AGILE	ITE

DOCUMENT TYPE: **TECHNICAL NOTE**

TITLE: **PROPOSAL FOR A BURST BACKGROUND FILTERING PROCEDURE**

DOCUMENT Ref. No.: AGILE-ITE-TN-008
IASF-BO Report 363/03

N° OF PAGES: i of iv, 13

ISSUE No.: 1.0

DATE: February, 2003

PROJECT Ref.: AGILE MCAL

PREPARED BY: E. CELESTI, C. LABANTI, M. MARISALDI

CHECKED BY: C. LABANTI

APPROVED BY:

SUBSYSTEM LEADER:	C. LABANTI	**DATE:**
PRINCIPAL INVESTIGATOR:	M. TAVANI	**DATE:**
PROGRAM MANAGER:		**DATE:**
PAPM:	R. A. BERNABEO	**DATE:**
CONFIGURATION:	C. MANGILI	**DATE:**

DISTRIBUTION LIST

POS.	NAME	DEPT.	N° OF COPIES	FULL COPY
	E. Celesti	**IASF-BO**	**1**	**YES**
	C. Labanti	**IASF-BO**	**1**	**YES**
	M. Marisaldi	**IASF-BO**	**1**	**YES**
	M. Tavani	**IASF-MI**	**1**	**YES**
	A. Argan	**IASF-MI**	**1**	**YES**
	G. Cafagna	**Laben**	**1**	**YES**
	L. Nicolini	**Laben**	**1**	**YES**
	M. Azzano	**Laben**	**1**	**YES**

CHANGE RECORD

ISSUE	DATE	PAGE	DESCRIPTION OF CHANGES	RELEASE
1.0	06/02/2003	All		

TABLE OF CONTENTS

1. INTRODUCTION

The scope of this procedure is to filter events that are originated by particles interacting in the MCAL and give rise to phosphorescence effects. In this case a fake burst trigger would be generated, particularly on short timescales (1, 16 ms).
This document shows a proposal of the method to use to reject such events, based on the fact that (in PICsIT) these phosphorescence events produce a sequence of counts originating from the same bar on a short timescale (of the order of a few milliseconds).

1.1 ACRONYMS

BBFP Burst Background Filtering Procedure

2. APPLICABLE AND REFERENCE DOCUMENTS

2.1 APPLICABLE DOCUMENTS

AD [1] *AGILE-DWG-SP-001 Issue 04*
AD [2] *AGILE-ITE-RE-002 Issue 1.1*

3. PHOSPHORESCENCE EFFECTS IN PICSIT

It was observed in PICsIT that occasionally a sudden burst in the rate of counts occurs, which is attributed to cosmic rays interacting with the detector's CsI(Tl) crystals. This interaction gives rise to tracks seen on the detector (with point like, linear or even elliptical shape), and a high number of counts is originated in a reduced time gap, from one or more pixels in sequence.
These counts have been attributed to phosphorescence effects originating in the crystals, due to interaction with high energy particles.
These counts have a 'trademark', which consists in their time and energy signature, as follows:

- Time 1: About 30 events are counted on the same pixel in a short time interval, typically 1.2 ms.
- Time 2: When more pixels give rise to phosphorescence effects, every pixel generates a train of counts occurring at different times, with a temporal distribution of about 100-200 ms.
- Energy: The equivalent energy of such events is near the threshold level (PICsIT threshold = 180 keV).

GCAL_TIME (μs)	PICSIT_PHA (chan)	PICSIT_Y (pixel)	PICSIT_Z (pixel)
0	**29**	**29**	**23**
61	**27**	**29**	**23**
61	121	32	55
61	**27**	**29**	**23**
122	**28**	**29**	**23**
122	**27**	**29**	**23**
183	1017	16	55
183	**29**	**29**	**23**
183	**30**	**29**	**23**
244	**30**	**29**	**23**
305	**30**	**29**	**23**
366	**27**	**29**	**23**
427	**29**	**29**	**23**
488	**27**	**29**	**23**
549	25	17	17
549	**29**	**29**	**23**
610	34	56	19
610	**27**	**29**	**23**
671	**34**	**29**	**23**
671	**27**	**29**	**23**
732	**30**	**29**	**23**
732	**27**	**29**	**23**
793	**28**	**29**	**23**
793	68	55	60
854	**27**	**29**	**23**
916	**32**	**29**	**23**
977	**30**	**29**	**23**
977	**27**	**29**	**23**
1038	**27**	**29**	**23**
1038	26	23	16
1099	**32**	**29**	**23**

In Table 3-1 is listed an example of such induced events as seen from the PICsIT detector.

