

SST Document

Generated by Doxygen 1.8.11

Contents

1	Main Page	1
2	Key Interfaces	3
3	Deprecated List	5
4	Hierarchical Index	7
4.1	Class Hierarchy	7
5	Class Index	15
5.1	Class List	15
6	Class Documentation	21
6.1	SST::Statistics::AccumulatorStatistic< NumberBase > Class Template Reference	21
6.1.1	Detailed Description	22
6.1.2	Member Function Documentation	22
6.1.2.1	addData_impl(NumberBase value) override	22
6.1.2.2	clearStatisticData() override	22
6.1.2.3	getArithmeticMean()	22
6.1.2.4	getCount()	23
6.1.2.5	getMax()	23
6.1.2.6	getMin()	23
6.1.2.7	getStandardDeviation()	23
6.1.2.8	getSum()	23
6.1.2.9	getSumSquared()	24
6.1.2.10	getVariance()	24

6.1.2.11	registerOutputFields(StatisticOutput *statOutput) override	24
6.1.3	Member Data Documentation	24
6.1.3.1	statParams	24
6.2	SST::Action Class Reference	25
6.2.1	Detailed Description	25
6.2.2	Member Function Documentation	25
6.2.2.1	endSimulation()	25
6.2.2.2	print(const std::string &header, Output &out) const override	25
6.3	SST::Activity Class Reference	26
6.3.1	Detailed Description	26
6.3.2	Member Function Documentation	26
6.3.2.1	execute(void)=0	26
6.3.2.2	getDeliveryTime() const	27
6.3.2.3	getPriority() const	27
6.3.2.4	print(const std::string &header, Output &out) const	27
6.3.2.5	setDeliveryTime(SimTime_t time)	27
6.3.2.6	setPriority(int priority)	27
6.3.2.7	setQueueOrder(uint64_t order)	27
6.4	SST::ActivityQueue Class Reference	27
6.4.1	Detailed Description	28
6.4.2	Member Function Documentation	28
6.4.2.1	empty()=0	28
6.4.2.2	front()=0	28
6.4.2.3	insert(Activity *activity)=0	28
6.4.2.4	pop()=0	28
6.4.2.5	size()=0	28
6.5	SST::ELI::Allocator< Base, T, Enable > Struct Template Reference	29
6.6	SST::ELI::Allocator< SSTELEMENTPythonModule, T > Struct Template Reference	29
6.7	SST::Core::ThreadSafe::Barrier Class Reference	29
6.7.1	Member Function Documentation	29

6.7.1.1	<code>resize(size_t newCount)</code>	29
6.7.1.2	<code>wait()</code>	30
6.8	SST::BaseComponent Class Reference	30
6.8.1	Detailed Description	32
6.8.2	Member Function Documentation	32
6.8.2.1	<code>complete(unsigned int UNUSED(phase))</code>	32
6.8.2.2	<code>configureLink(std::string name, TimeConverter *time_base, Event::HandlerBase *handler=NULL)</code>	32
6.8.2.3	<code>configureLink(std::string name, std::string time_base, Event::HandlerBase *handler=NULL)</code>	32
6.8.2.4	<code>configureLink(std::string name, Event::HandlerBase *handler=NULL)</code>	33
6.8.2.5	<code>configureSelfLink(std::string name, TimeConverter *time_base, Event::HandlerBase *handler=NULL)</code>	33
6.8.2.6	<code>configureSelfLink(std::string name, std::string time_base, Event::HandlerBase *handler=NULL)</code>	33
6.8.2.7	<code>configureSelfLink(std::string name, Event::HandlerBase *handler=NULL)</code>	34
6.8.2.8	<code>emergencyShutdown(void)</code>	34
6.8.2.9	<code>finish()</code>	34
6.8.2.10	<code>getCoordinates() const</code>	34
6.8.2.11	<code>getCurrentSimTime(TimeConverter *tc) const</code>	34
6.8.2.12	<code>getCurrentSimTime() const</code>	35
6.8.2.13	<code>getCurrentSimTime(std::string base)</code>	35
6.8.2.14	<code>getCurrentSimTimeMicro() const</code>	35
6.8.2.15	<code>getCurrentSimTimeMilli() const</code>	35
6.8.2.16	<code>getCurrentSimTimeNano() const</code>	35
6.8.2.17	<code>getId() const</code>	35
6.8.2.18	<code>getLocalSharedRegion(const std::string &key, size_t size)</code>	35
6.8.2.19	<code>getName() const</code>	35
6.8.2.20	<code>getNextClockCycle(TimeConverter *freq)</code>	35
6.8.2.21	<code>getParent() const</code> <small>__attribute__((deprecated("getParent() will be removed in SST version 10.0. With the new subcomponent structure</small>	36
6.8.2.22	<code>init(unsigned int UNUSED(phase))</code>	36
6.8.2.23	<code>isPortConnected(const std::string &name) const</code>	36

6.8.2.24	<code>loadModule(std::string type, Params &params)</code>	36
6.8.2.25	<code>loadModuleWithComponent(std::string type, Component *comp, Params &params) __attribute__((deprecated("loadModuleWithComponent will be removed in SST version 10.0. If the module needs access to the parent component</code>	36
6.8.2.26	<code>printStatus(Output &UNUSED(out))</code>	37
6.8.2.27	<code>registerClock(std::string freq, Clock::HandlerBase *handler, bool regAll=true)</code>	38
6.8.2.28	<code>registerOneShot(std::string timeDelay, OneShot::HandlerBase *handler)</code>	38
6.8.2.29	<code>registerStatistic(SST::Params &params, const std::string &statName, const std::string &statSubId="")</code>	38
6.8.2.30	<code>registerTimeBase(std::string base, bool regAll=true)</code>	39
6.8.2.31	<code>reregisterClock(TimeConverter *freq, Clock::HandlerBase *handler)</code>	39
6.8.2.32	<code>setDefaultTimeBase(TimeConverter *tc)</code>	39
6.8.2.33	<code>setup()</code>	39
6.8.2.34	<code>Status()</code>	39
6.8.2.35	<code>unregisterClock(TimeConverter *tc, Clock::HandlerBase *handler)</code>	39
6.8.2.36	<code>wasLoadedWithLegacyAPI() const</code>	39
6.8.3	Member Data Documentation	39
6.8.3.1	<code>loadUserSubComponentByIndex< T, ARGS... ></code>	39
6.9	<code>SST::Core::ThreadSafe::BoundedQueue< T ></code> Class Template Reference	40
6.10	<code>SST::ELI::Builder< Base, Args ></code> Struct Template Reference	40
6.11	<code>SST::ELI::BuilderDatabase</code> Struct Reference	41
6.12	<code>SST::ELI::BuilderInfoImpl< Policy, Policies ></code> Class Template Reference	41
6.13	<code>SST::ELI::BuilderInfoImpl< void ></code> Class Template Reference	41
6.14	<code>SST::ELI::BuilderLibrary< Base, CtorArgs ></code> Class Template Reference	42
6.15	<code>SST::ELI::BuilderLibraryDatabase< Base, CtorArgs ></code> Class Template Reference	42
6.16	<code>SST::RegionInfo::BulkMergeInfo</code> Class Reference	42
6.17	<code>SST::ELI::CachedAllocator< Base, T ></code> Struct Template Reference	43
6.18	<code>SST::ChangeSet</code> Class Reference	43
6.19	<code>SST::RegionInfo::ChangeSetMergeInfo</code> Class Reference	44
6.20	<code>SST::char_delimiter</code> Struct Reference	44
6.20.1	Member Function Documentation	45
6.20.1.1	<code>operator()(iter &first, iter last, std::string &token)</code>	45

6.21	SST::Core::Interprocess::CircularBuffer< T > Class Template Reference	45
6.22	SST::Clock Class Reference	45
6.22.1	Detailed Description	46
6.22.2	Constructor & Destructor Documentation	46
6.22.2.1	Clock(TimeConverter *period, int priority=CLOCKPRIORITY)	46
6.22.3	Member Function Documentation	46
6.22.3.1	getNextCycle()	46
6.22.3.2	print(const std::string &header, Output &out) const override	46
6.22.3.3	registerHandler(Clock::HandlerBase *handler)	46
6.22.3.4	schedule()	46
6.22.3.5	unregisterHandler(Clock::HandlerBase *handler, bool &empty)	47
6.23	SST::Component Class Reference	47
6.23.1	Detailed Description	47
6.23.2	Member Function Documentation	47
6.23.2.1	primaryComponentDoNotEndSim()	47
6.23.2.2	primaryComponentOKToEndSim()	48
6.23.2.3	registerAsPrimaryComponent()	48
6.23.2.4	registerExit() __attribute__((deprecated("'"registerExit is deprecated and will be removed in SST version 10.0. Please use registerAsPrimaryComponent() and primaryComponentDoNotEndSim() instead.'"))	48
6.23.2.5	SST_ELI_DECLARE_INFO_EXTERN(ELI::ProvidesParams, ELI::ProvidesSub↔ComponentSlots, ELI::ProvidesPorts, ELI::ProvidesStats, ELI::ProvidesCategory) Component(ComponentId_t id)	48
6.23.2.6	unregisterExit() __attribute__((deprecated("'"unregisterExit is deprecated and will be removed in SST version 10.0. Please use primaryComponentOKToEndSim() instead.'"))	49
6.24	SST::ComponentExtension Class Reference	49
6.24.1	Detailed Description	49
6.25	ComponentHolder Struct Reference	49
6.26	SST::ComponentInfo Class Reference	50
6.26.1	Member Typedef Documentation	51
6.26.1.1	statEnableList_t	51
6.27	SST::ComponentInfoMap Class Reference	51

6.28	ComponentPy_t Struct Reference	51
6.29	SST::Config Class Reference	52
6.29.1	Detailed Description	53
6.29.2	Constructor & Destructor Documentation	53
6.29.2.1	Config(RankInfo world_size)	53
6.29.3	Member Function Documentation	53
6.29.3.1	getLibPath(void) const	53
6.29.3.2	getVerboseLevel()	54
6.29.3.3	parseCmdLine(int argc, char *argv[])	54
6.29.3.4	Print()	54
6.29.3.5	print()	54
6.29.3.6	printTimingInfo()	54
6.29.3.7	setConfigEntryFromModel(const std::string &entryName, const std::string &value)	54
6.29.3.8	setStopAt(std::string stopAtStr)	54
6.29.3.9	setTimeBase(std::string timeBase)	54
6.29.4	Member Data Documentation	54
6.29.4.1	configFile	54
6.29.4.2	debugFile	54
6.29.4.3	dump_component_graph_file	55
6.29.4.4	enable_sig_handling	55
6.29.4.5	heartbeatPeriod	55
6.29.4.6	model_options	55
6.29.4.7	no_env_config	55
6.29.4.8	output_config_graph	55
6.29.4.9	output_core_prefix	55
6.29.4.10	output_directory	55
6.29.4.11	output_dot	55
6.29.4.12	output_json	55
6.29.4.13	output_xml	56
6.29.4.14	partitioner	56

6.29.4.15 print_env	56
6.29.4.16 print_timing	56
6.29.4.17 runMode	56
6.29.4.18 stopAfterSec	56
6.29.4.19 stopAtCycle	56
6.29.4.20 timeBase	56
6.29.4.21 timeVortex	56
6.29.4.22 verbose	56
6.29.4.23 world_size	57
6.30 SST::ConfigComponent Class Reference	57
6.30.1 Detailed Description	58
6.30.2 Constructor & Destructor Documentation	58
6.30.2.1 ConfigComponent(ComponentId_t id, std::string name, std::string type, float weight, RankInfo rank)	58
6.30.3 Member Function Documentation	58
6.30.3.1 print(std::ostream &os) const	58
6.30.4 Member Data Documentation	58
6.30.4.1 enabledStatistics	58
6.30.4.2 id	59
6.30.4.3 links	59
6.30.4.4 name	59
6.30.4.5 nextSubID	59
6.30.4.6 params	59
6.30.4.7 rank	59
6.30.4.8 slot_num	59
6.30.4.9 subComponents	59
6.30.4.10 type	59
6.30.4.11 ultimate_parent	59
6.30.4.12 weight	60
6.31 SST::ConfigGraph Class Reference	60
6.31.1 Detailed Description	61

6.31.2	Member Function Documentation	61
6.31.2.1	addComponent(ComponentId_t id, std::string name, std::string type, float weight, RankInfo rank)	61
6.31.2.2	addComponent(ComponentId_t id, std::string name, std::string type)	61
6.31.2.3	addLink(ComponentId_t comp_id, std::string link_name, std::string port, std::string latency_str, bool no_cut=false)	61
6.31.2.4	addStatisticOutputParameter(const std::string ¶m, const std::string &value)	61
6.31.2.5	addStatisticParameterForComponentName(const std::string &ComponentName, const std::string &statisticName, const std::string ¶m, const std::string &value)	61
6.31.2.6	checkForStructuralErrors()	61
6.31.2.7	checkRanks(RankInfo ranks)	61
6.31.2.8	containsComponentInRank(RankInfo rank)	62
6.31.2.9	enableStatisticForComponentName(std::string ComponentName, std::string statisticName)	62
6.31.2.10	getComponentMap()	62
6.31.2.11	getLinkMap()	62
6.31.2.12	postCreationCleanup()	62
6.31.2.13	print(std::ostream &os) const	62
6.31.2.14	setComponentRanks(RankInfo rank)	62
6.31.2.15	setLinkNoCut(std::string link_name)	62
6.31.2.16	setStatisticLoadLevel(uint8_t loadLevel)	62
6.31.2.17	setStatisticOutput(const std::string &name)	62
6.31.2.18	setStatisticOutputParams(const Params &p)	63
6.32	SST::Core::ConfigGraphOutput Class Reference	63
6.33	SST::Core::ConfigGraphOutputException Class Reference	63
6.34	SST::ConfigLink Class Reference	64
6.34.1	Detailed Description	64
6.34.2	Member Function Documentation	64
6.34.2.1	getMinLatency() const	64
6.34.2.2	print(std::ostream &os) const	64
6.34.3	Member Data Documentation	65
6.34.3.1	component	65

6.34.3.2	current_ref	65
6.34.3.3	id	65
6.34.3.4	latency	65
6.34.3.5	latency_str	65
6.34.3.6	name	65
6.34.3.7	no_cut	65
6.34.3.8	port	65
6.35	SST::ConfigStatGroup Class Reference	65
6.35.1	Member Function Documentation	66
6.35.1.1	verifyStatsAndComponents(const ConfigGraph *graph)	66
6.36	SST::ConfigStatOutput Class Reference	66
6.37	SST::ELI::CtorList< Base, Ctor, Ctors > Struct Template Reference	67
6.38	SST::ELI::CtorList< Base, void > Struct Template Reference	67
6.39	SST::ELI::DataBase< T > Class Template Reference	68
6.40	SST::decimal_fixedpoint< whole_words, fraction_words > Class Template Reference	68
6.40.1	Detailed Description	69
6.40.2	Constructor & Destructor Documentation	69
6.40.2.1	decimal_fixedpoint()	69
6.40.2.2	decimal_fixedpoint(std::string init)	69
6.40.2.3	decimal_fixedpoint(T init, typename std::enable_if< std::is_unsigned< T >::value >::type !=0)	70
6.40.2.4	decimal_fixedpoint(T init, typename std::enable_if< std::is_signed< T >::value &&std::is_integral< T >::value >::type !=0)	70
6.40.2.5	decimal_fixedpoint(const T init, typename std::enable_if< std::is_floating_point< T >::value >::type !=0)	70
6.40.2.6	decimal_fixedpoint(const decimal_fixedpoint &init)	70
6.40.3	Member Function Documentation	71
6.40.3.1	convert_to(typename std::enable_if< std::is_unsigned< T >::value >::type !=0) const	71
6.40.3.2	convert_to(typename std::enable_if< std::is_signed< T >::value &&std::is_integral< T >::value >::type !=0) const	71
6.40.3.3	convert_to(typename std::enable_if< std::is_floating_point< T >::value >::type !=0) const	71

6.40.3.4	<code>getFractionWords() const</code>	71
6.40.3.5	<code>getWholeWords() const</code>	71
6.40.3.6	<code>inverse()</code>	71
6.40.3.7	<code>negate()</code>	71
6.40.3.8	<code>operator!=(const decimal_fixedpoint &v) const</code>	71
6.40.3.9	<code>operator*=(const decimal_fixedpoint &v)</code>	72
6.40.3.10	<code>operator+=(const decimal_fixedpoint &v)</code>	72
6.40.3.11	<code>operator-=(const decimal_fixedpoint &v)</code>	72
6.40.3.12	<code>operator/=(const decimal_fixedpoint &v)</code>	72
6.40.3.13	<code>operator<(const decimal_fixedpoint &v) const</code>	73
6.40.3.14	<code>operator<=(const decimal_fixedpoint &v) const</code>	73
6.40.3.15	<code>operator=(const decimal_fixedpoint &v)</code>	73
6.40.3.16	<code>operator=(uint64_t v)</code>	73
6.40.3.17	<code>operator=(int64_t v)</code>	73
6.40.3.18	<code>operator=(double v)</code>	73
6.40.3.19	<code>operator=(std::string v)</code>	73
6.40.3.20	<code>operator==(const decimal_fixedpoint &v) const</code>	73
6.40.3.21	<code>operator>(const decimal_fixedpoint &v) const</code>	74
6.40.3.22	<code>operator>=(const decimal_fixedpoint &v) const</code>	74
6.40.3.23	<code>toDouble() const</code>	74
6.40.3.24	<code>toLong() const</code>	74
6.40.3.25	<code>toString(int32_t precision=6) const</code>	74
6.40.3.26	<code>toUnsignedLong() const</code>	75
6.41	<code>SST::ELI::DerivedBuilder< T, Base, Args ></code> Struct Template Reference	75
6.42	<code>SST::ELI::DerivedBuilder< SSTELEMENTPythonModule, SSTELEMENTPythonModuleOldELI, const std::string & ></code> Struct Template Reference	75
6.43	<code>SST::Core::DotConfigGraphOutput</code> Class Reference	76
6.44	<code>SST::ElementInfoParam</code> Struct Reference	76
6.44.1	Detailed Description	76
6.44.2	Member Data Documentation	77
6.44.2.1	<code>defaultValue</code>	77

6.44.2.2	description	77
6.44.2.3	name	77
6.45	SST::ElementInfoPort Struct Reference	77
6.45.1	Detailed Description	77
6.45.2	Member Data Documentation	77
6.45.2.1	description	77
6.45.2.2	name	77
6.45.2.3	validEvents	78
6.46	SST::ElementInfoPort2 Struct Reference	78
6.46.1	Detailed Description	78
6.46.2	Member Data Documentation	78
6.46.2.1	description	78
6.46.2.2	name	78
6.46.2.3	validEvents	78
6.47	SST::ElementInfoStatistic Struct Reference	79
6.47.1	Detailed Description	79
6.47.2	Member Data Documentation	79
6.47.2.1	description	79
6.47.2.2	enableLevel	79
6.47.2.3	name	79
6.47.2.4	units	79
6.48	SST::ElementInfoSubComponentSlot Struct Reference	79
6.49	SST::ELI::ElementsBuilder< Base, CtorTuple > Struct Template Reference	80
6.50	SST::ELI::ElementsBuilder< Base, std::tuple< Args... > > Struct Template Reference	80
6.51	SST::ELI::ElementsInfo< Base > Struct Template Reference	80
6.52	SST::ElemLoader Class Reference	80
6.52.1	Detailed Description	81
6.52.2	Constructor & Destructor Documentation	81
6.52.2.1	ElemLoader(const std::string &searchPaths)	81
6.52.3	Member Function Documentation	81

6.52.3.1	<code>getPotentialElements()</code>	81
6.52.3.2	<code>loadLibrary(const std::string &elemlib, bool showErrors)</code>	81
6.53	<code>SST::EmptyRankSync</code> Class Reference	81
6.53.1	Member Function Documentation	82
6.53.1.1	<code>registerLink(const RankInfo &UNUSED(to_rank), const RankInfo &UNUSED← D(from_rank), LinkId_t UNUSED(link_id), Link *UNUSED(link)) override</code>	82
6.54	<code>SST::EmptyThreadSync</code> Class Reference	82
6.54.1	Member Function Documentation	83
6.54.1.1	<code>registerLink(LinkId_t UNUSED(link_id), Link *UNUSED(link)) override</code>	83
6.55	<code>SST::Core::Environment::EnvironmentConfigGroup</code> Class Reference	83
6.56	<code>SST::Core::Environment::EnvironmentConfiguration</code> Class Reference	83
6.57	<code>SST::ComponentInfo::EqualsID</code> Struct Reference	84
6.58	<code>SST::ComponentInfo::EqualsName</code> Struct Reference	84
6.59	<code>SST::escaped_list_separator</code> Struct Reference	84
6.59.1	Member Function Documentation	84
6.59.1.1	<code>operator()(iter &first, iter last, std::string &token)</code>	84
6.60	<code>SST::Event</code> Class Reference	85
6.60.1	Detailed Description	85
6.60.2	Member Typedef Documentation	86
6.60.2.1	<code>id_type</code>	86
6.60.3	Member Function Documentation	86
6.60.3.1	<code>clone()</code>	86
6.60.3.2	<code>execute(void) override</code>	86
6.60.3.3	<code>generateUniqueld()</code>	86
6.60.3.4	<code>getDeliveryLink()</code>	86
6.60.3.5	<code>getLinkId(void) const</code>	86
6.60.3.6	<code>print(const std::string &header, Output &out) const override</code>	86
6.60.3.7	<code>setDeliveryLink(LinkId_t id, Link *link)</code>	86
6.60.3.8	<code>setRemoteEvent()</code>	87
6.60.4	Member Data Documentation	87
6.60.4.1	<code>delivery_link</code>	87

6.60.4.2	NO_ID	87
6.61	SST::Exit Class Reference	87
6.61.1	Detailed Description	87
6.61.2	Constructor & Destructor Documentation	87
6.61.2.1	Exit(int num_threads, TimeConverter *period, bool single_rank)	87
6.61.3	Member Function Documentation	88
6.61.3.1	execute(void) override	88
6.61.3.2	print(const std::string &header, Output &out) const override	88
6.61.3.3	refDec(ComponentId_t, uint32_t thread)	88
6.61.3.4	refInc(ComponentId_t, uint32_t thread)	88
6.62	SST::ELI::ExtendedCtor< NewCtor, OldCtor > Struct Template Reference	88
6.63	SST::Factory Class Reference	89
6.63.1	Detailed Description	90
6.63.2	Member Function Documentation	90
6.63.2.1	Create(const std::string &type, SST::Params &params, CtorArgs &&...args)	90
6.63.2.2	CreateComponent(ComponentId_t id, std::string &componentname, Params &params)	90
6.63.2.3	CreateModule(std::string type, Params &params)	90
6.63.2.4	CreateModuleWithComponent(std::string type, Component *comp, Params &params)	91
6.63.2.5	CreatePartitioner(std::string name, RankInfo total_ranks, RankInfo my_rank, int verbosity)	91
6.63.2.6	CreateStatistic(std::string type, BaseComponent *comp, const std::string &stat←Name, const std::string &stat, Params &params, Args...args)	91
6.63.2.7	CreateSubComponent(std::string type, Component *comp, Params &params)	91
6.63.2.8	DoesComponentInfoStatisticNameExist(const std::string &type, const std::string &statisticName)	92
6.63.2.9	DoesSubComponentSlotExist(const std::string &type, const std::string &slotName)	92
6.63.2.10	GetComponentInfoStatisticEnableLevel(const std::string &type, const std::string &statisticName)	92
6.63.2.11	GetComponentInfoStatisticUnits(const std::string &type, const std::string &statisticName)	93
6.63.2.12	getParamNames(const std::string &type)	93

6.63.2.13	<code>getPythonModule(std::string name)</code>	93
6.63.2.14	<code>hasLibrary(std::string elemLib)</code>	93
6.63.2.15	<code>isPortNameValid(const std::string &type, const std::string port_name)</code>	93
6.63.2.16	<code>isSubComponentLoadableUsingAPI(std::string type)</code>	94
6.63.2.17	<code>RequireEvent(std::string eventName)</code>	94
6.64	<code>SST::ELI::GetParams< T, Enable > Struct Template Reference</code>	94
6.65	<code>SST::ELI::GetParams< T, typename MethodDetect< decltype(T::ELI_getParams())>::type > Struct Template Reference</code>	95
6.66	<code>SST::Clock::Handler< classT, argT > Class Template Reference</code>	95
6.66.1	Detailed Description	95
6.66.2	Constructor & Destructor Documentation	95
6.66.2.1	<code>Handler(classT *const object, PtrMember member, argT data)</code>	95
6.66.3	Member Function Documentation	96
6.66.3.1	<code>operator()(Cycle_t cycle) override</code>	96
6.67	<code>SST::OneShot::Handler< classT, argT > Class Template Reference</code>	96
6.67.1	Detailed Description	96
6.67.2	Constructor & Destructor Documentation	97
6.67.2.1	<code>Handler(classT *const object, PtrMember member, argT data)</code>	97
6.67.3	Member Function Documentation	97
6.67.3.1	<code>operator>() override</code>	97
6.68	<code>SST::Event::Handler< classT, argT > Class Template Reference</code>	97
6.68.1	Detailed Description	98
6.68.2	Constructor & Destructor Documentation	98
6.68.2.1	<code>Handler(classT *const object, PtrMember member, argT data)</code>	98
6.68.3	Member Function Documentation	98
6.68.3.1	<code>operator()(Event *event)</code>	98
6.69	<code>SST::Interfaces::SimpleMem::Handler< classT, argT > Class Template Reference</code>	98
6.69.1	Detailed Description	98
6.69.2	Constructor & Destructor Documentation	99
6.69.2.1	<code>Handler(classT *const object, PtrMember member, argT data)</code>	99
6.69.3	Member Function Documentation	99

6.69.3.1	operator()(Request *req)	99
6.70	SST::Interfaces::SimpleNetwork::Handler< classT, argT > Class Template Reference	99
6.70.1	Detailed Description	99
6.70.2	Constructor & Destructor Documentation	100
6.70.2.1	Handler(classT *const object, PtrMember member, argT data)	100
6.71	SST::Clock::Handler< classT, void > Class Template Reference	100
6.71.1	Detailed Description	100
6.71.2	Constructor & Destructor Documentation	101
6.71.2.1	Handler(classT *const object, PtrMember member)	101
6.71.3	Member Function Documentation	101
6.71.3.1	operator()(Cycle_t cycle) override	101
6.72	SST::OneShot::Handler< classT, void > Class Template Reference	101
6.72.1	Detailed Description	101
6.72.2	Constructor & Destructor Documentation	102
6.72.2.1	Handler(classT *const object, PtrMember member)	102
6.72.3	Member Function Documentation	102
6.72.3.1	operator>() override	102
6.73	SST::Event::Handler< classT, void > Class Template Reference	102
6.73.1	Detailed Description	103
6.73.2	Constructor & Destructor Documentation	103
6.73.2.1	Handler(classT *const object, PtrMember member)	103
6.73.3	Member Function Documentation	103
6.73.3.1	operator()(Event *event)	103
6.74	SST::Interfaces::SimpleMem::Handler< classT, void > Class Template Reference	103
6.74.1	Detailed Description	103
6.74.2	Constructor & Destructor Documentation	104
6.74.2.1	Handler(classT *const object, PtrMember member)	104
6.74.3	Member Function Documentation	104
6.74.3.1	operator()(Request *req)	104
6.75	SST::Interfaces::SimpleNetwork::Handler< classT, void > Class Template Reference	104

6.75.1 Detailed Description	104
6.75.2 Constructor & Destructor Documentation	105
6.75.2.1 Handler(classT *const object, PtrMember member)	105
6.76 SST::Clock::HandlerBase Class Reference	105
6.76.1 Detailed Description	105
6.76.2 Member Function Documentation	105
6.76.2.1 operator()(Cycle_t)=0	105
6.77 SST::OneShot::HandlerBase Class Reference	106
6.77.1 Detailed Description	106
6.77.2 Member Function Documentation	106
6.77.2.1 operator>()=0	106
6.78 SST::Event::HandlerBase Class Reference	106
6.78.1 Detailed Description	106
6.78.2 Member Function Documentation	107
6.78.2.1 operator()(Event *)=0	107
6.79 SST::Interfaces::SimpleMem::HandlerBase Class Reference	107
6.79.1 Detailed Description	107
6.79.2 Member Function Documentation	107
6.79.2.1 operator()(Request *)=0	107
6.80 SST::Interfaces::SimpleNetwork::HandlerBase Class Reference	107
6.80.1 Detailed Description	108
6.81 SST::ComponentInfo::HashID Struct Reference	108
6.82 SST::ComponentInfo::HashName Struct Reference	108
6.83 SST::SyncQueue::Header Struct Reference	108
6.84 SST::Statistics::HistogramStatistic< BinDataType > Class Template Reference	109
6.84.1 Detailed Description	109
6.84.2 Member Function Documentation	110
6.84.2.1 addData_impl(BinDataType value) override	110
6.84.3 Member Data Documentation	110
6.84.3.1 statParams	110

6.85	SST::Statistics::ImplementsStatFields Struct Reference	110
6.86	SST::ELI::InfoDatabase Struct Reference	111
6.87	SST::ELI::InfoLibrary< Base > Class Template Reference	111
6.88	SST::ELI::InfoLibraryDatabase< Base > Class Template Reference	111
6.89	SST::ELI::InfoPorts< T, Enable > Struct Template Reference	112
6.90	SST::ELI::InfoPorts< T, typename MethodDetect< decltype(T::ELI_getPorts())>::type > Struct Template Reference	112
6.91	SST::ELI::InfoStats< T, Enable > Struct Template Reference	112
6.92	SST::ELI::InfoStats< T, typename MethodDetect< decltype(T::ELI_getStatistics())>::type > Struct Template Reference	112
6.93	SST::ELI::InfoSubs< T, Enable > Struct Template Reference	113
6.94	SST::ELI::InfoSubs< T, typename MethodDetect< decltype(T::ELI_getSubComponentSlots())>::type > Struct Template Reference	113
6.95	SST::InitQueue Class Reference	113
6.95.1	Detailed Description	113
6.95.2	Member Function Documentation	114
6.95.2.1	empty() override	114
6.95.2.2	front() override	114
6.95.2.3	insert(Activity *activity) override	114
6.95.2.4	pop() override	114
6.95.2.5	size() override	114
6.96	SST::ELI::InstantiateBuilder< Base, T > Struct Template Reference	114
6.97	SST::ELI::InstantiateBuilderInfo< Base, T > Struct Template Reference	115
6.98	SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType > Class Template Reference	115
6.98.1	Detailed Description	115
6.98.2	Constructor & Destructor Documentation	116
6.98.2.1	IPCTunnel(uint32_t comp_id, size_t numBuffers, size_t bufferSize, uint32_t expectedChildren=1)	116
6.98.2.2	IPCTunnel(const std::string ®ion_name)	116
6.98.2.3	~IPCTunnel()	116
6.98.3	Member Function Documentation	116
6.98.3.1	clearBuffer(size_t core)	116

6.98.3.2	getSharedData()	116
6.98.3.3	readMessage(size_t buffer)	117
6.98.3.4	readMessageNB(size_t buffer, MsgType *result)	117
6.98.3.5	shutdown(bool all=false)	117
6.98.3.6	writeMessage(size_t core, const MsgType &command)	117
6.98.4	Member Data Documentation	117
6.98.4.1	sharedData	117
6.99	SST::ELI::is_tuple_constructible< T, U > Struct Template Reference	117
6.100	SST::ELI::is_tuple_constructible< T, std::tuple< Args... > > Struct Template Reference	118
6.101	SST::Core::JSONConfigGraphOutput Class Reference	118
6.102	SST::Activity::less_time Class Reference	118
6.102.1	Detailed Description	119
6.102.2	Member Function Documentation	119
6.102.2.1	operator()(const Activity *lhs, const Activity *rhs) const	119
6.102.2.2	operator()(const Activity &lhs, const Activity &rhs) const	119
6.103	SST::Activity::less_time_priority Class Reference	119
6.103.1	Detailed Description	119
6.103.2	Member Function Documentation	119
6.103.2.1	operator()(const Activity *lhs, const Activity *rhs)	119
6.103.2.2	operator()(const Activity &lhs, const Activity &rhs)	119
6.104	SST::Activity::less_time_priority_order Class Reference	120
6.104.1	Detailed Description	120
6.104.2	Member Function Documentation	120
6.104.2.1	operator()(const Activity *lhs, const Activity *rhs) const	120
6.104.2.2	operator()(const Activity &lhs, const Activity &rhs) const	120
6.105	SST::Link Class Reference	120
6.105.1	Detailed Description	122
6.105.2	Constructor & Destructor Documentation	122
6.105.2.1	Link(LinkId_t id)	122
6.105.3	Member Function Documentation	122

6.105.3.1 addRecvLatency(int cycles, std::string timebase)	122
6.105.3.2 addRecvLatency(SimTime_t cycles, TimeConverter *timebase)	122
6.105.3.3 deliverEvent(Event *event) const	122
6.105.3.4 getDefaultTimeBase()	122
6.105.3.5 getId()	122
6.105.3.6 recv()	123
6.105.3.7 recvInitData()	123
6.105.3.8 recvUntimedData()	123
6.105.3.9 send(SimTime_t delay, TimeConverter *tc, Event *event)	123
6.105.3.10 send(SimTime_t delay, Event *event)	123
6.105.3.11 send(Event *event)	123
6.105.3.12 sendInitData(Event *init_data)	124
6.105.3.13 sendUntimedData(Event *data)	124
6.105.3.14 setDefaultTimeBase(TimeConverter *tc)	124
6.105.3.15 setFunctor(Event::HandlerBase *functor)	124
6.105.3.16 setLatency(Cycle_t lat)	124
6.105.3.17 setPolling()	124
6.105.4 Member Data Documentation	124
6.105.4.1 afterInitQueue	124
6.105.4.2 afterRunQueue	124
6.105.4.3 configuredQueue	125
6.105.4.4 defaultTimeBase	125
6.105.4.5 latency	125
6.105.4.6 pair_link	125
6.105.4.7 recvQueue	125
6.105.4.8 rFunctor	125
6.105.4.9 uninitQueue	125
6.105.4.10 untimedQueue	125
6.106 SST::LinkMap Class Reference	125
6.106.1 Detailed Description	126

6.106.2 Member Function Documentation	126
6.106.2.1 addSelfPort(std::string &name)	126
6.106.2.2 empty()	126
6.106.2.3 getLink(std::string name)	126
6.106.2.4 getLinkMap()	126
6.106.2.5 insertLink(std::string name, Link *link)	126
6.107SST::LinkPair Class Reference	127
6.107.1 Detailed Description	127
6.107.2 Constructor & Destructor Documentation	127
6.107.2.1 LinkPair(LinkId_t id)	127
6.107.3 Member Function Documentation	127
6.107.3.1 getId()	127
6.107.3.2 getLeft()	127
6.107.3.3 getRight()	127
6.108LinkPy_t Struct Reference	128
6.109SST::ELI::LoadedLibraries Class Reference	128
6.109.1 Member Function Documentation	128
6.109.1.1 addLoader(const std::string &lib, const std::string &name, std::function< void()> &&loader)	128
6.110SST::LoaderData Struct Reference	128
6.110.1 Detailed Description	129
6.110.2 Member Data Documentation	129
6.110.2.1 advise_handle	129
6.111lt__advise Struct Reference	129
6.112lt__handle Struct Reference	129
6.113lt__interface_id Struct Reference	130
6.114lt_dlinfo Struct Reference	130
6.115lt_dlsymlist Struct Reference	131
6.116lt_dlvtable Struct Reference	131
6.117lt_interface_data Struct Reference	131
6.118SST::RNG::MarsagliaRNG Class Reference	131

6.118.1 Detailed Description	132
6.118.2 Constructor & Destructor Documentation	132
6.118.2.1 MarsagliaRNG(unsigned int initial_z, unsigned int initial_w)	132
6.118.2.2 MarsagliaRNG()	132
6.118.3 Member Function Documentation	132
6.118.3.1 generateNextInt32() override	132
6.118.3.2 generateNextInt64() override	133
6.118.3.3 generateNextUInt32() override	133
6.118.3.4 generateNextUInt64() override	133
6.118.3.5 nextUniform() override	133
6.118.3.6 restart(unsigned int new_z, unsigned int new_w)	133
6.118.3.7 seed(uint64_t newSeed)	133
6.119SST::Core::MemPool Class Reference	134
6.119.1 Detailed Description	134
6.119.2 Constructor & Destructor Documentation	134
6.119.2.1 MemPool(size_t elementSize, size_t initialSize=(2<< 20))	134
6.119.3 Member Function Documentation	134
6.119.3.1 free(void *ptr)	134
6.119.3.2 getBytesMemUsed()	134
6.119.3.3 malloc()	135
6.119.4 Member Data Documentation	135
6.119.4.1 numAlloc	135
6.119.4.2 numFree	135
6.120SST::RNG::MersenneRNG Class Reference	135
6.120.1 Detailed Description	135
6.120.2 Constructor & Destructor Documentation	135
6.120.2.1 MersenneRNG(unsigned int seed)	135
6.120.2.2 MersenneRNG()	136
6.120.2.3 ~MersenneRNG()	136
6.120.3 Member Function Documentation	136

6.120.3.1 generateNextInt32() override	136
6.120.3.2 generateNextInt64() override	136
6.120.3.3 generateNextUInt32() override	136
6.120.3.4 generateNextUInt64() override	136
6.120.3.5 nextUniform() override	136
6.120.3.6 seed(uint64_t newSeed)	137
6.121 SST::ELI::MethodDetect< T > Struct Template Reference	137
6.122 SST::Module Class Reference	137
6.122.1 Detailed Description	137
6.123 ModuleLoaderPy_t Struct Reference	137
6.124 SST::Core::Serialization::need_delete_statics< T > Class Template Reference	138
6.125 SST::Interfaces::SimpleNetwork::NetworkInspector Class Reference	138
6.125.1 Detailed Description	138
6.125.2 Member Function Documentation	138
6.125.2.1 initialize(std::string id)=0	138
6.126 SST::NewRankSync Class Reference	138
6.126.1 Member Function Documentation	139
6.126.1.1 registerLink(const RankInfo &to_rank, const RankInfo &from_rank, LinkId_t link↔ _id, Link *link)=0	139
6.127 SST::NewThreadSync Class Reference	139
6.127.1 Member Function Documentation	140
6.127.1.1 registerLink(LinkId_t link_id, Link *link)=0	140
6.128 SST::ELI::NoValidConstructorsForDerivedType< NumValid > Struct Template Reference	140
6.129 SST::ELI::NoValidConstructorsForDerivedType< 0 > Class Template Reference	141
6.130 SST::NullEvent Class Reference	141
6.130.1 Detailed Description	141
6.130.2 Member Function Documentation	141
6.130.2.1 execute(void) override	141
6.130.2.2 print(const std::string &header, Output &out) const override	141
6.131 SST::Statistics::NullStatistic< T > Class Template Reference	142
6.131.1 Detailed Description	142

6.132 SST::Statistics::NullStatisticBase< T, B > Struct Template Reference	142
6.133 SST::Statistics::NullStatisticBase< std::tuple< Args... >, false > Struct Template Reference	143
6.134 SST::Statistics::NullStatisticBase< T, false > Struct Template Reference	143
6.135 SST::Statistics::NullStatisticBase< T, true > Struct Template Reference	143
6.136 SST::OneShot Class Reference	144
6.136.1 Detailed Description	144
6.136.2 Constructor & Destructor Documentation	144
6.136.2.1 OneShot(TimeConverter *timeDelay, int priority=ONESHOTPRIORITY)	144
6.136.3 Member Function Documentation	145
6.136.3.1 isScheduled()	145
6.136.3.2 print(const std::string &header, Output &out) const override	145
6.136.3.3 registerHandler(OneShot::HandlerBase *handler)	145
6.137 SST::Output Class Reference	145
6.137.1 Detailed Description	146
6.137.2 Member Enumeration Documentation	147
6.137.2.1 output_location_t	147
6.137.3 Constructor & Destructor Documentation	147
6.137.3.1 Output(const std::string &prefix, uint32_t verbose_level, uint32_t verbose_mask, output_location_t location, std::string localoutputfilename="")	147
6.137.3.2 Output()	148
6.137.4 Member Function Documentation	148
6.137.4.1 debug(uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) const __attribute__((format(printf	148
6.137.4.2 debugPrefix(const char *tempPrefix, uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) __ attribute__((format(printf	148
6.137.4.3 fatal(uint32_t line, const char *file, const char *func, int exit_code, const char *format,...) const __attribute__((format(printf	149
6.137.4.4 flush() const	149
6.137.4.5 getOutputLocation() const	149
6.137.4.6 getPrefix() const	149
6.137.4.7 getVerboseLevel() const	149
6.137.4.8 getVerboseMask() const	150

6.137.4.9	<code>init(const std::string &prefix, uint32_t verbose_level, uint32_t verbose_mask, output_location_t location, std::string localoutputfilename="")</code>	150
6.137.4.10	<code>output(uint32_t line, const char *file, const char *func, const char *format,...) const __attribute__((format(printf</code>	151
6.137.4.11	<code>output(const char *format,...) const __attribute__((format(printf</code>	151
6.137.4.12	<code>setFileName(const std::string &filename)</code>	151
6.137.4.13	<code>setOutputLocation(output_location_t location)</code>	151
6.137.4.14	<code>setPrefix(const std::string &prefix)</code>	151
6.137.4.15	<code>setVerboseLevel(uint32_t verbose_level)</code>	152
6.137.4.16	<code>setVerboseMask(uint32_t verbose_mask)</code>	152
6.137.4.17	<code>verbose(uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) const __attribute__((format(printf</code>	152
6.137.4.18	<code>verbosePrefix(const char *tempPrefix, uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) __attribute__((format(printf</code>	153
6.137.5	Member Data Documentation	153
6.137.5.1	file	153
6.137.5.2	line	154
6.138	OverallOutputter Class Reference	154
6.139	SST::Params Class Reference	154
6.139.1	Detailed Description	156
6.139.2	Member Typedef Documentation	156
6.139.2.1	key_type	156
6.139.2.2	KeySet_t	156
6.139.3	Constructor & Destructor Documentation	156
6.139.3.1	Params()	156
6.139.3.2	Params(const Params &old)	157
6.139.4	Member Function Documentation	157
6.139.4.1	clear()	157
6.139.4.2	contains(const key_type &k)	157
6.139.4.3	count(const key_type &k)	157
6.139.4.4	empty() const	157
6.139.4.5	enableVerify(bool enable)	157

6.139.4.6 enableVerify()	157
6.139.4.7 find(const std::string &k, T default_value, bool &found) const	158
6.139.4.8 find(const std::string &k, std::string default_value, bool &found) const	158
6.139.4.9 find(const std::string &k, const char *default_value, bool &found) const	158
6.139.4.10 find(const std::string &k, T default_value) const	158
6.139.4.11 find(const std::string &k, std::string default_value) const	159
6.139.4.12 find(const std::string &k, const char *default_value) const	159
6.139.4.13 find(const std::string &k) const	159
6.139.4.14 find(const std::string &k, bool &found) const	159
6.139.4.15 find_array(const key_type &k, std::vector< T > &vec) const	160
6.139.4.16 find_prefix_params(const std::string &prefix) const	160
6.139.4.17 getParamName(uint32_t id)	160
6.139.4.18 insert(std::string key, std::string value, bool overwrite=true)	160
6.139.4.19 operator=(const Params &old)	160
6.139.4.20 popAllowedKeys()	161
6.139.4.21 print_all_params(std::ostream &os, std::string prefix="") const	161
6.139.4.22 pushAllowedKeys(const KeySet_t &keys)	161
6.139.4.23 size() const	161
6.139.4.24 verifyParam(const key_type &k) const	161
6.140 SST::PartitionComponent Class Reference	161
6.141 SST::PartitionGraph Class Reference	162
6.141.1 Member Function Documentation	162
6.141.1.1 print(std::ostream &os) const	162
6.142 SST::PartitionLink Class Reference	162
6.142.1 Member Function Documentation	163
6.142.1.1 getMinLatency() const	163
6.142.1.2 print(std::ostream &os) const	163
6.143 SST::PollingLinkQueue Class Reference	163
6.143.1 Detailed Description	163
6.143.2 Member Function Documentation	163

6.143.2.1 empty() override	163
6.143.2.2 front() override	164
6.143.2.3 insert(Activity *activity) override	164
6.143.2.4 pop() override	164
6.143.2.5 size() override	164
6.144SST::Activity::pq_less_time_priority Class Reference	164
6.144.1 Detailed Description	164
6.144.2 Member Function Documentation	165
6.144.2.1 operator()(const Activity *lhs, const Activity *rhs) const	165
6.144.2.2 operator()(const Activity &lhs, const Activity &rhs) const	165
6.145SST::Activity::pq_less_time_priority_order Class Reference	165
6.145.1 Detailed Description	165
6.145.2 Member Function Documentation	165
6.145.2.1 operator()(const Activity *lhs, const Activity *rhs) const	165
6.145.2.2 operator()(const Activity &lhs, const Activity &rhs) const	165
6.146SST::ELI::ProvidesCategory Class Reference	166
6.147SST::ELI::ProvidesDefaultInfo Class Reference	166
6.148SST::ELI::ProvidesInterface Class Reference	167
6.149SST::ELI::ProvidesParams Class Reference	167
6.150SST::ELI::ProvidesPorts Class Reference	168
6.151SST::ELI::ProvidesStats Class Reference	168
6.152SST::ELI::ProvidesSubComponentSlots Class Reference	168
6.153PyComponent Struct Reference	169
6.154PySubComponent Struct Reference	169
6.155SST::Core::PythonConfigGraphOutput Class Reference	170
6.156SST::RankInfo Class Reference	171
6.156.1 Member Function Documentation	171
6.156.1.1 inRange(const RankInfo &other) const	171
6.157SST::RankSyncParallelSkip Class Reference	171
6.157.1 Constructor & Destructor Documentation	172

6.157.1.1 RankSyncParallelSkip(RankInfo num_ranks, TimeConverter *minPartTC)	172
6.157.2 Member Function Documentation	172
6.157.2.1 exchangeLinkUntimedData(int thread, std::atomic< int > &msg_count) override	172
6.157.2.2 finalizeLinkConfigurations() override	172
6.157.2.3 prepareForComplete() override	172
6.157.2.4 registerLink(const RankInfo &to_rank, const RankInfo &from_rank, LinkId_t link← _id, Link *link) override	172
6.158SST::RankSyncSerialSkip Class Reference	173
6.158.1 Constructor & Destructor Documentation	173
6.158.1.1 RankSyncSerialSkip(TimeConverter *minPartTC)	173
6.158.2 Member Function Documentation	173
6.158.2.1 exchangeLinkUntimedData(int thread, std::atomic< int > &msg_count) override	173
6.158.2.2 finalizeLinkConfigurations() override	173
6.158.2.3 prepareForComplete() override	173
6.158.2.4 registerLink(const RankInfo &to_rank, const RankInfo &from_rank, LinkId_t link← _id, Link *link) override	174
6.159SST::Core::Serialization::pvt::raw_ptr_wrapper< TPtr > Class Template Reference	174
6.160SST::RegionInfo Class Reference	174
6.160.1 Member Function Documentation	175
6.160.1.1 getMergeInfo()	175
6.161SST::RegionInfo::RegionMergeInfo Class Reference	175
6.162SST::Interfaces::SimpleMem::Request Class Reference	175
6.162.1 Detailed Description	177
6.162.2 Member Typedef Documentation	177
6.162.2.1 dataVec	177
6.162.2.2 flags_t	177
6.162.2.3 id_t	177
6.162.3 Member Enumeration Documentation	177
6.162.3.1 Command	177
6.162.3.2 Flags	177
6.162.4 Constructor & Destructor Documentation	178

6.162.4.1 Request(Command cmd, Addr addr, size_t size, dataVec &data, flags_t flags=0, flags_t memFlags=0)	178
6.162.4.2 Request(Command cmd, Addr addr, size_t size, flags_t flags=0, flags_t memFlags=0)	178
6.162.4.3 Request(Command cmd, Addr addr, size_t size, dataVec &data, uint32_t Opc, flags_t flags=0, flags_t memFlags=0)	178
6.162.4.4 Request(Command cmd, Addr addr, size_t size, uint32_t Opc, flags_t flags=0, flags_t memFlags=0)	178
6.162.5 Member Function Documentation	178
6.162.5.1 getCustomOpc(void)	178
6.162.5.2 getFlags(void)	178
6.162.5.3 getInstructionPointer()	178
6.162.5.4 getMemFlags(void)	179
6.162.5.5 getVirtualAddress()	179
6.162.5.6 setFlags(flags_t inValue)	179
6.162.5.7 setInstructionPointer(const Addr newIP)	179
6.162.5.8 setMemFlags(flags_t inValue)	179
6.162.5.9 setPayload(const std::vector< uint8_t > &data_in)	179
6.162.5.10 setPayload(uint8_t *data_in, size_t len)	179
6.162.5.11 setVirtualAddress(const Addr newVA)	180
6.162.6 Member Data Documentation	180
6.162.6.1 addr	180
6.162.6.2 addrs	180
6.162.6.3 cmd	180
6.162.6.4 custOpc	180
6.162.6.5 data	180
6.162.6.6 flags	180
6.162.6.7 id	180
6.162.6.8 instrPtr	181
6.162.6.9 memFlags	181
6.162.6.10 size	181
6.162.6.11 virtualAddr	181

