

Odin

*“Control and data acquisition framework
for parallel detector systems”*

Diamond Light Source

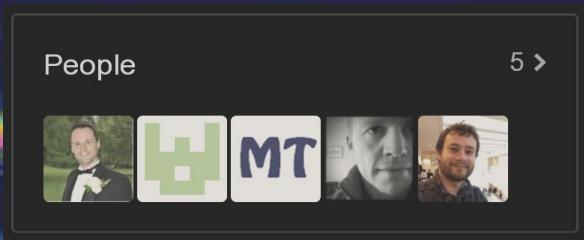
Gary Yendell



GitHub

The screenshot shows the GitHub interface for the 'odin-detector' repository. At the top, there's a logo consisting of three interlocking triangles in yellow, green, and blue. The repository name 'odin-detector' is displayed next to it. Below the logo, there are tabs for 'Repositories 2', 'People 5', 'Teams 2', and 'Projects 0'. A search bar with placeholder text 'Search repositories...' is followed by filters for 'Type: All' and 'Language: All'. A green button labeled 'New' is visible. The main content area displays two repositories: 'odin-control' and 'odin-data'. 'odin-control' is described as a 'Prototype of ODIN framework' and has tags for 'python', 'control', and 'detector'. It was last updated 4 days ago. 'odin-data' is described as 'DAQ software libraries for capturing and storing data from parallel detector systems' and has tags for 'C++', 'Apache-2.0', and 'detector'. It was also last updated 4 days ago. To the right, there's a 'Top languages' section showing C++ and Python, and a 'People' section showing five contributors with their profile pictures and initials (H, MT, etc.). There's also a small waveform graphic.

