My Project

Generated by Doxygen 1.8.14

Contents

Chapter 1

Namespace Index

1.1 Namespace List

Here is a list of all namespaces with brief descriptions:

Chapter 2

Hierarchical Index

2.1 Class Hierarchy

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

Chapter 3

Class Index

3.1 Class List

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

Chapter 4

File Index

4.1 File List

Here is a list of all files with brief descriptions:

Chapter 5

Namespace Documentation

5.1 simlib Namespace Reference

Namespaces

- action
- anchor
- archspec
- FSM
- hub
- node
- simulated
- simulationenvironment

5.2 simlib.action Namespace Reference

Classes

- class Action

  Classes ##.

Functions

- def fn (n)
- def fn2 (x, y)

Variables

- int errors = 0
- test_obj = None

5.2.1 Detailed Description

```
@package Action
This module defines the Action object, which is used, together with the ActionQueue class,
to implement delayed function calls.
```

5.2.2 Function Documentation

5.2.2.1 fn()

```
def simlib.action.fn (
              n )
```

5.2.2.2 fn2()

```
def simlib.action.fn2 (
              x,
              y )
```

5.2.3 Variable Documentation

5.2.3.1 errors

```
int simlib.action.errors = 0
```

5.2.3.2 test_obj

```
simlib.action.test_obj = None
```

5.3 simlib.anchor Namespace Reference

Classes

- class Anchor

5.4 simlib.archspec Namespace Reference

Classes

- class ArchSpec

  Classes ##.
- class MySubclass

Variables

- int errors = 0
- archspec_obj = ArchSpec(int , int , int)

5.4.1 Detailed Description

```
@package Architecture Specification
This module presents the simulation environment with the types of the hub, nodes, and
anchors.
```

5.4.2 Variable Documentation

5.4.2.1 archspec_obj

```
simlib.archspec.archspec_obj = ArchSpec( int , int , int )
```

5.4.2.2 errors

```
int simlib.archspec.errors = 0
```

5.5 simlib.FSM Namespace Reference

Classes

- class Device
- class DW1000
- class State

Functions

- def getOffCurrent (physical_data)

  This is dumb but i cant think of a way to do this while making it # general to each device.
- def getDeepSleepCurrent (physical_data)
- def getSleepCurrent (physical_data)
- def getInitCurrent (physical_data)
- def getIdleCurrent (physical_data)
- def getRXCurrent (physical_data)
- def getTXCurrent (physical_data)
- def getTime (physical_data)
- def getBattery (dev_data, state_data)

Variables

- int OSC_STARTUP_SEND_CURRENT = 3
- int OSC_STARTUP_SEND_TIME = 2000
- int PLL_STARTUP_SEND_CURRENT = 12
- int PLL_STARTUP_SEND_TIME = 7
- int WR_TX_DATA_SEND_CURRENT = 15
- int WR_TX_DATA_SEND_TIME = 10
- int TX_SHR_SEND_CURRENT = 65
- int TX_SHR_SEND_TIME = 135
- int TX_PHR_SEND_CURRENT = 48
- float TX_PHR_SEND_TIME = 1.33
- bool DEBUG = True
- OFF_STATE = State({'current' : getOffCurrent, 'time' : getTime, 'state' : 'off_state'})

  FSM for the DW1000, includes states, thier delays, and the current for each state.
- DEEPSLEEP_STATE = State({'current' : getDeepSleepCurrent, 'time' : getTime, 'state' : 'deepsleep_state'})
- SLEEP_STATE = State({'current' : getSleepCurrent, 'time' : getTime, 'state' : 'sleep_state'})
- INIT_STATE = State({'current' : getInitCurrent, 'time' : getTime, 'state' : 'init_state'})
- IDLE_STATE = State({'current' : getIdleCurrent, 'time' : getTime, 'state' : 'idle_state'})
- RX_STATE = State({'current' : getRXCurrent, 'time' : getTime, 'state' : 'rx_state'})
- TX_STATE = State({'current' : getTXCurrent, 'time' : getTime, 'state' : 'tx_state'})
- INITIAL_STATE = OFF_STATE
- dictionary DEVICE_DATA = {'batt_life' : getBattery}
- dictionary AVAILABLE_STATES
- battlife
- device = DW1000(AVAILABLE_STATES, INITIAL_STATE, DEVICE_DATA)
- int errors = 0
- list simlist = [ ]
- my_current = device.getState().getParam('current')

5.5.1 Function Documentation

5.5.1.1 getBattery()

```
def simlib.FSM.getBattery (
              dev_data,
              state_data )
```

5.5.1.2 getDeepSleepCurrent()

```
def simlib.FSM.getDeepSleepCurrent (
              physical_data )
```

5.5.1.3 getIdleCurrent()

```
def simlib.FSM.getIdleCurrent (
              physical_data )
```

5.5.1.4 getInitCurrent()

```
def simlib.FSM.getInitCurrent (
              physical_data )
```

5.5.1.5 getOffCurrent()

```
def simlib.FSM.getOffCurrent (
              physical_data )
```

This is dumb but i cant think of a way to do this while making it # general to each device.