Table 3-1: Event list for a typical phosphorescence induced sequence of counts in pixel (29, 23).

4. PHOSPHORESCENCE IN MCAL

It is likely that phosphorescence effects will arise on MCAL bars, when subject to interactions with high energy particles.
The time and energy signature will however be different compared to those seen on PICsIT, depending on a number of factors like:

- Saturation level of the preamplifiers.
- Energy threshold level.
- Geometrical shape of the detector.

A reliable model hasn't been found yet to describe this phenomenon in detail, and so a filtering procedure to reject them must be written, keeping in mind their general properties.

In the MCAL detector an high energy particle will probably hit more than one bar at the same time, and so more than one stream of events originating from different bars will be produced. These streams however should be spaced in time with gaps of tens of milliseconds, and so they will rarely overlap.
Within the phosphorescence induced stream of counts, there is the normal background count rate of about 2 kHz on the whole MCAL, this means about 2 counts/ms, versus a count rate of about 30 kHz during a phosphorescence event in PICsIT.

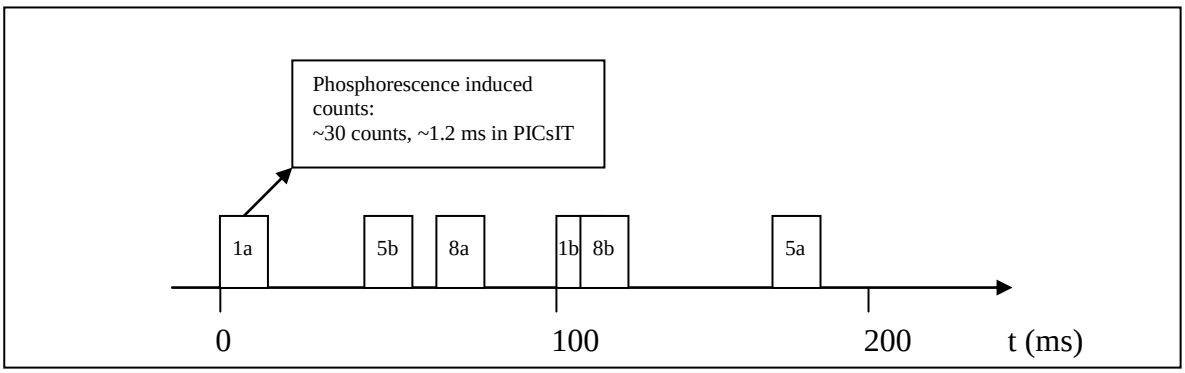

Figure 4-1: Supposed streams of counts induced by phosphorescence occurring on the MCAL detector.

A high count rate on AGILE could give rise to a burst trigger, particularly on short timescales (1 ms and 16 ms), and so a filtering procedure to discard these induced events is necessary.

4.1 BURST BACKGROUND FILTERING PROCEDURE PROPOSAL

The method proposed in this document will analyze the data stream and will be capable to identify subsequent events coming from the same bar.
In case of an overlapping between more than one induced count streams, the procedure is designed to work with up to a maximum of two bars triggering at the same time and producing events. It also keeps track of eventual spurious events infiltrating within the streams.

A list of the task to be accomplished by the filtering procedure (shown in Figure 4-2) follows:

- Every event is analyzed in sequence.
- Multiple events (hitting more than one bar in coincidence) are not considered as related to phosphorescence.
- When an event has the same bar address as one of the last two events occurred, the bar address is stored in a memory register (ref1), and a counter starts to count events coming from that bar (Counter1).
- There will be two memory registers for the storage of such events (ref1 and ref2), and so two bar streams can be tracked simultaneously.
- When a new stream starts, the respective bar address is stored in ref1, the old ref1 is shifted in ref2 and the old ref2 is deleted from memory. The same goes for the two respective counters (Counter2 goes in Counter1 and Counter1 is set to 1).
- When the current event has a different address with respect to the reference ones, this is considered as a spurious event, and a dedicated counter is increased (CounterNO).
- When a maximum number of spurious events within the stream are counted (CounterNO>THNO), the sequence is reset and the search procedure restarts, unless the reference address is empty (ref1=31).
- If the reference address is empty (ref1=31) no reset is performed, otherwise the procedure would loose track of the last two addresses stored (adr_event_n-1 and adr_event_n-2).
- When a predetermined number of events with the same bar address is counted (Counter1>TH1 or Counter2>TH2), a Flag is raised that indicates the occurrence of a phosphorescence effect.