Collaborators



Logo



Cryptic Name

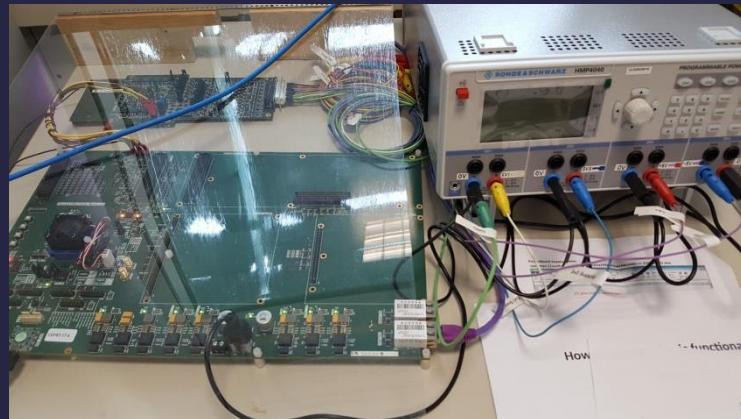
odin-detector



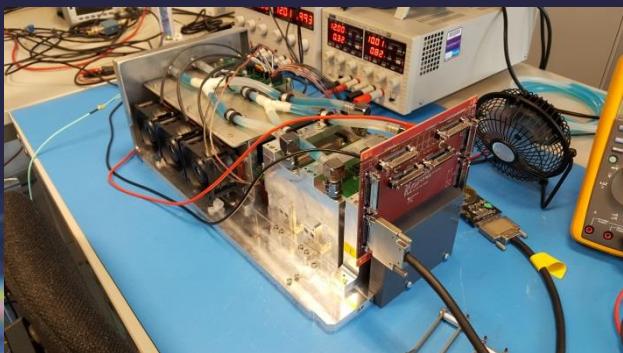
Odin Detectors



Excalibur



Percival

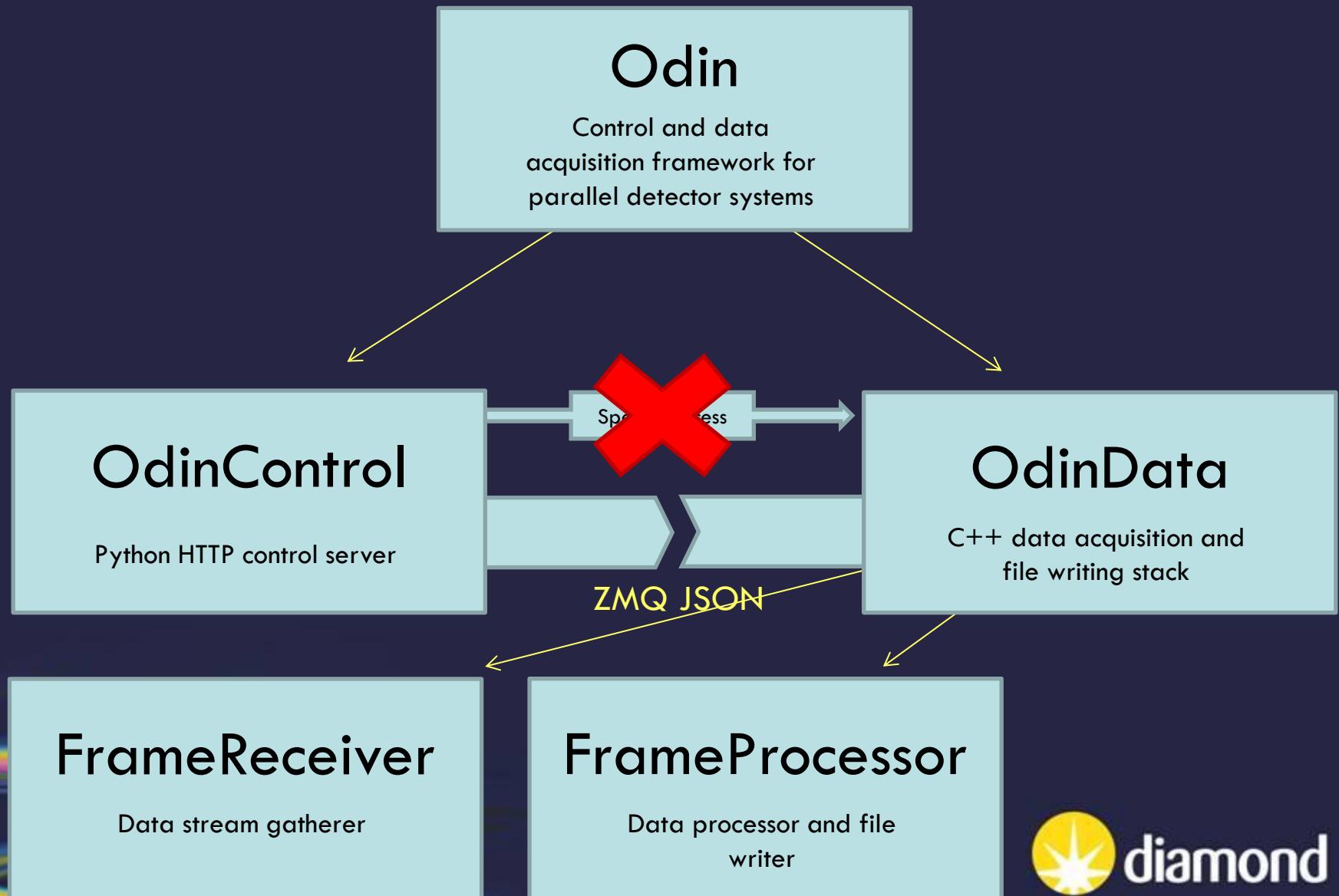


Tristan



Eiger...