I won't be adding comments to these # functions because they will be replaced when a better solution # is found. #

5.5.1.6 getRXCurrent()

```
def simlib.FSM.getRXCurrent (
              physical_data )
```

5.5.1.7 getSleepCurrent()

```
def simlib.FSM.getSleepCurrent (
              physical_data )
```

5.5.1.8 getTime()

```
def simlib.FSM.getTime (
              physical_data )
```

5.5.1.9 getTXCurrent()

```
def simlib.FSM.getTXCurrent (
              physical_data )
```

5.5.2 Variable Documentation

5.5.2.1 AVAILABLE_STATES

```
dictionary simlib.FSM.AVAILABLE_STATES
```

Initial value:

```
=  {OFF_STATE : [(INIT_STATE, 3000)],
                        DEEPSLEEP_STATE : [(INIT_STATE, 3000)],
                        SLEEP_STATE : [(INIT_STATE, 3000)],
                        INIT_STATE : [(IDLE_STATE, 5)],
                        IDLE_STATE : [(RX_STATE, 0) , (TX_STATE, 0)],
                        RX_STATE : [(IDLE_STATE, 0) , (SLEEP_STATE, 0) , (DEEPSLEEP_STATE, 0)],
                        TX_STATE : [(IDLE_STATE, 0) , (SLEEP_STATE, 0) , (DEEPSLEEP_STATE, 0)]}
```

5.5.2.2 battlife

```
simlib.FSM.battlife
```

5.5.2.3 DEBUG

```
bool simlib.FSM.DEBUG = True
```

5.5.2.4 DEEPSLEEP_STATE

```
simlib.FSM.DEEPSLEEP_STATE = State({'current' :  getDeepSleepCurrent, 'time' :  getTime, 'state'
:  'deepsleep_state'})
```

5.5.2.5 device

```
simlib.FSM.device = DW1000(AVAILABLE_STATES, INITIAL_STATE, DEVICE_DATA)
```

5.5.2.6 DEVICE_DATA

```
dictionary simlib.FSM.DEVICE_DATA = {'batt_life' :  getBattery}
```

5.5.2.7 errors

```
int simlib.FSM.errors = 0
```

5.5.2.8 IDLE_STATE

```
simlib.FSM.IDLE_STATE = State({'current' :  getIdleCurrent, 'time' :  getTime, 'state' ↩
:  'idle_state'})
```

5.5.2.9 INIT_STATE

```
simlib.FSM.INIT_STATE = State({'current' :  getInitCurrent, 'time' :  getTime, 'state' ↩
:  'init_state'})
```

5.5.2.10 INITIAL_STATE

```
simlib.FSM.INITIAL_STATE = OFF_STATE
```

5.5.2.11 my_current

```
simlib.FSM.my_current = device.getState().getParam('current')
```

5.5.2.12 OFF_STATE

```
simlib.FSM.OFF_STATE = State({'current' : getOffCurrent, 'time' : getTime, 'state' : 'off←
_state'})
```

FSM for the DW1000, includes states, thier delays, and the current for each state.

5.5.2.13 OSC_STARTUP_SEND_CURRENT

```
int simlib.FSM.OSC_STARTUP_SEND_CURRENT = 3
```

5.5.2.14 OSC_STARTUP_SEND_TIME

```
int simlib.FSM.OSC_STARTUP_SEND_TIME = 2000
```

5.5.2.15 PLL_STARTUP_SEND_CURRENT

```
int simlib.FSM.PLL_STARTUP_SEND_CURRENT = 12
```

5.5.2.16 PLL_STARTUP_SEND_TIME

```
int simlib.FSM.PLL_STARTUP_SEND_TIME = 7
```

5.5.2.17 RX_STATE

```
simlib.FSM.RX_STATE = State({'current' : getRXCurrent, 'time' : getTime, 'state' : 'rx_←
state'})
```

5.5.2.18 simlist

```
list simlib.FSM.simlist = [ ]
```

5.5.2.19 SLEEP_STATE

```
simlib.FSM.SLEEP_STATE = State({'current' :  getSleepCurrent, 'time' :  getTime, 'state' ↩
:  'sleep_state'})
```

5.5.2.20 TX_PHR_SEND_CURRENT

```
int simlib.FSM.TX_PHR_SEND_CURRENT = 48
```

5.5.2.21 TX_PHR_SEND_TIME

```
float simlib.FSM.TX_PHR_SEND_TIME = 1.33
```

5.5.2.22 TX_SHR_SEND_CURRENT

```
int simlib.FSM.TX_SHR_SEND_CURRENT = 65
```

5.5.2.23 TX_SHR_SEND_TIME

```
int simlib.FSM.TX_SHR_SEND_TIME = 135
```

5.5.2.24 TX_STATE

```
simlib.FSM.TX_STATE = State({'current' :  getTXCurrent, 'time' :  getTime, 'state' :  'tx_↩
state'})
```

5.5.2.25 WR_TX_DATA_SEND_CURRENT

```
int simlib.FSM.WR_TX_DATA_SEND_CURRENT = 15
```

5.5.2.26 WR_TX_DATA_SEND_TIME

```
int simlib.FSM.WR_TX_DATA_SEND_TIME = 10
```

5.6 simlib.hub Namespace Reference

Classes

- class Hub

Variables

- hub = Hub(0)

5.6.1 Variable Documentation

5.6.1.1 hub

```
simlib.hub.hub = Hub(0)
```

5.7 simlib.node Namespace Reference

Classes

- class Node

5.8 simlib.simulated Namespace Reference

Classes

- class ActionQueue
- class Simulated

  Classes ##.

