



# Hardware Triggered Scanning: Malcolm

**Philip Taylor, Emma Arandjelovic**  
**Observatory Sciences Limited**



# Software Stack: Reminder



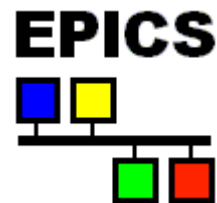
Data Analysis WorkbeNch  
- Analysis and visualization



Generic Data Acquisition  
- Experiment setup and supervision




**Malcolm**  
- Scan configuration



Experimental Physics &  
Industrial Control System  
- Low level control of hardware



# What is Malcolm?

- Generic and extensible framework for scanning
- Middle layer between GDA and control system
- Implemented in  python™
- Creates a s/w map of the h/w layer
- <https://pymalcolm.readthedocs.io>
- Web GUI called MalcolmJS
- <https://malcolmjs.readthedocs.io>



# Malcolm Process

Each Malcolm instance is a Python process managed by procServ  
On beamlines this runs on ixx-control

```
[p47user@localhost]$ ioc-list
```

```
BL47P-EA-IOC-01: pid = 2628, telnet port = 7002
```

```
BL47P-ML-MALC-01: pid = 2663, telnet port = 7003
```

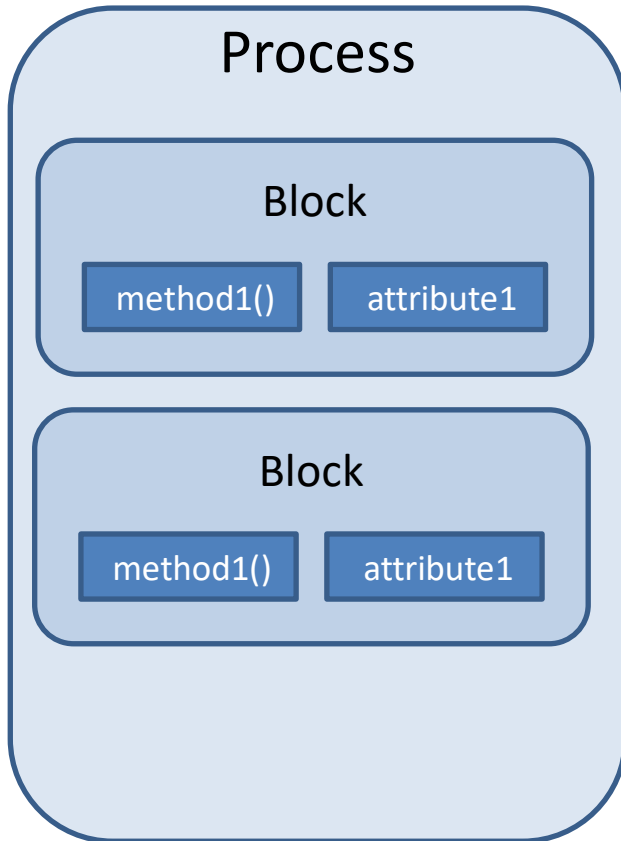
To reboot:

```
ioc-connect BL4xP-ML-MALC-01
```

```
Ctrl-X
```



# Malcolm Concepts

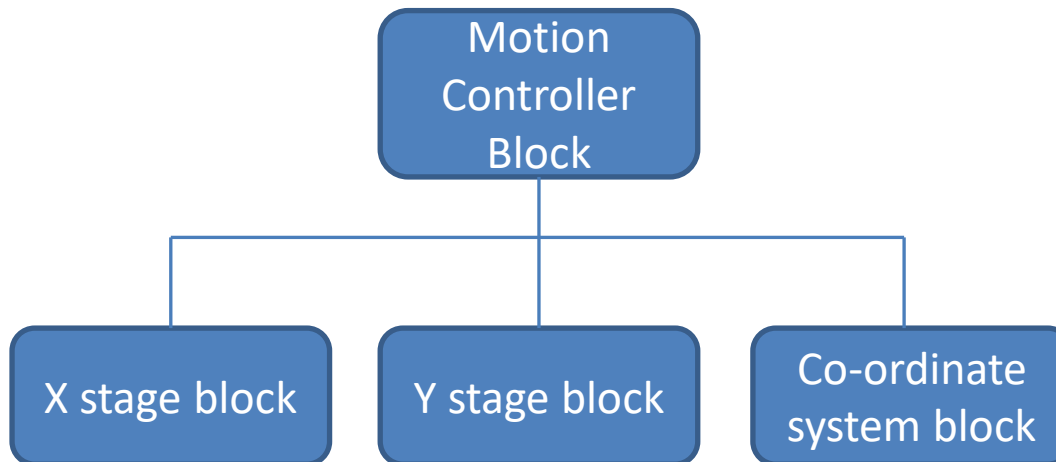


- A **block** is a user-centred view  
Examples:
  - Motion controller
  - Panda
  - Detector
  - 'Hello World' program
- Blocks contain:
  - **Methods** (actions)
  - **Attributes** (data)
- A **process** hosts multiple blocks
- Parent blocks contain child blocks connected together in a **design**



# Malcolm Concepts

- Each block has a unique name called an [MRI \(Malcolm Resource Identifier\)](#) e.g. BL47P-ML-SCAN-01
- A [device block](#) is a higher level block for synchronizing a number of child blocks