ADOdin

Thin AreaDetector wrapper API



RESTful API

OdinControl

Python HTTP control server

TCP

Tristan

Percival

Excalibur



OdinData

C++ data acquisition and
file writing stack

ZMQ JSON

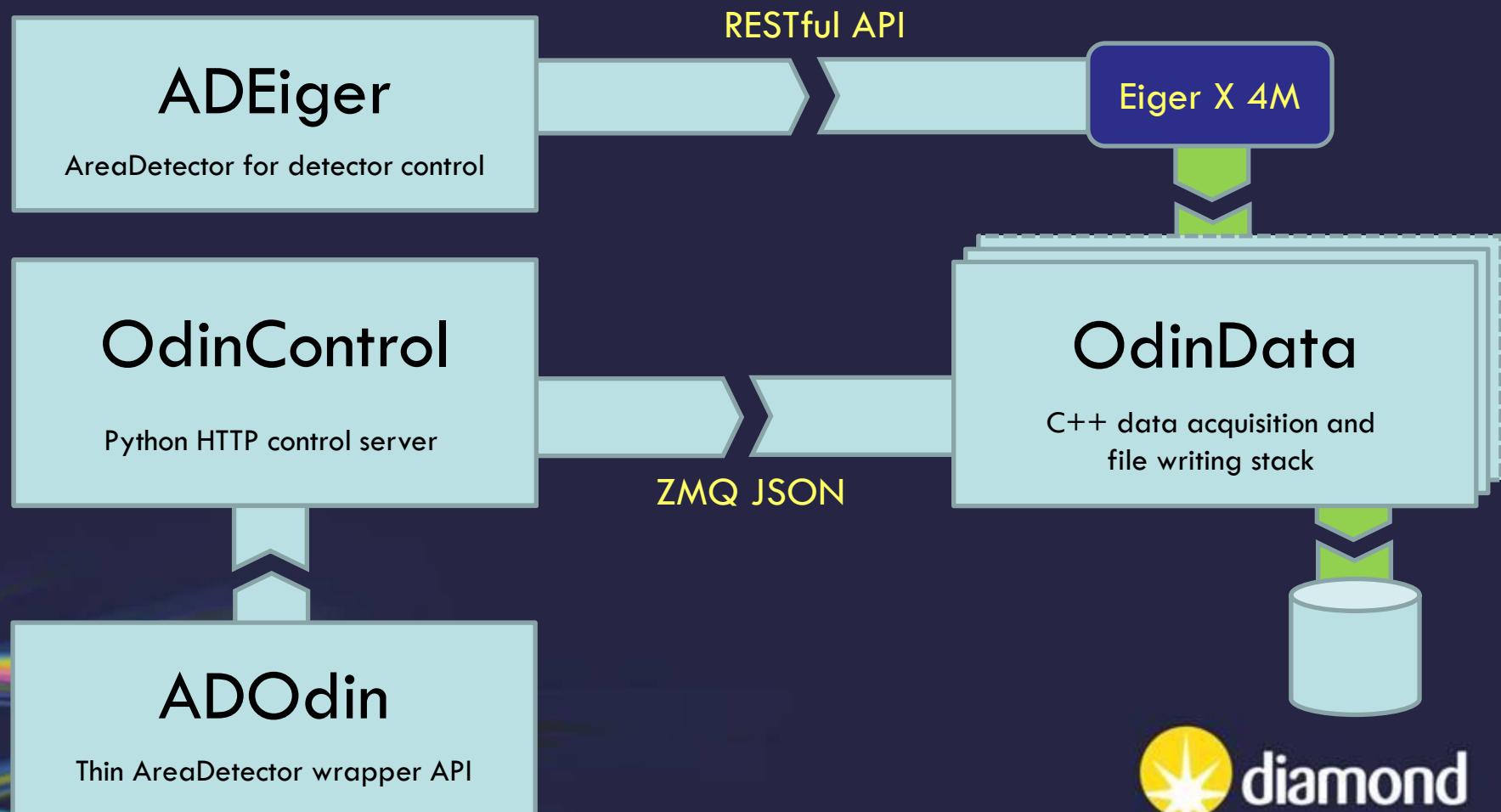


Control

Data

Control

Data



OdinControl

- Generic HTTP server
- Adapters provide functionality for each device
- Server generates RESTful API from arbitrary Adapter attributes and methods
- Control through simple webpages or client app

The screenshot displays two main sections of the OdinControl web interface:

Server Status

Attribute	Value
API Version:	0.1
Adapters Loaded:	percival
Server Start Time:	September 28, 2017 11:54:52
Server Up Time:	0:07:56.723233
Server Username:	gnx91527
Hardware Connection:	Address: 127.0.0.1 Port: 10001 Connected:
Database Connection:	Address: 127.0.0.1 Port: 8086 Name: percival Connected:

Auto-read Monitors (10 Hz): Stop

Control

Control Message Response:

Command: cmd_system_command
Parameters: name = start_acquisition
Execution Start Time: 2017-09-28 11:56:19.397969
Response: Completed
Message:

Download Channel Settings:

Initialise Channels:

System Command: start_acquisition

Set Channel Value: VDD_D2V5_1

Apply Set Point:

Upload Configuration File: ClockSettings.ini



OdinControl Server Configuration

- Configure server address
- Load Adapter libraries
- Configure specific Adapters

```
[server]
debug_mode = 1
http_port = 8888
http_addr = 0.0.0.0
adapters = excalibur, odin_data

[tornado]
logging = debug

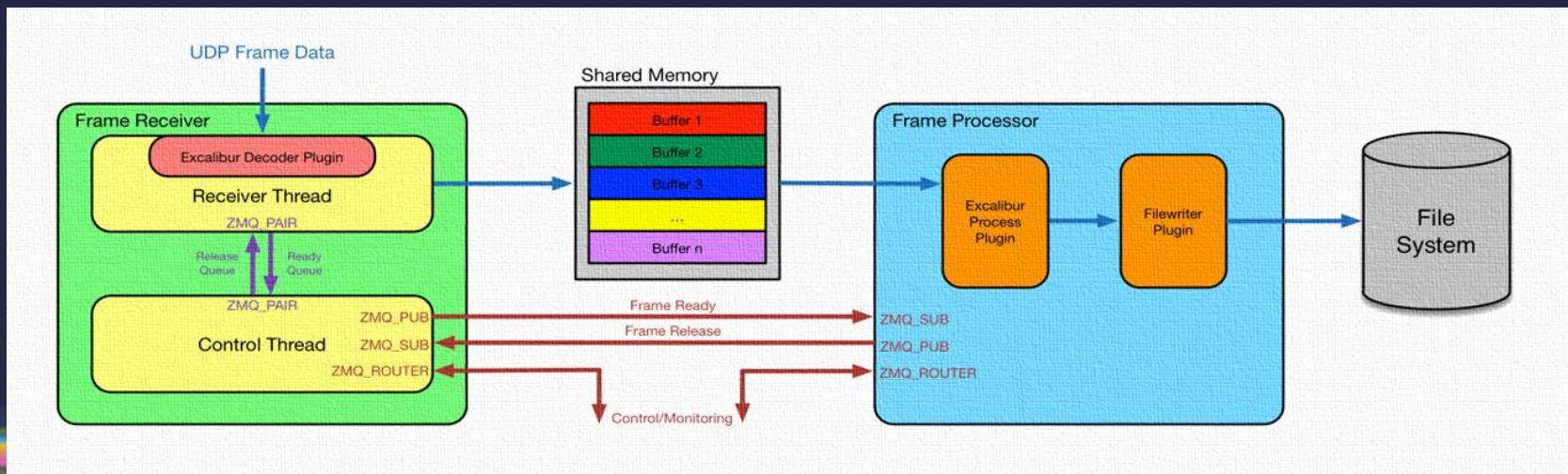
[adapter.excalibur]
module = excalibur.adapter.ExcaliburAdapter
detector_fems = 192.168.0.1:6969, 192.168.0.2:6969

[adapter.odin_data]
module = odin_data.odin_data_adapter.OdinDataAdapter
endpoints = 127.0.0.1:5004, 127.0.0.1:6004
update_interval = 0.5
```



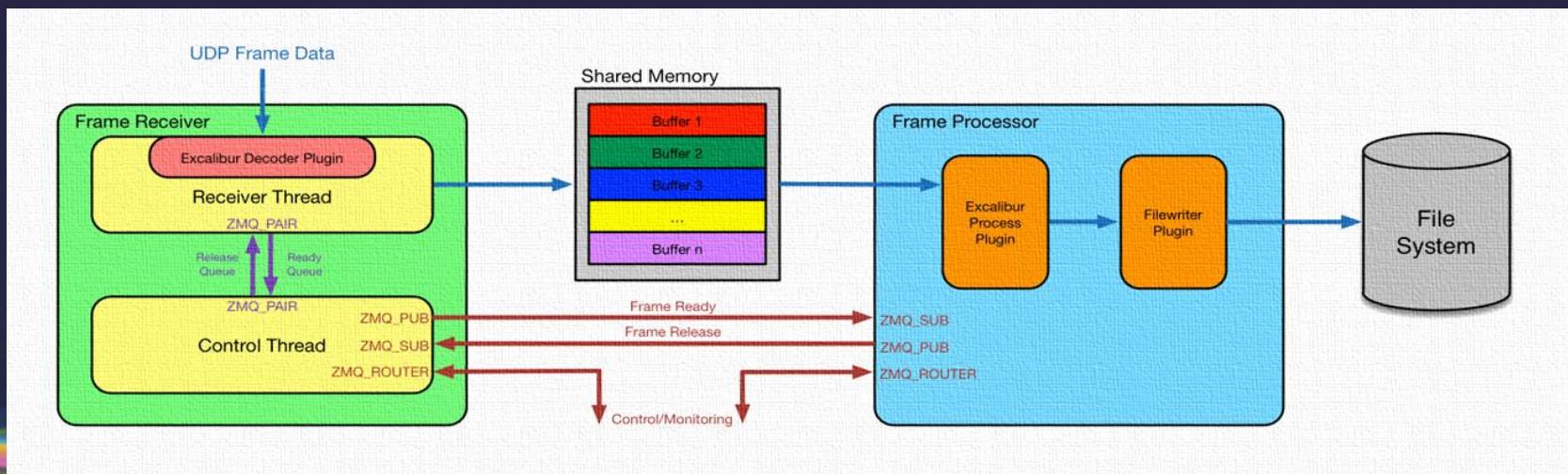
OdinData

- FrameReceiver gathers data from UDP or ZeroMQ stream
- Decoder plugin validates data packets and constructs a single frame
- Passes data frame to FrameProcessor through shared memory



