Contents

1 Punnets Main Page ................................. 1

2 Punnets Module Index .............................. 3
  2.1 Punnets Modules ................................. 3

3 Punnets Namespace Index .......................... 5
  3.1 Punnets Namespace List .......................... 5

4 Punnets Hierarchical Index ........................ 7
  4.1 Punnets Class Hierarchy ........................ 7

5 Punnets Compound Index ........................... 9
  5.1 Punnets Compound List ........................... 9

6 Punnets File Index ................................. 11
  6.1 Punnets File List ................................. 11

7 Punnets Page Index ................................. 13
  7.1 Punnets Related Pages ............................ 13

8 Punnets Module Documentation ..................... 15
  8.1 Logging (drawing a graph of neuron potentials) ..................... 15
  8.2 Variables for Statistics .......................... 16
  8.3 Neurons ............................................. 18
  8.4 Synapses ............................................. 19
  8.5 Scheduling .......................................... 20

9 Punnets Namespace Documentation .................. 21
  9.1 punnets Namespace Reference .................... 21
  9.2 punnets_common Namespace Reference ............... 23
  9.3 punnets_nodebug Namespace Reference ............... 29
CONTENTS

10.35 punnets private::tsynapse addfunc< debug > Class Template Reference 92
10.36 punnets common::tsynapse base Class Reference 94
10.37 punnets private::tsynapse fatigue< debug > Class Template Reference 96
10.38 punnets private::tsynapse message< debug > Class Template Reference 98
10.39 punnets private::tsynapse messfunc< debug > Class Template Reference 100

11 Punnets File Documentation 103
11.1 dlanguage.cpp File Reference 103
11.2 dlogger.cpp File Reference 105
11.3 dlogger.h File Reference 106
11.4 dneuron.cpp File Reference 107
11.5 dneuron.h File Reference 108
11.6 dsched.cpp File Reference 109
11.7 dsched.h File Reference 110
11.8 dttest.cpp File Reference 111
11.9 func.cpp File Reference 112
11.10 func.h File Reference 113
11.11 punnets.h File Reference 114
11.12 punnets_base.h File Reference 115

12 Punnets Page Documentation 117
12.1 Todo List 117
Chapter 1

Punnets Main Page

Punnets is a C++ library for Pulsed Neural Network Simulator. Based on its event-discrete (event-driven) manner of simulation, Punnets provides highly accurate and efficient simulation environment for rather complex neuron models. Please take a look at the sample programs, dtest.cpp and dlanguage.cpp, to see how to use the library.

Punnets provides two user-accessible namespaces, punnets and punnets_nodebug (p. 29). A user program may import one of the two namespaces to use the library. Although the punnets_nodebug (p. 29) version is more efficient in simulation, punnets version can provide debugging facility, which can be turned on for neuron-by-neuron basis. Stuffs in punnets_common (p. 23) namespace can be accessed via either namespace, punnets or punnets_nodebug (p. 29).

You can obtain the reference manual of this library by using doxygen, a document generation tool. If you have installed doxygen, it will be automatically generated and installed via a standard install process. You can also see the reference manual via the Punnets home page: http://www.snowelm.com/~t/research/software/punnets/
Chapter 2

Punnets Module Index

2.1 Punnets Modules

Here is a list of all modules:

- Logging (drawing a graph of neuron potentials) ............................................. 15
- Variables for Statistics .................................................................................. 16
- Neurons ........................................................................................................ 18
- Synapses ....................................................................................................... 19
- Scheduling .................................................................................................... 20
Chapter 3

Punnets Namespace Index

3.1 Punnets Namespace List

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

- punnets ......................................................... 21
- punnets\_common (This namespace provides commonly available classes. Users should use classes and functions in this namespace via either punnets or punnets\_nodebug (p. 29) namespace) ............... 23
- punnets\_nodebug ............................................. 29
- punnets\_private (This namespace provides some private classes used for punnets library. Users should not use classes and functions in this namespace directly) . 30
Chapter 4

Punnets Hierarchical Index

4.1 Punnets Class Hierarchy

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

- punnets_common::debugflag< false > .................................................. 31
- punnets_common::debugflag< true > .................................................. 32
- punnets_common::func_base .............................................................. 33
- punnets_common::func_const ............................................................. 36
- punnets_common::func_deriveq_base .................................................. 43
- punnets_common::func_const_int ......................................................... 38
- punnets_common::func_delta_int ......................................................... 40
- punnets_common::func_exp_int ........................................................... 51
- punnets_common::func_response ......................................................... 53
- punnets_common::func_sine_int ......................................................... 57
- punnets_common::func_sineshot_int ..................................................... 63
- punnets_common::func_exp .............................................................. 46
- punnets_common::func_exp_diff ......................................................... 48
- punnets_common::func_sine ............................................................. 55
- punnets_common::func_sineshot ......................................................... 59
- punnets_common::func_step ............................................................ 65
- punnets_common::greater_event ......................................................... 67
- punnets_common::message_base ......................................................... 68
- punnets_common::func_delta_int::message_add_pulse .............................. 42
- punnets_common::func_deriveq_base::message_set_lambda ....................... 44
- punnets_common::func_deriveq_base::message_set_zero_point ................... 45
- punnets_common::func_exp_diff::message_add_event_time ....................... 50
- punnets_common::func_sineshot::message_set_t0 .................................. 62

neuinfo

- punnets_common::taction .............................................................. 69
- punnets_common::tlogger .............................................................. 72
- punnets_common::tmessage
- punnets_common::tpulse
- punnets_common::tsentinel ............................................................ 89
- punnets_common::tsynapse_base ....................................................... 94
- punnets_private::tsynapse< debug > ................................................ 90
- punnets_private::tsynapse_addfunc< debug > ..................................... 92
<table>
<thead>
<tr>
<th>Class/Function</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>punnets::private::tneuron&lt; debug &gt;</td>
<td>Private member function</td>
<td>76</td>
</tr>
<tr>
<td>punnets::private::tneuron_ext&lt; debug &gt;</td>
<td>Private member function</td>
<td>81</td>
</tr>
<tr>
<td>punnets::private::valdomain</td>
<td>Private member function</td>
<td>88</td>
</tr>
<tr>
<td>punnets::common::tevent</td>
<td></td>
<td>71</td>
</tr>
<tr>
<td>punnets::common::tneuron_base</td>
<td></td>
<td>79</td>
</tr>
<tr>
<td>punnets::private::tneuron&lt; debug &gt;</td>
<td>Private member function</td>
<td>76</td>
</tr>
<tr>
<td>punnets::private::tneuron_ext&lt; debug &gt;</td>
<td>Private member function</td>
<td>81</td>
</tr>
<tr>
<td>tobserver</td>
<td></td>
<td>87</td>
</tr>
<tr>
<td>punnets::common::tsched_double</td>
<td></td>
<td>88</td>
</tr>
</tbody>
</table>
Chapter 5

Punnets Compound Index

5.1 Punnets Compound List

Here are the classes, structs, unions and interfaces with brief descriptions:

- punnets_common::debugflag< false > ............................... 31
- punnets_common::debugflag< true > (This instantiation of the class is used for punnets namespace) .............................. 32
- punnets_common::func_base ........................................... 33
- punnets_common::func_const (</classdef>) .......................... 36
- punnets_common::func_const_int ..................................... 38
- punnets_common::func_delta_int (</classdef>) ..................... 40
- punnets_common::func_delta_int::message_add_pulse (A message that adds a new pulse) ........................................ 42
- punnets_common::func_derivativeq_base ............................... 43
- punnets_common::func_derivativeq_base::message_set_lambda (A message that changes lambda (leak value)) ............................ 44
- punnets_common::func_derivativeq_base::message_set_zero_point (A message that changes zero point) ............................. 45
- punnets_common::func_exp .............................................. 46
- punnets_common::func_exp_diff ......................................... 48
- punnets_common::func_exp_diff::message_add_event_time ............ 50
- punnets_common::func_exp_int ........................................... 51
- punnets_common::func_response ........................................ 53
- punnets_common::func_sine ............................................. 55
- punnets_common::func_sine_int ......................................... 57
- punnets_common::func_sineshot ......................................... 59
- punnets_common::func_sineshot::message_set_t0 (A message that changes t0) ........................................... 62
- punnets_common::func_sineshot_int .................................... 63
- punnets_common::func_step (</classdef>) ............................. 65
- punnets_common::greater_te event .................................... 67
- punnets_common::message_base ........................................ 68
- punnets_common::taction .............................................. 69
- punnets_common::te vent ............................................. 71
- punnets_common::tlogger ............................................. 72
- punnets_private::tneuron< debug > .................................... 76
- punnets_common::tneuron_base ......................................... 79
- punnets_private::tneuron_ext< debug > ................................ 81
<table>
<thead>
<tr>
<th>Class Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>punnets_private::tneuron_ext_const&lt; debug &gt;</td>
<td>84</td>
</tr>
<tr>
<td>toobserver</td>
<td>87</td>
</tr>
<tr>
<td>punnets_common::tsched_double</td>
<td>88</td>
</tr>
<tr>
<td>punnets_common::tsentinel</td>
<td>89</td>
</tr>
<tr>
<td>punnets_private::tsynapse&lt; debug &gt;</td>
<td>90</td>
</tr>
<tr>
<td>punnets_private::tsynapse_addfunc&lt; debug &gt;</td>
<td>92</td>
</tr>
<tr>
<td>punnets_private::tsynapse_base</td>
<td>94</td>
</tr>
<tr>
<td>punnets_private::tsynapse_fatigue&lt; debug &gt;</td>
<td>96</td>
</tr>
<tr>
<td>punnets_private::tsynapse_message&lt; debug &gt;</td>
<td>98</td>
</tr>
<tr>
<td>punnets_private::tsynapse_messfunc&lt; debug &gt;</td>
<td>100</td>
</tr>
</tbody>
</table>

(Synapse class that adds a new function to the destination (postsynaptic) \texttt{tneuron\_ext} (p. 81))
Chapter 6

Punnets File Index

6.1 Punnets File List

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

- **config_punnets.h** ................................................................. ??
- **dlanguage.cpp** (Language simulation based on the punnets library) .................. 103
- **dlogger.cpp** (Logger) ............................................................. 105
- **dlogger.h** (Activation logging class) .......................................... 106
- **dneuron.cpp** (Neurons) ............................................................ 107
- **dneuron.h** (Neuron/Synapse class in discrete-event NN simulation) ............... 108
- **dsched.cpp** (Event, Action, Scheduler) ....................................... 109
- **dsched.h** (Distributed scheduler) .............................................. 110
- **dtest.cpp** (Test program of the punnets library) ............................. 111
- **func.cpp** (Functions) ............................................................... 112
- **func.h** (Function representation) ............................................. 113
- **punnets.h** (Punnets header file) ............................................. 114
- **punnets_base.h** (Base class of Punnets) .................................... 115
Chapter 7

Punnets Page Index

7.1 Punnets Related Pages

Here is a list of all related documentation pages:

Todo List ................................................................. 117
Chapter 8

Punnets Module Documentation

8.1 Logging (drawing a graph of neuron potentials)

Compounds

- class tlogger
8.2 Variables for Statistics

Variables

- \texttt{u_int64_t totalfire} = 0
  
  This variable counts the total number of firing of the neurons.

- \texttt{u_int64_t totalpulse} = 0
  
  This variable counts the total pulse arrivals to any of the neurons.

- \texttt{u_int64_t totalpartition} = 0
  
  This variable counts the total number of partitions.

- \texttt{u_int64_t totalpartition\_nonewton [4]} = \{0,0,0,0\}

- \texttt{u_int64_t totalpartition\_newton [4]} = \{0,0,0,0\}

- \texttt{u_int64_t totalpeaksearch [3]} = \{0,0,0\}

- \texttt{u_int64_t totalpeakenclosing} = 0
  
  This variable counts the total number of peak enclosings. One peak search contains several enclosings.

- \texttt{u_int64_t totalrescheduled} = \{0\}
  
  This variable counts the total number of re-scheduling.

- \texttt{u_int64_t totalfiltered\_maxgrad} = 0
  
  This variable counts the total filtering caused by maximum gradient check.

- \texttt{u_int64_t totalfiltered\_incontinuity} = 0
  
  This variable counts the total filtering caused by nearby incontinuity.

- \texttt{u_int64_t totalfiltered\_nextpulse} = 0
  
  This variable counts the total filtering caused by nearby next pulse.

8.2.1 Variable Documentation

8.2.1.1 \texttt{u_int64_t punnets\_common::totalpartition\_newton} = \{0,0,0,0\}

This variable counts the number of partitions with Newton-Raphson search. Each element contains the number of 0th, 1st, 2nd. and delta partitions, respectively.

Definition at line 42 of file dneuron.cpp.

Referenced by punnets\_private::tneuron\_ext\textless \ debug \textgreater ::scheduleFire().

8.2.1.2 \texttt{u_int64_t punnets\_common::totalpartition\_nonewton} = \{0,0,0,0\}

This variable counts the number of partitions without Newton-Raphson search. Each element contains the number of 0th, 1st, 2nd. and delta partitions, respectively.

Definition at line 41 of file dneuron.cpp.

Referenced by punnets\_private::tneuron\_ext\textless \ debug \textgreater ::scheduleFire().
8.2.1.3  \texttt{\_int64\_t punnets\_common::totalpeaksearch = \{0,0,0\}}

This variable counts the number of peak searches. Each element contains the number of searches with peak below threshold (no crossing), peak above threshold (crossing), no peak (not convex), respectively.

Definition at line 43 of file dneuron.cpp.

Referenced by punnets\_private::tneuron\_ext< debug >::scheduleFire().
8.3 Neurons

Compounds

- `class tneuron`
- `class tneuron_base`
- `class tneuron_ext`
- `class tneuron_ext_const`
8.4 Synapses

Compounds

- class tsynapse
- class tsynapse_addfunc

  *Synapse class that adds a new function to the destination (postsynaptic) neuron_ext (p. 81).*

- class tsynapse_base
- class tsynapse_fatigue
- class tsynapse_message
- class tsynapse_messfunc
8.5 Scheduling

Compounds

- struct greater_tevent
- class taction
- class tevent
- class sched_double
Chapter 9

Punnets Namespace Documentation

9.1 punnets Namespace Reference

Typedefs

- typedef punnets::tsynapse< true > tsynapse
  a typedef referring punnets::tsynapse (p. 90) class.

- typedef punnets::tsynapse_message< true > tsynapse_message
  a typedef referring punnets::tsynapse_message (p. 98) class.

- typedef punnets::tsynapse_fatigue< true > tsynapse_fatigue
  a typedef referring punnets::tsynapse_fatigue (p. 96) class.

- typedef punnets::tsynapse_addfunc< true > tsynapse_addfunc
  a typedef referring punnets::tsynapse_addfunc (p. 92) class.

- typedef punnets::tsynapse_messfunc< true > tsynapse_messfunc
  a typedef referring punnets::tsynapse_messfunc (p. 100) class.

- typedef punnets::tneuron< true > tneuron
  a typedef referring punnets::tneuron (p. 76) class.

- typedef punnets::tneuron_ext< const< true > tneuron_ext const
  a typedef referring punnets::tneuron_ext< const (p. 84) class.

- typedef punnets::tneuron_ext< true > tneuron_ext
  a typedef referring punnets::tneuron_ext (p. 81) class.
9.1.1 Detailed Description

This namespace provides Punnets library. Importing this namespace, a user can access the classes of Punnets library. Alternatively, if you need efficiency, you may import `punnets_nodebug` (p. 29) namespace. For most of the available classes, refer to the `punnets_common` (p. 23) namespace.

This namespace imports `punnets_common` (p. 23) so that every definitions in the `punnets_common` (p. 23) namespace can be transparently used in this namespace.
9.2 punnets_common Namespace Reference

This namespace provides commonly available classes. Users should use classes and functions in this namespace via either punnets or punnets_nodebug (p. 29) namespace.

Compounds

- class debugflag<false>
- class debugflag<true>

  *This instantiation of the class is used for punnets namespace.*

- class func_base
- class func_const

  </classdef>

- class func_const_int
- class func_delta_int

  </classdef>

- class func_deriveq_base
- class func_exp
- class func_exp_diff
- class func_exp_int
- class func_response
- class func_sine
- class func_sine_int
- class func_sineshot
- class func_sineshot_int
- class func_step

  </classdef>

- struct greater_tevent
- class message_add_pulse

  *A message that adds a new pulse.*

- class message_add_event_time
- class message_base
- class message_set_lambda

  *A message that changes lambda (leak value).*

- class message_set_zero_point

  *A message that changes zero point.*

- class message_set_t0

  *A message that changes t0.*

- struct less_sched_entry

  *This class is used to construct a global priority queue on STL.*

- class taction
- class tevent
- struct neuentry
  
  A class to keep a list of logging neurons.

- class tlogger
- class tmessage
- class tneuron_base
- class tpulse
- class tsched_double
- class tsentinel
- class tsynapse_base

**Typedefs**

- typedef double real
- typedef double ntime_t

**Functions**

- taction & makePulse (tneuron_base &idest, real ilevel)
  
  Generate a pulse to the specified destination.

- taction & makePulse (tneuron_base &idest, message_base *mess)
  
  Generate a pulse to the specified destination.

- template<bool b> taction & setExtInput (punnets_private::tneuron_ext const<b &idest, real ilevel)
  
  Set an external input to a specified level.

