S.P.L.A.T ISA BOARD
IO-953
MANUAL

Brookhaven National Laboratory
Version 1.0

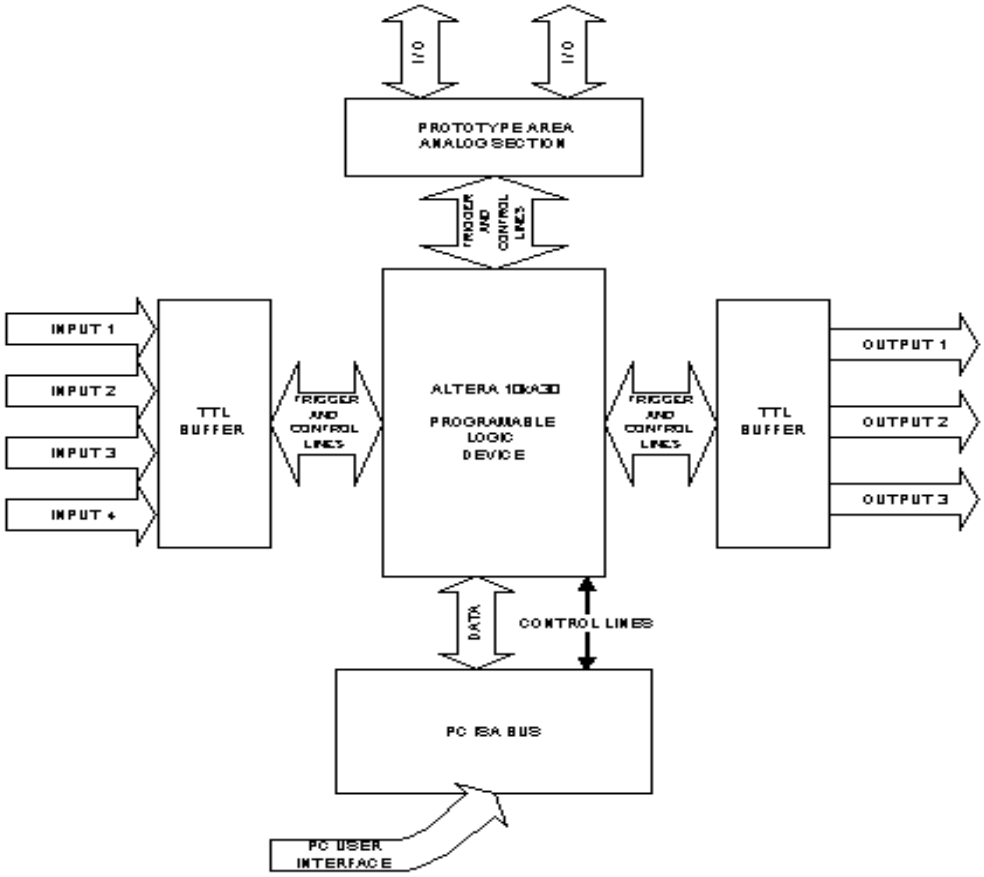
BY Jack Fried (email jfried@bnl.gov)
Web page (http://www.inst.bnl.gov/~jfried)

INTRODUCTION

The objective behind the SPLAT ISA board was to create a timing and control system for an aerosol particle mass spectroscopy experiment. The board will be used to measure the time of flight of particles in air to trigger an ablation laser. To accurately measure the time of flight a high-speed clock with a resolution of 25ns is used. All of the boards functions are implemented in an Altera 30K PLD, which allows the board to be modified fairly simply. At present the PLD contains integrators, counters, multipliers and a PC ISA interface to implement the required function.

BOARD FEATURES

- **40 MHz Clock**
- **10 MHz Clock**
- **Seven I/O ports**
- **Removable Prototype Area**
- **PC ISA Interface**
- **PC Interrupt Compatible**
- **High Speed PLD**
- **VC++ Software Interface**
- **Single Event Mode**
- **Continuos Event Mode**
- **Fully controllable registers**



SPLAT ISA BOARD
Block Diagram

SPECIFICATIONS

INTEGRATORS (PMT1 & 2)

1. Threshold count 1 – 65,5360
2. Integration window 25ns – 6.4us
3. Saturation width 25ns – 6.375us

INTEGRATOR (CHANNELTRON)

1. Threshold count 1-65,5360 (wait for integration window to end)
2. Integration window 25ns-6.4s
3. Saturation width NONE

TRIGGER OUT DELAY (PMT 1 & 2)

1. delay trigger by -200ns - 1.6384ms

TRIGGER OUT DELAY (CHANNELTRON)