Variables

- bool DEBUG = True
- int errors = 0
- aq = ActionQueue( )
- action1 = Action(aq.test, ["I was the first action added"])
- action2 = Action(aq.test, ["then me (:"])
- action3 = Action(aq.test, ["lastly me (;"])
- int i = 0

5.8.1 Variable Documentation

5.8.1.1 action1

```
simlib.simulated.action1 = Action( aq.test, ["I was the first action added"] )
```

5.8.1.2 action2

```
simlib.simulated.action2 = Action( aq.test, ["then me (:"] )
```

5.8.1.3 action3

```
simlib.simulated.action3 = Action( aq.test, ["lastly me (;"] )
```

5.8.1.4 aq

```
simlib.simulated.aq = ActionQueue( )
```

5.8.1.5 DEBUG

```
bool simlib.simulated.DEBUG = True
```

5.8.1.6 errors

```
int simlib.simulated.errors = 0
```

5.8.1.7 i

```
int simlib.simulated.i = 0
```

5.9 simlib.simulationenvironment Namespace Reference

Classes

- class SimulationEnvironment

Functions

- def superSmartTrilaterationAlgorithm

Variables

- int errors = 0
- archspec = ArchSpec(Hub, Anchor, Node)
- simEnv = SimulationEnvironment()
- hub = simEnv.createHub(superSmartTrilaterationAlgorithm)
- anchor = simEnv.createAnchor(0, 0, i)
- node = simEnv.createNode(3, 3, 3, 0, 0, 0)

5.9.1 Function Documentation

5.9.1.1 superSmartTrilaterationAlgorithm()

```
def simlib.simulationenvironment.superSmartTrilaterationAlgorithm (
              mapDict )
```

5.9.2 Variable Documentation

5.9.2.1 anchor

```
simlib.simulationenvironment.anchor = simEnv.createAnchor(0, 0, i)
```

5.9.2.2 archspec

```
simlib.simulationenvironment.archspec = ArchSpec(Hub, Anchor, Node)
```

5.9.2.3 errors

```
int simlib.simulationenvironment.errors = 0
```

5.9.2.4 hub

```
simlib.simulationenvironment.hub = simEnv.createHub(superSmartTrilaterationAlgorithm)
```

5.9.2.5 node

```
simlib.simulationenvironment.node = simEnv.createNode(3, 3, 3, 0, 0, 0)
```

5.9.2.6 simEnv

```
simlib.simulationenvironment.simEnv = SimulationEnvironment()
```

Chapter 6

Class Documentation

6.1 simlib.action.Action Class Reference

Classes ##.

Public Member Functions

- def __init__
- def decrement
- def set_fn
- def set_ctr
- def get_fn (self)
- def get_args (self)
- def get_ctr (self)

6.1.1 Detailed Description

Classes ##.

```
Element of an action queue
```

6.1.2 Constructor & Destructor Documentation

6.1.2.1 __init__()

```
def simlib.action.Action.__init__ (
              self,
              fn_to_be_called )
```

6.1.3 Member Function Documentation

6.1.3.1 decrement()

```
def simlib.action.Action.decrement (
              self,
              n )
```

6.1.3.2 get_args()

```
def simlib.action.Action.get_args (
              self,
              list )
```

```
Returns the function args associated with this action.
```

6.1.3.3 get_ctr()

```
def simlib.action.Action.get_ctr (
              self,
              int )
```

```
Returns the counter value associated with this action.
```

6.1.3.4 get_fn()

```
def simlib.action.Action.get_fn (
              self,
              types,
              FunctionType )
```

```
Returns the function associated with this action.
```

6.1.3.5 set_ctr()

```
def simlib.action.Action.set_ctr (
              self,
              ctr_val )
```

6.1.3.6 set_fn()

```
def simlib.action.Action.set_fn (
              self,
              fn_to_be_called )
```

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

- action.py

6.2 simlib.simulated.ActionQueue Class Reference

Public Member Functions

- def __init__ (self)
- def addToQueue
- def popAction (self)
- def update (self)
- def test (self, test)

Public Attributes

- queue

6.2.1 Detailed Description

```
Creates instance of Action Queue
```

6.2.2 Constructor & Destructor Documentation

6.2.2.1 __init__()

```
def simlib.simulated.ActionQueue.__init__ (
              self )
```

6.2.3 Member Function Documentation

6.2.3.1 addToQueue()

```
def simlib.simulated.ActionQueue.addToQueue (
              self,
              newAction )
```

6.2.3.2 popAction()

```
def simlib.simulated.ActionQueue.popAction (
              self )
```

```
removes the oldest Action from the list
```

6.2.3.3 test()

```
def simlib.simulated.ActionQueue.test (
              self,
              test )
```

```
arbitrary function added for testing purposes
```

6.2.3.4 update()

```
def simlib.simulated.ActionQueue.update (
              self )
```

```
Every time the update function is called, decrement the counter of
the first action in the action queue. If the the counter is zero,
pop the action off the action queue
```

