

Slow Control Software

AGATA Week 2026

Summary

- Interface with Topology manager and respective roles
- Slow Control :
 - New crystal colors on the main page and legend
 - Cards IPus connection and IDs checking
 - Card information page
 - Actions (STARE and PACE python functions)
 - Arguments page customized for each one
 - Automatic interpretation from documentation
 - Automated tests
- Repositories (github, gitlab, FW, SW) and software name

Interface with TopologyManager

- TM information about available STARE cards

```
{
  "connections": [
    {
      "id": "stare.0",
      "uri": "chtcp-2.0://10.81.17.221:10203?target=10.81.17.157:50001",
      "address_table": "file:///... /Stare_Utility/STARE.xml",
      "cluster": "ATC01", "position": 11, "crystal": "A",
      "host1": "10.10.1.6", "port1": 5001, "host2": "10.10.5.6", "port2": 5001,
      "data_from": "FROMSTARE",
      "stare": "stare06-A",
      "stare_bitstream_id": "DEADC0DE", "pace_bitstream_id": "FADEF00D"
    },
    ...
  ]
}
```

controlhub IP

card IP

- TM takes care of the network configuration and data flows
- SC configures and drives the cards, Oscilloscope shows inspection signals

Topology Manager Connections

Slow Control Crystals view Selection Configuration Action Administration About Logout

List Circles Planisphere **Topology**

Topology Manager Connections

Datetime: 2026-09-09 07:39:34 Reload

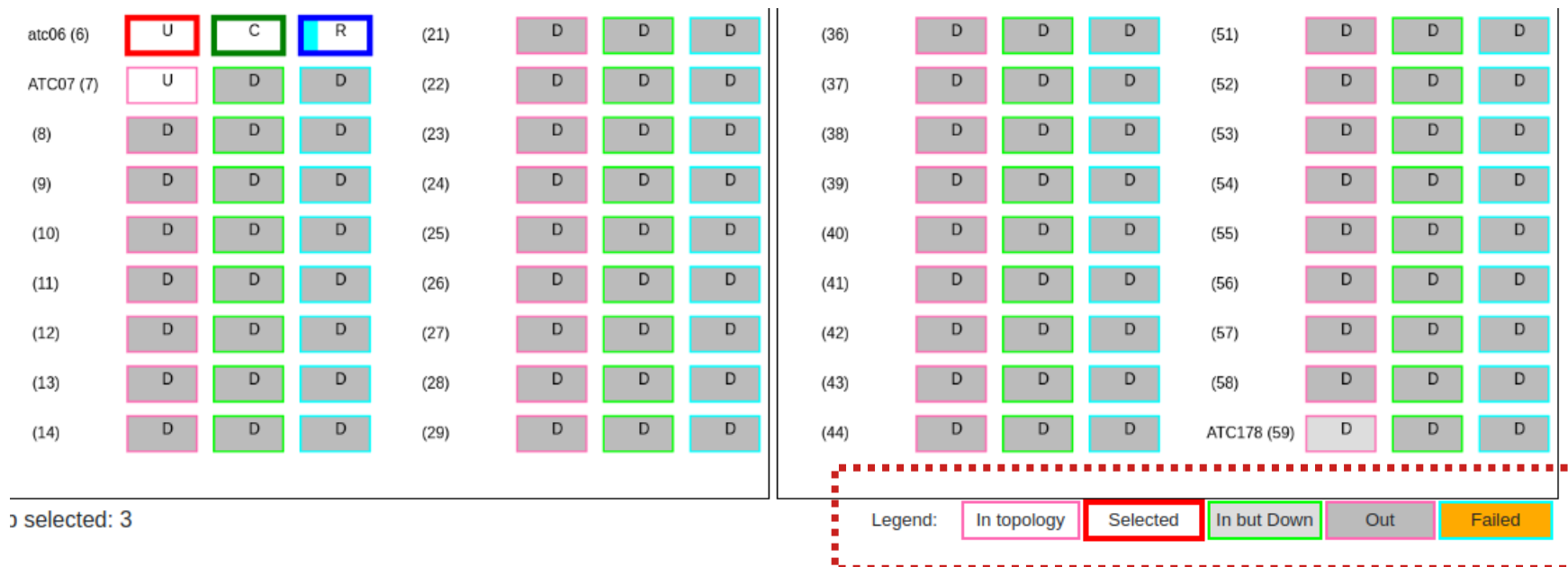
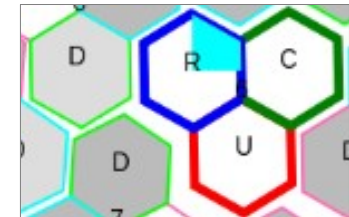
ID	URI	Cluster	Position	Crystal	STARE	Host1	Host2
2	chtcp-2.0://10.81.17.221:10203?target=10.81.17.164:50001	ATC01	0	B	stare000-B	10.81.29.40:5047	10.81.30.40:5048
20	chtcp-2.0://10.81.17.221:10203?target=10.81.17.190:50001	atc06	6	B	stare040-1	10.81.29.41:5049	10.81.30.41:5050
1	ipbusudp-2.0://172.16.2.34:50001	ATC00	0	A	stare010-1	10.10.1.6:5001	10.10.5.6:5001

