SSD slow control (old)

<http://www.star.bnl.gov/public/ssd/>

**The code can be found at ssdTop.zip.**

SSD slow control is written based on EPICS. There are mainly two independent control interface for the detector and power supply.

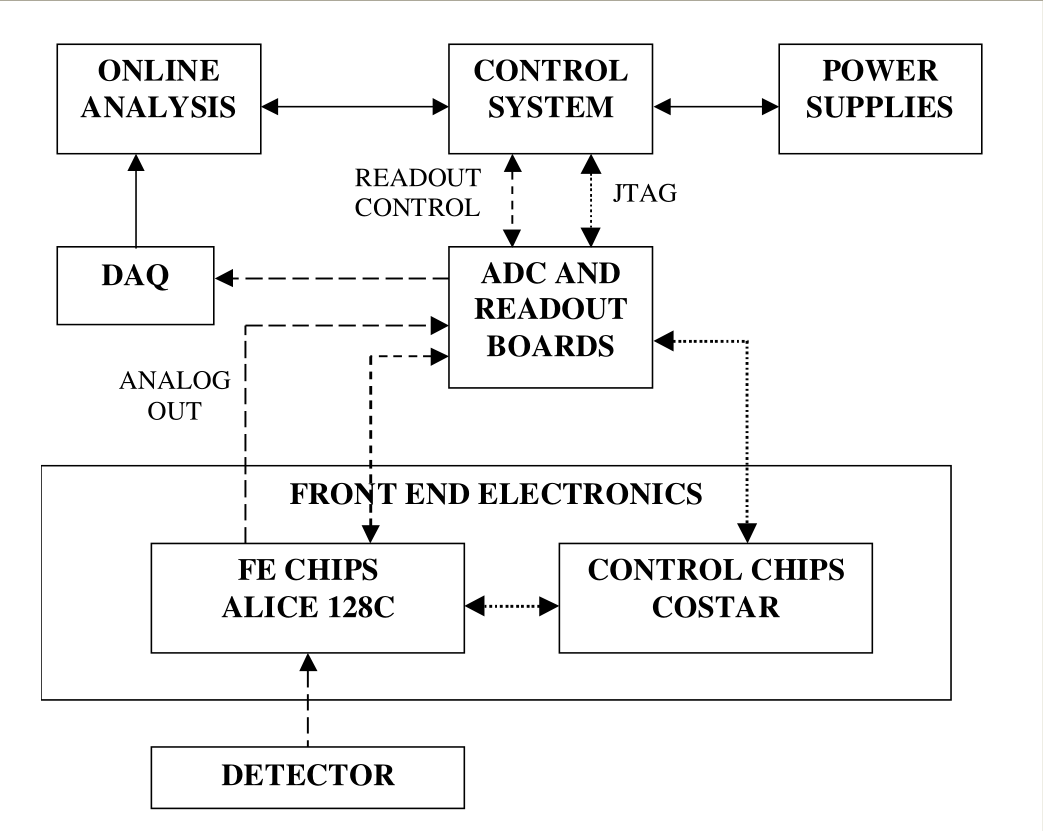


Figure : Block diagram of the system

# Detector

Electrically the SSD is divided in four subsystems (one P-side and one N-side on each clamshell), which correspond in terms of data to four half of ten ladders. These subsystems are identical. They can be represented as Figure 2.

The readout board is a kind of “hub” in the readout chain and slow control system of the SSD. It is the interface of the barrel and the “outside world” (slow control, trigger and DAQ). The main part of the readout board is FPGA.

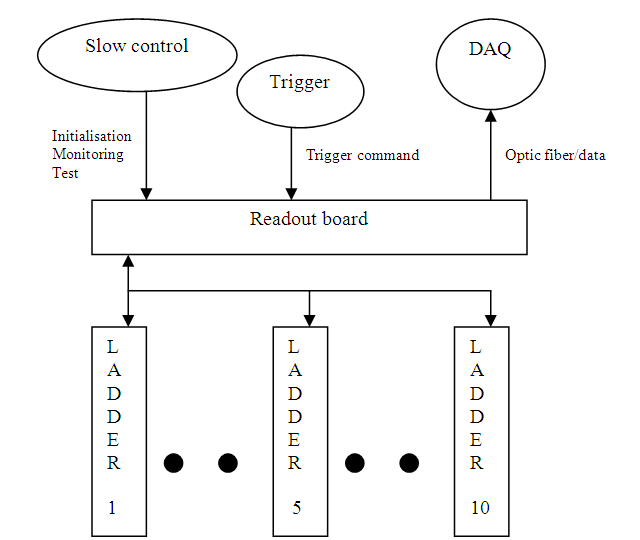


Figure : Readout chain

One ladder has 16 modules. To control the modules and convert the signal coming from the FEE, two electronic boards sit at each end of the ladder:

1. Control/Connection/C2D2 board: mainly dedicated to detect and to switch off the power supply if any latch-up occurs in the FEE of one module.
2. ADC board: convert the analog signal coming from the modules.