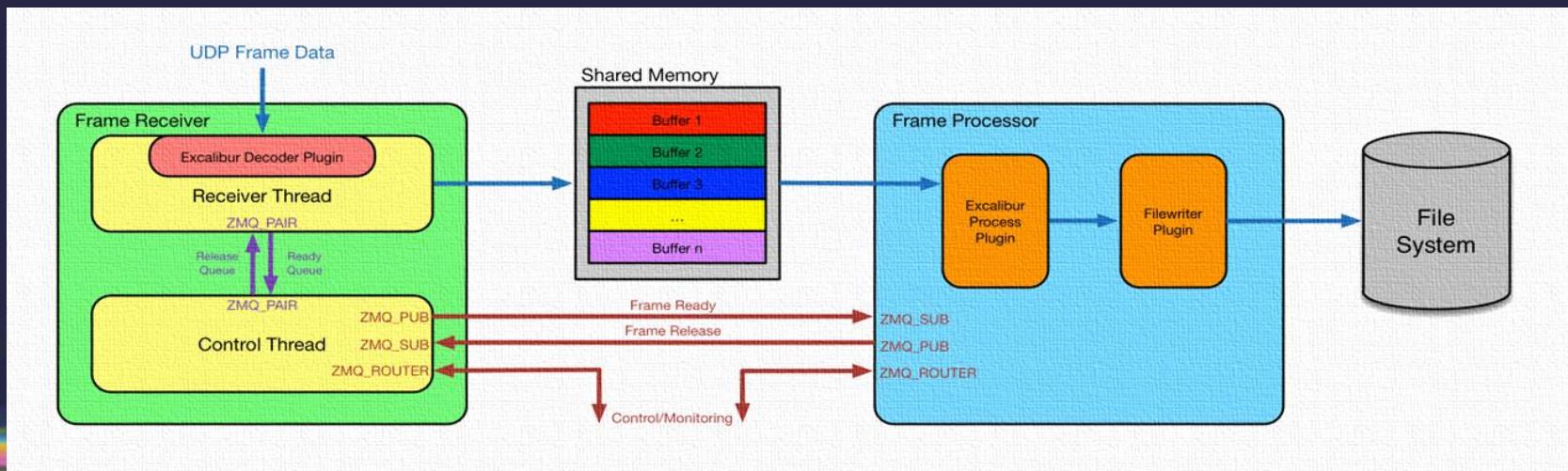
OdinData

- FrameProcessor receives message with frame pointer
- Process plugin performs data processing on the frame
- Detector agnostic FileWriterPlugin writes pre-chunked data directly to HDF5 dataset



OdinData

- Write data using Direct Chunk Write
- Only the raw dataset for best performance
- A single, live, view of the data produced using Virtual Dataset and SWMR functionality



Plugins

- Simple to write plugins to extend functionality
- Decoder plugin to convert data packets to frames
- Process plugin to construct data chunk



Plugins

- Simple to write plugins to extend functionality
- Decoder plugin to convert data packets to frames
- Process plugin to construct data chunk