Cannot find file: file://config/notfound.xml

- Coherence tests, errors signaled by background color and tooltip
 - Unicity of : names, tuple [Position, Crystal]
 - missing fields
 - Existence of the IPbus address file
- Detection of new available topology
- JSON file download for visualization or error detection

Cards colors and legend

- Crystal background colors : Out, In, Down, Failed
- Crystal border widths : selection
- Partial filling : trigger rate of detector
- Letter : status of Finite State Machine (D, U, C, I, R)



Cards checking

- Test IPbus connection
- Check that firmware ID bitstreams match with TM
 - STARE
 - PACE

Configure Setup Go Stop Reset **Check**

Status: Up: 4, Configured: 1, Idle: 0, Running: 1, Warning/Error(0, 0)

ATC01 (0)	U	D	D	(15)	D	D	D	(30)	D	D	D	(45)	D	D	D
ATC03 (1)	D	D	D	(16)	D	D	D	(31)	D	D	D	(46)	D	D	D
(2)	D	D	D	(17)	D	D	D	(32)	D	D	D	(47)	D	D	D
(3)	D	D	D	(18)	D	D	D	(33)	D	D	D	(48)	D	D	D
(4)	D	D	D	(19)	D	D	D	(34)	D	D	D	(49)	D	D	D
(5)	D	D	D	(20)	D	D	D	(35)	D	D	D	ATC50 (50)	D	D	D
atc06 (6)	U	D	D		D	D	D		D	D	D	(51)	D	D	D
ATC07 (7)	U	D	D		D	D	D		D	D	D	(52)	D	D	D
(8)	D	D	D		D	D	D		D	D	D	(53)	D	D	D
(9)	D	D	D		D	D	D		D	D	D	(54)	D	D	D
(10)	D	D	D		D	D	D		D	D	D	(55)	D	D	D
(11)	D	D	D		D	D	D		D	D	D	(56)	D	D	D
(12)	D	D	D	(27)	D	D	D	(42)	D	D	D	(57)	D	D	D

ATC07, Up
Position: 7A
Rate: 14 kHz
Check error:
Expected PACE ID bitstream: FADEF00D, found: 00000000
Expected STARE ID bitstream: DEADCODE, found: 00000000

Cards information page

i Card Information

stare040-1 (Position: 19)

Name	stare040-1
Position	19
Address	chtcp-2.0://10.81.17.221:10203?target=10.81.17.190:50001
Table	file://config/STARE.xml
Host1	10.81.29.41
Port1	5049
Host2	10.81.30.41
Port2	5050
Pace	null
Digitiser	digi000
STARE Bitstream ID	A1B4F59D
Pace Bitstream ID	12345678
Data From	FROMSTARE

[← Home](#)

Actions documentation

- Show undocumented actions (objective : 0)
- Category inside class
- Versions of embedded python scripts
 - PACE functions
 - STARE functions
- Use of help content to generate HTML form