6.163SST::Interfaces::SimpleNetwork::Request Class Reference	181
6.163.1 Detailed Description	182
6.163.2 Member Enumeration Documentation	182
6.163.2.1 TraceType	182
6.163.3 Constructor & Destructor Documentation	182
6.163.3.1 Request()	182
6.163.4 Member Function Documentation	182
6.163.4.1 givePayload(Event *event)	182
6.163.4.2 inspectPayload()	183
6.163.4.3 takePayload()	183
6.163.5 Member Data Documentation	183
6.163.5.1 allow_adaptive	183
6.163.5.2 dest	183
6.163.5.3 head	183
6.163.5.4 size_in_bits	183
6.163.5.5 src	183
6.163.5.6 tail	184
6.163.5.7 vn	184
6.164SST::SelfLink Class Reference	184
6.164.1 Detailed Description	184
6.165SST::Core::Serialization::pvt::ser_array_wrapper< TPtr, IntType > Class Template Reference	184
6.166SST::Core::Serialization::pvt::ser_buffer_accessor Class Reference	185
6.167SST::Core::Serialization::pvt::ser_buffer_overrun Class Reference	185
6.168SST::Core::Serialization::pvt::ser_packer Class Reference	185
6.168.1 Member Function Documentation	186
6.168.1.1 pack_buffer(void *buf, int size)	186
6.169SST::Core::Serialization::pvt::ser_sizer Class Reference	186
6.170SST::Core::Serialization::pvt::ser_unpacker Class Reference	187
6.170.1 Member Function Documentation	187
6.170.1.1 unpack_buffer(void *buf, int size)	187

6.171 SST::Core::Serialization::serializable Class Reference	187
6.172 SST::Core::Serialization::serializable_builder Class Reference	188
6.173 SST::Core::Serialization::serializable_builder_impl< T > Class Template Reference	188
6.173.1 Member Data Documentation	189
6.173.1.1 cls_id_	189
6.174 SST::Core::Serialization::serializable_factory Class Reference	189
6.174.1 Member Function Documentation	190
6.174.1.1 add_builder(serializable_builder *builder, const char *name)	190
6.175 sprockit::serializable_ptr_type Class Reference	190
6.176 SST::Core::Serialization::serializable_type< T > Class Template Reference	190
6.177 SST::Core::Serialization::serialize< T, Enable > Class Template Reference	190
6.177.1 Detailed Description	191
6.178 SST::Core::Serialization::serialize< bool > Class Template Reference	191
6.178.1 Detailed Description	191
6.179 SST::Core::Serialization::serialize< pvt::raw_ptr_wrapper< TPtr > > Class Template Reference	191
6.179.1 Detailed Description	192
6.180 SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_↵ _if< std::is_fundamental< T >::value std::is_enum< T >::value >::type > Class Template Refer- ence	192
6.180.1 Detailed Description	192
6.181 SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_↵ _if< !std::is_fundamental< T >::value && !std::is_enum< T >::value >::type > Class Template Ref- erence	192
6.181.1 Detailed Description	193
6.182 SST::Core::Serialization::serialize< pvt::ser_array_wrapper< void, IntType > > Class Template Reference	193
6.182.1 Detailed Description	193
6.183 SST::Core::Serialization::serialize< serializable * > Class Template Reference	193
6.184 SST::Core::Serialization::serialize< SST::SparseVectorMap< keyT, classT > > Class Template Reference	194
6.185 SST::Core::Serialization::serialize< std::deque< T > > Class Template Reference	194
6.186 SST::Core::Serialization::serialize< std::list< T > > Class Template Reference	194
6.187 SST::Core::Serialization::serialize< std::map< Key, Value > > Class Template Reference	194

6.188 SST::Core::Serialization::serialize< std::pair< U, V > > Class Template Reference	195
6.188.1 Detailed Description	195
6.189 SST::Core::Serialization::serialize< std::set< T > > Class Template Reference	195
6.190 SST::Core::Serialization::serialize< std::string > Class Template Reference	195
6.191 SST::Core::Serialization::serialize< std::vector< T > > Class Template Reference	196
6.192 SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type > Class Template Reference	196
6.193 SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type > Class Template Reference	196
6.193.1 Detailed Description	196
6.194 SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type > Class Template Reference	197
6.195 SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type > Class Template Reference	197
6.195.1 Detailed Description	197
6.196 SST::Core::Serialization::serialize< T[N], typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type > Class Template Reference	197
6.196.1 Detailed Description	198
6.197 SST::Core::Serialization::serialize< T[N], typename std::enable_if< std::is_fundamental< T >::value && std::is_enum< T >::value >::type > Class Template Reference	198
6.197.1 Detailed Description	198
6.198 SST::Core::Serialization::serializer Class Reference	198
6.198.1 Detailed Description	199
6.199 SST::SharedRegion Class Reference	199
6.199.1 Member Function Documentation	200
6.199.1.1 getLocalShareID() const	200
6.199.1.2 getPtr() const	200
6.199.1.3 getRawPtr()	200
6.199.1.4 getSize() const	200
6.199.1.5 isReady() const	201
6.199.1.6 modifyRegion(size_t offset, size_t length, const void *data)	201
6.199.1.7 publish()	201
6.200 SST::SharedRegionImpl Class Reference	201

6.201 SST::SharedRegionInitializedMerger Class Reference	201
6.201.1 Member Function Documentation	202
6.201.1.1 merge(uint8_t *target, const uint8_t *newData, size_t size) override	202
6.202 SST::SharedRegionManager Class Reference	202
6.203 SST::SharedRegionManagerImpl Class Reference	203
6.204 SST::SharedRegionMerger Class Reference	203
6.204.1 Detailed Description	204
6.204.2 Member Function Documentation	204
6.204.2.1 merge(uint8_t *target, const uint8_t *newData, size_t size)	204
6.205 SST::Interfaces::SimpleMem Class Reference	204
6.205.1 Detailed Description	205
6.205.2 Member Typedef Documentation	205
6.205.2.1 Addr	205
6.205.3 Constructor & Destructor Documentation	205
6.205.3.1 SimpleMem(SST::Component *comp, Params &UNUSED(params))	205
6.205.3.2 SimpleMem(SST::ComponentId_t id, Params &UNUSED(params))	205
6.205.4 Member Function Documentation	205
6.205.4.1 getLink(void) const =0	205
6.205.4.2 initialize(const std::string &linkName, HandlerBase *handler=NULL)=0	206
6.205.4.3 recvInitData()	206
6.205.4.4 recvResponse(void)=0	206
6.205.4.5 sendInitData(Request *req)=0	206
6.205.4.6 sendInitData(SST::Event *ev)	206
6.205.4.7 sendRequest(Request *req)=0	206
6.206 SST::Interfaces::SimpleNetwork Class Reference	207
6.206.1 Detailed Description	208
6.206.2 Member Typedef Documentation	208
6.206.2.1 nid_t	208
6.206.3 Constructor & Destructor Documentation	208
6.206.3.1 SimpleNetwork(SST::Component *comp)	208

6.206.3.2 SimpleNetwork(SST::ComponentId_t id)	208
6.206.4 Member Function Documentation	208
6.206.4.1 complete(unsigned int UNUSED(phase)) override	208
6.206.4.2 finish() override	208
6.206.4.3 getEndpointID() const =0	208
6.206.4.4 getLinkBW() const =0	209
6.206.4.5 init(unsigned int UNUSED(phase)) override	209
6.206.4.6 initialize(const std::string &portName, const UnitAlgebra &link_bw, int vns, const UnitAlgebra &in_buf_size, const UnitAlgebra &out_buf_size)=0	209
6.206.4.7 isNetworkInitialized() const =0	209
6.206.4.8 recv(int vn)=0	209
6.206.4.9 recvInitData()=0	210
6.206.4.10 recvUntimedData()	210
6.206.4.11 requestToReceive(int vn)=0	210
6.206.4.12 send(Request *req, int vn)=0	210
6.206.4.13 sendInitData(Request *req)=0	211
6.206.4.14 sendUntimedData(Request *req)	211
6.206.4.15 setNotifyOnReceive(HandlerBase *functor)=0	211
6.206.4.16 setNotifyOnSend(HandlerBase *functor)=0	211
6.206.4.17 setup() override	211
6.206.4.18 spaceToSend(int vn, int num_bits)=0	211
6.207 SST::IMPL::Partition::SimplePartitioner Class Reference	212
6.207.1 Member Function Documentation	212
6.207.1.1 performPartition(PartitionGraph *graph) override	212
6.207.1.2 performPartition(ConfigGraph *graph) override	213
6.208 SimThreadInfo_t Struct Reference	213
6.209 SST::Simulation Class Reference	213
6.209.1 Detailed Description	215
6.209.2 Member Typedef Documentation	215
6.209.2.1 clockMap_t	215
6.209.2.2 oneShotMap_t	215

6.209.3 Member Enumeration Documentation	215
6.209.3.1 Mode_t	215
6.209.4 Member Function Documentation	215
6.209.4.1 complete()	215
6.209.4.2 createSimulation(Config *config, RankInfo my_rank, RankInfo num_ranks, SimTime_t min_part)	215
6.209.4.3 GetComponent(const ComponentId_t &id) const	216
6.209.4.4 GetComponentInfoMap(void)	216
6.209.4.5 GetComponentLinkMap(ComponentId_t id) const	216
6.209.4.6 getCurrentPriority() const	216
6.209.4.7 getCurrentSimCycle() const	216
6.209.4.8 getElapsedSimTime() const	216
6.209.4.9 getEndSimCycle() const	216
6.209.4.10 getExit() const	216
6.209.4.11 getFinalSimTime() const	216
6.209.4.12 getLocalMinimumNextActivityTime()	217
6.209.4.13 getNextActivityTime() const	217
6.209.4.14 getNextClockCycle(TimeConverter *tc, int priority=CLOCKPRIORITY)	217
6.209.4.15 getNumRanks() const	217
6.209.4.16 getOutputDirectory()	217
6.209.4.17 getRank() const	217
6.209.4.18 getSharedRegionManager()	217
6.209.4.19 getSimulation()	217
6.209.4.20 getSimulationMode() const	217
6.209.4.21 getSimulationOutput()	218
6.209.4.22 getStatisticsProcessingEngine(void) const	218
6.209.4.23 getTimeLord(void)	218
6.209.4.24 initialize()	218
6.209.4.25 insertActivity(SimTime_t time, Activity *ev)	218
6.209.4.26 sWireUpFinished()	218

6.209.4.27	performWireUp(ConfigGraph &graph, const RankInfo &myRank, SimTime_↵ t min_part)	218
6.209.4.28	printStatus(bool fullStatus)	218
6.209.4.29	processGraphInfo(ConfigGraph &graph, const RankInfo &myRank, SimTime_↵ t min_part)	218
6.209.4.30	registerClock(const std::string &freq, Clock::HandlerBase *handler, int priority)	219
6.209.4.31	registerOneShot(std::string timeDelay, OneShot::HandlerBase *handler, int priority)	219
6.209.4.32	requireEvent(std::string name)	219
6.209.4.33	reregisterClock(TimeConverter *tc, Clock::HandlerBase *handler, int priority)	219
6.209.4.34	setOutputDirectory(std::string &outDir)	219
6.209.4.35	setSignal(int signal)	219
6.209.4.36	setStopAtCycle(Config *cfg)	219
6.209.4.37	setup()	220
6.209.4.38	shutdown()	220
6.209.4.39	unregisterClock(TimeConverter *tc, Clock::HandlerBase *handler, int priority)	220
6.210	SST::SimulatorHeartbeat Class Reference	220
6.210.1	Detailed Description	220
6.210.2	Constructor & Destructor Documentation	220
6.210.2.1	SimulatorHeartbeat(Config *cfg, int this_rank, Simulation *sim, TimeConverter *period)	220
6.211	SST::ELI::SingleCtor< Base, Args > Struct Template Reference	221
6.212	slist Struct Reference	221
6.213	SST::SparseVectorMap< keyT, classT > Class Template Reference	221
6.214	SST::SparseVectorMap< keyT, keyT > Class Template Reference	222
6.215	SST::Core::ThreadSafe::Spinlock Class Reference	223
6.216	SST::SST_ELI_element_version_extraction Struct Reference	223
6.217	SST::RNG::SSTConstantDistribution Class Reference	223
6.217.1	Detailed Description	224
6.217.2	Constructor & Destructor Documentation	224
6.217.2.1	SSTConstantDistribution(double v)	224
6.217.2.2	~SSTConstantDistribution()	224
6.217.3	Member Function Documentation	224

6.217.3.1 <code>getMean()</code>	224
6.217.3.2 <code>getNextDouble()</code>	224
6.217.4 Member Data Documentation	225
6.217.4.1 <code>mean</code>	225
6.218SST::RNG::SSTDiscreteDistribution Class Reference	225
6.218.1 Detailed Description	225
6.218.2 Constructor & Destructor Documentation	225
6.218.2.1 <code>SSTDiscreteDistribution(const double *probs, const uint32_t probsCount)</code>	225
6.218.2.2 <code>SSTDiscreteDistribution(const double *probs, const uint32_t probsCount, SST↵ Random *baseDist)</code>	226
6.218.2.3 <code>~SSTDiscreteDistribution()</code>	226
6.218.3 Member Function Documentation	226
6.218.3.1 <code>getNextDouble()</code>	226
6.218.4 Member Data Documentation	226
6.218.4.1 <code>baseDistrib</code>	226
6.218.4.2 <code>deleteDistrib</code>	226
6.218.4.3 <code>probabilities</code>	226
6.218.4.4 <code>probCount</code>	227
6.219SST::SSTElementPythonModule Class Reference	227
6.219.1 Detailed Description	227
6.219.2 Constructor & Destructor Documentation	227
6.219.2.1 <code>SSTElementPythonModule(std::string library)</code>	227
6.219.3 Member Function Documentation	228
6.219.3.1 <code>createPrimaryModule(char *code=NULL, std::string filename="")</code>	228
6.220SST::SSTElementPythonModuleCode Class Reference	228
6.220.1 Detailed Description	228
6.220.2 Member Function Documentation	229
6.220.2.1 <code>addSubModule(const std::string &module_name, char *code=NULL, std::string filename="")</code>	229
6.220.2.2 <code>getFullModuleName()</code>	229
6.221 SST::SSTElementPythonModuleOldELI Class Reference	229

6.222SST::RNG::SSTExponentialDistribution Class Reference	230
6.222.1 Detailed Description	230
6.222.2 Constructor & Destructor Documentation	230
6.222.2.1 SSTExponentialDistribution(const double mn)	230
6.222.2.2 SSTExponentialDistribution(const double mn, SSTRandom *baseDist)	231
6.222.2.3 ~SSTExponentialDistribution()	231
6.222.3 Member Function Documentation	231
6.222.3.1 getLambda()	231
6.222.3.2 getNextDouble()	231
6.222.4 Member Data Documentation	231
6.222.4.1 baseDistrib	231
6.222.4.2 deleteDistrib	231
6.222.4.3 lambda	232
6.223SST::RNG::SSTGaussianDistribution Class Reference	232
6.223.1 Detailed Description	232
6.223.2 Constructor & Destructor Documentation	232
6.223.2.1 SSTGaussianDistribution(double mn, double sd)	232
6.223.2.2 SSTGaussianDistribution(double mn, double sd, SSTRandom *baseRNG)	233
6.223.2.3 ~SSTGaussianDistribution()	233
6.223.3 Member Function Documentation	233
6.223.3.1 getMean()	233
6.223.3.2 getNextDouble()	233
6.223.3.3 getStandardDev()	233
6.223.4 Member Data Documentation	234
6.223.4.1 baseDistrib	234
6.223.4.2 deleteDistrib	234
6.223.4.3 mean	234
6.223.4.4 stddev	234
6.223.4.5 unusedPair	234
6.223.4.6 usePair	234

6.224SST::SSTInfoConfig Class Reference	234
6.224.1 Detailed Description	235
6.224.2 Constructor & Destructor Documentation	235
6.224.2.1 SSTInfoConfig()	235
6.224.3 Member Function Documentation	235
6.224.3.1 debugEnabled() const	235
6.224.3.2 getElementsToProcessArray()	235
6.224.3.3 getFilterMap()	235
6.224.3.4 getOptionBits()	235
6.224.3.5 getXMLFilePath()	235
6.224.3.6 parseCmdLine(int argc, char *argv[])	235
6.225SST::SSTLibraryInfo Class Reference	236
6.225.1 Detailed Description	236
6.225.2 Constructor & Destructor Documentation	236
6.225.2.1 SSTLibraryInfo(const std::string &name)	236
6.225.3 Member Function Documentation	236
6.225.3.1 getLibraryName()	237
6.225.3.2 outputHumanReadable(std::ostream &os, int LibIndex)	237
6.225.3.3 outputXML(int Index, TiXmlNode *XMLParentElement)	237
6.226SST::IMPL::Partition::SSTLinearPartition Class Reference	237
6.226.1 Detailed Description	238
6.226.2 Constructor & Destructor Documentation	238
6.226.2.1 SSTLinearPartition(RankInfo rankCount, RankInfo my_rank, int verbosity)	238
6.226.3 Member Function Documentation	238
6.226.3.1 performPartition(PartitionGraph *graph) override	238
6.226.3.2 performPartition(ConfigGraph *graph) override	238
6.226.4 Member Data Documentation	239
6.226.4.1 partOutput	239
6.227SST::sstLongOpts_s Struct Reference	239
6.228SST::SSTModelDescription Class Reference	239

6.228.1 Detailed Description	239
6.228.2 Member Function Documentation	240
6.228.2.1 createConfigGraph() \equiv 0	240
6.229 SST::Core::Interprocess::SSTMutex Class Reference	240
6.230 SST::Partition::SSTPartitioner Class Reference	240
6.230.1 Detailed Description	241
6.230.2 Member Function Documentation	241
6.230.2.1 performPartition(PartitionGraph *UNUSED(graph))	241
6.230.2.2 performPartition(PartitionGraph *graph)	241
6.230.2.3 performPartition(ConfigGraph *UNUSED(graph))	241
6.230.2.4 performPartition(ConfigGraph *graph)	241
6.231 SST::RNG::SSTPoissonDistribution Class Reference	242
6.231.1 Detailed Description	242
6.231.2 Constructor & Destructor Documentation	242
6.231.2.1 SSTPoissonDistribution(const double mn)	242
6.231.2.2 SSTPoissonDistribution(const double mn, SSTRandom *baseDist)	242
6.231.2.3 \sim SSTPoissonDistribution()	243
6.231.3 Member Function Documentation	243
6.231.3.1 getLambda()	243
6.231.3.2 getNextDouble()	243
6.231.4 Member Data Documentation	243
6.231.4.1 baseDistrib	243
6.231.4.2 deleteDistrib	243
6.231.4.3 lambda	243
6.232 SST::RNG::SSTRandom Class Reference	244
6.232.1 Detailed Description	244
6.232.2 Constructor & Destructor Documentation	244
6.232.2.1 \sim SSTRandom()	244
6.232.3 Member Function Documentation	244
6.232.3.1 generateNextInt32() \equiv 0	244

6.232.3.2 generateNextInt64()=0	244
6.232.3.3 generateNextUInt32()=0	244
6.232.3.4 generateNextUInt64()=0	245
6.232.3.5 nextUniform()=0	245
6.233SST::RNG::SSTRandomDistribution Class Reference	245
6.233.1 Detailed Description	245
6.233.2 Constructor & Destructor Documentation	245
6.233.2.1 ~SSTRandomDistribution()	245
6.233.2.2 SSTRandomDistribution()	245
6.233.3 Member Function Documentation	246
6.233.3.1 getNextDouble()=0	246
6.234SST::IMPL::Partition::SSTRoundRobinPartition Class Reference	246
6.234.1 Member Function Documentation	246
6.234.1.1 performPartition(PartitionGraph *graph) override	246
6.234.1.2 performPartition(ConfigGraph *graph) override	247
6.235SST::IMPL::Partition::SSTSelfPartition Class Reference	247
6.235.1 Detailed Description	247
6.235.2 Member Function Documentation	247
6.235.2.1 performPartition(ConfigGraph *UNUSED(graph)) override	247
6.235.2.2 performPartition(PartitionGraph *UNUSED(graph)) override	248
6.235.2.3 SST_ELI_REGISTER_PARTITIONER(SSTSelfPartition,"sst","self", SST_ELI_ELEMENT_VERSION(1, 0, 0),"Used when partitioning is already specified in the configuration file.") SSTSelfPartition(RankInfo UNUSED(total_ranks)	248
6.236SST::IMPL::Partition::SSTSinglePartition Class Reference	248
6.236.1 Detailed Description	249
6.236.2 Member Function Documentation	249
6.236.2.1 performPartition(PartitionGraph *graph) override	249
6.236.2.2 performPartition(ConfigGraph *graph) override	249
6.236.2.3 SST_ELI_REGISTER_PARTITIONER(SSTSinglePartition,"sst","single", S↵ST_ELI_ELEMENT_VERSION(1, 0, 0),"Allocates all components to rank 0. Automatically selected for serial jobs.") SSTSinglePartition(RankInfo total_ranks	249
6.237SST::RNG::SSTUniformDistribution Class Reference	249

6.237.1 Detailed Description	250
6.237.2 Constructor & Destructor Documentation	250
6.237.2.1 SSTUniformDistribution(const uint32_t probsCount)	250
6.237.2.2 SSTUniformDistribution(const uint32_t probsCount, SSTRandom *baseDist)	250
6.237.2.3 ~SSTUniformDistribution()	250
6.237.3 Member Function Documentation	250
6.237.3.1 getNextDouble()	251
6.237.4 Member Data Documentation	251
6.237.4.1 baseDistrib	251
6.237.4.2 deleteDistrib	251
6.237.4.3 probCount	251
6.238 StatGroupPy_t Struct Reference	251
6.239 SST::Core::Serialization::statics Class Reference	252
6.240 SST::Statistics::Statistic< T > Class Template Reference	252
6.240.1 Detailed Description	252
6.240.2 Constructor & Destructor Documentation	253
6.240.2.1 Statistic(BaseComponent *comp, const std::string &statName, const std::string &statSubId, Params &statParams)	253
6.240.3 Member Function Documentation	253
6.240.3.1 addData(InArgs &&...args)	253
6.241 SST::Statistics::StatisticBase Class Reference	253
6.241.1 Detailed Description	255
6.241.2 Member Enumeration Documentation	255
6.241.2.1 StatMode_t	255
6.241.3 Constructor & Destructor Documentation	255
6.241.3.1 StatisticBase(BaseComponent *comp, const std::string &statName, const std::string &statSubId, Params &statParams)	255
6.241.4 Member Function Documentation	255
6.241.4.1 clearStatisticData()	255
6.241.4.2 delayCollection(const char *delayTime)	255
6.241.4.3 delayOutput(const char *delayTime)	256

6.241.4.4 disable()	256
6.241.4.5 enable()	256
6.241.4.6 getCollectionCount() const	256
6.241.4.7 getCollectionCountLimit() const	256
6.241.4.8 getCompName() const	256
6.241.4.9 getComponent() const	256
6.241.4.10 getFlagClearDataOnOutput() const	256
6.241.4.11 getFlagOutputAtEndOfSim() const	256
6.241.4.12 getFlagResetCountOnOutput() const	257
6.241.4.13 getFullStatName() const	257
6.241.4.14 getParams()	257
6.241.4.15 getRegisteredCollectionMode() const	257
6.241.4.16 getStatDataType() const	257
6.241.4.17 getStatDataTypeFullName() const	257
6.241.4.18 getStatDataTypeShortName() const	257
6.241.4.19 getStatName() const	257
6.241.4.20 getStatSubId() const	257
6.241.4.21 getStatTypeName() const	257
6.241.4.22 incrementCollectionCount()	258
6.241.4.23 isEnabled() const	258
6.241.4.24 isNullStatistic() const	258
6.241.4.25 isOutputEnabled() const	258
6.241.4.26 isReady() const	258
6.241.4.27 resetCollectionCount()	258
6.241.4.28 setCollectionCount(uint64_t newCount)	258
6.241.4.29 setCollectionCountLimit(uint64_t newLimit)	258
6.241.4.30 setFlagClearDataOnOutput(bool flag)	258
6.241.4.31 setFlagOutputAtEndOfSim(bool flag)	258
6.241.4.32 setFlagResetCountOnOutput(bool flag)	259
6.241.4.33 setStatisticDataType(const StatisticFieldInfo::fieldType_t dataType)	259

6.241.4.34	setStatisticTypeName(const char *typeName)	259
6.242	SST::Statistics::StatisticCollector< T, F > Class Template Reference	259
6.242.1	Detailed Description	259
6.243	SST::Statistics::StatisticCollector< std::tuple< Args... >, false > Struct Template Reference	259
6.244	SST::Statistics::StatisticCollector< T, true > Struct Template Reference	260
6.245	SST::Statistics::StatisticFieldInfo Class Reference	260
6.245.1	Detailed Description	260
6.245.2	Constructor & Destructor Documentation	260
6.245.2.1	StatisticFieldInfo(const char *statName, const char *fieldName, fieldType_↔ t fieldType)	260
6.245.3	Member Function Documentation	261
6.245.3.1	getFieldHandle()	261
6.245.3.2	getFieldName() const	261
6.245.3.3	getFieldType() const	261
6.245.3.4	getFieldUniqueName() const	261
6.245.3.5	getStatName() const	261
6.245.3.6	operator==(StatisticFieldInfo &FieldInfo1)	261
6.245.3.7	setFieldHandle(fieldHandle_t handle)	262
6.246	SST::Statistics::StatisticFieldType< T > Class Template Reference	262
6.247	SST::Statistics::StatisticFieldTypeBase Class Reference	262
6.248	SST::Statistics::StatisticGroup Class Reference	263
6.249	SST::Statistics::StatisticInfo Class Reference	264
6.250	SST::Statistics::StatisticOutput Class Reference	264
6.250.1	Detailed Description	266
6.250.2	Constructor & Destructor Documentation	266
6.250.2.1	StatisticOutput(Params &outputParameters)	266
6.250.3	Member Function Documentation	266
6.250.3.1	acceptsGroups() const	266
6.250.3.2	checkOutputParameters()=0	266
6.250.3.3	endOfSimulation()=0	266
6.250.3.4	getFieldInfoArray()	266

6.250.3.5	<code>getFieldTypeShortName(fieldType_t type)</code>	266
6.250.3.6	<code>getOutputParameters()</code>	267
6.250.3.7	<code>getRegisteredField(fieldHandle_t fieldHandle)</code>	267
6.250.3.8	<code>getRegisteredField(const char *statisticName, const char *fieldName)</code>	267
6.250.3.9	<code>getStatisticOutputName()</code>	267
6.250.3.10	<code>implStartOutputEntries(StatisticBase *statistic)=0</code>	268
6.250.3.11	<code>implStopOutputEntries()=0</code>	268
6.250.3.12	<code>outputField(fieldHandle_t fieldHandle, int32_t data)</code>	268
6.250.3.13	<code>printUsage()=0</code>	268
6.250.3.14	<code>registerField(const char *fieldName)</code>	268
6.250.3.15	<code>startOfSimulation()=0</code>	269
6.251	<code>SST::Statistics::StatisticOutputConsole Class Reference</code>	269
6.251.1	Detailed Description	270
6.251.2	Member Function Documentation	270
6.251.2.1	<code>checkOutputParameters()</code> override	270
6.251.2.2	<code>endOfSimulation()</code> override	270
6.251.2.3	<code>implStartOutputEntries(StatisticBase *statistic)</code> override	270
6.251.2.4	<code>implStopOutputEntries()</code> override	270
6.251.2.5	<code>outputField(fieldHandle_t fieldHandle, int32_t data)</code> override	271
6.251.2.6	<code>printUsage()</code> override	271
6.251.2.7	<code>SST_ELI_REGISTER_DERIVED(StatisticOutput, StatisticOutputConsole, "sst", "statoutputconsole", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Output directly to console screen")</code> <code>StatisticOutputConsole(Params &outputParameters)</code>	271
6.251.2.8	<code>startOfSimulation()</code> override	271
6.252	<code>SST::Statistics::StatisticOutputCSV Class Reference</code>	271
6.252.1	Detailed Description	272
6.252.2	Member Function Documentation	272
6.252.2.1	<code>checkOutputParameters()</code> override	272
6.252.2.2	<code>endOfSimulation()</code> override	272
6.252.2.3	<code>implStartOutputEntries(StatisticBase *statistic)</code> override	272
6.252.2.4	<code>implStopOutputEntries()</code> override	273

6.252.2.5 outputField(fieldHandle_t fieldHandle, int32_t data) override	273
6.252.2.6 printUsage() override	273
6.252.2.7 SST_ELI_REGISTER_DERIVED(StatisticOutput, StatisticOutputCSV, ""sst"", ""statoutputcsv"", SST_ELI_ELEMENT_VERSION(1, 0, 0), ""Output directly to console screen"") StatisticOutputCSV(Params &outputParameters)	273
6.252.2.8 startOfSimulation() override	274
6.253SST::Statistics::StatisticOutputHDF5 Class Reference	274
6.253.1 Detailed Description	275
6.253.2 Member Function Documentation	275
6.253.2.1 acceptsGroups() const override	275
6.253.2.2 checkOutputParameters() override	275
6.253.2.3 endOfSimulation() override	275
6.253.2.4 implStartOutputEntries(StatisticBase *statistic) override	275
6.253.2.5 implStopOutputEntries() override	276
6.253.2.6 outputField(fieldHandle_t fieldHandle, int32_t data) override	276
6.253.2.7 printUsage() override	276
6.253.2.8 SST_ELI_REGISTER_DERIVED(StatisticOutput, StatisticOutputHDF5, ""sst"", ""statoutputhdf5"", SST_ELI_ELEMENT_VERSION(1, 0, 0), ""Output to an HDF5 file"") Statistic↵ OutputHDF5(Params &outputParameters)	276
6.253.2.9 startOfSimulation() override	276
6.254SST::Statistics::StatisticOutputJSON Class Reference	277
6.254.1 Detailed Description	277
6.254.2 Member Function Documentation	277
6.254.2.1 checkOutputParameters() override	277
6.254.2.2 endOfSimulation() override	278
6.254.2.3 implStartOutputEntries(StatisticBase *statistic) override	278
6.254.2.4 implStopOutputEntries() override	278
6.254.2.5 outputField(fieldHandle_t fieldHandle, int32_t data) override	278
6.254.2.6 printUsage() override	278
6.254.2.7 SST_ELI_REGISTER_DERIVED(StatisticOutput, StatisticOutputJSON, ""sst"", ""statoutputjson"", SST_ELI_ELEMENT_VERSION(1, 0, 0), ""Output to a JSON file"") Statistic↵ OutputJSON(Params &outputParameters)	279
6.254.2.8 startOfSimulation() override	279

6.255 SST::Statistics::StatisticOutputTxt Class Reference	279
6.255.1 Detailed Description	280
6.255.2 Member Function Documentation	280
6.255.2.1 checkOutputParameters() override	280
6.255.2.2 endOfSimulation() override	280
6.255.2.3 implStartOutputEntries(StatisticBase *statistic) override	280
6.255.2.4 implStopOutputEntries() override	280
6.255.2.5 outputField(fieldHandle_t fieldHandle, int32_t data) override	281
6.255.2.6 printUsage() override	281
6.255.2.7 SST_ELI_REGISTER_DERIVED(StatisticOutput, StatisticOutputTxt, "sst", "statoutputtxt", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Output directly to console screen") StatisticOutputTxt(Params &outputParameters)	281
6.255.2.8 startOfSimulation() override	281
6.256 SST::Statistics::StatisticProcessingEngine Class Reference	281
6.256.1 Detailed Description	282
6.256.2 Member Function Documentation	282
6.256.2.1 performGlobalStatisticOutput(bool endOfSimFlag=false)	282
6.256.2.2 performStatisticOutput(StatisticBase *stat, bool endOfSimFlag=false)	282
6.257 StatOutputPy_t Struct Reference	283
6.258 SST::StopAction Class Reference	283
6.258.1 Detailed Description	283
6.258.2 Constructor & Destructor Documentation	283
6.258.2.1 StopAction(std::string msg)	283
6.258.3 Member Function Documentation	284
6.258.3.1 execute() override	284
6.258.3.2 print(const std::string &header, Output &out) const override	284
6.259 SST::Interfaces::StringEvent Class Reference	284
6.259.1 Detailed Description	284
6.259.2 Constructor & Destructor Documentation	284
6.259.2.1 StringEvent(const std::string &str)	284
6.259.2.2 StringEvent(const StringEvent &me)	285

6.259.2.3 StringEvent(const StringEvent *me)	285
6.259.3 Member Function Documentation	285
6.259.3.1 getString(void)	285
6.260SST::SubComponent Class Reference	285
6.260.1 Detailed Description	286
6.260.2 Member Function Documentation	286
6.260.2.1 finish() override	286
6.260.2.2 init(unsigned int UNUSED(phase)) override	286
6.260.2.3 setup() override	286
6.261SST::SubComponentSlotInfo Class Reference	286
6.261.1 Detailed Description	287
6.261.2 Member Function Documentation	287
6.261.2.1 createAll(std::vector< T * > &vec, uint64_t share_flags, ARGS...args) const	287
6.261.2.2 createAllSparse(std::vector< std::pair< int, T * > > &vec, uint64_t share_flags, ARGS...args) const	288
6.261.2.3 createAllSparse(std::vector< T * > &vec, uint64_t share_flags, ARGS...args) const	288
6.261.3 Member Data Documentation	288
6.261.3.1 const	288
6.261.3.2 insertNulls	289
6.262symlist_chain Struct Reference	289
6.263SST::SyncBase Class Reference	289
6.263.1 Detailed Description	290
6.263.2 Constructor & Destructor Documentation	290
6.263.2.1 SyncBase()	290
6.263.3 Member Function Documentation	290
6.263.3.1 exchangeLinkUntimedData(int msg_count)=0	290
6.263.3.2 finalizeLinkConfigurations()=0	290
6.263.3.3 registerLink(const RankInfo &to_rank, const RankInfo &from_rank, LinkId_t link↔_id, Link *link)=0	291
6.264SST::SyncManager Class Reference	291
6.264.1 Member Function Documentation	291

6.264.1.1	<code>exchangeLinkUntimedData(std::atomic< int > &msg_count)</code>	291
6.264.1.2	<code>execute(void)</code> override	291
6.264.1.3	<code>finalizeLinkConfigurations()</code>	292
6.264.1.4	<code>prepareForComplete()</code>	292
6.264.1.5	<code>print(const std::string &header, Output &out)</code> const override	292
6.264.1.6	<code>registerLink(const RankInfo &to_rank, const RankInfo &from_rank, LinkId_t link_id, Link *link)</code>	292
6.265	<code>SST::SyncQueue</code> Class Reference	292
6.265.1	Detailed Description	293
6.265.2	Member Function Documentation	293
6.265.2.1	<code>clear()</code>	293
6.265.2.2	<code>empty()</code> override	293
6.265.2.3	<code>front()</code> override	293
6.265.2.4	<code>getData()</code>	293
6.265.2.5	<code>insert(Activity *activity)</code> override	293
6.265.2.6	<code>pop()</code> override	293
6.265.2.7	<code>size()</code> override	294
6.266	<code>SST::Interfaces::TestEvent</code> Class Reference	294
6.266.1	Detailed Description	294
6.266.2	Member Data Documentation	294
6.266.2.1	<code>count</code>	294
6.266.2.2	<code>print_on_delete</code>	295
6.267	<code>SST::ThreadSync</code> Class Reference	295
6.267.1	Constructor & Destructor Documentation	295
6.267.1.1	<code>ThreadSync(int num_threads, Simulation *sim)</code>	295
6.267.2	Member Function Documentation	295
6.267.2.1	<code>execute(void)</code> override	295
6.267.2.2	<code>finalizeLinkConfigurations()</code>	296
6.267.2.3	<code>print(const std::string &header, Output &out)</code> const override	296
6.267.2.4	<code>processLinkUntimedData()</code>	296
6.267.2.5	<code>registerLink(LinkId_t link_id, Link *link)</code>	296

6.268SST::ThreadSyncQueue Class Reference	296
6.268.1 Detailed Description	296
6.268.2 Member Function Documentation	297
6.268.2.1 empty() override	297
6.268.2.2 front() override	297
6.268.2.3 insert(Activity *activity) override	297
6.268.2.4 pop() override	297
6.268.2.5 size() override	297
6.269SST::ThreadSyncSimpleSkip Class Reference	297
6.269.1 Constructor & Destructor Documentation	298
6.269.1.1 ThreadSyncSimpleSkip(int num_threads, int thread, Simulation *sim)	298
6.269.2 Member Function Documentation	298
6.269.2.1 finalizeLinkConfigurations() override	298
6.269.2.2 processLinkUntimedData() override	298
6.269.2.3 registerLink(LinkId_t link_id, Link *link) override	298
6.270SST::TimeConverter Class Reference	299
6.270.1 Detailed Description	299
6.270.2 Member Function Documentation	299
6.270.2.1 convertFromCoreTime(SimTime_t time)	299
6.270.2.2 convertToCoreTime(SimTime_t time)	299
6.270.2.3 getFactor()	299
6.271SST::TimeLord Class Reference	300
6.271.1 Detailed Description	300
6.271.2 Member Function Documentation	300
6.271.2.1 getMicro()	300
6.271.2.2 getMilli()	300
6.271.2.3 getNano()	300
6.271.2.4 getSimCycles(std::string timeString, std::string where)	300
6.271.2.5 getTimeBase() const	300
6.271.2.6 getTimeConverter(std::string ts)	300

6.271.2.7 getTimeConverter(const UnitAlgebra &ts)	301
6.272SST::TimeVortex Class Reference	301
6.272.1 Detailed Description	301
6.272.2 Member Function Documentation	302
6.272.2.1 empty() override=0	302
6.272.2.2 front() override=0	302
6.272.2.3 insert(Activity *activity) override=0	302
6.272.2.4 pop() override=0	302
6.272.2.5 print(Output &out) const =0	302
6.272.2.6 size() override=0	302
6.273SST::IMPL::TimeVortexPQ Class Reference	303
6.273.1 Detailed Description	303
6.273.2 Member Function Documentation	303
6.273.2.1 empty() override	303
6.273.2.2 front() override	303
6.273.2.3 insert(Activity *activity) override	303
6.273.2.4 pop() override	304
6.273.2.5 print(Output &out) const override	304
6.273.2.6 size() override	304
6.274TiXmlAttribute Class Reference	304
6.274.1 Detailed Description	305
6.274.2 Member Function Documentation	305
6.274.2.1 Print(FILE *cfile, int depth) const	305
6.274.2.2 QueryIntValue(int *_value) const	306
6.275TiXmlAttributeSet Class Reference	306
6.276TiXmlBase Class Reference	306
6.276.1 Detailed Description	308
6.276.2 Member Function Documentation	308
6.276.2.1 EncodeString(const TIXML_STRING &str, TIXML_STRING *out)	308
6.276.2.2 Print(FILE *cfile, int depth) const =0	308

6.276.2.3 Row() const	309
6.276.2.4 SetCondenseWhiteSpace(bool condense)	309
6.276.3 Member Data Documentation	309
6.276.3.1 errorString	309
6.276.3.2 utf8ByteTable	310
6.277TiXmlComment Class Reference	310
6.277.1 Detailed Description	311
6.277.2 Member Function Documentation	311
6.277.2.1 Accept(TiXmlVisitor *visitor) const	311
6.277.2.2 Print(FILE *cfile, int depth) const	311
6.278TiXmlCursor Struct Reference	311
6.279TiXmlDeclaration Class Reference	312
6.279.1 Detailed Description	312
6.279.2 Member Function Documentation	313
6.279.2.1 Accept(TiXmlVisitor *visitor) const	313
6.279.2.2 Print(FILE *cfile, int depth) const	313
6.280TiXmlDocument Class Reference	313
6.280.1 Detailed Description	314
6.280.2 Member Function Documentation	314
6.280.2.1 Accept(TiXmlVisitor *content) const	314
6.280.2.2 ClearError()	314
6.280.2.3 Clone() const	315
6.280.2.4 Error() const	315
6.280.2.5 ErrorId() const	315
6.280.2.6 ErrorRow() const	315
6.280.2.7 LoadFile(TiXmlEncoding encoding=TIXML_DEFAULT_ENCODING)	315
6.280.2.8 LoadFile(FILE *, TiXmlEncoding encoding=TIXML_DEFAULT_ENCODING)	315
6.280.2.9 Parse(const char *p, TiXmlParsingData *data=0, TiXmlEncoding encoding=TI↵ XML_DEFAULT_ENCODING)	315
6.280.2.10Print() const	316
6.280.2.11RootElement() const	316

6.280.2.12SetTabSize(int _tabsize)	316
6.281 TiXmlElement Class Reference	316
6.281.1 Detailed Description	317
6.281.2 Member Function Documentation	318
6.281.2.1 Accept(TiXmlVisitor *visitor) const	318
6.281.2.2 Attribute(const char *name) const	318
6.281.2.3 Attribute(const char *name, int *i) const	318
6.281.2.4 Attribute(const char *name, double *d) const	318
6.281.2.5 GetText() const	318
6.281.2.6 Print(FILE *cfile, int depth) const	319
6.281.2.7 QueryBoolAttribute(const char *name, bool *_value) const	319
6.281.2.8 QueryIntAttribute(const char *name, int *_value) const	319
6.281.2.9 RemoveAttribute(const char *name)	319
6.281.2.10SetAttribute(const char *name, const char *_value)	319
6.281.2.11SetAttribute(const char *name, int value)	319
6.281.2.12SetDoubleAttribute(const char *name, double value)	319
6.282 TiXmlHandle Class Reference	320
6.282.1 Detailed Description	320
6.282.2 Member Function Documentation	321
6.282.2.1 Child(const char *value, int index) const	321
6.282.2.2 Child(int index) const	322
6.282.2.3 ChildElement(const char *value, int index) const	322
6.282.2.4 ChildElement(int index) const	322
6.282.2.5 Element() const	322
6.282.2.6 Node() const	322
6.282.2.7 Text() const	322
6.282.2.8 ToElement() const	322
6.282.2.9 ToNode() const	322
6.282.2.10ToText() const	322
6.282.2.11ToUnknown() const	323

6.282.2.12Unknown() const	323
6.283TiXmlNode Class Reference	323
6.283.1 Detailed Description	325
6.283.2 Member Enumeration Documentation	326
6.283.2.1 NodeType	326
6.283.3 Member Function Documentation	326
6.283.3.1 Accept(TiXmlVisitor *visitor) const =0	326
6.283.3.2 Clone() const =0	326
6.283.3.3 FirstChild(const char *value) const	326
6.283.3.4 GetDocument() const	326
6.283.3.5 InsertAfterChild(TiXmlNode *afterThis, const TiXmlNode &addThis)	327
6.283.3.6 InsertBeforeChild(TiXmlNode *beforeThis, const TiXmlNode &addThis)	327
6.283.3.7 InsertEndChild(const TiXmlNode &addThis)	327
6.283.3.8 IterateChildren(const TiXmlNode *previous) const	327
6.283.3.9 LinkEndChild(TiXmlNode *addThis)	327
6.283.3.10NextSiblingElement() const	327
6.283.3.11NextSiblingElement(const char *) const	328
6.283.3.12ReplaceChild(TiXmlNode *replaceThis, const TiXmlNode &withThis)	328
6.283.3.13SetValue(const char *_value)	328
6.283.3.14Type() const	328
6.283.3.15Value() const	328
6.284TiXmlOutputStream Class Reference	328
6.285TiXmlParsingData Class Reference	329
6.286TiXmlPrinter Class Reference	329
6.286.1 Detailed Description	330
6.286.2 Member Function Documentation	331
6.286.2.1 SetIndent(const char *_indent)	331
6.286.2.2 SetLineBreak(const char *_lineBreak)	331
6.286.2.3 SetStreamPrinting()	331
6.287TiXmlString Class Reference	331

6.288TiXmlText Class Reference	332
6.288.1 Detailed Description	333
6.288.2 Constructor & Destructor Documentation	333
6.288.2.1 TiXmlText(const char *initValue)	333
6.288.3 Member Function Documentation	333
6.288.3.1 Accept(TiXmlVisitor *content) const	333
6.288.3.2 Print(FILE *cfile, int depth) const	333
6.289TiXmlUnknown Class Reference	333
6.289.1 Detailed Description	334
6.289.2 Member Function Documentation	334
6.289.2.1 Accept(TiXmlVisitor *content) const	334
6.289.2.2 Print(FILE *cfile, int depth) const	334
6.290TiXmlVisitor Class Reference	335
6.290.1 Detailed Description	335
6.291SST::Tokenizer< TokenizerFunc > Class Template Reference	336
6.292SST::TraceFunction Class Reference	336
6.292.1 Member Function Documentation	336
6.292.1.1 output(const char *format,...) const __attribute__((format(printf	336
6.293SST::Core::ThreadSafe::UnboundedQueue< T > Class Template Reference	336
6.294SST::UninitializedQueue Class Reference	337
6.294.1 Detailed Description	337
6.294.2 Constructor & Destructor Documentation	337
6.294.2.1 UninitializedQueue(std::string message)	337
6.294.3 Member Function Documentation	338
6.294.3.1 empty() override	338
6.294.3.2 front() override	338
6.294.3.3 insert(Activity *activity) override	338
6.294.3.4 pop() override	338
6.294.3.5 size() override	338
6.295SST::Statistics::UniqueCountStatistic< T > Class Template Reference	338

6.295.1 Detailed Description	339
6.295.2 Member Function Documentation	339
6.295.2.1 addData_impl(T data) override	339
6.295.3 Member Data Documentation	339
6.295.3.1 statParams	340
6.296SST::UnitAlgebra Class Reference	340
6.296.1 Detailed Description	341
6.296.2 Constructor & Destructor Documentation	341
6.296.2.1 UnitAlgebra(std::string val)	341
6.296.3 Member Function Documentation	341
6.296.3.1 getRoundedValue() const	341
6.296.3.2 getValue() const	341
6.296.3.3 hasUnits(std::string u) const	341
6.296.3.4 invert()	341
6.296.3.5 operator*=(const UnitAlgebra &v)	341
6.296.3.6 operator*=(const T &v)	342
6.296.3.7 operator+=(const UnitAlgebra &v)	342
6.296.3.8 operator+=(const T &v)	342
6.296.3.9 operator-=(const UnitAlgebra &v)	342
6.296.3.10operator-=(const T &v)	342
6.296.3.11operator/=(const UnitAlgebra &v)	342
6.296.3.12operator/=(const T &v)	342
6.296.3.13operator<(const UnitAlgebra &v) const	342
6.296.3.14operator<=(const UnitAlgebra &v) const	342
6.296.3.15operator>(const UnitAlgebra &v) const	342
6.296.3.16operator>=(const UnitAlgebra &v) const	343
6.296.3.17print(std::ostream &stream)	343
6.296.3.18printWithBestSI(std::ostream &stream)	343
6.296.3.19toString() const	343
6.296.3.20toStringBestSI() const	343

6.297SST::Units Class Reference	343
6.297.1 Detailed Description	344
6.297.2 Constructor & Destructor Documentation	344
6.297.2.1 Units(std::string units, sst_big_num &multiplier)	344
6.297.3 Member Function Documentation	344
6.297.3.1 invert()	344
6.297.3.2 operator!=(const Units &lhs) const	344
6.297.3.3 operator*=(const Units &v)	344
6.297.3.4 operator/=(const Units &v)	344
6.297.3.5 operator=(const Units &v)	344
6.297.3.6 operator==(const Units &lhs) const	345
6.297.3.7 registerBaseUnit(std::string u)	345
6.297.3.8 registerCompoundUnit(std::string u, std::string v)	345
6.297.3.9 toString() const	345
6.298SST::Core::XMLConfigGraphOutput Class Reference	345
6.299SST::RNG::XORShiftRNG Class Reference	346
6.299.1 Detailed Description	346
6.299.2 Constructor & Destructor Documentation	346
6.299.2.1 XORShiftRNG(unsigned int seed)	346
6.299.2.2 XORShiftRNG()	346
6.299.2.3 ~XORShiftRNG()	347
6.299.3 Member Function Documentation	347
6.299.3.1 generateNextInt32() override	347
6.299.3.2 generateNextInt64() override	347
6.299.3.3 generateNextUInt32() override	347
6.299.3.4 generateNextUInt64() override	347
6.299.3.5 nextUniform() override	347
6.299.3.6 seed(uint64_t newSeed)	347

Chapter 1

Main Page

Structural Simulation Toolkit (SST)

What it does.

Understanding the performance of large-scale, capability-class , high performance computing (HPC) systems is possibly the most significant challenge in the development of such a system. Because it is impractical to construct prototype systems for evaluation at the desired scale, architects must turn to analytical models and simulation to guide design decisions. Historically, the architecture community has lacked the tools needed for a reliable and integrated evaluation of future architectures and workloads.

The Structural Simulation Toolkit (SST) aims address this problem. The SST provides a framework for simulating large-scale HPC systems. This simulator allows parallel simulation of large machines at multiple levels of detail. The SST couples models for processors, memory, and network subsystems. The SST aims, over time, to become a standard simulation environment for designing HPC systems by helping Industry, Academia, and the National Labs in designing and evaluating future architectures.

Key Interfaces for Component Writers

A quick overview of some key interfaces and data structures in SST can be found here: [Key Interfaces](#)

Install

Installation instructions can be found [here](#).

Tutorials

[Tutorials](#)

License

Copyright 2009–2019 NTESS. Under the terms of Contract DE-NA0003525 with NTESS, the U.S. Government retains certain rights in this software.

Copyright (c) 2009–2019, NTESS

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- * Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- * Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- * Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Contact

sst@sandia.gov

Chapter 2

Key Interfaces

The Component

The most important class for is `SST::Component`, the base class from which all simulation components inherit. At the very least, a component writer must create a class which inherits from `SST::Component` and which contains a constructor. This class should be created in a dynamic library (.so or .dylib file) and should include a C-style function which returns a pointer to a newly instantiated component.