**Variables**

- u_int64_t totalfire = 0
  
  This variable counts the total number of firing of the neurons.

- u_int64_t totalpulse = 0
  
  This variable counts the total pulse arrivals to any of the neurons.

- u_int64_t totalpartition = 0
  
  This variable counts the total number of partitions.

- u_int64_t totalpartition_nonewton [4] = {0,0,0,0}
- u_int64_t totalpartition_newton [4] = {0,0,0,0}
- u_int64_t totalpeaksearch [3] = {0,0,0}
- u_int64_t totalpeakenclosing = 0
  
  This variable counts the total number of peak enclosings. One peak search contains several enclosings.

- u_int64_t totalrescheduled = {0}
9.2 punnets_common Namespace Reference

This variable counts the total number of re-scheduling.

- \texttt{u_int64\_t totalfiltered\_maxgrad = 0}
  
  This variable counts the total filtering caused by maximum gradient check.

- \texttt{u_int64\_t totalfiltered\_incontinuity = 0}
  
  This variable counts the total filtering caused by nearby incontinuity.

- \texttt{u_int64\_t totalfiltered\_nextpulse = 0}
  
  This variable counts the total filtering caused by nearby next pulse.

- \texttt{const real epsilon = 1e-10}
  
  This variable specifies the minimum error value.

9.2.1 Detailed Description

This namespace provides commonly available classes. Users should use classes and functions in this namespace via either punnets or \texttt{punnets\_nodebug} (p. 29) namespace.

9.2.2 Typedef Documentation

9.2.2.1 typedef double punnets_common::ntime\_t

Time representation type.

The type that represents a simulation time.

Definition at line 34 of file punnets_base.h.

Referenced by punnets_common::tsentinel::activate(), punnets_private::tsynapse::messfunc\<debug >::activate(), punnets_private::tsynapse::addfunc\<debug >::activate(), punnets_private::neuron\_ext\<debug >::activate(), punnets_private::neuron\<debug >::activate(), punnets_private::tsynapse\_fatigue\<debug >::activate(), punnets_private::tsynapse\_message\<debug >::activate(), punnets_common::tlogger::activate(), punnets_common::tlogger::add(), punnets_private::tsynapse\_fatigue\<debug >::addDelay(), punnets_private::tsynapse\<debug >::addDelay(), punnets_private::neuron\_ext\<debug >::broadcastMessage(), punnets_private::neuron\_ext\<debug >::calcSignal(), punnets_private::neuron\_ext\<debug >::fire(), punnets_common::func\_delta\_int::func\_delta\_int(), punnets_common::func\_exp\_diff::get1stDeriv(), punnets_common::func\_exp\_int::get1stDeriv(), punnets_common::func\_exp::get1stDeriv(), punnets_common::func\_sineshot\_int::get1stDeriv(), punnets_common::func\_sine\_int::get1stDeriv(), punnets_common::func\_response::get1stDeriv(), punnets_common::func\_delta\_int::get1stDeriv(), punnets_common::func\_sineshot\_int::get1stDeriv(), punnets_common::func\_sine\_int::get1stDeriv(), punnets_common::func\_response::get1stDeriv(), punnets_common::func\_delta\_int::get2ndDeriv(), punnets_common::func\_sineshot\_int::get2ndDeriv(), punnets_common::func\_sine\_int::get2ndDeriv(), punnets_common::func\_response::get2ndDeriv().
9.2 punnets_common Namespace Reference

9.2.2.2 typedef double punnets_common::real

Real number type.

The type that represents a real number.

Definition at line 28 of file punnets_base.h.

Referenced by punnets_private::tsynapse<T, const>::scheduleFire(), punnets_private::tneuron<T, debug>::scheduleFire(), punnets_private::tneuron<T, debug>::sendMessage(), punnets_private::tneuron<T, debug>::setExtInput(), punnets_private::tneuron<T, debug>::setConvergeLevel(), punnets_private::tneuron<T, debug>::setLoopBack(), punnets_common::func::tneuron<T, ext>::shouldDelete(), punnets_common::func::base<T>::shouldDelete(), punnets_private::tneuron<T, debug>::simulateElapse(), punnets_private::tneuron<T, debug>::simulateElapse(), punnets_common::tlogger<T>::tlogger(), punnets_private::tneuron<T, debug>::tneuron(), punnets_private::tneuron<T, debug>::tneuron<T, ext>::const<debug>::convergeLevel(), punnets_private::tneuron<T, ext>::const<debug>::getConvergeLevel(), punnets_private::tneuron<T, ext>::const<debug>::getLoopBack(), punnets_private::tneuron<T, ext>::const<debug>::messageSend(), punnets_private::tneuron<T, ext>::const<debug>::step(), punnets_private::tneuron<T, ext>::const<debug>::tneuron(), punnets_private::tneuron<T, ext>::const<debug>::tsynapse(), and punnets_common::tsynapse<T, const>::tsynapse<T, const>::base().
::getCurrentExtInput(), punnets_common::tneuron_base::getCurrentExtInput(), punnets_private::tneuron_ext< debug >::getCurrentExtInput(), punnets_private::tneuron< debug >::getCurrentExtInput(), punnets_common::tneuron_base::getCurrentSigLevel(), punnets_private::tneuron_ext< debug >::getCurrentSigLevel(), punnets_private::tneuron< debug >::getCurrentSigLevel(), punnets_common::tneuron_base::getCurrentThrLevel(), punnets_private::tneuron_ext< debug >::getCurrentThrLevel(), punnets_private::tneuron< debug >::getCurrentThrLevel(), punnets_common::tneuron_base::getMaxGradient(), punnets_common::func exp::getMaxGradient(), punnets_common::func exp::getValue(), punnets_common::func exp::getValueDomain(), punnets_private::tsynapse addfunc< debug >::getWeight(), punnets_private::tsynapse fatigue< debug >::getWeight(), makePulse(), toserver::pulseArrive(), punnets_private::tneuron< debug >::pulseArrive(), punnets_private::tneuron< debug >:: schedulesetConvergeLevel(), punnets_private::tneuron_ext< debug >::setConvergeLevel(), punnets_private::tneuron_ext< debug >::setExtInput(), punnets_common::func deriveq::setZeroPoint(), punnets_common::func deriveq::getValueDomain(), punnets_private::tsynapse addfunc< debug >::maxPulse(), punnets_private::tsynapse message< debug >::maxPulse(), and punnets_private::tsynapse messfunc< debug >::maxPulse()
9.3 punnets_nodebug Namespace Reference

Typedefs

- typedef punnets_private::tsynapse< false > tsynapse
tsynapse class.

- typedef punnets_private::tsynapse_message< false > tsynapse_message
tsynapse_message class.

- typedef punnets_private::tsynapse_fatigue< false > tsynapse_fatigue
tsynapse_fatigue class.

- typedef punnets_private::tsynapse_addfunc< false > tsynapse_addfunc
tsynapse_addfunc class.

- typedef punnets_private::tsynapse_messfunc< false > tsynapse_messfunc
tsynapse_messfunc class.

- typedef punnets_private::tneuron< false > tneuron
tneuron class.

- typedef punnets_private::tneuron_ext_const< false > tneuron_ext Const
tneuron_ext Const class.

- typedef punnets_private::tneuron_ext< false > tneuron_ext
tneuron_ext class.

9.3.1 Detailed Description

This namespace provides Punnets library without debugging facility. Importing this namespace, a user can access an efficient variant of the Punnets library. It is recommended for a user to use punnets (p. 21) namespace, at least in the period of development. For most of the available classes, refer to the punnets_common (p. 23) namespace.

This namespace imports punnets_common (p. 23) so that every definitions in the punnets_common (p. 23) namespace can be transparently used in this namespace.
9.4 punnets_private Namespace Reference

This namespace provides some private classes used for punnets library. Users should not use classes and functions in this namespace directly.

Compounds

- class tneuron
- class tneuron_ext
- class tneuron_ext_const
- class tsynapse
- class tsynapse_addfunc
  
  Synapse class that adds a new function to the destination (postsynaptic) tneuron_ext (p. 81).

- class tsynapse_fatigue
- class tsynapse_message
- class tsynapse_messfunc
- struct valdomain

9.4.1 Detailed Description

This namespace provides some private classes used for punnets library. Users should not use classes and functions in this namespace directly.
Chapter 10

Punnets Class Documentation

10.1 punnets_common::debugflag< false > Class Template Reference

#include <dneuron.h>

Public Methods

- bool getDeb () const
  Get debugging condition, which is always false.

- void setDeb (bool)
  Set debugging condition, which always fails.

10.1.1 Detailed Description

template<> class punnets_common::debugflag< false >

This instantiation of the class is used for punnets_nodebug (p. 29) namespace. It provides always false value to the getDeb() (p. 31), so that the debugging code is optimized out.

Definition at line 114 of file dneuron.h.

The documentation for this class was generated from the following file:

- dneuron.h
10.2 punnets_common::debugflag< true > Class Template Reference

This instantiation of the class is used for punnets namespace.

#include <dneuron.h>

Public Methods

- **debugflag ()**
  
  Constructor initialized the debugging condition to false.

- **bool getDeb () const**
  
  Get debugging condition.

- **void setDeb (bool b)**
  
  Set debugging condition.

Protected Attributes

- **bool debug**
  
  The debugging condition.

10.2.1 Detailed Description

**template<> class punnets_common::debugflag< true >**

This instantiation of the class is used for punnets namespace.

Definition at line 125 of file dneuron.h.

The documentation for this class was generated from the following file:

- **dneuron.h**
#include <func.h>

Inheritance diagram for punnets_common::func_base::

Public Methods

- virtual bool shouldDelete (ntime_t t)
  Return true if the function will return only zeros after the specified time.

- virtual bool processMessage (ntime_t, const message_base &)

- virtual void setLambda (real)
  Change leak value on a leaky integrate function. Redefined in func deriveq_base (p. 43).

- virtual void setZeroPoint (real)
  Change zero point on a leaky integrate function. Redefined in func deriveq_base (p. 43).

- virtual real getMaxGradient (ntime_t t) const=0
  A pure virtual function that returns the max gradient of the function after the time t.

- virtual ntime_t getNextIncontinuity (ntime_t t) const

- virtual real getValue (ntime_t t) const=0
  A pure virtual function that returns f(t).

- virtual real get1stDeriv (ntime_t t) const=0
  A pure virtual function that returns df(t)/dt.

- virtual real get2ndDeriv (ntime_t t) const=0
  A pure virtual function that returns d²f(t)/dt².

- virtual void getValueDomain (ntime_t t, real &upslope, real &ceil, real &downslope,
  real &floor) const=0
  Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil,
  f(t)+α<ceil; downslope < f(t+α) < f(t)+α<upslope.

- virtual void get1stDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope,
  real &floor) const=0
  Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil,
  f(t)+α<ceil; downslope < f(t+α) < f(t)+α<upslope.

- virtual void get2ndDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope,
  real &floor) const=0
  Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil,
  f(t)+α<ceil; downslope < f(t+α) < f(t)+α<upslope.

- virtual func_base * clone ()=0
Allocates and returns another instance of this function.

- virtual std::string getDescription ()=0
  Obtains a human-readable description string of this function.

Protected Methods

- virtual void valueChange ()

10.3.1 Detailed Description

Abstract base class of a function.

A base class that represents a numerical function. Some functions need a reference point to be calculated based on differential equation. Such a function is derived from `func_deriveq_base` (p. 43), but `func_base` (p. 33) contains interface for such a function.

Definition at line 69 of file func.h.

10.3.2 Member Function Documentation

10.3.2.1 virtual ntime_t punnets_common::func_base::getNextIncontinuity
          (ntime_t) const [inline, virtual]

A virtual function that returns the next incontinuity point after the time t. In `func_base` (p. 33), the function always returns infinity.

Reimplemented in `punnets_common::func_step` (p. 66), `punnets_common::func_delta_int` (p. 41), `punnets_common::func_response` (p. 54), `punnets_common::func_sine` (p. 56), `punnets_common::func_sineshot` (p. 60), `punnets_common::func_exp` (p. 47), `punnets_common::func_sineshot` (p. 52), and `punnets_common::func_exp_diff` (p. 49).

Definition at line 96 of file func.h.

References punnets_common::ntime_t.

96 { return mak::Infinity; };

10.3.2.2 virtual bool punnets_common::func_base::processMessage (ntime_t, const
          message_base &) [inline, virtual]

Process the specified message at the specified time. Return true if the message is processed. In `func_base` (p. 33), this function always returns false (processes no message).

Reimplemented in `punnets_common::func_deriveq_base` (p. 43), `punnets_common::func_delta_int` (p. 40), `punnets_common::func_sineshot` (p. 61), and `punnets_common::func_exp_diff` (p. 49).

Definition at line 85 of file func.h.

References punnets_common::ntime_t.

85 { return false; }
10.3.2.3 virtual void punnets_common::func_base::valueChange () [inline, protected, virtual]

An entrance to recalculate coefficient at the value change caused by some external factor. In func_base (p. 33) this function do nothing. This function will be defined in the derived classes with the requirement of the recalculation.

Definition at line 75 of file func.h.

Referenced by punnets_common::func_const_int::func_const_int(), punnets_common::func_delta_int::func_delta_int(), punnets_common::func_exp_diff::func_exp_diff(), punnets_common::func_exp_int::func_exp_int(), punnets_common::func_sine::func_sine(), punnets_common::func_sine_int::func_sine_int(), punnets_common::func_sineshot::func_sineshot(), punnets_common::func_sineshot_int::func_sineshot_int(), punnets_common::func_exp_diff::processMessage(), and punnets_common::func_deriveq_base::setLambda().

75 { }

The documentation for this class was generated from the following file:

* func.h
10.4 punnets_common::func_const Class Reference

```cpp
#include <func.h>

Inheritance diagram for punnets_common::func_const:

```
```

Public Methods

- **func_const (real ic)**
  
  *constructs a func_const (p.36) with the constant c.*

- **virtual real getMaxGradient (ntime t) const**
  
  *A pure virtual function that returns the max gradient of the function after the time t.*

- **virtual real getValue (ntime t) const**
  
  *A pure virtual function that returns f(t).*

- **virtual real get1stDeriv (ntime t) const**
  
  *A pure virtual function that returns df(t)/dt.*

- **virtual real get2ndDeriv (ntime t) const**
  
  *A pure virtual function that returns d²f(t)/dt².*

- **virtual void getValueDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  *Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil;downslope < f(t)+α<ceil, f(t)+α<upslope.*

- **virtual void get1stDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  *Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil;downslope < f(t)+α<ceil, f(t)+α<upslope.*

- **virtual void get2ndDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  *Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil;downslope < f(t)+α<ceil, f(t)+α<upslope.*

- **virtual func_const * clone ()**
  
  *Allocates and returns another instance of this function.*

- **virtual std::string getDescription ()**
  
  *Obtains a human-readable description string of this function.*
10.4.1 Detailed Description

A constant function.
This class represents a constant function \( f(t) = c \).
Definition at line 201 of file func.h.
The documentation for this class was generated from the following file:

* func.h
10.5 punnets_common::func_const_int Class Reference

```c
#include <func.h>
```

Inheritance diagram for punnets_common::func_const_int:

```
punnets_common::func_base

punnets_common::func_deriveq_base

punnets_common::func_const_int
```

Public Methods

- **func_const_int (real ic)**
  
  Constructs a **func_const_int** (p. 38) with the constant c.

- **virtual real getMaxGradient (ntime, t) const**
  
  A pure virtual function that returns the max gradient of the function after the time t.

- **virtual real getValue (ntime, t) const**
  
  A pure virtual function that returns f(t).

- **virtual real get1stDeriv (ntime, t) const**
  
  A pure virtual function that returns df(t)/dt.

- **virtual real get2ndDeriv (ntime, t) const**
  
  A pure virtual function that returns d^2f(t)/dt^2.

- **virtual void getValueDomain (ntime, t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  Obtains a zeroth-order linear envelope of the function. floor < f(t+\alpha) < ceil, f(t)+\alpha;downslope < f(t+\alpha) < f(t)+\alpha;upslope.

- **virtual void get1stDerivDomain (ntime, t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  Obtains a first-order linear envelope of the function. floor < f(t+\alpha) < ceil, f(t)+\alpha;downslope < f(t+\alpha) < f(t)+\alpha;upslope.

- **virtual void get2ndDerivDomain (ntime, t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  Obtains a second-order linear envelope of the function. floor < f(t+\alpha) < ceil, f(t)+\alpha;downslope < f(t+\alpha) < f(t)+\alpha;upslope.

- **virtual func_const_int * clone ()**
  
  Allocates and returns another instance of this function.

- **virtual std::string getDescription ()**

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
Obtains a human-readable description string of this function.

10.5.1 Detailed Description

An integration of a constant function.
The result of solving $dx/dt = c - \lambda x$. That is, $x = c/\lambda + C e^{-\lambda t}$, where $C$ is an integration constant.
Definition at line 238 of file func.h.
The documentation for this class was generated from the following files:

- `func.h`
- `func.cpp`
public:

- `func_delta_int(real ir, ntime_t it0)`
  
  Constructs `func_delta_int` (p. 40) with pulse amplitude r, and pulse arrival time t0.

- `virtual real getMaxGradient(ntime_t t) const`  
  A pure virtual function that returns the max gradient of the function after the time t.

- `virtual real getValue(ntime_t t) const`

- `virtual real get1stDeriv(ntime_t t) const`
  A pure virtual function that returns df(t)/dt.

- `virtual real get2ndDeriv(ntime_t t) const`
  A pure virtual function that returns d²f(t)/dt².

- `virtual ntime_t getNextIncontinuity(ntime_t from) const`

- `virtual void getValueDomain(ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const`  
  Obtains a zeroth-order linear envelope of the function. floor < f(t+½alpha; ) < ceil, f(t)+½alpha;downslope < f(t+½alpha; ) < f(t)+½alpha;upslope.

- `virtual void get1stDerivDomain(ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const`  
  Obtains a first-order linear envelope of the function. floor < f(t+½alpha; ) < ceil, f(t)+½alpha;downslope < f(t+½alpha; ) < f(t)+½alpha;upslope.

- `virtual void get2ndDerivDomain(ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const`  
  Obtains a second-order linear envelope of the function. floor < f(t+½alpha; ) < ceil, f(t)+½alpha;downslope < f(t+½alpha; ) < f(t)+½alpha;upslope.

- `virtual std::string getDescription()`  
  Obtains a human-readable description string of this function.

- `virtual bool processMessage(ntime_t t, const message_base &m)`
Processes message_set::lambda and message_set::zero_point messages.

- virtual func::delta_int * clone ()
  Allocates and returns another instance of this function.

10.6.1 Detailed Description

\[ \frac{df}{dt} = r \delta(t - t_0) - \lambda f \]
Definition at line 316 of file func.h.

10.6.2 Member Function Documentation

10.6.2.1 virtual ntime_t punnets_common::func::delta_int::getNextIncontinuity (ntime_t from) const [inline, virtual]
A virtual function that returns the next incontinuity point after the time t. In func_base (p. 33), the function always returns infinity.
Reimplemented from punnets_common::func_base (p. 34).
Definition at line 345 of file func.h.
References punnets_common::ntime_t.

345 { return from < t0 ? t0 : mak::Infinity; }

10.6.2.2 virtual real punnets_common::func::delta_int::getValue (ntime_t t) const
[inline, virtual]
A function that returns a leaky integrated value between the last pulse arrival time \( t_0 \) and t. \( \frac{dx}{dt} = \frac{1}{\lambda} \int_{t_0}^{t} f(t) dt + C \)
Implements punnets_common::func_base (p. 33).
Definition at line 331 of file func.h.
References punnets_common::ntime_t, and punnets_common::real.
Referenced by processMessage().

334 { 
335 // std::cout << "zerop=" << zerop << ", t0=" << t0 << ",t=" << t << std::endl;
336 return (zerop < t0 && t0 <= t) ? r * \exp(-\lambda (t - t0)) : 0.0; }

The documentation for this class was generated from the following files:
- func.h
- func.cpp
10.7 punnets_common::func_delta_int::message_add_pulse

Class Reference

A message that adds a new pulse.

#include <func.h>

Inheritance diagram for punnets_common::func_delta_int::message_add_pulse:

```
punnets_common::message_base
```

Public Methods

- virtual const char * getMessageId () const

10.7.1 Detailed Description

A message that adds a new pulse.

Definition at line 354 of file func.h.

10.7.2 Member Function Documentation

10.7.2.1 virtual const char * punnets_common::func_delta_int::message_add_pulse::getMessageId () [inline, virtual]

Returns the pointer of a string that represents the class. The pointer must be same for any instance of a given class. A function can check the type of the message by comparing the pointer.

Implements punnets_common::message_base (p. 68).

Definition at line 360 of file func.h.

360 { return messageId; }

The documentation for this class was generated from the following files:

- func.h
- func.cpp
Public Methods

- virtual void setLambda (real new_lambda)
  
  Change leak value on a leaky integrate function.

- virtual void setZeroPoint (real new_zeropoint)

  Change zero point on a leaky integrate function.

- virtual bool processMessage (ntime_t t, const message_base &m)
  
  Processes message_set_lambda (p. 44) and message_set_zero_point (p. 45) messages.

Protected Methods

- virtual void zeropChange ()

10.8.1 Detailed Description

Abstract base class of a leaky integration function.

The function has a form of the following differential equation. \( \frac{dx}{dt} = f(t) - \lambda \cdot x \) The integration constant C is determined as \( f(\text{zerop})=0 \) stands.

Definition at line 125 of file func.h.

10.8.2 Member Function Documentation

10.8.2.1 virtual void punnets_common::func_deriveq_base::zeropChange ()

An entrance to recalculate coefficient at the zero-pointer change. In func_base (p. 33) this function do nothing. This function will be defined in the derived classes with the requirement of the recalculation.

Definition at line 134 of file func.h.

Referenced by punnets_common::func_response::setZeroPoint(), and setZeroPoint().

134 { }

The documentation for this class was generated from the following file:

- func.h
10.9 punnets_common::func_deriveq_base::message_set_lambda Class Reference

A message that changes lambda (leak value).

```
#include <func.h>
```

Inheritance diagram for punnets_common::func_deriveq_base::message_set_lambda:

```
```

Public Methods

- `virtual const char * getMessageId () const`

10.9.1 Detailed Description

A message that changes lambda (leak value).

Definition at line 146 of file func.h.

10.9.2 Member Function Documentation

10.9.2.1 `virtual const char * punnets_common::func_deriveq_base::message_set_lambda::getMessageId () [inline, virtual]`

Returns the pointer of a string that represents the class. The pointer must be same for any instance of a given class. A function can check the type of the message by comparing the pointer.

Implements `punnets_common::message_base` (p. 68).

Definition at line 152 of file func.h.

```
152 { return messageId; }
```

The documentation for this class was generated from the following files:

- `func.h`
- `func.cpp`
A message that changes zero point.

#include <func.h>

Inheritance diagram for punnets_common::func_deriveq_base::message_set_zero_point:

```
        punnets_common::func_deriveq_base::message_set_zero_point
            `punnets_common::message_base
```

Public Methods

- virtual const char * getMessageId () const

10.10.1 Detailed Description

A message that changes zero point.

Definition at line 156 of file func.h.

10.10.2 Member Function Documentation

10.10.2.1 virtual const char * punnets_common::func_deriveq_base::message_set_zero_point::getMessageId () [inline, virtual]

Returns the pointer of a string that represents the class. The pointer must be same for any instance of a given class. A function can check the type of the message by comparing the pointer.

Implements punnets_common::message_base (p. 68).

Definition at line 162 of file func.h.

162 { return messageId; }

The documentation for this class was generated from the following files:

- func.h
- func.cpp
10.11 punnets_common::func_exp Class Reference

#include <func.h>

Inheritance diagram for punnets_common::func_exp:

```
punnets_common::func_base
    ^
punnets_common::func_exp
```

Public Methods

- **func_exp (real ir, real psi, real it0)**
  Constructs a `func_exp` (p. 46) with initial value `r` at time `t0`, and decay constant `psi`.

- **virtual real getMaxGradient (ntime_t t) const**
  A pure virtual function that returns the max gradient of the function after the time `t`.

- **virtual real getValue (ntime_t t) const**
  A pure virtual function that returns `f(t)`.

- **virtual real get1stDeriv (ntime_t t) const**
  A pure virtual function that returns `df(t)/dt`.

- **virtual real get2ndDeriv (ntime_t t) const**
  A pure virtual function that returns `d^2f(t)/dt^2`.

- **virtual ntime_t getNextIncontinuity (ntime_t t) const**

- **virtual void getValueDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const**
  Obtains a zeroth-order linear envelope of the function. `floor < f(t+alpha) < ceil, f(t)+alpha; downslope < f(t+alpha) < f(t)+alpha; upslope.`

- **virtual void get1stDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const**
  Obtains a first-order linear envelope of the function. `floor < f(t+alpha) < ceil, f(t)+alpha; downslope < f(t+alpha) < f(t)+alpha; upslope.`

- **virtual void get2ndDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const**
  Obtains a second-order linear envelope of the function. `floor < f(t+alpha) < ceil, f(t)+alpha; downslope < f(t+alpha) < f(t)+alpha; upslope.`

- **virtual func_exp * clone ()**
  Allocates and returns another instance of this function.

- **virtual std::string getDescription ()**
  Obtains a human-readable description string of this function.
10.11 punnets_common::func_exp Class Reference

10.11.1 Detailed Description

Exponential function
Exponentially decaying function \( f(t) = r \exp(-\psi(t-t_0)) \).
Definition at line 655 of file func.h.

10.11.2 Member Function Documentation

10.11.2.1 virtual ntime_t punnets_common::func_exp::getNextIncontinuity
(ntime_t) const [inline, virtual]

A virtual function that returns the next incontinuity point after the time \( t \). In \texttt{func_base} (p. 33),
the function always returns infinity.
Reimplemented from \texttt{punnets_common::func_base} (p. 34).
Definition at line 679 of file func.h.
References \texttt{punnets_common::ntime_t}.

679 { return mak::Infinity; };

The documentation for this class was generated from the following file:

- \texttt{func.h}
10.12  punnets_common::func_exp_diff Class Reference

#include <func.h>

Inheritance diagram for punnets_common::func_exp_diff:

```
+----------------+           +----------------+           +-----------------
| punnets_common: |           | punnets_common: |           | punnets_common: |
| func_base      |           | func_exp_diff  |           | func_exp_diff  |
```

Public Methods

- **func_exp_diff** (real ir1, real ipsi1, real ir2, real ipsi2, real it0)
  constructs **func_exp_diff** (p. 48).

- **virtual real getMaxGradient** (ntime t) const

  A pure virtual function that returns the max gradient of the function after the time t.

- **virtual real getValue** (ntime t) const

  A pure virtual function that returns f(t).

- **virtual real get1stDeriv** (ntime t) const

  A pure virtual function that returns df(t)/dt.

- **virtual real get2ndDeriv** (ntime t) const

  A pure virtual function that returns d²f(t)/dt².

- **virtual ntime_t getNextIncontinuity** (ntime t) const

- **virtual void getValueDomain** (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const

  Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t)+αalpha; downslope < f(t)+αalpha; floor < f(t)+αalpha; upslope.

- **virtual void get1stDerivDomain** (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const

  Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t)+αalpha; downslope < f(t)+αalpha; floor < f(t)+αalpha; upslope.

- **virtual void get2ndDerivDomain** (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const

  Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t)+αalpha; downslope < f(t)+αalpha; floor < f(t)+αalpha; upslope.

- **virtual func_exp_diff * clone ()**

  Allocates and returns another instance of this function.

- **virtual std::string getDescription ()**

  Obtains a human-readable description string of this function.

- **virtual bool processMessage** (ntime t, const message_base &m)
10.12.1 Detailed Description

Difference of two exponential functions

External input of a form \( f(t) = r_1 \exp(- &psi;_1 (t - t_0)) - r_2 \exp(- &psi;_2 (t - t_0)) \). Here \( r_1 > 0 \) and \( 0 < &psi;_1 < &psi;_2 \).

Definition at line 801 of file func.h.

10.12.2 Member Function Documentation

10.12.2.1 virtual ntime_t punnets_common::func_exp_diff::getNextIncontinuity (ntime_t) const [inline, virtual]

A virtual function that returns the next incontinuity point after the time \( t \). In func_base (p. 33), the function always returns infinity.

Reimplemented from punnets_common::func_base (p. 34).

Definition at line 834 of file func.h.

References punnets_common::ntime_t.

834 { return mak::Infinity; }

10.12.2.2 bool punnets_common::func_exp_diff::processMessage (ntime_t t, const message_base & m) [virtual]

Process the specified message at the specified time. Return true if the message is processed. In func_base (p. 33), this function always returns false (processes no message).

Reimplemented from punnets_common::func_base (p. 34).

Definition at line 529 of file func.cpp.

References punnets_common::message_base::getMessageId(), punnets_common::ntime_t, and punnets_common::func_base::valueChange().

530 {
531 #ifdef USE_DYNAMIC
532 if( dynamic_cast<const message_add_event_time *>(m) != NULL )
533 #else
534 if( m.getMessageId() == message_add_event_time::messageId )
535 #endif
536 {
537 ir1 = r1orig + ir1 * exp(- &psi;_1 * (t-\( t_0 \))) ;
538 ir2 = r2orig + ir2 * exp(- &psi;_2 * (t-\( t_0 \))) ;
539 \( t_0 = t \);
540 valueChange();
541 return true;
542 }
543 return func_base::processMessage(t, m);
544 }

The documentation for this class was generated from the following files:

- func.h
- func.cpp
10.13 punnets_common::func_exp_diff::message_add_event_time Class Reference

```cpp
#include <func.h>
```

Inheritance diagram for punnets_common::func_exp_diff::message_add_event_time::

```
punnets_common::message_base
    `
punnets_common::func_exp_diff::message_add_event_time
```

Public Methods

- virtual const char * `getMessageId()` const

10.13.1 Detailed Description

Message to add an event. ????

Definition at line 846 of file func.h.

10.13.2 Member Function Documentation

10.13.2.1 virtual const char * punnets_common::func_exp_diff::message_add_event_time::getMessageId () [inline, virtual]

Returns the pointer of a string that represents the class. The pointer must be same for any instance of a given class. A function can check the type of the message by comparing the pointer.

Implements `punnets_common::message_base` (p. 68).

Definition at line 851 of file func.h.

851 { return messageId; }

The documentation for this class was generated from the following files:

- `func.h`
- `func.cpp`
Public Methods

- **func_exp_int**(real ir, real psi, real it0)
  Constructs a func_exp (p. 46) with initial value r at time t0, and decay constant psi.

- **virtual real getMaxGradient**(ntime t) const
  A pure virtual function that returns the max gradient of the function after the time t.

- **virtual real getValue**(ntime t) const
  A pure virtual function that returns f(t).

- **virtual real get1stDeriv**(ntime t) const
  A pure virtual function that returns df(t)/dt.

- **virtual real get2ndDeriv**(ntime t) const
  A pure virtual function that returns d²f(t)/dt².

- **virtual ntime getNextIncontinuity**(ntime t) const
- **virtual void getValueDomain**(ntime t, real &upslope, real &ceil, real &downslope, real &floor) const
  Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t+α) < f(t)+αupslope.

- **virtual void get1stDerivDomain**(ntime t, real &, real &, real &, real &) const
  Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t+α) < f(t)+αupslope.

- **virtual void get2ndDerivDomain**(ntime t, real &, real &, real &, real &) const
  Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t+α) < f(t)+αupslope.

- **virtual func_exp_int * clone**()
  Allocates and returns another instance of this function.

- **virtual std::string getDescription**()
  Obtains a human-readable description string of this function.
10.14.1 Detailed Description

Integration of an exponential function
leaky integration of external, exponential input $f(t) = r \exp(-\psi(t - t_0))$.

Todo:
    implement get1stDerivDomain etc. for this function

Definition at line 705 of file func.h.

10.14.2 Member Function Documentation

10.14.2.1 virtual ntime_t punnets_common::func_exp_int::getNextIncontinuity
(ntime_t) const [inline, virtual]

A virtual function that returns the next incontinuity point after the time t. In func_base (p. 33),
the function always returns infinity.

Reimplemented from punnets_common::func_base (p. 34).

Definition at line 744 of file func.h.

References punnets_common::ntime_t.

744 { return mak::Infinity; };

The documentation for this class was generated from the following file:

- func.h
10.15 punnets_common::func_response Class Reference

#include <func.h>

Inheritance diagram for punnets_common::func_response:

```
    punnets_common::func_base
     |                    |
     v                    v
punnets_common::func_deriveq_base
     |                    |
     v                    v
punnets_common::func_response
```

Public Methods

- virtual real getMaxGradient (ntime t) const
  
  A pure virtual function that returns the max gradient of the function after the time t.

- virtual real getValue (ntime t) const
  
  A pure virtual function that returns f(t).

- virtual real get1stDeriv (ntime t) const
  
  A pure virtual function that returns df(t)/dt.

- virtual real get2ndDeriv (ntime t) const
  
  A pure virtual function that returns d²f(t)/dt².

- virtual ntime_t getNextIncontinuity (ntime t) const

- virtual void getValueDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<upslope; downslope < f(t+α) < f(t)+α<ceil; upslope.

- virtual void get1stDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<upslope; downslope < f(t+α) < f(t)+α<ceil; upslope.

- virtual void get2ndDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<upslope; downslope < f(t+α) < f(t)+α<ceil; upslope.

- virtual std::string getDescription ()
  
  Obtains a human-readable description string of this function.

- virtual func_response * clone ()
  
  Allocates and returns another instance of this function.
virtual void setZeroPoint (real new_zeropoint)

Change zero point on a leaky integrate function.

10.15.1 Detailed Description

A response function.

Todo:

document this function

Definition at line 387 of file func.h.

10.15.2 Member Function Documentation

10.15.2.1 virtual ntime_t punnets_common::func_response::getNextIncontinuity

(ntime_t) const [inline, virtual]

A virtual function that returns the next incontinuity point after the time t. In func_base (p. 33),
the function always returns infinity.
Reimplemented from punnets_common::func_base (p. 34).
Definition at line 409 of file func.h.
References punnets_common::ntime_t.

409 { return mak::Infinity; };

The documentation for this class was generated from the following files:

- func.h
- func.cpp
10.16 punnets_common::func_sine Class Reference

#include <func.h>

Inheritance diagram for punnets_common::func_sine:

```
punnets_common::func_base
  |
  v
punnets_common::func_sine
```

Public Methods

- **func_sine** (real ir, real iomega, real itheta)
  
  Constructs a func_sine (p. 55) with amplitude r, angle velocity omega, and phase theta.

- virtual real getMaxGradient (ntime t) const
  
  A pure virtual function that returns the max gradient of the function after the time t.

- virtual real getValue (ntime t t) const
  
  A pure virtual function that returns f(t).

- virtual real get1stDeriv (ntime t t) const
  
  A pure virtual function that returns df(t)/dt.

- virtual real get2ndDeriv (ntime t t) const
  
  A pure virtual function that returns d²f(t)/dt².

- virtual ntime_t getNextIncontinuity (ntime t t) const

- virtual void getValueDomain (ntime t t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a zeroth-order linear envelope of the function. floor < f(t+&alpha;) < ceil, f(t)+&alpha;downslope < f(t+&alpha;) < f(t)+&alpha;upslope.

- virtual void get1stDerivDomain (ntime t t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a first-order linear envelope of the function. floor < f(t+&alpha;) < ceil, f(t)+&alpha;downslope < f(t+&alpha;) < f(t)+&alpha;upslope.

- virtual void get2ndDerivDomain (ntime t t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a second-order linear envelope of the function. floor < f(t+&alpha;) < ceil, f(t)+&alpha;downslope < f(t+&alpha;) < f(t)+&alpha;upslope.

- virtual func_sine * clone ()
  
  Allocates and returns another instance of this function.

- virtual std::string getDescription ()
  
  Obtains a human-readable description string of this function.
10.16.1 Detailed Description

Sinusoidal function

This function represents a sinusoidal function \( f(t) = r \sin(\omega t + \theta) \).

Definition at line 431 of file func.h.

10.16.2 Member Function Documentation

10.16.2.1 virtual ntime_t punnets_common::func_sine::getNextIncontinuity
(ntime_t) const  [inline, virtual]

A virtual function that returns the next incontinuity point after the time t. In func_base (p. 33),
the function always returns infinity.

Reimplemented from punnets_common::func_base (p. 34).

Definition at line 460 of file func.h.

References punnets_common::ntime_t.

460 { return mak::Infinity; };

The documentation for this class was generated from the following files:

- func.h
- func.cpp
#include `<func.h>`

Inheritance diagram for punnets_common::func_sine_int::

```
    punnets_common::func_base
    /       \
   /        \
 punnets_common::func_deriveq_base
 p
   \
   \
    punnets_common::func_sine_int
```

**Public Methods**

- **func_sine_int** (real ir, real iomega, real itheta)
  
  Constructs a *func_sine_int* (p. 57) with amplitude r, angle velocity omega, and phase theta.

- virtual real getMaxGradient (ntime t) const
  
  A pure virtual function that returns the max gradient of the function after the time t.

- virtual real getValue (ntime t) const
  
  A pure virtual function that returns f(t).

- virtual real get1stDeriv (ntime t) const
  
  A pure virtual function that returns df(t)/dt.

- virtual real get2ndDeriv (ntime t) const
  
  A pure virtual function that returns d²f(t)/dt².

- virtual void getValueDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil; downslope < f(t+α) < f(t)+α<upslope.

- virtual void get1stDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil; downslope < f(t+α) < f(t)+α<upslope.

- virtual void get2ndDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const
  
  Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil; downslope < f(t+α) < f(t)+α<upslope.

- virtual func_sine_int * clone ()
  
  Allocates and returns another instance of this function.

- virtual std::string getDescription ()

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
10.17.1 Detailed Description

Integrated function of `func_sine` (p. 55)
This function represents an integration of a sinusoidal function.
Definition at line 548 of file `func.h`.
The documentation for this class was generated from the following file:

- `func.h`
Public Methods

- **func_sineshot (real ir, real iomega, real it0)**
  
  Constructs a **func_sineshot** (p. 59) with amplitude r, angle velocity omega, and origin t0.

- **virtual real getMaxGradient (ntime_t) const**
  
  A pure virtual function that returns the max gradient of the function after the time t.

- **virtual bool shouldDelete (ntime_t current)**
  
  Return true if the function will return only zeros after the specified time.

- **virtual real getValue (ntime_t t) const**
  
  A pure virtual function that returns f(t).

- **virtual real get1stDeriv (ntime_t t) const**
  
  A pure virtual function that returns df(t)/dt.

- **virtual real get2ndDeriv (ntime_t t) const**
  
  A pure virtual function that returns d²f(t)/dt².

- **virtual ntime_t getNextIncontinuity (ntime_t from) const**

- **virtual void getValueDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const**

- **virtual void get1stDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  Obtains a first-order linear envelope of the function.  
  floor < f(t+α); < ceil,  
  f(t)+α; < downslope < f(t+α; < f(t)+α; upslope.

- **virtual void get2ndDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &floor) const**
  
  Obtains a second-order linear envelope of the function.  
  floor < f(t+α; < ceil,  
  f(t)+α; < downslope < f(t+α; < f(t)+α; upslope.

- **virtual func_sineshot * clone ()**
  
  Allocates and returns another instance of this function.

- **virtual std::string getDescription ()**
  
  Obtains a human-readable description string of this function.
• virtual bool processMessage (ntime_t t, const message_base &m)

10.18.1 Detailed Description

A single shot of sinusoidal function

This function represents a single shot (one cycle) of a sinusoidal function \( f(t) = r \sin^2(\omega(t - t0)) \) \( (t0 \leq t \leq t0 + \pi/\omega) \).

Definition at line 477 of file func.h.

10.18.2 Member Function Documentation

10.18.2.1 virtual ntime_t punnets_common::func_sineshot::getNextIncontinuity (ntime_t from) const [inline, virtual]

A virtual function that returns the next incontinuity point after the time t. In func_base (p. 33), the function always returns infinity.

Reimplemented from punnets_common::func_base (p. 34).

Definition at line 509 of file func.h.

References punnets_common::ntime_t.

509 { return from < t0 ? t0 : mak::Infinity; };

10.18.2.2 void punnets_common::func_sineshot::getValueDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &oor) const [virtual]

A virtual function that returns the next incontinuity point after the time t. Although this function has no incontinuity, it returns t0 because the linear envelope is divided at the point.

Implements punnets_common::func_base (p. 33).

Definition at line 175 of file func.cpp.

References punnets_common::ntime_t, and punnets_common::real.

176 {
177    upslope = ceil = downslope = floor = 0.0;
178    if( t < t0 || t >= t0 + duration )
179        return;
180    else
181    {
182        real phase = omega_2 * (t - t0);
183        real phi = phase + M_PI;
184        if( phi >= 2*M_PI ) phi -= 2*M_PI;
185        ( r > 0 ? ceil : floor ) = r;
186        if( phase < 0.5 * M_PI )
187            { ( r > 0 ? upslope : downslope ) = omega_2 * r_div_2 * cos( alpha * cos( beta * phi ) + 0.5*M_PI - alpha); }
188            ( r > 0 ? downslope : upslope ) = - omega_2 * r_div_2 * cos( alpha * cos( beta * (phi-M_PI) ) + 0.5*M_PI - alpha);
189        } else if( phase < M_PI )
190            { ( r > 0 ? downslope : upslope ) = r_div_2 * omega_2 * sin(phase );

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
virtual bool punnets_common::func_sineshot::processMessage(ntime_t t, const message_base & m) [inline, virtual]

Process the specified message at the specified time. Return true if the message is processed. In func_base (p. 33), this function always returns false (processes no message).

Reimplemented from punnets_common::func_base (p. 34).

Definition at line 528 of file func.h.

References punnets_common::message_base::getMessageId(), and punnets_common::ntime_t.

The documentation for this class was generated from the following files:

- func.h
- func.cpp

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
10.19  

punnets_common::func_sineshot::message_set_t0

Class Reference

A message that changes t0.

#include <func.h>

Inheritance diagram for punnets_common::func_sineshot::message_set_t0:

```
punnets_common::func_sineshot::message_set_t0
```

Public Methods

- virtual const char * getMessageId () const

10.19.1  Detailed Description

A message that changes t0.
Definition at line 520 of file func.h.

10.19.2  Member Function Documentation

10.19.2.1  virtual const char* punnets_common::func_sineshot::message_set_t0::get-
MessageId () [inline, virtual]

Returns the pointer of a string that represents the class. The pointer must be same for any
instance of a given class. A function can check the type of the message by comparing the pointer.
Implements punnets_common::message_base (p. 68).
Definition at line 525 of file func.h.

525 { return messageId; }

The documentation for this class was generated from the following files:

- func.h
- func.cpp
#include <func.h>

Inheritance diagram for punnets_common::func_sineshot_int::

```
  punnets_common::func_base
   `-- punnets_common::func_deriveq_base
       `-- punnets_common::func_sineshot_int
```

## Public Methods

- **func_sineshot_int** (real ir, real iomega)
  
  Constructs a `func_sineshot_int` (p. 63) with amplitude r and angle velocity omega.

- **virtual real getMaxGradient (ntime_t) const**
  
  A pure virtual function that returns the max gradient of the function after the time t.

- **virtual real getValue (ntime_t t) const**
  
  A pure virtual function that returns f(t).

- **virtual real get1stDeriv (ntime_t t) const**
  
  A pure virtual function that returns df(t)/dt.

- **virtual real get2ndDeriv (ntime_t t) const**
  
  A pure virtual function that returns d²f(t)/dt².

- **virtual void getValueDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &floord) const**
  
  Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil; downslope < f(t)+α<ceil; upslope.

- **virtual void get1stDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &fordin) const**
  
  Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil; downslope < f(t)+α<ceil; upslope.

- **virtual void get2ndDerivDomain (ntime_t t, real &upslope, real &ceil, real &downslope, real &fordin) const**
  
  Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+α<ceil; downslope < f(t)+α<ceil; upslope.

- **virtual func_sineshot_int * clone ()**
  
  Allocates and returns another instance of this function.

- **virtual std::string getDescription ()**
10.20.1 Detailed Description

Integrated function of \texttt{func\_sineshot} (p. 59)

This function represents an integration of \texttt{func\_sineshot} (p. 59). $t_0$ is always equal to zerop.

Definition at line 612 of file func.h.

The documentation for this class was generated from the following files:

- \texttt{func.h}
- \texttt{func.cpp}
10.21 punnets_common::func_step Class Reference

#include <func.h>

Inheritance diagram for punnets_common::func_step:

```
<
classdef

punnets_common::func_base

// punnets_common::func_step
```

Public Methods

- virtual `real getMaxGradient (ntime t) const`
  
  A pure virtual function that returns the max gradient of the function after the time t.

- virtual `real getValue (ntime t) const`
  
  A pure virtual function that returns f(t).

- virtual `real get1stDeriv (ntime t) const`
  
  A pure virtual function that returns df(t)/dt.

- virtual `real get2ndDeriv (ntime t) const`
  
  A pure virtual function that returns d²f(t)/dt².

- virtual `ntime t getNextIncontinuity (ntime t from) const`

- virtual void `getValueDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const`
  
  Obtains a zeroth-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t+α) < f(t)+αupslope.

- virtual void `get1stDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const`
  
  Obtains a first-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t+α) < f(t)+αupslope.

- virtual void `get2ndDerivDomain (ntime t, real &upslope, real &ceil, real &downslope, real &floor) const`
  
  Obtains a second-order linear envelope of the function. floor < f(t+α) < ceil, f(t)+αdownslope < f(t+α) < f(t)+αupslope.

- virtual func_step * `clone ()`
  
  Allocates and returns another instance of this function.

- virtual std::string `getDescription ()`
  
  Obtains a human-readable description string of this function.
10.21.1 Detailed Description

A step function.
This class represents a step function, \( f(t) = r s(t - t_0) \).
Definition at line 277 of file func.h.

10.21.2 Member Function Documentation

10.21.2.1 virtual ntime_t punnets_common::func_step::getNextIncontinuity
(ntime_t from) const [inline, virtual]

A virtual function that returns the next incontinuity point after the time \( t \). In \texttt{func\_base} (p. 33),
the function always returns infinity.
Reimplemented from \texttt{punnets\_common::func\_base} (p. 34).
Definition at line 294 of file func.h.
References \texttt{punnets\_common::ntime\_t}.

294 { return from < t0 ? t0 : mak::Infinity; };

The documentation for this class was generated from the following file:

- \texttt{func.h}
10.22 punnets_common::greater_tevent Struct Reference

#include <dsched.h>

10.22.1 Detailed Description

Function class to compare event times
This class is used to construct a local priority queue on STL. In a queue, events are sorted in an ascending order of event time. This class is privately used internally in the punnets library.
Definition at line 102 of file dsched.h.
The documentation for this struct was generated from the following file:

- dsched.h
10.23 punnets_common::message_base Class Reference

#include <func.h>

Inheritance diagram for punnets_common::message_base::

```
```

Public Methods

- virtual const char * getMessageId () const=0

10.23.1 Detailed Description

Base class of a message to a function

A message is used to deliver a change to a function. Usually an event is delivered to a neuron, but a neuron has several functions. Moreover, we don’t make neuron to handle correspondence of a specific event to a specific function. For this purpose a message is used, as every function knows which message has correspondence to itself.

Definition at line 49 of file func.h.

10.23.2 Member Function Documentation

10.23.2.1 virtual const char * punnets_common::message_base::getMessageId ()

[pure virtual]

Returns the pointer of a string that represents the class. The pointer must be same for any instance of a given class. A function can check the type of the message by comparing the pointer.

Implemented in punnets_common::func_derivative_base::message_set_lambda (p. 44), punnets_common::func_derivative_base::message_set_zero_point (p. 45), punnets_common::func_derivative_base::message_add_times (p. 42), punnets_common::func_sineshot::message_add_event (p. 62), and punnets_common::func_exp_diff::message_add_event_time (p. 50).

Referenced by punnets_common::func_sineshot::processMessage(), punnets_common::func_derivative_base::processMessage(), and punnets_common::func_exp_diff::processMessage().

The documentation for this class was generated from the following file:

- func.h
#include <dsched.h>

Inheritance diagram for punnets_common::taction::

```
```

Public Methods

- virtual void **activate**(tscheduler &scheduler, ntime_t current_time)=0
- virtual tqueue * queue () const=0
  Obtain a local event queue of this action.

- virtual const char * **getClassName** () const=0
  Get the class name of this action. Primarily for debugging.

## 10.24.1 Detailed Description

Action (some affection to an entity)

Class taction is a abstract base class of "changing something", such as pulse arrival etc. Event is an instance of an action, represented by a pair of time and action. When simulation time reaches the event time, the scheduler triggers the event; the corresponding action is "activated" to perform the change. E.g. when a pulse arrival is activated, the potential of the destination neuron is changed.

Punnets adopts a distributed queue model, in which every action has a corresponding "local queue".

**Todo:**
write more

Definition at line 52 of file dsched.h.

## 10.24.2 Member Function Documentation

### 10.24.2.1 virtual void punnets_common::taction::activate (tscheduler & scheduler, ntime_t current_time) [pure virtual]

Activates the action at the specified time. When the corresponding new events are generated by the action, it is scheduled to the scheduler.

Implemented in punnets_common::tsentinel (p. 89), punnets_common::tlogger (p. 72), punnets_common::tsynapse_base (p. 95), punnets_private::tsynapse< debug > (p. 90), punnets_private::tsynapse_message< debug > (p. 98), punnets_private::tsynapse_fatigue< debug > (p. 97), punnets_private::tneuron< debug > (p. 76), punnets_private::tneuron_ext< debug > (p. 82), punnets_private::tsynapse_addfunc< debug > (p. 92), and punnets_private::tsynapse_messfunc< debug > (p. 101).

Referenced by punnets_common::tevent::activate().

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
The documentation for this class was generated from the following file:

- dsched.h
#include <dsched.h>

Public Methods

- **tevent** (ntime_t time, taction &iact)
  
  construct an event with the specified time and action.

- **ntime_t** **getTime** () const
  
  Obtain the time component of the event.

- **taction** & **getAction** () const
  
  Obtain the action component of the event.

- **void** **activate** (tscheduler &scheduler) const
  
  When the event time reaches, scheduler calls this member to trigger the event.

## 10.25.1 Detailed Description

Event (scheduled action)

Class tevent represents a scheduled event, which is a pair of a scheduled time and an action. When simulation time reaches the event time, the scheduler triggers the event; the corresponding action is "activated" to perform the change.

Definition at line 74 of file dsched.h.

The documentation for this class was generated from the following file:

- dsched.h
10.26 punnets_common::tlogger Class Reference

#include <dlogger.h>

Inheritance diagram for punnets_common::tlogger:

```
| punnets_common::taction
| ↓                    |
| punnets_common::tlogger
```

Public Types

- enum logoption { shownone = 0, showthr = 1, showext = 2, showpart = 4 }

Public Methods

- `tlogger (std::ostream &iout, ntime_t istep, ntime_t ifrom=0, ntime_t iuntil=mak::Infinity)`
  Constructs a logger with output stream iout, time step istep, and logging range between ifrom and iuntil.

- `virtual const char * getClassName () const`
  Get the class name of this action. Primarily for debugging.

- `virtual tqueue * queue () const`
  Obtain a local event queue of this action.

- `void add (tneuron_base &p, ntime_t delay, bool ishowthr, bool ishowext=false, real offset=0.0)`
  Add a neuron to be logged. Two boolean specifies logging options of thresholds and externals.

- `void add (tneuron_base &p, ntime_t delay=0.0, logoption logopt=showthr, real offset=0.0)`
  Add a neuron to be logged. logoption specifies logging options.

- `void add (tsynapse_base &p, ntime_t delay=0.0)`
  Add a synapse to be logged.

- `virtual void activate (tscheduler &scheduler, ntime_t current_time)`
  When activated by scheduler, the logger logs the current status to the log file.

- `void schedule (tscheduler &scheduler)`
  Schedule the logger itself to the specified scheduler.

- `void gnuplot_def (std::ostream &os, std::string file)`
10.26.1 Detailed Description

Activation logging class.

This class periodically probes the state (potential value) of neurons, and produces a log file of the state changes. To use logging, you need to do the followings:

- Register the neurons to be logged by calling add method of the logger. If you want to plot several neurons separately, specify offsets to displace the graph vertically. Or you may specify the horizontal displacement by delay. This is useful for analyzing events with temporal delays.
- Schedule the logger to the scheduler by calling schedule method of the logger.
- Create gnuplot definition file by calling gnuplot_def method of the logger.

Definition at line 51 of file dlogger.h.

10.26.2 Member Enumeration Documentation

10.26.2.1 enum punnets::common::tlogger::logoption

The option of the logging. You may log threshold, external inputs, and partitions. Multiple of them can be specified by logical or.

Enumeration values:

- shownone Nothing.
- showthr Threshold.
- showext External inputs.
- showpart Partitions.

Definition at line 58 of file dlogger.h.

Referenced by add().