Actions for card stare010-1

Action class:

pace_call
pace_basic
pace_gts
pace_rcv
pace_dtp
pace_read
pace_mon
pace_sc
stare

49

STARE_aurora_status	STARE_board_reset	STARE_change_emula
STARE_compare_firmware_flash	STARE_disable_package_generator	STARE_erase_and_pro
STARE_erase_sector_firmware_flash	STARE_exit	STARE_force_flush_au
STARE_r_full_eeprom	STARE_r_reg_eeprom	STARE_read_firmware
STARE_read_network_configuration	STARE_read_project	STARE_read_serial_nu
STARE_read_sys_manag	STARE_read_version	STARE_reset_status
STARE_send_random	STARE_sent_error	STARE_w_reg_eeprom
STARE_write_gateway_sfp	STARE_write_ip_server_sfp	STARE_write_ip_stare
STARE_write_mac_sfp	STARE_write_port_sfp	STARE_write_project
STARE_write_single_emulator_ram	STARE_write_submask_sfp	STARE_write_version
STARE_read_idbitstream		STARE_read_one_data

Read: Write a single 64bit value to an emulator RAM.

Arguments:

<address(dec or hex)> <data(hex)> <emulator_number (0-3)> <ram_type (0=adf_CPT, 1=adf_RAM)>

Example:

52 0x0011223344556677 0 1

deprecated: on this example adresse is decimal, and data in hexadecimal ; both are allowed

?

[Automatic use of actions help to generate arguments list](#)

Ph2_PACE_CMD version: v30.435 (2026-07-16)

stare version: v2.0.1 (2026-06-11)

Automatic help content interpretation

- Page explaining how help content is automatically used to generate arguments list (rules decided after the usage):

Use of actions help to generate arguments list

Usage: or **Arguments:** followed by the description of arguments on the same or next line:

Argument list enclosed in angle brackets `< >` for required arguments or square brackets `[ ]` for optional arguments, separated by spaces. Angle brackets are not required if no type is provided. Arguments can also span over multiple lines.

A description of each argument can be placed after the first `-` or `:` character.

Within parentheses:

- A type may be provided (*int*, *dec*, *hex*, *str*, ...).
- If multiple types are possible, they are separated by *or*, e.g., (*dec or hex*). Ranges are indicated by a `-`, e.g., `0-255`.
- Type hierarchy: *string* includes *hexadecimal*, which includes *int*, which includes intervals (0-100)
- *file* type allows the user to upload a file
- A list of possible choices separated by commas, described after an `=` sign. Example: (`1 = enable, 0 = disable`).
- *IP* and *MAC* types are used to mark arguments containing IP or MAC addresses

Example: will also be used to automatically test python functions

The keyword `_ignore_test_` can be used to mark functions to be excluded from automatic tests

Interpretation rules

- <required> or [optional] arguments
- Description of arguments after ':' or '-'
- Optional types
 - (int, dec, str, ...)
- Intervales (ex. : 0-255)
- Multiple choices (ex.: 1=enable, 0=disable)
- Types hierarchy: intervalles $\subset$ integer $\subset$ hexa $\subset$ string
- Several types provided (ex. : string **or** hexa)
- Addresses IP, MAC
- Files to be uploaded by the browser
- Provided examples used for automatic tests

Arguments edit zones

- Argument types examples :

- Intervales, integers
- Strings
- Multiple choices
- Files

- Functions without arguments:

- Direct launch or
- User confirmation

Action STARE_generator_initialize_and_enable_v4

Argument	Description	Default	Value	Type
flow*		-	flow	Mb/s
package_size*		-	package_size	bytes
channel*		-	-- Select --	str
adf_format*		-	-- Select --	str

[Cancel](#)

[Launch](#)

Initializes and enables STARE data generators on SFP modules (0, 1, 2, 3, all).
Usage:

<flow (Mb/s)> <package_size(bytes)> <channel (0,1,2,3,all)> <adf_format (0 = byteANDpackage_counter, 1 = adf_CPT, 2 = adf_RAM, 3 = adf_DTH)>

Example:

2000 8192 3 2

Action STARE_change_emulator_ram

Argument	Description	Default	Value	Type
filename*		-	Browse... No file selected.	file
emulator_number*		-	0	int [0-3]
ram_type*		-	1	int [0-1]

[Cancel](#)

[Launch](#)

Load emulator RAM content from a file.