1. delay trigger by 0 - 1.6384ms

MULTIPLIER

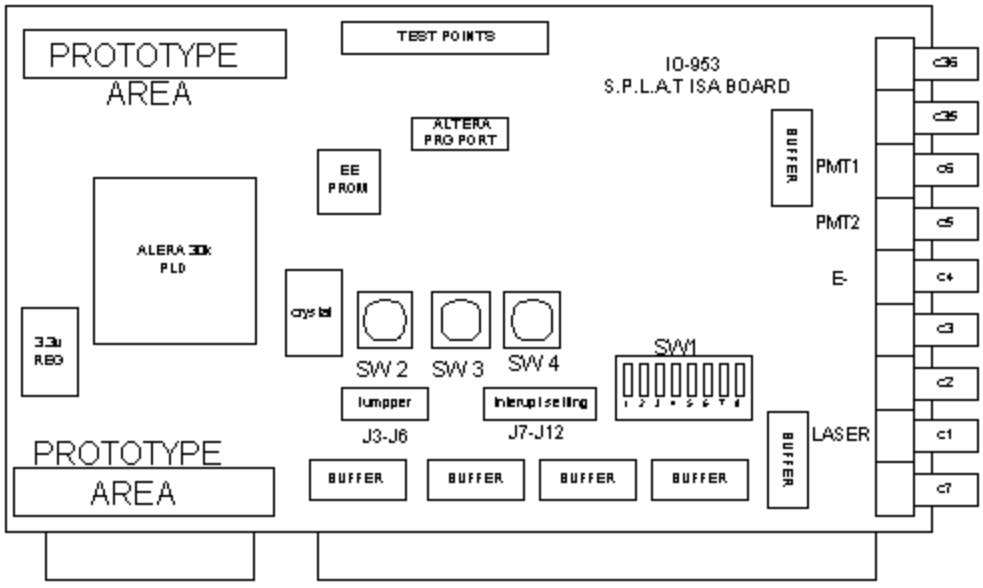
1. multiplication constant. 0 - 1.99999

LASER TIME OUT DELAY

1. Trigger Rest Delay 1.6us - 104.8576ms

TRIGGER OUT ACCURACY

1. Time between PMT1 & PMT2 ± 50ns
2. CHANNELTRON output trigger ± 25ns



DIP SWITCH SETTINGS SW1

PC I/O
Address (O = open C = Closed X= NOT USED)

	{ 1 2 3 4 5 678}
0x3e0	O O O O O XXX
0x3c0	O O O O C XXX
0x3a0	O O O C O XXX
0x380	O O O C C XXX
0x360	O O C O O XXX
0x340	O O C O C XXX
0x320	O O C C O XXX
0x300	O O C C C XXX
0x2e0	O C O O O XXX
0x2c0	O C O O C XXX
0x2a0	O C O C O XXX
0x280	O C O C C XXX
0x260	O C C O O XXX
0x240	O C C O C XXX
0x220	O C C C O XXX
0x100	C O C C C XXX
0x1e0	C O O O O XXX
0x1c0	C O O O C XXX
0x1a0	C O O C O XXX
0x180	C O O C C XXX
0x160	C O C O O XXX
0x140	C O C O C XXX
0x120	C O C C O XXX
0x100	C O C C C XXX

PC ISA REGISTERS

Board ID
BRDID 0x00

Function :
Return the board ID.
WRITE
NONE
READ
16 bit

Photo Multiplier 1&2 Threshold Setting
PMTTRS 0x02

Function :
This register controls the threshold for PMT1 and PMT2.
It sets the number of pulse counts the PMTS make before a trigger is generated.

WRITE

16 bit

READ

16 bit

Photo Multiplier 1&2 Integration window Setting
PMTWND 0x04

Function:

This register sets the Integration Windows for PMT1 and PMT2
RANGE 1-256 25ns - 6.4us
1 = 25ns
2 = 50ns
.
.
255 = 6.375us
256 = 6.400us

WRITE

8 bit

READ

8 bit

Photo Multiplier 1&2 saturation Setting
PMTSAT 0x06

Function:

This register sets the saturation setting for PMT1 and PMT2
RANGE 1-256 25ns - 6.4us
1 = 25ns
2 = 50ns
.
255 = 6.375us
256 = 6.400us

WRITE

8 bit

READ

8 bit

CHANNELTRON Threshold Setting
CHNTRS 0x08

Function :

This register controls the threshold for PMT1 and PMT2.
It sets the number of pulse counts the PMTS make before a trigger is generated.