`SST::Component` also contains useful functions for component setup (`SST::Component::Setup()` and `SST::Component::Init()`), cleanup (`SST::Component::Finish()`), controlling when the simulation stops (`SST::Component::registerExit()` and `SST::Component::unregisterExit()`), and for handling time (such as `SST::Component::getCurrentSimTime()`).

Making Event Handlers

SST components use event handling functors to handle interactions with other components (i.e. through an `SST::Event` sent over an `SST::Link`) and recurring events (i.e. component clock ticks). The Event Handler Class, `SST::EventHandler`, is templated to create either type of event handler by creating a functor which invokes a given member function whenever triggered. For example:

```
NICEventHandler = new Event::Handler<proc>
    (this, &proc::handle_nic_events);
...
bool proc::handle_nic_events(Event *event) {
    ...
}
```

creates an event handler which calls `proc::handle_nic_events()` with an argument of type `Event*` - the `SST::Event` to be processed. Similarly,

```
clockHandler = new Clock::Handler<proc>(this, &proc::preTic());
```

creates an clock handler which invokes the function `proc::preTic()` at a regular interval.

Using Event Handlers

Once created, an `SST::EventHandler` must be registered with the simulation core. This can be done with the `SST::Component::configureLink()` function for events coming from another component, or by `SST::Component::registerClock()`, for event handlers triggered by a clock. For example, the handlers created above could be registered in this way:

```
configureLink("mem0", NICEventHandler)
registerClock( "1Ghz", clockHandler );
```

Note that `SST::Component::registerClock()` can have its period or frequency expressed in SI units in a string. The allowed units are specified in `SST::TimeLord::getTimeConverter()` function.

Also note that the `SST::LinkMap::LinkAdd()` function does not require an event handler if the receiving component uses the "event pull" mechanism with `SST::Link::Recv()`.

Links

`SST::Component`s use `SST::Link` to communicate by passing events. An `SST::Link` is specified in the Python file used to configure the simulation. For example,

```
comp_A = sst.Component("cA", "simpleElementExample.simpleComponent")
comp_A.addParams({
    "workPerCycle" : ""1000"",
    "commSize" : ""100"",
})
comp_B = sst.Component("cB", "simpleElementExample.simpleComponent")
comp_B.addParams({
    "workPerCycle" : ""1000"",
    "commSize" : ""100"",
})

link_AB = sst.Link("link_AB")
link_AB.connect( (comp_A, "Slink", "10000ps"), (comp_B, "Nlink", "10000ps") )
```

specifies two components, `comp_A` and `comp_B`. These components are connected by an `SST::Link`. Each `link.connect` contains a pair of structures which specify the endpoint, port name, and latency of the link.

Other commonly used `SST::Link` functions are:

- `SST::Link::Send(CompEvent *event)` : Send an `SST::Event` across the link.
- `SST::Link::Send(SimTime_t delay, CompEvent *event)` : Send an `SST::Event` with an additional delay, where the delay is in units of the link frequency.
- `SST::Link::Send(SimTime_t delay, TimeConverter *tc, CompEvent event)` : Send an `SST::Event` with additional delay, where the delay is specified in units of the `SST::TimeConverter` object supplied.
- `SST::Link::Recv()` : Pull a pending `SST::Event` from the `SST::Link`. If there is no event handler registered with the Link, this function should be used to gather incoming events from the link. (I.e. a component "pulls" events from the link, rather than having them "pushed" into an event handler).

Chapter 3

Deprecated List

Member `TiXmlHandle::Element () const`

use `ToElement`. Return the handle as a `TiXmlElement`. This may return null.

Member `TiXmlHandle::Node () const`

use `ToNode`. Return the handle as a `TiXmlNode`. This may return null.

Member `TiXmlHandle::Text () const`

use `ToText()` Return the handle as a `TiXmlText`. This may return null.

Member `TiXmlHandle::Unknown () const`

use `ToUnknown()` Return the handle as a `TiXmlUnknown`. This may return null.

Chapter 4

Hierarchical Index

4.1 Class Hierarchy

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

SST::ActivityQueue	27
SST::InitQueue	113
SST::PollingLinkQueue	163
SST::SyncQueue	292
SST::ThreadSyncQueue	296
SST::TimeVortex	301
SST::IMPL::TimeVortexPQ	303
SST::UninitializedQueue	337
SST::ELI::Allocator< Base, T, Enable >	29
SST::Core::ThreadSafe::Barrier	29
SST::BaseComponent	30
SST::Component	47
SST::ComponentExtension	49
SST::SubComponent	285
SST::Interfaces::SimpleMem	204
SST::Interfaces::SimpleNetwork	207
SST::Interfaces::SimpleNetwork::NetworkInspector	138
SST::Core::ThreadSafe::BoundedQueue< T >	40
SST::Core::ThreadSafe::BoundedQueue< comm_recv_pair * >	40
SST::Core::ThreadSafe::BoundedQueue< comm_send_pair * >	40
SST::ELI::Builder< Base, Args >	40
SST::ELI::Builder< Base, Args... >	40
SST::ELI::DerivedBuilder< T, Base, Args >	75
SST::ELI::Builder< SSTELEMENTPythonModule, const std::string & >	40
SST::ELI::DerivedBuilder< SSTELEMENTPythonModule, SSTELEMENTPythonModuleOldELI, const std::string & >	75
SST::ELI::BuilderDatabase	41
SST::ELI::BuilderInfoImpl< void >	41
SST::ELI::BuilderLibrary< Base, CtorArgs >	42
SST::ELI::BuilderLibraryDatabase< Base, CtorArgs >	42
SST::ELI::CachedAllocator< Base, T >	43
SST::ELI::CachedAllocator< SSTELEMENTPythonModule, T >	43
SST::ELI::Allocator< SSTELEMENTPythonModule, T >	29
SST::char_delimiter	44

SST::Core::Interprocess::CircularBuffer< T >	45
ComponentHolder	49
PyComponent	169
PySubComponent	169
SST::ComponentInfo	50
SST::ComponentInfoMap	51
ComponentPy_t	51
SST::Core::ConfigGraphOutput	63
SST::Core::DotConfigGraphOutput	76
SST::Core::JSONConfigGraphOutput	118
SST::Core::PythonConfigGraphOutput	170
SST::Core::XMLConfigGraphOutput	345
SST::ELI::CtorList< Base, Ctor, Ctors >	67
SST::ELI::CtorList< Base, void >	67
SST::ELI::DataBase< T >	68
SST::decimal_fixedpoint< whole_words, fraction_words >	68
SST::decimal_fixedpoint< 3, 3 >	68
SST::ElementInfoParam	76
SST::ElementInfoPort	77
SST::ElementInfoPort2	78
SST::ElementInfoStatistic	79
SST::ElementInfoSubComponentSlot	79
SST::ELI::ElementsBuilder< Base, CtorTuple >	80
SST::ELI::ElementsBuilder< Base, std::tuple< Args... > >	80
SST::ELI::ElementsInfo< Base >	80
SST::ElemLoader	80
SST::Core::Environment::EnvironmentConfigGroup	83
SST::Core::Environment::EnvironmentConfiguration	83
SST::ComponentInfo::EqualsID	84
SST::ComponentInfo::EqualsName	84
SST::escaped_list_separator	84
exception	
SST::Core::ConfigGraphOutputException	63
SST::Core::Serialization::pvt::ser_buffer_overrun	185
SST::ELI::ExtendedCtor< NewCtor, OldCtor >	88
SST::Factory	89
false_type	
SST::ELI::is_tuple_constructible< T, U >	117
SST::ELI::GetParams< T, Enable >	94
SST::ELI::GetParams< T, typename MethodDetect< decltype(T::ELI_getParams())>::type >	95
SST::Clock::HandlerBase	105
SST::Clock::Handler< classT, argT >	95
SST::Clock::Handler< classT, void >	100
SST::OneShot::HandlerBase	106
SST::OneShot::Handler< classT, argT >	96
SST::OneShot::Handler< classT, void >	101
SST::Event::HandlerBase	106
SST::Event::Handler< classT, argT >	97
SST::Event::Handler< classT, void >	102
SST::Interfaces::SimpleMem::HandlerBase	107
SST::Interfaces::SimpleMem::Handler< classT, argT >	98
SST::Interfaces::SimpleMem::Handler< classT, void >	103
SST::Interfaces::SimpleNetwork::HandlerBase	107
SST::Interfaces::SimpleNetwork::Handler< classT, argT >	99
SST::Interfaces::SimpleNetwork::Handler< classT, void >	104
SST::ComponentInfo::HashID	108

SST::ComponentInfo::HashName	108
SST::SyncQueue::Header	108
SST::Statistics::ImplementsStatFields	110
SST::ELI::InfoDatabase	111
SST::ELI::InfoLibrary< Base >	111
SST::ELI::InfoLibraryDatabase< Base >	111
SST::ELI::InfoPorts< T, Enable >	112
SST::ELI::InfoPorts< T, typename MethodDetect< decltype(T::ELI_getPorts())>::type >	112
SST::ELI::InfoStats< T, Enable >	112
SST::ELI::InfoStats< T, typename MethodDetect< decltype(T::ELI_getStatistics())>::type >	112
SST::ELI::InfoSubs< T, Enable >	113
SST::ELI::InfoSubs< T, typename MethodDetect< decltype(T::ELI_getSubComponentSlots())>::type >	113
SST::ELI::InstantiateBuilder< Base, T >	114
SST::ELI::InstantiateBuilderInfo< Base, T >	115
SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >	115
is_constructible	
SST::ELI::is_tuple_constructible< T, std::tuple< Args... > >	118
SST::Activity::less_time	118
SST::Activity::less_time_priority	119
SST::Activity::less_time_priority_order	120
SST::Link	120
SST::SelfLink	184
SST::LinkMap	125
SST::LinkPair	127
LinkPy_t	128
SST::ELI::LoadedLibraries	128
SST::LoaderData	128
lt_advise	129
lt_handle	129
lt_interface_id	130
lt_dlnfo	130
lt_dlsymlist	131
lt_dlvtable	131
lt_interface_data	131
SST::Core::MemPool	134
SST::ELI::MethodDetect< T >	137
SST::Module	137
SST::Statistics::StatisticOutput	264
SST::Statistics::StatisticOutputConsole	269
SST::Statistics::StatisticOutputCSV	271
SST::Statistics::StatisticOutputHDF5	274
SST::Statistics::StatisticOutputJSON	277
SST::Statistics::StatisticOutputTxt	279
SST::SubComponent	285
ModuleLoaderPy_t	137
SST::Core::Serialization::need_delete_statics< T >	138
SST::NewRankSync	138
SST::EmptyRankSync	81
SST::RankSyncParallelSkip	171
SST::RankSyncSerialSkip	173
SST::NewThreadSync	139
SST::EmptyThreadSync	82
SST::ThreadSyncSimpleSkip	297
SST::ELI::NoValidConstructorsForDerivedType< NumValid >	140
SST::ELI::NoValidConstructorsForDerivedType< 0 >	141
SST::Statistics::NullStatisticBase< T, B >	142
SST::Statistics::NullStatisticBase< T >	142

SST::Statistics::NullStatistic< T >	142
SST::Output	145
OverallOutputter	154
SST::PartitionComponent	161
SST::PartitionGraph	162
SST::PartitionLink	162
Policy	
SST::ELI::BuilderInfoImpl< Policy, Policies >	41
SST::Activity::pq_less_time_priority	164
SST::Activity::pq_less_time_priority_order	165
SST::ELI::ProvidesCategory	166
SST::ELI::ProvidesDefaultInfo	166
SST::ELI::ProvidesInterface	167
SST::ELI::ProvidesParams	167
SST::ELI::ProvidesPorts	168
SST::ELI::ProvidesStats	168
SST::ELI::ProvidesSubComponentSlots	168
ptr_type	
sprockit::serializable_ptr_type	190
SST::Core::Serialization::pvt::raw_ptr_wrapper< TPtr >	174
SST::RegionInfo	174
SST::Interfaces::SimpleMem::Request	175
SST::Core::Serialization::pvt::ser_array_wrapper< TPtr, IntType >	184
SST::Core::Serialization::pvt::ser_buffer_accessor	185
SST::Core::Serialization::pvt::ser_packer	185
SST::Core::Serialization::pvt::ser_unpacker	187
SST::Core::Serialization::pvt::ser_sizer	186
serializable	
sprockit::serializable_ptr_type	190
SST::Core::Serialization::serializable	187
SST::Activity	26
SST::Action	25
SST::Clock	45
SST::Exit	87
SST::OneShot	144
SST::SimulatorHeartbeat	220
SST::StopAction	283
SST::SyncManager	291
SST::ThreadSync	295
SST::Event	85
SST::Interfaces::StringEvent	284
SST::Interfaces::TestEvent	294
SST::NullEvent	141
SST::ChangeSet	43
SST::Config	52
SST::ConfigComponent	57
SST::ConfigGraph	60
SST::ConfigLink	64
SST::ConfigStatGroup	65
SST::ConfigStatOutput	66
SST::Interfaces::SimpleNetwork::Request	181
SST::Params	154
SST::RankInfo	171
SST::RegionInfo::RegionMergeInfo	175
SST::RegionInfo::BulkMergeInfo	42
SST::RegionInfo::ChangeSetMergeInfo	44
SST::Statistics::StatisticInfo	264

SST::UnitAlgebra	340
SST::Core::Serialization::serializable_builder	188
SST::Core::Serialization::serializable_builder_impl< T >	188
SST::Core::Serialization::serializable_factory	189
SST::Core::Serialization::serializable_type< T >	190
SST::Core::Serialization::serializable_type< NullEvent >	190
SST::NullEvent	141
SST::Core::Serialization::serializable_type< Request >	190
SST::Interfaces::SimpleNetwork::Request	181
SST::Core::Serialization::serializable_type< StringEvent >	190
SST::Interfaces::StringEvent	284
SST::Core::Serialization::serializable_type< TestEvent >	190
SST::Interfaces::TestEvent	294
SST::Core::Serialization::serializable_type< UnitAlgebra >	190
SST::UnitAlgebra	340
SST::Core::Serialization::serialize< T, Enable >	190
SST::Core::Serialization::serialize< bool >	191
SST::Core::Serialization::serialize< pvt::raw_ptr_wrapper< TPtr > >	191
SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	192
SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if< !std::is_fundamental< T >::value && !std::is_enum< T >::value >::type >	192
SST::Core::Serialization::serialize< pvt::ser_array_wrapper< void, IntType > >	193
SST::Core::Serialization::serialize< serializable * >	193
SST::Core::Serialization::serialize< SST::SparseVectorMap< keyT, classT > >	194
SST::Core::Serialization::serialize< std::deque< T > >	194
SST::Core::Serialization::serialize< std::list< T > >	194
SST::Core::Serialization::serialize< std::map< Key, Value > >	194
SST::Core::Serialization::serialize< std::pair< U, V > >	195
SST::Core::Serialization::serialize< std::set< T > >	195
SST::Core::Serialization::serialize< std::string >	195
SST::Core::Serialization::serialize< std::vector< T > >	196
SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type >	196
SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	196
SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type >	197
SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	197
SST::Core::Serialization::serialize< T[N], typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	197
SST::Core::Serialization::serialize< T[N], typename std::enable_if< !std::is_fundamental< T >::value && !std::is_enum< T >::value >::type >	198
SST::Core::Serialization::serializer	198
SST::SharedRegion	199
SST::SharedRegionImpl	201
SST::SharedRegionManager	202
SST::SharedRegionManagerImpl	203
SST::SharedRegionMerger	203
SST::SharedRegionInitializedMerger	201
SimThreadInfo_t	213
SST::Simulation	213
SST::ELI::SingleCtor< Base, Args >	221
slist	221

SST::SparseVectorMap< keyT, classT >	221
SST::SparseVectorMap< ComponentId_t >	221
SST::SparseVectorMap< ComponentId_t, ConfigComponent >	221
SST::SparseVectorMap< ComponentId_t, PartitionComponent >	221
SST::SparseVectorMap< keyT, keyT >	222
SST::SparseVectorMap< LinkId_t, ConfigLink >	221
SST::SparseVectorMap< LinkId_t, PartitionLink >	221
SST::Core::ThreadSafe::Spinlock	223
SST::SST_ELI_element_version_extraction	223
SST::SSTElementPythonModule	227
SST::SSTElementPythonModuleOldELI	229
SST::SSTElementPythonModuleCode	228
SST::SSTInfoConfig	234
SST::SSTLibraryInfo	236
SST::sstLongOpts_s	239
SST::SSTModelDescription	239
SST::Core::Interprocess::SSTMutex	240
SST::Partition::SSTPartitioner	240
SST::IMPL::Partition::SimplePartitioner	212
SST::IMPL::Partition::SSTLinearPartition	237
SST::IMPL::Partition::SSTRoundRobinPartition	246
SST::IMPL::Partition::SSTSelfPartition	247
SST::IMPL::Partition::SSTSinglePartition	248
SST::RNG::SSTRandom	244
SST::RNG::MarsagliaRNG	131
SST::RNG::MersenneRNG	135
SST::RNG::XORShiftRNG	346
SST::RNG::SSTRandomDistribution	245
SST::RNG::SSTConstantDistribution	223
SST::RNG::SSTDiscreteDistribution	225
SST::RNG::SSTExponentialDistribution	230
SST::RNG::SSTGaussianDistribution	232
SST::RNG::SSTPoissonDistribution	242
SST::RNG::SSTUniformDistribution	249
StatGroupPy_t	251
SST::Core::Serialization::statics	252
SST::Statistics::StatisticBase	253
SST::Statistics::Statistic< T >	252
SST::Statistics::NullStatisticBase< T, false >	143
SST::Statistics::NullStatisticBase< T, true >	143
SST::Statistics::UniqueCountStatistic< T >	338
SST::Statistics::Statistic< BinDataType >	252
SST::Statistics::HistogramStatistic< BinDataType >	109
SST::Statistics::Statistic< NumberBase >	252
SST::Statistics::AccumulatorStatistic< NumberBase >	21
SST::Statistics::Statistic< std::tuple< Args... > >	252
SST::Statistics::NullStatisticBase< std::tuple< Args... >, false >	143
SST::Statistics::StatisticCollector< T, F >	259
SST::Statistics::StatisticCollector< BinDataType >	259
SST::Statistics::Statistic< BinDataType >	252
SST::Statistics::StatisticCollector< NumberBase >	259
SST::Statistics::Statistic< NumberBase >	252
SST::Statistics::StatisticCollector< std::tuple< Args... > >	259
SST::Statistics::Statistic< std::tuple< Args... > >	252
SST::Statistics::StatisticCollector< std::tuple< Args... >, false >	259

SST::Statistics::StatisticCollector< T >	259
SST::Statistics::Statistic< T >	252
SST::Statistics::StatisticCollector< T, true >	260
SST::Statistics::StatisticFieldInfo	260
SST::Statistics::StatisticFieldTypeBase	262
SST::Statistics::StatisticFieldType< T >	262
SST::Statistics::StatisticGroup	263
SST::Statistics::StatisticProcessingEngine	281
StatOutputPy_t	283
SST::SubComponentSlotInfo	286
symlist_chain	289
SST::SyncBase	289
SST::TimeConverter	299
SST::TimeLord	300
TiXmlAttributeSet	306
TiXmlBase	306
TiXmlAttribute	304
TiXmlNode	323
TiXmlComment	310
TiXmlDeclaration	312
TiXmlDocument	313
TiXmlElement	316
TiXmlText	332
TiXmlUnknown	333
TiXmlCursor	311
TiXmlHandle	320
TiXmlParsingData	329
TiXmlString	331
TiXmlOutputStream	328
TiXmlVisitor	335
TiXmlPrinter	329
SST::Tokenizer< TokenizerFunc >	336
SST::TraceFunction	336
SST::Core::ThreadSafe::UnboundedQueue< T >	336
SST::Core::ThreadSafe::UnboundedQueue< comm_recv_pair * >	336
SST::Units	343

Chapter 5

Class Index

5.1 Class List

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

SST::Statistics::AccumulatorStatistic< NumberBase >	21
SST::Action	25
SST::Activity	26
SST::ActivityQueue	27
SST::ELI::Allocator< Base, T, Enable >	29
SST::ELI::Allocator< SSTELEMENTPythonModule, T >	29
SST::Core::ThreadSafe::Barrier	29
SST::BaseComponent	30
SST::Core::ThreadSafe::BoundedQueue< T >	40
SST::ELI::Builder< Base, Args >	40
SST::ELI::BuilderDatabase	41
SST::ELI::BuilderInfoImpl< Policy, Policies >	41
SST::ELI::BuilderInfoImpl< void >	41
SST::ELI::BuilderLibrary< Base, CtorArgs >	42
SST::ELI::BuilderLibraryDatabase< Base, CtorArgs >	42
SST::RegionInfo::BulkMergeInfo	42
SST::ELI::CachedAllocator< Base, T >	43
SST::ChangeSet	43
SST::RegionInfo::ChangeSetMergeInfo	44
SST::char_delimiter	44
SST::Core::Interprocess::CircularBuffer< T >	45
SST::Clock	45
SST::Component	47
SST::ComponentExtension	49
ComponentHolder	49
SST::ComponentInfo	50
SST::ComponentInfoMap	51
ComponentPy_t	51
SST::Config	52
SST::ConfigComponent	57
SST::ConfigGraph	60
SST::Core::ConfigGraphOutput	63
SST::Core::ConfigGraphOutputException	63
SST::ConfigLink	64
SST::ConfigStatGroup	65

SST::ConfigStatOutput	66
SST::ELI::CtorList< Base, Ctor, Ctors >	67
SST::ELI::CtorList< Base, void >	67
SST::ELI::DataBase< T >	68
SST::decimal_fixedpoint< whole_words, fraction_words >	68
SST::ELI::DerivedBuilder< T, Base, Args >	75
SST::ELI::DerivedBuilder< SSTELEMENTPythonModule, SSTELEMENTPythonModuleOldELI, const std::string & >	75
SST::Core::DotConfigGraphOutput	76
SST::ElementInfoParam	76
SST::ElementInfoPort	77
SST::ElementInfoPort2	78
SST::ElementInfoStatistic	79
SST::ElementInfoSubComponentSlot	79
SST::ELI::ElementsBuilder< Base, CtorTuple >	80
SST::ELI::ElementsBuilder< Base, std::tuple< Args... > >	80
SST::ELI::ElementsInfo< Base >	80
SST::ElemLoader	80
SST::EmptyRankSync	81
SST::EmptyThreadSync	82
SST::Core::Environment::EnvironmentConfigGroup	83
SST::Core::Environment::EnvironmentConfiguration	83
SST::ComponentInfo::EqualsID	84
SST::ComponentInfo::EqualsName	84
SST::escaped_list_separator	84
SST::Event	85
SST::Exit	87
SST::ELI::ExtendedCtor< NewCtor, OldCtor >	88
SST::Factory	89
SST::ELI::GetParams< T, Enable >	94
SST::ELI::GetParams< T, typename MethodDetect< decltype(T::ELI_getParams())>::type >	95
SST::Clock::Handler< classT, argT >	95
SST::OneShot::Handler< classT, argT >	96
SST::Event::Handler< classT, argT >	97
SST::Interfaces::SimpleMem::Handler< classT, argT >	98
SST::Interfaces::SimpleNetwork::Handler< classT, argT >	99
SST::Clock::Handler< classT, void >	100
SST::OneShot::Handler< classT, void >	101
SST::Event::Handler< classT, void >	102
SST::Interfaces::SimpleMem::Handler< classT, void >	103
SST::Interfaces::SimpleNetwork::Handler< classT, void >	104
SST::Clock::HandlerBase	105
SST::OneShot::HandlerBase	106
SST::Event::HandlerBase	106
Functor classes for Event handling	106
SST::Interfaces::SimpleMem::HandlerBase	107
SST::Interfaces::SimpleNetwork::HandlerBase	107
SST::ComponentInfo::HashID	108
SST::ComponentInfo::HashName	108
SST::SyncQueue::Header	108
SST::Statistics::HistogramStatistic< BinDataType >	109
SST::Statistics::ImplementsStatFields	110
SST::ELI::InfoDatabase	111
SST::ELI::InfoLibrary< Base >	111
SST::ELI::InfoLibraryDatabase< Base >	111
SST::ELI::InfoPorts< T, Enable >	112
SST::ELI::InfoPorts< T, typename MethodDetect< decltype(T::ELI_getPorts())>::type >	112
SST::ELI::InfoStats< T, Enable >	112

SST::ELI::InfoStats< T, typename MethodDetect< decltype(T::ELI_getStatistics())>::type >	112
SST::ELI::InfoSubs< T, Enable >	113
SST::ELI::InfoSubs< T, typename MethodDetect< decltype(T::ELI_getSubComponentSlots())>::type >	113
SST::InitQueue	113
SST::ELI::InstantiateBuilder< Base, T >	114
SST::ELI::InstantiateBuilderInfo< Base, T >	115
SST::Core::Interprocess::IPTunnel< ShareDataType, MsgType >	115
SST::ELI::is_tuple_constructible< T, U >	117
SST::ELI::is_tuple_constructible< T, std::tuple< Args... > >	118
SST::Core::JSONConfigGraphOutput	118
SST::Activity::less_time	118
SST::Activity::less_time_priority	119
SST::Activity::less_time_priority_order	120
SST::Link	120
SST::LinkMap	125
SST::LinkPair	127
LinkPy_t	128
SST::ELI::LoadedLibraries	128
SST::LoaderData	128
lt_advice	129
lt_handle	129
lt_interface_id	130
lt_dlinfo	130
lt_dlsymlist	131
lt_dlvtable	131
lt_interface_data	131
SST::RNG::MarsagliaRNG	131
SST::Core::MemPool	134
SST::RNG::MersenneRNG	135
SST::ELI::MethodDetect< T >	137
SST::Module	137
ModuleLoaderPy_t	137
SST::Core::Serialization::need_delete_statics< T >	138
SST::Interfaces::SimpleNetwork::NetworkInspector	138
SST::NewRankSync	138
SST::NewThreadSync	139
SST::ELI::NoValidConstructorsForDerivedType< NumValid >	140
SST::ELI::NoValidConstructorsForDerivedType< 0 >	141
SST::NullEvent	141
SST::Statistics::NullStatistic< T >	142
SST::Statistics::NullStatisticBase< T, B >	142
SST::Statistics::NullStatisticBase< std::tuple< Args... >, false >	143
SST::Statistics::NullStatisticBase< T, false >	143
SST::Statistics::NullStatisticBase< T, true >	143
SST::OneShot	144
SST::Output	145
OverallOutputter	154
SST::Params	154
SST::PartitionComponent	161
SST::PartitionGraph	162
SST::PartitionLink	162
SST::PollingLinkQueue	163
SST::Activity::pq_less_time_priority	164
SST::Activity::pq_less_time_priority_order	165
SST::ELI::ProvidesCategory	166
SST::ELI::ProvidesDefaultInfo	166
SST::ELI::ProvidesInterface	167
SST::ELI::ProvidesParams	167

SST::ELI::ProvidesPorts	168
SST::ELI::ProvidesStats	168
SST::ELI::ProvidesSubComponentSlots	168
PyComponent	169
PySubComponent	169
SST::Core::PythonConfigGraphOutput	170
SST::RankInfo	171
SST::RankSyncParallelSkip	171
SST::RankSyncSerialSkip	173
SST::Core::Serialization::pvt::raw_ptr_wrapper< TPtr >	174
SST::RegionInfo	174
SST::RegionInfo::RegionMergeInfo	175
SST::Interfaces::SimpleMem::Request	175
SST::Interfaces::SimpleNetwork::Request	181
SST::SelfLink	184
SST::Core::Serialization::pvt::ser_array_wrapper< TPtr, IntType >	184
SST::Core::Serialization::pvt::ser_buffer_accessor	185
SST::Core::Serialization::pvt::ser_buffer_overrun	185
SST::Core::Serialization::pvt::ser_packer	185
SST::Core::Serialization::pvt::ser_sizer	186
SST::Core::Serialization::pvt::ser_unpacker	187
SST::Core::Serialization::serializable	187
SST::Core::Serialization::serializable_builder	188
SST::Core::Serialization::serializable_builder_impl< T >	188
SST::Core::Serialization::serializable_factory	189
sprockit::serializable_ptr_type	190
SST::Core::Serialization::serializable_type< T >	190
SST::Core::Serialization::serialize< T, Enable >	190
SST::Core::Serialization::serialize< bool >	191
SST::Core::Serialization::serialize< pvt::raw_ptr_wrapper< TPtr > >	191
SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	192
SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if< !std::is_fundamental< T >::value && !std::is_enum< T >::value >::type >	192
SST::Core::Serialization::serialize< pvt::ser_array_wrapper< void, IntType > >	193
SST::Core::Serialization::serialize< serializable * >	193
SST::Core::Serialization::serialize< SST::SparseVectorMap< keyT, classT > >	194
SST::Core::Serialization::serialize< std::deque< T > >	194
SST::Core::Serialization::serialize< std::list< T > >	194
SST::Core::Serialization::serialize< std::map< Key, Value > >	194
SST::Core::Serialization::serialize< std::pair< U, V > >	195
SST::Core::Serialization::serialize< std::set< T > >	195
SST::Core::Serialization::serialize< std::string >	195
SST::Core::Serialization::serialize< std::vector< T > >	196
SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type >	196
SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	196
SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type >	197
SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	197
SST::Core::Serialization::serialize< T[N], typename std::enable_if< std::is_fundamental< T >::value std::is_enum< T >::value >::type >	197
SST::Core::Serialization::serialize< T[N], typename std::enable_if< !std::is_fundamental< T >::value && !std::is_enum< T >::value >::type >	198
SST::Core::Serialization::serializer	198
SST::SharedRegion	199

SST::SharedRegionImpl	201
SST::SharedRegionInitializedMerger	201
SST::SharedRegionManager	202
SST::SharedRegionManagerImpl	203
SST::SharedRegionMerger	203
SST::Interfaces::SimpleMem	204
SST::Interfaces::SimpleNetwork	207
SST::IMPL::Partition::SimplePartitioner	212
SimThreadInfo_t	213
SST::Simulation	213
SST::SimulatorHeartbeat	220
SST::ELI::SingleCtor< Base, Args >	221
slist	221
SST::SparseVectorMap< keyT, classT >	221
SST::SparseVectorMap< keyT, keyT >	222
SST::Core::ThreadSafe::Spinlock	223
SST::SST_ELI_element_version_extraction	223
SST::RNG::SSTConstantDistribution	223
SST::RNG::SSTDiscreteDistribution	225
SST::SSTElementPythonModule	227
SST::SSTElementPythonModuleCode	228
SST::SSTElementPythonModuleOldELI	229
SST::RNG::SSTExponentialDistribution	230
SST::RNG::SSTGaussianDistribution	232
SST::SSTInfoConfig	234
SST::SSTLibraryInfo	236
SST::IMPL::Partition::SSTLinearPartition	237
SST::sstLongOpts_s	239
SST::SSTModelDescription	239
SST::Core::Interprocess::SSTMutex	240
SST::Partition::SSTPartitioner	240
SST::RNG::SSTPoissonDistribution	242
SST::RNG::SSTRandom	244
SST::RNG::SSTRandomDistribution	245
SST::IMPL::Partition::SSTRoundRobinPartition	246
SST::IMPL::Partition::SSTSelfPartition	247
SST::IMPL::Partition::SSTSinglePartition	248
SST::RNG::SSTUniformDistribution	249
StatGroupPy_t	251
SST::Core::Serialization::statics	252
SST::Statistics::Statistic< T >	252
SST::Statistics::StatisticBase	253
SST::Statistics::StatisticCollector< T, F >	259
SST::Statistics::StatisticCollector< std::tuple< Args... >, false >	259
SST::Statistics::StatisticCollector< T, true >	260
SST::Statistics::StatisticFieldInfo	260
SST::Statistics::StatisticFieldType< T >	262
SST::Statistics::StatisticFieldTypeBase	262
SST::Statistics::StatisticGroup	263
SST::Statistics::StatisticInfo	264
SST::Statistics::StatisticOutput	264
SST::Statistics::StatisticOutputConsole	269
SST::Statistics::StatisticOutputCSV	271
SST::Statistics::StatisticOutputHDF5	274
SST::Statistics::StatisticOutputJSON	277
SST::Statistics::StatisticOutputTxt	279
SST::Statistics::StatisticProcessingEngine	281
StatOutputPy_t	283

SST::StopAction	283
SST::Interfaces::StringEvent	284
SST::SubComponent	285
SST::SubComponentSlotInfo	286
symlist_chain	289
SST::SyncBase	289
SST::SyncManager	291
SST::SyncQueue	292
SST::Interfaces::TestEvent	294
SST::ThreadSync	295
SST::ThreadSyncQueue	296
SST::ThreadSyncSimpleSkip	297
SST::TimeConverter	299
SST::TimeLord	300
SST::TimeVortex	301
SST::IMPL::TimeVortexPQ	303
TiXmlAttribute	304
TiXmlAttributeSet	306
TiXmlBase	306
TiXmlComment	310
TiXmlCursor	311
TiXmlDeclaration	312
TiXmlDocument	313
TiXmlElement	316
TiXmlHandle	320
TiXmlNode	323
TiXmlOutputStream	328
TiXmlParsingData	329
TiXmlPrinter	329
TiXmlString	331
TiXmlText	332
TiXmlUnknown	333
TiXmlVisitor	335
SST::Tokenizer< TokenizerFunc >	336
SST::TraceFunction	336
SST::Core::ThreadSafe::UnboundedQueue< T >	336
SST::UninitializedQueue	
Used for debugging, and preventing accidentally sending messages into an incorrect queue	337
SST::Statistics::UniqueCountStatistic< T >	338
SST::UnitAlgebra	340
SST::Units	343
SST::Core::XMLConfigGraphOutput	345
SST::RNG::XORShiftRNG	346

Chapter 6

Class Documentation

6.1 SST::Statistics::AccumulatorStatistic< NumberBase > Class Template Reference

```
#include <stataccumulator.h>
```

Inheritance diagram for SST::Statistics::AccumulatorStatistic< NumberBase >:

Collaboration diagram for SST::Statistics::AccumulatorStatistic< NumberBase >:

Public Member Functions

- **SST_ELI_DECLARE_STATISTIC_TEMPLATE** ([AccumulatorStatistic](#), "sst", "[AccumulatorStatistic](#)", SST::T_ELI_ELEMENT_VERSION(1, 0, 0), "Accumulate all contributions to a statistic", "SST::Statistic<T>")
AccumulatorStatistic([BaseComponent](#) *comp)
- this **setStatisticTypeName** ("Accumulator")
- NumberBase [getSum](#) ()
- NumberBase [getMax](#) ()
- NumberBase [getMin](#) ()
- NumberBase [getSumSquared](#) ()
- NumberBase [getArithmeticMean](#) ()
- NumberBase [getVariance](#) ()
- NumberBase [getStandardDeviation](#) ()
- uint64_t [getCount](#) ()
- void [clearStatisticData](#) () override
- void [registerOutputFields](#) ([StatisticOutput](#) *statOutput) override
- void **outputStatisticData** ([StatisticOutput](#) *statOutput, bool UNUSED(EndOfSimFlag)) override
- bool **isStatModeSupported** ([StatisticBase::StatMode_t](#) mode) const override

Public Attributes

- const std::string & **statName**
- const std::string const std::string & **statSubId**
- const std::string const std::string [Params](#) & **statParams**: [Statistic](#)<NumberBase>(comp)
- const std::string const std::string [Params](#) **statName**
- const std::string const std::string [Params](#) **statSubId**
- const std::string const std::string [Params](#) **statParams**
- **m_sum_sq** = static_cast<NumberBase>(0)
- **m_min** = std::numeric_limits<NumberBase>::max()
- **m_max** = std::numeric_limits<NumberBase>::min()

Protected Member Functions

- void [addData_impl](#) (NumberBase value) override

Additional Inherited Members

6.1.1 Detailed Description

```
template<typename NumberBase>
class SST::Statistics::AccumulatorStatistic< NumberBase >
```

Allows the online gathering of statistical information about a single quantity. The basic statistics are captured online removing the need to keep a copy of the values of interest.

Template Parameters

<i>NumberBase</i>	A template for the basic numerical type of values
-------------------	---

6.1.2 Member Function Documentation

6.1.2.1 `template<typename NumberBase > void SST::Statistics::AccumulatorStatistic< NumberBase >::addData_impl (NumberBase value) [inline],[override],[protected]`

Present a new value to the class to be included in the statistics.

Parameters

<i>value</i>	New value to be presented
--------------	---------------------------

6.1.2.2 `template<typename NumberBase > void SST::Statistics::AccumulatorStatistic< NumberBase >::clearStatisticData () [inline],[override],[virtual]`

Inform the [Statistic](#) to clear its data

Reimplemented from [SST::Statistics::StatisticBase](#).

6.1.2.3 `template<typename NumberBase > NumberBase SST::Statistics::AccumulatorStatistic< NumberBase >::getArithmeticMean () [inline]`

Get the arithmetic mean of the values presented so far

Returns

The arithmetic mean of the values presented so far.

6.1.2.4 `template<typename NumberBase > uint64_t SST::Statistics::AccumulatorStatistic< NumberBase >::getCount
() [inline]`

Get a count of the number of elements presented to the statistics collection so far.

Returns

Count the number of values presented to the class.

6.1.2.5 `template<typename NumberBase > NumberBase SST::Statistics::AccumulatorStatistic< NumberBase
>::getMax() [inline]`

Provides the maximum value presented so far.

Returns

The maximum of values presented to the class so far

6.1.2.6 `template<typename NumberBase > NumberBase SST::Statistics::AccumulatorStatistic< NumberBase
>::getMin() [inline]`

Provides the minimum value presented so far.

Returns

The minimum of values presented to the class so far

6.1.2.7 `template<typename NumberBase > NumberBase SST::Statistics::AccumulatorStatistic< NumberBase
>::getStandardDeviation() [inline]`

Get the standard deviation of the values presented so far

Returns

The standard deviation of the values presented so far

6.1.2.8 `template<typename NumberBase > NumberBase SST::Statistics::AccumulatorStatistic< NumberBase
>::getSum() [inline]`

Provides the sum of the values presented so far.

Returns

The sum of values presented to the class so far.

6.1.2.9 `template<typename NumberBase > NumberBase SST::Statistics::AccumulatorStatistic< NumberBase >::getSumSquared () [inline]`

Provides the sum of each value squared presented to the class so far.

Returns

The sum of squared values presented to the class so far.

6.1.2.10 `template<typename NumberBase > NumberBase SST::Statistics::AccumulatorStatistic< NumberBase >::getVariance () [inline]`

Get the variance of the values presented so far

Returns

The variance of the values presented so far

6.1.2.11 `template<typename NumberBase > void SST::Statistics::AccumulatorStatistic< NumberBase >::registerOutputFields (StatisticOutput * statOutput) [inline],[override],[virtual]`

Called by the system to tell the [Statistic](#) to register its output fields. by calling `statOutput->registerField(...)`

Parameters

<i>statOutput</i>	- Pointer to the statistic output
-------------------	-----------------------------------

Implements [SST::Statistics::StatisticBase](#).

6.1.3 Member Data Documentation

6.1.3.1 `template<typename NumberBase > const std::string const std::string Params SST::Statistics::AccumulatorStatistic< NumberBase >::statParams`

Initial value:

```
{
    m_sum = static_cast<NumberBase>(0)
```

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

- `sst/core/statapi/stataccumulator.h`

6.2 SST::Action Class Reference

```
#include <action.h>
```

Inheritance diagram for SST::Action:

Collaboration diagram for SST::Action:

Public Member Functions

- void [print](#) (const std::string &header, [Output](#) &out) const override

Protected Member Functions

- void [endSimulation](#) ()
- void [endSimulation](#) (SimTime_t end)

Additional Inherited Members

6.2.1 Detailed Description

An [Action](#) is a schedulable [Activity](#) which is not an [Event](#).

6.2.2 Member Function Documentation

6.2.2.1 void SST::Action::endSimulation () [protected]

Called to signal to the [Simulation](#) object to end the simulation

6.2.2.2 void SST::Action::print (const std::string & header, [Output](#) & out) const [inline],[override],[virtual]

Generic print-print function for this [Activity](#). Subclasses should override this function.

Reimplemented from [SST::Activity](#).

Reimplemented in [SST::OneShot](#), [SST::SyncManager](#), [SST::Clock](#), [SST::Exit](#), [SST::ThreadSync](#), and [SST::StopAction](#).

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

- sst/core/action.h
- sst/core/action.cc

6.3 SST::Activity Class Reference

```
#include <activity.h>
```

Inheritance diagram for SST::Activity:

Collaboration diagram for SST::Activity:

Classes

- class [less_time](#)
- class [less_time_priority](#)
- class [less_time_priority_order](#)
- class [pq_less_time_priority](#)
- class [pq_less_time_priority_order](#)

Public Member Functions

- virtual void [execute](#) (void)=0
- void [setDeliveryTime](#) (SimTime_t time)
- SimTime_t [getDeliveryTime](#) () const
- int [getPriority](#) () const
- virtual void [print](#) (const std::string &header, [Output](#) &out) const
- void [setQueueOrder](#) (uint64_t order)

Protected Member Functions

- void [setPriority](#) (int priority)
- void [serialize_order](#) (SST::Core::Serialization::serializer &ser) override

Protected Attributes

- int32_t [enforce_link_order](#)

Additional Inherited Members

6.3.1 Detailed Description

Base class for all Activities in the SST [Event](#) Queue

6.3.2 Member Function Documentation

6.3.2.1 virtual void SST::Activity::execute (void) [pure virtual]

Function which will be called when the time for this [Activity](#) comes to pass.

Implemented in [SST::NullEvent](#), [SST::SyncManager](#), [SST::Exit](#), [SST::Event](#), [SST::StopAction](#), and [SST::ThreadSync](#).

6.3.2.2 `SimTime_t SST::Activity::getDeliveryTime () const` `[inline]`

Return the time at which this [Activity](#) will be delivered

6.3.2.3 `int SST::Activity::getPriority () const` `[inline]`

Return the Priority of this [Activity](#)

6.3.2.4 `virtual void SST::Activity::print (const std::string & header, Output & out) const` `[inline],[virtual]`

Generic print-print function for this [Activity](#). Subclasses should override this function.

Reimplemented in [SST::NullEvent](#), [SST::Event](#), [SST::OneShot](#), [SST::SyncManager](#), [SST::Clock](#), [SST::Exit](#), [SST::ThreadSync](#), [SST::StopAction](#), and [SST::Action](#).

6.3.2.5 `void SST::Activity::setDeliveryTime (SimTime_t time)` `[inline]`

Set the time for which this [Activity](#) should be delivered

6.3.2.6 `void SST::Activity::setPriority (int priority)` `[inline],[protected]`

Set the priority of the [Activity](#)

6.3.2.7 `void SST::Activity::setQueueOrder (uint64_t order)` `[inline]`

Set a new Queue order

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

- `sst/core/activity.h`

6.4 SST::ActivityQueue Class Reference

```
#include <activityQueue.h>
```

Inheritance diagram for SST::ActivityQueue:

Public Member Functions

- virtual bool `empty` ()=0
- virtual int `size` ()=0
- virtual [Activity](#) * `pop` ()=0
- virtual void `insert` ([Activity](#) *activity)=0
- virtual [Activity](#) * `front` ()=0

6.4.1 Detailed Description

Base Class for a queue of Activities

6.4.2 Member Function Documentation

6.4.2.1 `virtual bool SST::ActivityQueue::empty () [pure virtual]`

Returns true if the queue is empty

Implemented in [SST::IMPL::TimeVortexPQ](#), [SST::SyncQueue](#), [SST::TimeVortex](#), [SST::UninitializedQueue](#), [SST::ThreadSyncQueue](#), [SST::InitQueue](#), and [SST::PollingLinkQueue](#).

6.4.2.2 `virtual Activity* SST::ActivityQueue::front () [pure virtual]`

Returns the next activity

Implemented in [SST::IMPL::TimeVortexPQ](#), [SST::ThreadSyncQueue](#), [SST::SyncQueue](#), [SST::TimeVortex](#), [SST::UninitializedQueue](#), [SST::InitQueue](#), and [SST::PollingLinkQueue](#).

6.4.2.3 `virtual void SST::ActivityQueue::insert (Activity * activity) [pure virtual]`

Insert a new activity into the queue

Implemented in [SST::IMPL::TimeVortexPQ](#), [SST::ThreadSyncQueue](#), [SST::SyncQueue](#), [SST::TimeVortex](#), [SST::UninitializedQueue](#), [SST::InitQueue](#), and [SST::PollingLinkQueue](#).

6.4.2.4 `virtual Activity* SST::ActivityQueue::pop () [pure virtual]`

Remove and return the next activity

Implemented in [SST::IMPL::TimeVortexPQ](#), [SST::SyncQueue](#), [SST::ThreadSyncQueue](#), [SST::TimeVortex](#), [SST::UninitializedQueue](#), [SST::InitQueue](#), and [SST::PollingLinkQueue](#).

6.4.2.5 `virtual int SST::ActivityQueue::size () [pure virtual]`

Returns the number of activities in the queue

Implemented in [SST::IMPL::TimeVortexPQ](#), [SST::SyncQueue](#), [SST::TimeVortex](#), [SST::ThreadSyncQueue](#), [SST::UninitializedQueue](#), [SST::InitQueue](#), and [SST::PollingLinkQueue](#).

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

- [sst/core/activityQueue.h](#)

6.5 SST::ELI::Allocator< Base, T, Enable > Struct Template Reference

Public Member Functions

- `template<class... Args>`
`T * operator() (Args &&...args)`

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

- `sst/core/eli/elementbuilder.h`

6.6 SST::ELI::Allocator< SSTELEMENTPythonModule, T > Struct Template Reference

Inheritance diagram for SST::ELI::Allocator< SSTELEMENTPythonModule, T >:

Collaboration diagram for SST::ELI::Allocator< SSTELEMENTPythonModule, T >:

Additional Inherited Members

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

- `sst/core/model/element_python.h`

6.7 SST::Core::ThreadSafe::Barrier Class Reference

Public Member Functions

- **Barrier** (size_t count)
- void **resize** (size_t newCount)
- double **wait** ()
- void **disable** ()

6.7.1 Member Function Documentation

6.7.1.1 void SST::Core::ThreadSafe::Barrier::resize (size_t *newCount*) `[inline]`

ONLY call this while nobody is in **wait()**

6.7.1.2 double SST::Core::ThreadSafe::Barrier::wait () [inline]

Wait for all threads to reach this point.

Returns

0.0, or elapsed time spent waiting, if configured with `--enable-profile`

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

- sst/core/threadsafe.h

6.8 SST::BaseComponent Class Reference

```
#include <baseComponent.h>
```

Inheritance diagram for SST::BaseComponent:

Collaboration diagram for SST::BaseComponent:

Public Member Functions

- **BaseComponent** (ComponentId_t id)
- **BaseComponent** * **getParent** () const `__attribute__((deprecated("getParent() will be removed in SST version 10.0. With the new subcomponent structure`
- const std::string & **getType** () const
- ComponentId_t **getId** () const
- virtual void **emergencyShutdown** (void)
- const std::string & **getName** () const
- virtual void **init** (unsigned int UNUSED(phase))
- virtual void **complete** (unsigned int UNUSED(phase))
- virtual void **setup** ()
- virtual void **finish** ()
- virtual bool **Status** ()
- virtual void **printStatus** (Output &UNUSED(out))
- bool **isPortConnected** (const std::string &name) const
- Link * **configureLink** (std::string name, TimeConverter *time_base, Event::HandlerBase *handler=NULL)
- Link * **configureLink** (std::string name, std::string time_base, Event::HandlerBase *handler=NULL)
- Link * **configureLink** (std::string name, Event::HandlerBase *handler=NULL)
- Link * **configureSelfLink** (std::string name, TimeConverter *time_base, Event::HandlerBase *handler=NULL)
- Link * **configureSelfLink** (std::string name, std::string time_base, Event::HandlerBase *handler=NULL)
- Link * **configureSelfLink** (std::string name, Event::HandlerBase *handler=NULL)
- TimeConverter * **registerClock** (std::string freq, Clock::HandlerBase *handler, bool regAll=true)
- TimeConverter * **registerClock** (const UnitAlgebra &freq, Clock::HandlerBase *handler, bool regAll=true)
- void **unregisterClock** (TimeConverter *tc, Clock::HandlerBase *handler)
- Cycle_t **reregisterClock** (TimeConverter *freq, Clock::HandlerBase *handler)
- Cycle_t **getNextClockCycle** (TimeConverter *freq)
- TimeConverter * **registerOneShot** (std::string timeDelay, OneShot::HandlerBase *handler)
- TimeConverter * **registerOneShot** (const UnitAlgebra &timeDelay, OneShot::HandlerBase *handler)
- TimeConverter * **registerTimeBase** (std::string base, bool regAll=true)
- TimeConverter * **getTimeConverter** (const std::string &base)

- [TimeConverter](#) * **getTimeConverter** (const [UnitAlgebra](#) &base)
- [SimTime_t](#) **getCurrentSimTime** ([TimeConverter](#) *tc) const
- [SimTime_t](#) **getCurrentSimTime** () const
- [SimTime_t](#) **getCurrentSimTime** (std::string base)
- [SimTime_t](#) **getCurrentSimTimeNano** () const
- [SimTime_t](#) **getCurrentSimTimeMicro** () const
- [SimTime_t](#) **getCurrentSimTimeMilli** () const
- bool **isStatisticShared** (const std::string &statName, bool include_me=false)
- template<typename T >
[Statistic](#)< T > * **registerStatistic** ([SST::Params](#) ¶ms, const std::string &statName, const std::string &statSubId="")
- template<typename T >
[Statistic](#)< T > * **registerStatistic** (const std::string &statName, const std::string &statSubId="")
- template<typename... Args>
[Statistic](#)< std::tuple< Args... > > * **registerMultiStatistic** (const std::string &statName, const std::string &statSubId="")
- template<typename... Args>
[Statistic](#)< std::tuple< Args... > > * **registerMultiStatistic** ([SST::Params](#) ¶ms, const std::string &statName, const std::string &statSubId="")
- template<typename T >
[Statistic](#)< T > * **registerStatistic** (const char *statName, const char *statSubId="")
- [Module](#) * **loadModule** (std::string type, [Params](#) ¶ms)
- [Module](#) * **loadModuleWithComponent** (std::string type, [Component](#) *comp, [Params](#) ¶ms) __attribute__((deprecated("loadModuleWithComponent will be removed in SST version 10.0. If the module needs access to the parent component"))
- [SubComponentSlotInfo](#) * **getSubComponentSlotInfo** (std::string name, bool fatalOnEmptyIndex=false)
- const std::vector< double > & **getCoordinates** () const
- bool **wasLoadedWithLegacyAPI** () const

Public Attributes

- [BaseComponent](#) direct access to your parent is not **allowed** { return my_info->parent_info->component
- [Module](#) please use SubComponents instead of **Modules**
- [SubComponent](#) *loadSubComponent(std::string type, [Component](#) *comp, [Params](#) ¶ms) __attribute__((deprecated("This version of loadSubComponent will be removed in SST version 10.0. Please switch to new user defined API (LoadUserSubComponent(std retur [loadUserSubComponentByIndex](#)< T, ARGS... >)(slot_name, index, share_flags, args...)"))