The Flag will be considered by the DH in the Burst trigger generation sequence. Only the DH has the capability to reset the Flag.

N.B. During initialization, registers are set equal to 31, i.e. the registers are set empty. As there are only 30 bars this value will not interfere with the correct execution of the procedure.

4.2 REGISTERS LIST

Register name	Register function	Bits used
Ref1	Stores the address of the first reference bar	5
Ref2	Stores the address of the first reference bar	5
Adr_event_n	Stores the address of the bar being currently analyzed	5
Adr_event_n-1	Stores the address of the last bar analyzed	5
Adr_event_n-2	Stores the address of the last but one bar analyzed	5

4.3 COUNTERS LIST

Counter name	Counter function	Bits used
Counter1	Counts the events with the address equal to ref1	4
Counter2	Counts the events with the address equal to ref2	4
CounterNO	Counts the spurious events in a data stream	4

4.4 THRESHOLDS LIST

Counter name	Counter function	Bits used
TH1	Threshold on Counter1	4
TH2	Threshold on Counter2	4
THNO	Threshold on CounterNO	4

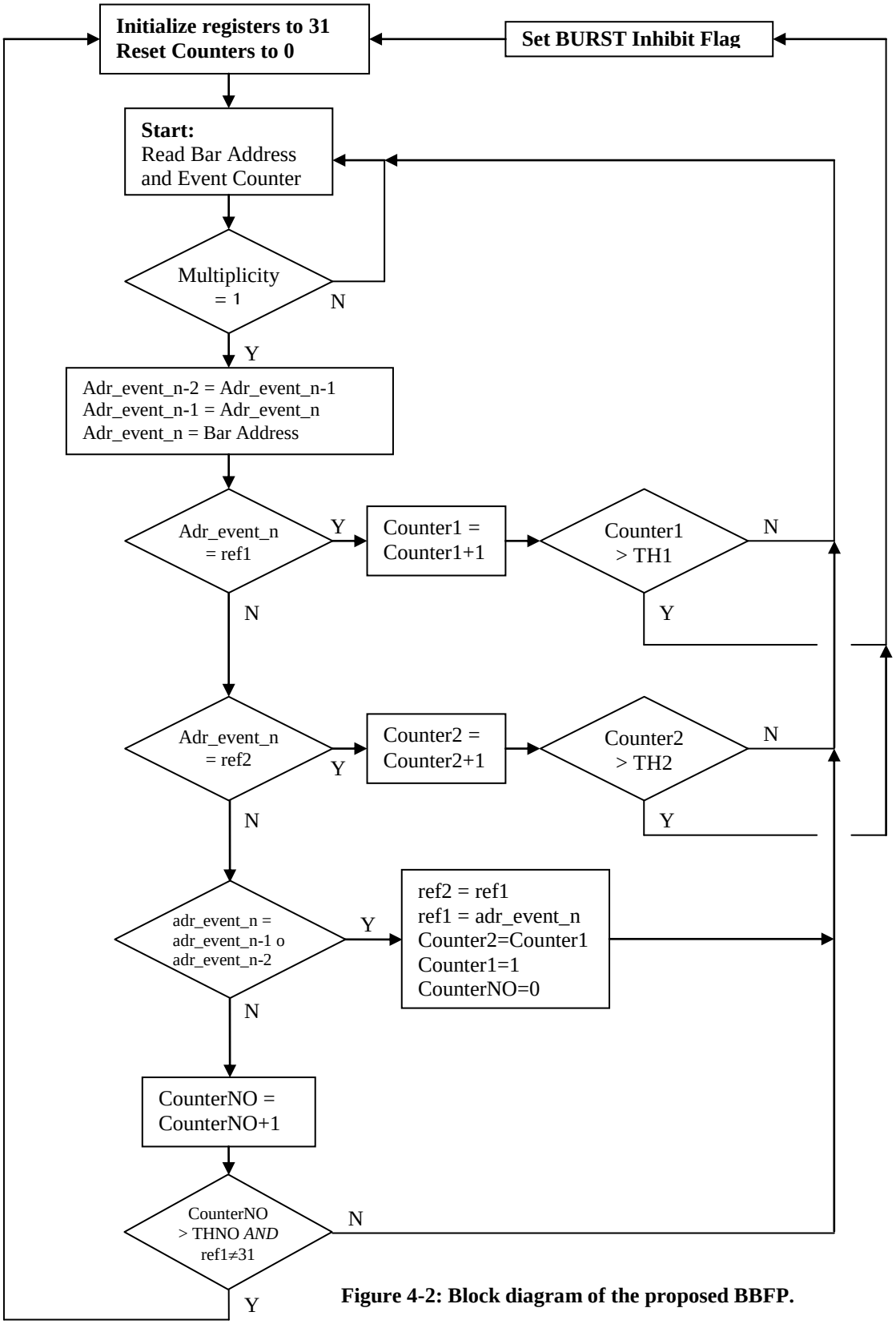

Figure 4-2: Block diagram of the proposed BBFP.

5. PHOSPHORESCENCE EVENTS SIMULATION

In order to test the filtering procedure, an algorithm has been written to simulate the procedure behavior, and a series of patterns of counts similar to those expected in MCAL has been simulated. For testing purposes only bar address values between 0 and 9 were taken.
Follows the simulation algorithm written in IDL language:

```
--------------------------------------------------------------------------------
PRO BBFP_SIM
;
;---BBFP simulator
;---Does not take into account the bar multiplicity
;

;---The input sequence s is a sequence of numbers indicating the bar addresses
;---In this simulation bar address ranges from 0 to 9
s=''
read, 'Input sequence: ', s
n=strlen(s)-1

;---Initialization of thresholds
TH1=6
TH2=6
THNO=3

;---Initialization of registers and counters
ref1=31
ref2=31
Adr_event_n=31
Adr_event_n1=31
Adr_event_n2=31
Counter1=0
Counter2=0
CounterNO=0
Flag = 0

print, 'adr_n      adr_n1  adr_n2  ref1      cnt1      ref2      cnt2      cntNO    flag'

for i=0, n do begin
