Alpide Dataflow SystemC Model

Generated by Doxygen 1.8.11

Contents

1	Intro	oduction	n	1
2	Buil	ding an	nd Running the code	3
3	Todo	o List		5
4	Mod	lule Ind	ex	7
	4.1	Module	es	7
5	Hier	archica	ıl Index	9
	5.1	Class	Hierarchy	9
6	File	Index		11
	6.1	File Lis	st	11
7	Mod	lule Doo	cumentation	13
	7.1	Alpide	SystemC Model	13
		7.1.1	Detailed Description	14
		7.1.2	Class Documentation	14
			7.1.2.1 class Alpide	14
			7.1.2.2 class TopReadoutUnit	21
	7.2	Alpide	Constants	26
		7.2.1	Detailed Description	26
	7.3	Alpide	Data Format	27
		7.3.1	Detailed Description	28
		7.3.2	Class Documentation	28

iv CONTENTS

		7.3.2.1	struct FrameStartFifoWord	28
		7.3.2.2	struct FrameEndFifoWord	28
		7.3.2.3	class AlpideDataWord	29
		7.3.2.4	class Alpideldle	32
		7.3.2.5	class AlpideChipHeader	33
		7.3.2.6	class AlpideChipTrailer	34
		7.3.2.7	class AlpideChipEmptyFrame	35
		7.3.2.8	class AlpideRegionHeader	36
		7.3.2.9	class AlpideRegionTrailer	37
		7.3.2.10	class AlpideDataShort	38
		7.3.2.11	class AlpideDataLong	39
		7.3.2.12	class AlpideBusyOn	40
		7.3.2.13	class AlpideBusyOff	41
		7.3.2.14	class AlpideComma	42
	7.3.3	Variable	Documentation	42
		7.3.3.1	DW_COMMA	42
		7.3.3.2	DW_IDLE	42
7.4	Alpide	Data Pars	er	44
	7.4.1	Detailed	Description	44
	7.4.2	Class Do	ocumentation	44
		7.4.2.1	struct AlpideDataParsed	44
		7.4.2.2	class AlpideEventFrame	45
		7.4.2.3	class AlpideEventBuilder	46
		7.4.2.4	class AlpideDataParser	49
	7.4.3	Enumera	tion Type Documentation	51
		7.4.3.1	AlpideDataTypes	51
7.5	Pixel N	Matrix, Colu	umns, and Priority Encoder	52
	7.5.1	Detailed	Description	52
	7.5.2	Class Do	ocumentation	52
		7.5.2.1	class PixelData	52

CONTENTS

		7.5.2.2	class PixelPriorityEncoder	55
		7.5.2.3	class PixelDoubleColumn	56
		7.5.2.4	class PixelMatrix	58
7.6	Region	Readout		65
	7.6.1	Detailed	Description	65
	7.6.2	Class Do	cumentation	65
		7.6.2.1	class RegionReadoutUnit	65
7.7	Event (Generation	1	71
	7.7.1	Detailed	Description	71
	7.7.2	Class Do	cumentation	71
		7.7.2.1	class EventGenerator	71
		7.7.2.2	class Hit	81
		7.7.2.3	class TriggerEvent	83
7.8	Miscell	aneous fu	nctions	87
	7.8.1	Detailed	Description	87
	7.8.2	Function	Documentation	87
		7.8.2.1	addTrace(sc_trace_file *wf, std::string name_prefix, std::string signal_name, T &signal)	87
7.9	Setting	s		88
	7.9.1	Detailed	Description	88
	7.9.2	Function	Documentation	88
		7.9.2.1	getSimSettings(const char *fileName=""settings.txt"")	88
		7.9.2.2	setDefaultSimSettings(QSettings *readoutSimSettings)	89
7.10	Main A	lpide Simu	ulation Testbench	91
	7.10.1	Detailed	Description	91
	7.10.2	Class Do	ocumentation	91
		7.10.2.1	class Stimuli	91

vi

8	File I	Documentation	95
	8.1	src/alpide/alpide.h File Reference	95
		8.1.1 Detailed Description	96
	8.2	src/alpide/alpide_constants.h File Reference	97
		8.2.1 Detailed Description	97
	8.3	src/alpide/alpide_data_format.h File Reference	98
		8.3.1 Detailed Description	99
	8.4	src/alpide/alpide_data_parser.h File Reference	100
		8.4.1 Detailed Description	100
	8.5	src/alpide/pixel_col.h File Reference	101
		8.5.1 Detailed Description	102
	8.6	src/alpide/pixel_matrix.cpp File Reference	103
		8.6.1 Detailed Description	103
	8.7	src/alpide/pixel_matrix.h File Reference	103
		8.7.1 Detailed Description	104
	8.8	src/alpide/region_readout.cpp File Reference	105
		8.8.1 Detailed Description	105
	8.9	src/alpide/region_readout.h File Reference	105
		8.9.1 Detailed Description	107
	8.10	src/alpide/top_readout.cpp File Reference	107
		8.10.1 Detailed Description	107
	8.11	src/alpide/top_readout.h File Reference	108
		8.11.1 Detailed Description	109
	8.12	src/event/event_generator.cpp File Reference	109
		8.12.1 Detailed Description	109
		8.12.2 Macro Definition Documentation	110
		8.12.2.1 print_function_timestamp	110
	8.13	src/event/event_generator.h File Reference	110
		8.13.1 Detailed Description	111
	8.14	src/event/hit.h File Reference	111

CONTENTS vii

	8.14.1	Detailed Description	112
8.15	src/eve	ent/trigger_event.cpp File Reference	113
	8.15.1	Detailed Description	113
8.16	src/eve	ent/trigger_event.h File Reference	114
	8.16.1	Detailed Description	115
8.17	src/mis	cc/vcd_trace.h File Reference	115
	8.17.1	Detailed Description	116
8.18	src/sett	tings/settings.cpp File Reference	116
	8.18.1	Detailed Description	117
8.19	src/sett	tings/settings.h File Reference	117
	8.19.1	Detailed Description	119
8.20	src/test	tbench/main.cpp File Reference	119
	8.20.1	Detailed Description	120
	8.20.2	Function Documentation	120
		8.20.2.1 create_output_dir(const QSettings *settings)	120
		8.20.2.2 sc_main(int argc, char **argv)	121
8.21	src/test	tbench/stimuli.cpp File Reference	121
	8.21.1	Detailed Description	122
	8.21.2	Function Documentation	122
		8.21.2.1 print_event_rate(const std::list< int > &t_delta_queue)	122
8.22	src/test	tbench/stimuli.h File Reference	123
	8.22.1	Detailed Description	123
Index			125

Introduction

This is the Alpide Dataflow Model documentation, a simulation model of the Alpide chip written in C++ and using the SystemC framework.

The simulation model aims to be a fairly accurate model of the Alpide chip's readout chain, but with the performance benefits that C++ and SystemC offer compared to traditional HDL simulations.

Building the code and getting started

Instructions from the README on how to build, use and run the code can be found here:

· Building and Running the code

Codying style

Below is a brief description of the coding style used for this code.

Class member names

- SystemC port and signal names: all lowercase, with s_ prefix, and _in or _out postfix to indicate input/output. They should also be at the top of the class definition, separated from normal C++ member variables.
- · Class names: Camel case names, starting with capital letter
- · Class methods: Camel case names, starting with lower case letter (except for constructor etc. obviously)
- Class member variables: camel case names, starting with an m to indicate member variable, e.g. mNum

 Events.
- Class function parameters: all lowercase, words separated with underscore, to separate from class member variables. Example: num_events.

Documentation

Doxygen style comments

2 Introduction

Building and Running the code

A simple Dataflow SystemC Model of ITS and the Alpide chip

Getting started - building the project

Building the project requires GCC with C++11 support (included from version 4.8.x I think).

Required Libraries

- Requires qmake and boost libraries. On ubuntu, they can be installed with:
- qmake:

```
1 sudo apt-get install qt5-default
```

boost (full installation):

```
1 sudo apt-get install libboost-all-dev
```

· SystemC:

Can be downloaded from http://accellera.org/downloads/standards/systemc

The code has been tested and developed with version 2.3.1 of SystemC. Refer to SystemC documentation for build/install instructions.

Environment

The project's makefile expects to find the path to the base directory of the SystemC installation in \$SYSTEM ← C_HOME. It may also be necessary to add the path to the SystemC libraries to \$LD_LIBRARY_PATH. Adding something like this to \$HOME/.profile should do (for version 2.3.1):

```
1 export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/path/to/systemc-2.3.1/lib-linux64/
2 export SYSTEMC_HOME=/path/to/systemc-2.3.1
```

Compiling the code

```
1 cd alpide_dataflow_model 2 make
```

To run:

```
{\tt 1./alpide\_dataflow\_model}
```

The program requires a settings.txt file with simulation settings. If it does not exist, running the program generates a settings.txt file with default settings. This file can be edited and the simulation rerun to use those settings. The settings.txt file will not be overwritten.

Simulation results will be saved in sim_output/Run {timestamp}/

Simulation output is stored in alpide_dataflow_model/sim_output/{timestamp}/

To process simulation data:

```
1 cd alpide_dataflow_model/sim_output/{timestamp}/
2 root -b -q -l '.../process/process_event_data.C+("physics_events_data.csv")'
```

Todo List

Member Alpide::dataTransmission (void)

Implement more advanced data transmission method.

Member Alpide::frameReadout (void)

Strobe extended not implemented yet

Should the busy_transition flag always be set like this when chip is busy, or should it only happen when the chip goes into or out of busy state?

Member Alpide::mainProcess (void)

Implement more advanced data transmission method.

Member Alpide::s event buffers used debug

Should these signals be private maybe?

Member Alpide::strobelnput (void)

What should I do in data overrun mode (when readout_abort is set)? Should I still accept events? I need the frame end word to be added, for the normal transmission of CHIP HEADER/TRAILER words. This is currently done by the frameReadout() method, which requires there to be events in the MEB.

Should rejected triggers count be increased in data overrun mode?

The FATAL overflow bit/signal has to be cleared by a RORST/GRST command in the Alpide chip, it will not be cleared by automatically.

Need to clear RRU FIFOs, and MEBs when entering this state

File alpide data parser.h

Move this class to a separate directory/module. Don't mix it with the Alpide simulation model.

Member AlpideDataWord::operator<< (std::ostream &stream, const AlpideDataWord &alpide_dw)

Overload this for all AlpideDataWord classes, so SystemC can print them to trace files properly?

Member AlpideEventBuilder::inputDataWord (AlpideDataWord dw)

Busy on here

Busy off here

Unknown Alpide data word received. Do something smart here?

Member EventGenerator::addHitsToTriggerEvent (TriggerEvent &e)

Is this check worth it performance wise, or is it better to just iterate through the whole list?

Member EventGenerator::generateNextPhysicsEvent (void)

Account larger/bigger clusters here (when implemented)

Remove?

6 Todo List

Member EventGenerator::generateNextTriggerEvent (int64 t event start, int64 t event end, int chip id)

Should I check distance between start time of two triggers? Or the distance in time between the end of the first trigger and the start of the next trigger?

Member EventGenerator::mDataPath

This is currently used.. remove or update code that uses it..

Member EventGenerator::physicsEventProcess (void)

Maybe do this only on strobe falling edge? Saves some CPU cycles that way?

Member EventGenerator::scaleDiscreteDistribution (std::vector< double > &dist_vector, double new_← mean_value)

This changes the mean value slightly.. and the sum isn't that far off 1.0 before this anyway...

Member EventGenerator::setRandomSeed (int seed)

More than one seed? What if seed is set after random number generators have been started?

Member FrameEndFifoWord::operator<< (std::ostream &stream, const FrameEndFifoWord &dw)

Overload this for all FrameEndFifoWord classes, so SystemC can print them to trace files properly?

Member FrameStartFifoWord::operator<< (std::ostream &stream, const FrameStartFifoWord &dw)

Overload this for all FrameStartFifoWord classes, so SystemC can print them to trace files properly?

Member PixelMatrix::mColumnBuffs

Implement event ID somewhere. Maybe make an MEB class, and use it as the datatype for this queue?

Member print_event_rate (const std::list< int > &t_delta_queue)

Update/fix/remove this function.. currently not used..

Member RegionReadoutUnit::addTraces (sc_trace_file *wf, std::string name_prefix) const

Probably need to a stream << operator to allow values from fifo to be printed to trace file

Probably need to a stream << operator to allow values from fifo to be printed to trace file

Member sc_main (int argc, char **argv)

Pass vcd trace object to constructor of Stimuli class and Alpide classes?

Add a warning here if user tries to simulate over 1000 events with this option enabled, because it will consume 100s of megabytes

Member Stimuli::mNumEvents

Make it a 64-bit int?

Member Stimuli::stimuliEventProcess (void)

Check if there are actually events? Throw an error if we get notification but there are not events?

Member TopReadoutUnit::topRegionReadoutProcess (void)

Update state machine pictures with Alpide documentation + simplified FSM diagram

Should I maybe use the empty signal for the current region here??

File trigger_event.h

Use SystemC time data type instead of int64 t?

Member TriggerEvent::mEventFilteredFlag

With the new way of doing things, I don't need to have an Event object to keep track of hits, they are stored in the EventGenerator object. So I can get rid off this?

Member TriggerEvent::writeToFile (const std::string path="")

Note in use.. Revisit this function, since I have changed this class a lot...

Implement layers etc.

Module Index

4.1 Modules

Here is a list of all modules:

Alpide SystemC Model	3
Alpide Constants	26
Alpide Data Format	27
Alpide Data Parser	14
Pixel Matrix, Columns, and Priority Encoder	52
Region Readout	35
Event Generation	71
Miscellaneous functions	37
Settings	38
Main Alpide Simulation Testbench)1

8 Module Index

Hierarchical Index

5.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

AlpideDataParsed		44
AlpideDataWord		27
AlpideBusyOff		27
AlpideBusyOn		27
AlpideChipEmptyFrame		27
AlpideChipHeader		27
AlpideChipTrailer		
AlpideComma		
AlpideDataLong		
AlpideDataShort		
AlpideIdle		
AlpideRegionHeader		
AlpideRegionTrailer		27
AlpideEventBuilder		44
AlpideDataParser		44
AlpideEventFrame		44
FrameEndFifoWord		27
FrameStartFifoWord		27
PixelData		52
Hit		71
PixelDoubleColumn		52
PixelMatrix		52
Alpide		13
PixelPriorityEncoder		52
sc_module		J <u>Z</u>
Alpide		13
AlpideDataParser		
EventGenerator		
RegionReadoutUnit		
Stimuli		
TopReadoutUnit		
TriggerEvent		

10 Hierarchical Index

File Index

6.1 File List

Here is a list of all documented files with brief descriptions:

src/alpide/alpide.h	
Source file for Alpide class	??
src/alpide/alpide_constants.h	
Various constants for alpide chip, such as pixel matrix width and heigh, fifo depths, etc	??
src/alpide/alpide_data_format.h	
Definitions for data format used in Alpide chip	??
src/alpide/alpide_data_parser.h	
Classes for parsing serial data from Alpide chip, and building/reconstructing events/frames from	
the data	??
src/alpide/pixel_col.h	
Source file for pixel column, double column, and priority encoder classes	??
src/alpide/pixel_matrix.cpp	
Source file for pixel matrix class	??
src/alpide/pixel_matrix.h	
Header file for pixel matrix class	??
src/alpide/region_readout.cpp	
Class for implementing the Region Readout Unit (RRU) in the Alpide chip	??
src/alpide/region_readout.h	
Class for implementing the Region Readout Unit (RRU) in the Alpide chip	??
src/alpide/top_readout.cpp	
Class for implementing the Top Readout Unit (TRU) in the Alpide chip	??
src/alpide/top_readout.h	
Class for implementing the Top Readout Unit (TRU) in the Alpide chip	??
src/event/event_generator.cpp	??
src/event/event_generator.h	
A simple event generator for Alpide SystemC simulation model	??
src/event/hit.h	
Source file for PixelData and Hit classes. These classes hold the coordinates for a discrete hit in	
the Alpide chip, along with information about when the hit is active (equivalent to when the analog	
pulse out of the amplifier and shaping stage in the analog front end goes above the threshold)	??
src/event/trigger_event.cpp	
Event class for Alpide SystemC simulation model. This class holds all the pixel hits for an event	
for the whole detector. The philosophy behind this class is that the shaping etc. is performed by	
this class and the EventGenerator class, and that the pixel hits here can be fed directly to the	
Alpide chip at the given time	??

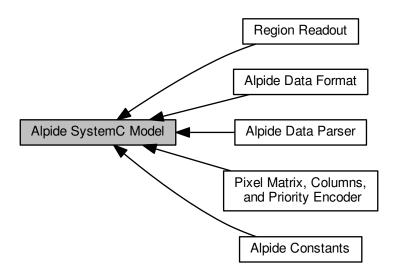
12 File Index

src/event/trigger_event.h	
Trigger event class for Alpide SystemC simulation model. This class holds all the pixel hits for a trigger event for the whole detector. The philosophy behind this class is that the shaping etc.	
is performed by this class and the EventGenerator class, and that the pixel hits here can be fed directly to the Alpide chip at the given time	??
src/misc/vcd_trace.h	
Common function for adding SystemC signals etc. to Value Change Dump (VCD) file	??
src/settings/settings.cpp	
Source file for simulation settings file	??
src/settings/settings.h	
Header file for simulation settings file handling	??
src/testbench/main.cpp	
Main source file for Alpide Dataflow SystemC simulation testbench	??
src/testbench/stimuli.cpp	
Source file for stimuli function for Alpide Dataflow SystemC model	??
src/testbench/stimuli.h	
Header file for stimuli function for Alpide Dataflow SystemC model	??

Module Documentation

7.1 Alpide SystemC Model

Collaboration diagram for Alpide SystemC Model:



Modules

- Alpide Constants
- Alpide Data Format
- Alpide Data Parser
- Pixel Matrix, Columns, and Priority Encoder
- Region Readout

Classes

- · class Alpide
- · class TopReadoutUnit

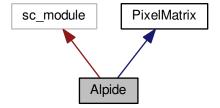
7.1.1 Detailed Description

7.1.2 Class Documentation

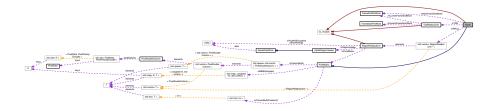
7.1.2.1 class Alpide

Alpide main class. Currently it only implements the MEBs, no RRU FIFOs, and no TRU FIFO. It will be used to run some initial estimations for probability of MEB overflow (busy).

Inheritance diagram for Alpide:



Collaboration diagram for Alpide:



Public Member Functions

• Alpide (sc_core::sc_module_name name, int chip_id, int region_fifo_size, int dmu_fifo_size, bool enable_← clustering, bool continuous_mode, bool matrix_readout_speed)

Constructor for Alpide.

- int getChipId (void)
- void addTraces (sc_trace_file *wf, std::string name_prefix) const Add SystemC signals to log in VCD trace file.
- uint64_t getTriggerEventsAcceptedCount (void) const
- uint64_t getTriggerEventsRejectedCount (void) const

Public Attributes

```
sc_in_clk s_system_clk_in
     40MHz LHC clock
sc in< bool > s strobe n in

 sc out< bool > s chip ready out

     Indicates that the chip is ready to accept hits and setPixel() can be called.

    sc out< sc uint< 24 >> s serial data output

    sc_signal< sc_uint< 8 >> s_event_buffers_used_debug

     Number of events stored in the chip at any given time.

    sc_signal< sc_uint< 8 >> s_frame_start_fifo_size_debug

    sc_signal< sc_uint< 8 >> s_frame_end_fifo_size_debug

• sc signal < sc uint < 32 > > s total number of hits
     Sum of all hits in all multi event buffers.

    sc_signal < sc_uint < 32 >> s_oldest_event_number_of_hits

     Number of hits in oldest multi event buffer.
• sc signal< bool > s region fifo empty [N REGIONS]

    sc signal < bool > s region valid [N REGIONS]

    sc_signal< bool > s_region_data_read [N_REGIONS]

    sc_signal< bool > s_region_event_start

    sc_signal < bool > s_region_event_pop

    sc signal < AlpideDataWord > s region data [N REGIONS]

    sc signal < bool > s frame readout start

     Frame Readout Managment Unit (FROMU) signals.

    sc_signal < bool > s_frame_readout_done [N_REGIONS]

    sc_signal< bool > s_frame_readout_done_all

    sc signal< bool > s frame fifo busy

    sc signal< bool > s multi event buffers busy

    sc signal < bool > s fatal state

    sc_signal< bool > s_readout_abort

    sc_signal< bool > s_flushed_incomplete

    sc signal< bool > s busy violation

• sc signal< bool > s busy status

    sc fifo< AlpideDataWord > s dmu fifo
```

Private Types

enum FROMU_readout_state_t { WAIT_FOR_EVENTS = 0, REGION_READOUT_START = 1, WAIT_F
 OR_REGION_READOUT = 2, REGION_READOUT_DONE = 3 }

Private Member Functions

void mainProcess (void)

sc_signal < sc_uint < 8 > > s_dmu_fifo_size
 sc_signal < bool > s_chip_ready_internal

Data transmission SystemC method. Currently runs on 40MHz clock.

void strobelnput (void)

This function handles the strobe input to the Alpide class object. Controls creation of new Multi Event Buffers (MEBs). Together with the frameReadout function, this process essentially does the same as the FROMU (Frame Read Out Management Unit) in the Alpide chip. Note: it is assumed that STROBE is synchronous to the clock. It will not be "dangerous" if it is not, but it will deviate from the real chip implementation.

void frameReadout (void)

Frame readout SystemC method @ 40MHz (system clock). Together with the strobeProcess, this function essentially does the same job as the FROMU (Frame Read Out Management Unit) in the Alpide chip.

void dataTransmission (void)

Read out DMU FIFOs and output data on "serial" line. Should be called one time per clock cycle.

bool getFrameReadoutDone (void)

Get logical AND/product of all regions' frame_readout_done signals.