Protected Member Functions

- bool **isAnonymous** ()
- bool **isUser** ()
- void **setDefaultTimeBase** ([TimeConverter](#) *tc)
- [TimeConverter](#) * **getDefaultTimeBase** ()
- bool **doesSubComponentExist** (std::string type)
- [SharedRegion](#) * **getLocalSharedRegion** (const std::string &key, size_t size)
- [SharedRegion](#) * **getGlobalSharedRegion** (const std::string &key, size_t size, [SharedRegionMerger](#) *merger=NULL)
- [Simulation](#) * **getSimulation** () const
- virtual bool **doesComponentInfoStatisticExist** (const std::string &statisticName) const
- uint8_t **getComponentInfoStatisticEnableLevel** (const std::string &statisticName) const
- [Component](#) * **getTrueComponent** () const __attribute__((deprecated("getTrueParent will be removed in SST version 10.0. With the new subcomponent structure"))

Protected Attributes

- [Component](#) direct access to your parent component is not **allowed**
- [Simulation](#) * **sim**

Friends

- class **SubComponentSlotInfo**
- class **SubComponent**
- class **ComponentInfo**
- class **ComponentExtension**
- class **SST::Statistics::StatisticProcessingEngine**

6.8.1 Detailed Description

Main component object for the simulation.

6.8.2 Member Function Documentation

6.8.2.1 virtual void SST::BaseComponent::complete (unsigned int *UNUSEDphase*) [inline],[virtual]

Used during the init phase. The method will be called each phase of initialization. Initialization ends when no components have sent any data.

Reimplemented in [SST::Interfaces::SimpleNetwork](#).

6.8.2.2 Link * SST::BaseComponent::configureLink (std::string *name*, TimeConverter * *time_base*, Event::HandlerBase * *handler* = NULL)

Configure a [Link](#)

Parameters

<i>name</i>	- Port Name on which the link to configure is attached.
<i>time_base</i>	- Time Base of the link. If NULL is passed in, then it will use the Component defaultTimeBase
<i>handler</i>	- Optional Handler to be called when an Event is received

Returns

A pointer to the configured link, or NULL if an error occurred.

6.8.2.3 Link * SST::BaseComponent::configureLink (std::string *name*, std::string *time_base*, Event::HandlerBase * *handler* = NULL)

Configure a [Link](#)

Parameters

<i>name</i>	- Port Name on which the link to configure is attached.
<i>time_base</i>	- Time Base of the link as a string
<i>handler</i>	- Optional Handler to be called when an Event is received

Returns

A pointer to the configured link, or NULL if an error occurred.

6.8.2.4 `Link * SST::BaseComponent::configureLink (std::string name, Event::HandlerBase * handler = NULL)`

Configure a [Link](#)

Parameters

<i>name</i>	- Port Name on which the link to configure is attached.
<i>handler</i>	- Optional Handler to be called when an Event is received

Returns

A pointer to the configured link, or NULL if an error occurred.

6.8.2.5 `Link * SST::BaseComponent::configureSelfLink (std::string name, TimeConverter * time_base, Event::HandlerBase * handler = NULL)`

Configure a [SelfLink](#) (Loopback link)

Parameters

<i>name</i>	- Name of the self-link port
<i>time_base</i>	- Time Base of the link. If NULL is passed in, then it will use the Component defaultTimeBase
<i>handler</i>	- Optional Handler to be called when an Event is received

Returns

A pointer to the configured link, or NULL if an error occurred.

6.8.2.6 `Link * SST::BaseComponent::configureSelfLink (std::string name, std::string time_base, Event::HandlerBase * handler = NULL)`

Configure a [SelfLink](#) (Loopback link)

Parameters

<i>name</i>	- Name of the self-link port
<i>time_base</i>	- Time Base of the link
<i>handler</i>	- Optional Handler to be called when an Event is received

Returns

A pointer to the configured link, or NULL if an error occurred.

6.8.2.7 `Link * SST::BaseComponent::configureSelfLink (std::string name, Event::HandlerBase * handler = NULL)`

Configure a [SelfLink](#) (Loopback link)

Parameters

<i>name</i>	- Name of the self-link port
<i>handler</i>	- Optional Handler to be called when an Event is received

Returns

A pointer to the configured link, or NULL if an error occurred.

6.8.2.8 `virtual void SST::BaseComponent::emergencyShutdown (void) [inline], [virtual]`

Called when SIGINT or SIGTERM has been seen. Allows components opportunity to clean up external state.

6.8.2.9 `virtual void SST::BaseComponent::finish () [inline], [virtual]`

Called after simulation completes, but before objects are destroyed. A good place to print out statistics.

Reimplemented in [SST::Interfaces::SimpleNetwork](#), and [SST::SubComponent](#).

6.8.2.10 `const std::vector<double>& SST::BaseComponent::getCoordinates () const [inline]`

Retrieve the X,Y,Z coordinates of this component

6.8.2.11 `SimTime_t SST::BaseComponent::getCurrentSimTime (TimeConverter * tc) const`

return the time since the simulation began in units specified by the parameter.

Parameters

<i>tc</i>	TimeConverter specifying the units
-----------	--

6.8.2.12 `SimTime_t SST::BaseComponent::getCurrentSimTime () const` `[inline]`

return the time since the simulation began in the default timebase

6.8.2.13 `SimTime_t SST::BaseComponent::getCurrentSimTime (std::string base)`

return the time since the simulation began in timebase specified

Parameters

<i>base</i>	Timebase frequency in SI Units
-------------	--

6.8.2.14 `SimTime_t SST::BaseComponent::getCurrentSimTimeMicro () const`

Utility function to return the time since the simulation began in microseconds

6.8.2.15 `SimTime_t SST::BaseComponent::getCurrentSimTimeMilli () const`

Utility function to return the time since the simulation began in milliseconds

6.8.2.16 `SimTime_t SST::BaseComponent::getCurrentSimTimeNano () const`

Utility function to return the time since the simulation began in nanoseconds

6.8.2.17 `ComponentId_t SST::BaseComponent::getId () const` `[inline]`

Returns unique component ID

6.8.2.18 `SharedRegion * SST::BaseComponent::getLocalSharedRegion (const std::string & key, size_t size)`
`[protected]`

Find a lookup table

6.8.2.19 `const std::string& SST::BaseComponent::getName () const` `[inline]`

Returns component Name

6.8.2.20 `Cycle_t SST::BaseComponent::getNextClockCycle (TimeConverter * freq)`

Returns the next Cycle that the [TimeConverter](#) would fire

6.8.2.21 `BaseComponent* SST::BaseComponent::getParent () const` [new]

Returns a pointer to the parent [BaseComponent](#)

6.8.2.22 `virtual void SST::BaseComponent::init (unsigned int UNUSEDphase)` [inline],[virtual]

Used during the init phase. The method will be called each phase of initialization. Initialization ends when no components have sent any data.

Reimplemented in [SST::Interfaces::SimpleNetwork](#), and [SST::SubComponent](#).

6.8.2.23 `bool SST::BaseComponent::isPortConnected (const std::string & name) const`

Determine if a port name is connected to any links

6.8.2.24 `Module * SST::BaseComponent::loadModule (std::string type, Params & params)`

Loads a module from an element Library

Parameters

<i>type</i>	Fully Qualified library.moduleName
<i>params</i>	Parameters the module should use for configuration

Returns

handle to new instance of module, or NULL on failure.

6.8.2.25 `Module * SST::BaseComponent::loadModuleWithComponent (std::string type, Component * comp, Params & params)`

Loads a module from an element Library

Parameters

<i>type</i>	Fully Qualified library.moduleName
<i>comp</i>	Pointer to component to pass to Module's constructor
<i>params</i>	Parameters the module should use for configuration

Returns

handle to new instance of module, or NULL on failure.

6.8.2.26 virtual void SST::BaseComponent::printStats (Output & *UNUSEDout*) [inline],[virtual]

Called by the [Simulation](#) to request that the component print it's current status. Useful for debugging.

Parameters

<i>out</i>	The Output class which should be used to print component status.
------------	--

6.8.2.27 `TimeConverter * SST::BaseComponent::registerClock (std::string freq, Clock::HandlerBase * handler, bool regAll = true)`

Registers a clock for this component.

Parameters

<i>freq</i>	Frequency for the clock in SI units
<i>handler</i>	Pointer to Clock::HandlerBase which is to be invoked at the specified interval
<i>regAll</i>	Should this clock period be used as the default time base for all of the links connected to this component

6.8.2.28 `TimeConverter * SST::BaseComponent::registerOneShot (std::string timeDelay, OneShot::HandlerBase * handler)`

Registers a [OneShot](#) event for this component. Note: [OneShot](#) cannot be canceled, and will always callback after the timedelay.

Parameters

<i>timeDelay</i>	Time delay for the OneShot in SI units
<i>handler</i>	Pointer to OneShot::HandlerBase which is to be invoked at the specified interval

6.8.2.29 `template<typename T> Statistic<T>* SST::BaseComponent::registerStatistic (SST::Params & params, const std::string & statName, const std::string & statSubId = " ") [inline]`

Registers a statistic. If Statistic is allowed to run (controlled by Python runtime parameters), then a statistic will be created and returned. If not allowed to run, then a NullStatistic will be returned. In either case, the returned value should be used for all future Statistic calls. The type of Statistic and the Collection Rate is set by Python runtime parameters. If no type is defined, then an Accumulator Statistic will be provided by default. If rate set to 0 or not provided, then the statistic will output results only at end of sim (if output is enabled).

Parameters

<i>statName</i>	Primary name of the statistic. This name must match the defined ElementInfoStatistic in the component, and must also be enabled in the Python input file.
<i>statSubId</i>	An additional sub name for the statistic

Returns

Either a created statistic of desired type or a NullStatistic depending upon runtime settings.

6.8.2.30 TimeConverter * SST::BaseComponent::registerTimeBase (std::string *base*, bool *regAll* = true)

Registers a default time base for the component and optionally sets the the component's links to that timebase. Useful for components which do not have a clock, but would like a default timebase.

Parameters

<i>base</i>	Frequency for the clock in SI units
<i>regAll</i>	Should this clock period be used as the default time base for all of the links connected to this component

6.8.2.31 Cycle_t SST::BaseComponent::reregisterClock (TimeConverter * *freq*, Clock::HandlerBase * *handler*)

Reactivates an existing [Clock](#) and Handler

Returns

time of next time clock handler will fire

6.8.2.32 void SST::BaseComponent::setDefaultTimeBase (TimeConverter * *tc*) [inline], [protected]

Manually set the default defaultTimeBase

6.8.2.33 virtual void SST::BaseComponent::setup () [inline], [virtual]

Called after all components have been constructed and initialization has completed, but before simulation time has begun.

Reimplemented in [SST::Interfaces::SimpleNetwork](#), and [SST::SubComponent](#).

6.8.2.34 virtual bool SST::BaseComponent::Status () [inline], [virtual]

Currently unused function

6.8.2.35 void SST::BaseComponent::unregisterClock (TimeConverter * *tc*, Clock::HandlerBase * *handler*)

Removes a clock handler from the component

6.8.2.36 bool SST::BaseComponent::wasLoadedWithLegacyAPI () const [inline]

Temporary function to help provide backward compatibility to old [SubComponent](#) API.

Returns

true if subcomponent loaded with old API, false if loaded with new

6.8.3 Member Data Documentation**6.8.3.1 SubComponent* loadSubComponent (std::string type, Component* comp, Params& params) __attribute__((deprecated("This version of loadSubComponent will be removed in SST version 10.0. Please switch to new user defined API (LoadUserSubComponent(std retur SST::BaseComponent::loadUserSubComponentByIndex< T, ARGS...>)(slot_name, index, share_flags, args...) [new])**

Loads a [SubComponent](#) from an element Library

Parameters

<i>type</i>	Fully Qualified library.moduleName
<i>comp</i>	Pointer to component to pass to SuBaseComponent's constructor
<i>params</i>	Parameters the module should use for configuration

Returns

handle to new instance of [SubComponent](#), or NULL on failure.

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

- sst/core/baseComponent.h
- sst/core/baseComponent.cc

6.9 SST::Core::ThreadSafe::BoundedQueue< T > Class Template Reference

Public Member Functions

- **BoundedQueue** (size_t maxSize)
- void **initialize** (size_t maxSize)
- size_t **size** () const
- bool **empty** () const
- bool **try_insert** (const T &arg)
- bool **try_remove** (T &res)
- T **remove** ()

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

- sst/core/threadsafe.h

6.10 SST::ELI::Builder< Base, Args > Struct Template Reference

Public Types

- typedef Base *(* **createFxn**) (Args...)
- template<class NewBase >
using **ChangeBase** = [Builder](#)< NewBase, Args... >

Public Member Functions

- virtual Base * **create** (Args...ctorArgs)=0

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

- sst/core/eli/elementbuilder.h

6.11 SST::ELI::BuilderDatabase Struct Reference

Static Public Member Functions

- `template<class T, class... Args>`
`static BuilderLibrary< T, Args... > * getLibrary (const std::string &name)`

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

- `sst/core/eli/elementbuilder.h`

6.12 SST::ELI::BuilderInfoImpl< Policy, Policies > Class Template Reference

Inheritance diagram for SST::ELI::BuilderInfoImpl< Policy, Policies >:

Collaboration diagram for SST::ELI::BuilderInfoImpl< Policy, Policies >:

Public Member Functions

- `template<class... Args>`
`BuilderInfoImpl (const std::string &elemlib, const std::string &elem, Args &&...args)`
- `template<class XMLNode >`
`void outputXML (XMLNode *node)`
- `void toString (std::ostream &os) const`

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

- `sst/core/eli/elementinfo.h`

6.13 SST::ELI::BuilderInfoImpl< void > Class Template Reference

Protected Member Functions

- `template<class... Args>`
`BuilderInfoImpl (Args &&...UNUSED(args))`
- `template<class XMLNode >`
`void outputXML (XMLNode *UNUSED(node))`
- `void toString (std::ostream &UNUSED(os)) const`

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

- `sst/core/eli/elementinfo.h`

6.14 SST::ELI::BuilderLibrary< Base, CtorArgs > Class Template Reference

Public Types

- using **BaseBuilder** = [Builder](#)< Base, CtorArgs... >
- template<class NewBase >
using **ChangeBase** = [BuilderLibrary](#)< NewBase, CtorArgs... >

Public Member Functions

- **BuilderLibrary** (const std::string &name)
- [BaseBuilder](#) * **getBuilder** (const std::string &name)
- const std::map< std::string, [BaseBuilder](#) * > & **getMap** () const
- void **readBuilder** (const std::string &name, [BaseBuilder](#) *fact)
- bool **addBuilder** (const std::string &elem, [BaseBuilder](#) *fact)

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

- sst/core/eli/elementbuilder.h

6.15 SST::ELI::BuilderLibraryDatabase< Base, CtorArgs > Class Template Reference

Public Types

- using **Library** = [BuilderLibrary](#)< Base, CtorArgs... >
- using **BaseFactory** = typename [Library::BaseBuilder](#)
- using **Map** = std::map< std::string, [Library](#) * >
- template<class NewBase >
using **ChangeBase** = [BuilderLibraryDatabase](#)< NewBase, CtorArgs... >

Static Public Member Functions

- static [Library](#) * **getLibrary** (const std::string &name)

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

- sst/core/eli/elementbuilder.h

6.16 SST::RegionInfo::BulkMergeInfo Class Reference

Inheritance diagram for SST::RegionInfo::BulkMergeInfo:

Collaboration diagram for SST::RegionInfo::BulkMergeInfo:

Public Member Functions

- **BulkMergeInfo** (int rank, const std::string &key, void *data, size_t length)
- bool **merge** ([RegionInfo](#) *ri) override
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Protected Attributes

- size_t **length**
- void * **data**

Additional Inherited Members

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

- sst/core/sharedRegionImpl.h

6.17 SST::ELI::CachedAllocator< Base, T > Struct Template Reference

Collaboration diagram for SST::ELI::CachedAllocator< Base, T >:

Public Member Functions

- template<class... Args>
Base * **operator()** (Args &&...ctorArgs)

Static Public Attributes

- static Base * **cached_** = nullptr

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

- sst/core/eli/elementbuilder.h

6.18 SST::ChangeSet Class Reference

Inheritance diagram for SST::ChangeSet:

Collaboration diagram for SST::ChangeSet:

Public Member Functions

- **ChangeSet** (size_t offset, size_t length, uint8_t *data=NULL)
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Public Attributes

- `size_t` **offset**
- `size_t` **length**
- `uint8_t *` **data**

Additional Inherited Members

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

- `sst/core/sharedRegionImpl.h`

6.19 SST::RegionInfo::ChangeSetMergeInfo Class Reference

Inheritance diagram for `SST::RegionInfo::ChangeSetMergeInfo`:

Collaboration diagram for `SST::RegionInfo::ChangeSetMergeInfo`:

Public Member Functions

- **ChangeSetMergeInfo** (int rank, const std::string &key, std::vector< [ChangeSet](#) > &changeSets)
- bool **merge** ([RegionInfo](#) *ri) override
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Protected Attributes

- std::vector< [ChangeSet](#) > **changeSets**

Additional Inherited Members

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

- `sst/core/sharedRegionImpl.h`

6.20 SST::char_delimiter Struct Reference

Public Types

- typedef std::string::const_iterator **iter**

Public Member Functions

- **char_delimiter** (const std::string &delim=" \t\v\f\n\r")
- void **operator()** (iter &first, iter last, std::string &token)

Public Attributes

- const std::string **delim**

6.20.1 Member Function Documentation

6.20.1.1 void SST::char_delimiter::operator()(iter & *first*, iter *last*, std::string & *token*) [inline]

Returns

pair<iter, iter> = <tok_end, next_tok>

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

- sst/core/stringize.h

6.21 SST::Core::Interprocess::CircularBuffer< T > Class Template Reference

Public Member Functions

- **CircularBuffer** (size_t mSize=0)
- void **setBufferSize** (const size_t bufferSize)
- T **read** ()
- bool **readNB** (T *result)
- void **write** (const T &v)
- void **clearBuffer** ()

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

- sst/core/interprocess/circularBuffer.h

6.22 SST::Clock Class Reference

```
#include <clock.h>
```

Inheritance diagram for SST::Clock:

Collaboration diagram for SST::Clock:

Classes

- class [Handler](#)
- class [Handler< classT, void >](#)
- class [HandlerBase](#)

Public Member Functions

- [Clock](#) ([TimeConverter](#) *period, int priority=CLOCKPRIORITY)
- void [schedule](#) ()
- Cycle_t [getNextCycle](#) ()
- bool [registerHandler](#) ([Clock::HandlerBase](#) *handler)
- bool [unregisterHandler](#) ([Clock::HandlerBase](#) *handler, bool &empty)
- void [print](#) (const std::string &header, [Output](#) &out) const override

Additional Inherited Members

6.22.1 Detailed Description

A [Clock](#) class.

Calls callback functions (handlers) on a specified period

6.22.2 Constructor & Destructor Documentation

6.22.2.1 SST::Clock::Clock ([TimeConverter](#) * *period*, int *priority* = `CLOCKPRIORITY`)

Create a new clock with a specified period

6.22.3 Member Function Documentation

6.22.3.1 Cycle_t SST::Clock::getNextCycle ()

Return the time of the next clock tick

6.22.3.2 void SST::Clock::print (const std::string & *header*, [Output](#) & *out*) const [override],[virtual]

Generic print-print function for this [Activity](#). Subclasses should override this function.

Reimplemented from [SST::Action](#).

6.22.3.3 bool SST::Clock::registerHandler ([Clock::HandlerBase](#) * *handler*)

Add a handler to be called on this clock's tick

6.22.3.4 void SST::Clock::schedule ()

Activates this clock object, by inserting into the simulation's timeVortex for future execution.

6.22.3.5 bool SST::Clock::unregisterHandler (Clock::HandlerBase * handler, bool & empty)

Remove a handler from the list of handlers to be called on the clock tick

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

- sst/core/clock.h
- sst/core/clock.cc

6.23 SST::Component Class Reference

```
#include <component.h>
```

Inheritance diagram for SST::Component:

Collaboration diagram for SST::Component:

Public Member Functions

- [SST_ELI_DECLARE_INFO_EXTERN](#) (ELI::ProvidesParams, ELI::ProvidesSubComponentSlots, ELI::ProvidesPorts, ELI::ProvidesStats, ELI::ProvidesCategory) [Component](#)(ComponentId_t id)
- bool [registerExit](#) () [__attribute__\(\(deprecated\("registerExit is deprecated and will be removed in SST version 10.0. Please use \[registerAsPrimaryComponent\]\(#\)\(\) and \[primaryComponentDoNotEndSim\]\(#\)\(\) instead."\)\)\)](#)
- bool [unregisterExit](#) () [__attribute__\(\(deprecated\("unregisterExit is deprecated and will be removed in SST version 10.0. Please use \[primaryComponentOKToEndSim\]\(#\)\(\) instead."\)\)\)](#)
- void [registerAsPrimaryComponent](#) ()
- void [primaryComponentDoNotEndSim](#) ()
- void [primaryComponentOKToEndSim](#) ()

Friends

- class [SubComponent](#)

Additional Inherited Members

6.23.1 Detailed Description

Main component object for the simulation. All models inherit from this.

6.23.2 Member Function Documentation

6.23.2.1 void SST::Component::primaryComponentDoNotEndSim ()

Tells the simulation that it should not exit. The component will remain in this state until a call to [primaryComponentOKToEndSim](#)().

```
@sa Component::registerAsPrimaryComponent ()
@sa Component::primaryComponentOKToEndSim ()
```

6.23.2.2 void SST::Component::primaryComponentOKToEndSim ()

Tells the simulation that it is now OK to end simulation. [Simulation](#) will not end until all primary components have called this function.

```
@sa Component::registerAsPrimaryComponent()
@sa Component::primaryComponentDoNotEndSim()
```

6.23.2.3 void SST::Component::registerAsPrimaryComponent ()

Register as a primary component, which allows the component to specify when it is and is not OK to end simulation. The simulator will not end simulation naturally (through use of the [Exit](#) object) while any primary component has specified [primaryComponentDoNotEndSim\(\)](#). However, it is still possible for Actions other than [Exit](#) to end simulation. Once all primary components have specified [primaryComponentOKToEndSim\(\)](#), the [Exit](#) object will trigger and end simulation.

This must be called during simulation wireup (i.e during the constructor for the component). By default, the state of the primary component is set to OKToEndSim.

If no component registers as a primary component, then the [Exit](#) object will not be used for that simulation and simulation termination must be accomplished through some other mechanism (e.g. `-stopAt` flag, or some other [Action](#) object).

```
@sa Component::primaryComponentDoNotEndSim()
@sa Component::primaryComponentOKToEndSim()
```

6.23.2.4 bool SST::Component::registerExit ()

Register that the simulation should not end until this component says it is OK to. Calling this function (generally done in [Component::setup\(\)](#) or in component constructor) increments a global counter. Calls to [Component::unregisterExit\(\)](#) decrements the counter. The simulation cannot end unless this counter reaches zero, or the simulation time limit is reached. This counter is synchronized periodically with the other nodes.

See also

[Component::unregisterExit\(\)](#)

6.23.2.5 SST::Component::SST_ELI_DECLARE_INFO_EXTERN (ELI::ProvidesParams , ELI::Provides↵ SubComponentSlots , ELI::ProvidesPorts , ELI::ProvidesStats , ELI::ProvidesCategory)

Constructor. Generally only called by the factory class.

Parameters

<i>id</i>	Unique component ID
-----------	---------------------

6.23.2.6 bool SST::Component::unregisterExit ()

Indicate permission for the simulation to end. This function is the mirror of [Component::registerExit\(\)](#). It decrements the global counter, which, upon reaching zero, indicates that the simulation can terminate.

See also

[Component::registerExit\(\)](#)

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

- sst/core/component.h
- sst/core/component.cc

6.24 SST::ComponentExtension Class Reference

```
#include <componentExtension.h>
```

Inheritance diagram for SST::ComponentExtension:

Collaboration diagram for SST::ComponentExtension:

Public Member Functions

- **ComponentExtension** (ComponentId_t id)

Additional Inherited Members

6.24.1 Detailed Description

[ComponentExtension](#) is a class that can be loaded using `loadComponentExtension<T>(...)`. All the calls to the [BaseComponent](#) APIU will act like they are happening in the nearest SubComponent or [Component](#) parent. Hierarchy will not be kept in the case were a [ComponentExtension](#) is loaded into a [ComponentExtension](#); they will both act like they are in the parent.

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

- sst/core/componentExtension.h
- sst/core/componentExtension.cc

6.25 ComponentHolder Struct Reference

Inheritance diagram for ComponentHolder:

6.26 SST::ComponentInfo Class Reference

Classes

- struct [EqualsID](#)
- struct [EqualsName](#)
- struct [HashID](#)
- struct [HashName](#)

Public Types

- typedef std::vector< [Statistics::StatisticInfo](#) > [statEnableList_t](#)

Public Member Functions

- **ComponentInfo** (const std::string &type, const [Params](#) *params, const [ComponentInfo](#) *parent_info)
- **ComponentInfo** (ComponentId_t id, [ComponentInfo](#) *parent_info, const std::string &type, const std::string &slot_name, int slot_num, uint64_t share_flags)
- **ComponentInfo** ([ConfigComponent](#) *ccomp, const std::string &name, [ComponentInfo](#) *parent_info, [LinkMap](#) *link_map)
- **ComponentInfo** ([ComponentInfo](#) &&o)
- bool **isAnonymous** ()
- bool **isUser** ()
- ComponentId_t **getID** () const
- const std::string & **getName** () const
- const std::string & **getSlotName** () const
- int **getSlotNum** () const
- const std::string & **getType** () const
- [BaseComponent](#) * **getComponent** () const
- [LinkMap](#) * **getLinkMap** ()
- const [Params](#) * **getParams** () const
- std::map< ComponentId_t, [ComponentInfo](#) > & **getSubComponents** ()
- [ComponentInfo](#) * **findSubComponent** (std::string slot, int slot_num)
- [ComponentInfo](#) * **findSubComponent** (ComponentId_t id)
- std::vector< LinkId_t > **getAllLinkIds** () const
- [statEnableList_t](#) * **getStatEnableList** ()

Static Public Attributes

- static const uint64_t **SHARE_PORTS** = 0x1
- static const uint64_t **SHARE_STATS** = 0x2
- static const uint64_t **INSERT_STATS** = 0x4
- static const uint64_t **IS_LEGACY_SUBCOMPONENT** = 0x32
- static const uint64_t **SHARE_NONE** = 0x0

Friends

- class **Simulation**
- class **BaseComponent**
- class **ComponentInfoMap**

6.26.1 Member Typedef Documentation

6.26.1.1 typedef std::vector<Statistics::StatisticInfo> SST::ComponentInfo::statEnableList_t

List of Enabled Statistics

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

- sst/core/componentInfo.h
- sst/core/componentInfo.cc

6.27 SST::ComponentInfoMap Class Reference

Public Types

- typedef std::unordered_set< [ComponentInfo](#) *, [ComponentInfo::HashID](#), [ComponentInfo::EqualsID](#) >↔
::const_iterator **const_iterator**

Public Member Functions

- const_iterator **begin** () const
- const_iterator **end** () const
- void **insert** ([ComponentInfo](#) *info)
- [ComponentInfo](#) * **getByID** (const ComponentId_t key) const
- bool **empty** ()
- void **clear** ()

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

- sst/core/componentInfo.h

6.28 ComponentPy_t Struct Reference

Collaboration diagram for ComponentPy_t:

Public Attributes

- PyObject_HEAD [ComponentHolder](#) * **obj**

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

- sst/core/model/pymodel_comp.h

6.29 SST::Config Class Reference

```
#include <config.h>
```

Inheritance diagram for SST::Config:

Collaboration diagram for SST::Config:

Public Types

- typedef bool(Config::* **flagFunction**) (void)
- typedef bool(Config::* **argFunction**) (const std::string &arg)

Public Member Functions

- [Config](#) ([RankInfo](#) world_size)
- int [parseCmdLine](#) (int argc, char *argv[])
- bool [setConfigEntryFromModel](#) (const std::string &entryName, const std::string &value)
- uint32_t [getVerboseLevel](#) ()
- bool [printTimingInfo](#) ()
- void [setStopAt](#) (std::string stopAtStr)
- void [setTimeBase](#) (std::string timeBase)
- void [Print](#) ()
- bool [usage](#) ()
- bool [printVersion](#) ()
- bool [incrVerbose](#) ()
- bool [setVerbosity](#) (const std::string &arg)
- bool [disableSigHandlers](#) ()
- bool [disableEnvConfig](#) ()
- bool [enablePrintTiming](#) ()
- bool [enablePrintEnv](#) ()
- bool [setConfigFile](#) (const std::string &arg)
- bool [setDebugFile](#) (const std::string &arg)
- bool [setLibPath](#) (const std::string &arg)
- bool [addLibPath](#) (const std::string &arg)
- bool [setRunMode](#) (const std::string &arg)
- bool [setStopAt](#) (const std::string &arg)
- bool [setStopAfter](#) (const std::string &arg)
- bool [setHeartbeat](#) (const std::string &arg)
- bool [setTimebase](#) (const std::string &arg)
- bool [setPartitioner](#) (const std::string &arg)
- bool [setTimeVortex](#) (const std::string &arg)
- bool [setOutputDir](#) (const std::string &arg)
- bool [setWriteConfig](#) (const std::string &arg)
- bool [setWriteDot](#) (const std::string &arg)
- bool [setWriteXML](#) (const std::string &arg)
- bool [setWriteJSON](#) (const std::string &arg)
- bool [setWritePartition](#) (const std::string &arg)
- bool [setOutputPrefix](#) (const std::string &arg)
- bool [setModelOptions](#) (const std::string &arg)
- bool [setNumThreads](#) (const std::string &arg)
- [Simulation::Mode_t](#) [getRunMode](#) ()
- void [print](#) ()
- std::string [getLibPath](#) (void) const
- uint32_t [getNumRanks](#) ()
- uint32_t [getNumThreads](#) ()
- uint32_t [setNumThreads](#) (uint32_t nthr)
- void [serialize_order](#) ([SST::Core::Serialization::serializer](#) &ser) override

Public Attributes

- `std::string debugFile`
- `Simulation::Mode_t runMode`
- `std::string configFile`
- `std::string stopAtCycle`
- `uint32_t stopAfterSec`
- `std::string heartbeatPeriod`
- `std::string timeBase`
- `std::string partitioner`
- `std::string timeVortex`
- `std::string output_config_graph`
- `std::string output_dot`
- `std::string output_xml`
- `std::string output_json`
- `std::string output_directory`
- `std::string model_options`
- `std::string dump_component_graph_file`
- `std::string output_core_prefix`
- `RankInfo world_size`
- `uint32_t verbose`
- `bool no_env_config`
- `bool enable_sig_handling`
- `bool print_timing`
- `bool print_env`

Additional Inherited Members

6.29.1 Detailed Description

Class to contain SST [Simulation](#) Configuration variables

6.29.2 Constructor & Destructor Documentation

6.29.2.1 SST::Config::Config (RankInfo world_size)

Create a new [Config](#) object.

Parameters

<i>my_rank</i>	- parallel rank of this instance
<i>world_size</i>	- number of parallel ranks in the simulation

6.29.3 Member Function Documentation

6.29.3.1 std::string SST::Config::getLibPath (void) const

Return the library search path

6.29.3.2 `uint32_t SST::Config::getVerboseLevel ()`

Return the current Verbosity level

6.29.3.3 `int SST::Config::parseCmdLine (int argc, char * argv[])`

Parse command-line arguments to update configuration values

6.29.3.4 `void SST::Config::Print ()`

Print the current configuration to stdout

6.29.3.5 `void SST::Config::print ()` `[inline]`

Print to stdout the current configuration

6.29.3.6 `bool SST::Config::printTimingInfo ()`

Print the SST core timing information

6.29.3.7 `bool SST::Config::setConfigEntryFromModel (const std::string & entryName, const std::string & value)`

Set a configuration string to update configuration values

6.29.3.8 `void SST::Config::setStopAt (std::string stopAtStr)`

Set the cycle at which to stop the simulation

6.29.3.9 `void SST::Config::setTimeBase (std::string timeBase)`

Sets the default timebase of the simulation

6.29.4 **Member Data Documentation****6.29.4.1** `std::string SST::Config::configFile`

Graph generation file

6.29.4.2 `std::string SST::Config::debugFile`

File to which debug information should be written

6.29.4.3 `std::string SST::Config::dump_component_graph_file`

File to dump component graph

6.29.4.4 `bool SST::Config::enable_sig_handling`

Enable signal handling

6.29.4.5 `std::string SST::Config::heartbeatPeriod`

Sets the heartbeat period for the simulation

6.29.4.6 `std::string SST::Config::model_options`

Options to pass to Python Model generator

6.29.4.7 `bool SST::Config::no_env_config`

Bypass compile-time environmental configuration

6.29.4.8 `std::string SST::Config::output_config_graph`

File to dump configuration graph

6.29.4.9 `std::string SST::Config::output_core_prefix`

Set the [SST::Output](#) prefix for the core

6.29.4.10 `std::string SST::Config::output_directory`

[Output](#) directory to dump all files to

6.29.4.11 `std::string SST::Config::output_dot`

File to dump dot output

6.29.4.12 `std::string SST::Config::output_json`

File to dump JSON output

6.29.4.13 `std::string SST::Config::output_xml`

File to dump XML output

6.29.4.14 `std::string SST::Config::partitioner`

Partitioner to use

6.29.4.15 `bool SST::Config::print_env`

Print SST environment

6.29.4.16 `bool SST::Config::print_timing`

Print SST timing information

6.29.4.17 `Simulation::Mode_t SST::Config::runMode`

Run Mode (Init, Both, Run-only)

6.29.4.18 `uint32_t SST::Config::stopAfterSec`

When (wall-time) to stop the simulation

6.29.4.19 `std::string SST::Config::stopAtCycle`

When to stop the simulation

6.29.4.20 `std::string SST::Config::timeBase`

Timebase of simulation

6.29.4.21 `std::string SST::Config::timeVortex`

[TimeVortex](#) implementation to use

6.29.4.22 `uint32_t SST::Config::verbose`

Verbosity

6.29.4.23 RankInfo SST::Config::world_size

Number of ranks, threads which should be invoked per rank

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

- sst/core/config.h
- sst/core/config.cc

6.30 SST::ConfigComponent Class Reference

```
#include <configGraph.h>
```

Inheritance diagram for SST::ConfigComponent:

Collaboration diagram for SST::ConfigComponent:

Public Member Functions

- const ComponentId_t & **key** () const
- void **print** (std::ostream &os) const
- **ConfigComponent** **cloneWithoutLinks** () const
- **ConfigComponent** **cloneWithoutLinksOrParams** () const
- void **setRank** (**RankInfo** r)
- void **setWeight** (double w)
- void **setCoordinates** (const std::vector< double > &c)
- void **addParameter** (const std::string &key, const std::string &value, bool overwrite)
- **ConfigComponent** * **addSubComponent** (ComponentId_t, const std::string &name, const std::string &type, int slot)
- **ConfigComponent** * **findSubComponent** (ComponentId_t)
- const **ConfigComponent** * **findSubComponent** (ComponentId_t) const
- void **enableStatistic** (const std::string &statisticName, bool recursively=false)
- void **addStatisticParameter** (const std::string &statisticName, const std::string ¶m, const std::string &value, bool recursively=false)
- void **setStatisticParameters** (const std::string &statisticName, const **Params** ¶ms, bool recursively=false)
- std::vector< LinkId_t > **allLinks** () const
- void **serialize_order** (SST::Core::Serialization::serializer &ser) override
- ImplementSerializable(SST::ConfigComponent) private **ConfigComponent** (ComponentId_t id, std::string name, std::string type, float weight, **RankInfo** rank)
- **ConfigComponent** (ComponentId_t id, std::string name, int slot_num, std::string type, float weight, **RankInfo** rank)

Public Attributes

- ComponentId_t [id](#)
- [ConfigComponent](#) * [ultimate_parent](#)
- std::string [name](#)
- int [slot_num](#)
- std::string [type](#)
- float [weight](#)
- [RankInfo](#) [rank](#)
- std::vector< [LinkId_t](#) > [links](#)
- [Params](#) [params](#)
- std::vector< [Statistics::StatisticInfo](#) > [enabledStatistics](#)
- std::vector< [ConfigComponent](#) > [subComponents](#)
- std::vector< double > **[coords](#)**
- uint16_t [nextSubID](#)

Static Public Attributes

- static constexpr ComponentId_t **[null_id](#)** = std::numeric_limits<ComponentId_t>::max()

Additional Inherited Members

6.30.1 Detailed Description

Represents the configuration of a generic component

6.30.2 Constructor & Destructor Documentation

6.30.2.1 ImplementSerializable (SST::ConfigComponent) private SST::ConfigComponent::ConfigComponent (ComponentId_t *id*, std::string *name*, std::string *type*, float *weight*, [RankInfo](#) *rank*) [\[inline\]](#)

Create a new [Component](#)

6.30.3 Member Function Documentation

6.30.3.1 void SST::ConfigComponent::print (std::ostream & *os*) const

Print [Component](#) information

6.30.4 Member Data Documentation

6.30.4.1 std::vector<[Statistics::StatisticInfo](#)> SST::ConfigComponent::enabledStatistics

List of statistics to be enabled

6.30.4.2 ComponentId_t SST::ConfigComponent::id

Unique ID of this component

6.30.4.3 std::vector<LinkId_t> SST::ConfigComponent::links

List of links connected

6.30.4.4 std::string SST::ConfigComponent::name

Name of this component, or slot name for subcomp

6.30.4.5 uint16_t SST::ConfigComponent::nextSubID

Next subID to use for children

6.30.4.6 Params SST::ConfigComponent::params

Set of Parameters

6.30.4.7 RankInfo SST::ConfigComponent::rank

Parallel Rank for this component

6.30.4.8 int SST::ConfigComponent::slot_num

Slot number. Only valid for subcomponents

6.30.4.9 std::vector<ConfigComponent> SST::ConfigComponent::subComponents

List of subcomponents

6.30.4.10 std::string SST::ConfigComponent::type

Type of this component

6.30.4.11 ConfigComponent* SST::ConfigComponent::ultimate_parent

Ultimate parent of this component

6.30.4.12 float SST::ConfigComponent::weight

Partitioning weight for this component

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

- sst/core/configGraph.h
- sst/core/configGraph.cc

6.31 SST::ConfigGraph Class Reference

```
#include <configGraph.h>
```

Inheritance diagram for SST::ConfigGraph:

Collaboration diagram for SST::ConfigGraph:

Public Member Functions

- void [print](#) (std::ostream &os) const
- size_t [getNumComponents](#) ()
- void [setComponentRanks](#) ([RankInfo](#) rank)
- bool [containsComponentInRank](#) ([RankInfo](#) rank)
- bool [checkRanks](#) ([RankInfo](#) ranks)
- ComponentId_t [addComponent](#) (ComponentId_t id, std::string name, std::string type, float weight, [RankInfo](#) rank)
- ComponentId_t [addComponent](#) (ComponentId_t id, std::string name, std::string type)
- void [setStatisticOutput](#) (const std::string &name)
- void [addStatisticOutputParameter](#) (const std::string ¶m, const std::string &value)
- void [setStatisticOutputParams](#) (const [Params](#) &p)
- void [setStatisticLoadLevel](#) (uint8_t loadLevel)
- void [enableStatisticForComponentName](#) (std::string componentName, std::string statisticName)
- void [enableStatisticForComponentType](#) (std::string componentType, std::string statisticName)
- void [addStatisticParameterForComponentName](#) (const std::string &componentName, const std::string &statisticName, const std::string ¶m, const std::string &value)
- void [addStatisticParameterForComponentType](#) (const std::string &componentType, const std::string &statisticName, const std::string ¶m, const std::string &value)
- std::vector< [ConfigStatOutput](#) > & [getStatOutputs](#) ()
- const [ConfigStatOutput](#) & [getStatOutput](#) (size_t index=0) const
- long [getStatLoadLevel](#) () const
- void [addLink](#) (ComponentId_t comp_id, std::string link_name, std::string port, std::string latency_str, bool no_cut=false)
- void [setLinkNoCut](#) (std::string link_name)
- void [postCreationCleanup](#) ()
- bool [checkForStructuralErrors](#) ()
- [ConfigComponentMap_t](#) & [getComponentMap](#) ()
- const std::map< std::string, [ConfigStatGroup](#) > & [getStatGroups](#) () const
- [ConfigStatGroup](#) * [getStatGroup](#) (const std::string &name)
- bool [containsComponent](#) (ComponentId_t id) const
- [ConfigComponent](#) * [findComponent](#) (ComponentId_t)
- const [ConfigComponent](#) * [findComponent](#) (ComponentId_t) const
- [ConfigLinkMap_t](#) & [getLinkMap](#) ()
- [ConfigGraph](#) * [getSubGraph](#) (uint32_t start_rank, uint32_t end_rank)
- [ConfigGraph](#) * [getSubGraph](#) (const std::set< uint32_t > &rank_set)
- [PartitionGraph](#) * [getPartitionGraph](#) ()
- [PartitionGraph](#) * [getCollapsedPartitionGraph](#) ()
- void [annotateRanks](#) ([PartitionGraph](#) *graph)
- void [getConnectedNoCutComps](#) (ComponentId_t start, [ComponentIdMap_t](#) &group)
- void [serialize_order](#) ([SST::Core::Serialization::serializer](#) &ser) override

Friends

- class **Simulation**
- class **SSTSDLModelDefinition**

Additional Inherited Members

6.31.1 Detailed Description

A Configuration Graph A graph representing Components and Links

6.31.2 Member Function Documentation

6.31.2.1 `ComponentId_t SST::ConfigGraph::addComponent (ComponentId_t id, std::string name, std::string type, float weight, RankInfo rank)`

Create a new component with weight and rank

6.31.2.2 `ComponentId_t SST::ConfigGraph::addComponent (ComponentId_t id, std::string name, std::string type)`

Create a new component

6.31.2.3 `void SST::ConfigGraph::addLink (ComponentId_t comp_id, std::string link_name, std::string port, std::string latency_str, bool no_cut = false)`

Add a [Link](#) to a [Component](#) on a given Port

6.31.2.4 `void SST::ConfigGraph::addStatisticOutputParameter (const std::string & param, const std::string & value)`

Add parameter to the statistic output module

6.31.2.5 `void SST::ConfigGraph::addStatisticParameterForComponentName (const std::string & ComponentName, const std::string & statisticName, const std::string & param, const std::string & value)`

Add Parameters for a Statistic

6.31.2.6 `bool SST::ConfigGraph::checkForStructuralErrors ()`

Check the graph for Structural errors

6.31.2.7 `bool SST::ConfigGraph::checkRanks (RankInfo ranks)`

Verify that all components have valid Ranks assigned

6.31.2.8 `bool SST::ConfigGraph::containsComponentInRank (RankInfo rank)`

Checks to see if rank contains at least one component

6.31.2.9 `void SST::ConfigGraph::enableStatisticForComponentName (std::string ComponentName, std::string statisticName)`

Enable a Statistics assigned to a component

6.31.2.10 `ConfigComponentMap_t& SST::ConfigGraph::getComponentMap () [inline]`

Return the map of components

6.31.2.11 `ConfigLinkMap_t& SST::ConfigGraph::getLinkMap () [inline]`

Return the map of links

6.31.2.12 `void SST::ConfigGraph::postCreationCleanup ()`

Perform any post-creation cleanup processes

6.31.2.13 `void SST::ConfigGraph::print (std::ostream & os) const [inline]`

Print the configuration graph

6.31.2.14 `void SST::ConfigGraph::setComponentRanks (RankInfo rank)`

Helper function to set all the ranks to the same value

6.31.2.15 `void SST::ConfigGraph::setLinkNoCut (std::string link_name)`

Set a [Link](#) to be no-cut

6.31.2.16 `void SST::ConfigGraph::setStatisticLoadLevel (uint8_t loadLevel)`

Set the statistic system load level

6.31.2.17 `void SST::ConfigGraph::setStatisticOutput (const std::string & name)`

Set the statistic output module

6.31.2.18 void SST::ConfigGraph::setStatisticOutputParams (const Params & p)

Set a set of parameter to the statistic output module

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

- sst/core/configGraph.h
- sst/core/configGraph.cc

6.32 SST::Core::ConfigGraphOutput Class Reference

Inheritance diagram for SST::Core::ConfigGraphOutput:

Public Member Functions

- **ConfigGraphOutput** (const char *path)
- virtual void **generate** (const Config *cfg, ConfigGraph *graph)=0

Protected Attributes

- FILE * **outputFile**

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

- sst/core/configGraphOutput.h

6.33 SST::Core::ConfigGraphOutputException Class Reference

Inheritance diagram for SST::Core::ConfigGraphOutputException:

Collaboration diagram for SST::Core::ConfigGraphOutputException:

Public Member Functions

- **ConfigGraphOutputException** (const char *msg)
- virtual const char * **what** () const noexcept override

Protected Attributes

- char * **exMsg**

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

- sst/core/configGraphOutput.h

6.34 SST::ConfigLink Class Reference

```
#include <configGraph.h>
```

Inheritance diagram for SST::ConfigLink:

Collaboration diagram for SST::ConfigLink:

Public Member Functions

- LinkId_t **key** () const
- SimTime_t **getMinLatency** () const
- void **print** (std::ostream &os) const
- void **serialize_order** (SST::Core::Serialization::serializer &ser) override
- ImplementSerializable(SST::ConfigLink) private **ConfigLink** (LinkId_t id)
- **ConfigLink** (LinkId_t id, const std::string &n)
- void **updateLatencies** (TimeLord *)

Public Attributes

- LinkId_t id
- std::string name
- ComponentId_t component [2]
- std::string port [2]
- SimTime_t latency [2]
- std::string latency_str [2]
- int current_ref
- bool no_cut

Additional Inherited Members

6.34.1 Detailed Description

Represents the configuration of a generic [Link](#)

6.34.2 Member Function Documentation

6.34.2.1 SimTime_t SST::ConfigLink::getMinLatency () const `[inline]`

Return the minimum latency of this link (from both sides)

6.34.2.2 void SST::ConfigLink::print (std::ostream & os) const `[inline]`

Print the [Link](#) information

6.34.3 Member Data Documentation

6.34.3.1 ComponentId_t SST::ConfigLink::component[2]

IDs of the connected components

6.34.3.2 int SST::ConfigLink::current_ref

Number of components currently referring to this [Link](#)

6.34.3.3 LinkId_t SST::ConfigLink::id

ID of this link

6.34.3.4 SimTime_t SST::ConfigLink::latency[2]

Latency from each side

6.34.3.5 std::string SST::ConfigLink::latency_str[2]

Temp string holding latency

6.34.3.6 std::string SST::ConfigLink::name

Name of this link

6.34.3.7 bool SST::ConfigLink::no_cut

If set to true, partitioner will not make a cut through this [Link](#)

6.34.3.8 std::string SST::ConfigLink::port[2]

Names of the connected ports

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

- sst/core/configGraph.h
- sst/core/configGraph.cc

6.35 SST::ConfigStatGroup Class Reference

Inheritance diagram for SST::ConfigStatGroup:

Collaboration diagram for SST::ConfigStatGroup:

Public Member Functions

- **ConfigStatGroup** (const std::string &name)
- bool **addComponent** (ComponentId_t id)
- bool **addStatistic** (const std::string &name, [Params](#) &p)
- bool **setOutput** (size_t id)
- bool **setFrequency** (const std::string &freq)
- std::pair< bool, std::string > [verifyStatsAndComponents](#) (const [ConfigGraph](#) *graph)
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Public Attributes

- std::string **name**
- std::map< std::string, [Params](#) > **statMap**
- std::vector< ComponentId_t > **components**
- size_t **outputID**
- [UnitAlgebra](#) **outputFrequency**

Additional Inherited Members

6.35.1 Member Function Documentation

6.35.1.1 `std::pair< bool, std::string > SST::ConfigStatGroup::verifyStatsAndComponents (const ConfigGraph * graph)`

Checks to make sure that all components in the group support all of the statistics as configured in the group.