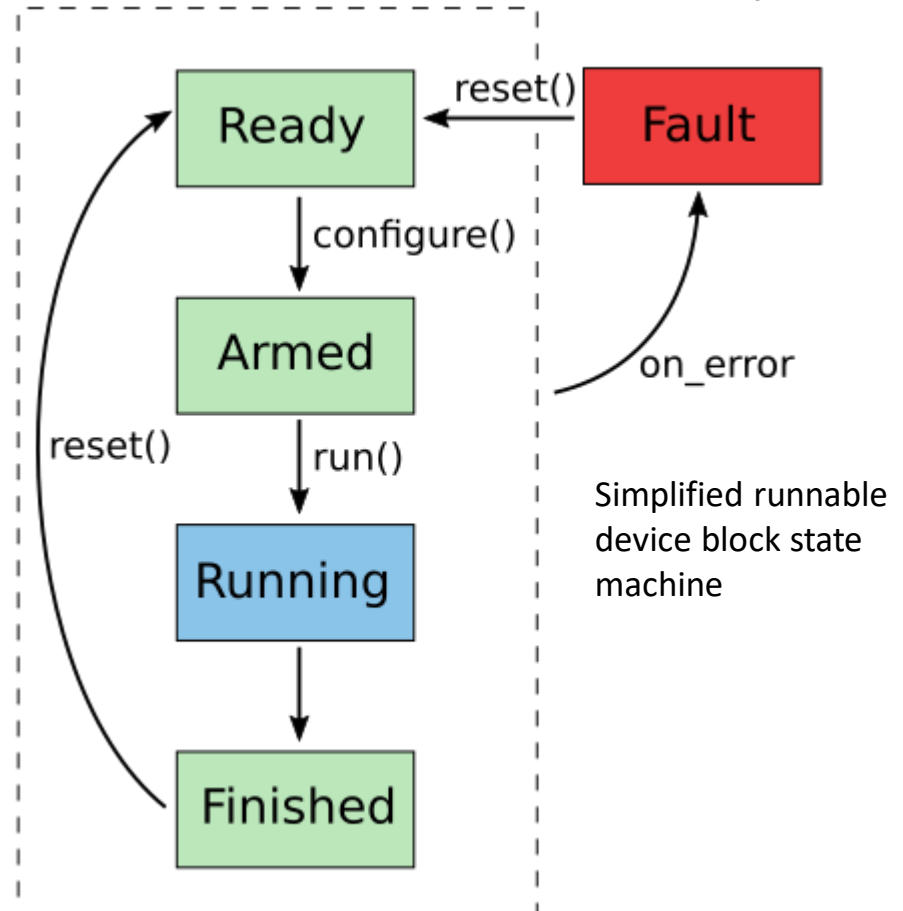




# Malcolm Concepts

- A *runnable device block* adds state machine functionality with two main methods:

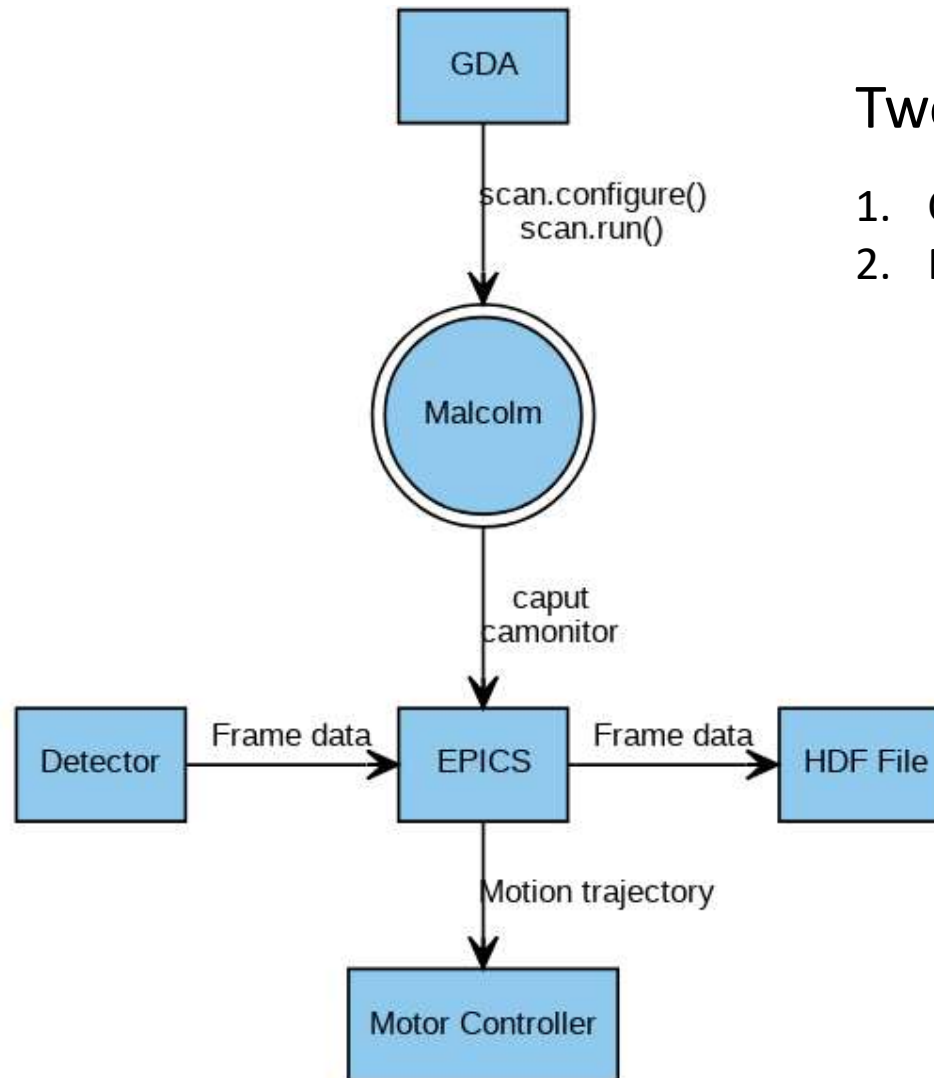
1. **configure(params)**
  - setup all child blocks
2. **run()**
  - start all children and supervise their progress



# Control Flow

Two step process:

1. Configure the scan
2. Run the scan







# Web GUI: Overview

*Left: parent block  
(in focus)*

*Central panel: design layout  
/ attribute view*

*Right: inspector*

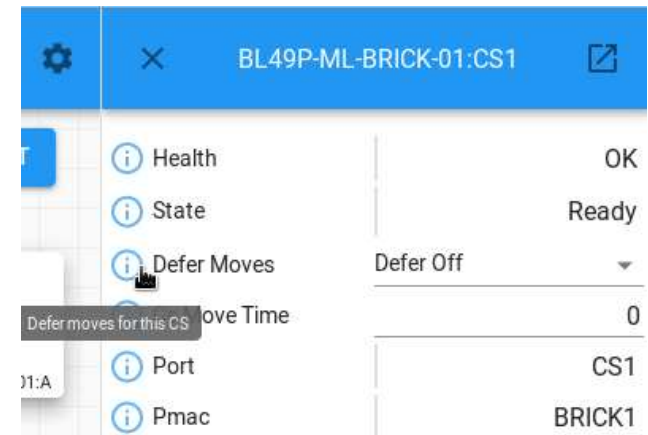
The screenshot displays the Web GUI interface, divided into three main panels:

- Left Panel (Parent Block):** Shows the 'BL49P-ML-BRICK-01' block. It includes a list of attributes (Health, State, Layout, Design, Exports, Modified, Label, Points Scanned) and a set of controls (DISABLE, RESET, EXECUTE PROFILE, ABORT PROFILE). The 'Layout' attribute is highlighted in green.
- Central Panel (Design Layout):** Displays a visual representation of the design layout. It shows a central 'CS1' block connected to 'stagex', 'stagea', 'STATUS', and 'TRAJ' blocks. The 'CS1' block is highlighted with a green border.
- Right Panel (Inspector):** Shows the 'BL49P-ML-BRICK-01:CS1' block. It includes a list of attributes (Health, State, Defer Moves, Cs Move Time, Port, Pmac, Demand A, Demand B, Demand C, Demand U, Demand V, Demand W, Demand X, Demand Y, Demand Z) and a set of controls (DISABLE, RESET, ABORT).