· void updateBusyStatus (void)

Update internal busy status signals.

Private Attributes

- tlm::tlm fifo< FrameStartFifoWord > s frame start fifo
- tlm::tlm fifo< FrameEndFifoWord > s frame end fifo
- FrameEndFifoWord mNextFrameEndWord
- sc_signal < sc_uint < 8 > > s_fromu_readout_state
- · int mChipId
- bool mEnableReadoutTraces
- bool mStrobeActive
- uint16 t mBunchCounter
- uint64_t mTriggerEventsAccepted = 0

Number of (trigger) events that are accepted into an MEB by the chip.

• uint64 t mTriggerEventsRejected = 0

Triggered mode: If 3 MEBs are already full, the chip will not accept more events until one of those 3 MEBs have been read out. This variable is counted up for each event that is not accepted.

uint64 t mTriggerEventsFlushed = 0

Continuous mode only. The Alpide chip will try to guarantee that there is a free MEB slice in continuous mode. It does this by deleting the oldest MEB slice (even if it has not been read out) when the 3rd one is filled. This variable counts up in that case.

- std::vector< RegionReadoutUnit *> mRRUs
- TopReadoutUnit * mTRU

Additional Inherited Members

7.1.2.1.1 Constructor & Destructor Documentation

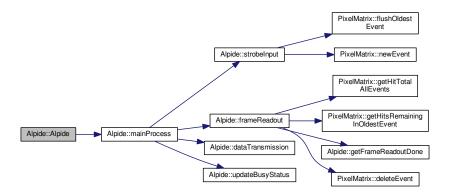
7.1.2.1.1.1 Alpide::Alpide (sc_core::sc_module_name name, int chip_id, int region_fifo_size, int dmu_fifo_size, bool enable_clustering, bool continuous_mode, bool matrix_readout_speed)

Constructor for Alpide.

Parameters

in	name	SystemC module name
in	chip_id	Desired chip id
in	region_fifo_size	Depth of Region Readout Unit (RRU) FIFOs
in	dmu_fifo_size	Depth of DMU (Data Management Unit) FIFO.
in	enable_clustering	Enable clustering and use of DATA LONG words
in	continuous_mode	Enable continuous mode (triggered mode if false)
in	matrix_readout_speed	True for fast readout (2 clock cycles), false is slow (4 cycles).

Here is the call graph for this function:



7.1.2.1.2 Member Function Documentation

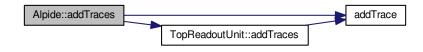
7.1.2.1.2.1 void Alpide::addTraces (sc_trace_file * wf, std::string name_prefix) const

Add SystemC signals to log in VCD trace file.

Parameters

in,out	wf	Pointer to VCD trace file object
in	name_prefix	Name prefix to be added to all the trace names

Here is the call graph for this function:



7.1.2.1.2.2 void Alpide::dataTransmission (void) [private]

Read out DMU FIFOs and output data on "serial" line. Should be called one time per clock cycle.

Todo Implement more advanced data transmission method.

Referenced by mainProcess().

Here is the caller graph for this function:



7.1.2.1.2.3 void Alpide::frameReadout (void) [private]

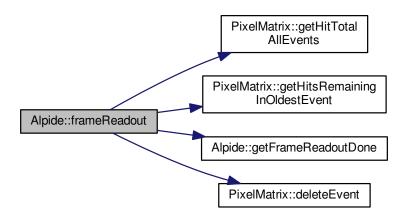
Frame readout SystemC method @ 40MHz (system clock). Together with the strobeProcess, this function essentially does the same job as the FROMU (Frame Read Out Management Unit) in the Alpide chip.

Todo Strobe extended not implemented yet

Todo Should the busy_transition flag always be set like this when chip is busy, or should it only happen when the chip goes into or out of busy state?

Referenced by mainProcess().

Here is the call graph for this function:



Here is the caller graph for this function:



7.1.2.1.2.4 bool Alpide::getFrameReadoutDone(void) [private]

Get logical AND/product of all regions' frame_readout_done signals.

Returns

True when frame_readout_done is set in all regions

Referenced by frameReadout().

Here is the caller graph for this function:



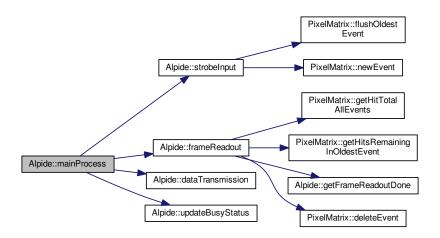
7.1.2.1.2.5 void Alpide::mainProcess (void) [private]

Data transmission SystemC method. Currently runs on 40MHz clock.

Todo Implement more advanced data transmission method.

Referenced by Alpide().

Here is the call graph for this function:



Here is the caller graph for this function:



7.1.2.1.2.6 void Alpide::strobelnput (void) [private]

This function handles the strobe input to the Alpide class object. Controls creation of new Multi Event Buffers (ME← Bs). Together with the frameReadout function, this process essentially does the same as the FROMU (Frame Read Out Management Unit) in the Alpide chip. Note: it is assumed that STROBE is synchronous to the clock. It will not be "dangerous" if it is not, but it will deviate from the real chip implementation.

Todo What should I do in data overrun mode (when readout_abort is set)? Should I still accept events? I need the frame end word to be added, for the normal transmission of CHIP HEADER/TRAILER words. This is currently done by the frameReadout() method, which requires there to be events in the MEB.

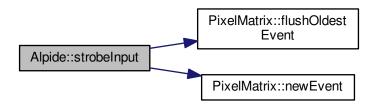
Todo Should rejected triggers count be increased in data overrun mode?

Todo The FATAL overflow bit/signal has to be cleared by a RORST/GRST command in the Alpide chip, it will not be cleared by automatically.

Todo Need to clear RRU FIFOs, and MEBs when entering this state

Referenced by mainProcess().

Here is the call graph for this function:



Here is the caller graph for this function:



7.1.2.1.3 Member Data Documentation

7.1.2.1.3.1 sc_signal < sc_uint < 8 > > Alpide::s_event_buffers_used_debug

Number of events stored in the chip at any given time.

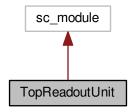
Todo Should these signals be private maybe?

Referenced by addTraces(), Alpide(), and frameReadout().

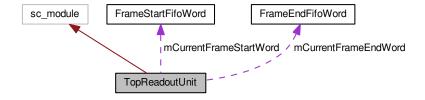
7.1.2.2 class TopReadoutUnit

The TopReadoutUnit (TRU) class is a simple representation of the TRU in the Alpide chip. It should be connected to the Region Readout Unit (RRU) in the Alpide object, and will be responsible for reading out from the RRUs with the topRegionReadoutProcess, which should run at the system clock (40MHz).

Inheritance diagram for TopReadoutUnit:



Collaboration diagram for TopReadoutUnit:



Public Member Functions

• TopReadoutUnit (sc_core::sc_module_name name, unsigned int chip_id)

Constructor for TopReadoutUnit.

void topRegionReadoutProcess (void)

SystemC method that controls readout from regions, should run on the 40MHz clock. The regions are read out in ascending order, and each event is encapsulated with a CHIP_HEADER and CHIP_TRAILER word. See the state machine diagram for a better explanation.

• void addTraces (sc_trace_file *wf, std::string name_prefix) const

Add SystemC signals to log in VCD trace file.

Public Attributes

```
    sc_in_clk s_clk_in
        Alpide chip clock (typically 40MHz)
    sc_in< bool > s_readout_abort_in
    sc_in< bool > s_fatal_state_in
    sc_in< bool > s_region_fifo_empty_in [N_REGIONS]
    sc_in< bool > s_region_valid_in [N_REGIONS]
    sc_in< AlpideDataWord > s_region_data_in [N_REGIONS]
    sc_out< bool > s_region_event_pop_out
    sc_out< bool > s_region_event_start_out
    sc_out< bool > s_region_data_read_out [N_REGIONS]
    sc_port< tlm::tlm_nonblocking_get_peek_if< FrameStartFifoWord > s_frame_start_fifo_output
    sc_port< tlm::tlm_nonblocking_get_peek_if< FrameEndFifoWord > s_frame_end_fifo_output
    sc_port< sc_fifo_out_if< AlpideDataWord > s_dmu_fifo_input
    Output from TRU.
```

Private Types

```
    enum TRU_state_t {
    EMPTY = 0, IDLE = 1, WAIT_REGION_DATA = 2, CHIP_HEADER = 3,
    BUSY_VIOLATION = 4, REGION_DATA = 5, WAIT = 6, CHIP_TRAILER = 7 }
```

Private Member Functions

• bool getNextRegion (unsigned int ®ion_out)

Find the first valid region, and return its region id.

bool getAllRegionsEmpty (void)

AND all region empty signals together.

Private Attributes

```
    sc_signal < sc_uint < 8 >> s_tru_state
    sc_signal < sc_uint < 8 >> s_previous_region
    sc_signal < bool > s_all_regions_empty_debug
    Signal copy of all_regions_empty variable, 1 cycle delayed.
    sc_signal < bool > s_no_regions_valid_debug
    Signal copy of no_regions_valid variable, 1 cycle delayed.
```

- · unsigned int mChipId
- FrameStartFifoWord mCurrentFrameStartWord
- FrameEndFifoWord mCurrentFrameEndWord

7.1.2.2.1 Constructor & Destructor Documentation

 $7.1.2.2.1.1 \quad \text{TopReadoutUnit::} \\ \text{TopReadoutUnit (sc_core::sc_module_name } \\ \textit{name, unsigned int } \\ \textit{chip_id)} \\$

Constructor for TopReadoutUnit.

Parameters

in	name	SystemC module name
in	chip←	Chip ID number
	_id	

Here is the call graph for this function:



7.1.2.2.2 Member Function Documentation

7.1.2.2.2.1 void TopReadoutUnit::addTraces ($sc_trace_file* \textit{wf}, std::string \textit{name_prefix}$) const

Add SystemC signals to log in VCD trace file.

Parameters

in,out	wf	Pointer to VCD trace file object
in	name_prefix	Name prefix to be added to all the trace names

Referenced by Alpide::addTraces().

Here is the call graph for this function:



Here is the caller graph for this function:



7.1.2.2.2.2 bool TopReadoutUnit::getAllRegionsEmpty (void) [private]

AND all region empty signals together.

Returns

true if all regions are empty

Referenced by topRegionReadoutProcess().

Here is the caller graph for this function:



7.1.2.2.2.3 bool TopReadoutUnit::getNextRegion (unsigned int & region_out) [private]

Find the first valid region, and return its region id.

Parameters

0	ut	region_out	Reference to an integer that will hold the region id.
---	----	------------	---

Returns

True if a valid region was found.

Referenced by topRegionReadoutProcess().

Here is the caller graph for this function:



7.1.2.2.2.4 void TopReadoutUnit::topRegionReadoutProcess (void)

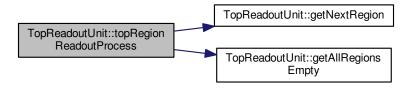
SystemC method that controls readout from regions, should run on the 40MHz clock. The regions are read out in ascending order, and each event is encapsulated with a CHIP_HEADER and CHIP_TRAILER word. See the state machine diagram for a better explanation.

Todo Update state machine pictures with Alpide documentation + simplified FSM diagram

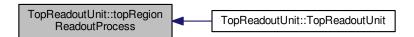
Todo Should I maybe use the empty signal for the current region here??

Referenced by TopReadoutUnit().

Here is the call graph for this function:

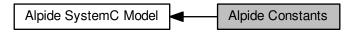


Here is the caller graph for this function:



7.2 Alpide Constants

Collaboration diagram for Alpide Constants:



Macros

- #define N_REGIONS 32
- #define N_PIXEL_ROWS 512
- #define N_PIXEL_COLS 1024
- #define **N_PIXEL_COLS_PER_REGION** (N_PIXEL_COLS/N_REGIONS)
- #define N_PIXEL_DOUBLE_COLS_PER_REGION (N_PIXEL_COLS_PER_REGION/2)
- #define N_PIXELS_PER_REGION (N_PIXEL_COLS/N_REGIONS)
- #define TRU_FRAME_FIFO_ALMOST_FULL1 48
- #define TRU FRAME FIFO ALMOST FULL2 56
- #define TRU_FRAME_FIFO_SIZE 64
- #define DATA_LONG_PIXMAP_SIZE ((unsigned int) 7)
- #define LHC ORBIT BUNCH COUNT 3564
- #define CHIP_WIDTH_CM 3
- #define CHIP_HEIGHT_CM 1.5

7.2.1 Detailed Description

7.3 Alpide Data Format 27

7.3 Alpide Data Format

Collaboration diagram for Alpide Data Format:



Classes

struct FrameStartFifoWord

Data word stored in FRAME START FIFO. More ...

struct FrameEndFifoWord

Data word stored in FRAME END FIFO. More...

· class AlpideDataWord

The FIFOs in the Alpide chip are 24 bits, or 3 bytes, wide. This is a base class for the data words that holds 3 bytes, and is used as the data type in the SystemC FIFO templates. This class shouldn't be used on its own, the various types of data words are implemented in derived classes. More...

- · class AlpideIdle
- class AlpideChipHeader
- · class AlpideChipTrailer
- · class AlpideChipEmptyFrame
- · class AlpideRegionHeader
- · class AlpideRegionTrailer
- · class AlpideDataShort
- · class AlpideDataLong
- · class AlpideBusyOn
- · class AlpideBusyOff
- class AlpideComma

Variables

- const uint8_t DW_IDLE = 0b111111111
- const uint8_t DW_CHIP_HEADER = 0b10100000
- const uint8_t DW_CHIP_TRAILER = 0b10110000
- const uint8_t **DW_CHIP_EMPTY_FRAME** = 0b11100000
- const uint8_t DW_REGION_HEADER = 0b11000000
- const uint8_t **DW_REGION_TRAILER** = 0b11110011
- const uint8_t DW_DATA_SHORT = 0b01000000
- const uint8_t DW_DATA_LONG = 0b000000000
- const uint8_t DW_BUSY_ON = 0b11110001
- const uint8_t **DW_BUSY_OFF** = 0b11110000
- const uint8 t DW COMMA = 0b111111110
- const uint8_t READOUT_FLAGS_BUSY_VIOLATION = 0b00001000
- const uint8_t **READOUT_FLAGS_FLUSHED_INCOMPLETE** = 0b00000100
- const uint8_t READOUT_FLAGS_STROBE_EXTENDED = 0b00000010

- const uint8_t READOUT_FLAGS_BUSY_TRANSITION = 0b00000001
- const uint8_t MASK_IDLE_BUSY_COMMA = 0b111111111

Mask for busy, idle and comma words.

• const uint8_t MASK_CHIP = 0b11110000

Mask for chip header/trailer/empty frame words.

• const uint8_t MASK_REGION_HEADER = 0b11100000

Mask for region header word.

const uint8 t MASK DATA = 0b11000000

Mask for data short/long words.

- 7.3.1 Detailed Description
- 7.3.2 Class Documentation
- 7.3.2.1 struct FrameStartFifoWord

Data word stored in FRAME START FIFO.

Public Member Functions

- bool operator== (const FrameStartFifoWord &rhs) const
- FrameStartFifoWord & operator= (const FrameStartFifoWord &rhs)

Public Attributes

- bool busy_violation
- uint16_t BC_for_frame

Friends

- void **sc_trace** (sc_trace_file *tf, const FrameStartFifoWord &dw, const std::string &name)
- std::ostream & operator<< (std::ostream &stream, const FrameStartFifoWord &dw)
- 7.3.2.1.1 Friends And Related Function Documentation
- 7.3.2.1.1.1 std::ostream& operator<<(std::ostream & stream, const FrameStartFifoWord & dw) [friend]
- Todo Overload this for all FrameStartFifoWord classes, so SystemC can print them to trace files properly?
- 7.3.2.2 struct FrameEndFifoWord

Data word stored in FRAME END FIFO.

7.3 Alpide Data Format 29

Public Member Functions

- bool operator== (const FrameEndFifoWord &rhs) const
- FrameEndFifoWord & operator= (const FrameEndFifoWord &rhs)

Public Attributes

- · bool flushed_incomplete
- bool strobe_extended
- bool busy_transition

Friends

- void sc_trace (sc_trace_file *tf, const FrameEndFifoWord &dw, const std::string &name)
- std::ostream & operator<< (std::ostream &stream, const FrameEndFifoWord &dw)

7.3.2.2.1 Friends And Related Function Documentation

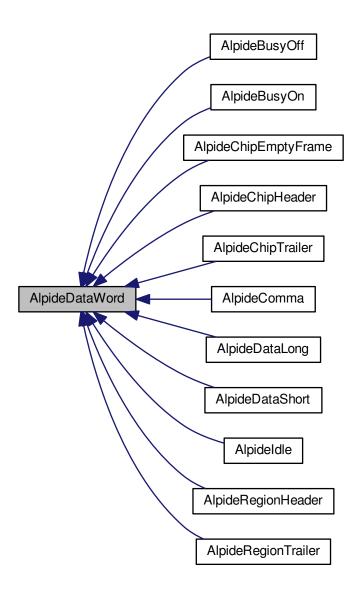
7.3.2.2.1.1 std::ostream& operator<<< (std::ostream & stream, const FrameEndFifoWord & dw) [friend]

Todo Overload this for all FrameEndFifoWord classes, so SystemC can print them to trace files properly?

7.3.2.3 class AlpideDataWord

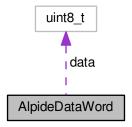
The FIFOs in the Alpide chip are 24 bits, or 3 bytes, wide. This is a base class for the data words that holds 3 bytes, and is used as the data type in the SystemC FIFO templates. This class shouldn't be used on its own, the various types of data words are implemented in derived classes.

Inheritance diagram for AlpideDataWord:



7.3 Alpide Data Format 31

Collaboration diagram for AlpideDataWord:



Public Member Functions

- bool signalBusyOn (void)
- bool signalBusyOff (void)
- bool operator== (const AlpideDataWord &rhs) const
- AlpideDataWord & operator= (const AlpideDataWord &rhs)

Public Attributes

• uint8_t data [3]

Friends

- void **sc_trace** (sc_trace_file *tf, const AlpideDataWord &dw, const std::string &name)
- std::ostream & operator<< (std::ostream &stream, const AlpideDataWord &alpide_dw)

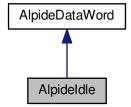
7.3.2.3.1 Friends And Related Function Documentation

7.3.2.3.1.1 std::ostream& operator<<(std::ostream & stream, const AlpideDataWord & alpide_dw) [friend]

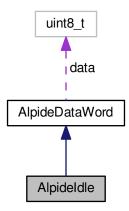
Todo Overload this for all AlpideDataWord classes, so SystemC can print them to trace files properly?

7.3.2.4 class Alpideldle

Inheritance diagram for Alpideldle:



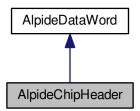
Collaboration diagram for Alpideldle:



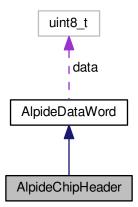
7.3 Alpide Data Format 33

7.3.2.5 class AlpideChipHeader

Inheritance diagram for AlpideChipHeader:



Collaboration diagram for AlpideChipHeader:

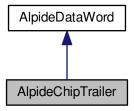


Public Member Functions

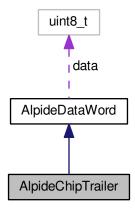
- AlpideChipHeader (uint8_t chip_id, uint16_t bunch_counter)
- AlpideChipHeader (uint8_t chip_id, FrameStartFifoWord &frame_start)

7.3.2.6 class AlpideChipTrailer

Inheritance diagram for AlpideChipTrailer:



Collaboration diagram for AlpideChipTrailer:



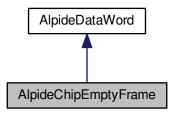
Public Member Functions

- AlpideChipTrailer (uint8_t readout_flags)
- AlpideChipTrailer (FrameStartFifoWord frame_start, FrameEndFifoWord frame_end, bool fatal_state, bool readout_abort)

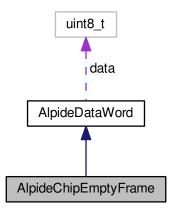
7.3 Alpide Data Format 35

7.3.2.7 class AlpideChipEmptyFrame

Inheritance diagram for AlpideChipEmptyFrame:



Collaboration diagram for AlpideChipEmptyFrame:

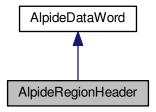


Public Member Functions

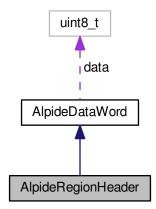
- AlpideChipEmptyFrame (uint8_t chip_id, uint16_t bunch_counter)
- AlpideChipEmptyFrame (uint8_t chip_id, FrameStartFifoWord &frame_start)

7.3.2.8 class AlpideRegionHeader

Inheritance diagram for AlpideRegionHeader:



Collaboration diagram for AlpideRegionHeader:



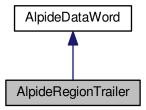
Public Member Functions

• AlpideRegionHeader (uint8_t region_id)

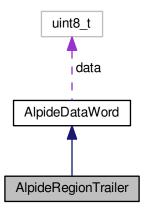
7.3 Alpide Data Format 37

7.3.2.9 class AlpideRegionTrailer

Inheritance diagram for AlpideRegionTrailer:

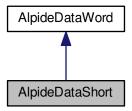


Collaboration diagram for AlpideRegionTrailer:

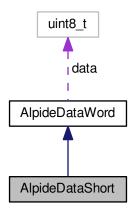


7.3.2.10 class AlpideDataShort

Inheritance diagram for AlpideDataShort:



Collaboration diagram for AlpideDataShort:



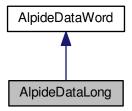
Public Member Functions

• AlpideDataShort (uint8_t encoder_id, uint16_t addr)

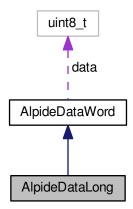
7.3 Alpide Data Format 39

7.3.2.11 class AlpideDataLong

Inheritance diagram for AlpideDataLong:



Collaboration diagram for AlpideDataLong:

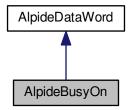


Public Member Functions

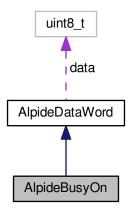
• AlpideDataLong (uint8_t encoder_id, uint16_t addr, uint8_t hitmap)

7.3.2.12 class AlpideBusyOn

Inheritance diagram for AlpideBusyOn:



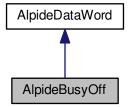
Collaboration diagram for AlpideBusyOn:



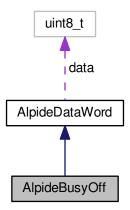
7.3 Alpide Data Format 41

7.3.2.13 class AlpideBusyOff

Inheritance diagram for AlpideBusyOff:

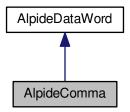


Collaboration diagram for AlpideBusyOff:

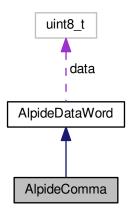


7.3.2.14 class AlpideComma

Inheritance diagram for AlpideComma:



Collaboration diagram for AlpideComma:



Additional Inherited Members

7.3.3 Variable Documentation

7.3.3.1 const uint8_t DW_COMMA = 0b111111110

This is not the correct COMMA word, but using this value instead makes this simulation model a bit simpler because the word cannot be confused with CHIP TRAILER 1111 1110 used to avoid confusion with CHIP TRAILER).