Arguments:

<filename (file)> <emulator_number (0-3)> <ram_type (0-1)>

Automated tests

- PACE and STARE functions represent 20,000 code lines
- Automatic tests procedure:
 - Test empty Database creation
 - Actions launched (examples, min, max values)
 - Approximately 200 automated tests
 - Fake arguments of type file generated
 - Use of an IPbus dummy machine device
- Actions can be marked to be ignored by tests (only 2)

Source code coverage

Modules	statements	missing	excluded	coverage
slowControl/pace/lib/__init__.py	0	0	0	100%
slowControl/pace/lib/aEth.py	812	444	0	45%
slowControl/pace/lib/agg.py	369	306	0	17%
slowControl/pace/lib/clock.py	1679	1563	0	7%
slowControl/pace/lib/constants.py	19	2	0	89%
slowControl/pace/lib/digi.py	1384	1054	0	24%
slowControl/pace/lib/sensors.py	906	726	0	20%
slowControl/pace/paceIPbus.py	246	163	0	34%
slowControl/Ph2_PACE_CMD.py	110	47	0	57%
slowControl/lib/pace_basic.py	854	606	0	29%
slowControl/lib/pace_call.py	305	147	0	52%
slowControl/lib/pace_dtph.py	993	309	0	69%
slowControl/lib/pace_gts.py	523	209	0	60%
slowControl/lib/pace_mon.py	1591	1372	0	14%
slowControl/lib/pace_read.py	1305	827	0	37%
slowControl/lib/pace_recv.py	1640	1378	0	16%
slowControl/lib/pace_sc.py	801	390	0	51%
slowControl/lib/stare.py	1123	362	0	68%

- Once the tests finished, results can be seen at: </coverage/index.html>

Source code coverage

- Reasons for uncovered lines:
 - Undocumented or not well documented functions (PACE not yet done)
 - Explicitely ignored functions (a few ones)
 - Tests for invalid argument values already done at higher level
 - Variable numbers of arguments
 - Actions depending on values read through IPbus
 - Exceptions processing
 - Old code lines or other never called functions

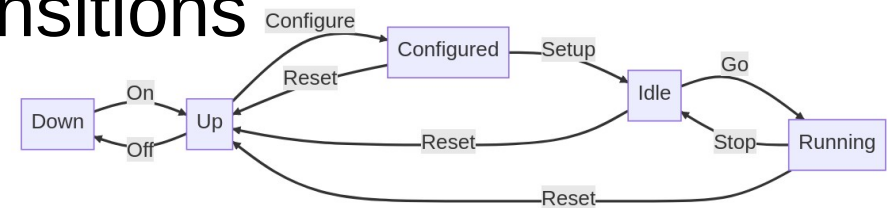
Console

- `./console.sh` **(on the Slow Control server)**
- `csl.launchActionCard(0, "A", "STARE_read_idbitstream", "")`
- `csl.history([n])` **prints [n lines] history of actions**
- `csl.getParamValue(0, "A", "sys_manag.ID_bitstream")`
- `csl.setDefaultCard(0, "A")` **or** `csl.setDefaultCard(name)`
- `csl.setDefaultAction("STARE_10g_status 3")`
- `csl.quickLaunch([""])`
- `c=csl.getCardCmd(0, "A")` **or** `csl.getCardCmd(name)`
- **Examples**
 - `c.do_STARE_read_idbitstream("arguments")`
 - `c.help_PACE_init()`

Name and Repositories

- Slow Control software name : **AgaPilot**
- ~~Github since february~~
- Firmware repository: ~~IN2P3 Gitlab agata_stare~~
- Software repository [IN2P3 Gitlab agata/daq/pacecontrol](#)
- Software repository [IN2P3 Gitlab agata/daq/starecontrol](#)
 - Contains then integrated PaceControl sources

- Test with several PACE/STARE cards simultaneously
- Increase source code coverage with automated tests
- Continue PACE/STARE functions integration work
 - Associate actions to FSM transitions



- Automated tests with non-empty dummy IPbus machines
- Update technical documentation

Thank you for your attention