# Web GUI: Block View

- Attributes displayed depend on the type of block
- Tool-tip text provide descriptions
- Any problems connecting to the h/w are displayed in the tool-tip



Demand A	0.0000
Demand B	0.0000
Demand C	0.0000
Demand U	0.0000
Demand V	0.0000
Demand W	0.0000
Demand X	0.0000



# Web GUI: Attribute View

Displays table or plot of historical data values





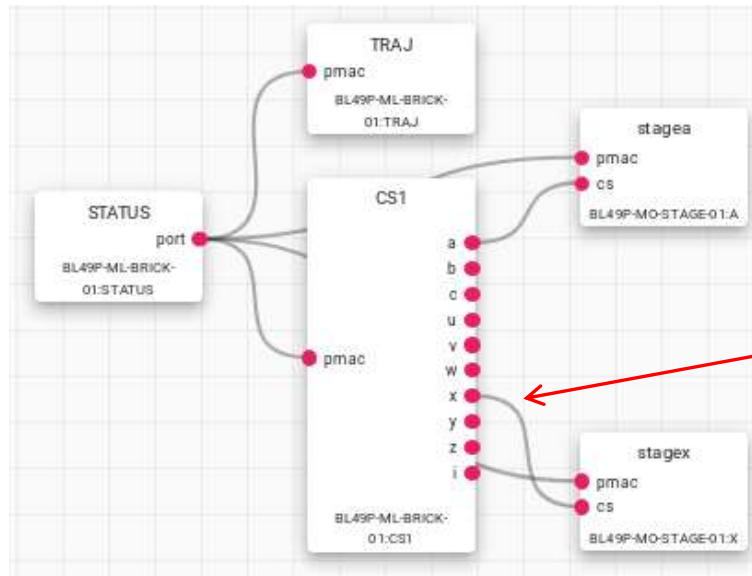
# Web GUI: Layout View

Graphical view of selected root block, with automatic layout feature

A screenshot of the Web GUI Layout View for a LEGO Mindstorms brick. The interface is divided into a left sidebar and a main canvas. The sidebar on the left contains a list of properties: Health (OK), State (Ready), Layout (selected and circled in red), Design (a\_z\_in\_cs1), Exports, Modified, Label (Brick with X and A Samp), and Points Scanned (12297 / 12297). Below the properties are buttons for DISABLE, RESET, Save, Move C S1, Servo Frequency, and Write Profile, followed by EXECUTE PROFILE and ABORT PROFILE. The main canvas on the right shows a graphical layout of the brick's internal components. At the top of the canvas, there are three green circular icons (magnifying glass, zoom in, zoom out) and a blue button labeled "AUTO LAYOUT" which is circled in red. The layout itself consists of several blocks: a central "CS1" block, a "STATUS" block to its left, a "TRAJ" block above it, and two "stage" blocks ("stagea" and "stagex") to its right. Each block has ports labeled with letters (a-z) and "i". The "STATUS" block has a "port" label. The "TRAJ" block has a "pmac" label. The "stagea" and "stagex" blocks have "pmac" and "CS" labels. The "CS1" block has a "pmac" label. The "CS1" block is connected to the "TRAJ" block, the "stagea" block, and the "stagex" block. The "STATUS" block is connected to the "CS1" block. The "CS1" block is also connected to a vertical column of ports labeled a, b, c, u, v, w, x, y, z, and i. The "AUTO LAYOUT" button is located in the top right corner of the canvas.