Referenced by AlpideEventBuilder::parseDataWord(), and AlpideEventBuilder::parseNonHeaderBytes().

7.3.3.2 const uint8_t DW_IDLE = 0b11111111

Alpide Data format and valid data words (from Alpide manual)

Data word	Header bits	Parameter bits
IDLE	1111 1111	None
CHIP HEADER	1010	<pre><chip id[3:0]=""><bunch counter="" for="" frame[10:3]=""></bunch></chip></pre>
CHIP TRAILER	1011	<readout flags[3:0]=""></readout>
CHIP EMPTY FRAME	1110	<pre><chip id[3:0]=""><bunch counter="" for="" frame[10:3]=""></bunch></chip></pre>
REGION HEADER	110	<region id[4:0]=""></region>
DATA SHORT	01	<pre><encoder id[3:0]=""><addr[9:0]></addr[9:0]></encoder></pre>
DATA LONG	00	<pre><encoder id[3:0]=""><addr[9:0]> 0 <hit map[6:0]=""> 1111 0001 1111</hit></addr[9:0]></encoder></pre>
BUSY ON	1111 0001	None
BUSY OFF	1111 0000	None
COMMA	1011 1100	Note: 1111 1110 used instead in this code

Alpide data words, used to initialize the 24-bit FIFOs in the Alpide chip. The MSBs in the words identify datawords, the LSBs are parameters. Note 1: There is not a fixed width for the MSB identifier part. Note 2: Not to be confused with the definitions in AlpideDataTypes in alpide_data_parser.h, which is for identifying individual bytes in a datastream.

Referenced by AlpideEventBuilder::parseDataWord(), and AlpideEventBuilder::parseNonHeaderBytes().

7.4 Alpide Data Parser

Collaboration diagram for Alpide Data Parser:



Classes

- struct AlpideDataParsed
- class AlpideEventFrame
- · class AlpideEventBuilder
- class AlpideDataParser

Enumerations

- enum AlpideDataTypes {
 - ALPIDE_IDLE, ALPIDE_CHIP_HEADER1, ALPIDE_CHIP_HEADER2, ALPIDE_CHIP_TRAILER, ALPIDE_CHIP_EMPTY_FRAME1, ALPIDE_CHIP_EMPTY_FRAME2, ALPIDE_REGION_HEADER, ALPIDE_DATA_SHORT1,
 - $\label{long1} \begin{tabular}{ll} ALPIDE_DATA_SHORT2, ALPIDE_DATA_LONG1, ALPIDE_DATA_LONG2, ALPIDE_DATA_LONG3, ALPIDE_BUSY_OFF, ALPIDE_COMMA, ALPIDE_UNKNOWN \end{tabular}$
- 7.4.1 Detailed Description
- 7.4.2 Class Documentation
- 7.4.2.1 struct AlpideDataParsed

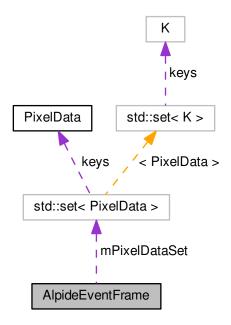
Public Attributes

• AlpideDataTypes data [3]

7.4 Alpide Data Parser 45

7.4.2.2 class AlpideEventFrame

Collaboration diagram for AlpideEventFrame:



Public Member Functions

- bool pixelHitInEvent (PixelData &pixel) const Look for a pixel hit in this event frame.
- void setFrameCompleted (bool val)
- bool getFrameCompleted (void)
- unsigned int getEventSize (void) const
- void addPixelHit (const PixelData &pixel)
- std::set< PixelData >::const_iterator getPixelSetIterator (void) const
- std::set< PixelData >::const_iterator getPixelSetEnd (void) const

Private Attributes

- std::set< PixelData > mPixelDataSet
- bool mFrameCompleted

7.4.2.2.1 Member Function Documentation

7.4.2.2.1.1 bool AlpideEventFrame::pixelHitlnEvent (PixelData & pixel) const

Look for a pixel hit in this event frame.

Parameters

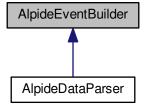
in pixel Reference to PixelData objection	ct
---	----

Returns

True if pixel is in event frame, false if not.

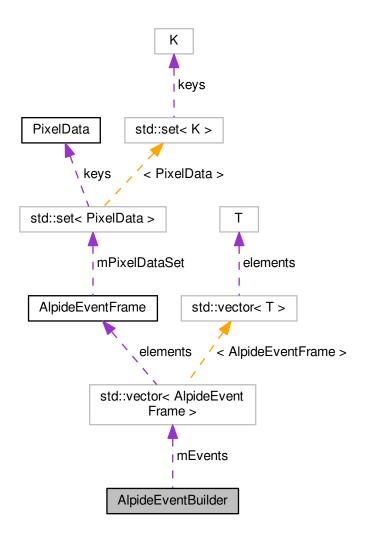
7.4.2.3 class AlpideEventBuilder

Inheritance diagram for AlpideEventBuilder:



7.4 Alpide Data Parser 47

Collaboration diagram for AlpideEventBuilder:



Public Member Functions

- unsigned int getNumEvents (void) const
- const AlpideEventFrame * getNextEvent (void) const

Get a reference to the next event. This does not delete the event, and successive calls will return the same event unless popEvent() has been called.

void popEvent (void)

Pop/remove the oldest event (if there are any events, otherwise do nothing).

void inputDataWord (AlpideDataWord dw)

Takes a 3 byte Alpide data word as input, parses it, and depending on the data: 1) If this is a new Alpide data frame, a new AlpideEventFrame is created in mEvents 2) If this is data that belongs to the existing and most recent frame, hit data is added to that frame. 3) If these are just idle words etc., nothing is done with them.

AlpideDataParsed parseDataWord (AlpideDataWord dw)

Parse 3-byte Alpide data words the, and increase counters for the different types of data words. Note: the function only discovers what type of data word it is, it does nothing with the data word's parameters.

Private Member Functions

AlpideDataTypes parseNonHeaderBytes (uint8_t data)

Use this to parse the last 1-2 (least significant) bytes of a 24-bit Alpide data word, for words which are known to not utilize these bytes, e.g.: Data long uses all 3 bytes - don't use this function Data short uses first 2 bytes - use this function for the last byte Region header uses the first byte - use this function for the last two bytes The function will return either IDLE, BUSY_ON, BUSY_OFF, or UNKNOWN for these bytes. It will also increase counters for the corresponding words.

Private Attributes

- std::vector< AlpideEventFrame > mEvents
- unsigned int mCurrentRegion = 0
- long mCommaCount
- long mldleCount
- · long mldleByteCount
- · long mBusyOnCount
- · long mBusyOffCount
- long mDataShortCount
- long mDataLongCount
- long mRegionHeaderCount
- long mChipHeaderCount
- long mChipTrailerCount
- long mChipEmptyFrameCount
- long mUnknownDataWordCount

7.4.2.3.1 Member Function Documentation

7.4.2.3.1.1 const AlpideEventFrame * AlpideEventBuilder::getNextEvent (void) const

Get a reference to the next event. This does not delete the event, and successive calls will return the same event unless popEvent() has been called.

Returns

Pointer to the next event if there are more events, nullptr if there are no events.

7.4.2.3.1.2 void AlpideEventBuilder::inputDataWord (AlpideDataWord dw)

Takes a 3 byte Alpide data word as input, parses it, and depending on the data: 1) If this is a new Alpide data frame, a new AlpideEventFrame is created in mEvents 2) If this is data that belongs to the existing and most recent frame, hit data is added to that frame. 3) If these are just idle words etc., nothing is done with them.

Parameters

	in	dw	AlpideDataWord input to parse.
--	----	----	--------------------------------

Todo Busy on here

Todo Busy off here

7.4 Alpide Data Parser 49

Todo Unknown Alpide data word received. Do something smart here?

7.4.2.3.1.3 AlpideDataParsed AlpideEventBuilder::parseDataWord (AlpideDataWord dw)

Parse 3-byte Alpide data words the, and increase counters for the different types of data words. Note: the function only discovers what type of data word it is, it does nothing with the data word's parameters.

Parameters

in	dw	AlpideDataWord input to parse.
----	----	--------------------------------

Returns

AlpideDataParsed object with parsed data word type filled in for each byte

7.4.2.3.1.4 AlpideDataTypes AlpideEventBuilder::parseNonHeaderBytes (uint8_t data) [private]

Use this to parse the last 1-2 (least significant) bytes of a 24-bit Alpide data word, for words which are known to not utilize these bytes, e.g.: Data long uses all 3 bytes - don't use this function Data short uses first 2 bytes - use this function for the last byte Region header uses the first byte - use this function for the last two bytes The function will return either IDLE, BUSY_ON, BUSY_OFF, or UNKNOWN for these bytes. It will also increase counters for the corresponding words.

Parameters

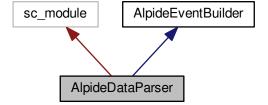
ir	data	One of the "additional" bytes in a data word to parse
----	------	---

Returns

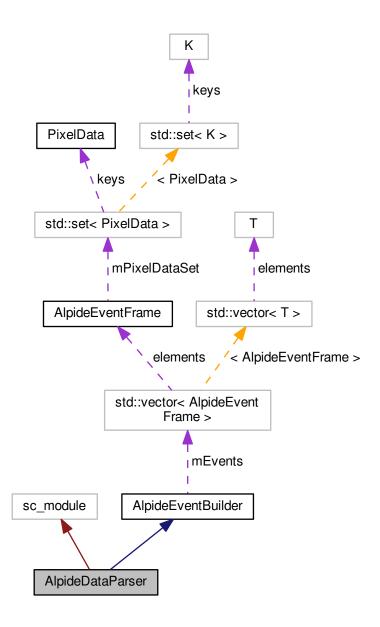
Data word type for "additional" byte provided in data argument

7.4.2.4 class AlpideDataParser

Inheritance diagram for AlpideDataParser:



Collaboration diagram for AlpideDataParser:



Public Member Functions

- AlpideDataParser (sc_core::sc_module_name name)
- void addTraces (sc_trace_file *wf, std::string name_prefix) const Add SystemC signals to log in VCD trace file.

Public Attributes

- sc_in< sc_uint< 24 >> s_serial_data_in
- sc_in_clk s_clk_in

7.4 Alpide Data Parser 51

Private Member Functions

· void parserInputProcess (void)

Matrix readout SystemC method. Expects a 3-byte word input on each clock edge. The 3-byte data word is passed to the underlying base class for processing and event frame generation.

7.4.2.4.1 Member Function Documentation

7.4.2.4.1.1 void AlpideDataParser::addTraces (sc_trace_file * wf, std::string name_prefix) const

Add SystemC signals to log in VCD trace file.

Parameters

in,out	wf	Pointer to VCD trace file object
in	name_prefix	Name prefix to be added to all the trace names

Here is the call graph for this function:



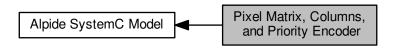
7.4.3 Enumeration Type Documentation

7.4.3.1 enum AlpideDataTypes

Enumerations used to identify the meaning of the different bytes in the data stream from the Alpide chip. Not to be confused with the definitions in alpide_data_format.h which are used to initialize the 24-bit FIFO words.

7.5 Pixel Matrix, Columns, and Priority Encoder

Collaboration diagram for Pixel Matrix, Columns, and Priority Encoder:



Classes

class PixelData

A struct that indicates a hit in a region, at the pixel identified by the col and row variables. For each hit an object of this type will be inserted into the std::set container in regionDataVector. For the pixels that don't have hits there will not be an object of this type inserted. Column should be 0 or 1. Row can be any value from 0 to N_PIXEL_ROWS-1. More...

· class PixelPriorityEncoder

Comparator class/function for use with the PixelData class in the std::set container, which allows the container to sort the PixelData entries in a meaningful way. The picture below is from the ALPIDE operations manual, and shows: More...

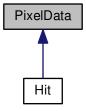
- class PixelDoubleColumn
- class PixelMatrix

Functions

- const PixelData NoPixelHit (-1,-1)
- 7.5.1 Detailed Description
- 7.5.2 Class Documentation
- 7.5.2.1 class PixelData

A struct that indicates a hit in a region, at the pixel identified by the col and row variables. For each hit an object of this type will be inserted into the std::set container in regionDataVector. For the pixels that don't have hits there will not be an object of this type inserted. Column should be 0 or 1. Row can be any value from 0 to N_PIXEL_ROWS-1.

Inheritance diagram for PixelData:



Public Member Functions

- PixelData (int col=0, int row=0)
- PixelData (int region, int pri enc, int addr)

Constructor for pixel data based on region number, priority encoder number in region, and pixel address in priority encoder. This is how the data are specified when they are transmitted as data long/short words.

- PixelData (const PixelData &p)
- bool operator== (const PixelData &rhs) const
- bool operator> (const PixelData &rhs) const
- bool operator< (const PixelData &rhs) const
- bool operator>= (const PixelData &rhs) const
- bool operator<= (const PixelData &rhs) const
- int getCol (void) const
- int getRow (void) const
- void setCol (const int col)
- void setRow (const int row)
- unsigned int getPriEncPixelAddress (void) const

Get the "address" of this pixel within it's double column, that is the priority that this pixel has in the priority encoder would

• unsigned int getPriEncNumInRegion (void) const

Get the priority encoder that this pixel (column) belongs to, within the column's region. Hardcoded for 16 double columns per region.

Private Attributes

- int mCol
- int mRow

Friends

- class PixelPriorityEncoder
- 7.5.2.1.1 Constructor & Destructor Documentation
- 7.5.2.1.1.1 PixelData::PixelData (int region, int pri_enc, int addr)

Constructor for pixel data based on region number, priority encoder number in region, and pixel address in priority encoder. This is how the data are specified when they are transmitted as data long/short words.

Parameters

in	region	Region number
in	pri_enc	Priority encoder number in region (ie. double column number in region).
in	addr	Prioritized address in priority encoder

7.5.2.1.2 Member Function Documentation

7.5.2.1.2.1 unsigned int PixelData::getPriEncNumInRegion (void) const [inline]

Get the priority encoder that this pixel (column) belongs to, within the column's region. Hardcoded for 16 double columns per region.

Returns

Priority encoder number.

Referenced by RegionReadoutUnit::readoutNextPixel().

Here is the caller graph for this function:



7.5.2.1.2.2 unsigned int PixelData::getPriEncPixelAddress (void) const [inline]

Get the "address" of this pixel within it's double column, that is the priority that this pixel has in the priority encoder would.

Returns

Pixel's priority encoder address/priority

Referenced by RegionReadoutUnit::readoutNextPixel().

Here is the caller graph for this function:



7.5.2.2 class PixelPriorityEncoder

Comparator class/function for use with the PixelData class in the std::set container, which allows the container to sort the PixelData entries in a meaningful way. The picture below is from the ALPIDE operations manual, and shows:

- Left: 512 rows x 1024 columns of pixels, divided into 32 regions
- Middle: 32 columns (16 double columns) x 512 rows in a region
- Right: Index/numbering/address of pixels within a double column, and the priority encoder between the columns. The priority encoder starts with the pixel that has the lowest address, and prioritizes them in increasing order.

The regionDataVector, which is declared as std::vector<std::set<PixelData, PixelComparer> > regionData← Vector; attempts to implement the priority encoder to reflect what is on the ALPIDE chip. Only pixels that have hits will be stored in the set, the pixels in the set are read out in increasing order (starting with index 0), and the PixelComparer type implements the actual prioritization of the pixels in the set.

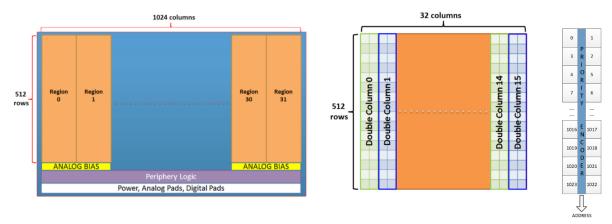


Figure 7.1 Overview of how regions and columns are indexed, and how pixels are indexed in double columns, in the Alpide chip.

Public Member Functions

• bool operator() (const PixelData &leftIn, const PixelData &rightIn)

Overloaded () function, allows the std::set to use this function to compare two PixelData classes in the set, and determine which of them should come first when sorting them. The prioritization works like this:

7.5.2.2.1 Member Function Documentation

7.5.2.2.1.1 bool PixelPriorityEncoder::operator() (const PixelData & leftln, const PixelData & rightln) [inline]

Overloaded () function, allows the std::set to use this function to compare two PixelData classes in the set, and determine which of them should come first when sorting them. The prioritization works like this:

- · Lower rows prioritized first
- For even rows, the column 0 pixel comes first
- · For odd rows, the column 1 pixel comes first

Parameters

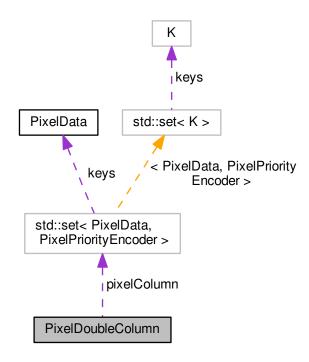
leftIn	Left side argument
right⊷ In	Right side argument

Returns

True if leftIn has highest priority, false if rightIn has higest priority

7.5.2.3 class PixelDoubleColumn

Collaboration diagram for PixelDoubleColumn:



Public Member Functions

- void setPixel (unsigned int col_num, unsigned int row_num)
 - Set a pixel in a pixel double column object.
- void clear (void)
 - Clear (flush) contents of double column.
- bool inspectPixel (unsigned int col_num, unsigned int row_num)
 - Check if there is a hit or not for the pixel specified by col_num and row_num, without deleting the pixel from the MEB.
- PixelData readPixel (void)
 - Read out the next pixel from this double column, and erase it from the MEB. Pixels are read out in an order corresponding to that of the priority encoder in the Alpide chip.
- unsigned int pixelHitsRemaining (void)
 - Returns how many pixel hits (in this double column) that have not been read out from the MEBs yet.

Private Attributes

std::set< PixelData, PixelPriorityEncoder > pixelColumn

7.5.2.3.1 Member Function Documentation

7.5.2.3.1.1 bool PixelDoubleColumn::inspectPixel (unsigned int col_num, unsigned int row_num)

Check if there is a hit or not for the pixel specified by col_num and row_num, without deleting the pixel from the MEB.

Parameters

in	col_num	column number of pixel, must be 0 or 1.
in	row_num	row number of pixel, must be in the range 0 to N_PIXEL_ROWS-1

Returns

True if there is a hit, false if not.

Exceptions

	std::out of range	if col_num or row_num is not in the specified range.	
- 1			

7.5.2.3.1.2 PixelData PixelDoubleColumn::readPixel (void)

Read out the next pixel from this double column, and erase it from the MEB. Pixels are read out in an order corresponding to that of the priority encoder in the Alpide chip.

Returns

PixelData with hit coordinates. If no pixel hits exist, NoPixelHit is returned (PixelData object with coords = (-1,-1)).

7.5.2.3.1.3 void PixelDoubleColumn::setPixel (unsigned int col_num, unsigned int row_num)

Set a pixel in a pixel double column object.

Parameters

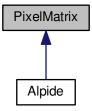
in	col_num	column number of pixel, must be 0 or 1.
in	row_num	row number of pixel, must be in the range 0 to N_PIXEL_ROWS-1

Exceptions

std::out_of_range	if col_num or row_num is not in the specified range.
-------------------	--

7.5.2.4 class PixelMatrix

Inheritance diagram for PixelMatrix:



Collaboration diagram for PixelMatrix:



Public Member Functions

• PixelMatrix (bool continuous_mode)

PixelMatrix Constructor.

void newEvent (uint64_t event_time)

Indicate to the Alpide that we are starting on a new event. If the call is successful a new MEB slice is created, and the next calls to setPixel will add pixels to the new event.

void deleteEvent (uint64_t event_time)

Delete the oldest event from the MEB (if there are any events at all, calling this function with no events is fine).

void flushOldestEvent (void)

Flush the oldest event by clearing all double columns and setting its size to zero.

void setPixel (unsigned int col, unsigned int row)

Set the pixel (ie. the pixel is hit) specified by col_num and row_num, in the most recent event buffer.