Returns

pair of: bool for OK, string for error message (if any)

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

- sst/core/configGraph.h
- sst/core/configGraph.cc

6.36 SST::ConfigStatOutput Class Reference

Inheritance diagram for SST::ConfigStatOutput:

Collaboration diagram for SST::ConfigStatOutput:

Public Member Functions

- **ConfigStatOutput** (const std::string &type)
- void **addParameter** (const std::string &key, const std::string &val)
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Public Attributes

- `std::string` **type**
- [Params](#) **params**

Additional Inherited Members

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

- `sst/core/configGraph.h`

6.37 SST::ELI::CtorList< Base, Ctor, Ctors > Struct Template Reference

Public Types

- `template<class NewBase >`
using **ChangeBase** = [CtorList](#)< NewBase, Ctor, Ctors... >

Static Public Member Functions

- `template<class T , int NumValid = 0, class U = T>`
static `std::enable_if< std::is_abstract< U >::value || is_tuple_constructible< U, Ctor >::value, bool >::type`
add ()
- `template<class T , int NumValid = 0, class U = T>`
static `std::enable_if< !std::is_abstract< U >::value && !is_tuple_constructible< U, Ctor >::value, bool >::type`
add ()

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

- `sst/core/eli/elementbuilder.h`

6.38 SST::ELI::CtorList< Base, void > Struct Template Reference

Static Public Member Functions

- `template<class T , int numValidCtors>`
static `bool` **add** ()

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

- `sst/core/eli/elementbuilder.h`

6.39 SST::ELI::DataBase< T > Class Template Reference

Static Public Member Functions

- static T * **get** (const std::string &elemlib, const std::string &elem)
- static void **add** (const std::string &elemlib, const std::string &elem, T *info)

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

- sst/core/eli/elementinfo.h

6.40 SST::decimal_fixedpoint< whole_words, fraction_words > Class Template Reference

```
#include <decimal_fixedpoint.h>
```

Public Member Functions

- constexpr int [getWholeWords](#) () const
- constexpr int [getFractionWords](#) () const
- [decimal_fixedpoint](#) ()
- [decimal_fixedpoint](#) (std::string init)
- template<class T >
[decimal_fixedpoint](#) (T init, typename std::enable_if< std::is_unsigned< T >::value >::type !=0)
- template<class T >
[decimal_fixedpoint](#) (T init, typename std::enable_if< std::is_signed< T >::value &&std::is_integral< T >::value >::type !=0)
- template<class T >
[decimal_fixedpoint](#) (const T init, typename std::enable_if< std::is_floating_point< T >::value >::type !=0)
- [decimal_fixedpoint](#) (const [decimal_fixedpoint](#) &init)
- [decimal_fixedpoint](#) & [operator=](#) (const [decimal_fixedpoint](#) &v)
- [decimal_fixedpoint](#) & [operator=](#) (uint64_t v)
- [decimal_fixedpoint](#) & [operator=](#) (int64_t v)
- [decimal_fixedpoint](#) & [operator=](#) (double v)
- [decimal_fixedpoint](#) & [operator=](#) (std::string v)
- void [negate](#) ()
- double [toDouble](#) () const
- int64_t [toLong](#) () const
- uint64_t [toUnsignedLong](#) () const
- template<typename T >
T [convert_to](#) (typename std::enable_if< std::is_unsigned< T >::value >::type !=0) const
- template<typename T >
T [convert_to](#) (typename std::enable_if< std::is_signed< T >::value &&std::is_integral< T >::value >::type !=0) const
- template<typename T >
T [convert_to](#) (typename std::enable_if< std::is_floating_point< T >::value >::type !=0) const
- std::string [toString](#) (int32_t precision=6) const
- [decimal_fixedpoint](#) & [operator+=](#) (const [decimal_fixedpoint](#) &v)
- [decimal_fixedpoint](#) & [operator-=](#) (const [decimal_fixedpoint](#) &v)
- [decimal_fixedpoint](#) & [operator*=](#) (const [decimal_fixedpoint](#) &v)

- `decimal_fixedpoint & operator/=` (const `decimal_fixedpoint` &*v*)
- `decimal_fixedpoint & inverse` ()
- `bool operator==` (const `decimal_fixedpoint` &*v*) const
- `bool operator!=` (const `decimal_fixedpoint` &*v*) const
- `bool operator>` (const `decimal_fixedpoint` &*v*) const
- `bool operator>=` (const `decimal_fixedpoint` &*v*) const
- `bool operator<` (const `decimal_fixedpoint` &*v*) const
- `bool operator<=` (const `decimal_fixedpoint` &*v*) const

Static Public Attributes

- static constexpr uint32_t **storage_radix** = 100000000
- static constexpr uint64_t **storage_radix_long** = 1000000000l
- static constexpr int32_t **digits_per_word** = 8

Friends

- template<int A, int B>
class **sst_dec_fixed**

6.40.1 Detailed Description

```
template<int whole_words, int fraction_words>
class SST::decimal_fixedpoint< whole_words, fraction_words >
```

Class that implements a decimal fixed-point number.

Fixed point class that stores digits in radix-10. Size is specified in words, and each word represents 8 digits.

Template Parameters

<i>whole_words</i>	Number of words used to represent the digits to the left of the decimal point. Each word represents 8 decimal digits.
<i>fraction_words</i>	Number of words used to represent the digits to the right of the decimal point. Each word represents 8 decimal digits.

6.40.2 Constructor & Destructor Documentation

6.40.2.1 `template<int whole_words, int fraction_words> SST::decimal_fixedpoint< whole_words, fraction_words >::decimal_fixedpoint ()` `[inline]`

Default constructor.

Builds a `decimal_fixedpoint` with the value 0;

6.40.2.2 `template<int whole_words, int fraction_words> SST::decimal_fixedpoint< whole_words, fraction_words >::decimal_fixedpoint (std::string init)` `[inline]`

Build a `decimal_fixedpoint` using a string initializer.

Parameters

<i>init</i>	Initialization string. The format is similar to the c++ double precision strings. For example 1.234, -1.234, 0.234, 1.234e14, 1.234e14, etc.
-------------	--

6.40.2.3 `template<int whole_words, int fraction_words> template<class T > SST::decimal_fixedpoint< whole_words, fraction_words >::decimal_fixedpoint (T init, typename std::enable_if< std::is_unsigned< T >::value >::type * = 0) [inline]`

Build a [decimal_fixedpoint](#) using a 64-bit unsigned number.

Parameters

<i>init</i>	Initialization value.
-------------	-----------------------

6.40.2.4 `template<int whole_words, int fraction_words> template<class T > SST::decimal_fixedpoint< whole_words, fraction_words >::decimal_fixedpoint (T init, typename std::enable_if< std::is_signed< T >::value &&std::is_integral< T >::value >::type * = 0) [inline]`

Build a [decimal_fixedpoint](#) using a 64-bit signed number.

Parameters

<i>init</i>	Initialization value.
-------------	-----------------------

6.40.2.5 `template<int whole_words, int fraction_words> template<class T > SST::decimal_fixedpoint< whole_words, fraction_words >::decimal_fixedpoint (const T init, typename std::enable_if< std::is_floating_point< T >::value >::type * = 0) [inline]`

Build a [decimal_fixedpoint](#) using a double.

Parameters

<i>init</i>	Initialization value.
-------------	-----------------------

6.40.2.6 `template<int whole_words, int fraction_words> SST::decimal_fixedpoint< whole_words, fraction_words >::decimal_fixedpoint (const decimal_fixedpoint< whole_words, fraction_words > & init) [inline]`

Build a [decimal_fixedpoint](#) using another [decimal_fixedpoint](#).

Parameters

<i>init</i>	Initialization value.
-------------	-----------------------

6.40.3 Member Function Documentation

6.40.3.1 `template<int whole_words, int fraction_words> template<typename T > T SST::decimal_fixedpoint< whole_words, fraction_words >::convert_to (typename std::enable_if< std::is_unsigned< T >::value >::type * = 0) const [inline]`

Templated conversion function for unsigned types.

6.40.3.2 `template<int whole_words, int fraction_words> template<typename T > T SST::decimal_fixedpoint< whole_words, fraction_words >::convert_to (typename std::enable_if< std::is_signed< T >::value &&std::is_integral< T >::value >::type * = 0) const [inline]`

Templated conversion function for signed integral types.

6.40.3.3 `template<int whole_words, int fraction_words> template<typename T > T SST::decimal_fixedpoint< whole_words, fraction_words >::convert_to (typename std::enable_if< std::is_floating_point< T >::value >::type * = 0) const [inline]`

Templated conversion function for floating point types.

6.40.3.4 `template<int whole_words, int fraction_words> constexpr int SST::decimal_fixedpoint< whole_words, fraction_words >::getFractionWords () const [inline]`

Get the value of fraction_words template parameter.

6.40.3.5 `template<int whole_words, int fraction_words> constexpr int SST::decimal_fixedpoint< whole_words, fraction_words >::getWholeWords () const [inline]`

Get the value of whole_words template parameter.

6.40.3.6 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::inverse () [inline]`

Inverts the number (1 divided by this number)

6.40.3.7 `template<int whole_words, int fraction_words> void SST::decimal_fixedpoint< whole_words, fraction_words >::negate () [inline]`

Negate the value (change the sign bit).

6.40.3.8 `template<int whole_words, int fraction_words> bool SST::decimal_fixedpoint< whole_words, fraction_words >::operator!= (const decimal_fixedpoint< whole_words, fraction_words > & v) const [inline]`

Checks to see if two numbers are not equal

Parameters

v	Number to check equality against
-----	----------------------------------

6.40.3.9 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator*= (const decimal_fixedpoint< whole_words, fraction_words > & v)`
`[inline]`

Multiplies another number to this one and sets it equal to the result.

Parameters

v	Number to multiply to this one
-----	--------------------------------

6.40.3.10 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator+= (const decimal_fixedpoint< whole_words, fraction_words > & v)`
`[inline]`

Adds another number to this one and sets it equal to the result.

Parameters

v	Number to add to this one
-----	---------------------------

6.40.3.11 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator-= (const decimal_fixedpoint< whole_words, fraction_words > & v)`
`[inline]`

Subtracts another number from this one and sets it equal to the result.

Parameters

v	Number to subtract from this one
-----	----------------------------------

6.40.3.12 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator/= (const decimal_fixedpoint< whole_words, fraction_words > & v)`
`[inline]`

Divides another number from this one and sets it equal to the result.

Parameters

v	Number to divide from this one
-----	--------------------------------

6.40.3.13 `template<int whole_words, int fraction_words> bool SST::decimal_fixedpoint< whole_words, fraction_words >::operator< (const decimal_fixedpoint< whole_words, fraction_words > & v) const` `[inline]`

Checks to see if this number is less than another number

Parameters

<code>v</code>	Number to compare to
----------------	----------------------

6.40.3.14 `template<int whole_words, int fraction_words> bool SST::decimal_fixedpoint< whole_words, fraction_words >::operator<= (const decimal_fixedpoint< whole_words, fraction_words > & v) const` `[inline]`

Checks to see if this number is less than or equal to another number

Parameters

<code>v</code>	Number to compare to
----------------	----------------------

6.40.3.15 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator= (const decimal_fixedpoint< whole_words, fraction_words > & v)` `[inline]`

Equal operator for other [decimal_fixedpoint](#) objects.

6.40.3.16 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator= (uint64_t v)` `[inline]`

Equal operator for 64-bit unsigned int.

6.40.3.17 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator= (int64_t v)` `[inline]`

Equal operator for 64-bit signed int.

6.40.3.18 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator= (double v)` `[inline]`

Equal operator for double.

6.40.3.19 `template<int whole_words, int fraction_words> decimal_fixedpoint& SST::decimal_fixedpoint< whole_words, fraction_words >::operator= (std::string v)` `[inline]`

Equal operator for string.

6.40.3.20 `template<int whole_words, int fraction_words> bool SST::decimal_fixedpoint< whole_words, fraction_words >::operator== (const decimal_fixedpoint< whole_words, fraction_words > & v) const` `[inline]`

Checks to see if two numbers are equal

Parameters

<i>v</i>	Number to check equality against
----------	----------------------------------

6.40.3.21 `template<int whole_words, int fraction_words> bool SST::decimal_fixedpoint< whole_words, fraction_words >::operator> (const decimal_fixedpoint< whole_words, fraction_words > & v) const` `[inline]`

Checks to see if this number is greater than another number

Parameters

<i>v</i>	Number to compare to
----------	----------------------

6.40.3.22 `template<int whole_words, int fraction_words> bool SST::decimal_fixedpoint< whole_words, fraction_words >::operator>= (const decimal_fixedpoint< whole_words, fraction_words > & v) const` `[inline]`

Checks to see if this number is greater than or equal to another number

Parameters

<i>v</i>	Number to compare to
----------	----------------------

6.40.3.23 `template<int whole_words, int fraction_words> double SST::decimal_fixedpoint< whole_words, fraction_words >::toDouble () const` `[inline]`

Return a double precision version of the [decimal_fixedpoint](#). There is possible precision loss in this conversion.

6.40.3.24 `template<int whole_words, int fraction_words> int64_t SST::decimal_fixedpoint< whole_words, fraction_words >::toLong () const` `[inline]`

Return a int64_t version of the [decimal_fixedpoint](#). There is possible precision loss in this conversion.

6.40.3.25 `template<int whole_words, int fraction_words> std::string SST::decimal_fixedpoint< whole_words, fraction_words >::toString (int32_t precision = 6) const` `[inline]`

Create a string representation of this [decimal_fixedpoint](#).

Parameters

<i>precision</i>	Precision to use when printing number
------------------	---------------------------------------

6.40.3.26 `template<int whole_words, int fraction_words> uint64_t SST::decimal_fixedpoint< whole_words, fraction_words >::toUnsignedLong () const [inline]`

Return a uint64_t version of the [decimal_fixedpoint](#). There is possible precision loss in this conversion.

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

- `sst/core/decimal_fixedpoint.h`

6.41 SST::ELI::DerivedBuilder< T, Base, Args > Struct Template Reference

Inheritance diagram for SST::ELI::DerivedBuilder< T, Base, Args >:

Collaboration diagram for SST::ELI::DerivedBuilder< T, Base, Args >:

Public Member Functions

- Base * **create** (Args...ctorArgs) override

Additional Inherited Members

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

- `sst/core/eli/elementbuilder.h`

6.42 SST::ELI::DerivedBuilder< SSTELEMENTPythonModule, SSTELEMENTPythonModuleOldELI, const std::string & > Struct Template Reference

Inheritance diagram for SST::ELI::DerivedBuilder< SSTELEMENTPythonModule, SSTELEMENTPythonModuleOldELI, const std::string & >:

Collaboration diagram for SST::ELI::DerivedBuilder< SSTELEMENTPythonModule, SSTELEMENTPythonModuleOldELI, const std::string & >:

Public Member Functions

- [SSTELEMENTPythonModule](#) * **create** (const std::string &lib) override
- **DerivedBuilder** (genPythonModuleFunction func)

Public Attributes

- genPythonModuleFunction **func_**

Additional Inherited Members

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

- sst/core/model/element_python.h

6.43 SST::Core::DotConfigGraphOutput Class Reference

Inheritance diagram for SST::Core::DotConfigGraphOutput:

Collaboration diagram for SST::Core::DotConfigGraphOutput:

Public Member Functions

- **DotConfigGraphOutput** (const char *path)
- virtual void **generate** (const [Config](#) *cfg, [ConfigGraph](#) *graph) override

Protected Member Functions

- void **generateDot** (const [ConfigComponent](#) &comp, const [ConfigLinkMap_t](#) &linkMap) const
- void **generateDot** (const [ConfigLink](#) &link) const

Additional Inherited Members

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

- sst/core/cfgoutput/dotConfigOutput.h
- sst/core/cfgoutput/dotConfigOutput.cc

6.44 SST::ElementInfoParam Struct Reference

```
#include <elibase.h>
```

Public Attributes

- const char * [name](#)
- const char * [description](#)
- const char * [defaultValue](#)

6.44.1 Detailed Description

Describes Parameters to a [Component](#).

6.44.2 Member Data Documentation

6.44.2.1 `const char* SST::ElementInfoParam::defaultValue`

Default value (if any) NULL == required parameter with no default, "" == optional parameter, blank default, "foo" == default value

6.44.2.2 `const char* SST::ElementInfoParam::description`

Brief description of the parameter (ie, what it controls)

6.44.2.3 `const char* SST::ElementInfoParam::name`

Name of the parameter

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

- `sst/core/eli/elibase.h`

6.45 SST::ElementInfoPort Struct Reference

```
#include <elibase.h>
```

Public Attributes

- `const char * name`
- `const char * description`
- `const char ** validEvents`

6.45.1 Detailed Description

Describes Ports that the [Component](#) can use

6.45.2 Member Data Documentation

6.45.2.1 `const char* SST::ElementInfoPort::description`

Brief description of the port (ie, what it is used for)

6.45.2.2 `const char* SST::ElementInfoPort::name`

Name of the port. Can contain d for a dynamic port, also `%(xxx)d` for dynamic port with xxx being the controlling component parameter

6.45.2.3 `const char** SST::ElementInfoPort::validEvents`

List of fully-qualified event types that this Port expects to send or receive

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

- `sst/core/eli/elibase.h`

6.46 SST::ElementInfoPort2 Struct Reference

```
#include <elibase.h>
```

Public Member Functions

- **ElementInfoPort2** (const [ElementInfoPort](#) *port)
- **ElementInfoPort2** (const char *name, const char *description, const std::vector< std::string > validEvents)

Public Attributes

- const char * name
- const char * description
- const std::vector< std::string > validEvents

6.46.1 Detailed Description

Describes Ports that the [Component](#) can use

6.46.2 Member Data Documentation

6.46.2.1 `const char* SST::ElementInfoPort2::description`

Brief description of the port (ie, what it is used for)

6.46.2.2 `const char* SST::ElementInfoPort2::name`

Name of the port. Can contain d for a dynamic port, also %(xxx)d for dynamic port with xxx being the controlling component parameter

6.46.2.3 `const std::vector<std::string> SST::ElementInfoPort2::validEvents`

List of fully-qualified event types that this Port expects to send or receive

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

- `sst/core/eli/elibase.h`

6.47 SST::ElementInfoStatistic Struct Reference

```
#include <elibase.h>
```

Public Attributes

- const char * [name](#)
- const char * [description](#)
- const char * [units](#)
- const uint8_t [enableLevel](#)

6.47.1 Detailed Description

Describes Statistics used by a [Component](#).

6.47.2 Member Data Documentation

6.47.2.1 const char* SST::ElementInfoStatistic::description

Brief description of the Statistic

6.47.2.2 const uint8_t SST::ElementInfoStatistic::enableLevel

Level to meet to enable statistic 0 = disabled

6.47.2.3 const char* SST::ElementInfoStatistic::name

Name of the Statistic to be Enabled

6.47.2.4 const char* SST::ElementInfoStatistic::units

[Units](#) associated with this Statistic value

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

- sst/core/eli/elibase.h

6.48 SST::ElementInfoSubComponentSlot Struct Reference

Public Attributes

- const char * **name**
- const char * **description**
- const char * **superclass**

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

- sst/core/eli/elibase.h

6.49 SST::ELI::ElementsBuilder< Base, CtorTuple > Struct Template Reference

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

- `sst/core/eli/elementbuilder.h`

6.50 SST::ELI::ElementsBuilder< Base, std::tuple< Args... > > Struct Template Reference

Static Public Member Functions

- static `BuilderLibrary`< Base, Args... > * **getLibrary** (const std::string &name)
- template<class T >
static `Builder`< Base, Args... > * **makeBuilder** ()

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

- `sst/core/eli/elementbuilder.h`

6.51 SST::ELI::ElementsInfo< Base > Struct Template Reference

Static Public Member Functions

- static `InfoLibrary`< Base > * **getLibrary** (const std::string &name)
- template<class T >
static bool **add** ()

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

- `sst/core/eli/elementinfo.h`

6.52 SST::ElemLoader Class Reference

```
#include <elemLoader.h>
```

Public Member Functions

- `ElemLoader` (const std::string &searchPaths)
- void `loadLibrary` (const std::string &elemLib, bool showErrors)
- std::vector< std::string > `getPotentialElements` ()

6.52.1 Detailed Description

Class to load Element Libraries

6.52.2 Constructor & Destructor Documentation

6.52.2.1 SST::ElemLoader::ElemLoader (const std::string & *searchPaths*)

Create a new ElementLoader with a given searchpath of directories

6.52.3 Member Function Documentation

6.52.3.1 std::vector< std::string > SST::ElemLoader::getPotentialElements ()

Returns a list of potential element libraries in the search path

6.52.3.2 void SST::ElemLoader::loadLibrary (const std::string & *elemlib*, bool *showErrors*)

Attempt to load a library

Parameters

<i>elemlib</i>	- The name of the Element Library to load
<i>showErrors</i>	- Print errors associated with attempting to find and load the library

Returns

Informational structure of the library, or NULL if it failed to load.

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

- sst/core/elemLoader.h
- sst/core/elemLoader.cc

6.53 SST::EmptyRankSync Class Reference

Inheritance diagram for SST::EmptyRankSync:

Collaboration diagram for SST::EmptyRankSync:

Public Member Functions

- [ActivityQueue](#) * **registerLink** (const [RankInfo](#) &UNUSED(to_rank), const [RankInfo](#) &UNUSED(from_rank), LinkId_t UNUSED(link_id), [Link](#) *UNUSED(link)) override
- void **execute** (int UNUSED(thread)) override
- void **exchangeLinkUntimedData** (int UNUSED_WO_MPI(thread), std::atomic< int > &UNUSED_WO_MPI(msg_count)) override
- void **finalizeLinkConfigurations** () override
- void **prepareForComplete** () override
- SimTime_t **getNextSyncTime** () override
- [TimeConverter](#) * **getMaxPeriod** ()
- uint64_t **getDataSize** () const override

Additional Inherited Members

6.53.1 Member Function Documentation

6.53.1.1 **ActivityQueue*** SST::EmptyRankSync::registerLink (const [RankInfo](#) & *UNUSEDto_rank*, const [RankInfo](#) & *UNUSEDfrom_rank*, LinkId_t *UNUSEDlink_id*, [Link](#) * *UNUSEDlink*) [inline],[override]

Register a [Link](#) which this Sync Object is responsible for

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

- sst/core/syncManager.cc

6.54 SST::EmptyThreadSync Class Reference

Inheritance diagram for SST::EmptyThreadSync:

Collaboration diagram for SST::EmptyThreadSync:

Public Member Functions

- void **before** () override
- void **after** () override
- void **execute** () override
- void **processLinkUntimedData** () override
- void **finalizeLinkConfigurations** () override
- void **prepareForComplete** () override
- void **registerLink** (LinkId_t UNUSED(link_id), [Link](#) *UNUSED(link)) override
- [ActivityQueue](#) * **getQueueForThread** (int UNUSED(tid)) override

Additional Inherited Members

6.54.1 Member Function Documentation

6.54.1.1 `void SST::EmptyThreadSync::registerLink (LinkId_t UNUSEDlink_id, Link * UNUSEDlink) [inline], [override]`

Register a [Link](#) which this Sync Object is responsible for

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

- sst/core/syncManager.cc

6.55 SST::Core::Environment::EnvironmentConfigGroup Class Reference

Public Member Functions

- **EnvironmentConfigGroup** (std::string name)
- std::string **getName** () const
- std::set< std::string > **getKeys** () const
- std::string **getValue** (std::string key)
- void **setValue** (std::string key, std::string value)
- void **print** ()
- void **writeTo** (FILE *outFile)

Protected Attributes

- std::string **groupName**
- std::map< std::string, std::string > **params**

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

- sst/core/env/envconfig.h
- sst/core/env/envconfig.cc

6.56 SST::Core::Environment::EnvironmentConfiguration Class Reference

Public Member Functions

- [EnvironmentConfigGroup](#) * **createGroup** (std::string groupName)
- void **removeGroup** (std::string groupName)
- std::set< std::string > **getGroupNames** ()
- [EnvironmentConfigGroup](#) * **getGroupByName** (std::string groupName)
- void **print** ()
- void **writeTo** (std::string filePath)
- void **writeTo** (FILE *outFile)

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

- sst/core/env/envconfig.h
- sst/core/env/envconfig.cc

6.57 SST::ComponentInfo::EqualsID Struct Reference

Public Member Functions

- bool **operator()** (const [ComponentInfo](#) *lhs, const [ComponentInfo](#) *rhs) const

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

- sst/core/componentInfo.h

6.58 SST::ComponentInfo::EqualsName Struct Reference

Public Member Functions

- bool **operator()** (const [ComponentInfo](#) *lhs, const [ComponentInfo](#) *rhs) const

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

- sst/core/componentInfo.h

6.59 SST::escaped_list_separator Struct Reference

Public Types

- typedef std::string::const_iterator **iter**

Public Member Functions

- **escaped_list_separator** (const std::string &esc="\\", const std::string &sep=",", const std::string "e="")
- void [operator\(\)](#) (iter &first, iter last, std::string &token)

Public Attributes

- std::string **e**
- std::string **q**
- std::string **s**

6.59.1 Member Function Documentation

6.59.1.1 void SST::escaped_list_separator::operator() (iter & *first*, iter *last*, std::string & *token*) [inline]

Returns

pair<iter, iter> = <tok_end, next_tok>

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

- sst/core/stringize.h

6.60 SST::Event Class Reference

```
#include <event.h>
```

Inheritance diagram for SST::Event:

Collaboration diagram for SST::Event:

Classes

- class [Handler](#)
- class [Handler< classT, void >](#)
- class [HandlerBase](#)

Functor classes for [Event](#) handling.

Public Types

- typedef std::pair< uint64_t, int > [id_type](#)

Public Member Functions

- void [execute](#) (void) override
- virtual [Event](#) * [clone](#) ()
- void [setDeliveryLink](#) (LinkId_t id, [Link](#) *link)
- [Link](#) * [getDeliveryLink](#) ()
- void [setRemoteEvent](#) ()
- LinkId_t [getLinkId](#) (void) const
- virtual void [print](#) (const std::string &header, [Output](#) &out) const override
- void [serialize_order](#) ([SST::Core::Serialization::serializer](#) &ser) override

Static Public Attributes

- static const [id_type](#) [NO_ID](#) = std::make_pair(0, -1)

Protected Member Functions

- [id_type](#) [generateUniqueId](#) ()

Protected Attributes

- [Link](#) * [delivery_link](#)

Additional Inherited Members

6.60.1 Detailed Description

Base class for Events - Items sent across links to communicate between components.

6.60.2 Member Typedef Documentation

6.60.2.1 `typedef std::pair<uint64_t, int> SST::Event::id_type`

Type definition of unique identifiers

6.60.3 Member Function Documentation

6.60.3.1 `Event * SST::Event::clone () [virtual]`

Clones the event in for the case of a broadcast

6.60.3.2 `void SST::Event::execute (void) [override],[virtual]`

Cause this event to fire

Implements [SST::Activity](#).

Reimplemented in [SST::NullEvent](#).

6.60.3.3 `Event::id_type SST::Event::generateUniqueld () [protected]`

Generates an ID that is unique across ranks, components and events.

6.60.3.4 `Link* SST::Event::getDeliveryLink () [inline]`

Gets the link id used for delivery. For use by SST Core only

6.60.3.5 `LinkId_t SST::Event::getLinkId (void) const [inline]`

Gets the link id associated with this event. For use by SST Core only

6.60.3.6 `virtual void SST::Event::print (const std::string & header, Output & out) const [inline],[override],[virtual]`

Virtual function to "pretty-print" this event. Should be implemented by subclasses.

Reimplemented from [SST::Activity](#).

Reimplemented in [SST::NullEvent](#).

6.60.3.7 `void SST::Event::setDeliveryLink (LinkId_t id, Link * link) [inline]`

Sets the link id used for delivery. For use by SST Core only

6.60.3.8 `void SST::Event::setRemoteEvent () [inline]`

For use by SST Core only

6.60.4 Member Data Documentation

6.60.4.1 `Link* SST::Event::delivery_link [protected]`

[Link](#) used for delivery

6.60.4.2 `const SST::Event::id_type SST::Event::NO_ID = std::make_pair(0, -1) [static]`

Constant, default value for id_types

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

- `sst/core/event.h`
- `sst/core/event.cc`

6.61 SST::Exit Class Reference

```
#include <exit.h>
```

Inheritance diagram for SST::Exit:

Collaboration diagram for SST::Exit:

Public Member Functions

- [Exit](#) (int num_threads, [TimeConverter](#) *period, bool single_rank)
- bool [refInc](#) (ComponentId_t, uint32_t thread)
- bool [refDec](#) (ComponentId_t, uint32_t thread)
- unsigned int [getRefCount](#) ()
- SimTime_t [getEndTime](#) ()
- void [execute](#) (void) override
- void [check](#) ()
- void [print](#) (const std::string &header, [Output](#) &out) const override
- unsigned int [getGlobalCount](#) ()

Additional Inherited Members

6.61.1 Detailed Description

[Exit Event Action](#)

Causes the simulation to halt

6.61.2 Constructor & Destructor Documentation

6.61.2.1 `SST::Exit::Exit (int num_threads, TimeConverter * period, bool single_rank)`

Create a new ExitEvent

Parameters

<i>sim</i>	- Simulation Object
<i>period</i>	- Period upon which to check for exit status
<i>single_rank</i>	- True if there are no parallel ranks

[Exit](#) needs to register a handler during constructor time, which requires a simulation object. But the simulation class creates an [Exit](#) object during it's construction, meaning that [Simulation::getSimulation\(\)](#) won't work yet. So [Exit](#) is the one exception to the "constructors shouldn't take simulation pointers" rule. However, it still needs to follow the "classes shouldn't contain pointers back to Simulation" rule.

6.61.3 Member Function Documentation

6.61.3.1 `void SST::Exit::execute (void) [override],[virtual]`

Function which will be called when the time for this [Activity](#) comes to pass.

Implements [SST::Activity](#).

6.61.3.2 `void SST::Exit::print (const std::string & header, Output & out) const [inline],[override],[virtual]`

Generic print-print function for this [Activity](#). Subclasses should override this function.

Reimplemented from [SST::Action](#).

6.61.3.3 `bool SST::Exit::refDec (ComponentId_t id, uint32_t thread)`

Decrement Reference Count for a given [Component](#) ID

6.61.3.4 `bool SST::Exit::refInc (ComponentId_t id, uint32_t thread)`

Increment Reference Count for a given [Component](#) ID

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

- sst/core/exit.h
- sst/core/exit.cc

6.62 SST::ELI::ExtendedCtor< NewCtor, OldCtor > Struct Template Reference

Public Types

- `template<class __NewCtor >`
using **ExtendCtor** = [ExtendedCtor](#)< __NewCtor, [ExtendedCtor](#)< NewCtor, OldCtor >>
- `template<class NewBase >`
using **ChangeBase** = typename NewCtor::template ChangeBase< NewBase >

Static Public Member Functions

- `template<class T >`
`static bool add ()`

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

- `sst/core/eli/elementbuilder.h`

6.63 SST::Factory Class Reference

```
#include <factory.h>
```

Collaboration diagram for SST::Factory:

Public Member Functions

- `bool isPortNameValid (const std::string &type, const std::string port_name)`
- `const Params::KeySet_t & getParamNames (const std::string &type)`
- `Component * CreateComponent (ComponentId_t id, std::string &componentname, Params ¶ms)`
- `void RequireEvent (std::string eventname)`
- `Module * CreateModule (std::string type, Params ¶ms)`
- `Module * CreateModuleWithComponent (std::string type, Component *comp, Params ¶ms)`
- `SubComponent * CreateSubComponent (std::string type, Component *comp, Params ¶ms)`
- `bool doesSubComponentExist (std::string type)`
- `Partition::SSTPartitioner * CreatePartitioner (std::string name, RankInfo total_ranks, RankInfo my_rank, int verbosity)`
- `template<class Base >`
`bool isSubComponentLoadableUsingAPI (std::string type)`
- `template<class Base , class... CtorArgs>`
`Base * Create (const std::string &type, SST::Params ¶ms, CtorArgs &&...args)`
- `template<class T , class... Args>`
`Statistics::Statistic< T > * CreateStatistic (std::string type, BaseComponent *comp, const std::string &statName, const std::string &stat, Params ¶ms, Args...args)`
- `SSTElementPythonModule * getPythonModule (std::string name)`
- `bool hasLibrary (std::string elemLib)`
- `void requireLibrary (std::string &elemLib)`
- `void getLoadedLibraryNames (std::set< std::string > &lib_names)`
- `void loadUnloadedLibraries (const std::set< std::string > &lib_names)`
- `bool DoesSubComponentSlotExist (const std::string &type, const std::string &slotName)`
- `bool DoesComponentInfoStatisticNameExist (const std::string &type, const std::string &statisticName)`
- `uint8_t GetComponentInfoStatisticEnableLevel (const std::string &type, const std::string &statisticName)`
- `std::string GetComponentInfoStatisticUnits (const std::string &type, const std::string &statisticName)`

Static Public Member Functions

- `static Factory * getFactory ()`

Protected Attributes

- [Output](#) & **out**

6.63.1 Detailed Description

Class for instantiating Components, Links and the like out of element libraries.

6.63.2 Member Function Documentation

6.63.2.1 `template<class Base , class... CtorArgs> Base* SST::Factory::Create (const std::string & type, SST::Params & params, CtorArgs &&... args)` `[inline]`

General function for a given base class

Parameters

<i>type</i>	
<i>params</i>	
<i>args</i>	Constructor arguments

6.63.2.2 `Component * SST::Factory::CreateComponent (ComponentId_t id, std::string & componentname, Params & params)`

Attempt to create a new [Component](#) instantiation

Parameters

<i>id</i>	- The unique ID of the component instantiation
<i>componentname</i>	- The fully qualified elementlibname.componentname type of component
<i>params</i>	- The params to pass to the component's constructor

Returns

Newly created component

6.63.2.3 `Module * SST::Factory::CreateModule (std::string type, Params & params)`

Instantiate a new [Module](#)

Parameters

<i>type</i>	- Fully qualified elementlibname.modulename type
<i>params</i>	- Parameters to pass to the Module 's constructor

6.63.2.4 **Module** * SST::Factory::CreateModuleWithComponent (std::string *type*, Component * *comp*, Params & *params*)

Instantiate a new [Module](#)

Parameters

<i>type</i>	- Fully qualified elementlibname.modulename type
<i>comp</i>	- Component instance to pass to the Module 's constructor
<i>params</i>	- Parameters to pass to the Module 's constructor

6.63.2.5 **Partitioner::SSTPartitioner** * SST::Factory::CreatePartitioner (std::string *name*, RankInfo *total_ranks*, RankInfo *my_rank*, int *verbosity*)

Return partitioner function

Parameters

<i>name</i>	- Fully qualified elementlibname.partitionner type name
-------------	---

6.63.2.6 **template<class T , class... Args> Statistics::Statistic<T>*** SST::Factory::CreateStatistic (std::string *type*, BaseComponent * *comp*, const std::string & *statName*, const std::string & *stat*, Params & *params*, Args... *args*) [inline]

Instantiate a new Statistic

Parameters

<i>comp</i>	- Owning component
<i>type</i>	- Fully qualified elementlibname.statisticname type
<i>statName</i>	- Name of the statistic
<i>stat↔ SubId</i>	- Name of the sub statistic
<i>params</i>	- Parameters to pass to the Statistics's constructor
<i>fieldType</i>	- Type of data stored in statistic

6.63.2.7 **SubComponent** * SST::Factory::CreateSubComponent (std::string *type*, Component * *comp*, Params & *params*)

Instantiate a new [Module](#)

Parameters

<i>type</i>	- Fully qualified elementlibname.modulename type
<i>comp</i>	- Component instance to pass to the SubComponent 's constructor
<i>params</i>	- Parameters to pass to the SubComponent 's constructor

6.63.2.8 `bool SST::Factory::DoesComponentInfoStatisticNameExist (const std::string & type, const std::string & statisticName)`

Determine if a statistic is defined in a components [ElementInfoStatistic](#)

Parameters

<i>type</i>	- The name of the component
<i>statisticName</i>	- The name of the statistic

Returns

True if the statistic is defined in the component's [ElementInfoStatistic](#)

6.63.2.9 `bool SST::Factory::DoesSubComponentSlotExist (const std::string & type, const std::string & slotName)`

Determine if a SubComponentSlot is defined in a components [ElementInfoStatistic](#)

Parameters

<i>type</i>	- The name of the component/subcomponent
<i>slotName</i>	- The name of the SubComponentSlot

Returns

True if the SubComponentSlot is defined in the component's ELI

6.63.2.10 `uint8_t SST::Factory::GetComponentInfoStatisticEnableLevel (const std::string & type, const std::string & statisticName)`

Determine if a statistic is defined in a subcomponents [ElementInfoStatistic](#)

Parameters

<i>type</i>	- The name of the subcomponent
<i>statisticName</i>	- The name of the statistic

Returns

True if the statistic is defined in the component's [ElementInfoStatistic](#) Get the enable level of a statistic defined in the component's [ElementInfoStatistic](#)

Parameters

<i>componentname</i>	- The name of the component
<i>statisticName</i>	- The name of the statistic

Returns

The Enable Level of the statistic from the [ElementInfoStatistic](#)

6.63.2.11 `std::string SST::Factory::GetComponentInfoStatisticUnits (const std::string & type, const std::string & statisticName)`

Get the units of a statistic defined in the component's [ElementInfoStatistic](#)

Parameters

<i>componentname</i>	- The name of the component
<i>statisticName</i>	- The name of the statistic

Returns

The units string of the statistic from the [ElementInfoStatistic](#)

6.63.2.12 `const Params::KeySet_t & SST::Factory::getParamNames (const std::string & type)`

Get a list of allowed param keys for a given component type.

Parameters

<i>type</i>	- Name of component in lib.name format
-------------	--

Returns

True if this is a valid portname

6.63.2.13 `SSTElementPythonModule * SST::Factory::getPythonModule (std::string name)`

Return Python [Module](#) creation function

Parameters

<i>name</i>	- Fully qualified elementlibname.pythonModName type name
-------------	--

6.63.2.14 `bool SST::Factory::hasLibrary (std::string elemlib)`

Checks to see if library exists and can be loaded

6.63.2.15 `bool SST::Factory::isPortNameValid (const std::string & type, const std::string port_name)`

Get a list of allowed ports for a given component type.

Parameters

<i>type</i>	- Name of component in lib.name format
-------------	--

Returns

True if this is a valid portname

6.63.2.16 `template<class Base > bool SST::Factory::isSubComponentLoadableUsingAPI (std::string type)` `[inline]`

Check to see if a given element type is loadable with a particular API

Parameters

<i>name</i>	- Name of element to check in lib.name format
-------------	---

Returns

True if loadable as the API specified as the template parameter

6.63.2.17 `void SST::Factory::RequireEvent (std::string eventname)`

Ensure that an element library containing the required event is loaded

Parameters

<i>eventname</i>	- The fully qualified elementlibname.eventname type
------------------	---

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

- sst/core/factory.h
- sst/core/factory.cc

6.64 SST::ELI::GetParams< T, Enable > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoParam](#) > & **get** ()

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

- sst/core/eli/paramsInfo.h

6.65 SST::ELI::GetParams< T, typename MethodDetect< decltype(T::ELI_getParams())>::type > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoParam](#) > & **get** ()

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

- sst/core/eli/paramsInfo.h

6.66 SST::Clock::Handler< classT, argT > Class Template Reference

```
#include <clock.h>
```

Inheritance diagram for SST::Clock::Handler< classT, argT >:

Collaboration diagram for SST::Clock::Handler< classT, argT >:

Public Member Functions

- [Handler](#) (classT *const object, PtrMember member, argT data)
- bool [operator\(\)](#) (Cycle_t cycle) override

6.66.1 Detailed Description

```
template<typename classT, typename argT = void>
class SST::Clock::Handler< classT, argT >
```

[Event Handler](#) class with user-data argument

Template Parameters

<i>classT</i>	Type of the Object
<i>argT</i>	Type of the argument

6.66.2 Constructor & Destructor Documentation

6.66.2.1 `template<typename classT, typename argT = void> SST::Clock::Handler< classT, argT >::Handler (classT *const object, PtrMember member, argT data) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler
<i>data</i>	- Additional argument to pass to handler

6.66.3 Member Function Documentation

6.66.3.1 `template<typename classT , typename argT = void> bool SST::Clock::Handler< classT, argT >::operator() (Cycle_t) [inline],[override],[virtual]`

Function called when [Handler](#) is invoked

Implements [SST::Clock::HandlerBase](#).

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

- `sst/core/clock.h`

6.67 SST::OneShot::Handler< classT, argT > Class Template Reference

```
#include <oneshot.h>
```

Inheritance diagram for `SST::OneShot::Handler< classT, argT >`:

Collaboration diagram for `SST::OneShot::Handler< classT, argT >`:

Public Member Functions

- [Handler](#) (`classT *const object`, `PtrMember member`, `argT data`)
- void [operator\(\)](#) () override

6.67.1 Detailed Description

```
template<typename classT, typename argT = void>
class SST::OneShot::Handler< classT, argT >
```

[Event Handler](#) class with user-data argument

Template Parameters

<i>classT</i>	Type of the Object
<i>argT</i>	Type of the argument

6.67.2 Constructor & Destructor Documentation

6.67.2.1 `template<typename classT , typename argT = void> SST::OneShot::Handler< classT, argT >::Handler (classT *const object, PtrMember member, argT data)` `[inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler
<i>data</i>	- Additional argument to pass to handler

6.67.3 Member Function Documentation

6.67.3.1 `template<typename classT , typename argT = void> void SST::OneShot::Handler< classT, argT >::operator() ()` `[inline]`, `[override]`, `[virtual]`

Operator to callback [OneShot Handler](#); called by the [OneShot](#) object to execute the users callback.

Parameters

<i>data</i>	- The data passed in when the handler was created
-------------	---

Implements [SST::OneShot::HandlerBase](#).

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

- `sst/core/oneshot.h`

6.68 SST::Event::Handler< classT, argT > Class Template Reference

```
#include <event.h>
```

Inheritance diagram for SST::Event::Handler< classT, argT >:

Collaboration diagram for SST::Event::Handler< classT, argT >:

Public Member Functions

- [Handler](#) (classT *const *object*, PtrMember *member*, argT *data*)
- void [operator\(\)](#) ([Event](#) **event*)

6.68.1 Detailed Description

```
template<typename classT, typename argT = void>
class SST::Event::Handler< classT, argT >
```

[Event Handler](#) class with user-data argument

6.68.2 Constructor & Destructor Documentation

6.68.2.1 `template<typename classT , typename argT = void> SST::Event::Handler< classT, argT >::Handler (classT *const object, PtrMember member, argT data)` `[inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler
<i>data</i>	- Additional argument to pass to handler

6.68.3 Member Function Documentation

6.68.3.1 `template<typename classT , typename argT = void> void SST::Event::Handler< classT, argT >::operator() (Event *)` `[inline]`, `[virtual]`

[Handler](#) function

Implements [SST::Event::HandlerBase](#).

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

- `sst/core/event.h`

6.69 SST::Interfaces::SimpleMem::Handler< classT, argT > Class Template Reference

```
#include <simpleMem.h>
```

Inheritance diagram for SST::Interfaces::SimpleMem::Handler< classT, argT >:

Collaboration diagram for SST::Interfaces::SimpleMem::Handler< classT, argT >:

Public Member Functions

- [Handler](#) (classT *const object, PtrMember member, argT data)
- void [operator\(\)](#) ([Request](#) *req)

6.69.1 Detailed Description

```
template<typename classT, typename argT = void>
class SST::Interfaces::SimpleMem::Handler< classT, argT >
```

[Event Handler](#) class with user-data argument

Template Parameters

<i>classT</i>	Type of the Object
<i>argT</i>	Type of the argument

6.69.2 Constructor & Destructor Documentation

6.69.2.1 `template<typename classT , typename argT = void> SST::Interfaces::SimpleMem::Handler< classT, argT >::Handler (classT *const object, PtrMember member, argT data) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler
<i>data</i>	- Additional argument to pass to handler

6.69.3 Member Function Documentation

6.69.3.1 `template<typename classT , typename argT = void> void SST::Interfaces::SimpleMem::Handler< classT, argT >::operator() (Request*) [inline],[virtual]`

Function called when [Handler](#) is invoked

Implements [SST::Interfaces::SimpleMem::HandlerBase](#).

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

- sst/core/interfaces/simpleMem.h

6.70 SST::Interfaces::SimpleNetwork::Handler< classT, argT > Class Template Reference

```
#include <simpleNetwork.h>
```

Inheritance diagram for SST::Interfaces::SimpleNetwork::Handler< classT, argT >:

Collaboration diagram for SST::Interfaces::SimpleNetwork::Handler< classT, argT >:

Public Member Functions

- [Handler](#) (classT *const object, PtrMember member, argT data)
- bool **operator()** (int vn)

6.70.1 Detailed Description

```
template<typename classT, typename argT = void>
class SST::Interfaces::SimpleNetwork::Handler< classT, argT >
```

[Event Handler](#) class with user-data argument

Template Parameters

<i>classT</i>	Type of the Object
<i>argT</i>	Type of the argument

6.70.2 Constructor & Destructor Documentation

6.70.2.1 `template<typename classT , typename argT = void> SST::Interfaces::SimpleNetwork::Handler< classT, argT >::Handler (classT *const object, PtrMember member, argT data) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler
<i>data</i>	- Additional argument to pass to handler

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

- sst/core/interfaces/simpleNetwork.h

6.71 SST::Clock::Handler< classT, void > Class Template Reference

```
#include <clock.h>
```

Inheritance diagram for SST::Clock::Handler< classT, void >:

Collaboration diagram for SST::Clock::Handler< classT, void >:

Public Member Functions

- [Handler](#) (classT *const *object*, PtrMember *member*)
- bool [operator\(\)](#) (Cycle_t *cycle*) override

6.71.1 Detailed Description

```
template<typename classT>
class SST::Clock::Handler< classT, void >
```

[Event Handler](#) class without user-data

Template Parameters

<i>classT</i>	Type of the Object
---------------	--------------------

6.71.2 Constructor & Destructor Documentation

6.71.2.1 `template<typename classT > SST::Clock::Handler< classT, void >::Handler (classT *const object, PtrMember member) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler

6.71.3 Member Function Documentation

6.71.3.1 `template<typename classT > bool SST::Clock::Handler< classT, void >::operator() (Cycle_t) [inline], [override], [virtual]`

Function called when [Handler](#) is invoked

Implements [SST::Clock::HandlerBase](#).

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

- `sst/core/clock.h`

6.72 SST::OneShot::Handler< classT, void > Class Template Reference

```
#include <oneshot.h>
```

Inheritance diagram for SST::OneShot::Handler< classT, void >:

Collaboration diagram for SST::OneShot::Handler< classT, void >:

Public Member Functions

- [Handler](#) (classT *const object, PtrMember member)
- void [operator\(\)](#) () override

6.72.1 Detailed Description

```
template<typename classT>
class SST::OneShot::Handler< classT, void >
```

[Event Handler](#) class without user-data

Template Parameters

<i>classT</i>	Type of the Object
---------------	--------------------

6.72.2 Constructor & Destructor Documentation

6.72.2.1 `template<typename classT > SST::OneShot::Handler< classT, void >::Handler (classT *const object, PtrMember member) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler

6.72.3 Member Function Documentation

6.72.3.1 `template<typename classT > void SST::OneShot::Handler< classT, void >::operator() () [inline], [override], [virtual]`

Operator to callback [OneShot Handler](#); called by the [OneShot](#) object to execute the users callback.

Implements [SST::OneShot::HandlerBase](#).

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

- `sst/core/oneshot.h`

6.73 SST::Event::Handler< classT, void > Class Template Reference

```
#include <event.h>
```

Inheritance diagram for SST::Event::Handler< classT, void >:

Collaboration diagram for SST::Event::Handler< classT, void >:

Public Member Functions

- [Handler](#) (classT *const object, PtrMember member)
- void [operator\(\)](#) ([Event](#) *event)

6.73.1 Detailed Description

```
template<typename classT>
class SST::Event::Handler< classT, void >
```

[Event Handler](#) class with no user-data.

6.73.2 Constructor & Destructor Documentation

6.73.2.1 `template<typename classT > SST::Event::Handler< classT, void >::Handler (classT *const object, PtrMember member) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler

6.73.3 Member Function Documentation

6.73.3.1 `template<typename classT > void SST::Event::Handler< classT, void >::operator() (Event *) [inline], [virtual]`

[Handler](#) function

Implements [SST::Event::HandlerBase](#).

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

- `sst/core/event.h`

6.74 SST::Interfaces::SimpleMem::Handler< classT, void > Class Template Reference

```
#include <simpleMem.h>
```

Inheritance diagram for SST::Interfaces::SimpleMem::Handler< classT, void >:

Collaboration diagram for SST::Interfaces::SimpleMem::Handler< classT, void >:

Public Member Functions

- [Handler](#) (classT *const object, PtrMember member)
- void [operator\(\)](#) ([Request](#) *req)

6.74.1 Detailed Description

```
template<typename classT>
class SST::Interfaces::SimpleMem::Handler< classT, void >
```

[Event Handler](#) class without user-data

Template Parameters

<i>classT</i>	Type of the Object
---------------	--------------------

6.74.2 Constructor & Destructor Documentation

6.74.2.1 `template<typename classT > SST::Interfaces::SimpleMem::Handler< classT, void >::Handler (classT *const object, PtrMember member) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler

6.74.3 Member Function Documentation

6.74.3.1 `template<typename classT > void SST::Interfaces::SimpleMem::Handler< classT, void >::operator() (Request *) [inline],[virtual]`

Function called when [Handler](#) is invoked

Implements [SST::Interfaces::SimpleMem::HandlerBase](#).