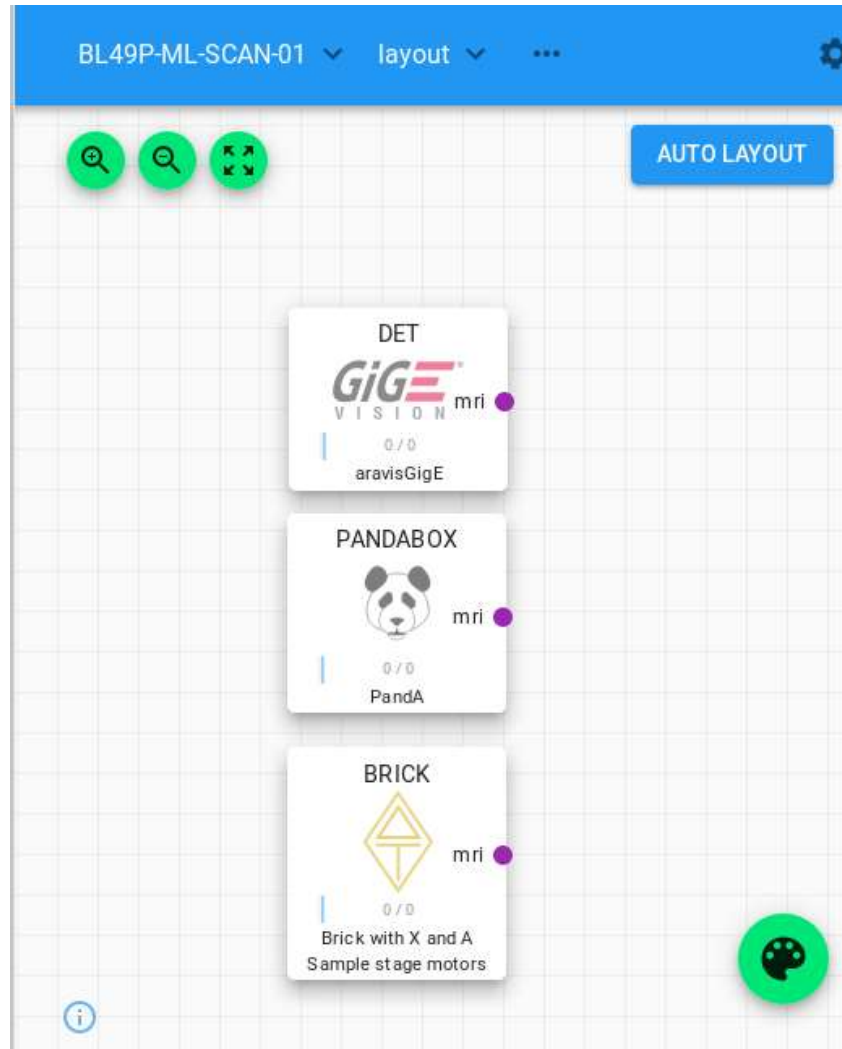
# Web GUI: Layout View



‘Wire-up’  
components visually



# Web GUI: Layout View

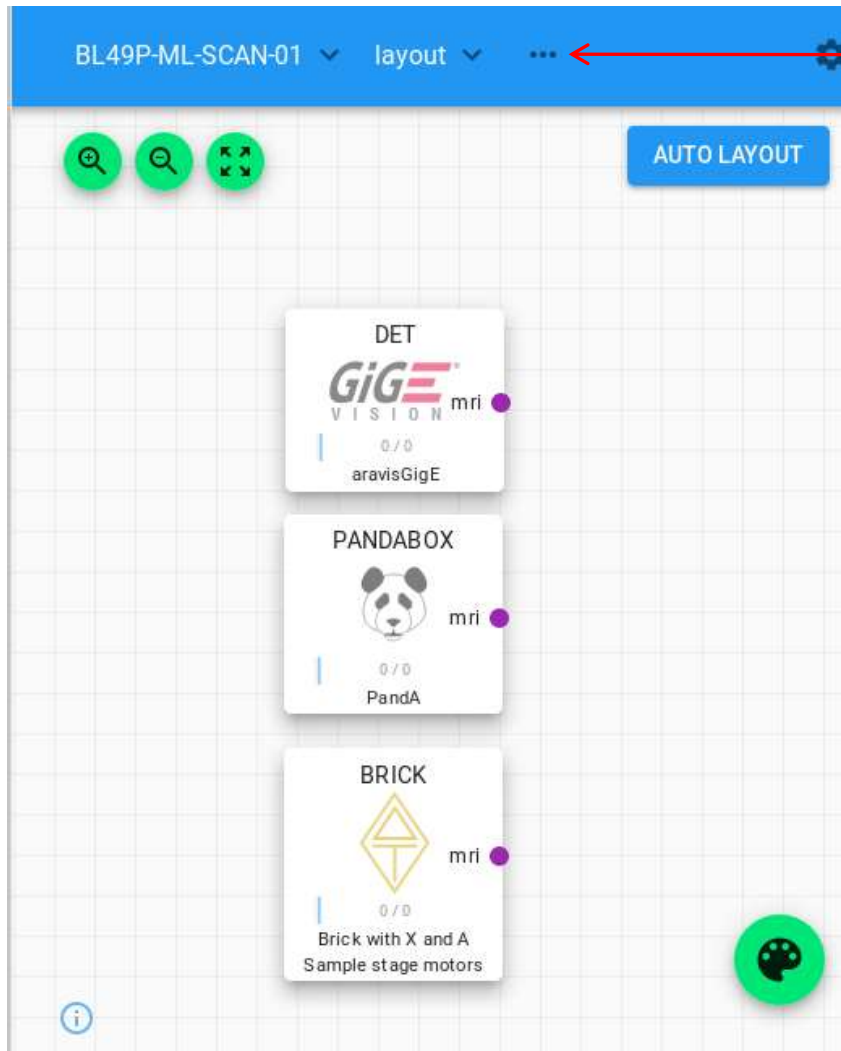


Auto Layout – first step for complex layouts

Palette – drag and drop new items



# Web GUI: Navigation



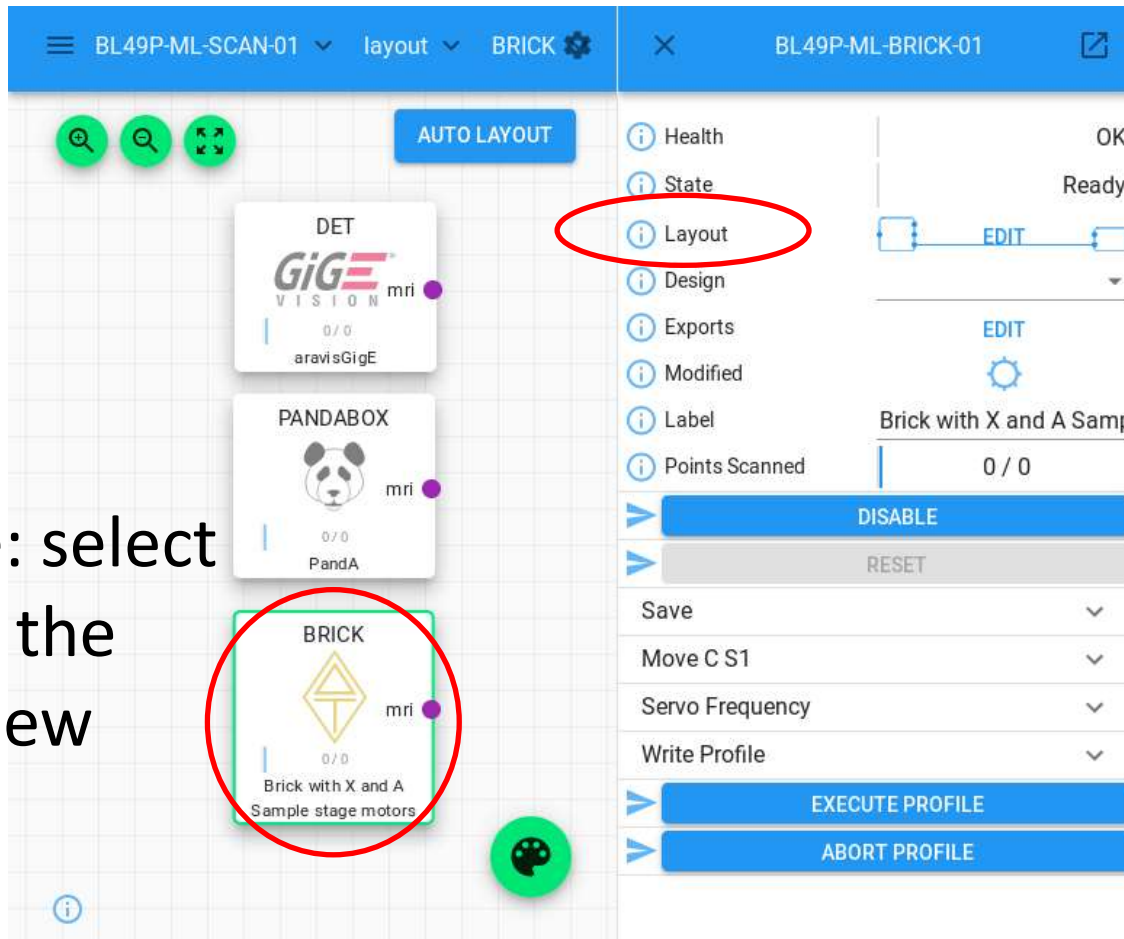
Breadcrumb trail in central panel

Starting with the selected root block, you can drill down into the design