• bool regionEmpty (int start_double_col, int stop_double_col)

Check if the region denoted by start_double_col and stop_double_col is empty.

bool regionEmpty (int region)

Check if a region of the pixel matrix is empty.

• PixelData readPixel (uint64 t time now, int start double col=0, int stop double col=N PIXEL COLS/2)

Read out the next pixel from the pixel matrix, and erase it from the MEB. This member function will read out pixels from the oldest event buffer. The pixels will be by default be read out from the double columns in consecutive order from 0 to (N_PIXEL_COLS/2)-1, or optionally from the double column range specified by start_double_col and stop—double_col. Regions are not read out in parallel with this function. But note that within a double column the pixels will be read out with the order used by the priority encoder in the Alpide chip.

PixelData readPixelRegion (int region, uint64_t time_now)

Read out the next pixel from the specified region in the pixel matrix, and erase it from the MEB. This member function will read out pixels from the oldest event buffer. The pixels in the desired region will be read out from the double columns in consecutive order from 0 to N_PIXEL_DOUBLE_COLS_PER_REGION-1. Note that within a double column the pixels will be read out with the order used by the priority encoder in the Alpide chip.

- int **getNumEvents** (void)
- int getHitsRemainingInOldestEvent (void)

Return the number of hits in the oldest of the events stored in multi event buffers.

int getHitTotalAllEvents (void)

Get total number of hits in all Multi Event Buffers.

std::map< unsigned int, std::uint64 t > getMEBHisto (void) const

Protected Attributes

• bool mContinuousMode

True: Continuous, False: Triggered.

Private Attributes

• std::queue< std::vector< PixelDoubleColumn >> mColumnBuffs

mColumnBuffs holds multi event buffers of pixel columns The queue represent the MEBs, and the vector the pixel columns.

std::list< int > mColumnBuffsPixelsLeft

Each entry here corresponds to one entry in mColumnBuffs. This variable keeps track of the number of pixel left in the columns in each entry in mColumnBuffs.

std::map< unsigned int, std::uint64_t > mMEBHistogram

This map contains histogram values over MEB usage. The key is the number of MEBs in use, and the value is the total time duration for that key.

uint64_t mMEBHistoLastUpdateTime = 0

Last time the MEB histogram was updated.

7.5.2.4.1 Constructor & Destructor Documentation

7.5.2.4.1.1 PixelMatrix::PixelMatrix (bool continuous_mode)

PixelMatrix Constructor.

Parameters

in	continuous_mode	True: continuous mode, false: triggered mode
----	-----------------	--

7.5.2.4.2 Member Function Documentation

7.5.2.4.2.1 void PixelMatrix::deleteEvent (uint64_t time_now)

Delete the oldest event from the MEB (if there are any events at all, calling this function with no events is fine).

Parameters

in	time_now	Simulation time when this readout is occuring

Exceptions

out_of_range	If there are no events, or if col or row is outside the allowed range
--------------	---

Referenced by Alpide::frameReadout().

Here is the caller graph for this function:



7.5.2.4.2.2 int PixelMatrix::getHitsRemainingInOldestEvent (void)

Return the number of hits in the oldest of the events stored in multi event buffers.

Returns

Number of hits in oldest event. If there are no events left, return zero.

Referenced by Alpide::frameReadout().

Here is the caller graph for this function:



7.5.2.4.2.3 int PixelMatrix::getHitTotalAllEvents (void)

Get total number of hits in all Multi Event Buffers.

Returns

Total number of hits.

Referenced by Alpide::frameReadout().

Here is the caller graph for this function:



7.5.2.4.2.4 void PixelMatrix::newEvent (uint64_t event_time)

Indicate to the Alpide that we are starting on a new event. If the call is successful a new MEB slice is created, and the next calls to setPixel will add pixels to the new event.

Parameters

in	event time	Simulation time when the event is pushed/latched into MEB (use current simulation time).
----	------------	--

Referenced by Alpide::strobeInput().

Here is the caller graph for this function:



7.5.2.4.2.5 PixelData PixelMatrix::readPixel (uint64_t time_now, int start_double_col = 0, int stop_double_col = N_PIXEL_COLS/2)

Read out the next pixel from the pixel matrix, and erase it from the MEB. This member function will read out pixels from the oldest event buffer. The pixels will be by default be read out from the double columns in consecutive order from 0 to (N_PIXEL_COLS/2)-1, or optionally from the double column range specified by start_double_col and stop_double_col. Regions are not read out in parallel with this function. But note that within a double column the pixels will be read out with the order used by the priority encoder in the Alpide chip.

Parameters

	in	time_now	Simulation time when this readout is occuring
	in	start_double_col	Start double column to start searching for pixels to readout from
Ī	in	stop_double_col Stop searching for pixels to read out when reaching this colur	

Returns

PixelData with hit coordinates. If no pixel hits exist, NoPixelHit is returned (PixelData object with coords = (-1,-1)).

Exceptions

	std::out_of_range	if start_double_col is less than zero, or larger than (N_PIXEL_COLS/2)-1.
std::out_of_range if stop_double_col is less than one, or larger than N_PIXEL_Co		if stop_double_col is less than one, or larger than N_PIXEL_COLS/2.
	std::out_of_range	if stop_double_col is greater than or equal to start_double_col

Referenced by readPixelRegion().

Here is the caller graph for this function:



7.5.2.4.2.6 PixelData PixelMatrix::readPixelRegion (int region, uint64_t time_now)

Read out the next pixel from the specified region in the pixel matrix, and erase it from the MEB. This member function will read out pixels from the oldest event buffer. The pixels in the desired region will be read out from the double columns in consecutive order from 0 to N_PIXEL_DOUBLE_COLS_PER_REGION-1. Note that within a double column the pixels will be read out with the order used by the priority encoder in the Alpide chip.

Parameters

in	region	The region number to read out a pixel from	
in	time_now	Simulation time when this readout is occuring. Required for updating histogram data in case an MEB is done reading out.	

Returns

PixelData with hit coordinates. If no pixel hits exist, NoPixelHit is returned (PixelData object with coords = (-1,-1)).

Exceptions

	std::out_of_range	if region is less than zero, or greater than N_REGIONS-1	
--	-------------------	--	--

Referenced by RegionReadoutUnit::readoutNextPixel().

Here is the call graph for this function:



Here is the caller graph for this function:



7.5.2.4.2.7 bool PixelMatrix::regionEmpty (int start_double_col, int stop_double_col)

Check if the region denoted by start_double_col and stop_double_col is empty.

Parameters

in	start_double_col	Start of region in terms of double columns
	-411-	Final of an along in Assess of Assessing a discount
ın	stop_aouble_col	End of region in terms of double columns

Returns

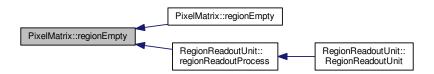
True if empty

Exceptions

std::out_of_range	if start_double_col is less than zero, or larger than (N_PIXEL_COLS/2)-1.
std::out_of_range	if stop_double_col is less than one, or larger than N_PIXEL_COLS/2.
std::out_of_range	if stop_double_col is greater than or equal to start_double_col

 $Referenced\ by\ region Empty(),\ and\ Region Readout Unit:: region Readout Process().$

Here is the caller graph for this function:



7.5.2.4.2.8 bool PixelMatrix::regionEmpty (int region)

Check if a region of the pixel matrix is empty.

Parameters

in	region	The region number to check

Returns

PixelData with hit coordinates. If no pixel hits exist, NoPixelHit is returned (PixelData object with coords = (-1,-1)).

Exceptions

std::out_of_range	if region is less than zero,	, or greater than N_REGIONS-1
-------------------	------------------------------	-------------------------------

Here is the call graph for this function:



7.5.2.4.2.9 void PixelMatrix::setPixel (unsigned int col, unsigned int row)

Set the pixel (ie. the pixel is hit) specified by col_num and row_num, in the most recent event buffer.

Parameters

in	col	Column (0 to N_PIXEL_COLS-1).
in	row	Row (0 to N_PIXEL_ROWS-1).

Exceptions

	out_of_range	If there are no events, or if col or row is outside the allowed range	
--	--------------	---	--

Referenced by TriggerEvent::feedHitsToChip().

Here is the caller graph for this function:



7.5.2.4.3 Member Data Documentation

7.5.2.4.3.1 std::queue< std::vector<PixelDoubleColumn> > PixelMatrix::mColumnBuffs [private]

mColumnBuffs holds multi event buffers of pixel columns The queue represent the MEBs, and the vector the pixel columns.

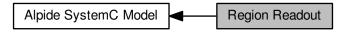
Todo Implement event ID somewhere. Maybe make an MEB class, and use it as the datatype for this queue?

Referenced by deleteEvent(), flushOldestEvent(), getHitsRemainingInOldestEvent(), getHitTotalAllEvents(), new Event(), readPixel(), regionEmpty(), and setPixel().

7.6 Region Readout 65

7.6 Region Readout

Collaboration diagram for Region Readout:

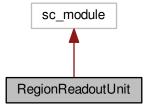


Classes

- · class RegionReadoutUnit
- 7.6.1 Detailed Description
- 7.6.2 Class Documentation
- 7.6.2.1 class RegionReadoutUnit

The RegionReadoutUnit class is a simple representation of the RRU in the Alpide chip. It has a member function that accepts pixel hits inputs, the RRU class will hold on to these pixels to determine if there are several pixels in the same cluster, and then put DATA_LONG or DATA_SHORT words into a SystemC FIFO.

Inheritance diagram for RegionReadoutUnit:



Collaboration diagram for RegionReadoutUnit:



Public Member Functions

RegionReadoutUnit (sc_core::sc_module_name name, PixelMatrix *matrix, unsigned int region_num, unsigned int fifo size, bool matrix readout speed, bool cluster enable)

Constructor for RegionReadoutUnit class.

void regionReadoutProcess (void)

SystemC process/method that implements the logic in the Region Readout Unit (RRU). NOTE: Should run at system clock frequency (40MHz).

- void regionMatrixReadoutFSM (void)
- void regionValidFSM (void)

SystemC process/method that implements the state machine that determines if the region is valid (has data this frame) Note: should run on Alpide system clock frequency.

void regionHeaderFSM (void)

SystemC process/method that implements the state machine that determines when the region header should be outputted Note: should run on Alpide system clock frequency.

void addTraces (sc_trace_file *wf, std::string name_prefix) const

Add SystemC signals to log in VCD trace file.

Public Attributes

• sc_in_clk s_system_clk_in

40MHz LHC clock

- sc in < bool > s frame readout start in
- sc_in< bool > s_readout_abort_in
- sc_in< bool > s_region_event_start_in

This comes from TRU, when reaodut of next frame from region FIFO to TRU FIFO should start.

• sc_in< bool > s_region_event_pop_in

This comes from TRU, when reaodut of next frame from region FIFO to TRU FIFO should start.

- $sc_in < bool > s_region_data_read_in$
- sc_out< bool > s_frame_readout_done_out
- sc_out< bool > s_region_fifo_empty_out
- $sc_out < bool > s_region_valid_out$
- sc_out< AlpideDataWord > s_region_data_out

Private Member Functions

• bool readoutNextPixel (PixelMatrix &matrix)

Read out the next pixel from this region's priority encoder. NOTE: This function should be called from a process that runs at the priority encoder readout clock. The function here will look for pixel clusters and generate DATA LONG words when possible if clustering is enabled, otherwise it will only send DATA SHORT words. See the flowchart for a better explanation of how this function works.

void flushRegionFifo (void)

Flush the region fifo. Used in data overrun mode. The function assumes that the fifo can be flushed in one clock cycle.

7.6 Region Readout 67

Private Attributes

- sc_signal< sc_uint< 8 >> s_rru_readout_state
- sc_signal< sc_uint< 8 >> s_rru_valid_state
- sc_signal< sc_uint< 1 >> s_rru_header_state
- sc_signal< bool > s_generate_region_header
- sc_signal < bool > s_region_matrix_empty_debug

Delayed one clock cycle compared to when it is used..

- sc signal< sc uint< 2 >> s matrix readout delay counter
- tlm::tlm fifo< AlpideDataWord > s_region_fifo
- sc_signal< sc_uint< 8 >> s_region_fifo_size
- AlpideRegionHeader mRegionHeader
- unsigned int mRegionId

The region handled by this RRU.

- bool mMatrixReadoutSpeed
- bool mMatrixReadoutCounter

Used with mMatrixReadoutSpeed to implement a delay when readout out pixel matrix.

std::uint16 t mPixelHitBaseAddr

Corresponds to pixel address in DATA SHORT/LONG words, in priority encoder order.

- std::uint8_t mPixelHitEncoderId
- std::uint8_t mPixelHitmap

Corresponds to hitmap in DATA LONG word.

- unsigned int mFifoSizeLimit
- bool mFifoSizeLimitEnabled
- bool mBusySignaled
- bool mClusteringEnabled
- bool mClusterStarted

Used in conjunction with mClusteringEnabled. Indicates that we have already received the first pixel in a potential cluster (stored in mPixelHitBaseAddr), and should continue building this cluster with subsequent hits that fall into the same pixel cluster range.

• PixelMatrix * mPixelMatrix

7.6.2.1.1 Constructor & Destructor Documentation

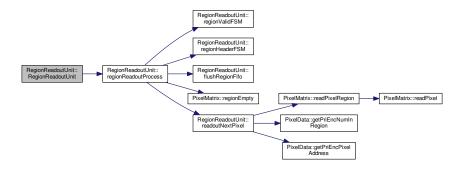
7.6.2.1.1.1 RegionReadoutUnit::RegionReadoutUnit (sc_core::sc_module_name name, PixelMatrix * matrix, unsigned int region_num, unsigned int fifo_size, bool matrix_readout_speed, bool cluster_enable)

Constructor for RegionReadoutUnit class.

Parameters

in	name	SystemC module name
in	matrix	Reference to pixel matrix
in	region_num	The region number that this RRU is assigned to
in	fifo_size	Size limit on the RRU's FIFO. 0 for no limit.
in	matrix_readout_speed	True for fast readout (2 clock cycles), false is slow (4 cycles).
in	cluster_enable	Enable/disable clustering and use of DATA LONG data words

Here is the call graph for this function:



7.6.2.1.2 Member Function Documentation

7.6.2.1.2.1 void RegionReadoutUnit::addTraces (sc_trace_file * wf, std::string name_prefix) const

Add SystemC signals to log in VCD trace file.

Parameters

in,out	wf	Pointer to VCD trace file object
in	name_prefix	Name prefix to be added to all the trace names

Todo Probably need to a stream << operator to allow values from fifo to be printed to trace file

 ${\color{red}\textbf{Todo}} \ \ \textbf{Probably need to a stream} << \textbf{operator to allow values from fifo to be printed to trace file}$

Here is the call graph for this function:



7.6.2.1.2.2 bool RegionReadoutUnit::readoutNextPixel(PixelMatrix & matrix) [private]

Read out the next pixel from this region's priority encoder. NOTE: This function should be called from a process that runs at the priority encoder readout clock. The function here will look for pixel clusters and generate DATA LONG words when possible if clustering is enabled, otherwise it will only send DATA SHORT words. See the flowchart for a better explanation of how this function works.

7.6 Region Readout 69

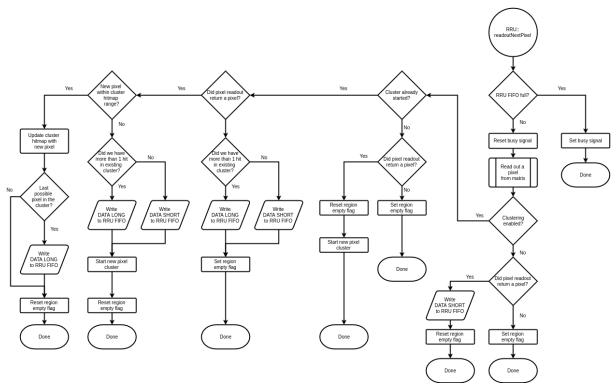


Figure 7.2 Flowchart for pixel readout and clustering in readoutNextPixel()

Parameters

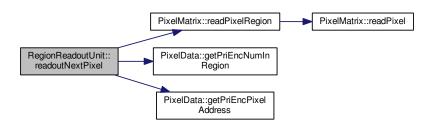
	in	matrix	Reference to pixel matrix
--	----	--------	---------------------------

Returns

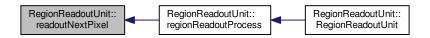
False if matrix is empty and no pixel was read out

Referenced by regionReadoutProcess().

Here is the call graph for this function:



Here is the caller graph for this function:



7.6.2.1.3 Member Data Documentation

7.6.2.1.3.1 bool RegionReadoutUnit::mMatrixReadoutSpeed [private]

Corresponds to Matrix Readout Speed bit in 0x0001 Mode Control register in Alpide chip. True: 20MHz readout. False: 10MHz readout.

Referenced by regionReadoutProcess().

7.6.2.1.3.2 std::uint8_t RegionReadoutUnit::mPixelHitEncoderId [private]

Corresponds to priority encoder id in DATA SHORT/LONG words, which is the priority encoder id (within the current region) that the current pixel belongs to.

Referenced by readoutNextPixel().

7.6.2.1.3.3 sc_in

| RegionReadoutUnit::s_frame_readout_start_in

This signal comes from FROMU, on deassertion of trigger, and indicates that start of readout from current pixel matrix event buffer to region FIFO can start

Referenced by addTraces(), and regionReadoutProcess().

7.7 Event Generation

Classes

· class EventGenerator

A simple event generator for Alpide SystemC simulation model. More...

- class Hit
- class TriggerEvent

Macros

• #define N_CHIPS 108

Variables

· const TriggerEvent NoTriggerEvent

A TriggerEvent that equals NoTriggerEvent is returned by some of the EventGenerator's functions which return a reference to an event, when there is no TriggerEvent to return.

7.7.1 Detailed Description

7.7.2 Class Documentation

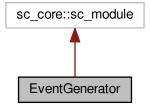
7.7.2.1 class EventGenerator

A simple event generator for Alpide SystemC simulation model.

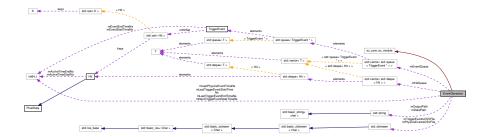
Physics events are generated at a rate that has an exponential distribution, with Lambda = 1 / average rate. The number of hits generated (hit multiplicity) per event can be based on a gaussian distribution or a user-defined discrete distribution. The ROOT macro export_multiplicity_data.cxx found under process/Multiplicity_distribution can be used to generate a discrete distribution based on real multiplicity data from ALICE.

The hits will currently be disributed randomly (with a flat/uniform distribution) among the different chips and over a chip's x/y coordinates. For each hit a fixed 2x2 pixel cluster is generated on the chip (this might be replaced with a more advanced random distribution in the future).

Inheritance diagram for EventGenerator:



Collaboration diagram for EventGenerator:



Public Member Functions

- EventGenerator (sc_core::sc_module_name name, const QSettings *settings, std::string output_path)
 Constructor for EventGenerator.
- void generateNextEvent ()
- void generateNextEvents (int n_events)
- const TriggerEvent & getNextTriggerEvent (void)

Get a reference to the next event (if there is one). Note: this function will keep returning the same event until it has been removeOldestEvent().

· void setBunchCrossingRate (int rate_ns)

Sets the bunch crossing rate, and recalculates the average crossing rate.

void setRandomSeed (int seed)

Sets the random seed used by random number generators.

void initRandomNumGenerator (void)

Initialize random number generators.

- void setPath (const std::string &path)
- void enableWriteToDisk (void)
- void disableWriteToDisk (void)
- void setNumEventsInMemAllowed (int n)
- int getTriggerFilterTime (void) const
- int getEventsInMem (void) const
- int getPhysicsEventCount (void) const
- int getTriggerEventCount (void) const
- void removeOldestEvent (void)

Remove the oldest event from the event queue (if there are any events in the queue, otherwise do nothing).

void physicsEventProcess (void)

SystemC controlled method, should be sensitive to the positive edge of the clock. Responsible for 1) Creating new physics events (hits) 2) Deleting old inactive hits.

void triggerEventProcess (void)

SystemC controlled method. It should be sensitive to the strobe signal, (both rising and falling edge) and is responsible for creating the triggerEvent objects after a STROBE pulse.

Public Attributes

- sc in< bool > s strobe in
- sc in clk s clk in
- sc_event_queue_port E_trigger_event_available
- sc_out< bool > s_physics_event_out

Private Member Functions

- void calculateAverageCrossingRate (void)
- void eventMemoryCountLimiter (void)

Limit the number of events stored in memory, as specified by mNumEventsInMemoryAllowed. The oldest events will be removed to bring the count below the threshold. If mWriteEventsToDisk is true, then the events that are removed will be written to disk.

TriggerEvent * generateNextTriggerEvent (int64_t event_start, int64_t event_end, int chip_id)

Create a new trigger event at the given start time. It checks if trigger event should be filtered or not, and updates trigger ID count.

int64 t generateNextPhysicsEvent (void)

Generate the next physics event (in the future). 1) Generate time till the next physics event 2) Generate hits for the next event, and put them on the hit queue 3) Update counters etc.

void readDiscreteDistributionFile (const char *filename, std::vector< double > &dist_vector) const

Read a discrete distribution from file and store it in a vector. The file format is a simple text file, with the following format: X0 Y0 X1 Y1 ... Xn Yn.

void scaleDiscreteDistribution (std::vector< double > &dist vector, double new mean value)

Scale the x axis of a discrete distribution, so that the distribution gets a new mean value.

unsigned int getRandomMultiplicity (void)

Return a random number of hits (multiplicity) based on the chosen distribution for multiplicity.

void addHitsToTriggerEvent (TriggerEvent &e)

Iterate through the hit queue corresponding to the chip_id associated with the event referenced by e, and add the active hits to it.

· void removeInactiveHits (void)

Remove old hits. Start at the front of the hit queue, and pop (remove) hits from the front while the hits are no longer active at current simulation time, and older than the oldest trigger event (so we don't delete hits that may be still be used in a trigger event that hasn't been processed yet).