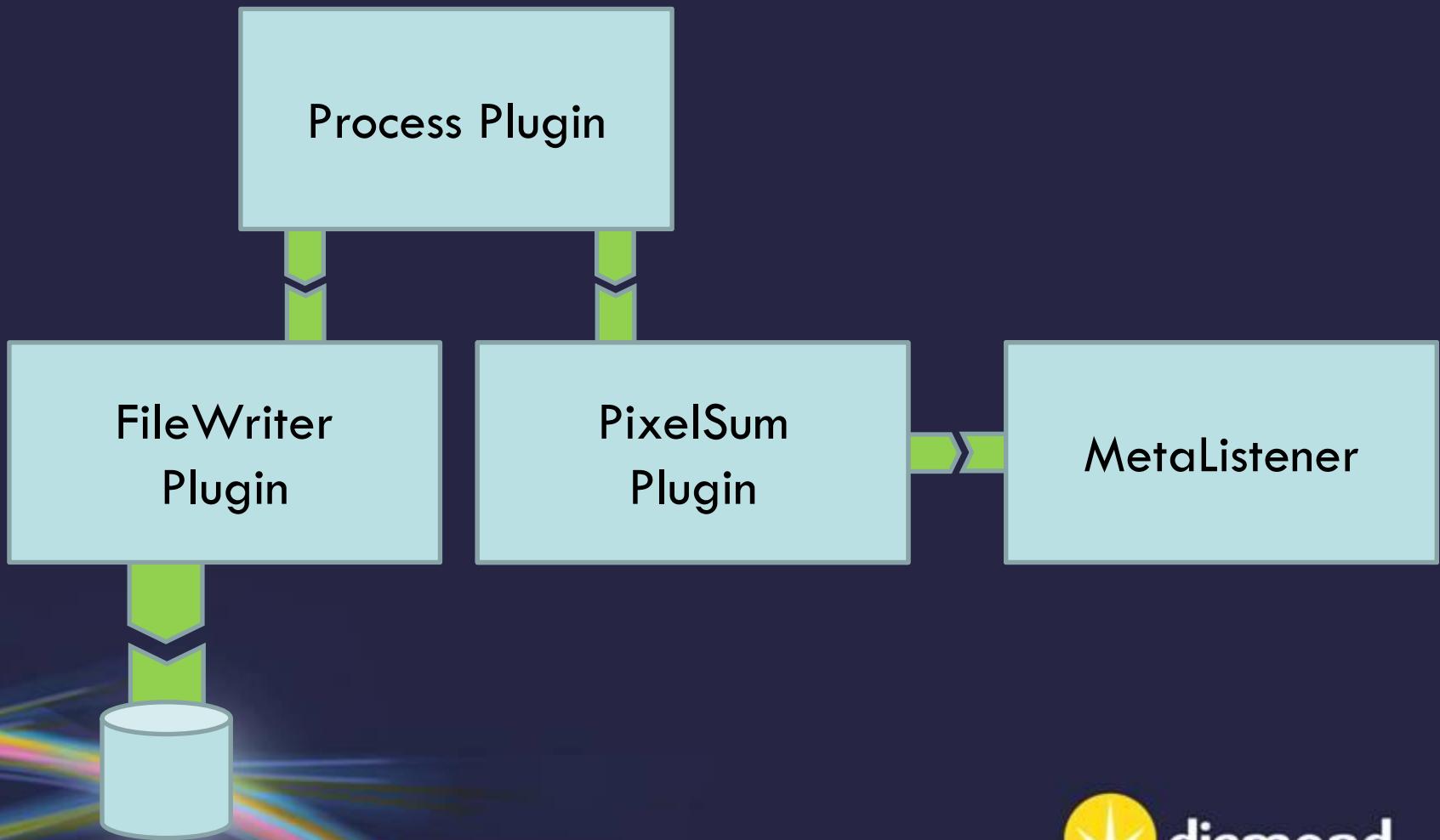


Processor Plugins

- Process plugin to construct data chunk
 - Inherit from FrameProcessorPlugin class
 - Implement process_frame()
 - connect() to an upstream plugin at runtime
- Resource-intensive plugins parallel to file writing plugin chain



Processor Plugins



OdinData Features

- Robust to packet loss, out of order packets, missing frames
- Perform bare minimum processing
- Write straight to disk with direct chunk write
- Scalable by addition of OdinData processes / server nodes



HDF5 Usage

- Direct Chunk Write: Fast writes; Pre-compression support
- SWMR: Live dataset view; Ensures file integrity after failure
- VDS: Single, coherent view of dataset split across many nodes



ADOdin

OdinData Configuration

HDF5

File Directory	/tmp		
File Name	test	test	
File Extension	h5	h5	
Frame Count	100	100	15
Start Timeout Stop Advanced			

Advanced HDF5 Configuration

Dataset Configuration

Image Width / Height:	2048	512	
Chunk Depth / Width / Height:	1	2048	512
Data Type:	UInt16		
Compression Type:	None		
File Configuration			
Initial Frame Offset:	0	0	
Timeout Period (ms):	0	0	
Frames per Block:	1	1	
Blocks per File:	0	0	
Chunk boundary alignment:	1 bytes	1 bytes	
Chunk boundary threshold:	1 bytes	1 bytes	
Earliest HDF5 Version:	No		

OdinData Status

	FR/FP 1	FR/FP 2
Status	█ █ █	█ █ █
Rank	0	1
Free Buffers	459	459
Frames Written	8	7
Total In Acq	50	50



ADOdin

Current Excalibur API

Info

Port	Run	State	Idle
Model		BL-ID-EA-EXCALIBUR-DI	
Image			
Sensor Size		X 2048	Y 1796
Image Size		2048	1796
Image Bytes		731648	
Test Pulse Enable		Off	<input type="checkbox"/>
Node Status			
IOC	Cal	Arrays	
1:		3	3
2:		3	3
3:		3	3
4:		3	3
5:		3	3
6:		3	3
DAC Scan			
DAC #	0		
Start	30		
Stop	230		
Step	1		
Configuration			
Operation Mode	Normal	Read/Write Mode	Sequential
Counter Depth	12 bit	24 Bit Discriminator	DiscL
Counter Select	A	Equalization Mode	Off
Auto Load	Auto	Pixel Mode	Single Pixel
Number of Test Pulses	0	Gain Mode	Super High
Burst Submit Period	0.000 s	Collection Polarity	Hole
Energy threshold	4.5000	LFSR Bypass	Off
Cal File Root	/dls_sw/l13-1/epics/excalibur/V2.7		