# Web GUI: Navigation



Example: select *BRICK* in the layout view

Then choose *Layout* in the right hand inspector

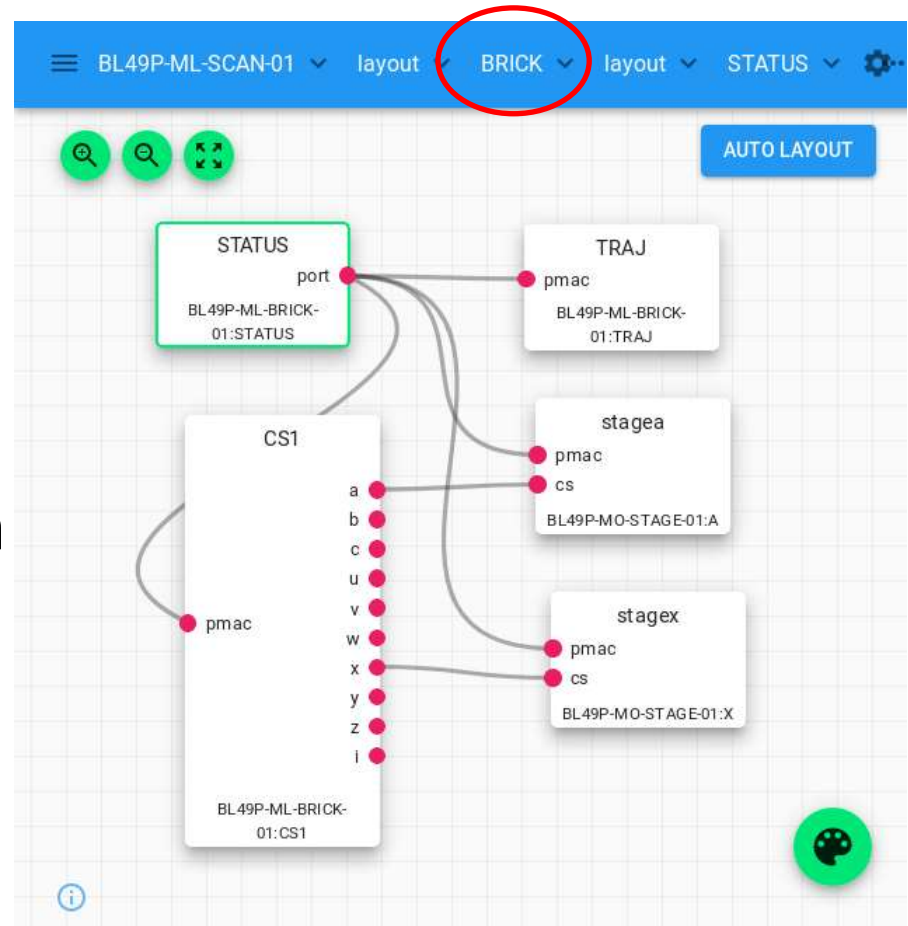




# Web GUI: Navigation

Now focused on the BRICK block:

1. Controller
2. Trajectory
3. Co-ordinate system
4. Axes
5. Status



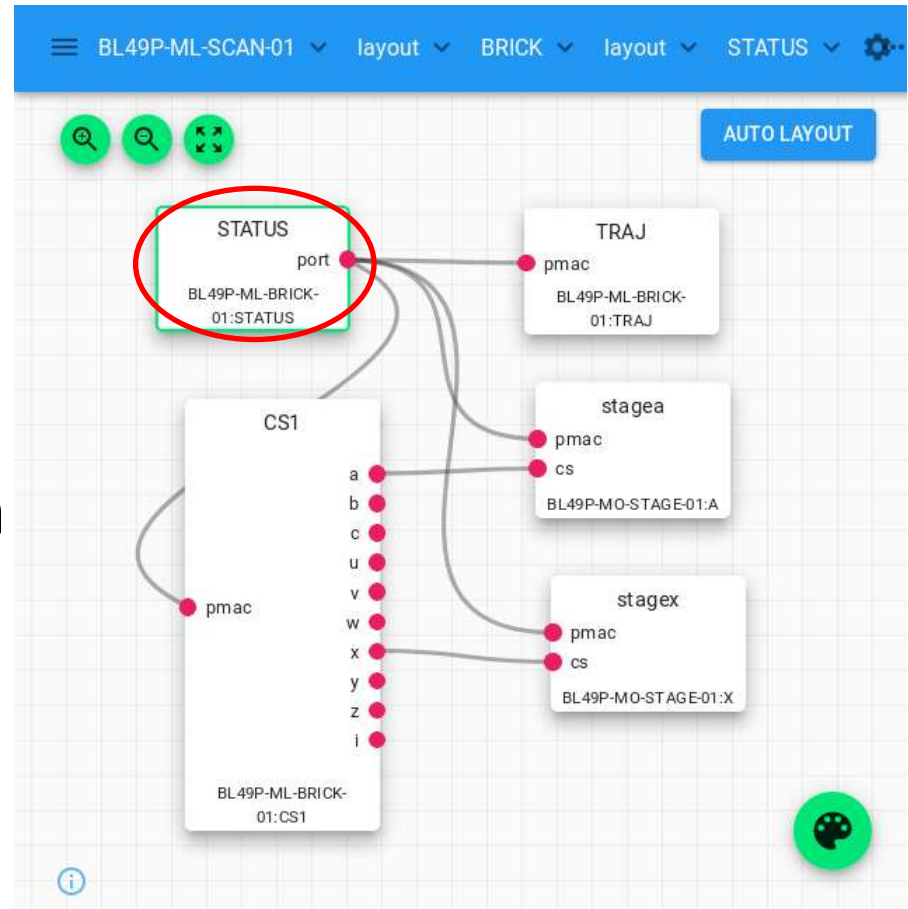


# Web GUI: Navigation

Now focused on the  
BRICK block:

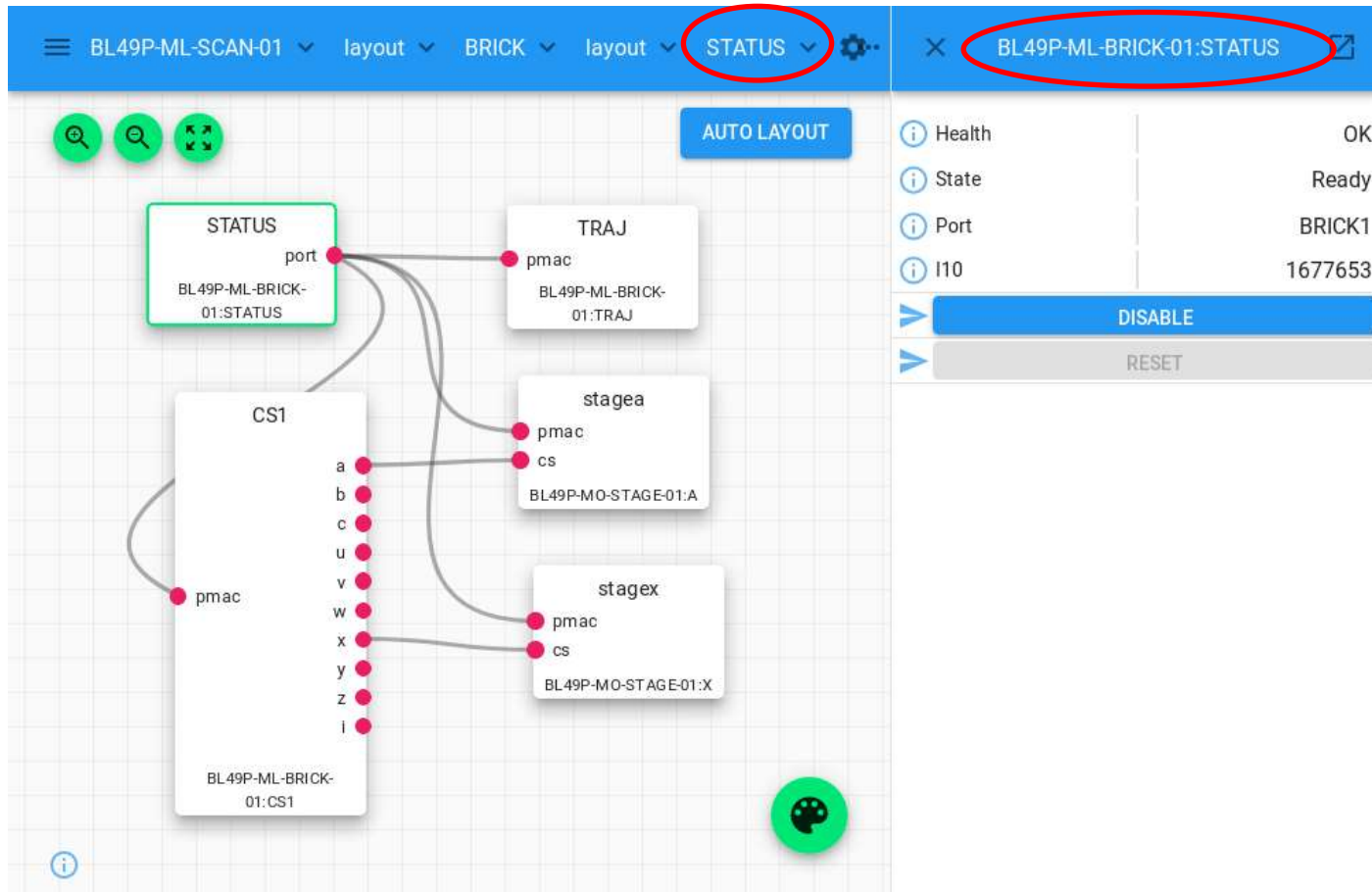
1. Controller
2. Trajectory
3. Co-ordinate system
4. Axes
5. Status

Select STATUS





# Web GUI: Navigation





# Exercise: Web GUI



- Navigate to the PANDA block
- Try to toggle on/off the light controlled by **TTLOUT2**
- For more info on the web GUI:  
<https://malcolmjs.readthedocs.io>