6.2.4 Member Data Documentation

6.2.4.1 queue

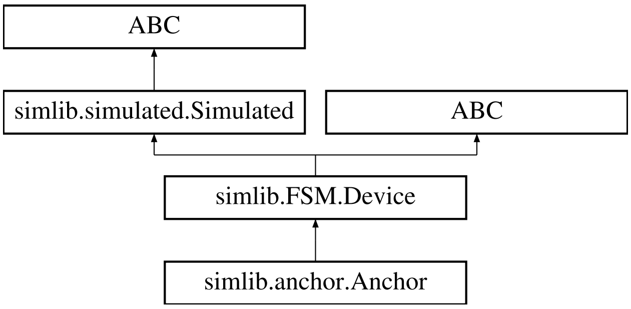
```
simlib.simulated.ActionQueue.queue
```

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

- simulated.py

6.3 simlib.anchor.Anchor Class Reference

Inheritance diagram for simlib.anchor.Anchor:



Public Member Functions

- def __init__
- def setTime
- def getID ()
- def listenForSignal ()
- def requestPing
- def pingNode
- def waitForReply
- def addAction
- def prependAction

Public Attributes

- time
- ID
- xPos
- yPos
- zPos
- pingedNodeID
- measurement
- signalList
- distance

6.3.1 Constructor & Destructor Documentation

6.3.1.1 __init__()

```
def simlib.anchor.Anchor.__init__ (
              self,
              ID )
```

6.3.2 Member Function Documentation

6.3.2.1 addAction()

```
def simlib.anchor.Anchor.addAction (
              self,
              function )
```

6.3.2.2 getID()

```
def simlib.anchor.Anchor.getID (
              int )
```

6.3.2.3 listenForSignal()

```
def simlib.anchor.Anchor.listenForSignal (
              int )
```

```
Searches to see if it is the target of any signals. Returns the sending ID
if a signal is found or 0 if no signal is found.
```

6.3.2.4 pingNode()

```
def simlib.anchor.Anchor.pingNode (
              self,
              args )
```

6.3.2.5 prependAction()

```
def simlib.anchor.Anchor.prependAction (
              self,
              function )
```

6.3.2.6 requestPing()

```
def simlib.anchor.Anchor.requestPing (
              self,
              node )
```

6.3.2.7 setTime()

```
def simlib.anchor.Anchor.setTime (
              time )
```

6.3.2.8 waitForReply()

```
def simlib.anchor.Anchor.waitForReply (
              self,
              args )
```

6.3.3 Member Data Documentation

6.3.3.1 distance

```
simlib.anchor.Anchor.distance
```

6.3.3.2 ID

```
simlib.anchor.Anchor.ID
```

6.3.3.3 measurement

```
simlib.anchor.Anchor.measurement
```

6.3.3.4 pingedNodeID

```
simlib.anchor.Anchor.pingedNodeID
```

6.3.3.5 signalList

```
simlib.anchor.Anchor.signalList
```

6.3.3.6 time

```
simlib.anchor.Anchor.time
```

6.3.3.7 xPos

```
simlib.anchor.Anchor.xPos
```

6.3.3.8 yPos

```
simlib.anchor.Anchor.yPos
```

6.3.3.9 zPos

```
simlib.anchor.Anchor.zPos
```

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

- anchor.py

6.4 simlib.archspec.ArchSpec Class Reference

Classes ##.

Public Member Functions

- def __init__
- def get_hubclass (self)
- def get_anchorclass (self)
- def get_nodeclass (self)
- def set_hubclass
- def set_anchorclass
- def set_nodeclass

6.4.1 Detailed Description

Classes ##.

```
Architecture specification
Defines what classes to use for hub, anchors, and nodes.


The default constructor
@param hubclass The type of the central hub
@param anchorclass The type of the anchor
@param nodeclass The type of the node
```

6.4.2 Constructor & Destructor Documentation

6.4.2.1 __init__()

```
def simlib.archspec.ArchSpec.__init__ (
              self,
              hubclass )
```

6.4.3 Member Function Documentation

6.4.3.1 get_anchorclass()

```
def simlib.archspec.ArchSpec.get_anchorclass (
              self,
              type )
```

6.4.3.2 get_hubclass()

```
def simlib.archspec.ArchSpec.get_hubclass (
              self,
              type )
```

6.4.3.3 get_nodeclass()

```
def simlib.archspec.ArchSpec.get_nodeclass (
              self,
              type )
```

6.4.3.4 set_anchorclass()

```
def simlib.archspec.ArchSpec.set_anchorclass (
              self,
              anchorclass )
```

6.4.3.5 set_hubclass()

```
def simlib.archspec.ArchSpec.set_hubclass (
              self,
              hubclass )
```

6.4.3.6 set_nodeclass()

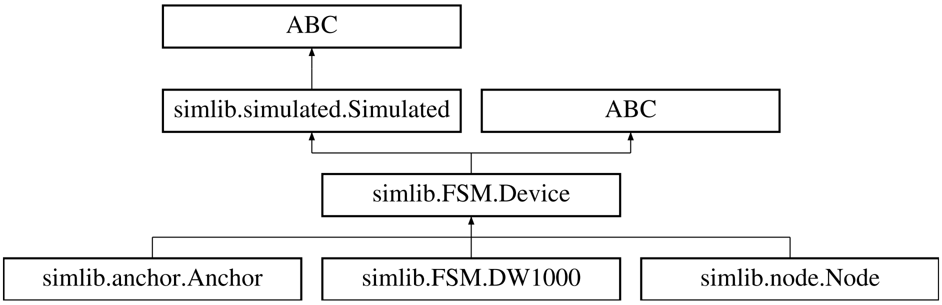
```
def simlib.archspec.ArchSpec.set_nodeclass (
              self,
              nodeclass )
```