Private Attributes

- std::vector< std::gueue< TriggerEvent * > > mEventQueue
- std::vector< std::deque< Hit > > mHitQueue
- · int mNumChips
- int mBunchCrossingRateNs
- int mAverageEventRateNs
- int mNumEventsInMemoryAllowed = 0

Number of events to keep in memory at a time. 0 = infinite.

• int mPhysicsEventCount = 0

Total number of physics and trigger events generated.

- int mTriggerEventIdCount = 0
- int64_t mLastPhysicsEventTimeNs = 0

Time of the last physics event that was generated.

- int64_t mLastTriggerEventStartTimeNs = 0
- int64_t mLastTriggerEventEndTimeNs = 0
- bool mStrobeActive = false
- int64_t mNextTriggerEventStartTimeNs = 0
- int mNextTriggerEventChipId = 0
- int mPixelDeadTime
- int mPixelActiveTime
- int mTriggerFilterTimeNs
- bool mTriggerFilteringEnabled = false
- bool mContinuousMode = false
- std::string mDataPath = "data"

- std::string mOutputPath
- bool mWriteEventsToDisk = false
- bool mCreateCSVFile = true
- std::ofstream mPhysicsEventsCSVFile
- std::ofstream mTriggerEventsCSVFile
- · int mRandomSeed
- boost::random::mt19937 mRandHitGen
- boost::random::mt19937 mRandHitMultiplicityGen
- boost::random::mt19937 mRandEventTimeGen
- $\bullet \ boost:: random:: uniform_int_distribution < int > * \ mRandHitChipID \\$

Uniform distribution used generating hit coordinates.

- boost::random::uniform_int_distribution< int > * mRandHitChipX
- boost::random::uniform_int_distribution< int > * mRandHitChipY
- boost::random::discrete_distribution * mRandHitMultiplicityDiscrete
- boost::random::normal_distribution< double > * mRandHitMultiplicityGauss
- $\bullet \ \ boost:: random:: exponential_distribution < double > * mRandEventTime \\$

Exponential distribution used for time between events.

- int mHitMultiplicityGaussAverage
- int mHitMultiplicityGaussDeviation

7.7.2.1.1 Constructor & Destructor Documentation

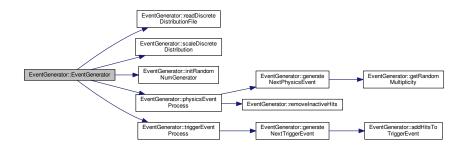
7.7.2.1.1.1 EventGenerator::EventGenerator (sc_core::sc_module_name name, const QSettings * settings, std::string output_path)

Constructor for EventGenerator.

Parameters

in	name	SystemC module name
in	settings	QSettings object with simulation settings.
in	output_path	Directory path to store simulation output data in

Here is the call graph for this function:



7.7.2.1.2 Member Function Documentation

7.7.2.1.2.1 void EventGenerator::addHitsToTriggerEvent (TriggerEvent & e) [private]

Iterate through the hit queue corresponding to the chip_id associated with the event referenced by e, and add the active hits to it.

Parameters

Todo Is this check worth it performance wise, or is it better to just iterate through the whole list?

Referenced by generateNextTriggerEvent().

Here is the caller graph for this function:



7.7.2.1.2.2 int64_t EventGenerator::generateNextPhysicsEvent (void) [private]

Generate the next physics event (in the future). 1) Generate time till the next physics event 2) Generate hits for the next event, and put them on the hit queue 3) Update counters etc.

Returns

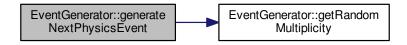
The number of clock cycles until this event will actually occur

Todo Account larger/bigger clusters here (when implemented)

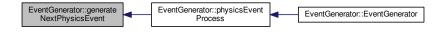
Todo Remove?

Referenced by physicsEventProcess().

Here is the call graph for this function:



Here is the caller graph for this function:



7.7.2.1.2.3 TriggerEvent * EventGenerator::generateNextTriggerEvent (int64_t event_start, int64_t event_end, int chip_id)

[private]

Create a new trigger event at the given start time. It checks if trigger event should be filtered or not, and updates trigger ID count.

Parameters

in	event_start	Start time of trigger event (time when strobe signal went high).
in	event_end	End time of trigger event (time when strobe signal went low again).
in	chip_id	Chip ID to generate event for

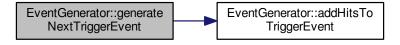
Returns

Pointer to new TriggerEvent object that was allocated on the stack. Caller must remember to delete it when done in order to free memory.

Todo Should I check distance between start time of two triggers? Or the distance in time between the end of the first trigger and the start of the next trigger?

Referenced by triggerEventProcess().

Here is the call graph for this function:



Here is the caller graph for this function:



7.7.2.1.2.4 const TriggerEvent & EventGenerator::getNextTriggerEvent (void)

Get a reference to the next event (if there is one). Note: this function will keep returning the same event until it has been removed by removeOldestEvent().

Returns

Reference to next event. If there are no events, then a reference to NoTriggerEvent (with event id = -1) is returned.

Referenced by Stimuli::stimuliEventProcess().

Here is the caller graph for this function:



7.7.2.1.2.5 unsigned int EventGenerator::getRandomMultiplicity (void) [private]

Return a random number of hits (multiplicity) based on the chosen distribution for multiplicity.

Returns

Number of hits

Exceptions

runtime_error if the EventGenerator for some reason does not have a multiplicity distribution initialized.

Referenced by generateNextPhysicsEvent().

Here is the caller graph for this function:



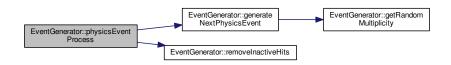
7.7.2.1.2.6 void EventGenerator::physicsEventProcess (void)

SystemC controled method, should be sensitive to the positive edge of the clock. Responsible for 1) Creating new physics events (hits) 2) Deleting old inactive hits.

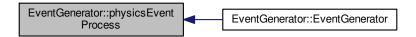
Todo Maybe do this only on strobe falling edge? Saves some CPU cycles that way?

Referenced by EventGenerator().

Here is the call graph for this function:



Here is the caller graph for this function:



7.7.2.1.2.7 void EventGenerator::readDiscreteDistributionFile (const char * filename, std::vector < double > & dist_vector)

const [private]

Read a discrete distribution from file and store it in a vector. The file format is a simple text file, with the following format: X0 Y0 X1 Y1 ... Xn Yn.

Where X-values correspond to the possible range of values for the random distribution, and the Y-values correspond to probability for a given X-value. X and Y is separated by whitespace. All X-values must be unsigned integers, and Y-values are assumed to be (positive) floating point.

The boost::random::discrete_distribution expects a list of probability values, where the index in the list corresponds to the X-value. This function generates a vector to represent that list. Missing X-values is allowed in the file, for example: 0 0.12 1 0.23 3 0.45

In the above example, an entry for the X-value of 2 with probability (Y) 0.0 will be inserted to the vector by this function.

Parameters

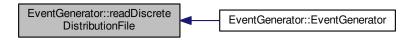
in	filename	Relative or absolute path and filename to open
out	dist_vector	Reference to vector to store the distribution in

Exceptions

runtime_error	If the file can not be opened
domain_error	If a negative x-value (hits) or y-value (probability) is encountered in the file

Referenced by EventGenerator().

Here is the caller graph for this function:



7.7.2.1.2.8 void EventGenerator::scaleDiscreteDistribution ($std::vector < double > \& dist_vector$, double new_mean_value) [private]

Scale the x axis of a discrete distribution, so that the distribution gets a new mean value.

Parameters

in,out	dist_vector	Distribution to scale. The original distribution in this vector will be overwritten and replaced with the new, scaled, distribution.
in	new_mean_value	The desired mean value of the new distribution.

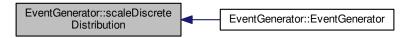
Exceptions

runtime_error	If dist_vector is empty, a runtime_error is thrown.
---------------	---

Todo This changes the mean value slightly.. and the sum isn't that far off 1.0 before this anyway...

Referenced by EventGenerator().

Here is the caller graph for this function:

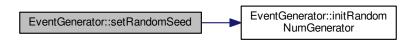


7.7.2.1.2.9 void EventGenerator::setRandomSeed (int seed)

Sets the random seed used by random number generators.

Todo More than one seed? What if seed is set after random number generators have been started?

Here is the call graph for this function:



7.7.2.1.3 Member Data Documentation

7.7.2.1.3.1 std::string EventGenerator::mDataPath = "data" [private]

Todo This is currently used.. remove or update code that uses it..

Referenced by removeOldestEvent().

7.7.2.1.3.2 std::vector<std::queue<TriggerEvent*>> EventGenerator::mEventQueue [private]

This is the trigger event queue (ie. the hits that occur between a strobe, which are fed to the Alpide chips). Each Alpide chip has its own queue (corresponding to an index in the vector).

 $Referenced\ by\ EventGenerator(),\ getNextTriggerEvent(),\ removeOldestEvent(),\ and\ triggerEventProcess().$

```
7.7.2.1.3.3 std::vector<std::deque<Hit>> EventGenerator::mHitQueue [private]
```

New hits will be push at the back, and old (expired) hits popped at the front. We need to be able to iterate over the queue, so a normal std::queue would not work. And deque seems faster than a list for our purpose: http://stackoverflow.com/questions/14574831/stddeque-or-stdlist But that should probably be tested:) Each Alpide chip has its own queue (corresponding to an index in the vector).

Referenced by addHitsToTriggerEvent(), EventGenerator(), generateNextPhysicsEvent(), and removeInactiveHits().

```
7.7.2.1.3.4 int64_t EventGenerator::mLastTriggerEventStartTimeNs = 0 [private]
```

Time of the last trigger event that was generated (time of last strobe) Will not be updated if trigger was filtered out.

Referenced by generateNextTriggerEvent(), and triggerEventProcess().

```
7.7.2.1.3.5 int EventGenerator::mNextTriggerEventChipId = 0 [private]
```

Used by getNextTriggerEvent() so it doesn't have to start iterating from the beginning of the event queue vector each time it is called. Also used by removeOldestEvent().

Referenced by getNextTriggerEvent(), removeOldestEvent(), and triggerEventProcess().

```
7.7.2.1.3.6 int64_t EventGenerator::mNextTriggerEventStartTimeNs = 0 [private]
```

Start time of next trigger event (start time recorded on STROBE rising edge). Event actually created and hits assigned to it on STROBE falling edge.

Referenced by triggerEventProcess().

```
7.7.2.1.3.7 boost::random::discrete_distribution* EventGenerator::mRandHitMultiplicityDiscrete [private]
```

Choice of discrete distribution (based on discrete list of N_hits vs Probability), or gaussian distribution.

Referenced by EventGenerator(), and getRandomMultiplicity().

```
7.7.2.1.3.8 int EventGenerator::mTriggerFilterTimeNs [private]
```

Minimum time between two triggers/events. Triggers/events that come sooner than this will be filtered out (but their hits will still be stored).

Referenced by EventGenerator(), and generateNextTriggerEvent().

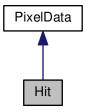
```
7.7.2.1.3.9 sc_out<bool> EventGenerator::s_physics_event_out
```

Active for one clock pulse every time we have a "physics event". Not really used for anything, just to indicate physics events in waveforms

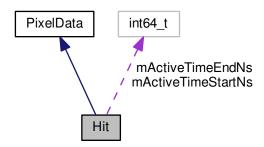
 $Referenced \ by \ physics Event Process (), \ and \ Stimuli::Stimuli().$

7.7.2.2 class Hit

Inheritance diagram for Hit:



Collaboration diagram for Hit:



Public Member Functions

- Hit (int col, int row, int64_t time_now_ns, int dead_time_ns, int active_time_ns)
 Constructor that calculates active start and end times based on current simulation time, and dead times and active times.
- Hit (int col, int row, int64_t time_active_start_ns, int64_t time_active_end_ns)

 Constructor that takes active start and end times directly.
- Hit (const Hit &h)
- bool operator== (const Hit &rhs) const
- bool operator> (const Hit &rhs) const
- bool operator< (const Hit &rhs) const
- bool operator>= (const Hit &rhs) const
- bool operator<= (const Hit &rhs) const
- Hit & operator= (const Hit &rhs)
- int64_t getActiveTimeStart (void) const
- int64_t getActiveTimeEnd (void) const
- bool isActive (int64_t time_now_ns) const

Check if this hit is currently active (which is equivalent to when analog pulse shape is over threshold).

• bool isActive (int64_t strobe_start_time_ns, int64_t strobe_end_time_ns) const

Check if this hit is active at any time during the specified time duration (between strobe_start_time_ns and strobe_← end_time_ns).

Private Attributes

- int64_t mActiveTimeStartNs
- int64_t mActiveTimeEndNs

7.7.2.2.1 Constructor & Destructor Documentation

7.7.2.2.1.1 Hit::Hit (int col, int row, int64_t time_now_ns, int dead_time_ns, int active_time_ns)

Constructor that calculates active start and end times based on current simulation time, and dead times and active times.

Parameters

in	col	Column number.
in	row	Row number.
in	time_now_ns	Time (in nanoseconds) when this hit occured (ie. current simulation time).
in	dead_time_ns	Dead time (in nanoseconds) before the hit "becomes active". This is equivalent to the time it takes for the analog signal to go above the threshold after a hit.
in	active_time_ns	Specifies (in nanoseconds) how long the hit stays active (ie. pixel is triggered) after the dead time has passed. This is equivalent to the amount time the analog pulse into the discriminator/comparator is over threshold.

7.7.2.2.1.2 Hit::Hit (int col, int row, int64_t time_active_start_ns, int64_t time_active_end_ns)

Constructor that takes active start and end times directly.

Parameters

in	col	Column number.
in	row	Row number.
in	time_active_start_ns	Absolute simulation time (in nanoseconds) for when the hit becomes active, which is equivalent to the analog signal going above the threshold after a hit.
in	time_active_end_ns	Absolute simulation time (in nanoseconds) for when the hit stops being active, which is equivalent to when the analog signal goes below the threshold again after having been active.

7.7.2.2.2 Member Function Documentation

7.7.2.2.2.1 bool Hit::isActive (int64_t time_now_ns) const [inline]

Check if this hit is currently active (which is equivalent to when analog pulse shape is over threshold).

Parameters

time_now_ns	Current simulation time (in nanoseconds).
-------------	---

Returns

True if active, false if not.

7.7.2.2.2.2 bool Hit::isActive (int64_t strobe_start_time_ns, int64_t strobe_end_time_ns) const [inline]

Check if this hit is active at any time during the specified time duration (between strobe_start_time_ns and strobe ← _end_time_ns).

Parameters

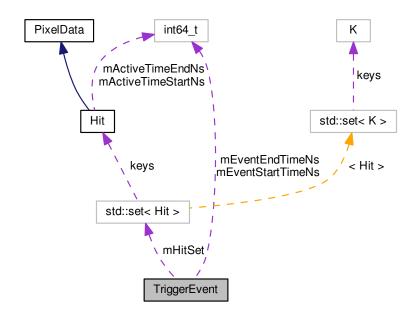
strobe_start_time_ns	Strobe start time	
strobe_end_time_ns	Strobe end time	

Returns

True if active, false if not.

7.7.2.3 class TriggerEvent

Collaboration diagram for TriggerEvent:



Public Member Functions

TriggerEvent (int64_t event_start_time_ns, int64_t event_end_time_ns, int chip_id, int event_id, bool filter_
 event=false)

Standard constructor.

• TriggerEvent (const TriggerEvent &e)

Copy constructor.

- void addHit (const Hit &h)
- void feedHitsToChip (PixelMatrix &matrix) const

Feed this event to the pixel matrix of the specified chip. If the trigger filter flag is set, or if there are no hits in the event, nothing will be sent to the chip, and a new event/MEB will not be created in the Alpide chip / pixel matrix object.

void writeToFile (const std::string path="")

Write this event to file, in XML format. The filename will be: "path/event< mEventId>.xml".

- void setEventFilteredFlag (bool value)
- int getEventSize (void) const
- · int getChipId (void) const
- · int getEventId (void) const
- int64_t getEventStartTime (void) const
- int64_t getEventEndTime (void) const
- · bool getEventFilteredFlag (void) const

Private Attributes

• int64 t mEventStartTimeNs

Absolute start time of event.

int64_t mEventEndTimeNs

Absolute end time of event.

- · int mEventId
- · int mChipld
- std::set< Hit > mHitSet
- · bool mEventFilteredFlag

This flag indicates that this event/trigger came too soon, and that it has been filtered out. The class object is still created to keep track of the pixels that are hit, but they will not be fed to the Alpide chip.

7.7.2.3.1 Constructor & Destructor Documentation

7.7.2.3.1.1 TriggerEvent::TriggerEvent (int64_t event_start_time_ns, int64_t event_end_time_ns, int chip_id, int event_id, bool filter_event = false)

Standard constructor.

Parameters

in	event_start_time_ns	Start time of trigger event (time when strobe was asserted)	
in	event_end_time_ns	nd_time_ns	
in	chip_id	Chip ID	
in	event_id	Event ID	
in	filter_event	Flag that indicates whether this trigger should be filtered or not (when trigger	
		filtering is enabled, and trigger came too close to last event)	

7.7.2.3.2 Member Function Documentation

7.7.2.3.2.1 void TriggerEvent::feedHitsToChip (PixelMatrix & matrix) const

Feed this event to the pixel matrix of the specified chip. If the trigger filter flag is set, or if there are no hits in the event, nothing will be sent to the chip, and a new event/MEB will not be created in the Alpide chip / pixel matrix object.

Parameters

out	matrix	Pixel matrix for the chip
-----	--------	---------------------------

Referenced by Stimuli::stimuliEventProcess().

Here is the call graph for this function:



Here is the caller graph for this function:



7.7.2.3.2.2 void TriggerEvent::writeToFile (const std::string path = " ")

Write this event to file, in XML format. The filename will be: "path/event<mEventId>.xml".

Todo Note in use.. Revisit this function, since I have changed this class a lot...

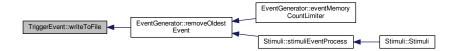
Parameters

in path Path to store file in.

Todo Implement layers etc.

Referenced by EventGenerator::removeOldestEvent().

Here is the caller graph for this function:



7.7.2.3.3 Member Data Documentation

7.7.2.3.3.1 bool TriggerEvent::mEventFilteredFlag [private]

This flag indicates that this event/trigger came too soon, and that it has been filtered out. The class object is still created to keep track of the pixels that are hit, but they will not be fed to the Alpide chip.

Todo With the new way of doing things, I don't need to have an Event object to keep track of hits, they are stored in the EventGenerator object. So I can get rid off this?

Referenced by feedHitsToChip(), and TriggerEvent().

7.8 Miscellaneous functions

Functions

template < class T >
 static void addTrace (sc_trace_file *wf, std::string name_prefix, std::string signal_name, T & signal)
 Add a SystemC signal/trace to VCD file, with desired signal hierarchy given by name_prefix.

7.8.1 Detailed Description

7.8.2 Function Documentation

7.8.2.1 template < class T > static void addTrace (sc_trace_file * wf, std::string name_prefix, std::string signal_name, T & signal) [inline], [static]

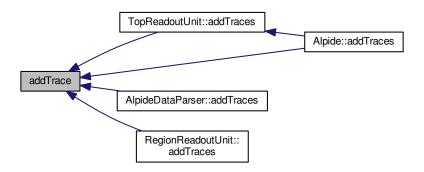
Add a SystemC signal/trace to VCD file, with desired signal hierarchy given by name_prefix.

Parameters

wf	VCD waveform file pointer	
name_prefix	Prefix to be added before signal name, used for signal hierarchy. A period (.) separates levels of hierarchy.	
signal_name	Name of the signal	
signal	The SystemC signal object	

Referenced by TopReadoutUnit::addTraces(), AlpideDataParser::addTraces(), Alpide::addTraces(), and Region \leftarrow ReadoutUnit::addTraces().

Here is the caller graph for this function:



7.9 Settings

Macros

- #define DEFAULT DATA OUTPUT WRITE VCD "true"
- #define DEFAULT DATA OUTPUT WRITE VCD CLOCK "false"
- #define DEFAULT DATA OUTPUT WRITE EVENT CSV "true"
- #define DEFAULT_SIMULATION_N_CHIPS "25000"
- #define DEFAULT_SIMULATION_N_EVENTS "10000"
- #define DEFAULT SIMULATION CONTINUOUS MODE "false"
- #define DEFAULT SIMULATION RANDOM SEED "0"
- #define DEFAULT_EVENT_HIT_MULTIPLICITY_DISTRIBUTION_TYPE "discrete"
- #define DEFAULT EVENT HIT MULTIPLICITY DISTRIBUTION FILE "multipl dist raw bins.txt"
- #define DEFAULT_EVENT_HIT_MULTIPLICITY_GAUSS_AVG "2000"
- #define DEFAULT EVENT HIT MULTIPLICITY GAUSS STDDEV "350"
- #define DEFAULT EVENT HIT DENSITY MIN BIAS PER CM2 "19"
- #define DEFAULT_EVENT_BUNCH_CROSSING_RATE_NS "25"
- #define DEFAULT EVENT AVERAGE EVENT RATE NS "2500"
- #define **DEFAULT_EVENT_TRIGGER_DELAY_NS** "1000"
- #define DEFAULT_EVENT_TRIGGER_FILTER_TIME_NS "10000"
- #define DEFAULT_EVENT_TRIGGER_FILTER_ENABLE "true"
- #define DEFAULT EVENT STROBE ACTIVE LENGTH NS "4800"
- #define DEFAULT_EVENT_STROBE_INACTIVE_LENGTH_NS "200"
- #define DEFAULT_ALPIDE_CLUSTERING_ENABLE "true"
- #define DEFAULT ALPIDE REGION FIFO SIZE "128"
- #define DEFAULT_ALPIDE_DMU_FIFO_SIZE "64"
- #define DEFAULT_ALPIDE_REGION_SIZE "32"
- #define DEFAULT ALPIDE PIXEL SHAPING DEAD TIME NS "200"
- #define DEFAULT ALPIDE PIXEL SHAPING ACTIVE TIME NS "6000"
- #define DEFAULT_ALPIDE_MATRIX_READOUT_SPEED_FAST "true"

Functions

QSettings * getSimSettings (const char *fileName="settings.txt")

Open a file with simulation settings. If the file does not exist, it will be created. If any settings are missing, they will be initialized with default values. If no filename is specified, the default settings.txt file is used in the current directory.

void setDefaultSimSettings (QSettings *readoutSimSettings)

Set default settings for each setting that is missing in the QSettings object.