WRITE

16 bit

READ

16 bit

CHANNELTRON Integration window Setting
CHNWND 0x0a

Function:

This register sets the Integration Windows for PMT1 and PMT2
RANGE 1-256 25ns - 6.4us
1 = 25ns
2 = 50ns

.
.
255 = 6.375us
256 = 6.400us

WRITE

8 bit

READ

8 bit

NOT USED

CHNENC 0x0c

Function :

NONE

WRITE

NONE

READ

NONE

TRIGGER reset delay

LTIMEOUT 0x0e

Function :

Wait time after the trigger is generated before restarting cycle.

Range 1.6us – 104.8576ms in 1.6us steps

0 = 1.6us
1 = 3.2us
2 = 4.8us
.
.

WRITE

16 bit

READ

16 bit

WAIT After PMT1 Trigger Before Allowing PMT2 to Trigger

T1WAIT 0x10

Function :

Wait time after PMT1 trigger is generated before allowing PMT2.

Range 25ns – 1.638ms

1 = 25ns
2 = 50ns
2 = 75ns
.

WRITE

16 bit

READ

16 bit

TIME OUT after PMT1 Triggered But no PMT2 Trigger Arrived

T2WAIT 0x12

Function :

TIME OUT after PMT1 triggered but NO PMT2 arrives

Range 25ns – 1.638ms

1 = 25ns
2 = 50ns

2 = 75ns
.

WRITE
16 bit
READ
16 bit

CHANNELTRON Delay output Trigger
T3WAIT 0x14

Function : CHANNELTRON Delay output Trigger
Range 0ns – 1.638ms
0 = 0
1 = 25ns
2 = 50ns
2 = 75ns
.

WRITE
16 bit
READ
16 bit

NOT USED
MULTSUM 0x16

Function :
NONE
WRITE
NONE
READ
NONE

MULTIPLIER CONSTANT
ERRC x18

Function : multiplier Constant
Range 0 – 1.99999

C1*4096 = 16 bit value
Example
T1 = 5430
C1= 0.98
0.98*4096 = 4014.08 (float)
reg_load -> = 4014 (16 bit)

true result = 5321.4
mult result = 5321.293

WRITE
16 bit
READ
16 bit

Photo Multiplier trigger output delay
PMTDEL 0x1a

Function : CHANNELTRON Delay output Trigger
Range -175ns – 1.6378ms
1 = -175ns delay

2 = -150ns delay
3 = -125ns delay
.
7 = -25ns delay
8 = 0ns delay
9 = 25ns delay
.
.

WRITE
16 bit
READ
16 bit

USER REGISTERS
STAR 0x1c

Function : USER REGISTERS

Bit 0 :generate trigger 0 -> 1
Bit 1 :set 0 = single event
:set 1 = continues trigger
Bit 2 :set 0 = PMT delay disabled
:set 1 = PMT delay enable
Bit 4 :set 1 = reset board
:set 0 = Normal run

Bit 14: if = 0 data from PMT
If = 1 data from CHANNELTRON
Bit 15: if = 1 trigger fired

WRITE
14 bit
READ
16 bit

EVENT DATA OUTPUT
PTRVLT 0x1e
Function : READ the time of flight or the # of electrons
WRITE
NONE
READ
16 bit

SOFTWARE EXAMPLE:

EXAMPLE 1

```
#include “splatisa.h”

splatisa splat;    /* declare the class splatisa*/


function()
```

```
{
        /* set up splat board */
    splat.reset_board();
    splat.set_reg(PMTTRS,0x100);        //Set PMT threshold to 256
    splat.set_reg(PMTWND,0x150);        //Set window to  3.75us
    splat.set_reg(LTIMEOUT,12500)       // Set timeout to 20ms
        /*collect data */

    OPEN FILE  TO SAVE DATA

    splat.cont_trigger();               // Have the board generate continuos triggers
    while(!splat.check_trigger() && !STOP)
    {
        Save to file ← splat.read_trvl();       // Save data collected from splat
        Save to file ← splat.reg.start();       // Save what generated the trigger
        Save to file ← PCI DATA                 // read data from PCI card
    }

    CLOSE FILE
}
```

EXAMPLE 2

```
#include   "splatisa.h"

splatisa splat;    /* declare the class  splatisa*/


function()
{
        /* set up splat board */
    splat.reset_board();
    splat.set_reg(PMTTRS,0x100);        //Set PMT threshold to 256
    splat.set_reg(PMTWND,0x150);        //Set window to  3.75us
        /*collect data */

    OPEN FILE  TO SAVE DATA

    splat.single_trigger();             // Have the board generate single  triggers
    while(!splat.check_trigger() && !STOP)
    {
        Save to file ← splat.read_trvl();       // Save data collected from splat
        Save to file ← splat.reg.start();       // Save what generated the trigger
        Save to file ← PCI DATA                 // read data from PCI card
        splat.single_trigger();                 // generate a single triggers

    }
```

```
        CLOSE FILE
}
```