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

- archspec.py

6.5 simlib.FSM.Device Class Reference

Inheritance diagram for simlib.FSM.Device:



Public Member Functions

- def __init__
- def getState (self)
- def setNextState
- def getParam

Public Attributes

- available_states
- initial_state
- dev_state
- physical_data
- next_states

6.5.1 Detailed Description

```
Creates an instance of a device dependent on the FSM,
physical data, and available states used to accurately
model the DW1000 or other like devices.
```

6.5.2 Constructor & Destructor Documentation

6.5.2.1 __init__()

```
def simlib.FSM.Device.__init__ (
              self,
              available_states )
```

6.5.3 Member Function Documentation

6.5.3.1 getParam()

```
def simlib.FSM.Device.getParam (
              self,
              param )
```

6.5.3.2 getState()

```
def simlib.FSM.Device.getState (
              self,
              State )
```

```
Returns the current state of the device.
```

6.5.3.3 setNextState()

```
def simlib.FSM.Device.setNextState (
              self,
              state )
```

6.5.4 Member Data Documentation

6.5.4.1 available_states

```
simlib.FSM.Device.available_states
```

6.5.4.2 dev_state

```
simlib.FSM.Device.dev_state
```

6.5.4.3 initial_state

`simlib.FSM.Device.initial_state`

6.5.4.4 next_states

`simlib.FSM.Device.next_states`

6.5.4.5 physical_data

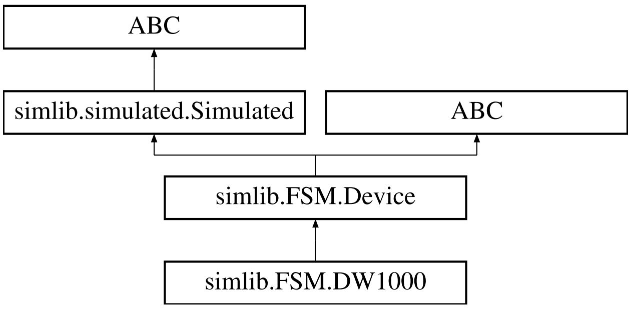
`simlib.FSM.Device.physical_data`

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

- FSM.py

6.6 simlib.FSM.DW1000 Class Reference

Inheritance diagram for simlib.FSM.DW1000:



Public Member Functions

- def __init__
- def mainloop

Additional Inherited Members

6.6.1 Constructor & Destructor Documentation

6.6.1.1 __init__()

```
def simlib.FSM.DW1000.__init__ (
              self,
              available_states )
```

6.6.2 Member Function Documentation

6.6.2.1 mainloop()

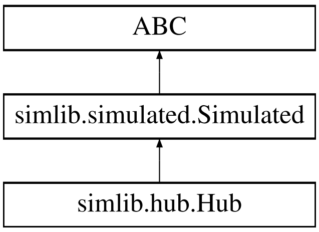
```
def simlib.FSM.DW1000.mainloop (
              self,
              simlist )
```

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

- FSM.py

6.7 simlib.hub.Hub Class Reference

Inheritance diagram for simlib.hub.Hub:



Public Member Functions

- def __init__ (self, algorithm)
- def setTime
- def containsAnchor
- def addAnchor
- def removeAnchor
- def containsNode
- def addNode
- def removeNode
- def resetMap (self)
- def mapDistance
- def getNodePosition
- def mapAnchorAndNode
- def triliterateNode
- def generateCompleteMap (self)
- def addAction
- def prependAction
- def mainloop ()

Public Attributes

- time
- anchors
- nodes
- map
- nodePositions
- algorithm

6.7.1 Constructor & Destructor Documentation

6.7.1.1 __init__()

```
def simlib.hub.Hub.__init__ (
              self,
              algorithm )
```

6.7.2 Member Function Documentation

6.7.2.1 addAction()

```
def simlib.hub.Hub.addAction (
              self,
              function )
```

6.7.2.2 addAnchor()

```
def simlib.hub.Hub.addAnchor (
              self,
              anchor )
```

6.7.2.3 addNode()

```
def simlib.hub.Hub.addNode (
              self,
              node )
```

6.7.2.4 containsAnchor()

```
def simlib.hub.Hub.containsAnchor (
              self,
              anchor )
```

6.7.2.5 containsNode()

```
def simlib.hub.Hub.containsNode (
              self,
              node )
```

6.7.2.6 generateCompleteMap()

```
def simlib.hub.Hub.generateCompleteMap (
              self )
```

```
Generates a complete map of anchors and nodes
```

6.7.2.7 getNodePosition()

```
def simlib.hub.Hub.getNodePosition (
              self,
              node )
```

6.7.2.8 mainloop()

```
def simlib.hub.Hub.mainloop ( )
```

6.7.2.9 mapAnchorAndNode()

```
def simlib.hub.Hub.mapAnchorAndNode (
              self,
              args )
```