7.9.1 Detailed Description

7.9.2 Function Documentation

7.9.2.1 QSettings* getSimSettings (const char * fileName)

Open a file with simulation settings. If the file does not exist, it will be created. If any settings are missing, they will be initialized with default values. If no filename is specified, the default settings.txt file is used in the current directory.

7.9 Settings 89

Parameters

in	fileName	File to open, relative to current directory. Defaults to settings.txt if not supplied.
		· · · · · · · · · · · · · · · · · · ·

Returns

Pointer to QSettings object initialized with all settings, either from settings file or with default settings if any settings were missing.

Referenced by sc_main().

Here is the call graph for this function:



Here is the caller graph for this function:



7.9.2.2 void setDefaultSimSettings (QSettings * readoutSimSettings)

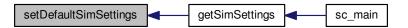
Set default settings for each setting that is missing in the QSettings object.

Parameters

in, out readoutSimSettings Pointer to QSettings object
--

Referenced by getSimSettings().

Here is the caller graph for this function:

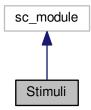


7.10 Main Alpide Simulation Testbench

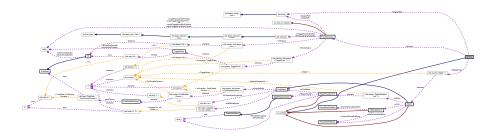
Classes

- · class Stimuli
- 7.10.1 Detailed Description
- 7.10.2 Class Documentation
- 7.10.2.1 class Stimuli

Inheritance diagram for Stimuli:



Collaboration diagram for Stimuli:



Public Member Functions

- Stimuli (sc_core::sc_module_name name, QSettings *settings, std::string output_path)

 Constructor for stimuli class. Instantiates and initializes the EventGenerator and Alpide objects, connects the SystemC ports.
- void stimuliMainProcess (void)

Main control of simulation stimuli, which mainly involves controlling the strobe signal and stop the simulation after the desired number of events.

void stimuliEventProcess (void)

SystemC controlled method. Waits for EventGenerator to notify the E_trigger_event_available notification queue that a new trigger event is available. When a trigger event is available it is fed to the Alpide chip(s).

void addTraces (sc_trace_file *wf) const

Add SystemC signals to log in VCD trace file.

void writeDataToFile (void) const

Write simulation data to file. Histograms for MEB usage from the Alpide chips, and trigger event statistics (number of accepted/rejected) in the chips are recorded here.

Public Attributes

- sc_in_clk clock
- sc_signal< bool > s_strobe_n
- sc_signal< bool > s_physics_event
- sc_signal < bool > s_chip_ready [100]
- sc_signal< sc_uint< 24 >> s_alpide_serial_data [100]
- sc_event_queue E_trigger_event_available

Private Attributes

- EventGenerator * mEvents
- std::vector< Alpide * > mAlpideChips
- const QSettings * mSettings
- std::string mOutputPath
- bool simulation done = false
- bool mContinuousMode
- int mNumEvents
- int mNumChips
- int mStrobeActiveNs
- · int mStrobelnactiveNs
- int mTriggerDelayNs

7.10.2.1.1 Constructor & Destructor Documentation

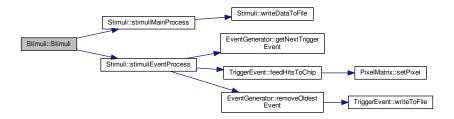
7.10.2.1.1.1 Stimuli::Stimuli (sc_core::sc_module_name name, QSettings * settings, std::string output_path)

Constructor for stimuli class. Instantiates and initializes the EventGenerator and Alpide objects, connects the SystemC ports.

Parameters

in	name	SystemC module name QSettings object with simulation settings.	
in	settings		
in	in output_path Path to store output files generated by the Stim		

Here is the call graph for this function:



7.10.2.1.2 Member Function Documentation

7.10.2.1.2.1 void Stimuli::addTraces (sc_trace_file * wf) const

Add SystemC signals to log in VCD trace file.

Parameters

in,out	wf	VCD waveform file pointer
--------	----	---------------------------

Referenced by sc_main().

Here is the caller graph for this function:



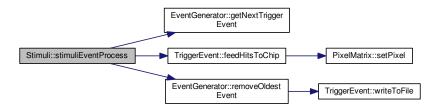
7.10.2.1.2.2 void Stimuli::stimuliEventProcess (void)

SystemC controlled method. Waits for EventGenerator to notify the E_trigger_event_available notification queue that a new trigger event is available. When a trigger event is available it is fed to the Alpide chip(s).

Todo Check if there are actually events? Throw an error if we get notification but there are not events?

Referenced by Stimuli().

Here is the call graph for this function:



Here is the caller graph for this function:



7.10.2.1.3 Member Data Documentation

7.10.2.1.3.1 int Stimuli::mNumEvents [private]

Todo Make it a 64-bit int?

 $Referenced \ by \ Stimuli(), \ stimuliMainProcess(), \ and \ writeDataToFile().$

Chapter 8

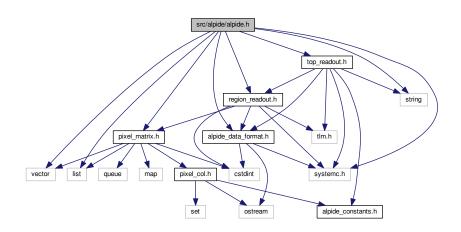
File Documentation

8.1 src/alpide/alpide.h File Reference

Source file for Alpide class.

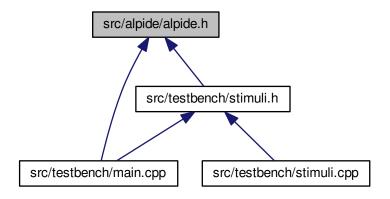
```
#include "alpide_data_format.h"
#include "pixel_matrix.h"
#include "region_readout.h"
#include "top_readout.h"
#include <systemc.h>
#include <vector>
#include <list>
#include <string>
```

Include dependency graph for alpide.h:



96 File Documentation

This graph shows which files directly or indirectly include this file:



Classes

• class Alpide

8.1.1 Detailed Description

Source file for Alpide class.

Header file for Alpide class.

Author

Simon Voigt Nesbo

Date

December 12, 2016

Author

Simon Voigt Nesbo

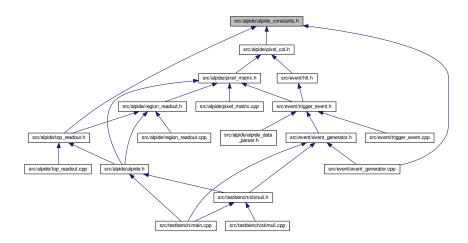
Date

December 11, 2016

8.2 src/alpide/alpide_constants.h File Reference

Various constants for alpide chip, such as pixel matrix width and heigh, fifo depths, etc.

This graph shows which files directly or indirectly include this file:



Macros

- #define N_REGIONS 32
- #define N_PIXEL_ROWS 512
- #define N_PIXEL_COLS 1024
- #define **N_PIXEL_COLS_PER_REGION** (N_PIXEL_COLS/N_REGIONS)
- #define N_PIXEL_DOUBLE_COLS_PER_REGION (N_PIXEL_COLS_PER_REGION/2)
- #define **N_PIXELS_PER_REGION** (N_PIXEL_COLS/N_REGIONS)
- #define TRU_FRAME_FIFO_ALMOST_FULL1 48
- #define TRU_FRAME_FIFO_ALMOST_FULL2 56
- #define TRU_FRAME_FIFO_SIZE 64
- #define **DATA_LONG_PIXMAP_SIZE** ((unsigned int) 7)
- #define LHC_ORBIT_BUNCH_COUNT 3564
- #define CHIP_WIDTH_CM 3
- #define CHIP_HEIGHT_CM 1.5

8.2.1 Detailed Description

Various constants for alpide chip, such as pixel matrix width and heigh, fifo depths, etc.

Author

Simon Voigt Nesbo

Date

November 27, 2016

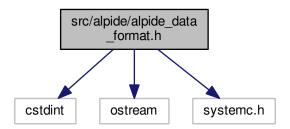
98 File Documentation

8.3 src/alpide/alpide_data_format.h File Reference

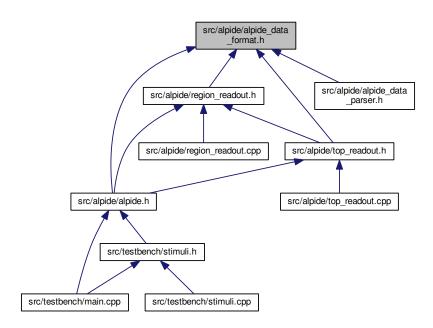
Definitions for data format used in Alpide chip.

```
#include <cstdint>
#include <ostream>
#include <systemc.h>
```

Include dependency graph for alpide_data_format.h:



This graph shows which files directly or indirectly include this file:



Classes

· struct FrameStartFifoWord

Data word stored in FRAME START FIFO. More ...

struct FrameEndFifoWord

Data word stored in FRAME END FIFO. More...

· class AlpideDataWord

The FIFOs in the Alpide chip are 24 bits, or 3 bytes, wide. This is a base class for the data words that holds 3 bytes, and is used as the data type in the SystemC FIFO templates. This class shouldn't be used on its own, the various types of data words are implemented in derived classes. More...

- · class Alpideldle
- · class AlpideChipHeader
- · class AlpideChipTrailer
- · class AlpideChipEmptyFrame
- · class AlpideRegionHeader
- · class AlpideRegionTrailer
- · class AlpideDataShort
- · class AlpideDataLong
- · class AlpideBusyOn
- · class AlpideBusyOff
- · class AlpideComma

Variables

- const uint8_t DW_IDLE = 0b111111111
- const uint8_t DW_CHIP_HEADER = 0b10100000
- const uint8_t DW_CHIP_TRAILER = 0b10110000
- const uint8 t DW_CHIP_EMPTY_FRAME = 0b11100000
- const uint8 t DW REGION HEADER = 0b11000000
- const uint8 t DW_REGION_TRAILER = 0b11110011
- const uint8_t DW_DATA_SHORT = 0b01000000
- const uint8_t DW_DATA_LONG = 0b000000000
- const uint8_t DW_BUSY_ON = 0b111110001
- const uint8_t DW_BUSY_OFF = 0b11110000
- const uint8_t DW_COMMA = 0b111111110
- const uint8_t **READOUT_FLAGS_BUSY_VIOLATION** = 0b00001000
- const uint8 t READOUT FLAGS FLUSHED INCOMPLETE = 0b00000100
- const uint8_t READOUT_FLAGS_STROBE_EXTENDED = 0b00000010
- const uint8 t READOUT_FLAGS_BUSY_TRANSITION = 0b000000001
- const uint8_t MASK_IDLE_BUSY_COMMA = 0b111111111

Mask for busy, idle and comma words.

• const uint8_t MASK_CHIP = 0b11110000

Mask for chip header/trailer/empty frame words.

• const uint8_t MASK_REGION_HEADER = 0b11100000

Mask for region header word.

const uint8_t MASK_DATA = 0b11000000

Mask for data short/long words.

8.3.1 Detailed Description

Definitions for data format used in Alpide chip.

Author

Simon Voigt Nesbo

Date

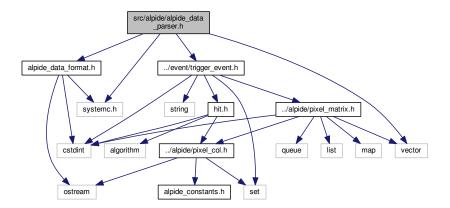
February 20, 2017

100 File Documentation

8.4 src/alpide/alpide_data_parser.h File Reference

Classes for parsing serial data from Alpide chip, and building/reconstructing events/frames from the data.

```
#include "alpide_data_format.h"
#include "../event/trigger_event.h"
#include <vector>
#include <systemc.h>
Include dependency graph for alpide_data_parser.h:
```



Classes

- struct AlpideDataParsed
- · class AlpideEventFrame
- · class AlpideEventBuilder
- · class AlpideDataParser

Enumerations

enum AlpideDataTypes {

ALPIDE_IDLE, ALPIDE_CHIP_HEADER1, ALPIDE_CHIP_HEADER2, ALPIDE_CHIP_TRAILER, ALPIDE_CHIP_EMPTY_FRAME1, ALPIDE_CHIP_EMPTY_FRAME2, ALPIDE_REGION_HEADER, AL⇔ PIDE_DATA_SHORT1,

ALPIDE_DATA_SHORT2, ALPIDE_DATA_LONG1, ALPIDE_DATA_LONG2, ALPIDE_DATA_LONG3, ALPIDE_BUSY_ON, ALPIDE_BUSY_OFF, ALPIDE_COMMA, ALPIDE_UNKNOWN }

8.4.1 Detailed Description

Classes for parsing serial data from Alpide chip, and building/reconstructing events/frames from the data.

Author

Simon Voigt Nesbo

Date

March 6, 2017

Todo Move this class to a separate directory/module. Don't mix it with the Alpide simulation model.

Author

Simon Voigt Nesbo

Date

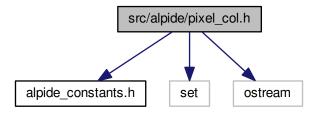
March 3, 2017

8.5 src/alpide/pixel_col.h File Reference

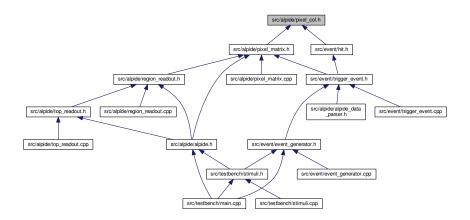
Source file for pixel column, double column, and priority encoder classes.

```
#include "alpide_constants.h"
#include <set>
#include <ostream>
```

Include dependency graph for pixel_col.h:



This graph shows which files directly or indirectly include this file:



Classes

· class PixelData

A struct that indicates a hit in a region, at the pixel identified by the col and row variables. For each hit an object of this type will be inserted into the std::set container in regionDataVector. For the pixels that don't have hits there will not be an object of this type inserted. Column should be 0 or 1. Row can be any value from 0 to N_PIXEL_ROWS-1. More...

· class PixelPriorityEncoder

Comparator class/function for use with the PixelData class in the std::set container, which allows the container to sort the PixelData entries in a meaningful way. The picture below is from the ALPIDE operations manual, and shows:

• class PixelDoubleColumn

Functions

• const PixelData NoPixelHit (-1,-1)

8.5.1 Detailed Description

Source file for pixel column, double column, and priority encoder classes.

Pixel column, double column, and priority encoder classes.

Author

Simon Voigt Nesbo

Date

November 27, 2016

Author

Simon Voigt Nesbo

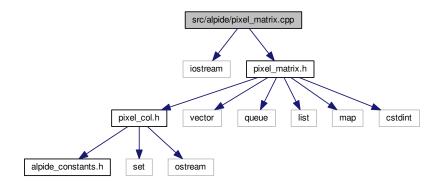
Date

November 27, 2016 Detailed description of file.

8.6 src/alpide/pixel_matrix.cpp File Reference

Source file for pixel matrix class.

```
#include <iostream>
#include "pixel_matrix.h"
Include dependency graph for pixel_matrix.cpp:
```



8.6.1 Detailed Description

Source file for pixel matrix class.

Author

Simon Voigt Nesbo

Date

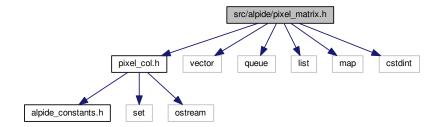
December 11, 2016 Pixel matrix class comprises all the pixel regions, which allows to interface in terms of absolute coordinates with the pixel matrix. Special version for the Alpide Dataflow SystemC model.

8.7 src/alpide/pixel_matrix.h File Reference

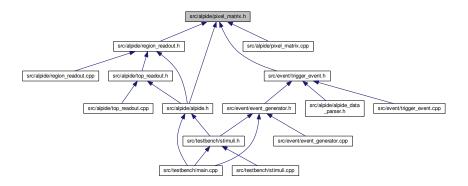
Header file for pixel matrix class.

```
#include "pixel_col.h"
#include <vector>
#include <queue>
#include <list>
#include <map>
#include <cstdint>
```

Include dependency graph for pixel_matrix.h:



This graph shows which files directly or indirectly include this file:



Classes

class PixelMatrix

8.7.1 Detailed Description

Header file for pixel matrix class.

Author

Simon Voigt Nesbo

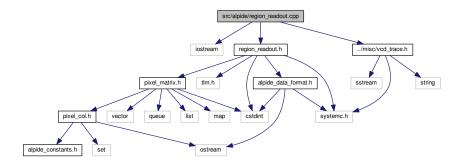
Date

December 11, 2016 Pixel matrix class comprises all the pixel regions, which allows to interface in terms of absolute coordinates with the pixel matrix. Special version for the Alpide Dataflow SystemC model.

8.8 src/alpide/region_readout.cpp File Reference

Class for implementing the Region Readout Unit (RRU) in the Alpide chip.

```
#include <iostream>
#include "region_readout.h"
#include "../misc/vcd_trace.h"
Include dependency graph for region_readout.cpp:
```



Functions

• SC_HAS_PROCESS (RegionReadoutUnit)

8.8.1 Detailed Description

Class for implementing the Region Readout Unit (RRU) in the Alpide chip.

Author

Simon Voigt Nesbo

Date

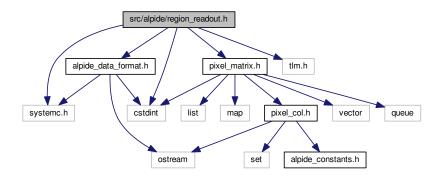
February 20, 2017

8.9 src/alpide/region_readout.h File Reference

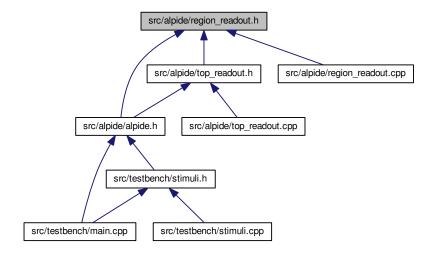
Class for implementing the Region Readout Unit (RRU) in the Alpide chip.

```
#include "alpide_data_format.h"
#include "pixel_matrix.h"
#include <cstdint>
#include <systemc.h>
#include <tlm.h>
```

Include dependency graph for region readout.h:



This graph shows which files directly or indirectly include this file:



Classes

• class RegionReadoutUnit

Enumerations

- enum { IDLE = 0, START_READOUT = 1, READOUT_AND_CLUSTERING = 2, REGION_TRAILER = 3 }
- enum { IDLE = 0, EMPTY = 1, VALID = 2, POP = 3 }
- enum { **HEADER** = 0, **DATA** = 1 }

8.9.1 Detailed Description

Class for implementing the Region Readout Unit (RRU) in the Alpide chip.

Author

Simon Voigt Nesbo

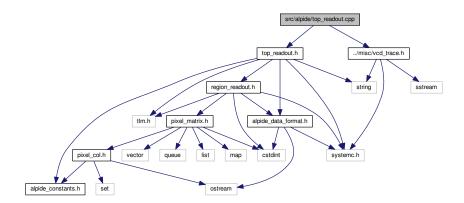
Date

February 20, 2017

8.10 src/alpide/top_readout.cpp File Reference

Class for implementing the Top Readout Unit (TRU) in the Alpide chip.

```
#include "top_readout.h"
#include "../misc/vcd_trace.h"
Include dependency graph for top_readout.cpp:
```



Functions

• SC_HAS_PROCESS (TopReadoutUnit)

8.10.1 Detailed Description

Class for implementing the Top Readout Unit (TRU) in the Alpide chip.

Author

Simon Voigt Nesbo

Date

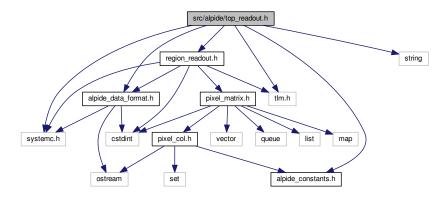
February 20, 2017

8.11 src/alpide/top_readout.h File Reference

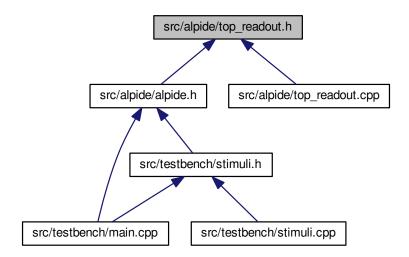
Class for implementing the Top Readout Unit (TRU) in the Alpide chip.

```
#include "region_readout.h"
#include "alpide_constants.h"
#include "alpide_data_format.h"
#include <string>
#include <systemc.h>
#include <tlm.h>
```

Include dependency graph for top_readout.h:



This graph shows which files directly or indirectly include this file:



Classes

· class TopReadoutUnit

8.11.1 Detailed Description

Class for implementing the Top Readout Unit (TRU) in the Alpide chip.