        ;---Spostamento registri
        Adr_event_n2 = Adr_event_n1
        Adr_event_n1 = Adr_event_n
        Adr_event_n = fix(strmid(s, i, 1))

        case Adr_event_n of
        ref1    :begin
                Counter1 = Counter1 + 1
                if (Counter1 GT TH1) then begin
                        flag=1
                        print, format='(i4, i8, i8, i8, i8, i8, i8, i8, i8)', Adr_event_n, Adr_event_n1, Adr_event_n2, $
                                ref1, Counter1, ref2, Counter2, CounterNO, flag
                        goto, endproc
```

```
            endif
        end

        ref2:   begin
            Counter2 = Counter2+1
            if (Counter2 GT TH2) then begin
                flag=1
                print, format='(i4, i8, i8, i8, i8, i8, i8, i8, i8)', Adr_event_n, Adr_event_n1, Adr_event_n2, $
                                ref1, Counter1, ref2, Counter2, CounterNO, flag
                goto, endproc
            endif
        end

        Adr_event_n1:   begin
            ref2 = ref1
            ref1 = Adr_event_n
            Counter2 = Counter1
            Counter1 = 1
            CounterNO = 0
        end

        Adr_event_n2:   begin
            ref2 = ref1
            ref1 = Adr_event_n
            Counter2 = Counter1
            Counter1 = 1
            CounterNO = 0
        end

        else:   begin
        CounterNO = CounterNO+1
        end
        endcase

        print, format='(i4, i8, i8, i8, i8, i8, i8, i8, i8)', Adr_event_n, Adr_event_n1, Adr_event_n2, $
                        ref1, Counter1, ref2, Counter2, CounterNO, flag

        if (CounterNO GT THNO) and (ref1 NE 31) then begin
            ;---Initialization of registers and counters
            ref1=31
            ref2=31
            Adr_event_n=31
            Adr_event_n1=31
            Adr_event_n2=31
            Counter1=0
            Counter2=0
            CounterNO=0
        endif
endfor

ENDPROC:

END
--------------------------------------------------------------------------------
```

Below are listed some examples of relevant sequences sent to the simulator. In this simulation the following values of the thresholds were set:
TH1 = TH2 = 6; THNO = 3