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

- sst/core/interfaces/simpleMem.h

6.75 SST::Interfaces::SimpleNetwork::Handler< classT, void > Class Template Reference

```
#include <simpleNetwork.h>
```

Inheritance diagram for SST::Interfaces::SimpleNetwork::Handler< classT, void >:

Collaboration diagram for SST::Interfaces::SimpleNetwork::Handler< classT, void >:

Public Member Functions

- [Handler](#) (classT *const object, PtrMember member)
- bool **operator()** (int vn)

6.75.1 Detailed Description

```
template<typename classT>
class SST::Interfaces::SimpleNetwork::Handler< classT, void >
```

[Event Handler](#) class without user-data

Template Parameters

<i>classT</i>	Type of the Object
---------------	--------------------

6.75.2 Constructor & Destructor Documentation

6.75.2.1 `template<typename classT> SST::Interfaces::SimpleNetwork::Handler< classT, void >::Handler (classT *const object, PtrMember member) [inline]`

Constructor

Parameters

<i>object</i>	- Pointer to Object upon which to call the handler
<i>member</i>	- Member function to call as the handler

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

- sst/core/interfaces/simpleNetwork.h

6.76 SST::Clock::HandlerBase Class Reference

```
#include <clock.h>
```

Inheritance diagram for SST::Clock::HandlerBase:

Public Member Functions

- virtual bool [operator\(\)](#) (Cycle_t)=0

6.76.1 Detailed Description

Functor classes for [Clock](#) handling

6.76.2 Member Function Documentation

6.76.2.1 `virtual bool SST::Clock::HandlerBase::operator() (Cycle_t) [pure virtual]`

Function called when [Handler](#) is invoked

Implemented in [SST::Clock::Handler< classT, void >](#), and [SST::Clock::Handler< classT, argT >](#).

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

- sst/core/clock.h

6.77 SST::OneShot::HandlerBase Class Reference

```
#include <oneshot.h>
```

Inheritance diagram for SST::OneShot::HandlerBase:

Public Member Functions

- virtual void [operator\(\)](#) ()=0

6.77.1 Detailed Description

Functor classes for [OneShot](#) handling

6.77.2 Member Function Documentation

6.77.2.1 virtual void SST::OneShot::HandlerBase::operator() () [pure virtual]

Function called when [Handler](#) is invoked

Implemented in [SST::OneShot::Handler< classT, void >](#), and [SST::OneShot::Handler< classT, argT >](#).

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

- sst/core/oneshot.h

6.78 SST::Event::HandlerBase Class Reference

Functor classes for [Event](#) handling.

```
#include <event.h>
```

Inheritance diagram for SST::Event::HandlerBase:

Public Member Functions

- virtual void [operator\(\)](#) ([Event](#) *)=0

6.78.1 Detailed Description

Functor classes for [Event](#) handling.

6.78.2 Member Function Documentation

6.78.2.1 `virtual void SST::Event::HandlerBase::operator()(Event *) [pure virtual]`

[Handler](#) function

Implemented in [SST::Event::Handler< classT, void >](#), and [SST::Event::Handler< classT, argT >](#).

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

- `sst/core/event.h`

6.79 SST::Interfaces::SimpleMem::HandlerBase Class Reference

```
#include <simpleMem.h>
```

Inheritance diagram for SST::Interfaces::SimpleMem::HandlerBase:

Public Member Functions

- `virtual void operator() (Request *)=0`

6.79.1 Detailed Description

Functor classes for [Clock](#) handling

6.79.2 Member Function Documentation

6.79.2.1 `virtual void SST::Interfaces::SimpleMem::HandlerBase::operator() (Request *) [pure virtual]`

Function called when [Handler](#) is invoked

Implemented in [SST::Interfaces::SimpleMem::Handler< classT, void >](#), and [SST::Interfaces::SimpleMem::Handler< classT, argT >](#).

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

- `sst/core/interfaces/simpleMem.h`

6.80 SST::Interfaces::SimpleNetwork::HandlerBase Class Reference

```
#include <simpleNetwork.h>
```

Inheritance diagram for SST::Interfaces::SimpleNetwork::HandlerBase:

Public Member Functions

- virtual bool **operator()** (int)=0

6.80.1 Detailed Description

Functor classes for handling of callbacks

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

- sst/core/interfaces/simpleNetwork.h

6.81 SST::ComponentInfo::HashID Struct Reference

Public Member Functions

- size_t **operator()** (const [ComponentInfo](#) *info) const

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

- sst/core/componentInfo.h

6.82 SST::ComponentInfo::HashName Struct Reference

Public Member Functions

- size_t **operator()** (const [ComponentInfo](#) *info) const

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

- sst/core/componentInfo.h

6.83 SST::SyncQueue::Header Struct Reference

Public Attributes

- uint32_t **mode**
- uint32_t **count**
- uint32_t **buffer_size**

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

- sst/core/syncQueue.h

6.84 SST::Statistics::HistogramStatistic< BinDataType > Class Template Reference

```
#include <stathistogram.h>
```

Inheritance diagram for SST::Statistics::HistogramStatistic< BinDataType >:

Collaboration diagram for SST::Statistics::HistogramStatistic< BinDataType >:

Public Member Functions

- **SST_ELI_DECLARE_STATISTIC_TEMPLATE** ([HistogramStatistic](#), "sst", "[HistogramStatistic](#)", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Track distribution of statistic across bins", "SST::Statistic<T>") [HistogramStatistic](#)([BaseComponent](#) *comp)
- allowedKeySet **insert** ("minvalue")
- allowedKeySet **insert** ("binwidth")
- allowedKeySet **insert** ("numbins")
- allowedKeySet **insert** ("dumpbinsonoutput")
- allowedKeySet **insert** ("includeoutofbounds")
- statParams **pushAllowedKeys** (allowedKeySet)
- this **setCollectionCount** (0)
- this **setStatisticTypeName** ("Histogram")

Public Attributes

- const std::string & **statName**
- const std::string const std::string & **statSubId**
- const std::string const std::string [Params](#) & **statParams**: [Statistic](#)<BinDataType>(comp
- const std::string const std::string [Params](#) **statName**
- const std::string const std::string [Params](#) **statSubId**
- const std::string const std::string [Params](#) **statParams**
- **m_minValue** = statParams.find<BinDataType>("minvalue", 0)
- **m_binWidth** = statParams.find<NumBinsType>("binwidth", 5000)
- **m_numBins** = statParams.find<NumBinsType>("numbins", 100)
- **m_dumpBinsOnOutput** = statParams.find<bool>("dumpbinsonoutput", true)
- **m_includeOutOfBounds** = statParams.find<bool>("includeoutofbounds", true)
- **m_totalSummed** = 0
- **m_totalSummedSqr** = 0
- **m_OOBMinCount** = 0
- **m_OOBMaxCount** = 0
- **m_itemsBinnedCount** = 0

Protected Member Functions

- void [addData_impl](#) (BinDataType value) override

Additional Inherited Members

6.84.1 Detailed Description

```
template<class BinDataType>
class SST::Statistics::HistogramStatistic< BinDataType >
```

Holder of data grouped into pre-determined width bins.

Template Parameters

<i>BinDataType</i>	is the type of the data held in each bin (i.e. what data type described the width of the bin)
--------------------	---

6.84.2 Member Function Documentation

6.84.2.1 `template<class BinDataType > void SST::Statistics::HistogramStatistic< BinDataType >::addData_impl (BinDataType value) [inline], [override], [protected]`

Adds a new value to the histogram. The correct bin is identified and then incremented. If no bin can be found to hold the value then a new bin is created.

6.84.3 Member Data Documentation

6.84.3.1 `template<class BinDataType > const std::string const std::string Params SST::Statistics::HistogramStatistic< BinDataType >::statParams`

Initial value:

```
{
    Params::KeySet_t allowedKeySet
```

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

- sst/core/statapi/stathistogram.h

6.85 SST::Statistics::ImplementsStatFields Struct Reference

Public Member Functions

- `std::string toString () const`
- `Statistics::fieldType_t fieldId () const`
- `const char * fieldName () const`
- `const char * fieldShortName () const`

Protected Member Functions

- `template<class T > ImplementsStatFields (T *UNUSED(t))`

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

- sst/core/statapi/statbase.h

6.86 SST::ELI::InfoDatabase Struct Reference

Static Public Member Functions

- `template<class T >`
`static InfoLibrary< T > * getLibrary (const std::string &name)`

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

- `sst/core/eli/elementinfo.h`

6.87 SST::ELI::InfoLibrary< Base > Class Template Reference

Public Types

- using **BaseInfo** = typename Base::BuilderInfo

Public Member Functions

- **InfoLibrary** (const std::string &name)
- BaseInfo * **getInfo** (const std::string &name)
- bool **hasInfo** (const std::string &name) const
- int **numEntries** () const
- const std::map< std::string, BaseInfo * > & **getMap** () const
- void **readdInfo** (const std::string &name, BaseInfo *info)
- bool **addInfo** (const std::string &elem, BaseInfo *info)

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

- `sst/core/eli/elementinfo.h`

6.88 SST::ELI::InfoLibraryDatabase< Base > Class Template Reference

Public Types

- using **Library** = InfoLibrary< Base >
- using **BaseInfo** = typename Library::BaseInfo
- using **Map** = std::map< std::string, Library * >

Static Public Member Functions

- static **Library** * **getLibrary** (const std::string &name)

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

- `sst/core/eli/elementinfo.h`

6.89 SST::ELI::InfoPorts< T, Enable > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoPort2](#) > & get ()

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

- sst/core/eli/portsInfo.h

6.90 SST::ELI::InfoPorts< T, typename MethodDetect< decltype(T::ELI_getPorts())>::type > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoPort2](#) > & get ()

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

- sst/core/eli/portsInfo.h

6.91 SST::ELI::InfoStats< T, Enable > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoStatistic](#) > & get ()

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

- sst/core/eli/statsInfo.h

6.92 SST::ELI::InfoStats< T, typename MethodDetect< decltype(T::ELI_getStatistics())>::type > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoStatistic](#) > & get ()

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

- sst/core/eli/statsInfo.h

6.93 SST::ELI::InfoSubs< T, Enable > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoSubComponentSlot](#) > & **get** ()

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

- sst/core/eli/subcompSlotInfo.h

6.94 SST::ELI::InfoSubs< T, typename MethodDetect< decltype(T::ELI_getSubComponentSlots())>::type > Struct Template Reference

Static Public Member Functions

- static const std::vector< [SST::ElementInfoSubComponentSlot](#) > & **get** ()

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

- sst/core/eli/subcompSlotInfo.h

6.95 SST::InitQueue Class Reference

```
#include <initQueue.h>
```

Inheritance diagram for SST::InitQueue:

Collaboration diagram for SST::InitQueue:

Public Member Functions

- bool [empty](#) () override
- int [size](#) () override
- void [insert](#) ([Activity](#) *activity) override
- [Activity](#) * [pop](#) () override
- [Activity](#) * [front](#) () override

6.95.1 Detailed Description

[ActivityQueue](#) for use during the init() phase

6.95.2 Member Function Documentation

6.95.2.1 `bool SST::InitQueue::empty () [override],[virtual]`

Returns true if the queue is empty

Implements [SST::ActivityQueue](#).

6.95.2.2 `Activity * SST::InitQueue::front () [override],[virtual]`

Returns the next activity

Implements [SST::ActivityQueue](#).

6.95.2.3 `void SST::InitQueue::insert (Activity * activity) [override],[virtual]`

Insert a new activity into the queue

Implements [SST::ActivityQueue](#).

6.95.2.4 `Activity * SST::InitQueue::pop () [override],[virtual]`

Remove and return the next activity

Implements [SST::ActivityQueue](#).

6.95.2.5 `int SST::InitQueue::size () [override],[virtual]`

Returns the number of activities in the queue

Implements [SST::ActivityQueue](#).

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

- `sst/core/initQueue.h`
- `sst/core/initQueue.cc`

6.96 SST::ELI::InstantiateBuilder< Base, T > Struct Template Reference

Static Public Member Functions

- static bool **isLoaded** ()

Static Public Attributes

- static const bool **loaded** = Base::Ctor::template add<T>()

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

- sst/core/eli/elementbuilder.h

6.97 SST::ELI::InstantiateBuilderInfo< Base, T > Struct Template Reference

Static Public Member Functions

- static bool **isLoaded** ()

Static Public Attributes

- static const bool **loaded** = ElementsInfo<Base>::template add<T>()

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

- sst/core/eli/elementinfo.h

6.98 SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType > Class Template Reference

```
#include <ipctunnel.h>
```

Public Member Functions

- [IPCTunnel](#) (uint32_t comp_id, size_t numBuffers, size_t bufferSize, uint32_t expectedChildren=1)
- [IPCTunnel](#) (const std::string ®ion_name)
- virtual [~IPCTunnel](#) ()
- void [shutdown](#) (bool all=false)
- const std::string & [getRegionName](#) (void) const
- ShareDataType * [getSharedData](#) ()
- void [writeMessage](#) (size_t core, const MsgType &command)
- MsgType [readMessage](#) (size_t buffer)
- bool [readMessageNB](#) (size_t buffer, MsgType *result)
- void [clearBuffer](#) (size_t core)

Protected Attributes

- ShareDataType * [sharedData](#)

6.98.1 Detailed Description

```
template<typename ShareDataType, typename MsgType>
class SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >
```

Tunneling class between two processes, connected by shared memory. Supports multiple circular-buffer queues, and a generic region of memory for shared data.

Template Parameters

<i>ShareDataType</i>	Type to put in the shared data region
<i>MsgType</i>	Type of messages being sent in the circular buffers

6.98.2 Constructor & Destructor Documentation

6.98.2.1 `template<typename ShareDataType , typename MsgType > SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::IPCTunnel (uint32_t comp_id, size_t numBuffers, size_t bufferSize, uint32_t expectedChildren = 1) [inline]`

Construct a new Tunnel for IPC Communications

Parameters

<i>comp_id</i>	Component ID of owner
<i>numBuffers</i>	Number of buffers for which we should tunnel
<i>bufferSize</i>	How large each core's buffer should be

6.98.2.2 `template<typename ShareDataType , typename MsgType > SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::IPCTunnel (const std::string & region_name) [inline]`

Access an existing Tunnel

Parameters

<i>region_name</i>	Name of the shared-memory region to access
--------------------	--

6.98.2.3 `template<typename ShareDataType , typename MsgType > virtual SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::~IPCTunnel () [inline],[virtual]`

Destructor

6.98.3 Member Function Documentation

6.98.3.1 `template<typename ShareDataType , typename MsgType > void SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::clearBuffer (size_t core) [inline]`

Empty the messages in the buffer

6.98.3.2 `template<typename ShareDataType , typename MsgType > ShareDataType* SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::getSharedData () [inline]`

return a pointer to the ShareDataType region

6.98.3.3 `template<typename ShareDataType , typename MsgType > MsgType SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::readMessage (size_t buffer) [inline]`

Blocks until a command is available

6.98.3.4 `template<typename ShareDataType , typename MsgType > bool SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::readMessageNB (size_t buffer, MsgType * result) [inline]`

Non-blocking version of readMessage

6.98.3.5 `template<typename ShareDataType , typename MsgType > void SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::shutdown (bool all = false) [inline]`

Shutdown

6.98.3.6 `template<typename ShareDataType , typename MsgType > void SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::writeMessage (size_t core, const MsgType & command) [inline]`

Blocks until space is available

6.98.4 Member Data Documentation

6.98.4.1 `template<typename ShareDataType , typename MsgType > ShareDataType* SST::Core::Interprocess::IPCTunnel< ShareDataType, MsgType >::sharedData [protected]`

Pointer to the Shared Data Region

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

- sst/core/interprocess/ipctunnel.h

6.99 SST::ELI::is_tuple_constructible< T, U > Struct Template Reference

Inheritance diagram for SST::ELI::is_tuple_constructible< T, U >:

Collaboration diagram for SST::ELI::is_tuple_constructible< T, U >:

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

- sst/core/eli/elementbuilder.h

6.100 SST::ELI::is_tuple_constructible< T, std::tuple< Args... > > Struct Template Reference

Inheritance diagram for SST::ELI::is_tuple_constructible< T, std::tuple< Args... > >:

Collaboration diagram for SST::ELI::is_tuple_constructible< T, std::tuple< Args... > >:

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

- sst/core/eli/elementbuilder.h

6.101 SST::Core::JSONConfigGraphOutput Class Reference

Inheritance diagram for SST::Core::JSONConfigGraphOutput:

Collaboration diagram for SST::Core::JSONConfigGraphOutput:

Public Member Functions

- **JSONConfigGraphOutput** (const char *path)
- virtual void **generate** (const [Config](#) *cfg, [ConfigGraph](#) *graph) override

Protected Member Functions

- void **generateJSON** (const std::string indent, const [ConfigComponent](#) &comp, const [ConfigLinkMap_t](#) &link↔Map) const

Additional Inherited Members

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

- sst/core/cfgoutput/jsonConfigOutput.h
- sst/core/cfgoutput/jsonConfigOutput.cc

6.102 SST::Activity::less_time Class Reference

```
#include <activity.h>
```

Public Member Functions

- bool **operator()** (const [Activity](#) *lhs, const [Activity](#) *rhs) const
- bool **operator()** (const [Activity](#) &lhs, const [Activity](#) &rhs) const

6.102.1 Detailed Description

Comparator class to use with STL container classes.

6.102.2 Member Function Documentation

6.102.2.1 `bool SST::Activity::less_time::operator() (const Activity * lhs, const Activity * rhs) const` `[inline]`

Compare pointers

6.102.2.2 `bool SST::Activity::less_time::operator() (const Activity & lhs, const Activity & rhs) const` `[inline]`

Compare references

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

- `sst/core/activity.h`

6.103 SST::Activity::less_time_priority Class Reference

```
#include <activity.h>
```

Public Member Functions

- `bool operator() (const Activity *lhs, const Activity *rhs)`
- `bool operator() (const Activity &lhs, const Activity &rhs)`

6.103.1 Detailed Description

Comparator class to use with STL container classes.

6.103.2 Member Function Documentation

6.103.2.1 `bool SST::Activity::less_time_priority::operator() (const Activity * lhs, const Activity * rhs)` `[inline]`

Compare based off pointers

6.103.2.2 `bool SST::Activity::less_time_priority::operator() (const Activity & lhs, const Activity & rhs)` `[inline]`

Compare based off references

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

- `sst/core/activity.h`

6.104 SST::Activity::less_time_priority_order Class Reference

```
#include <activity.h>
```

Public Member Functions

- bool `operator()` (const `Activity` *lhs, const `Activity` *rhs) const
- bool `operator()` (const `Activity` &lhs, const `Activity` &rhs) const

6.104.1 Detailed Description

To use with STL container classes.

6.104.2 Member Function Documentation

6.104.2.1 bool SST::Activity::less_time_priority_order::operator() (const `Activity` * lhs, const `Activity` * rhs) const
[inline]

Compare based off pointers

6.104.2.2 bool SST::Activity::less_time_priority_order::operator() (const `Activity` & lhs, const `Activity` & rhs) const
[inline]

Compare based off references

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

- sst/core/activity.h

6.105 SST::Link Class Reference

```
#include <link.h>
```

Inheritance diagram for SST::Link:

Collaboration diagram for SST::Link:

Public Member Functions

- [Link](#) (LinkId_t id)
- void [setLatency](#) (Cycle_t lat)
- void [addRecvLatency](#) (int cycles, std::string timebase)
- void [addRecvLatency](#) (SimTime_t cycles, [TimeConverter](#) *timebase)
- void [setFunctor](#) ([Event::HandlerBase](#) *functor)
- void [setPolling](#) ()
- void [send](#) (SimTime_t delay, [TimeConverter](#) *tc, [Event](#) *event)
- void [send](#) (SimTime_t delay, [Event](#) *event)
- void [send](#) ([Event](#) *event)
- [Event](#) * [recv](#) ()
- void [setDefaultTimeBase](#) ([TimeConverter](#) *tc)
- [TimeConverter](#) * [getDefaultTimeBase](#) ()
- void [deliverEvent](#) ([Event](#) *event) const
- LinkId_t [getId](#) ()
- void [sendUntimedData](#) ([Event](#) *data)
- [Event](#) * [recvUntimedData](#) ()
- void [sendInitData](#) ([Event](#) *init_data)
- [Event](#) * [recvInitData](#) ()
- void [setAsConfigured](#) ()
- bool [isConfigured](#) ()

Protected Attributes

- [ActivityQueue](#) * [recvQueue](#)
- [ActivityQueue](#) * [untimedQueue](#)
- [ActivityQueue](#) * [configuredQueue](#)
- [Event::HandlerBase](#) * [rFunctor](#)
- [TimeConverter](#) * [defaultTimeBase](#)
- SimTime_t [latency](#)
- [Link](#) * [pair_link](#)

Static Protected Attributes

- static [ActivityQueue](#) * [uninitQueue](#) = new [UninitializedQueue](#)("ERROR: Trying to [send](#) or [recv](#) from link during initialization. Send and Recv cannot be called before setup.")
- static [ActivityQueue](#) * [afterInitQueue](#) = new [UninitializedQueue](#)("ERROR: Trying to call [sendUntimedData](#)/[sendInitData](#) or [recvUntimedData](#)/[recvInitData](#) during the run phase.")
- static [ActivityQueue](#) * [afterRunQueue](#) = new [UninitializedQueue](#)("ERROR: Trying to call [send](#) or [recv](#) during complete phase.")

Friends

- class [LinkPair](#)
- class [NewRankSync](#)
- class [NewThreadSync](#)
- class [Simulation](#)
- class [SyncBase](#)
- class [ThreadSync](#)
- class [SyncManager](#)
- class [ComponentInfo](#)

6.105.1 Detailed Description

[Link](#) between two components. Carries events

6.105.2 Constructor & Destructor Documentation

6.105.2.1 SST::Link::Link (LinkId_t *id*)

Create a new link with a given ID

6.105.3 Member Function Documentation

6.105.3.1 void SST::Link::addRecvLatency (int *cycles*, std::string *timebase*)

Set additional Latency to be added on to events coming in on this link.

Parameters

<i>cycles</i>	Number of Cycles to be added
<i>timebase</i>	Base Units of cycles

6.105.3.2 void SST::Link::addRecvLatency (SimTime_t *cycles*, TimeConverter * *timebase*)

Set additional Latency to be added on to events coming in on this link.

Parameters

<i>cycles</i>	Number of Cycles to be added
<i>timebase</i>	Base Units of cycles

6.105.3.3 void SST::Link::deliverEvent (Event * *event*) const [inline]

Causes an event to be delivered to the registered callback

6.105.3.4 TimeConverter * SST::Link::getDefaultTimeBase ()

Return the default Time Base for this link

6.105.3.5 LinkId_t SST::Link::getId () [inline]

Return the ID of this link

6.105.3.6 Event * SST::Link::recv ()

Retrieve a pending event from the [Link](#). For links which do not have a set event handler, they can be polled with this function. Returns NULL if there is no pending event.

6.105.3.7 Event* SST::Link::recvInitData () [inline]

Receive an event (if any) during the complete() phase

6.105.3.8 Event * SST::Link::recvUntimedData ()

Receive an event (if any) during the init() phase

6.105.3.9 void SST::Link::send (SimTime_t delay, TimeConverter * tc, Event * event)

Send an event over the link with additional delay. Sends an event over a link with an additional delay specified with a [TimeConverter](#). I.e. the total delay is the link's delay + the additional specified delay.

Parameters

<i>delay</i>	- additional delay
<i>tc</i>	- time converter to specify units for the additional delay
<i>event</i>	- the Event to send

6.105.3.10 void SST::Link::send (SimTime_t delay, Event * event) [inline]

Send an event with additional delay. Sends an event over a link with additional delay specified by the [Link](#)'s default timebase.

Parameters

<i>delay</i>	The additional delay, in units of the default Link timebase
<i>event</i>	The event to send

6.105.3.11 void SST::Link::send (Event * event) [inline]

Send an event with the [Link](#)'s default delay

Parameters

<i>event</i>	The event to send
--------------	-------------------

6.105.3.12 `void SST::Link::sendInitData (Event * init_data) [inline]`

Send data during the complete() phase.

6.105.3.13 `void SST::Link::sendUntimedData (Event * data)`

Send data during the init() phase.

6.105.3.14 `void SST::Link::setDefaultTimeBase (TimeConverter * tc)`

Manually set the default defaultTimeBase

Parameters

<i>tc</i>	TimeConverter object for the timebase
-----------	---

6.105.3.15 `void SST::Link::setFunctor (Event::HandlerBase * functor) [inline]`

Set the callback function to be called when a message is delivered.

6.105.3.16 `void SST::Link::setLatency (Cycle_t lat)`

set minimum link latency

6.105.3.17 `void SST::Link::setPolling ()`

Specifies that this link has no callback, and is poll-based only

6.105.4 Member Data Documentation

6.105.4.1 `ActivityQueue * SST::Link::afterInitQueue = new UninitializedQueue("ERROR: Trying to call sendUntimedData/sendInitData or recvUntimedData/recvInitData during the run phase.") [static], [protected]`

Uninitialized queue. Used for error detection

6.105.4.2 `ActivityQueue * SST::Link::afterRunQueue = new UninitializedQueue("ERROR: Trying to call send or recv during complete phase.") [static], [protected]`

Uninitialized queue. Used for error detection

6.105.4.3 `ActivityQueue* SST::Link::configuredQueue` `[protected]`

Currently active Queue

6.105.4.4 `TimeConverter* SST::Link::defaultTimeBase` `[protected]`

Timebase used if no other timebase is specified. Used to specify the units for added delays when sending, such as in `Link::send()`. Often set by the `Component::registerClock()` function if the `regAll` argument is true.

6.105.4.5 `SimTime_t SST::Link::latency` `[protected]`

Latency of the link. It is used by the partitioner as the weight. This latency is added to the delay put on the event by the component.

6.105.4.6 `Link* SST::Link::pair_link` `[protected]`

Pointer to the opposite side of this link

6.105.4.7 `ActivityQueue* SST::Link::recvQueue` `[protected]`

Queue of events to be received by the owning component

6.105.4.8 `Event::HandlerBase* SST::Link::rFuncor` `[protected]`

Receive functor. This functor is set when the link is connected. Determines what the receiver wants to be called

6.105.4.9 `ActivityQueue * SST::Link::uninitQueue = new UninitializedQueue("ERROR: Trying to send or recv from link during initialization. Send and Recv cannot be called before setup.")` `[static], [protected]`

Uninitialized queue. Used for error detection

6.105.4.10 `ActivityQueue* SST::Link::untimedQueue` `[protected]`

Queue of events to be received during init by the owning component

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

- `sst/core/link.h`
- `sst/core/link.cc`

6.106 `SST::LinkMap` Class Reference

```
#include <linkMap.h>
```

Public Member Functions

- void [addSelfPort](#) (std::string &name)
- bool **isSelfPort** (const std::string &name) const
- void [insertLink](#) (std::string name, [Link](#) *link)
- void **removeLink** (std::string name)
- [Link](#) * [getLink](#) (std::string name)
- bool [empty](#) ()
- std::map< std::string, [Link](#) * > & [getLinkMap](#) ()

6.106.1 Detailed Description

Maps port names to the Links that are connected to it

6.106.2 Member Function Documentation

6.106.2.1 void SST::LinkMap::addSelfPort (std::string & *name*) [inline]

Set the list of allowed port names from the [ElementInfoPort](#) Add a port name to the list of allowed ports. Used by SelfLinks, as these are undocumented.

6.106.2.2 bool SST::LinkMap::empty () [inline]

Checks to see if [LinkMap](#) is empty.

Returns

True if [Link](#) map is empty, false otherwise

6.106.2.3 [Link](#)* SST::LinkMap::getLink (std::string *name*) [inline]

Returns a [Link](#) pointer for a given name

6.106.2.4 std::map<std::string,[Link](#)*>& SST::LinkMap::getLinkMap () [inline]

Return a reference to the internal map

6.106.2.5 void SST::LinkMap::insertLink (std::string *name*, [Link](#) * *link*) [inline]

Inserts a new pair of name and link into the map

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

- sst/core/linkMap.h

6.107 SST::LinkPair Class Reference

```
#include <linkPair.h>
```

Public Member Functions

- [LinkPair](#) (LinkId_t id)
- LinkId_t [getId](#) ()
- [Link](#) * [getLeft](#) ()
- [Link](#) * [getRight](#) ()

6.107.1 Detailed Description

Defines a pair of links (to define a connected link)

6.107.2 Constructor & Destructor Documentation

6.107.2.1 SST::LinkPair::LinkPair (LinkId_t *id*) [inline]

Create a new [LinkPair](#) with specified ID

6.107.3 Member Function Documentation

6.107.3.1 LinkId_t SST::LinkPair::getId () [inline]

return the ID of the [LinkPair](#)

6.107.3.2 [Link](#)* SST::LinkPair::getLeft () [inline]

Return the Left [Link](#)

6.107.3.3 [Link](#)* SST::LinkPair::getRight () [inline]

Return the Right [Link](#)

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

- sst/core/linkPair.h

6.108 LinkPy_t Struct Reference

Public Attributes

- PyObject_HEAD char * **name**
- bool **no_cut**
- char * **latency**

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

- sst/core/model/pymodel_link.h

6.109 SST::ELI::LoadedLibraries Class Reference

Public Types

- using **InfoMap** = std::map< std::string, std::function< void()>>
- using **LibraryMap** = std::map< std::string, InfoMap >

Static Public Member Functions

- static bool **isLoaded** (const std::string &name)
- static bool **addLoader** (const std::string &lib, const std::string &name, std::function< void()> &&loader)
- static const LibraryMap & **getLoaders** ()

6.109.1 Member Function Documentation

6.109.1.1 bool SST::ELI::LoadedLibraries::addLoader (const std::string & *lib*, const std::string & *name*, std::function< void()> && *loader*) [static]

Returns

A boolean indicated successfully added

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

- sst/core/eli/elibase.h
- sst/core/eli/elibase.cc

6.110 SST::LoaderData Struct Reference

Collaboration diagram for SST::LoaderData:

Public Attributes

- [It_dladvise advise_handle](#)

6.110.1 Detailed Description

This structure exists so that we don't need to have any libtool-specific code (and therefore need the libtool headers) in [factory.h](#)

6.110.2 Member Data Documentation

6.110.2.1 It_dladvise SST::LoaderData::advise_handle

Handle from Libtool

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

- [sst/core/elemLoader.cc](#)

6.111 It__advise Struct Reference

Public Attributes

- unsigned int **try_ext**:1
- unsigned int **is_resident**:1
- unsigned int **is_symglobal**:1
- unsigned int **is_symlocal**:1
- unsigned int **try_preload_only**:1

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

- [sst/core/libltdl/libltdl/lt__private.h](#)

6.112 It__handle Struct Reference

Collaboration diagram for It__handle:

Public Attributes

- [lt_dlhandle](#) **next**
- const [lt_dlvtable](#) * **vtable**
- [lt_dlinfo](#) **info**
- int **depcount**
- [lt_dlhandle](#) * **deplibs**
- [lt_module](#) **module**
- void * **system**
- [lt_interface_data](#) * **interface_data**
- int **flags**

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

- sst/core/libltdl/libltdl/lt__private.h

6.113 lt__interface_id Struct Reference

Public Attributes

- char * **id_string**
- [lt_dlhandle_interface](#) * **iface**

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

- sst/core/libltdl/ltidl.c

6.114 lt_dlinfo Struct Reference

Public Attributes

- char * **filename**
- char * **name**
- int **ref_count**
- unsigned int **is_resident**:1
- unsigned int **is_symglobal**:1
- unsigned int **is_symlocal**:1

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

- sst/core/libltdl/ltidl.h

6.115 It_dlsymlist Struct Reference

Public Attributes

- const char * **name**
- void * **address**

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

- sst/core/libltdl/ltl.h

6.116 It_dlvtable Struct Reference

Public Attributes

- const char * **name**
- const char * **sym_prefix**
- lt_module_open * **module_open**
- lt_module_close * **module_close**
- lt_find_sym * **find_sym**
- lt_dloader_init * **dloader_init**
- lt_dloader_exit * **dloader_exit**
- lt_user_data **dloader_data**
- lt_dloader_priority **priority**

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

- sst/core/libltdl/libltdl/lt_dloader.h

6.117 It_interface_data Struct Reference

Public Attributes

- lt_dinterface_id **key**
- void * **data**

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

- sst/core/libltdl/libltdl/lt__private.h

6.118 SST::RNG::MarsagliaRNG Class Reference

```
#include "sst/core/rng/marsaglia.h"
```

Inheritance diagram for SST::RNG::MarsagliaRNG:

Collaboration diagram for SST::RNG::MarsagliaRNG:

Public Member Functions

- [MarsagliaRNG](#) (unsigned int initial_z, unsigned int initial_w)
- [MarsagliaRNG](#) ()
- void [restart](#) (unsigned int new_z, unsigned int new_w)
- double [nextUniform](#) () override
- uint32_t [generateNextUInt32](#) () override
- uint64_t [generateNextUInt64](#) () override
- int64_t [generateNextInt64](#) () override
- int32_t [generateNextInt32](#) () override
- void [seed](#) (uint64_t newSeed)

6.118.1 Detailed Description

Implements a random number generator using the Marsaglia method. This method is computationally cheap and provides a reasonable distribution of random numbers. If you need additional strength in the random numbers you may want to consider the Mersenne RNG.

For more information see the [Multiply-with-carry Random Number Generator](#) article

at Wikipedia (<http://en.wikipedia.org/wiki/Multiply-with-carry>).

6.118.2 Constructor & Destructor Documentation

6.118.2.1 MarsagliaRNG::MarsagliaRNG (unsigned int *initial_z*, unsigned int *initial_w*)

Creates a new Marsaglia RNG using the initial seeds.

Parameters

in	<i>initial_z</i>	One of the random seed pairs
in	<i>initial_w</i>	One of the random seed pairs.

6.118.2.2 MarsagliaRNG::MarsagliaRNG ()

Creates a new Marsaglia RNG using random initial seeds (which are read from the system clock). Note that these will create variation between runs and between platforms.

6.118.3 Member Function Documentation

6.118.3.1 int32_t MarsagliaRNG::generateNextInt32 () [override],[virtual]

Generates the next number as a signed 32-bit integer.

Implements [SST::RNG::SSTRandom](#).

6.118.3.2 `int64_t MarsagliaRNG::generateNextInt64 () [override],[virtual]`

Generates the next number as a signed 64-bit integer.

Implements [SST::RNG::SSTRandom](#).

6.118.3.3 `uint32_t MarsagliaRNG::generateNextUInt32 () [override],[virtual]`

Generates the next random number as an unsigned 32-bit integer.

Implements [SST::RNG::SSTRandom](#).

6.118.3.4 `uint64_t MarsagliaRNG::generateNextUInt64 () [override],[virtual]`

Generates the next random number as an unsigned 64-bit integer.

Implements [SST::RNG::SSTRandom](#).

6.118.3.5 `double MarsagliaRNG::nextUniform () [override],[virtual]`

Generates the next random number as a double in the range 0 to 1.

Implements [SST::RNG::SSTRandom](#).

6.118.3.6 `void MarsagliaRNG::restart (unsigned int new_z, unsigned int new_w)`

Restart the random number generator with new seeds

Parameters

in	<i>new</i> ↔ _z	A new Z-seed
in	<i>new</i> ↔ _w	A new W-seed

6.118.3.7 `void MarsagliaRNG::seed (uint64_t newSeed)`

Seed the XOR RNG

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

- `sst/core/rng/marsaglia.h`
- `sst/core/rng/marsaglia.cc`

6.119 SST::Core::MemPool Class Reference

```
#include <mempool.h>
```

Public Member Functions

- [MemPool](#) (size_t elementSize, size_t initialSize=(2<< 20))
- void * [malloc](#) ()
- void [free](#) (void *ptr)
- uint64_t [getBytesMemUsed](#) ()
- uint64_t [getUndeletedEntries](#) ()
- size_t [getArenaSize](#) () const
- size_t [getElementSize](#) () const
- const std::list< uint8_t * > & [getArenas](#) ()

Public Attributes

- std::atomic< uint64_t > [numAlloc](#)
- std::atomic< uint64_t > [numFree](#)

6.119.1 Detailed Description

Simple Memory Pool class

6.119.2 Constructor & Destructor Documentation

6.119.2.1 SST::Core::MemPool::MemPool (size_t *elementSize*, size_t *initialSize* = (2<<20)) [inline]

Create a new Memory Pool.

Parameters

<i>elementSize</i>	- Size of each Element
<i>initialSize</i>	- Size of the memory pool (in bytes)

6.119.3 Member Function Documentation

6.119.3.1 void SST::Core::MemPool::free (void * *ptr*) [inline]

Return an element to the memory pool

6.119.3.2 uint64_t SST::Core::MemPool::getBytesMemUsed () [inline]

Approximates the current memory usage of the mempool. Some overheads are not taken into account.

6.119.3.3 `void* SST::Core::MemPool::malloc () [inline]`

Allocate a new element from the memory pool

6.119.4 Member Data Documentation

6.119.4.1 `std::atomic<uint64_t> SST::Core::MemPool::numAlloc`

Counter: Number of times elements have been allocated

6.119.4.2 `std::atomic<uint64_t> SST::Core::MemPool::numFree`

Counter: Number times elements have been freed

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

- `sst/core/mempool.h`

6.120 SST::RNG::MersenneRNG Class Reference

```
#include "sst/core/rng/mersenne.h"
```

Inheritance diagram for SST::RNG::MersenneRNG:

Collaboration diagram for SST::RNG::MersenneRNG:

Public Member Functions

- [MersenneRNG](#) (unsigned int [seed](#))
- [MersenneRNG](#) ()
- double [nextUniform](#) () override
- uint32_t [generateNextUInt32](#) () override
- uint64_t [generateNextUInt64](#) () override
- int64_t [generateNextInt64](#) () override
- int32_t [generateNextInt32](#) () override
- void [seed](#) (uint64_t newSeed)
- [~MersenneRNG](#) ()

6.120.1 Detailed Description

Implements a Mersenne-based RNG for use in the SST core or components. The Mersenne RNG provides a better "randomness" to the distribution of outputs but is computationally more expensive than the Marsaglia RNG.

6.120.2 Constructor & Destructor Documentation

6.120.2.1 `MersenneRNG::MersenneRNG (unsigned int seed)`

Create a new Mersenne RNG with a specified seed

Parameters

in	<i>seed</i>	The seed for this RNG
----	-------------	-----------------------

6.120.2.2 MersenneRNG::MersenneRNG ()

Creates a new Mersenne using a random seed which is obtained from the system clock. Note this will give different results on different platforms and between runs.

6.120.2.3 MersenneRNG::~~MersenneRNG ()

Destructor for Mersenne

6.120.3 Member Function Documentation

6.120.3.1 int32_t MersenneRNG::generateNextInt32 () [override],[virtual]

Generates the next random number as a signed 32-bit integer

Implements [SST::RNG::SSTRandom](#).

6.120.3.2 int64_t MersenneRNG::generateNextInt64 () [override],[virtual]

Generates the next random number as a signed 64-bit integer

Implements [SST::RNG::SSTRandom](#).

6.120.3.3 uint32_t MersenneRNG::generateNextUInt32 () [override],[virtual]

Generates the next random number as an unsigned 32-bit integer

Implements [SST::RNG::SSTRandom](#).

6.120.3.4 uint64_t MersenneRNG::generateNextUInt64 () [override],[virtual]

Generates the next random number as an unsigned 64-bit integer

Implements [SST::RNG::SSTRandom](#).

6.120.3.5 double MersenneRNG::nextUniform () [override],[virtual]

Generates the next random number as a double value between 0 and 1.

Implements [SST::RNG::SSTRandom](#).

6.120.3.6 void MersenneRNG::seed (uint64_t newSeed)

Seed the XOR RNG

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

- sst/core/rng/mersenne.h
- sst/core/rng/mersenne.cc

6.121 SST::ELI::MethodDetect< T > Struct Template Reference

Public Types

- using **type** = void

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

- sst/core/eli/elibase.h

6.122 SST::Module Class Reference

```
#include <module.h>
```

Inheritance diagram for SST::Module:

Public Member Functions

- **SST_ELI_DECLARE_CTORS** (ELI_CTOR(SST::Params &),) [Module\(\)](#)

6.122.1 Detailed Description

[Module](#) is a tag class used with the loadModule function.

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

- sst/core/module.h

6.123 ModuleLoaderPy_t Struct Reference

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

- sst/core/model/pymodel.cc

6.124 SST::Core::Serialization::need_delete_statics< T > Class Template Reference

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

- sst/core/serialization/statics.h

6.125 SST::Interfaces::SimpleNetwork::NetworkInspector Class Reference

```
#include <simpleNetwork.h>
```

Inheritance diagram for SST::Interfaces::SimpleNetwork::NetworkInspector:

Collaboration diagram for SST::Interfaces::SimpleNetwork::NetworkInspector:

Public Member Functions

- **NetworkInspector** ([Component](#) *parent)
- **NetworkInspector** (ComponentId_t id)
- virtual void **inspectNetworkData** ([Request](#) *req)=0
- virtual void **initialize** (std::string id)=0

Additional Inherited Members

6.125.1 Detailed Description

Class used to inspect network requests going through the network.

6.125.2 Member Function Documentation

6.125.2.1 virtual void SST::Interfaces::SimpleNetwork::NetworkInspector::initialize (std::string id) [pure virtual]

The ID uniquely identifies the component in which this subcomponent is instantiated. It does not uniquely define this particular [NetworkInspector](#), and all NetworkInspectors instantiated in the same component will get the same ID. If registering statistics, the ID is intended to be used as the subfield of the statistic.

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

- sst/core/interfaces/simpleNetwork.h

6.126 SST::NewRankSync Class Reference

Inheritance diagram for SST::NewRankSync:

Collaboration diagram for SST::NewRankSync:

Public Member Functions

- virtual [ActivityQueue](#) * [registerLink](#) (const [RankInfo](#) &to_rank, const [RankInfo](#) &from_rank, LinkId_t link_id, [Link](#) *link)=0
- virtual void [execute](#) (int thread)=0
- virtual void [exchangeLinkUntimedData](#) (int thread, std::atomic< int > &msg_count)=0
- virtual void [finalizeLinkConfigurations](#) ()=0
- virtual void [prepareForComplete](#) ()=0
- virtual SimTime_t [getNextSyncTime](#) ()
- [TimeConverter](#) * [getMaxPeriod](#) ()
- virtual uint64_t [getDataSize](#) () const =0

Protected Member Functions

- void [finalizeConfiguration](#) ([Link](#) *link)
- void [prepareForCompleteInt](#) ([Link](#) *link)
- void [sendUntimedData_sync](#) ([Link](#) *link, [Event](#) *data)

Protected Attributes

- SimTime_t [nextSyncTime](#)
- [TimeConverter](#) * [max_period](#)

6.126.1 Member Function Documentation

6.126.1.1 virtual [ActivityQueue](#)* SST::NewRankSync::registerLink (const [RankInfo](#) & to_rank, const [RankInfo](#) & from_rank, LinkId_t link_id, [Link](#) * link) [pure virtual]

Register a [Link](#) which this Sync Object is responsible for

Implemented in [SST::RankSyncParallelSkip](#), and [SST::RankSyncSerialSkip](#).

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

- sst/core/syncManager.h

6.127 SST::NewThreadSync Class Reference

Inheritance diagram for SST::NewThreadSync:

Collaboration diagram for SST::NewThreadSync:

Public Member Functions

- virtual void **before** ()=0
- virtual void **after** ()=0
- virtual void **execute** ()=0
- virtual void **processLinkUntimedData** ()=0
- virtual void **finalizeLinkConfigurations** ()=0
- virtual void **prepareForComplete** ()=0
- virtual SimTime_t **getNextSyncTime** ()
- void **setMaxPeriod** (TimeConverter *period)
- TimeConverter * **getMaxPeriod** ()
- virtual void **registerLink** (LinkId_t link_id, Link *link)=0
- virtual ActivityQueue * **getQueueForThread** (int tid)=0

Protected Member Functions

- void **finalizeConfiguration** (Link *link)
- void **prepareForCompleteInt** (Link *link)
- void **sendUntimedData_sync** (Link *link, Event *data)

Protected Attributes

- SimTime_t **nextSyncTime**
- TimeConverter * **max_period**

6.127.1 Member Function Documentation

6.127.1.1 virtual void SST::NewThreadSync::registerLink (LinkId_t *link_id*, Link * *link*) [pure virtual]

Register a [Link](#) which this Sync Object is responsible for

Implemented in [SST::ThreadSyncSimpleSkip](#).

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

- sst/core/syncManager.h

6.128 SST::ELI::NoValidConstructorsForDerivedType< NumValid > Struct Template Reference

Static Public Attributes

- static constexpr bool **atLeastOneValidCtor** = true

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

- sst/core/eli/elementbuilder.h

6.129 SST::ELI::NoValidConstructorsForDerivedType< 0 > Class Template Reference

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

- `sst/core/eli/elementbuilder.h`

6.130 SST::NullEvent Class Reference

```
#include <event.h>
```

Inheritance diagram for SST::NullEvent:

Collaboration diagram for SST::NullEvent:

Public Member Functions

- void `execute` (void) override
- virtual void `print` (const std::string &header, `Output` &out) const override

Additional Inherited Members

6.130.1 Detailed Description

Null `Event`. Does nothing.

6.130.2 Member Function Documentation

6.130.2.1 void SST::NullEvent::execute (void) `[override]`,`[virtual]`

Cause this event to fire

Reimplemented from `SST::Event`.

6.130.2.2 virtual void SST::NullEvent::print (const std::string & *header*, `Output` & *out*) const `[inline]`,
`[override]`,`[virtual]`

Virtual function to "pretty-print" this event. Should be implemented by subclasses.

Reimplemented from `SST::Event`.

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

- `sst/core/event.h`
- `sst/core/event.cc`

6.131 SST::Statistics::NullStatistic< T > Class Template Reference

```
#include <statnull.h>
```

Inheritance diagram for SST::Statistics::NullStatistic< T >:

Collaboration diagram for SST::Statistics::NullStatistic< T >:

Public Member Functions

- **SST_ELI_DECLARE_STATISTIC_TEMPLATE** (NullStatistic,"sst","NullStatistic", SST_ELI_ELEMENT_VERSION(1, 0, 0),"Null object it ignore all collections","SST::Statistic<T>") NullStatistic(BaseComponent *comp

Public Attributes

- const std::string & **statName**
- const std::string const std::string & **statSubId**
- const std::string const std::string Params & **statParam**: NullStatisticBase<T>(comp
- const std::string const std::string Params **statName**
- const std::string const std::string Params **statSubId**
- const std::string const std::string Params **statParam**

6.131.1 Detailed Description

```
template<class T>
class SST::Statistics::NullStatistic< T >
```

An empty statistic place holder.

Template Parameters

<i>T</i>	A template for holding the main data type of this statistic
----------	---

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

- sst/core/statapi/statnull.h

6.132 SST::Statistics::NullStatisticBase< T, B > Struct Template Reference

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

- sst/core/statapi/statnull.h

6.133 SST::Statistics::NullStatisticBase< std::tuple< Args... >, false > Struct Template Reference

Inheritance diagram for SST::Statistics::NullStatisticBase< std::tuple< Args... >, false >:

Collaboration diagram for SST::Statistics::NullStatisticBase< std::tuple< Args... >, false >:

Public Member Functions

- **NullStatisticBase** ([BaseComponent](#) *comp, const std::string &statName, const std::string &statSubId, [Params](#) &statParams)
- void **addData_impl** (Args...UNUSED(data)) override

Additional Inherited Members

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

- sst/core/statapi/statnull.h

6.134 SST::Statistics::NullStatisticBase< T, false > Struct Template Reference

Inheritance diagram for SST::Statistics::NullStatisticBase< T, false >:

Collaboration diagram for SST::Statistics::NullStatisticBase< T, false >:

Public Member Functions

- **NullStatisticBase** ([BaseComponent](#) *comp, const std::string &statName, const std::string &statSubId, [Params](#) &statParams)
- void **addData_impl** (T &&UNUSED(data)) override
- void **addData_impl** (const T &UNUSED(data)) override

Additional Inherited Members

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

- sst/core/statapi/statnull.h

6.135 SST::Statistics::NullStatisticBase< T, true > Struct Template Reference

Inheritance diagram for SST::Statistics::NullStatisticBase< T, true >:

Collaboration diagram for SST::Statistics::NullStatisticBase< T, true >:

Public Member Functions

- **NullStatisticBase** ([BaseComponent](#) *comp, const std::string &statName, const std::string &statSubId, [Params](#) &statParams)
- void **addData_impl** (T UNUSED(data)) override

Additional Inherited Members

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

- sst/core/statapi/statnull.h

6.136 SST::OneShot Class Reference

```
#include <oneshot.h>
```

Inheritance diagram for SST::OneShot:

Collaboration diagram for SST::OneShot:

Classes

- class [Handler](#)
- class [Handler< classT, void >](#)
- class [HandlerBase](#)

Public Member Functions

- [OneShot](#) ([TimeConverter](#) *timeDelay, int priority=ONESHOTPRIORITY)
- bool [isScheduled](#) ()
- void [registerHandler](#) ([OneShot::HandlerBase](#) *handler)
- void [print](#) (const std::string &header, [Output](#) &out) const override

Additional Inherited Members

6.136.1 Detailed Description

A [OneShot Event](#) class.