6.7.2.10 mapDistance()

```
def simlib.hub.Hub.mapDistance (
              self,
              xVal )
```

6.7.2.11 prependAction()

```
def simlib.hub.Hub.prependAction (
              self,
              function )
```

6.7.2.12 removeAnchor()

```
def simlib.hub.Hub.removeAnchor (
              self,
              anchor )
```

6.7.2.13 removeNode()

```
def simlib.hub.Hub.removeNode (
              self,
              node )
```

6.7.2.14 resetMap()

```
def simlib.hub.Hub.resetMap (
              self )
```

```
Resets the map to [][]
```

6.7.2.15 setTime()

```
def simlib.hub.Hub.setTime (
              time )
```

6.7.2.16 triliterateNode()

```
def simlib.hub.Hub.triliterateNode (
              node )
```

6.7.3 Member Data Documentation

6.7.3.1 algorithm

```
simlib.hub.Hub.algorithm
```

6.7.3.2 anchors

```
simlib.hub.Hub.anchors
```

6.7.3.3 map

```
simlib.hub.Hub.map
```

6.7.3.4 nodePositions

```
simlib.hub.Hub.nodePositions
```

6.7.3.5 nodes

```
simlib.hub.Hub.nodes
```

6.7.3.6 time

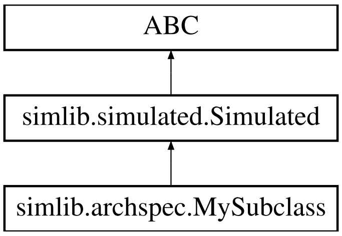
```
simlib.hub.Hub.time
```

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

- hub.py

6.8 simlib.archspec.MySubclass Class Reference

Inheritance diagram for simlib.archspec.MySubclass:



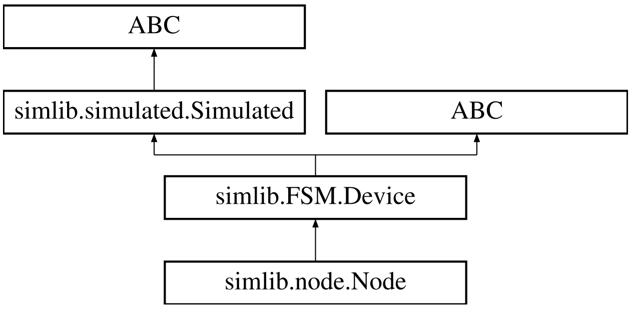
Additional Inherited Members

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

- archspec.py

6.9 simlib.node.Node Class Reference

Inheritance diagram for simlib.node.Node:



Public Member Functions

- def __init__
- def setTime
- def getID ()
- def listenForSignal ()
- def addAction
- def prependAction

Public Attributes

- time
- ID
- xPos
- yPos
- zPos
- xVel
- yVel
- zVel
- signalList

6.9.1 Constructor & Destructor Documentation

6.9.1.1 __init__()

```
def simlib.node.Node.__init__ (
              self,
              ID )
```

6.9.2 Member Function Documentation

6.9.2.1 addAction()

```
def simlib.node.Node.addAction (
              self,
              function )
```

6.9.2.2 getID()

```
def simlib.node.Node.getID (
              int )
```

6.9.2.3 listenForSignal()

```
def simlib.node.Node.listenForSignal (
              int )
```

```
Searches to see if it is the target of any signals. Returns the sending ID
if a signal is found or 0 if no signal is found.
```

6.9.2.4 prependAction()

```
def simlib.node.Node.prependAction (
              self,
              function )
```

6.9.2.5 setTime()

```
def simlib.node.Node.setTime (
              time )
```

6.9.3 Member Data Documentation

6.9.3.1 ID

```
simlib.node.Node.ID
```

6.9.3.2 signalList

```
simlib.node.Node.signalList
```

6.9.3.3 time

```
simlib.node.Node.time
```

6.9.3.4 xPos

```
simlib.node.Node.xPos
```

6.9.3.5 xVel

```
simlib.node.Node.xVel
```

6.9.3.6 yPos

```
simlib.node.Node.yPos
```

6.9.3.7 yVel

```
simlib.node.Node.yVel
```

6.9.3.8 zPos

```
simlib.node.Node.zPos
```

6.9.3.9 zVel

```
simlib.node.Node.zVel
```

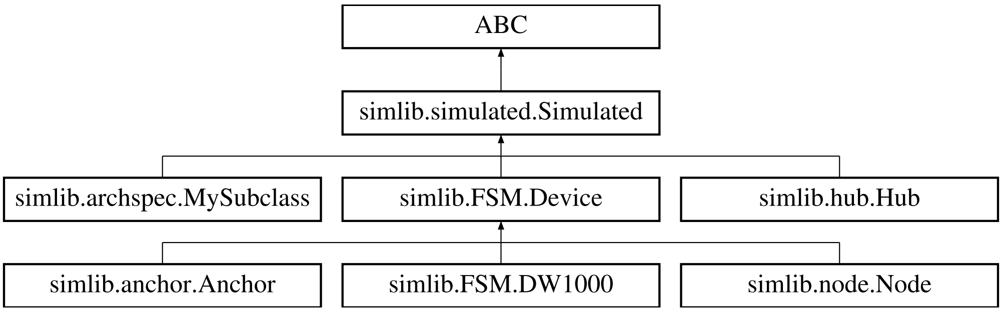
The documentation for this class was generated from the following file:

- node.py

6.10 simlib.simulated.Simulated Class Reference

Classes ##.