# Designs

Different scan requirements can be implemented using multiple **designs**

**Example: a new experiment requires a different PandA configuration**

**Step 1:** Open the PANDA block in the layout view

**Step 2:** Modify the design as required

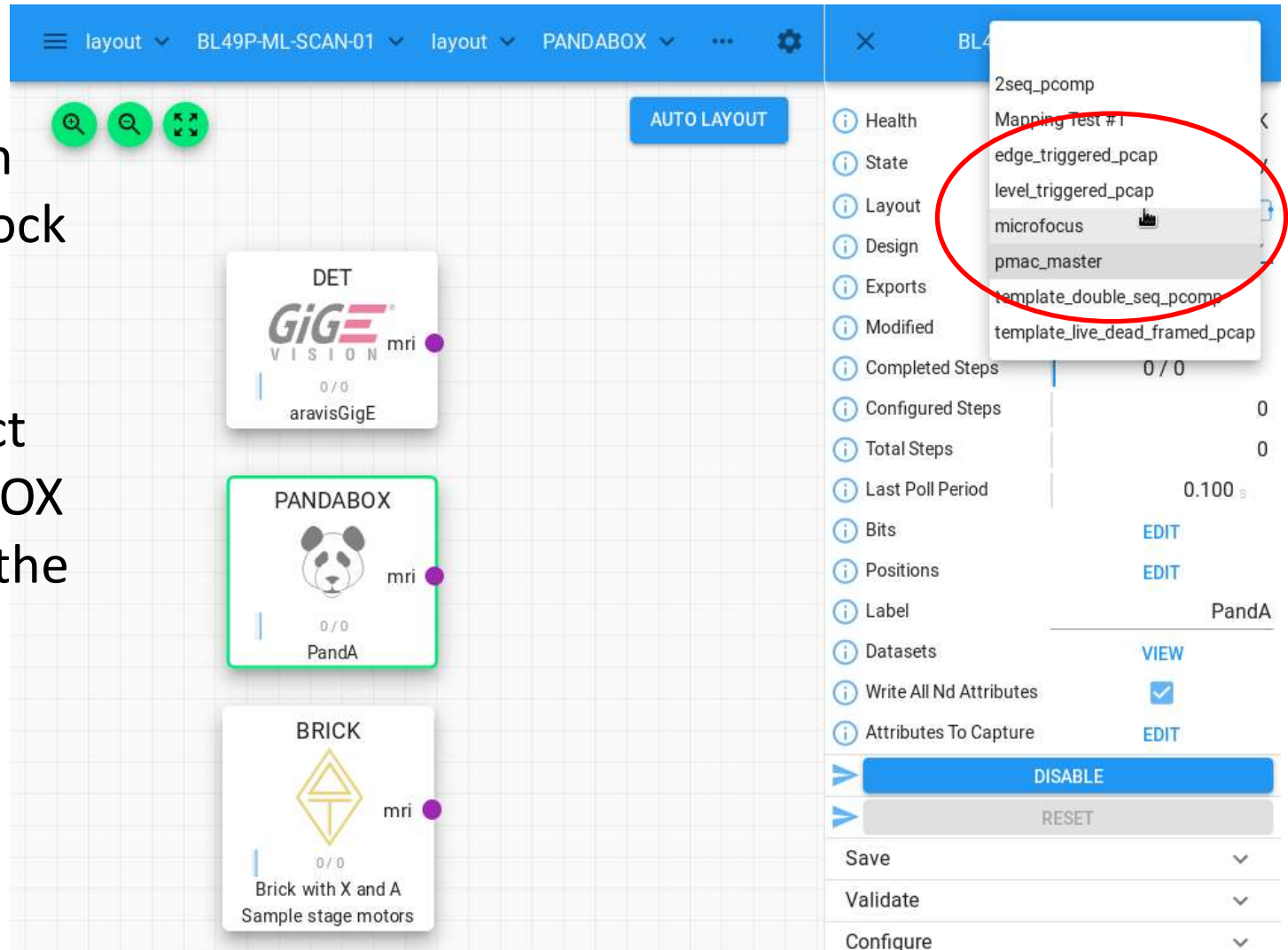
**Step 3:** Choose a name and save

A screenshot of the BL49P-ML-PANDA-01 configuration interface. The interface has a blue header with the title "BL49P-ML-PANDA-01". Below the header is a list of configuration items on the left and their corresponding values or controls on the right. The items are: Health (OK), State (Ready), Layout (EDIT), Design (pmac\_master), Exports (EDIT), Modified (gear icon), Completed Steps (0 / 0), Configured Steps (0), Total Steps (0), Last Poll Period (0.100 s), Bits (EDIT), Positions (EDIT), Label (PandA), Datasets (VIEW), Write All Nd Attributes (checked), and Attributes To Capture (EDIT). Below the list are two buttons: "DISABLE" and "RESET". At the bottom, there is a "Save" section with a "Design Name" field containing "microfocus" and a "SAVE" button. A red oval highlights the "Save" section.

# Designs

**Step 4:** Open the SCAN block layout

**Step 5:** Select the PANDABOX and choose the new design



The screenshot shows the PANDABOX software interface. The top bar displays the layout name 'BL49P-ML-SCAN-01' and the design name 'PANDABOX'. The main workspace shows three blocks: 'DET' (GigE VISION mri), 'PANDABOX' (Panda), and 'BRICK' (Brick with X and A Sample stage motors). The 'PANDABOX' block is highlighted with a green border. A dropdown menu is open over the 'PANDABOX' block, showing a list of designs. The design 'edge\_triggered\_pcap' is selected and highlighted with a red circle. The right sidebar shows the 'Health' tab with various status indicators and a 'DISABLE' button.



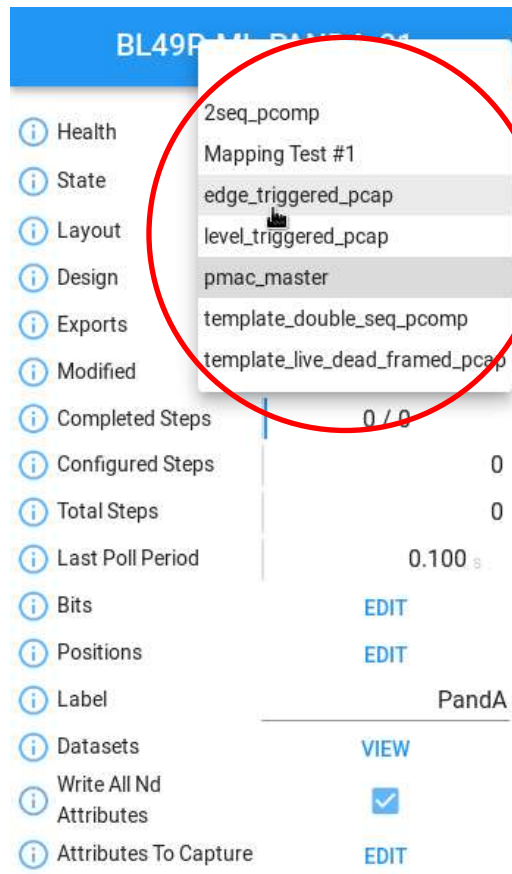
# Designs

**Step 6:** Using the left hand panel, save this SCAN configuration as a new design

The screenshot shows the configuration interface for "BL49P-ML-PANDA-01". The left-hand panel lists various configuration items, each with an information icon (i) to its left. The "Modified" item is circled in red. The right-hand panel displays the values for these items, with "OK" for Health, "Ready" for State, and "PandA" for Label. The "Completed Steps" section shows "0 / 0". The "Bits" and "Positions" sections have "EDIT" buttons. The "Attributes To Capture" section has an "EDIT" button. At the bottom, the "Save" section is circled in red, showing a "Design Name" field with the value "scan1" and a "SAVE" button.

BL49P-ML-PANDA-01	
Health	OK
State	Ready
Layout	EDIT
Design	▼
Exports	EDIT
Modified	
Completed Steps	0 / 0
Configured Steps	0
Total Steps	0
Last Poll Period	0.100 s
Bits	EDIT
Positions	EDIT
Label	PandA
Datasets	VIEW
Write All Nd Attributes	<input checked="" type="checkbox"/>
Attributes To Capture	EDIT
DISABLE	
RESET	
Save	
Design Name	scan1
SAVE	

# Designs



To load a design in the Web GUI:

1. Navigate to the root block
2. Select the design from the drop down list in the left hand panel





# Malcolm Configuration

- Blocks are reusable components that require configuration

Examples:

- PV prefix (for EPICS devices), IP address (for Panda)
  - Initial design
  - Runtime config directories
- Configuration specified in YAML  
(‘YAML Ain’t Markup Language’!)

```
[p49user@p49-pw001 ~]$ configure-ioc show BL49P-ML-MALC-01  
BL49P-ML-MALC-01 /dls_sw/work/R3.14.12.7/support/BL46P-BUILDER/etc/malcolm/BL49P-ML-MALC-01.yaml
```



# YAML Files

```
BL49P-ML-MALC-01.yaml
/dls_sw/work/R3.14.12.7/support/BL46P-BUILDER/etc/malcolm

1 #! /dls_sw/prod/common/python/RHEL7-x86_64/pymalcolm/4-2b3/malcolm/imalcolm.py
2 # This is intended for use with the test lab setup BL46P-MAP-IOC-01
3
4 - builtin.defines.module_path:
5     name: TestRigs
6     path: $(yamldir)
7
8 - TestRigs.includes.BL4xP_template:
9     rig_number: 49
10
```