Calls callback functions (handlers) on a specified period

6.136.2 Constructor & Destructor Documentation

6.136.2.1 SST::OneShot::OneShot ([TimeConverter](#) * *timeDelay*, int *priority* = ONESHOTPRIORITY)

Create a new One Shot for a specified time that will callback the handler function. Note: [OneShot](#) cannot be canceled, and will always callback after the timedelay.

6.136.3 Member Function Documentation

6.136.3.1 bool SST::OneShot::isScheduled () [inline]

Is [OneShot](#) scheduled

6.136.3.2 void SST::OneShot::print (const std::string & *header*, *Output* & *out*) const [override],[virtual]

Print details about the [OneShot](#)

Reimplemented from [SST::Action](#).

6.136.3.3 void SST::OneShot::registerHandler (*OneShot::HandlerBase* * *handler*)

Add a handler to be called on this [OneShot Event](#)

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

- sst/core/oneshot.h
- sst/core/oneshot.cc

6.137 SST::Output Class Reference

```
#include <output.h>
```

Public Types

- enum [output_location_t](#) { [NONE](#), [STDOUT](#), [STDERR](#), [FILE](#) }

Public Member Functions

- [Output](#) (const std::string &prefix, uint32_t verbose_level, uint32_t verbose_mask, [output_location_t](#) location, std::string localoutputfilename="")
- [Output](#) ()
- void [init](#) (const std::string &prefix, uint32_t verbose_level, uint32_t verbose_mask, [output_location_t](#) location, std::string localoutputfilename="")
- void [output](#) (uint32_t line, const char *file, const char *func, const char *format,...) const __attribute__((format(printf
- void [if](#) (true==m_objInitialized &&NONE!=m_targetLoc)
- void [output](#) (const char *format,...) const __attribute__((format(printf
- void [if](#) (true==m_objInitialized &&NONE!=m_targetLoc)
- void [verbose](#) (uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) const __attribute__((format(printf
- void [if](#) (true==m_objInitialized &&NONE!=m_targetLoc)
- void [verbosePrefix](#) (const char *tempPrefix, uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) __attribute__((format(printf
- void [if](#) (true==m_objInitialized &&NONE!=m_targetLoc)

- void [debugPrefix](#) (const char *tempPrefix, uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) `__attribute__((format(printf`
- void [debug](#) (uint32_t line, const char *file, const char *func, uint32_t output_level, uint32_t output_bits, const char *format,...) const `__attribute__((format(printf`
- void [fatal](#) (uint32_t line, const char *file, const char *func, int exit_code, const char *format,...) const `__attribute__((format(printf`
- void [setPrefix](#) (const std::string &prefix)
- std::string [getPrefix](#) () const
- void [setVerboseMask](#) (uint32_t verbose_mask)
- uint32_t [getVerboseMask](#) () const
- void [setVerboseLevel](#) (uint32_t verbose_level)
- uint32_t [getVerboseLevel](#) () const
- void [setOutputLocation](#) ([output_location_t](#) location)
- [output_location_t](#) [getOutputLocation](#) () const
- void [flush](#) () const

Static Public Member Functions

- static void [setFileName](#) (const std::string &filename)
- static [Output](#) & [getDefaultObject](#) ()

Public Attributes

- void void **line**
- void **file**
- void **func**
- void **output_level**
- void **output_bits**
- void **format**
- void void **file**

Static Public Attributes

- static constexpr uint32_t **PrintAll** = std::numeric_limits<uint32_t>::max()

Friends

- class **TraceFunction**

6.137.1 Detailed Description

[Output](#) object provides consistent method for outputting data to stdout, stderr and/or sst debug file. All components should use this class to log any information.

6.137.2 Member Enumeration Documentation

6.137.2.1 enum SST::Output::output_location_t

Choice of output location

Enumerator

NONE No output
STDOUT Print to stdout
STDERR Print to stderr
FILE Print to a file

6.137.3 Constructor & Destructor Documentation

6.137.3.1 SST::Output::Output (const std::string & *prefix*, uint32_t *verbose_level*, uint32_t *verbose_mask*, output_location_t *location*, std::string *localoutputfilename* = " ")

Constructor. Set up output configuration.

Parameters

<i>prefix</i>	<p>Prefix to be prepended to all strings emitted by calls to debug(), verbose(), fatal() and possibly output(). NOTE: No space will be inserted between the prepended prefix string and the normal output string. Prefix can contain the following escape codes:</p> <ul style="list-style-type: none"> • @f Name of the file in which output call was made. • @l Line number in the file in which output call was made. • @p Name of the function from which output call was made. • @r MPI rank of the calling process. Will be empty if MPI_COMM_WORLD size is 1. • @R MPI rank of the calling process. Will be 0 if MPI_COMM_WORLD size is 1. • @i Thread Id of the calling process. Will be empty if number of threads is 1. • @I Thread Id of the calling process. Will be 0 if number of threads is 1. • @x Rank information of the calling process. Will be empty if number of MPI ranks and number of threads are both 1 Same as [@r:@i] • @X Rank information of the calling process. Will be [0.0] if number of MPI ranks and number of threads are both 1 Same as [@R:@I] • @t Simulation time. Will be the raw simulation cycle time retrieved from the SST Core.
<i>verbose_level</i>	<p>Debugging output level. Calls to debug(), verbose() and fatal() are only output if their output_level parameter is less than or equal to the verbose_level currently set for the object</p>
<i>verbose_mask</i>	<p>Bitmask of allowed message types for debug(), verbose() and fatal(). The Output object will only output the message if the set bits of the output_bits parameter are set in the verbose_mask of the object. It uses this logic: if (\simverbose_mask & output_bits == 0) then output is enabled.</p>

Parameters

<i>location</i>	Output location. Output will be directed to STDOUT, STDERR, FILE, or NONE. If FILE output is selected, the output will be directed to the file defined by the <code>--debug</code> runtime parameter, or to the file 'sst_output' if the <code>--debug</code> parameter is not defined. If the size of MPI_COMM_WORLD is > 1, then the rank process will be appended to the file name.
<i>localoutputfilename.</i>	Send the output of this class to the file identified in localoutputfilename instead of the of the normal output file set by the run time parameter <code>--debug-file</code> . location parameter must be set to FILE. This parameter is intended for special case debug purposes only.

6.137.3.2 SST::Output::Output ()

Default Constructor. User must call [init\(\)](#) to properly initialize obj. Until [init\(\)](#) is called, no output will occur.

6.137.4 Member Function Documentation

6.137.4.1 void SST::Output::debug (uint32_t *line*, const char * *file*, const char * *func*, uint32_t *output_level*, uint32_t *output_bits*, const char * *format*, ...) const

[Output](#) the debug message with formatting as specified by the format parameter. [Output](#) will only occur if specified output_level and output_bits meet criteria defined by object. The output will be prepended with the expanded prefix set in the object. NOTE: Debug outputs will only occur if the **SST_DEBUG_OUTPUT** is defined. this define can be set in source code or by setting the `--enable-debug` option during SST configuration.

Parameters

<i>line</i>	Line number of calling function (use CALL_INFO macro)
<i>file</i>	File name calling function (use CALL_INFO macro)
<i>func</i>	Function name calling function (use CALL_INFO macro)
<i>output_level</i>	For output to occur, output_level must be less than or equal to verbose_level set in object
<i>output_bits</i>	The Output object will only output the message if the set bits of the output_bits parameter are set in the verbose_mask of the object. It uses this logic: if (\sim verbose_mask & output_bits == 0) then output is enabled.
<i>format</i>	Format string. All valid formats for printf are available.
<i>...</i>	Arguments for format.

6.137.4.2 void SST::Output::debugPrefix (const char * *tempPrefix*, uint32_t *line*, const char * *file*, const char * *func*, uint32_t *output_level*, uint32_t *output_bits*, const char * *format*, ...)

[Output](#) the debug message with formatting as specified by the format parameter. [Output](#) will only occur if specified output_level and output_bits meet criteria defined by object. The output will be prepended with the expanded prefix set in the object.

Parameters

<i>tempPrefix</i>	For just this call use this prefix
<i>line</i>	Line number of calling function (use CALL_INFO macro)

Parameters

<i>file</i>	File name calling function (use CALL_INFO macro)
<i>func</i>	Function name calling function (use CALL_INFO macro)
<i>output_level</i>	For output to occur, output_level must be less than or equal to verbose_level set in object
<i>output_bits</i>	The Output object will only output the message if the set bits of the output_bits parameter are set in the verbose_mask of the object. It uses this logic: if (\sim verbose_mask & output_bits == 0) then output is enabled.
<i>format</i>	Format string. All valid formats for printf are available.
...	Arguments for format.

6.137.4.3 void SST::Output::fatal (uint32_t *line*, const char * *file*, const char * *func*, int *exit_code*, const char * *format*, ...) const

[Output](#) the fatal message with formatting as specified by the format parameter. Message will be sent to the output location and to stderr. The output will be prepended with the expanded prefix set in the object. NOTE: [fatal\(\)](#) will call MPI_Abort(exit_code) to terminate simulation.

Parameters

<i>line</i>	Line number of calling function (use CALL_INFO macro)
<i>file</i>	File name calling function (use CALL_INFO macro)
<i>func</i>	Function name calling function (use CALL_INFO macro)
<i>exit_code</i>	The exit code used for termination of simulation. will be passed to MPI_Abort()
<i>format</i>	Format string. All valid formats for printf are available.
...	Arguments for format.

6.137.4.4 void SST::Output::flush () const [inline]

This method allows for the manual flushing of the output.

6.137.4.5 Output::output_location_t SST::Output::getOutputLocation () const

Returns object output location

6.137.4.6 std::string SST::Output::getPrefix () const

Returns object prefix

6.137.4.7 uint32_t SST::Output::getVerboseLevel () const

Returns object verbose level

6.137.4.8 uint32_t SST::Output::getVerboseMask () const

Returns object verbose mask

6.137.4.9 void SST::Output::init (const std::string & prefix, uint32_t verbose_level, uint32_t verbose_mask, output_location_t location, std::string localoutputfilename = " ")

Initialize the object after construction

Parameters

<i>prefix</i>	<p>Prefix to be prepended to all strings emitted by calls to debug(), verbose(), fatal() and possibly output(). NOTE: No space will be inserted between the prepended prefix string and the normal output string. Prefix can contain the following escape codes:</p> <ul style="list-style-type: none"> • @f Name of the file in which output call was made. • @l Line number in the file in which output call was made. • @p Name of the function from which output call was made. • @r MPI rank of the calling process. Will be empty if MPI_COMM_WORLD size is 1. • @R MPI rank of the calling process. Will be 0 if MPI_COMM_WORLD size is 1. • @i Thread Id of the calling process. Will be empty if number of threads is 1. • @I Thread Id of the calling process. Will be 0 if number of threads is 1. • @x Rank information of the calling process. Will be empty if number of MPI ranks and number of threads are both 1 Same as [@r:@i] • @X Rank information of the calling process. Will be [0.0] if number of MPI ranks and number of threads are both 1 Same as [@R:@I] • @t Simulation time. Will be the raw simulation cycle time retrieved from the SST Core.
<i>verbose_level</i>	Debugging output level. Calls to debug() , verbose() and fatal() are only output if their output_level parameter is less than or equal to the verbose_level currently set for the object
<i>verbose_mask</i>	Bitmask of allowed message types for debug() , verbose() and fatal() . The Output object will only output the message if the set bits of the output_bits parameter are set in the verbose_mask of the object. It uses this logic: if ($\sim\text{verbose_mask} \& \text{output_bits} == 0$) then output is enabled.
<i>location</i>	Output location. Output will be directed to STDOUT, STDERR, FILE, or NONE. If FILE output is selected, the output will be directed to the file defined by the <code>--debug</code> runtime parameter, or to the file 'sst_output' if the <code>--debug</code> parameter is not defined. If the size of MPI_COMM_WORLD is > 1 , then the rank process will be appended to the file name.
<i>localoutputfilename.</i>	Send the output of this class to the file identified in localoutputfilename instead of the of the normal output file set by the run time parameter <code>--debug-file</code> . location parameter must be set to FILE. This parameter is intended for special case debug purposes only.

6.137.4.10 `void SST::Output::output (uint32_t line, const char * file, const char * func, const char * format, ...) const`

[Output](#) the message with formatting as specified by the format parameter. The output will be prepended with the expanded prefix set in the object.

Parameters

<i>line</i>	Line number of calling function (use CALL_INFO macro)
<i>file</i>	File name calling function (use CALL_INFO macro)
<i>func</i>	Function name calling function (use CALL_INFO macro)
<i>format</i>	Format string. All valid formats for printf are available.
...	Argument strings for format.

6.137.4.11 `void SST::Output::output (const char * format, ...) const`

[Output](#) the message with formatting as specified by the format parameter.

Parameters

<i>format</i>	Format string. All valid formats for printf are available.
...	Arguments for format.

6.137.4.12 `void SST::Output::setFileName (const std::string & filename) [static]`

This method sets the static filename used by SST. It can only be called once, and is automatically called by the SST Core. No components should call this method.

6.137.4.13 `void SST::Output::setOutputLocation (output_location_t location)`

Sets object output location

Parameters

<i>location</i>	Output location. Output will be directed to STDOUT, STDERR, FILE, or NONE. If FILE output is selected, the output will be directed to the file defined by the <code>--debug</code> runtime parameter, or to the file 'sst_output' if the <code>--debug</code> parameter is not defined. If the size of MPI_COMM_WORLD is > 1, then the rank process will be appended to the file name.
-----------------	--

6.137.4.14 `void SST::Output::setPrefix (const std::string & prefix)`

Sets object prefix

Parameters

<i>prefix</i>	<p>Prefix to be prepended to all strings emitted by calls to debug(), verbose(), fatal() and possibly output(). NOTE: No space will be inserted between the prepended prefix string and the normal output string. Prefix can contain the following escape codes:</p> <ul style="list-style-type: none"> • @f Name of the file in which output call was made. • @l Line number in the file in which output call was made. • @p Name of the function from which output call was made. • @r MPI rank of the calling process. Will be empty if MPI_COMM_WORLD size is 1. • @R MPI rank of the calling process. Will be 0 if MPI_COMM_WORLD size is 1. • @i Thread Id of the calling process. Will be empty if number of threads is 1. • @I Thread Id of the calling process. Will be 0 if number of threads is 1. • @x Rank information of the calling process. Will be empty if number of MPI ranks and number of threads are both 1 Same as [@r:@i] • @X Rank information of the calling process. Will be [0.0] if number of MPI ranks and number of threads are both 1 Same as [@R:@I] • @t Simulation time. Will be the raw simulation cycle time retrieved from the SST Core.
---------------	--

6.137.4.15 void SST::Output::setVerboseLevel (uint32_t *verbose_level*)

Sets object verbose level

Parameters

<i>verbose_level</i>	Debugging output level. Calls to debug() , verbose() and fatal() are only output if their output_level parameter is less than or equal to the verbose_level currently set for the object
----------------------	--

6.137.4.16 void SST::Output::setVerboseMask (uint32_t *verbose_mask*)

Sets object verbose mask

Parameters

<i>verbose_mask</i>	Bitmask of allowed message types for debug() , verbose() and fatal() . The Output object will only output the message if the set bits of the output_bits parameter are set in the verbose_mask of the object. It uses this logic: if (\sim verbose_mask & output_bits == 0) then output is enabled.
---------------------	--

6.137.4.17 void SST::Output::verbose (uint32_t *line*, const char * *file*, const char * *func*, uint32_t *output_level*, uint32_t *output_bits*, const char * *format*, ...) const

[Output](#) the verbose message with formatting as specified by the format parameter. [Output](#) will only occur if specified output_level and output_bits meet criteria defined by object. The output will be prepended with the expanded prefix

set in the object.

Parameters

<i>line</i>	Line number of calling function (use CALL_INFO macro)
<i>file</i>	File name calling function (use CALL_INFO macro)
<i>func</i>	Function name calling function (use CALL_INFO macro)
<i>output_level</i>	For output to occur, output_level must be less than or equal to verbose_level set in object
<i>output_bits</i>	The Output object will only output the message if the set bits of the output_bits parameter are set in the verbose_mask of the object. It uses this logic: if (\sim verbose_mask & output_bits == 0) then output is enabled.
<i>format</i>	Format string. All valid formats for printf are available.
...	Arguments for format.

6.137.4.18 void SST::Output::verbosePrefix (const char * *tempPrefix*, uint32_t *line*, const char * *file*, const char * *func*, uint32_t *output_level*, uint32_t *output_bits*, const char * *format*, ...)

[Output](#) the verbose message with formatting as specified by the format parameter. [Output](#) will only occur if specified output_level and output_bits meet criteria defined by object. The output will be prepended with the expanded prefix set in the object.

Parameters

<i>tempPrefix</i>	For just this call use this prefix
<i>line</i>	Line number of calling function (use CALL_INFO macro)
<i>file</i>	File name calling function (use CALL_INFO macro)
<i>func</i>	Function name calling function (use CALL_INFO macro)
<i>output_level</i>	For output to occur, output_level must be less than or equal to verbose_level set in object
<i>output_bits</i>	The Output object will only output the message if the set bits of the output_bits parameter are set in the verbose_mask of the object. It uses this logic: if (\sim verbose_mask & output_bits == 0) then output is enabled.
<i>format</i>	Format string. All valid formats for printf are available.
...	Arguments for format.

6.137.5 Member Data Documentation

6.137.5.1 void void SST::Output::file

Initial value:

```
{
```

```
(void) line
```

6.137.5.2 void void SST::Output::line

Initial value:

```
{
```

```
(void) tempPrefix
```

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

- sst/core/output.h
- sst/core/output.cc

6.138 OverallOutputter Class Reference

Public Member Functions

- void **outputHumanReadable** (std::ostream &os)
- void **outputXML** ()

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

- sst/core/sstinfo.cc

6.139 SST::Params Class Reference

```
#include <params.h>
```

Inheritance diagram for SST::Params:

Collaboration diagram for SST::Params:

Public Types

- typedef std::string [key_type](#)
- typedef std::set< [key_type](#), KeyCompare > [KeySet_t](#)

Public Member Functions

- bool [enableVerify](#) (bool enable)
- size_t [size](#) () const
- bool [empty](#) () const
- [Params](#) ()
- [Params](#) (const [Params](#) &old)
- [Params](#) & [operator=](#) (const [Params](#) &old)
Assignment operator.
- void [clear](#) ()
- size_t [count](#) (const [key_type](#) &k)
Finds the number of elements with given key.
- template<class T >
std::enable_if< not std::is_same< std::string, T >::value, T >::type [find](#) (const std::string &k, T default_value, bool &found) const
- template<class T >
T [find](#) (const std::string &k, std::string default_value, bool &found) const
- template<class T >
std::enable_if< std::is_same< bool, T >::value, T >::type [find](#) (const std::string &k, const char *default_value, bool &found) const
- template<class T >
T [find](#) (const std::string &k, T default_value) const
- template<class T >
T [find](#) (const std::string &k, std::string default_value) const
- template<class T >
std::enable_if< std::is_same< bool, T >::value, T >::type [find](#) (const std::string &k, const char *default_value) const
- template<class T >
T [find](#) (const std::string &k) const
- template<class T >
std::enable_if< not std::is_same< bool, T >::value, T >::type [find](#) (const std::string &k, bool &found) const
- template<class T >
void [find_array](#) (const [key_type](#) &k, std::vector< T > &vec) const
- void [print_all_params](#) (std::ostream &os, std::string prefix="") const
- void [print_all_params](#) ([Output](#) &out, std::string prefix="") const
- void [insert](#) (std::string key, std::string value, bool overwrite=true)
- void [insert](#) (const [Params](#) ¶ms)
- std::set< std::string > [getKeys](#) () const
- [Params](#) [find_prefix_params](#) (const std::string &prefix) const
- [Params](#) [find_scoped_params](#) (const std::string &scope, const char *delims="..") const
- bool [contains](#) (const [key_type](#) &k)
- void [pushAllowedKeys](#) (const [KeySet_t](#) &keys)
- void [popAllowedKeys](#) ()
- void [verifyParam](#) (const [key_type](#) &k) const
- void [serialize_order](#) ([SST::Core::Serialization::serializer](#) &ser) override
- uint32_t [getKey](#) (const std::string &str) const
- uint32_t [getKey](#) (const std::string &str)
- friend [int::main](#) (int argc, char *argv[])

Static Public Member Functions

- static void [enableVerify](#) ()
- static const std::string & [getParamName](#) (uint32_t id)

Public Attributes

- ImplementSerializable([SST::Params](#)) private std::vector< [KeySet_t](#) > **allowedKeys**
- bool **verify_enabled**

Static Public Attributes

- static bool **g_verify_enabled** = false
- static std::map< std::string, uint32_t > **keyMap**
- static std::vector< std::string > **keyMapReverse**
- static [SST::Core::ThreadSafe::Spinlock](#) **keyLock**
- static uint32_t **nextKeyID**

Additional Inherited Members

6.139.1 Detailed Description

Parameter store.

Stores key-value pairs as std::strings and provides a templated find method for finding values and converting them to arbitrary types (

See also

[find\(\)](#)).

6.139.2 Member Typedef Documentation

6.139.2.1 typedef std::string SST::Params::key_type

Type of key (string)

6.139.2.2 typedef std::set<key_type, KeyCompare> SST::Params::KeySet_t

Type of a set of keys

6.139.3 Constructor & Destructor Documentation

6.139.3.1 SST::Params::Params ()

Create a new, empty [Params](#)

6.139.3.2 SST::Params::Params (const Params & old)

Create a copy of a [Params](#) object

6.139.4 Member Function Documentation

6.139.4.1 void SST::Params::clear ()

Erases all elements.

6.139.4.2 bool SST::Params::contains (const key_type & k)

Parameters

<i>k</i>	Key to search for
----------	-------------------

Returns

True if the params contains the key, false otherwise

6.139.4.3 size_t SST::Params::count (const key_type & k)

Finds the number of elements with given key.

Parameters

<i>k</i>	Key of (key, value) pairs to be located.
----------	--

Returns

Number of elements with specified key (either 1 or 0).

6.139.4.4 bool SST::Params::empty () const

Returns true if the [Params](#) is empty. (Thus begin() would equal end().)

6.139.4.5 bool SST::Params::enableVerify (bool enable) [inline]

Enable or disable parameter verification on an instance of [Params](#). Useful when generating a new set of [Params](#) to pass off to a module.

6.139.4.6 static void SST::Params::enableVerify () [inline],[static]

Enable, on a global scale, parameter verification. Used after construction of the config graph so that warnings are not generated during construction.

6.139.4.7 `template<class T > std::enable_if<not std::is_same<std::string,T>::value, T>::type SST::Params::find (const std::string & k, T default_value, bool & found) const [inline]`

Find a Parameter value in the set, and return its value as a type T. Type T must be either a basic numeric type (including bool) , a std::string, or a class that has a constructor with a std::string as its only parameter. This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>default_value</i>	- Default value to return if parameter isn't found
<i>found</i>	- set to true if the the parameter was found

Exceptions

<i>std::invalid_argument</i>	If value in (key, value) can't be converted to type T, an invalid_argument exception is thrown.
------------------------------	---

6.139.4.8 `template<class T > T SST::Params::find (const std::string & k, std::string default_value, bool & found) const [inline]`

Find a Parameter value in the set, and return its value as a type T. Type T must be either a basic numeric type (including bool) , a std::string, or a class that has a constructor with a std::string as its only parameter. This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>default_value</i>	- Default value to return if parameter isn't found, specified as a string
<i>found</i>	- set to true if the the parameter was found

6.139.4.9 `template<class T > std::enable_if<std::is_same<bool,T>::value, T>::type SST::Params::find (const std::string & k, const char * default_value, bool & found) const [inline]`

Find a Parameter value in the set, and return its value as a type T. This version of find is only enabled for bool. This is required because a string literal will be preferentially cast to a bool rather than a string. This ensures that find<bool> works correctly for string literals. This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>default_value</i>	- Default value to return if parameter isn't found, specified as a string literal

6.139.4.10 `template<class T > T SST::Params::find (const std::string & k, T default_value) const [inline]`

Find a Parameter value in the set, and return its value as a type T. Type T must be either a basic numeric type (including bool), a std::string, or a class that has a constructor with a std::string as its only parameter. This class

uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>default_value</i>	- Default value to return if parameter isn't found

6.139.4.11 `template<class T> T SST::Params::find (const std::string & k, std::string default_value) const` `[inline]`

Find a Parameter value in the set, and return its value as a type T. Type T must be either a basic numeric type (including bool) , a std::string, or a class that has a constructor with a std::string as its only parameter. This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>default_value</i>	- Default value to return if parameter isn't found, specified as a string

6.139.4.12 `template<class T> std::enable_if<std::is_same<bool,T>::value, T>::type SST::Params::find (const std::string & k, const char * default_value) const` `[inline]`

Find a Parameter value in the set, and return its value as a type T. This version of find is only enabled for bool. This is required because a string literal will be preferentially cast to a bool rather than a string. This ensures that find<bool> works correctly for string literals. This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>default_value</i>	- Default value to return if parameter isn't found, specified as a string literal

6.139.4.13 `template<class T> T SST::Params::find (const std::string & k) const` `[inline]`

Find a Parameter value in the set, and return its value as a type T. Type T must be either a basic numeric type (including bool) , a std::string, or a class that has a constructor with a std::string as its only parameter. This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
----------	------------------

6.139.4.14 `template<class T> std::enable_if<not std::is_same<bool, T>::value, T>::type SST::Params::find (const std::string & k, bool & found) const` `[inline]`

Find a Parameter value in the set, and return its value as a type T. Type T must be either a basic numeric type , a std::string, or a class that has a constructor with a std::string as its only parameter. This version of find is not enabled

for bool as it conflicts with find<bool>(string key, bool default_value). This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>found</i>	- set to true if the the parameter was found

```
6.139.4.15  template<class T > void SST::Params::find_array ( const key_type & k, std::vector< T > & vec ) const
            [inline]
```

Find a Parameter value in the set, and return its value as a vector of T's. The array will be appended to the end of the vector. Type T must be either a basic numeric type (including bool) , a std::string, or a class that has a constructor with a std::string as its only parameter. This class uses SST::Core::from_string to do the conversion.

Parameters

<i>k</i>	- Parameter name
<i>vec</i>	- vector to append array items to

```
6.139.4.16  Params SST::Params::find_prefix_params ( const std::string & prefix ) const
```

Returns a new parameter object with parameters that match the specified prefix.

```
6.139.4.17  const std::string & SST::Params::getParamName ( uint32_t id )  [static]
```

Given a Parameter Key ID, return the Name of the matching parameter

Parameters

<i>id</i>	Key ID to look up
-----------	-------------------

Returns

String name of the parameter

```
6.139.4.18  void SST::Params::insert ( std::string key, std::string value, bool overwrite = true )
```

Add a key value pair into the param object.

```
6.139.4.19  Params & SST::Params::operator= ( const Params & old )
```

Assignment operator.

Parameters

<i>old</i>	Param to be copied
------------	--------------------

All the elements of *old* are copied,

6.139.4.20 void SST::Params::popAllowedKeys ()

Removes the most recent set of keys considered allowed

6.139.4.21 void SST::Params::print_all_params (std::ostream & *os*, std::string *prefix* = " ") const

Print all key/value parameter pairs to specified ostream

6.139.4.22 void SST::Params::pushAllowedKeys (const KeySet_t & *keys*)

Parameters

<i>keys</i>	Set of keys to consider valid to add to the stack of legal keys
-------------	---

6.139.4.23 size_t SST::Params::size () const

Returns the size of the [Params](#).

6.139.4.24 void SST::Params::verifyParam (const key_type & *k*) const

Parameters

<i>k</i>	Key to check for validity
----------	---------------------------

Returns

True if the key is considered allowed

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

- sst/core/params.h
- sst/core/params.cc

6.140 SST::PartitionComponent Class Reference

Collaboration diagram for SST::PartitionComponent:

Public Member Functions

- **PartitionComponent** (const [ConfigComponent](#) &cc)
- **PartitionComponent** (LinkId_t id)
- void **print** (std::ostream &os, const [PartitionGraph](#) *graph) const
- ComponentId_t **key** () const

Public Attributes

- ComponentId_t **id**
- float **weight**
- [RankInfo](#) **rank**
- LinkIdMap_t **links**
- [ComponentIdMap_t](#) **group**

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

- sst/core/configGraph.h
- sst/core/configGraph.cc

6.141 SST::PartitionGraph Class Reference

Public Member Functions

- void [print](#) (std::ostream &os) const
- [PartitionComponentMap_t](#) & **getComponentMap** ()
- [PartitionLinkMap_t](#) & **getLinkMap** ()
- const [PartitionLink](#) & **getLink** (LinkId_t id) const
- size_t **getNumComponents** ()

6.141.1 Member Function Documentation

6.141.1.1 void SST::PartitionGraph::print (std::ostream & os) const [inline]

Print the configuration graph

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

- sst/core/configGraph.h

6.142 SST::PartitionLink Class Reference

Public Member Functions

- **PartitionLink** (const [ConfigLink](#) &cl)
- LinkId_t **key** () const
- SimTime_t [getMinLatency](#) () const
- void [print](#) (std::ostream &os) const

Public Attributes

- LinkId_t **id**
- ComponentId_t **component** [2]
- SimTime_t **latency** [2]
- bool **no_cut**

6.142.1 Member Function Documentation

6.142.1.1 SimTime_t SST::PartitionLink::getMinLatency () const [inline]

Return the minimum latency of this link (from both sides)

6.142.1.2 void SST::PartitionLink::print (std::ostream & os) const [inline]

Print the [Link](#) information

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

- sst/core/configGraph.h

6.143 SST::PollingLinkQueue Class Reference

```
#include <pollingLinkQueue.h>
```

Inheritance diagram for SST::PollingLinkQueue:

Collaboration diagram for SST::PollingLinkQueue:

Public Member Functions

- bool [empty](#) () override
- int [size](#) () override
- void [insert](#) (Activity *activity) override
- Activity * [pop](#) () override
- Activity * [front](#) () override

6.143.1 Detailed Description

A link queue which is used for polling only.

6.143.2 Member Function Documentation

6.143.2.1 bool SST::PollingLinkQueue::empty () [override],[virtual]

Returns true if the queue is empty

Implements [SST::ActivityQueue](#).

6.143.2.2 `Activity * SST::PollingLinkQueue::front ()` [override],[virtual]

Returns the next activity

Implements [SST::ActivityQueue](#).

6.143.2.3 `void SST::PollingLinkQueue::insert (Activity * activity)` [override],[virtual]

Insert a new activity into the queue

Implements [SST::ActivityQueue](#).

6.143.2.4 `Activity * SST::PollingLinkQueue::pop ()` [override],[virtual]

Remove and return the next activity

Implements [SST::ActivityQueue](#).

6.143.2.5 `int SST::PollingLinkQueue::size ()` [override],[virtual]

Returns the number of activities in the queue

Implements [SST::ActivityQueue](#).

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

- sst/core/pollingLinkQueue.h
- sst/core/pollingLinkQueue.cc

6.144 SST::Activity::pq_less_time_priority Class Reference

```
#include <activity.h>
```

Public Member Functions

- `bool operator() (const Activity *lhs, const Activity *rhs) const`
- `bool operator() (const Activity &lhs, const Activity &rhs) const`

6.144.1 Detailed Description

To use with STL priority queues, that order in reverse.

6.144.2 Member Function Documentation

6.144.2.1 `bool SST::Activity::pq_less_time_priority::operator() (const Activity * lhs, const Activity * rhs) const`
`[inline]`

Compare based off pointers

6.144.2.2 `bool SST::Activity::pq_less_time_priority::operator() (const Activity & lhs, const Activity & rhs) const`
`[inline]`

Compare based off references

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

- sst/core/activity.h

6.145 SST::Activity::pq_less_time_priority_order Class Reference

```
#include <activity.h>
```

Public Member Functions

- `bool operator() (const Activity *lhs, const Activity *rhs) const`
- `bool operator() (const Activity &lhs, const Activity &rhs) const`

6.145.1 Detailed Description

To use with STL priority queues, that order in reverse.

6.145.2 Member Function Documentation

6.145.2.1 `bool SST::Activity::pq_less_time_priority_order::operator() (const Activity * lhs, const Activity * rhs) const`
`[inline]`

Compare based off pointers

6.145.2.2 `bool SST::Activity::pq_less_time_priority_order::operator() (const Activity & lhs, const Activity & rhs) const`
`[inline]`

Compare based off references

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

- sst/core/activity.h

6.146 SST::ELI::ProvidesCategory Class Reference

Public Member Functions

- uint32_t **category** () const
- void **toString** (std::ostream &UNUSED(os)) const
- template<class XMLNode >
void **outputXML** (XMLNode *UNUSED(node))

Static Public Member Functions

- static const char * **categoryName** (int cat)

Protected Member Functions

- template<class T >
ProvidesCategory (T *UNUSED(t))

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

- sst/core/eli/categoryInfo.h

6.147 SST::ELI::ProvidesDefaultInfo Class Reference

Public Member Functions

- const std::string & **getLibrary** () const
- const std::string & **getDescription** () const
- const std::string & **getName** () const
- const std::vector< int > & **getVersion** () const
- const std::string & **getCompileFile** () const
- const std::string & **getCompileDate** () const
- const std::vector< int > & **getELICompiledVersion** () const
- std::string **getELIVersionString** () const
- void **toString** (std::ostream &os) const
- template<class XMLNode >
void **outputXML** (XMLNode *node) const
- template<class T >
ProvidesDefaultInfo (const std::string &lib, const std::string &name, T *UNUSED(t))

Protected Member Functions

- template<class T >
ProvidesDefaultInfo (T *t)

Friends

- class **ModuleDocOldEli**

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

- sst/core/eli/defaultInfo.h
- sst/core/eli/elementinfo.cc

6.148 SST::ELI::ProvidesInterface Class Reference

Public Member Functions

- const std::string & **getInterface** () const
- void **toString** (std::ostream &os) const
- template<class XMLNode >
void **outputXML** (XMLNode *node) const

Protected Member Functions

- template<class T >
ProvidesInterface (T *UNUSED(t))

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

- sst/core/eli/interfaceInfo.h

6.149 SST::ELI::ProvidesParams Class Reference

Public Member Functions

- const std::vector< [ElementInfoParam](#) > & **getValidParams** () const
- void **toString** (std::ostream &os) const
- template<class XMLNode >
void **outputXML** (XMLNode *node) const
- const [Params::KeySet_t](#) & **getParamNames** () const

Protected Member Functions

- template<class T >
ProvidesParams (T *UNUSED(t))

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

- sst/core/eli/paramsInfo.h
- sst/core/eli/elementinfo.cc

6.150 SST::ELI::ProvidesPorts Class Reference

Public Member Functions

- `const std::vector< std::string > & getPortnames ()`
- `const std::vector< ElementInfoPort2 > & getValidPorts () const`
- `void toString (std::ostream &os) const`
- `template<class XMLNode >`
`void outputXML (XMLNode *node) const`

Protected Member Functions

- `template<class T >`
`ProvidesPorts (T *UNUSED(t))`

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

- `sst/core/eli/portsInfo.h`
- `sst/core/eli/elementinfo.cc`

6.151 SST::ELI::ProvidesStats Class Reference

Public Member Functions

- `const std::vector< ElementInfoStatistic > & getValidStats () const`
- `const std::vector< std::string > & getStatnames () const`
- `void toString (std::ostream &os) const`
- `template<class XMLNode >`
`void outputXML (XMLNode *node) const`

Protected Member Functions

- `template<class T >`
`ProvidesStats (T *UNUSED(t))`

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

- `sst/core/eli/statsInfo.h`
- `sst/core/eli/elementinfo.cc`

6.152 SST::ELI::ProvidesSubComponentSlots Class Reference

Public Member Functions

- `const std::vector< ElementInfoSubComponentSlot > & getSubComponentSlots () const`
- `void toString (std::ostream &os) const`
- `template<class XMLNode >`
`void outputXML (XMLNode *node) const`

Protected Member Functions

- `template<class T >`
ProvidesSubComponentSlots (T *UNUSED(t))

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

- `sst/core/eli/subcompSlotInfo.h`
- `sst/core/eli/elementinfo.cc`

6.153 PyComponent Struct Reference

Inheritance diagram for PyComponent:

Collaboration diagram for PyComponent:

Public Member Functions

- **PyComponent** ([ComponentPy_t](#) *pobj)
- `const char *` **getName** () `const` override
- `ConfigComponent *` **getComp** () `override`
- [PyComponent](#) * **getBaseObj** () `override`
- `int` **compare** ([ComponentHolder](#) *other) `override`

Public Attributes

- `ComponentId_t` **id**
- `uint16_t` **subCompId**

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

- `sst/core/model/pymodel_comp.h`
- `sst/core/model/pymodel_comp.cc`

6.154 PySubComponent Struct Reference

Inheritance diagram for PySubComponent:

Collaboration diagram for PySubComponent:

Public Member Functions

- **PySubComponent** ([ComponentPy_t](#) *pobj)
- `const char *` **getName** () `const` override
- `ConfigComponent *` **getComp** () `override`
- [PyComponent](#) * **getBaseObj** () `override`
- `int` **compare** ([ComponentHolder](#) *other) `override`
- `int` **getSlot** () `const`

Public Attributes

- [ComponentHolder](#) * **parent**
- int **slot**

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

- sst/core/model/pymodel_comp.h
- sst/core/model/pymodel_comp.cc

6.155 SST::Core::PythonConfigGraphOutput Class Reference

Inheritance diagram for SST::Core::PythonConfigGraphOutput:

Collaboration diagram for SST::Core::PythonConfigGraphOutput:

Public Member Functions

- **PythonConfigGraphOutput** (const char *path)
- virtual void **generate** (const [Config](#) *cfg, [ConfigGraph](#) *graph) override

Protected Member Functions

- [ConfigGraph](#) * **getGraph** ()
- void **generateParams** (const [Params](#) ¶ms)
- void **generateCommonComponent** (const char *objName, const [ConfigComponent](#) &comp)
- void **generateSubComponent** (const char *owner, const [ConfigComponent](#) &comp)
- void **generateComponent** (const [ConfigComponent](#) &comp)
- void **generateStatGroup** (const [ConfigGraph](#) *graph, const [ConfigStatGroup](#) &grp)
- const std::string & **getLinkObject** (LinkId_t id)
- char * **makePythonSafeWithPrefix** (const std::string name, const std::string prefix) const
- void **makeBufferPythonSafe** (char *buffer) const
- char * **makeEscapeSafe** (const std::string input) const
- bool **strncmp** (const char *a, const char *b, const size_t n) const
- bool **isMultiLine** (const char *check) const
- bool **isMultiLine** (const std::string check) const

Additional Inherited Members

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

- sst/core/cfgoutput/pythonConfigOutput.h
- sst/core/cfgoutput/pythonConfigOutput.cc

6.156 SST::RankInfo Class Reference

Inheritance diagram for SST::RankInfo:

Collaboration diagram for SST::RankInfo:

Public Member Functions

- **RankInfo** (uint32_t rank, uint32_t thread)
- bool **isAssigned** () const
- bool **inRange** (const [RankInfo](#) &other) const
- bool **operator==** (const [RankInfo](#) &other) const
- bool **operator!=** (const [RankInfo](#) &other) const
- bool **operator<** (const [RankInfo](#) &other) const
- bool **operator<=** (const [RankInfo](#) &other) const
- bool **operator>** (const [RankInfo](#) &other) const
- bool **operator>=** (const [RankInfo](#) &other) const
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Public Attributes

- uint32_t **rank**
- uint32_t **thread**

Static Public Attributes

- static const uint32_t **UNASSIGNED** = (uint32_t)-1

Additional Inherited Members

6.156.1 Member Function Documentation

6.156.1.1 bool SST::RankInfo::inRange (const [RankInfo](#) & *other*) const [inline]

Returns

true if other's rank and thread are less than ours

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

- sst/core/rankInfo.h

6.157 SST::RankSyncParallelSkip Class Reference

Inheritance diagram for SST::RankSyncParallelSkip:

Collaboration diagram for SST::RankSyncParallelSkip:

Public Member Functions

- [RankSyncParallelSkip](#) ([RankInfo](#) num_ranks, [TimeConverter](#) *minPartTC)
- [ActivityQueue](#) * [registerLink](#) (const [RankInfo](#) &to_rank, const [RankInfo](#) &from_rank, [LinkId_t](#) link_id, [Link](#) *link) override
- void [execute](#) (int thread) override
- void [exchangeLinkUntimedData](#) (int thread, std::atomic< int > &msg_count) override
- void [finalizeLinkConfigurations](#) () override
- void [prepareForComplete](#) () override
- [SimTime_t](#) [getNextSyncTime](#) () override
- [uint64_t](#) [getDataSize](#) () const override

Additional Inherited Members

6.157.1 Constructor & Destructor Documentation

6.157.1.1 SST::RankSyncParallelSkip::RankSyncParallelSkip ([RankInfo](#) num_ranks, [TimeConverter](#) * minPartTC)

Create a new Sync object which fires with a specified period

6.157.2 Member Function Documentation

6.157.2.1 void SST::RankSyncParallelSkip::exchangeLinkUntimedData (int thread, std::atomic< int > & msg_count)
[override], [virtual]

Cause an exchange of Untimed Data to occur

Implements [SST::NewRankSync](#).

6.157.2.2 void SST::RankSyncParallelSkip::finalizeLinkConfigurations () [override], [virtual]

Finish link configuration

Implements [SST::NewRankSync](#).

6.157.2.3 void SST::RankSyncParallelSkip::prepareForComplete () [override], [virtual]

Prepare for complete() stage

Implements [SST::NewRankSync](#).

6.157.2.4 [ActivityQueue](#) * SST::RankSyncParallelSkip::registerLink (const [RankInfo](#) & to_rank, const [RankInfo](#) & from_rank, [LinkId_t](#) link_id, [Link](#) * link) [override], [virtual]

Register a [Link](#) which this Sync Object is responsible for

Implements [SST::NewRankSync](#).

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

- sst/core/rankSyncParallelSkip.h
- sst/core/rankSyncParallelSkip.cc

6.158 SST::RankSyncSerialSkip Class Reference

Inheritance diagram for SST::RankSyncSerialSkip:

Collaboration diagram for SST::RankSyncSerialSkip:

Public Member Functions

- [RankSyncSerialSkip](#) ([TimeConverter](#) *minPartTC)
- [ActivityQueue](#) * [registerLink](#) (const [RankInfo](#) &to_rank, const [RankInfo](#) &from_rank, LinkId_t link_id, [Link](#) *link) override
- void [execute](#) (int thread) override
- void [exchangeLinkUntimedData](#) (int thread, std::atomic< int > &msg_count) override
- void [finalizeLinkConfigurations](#) () override
- void [prepareForComplete](#) () override
- SimTime_t [getNextSyncTime](#) () override
- uint64_t [getDataSize](#) () const override

Additional Inherited Members

6.158.1 Constructor & Destructor Documentation

6.158.1.1 SST::RankSyncSerialSkip::RankSyncSerialSkip ([TimeConverter](#) * *minPartTC*)

Create a new Sync object which fires with a specified period

6.158.2 Member Function Documentation

6.158.2.1 void SST::RankSyncSerialSkip::exchangeLinkUntimedData (int *thread*, std::atomic< int > & *msg_count*) [*override*], [*virtual*]

Cause an exchange of Untimed Data to occur

Implements [SST::NewRankSync](#).

6.158.2.2 void SST::RankSyncSerialSkip::finalizeLinkConfigurations () [*override*], [*virtual*]

Finish link configuration

Implements [SST::NewRankSync](#).

6.158.2.3 void SST::RankSyncSerialSkip::prepareForComplete () [*override*], [*virtual*]

Prepare for the complete() stage

Implements [SST::NewRankSync](#).

6.158.2.4 **ActivityQueue** * SST::RankSyncSerialSkip::registerLink (const RankInfo & *to_rank*, const RankInfo & *from_rank*, LinkId_t *link_id*, Link * *link*) [override], [virtual]

Register a [Link](#) which this Sync Object is responsible for

Implements [SST::NewRankSync](#).

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

- sst/core/rankSyncSerialSkip.h
- sst/core/rankSyncSerialSkip.cc

6.159 SST::Core::Serialization::pvt::raw_ptr_wrapper< TPtr > Class Template Reference

Public Member Functions

- **raw_ptr_wrapper** (TPtr *&ptr)

Public Attributes

- TPtr *& **bufptr**

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

- sst/core/serialization/serialize_array.h

6.160 SST::RegionInfo Class Reference

Classes

- class [BulkMergeInfo](#)
- class [ChangeSetMergeInfo](#)
- class [RegionMergeInfo](#)

Public Member Functions

- bool **initialize** (const std::string &key, size_t size, uint8_t initByte, [SharedRegionMerger](#) *mergeObj)
- bool **isInitialized** () const
- bool **isReady** () const
- [SharedRegionImpl](#) * **addSharer** ([SharedRegionManager](#) *manager)
- void **removeSharer** ([SharedRegionImpl](#) *sri)
- void **modifyRegion** (size_t offset, size_t length, const void *data)
- void **publish** ()
- void **updateState** (bool finalize)
- const std::string & **getKey** () const
- void * **getMemory** ()
- const void * **getConstPtr** () const
- size_t **getSize** () const
- size_t **getNumSharers** () const
- bool **shouldMerge** () const
- [SharedRegionMerger](#) * **getMerger** ()
- [RegionMergeInfo](#) * **getMergeInfo** ()
- void **setProtected** (bool readOnly)

6.160.1 Member Function Documentation

6.160.1.1 RegionInfo::RegionMergeInfo * SST::RegionInfo::getMergeInfo ()

Returns the size of the data to be transferred

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

- sst/core/sharedRegionImpl.h
- sst/core/sharedRegion.cc

6.161 SST::RegionInfo::RegionMergeInfo Class Reference

Inheritance diagram for SST::RegionInfo::RegionMergeInfo:

Collaboration diagram for SST::RegionInfo::RegionMergeInfo:

Public Member Functions

- **RegionMergeInfo** (int rank, const std::string &key)
- virtual bool **merge** ([RegionInfo](#) *UNUSED(ri))
- const std::string & **getKey** () const
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Protected Attributes

- int **rank**
- std::string **key**

Additional Inherited Members

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

- sst/core/sharedRegionImpl.h

6.162 SST::Interfaces::SimpleMem::Request Class Reference

```
#include <simpleMem.h>
```

Public Types

- enum [Command](#) {
[Read](#), [Write](#), [ReadResp](#), [WriteResp](#),
[FlushLine](#), [FlushLineInv](#), [FlushLineResp](#), [TxBegin](#),
[TxEnd](#), [TxResp](#), [TxAbort](#), [TxCommit](#),
[CustomCmd](#) }
- enum [Flags](#) {
[F_NONCACHEABLE](#) = 1<<1, [F_LOCKED](#) = 1<<2, [F_LLSC](#) = 1<<3, [F_LLSC_RESP](#) = 1<<4,
[F_FLUSH_SUCCESS](#) = 1<<5, [F_TRANSACTION](#) = 1<<6 }
- typedef uint64_t [id_t](#)
- typedef uint32_t [flags_t](#)
- typedef std::vector< uint8_t > [dataVec](#)

Public Member Functions

- [Request](#) ([Command](#) cmd, [Addr](#) addr, [size_t](#) size, [dataVec](#) &data, [flags_t](#) flags=0, [flags_t](#) memFlags=0)
- [Request](#) ([Command](#) cmd, [Addr](#) addr, [size_t](#) size, [flags_t](#) flags=0, [flags_t](#) memFlags=0)
- [Request](#) ([Command](#) cmd, [Addr](#) addr, [size_t](#) size, [dataVec](#) &data, uint32_t Opc, [flags_t](#) flags=0, [flags_t](#) memFlags=0)
- [Request](#) ([Command](#) cmd, [Addr](#) addr, [size_t](#) size, uint32_t Opc, [flags_t](#) flags=0, [flags_t](#) memFlags=0)
- void [addAddress](#) ([Addr](#) addr)
- void [setPayload](#) (const std::vector< uint8_t > &data_in)
Set the contents of the payload / data field.
- void [setPayload](#) (uint8_t *data_in, [size_t](#) len)
Set the contents of the payload / data field.
- void [setVirtualAddress](#) (const [Addr](#) newVA)
Set the virtual address associated with the operation.
- uint64_t [getVirtualAddress](#) ()
- void [setInstructionPointer](#) (const [Addr](#) newIP)
Sets the instruction pointer associated with the operation.
- [Addr](#) [getInstructionPointer](#) ()
- void [clearFlags](#) (void)
Clears the flags associated with the operation.
- void [setFlags](#) ([flags_t](#) inValue)
- [flags_t](#) [getFlags](#) (void)
- void [clearMemFlags](#) (void)
Clears the memory flags associated with the operation.
- void [setMemFlags](#) ([flags_t](#) inValue)
- [flags_t](#) [getMemFlags](#) (void)
- uint32_t [getCustomOpc](#) (void)

Public Attributes

- [Command](#) cmd
- std::vector< [Addr](#) > [addrs](#)
- [Addr](#) addr
- [size_t](#) size
- [dataVec](#) data
- [flags_t](#) flags
- [flags_t](#) memFlags
- [id_t](#) id
- [Addr](#) instrPtr
- [Addr](#) virtualAddr
- uint32_t [custOpc](#)

6.162.1 Detailed Description

Represents both memory requests and responses.