Author

Simon Voigt Nesbo

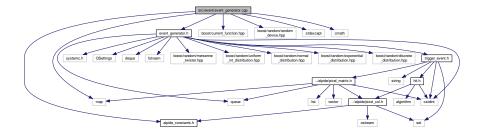
Date

February 20, 2017

8.12 src/event/event_generator.cpp File Reference

```
#include "event_generator.h"
#include "../alpide/alpide_constants.h"
#include <boost/current_function.hpp>
#include <boost/random/random_device.hpp>
#include <stdexcept>
#include <cmath>
#include <map>
```

Include dependency graph for event_generator.cpp:



Macros

#define print_function_timestamp()

Functions

• SC_HAS_PROCESS (EventGenerator)

8.12.1 Detailed Description

Author

Simon Voigt Nesbo

Date

December 22, 2016

A simple event generator for Alpide SystemC simulation model.

8.12.2 Macro Definition Documentation

8.12.2.1 #define print_function_timestamp()

Value:

```
std::cout << std::endl << "@ " << sc_time_stamp().value() << " ns\t"; \
    std::cout << BOOST_CURRENT_FUNCTION << ":" << std::endl; \
    std::cout << "-----"; \
    std::cout << "----" << std::endl;</pre>
```

8.13 src/event/event_generator.h File Reference

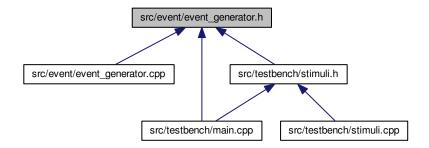
A simple event generator for Alpide SystemC simulation model.

```
#include "trigger_event.h"
#include <systemc.h>
#include <QSettings>
#include <queue>
#include <deque>
#include <fstream>
#include <boost/random/mersenne_twister.hpp>
#include <boost/random/uniform_int_distribution.hpp>
#include <boost/random/normal_distribution.hpp>
#include <boost/random/exponential_distribution.hpp>
#include <boost/random/exponential_distribution.hpp>
#include <cstdint>
```

Include dependency graph for event_generator.h:



This graph shows which files directly or indirectly include this file:



Classes

· class EventGenerator

A simple event generator for Alpide SystemC simulation model. More...

Macros

• #define N_CHIPS 108

8.13.1 Detailed Description

A simple event generator for Alpide SystemC simulation model.

Author

Simon Voigt Nesbo

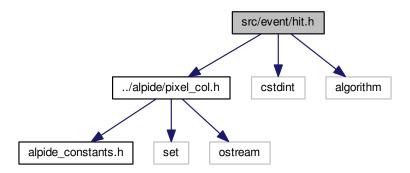
Date

December 22, 2016

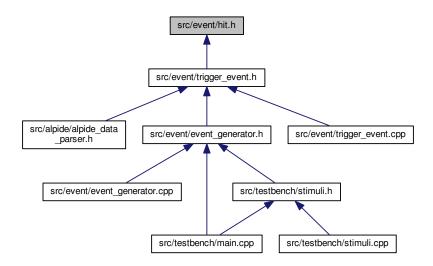
8.14 src/event/hit.h File Reference

Source file for PixelData and Hit classes. These classes hold the coordinates for a discrete hit in the Alpide chip, along with information about when the hit is active (equivalent to when the analog pulse out of the amplifier and shaping stage in the analog front end goes above the threshold).

```
#include "../alpide/pixel_col.h"
#include <cstdint>
#include <algorithm>
Include dependency graph for hit.h:
```



This graph shows which files directly or indirectly include this file:



Classes

class Hit

8.14.1 Detailed Description

Source file for PixelData and Hit classes. These classes hold the coordinates for a discrete hit in the Alpide chip, along with information about when the hit is active (equivalent to when the analog pulse out of the amplifier and shaping stage in the analog front end goes above the threshold).

Header file for PixelData and Hit classes. These classes hold the coordinates for a discrete hit in the Alpide chip, along with information about when the hit is active (equivalent to when the analog pulse out of the amplifier and shaping stage in the analog front end goes above the threshold).

Author

Simon Voigt Nesbo

Date

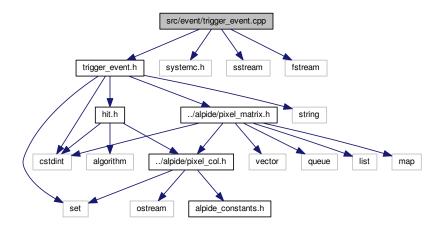
December 12, 2016

8.15 src/event/trigger_event.cpp File Reference

Event class for Alpide SystemC simulation model. This class holds all the pixel hits for an event for the whole detector. The philosophy behind this class is that the shaping etc. is performed by this class and the Event Generator class, and that the pixel hits here can be fed directly to the Alpide chip at the given time.

```
#include "trigger_event.h"
#include <systemc.h>
#include <sstream>
#include <fstream>
```

Include dependency graph for trigger event.cpp:



Variables

• const TriggerEvent NoTriggerEvent (0, 0,-1,-1)

A TriggerEvent that equals NoTriggerEvent is returned by some of the EventGenerator's functions which return a reference to an event, when there is no TriggerEvent to return.

8.15.1 Detailed Description

Event class for Alpide SystemC simulation model. This class holds all the pixel hits for an event for the whole detector. The philosophy behind this class is that the shaping etc. is performed by this class and the Event—Generator class, and that the pixel hits here can be fed directly to the Alpide chip at the given time.

Author

Simon Voigt Nesbo

Date

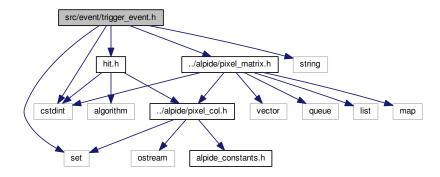
December 12, 2016

8.16 src/event/trigger_event.h File Reference

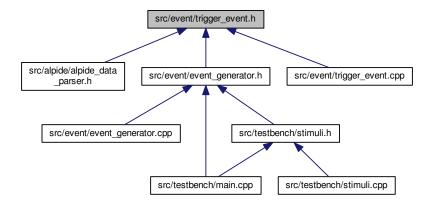
Trigger event class for Alpide SystemC simulation model. This class holds all the pixel hits for a trigger event for the whole detector. The philosophy behind this class is that the shaping etc. is performed by this class and the EventGenerator class, and that the pixel hits here can be fed directly to the Alpide chip at the given time.

```
#include "hit.h"
#include "../alpide/pixel_matrix.h"
#include <string>
#include <set>
#include <cstdint>
```

Include dependency graph for trigger_event.h:



This graph shows which files directly or indirectly include this file:



Classes

class TriggerEvent

Variables

· const TriggerEvent NoTriggerEvent

A TriggerEvent that equals NoTriggerEvent is returned by some of the EventGenerator's functions which return a reference to an event, when there is no TriggerEvent to return.

8.16.1 Detailed Description

Trigger event class for Alpide SystemC simulation model. This class holds all the pixel hits for a trigger event for the whole detector. The philosophy behind this class is that the shaping etc. is performed by this class and the EventGenerator class, and that the pixel hits here can be fed directly to the Alpide chip at the given time.

Author

Simon Voigt Nesbo

Date

January 2, 2017

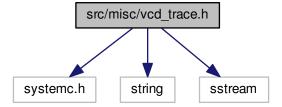
Todo Use SystemC time data type instead of int64_t?

8.17 src/misc/vcd_trace.h File Reference

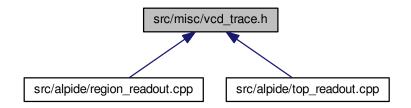
Common function for adding SystemC signals etc. to Value Change Dump (VCD) file.

```
#include <systemc.h>
#include <string>
#include <sstream>
```

Include dependency graph for vcd trace.h:



This graph shows which files directly or indirectly include this file:



Functions

template < class T >
 static void addTrace (sc_trace_file *wf, std::string name_prefix, std::string signal_name, T & signal)
 Add a SystemC signal/trace to VCD file, with desired signal hierarchy given by name_prefix.

8.17.1 Detailed Description

Common function for adding SystemC signals etc. to Value Change Dump (VCD) file.

Author

Simon Voigt Nesbo

Date

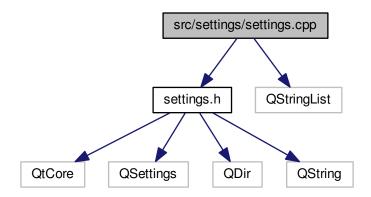
March 3, 2017

8.18 src/settings/settings.cpp File Reference

Source file for simulation settings file.

```
#include "settings.h"
#include <QStringList>
```

Include dependency graph for settings.cpp:



Functions

• QSettings * getSimSettings (const char *fileName)

Open a file with simulation settings. If the file does not exist, it will be created. If any settings are missing, they will be initialized with default values. If no filename is specified, the default settings.txt file is used in the current directory.

void setDefaultSimSettings (QSettings *readoutSimSettings)

Set default settings for each setting that is missing in the QSettings object.

8.18.1 Detailed Description

Source file for simulation settings file.

Author

Simon Voigt Nesbo svn@hib.no

Date

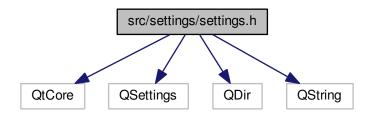
November 3, 2016 Some functions for reading the simulation settings file, and for initializing default settings if the settings file, or certain settings, are missing.

8.19 src/settings/settings.h File Reference

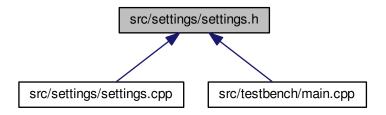
Header file for simulation settings file handling.

#include <QtCore>
#include <QSettings>
#include <QDir>
#include <QString>

Include dependency graph for settings.h:



This graph shows which files directly or indirectly include this file:



Macros

- #define **DEFAULT_DATA_OUTPUT_WRITE_VCD** "true"
- #define DEFAULT_DATA_OUTPUT_WRITE_VCD_CLOCK "false"
- #define DEFAULT_DATA_OUTPUT_WRITE_EVENT_CSV "true"
- #define DEFAULT SIMULATION N CHIPS "25000"
- #define DEFAULT_SIMULATION_N_EVENTS "10000"
- #define **DEFAULT_SIMULATION_CONTINUOUS_MODE** "false"
- #define DEFAULT SIMULATION RANDOM SEED "0"
- #define DEFAULT_EVENT_HIT_MULTIPLICITY_DISTRIBUTION_TYPE "discrete"
- #define DEFAULT_EVENT_HIT_MULTIPLICITY_DISTRIBUTION_FILE "multipl_dist_raw_bins.txt"
- #define DEFAULT_EVENT_HIT_MULTIPLICITY_GAUSS_AVG "2000"
- #define DEFAULT_EVENT_HIT_MULTIPLICITY_GAUSS_STDDEV "350"
- #define DEFAULT EVENT HIT DENSITY MIN BIAS PER CM2 "19"
- #define DEFAULT_EVENT_BUNCH_CROSSING_RATE_NS "25"
- #define DEFAULT EVENT_AVERAGE EVENT_RATE_NS "2500"
- #define **DEFAULT_EVENT_TRIGGER_DELAY_NS** "1000"

- #define DEFAULT_EVENT_TRIGGER_FILTER_TIME_NS "10000"
- #define DEFAULT_EVENT_TRIGGER_FILTER_ENABLE "true"
- #define DEFAULT_EVENT_STROBE_ACTIVE_LENGTH_NS "4800"
- #define DEFAULT EVENT STROBE INACTIVE LENGTH NS "200"
- #define DEFAULT_ALPIDE_CLUSTERING_ENABLE "true"
- #define DEFAULT_ALPIDE_REGION_FIFO_SIZE "128"
- #define DEFAULT_ALPIDE_DMU_FIFO_SIZE "64"
- #define DEFAULT_ALPIDE_REGION_SIZE "32"
- #define DEFAULT ALPIDE PIXEL SHAPING DEAD TIME NS "200"
- #define DEFAULT ALPIDE PIXEL SHAPING ACTIVE TIME NS "6000"
- #define DEFAULT_ALPIDE_MATRIX_READOUT_SPEED_FAST "true"

Functions

QSettings * getSimSettings (const char *fileName="settings.txt")

Open a file with simulation settings. If the file does not exist, it will be created. If any settings are missing, they will be initialized with default values. If no filename is specified, the default settings.txt file is used in the current directory.

void setDefaultSimSettings (QSettings *readoutSimSettings)

Set default settings for each setting that is missing in the QSettings object.

8.19.1 Detailed Description

Header file for simulation settings file handling.

Author

Simon Voigt Nesbo svn@hib.no

Date

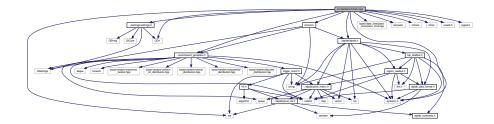
November 3, 2016 This file has definitions for default simulation settings, which can be used as default values or for generating the settings file if it is missing.

8.20 src/testbench/main.cpp File Reference

Main source file for Alpide Dataflow SystemC simulation testbench.

```
#include "../settings/settings.h"
#include "../event/event_generator.h"
#include "../alpide/alpide.h"
#include "stimuli.h"
#include <systemc.h>
#include "boost/date_time/posix_time/posix_time.hpp"
#include <iostream>
#include <chrono>
#include <ctime>
#include <QDir>
#include <unistd.h>
#include <signal.h>
```

Include dependency graph for main.cpp:



Functions

- void signal_callback_handler (int signum)
 - Callback function for CTRL+C (SIGINT) signal, used for exiting the simulation nicely and not lose data if the user presses CTRL+C on the command line.
- std::string create_output_dir (const QSettings *settings)
 - Create output directory "\$PWD/sim_output/Run < timestamp>". Also writes a copy of the settings file used for the simulation to this path.
- int sc_main (int argc, char **argv)

Variables

• volatile bool **g_terminate_program** = false

8.20.1 Detailed Description

Main source file for Alpide Dataflow SystemC simulation testbench.

Author

Simon Voigt Nesbo

Date

December 11, 2016

8.20.2 Function Documentation

8.20.2.1 std::string create_output_dir (const QSettings * settings)

Create output directory "\$PWD/sim_output/Run <timestamp>". Also writes a copy of the settings file used for the simulation to this path.

Returns

Output directory path string

Referenced by sc_main().

Here is the caller graph for this function:

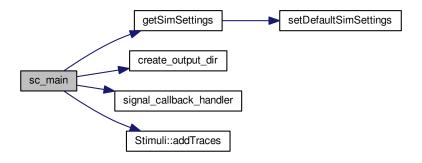


8.20.2.2 int sc_main (int argc, char ** argv)

Todo Pass vcd trace object to constructor of Stimuli class and Alpide classes?

Todo Add a warning here if user tries to simulate over 1000 events with this option enabled, because it will consume 100s of megabytes

Here is the call graph for this function:

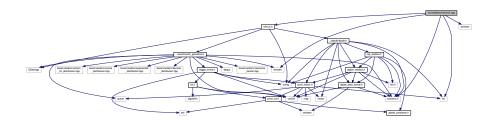


8.21 src/testbench/stimuli.cpp File Reference

Source file for stimuli function for Alpide Dataflow SystemC model.

```
#include "stimuli.h"
#include <systemc.h>
#include <list>
#include <sstream>
#include <fstream>
```

Include dependency graph for stimuli.cpp:



Functions

- void print_event_rate (const std::list< int > &t_delta_queue)
 Takes a list of t_delta values (time between events) for the last events, calculates the average event rate over those events, and prints it to std::cout. The list must be maintained by the caller.
- SC HAS PROCESS (Stimuli)

Variables

volatile bool g_terminate_program

8.21.1 Detailed Description

Source file for stimuli function for Alpide Dataflow SystemC model.

Author

Simon Voigt Nesbo

Date

December 12, 2016

8.21.2 Function Documentation

8.21.2.1 void print_event_rate (const std::list< int > & t_delta_queue)

Takes a list of t_delta values (time between events) for the last events, calculates the average event rate over those events, and prints it to std::cout. The list must be maintained by the caller.

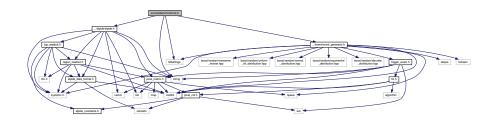
Todo Update/fix/remove this function.. currently not used..

8.22 src/testbench/stimuli.h File Reference

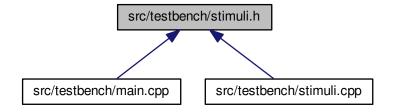
Header file for stimuli function for Alpide Dataflow SystemC model.

```
#include "../alpide/alpide.h"
#include "../event/event_generator.h"
#include <QSettings>
#include <string>
```

Include dependency graph for stimuli.h:



This graph shows which files directly or indirectly include this file:



Classes

• class Stimuli

8.22.1 Detailed Description

Header file for stimuli function for Alpide Dataflow SystemC model.

Author

Simon Voigt Nesbo

Date

December 12, 2016

Index

\sim EventGenerator	DW_DATA_LONG, 95
EventGenerator, 29	DW_DATA_SHORT, 95
	DW_IDLE, 96
addHit	DW_REGION_HEADER, 96
TriggerEvent, 84	operator<<, 95
addHitsToTriggerEvent	AlpideBusyOff, 15
EventGenerator, 29	AlpideBusyOff, 16
addTraces	AlpideBusyOn, 16
Alpide, 11	AlpideBusyOn, 17
RegionReadoutUnit, 69	AlpideChipEmptyFrame, 17
Stimuli, 74	AlpideChipEmptyFrame, 18
TopReadoutUnit, 80	AlpideChipHeader, 18
Alpide, 9	AlpideChipHeader, 19
addTraces, 11	AlpideChipTrailer, 19
Alpide, 10	AlpideChipTrailer, 20
dataTransmission, 12	AlpideDataLong, 20
getChipId, 12	AlpideDataLong, 21
mChipld, 13	AlpideDataShort, 21
mEnableReadoutTraces, 13	AlpideDataShort, 22
mRRUs, 13	AlpideDataWord, 22
mTRU, 13	data, 23
matrixReadout, 12	Alpideldle, 24
s_event_buffers_used, 14	Alpideldle, 24
s_matrix_readout_clk_in, 14	AlpideRegionHeader, 25
s_oldest_event_number_of_hits, 14	AlpideRegionHeader, 25
s_region_empty, 14	Alpidor logioni loddor, 20
s_serial_data_output, 14	BOOST_AUTO_TEST_CASE
s_system_clk_in, 14	pixel_col_test.cpp, 126
s_top_readout_fifo, 14	pixel_matrix_test.cpp, 127
s_total_number_of_hits, 14	BOOST_TEST_MODULE
s_tru_fifo_size, 15	pixel_col_test.cpp, 126
alpide.cpp	pixel_matrix_test.cpp, 127
SC_HAS_PROCESS, 89	p.merau.m_teetiepp;
alpide constants.h	CHIP_EMPTY_FRAME
CHIP_HEIGHT_CM, 92	TopReadoutUnit, 79
CHIP WIDTH CM, 92	CHIP HEADER
DATA_LONG_PIXMAP_SIZE, 92	TopReadoutUnit, 79
LHC_ORBIT_BUNCH_COUNT, 92	CHIP_HEIGHT_CM
N_PIXEL_COLS_PER_REGION, 92	alpide_constants.h, 92
N PIXEL COLS, 92	CHIP TRAILER
N_PIXEL_DOUBLE_COLS_PER_REGION, 93	TopReadoutUnit, 79
N PIXEL ROWS, 93	CHIP_WIDTH_CM
N PIXELS PER REGION, 93	alpide constants.h, 92
N_REGIONS, 93	calculateAverageCrossingRate
alpide data format.h	EventGenerator, 30
DW BUSY OFF, 95	clock
DW_BUSY_ON, 95	Stimuli, 76
DW_CHIP_EMPTY_FRAME, 95	compile_instructions.txt
DW_CHIP_HEADER, 95	yet, 96
DW CHIP TRAILER, 95	create output dir