Each module is composed of a double sided silicon strip detector and two hybrid circuits (6 Alice128C readout chips + 1 COSTAR control chip). There are also two Tape automated bonding (TAB) is used for the connection between the strips on the detector and the analog inputs of the ALICE128C.

The ALICE128C and COSTAR chips are all remotely controlled by a JTAG protocol. The readout boards and connection boards are also in the JTAG chain.

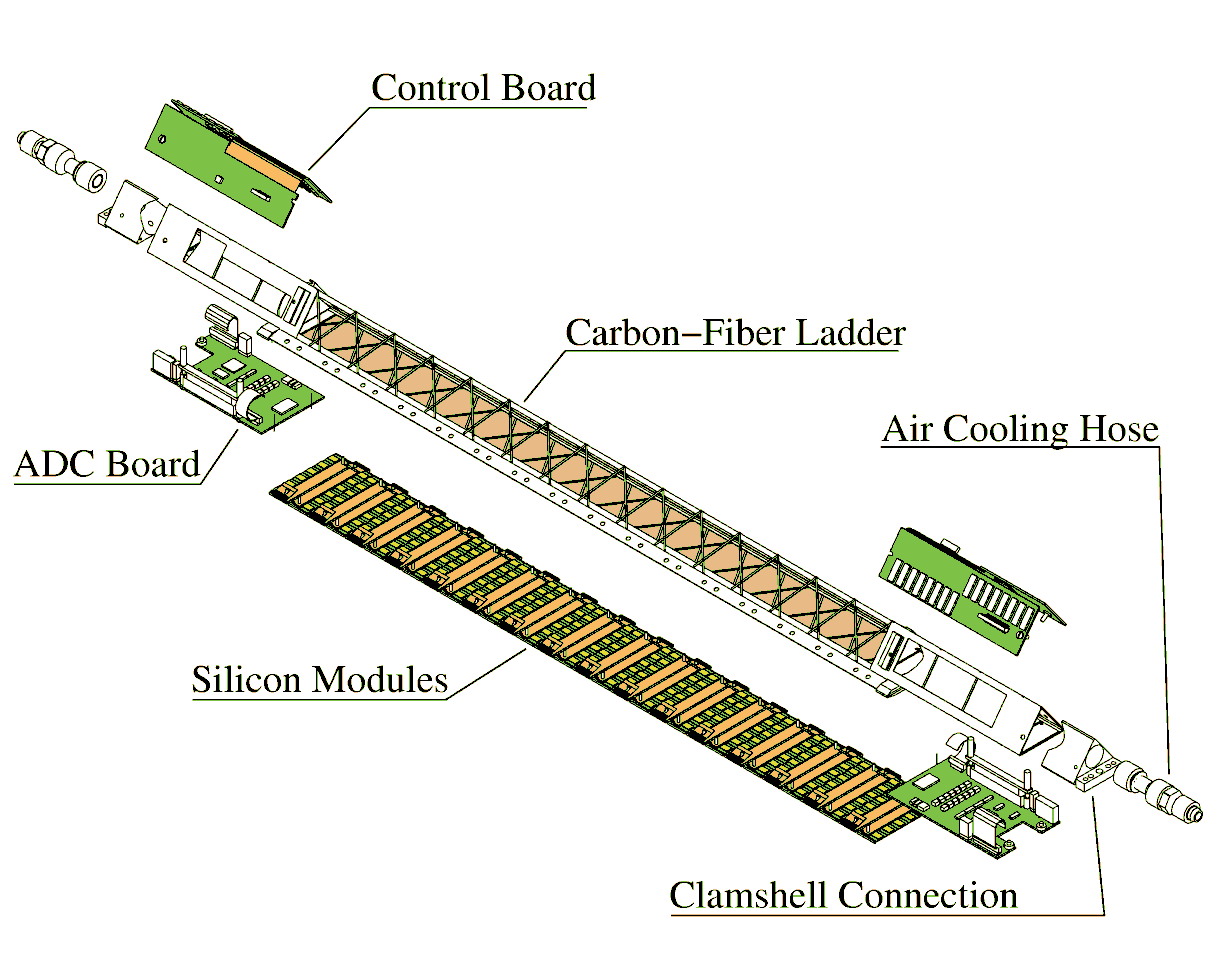


Figure : Exploded view of an equipped ladder

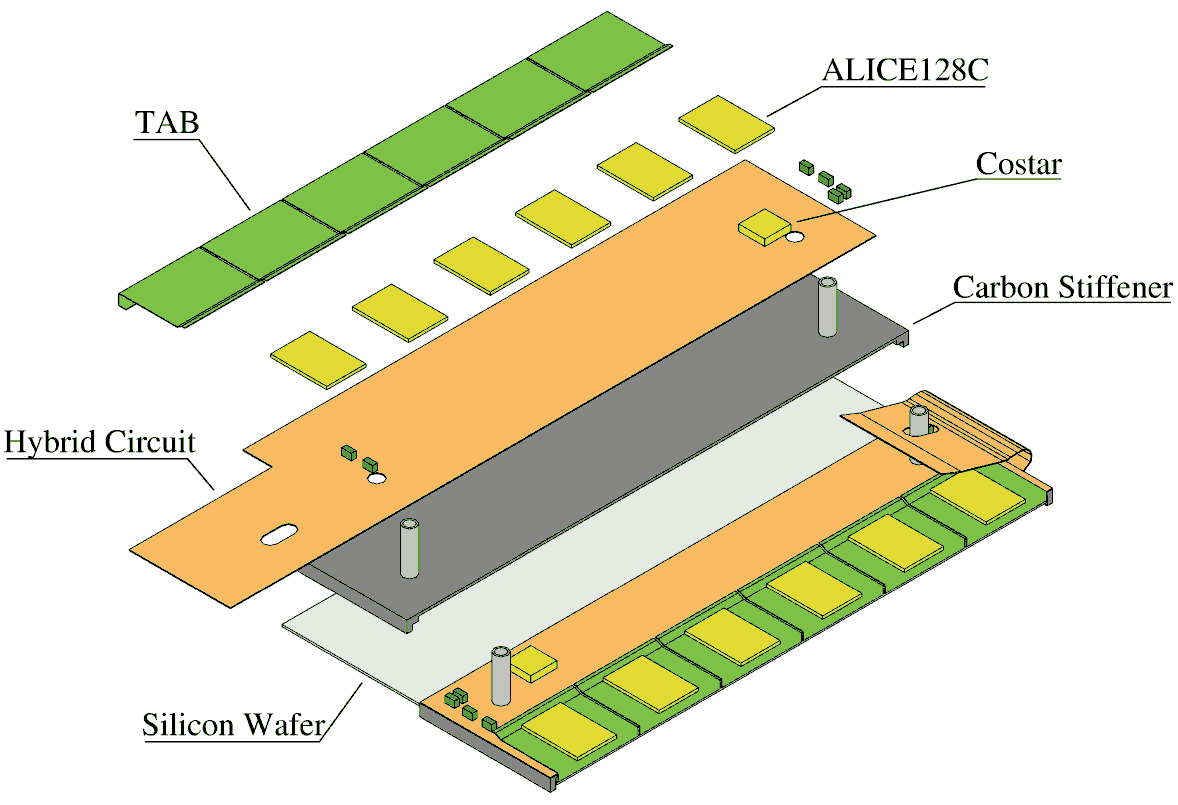


Figure : Exploded view of a detection module

# JTAG

The control signals are generated by Corelis CVME 1149.1 board and the core is SN74ACT8990 Test Bus Controller. The board provides some high-level function for the JTAG communication. The most important three functions are:

**scan\_ir(unsigned short \*output, unsigned short length, unsigned short \*input)**

**scan\_dr(unsigned short \*output, unsigned short length, unsigned short \*input)**

**circulate\_dr(unsigned short length, unsigned short \*data)**

## ALICE128C

/\* Alice128C registers size \*/

#define AL\_IR\_SZ 5

#define AL\_BSR\_SZ 8

#define AL\_BYPASS\_SZ 1

#define AL\_BIAS\_SZ 56

#define AL\_PULSE\_SZ 8

#define AL\_TEST\_SZ 128

#define AL\_READ\_SZ 258

#define AL\_ONOFF\_SZ 1

#define AL\_TOKEN\_SZ 1

/\* Alice128C commands \*/

#define AL\_BYPASS 0x1f

#define AL\_EXT\_TEST 0x00 /\* not used \*/

#define AL\_BSR\_SP 0x05 /\* not used \*/

#define AL\_BIAS 0x09

#define AL\_PULSE 0x0d

#define AL\_TEST 0x11

#define AL\_READ 0x15

#define AL\_ONOFF 0x19

#define AL\_TOKEN 0x1d

## COSTAR

/\* COSTAR REGISTERS SIZE \*/

#define CO\_IR\_SZ 5

#define CO\_BSR\_SZ 20

#define CO\_BYPASS\_SZ 1

#define CO\_CTRL\_SZ 8

#define CO\_IDENT\_SZ 8 /\* value: 0xaf \*/

#define CO\_ADC\_SZ 32

#define CO\_ADC\_TEST\_SZ 4

#define CO\_DAC\_SZ 32

#define CO\_DINOUT\_SZ 8

/\* COSTAR COMMANDS \*/

#define CO\_EXT\_TEST 0x00 /\* not used \*/

#define CO\_HIGHZ 0x01 /\* not used \*/

#define CO\_BSR\_SP 0x02 /\* not used \*/

#define CO\_IN\_TEST 0x03 /\* not used \*/

#define CO\_CLAMP 0x04 /\* not used \*/

#define CO\_ID 0x05 /\* supp 10/02/99 cf doc not used\*/

#define CO\_CTRL\_1 0x10

#define CO\_ADC\_TEST\_0 0x11

#define CO\_ADC\_0 0x12

#define CO\_ADC\_TEST\_1 0x13

#define CO\_ADC\_1 0x14

#define CO\_DINOUT 0x17 /\* not used \*/

#define CO\_DAC\_0 0x18

#define CO\_CTRL\_2 0x1A

#define CO\_IDENT 0x1B

#define CO\_BYPASS 0x1F

## CONNECTION BOARD

/\* CONNEXION REGISTERS SIZE \*/

#define CX\_IR\_SZ 5

#define CX\_BYPASS\_SZ 1

#define CX\_IDENT\_SZ 8 /\* value: 0x90 \*/

#define CX\_LATCHUP\_SZ 1

#define CX\_STATUS\_SZ 16

#define CX\_VREF\_SZ 2

#define CX\_POWER\_SZ 16

#define CX\_HYB\_BYP\_SZ 16

#define CX\_TOKENOUT\_SZ 1

#define CX\_MUX\_SZ 160

#define CX\_TEMP\_START\_SZ 1

#define CX\_TEMP\_SZ 36

/\* FPGA Connexion commands \*/

#define CX\_BYPASS 0x1F

#define CX\_IDENT 0x1B

#define CX\_LATCHUP 0x06

#define CX\_STATUS 0x07

#define CX\_VREF 0x08

#define CX\_POWER 0x09

#define CX\_HYB\_BYP 0x0B

#define CX\_TOKENOUT 0x0C /\* not used \*/

#define CX\_MUX 0x05

#define CX\_TEMP\_START 0x0D

#define CX\_TEMP 0x0E

## READOUT

/\* READOUT REGISTERS SIZE \*/

#define RD\_IR\_SZ 5

#define RD\_BYPASS\_SZ 1

#define RD\_IDENT\_SZ 8 /\* value: 0x60 \*/

#define RD\_VERSION\_SZ 48

#define RD\_CONFIG\_SZ 40

#define RD\_STATUS\_SZ 16

#define RD\_TEMP1\_SZ 24

#define RD\_TEMP2\_SZ 46

/\* #define RD\_OPTIC\_SZ ? To be defined \*/

#define RD\_LATCHUP\_SZ 8

#define RD\_CHAIN\_SZ 2

#define RD\_NB\_EVENT\_SZ 48

#define RD\_OPTIC\_VAR\_SZ 48

#define RD\_OPTIC\_FAB\_SZ 64

#define RD\_DAQ\_STATUS\_SZ 40

#define RD\_PEDESTAL\_SZ 20

#define RD\_ADJUST\_SZ 8

/\* FPGA Readout commands \*/

#define RD\_BYPASS 0x1F

#define RD\_IDENT 0x1B

#define RD\_VERSION 0x0C

#define RD\_CONFIG 0x0B

#define RD\_STATUS 0x09

#define RD\_TEMP1 0x08

#define RD\_TEMP2 0x0A /\* not used \*/

#define RD\_NUM\_CONFIG 0x0D /\* not used \*/

#define RD\_LATCHUP 0x07

#define RD\_CHAIN 0x06

#define RD\_NB\_EVENT 0x03

#define RD\_OPTIC\_VAR 0x04

#define RD\_OPTIC\_FAB 0x05

#define RD\_DAQ\_STATUS 0x02

#define RD\_PEDESTAL 0x0E

#define RD\_ADJUST 0x12

# Power supply

The power supply will change from CAEN to WIENER. The related code need to be rewritten completely.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Last modified: 12-10-2012

[yanwh@rcf.rhic.bnl.gov](mailto:yanwh@rcf.rhic.bnl.gov)