Inheritance diagram for simlib.simulated.Simulated:



Public Member Functions

- def __init__ (self)
- def mainloop
- def run_timestep

Public Attributes

- actionQueue

6.10.1 Detailed Description

Classes ##.

6.10.2 Constructor & Destructor Documentation

6.10.2.1 __init__()

```
def simlib.simulated.Simulated.__init__ (
              self )
```

6.10.3 Member Function Documentation

6.10.3.1 mainloop()

```
def simlib.simulated.Simulated.mainloop (
              self,
              simlist )
```

6.10.3.2 run_timestep()

```
def simlib.simulated.Simulated.run_timestep (
              self,
              simlist )
```

6.10.4 Member Data Documentation

6.10.4.1 actionQueue

```
simlib.simulated.Simulated.actionQueue
```

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

- simulated.py

6.11 simlib.simulationenvironment.SimulationEnvironment Class Reference

Public Member Functions

- def __init__ (self)
- def createHub
- def createNode
- def getNodeByID
- def associateNode
- def dissassociateNode
- def deleteNode
- def createAnchor
- def getAnchorByID
- def associateAnchor
- def dissassociateAnchor
- def deleteAnchor
- def mainloop ()

Public Attributes

- time
- hubs
- anchors
- nodes
- nextID
- signalList

6.11.1 Constructor & Destructor Documentation

6.11.1.1 __init__()

```
def simlib.simulationenvironment.SimulationEnvironment.__init__ (
              self )
```

6.11.2 Member Function Documentation

6.11.2.1 associateAnchor()

```
def simlib.simulationenvironment.SimulationEnvironment.associateAnchor (
              self,
              hub )
```

6.11.2.2 associateNode()

```
def simlib.simulationenvironment.SimulationEnvironment.associateNode (
              self,
              hub )
```

6.11.2.3 createAnchor()

```
def simlib.simulationenvironment.SimulationEnvironment.createAnchor (
              self,
              xPos )
```

6.11.2.4 createHub()

```
def simlib.simulationenvironment.SimulationEnvironment.createHub (
              self,
              algorithm )
```

6.11.2.5 createNode()

```
def simlib.simulationenvironment.SimulationEnvironment.createNode (
              self,
              xPos )
```

6.11.2.6 deleteAnchor()

```
def simlib.simulationenvironment.SimulationEnvironment.deleteAnchor (
              self,
              anchor )
```

6.11.2.7 deleteNode()

```
def simlib.simulationenvironment.SimulationEnvironment.deleteNode (
              self,
              node )
```

6.11.2.8 dissassociateAnchor()

```
def simlib.simulationenvironment.SimulationEnvironment.dissassociateAnchor (
              hub )
```

6.11.2.9 dissassociateNode()

```
def simlib.simulationenvironment.SimulationEnvironment.dissassociateNode (
              self,
              hub )
```

6.11.2.10 getAnchorByID()

```
def simlib.simulationenvironment.SimulationEnvironment.getAnchorByID (
              self,
              ID )
```

6.11.2.11 getNodeByID()

```
def simlib.simulationenvironment.SimulationEnvironment.getNodeByID (
              self,
              ID )
```

6.11.2.12 mainloop()

```
def simlib.simulationenvironment.SimulationEnvironment.mainloop ( )
```

6.11.3 Member Data Documentation

6.11.3.1 anchors

```
simlib.simulationenvironment.SimulationEnvironment.anchors
```

6.11.3.2 hubs

```
simlib.simulationenvironment.SimulationEnvironment.hubs
```

6.11.3.3 nextID

```
simlib.simulationenvironment.SimulationEnvironment.nextID
```

6.11.3.4 nodes

```
simlib.simulationenvironment.SimulationEnvironment.nodes
```

6.11.3.5 signalList

```
simlib.simulationenvironment.SimulationEnvironment.signalList
```

6.11.3.6 time

```
simlib.simulationenvironment.SimulationEnvironment.time
```

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

- simulationenvironment.py

6.12 simlib.FSM.State Class Reference

Public Member Functions

- def __init__
- def getParam

Public Attributes

- physical_data

6.12.1 Detailed Description

```
Creates instance of a State for the device.
```

6.12.2 Constructor & Destructor Documentation

6.12.2.1 __init__()

```
def simlib.FSM.State.__init__ (
              self,
              physical_data )
```

6.12.3 Member Function Documentation

6.12.3.1 getParam()

```
def simlib.FSM.State.getParam (
              self,
              param )
```

6.12.4 Member Data Documentation

6.12.4.1 physical_data

```
simlib.FSM.State.physical_data
```

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

- FSM.py

Chapter 7

File Documentation

7.1 __init__.py File Reference

Namespaces

- simlib

7.2 action.py File Reference

Classes

- class simlib.action.Action

  Classes ##.