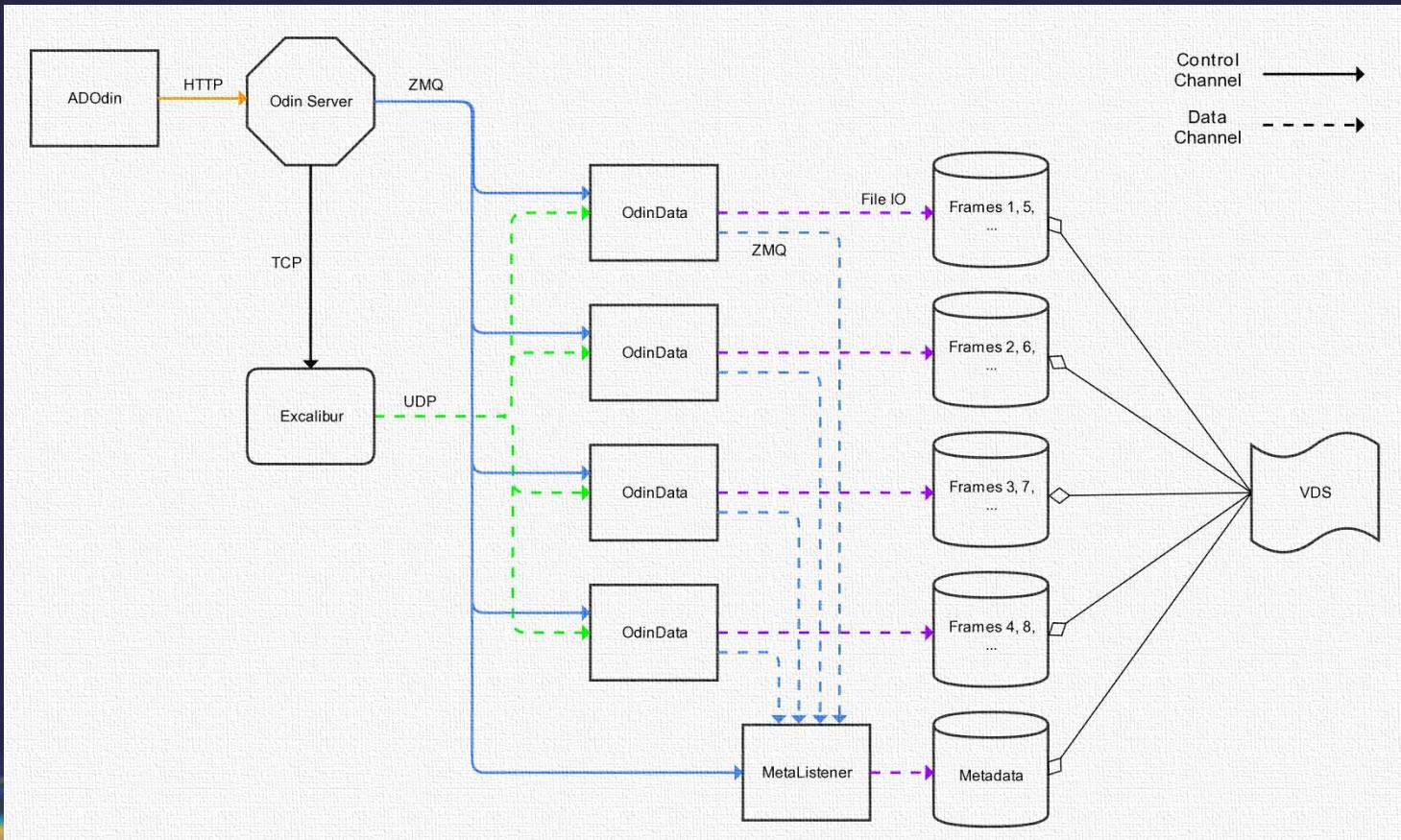
New Excalibur API

Info

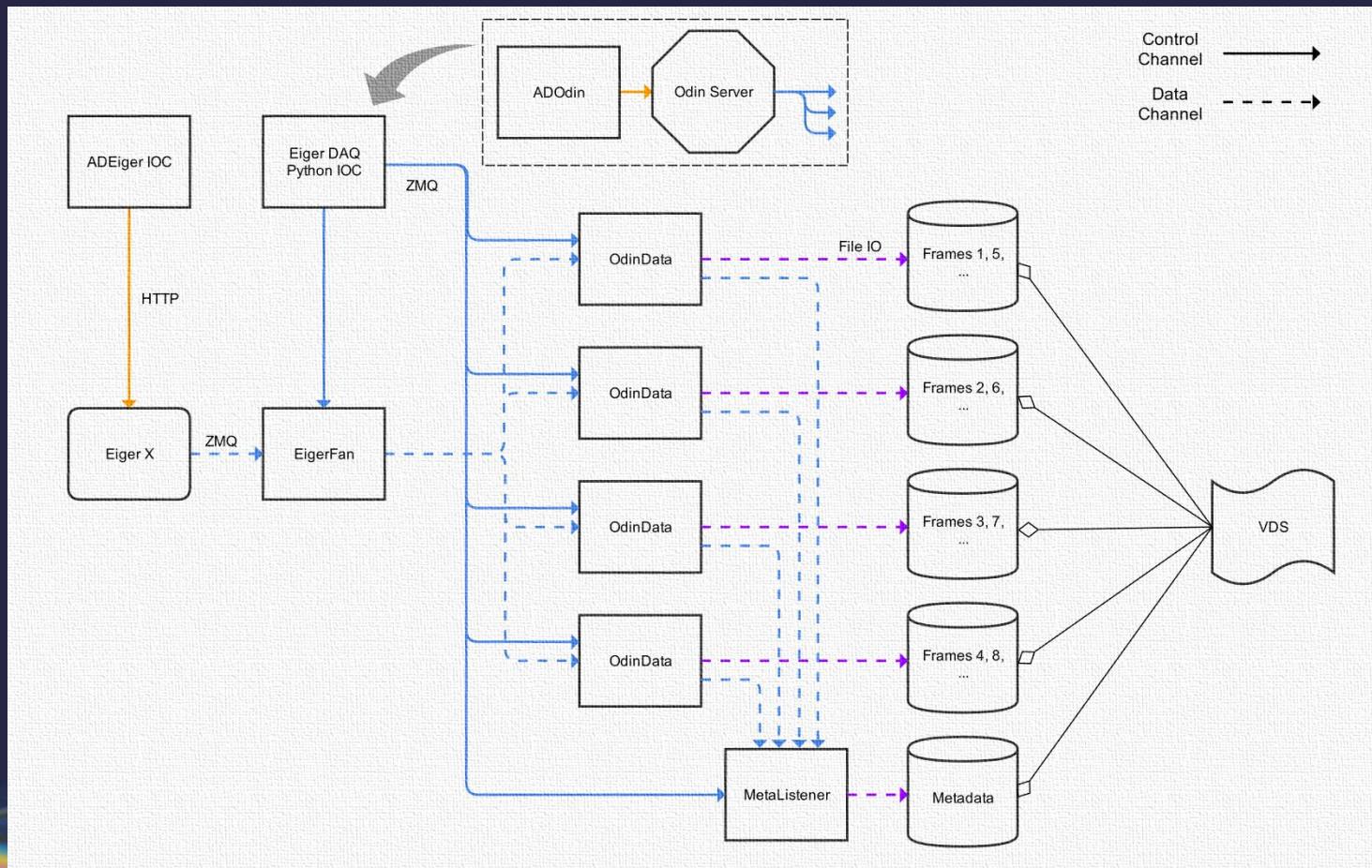
Port	ODIN-CAM	State	Acquire
Acquiring ...			
Model		BL-ID-EA-EXCALIBUR-DI	
Image			
Sensor Size		X 2048	Y 517
Image Size		8	8
Image Bytes		0	
Test Pulse Enable		Off	<input type="checkbox"/>
FEM Status			
Calibrated		Frames	
1:		5	
2:		5	
DAC Scan			
DAC #	0		
Start	0		
Stop	0		
Step	0		
Status			
Counter	0	9	
Array Rate (fps)	0.00 Hz		
Time	Exp.	0.000	Image
Remain	0	1	
Power Supplies			
Low Voltage			
High Voltage			
Configuration			
Operation Mode	Normal	Read/Write Mode	Sequential
Counter Depth	12 bit	24 Bit Discriminator	DiscL
Counter Select	A	Equalization Mode	Off
Auto Load	Manual	Pixel Mode	Single Pixel
Number of Test Pulses	0	Gain Mode	Super High
Burst Submit Period	0.000 s	Collection Polarity	Hole
Energy threshold	0.0000	LFSR Bypass	Off
Cal File Root			



Excalibur Overview



Eiger X 4M Overview



In Summary...

- OdinControl: a control system agnostic single point of control for arbitrary devices
- OdinData: a fast, scalable data acquisition framework
- Odin integrates with and replaces some parts of AreaDetector
- Will become an integral part of the control system at DLS



Thanks for Listening!

Questions?

Acknowledgements

Ulrik Pedersen, Alan Greer, Matthew Taylor | DLS
Tim Nicholls | STFC

More Info

Contact: gary.yendell@diamond.ac.uk
Paper: ICALEPCS 2017 – TUPHA212
GitHub: [odin-detector](#) and [dls-controls](#)



EPICS REST Interface