main.cpp, 121	settings.h, 119
DATA LONG BIYMAD CIZE	DEFAULT_SIMULATION_N_EVENTS
DATA_LONG_PIXMAP_SIZE	settings.h, 119
alpide_constants.h, 92	DEFAULT_SIMULATION_RANDOM_SEED
DEFAULT_ALPIDE_CLUSTERING_ENABLE	settings.h, 119
settings.h, 117	DW_BUSY_OFF
DEFAULT_ALPIDE_MATRIX_READOUT_PERIOD_←	alpide_data_format.h, 95
NS	DW_BUSY_ON
settings.h, 117	alpide_data_format.h, 95
DEFAULT_ALPIDE_PIXEL_SHAPING_ACTIVE_TIM←	DW_CHIP_EMPTY_FRAME
E_NS	alpide_data_format.h, 95
settings.h, 117	DW_CHIP_HEADER
DEFAULT_ALPIDE_PIXEL_SHAPING_DEAD_TIME↔	alpide_data_format.h, 95
_NS	DW_CHIP_TRAILER
settings.h, 117	alpide_data_format.h, 95
DEFAULT_ALPIDE_REGION_FIFO_SIZE	DW_DATA_LONG
settings.h, 117	alpide_data_format.h, 95
DEFAULT_ALPIDE_REGION_SIZE	DW DATA SHORT
settings.h, 117	alpide_data_format.h, 95
DEFAULT_ALPIDE_TRU_FIFO_SIZE	DW IDLE
settings.h, 117	alpide_data_format.h, 96
DEFAULT_DATA_OUTPUT_WRITE_EVENT_CSV	DW_REGION_HEADER
settings.h, 117	alpide_data_format.h, 96
DEFAULT_DATA_OUTPUT_WRITE_VCD_CLOCK	data
settings.h, 117	AlpideDataWord, 23
DEFAULT_DATA_OUTPUT_WRITE_VCD	dataTransmission
settings.h, 117	Alpide, 12
DEFAULT_EVENT_AVERAGE_EVENT_RATE_NS	disableWriteToDisk
settings.h, 117	EventGenerator, 30
DEFAULT_EVENT_BUNCH_CROSSING_RATE_NS	Evenidenerator, 30
settings.h, 118	E_trigger_event_available
DEFAULT_EVENT_HIT_DENSITY_MIN_BIAS_PER	EventGenerator, 39
	Stimuli, 76
settings.h, 118	enableWriteToDisk
DEFAULT_EVENT_HIT_MULTIPLICITY_DISTRIBU←	EventGenerator, 30
TION FILE	event generator.cpp
settings.h, 118	print_function_timestamp, 106
DEFAULT EVENT HIT MULTIPLICITY DISTRIBU	SC_HAS_PROCESS, 106
TION TYPE	event_generator.h
settings.h, 118	N_CHIPS, 107
DEFAULT_EVENT_HIT_MULTIPLICITY_GAUSS_AVG	EventGenerator, 26
settings.h, 118	\sim EventGenerator, 29
DEFAULT_EVENT_HIT_MULTIPLICITY_GAUSS_S↔	addHitsToTriggerEvent, 29
TDDEV	calculateAverageCrossingRate, 30
settings.h, 118	disableWriteToDisk, 30
DEFAULT_EVENT_STROBE_ACTIVE_LENGTH_NS	E_trigger_event_available, 39
settings.h, 118	enableWriteToDisk, 30
DEFAULT EVENT STROBE INACTIVE LENGTH ←	EventGenerator, 29
NS	eventMemoryCountLimiter, 30
settings.h, 118	generateNextEvent, 31
DEFAULT_EVENT_TRIGGER_DELAY_NS	generateNextEvents, 31
settings.h, 118	generateNextPhysicsEvent, 31
DEFAULT_EVENT_TRIGGER_FILTER_ENABLE	generateNextTriggerEvent, 31
	getEventsInMem, 32
settings.h, 118	
DEFAULT_EVENT_TRIGGER_FILTER_TIME_NS	getNextTriggerEvent, 32
settings.h, 119	getPhysicsEventCount, 33
DEFAULT_SIMULATION_CONTINUOUS_MODE	getRandomMultiplicity, 33
settings.h, 119	getTriggerEventCount, 33
DEFAULT_SIMULATION_N_CHIPS	getTriggerFilterTime, 34

initRandomNumGenerator, 34	EventGenerator, 31
mAverageEventRateNs, 39	generateNextPhysicsEvent
mBunchCrossingRateNs, 39	EventGenerator, 31
mContinuousMode, 39	generateNextTriggerEvent
mCreateCSVFile, 39	EventGenerator, 31
mDataPath, 39	getActiveTimeEnd
mEventQueue, 39	Hit, 46
mHitMultiplicityGaussAverage, 39	getActiveTimeStart
mHitMultiplicityGaussDeviation, 40	Hit, 46
mHitQueue, 40	getChipId
mLastPhysicsEventTimeNs, 40	Alpide, 12
mLastTriggerEventEndTimeNs, 40	TriggerEvent, 85
mLastTriggerEventStartTimeNs, 40	getCol
mNextTriggerEventChipId, 40	PixelData, 50
mNextTriggerEventStartTimeNs, 40	getEventEndTime
mNumChips, 40	TriggerEvent, 85
mNumEventsInMemoryAllowed, 41	getEventFilteredFlag
mOutputPath, 41	TriggerEvent, 85
mPhysicsEventCount, 41	getEventId
mPhysicsEventsCSVFile, 41	TriggerEvent, 85
mPixelActiveTime, 41	getEventSize
mPixelDeadTime, 41	TriggerEvent, 86
mRandEventTime, 41	getEventStartTime
mRandEventTimeGen, 41	TriggerEvent, 86
mRandHitChipID, 41	getEventsInMem
mRandHitChipX, 42	EventGenerator, 32
mRandHitChipY, 42	getHitTotalAllEvents
mRandHitGen, 42	PixelMatrix, 60
mRandHitMultiplicityDiscrete, 42	getHitsRemainingInOldestEvent
mRandHitMultiplicityGauss, 42	PixelMatrix, 60
mRandHitMultiplicityGen, 42	getMEBHisto
mRandomSeed, 42	PixelMatrix, 60
mTriggerEventIdCount, 42	getNextTriggerEvent
mTriggerEventsCSVFile, 42	EventGenerator, 32
mTriggerFilterTimeNs, 43	getNumEvents
mTriggerFilteringEnabled, 43	PixelMatrix, 61
mWriteEventsToDisk, 43	getPhysicsEventCount
physicsEventProcess, 34	EventGenerator, 33
readDiscreteDistributionFile, 35	getPriEncNumInRegion
removeInactiveHits, 36	PixelData, 50
removeOldestEvent, 36	getPriEncPixelAddress
s_clk_in, 43	PixelData, 50
s_physics_event_out, 43	getRandomMultiplicity
s_strobe_in, 43	EventGenerator, 33
scaleDiscreteDistribution, 37	getRow
setBunchCrossingRate, 37	PixelData, 51
setNumEventsInMemAllowed, 37	getSimSettings
setPath, 38	settings.cpp, 113
setRandomSeed, 38	settings.h, 119
triggerEventProcess, 38	getTriggerEventCount
eventMemoryCountLimiter	EventGenerator, 33
EventGenerator, 30	getTriggerEventsAcceptedCount
(HE T OL:	PixelMatrix, 61
feedHitsToChip	getTriggerEventsRejectedCount
TriggerEvent, 84	PixelMatrix, 61
generateNextEvent	getTriggerFilterTime
EventGenerator, 31	EventGenerator, 34
generateNextEvents	Hit, 44
generalet vertes	rm, 44

getActiveTimeEnd, 46	mDataPath
getActiveTimeStart, 46	EventGenerator, 39
Hit, 45	mEnableReadoutTraces
isActive, 46	Alpide, 13
mActiveTimeEndNs, 47	mEventEndTimeNs
mActiveTimeStartNs, 47	TriggerEvent, 87
operator<, 46	mEventFilteredFlag
operator<=, 46	TriggerEvent, 87
operator>, 47	mEventId
operator>=, 47	TriggerEvent, 87
operator=, 46	mEventQueue
operator==, 47	EventGenerator, 39
oporator—, m	mEventStartTimeNs
IDLE	
TopReadoutUnit, 79	TriggerEvent, 87
initRandomNumGenerator	mEvents
EventGenerator, 34	Stimuli, 77
inspectPixel	mFifoSizeLimit
PixelDoubleColumn, 55	RegionReadoutUnit, 70
isActive	mFifoSizeLimitEnabled
Hit, 46	RegionReadoutUnit, 70
Till, 40	mHitMultiplicityGaussAverage
LHC_ORBIT_BUNCH_COUNT	EventGenerator, 39
alpide_constants.h, 92	mHitMultiplicityGaussDeviation
aipide_constants.n, 92	EventGenerator, 40
mActiveTimeEndNs	mHitQueue
Hit, 47	EventGenerator, 40
mActiveTimeStartNs	mHitSet
Hit, 47	TriggerEvent, 88
mAlpideChips	mLastPhysicsEventTimeNs
·	EventGenerator, 40
Stimuli, 76	mLastTriggerEventEndTimeNs
mAverageEventRateNs	EventGenerator, 40
EventGenerator, 39	mLastTriggerEventStartTimeNs
mBunchCounter	EventGenerator, 40
TopReadoutUnit, 81	mMEBHistoLastUpdateTime
mBunchCrossingRateNs	PixelMatrix, 65
EventGenerator, 39	
mBusySignaled	mMEBHistogram
RegionReadoutUnit, 70	PixelMatrix, 65
mChipId	mNextTriggerEventChipId
Alpide, 13	EventGenerator, 40
TopReadoutUnit, 81	mNextTriggerEventStartTimeNs
TriggerEvent, 87	EventGenerator, 40
mClusterStarted	mNumChips
RegionReadoutUnit, 70	EventGenerator, 40
mClusteringEnabled	Stimuli, 77
RegionReadoutUnit, 70	mNumEvents
mCol	Stimuli, 77
PixelData, 54	mNumEventsInMemoryAllowed
mColumnBuffs	EventGenerator, 41
PixelMatrix, 65	mOutputPath
mColumnBuffsPixelsLeft	EventGenerator, 41
PixelMatrix, 65	Stimuli, 77
mContinuousMode	mPhysicsEventCount
EventGenerator, 39	EventGenerator, 41
PixelMatrix, 65	mPhysicsEventsCSVFile
Stimuli, 76	EventGenerator, 41
mCreateCSVFile	mPixelActiveTime
EventGenerator, 39	EventGenerator, 41
Eventaenerator, 33	Evenidencialor, 41

mPixelDeadTime	main.cpp
EventGenerator, 41	create_output_dir, 121
mPixelHitBaseAddr	sc_main, 122
RegionReadoutUnit, 70	matrix_readout_clock
mPixelHitEncoderId	Stimuli, 76
RegionReadoutUnit, 70	matrixReadout
mPixelHitmap	Alpide, 12
RegionReadoutUnit, 70	•
mRRUs	N_CHIPS
Alpide, 13	event_generator.h, 107
mRandEventTime	N_PIXEL_COLS_PER_REGION
EventGenerator, 41	alpide_constants.h, 92
mRandEventTimeGen	N_PIXEL_COLS
EventGenerator, 41	alpide_constants.h, 92
mRandHitChipID	N_PIXEL_DOUBLE_COLS_PER_REGION
EventGenerator, 41	alpide_constants.h, 93
mRandHitChipX	N_PIXEL_ROWS
EventGenerator, 42	alpide_constants.h, 93
mRandHitChipY	N_PIXELS_PER_REGION
EventGenerator, 42	alpide_constants.h, 93
mRandHitGen	N REGIONS
EventGenerator, 42	alpide_constants.h, 93
•	newEvent
mRandHitMultiplicityDiscrete	PixelMatrix, 61
EventGenerator, 42	NoPixelHit
mRandHitMultiplicityGauss	pixel_col.h, 98
EventGenerator, 42	NoTriggerEvent
mRandHitMultiplicityGen	trigger_event.cpp, 111
EventGenerator, 42	trigger_event.h, 112
mRandomSeed	<u>9</u> 900,
EventGenerator, 42	operator<
mRegionId	Hit, 46
RegionReadoutUnit, 71	PixelData, 51
mRow	operator<<
PixelData, 54	alpide_data_format.h, 95
mSettings	operator<=
Stimuli, 77	Hit, 46
mStrobeActiveNs	PixelData, 51
Stimuli, 77	operator>
mStrobelnactiveNs	Hit, 47
Stimuli, 77	PixelData, 52
mTRU	operator>=
Alpide, 13	Hit, 47
mTriggerDelayNs	PixelData, 52
Stimuli, 77	operator()
mTriggerEventIdCount	PixelPriorityEncoder, 67
EventGenerator, 42	operator=
mTriggerEventsAccepted	Hit, 46
PixelMatrix, 65	
mTriggerEventsCSVFile	operator==
EventGenerator, 42	Hit, 47
mTriggerEventsRejected	PixelData, 52
PixelMatrix, 66	physicsEventProcess
mTriggerFilterTimeNs	• •
	EventGenerator, 34
EventGenerator, 43	pixel_col.h
mTriggerFilteringEnabled	NoPixelHit, 98
EventGenerator, 43	pixel_col_test.cpp
mWriteEventsToDisk	BOOST_AUTO_TEST_CASE, 126
EventGenerator, 43	BOOST_TEST_MODULE, 126

pixel_matrix_test.cpp	TopReadoutUnit, 79
BOOST_AUTO_TEST_CASE, 127	readDiscreteDistributionFile
BOOST_TEST_MODULE, 127	EventGenerator, 35
pixelColumn	readPixel
PixelDoubleColumn, 58	PixelDoubleColumn, 56
PixelData, 48	PixelMatrix, 62
getCol, 50	readPixelRegion
getPriEncNumInRegion, 50	PixelMatrix, 63
getPriEncPixelAddress, 50	readoutNextPixel
getRow, 51	RegionReadoutUnit, 69
mCol, 54	RegionReadoutUnit, 67
mRow, 54	addTraces, 69
operator<, 51	mBusySignaled, 70
operator<=, 51 operator>, 52	mClusterStarted, 70
operator>, 52 operator>=, 52	mClusteringEnabled, 70 mFifoSizeLimit, 70
operator==, 52	mFifoSizeLimitEnabled, 70
PixelData, 49	mPixelHitBaseAddr, 70
PixelPriorityEncoder, 54	mPixelHitEncoderId, 70
setCol, 53	mPixelHitmap, 70
setRow, 53	mRegionId, 71
PixelDoubleColumn, 54	readoutNextPixel, 69
inspectPixel, 55	RegionReadoutUnit, 69
pixelColumn, 58	s_busy_out, 71
pixelHitsRemaining, 56	s_region_empty_out, 71
readPixel, 56	s_region_fifo, 71
setPixel, 57	s_region_fifo_out, 71
pixelHitsRemaining	s_region_fifo_size, 71
PixelDoubleColumn, 56	removeInactiveHits
PixelMatrix, 58	EventGenerator, 36
getHitTotalAllEvents, 60	removeOldestEvent
getHitsRemainingInOldestEvent, 60	EventGenerator, 36
getMEBHisto, 60	
getNumEvents, 61	s_busy_out
getTriggerEventsAcceptedCount, 61	RegionReadoutUnit, 71
getTriggerEventsRejectedCount, 61	s_clk_in
mColumnBuffs, 65	EventGenerator, 43
mColumnBuffsPixelsLeft, 65	TopReadoutUnit, 81
mContinuousMode, 65	s_current_event_hits_left_in
mMEBHistoLastUpdateTime, 65	TopReadoutUnit, 81
mMEBHistogram, 65	s_current_region
mTriggerEventsAccepted, 65	TopReadoutUnit, 81
mTriggerEventsRejected, 66	s_event_buffers_used Alpide, 14
newEvent, 61	s_event_buffers_used_in
PixelMatrix, 59	TopReadoutUnit, 81
readPixel, 62	s matrix readout clk in
readPixelRegion, 63	Alpide, 14
setPixel, 64	s_oldest_event_number_of_hits
PixelPriorityEncoder, 66	Alpide, 14
operator(), 67	s_physics_event
PixelData, 54	Stimuli, 77
print_event_rate	s_physics_event_out
stimuli.cpp, 123	EventGenerator, 43
print_function_timestamp	s_region_empty
event_generator.cpp, 106	Alpide, 14
REGION DATA	s_region_empty_in
TopReadoutUnit, 79	TopReadoutUnit, 81
REGION_HEADER	s_region_empty_out
_	_ 3 1-7

RegionReadoutUnit, 71	setDefaultSimSettings, 114
s_region_fifo	settings.h
RegionReadoutUnit, 71	DEFAULT_ALPIDE_CLUSTERING_ENABLE, 117
s_region_fifo_in	DEFAULT_ALPIDE_MATRIX_READOUT_PERI↔
TopReadoutUnit, 81	OD_NS, 117
s_region_fifo_out	DEFAULT_ALPIDE_PIXEL_SHAPING_ACTIVE↔
RegionReadoutUnit, 71	_TIME_NS, 117
s_region_fifo_size	DEFAULT_ALPIDE_PIXEL_SHAPING_DEAD_↔
RegionReadoutUnit, 71	TIME_NS, 117
s_serial_data_output	DEFAULT_ALPIDE_REGION_FIFO_SIZE, 117
Alpide, 14	DEFAULT ALPIDE REGION SIZE, 117
•	DEFAULT_ALPIDE_TRU_FIFO_SIZE, 117
s_strobe Stimuli, 77	
	DEFAULT_DATA_OUTPUT_WRITE_EVENT_C↔
s_strobe_in	SV, 117
EventGenerator, 43	DEFAULT_DATA_OUTPUT_WRITE_VCD_CLO↔
s_system_clk_in	CK, 117
Alpide, 14	DEFAULT_DATA_OUTPUT_WRITE_VCD, 117
s_top_readout_fifo	DEFAULT_EVENT_AVERAGE_EVENT_RATE_←
Alpide, 14	NS, 117
s_total_number_of_hits	DEFAULT_EVENT_BUNCH_CROSSING_RAT↔
Alpide, 14	E_NS, 118
s_tru_fifo_out	DEFAULT_EVENT_HIT_DENSITY_MIN_BIAS_←
TopReadoutUnit, 82	PER_CM2, 118
s_tru_fifo_size	DEFAULT_EVENT_HIT_MULTIPLICITY_DISTR↔
Alpide, 15	IBUTION_FILE, 118
s_tru_state	DEFAULT_EVENT_HIT_MULTIPLICITY_DISTR↔
TopReadoutUnit, 82	IBUTION_TYPE, 118
SC_HAS_PROCESS	DEFAULT_EVENT_HIT_MULTIPLICITY_GAUS↔
alpide.cpp, 89	S_AVG, 118
event_generator.cpp, 106	DEFAULT_EVENT_HIT_MULTIPLICITY_GAUS↔
stimuli.cpp, 123	S_STDDEV, 118
top_readout.cpp, 104	DEFAULT_EVENT_STROBE_ACTIVE_LENGT←
sc_main	H_NS, 118
main.cpp, 122	DEFAULT_EVENT_STROBE_INACTIVE_LEN←
scaleDiscreteDistribution	GTH_NS, 118
EventGenerator, 37	DEFAULT_EVENT_TRIGGER_DELAY_NS, 118
setBunchCrossingRate	DEFAULT_EVENT_TRIGGER_FILTER_ENABLE,
EventGenerator, 37	118
setCol	DEFAULT_EVENT_TRIGGER_FILTER_TIME_←
PixelData, 53	NS, 119
setDefaultSimSettings	DEFAULT_SIMULATION_CONTINUOUS_MODE,
settings.cpp, 114	119
settings.h, 120	DEFAULT_SIMULATION_N_CHIPS, 119
setEventFilteredFlag	DEFAULT_SIMULATION_N_EVENTS, 119
TriggerEvent, 86	DEFAULT_SIMULATION_RANDOM_SEED, 119
setNumEventsInMemAllowed	getSimSettings, 119
EventGenerator, 37	setDefaultSimSettings, 120
setPath	simulation_done
EventGenerator, 38	Stimuli, 78
setPixel	src/alpide/alpide.cpp, 89
PixelDoubleColumn, 57	src/alpide/alpide.h, 90
PixelMatrix, 64	src/alpide/alpide_constants.h, 91
setRandomSeed	src/alpide/alpide_data_format.h, 93 src/alpide/compile_instructions.txt, 96
EventGenerator, 38	• -
setRow PivelDate F0	src/alpide/pixel_col.cpp, 96
PixelData, 53	src/alpide/pixel_col.h, 97
settings.cpp	src/alpide/pixel_matrix.cpp, 99
getSimSettings, 113	src/alpide/pixel_matrix.h, 99

src/alpide/region_readout.cpp, 101	IDLE, 79
src/alpide/region_readout.h, 101	mBunchCounter, 81
src/alpide/top_readout.cpp, 103	mChipId, 81
src/alpide/top_readout.h, 104	REGION_DATA, 79
src/event/event_cpp.d, 105	REGION HEADER, 79
	s_clk_in, 81
src/event/event_generator.cpp, 105	
src/event/event_generator.h, 106	s_current_event_hits_left_in, 81
src/event/event_generator_cpp.d, 107	s_current_region, 81
src/event/hit.cpp, 108	s_event_buffers_used_in, 81
src/event/hit.h, 108	s_region_empty_in, 81
src/event/hit_cpp.d, 110	s_region_fifo_in, 81
src/event/trigger_event.cpp, 110	s_tru_fifo_out, 82
src/event/trigger_event.h, 111	s_tru_state, 82
src/settings/settings.cpp, 113	TRU_state_t, 79
src/settings/settings.h, 115	TopReadoutUnit, 80
src/testbench/main.cpp, 120	topRegionReadoutProcess, 80
src/testbench/stimuli.cpp, 122	topRegionReadoutProcess
src/testbench/stimuli.h, 124	TopReadoutUnit, 80
src/unit_tests/pixel_col_test.cpp, 125	trigger event.cpp
src/unit_tests/pixel_matrix_test.cpp, 126	NoTriggerEvent, 111
Stimuli, 72	trigger_event.h
addTraces, 74	NoTriggerEvent, 112
clock, 76	TriggerEvent, 82
E_trigger_event_available, 76	addHit, 84
mAlpideChips, 76	feedHitsToChip, 84
mContinuousMode, 76	getChipId, 85
mEvents, 77	getEventEndTime, 85
mNumChips, 77	getEventFilteredFlag, 85
mNumEvents, 77	getEventId, 85
mOutputPath, 77	getEventSize, 86
mSettings, 77	getEventStartTime, 86
mStrobeActiveNs, 77	mChipId, 87
mStrobeInactiveNs, 77	mEventEndTimeNs, 87
mTriggerDelayNs, 77	mEventFilteredFlag, 87
matrix_readout_clock, 76	mEventId, 87
s physics event, 77	mEventStartTimeNs, 87
s_strobe, 77	mHitSet, 88
simulation_done, 78	setEventFilteredFlag, 86
Stimuli, 73	TriggerEvent, 83
stimuliEventProcess, 74	writeToFile, 86
stimuliMainProcess, 75	triggerEventProcess
writeDataToFile, 76	EventGenerator, 38
stimuli.cpp	versita Data Ta Fila
print_event_rate, 123	writeDataToFile
SC_HAS_PROCESS, 123	Stimuli, 76
stimuliEventProcess	writeToFile
Stimuli, 74	TriggerEvent, 86
stimuliMainProcess	
Stimuli, 75	yet
	compile_instructions.txt, 96
TRU_state_t	
TopReadoutUnit, 79	
top_readout.cpp	
SC_HAS_PROCESS, 104	
TopReadoutUnit, 78	
addTraces, 80	
CHIP_EMPTY_FRAME, 79	
CHIP_HEADER, 79	
CHIP TRAILER, 79	
SIN TIVNELLE 10	