Namespaces

- simlib.action

Functions

- def simlib.action.fn (n)
- def simlib.action.fn2 (x, y)

Variables

- int simlib.action.errors = 0
- simlib.action.test_obj = None

7.3 anchor.py File Reference

Classes

- class simlib.anchor.Anchor

Namespaces

- simlib.anchor

7.4 archspec.py File Reference

Classes

- class simlib.archspec.ArchSpec

  Classes ##.
- class simlib.archspec.MySubclass

Namespaces

- simlib.archspec

Variables

- int simlib.archspec.errors = 0
- simlib.archspec.archspec_obj = ArchSpec(int , int , int)

7.5 FSM.py File Reference

Classes

- class simlib.FSM.State
- class simlib.FSM.Device
- class simlib.FSM.DW1000

Namespaces

- simlib.FSM

Functions

- def simlib.FSM.getOffCurrent (physical_data)

  This is dumb but i cant think of a way to do this while making it # general to each device.
- def simlib.FSM.getDeepSleepCurrent (physical_data)
- def simlib.FSM.getSleepCurrent (physical_data)
- def simlib.FSM.getInitCurrent (physical_data)
- def simlib.FSM.getIdleCurrent (physical_data)
- def simlib.FSM.getRXCurrent (physical_data)
- def simlib.FSM.getTXCurrent (physical_data)
- def simlib.FSM.getTime (physical_data)
- def simlib.FSM.getBattery (dev_data, state_data)

Variables

- int simlib.FSM.OSC_STARTUP_SEND_CURRENT = 3
- int simlib.FSM.OSC_STARTUP_SEND_TIME = 2000
- int simlib.FSM.PLL_STARTUP_SEND_CURRENT = 12
- int simlib.FSM.PLL_STARTUP_SEND_TIME = 7
- int simlib.FSM.WR_TX_DATA_SEND_CURRENT = 15
- int simlib.FSM.WR_TX_DATA_SEND_TIME = 10
- int simlib.FSM.TX_SHR_SEND_CURRENT = 65
- int simlib.FSM.TX_SHR_SEND_TIME = 135
- int simlib.FSM.TX_PHR_SEND_CURRENT = 48
- float simlib.FSM.TX_PHR_SEND_TIME = 1.33
- bool simlib.FSM.DEBUG = True
- simlib.FSM.OFF_STATE = State({'current' : getOffCurrent, 'time' : getTime, 'state' : 'off_state'})

  FSM for the DW1000, includes states, thier delays, and the current for each state.
- simlib.FSM.DEEPSLEEP_STATE = State({'current' : getDeepSleepCurrent, 'time' : getTime, 'state' ↩ : 'deepsleep_state'})
- simlib.FSM.SLEEP_STATE = State({'current' : getSleepCurrent, 'time' : getTime, 'state' : 'sleep_state'})
- simlib.FSM.INIT_STATE = State({'current' : getInitCurrent, 'time' : getTime, 'state' : 'init_state'})
- simlib.FSM.IDLE_STATE = State({'current' : getIdleCurrent, 'time' : getTime, 'state' : 'idle_state'})
- simlib.FSM.RX_STATE = State({'current' : getRXCurrent, 'time' : getTime, 'state' : 'rx_state'})
- simlib.FSM.TX_STATE = State({'current' : getTXCurrent, 'time' : getTime, 'state' : 'tx_state'})
- simlib.FSM.INITIAL_STATE = OFF_STATE
- dictionary simlib.FSM.DEVICE_DATA = {'batt_life' : getBattery}
- dictionary simlib.FSM.AVAILABLE_STATES
- simlib.FSM.battlife
- simlib.FSM.device = DW1000(AVAILABLE_STATES, INITIAL_STATE, DEVICE_DATA)
- int simlib.FSM.errors = 0
- list simlib.FSM.simlist = [ ]
- simlib.FSM.my_current = device.getState().getParam('current')

7.6 hub.py File Reference

Classes

- class simlib.hub.Hub

Namespaces

- simlib.hub

Variables

- simlib.hub.hub = Hub(0)

7.7 node.py File Reference

Classes

- class simlib.node.Node

Namespaces

- simlib.node

7.8 simulated.py File Reference

Classes

- class simlib.simulated.Simulated

  Classes ##.
- class simlib.simulated.ActionQueue

Namespaces

- simlib.simulated

Variables

- bool simlib.simulated.DEBUG = True
- int simlib.simulated.errors = 0
- simlib.simulated.aq = ActionQueue( )
- simlib.simulated.action1 = Action(aq.test, ["I was the first action added"])
- simlib.simulated.action2 = Action(aq.test, ["then me (:"])
- simlib.simulated.action3 = Action(aq.test, ["lastly me (;"])
- int simlib.simulated.i = 0

7.9 simulationenvironment.py File Reference

Classes

- class simlib.simulationenvironment.SimulationEnvironment

Namespaces

- simlib.simulationenvironment

Functions

- def simlib.simulationenvironment.superSmartTrilaterationAlgorithm

Variables

- int simlib.simulationenvironment.errors = 0
- simlib.simulationenvironment.archspec = ArchSpec(Hub, Anchor, Node)
- simlib.simulationenvironment.simEnv = SimulationEnvironment()
- simlib.simulationenvironment.hub = simEnv.createHub(superSmartTrilaterationAlgorithm)
- simlib.simulationenvironment.anchor = simEnv.createAnchor(0, 0, i)
- simlib.simulationenvironment.node = simEnv.createNode(3, 3, 3, 0, 0, 0)

Index