YAML ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS



# YAML Files

```
Open ▾ [icon] /dls_sw/work/BL49P-ML-MALC-01.yaml BL4xP_template.yaml Save [menu] x
1 - builtin.parameters.string:
2   name: rig_number
3   description: Identifier number of Test Rig (e.g. 47 for p47-pw001)
4
5 - builtin.defines.string:
6   name: config_dir
7   value: /dls_sw/p$(rig_number)/epics/malcolm
8
9 - builtin.defines.string:
10  name: beamline
11  value: BL$(rig_number)P
12
13 # Motion Controller
14 - TestRigs.blocks.pmac_manager_block:
15   mri_prefix: $(beamline)-ML-BRICK-01
16   config_dir: $(config_dir)
17   beamline: $(beamline)
18
19 # Detectors
20 - aravisGigE.blocks.aravisGigE_runnable_block:
21   pv_prefix: $(beamline)-EA-DET-01
22   mri_prefix: $(beamline)-ML-DET-01
23   config_dir: $(config_dir)
24
25 - ADPAndABlocks.blocks.panda_runnable_block:
26   pv_prefix: $(beamline)-MO-PANDA-01
27   mri_prefix: $(beamline)-ML-PANDA-01
28   hostname: 192.168.250.18
29   config_dir: $(config_dir)
30
31 # Define the Scans
32 - TestRigs.blocks.scan_block:
33   mri_prefix: $(beamline)-ML-SCAN-01
34   beamline: $(beamline)
35   config_dir: $(config_dir)
36   initial_design: pmac_master_tomo
37
38 - system.defines.redirector_iocs:
39   name: iocs
40   value: $(beamline)-ML-DET-01
```

1 #! /dls\_sw/prod/common/python/R  
2 # This is intended for use with  
3  
4 - builtin.defines.module\_path:  
5 name: TestRigs  
6 path: \$(yamldir)  
7  
8 - TestRigs.includes.BL4xP\_template.yaml  
9 rig\_number: 49  
10



# YAML Files

```
1 #!/dls_sw/prod/common/python/R
2 # This is intended for use with
3
4 - builtin.defines.module_path:
5   name: TestRigs
6   path: $(yamldir)
7
8 - TestRigs.includes.BL4xP_template.yaml
9   rig_number: 49
10
11
12
13 - builtin.parameters.string:
14   name: rig_number
15   description: Identifier number of Test Rig (e.g. 47 for p47-pw001)
16
17 - builtin.defines.string:
18   name: config_dir
19   value: /dls_sw/p$(rig_number)/epics/malcolm
20
21 - builtin.defines.string:
22   name: beamline
23   value: BL$(rig_number)P
24
25 # Motion Controller
26 - TestRigs.blocks.pmac_manager_block:
27   mri_prefix: $(beamline)-ML-BRICK-01
28   config_dir: $(config_dir)
29   beamline: $(beamline)
30
31 # Detectors
32 - aravisGigE.blocks.aravisGigE_runnable_block:
33   pv_prefix: $(beamline)-EA-DET-01
34   mri_prefix: $(beamline)-ML-DET-01
35   config_dir: $(config_dir)
36
37 - ADPAndABlocks.blocks.panda_runnable_block:
38   pv_prefix: $(beamline)-MO-PANDA-01
39   mri_prefix: $(beamline)-ML-PANDA-01
40   hostname: 192.168.250.18
41   config_dir: $(config_dir)
42
43 # Define the Scans
44 - TestRigs.blocks.scan_block:
45   mri_prefix: $(beamline)-ML-SCAN-01
46   beamline: $(beamline)
47   config_dir: $(config_dir)
48   initial_design: pmac_master_tomo
49
50 - system.defines.redirector_iocs:
51   name: iocs
52   value: $(beamline)-ML-SCAN-01
```



# Exercise: pvAccess



- Malcolm<->GDA interface uses [pvAccess](#) (EPICS V4)
- Each block is represented as a V4 [process variable \(pv\)](#) with a number of [fields](#)
- Use *pvget* to dump the entire SCAN block structure  
[pvget BL4xP-ML-SCAN-01 -r ""](#)
- (If *pvget* not in PATH, run [module load pvatools](#))



# pvAccess continued



- -r '*request*' option specifies which fields to return  
`pvget BL4xP-ML-SCAN-01 -r health.value`
- What information comes back for the *layout* field?
- Other useful SCAN block fields include:
  - `totalSteps`
  - `completedSteps`
  - `state`



# Exercise: Scan States






- Start a scan in GDA and monitor the SCAN block in the Malcolm web GUI. What states does it go through?
- The -m (*'monitor'*) option to pvget monitors the PV for changes. Use this together with the -r option to monitor the *completedSteps* field from the command line as the scan progresses.





# Scan States Screenshot

BL49P-ML-SCAN-01		layout ▼ BL49P-ML-SCAN-01 ▼ state ▼ ...		
		TIME SET	VALUE	
Health	OK			
State	Ready	2019-12-02T13:37:29.521Z		Ready
Layout	 EDIT 	2019-12-02T13:45:35.840Z		Aborting
Design	pmac_master_tomo ▼	2019-12-02T13:45:36.111Z		Aborted
Exports	EDIT	2019-12-02T13:45:36.157Z		Resetting
Modified		2019-12-02T13:45:36.211Z		Ready
Completed Steps	0 / 6000	2019-12-02T13:45:36.538Z		Configuring
Configured Steps	0	2019-12-02T13:45:42.544Z		Armed
Total Steps	0	2019-12-02T13:45:44.300Z		Running
Label	PMAC Master Tomograp	2019-12-02T13:45:58.870Z		PostRun
Simultaneous Axes	EDIT	2019-12-02T13:45:58.890Z		Finished
Datasets	VIEW	2019-12-02T13:45:58.946Z		Resetting
Min Turnaround	0.000000 s	2019-12-02T13:46:00.890Z		Ready
Min Turnaround Interval	0.000000 s	2019-12-02T13:46:19.831Z		Aborting
DISABLE		2019-12-02T13:46:19.906Z		Aborted
RESET		2019-12-02T13:46:19.958Z		Resetting
Save	▼			
Validate	▼			





# Dataset table

BL49P-ML-SCAN-01

Health

State

Layout

Design

Exports

Modified

Completed Steps

Configured Steps

Total Steps

Label

Simultaneous Axes

Datasets

Min Turnaround

Min Turnaround Interval

DISABLE

RESET

Save

Validate

OK

Running

VIEW

pmac\_master\_tomo

VIEW

3000 / 10000

10000

10000

PMAC Master Tomogra

EDIT

VIEW

0.000000

0.006000

BL49P-ML-SCAN-01 datasets

	NAME	FILENAME	TYPE	RANK	PATH
	DET.data	htss-269-DET.h5	primary	4	/entry/detector/dete...
	DET.sum	htss-269-DET.h5	secondary	4	/entry/sum/sum
	stagex.value_set	htss-269-DET.h5	position_set	1	/entry/detector/stag...
	stagea.value_set	htss-269-DET.h5	position_set	1	/entry/detector/stag...
	stagex.value	htss-269-PANDABO...	position_value	4	/entry/stagex.value/...
	stagea.value	htss-269-PANDABO...	position_value	4	/entry/stagea.value/...
	stagex.value_set	htss-269-PANDABO...	position_set	1	/entry/detector/stag...
	stagea.value_set	htss-269-PANDABO...	position_set	1	/entry/detector/stag...

Up to date!

DISCARD CHANGES

SUBMIT