```cpp
59 { 
60    shownone = 0,
61    showthr = 1,
62    showext = 2,
63    showpart = 4
64    };
```

10.26.3 Member Function Documentation

10.26.3.1 void punnets::common::tlogger::gnuplot_def (std::ostream & os, std::string file)

Generate a GNUPLOT definition file to the specified stream. file is a file name of the log file.

Definition at line 90 of file dlogger.cpp.

References showext, showpart, and showthr.

```cpp
91 { 
92    // cerr << "min/max: " << delaymin << "/" << delaymax << endl;
93    // os << "set term X11" << endl;
```
os << "set y2tics" << endl;
os << "plot [" << from << ":" << until << "]";

string delim = "";
int column = 1;
for( vector< neuentry >::iterator i = neus.begin(); i != neus.end(); i++ )
{
    int xcol = (i->delay == 0.0 ? 1 : ++column);
os << delim << " \" << endl << "\" " <<
    "using " << xcol << ":" << (++column) << " axes xly1 " <<
    "title \"" << i->neuron->getClassName() << ":" << i->neuron->getName() << "";
    if( i->delay != 0.0 )
os << "(delay " << i->delay << ")";
    if( i->offset != 0.0 )
os << "(offset " << i->offset << ")";
os << " signal\" with lines";
    delim = ",";

    if( i->logopt & showthr )
    {
        os << delim << " \" << endl << "\" " <<
            "using " << xcol << ":" << (++column) << " axes xly1 " <<
            "title \"" << i->neuron->getClassName() << ":" << i->neuron->getName() << "";
        if( i->delay != 0.0 )
os << "(delay " << i->delay << ")";
        if( i->offset != 0.0 )
os << "(offset " << i->offset << ")";
os << " threshold\" with dots";
        delim = ",";
    }
    if( i->logopt & showext )
    {
        os << delim << " \" << endl << "\" " <<
            "using " << xcol << ":" << (++column) << " axes xly1 " <<
            "title \"" << i->neuron->getClassName() << ":" << i->neuron->getName() << "";
        if( i->delay != 0.0 )
os << "(delay " << i->delay << ")";
        if( i->offset != 0.0 )
os << "(offset " << i->offset << ")";
os << " external\" with dots";
        delim = ",";
    }
    if( i->logopt & showpart )
    {
        os << delim << " \" << endl << "\" " <<
            "using " << xcol << ":" << (++column) << " axes xly1 " <<
            "title \"" << i->neuron->getClassName() << ":" << i->neuron->getName() << "";
        if( i->delay != 0.0 )
os << "(delay " << i->delay << ")";
        if( i->offset != 0.0 )
os << "(offset " << i->offset << ")";
os << " partition 0\" with impulses";
        delim = ",";
    }
    os << delim << " \" << endl << "\" " <<
        "using " << xcol << ":" << (++column) << " axes xly1 " <<
        "title \"" << i->neuron->getClassName() << ":" << i->neuron->getName() << "";
    if( i->delay != 0.0 )
os << "(delay " << i->delay << ")";
    if( i->offset != 0.0 )
os << "(offset " << i->offset << ")";
os << " partition 1\" with impulses";
    delim = ",";
}

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
161 os << "(delay " << i->delay << ")";
162 if( i->offset != 0.0 )
163 os << "(offset " << i->offset << ")";
164 os << " partition 2" with impulses";
165 os << delim << " \" << endl << \t" << logfile << ": " << "using " << xcol << ":" << (++column) << " axes xly1 " << "title \"" << i->neuron->getClassName() << " \"" << i->neuron->getName() << "\"
166 if( i->delay != 0.0 )
167 os << "(delay " << i->delay << ")";
168 if( i->offset != 0.0 )
169 os << "(offset " << i->offset << ")";
170 os << " partition delta_t" with impulses";
171 }
172 for( vector<pair<tsynapse_base *, ntime_t> >::iterator i = syns.begin(); i != syns.end(); i++ )
173 {
174 int xcol = (i->second == 0.0 ? 1 : ++column);
175 os << delim << " \" << endl << \t" << logfile << ": " << "using " << xcol << ":" << (++column) << " axes xly2 " << "title \"" << i->first->getClassName() << " \"" << i->first->getSrc().getName() << " -> " << i->first->getDest().getName() << "\"
176 if( i->second != 0.0 )
177 os << "(delay " << i->second << ")";
178 os << "\" with lines";
179 delim = ";
180 }
181 os << endl;
182 os << "pause -1" << endl;
183 }
184 }

The documentation for this class was generated from the following files:

- dlogger.h
- dlogger.cpp
# include <dneuron.h>

Inheritance diagram for punnets::tneuron::debug::

```
+--- punnets_common::tneuron_base
|   +--- punnets_common::taction
|       +--- punnets_private::tneuron< debug >
|               +--- punnets_private::tneuron_ext_const< debug >
```

Public Methods

- **tneuron** (std::string iname="", ntime_t isig_hv_period=def_sig_hv_period, ntime_t ithr_hv_period=def_thr_hv_period, real imin_threshold=def_min_threshold, real imax_threshold=def_max_threshold)

- virtual real **getCurrentSigLevel** (ntime_t current_time)
  
  *Calculates the current signal level.*

- virtual real **getCurrentThrLevel** (ntime_t current_time)
  
  *Calculates the current threshold level.*

- virtual void **pulseArrive** (tscheduler &scheduler, ntime_t current_time, real pulse_level)
  
  *Processes the pulse arrival.*

- virtual void **addSynapse** (tsynapse_base *s)
  
  *Add an synapse whose destination (post-synapse) is this neuron.*

- virtual void **eraseSynapse** (tsynapse_base *s)
  
  *Remove an synapse from this neuron.*

- virtual ntime_t **getLastFire** () const
  
  *Get the time of the last firing.*

- virtual ntime_t **getLastSimulate** () const
  
  *Get the time of the last simulation.*

- virtual void **activate** (tscheduler &scheduler, ntime_t current_time)
  
  *When activated as an event, it re-calculates its state and check firing.*

- virtual const char * **getName** () const
  
  *Get the class name.*

- virtual tqueue * **queue** () const
  
  *Get the pointer to the queue object.*
const std::vector< tsynapse_base * > & getSynapses ()
// Get the vector of synapses.

Protected Methods

- void simulateElapse (ntime_t current_time)
  // This function simulates time elapse up to the specified time.

- virtual void scheduleFire (tscheduler &scheduler, ntime_t current_time, bool resched=true)
  // Schedule the next firing.

Protected Attributes

- ntime_t sig_hv_period
  // Signal half-value period.

- ntime_t thr_hv_period
  // Threshold half-value period.

- real min_threshold
  // Minimum threshold level.

- real max_threshold
  // Maximum threshold level just after one firing.

- real coeff_sigdecay
  // Signal decaying coefficient is calculated from the signal half-value period.

- real coeff_thrdecay
  // Threshold decaying coefficient is calculated from the threshold half-value period.

Static Protected Attributes

- const real sig_converge_level = 0.0
  // The converge level (resting potential) of this neuron.

10.27.1 Detailed Description

template<bool debug> class punnets_private::tneuron< debug >

The class defines a leaky integrate-and-fire neuron with threshold change, which receives only immediate pulses. A neuron itself behaves as an action of an event. When the next firing is predicted, the action is scheduled at the predicted firing time (called ‘loopback’). When activated at a certain simulation time, the neuron re-calculates its state and process firing if necessary.

Definition at line 431 of file dneuron.h.
10.27.2 Constructor & Destructor Documentation

10.27.2.1 template <bool debug> punnets_private::tneuron< debug >::tneuron
(std::string iname = "", ntime_t isig_hv_period = def_sig_hv_period,
ntime_t ithr_hv_period = def_thr_hv_period, real imin_threshold =
def_min_threshold, real imax_threshold = def_max_threshold)

Construct a neuron with the specified name, signal half-value period, threshold half-value period,
minimum threshold and maximum threshold.

Definition at line 122 of file dneuron.cpp.

References punnets_common::ntime_t, and punnets_common::real.

127 : neuron_base(iname),
128     sig_level(0.0), last_simulate(0.0), last_fire(0.0),
129     sig_hv_period(isig_hv_period), thr_hv_period(ithr_hv_period),
130     min_threshold(imin_threshold), max_threshold(imax_threshold),
131     coeff_sigdecay(- M_LN2 / isig_hv_period),
132     coeff_thrdecay( - M_LN2 / ithr_hv_period )
133 { }

The documentation for this class was generated from the following files:

- dneuron.h
- dneuron.cpp
# include <dneuron.h>

Inheritance diagram for punnets_common::tneuron_base:

```
+----+----+
|    |    |
|    | punnets_common::tneuron_base |
|    |    |
+----+----+
    |    |
    | punnets_private::tneuron< debug > |
    |    |
+----+----+
    |    |
    | punnets_private::tneuron_ext< debug > |
    |    |
+----+----+
    |    |
    | punnets_private::tneuron_ext_const< debug > |
    |    |
```

Public Methods

- **tneuron_base** (std::string iname="")
  
  *Construct a neuron with the name iname.*

- **const std::string & getName () const**
  
  *Returns the name of this neuron.*

- **virtual void pulseArrive (tscheduler &scheduler, ntime_t current_time, real pulse_level)=0**

- **virtual void addSynapse (tsynapse_base *)**
  
  *Add a synapse input (this neuron becomes postsynaptic). A derived class should redefine this method.*

- **virtual real getCurrentSigLevel (ntime_t)**
  
  *This function probes the current signal level of the neuron. A derived class should redefine this method.*

- **virtual real getCurrentThrLevel (ntime_t)**
  
  *This function probes the current threshold level of the neuron. A derived class should redefine this method.*

- **virtual real getCurrentExtInput (ntime_t)**
  
  *This function probes the current external input level of the neuron. A derived class should redefine this method.*

- **virtual ntime_t getLastFire () const**
  
  *This function returns the last time that the neuron fired. Used in STDP and such.*

- **virtual ntime_t getLastSimulate () const**
  
  *This function returns the last time that the neuron has been simulated.*

- **virtual int getLastSimulateType () const**
  
  *This function returns the type of the last simulation (0th/1st/2nd and so on).*

- **virtual tqueue * queue () const**
  
  *Obtains the pointer to the queue object of this neuron.*
- virtual const char * getClassName () const
  
  Obtains the class name of this neuron.

## Protected Attributes

- std::string name
  
  The name of this neuron. Used in debugging and logging.

- tqueue _queue
  
  The event queue of this neuron.

### 10.28.1 Detailed Description

The base class of a neuron.

This abstract base class provides several interfaces for neuron access.

Definition at line 147 of file dneuron.h.

### 10.28.2 Member Function Documentation

#### 10.28.2.1 virtual void punnets_common::tneuron_base::pulseArrive (tscheduler & scheduler, ntime_t current_time, real pulse_level) [pure virtual]

These methods handle pulse arrivals of the neuron. The pulse may be a real number (immediate value), message, or a function to be added. The latter two has a default handler that does nothing. A derived class should redefine these methods.

Implemented in punnets_private::tneuron< debug > (p. 76), and punnets_private::tneuron_ext< debug > (p. 81).

The documentation for this class was generated from the following file:

- dneuron.h
Public Methods

- **tneuron_ext** (std::string iname="", real sig_hv_period=1.0)
  
  Construct a **tneuron_ext** (p.81) class with the specified name and signal half-value period.

- virtual real **getCurrentSigLevel** (ntime_t current_time)
  
  Get the current signal level.

- virtual real **getCurrentThrLevel** (ntime_t)
  
  Get the current threshold level, which is always 0.0.

- virtual void **pulseArrive** (tscheduler &scheduler, ntime_t current_time, real pulse_level)
  
  Process the pulse arrival to this neuron.

- virtual void **pulseArrive** (tscheduler &scheduler, ntime_t current_time, message_base *mess)
  
  Process the message arrival to this neuron.

- virtual void **pulseArrive** (tscheduler &scheduler, ntime_t current_time, func_base *func)
  
  Process the function arrival to this neuron.

- virtual void **addSynapse** (tsynapse_base *s)
  
  Add an incoming synapse to this neuron.

- virtual void **eraseSynapse** (tsynapse_base *s)
  
  Erase an incoming synapse from this neuron.

- virtual void **addExt** (func_base *s)
  
  Add an external function to this neuron.

- virtual ntime_t **getLastFire** () const
  
  This function returns the last time that the neuron fired. Used in STDP and such.

- virtual ntime_t **getLastSimulate** () const
  
  This function returns the last time that the neuron has been simulated.
• virtual int \texttt{getLastSimulateType} () const  
  \textit{This function returns the type of the last simulation (0th/1st/2nd and so on).}

• virtual void \texttt{activate} (tscheduler &scheduler, ntime\_t current\_time)  
  \textit{When activated as an event (loopback), it re-calculates its state and check firing.}

• virtual const char * \texttt{getClassName} () const  
  \textit{Gets the class name.}

• const std::vector< tsynapse\_base * > & \texttt{getSynapses} ()  
  \textit{Gets the collection of the synapses.}

• virtual tqueue * \texttt{queue} () const  
  \textit{Get the queue object of this neuron.}

• void \texttt{setLoopBack} (tscheduler &scheduler, ntime\_t schedule\_time)

\textbf{Protected Methods}

• void \texttt{fire} (tscheduler &scheduler, ntime\_t current\_time)  
  \textit{Process firing at the specified time.}

• real \texttt{calcSignal} (ntime\_t current\_time)  
  \textit{Calculate the power of signal at the specified time.}

• virtual void \texttt{scheduleFire} (tscheduler &scheduler, ntime\_t current\_time, bool resched=true)  
  \textit{Schedule the next firing.}

• bool \texttt{sendMessage} (ntime\_t t, message\_base *mess)  
  \textit{Send the message to a function.}

• bool \texttt{broadcastMessage} (ntime\_t t, message\_base *mess)  
  \textit{Send the message to all the functions.}

\textbf{Protected Attributes}

• std::vector< tsynapse\_base * > \texttt{synapses}  
  \textit{A collection of incoming synapses.}

• std::vector< func\_base * > \texttt{exts}  
  \textit{A collection of functions.}

• func\_delta\_int * \texttt{pulses}  
  \textit{A pulse function to support tsynapse.}

• ntime\_t \texttt{last\_simulate}  
  \textit{Last simulation time, only for simulate-logging.}
- ntime_t last_fire
  
  Last firing time.

- ntime_t loopback
  
  Loopback time. It is used to check the loopback time changes.

- real lambda
  
  The decaying coefficient for the inputs.

- int last_simulate_type
  
  The last simulation type, only for simulate-logging.

### 10.29.1 Detailed Description

template<bool debug> class punnets_private::tneuron_ext< debug >

The extended neuron class. This class of neurons calculates the potential as a sum of functions. You can specify arbitrary functions, if you can provide the linear envelopes of the function and 1st/2nd derivatives.

Definition at line 604 of file dneuron.h.

### 10.29.2 Member Function Documentation

#### 10.29.2.1 template<bool debug> void punnets_private::tneuron_ext< debug >::setLoopBack (tscheduler & scheduler, ntime_t schedule_time) [inline]

Specify the loopback at the time. If the time is infinity, the loopback is cancelled. This method is public because the loopback is explicitly set.

Definition at line 718 of file dneuron.h.

References punnets_common::ntime_t.

Referenced by punnets_private::tneuron_ext< debug >::scheduleFire().