Input sequence: 3566266686666666

adr_n	adr_n1	adr_n2	ref1	cnt1	ref2	cnt2	cntNO	flag
3	31	31	31	0	31	0	1	0
5	3	31	31	0	31	0	2	0
6	5	3	31	0	31	0	3	0
6	6	5	6	1	31	0	0	0
2	6	6	6	1	31	0	1	0
6	2	6	6	2	31	0	1	0
6	6	2	6	3	31	0	1	0
6	6	6	6	4	31	0	1	0
8	6	6	6	4	31	0	2	0
6	8	6	6	5	31	0	2	0
6	6	8	6	6	31	0	2	0
6	6	6	6	7	31	0	2	1

Table 5-1: Shows the normal behavior of the algorithm when a stream of

Input sequence: 1123232323232322

adr_n	adr_n1	adr_n2	ref1	cnt1	ref2	cnt2	cntNO	flag
1	31	31	31	0	31	0	1	0
1	1	31	1	1	31	0	0	0
2	1	1	1	1	31	0	1	0
3	2	1	1	1	31	0	2	0
2	3	2	2	1	1	1	0	0
3	2	3	3	1	2	1	0	0
2	3	2	3	1	2	2	0	0
3	2	3	3	2	2	2	0	0
2	3	2	3	2	2	3	0	0
3	2	3	3	3	2	3	0	0
2	3	2	3	3	2	4	0	0
3	2	3	3	4	2	4	0	0
2	3	2	3	4	2	5	0	0
3	2	3	3	5	2	5	0	0
2	3	2	3	5	2	6	0	0
2	2	3	3	5	2	7	0	1

counts is generated (in this case bar 6 shows phosphorescence).

Table 5-2: In this case are shown 2 stream of counts occurring simultaneously.

Input sequence: 11111532333436353111

adr_n	adr_n1	adr_n2	ref1	cnt1	ref2	cnt2	cntNO	flag
1	31	31	31	0	31	0	1	0
1	1	31	1	1	31	0	0	0
1	1	1	1	2	31	0	0	0
1	1	1	1	3	31	0	0	0
1	1	1	1	4	31	0	0	0
5	1	1	1	4	31	0	1	0
3	5	1	1	4	31	0	2	0
2	3	5	1	4	31	0	3	0
3	2	3	3	1	1	4	0	0
3	3	2	3	2	1	4	0	0
3	3	3	3	3	1	4	0	0
4	3	3	3	3	1	4	1	0
3	4	3	3	4	1	4	1	0
6	3	4	3	4	1	4	2	0
3	6	3	3	5	1	4	2	0
5	3	6	3	5	1	4	3	0
3	5	3	3	6	1	4	3	0
1	3	5	3	6	1	5	3	0
1	1	3	3	6	1	6	3	0
1	1	1	3	6	1	7	3	1

Table 5-3: This case shows that it is possible, in some particular circumstances, to have a large amount of spurious counts without the occurrence of a reset, in spite of a low value of THNO.

Input sequence: 3571162342115511551155 11

adr_n	adr_n1	adr_n2	ref1	cnt1	ref2	cnt2	cntNO	flag
3	31	31	31	0	31	0	1	0
5	3	31	31	0	31	0	2	0
7	5	3	31	0	31	0	3	0
1	7	5	31	0	31	0	4	0
1	1	7	1	1	31	0	0	0
6	1	1	1	1	31	0	1	0
2	6	1	1	1	31	0	2	0
3	2	6	1	1	31	0	3	0
4	3	2	1	1	31	0	4	0
2	31	31	31	0	31	0	1	0
1	2	31	31	0	31	0	2	0
1	1	2	1	1	31	0	0	0
5	1	1	1	1	31	0	1	0
5	5	1	5	1	1	1	0	0
1	5	5	5	1	1	2	0	0
1	1	5	5	1	1	3	0	0
5	1	1	5	2	1	3	0	0
5	5	1	5	3	1	3	0	0
1	5	5	5	3	1	4	0	0
1	1	5	5	3	1	5	0	0
5	1	1	5	4	1	5	0	0
5	5	1	5	5	1	5	0	0
1	5	5	5	5	1	6	0	0
1	1	5	5	5	1	7	0	1

Table 5-4: Sequence showing the functionality of the reset after CounterNO>THNO. Notice that information in Adr_event_n-1 and Adr_event_n-2 is lost.