```cpp
719 { 
720   loopback = schedule_time;
721   if( schedule_time < mak::Infinity )
722     scheduler.scheduleEvent( schedule_time, *this );
723 }
```

The documentation for this class was generated from the following files:

- dneuron.h
- dneuron.cpp

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
Public Methods

- *tneuron_ext_const* (std::string iname="", ntime t isig_hv_period=def_isig_hv_period, ntime t ithr_hv_period=def_ithr_hv_period, real imin_threshold=def_imin_threshold, real imax_threshold=def_imax_threshold, real iext_input=def_iext_input)
- virtual real *getCurrentSigLevel* (ntime t current_time)
  
  Calculates the current signal level.

- real *getExtInput* ()
  
  Get the weight of the external input.

- virtual real *getCurrentExtInput* (ntime t)
  
  Get the weight of the external input for logging.

- void *setExtInput* (tscheduler &scheduler, ntime t current_time, real val)
  
  Change the weight of the external input at the specified time.

- real *getConvergeLevel* ()
  
  Get the converge level (resting potential of this neuron).

- void *setConvergeLevel* (tscheduler &scheduler, ntime t current_time, real val)
- virtual const char * *getName* () const
  
  Get the class name.

Protected Methods

- void *simulateElapse* (ntime t current_time)
- virtual void *scheduleFire* (tscheduler &scheduler, ntime t current_time, bool resched=true)
  
  Schedule the next firing.
Protected Attributes

- real sig_converge_level

The converge level (resting potential) of this neuron.

10.30.1 Detailed Description

template<bool debug> class punnets_private::tneuron_ext_const< debug >

This class extends the tneuron class with a constant external input.
Definition at line 539 of file dneuron.h.

10.30.2 Constructor & Destructor Documentation

10.30.2.1 template<bool debug> punnets_private::tneuron_ext_const< debug >::tneuron_ext_const (std::string iname = "", ntime_t isig_hv_period = defsig_hv_period, ntime_t ithr_hv_period = defthr_hv_period, real imin_threshold = defmin_threshold, real imax_threshold = defmax_threshold, real iext_input = defext_input) [inline]

Construct a neuron with the specified name, signal half-value period, threshold half-value period, minimum threshold, maximum threshold, and the weight of the external input.
Definition at line 562 of file dneuron.h.
References punnets_common::ntime_t, and punnets_common::real.

568 : tneuron<debug>( iname, isig_hv_period, ithr_hv_period, imin_threshold, imax_threshold ),
569 ext_input(iext_input), sig_converge_level(iext_input / coeff_sigdecay) { }

10.30.3 Member Function Documentation

10.30.3.1 template<bool debug> void punnets_private::tneuron_ext_const< debug >::setConvergeLevel (tscheduler & scheduler, ntime_t current_time, real val) [inline]

Change the weight of the external input at the specified time, so that the converge level (resting potential of this neuron) becomes the specified value.
Definition at line 587 of file dneuron.h.
References punnets_common::ntime_t, and punnets_common::real.

588 { simulateElapse( current_time ); sig_converge_level = val; ext_input = sig_converge_level * coeff_sigdecay; scheduler->scheduleFire( scheduler, current_time ); }

10.30.3.2 template<bool debug> void punnets_private::tneuron_ext_const< debug >::simulateElapse (ntime_t current_time) [inline, protected]

This function simulates time elapse up to the specified time. Note that this function is non-virtual to speed-up the simulation.

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
Reimplemented from `punnets_private::tneuron< debug >` (p. 77).

Definition at line 548 of file dneuron.h.

References `punnets_common::ntime_t`.

549  
550      sig_level = sig_converge_level + 
551      ( ( sig_level - sig_converge_level ) 
552         * exp( (current_time - last_simulate) * coeff_sigdecay ) );  
553  last_simulate = current_time;  
554 

The documentation for this class was generated from the following files:

- `dneuron.h`
- `dneuron.cpp`
10.31 toobserver Class Reference

Public Methods

- toobserver (string iname, bool ishow=false)
  The constructor.

- virtual void pulseArrive (tscheduler &, ntime & current_time, real pulse_level)
  When a pulse is arrived, it counts and reports the pulse arrival.

- int getNPulses () const
  Get the number of pulses observed.

- void clearNPulses ()
  Clear the pulse counter.

10.31.1 Detailed Description

An observer neuron class.

When this neuron received a pulse, it counts the number of pulses. If constructed with true show parameter, it also reports the arrival of a pulse to the console.

Definition at line 51 of file dtest.cpp.

The documentation for this class was generated from the following file:

- dtest.cpp
10.32 punnets_common::tsched_double Class Reference

#include <dsched.h>

Public Methods

- **tsched_double ()**
  Constructs a scheduler with no events.

- **void scheduleEvent (const tevent &event)**
  Schedule an event.

- **void scheduleEvent (ntime_t t, taction &act)**
  Schedule an action for the specified time.

- **bool isScheduled () const**
  Tests if the scheduler is not empty.

- **ntime_t run (ntime_t until=HUGE_VAL)**
  Run the simulation (continue triggering events) until the specified time.

10.32.1 Detailed Description

Event scheduler class.
The scheduler class activates events in an order of time.
Definition at line 115 of file dsched.h.
The documentation for this class was generated from the following files:

- dsched.h
- dsched.cpp
10.33 punnets_common::tsentinel Class Reference

Inheritance diagram for punnets_common::tsentinel:

```
punnets_common::taction
   `-- punnets_common::tsentinel
```

Public Methods

- virtual void `activate` (tscheduler &, ntime_t)
- tqueue * `queue` () const
  
  Obtain a local event queue of this action.

- virtual const char * `getClassName` () const
  
  Get the class name of this action. Primarily for debugging.

10.33.1 Detailed Description

Sentinel action.

This class is used to mark the end of the simulation range.

Definition at line 37 of file dsched.cpp.

10.33.2 Member Function Documentation

10.33.2.1 virtual void `punnets_common::tsentinel::activate` (tscheduler &, ntime_t) [inline, virtual]

Activates the action at the specified time. When the corresponding new events are generated by
the action, it is scheduled to the scheduler.

Implements `punnets_common::taction` (p.69).

Definition at line 46 of file dsched.cpp.

References punnets_common::ntime_t.

```
47        {  processed = true;
48        }
```

The documentation for this class was generated from the following file:

- dsched.cpp
10.34 punnets_private::tsynapse< debug > Class Template Reference

#include <dneuron.h>

Inheritance diagram for punnets_private::tsynapse< debug >::

```
<table>
<thead>
<tr>
<th>punnets_common::taction</th>
</tr>
</thead>
<tbody>
<tr>
<td>punnets_common::tsynapse_base</td>
</tr>
<tr>
<td>punnets_private::tsynapse&lt; debug &gt;</td>
</tr>
</tbody>
</table>
```

### Public Methods

- **tsynapse** (tneuron_base &isrc, tneuron_base &idest, ntime_t idelay, real iweight)
  
  *Construct a synapse with the specified source, destination, delay and weight.*

- **tsynapse** (tneuron_base &idest, ntime_t idelay, real iweight)
- **virtual void setSrc** (tneuron_base &isrc)
  
  *Specify the source neuron.*

- **virtual real getWeight** () const
  
  *Get the weight value of this synapse.*

- **virtual tneuron_base & getSrc** () const
  
  *Get the source neuron (pre-synaptic).*

- **virtual tneuron_base & getDest** () const
  
  *Get the destination neuron (post-synaptic).*

- **void addWeight** (real delta_w)
  
  *Modify the weight value with the specified step size.*

- **void addDelay** (ntime_t delta_d)
  
  *Modify the delay with the specified step size.*

- **virtual void activate** (tscheduler &scheduler, ntime_t current_time)
  
  *The event handler delivers a pulse to the destination (post-synaptic) neuron.*

- **virtual tqueue * queue** () const
  
  *Obtains the pointer to the queue object of this event.*
10.34 punnets_private::tsynapse< debug > Class Template Reference

10.34.1 Detailed Description

template< bool debug > class punnets_private::tsynapse< debug >

The default synapse class. This class of synapses has an effect for the destination (post-synaptic) neuron to update its potential immediately with a constant amount of the weight.
Definition at line 265 of file dneuron.h.

10.34.2 Constructor & Destructor Documentation

10.34.2.1 template< bool debug > punnets_private::tsynapse< debug >::tsynapse
(tneuron_base & idest, ntime_t idelay, real iweight) [inline]

Construct a synapse with the specified destination, delay and weight. Source neuron will be specified after the construction via setSrc() (p. 90) method.
Definition at line 278 of file dneuron.h.
References punnets_common::ntime_t, and punnets_common::real.

279 : tsynapse_base(idelay), src(NULL), dest(&idest), weight(iweight) { }

The documentation for this class was generated from the following file:

- dneuron.h


10.35 punnets private::tsynapse_addfunc< debug > Class Template Reference

Synapse class that adds a new function to the destination (postsynaptic) tneuron ext (p. 81).
#include <dneuron.h>

Inheritance diagram for punnets private::tsynapse_addfunc< debug >:

```
punnets_common::taction

    punnets_common::tsynapse_base

    punnets_private::tsynapse_addfunc< debug >
```

Public Methods

- `tsynapse_addfunc` (tneuron base &isrc, tneuron ext< debug > &idest, real idelay, func base *ifunc, message base *imess=NULL, real ilev=0.0)
- `tsynapse_addfunc` (tneuron ext< debug > &idest, real idelay, func base *ifunc, message base *imess=NULL, real ilev=0.0)
- virtual void `setSrc` (tneuron base &isrc) 
  
  Set the source neuron (presynaptic) of this synapse. A derived class should redefine this method.

- virtual tneuron base & `getSrc` () const
  
  Get the source neuron (presynaptic). A derived class should redefine this method.

- virtual tneuron ext< debug > (& `getDest` () const
  
  Get the destination neuron (postsynaptic). A derived class should redefine this method.

- virtual void `activate` (tscheduler &scheduler, ntime current time)
  
  When activated, The synapse clones the function and added it to the destination neuron.

- virtual tqueue * `queue` () const
  
  Obtain a local event queue of this action.

10.35.1 Detailed Description

template< bool debug> class punnets_private::tsynapse_addfunc< debug >

Synapse class that adds a new function to the destination (postsynaptic) tneuron ext (p. 81).
Definition at line 734 of file dneuron.h.
10.35.2 Constructor & Destructor Documentation

10.35.2.1 template<bool debug> punnets_private::tsynapse_addfunc< debug>
    ::tsynapse_addfunc (tneuron_base & isrc, tneuron_ext< debug > & idest, real idelay,
    func_base * ifunc, message_base * imess = NULL, real ilev = 0.0) [inline]

Constructs add-function synapse with the specified source, destination, delay, a pointer to a function, a pointer to a message that is sent before adding, and immediate pulse level.
Definition at line 746 of file dneuron.h.
References punnets_common::real.

747 : tsynapse_base(idelay), src(&isrc), dest(&idest), func(ifunc), mess(imess), lev(ilev) { }

10.35.2.2 template<bool debug> punnets_private::tsynapse_addfunc< debug>
    ::tsynapse_addfunc (tneuron_ext< debug > & idest, real idelay, func_base * ifunc,
    message_base * imess = NULL, real ilev = 0.0) [inline]

Constructs add-function synapse with the specified destination, delay, a pointer to a function, a pointer to a message that is sent before adding, and immediate pulse level.
Definition at line 750 of file dneuron.h.
References punnets_common::real.

751 : tsynapse_base(idelay), src(NULL), dest(&idest), func(ifunc), mess(imess), lev(ilev) { }

The documentation for this class was generated from the following file:

- dneuron.h
10.36 punnets_common::tsynapse_base Class Reference

#include <dneuron.h>

Inheritance diagram for punnets_common::tsynapse_base:

Public Methods

- **tsynapse_base (ntime_t idelay)**
  
  Construct a synapse with the specified delay.

- **ntime_t getDelay () const**
  
  Get the delay length.

- **virtual real getWeight () const**
  
  Get the weight of the synapse. A derived class should redefine this method.

- **virtual void setSrc (tneuron_base &)**
  
  Set the source neuron (presynaptic) of this synapse. A derived class should redefine this method.

- **virtual tneuron_base & getSrc () const**
  
  Get the source neuron (pre-synaptic). A derived class should redefine this method.

- **virtual tneuron_base & getDest () const**
  
  Get the destination neuron (post-synaptic). A derived class should redefine this method.

- **virtual void activate (tscheduler &scheduler, ntime_t current_time)=0**

- **virtual const char * getClassName () const**
  
  Obtains the class name of this neuron.

Protected Attributes

- **ntime_t ndelay**
  
  Synaptic delay.

10.36.1 Detailed Description

The base class of a synapse.

This abstract base class provides several interfaces for a synapse. A synapse itself behaves as an action of an event; i.e. when activated at a certain simulation time, a synapse updates the destination (post-synaptic) neuron.

Definition at line 217 of file dneuron.h.
10.36.2 Member Function Documentation

10.36.2.1 virtual void punnets_common::tsynapse_base::activate (tscheduler & scheduler, ntime_t current_time) [pure virtual]

The event handler. When the source (pre-synaptic) neuron fires, this function is called after the synaptic delay. A derived class should redefine this method to update the destination (post-synaptic) neuron.

Implements punnets_common::taction (p. 69).

Implemented in punnets_private::tsynapse< debug > (p. 90), punnets_private::tsynapse_message< debug > (p. 98), punnets_private::tsynapseFatigue< debug > (p. 97), punnets_private::tsynapse_addfunc< debug > (p. 92), and punnets_private::tsynapse_messagefunc< debug > (p. 101).

The documentation for this class was generated from the following file:

- dneuron.h
10.37 punnets_private::tsynapse_fatigue< debug >  Class Template Reference

#include <dneuron.h>

Inheritance diagram for punnets_private::tsynapse_fatigue< debug >::<

Public Methods

- **tsynapse_fatigue** (tneuron_base &isrc, tneuron_base &idest, real idelay, real iweight)
  
  Construct a synapse with the specified source, destination, delay and initial weight.

- **tsynapse_fatigue** (tneuron_base &idest, real idelay, real iweight)
  
  virtual real getWeight () const
  
  Get the weight value of this synapse.

- virtual tneuron_base & getSrc () const
  
  Get the source neuron (pre-synaptic).

- virtual tneuron_base & getDest () const
  
  Get the destination neuron (post-synaptic).

- void addWeight (real delta_w)
  
  Modify the weight value with the specified step size.

- void addDelay (ntime_t delta_d)
  
  Modify the delay with the specified step size.

- virtual void activate (tscheduler &scheduler, ntime_t current_time)

- virtual const char * getClassName () const
  
  Obtains the class name of this neuron.

Static Protected Attributes

- const ntime_t recover_hv_period = 2000
  
  The time of half-value period of the synapse weight decrease.

- const real fire_ratio = 0.002
  
  The ratio of the decay of the weight caused by one pulse deliver.
10.37 punnets_private::tsynapse_fatigue< debug > Class Template Reference

10.37.1 Detailed Description

template<bool debug> class punnets_private::tsynapse_fatigue< debug >

The "fatigue" synapse class. This class of synapses delivers a pulse to the destination (postsynaptic) neuron, whose strength decays on bursting (by lack of energy) and recovers gradually. This is a sample for constructing more complex synapse class.

Definition at line 363 of file dneuron.h.

10.37.2 Constructor & Destructor Documentation

10.37.2.1 template<bool debug> punnets_private::tsynapse_fatigue< debug >::tsynapse_fatigue (tneuron_base & idest, real idelay, real iweight)
[inline]

Construct a synapse with the specified destination, delay and initial weight. Source neuron will be specified after the construction via setSrc() (p. 94) method.

Definition at line 383 of file dneuron.h.

References punnets_common::real.

384 : tsynapse_base(idelay), src(NULL), dest(&idest), weight(iweight / fire_ratio), last_fire( -Infinity ), last_weight

10.37.3 Member Function Documentation

10.37.3.1 template<bool debug> virtual void punnets_private::tsynapse_fatigue< debug >::activate (tscheduler & scheduler, ntime_t current_time)
[inline, virtual]

The event handler delivers a pulse to the destination (postsynaptic) neuron. Then the last firing time is recorded to process the fatiguefulness.

Implements punnets_common::tsynapse_base (p. 95).

Definition at line 401 of file dneuron.h.

References punnets_common::ntime_t, and punnets_common::real.

402 {  
403     real this_weight = weight + (last_weight - weight) * exp( (current_time - last_fire) * (-M_LN2 / recover_hv_period) )  
404     if( getDeb() )  
405         std::cout << std::setw(9) << std::setiosflags(std::ios::fixed) << std::setprecision(debug_precision) << current_time << " Pulse Arrived from " << src->getName() << "( " << this_weight * fire_ratio << " / " << weight << " )" << std::endl;  
406         dest->pulseArrive( scheduler, current_time, this_weight * fire_ratio );  
407     last_fire = current_time;  
408     last_weight = this_weight * (1.0 - fire_ratio);  
409 }  

The documentation for this class was generated from the following file:

* dneuron.h

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
10.38 punnets_private::tsynapse_message< debug > Class Template Reference

#include <dneuron.h>
Inheritance diagram for punnets_private::tsynapse_message< debug >:

punnets_common::taction
   
   punnets_common::tsynapse_base
   
   punnets_private::tsynapse_message< debug >

Public Methods

- **tsynapse_message** (tneuron_base &isrc, tneuron_base &idest, real idelay, message_base *imess)
- **tsynapse_message** (tneuron_base &idest, real idelay, message_base *imess)
- virtual ~**tsynapse_message** ()

  Destructor deletes the message.

- virtual void **setSrc** (tneuron_base &isrc)

  Specify the source neuron.

- virtual tneuron_base & **getSrc** () const

  Get the source neuron (pre-synaptic).

- virtual tneuron_base & **getDest** () const

  Get the destination neuron (post-synaptic).

- virtual void **activate** (tscheduler &scheduler, ntime_t current_time)

  The event handler delivers a message to the destination (post-synaptic) neuron.

10.38.1 Detailed Description

template<bool debug> class punnets_private::tsynapse_message< debug >

The message synapse class. This class of synapses has an effect for the destination (post-synaptic) neuron to deliver a message. The effect of the message can be arbitrarily specified.

Definition at line 317 of file dneuron.h.
10.38.2 Constructor & Destructor Documentation

10.38.2.1 template<

Condition::

Class Template Reference 99

Construct a synapse with the specified source, destination, delay and a pointer to a message. The pointer to the message will be deleted at the destruction of this synapse.

Definition at line 327 of file dneuron.h.

References punnets::common::real.

328 : tsynapse_base(idelay), src(&isrc), dest(&idest), mess(imess) {}

10.38.2.2 template<

Condition::

Class Template Reference 99

Construct a synapse with the specified destination, delay and a pointer to a message. Source neuron will be specified after the construction via setSrc() (p. 98) method. The pointer to the message will be deleted at the destruction of this synapse.

Definition at line 332 of file dneuron.h.

References punnets::common::real.

333 : tsynapse_base(idelay), src(NULL), dest(&idest), mess(imess) {}

The documentation for this class was generated from the following file:

- dneuron.h
10.39 punnets_private::tsynapse_messfunc< debug > Class Template Reference

#include <dneuron.h>

Inheritance diagram for punnets_private::tsynapse_messfunc< debug >:

```
punnets_common::taction

punnets_common::tsynapse_base

punnets_private::tsynapse_messfunc< debug >
```

**Public Methods**

- **tsynapse_messfunc** (tneuron_base &isrc, tneuron_ext< debug > &idest, real idelay, func_base *ifunc, message_base *imess, real ilev=0.0)
- **tsynapse_messfunc** (tneuron_ext< debug > &idest, real idelay, func_base *ifunc, message_base *imess, real ilev=0.0)
- **virtual void setSrc** (tneuron_base &isrc)
  
  *Set the source neuron (presynaptic) of this synapse. A derived class should redefine this method.*

- **virtual tneuron_base & getSrc** () const
  
  *Get the source neuron (pre-synaptic). A derived class should redefine this method.*

- **virtual tneuron_ext< debug > & getDest** () const
  
  *Get the destination neuron (post-synaptic). A derived class should redefine this method.*

- **virtual void activate** (tscheduler &scheduler, ntime t current_time)
- **virtual tqueue * queue** () const
  
  *Obtain a local event queue of this action.*

### 10.39.1 Detailed Description

template<bool debug> class punnets_private::tsynapse_messfunc< debug >

Synapse class that sends a message to a function in a tneuron_ext (p. 81) neuron. It is useful for applying an effect of func_expdiff, on which several firing can be calculated by one function and message_add_event_time message.

Definition at line 779 of file dneuron.h.
10.39.2 Constructor & Destructor Documentation

10.39.2.1 template<bool debug> punnets_private::tsynapseMessfunc< debug >::tsynapseMessfunc (tneuron_base & isrc, tneuron_ext< debug > & idest, real idelay, func_base * ifunc, message_base * imess, real ilev = 0.0) [inline]

Constructs add-function synapse with the specified source, destination, delay, a pointer to a function, a pointer to a message that is sent for the function at every activation, and immediate pulse level. In the constructor the function is added to the destination neuron.

Definition at line 792 of file dneuron.h.

References punnets_private::tneuron_ext< debug >::addExt(), and punnets_common::real.

793 : tsynapse_base(idelay), src(&isrc), dest(&idest), func(ifunc), mess(imess), lev(ilev) { dest->addExt(func); }

10.39.2.2 template<bool debug> punnets_private::tsynapseMessfunc< debug >::tsynapseMessfunc (tneuron_ext< debug > & idest, real idelay, func_base * ifunc, message_base * imess, real ilev = 0.0) [inline]

Constructs add-function synapse with the specified destination, delay, a pointer to a function, a pointer to a message that is sent for the function at every activation, and immediate pulse level. In the constructor the function is added to the destination neuron.

Definition at line 797 of file dneuron.h.

References punnets_private::tneuron_ext< debug >::addExt(), and punnets_common::real.

798 : tsynapse_base(idelay), src(NULL), dest(&idest), func(ifunc), mess(imess), lev(ilev) { dest->addExt(func); }

10.39.3 Member Function Documentation

10.39.3.1 template<bool debug> virtual void punnets_private::tsynapseMessfunc< debug >::activate (tscheduler & scheduler, ntime_t current_time) [inline, virtual]

When activated, the synapse sends the message to the function and sends the pulse to the destination neuron (to notice the change of the function).

Implements punnets_common::tsynapse_base (p. 95).

Definition at line 809 of file dneuron.h.

References punnets_common::tneuron_base::getName(), punnets_common::ntime_t, and punnets_private::tneuron_ext< debug >::pulseArrive().

810 {
811   if( getDeb() )
812     std::cout << std::setw(9) << std::setiosflags(std::ios::fixed) << std::setprecision(debug_precision) << current_time << " " << dest->getName() << " Pulse Arrived (func) from " << src->getName() << std::endl;
813   func->processMessage( current_time, *mess );
814   // cout << f->getDescription() << endl;
815   dest->pulseArrive( scheduler, current_time, lev );
816 }

The documentation for this class was generated from the following file:

• dneuron.h

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
Chapter 11

Punnets File Documentation

11.1 dlanguage.cpp File Reference

Language simulation based on the punnets library.

```cpp
#include <stdlib.h>
#include <sstream>
#include <fstream>
#include <iomanip>
#include <sys/types.h>
#include <sys/wait.h>
#include <unistd.h>
#include <math.h>
#include <string>
#include <mak/hash_string.h>
#include "punnets.h"
```

Namespaces

- namespace __gnu_cxx
- namespace mak
- namespace std

Compounds

- struct neuinfo
- struct word_t

11.1.1 Detailed Description

Language simulation based on the punnets library.
See the paper “A Pulsed Neural Network for Language Understanding: Discrete-Event Simulation of a Short-Term Memory Mechanism and Sentence Understanding” for details. In short, we assign one representer neuron (\textit{exts}) to represent each word. Once fired, a representer neuron periodically fires to keep short-term memory of the word. Representer neurons are interconnected via two networks, named autoassociative network (\textit{sgates}) and heteroassociative network (\textit{dgates}). When the network receives several words in a sequence, the two networks computes the bindings of the words, and as a result, the meaning of the input sentence is represented by the activation patter of the representer neurons (synchronized neurons have been bound). The connection weights of the two networks are obtained from the external files.

\textbf{Author:}
Makino, Takaki <t-makino-punnets01@snowelm.com>

\textbf{Date:}
2003-05-01

\textbf{Version:}

\textbf{Id:}
dlanguage.cpp,v 1.7 2003/05/08 08:23:56 t Exp

Copyright (C) 2003 Makino, Takaki.
This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.
This program is distributed in the hope that it will be useful, but \textit{WITHOUT ANY WARRANTY}; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
Definition in file \textit{dlanguage.cpp}.

\subsection{Variable Documentation}

\subsection*{11.1.2.1 const string posstrs[]}\textit{\[]}

\textbf{Initial value:}

\begin{verbatim}
{ "pronoun 1st", "pronoun 3rd", "noun", "proper noun", "vi", "vt", "vt_subj", "det" }
\end{verbatim}

Definition at line 104 of file dlanguage.cpp.
11.2 dlogger.cpp File Reference

Logger.

```cpp
#include "dlogger.h"
#include <iomanip>
```

Namespaces

- namespace punnets_common

11.2.1 Detailed Description

Logger.

Author:

Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:

2003-05-01

Version:

Id:

dlogger.cpp,v 1.1 2003/05/01 10:57:43 t Exp

Copyright (C) 2003 Makino, Takaki.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file dlogger.cpp.
11.3  dlogger.h File Reference

Activation logging class.
#include "dneuron.h"
#include <mak/infinity.h>

Namespaces

- namespace punnets_common

11.3.1 Detailed Description

Activation logging class.

Author:
Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
2003-05-01

Version:

Id:
  dlogger.h,v 1.2 2003/05/08 07:24:56 t Exp

Copyright (C) 2003 Makino, Takaki.
This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.
This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
Definition in file dlogger.h.
11.4 dneuron.cpp File Reference

Neurons.
#include "dneuron.h"
#include <math.h>
#include <mak/profile.h>

Namespaces

- namespace punnets_common
- namespace punnets_private

11.4.1 Detailed Description

Neurons.

Author:
Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
2003-05-01

Version:

Id:
dneuron.cpp,v 1.7 2003/05/07 09:32:23 t Exp

Copyright (C) 2003 Makino, Takaki.
This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file dneuron.cpp.
11.5  dneuron.h File Reference

Neuron/Synapse class in discrete-event NN simulation.

```c
#include <vector>
#include <map>
#include <string>
#include <algorithm>
#include <math.h>
#include <queue>
#include <iostream>
#include <iomanip>
#include "dsched.h"
#include "func.h"
```

Namespaces

- namespace punnets
- namespace punnets_common
- namespace punnets_nodebug
- namespace punnets_private

11.5.1  Detailed Description

Neuron/Synapse class in discrete-event NN simulation.

**Author:**
Makino, Takaki <t-makino-punnets01@snowelm.com>

**Date:**
2003-05-01

**Version:**

**Id:**

dneuron.h,v 1.6 2003/05/08 07:24:56 t Exp

Copyright (C) 2003 Makino, Takaki.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file dneuron.h.
11.6 dsched.cpp File Reference

Event, Action, Scheduler.

#include "dsched.h"

Namespaces

- namespace punnets_common

11.6.1 Detailed Description

Event, Action, Scheduler.

Author:
Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
2003-05-01

Version:

Id:
dsched.cpp,v 1.1 2003/05/01 10:57:43 t Exp

Copyright (C) 2003 Makino, Takaki.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file dsched.cpp.
11.7 dsched.h File Reference

Distributed scheduler.

```c
#include <math.h>
#include <functional>
#include <map>
#include <set>
#include <vector>
#include <queue>
#include <string>
#include "punnets_base.h"
```

Namespaces

- namespace punnets_common

11.7.1 Detailed Description

Distributed scheduler.

**Author:**

Makino, Takaki <t-makino-punnets01@snowelm.com>

**Date:**

2003-05-01

**Version:**

**Id:**

dsched.h,v 1.2 2003/05/08 07:24:56 t Exp

Copyright (C) 2003 Makino, Takaki. All rights reserved.

Definition in file dsched.h.
11.8 dtest.cpp File Reference

Test program of the punnets library.

```cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include "dneuron.h"
#include "dsched.h"
#include "dlogger.h"
#include <mak/cmdopt.h>
```

Compounds

- class tobserver

11.8.1 Detailed Description

Test program of the punnets library.

This program tests the performance of delayed firing simulations. Every neuron has an sinusoidal external input and interconnecting synapses, which causes alpha-function-style (difference of two exponential functions) response. Two hundred pulses are injected to produce initial activity. You can change parameters by command-line options; try `dtest -h` to show the list of options.

Author:
Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
2003-05-01

Version:

Id:
dtest.cpp,v 1.7 2003/05/08 08:23:56 t Exp

Copyright (C) 2003 Makino, Takaki.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file `dtest.cpp`.

---

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
11.9 func.cpp File Reference

Functions.
#include "func.h"
#include <iostream>

Namespaces
  • namespace punnets_common

11.9.1 Detailed Description

Functions.

Author:
Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
2003-05-01

Version:

Id:
func.cpp,v 1.3 2003/05/02 09:14:21 t Exp

Copyright (C) 2003 Makino, Takaki.
This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file func.cpp.
11.10 func.h File Reference

Function representation.

#include <math.h>
#include <string>
#include <iostream>
#include <sstream>
#include <mak/infinity.h>
#include "punnets_base.h"

Namespaces

• namespace punnets_common

11.10.1 Detailed Description

Function representation.

Author:
Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
2003-05-01

Version:

Id:
func.h,v 1.2 2003/05/02 09:13:05 t Exp

Copyright (C) 2003 Makino, Takaki.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file func.h.
11.11 punnets.h File Reference

Punnets header file.

#include "config_punnets.h"
#include "func.h"
#include "dsched.h"
#include "dlogger.h"
#include "dneuron.h"

11.11.1 Detailed Description

Punnets header file.

Author:
   Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
   2003-05-01

Version:

Id:
   punnets.h,v 1.4 2003/05/08 08:23:56 t Exp

Copyright (C) 2003 Makino, Takaki.

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Definition in file punnets.h.
11.12 punnets_base.h File Reference

Base class of Punnets.

#include "config_punnets.h"

Namespaces

- namespace punnets
- namespace punnets_common
- namespace punnets_nodebug
- namespace punnets_private

11.12.1 Detailed Description

Base class of Punnets.

Author:
Makino, Takaki <t-makino-punnets01@snowelm.com>

Date:
2003-05-01

Version:

Id:
punnets_base.h,v 1.2 2003/05/08 07:24:56 t Exp

Copyright (C) 2003 Makino, Takaki. All rights reserved.
Definition in file punnets_base.h.
Chapter 12

Punnets Page Documentation

12.1 Todo List

Class punnets_common::func_exp_int implement get1stDerivDomain etc. for this function

Class punnets_common::func_response document this function

Class punnets_common::taction write more
Index

queue
  punnets_common::tneuron_base, 80
~tsynapse_message
  punnets_private::tsynapse_message, 98
activate
  punnets_common::taction, 69
  punnets_common::tevent, 71
  punnets_common::tlogger, 72
  punnets_common::tsentinel, 89
  punnets_common::tsynapse_base, 95
  punnets_private::tneuron, 76
  punnets_private::tneuron_ext, 82
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
add
  punnets_common::tlogger, 72
addDelay
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
addExt
  punnets_private::tneuron_ext, 81
addSynapse
  punnets_common::tneuron_base, 79
  punnets_private::tneuron, 76
  punnets_private::tneuron_ext, 81
addWeight
  punnets_private::tsynapse, 90
  punnets_private::tsynapse, 90
broadcastMessage
  punnets_private::tneuron_ext, 82
calcSignal
  punnets_private::tneuron_ext, 82
clearNPulses
  toobserver, 87
clone
  punnets_common::func_base, 33
  punnets_common::func_const, 36
func.cpp, 112
func.h, 113
func_const
  punnets_common::tsynapse_base, 95
  punnets_private::tneuron, 76
  punnets_private::tneuron_ext, 81
coeff_sigdecay
  punnets_private::tneuron, 77
coeff_thrdecay
  punnets_private::tneuron, 77
debug
  punnets_common::debugflag< true >,
    32
debugflag
  punnets_common::debugflag< true >,
    32
dlanguage.cpp, 103
dlanguage.h, 104
dlogger.cpp, 105
dlogger.h, 106
dneuron.cpp, 107
dneuron.h, 108
dsched.cpp, 109
dsched.h, 110
dtest.cpp, 111
epsilon
  punnets_common, 25
eraseSynapse
  punnets_private::tneuron, 76
  punnets_private::tneuron_ext, 81
exts
  punnets_private::tneuron_ext, 82
fire
  punnets_private::tneuron_ext, 82
fire_ratio
  punnets_private::tsynapse_base, 95
epsilon
  punnets_common, 25
eraseSynapse
  punnets_private::tneuron, 76
  punnets_private::tneuron_ext, 81
exts
  punnets_private::tneuron_ext, 82
fire
  punnets_private::tneuron_ext, 82
fire_ratio
  punnets_private::tsynapse_base, 95
punnets::common::func::const, 36
func::const
  punnets::common::func::const, 38
func::delta
  punnets::common::func::delta, 40
func::exp
  punnets::common::func::exp, 46
func::exp::diff
  punnets::common::func::exp::diff, 48
func::exp::int
  punnets::common::func::exp::int, 51
func::sine
  punnets::common::func::sine, 55
func::sine::int
  punnets::common::func::sine::int, 57
func::sineshot
  punnets::common::func::sineshot, 59
func::sineshot::int
  punnets::common::func::sineshot::int, 63
get1stDeriv
  punnets::common::func::base, 33
  punnets::common::func::const, 36
  punnets::common::func::const::int, 38
  punnets::common::func::exp, 46
  punnets::common::func::exp::diff, 48
  punnets::common::func::exp::int, 51
get1stDerivDomain
  punnets::common::func::base, 33
  punnets::common::func::const, 36
  punnets::common::func::const::int, 38
  punnets::common::func::exp, 46
  punnets::common::func::exp::diff, 48
  punnets::common::func::exp::int, 51
getAction
  punnets::common::tevent, 71
getClassName
  punnets::common::taction, 69
  punnets::common::tlogger, 72
  punnets::common::tsynapse, 94
getConvergeLevel
  punnets::private::tneuron::ext::const, 84
getCurrentExtInput
  punnets::common::tneuron::base, 79
  punnets::private::tneuron::ext::const, 84
gGetCurrentSigLevel
  punnets::common::tneuron::base, 79
  punnets::private::tneuron, 76
gGetCurrentThrLevel
  punnets::common::tneuron::base, 79
  punnets::private::tneuron::ext, 81
getDeb
  punnets::common::debugflag< false >, 31
  punnets::common::debugflag< true >, 32
gGetDelay
  Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
getDest
punnets\_common::tsynapse\_base, 94
punnets\_private::tsynapse, 90
punnets\_private::tsynapse\_addfunc, 92
punnets\_private::tsynapse\_fatigue, 96
punnets\_private::tsynapse\_message, 98
punnets\_private::tsynapse\_messfunc, 100
getExtInput
punnets\_private::tneuron\_ext\_const, 84
g getLastFire
punnets\_common::tneuron\_base, 79
punnets\_private::tneuron, 76
punnets\_private::tneuron\_ext, 81
g getLastSimulate
punnets\_common::tneuron\_base, 79
punnets\_private::tneuron, 76
punnets\_private::tneuron\_ext, 81
g getLastSimulateType
punnets\_common::tneuron\_base, 79
punnets\_private::tneuron, 76
punnets\_private::tneuron\_ext, 81
g getMaxGradient
punnets\_common::func\_base, 33
punnets\_common::func\_const, 36
punnets\_common::func\_const\_int, 38
punnets\_common::func\_delta\_int, 40
punnets\_common::func\_exp, 46
punnets\_common::func\_exp\_diff, 48
punnets\_common::func\_exp\_int, 51
punnets\_common::func\_response, 53
punnets\_common::func\_sine, 55
punnets\_common::func\_sine\_int, 57
punnets\_common::func\_sineshot, 59
punnets\_common::func\_sineshot\_int, 63
punnets\_common::func\_step, 65
g getMessageId
punnets\_common::func\_delta\_int::message\_add\_pulse, 42
punnets\_common::func\_deriveq\_base::message\_set\_lambda, 44
INDEX

punnets::func::exp, 46
punnets::func::exp_diff, 48
punnets::func::exp_int, 51
punnets::func::response, 53
punnets::func::sine, 55
punnets::func::sine_int, 57
punnets::func::sineshot, 60
punnets::func::sineshot_int, 63
punnets::func::step, 65
getWeight
  punnets::tsynapse::base, 94
  punnets::tsynapse::fatigue, 96
gnuplot
  punnets::tlogger, 73
isScheduled
  punnets::tsched::double, 88

lambda
  punnets::tneuron::ext, 83
last_fire
  punnets::tneuron::ext, 83
last_simulate
  punnets::tneuron::ext, 82
last_simulate_type
  punnets::tneuron::ext, 83
Logging (drawing a graph of neuron potentials), 15
logoption
  punnets::tlogger, 73
loopback
  punnets::tneuron::ext, 83
makePulse
  punnets::common, 24
max_threshold
  punnets::tneuron, 77
min_threshold
  punnets::tneuron, 77

name
  punnets::tneuron::base, 80
ndelay
  punnets::tsynapse::base, 94
Neurons, 18
ntime_t
  punnets::common, 25

posstrs
dlanguage.cpp, 104
processMessage
  punnets::func::base, 34
  punnets::func::delta_int, 40
  punnets::func::derivative_base, 43
  punnets::func::exp_diff, 49
  punnets::func::sineshot, 61
pulseArrive
  punnets::tneuron::base, 80
  punnets::tneuron, 76
  punnets::tneuron::ext, 81
tobserver, 87
pulses
  punnets::tneuron::ext, 82
punnets, 21
tneuron, 21
tneuron::ext, 21
tneuron const, 21
tsynapse, 21
tsynapse::addfunc, 21
tsynapse::fatigue, 21
tsynapse::message, 21
tsynapse::messfunc, 21
punnets.h, 114
punnets_base.h, 115
punnets::common, 23
epsilon, 25
makePulse, 24
ntime_t, 25
real, 27
setExtInput, 24
total::filtered_incontinuity, 25
total::filtered_maxgrad, 25
total::filtered_nextpulse, 25
totalfire, 24
total::partition, 24
total::peakenclosing, 24
totalpulse, 24
totalescheduled, 24
punnets::common::debugflag<false>, 31
getDeb, 31
setDeb, 31
punnets::common::debugflag<true>, 32
debug, 32
debugflag, 32
getDeb, 32
setDeb, 32
punnets::common::func::base, 33
cloner, 33
get1stDeriv, 33
get1stDerivDomain, 33
get2ndDeriv, 33
get2ndDerivDomain, 33
getAddress, 34
getAuto, 34
getMaxGradient, 33
getNextIncontinuity, 34
getValue, 33
getValueDomain, 33
processMessage, 34

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
setLambda, 33
setZeroPoint, 33
shouldDelete, 33
valueChange, 34
punnets::common::func const, 36
close, 36
func const, 36
get1stDeriv, 36
get1stDerivDomain, 36
get2ndDeriv, 36
get2ndDerivDomain, 36
description, 36
getMaxGradient, 36
getValue, 36
getValueDomain, 36
punnets::common::func const, 38
close, 38
func const, 38
get1stDeriv, 38
get1stDerivDomain, 38
get2ndDeriv, 38
get2ndDerivDomain, 38
description, 38
getMaxGradient, 38
getValue, 38
getValueDomain, 38
punnets::common::func delta int, 40
close, 41
func delta int, 40
get1stDeriv, 40
get1stDerivDomain, 40
get2ndDeriv, 40
get2ndDerivDomain, 40
description, 40
getMaxGradient, 40
getNextIncontinuity, 41
getValue, 41
getValueDomain, 40
processMessage, 40
punnets::common::func delta int::message, 40
  add pulse, 42
getMessageId, 42
punnets::common::func deriveq base, 43
  processMessage, 43
  setLambda, 43
  setZeroPoint, 43
  zeronChange, 43
punnets::common::func deriveq base::message set lambda, 44
getMessageId, 44
punnets::common::func deriveq base::message set zero point, 45
getMessageId, 45
punnets::common::func exp, 46
close, 46
func exp, 46
get1stDeriv, 46
get1stDerivDomain, 46
get2ndDeriv, 46
get2ndDerivDomain, 46
description, 46
getMaxGradient, 46
getNextIncontinuity, 47
getValue, 46
getValueDomain, 46
punnets::common::func exp_diff, 48
close, 48
func exp_diff, 48
get1stDeriv, 48
get1stDerivDomain, 48
get2ndDeriv, 48
get2ndDerivDomain, 48
description, 48
getMaxGradient, 48
getNextIncontinuity, 49
getValue, 48
getValueDomain, 48
processMessage, 49
punnets::common::func exp_diff::message, 50
  add event time, 50
  getId, 50
punnets::common::func exp_int, 51
close, 51
func exp_int, 51
get1stDeriv, 51
get1stDerivDomain, 51
get2ndDeriv, 51
get2ndDerivDomain, 51
description, 51
getMaxGradient, 51
getNextIncontinuity, 52
getValue, 51
getValueDomain, 51
punnets::common::func response, 53
close, 53
get1stDeriv, 53
get1stDerivDomain, 53
get2ndDeriv, 53
get2ndDerivDomain, 53
description, 53
getMaxGradient, 53
getNextIncontinuity, 54
getValue, 53
getValueDomain, 53
setZeroPoint, 54
punnets::common::func sine, 55
close, 55
func_sine, 55
get1stDeriv, 55
get1stDerivDomain, 55
get2ndDeriv, 55
get2ndDerivDomain, 55
getDescription, 55
getMaxGradient, 55
getNextIncontinuity, 56
getValue, 55
g getValueDomain, 55
punnets_common::func_sine_int, 57
c lone, 57
func_sine_int, 57
get1stDeriv, 57
get1stDerivDomain, 57
get2ndDeriv, 57
get2ndDerivDomain, 57
getDescription, 57
getMaxGradient, 57
getValue, 57
g getValueDomain, 57
punnets_common::func_sineshot, 59
c lone, 59
func_sineshot, 59
get1stDeriv, 59
get1stDerivDomain, 59
get2ndDeriv, 59
get2ndDerivDomain, 59
getDescription, 59
getMaxGradient, 59
getNextIncontinuity, 60
getValue, 59
g getValueDomain, 60
processMessage, 61
shouldDelete, 59
punnets_common::func_sineshot::message_set_t0, 62
g etMessageId, 62
punnets_common::func_sineshot::int, 63
c lone, 63
func_sineshot::int, 63
get1stDeriv, 63
get1stDerivDomain, 63
get2ndDeriv, 63
get2ndDerivDomain, 63
getDescription, 63
getMaxGradient, 63
getValue, 63
g getValueDomain, 63
punnets_common::func_step, 65
c lone, 65
get1stDeriv, 65
get1stDerivDomain, 65
get2ndDeriv, 65
g get2ndDerivDomain, 65
g etDescription, 65
getMaxGradient, 65
getNextIncontinuity, 66
getValue, 65
g getValueDomain, 65
punnets_common:::greater:::event, 67
punnets_common:::message:::base, 68
g etMessageId, 68
punnets_common:::taction, 69
activate, 69
g etClassName, 69
queue, 69
punnets_common:::tevent, 71
activate, 71
getAction, 71
g etTime, 71
tevent, 71
punnets_common:::tlogger, 72
activate, 72
add, 72
g etClassName, 72
gnuPlot_def, 73
logoption, 73
queue, 72
schedule, 72
showext, 73
shownone, 73
showpart, 73
showthr, 73
tlogger, 72
punnets_common:::tneuron:::base, 79
ueue, 80
addSynapse, 79
g etClassName, 80
g etCurrentExtInput, 79
g etCurrentSigLevel, 79
g etCurrentThrLevel, 79
g etLastFire, 79
g etLastSimulate, 79
g etLastSimulateType, 79
getName, 79
name, 80
pulseArrive, 80
queue, 79
tneuron_base, 79
n schedule_double, 88
isScheduled, 88
run, 88
scheduleEvent, 88
tsched_double, 88
punnets_common:::tsentinel, 89
activate, 89
g etClassName, 89

Generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
queue, 89
punnets::common::tsynapse::base, 94
activate, 95
getClassName, 94
getDelay, 94
getDest, 94
getSrc, 94
getWeight, 94
ndelay, 94
setSrc, 94
tsynapse::base, 94

punnets::nodebug, 29
tneuron, 29
tneuron_ext, 29
tneuron_ext::const, 29
tsynapse, 29
tsynapse::addfunc, 29
tsynapse::fatigue, 29
tsynapse::message, 29
tsynapse::messfunc, 29

punnets::private, 30
punnets::private::tneuron, 76
activate, 76
addSynapse, 76
coe::sigdecay, 77
coe::thrdecay, 77
eraseSynapse, 76
getClassName, 76
getCurrentSigLevel, 76
getCurrentThrLevel, 76
getLastFire, 76
getLastSimulate, 76
getLastSimulateType, 76
getSynapses, 76
lambda, 77
last_fire, 77
last_simulate, 77
last_simulate_type, 77
loopback, 77
pulseArrive, 77
pulses, 77
queue, 77
scheduleFire, 77
sendMessage, 77
setLoopBack, 77
synapses, 78
tneuron_ext, 78

punnets::private::tneuron::ext, 81
activate, 81
addExt, 81
addSynapse, 81
broadcastMessage, 81
calcSignal, 82
eraseSynapse, 81
eexts, 82
fire, 82
getClassName, 82
getCurrentSigLevel, 82

getClassName, 84
getCurrentExtInput, 84
getCurrentSigLevel, 84
getExtInput, 84
scheduleFire, 84
setConvergeLevel, 85
setExtInput, 85
sig_converge_level, 85
simulateElapse, 85
tneuron_ext::const, 85

punnets::private::tsynapse, 90
activate, 90
addDelay, 90
addWeight, 90
getDest, 90
getSrc, 90
getWeight, 90
queue, 90
setSrc, 90
tsynapse, 90, 91

punnets::private::tsynapse::addfunc, 92
activate, 92
getDest, 92
getSrc, 92
queue, 92
setSrc, 92
tsynapse::addfunc, 93

punnets::private::tsynapse::fatigue, 96
activate, 97
addDelay, 97
addWeight, 97
fire_ratio, 97
getClassName, 97
getDest, 97

generated on Mon Jun 16 15:42:30 2003 for Punnets by Doxygen
getSrc, 96
getWeight, 96
recover_hv_period, 96
tsynapse_fatigue, 96, 97
punnets::tsynapse::message, 98
~tsynapse::message, 98
activate, 98
getDest, 98
getSrc, 98
setSrc, 96
tsynapse::message, 99
punnets::tsynapse::message, 100
activate, 101
getDest, 100
getSrc, 100
queue, 100
setSrc, 100
synapse::message, 101
queue
  punnets::common::taction, 69
  punnets::common::tlogger, 72
  punnets::common::tneuron::base, 79
  punnets::common::tsentinel, 89
  punnets::private::tneuron, 76
  punnets::private::tneuron::ext, 82
  punnets::private::tsynapse, 90
  punnets::private::tsynapse::addfunc, 92
  punnets::private::tsynapse::messfunc, 100
real
  punnets::common, 27
recover_hv_period
  punnets::private::tsynapse::fatigue, 96
run
  punnets::common::tsched::double, 88
schedule
  punnets::common::tlogger, 72
scheduleEvent
  punnets::common::tsched::double, 88
scheduleFire
  punnets::private::tneuron, 77
  punnets::private::tneuron::ext, 82
  punnets::private::tneuron::ext::const, 84
Scheduling, 20
sendMessage
  punnets::private::tneuron::ext, 82
setConvergeLevel
  punnets::private::tneuron::ext::const, 85
setDebug
  punnets::common::debugflag< true >, 32
setExtInput
  punnets::common, 24
  punnets::private::tneuron::ext::const, 84
setLambda
  punnets::common::func::base, 33
  punnets::common::func::deriveq::base, 43
setLoopBack
  punnets::private::tneuron::ext, 83
setSrc
  punnets::common::tsynapse::base, 94
  punnets::private::tsynapse, 90
  punnets::private::tsynapse::addfunc, 92
  punnets::private::tsynapse::message, 98
  punnets::private::tsynapse::messfunc, 100
setZeroPoint
  punnets::common::func::base, 33
  punnets::common::func::deriveq::base, 43
  punnets::common::func::response, 54
shouldDelete
  punnets::common::func::base, 33
  punnets::common::func::sineshot, 59
showext
  punnets::common::tlogger, 73
shownone
  punnets::common::tlogger, 73
showpart
  punnets::common::tlogger, 73
showthr
  punnets::common::tlogger, 73
sig::converge::level
  punnets::private::tneuron, 77
  punnets::private::tneuron::ext::const, 85
sig::hv_period
  punnets::private::tneuron, 77
simulateElapse
  punnets::private::tneuron, 77
  punnets::private::tneuron::ext::const, 85
Synapses, 19
synapses
  punnets::private::tneuron::ext, 82
tevent
  punnets::common::tevent, 71
thr::hv_period
  punnets::private::tneuron, 77
tlogger
  punnets::common::tlogger, 72
tneuron
  punnets, 21
  punnets::nodebug, 29
  punnets::private::tneuron, 78
tneuron::base
punnett_common::tneuron_base, 79

tneuron_ext
punnett, 21
punnett_nodebug, 29
punnett_private::tneuron_ext, 81
tneuron_ext__const
punnett, 21
punnett_nodebug, 29
punnett_private::tneuron_ext__const, 85
tobserver, 87
clearNPulses, 87
getNPulses, 87
pulseArrive, 87
tobserver, 87
totalfiltered_incontinuity
punnett_common, 25
totalvars, 16
totalfiltered_maxgrad
punnett_common, 25
totalvars, 16
totalfiltered_nextpulse
punnett_common, 25
totalvars, 16
totalfire
punnett_common, 24
totalvars, 16
totalpartition
punnett_common, 24
totalvars, 16
totalpartition__newton
totalvars, 16
totalpartition__nonewton
totalvars, 16
totalpeakenclosing
punnett_common, 24
totalvars, 16
totalpeaksearch
totalvars, 16
totalpulse
punnett_common, 24
totalvars, 16
totalrescheduled
punnett_common, 24
totalvars, 16
totalvars
totalfiltered_incontinuity, 16
totalfiltered_maxgrad, 16
totalfiltered_nextpulse, 16
totalfire, 16
totalpartition, 16
totalpartition__newton, 16
totalpartition__nonewton, 16
totalpeakenclosing, 16
totalpeaksearch, 16
totalpulse, 16
totalrescheduled, 16
tsched_double
punnett_common::tsched_double, 88
tsynapse
punnett, 21
punnett_nodebug, 29
punnett_private::tsynapse, 90, 91
tsynapse_addfunc
punnett, 21
punnett_nodebug, 29
punnett_private::tsynapse_addfunc, 93
tsynapse_base
punnett_common::tsynapse_base, 94
tsynapse_fatigue
punnett, 21
punnett_nodebug, 29
punnett_private::tsynapse_fatigue, 96, 97
tsynapse_message
punnett, 21
punnett_nodebug, 29
punnett_private::tsynapse_message, 99
tsynapse_messfunc
punnett, 21
punnett_nodebug, 29
punnett_private::tsynapse_messfunc, 101
valueChange
punnett_common::func_base, 34
Variables for Statistics, 16
zeropChange
punnett_common::func__leriveq_base, 43