6.162.2 Member Typedef Documentation

6.162.2.1 `typedef std::vector<uint8_t> SST::Interfaces::SimpleMem::Request::dataVec`

Type of the payload or data

6.162.2.2 `typedef uint32_t SST::Interfaces::SimpleMem::Request::flags_t`

Flag type

6.162.2.3 `typedef uint64_t SST::Interfaces::SimpleMem::Request::id_t`

[Request](#) ID type

6.162.3 Member Enumeration Documentation

6.162.3.1 `enum SST::Interfaces::SimpleMem::Request::Command`

Commands and responses possible with a [Request](#) object

Enumerator

- Read*** Issue a Read from Memory
- Write*** Issue a Write to Memory
- ReadResp*** Response from Memory to a Read
- WriteResp*** Response from Memory to a Write
- FlushLine*** Cache flush request - writeback specified line throughout memory system
- FlushLineInv*** Cache flush request - writeback and invalidate specified line throughout memory system
- FlushLineResp*** Response to FlushLine; flag F_FLUSH_SUCCESS indicates success or failure
- TxBegin*** Start a new transaction
- TxEnd*** End the current lowest transaction
- CustomCmd*** Custom memory command: Must also set custCmd opcode

6.162.3.2 `enum SST::Interfaces::SimpleMem::Request::Flags`

Flags to specify conditions on a [Request](#)

Enumerator

- F_NONCACHEABLE*** This request should not be cached
- F_LOCKED*** This request should be locked. A LOCKED read should be soon followed by a LOCKED write (to unlock)
- F_FLUSH_SUCCESS*** This flag is set if the flush was successful. Flush may fail due to LOCKED lines

6.162.4 Constructor & Destructor Documentation

6.162.4.1 `SST::Interfaces::SimpleMem::Request::Request (Command cmd, Addr addr, size_t size, dataVec & data, flags_t flags = 0, flags_t memFlags = 0) [inline]`

Constructor

6.162.4.2 `SST::Interfaces::SimpleMem::Request::Request (Command cmd, Addr addr, size_t size, flags_t flags = 0, flags_t memFlags = 0) [inline]`

Constructor

6.162.4.3 `SST::Interfaces::SimpleMem::Request::Request (Command cmd, Addr addr, size_t size, dataVec & data, uint32_t Opc, flags_t flags = 0, flags_t memFlags = 0) [inline]`

Constructor

6.162.4.4 `SST::Interfaces::SimpleMem::Request::Request (Command cmd, Addr addr, size_t size, uint32_t Opc, flags_t flags = 0, flags_t memFlags = 0) [inline]`

Constructor

6.162.5 Member Function Documentation

6.162.5.1 `uint32_t SST::Interfaces::SimpleMem::Request::getCustomOpc (void) [inline]`

Returns

the custom opcode for custom request types

6.162.5.2 `flags_t SST::Interfaces::SimpleMem::Request::getFlags (void) [inline]`

Returns

the flags associated with the operation

6.162.5.3 `Addr SST::Interfaces::SimpleMem::Request::getInstructionPointer () [inline]`

Returns

the instruction pointer associated with the operation

6.162.5.4 `flags_t SST::Interfaces::SimpleMem::Request::getMemFlags (void) [inline]`

Returns

the memory flags associated with the operation

6.162.5.5 `uint64_t SST::Interfaces::SimpleMem::Request::getVirtualAddress () [inline]`

Returns

the virtual address associated with the operation

6.162.5.6 `void SST::Interfaces::SimpleMem::Request::setFlags (flags_t inValue) [inline]`

Parameters

in	<i>inValue</i>	Should be one of the flags beginning with F_ in simpleMem
----	----------------	---

6.162.5.7 `void SST::Interfaces::SimpleMem::Request::setInstructionPointer (const Addr newIP) [inline]`

Sets the instruction pointer associated with the operation.

Parameters

in	<i>new</i> ↔ <i>IP</i>	
----	---------------------------	--

6.162.5.8 `void SST::Interfaces::SimpleMem::Request::setMemFlags (flags_t inValue) [inline]`

Parameters

in	<i>inValue</i>	Should be one of the flags beginning with F_ in simpleMem
----	----------------	---

6.162.5.9 `void SST::Interfaces::SimpleMem::Request::setPayload (const std::vector< uint8_t > & data_in) [inline]`

Set the contents of the payload / data field.

Parameters

in	<i>data</i> ↔ <i>_in</i>	
----	-----------------------------	--

6.162.5.10 `void SST::Interfaces::SimpleMem::Request::setPayload (uint8_t * data_in, size_t len) [inline]`

Set the contents of the payload / data field.

Parameters

in	<i>data</i> ↔ <i>_in</i>	
----	-----------------------------	--

6.162.5.11 void SST::Interfaces::SimpleMem::Request::setVirtualAddress (const Addr *newVA*) [inline]

Set the virtual address associated with the operation.

Parameters

in	<i>newVA</i>	
----	--------------	--

6.162.6 Member Data Documentation

6.162.6.1 Addr SST::Interfaces::SimpleMem::Request::addr

Target address - DEPRECATED but included for backward compatibility, defaults to `addr[0]`

6.162.6.2 std::vector<Addr> SST::Interfaces::SimpleMem::Request::addrs

Target address(es)

6.162.6.3 Command SST::Interfaces::SimpleMem::Request::cmd

Command to issue

6.162.6.4 uint32_t SST::Interfaces::SimpleMem::Request::custOpc

Custom command opcode for CustomdCmd type commands

6.162.6.5 dataVec SST::Interfaces::SimpleMem::Request::data

Payload data (for Write, or ReadResp)

6.162.6.6 flags_t SST::Interfaces::SimpleMem::Request::flags

Flags associated with this request or response

6.162.6.7 id_t SST::Interfaces::SimpleMem::Request::id

Unique ID to identify responses with requests

6.162.6.8 Addr SST::Interfaces::SimpleMem::Request::instrPtr

Instruction pointer associated with the operation

6.162.6.9 flags_t SST::Interfaces::SimpleMem::Request::memFlags

Memory flags - ignored by caches except to be passed through with request to main memory

6.162.6.10 size_t SST::Interfaces::SimpleMem::Request::size

Size of this request or response

6.162.6.11 Addr SST::Interfaces::SimpleMem::Request::virtualAddr

Virtual address associated with the operation

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

- sst/core/interfaces/simpleMem.h
- sst/core/interfaces/simpleMem.cc

6.163 SST::Interfaces::SimpleNetwork::Request Class Reference

```
#include <simpleNetwork.h>
```

Inheritance diagram for SST::Interfaces::SimpleNetwork::Request:

Collaboration diagram for SST::Interfaces::SimpleNetwork::Request:

Public Types

- enum [TraceType](#) { [NONE](#), [ROUTE](#), [FULL](#) }

Public Member Functions

- void [givePayload](#) ([Event](#) *event)
- [Event](#) * [takePayload](#) ()
- [Event](#) * [inspectPayload](#) ()
- [Request](#) ()
- **Request** ([nid_t](#) dest, [nid_t](#) src, [size_t](#) size_in_bits, bool head, bool tail, [Event](#) *payload=NULL)
- [Request](#) * [clone](#) ()
- void [setTraceID](#) (int id)
- void [setTraceType](#) ([TraceType](#) type)
- int [getTraceID](#) ()
- [TraceType](#) [getTraceType](#) ()
- void [serialize_order](#) ([SST::Core::Serialization::serializer](#) &ser) override

Public Attributes

- [nid_t](#) `dest`
- [nid_t](#) `src`
- [int](#) `vn`
- [size_t](#) `size_in_bits`
- [bool](#) `head`
- [bool](#) `tail`
- [bool](#) `allow_adaptive`

Protected Attributes

- [TraceType](#) `trace`
- [int](#) `tracelD`

Additional Inherited Members

6.163.1 Detailed Description

Represents both network sends and receives

6.163.2 Member Enumeration Documentation

6.163.2.1 enum SST::Interfaces::SimpleNetwork::Request::TraceType

Trace types

Enumerator

NONE No tracing enabled

ROUTE Trace route information only

FULL Trace all movements of packets through network

6.163.3 Constructor & Destructor Documentation

6.163.3.1 SST::Interfaces::SimpleNetwork::Request::Request () [inline]

Constructor

6.163.4 Member Function Documentation

6.163.4.1 void SST::Interfaces::SimpleNetwork::Request::givePayload (Event * *event*) [inline]

Sets the payload field for this request

Parameters

<i>payload</i> ↔ _in	Event to set as payload.
-------------------------	--

6.163.4.2 **Event*** SST::Interfaces::SimpleNetwork::Request::inspectPayload () [inline]

Returns the payload for the request for inspection. This call does not set the payload to NULL, so deleting the request will also delete the payload. If the request is going to be deleted, use takePayload instead.

Returns

[Event](#) that was set as payload of the request.

6.163.4.3 **Event*** SST::Interfaces::SimpleNetwork::Request::takePayload () [inline]

Returns the payload for the request. This will also set the payload to NULL, so the call will only return valid data one time after each givePayload call.

Returns

[Event](#) that was set as payload of the request.

6.163.5 Member Data Documentation

6.163.5.1 **bool** SST::Interfaces::SimpleNetwork::Request::allow_adaptive

Indicates whether adaptive routing is allowed or not.

6.163.5.2 **nid_t** SST::Interfaces::SimpleNetwork::Request::dest

Node ID of destination

6.163.5.3 **bool** SST::Interfaces::SimpleNetwork::Request::head

True if this is the head of a stream

6.163.5.4 **size_t** SST::Interfaces::SimpleNetwork::Request::size_in_bits

Size of packet in bits

6.163.5.5 **nid_t** SST::Interfaces::SimpleNetwork::Request::src

Node ID of source

6.163.5.6 `bool SST::Interfaces::SimpleNetwork::Request::tail`

True if this is the tail of a steram

6.163.5.7 `int SST::Interfaces::SimpleNetwork::Request::vn`

Virtual network of packet

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

- `sst/core/interfaces/simpleNetwork.h`

6.164 SST::SelfLink Class Reference

```
#include <link.h>
```

Inheritance diagram for SST::SelfLink:

Collaboration diagram for SST::SelfLink:

Additional Inherited Members

6.164.1 Detailed Description

Self Links are links from a component to itself

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

- `sst/core/link.h`

6.165 SST::Core::Serialization::pvt::ser_array_wrapper< TPtr, IntType > Class Template Reference

Public Member Functions

- **`ser_array_wrapper`** (TPtr *&buf, IntType &size)

Public Attributes

- TPtr *& **`bufptr`**
- IntType & **`sizeptr`**

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

- `sst/core/serialization/serialize_array.h`

6.166 SST::Core::Serialization::pvt::ser_buffer_accessor Class Reference

Inheritance diagram for SST::Core::Serialization::pvt::ser_buffer_accessor:

Public Member Functions

- template<class T >
T * **next** ()
- char * **next_str** (size_t size)
- size_t **size** () const
- size_t **max_size** () const
- void **init** (void *buffer, size_t size)
- void **clear** ()
- void **reset** ()

Protected Attributes

- char * **bufstart_**
- char * **bufptr_**
- size_t **size_**
- size_t **max_size_**

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

- sst/core/serialization/serialize_buffer_accessor.h

6.167 SST::Core::Serialization::pvt::ser_buffer_overrun Class Reference

Inheritance diagram for SST::Core::Serialization::pvt::ser_buffer_overrun:

Collaboration diagram for SST::Core::Serialization::pvt::ser_buffer_overrun:

Public Member Functions

- **ser_buffer_overrun** (int UNUSED(maxsize))

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

- sst/core/serialization/serialize_buffer_accessor.h

6.168 SST::Core::Serialization::pvt::ser_packer Class Reference

Inheritance diagram for SST::Core::Serialization::pvt::ser_packer:

Collaboration diagram for SST::Core::Serialization::pvt::ser_packer:

Public Member Functions

- template<class T >
void **pack** (T &t)
- void [pack_buffer](#) (void *buf, int size)
pack_buffer
- void **pack_string** (std::string &str)

Additional Inherited Members

6.168.1 Member Function Documentation

6.168.1.1 void SST::Core::Serialization::pvt::ser_packer::pack_buffer (void * *buf*, int *size*)

pack_buffer

Parameters

<i>buf</i>	Must be non-null
<i>size</i>	Must be non-zero

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

- sst/core/serialization/serialize_packer.h
- sst/core/serialization/serializer.cc

6.169 SST::Core::Serialization::pvt::ser_sizer Class Reference

Public Member Functions

- template<class T >
void **size** (T &UNUSED(t))
- void **size_string** (std::string &str)
- void **add** (size_t s)
- size_t **size** () const
- void **reset** ()

Protected Attributes

- size_t **size_**

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

- sst/core/serialization/serialize_sizer.h
- sst/core/serialization/serializer.cc

6.170 SST::Core::Serialization::pvt::ser_unpacker Class Reference

Inheritance diagram for SST::Core::Serialization::pvt::ser_unpacker:

Collaboration diagram for SST::Core::Serialization::pvt::ser_unpacker:

Public Member Functions

- template<class T >
void **unpack** (T &t)
- void **unpack_buffer** (void *buf, int size)
unpack_buffer
- void **unpack_string** (std::string &str)

Additional Inherited Members

6.170.1 Member Function Documentation

6.170.1.1 void SST::Core::Serialization::pvt::ser_unpacker::unpack_buffer (void * *buf*, int *size*)

unpack_buffer

Parameters

<i>buf</i>	Must unpack to non-null buffer
<i>size</i>	Must be non-zero

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

- sst/core/serialization/serialize_unpacker.h
- sst/core/serialization/serializer.cc

6.171 SST::Core::Serialization::serializable Class Reference

Inheritance diagram for SST::Core::Serialization::serializable:

Public Member Functions

- virtual const char * **cls_name** () const =0
- virtual void **serialize_order** (serializer &ser)=0
- virtual uint32_t **cls_id** () const =0
- virtual std::string **serialization_name** () const =0

Static Public Attributes

- static constexpr uint32_t **NullClsId** = std::numeric_limits<uint32_t>::max()

Protected Types

- enum **cxn_flag_t** { **ConstructorFlag** }

Static Protected Member Functions

- static void **serializable_abort** (uint32_t line, const char *file, const char *func, const char *obj)

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

- sst/core/serialization/serializable.h
- sst/core/serialization/serializable.cc

6.172 SST::Core::Serialization::serializable_builder Class Reference

Inheritance diagram for SST::Core::Serialization::serializable_builder:

Public Member Functions

- virtual [serializable](#) * **build** () const =0
- virtual const char * **name** () const =0
- virtual uint32_t **cls_id** () const =0
- virtual bool **sanity** ([serializable](#) *ser)=0

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

- sst/core/serialization/serializable.h

6.173 SST::Core::Serialization::serializable_builder_impl< T > Class Template Reference

Inheritance diagram for SST::Core::Serialization::serializable_builder_impl< T >:

Collaboration diagram for SST::Core::Serialization::serializable_builder_impl< T >:

Public Member Functions

- [serializable](#) * **build** () const override
- const char * **name** () const override
- uint32_t **cls_id** () const override
- bool **sanity** ([serializable](#) *ser) override

Static Public Member Functions

- static uint32_t **static_cls_id** ()
- static const char * **static_name** ()

Static Protected Attributes

- static const char * **name_** = typeid(T).name()
- static const uint32_t **cls_id_**

6.173.1 Member Data Documentation

6.173.1.1 `template<class T> const uint32_t SST::Core::Serialization::serializable_builder_impl< T >::cls_id_`
`[static], [protected]`

Initial value:

```
= serializable_factory::add_builder(new serializable_builder_impl<T>,
    typeid(T).name())
```

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

- sst/core/serialization/serializable.h

6.174 SST::Core::Serialization::serializable_factory Class Reference

Static Public Member Functions

- static serializable * **get_serializable** (uint32_t cls_id)
- static uint32_t **add_builder** (serializable_builder *builder, const char *name)
- static bool **sanity** (serializable *ser, uint32_t cls_id)
- static void **delete_statics** ()

Protected Types

- typedef std::unordered_map< long, serializable_builder * > **builder_map**

Static Protected Attributes

- static builder_map * **builders_** = 0

6.174.1 Member Function Documentation

6.174.1.1 `uint32_t SST::Core::Serialization::serializable_factory::add_builder (serializable_builder * builder, const char * name) [static]`

Returns

The cls id for the given builder

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

- sst/core/serialization/serializable.h
- sst/core/serialization/serializable.cc

6.175 sprockit::serializable_ptr_type Class Reference

Inheritance diagram for sprockit::serializable_ptr_type:

Collaboration diagram for sprockit::serializable_ptr_type:

Public Types

- `typedef sprockit::refcount_ptr< serializable_ptr_type > ptr`

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

- sst/core/serialization/ser_ptr_type.h

6.176 SST::Core::Serialization::serializable_type< T > Class Template Reference

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

- sst/core/serialization/serializable.h

6.177 SST::Core::Serialization::serialize< T, Enable > Class Template Reference

```
#include <serialize.h>
```

Public Member Functions

- `void operator() (T &UNUSED(t), serializer &UNUSED(ser))`

6.177.1 Detailed Description

```
template<class T, class Enable = void>  
class SST::Core::Serialization::serialize< T, Enable >
```

Base serialize class. This is the default, which if hit will static_assert. All other instances are partial specializations of this class and do all the real serialization.

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

- sst/core/serialization/serialize.h

6.178 SST::Core::Serialization::serialize< bool > Class Template Reference

```
#include <serialize.h>
```

Public Member Functions

- void **operator()** (bool &t, [serializer](#) &ser)

6.178.1 Detailed Description

```
template<>  
class SST::Core::Serialization::serialize< bool >
```

Version of serialize that works for bool.

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

- sst/core/serialization/serialize.h

6.179 SST::Core::Serialization::serialize< pvt::raw_ptr_wrapper< TPtr > > Class Template Reference

```
#include <serialize_array.h>
```

Public Member Functions

- void **operator()** ([pvt::raw_ptr_wrapper](#)< TPtr > ptr, [serializer](#) &ser)

6.179.1 Detailed Description

```
template<class TPtr>
class SST::Core::Serialization::serialize< pvt::raw_ptr_wrapper< TPtr > >
```

Version of serialize that works for copying raw pointers (only copying the value of the pointer. Note that this is only useful if the value is going to be sent back to the originator, since it won't be valid on the other rank.

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

- sst/core/serialization/serialize_array.h

6.180 SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if< std::is_fundamental< T >::value||std::is_enum< T >::value >::type > Class Template Reference

```
#include <serialize_array.h>
```

Public Member Functions

- void **operator()** (pvt::ser_array_wrapper< T, IntType > arr, serializer &ser)

6.180.1 Detailed Description

```
template<class T, class IntType>
class SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if< std::is_fundamental< T >::value||std::is_enum< T >::value >::type >
```

Version of serialize that works for dynamically allocated arrays of fundamental types and enums.

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

- sst/core/serialization/serialize_array.h

6.181 SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if<!std::is_fundamental< T >::value &&!std::is_enum< T >::value >::type > Class Template Reference

```
#include <serialize_array.h>
```

Public Member Functions

- void **operator()** (pvt::ser_array_wrapper< T, IntType > arr, serializer &ser)

6.181.1 Detailed Description

```
template<class T, class IntType>
class SST::Core::Serialization::serialize< pvt::ser_array_wrapper< T, IntType >, typename std::enable_if<!std::is_
fundamental< T >::value &&!std::is_enum< T >::value >::type >
```

Version of serialize that works for dynamically allocated arrays of non base types.

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

- sst/core/serialization/serialize_array.h

6.182 SST::Core::Serialization::serialize< pvt::ser_array_wrapper< void, IntType > > Class Template Reference

```
#include <serialize_array.h>
```

Public Member Functions

- void **operator()** (pvt::ser_array_wrapper< void, IntType > arr, [serializer](#) &ser)

6.182.1 Detailed Description

```
template<class IntType>
class SST::Core::Serialization::serialize< pvt::ser_array_wrapper< void, IntType > >
```

Version of serialize that works for statically allocated arrays of void*.

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

- sst/core/serialization/serialize_array.h

6.183 SST::Core::Serialization::serialize< serializable * > Class Template Reference

Public Member Functions

- void **operator()** ([serializable](#) *&s, [serializer](#) &ser)

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

- sst/core/serialization/serialize_serializable.h

6.184 SST::Core::Serialization::serialize< SST::SparseVectorMap< keyT, classT > > Class Template Reference

Public Member Functions

- void **operator()** (SST::SparseVectorMap< keyT, classT > &v, SST::Core::Serialization::serializer &ser)

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

- sst/core/sparseVectorMap.h

6.185 SST::Core::Serialization::serialize< std::deque< T > > Class Template Reference

Public Member Functions

- void **operator()** (Deque &v, serializer &ser)

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

- sst/core/serialization/serialize_deque.h

6.186 SST::Core::Serialization::serialize< std::list< T > > Class Template Reference

Public Member Functions

- void **operator()** (List &v, serializer &ser)

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

- sst/core/serialization/serialize_list.h

6.187 SST::Core::Serialization::serialize< std::map< Key, Value > > Class Template Reference

Public Member Functions

- void **operator()** (Map &m, serializer &ser)

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

- sst/core/serialization/serialize_map.h

6.188 SST::Core::Serialization::serialize< std::pair< U, V > > Class Template Reference

```
#include <serialize.h>
```

Public Member Functions

- void **operator()** (std::pair< U, V > &t, [serializer](#) &ser)

6.188.1 Detailed Description

```
template<class U, class V>  
class SST::Core::Serialization::serialize< std::pair< U, V > >
```

Version of serialize that works for std::pair.

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

- sst/core/serialization/serialize.h

6.189 SST::Core::Serialization::serialize< std::set< T > > Class Template Reference

Public Member Functions

- void **operator()** (Set &v, [serializer](#) &ser)

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

- sst/core/serialization/serialize_set.h

6.190 SST::Core::Serialization::serialize< std::string > Class Template Reference

Public Member Functions

- void **operator()** (std::string &str, [serializer](#) &ser)

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

- sst/core/serialization/serialize_string.h

6.191 SST::Core::Serialization::serialize< std::vector< T > > Class Template Reference

Public Member Functions

- void **operator()** (Vector &v, [serializer](#) &ser)

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

- sst/core/serialization/serialize_vector.h

6.192 SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type > Class Template Reference

Public Member Functions

- void **operator()** (T *&s, [serializer](#) &ser)

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

- sst/core/serialization/serialize_serializable.h

6.193 SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_fundamental< T >::value || std::is_enum< T >::value >::type > Class Template Reference

```
#include <serialize.h>
```

Public Member Functions

- void **operator()** (T *&t, [serializer](#) &ser)

6.193.1 Detailed Description

```
template<class T>
class SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_fundamental< T >::value || std::is_enum< T >::value >::type >
```

Version of serialize that works for pointers to fundamental types and enums. Note that there is no pointer tracking. This only copies the value pointed to into the buffer. If multiple objects point to the same location, they will each have an independent copy after deserialization.

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

- sst/core/serialization/serialize.h

6.194 SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type > Class Template Reference

Public Member Functions

- void **operator()** (T &t, [serializer](#) &ser)

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

- sst/core/serialization/serialize_serializable.h

6.195 SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_fundamental< T >::value||std::is_enum< T >::value >::type > Class Template Reference

```
#include <serialize.h>
```

Public Member Functions

- void **operator()** (T &t, [serializer](#) &ser)

6.195.1 Detailed Description

```
template<class T>
class SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_fundamental< T >::value||std::is_enum< T >::value >::type >
```

Version of serialize that works for fundamental types and enums.

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

- sst/core/serialization/serialize.h

6.196 SST::Core::Serialization::serialize< T[N], typename std::enable_if< std::is_fundamental< T >::value||std::is_enum< T >::value >::type > Class Template Reference

```
#include <serialize_array.h>
```

Public Member Functions

- void **operator()** (T arr[N], [serializer](#) &ser)

6.196.1 Detailed Description

```
template<class T, int N>
class SST::Core::Serialization::serialize< T[N], typename std::enable_if< std::is_fundamental< T >::value || std::is_enum< T >::value >::type >
```

Version of serialize that works for statically allocated arrays of fundamental types and enums.

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

- sst/core/serialization/serialize_array.h

6.197 SST::Core::Serialization::serialize< T[N], typename std::enable_if<!std::is_fundamental< T >::value &&!std::is_enum< T >::value >::type > Class Template Reference

```
#include <serialize_array.h>
```

Public Member Functions

- void **operator()** (T arr[N], [serializer](#) &ser)

6.197.1 Detailed Description

```
template<class T, int N>
class SST::Core::Serialization::serialize< T[N], typename std::enable_if<!std::is_fundamental< T >::value &&!std::is_enum< T >::value >::type >
```

Version of serialize that works for statically allocated arrays of non base types.

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

- sst/core/serialization/serialize_array.h

6.198 SST::Core::Serialization::serializer Class Reference

```
#include <serializer.h>
```

Collaboration diagram for SST::Core::Serialization::serializer:

Public Types

- enum **SERIALIZE_MODE** { SIZER, PACK, UNPACK }

Public Member Functions

- [pvt::ser_packer](#) & **packer** ()
- [pvt::ser_unpacker](#) & **unpacker** ()
- [pvt::ser_sizer](#) & **sizer** ()
- template<class T >
void **size** (T &t)
- template<class T >
void **pack** (T &t)
- template<class T >
void **unpack** (T &t)
- SERIALIZE_MODE **mode** () const
- void **set_mode** (SERIALIZE_MODE mode)
- void **reset** ()
- template<typename T >
void **primitive** (T &t)
- template<class T , int N>
void **array** (T arr[N])
- template<typename T , typename Int >
void **binary** (T *&buffer, Int &size)
- template<typename Int >
void **binary** (void *&buffer, Int &size)
- void **string** (std::string &str)
- void **start_packing** (char *buffer, size_t size)
- void **start_sizing** ()
- void **start_unpacking** (char *buffer, size_t size)
- size_t **size** () const

Protected Attributes

- [pvt::ser_packer](#) **packer_**
- [pvt::ser_unpacker](#) **unpacker_**
- [pvt::ser_sizer](#) **sizer_**
- SERIALIZE_MODE **mode_**

6.198.1 Detailed Description

This class is basically a wrapper for objects to declare the order in which their members should be ser/des

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

- sst/core/serialization/serializer.h
- sst/core/serialization/serializer.cc

6.199 SST::SharedRegion Class Reference

Inheritance diagram for SST::SharedRegion:

Public Member Functions

- void **shutdown** ()
- size_t **getLocalShareID** () const
- size_t **getSize** () const
- void **publish** ()
- bool **isReady** () const
- void **modifyRegion** (size_t offset, size_t length, const void *data)
- template<typename T >
void **modifyArray** (size_t offset, const T &data)
- void * **getRawPtr** ()
- template<typename T >
T **getPtr** () const

Protected Member Functions

- **SharedRegion** (**SharedRegionManager** *manager, size_t id, size_t size)

6.199.1 Member Function Documentation

6.199.1.1 size_t SST::SharedRegion::getLocalShareID () const [inline]

Returns

The ID of this instance. (Number in range 0->N)

6.199.1.2 template<typename T > T SST::SharedRegion::getPtr () const [inline]

Returns

a const pointer to the shared memory region

6.199.1.3 void* SST::SharedRegion::getRawPtr () [inline]

Returns

a void* pointer to the shared memory region This pointer is only valid to write to before a call to [publish\(\)](#)

6.199.1.4 size_t SST::SharedRegion::getSize () const [inline]

Returns

The size of the shared memory region

6.199.1.5 `bool SST::SharedRegion::isReady () const [inline]`

Returns

True if the region is ready to use (all sharers have called [publish\(\)](#)).

6.199.1.6 `void SST::SharedRegion::modifyRegion (size_t offset, size_t length, const void * data) [inline]`

Before the region has published, apply a modification. (Copy this data in)

6.199.1.7 `void SST::SharedRegion::publish () [inline]`

Call to denote that you are done making any changes to this region

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

- `sst/core/sharedRegion.h`

6.200 SST::SharedRegionImpl Class Reference

Inheritance diagram for SST::SharedRegionImpl:

Collaboration diagram for SST::SharedRegionImpl:

Public Member Functions

- **SharedRegionImpl** ([SharedRegionManager](#) *manager, size_t id, size_t size, [RegionInfo](#) *region)
- `bool isPublished () const`
- `void setPublished ()`
- [RegionInfo](#) * **getRegion** () const

Additional Inherited Members

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

- `sst/core/sharedRegionImpl.h`

6.201 SST::SharedRegionInitializedMerger Class Reference

Inheritance diagram for SST::SharedRegionInitializedMerger:

Collaboration diagram for SST::SharedRegionInitializedMerger:

Public Member Functions

- **SharedRegionInitializedMerger** (uint8_t defaultValue)
- bool **merge** (uint8_t *target, const uint8_t *newData, size_t size) override
- bool **merge** (uint8_t *target, size_t size, const std::vector< [ChangeSet](#) > &changeSets) override

6.201.1 Member Function Documentation

6.201.1.1 bool SST::SharedRegionInitializedMerger::merge (uint8_t * *target*, const uint8_t * *newData*, size_t *size*)
[[override](#)], [[virtual](#)]

Merge the data from 'newData' into 'target'

Returns

True on success, False on failure

Reimplemented from [SST::SharedRegionMerger](#).

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

- sst/core/sharedRegion.h
- sst/core/sharedRegion.cc

6.202 SST::SharedRegionManager Class Reference

Inheritance diagram for SST::SharedRegionManager:

Public Member Functions

- virtual [SharedRegion](#) * **getLocalSharedRegion** (const std::string &key, size_t size, uint8_t initByte=0)=0
- virtual [SharedRegion](#) * **getGlobalSharedRegion** (const std::string &key, size_t size, [SharedRegionMerger](#) *merger=NULL, uint8_t initByte=0)=0
- virtual void **publishRegion** ([SharedRegion](#) *)=0
- virtual bool **isRegionReady** (const [SharedRegion](#) *)=0
- virtual void **shutdownSharedRegion** ([SharedRegion](#) *)=0
- virtual void **updateState** (bool finalize)=0

Protected Member Functions

- virtual void **modifyRegion** ([SharedRegion](#) *sr, size_t offset, size_t length, const void *data)=0
- virtual void * **getMemory** ([SharedRegion](#) *sr)=0
- virtual const void * **getConstPtr** (const [SharedRegion](#) *sr) const =0

Friends

- class **SharedRegion**

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

- sst/core/sharedRegion.h

6.203 SST::SharedRegionManagerImpl Class Reference

Inheritance diagram for SST::SharedRegionManagerImpl:

Collaboration diagram for SST::SharedRegionManagerImpl:

Public Member Functions

- virtual [SharedRegion](#) * **getLocalSharedRegion** (const std::string &key, size_t size, uint8_t initByte=0) override
- virtual [SharedRegion](#) * **getGlobalSharedRegion** (const std::string &key, size_t size, [SharedRegionMerger](#) *merger=NULL, uint8_t initByte=0) override
- virtual void **publishRegion** ([SharedRegion](#) *) override
- virtual bool **isRegionReady** (const [SharedRegion](#) *) override
- virtual void **shutdownSharedRegion** ([SharedRegion](#) *) override
- void **updateState** (bool finalize) override

Protected Member Functions

- void **modifyRegion** ([SharedRegion](#) *sr, size_t offset, size_t length, const void *data) override
- void * **getMemory** ([SharedRegion](#) *sr) override
- const void * **getConstPtr** (const [SharedRegion](#) *sr) const override

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

- sst/core/sharedRegionImpl.h
- sst/core/sharedRegion.cc

6.204 SST::SharedRegionMerger Class Reference

```
#include <sharedRegion.h>
```

Inheritance diagram for SST::SharedRegionMerger:

Public Member Functions

- virtual bool **merge** (uint8_t *target, const uint8_t *newData, size_t size)
- virtual bool **merge** (uint8_t *target, size_t size, const std::vector< [ChangeSet](#) > &changeSets)

6.204.1 Detailed Description

Utility class to define how to merge multiple pieces of shared memory regions Useful in the multi-MPI-rank, "Global Shared" model

6.204.2 Member Function Documentation

6.204.2.1 `virtual bool SST::SharedRegionMerger::merge (uint8_t * target, const uint8_t * newData, size_t size)`
[virtual]

Merge the data from 'newData' into 'target'

Returns

True on success, False on failure

Reimplemented in [SST::SharedRegionInitializedMerger](#).

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

- `sst/core/sharedRegion.h`
- `sst/core/sharedRegion.cc`

6.205 SST::Interfaces::SimpleMem Class Reference

```
#include <simpleMem.h>
```

Inheritance diagram for SST::Interfaces::SimpleMem:

Collaboration diagram for SST::Interfaces::SimpleMem:

Classes

- class [Handler](#)
- class [Handler< classT, void >](#)
- class [HandlerBase](#)
- class [Request](#)

Public Types

- typedef uint64_t [Addr](#)

Public Member Functions

- [SimpleMem](#) ([SST::Component](#) *comp, [Params](#) &UNUSED(params))
- [SimpleMem](#) ([SST::ComponentId_t](#) id, [Params](#) &UNUSED(params))
- virtual bool [initialize](#) (const std::string &linkName, [HandlerBase](#) *handler=NULL)=0
- virtual void [sendInitData](#) ([Request](#) *req)=0
- virtual void [sendInitData](#) ([SST::Event](#) *ev)
- virtual [SST::Event](#) * [recvInitData](#) ()
- virtual [SST::Link](#) * [getLink](#) (void) const =0
- virtual void [sendRequest](#) ([Request](#) *req)=0
- virtual [Request](#) * [recvResponse](#) (void)=0

Additional Inherited Members

6.205.1 Detailed Description

Simplified, generic interface to Memory models

6.205.2 Member Typedef Documentation

6.205.2.1 typedef uint64_t SST::Interfaces::SimpleMem::Addr

All Addresses can be 64-bit

6.205.3 Constructor & Destructor Documentation

6.205.3.1 SST::Interfaces::SimpleMem::SimpleMem (SST::Component * comp, Params & UNUSEDparams) [inline]

Constructor, designed to be used via 'loadSubComponent'.

6.205.3.2 SST::Interfaces::SimpleMem::SimpleMem (SST::ComponentId_t id, Params & UNUSEDparams) [inline]

Constructor, designed to be used via 'loadUserSubComponent' and 'loadAnonymousSubComponent'.

6.205.4 Member Function Documentation

6.205.4.1 virtual SST::Link* SST::Interfaces::SimpleMem::getLink (void) const [pure virtual]

Returns a handle to the underlying [SST::Link](#)

6.205.4.2 `virtual bool SST::Interfaces::SimpleMem::initialize (const std::string & linkName, HandlerBase * handler = NULL) [pure virtual]`

Second half of building the interface. Initialize with link name name, and handler, if any

Returns

true if the link was able to be configured.

6.205.4.3 `virtual SST::Event* SST::Interfaces::SimpleMem::recvInitData () [inline],[virtual]`

Receive any data during the [init\(\)](#) phase.

See also

[SST::Link::recvInitData\(\)](#)

6.205.4.4 `virtual Request* SST::Interfaces::SimpleMem::recvResponse (void) [pure virtual]`

Receive a [Request](#) response from the side of the link.

Use this method for polling-based applications. Register a handler for push-based notification of responses.

Returns

NULL if nothing is available.

Pointer to a [Request](#) response (that should be deleted)

6.205.4.5 `virtual void SST::Interfaces::SimpleMem::sendInitData (Request * req) [pure virtual]`

Sends a memory-based request during the [init\(\)](#) phase

6.205.4.6 `virtual void SST::Interfaces::SimpleMem::sendInitData (SST::Event * ev) [inline],[virtual]`

Sends a generic [Event](#) during the [init\(\)](#) phase (Mostly acts as a passthrough)

See also

[SST::Link::sendInitData\(\)](#)

6.205.4.7 `virtual void SST::Interfaces::SimpleMem::sendRequest (Request * req) [pure virtual]`

Send a [Request](#) to the other side of the link.

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

- `sst/core/interfaces/simpleMem.h`

6.206 SST::Interfaces::SimpleNetwork Class Reference

```
#include <simpleNetwork.h>
```

Inheritance diagram for SST::Interfaces::SimpleNetwork:

Collaboration diagram for SST::Interfaces::SimpleNetwork:

Classes

- class [Handler](#)
- class [Handler](#)< classT, void >
- class [HandlerBase](#)
- class [NetworkInspector](#)
- class [Request](#)

Public Types

- typedef int64_t [nid_t](#)

Public Member Functions

- [SimpleNetwork](#) (SST::Component *comp)
- [SimpleNetwork](#) (SST::ComponentId_t id)
- virtual bool [initialize](#) (const std::string &portName, const [UnitAlgebra](#) &link_bw, int vns, const [UnitAlgebra](#) &in_buf_size, const [UnitAlgebra](#) &out_buf_size)=0
- virtual void [sendInitData](#) ([Request](#) *req)=0
- virtual [Request](#) * [recvInitData](#) ()=0
- virtual void [sendUntimedData](#) ([Request](#) *req)
- virtual [Request](#) * [recvUntimedData](#) ()
- virtual bool [send](#) ([Request](#) *req, int vn)=0
- virtual [Request](#) * [recv](#) (int vn)=0
- virtual void [setup](#) () override
- virtual void [init](#) (unsigned int UNUSED(phase)) override
- virtual void [complete](#) (unsigned int UNUSED(phase)) override
- virtual void [finish](#) () override
- virtual bool [spaceToSend](#) (int vn, int num_bits)=0
- virtual bool [requestToReceive](#) (int vn)=0
- virtual void [setNotifyOnReceive](#) ([HandlerBase](#) *functor)=0
- virtual void [setNotifyOnSend](#) ([HandlerBase](#) *functor)=0
- virtual bool [isNetworkInitialized](#) () const =0
- virtual [nid_t](#) [getEndpointID](#) () const =0
- virtual const [UnitAlgebra](#) & [getLinkBW](#) () const =0

Static Public Attributes

- static const [nid_t](#) [INIT_BROADCAST_ADDR](#) = 0xffffffffffffff

Additional Inherited Members

6.206.1 Detailed Description

Generic network interface

6.206.2 Member Typedef Documentation

6.206.2.1 `typedef int64_t SST::Interfaces::SimpleNetwork::nid_t`

All Addresses can be 64-bit

6.206.3 Constructor & Destructor Documentation

6.206.3.1 `SST::Interfaces::SimpleNetwork::SimpleNetwork (SST::Component * comp) [inline]`

Constructor, designed to be used via 'loadSubComponent'.

6.206.3.2 `SST::Interfaces::SimpleNetwork::SimpleNetwork (SST::ComponentId_t id) [inline]`

Constructor, designed to be used via 'loadUserSubComponent' or 'loadAnonymousSubComponent'.

6.206.4 Member Function Documentation

6.206.4.1 `virtual void SST::Interfaces::SimpleNetwork::complete (unsigned int UNUSEDphase) [inline], [override], [virtual]`

Used during the init phase. The method will be called each phase of initialization. Initialization ends when no components have sent any data.

Reimplemented from [SST::BaseComponent](#).

6.206.4.2 `virtual void SST::Interfaces::SimpleNetwork::finish () [inline], [override], [virtual]`

Called after simulation completes, but before objects are destroyed. A good place to print out statistics.

Reimplemented from [SST::SubComponent](#).

6.206.4.3 `virtual nid_t SST::Interfaces::SimpleNetwork::getEndpointID () const [pure virtual]`

Returns the endpoint ID. Cannot be called until after the network is initialized.

Returns

Endpoint ID

6.206.4.4 `virtual const UnitAlgebra& SST::Interfaces::SimpleNetwork::getLinkBW () const` [pure virtual]

Returns the final BW of the link managed by the simpleNetwork instance. Cannot be called until after the network is initialized.

Returns

[Link](#) bandwidth of associated link

6.206.4.5 `virtual void SST::Interfaces::SimpleNetwork::init (unsigned int UNUSEDphase)` [inline],[override],[virtual]

Used during the init phase. The method will be called each phase of initialization. Initialization ends when no components have sent any data.

Reimplemented from [SST::SubComponent](#).

6.206.4.6 `virtual bool SST::Interfaces::SimpleNetwork::initialize (const std::string & portName, const UnitAlgebra & link_bw, int vns, const UnitAlgebra & in_buf_size, const UnitAlgebra & out_buf_size)` [pure virtual]

Second half of building the interface. Initialize network interface

Parameters

<i>portName</i>	- Name of port to connect to
<i>link_bw</i>	- Bandwidth of the link
<i>vns</i>	- Number of virtual networks to be provided
<i>in_buf_size</i>	- Size of input buffers (from router)
<i>out_buf_size</i>	- Size of output buffers (to router)

Returns

true if the link was able to be configured.

6.206.4.7 `virtual bool SST::Interfaces::SimpleNetwork::isNetworkInitialized () const` [pure virtual]

Check to see if network is initialized. If network is not initialized, then no other functions other than [init\(\)](#) can be called on the interface.

Returns

true if network is initialized, false otherwise

6.206.4.8 `virtual Request* SST::Interfaces::SimpleNetwork::recv (int vn)` [pure virtual]

Receive a [Request](#) from the network.

Use this method for polling-based applications. Register a handler for push-based notification of responses.

Parameters

<i>vn</i>	Virtual network to receive on
-----------	-------------------------------

Returns

NULL if nothing is available.
 Pointer to a [Request](#) response (that should be deleted)

6.206.4.9 `virtual Request* SST::Interfaces::SimpleNetwork::recvInitData () [pure virtual]`

Receive any data during the [init\(\)](#) phase.

See also

[SST::Link::recvInitData\(\)](#)

6.206.4.10 `virtual Request* SST::Interfaces::SimpleNetwork::recvUntimedData () [inline],[virtual]`

Receive any data during untimed phases ([init\(\)](#) and [complete\(\)](#)).

See also

[SST::Link::recvUntimedData\(\)](#)

For now, simply call `recvInitData`. Once that call is deprecated and removed, this will become a pure virtual function. This means that when classes implement [SimpleNetwork](#), they will need to overload both `recvUntimedData` and `recvInitData` with identical functionality (or having the `Init` version call the `Untimed` version) until `recvInitData` is removed in SST 9.0.

6.206.4.11 `virtual bool SST::Interfaces::SimpleNetwork::requestToReceive (int vn) [pure virtual]`

Checks if there is a waiting network request request pending in the specified virtual network.

Parameters

<i>vn</i>	Virtual network to check
-----------	--------------------------

Returns

true if a network request is pending in the specified virtual network, false otherwise

6.206.4.12 `virtual bool SST::Interfaces::SimpleNetwork::send (Request * req, int vn) [pure virtual]`

Returns a handle to the underlying [SST::Link](#) Send a [Request](#) to the network.

6.206.4.13 `virtual void SST::Interfaces::SimpleNetwork::sendInitData (Request * req) [pure virtual]`

Sends a network request during the [init\(\)](#) phase

6.206.4.14 `virtual void SST::Interfaces::SimpleNetwork::sendUntimedData (Request * req) [inline],[virtual]`

Sends a network request during untimed phases ([init\(\)](#) and [complete\(\)](#)).

See also

[SST::Link::sendUntimedData\(\)](#)

For now, simply call `sendInitData`. Once that call is deprecated and removed, this will become a pure virtual function. This means that when classes implement [SimpleNetwork](#), they will need to overload both `sendUntimedData` and `sendInitData` with identical functionality (or having the `Init` version call the `Untimed` version) until `sendInitData` is removed in SST 9.0.

6.206.4.15 `virtual void SST::Interfaces::SimpleNetwork::setNotifyOnReceive (HandlerBase * functor) [pure virtual]`

Registers a functor which will fire when a new request is received from the network. Note, the actual request that was received is not passed into the functor, it is only a notification that something is available.

Parameters

<i>functor</i>	Functor to call when request is received
----------------	--

6.206.4.16 `virtual void SST::Interfaces::SimpleNetwork::setNotifyOnSend (HandlerBase * functor) [pure virtual]`

Registers a functor which will fire when a request is sent to the network. Note, this only tells you when data is sent, it does not guarantee any specified amount of available space.

Parameters

<i>functor</i>	Functor to call when request is sent
----------------	--------------------------------------

6.206.4.17 `virtual void SST::Interfaces::SimpleNetwork::setup () [inline],[override],[virtual]`

Called after all components have been constructed and initialization has completed, but before simulation time has begun.

Reimplemented from [SST::SubComponent](#).

6.206.4.18 `virtual bool SST::Interfaces::SimpleNetwork::spaceToSend (int vn, int num_bits) [pure virtual]`

Checks if there is sufficient space to send on the specified virtual network

Parameters

<i>vn</i>	Virtual network to check
<i>num_bits</i>	Minimum size in bits required to have space to send

Returns

true if there is space in the output, false otherwise

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

- sst/core/interfaces/simpleNetwork.h
- sst/core/interfaces/simpleNetwork.cc

6.207 SST::IMPL::Partition::SimplePartitioner Class Reference

Inheritance diagram for SST::IMPL::Partition::SimplePartitioner:

Collaboration diagram for SST::IMPL::Partition::SimplePartitioner:

Public Member Functions

- [RankInfo](#) **convertPartNum** (uint32_t partNum)
- void **simple_partition_step** ([PartitionComponentMap_t](#) &component_map, ComponentId_t *setA, const int lengthA, int rankA, ComponentId_t *setB, const int lengthB, int rankB, std::map< ComponentId_t, std::map< ComponentId_t, SimTime_t > > timeTable, int step)
- **SimplePartitioner** ([RankInfo](#) total_ranks, [RankInfo](#) my_rank, int verbosity)
- void **performPartition** ([PartitionGraph](#) *graph) override
- void **performPartition** ([ConfigGraph](#) *graph) override
- bool **requiresConfigGraph** () override
- bool **spawnOnAllRanks** () override

Public Attributes

- SST_ELI_REGISTER_PARTITIONER([SimplePartitioner](#),"sst","simple", SST_ELI_ELEMENT_VERSION(1, 0, 0),"Simple partitioning scheme which attempts to partition on high latency links while balancing number of components per rank.") private uint32_t **total_parts**

6.207.1 Member Function Documentation

6.207.1.1 void SST::IMPL::Partition::SimplePartitioner::performPartition ([PartitionGraph](#) * *graph*) [override],
[virtual]

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [PartitionGraph](#) object.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.207.1.2 `void SST::IMPL::Partition::SimplePartitioner::performPartition (ConfigGraph * graph) [inline], [override], [virtual]`

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [ConfigGraph](#) object. The consequence of using ConfigGraphs is that no-cut links are not supported.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented from [SST::Partition::SSTPartitioner](#).

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

- sst/core/impl/partitioners/simplepart.h
- sst/core/impl/partitioners/simplepart.cc

6.208 SimThreadInfo_t Struct Reference

Collaboration diagram for SimThreadInfo_t:

Public Attributes

- [RankInfo](#) **myRank**
- [RankInfo](#) **world_size**
- [Config](#) * **config**
- [ConfigGraph](#) * **graph**
- SimTime_t **min_part**
- double **build_time**
- double **run_time**
- [UnitAlgebra](#) **simulated_time**
- uint64_t **max_tv_depth**
- uint64_t **current_tv_depth**
- uint64_t **sync_data_size**

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

- sst/core/main.cc

6.209 SST::Simulation Class Reference

```
#include <simulation.h>
```

Public Types

- enum [Mode_t](#) { UNKNOWN, INIT, RUN, BOTH }
- typedef std::map< std::pair< SimTime_t, int >, [Clock](#) * > [clockMap_t](#)
- typedef std::map< std::pair< SimTime_t, int >, [OneShot](#) * > [oneShotMap_t](#)

Public Member Functions

- void [printStatus](#) (bool fullStatus)
- void [processGraphInfo](#) ([ConfigGraph](#) &graph, const [RankInfo](#) &myRank, [SimTime_t](#) min_part)
- int [performWireUp](#) ([ConfigGraph](#) &graph, const [RankInfo](#) &myRank, [SimTime_t](#) min_part)
- void [setStopAtCycle](#) ([Config](#) *cfg)
- void [initialize](#) ()
- void [complete](#) ()
- void [setup](#) ()
- void [run](#) ()
- void [finish](#) ()
- bool [isIndependentThread](#) ()
- [Mode_t](#) [getSimulationMode](#) () const
- const [SimTime_t](#) & [getCurrentSimCycle](#) () const
- [SimTime_t](#) [getEndSimCycle](#) () const
- int [getCurrentPriority](#) () const
- [UnitAlgebra](#) [getElapsedSimTime](#) () const
- [UnitAlgebra](#) [getFinalSimTime](#) () const
- [RankInfo](#) [getRank](#) () const
- [RankInfo](#) [getNumRanks](#) () const
- [TimeConverter](#) * [registerClock](#) (const std::string &freq, [Clock::HandlerBase](#) *handler, int priority)
- [TimeConverter](#) * [registerClock](#) (const [UnitAlgebra](#) &freq, [Clock::HandlerBase](#) *handler, int priority)
- [TimeConverter](#) * [registerClock](#) ([TimeConverter](#) *tcFreq, [Clock::HandlerBase](#) *handler, int priority)
- void [unregisterClock](#) ([TimeConverter](#) *tc, [Clock::HandlerBase](#) *handler, int priority)
- [Cycle_t](#) [reregisterClock](#) ([TimeConverter](#) *tc, [Clock::HandlerBase](#) *handler, int priority)
- [Cycle_t](#) [getNextClockCycle](#) ([TimeConverter](#) *tc, int priority=CLOCKPRIORITY)
- [TimeConverter](#) * [registerOneShot](#) (std::string timeDelay, [OneShot::HandlerBase](#) *handler, int priority)
- [TimeConverter](#) * [registerOneShot](#) (const [UnitAlgebra](#) &timeDelay, [OneShot::HandlerBase](#) *handler, int priority)
- void [insertActivity](#) ([SimTime_t](#) time, [Activity](#) *ev)
- [Exit](#) * [getExit](#) () const
- const std::vector< [SimTime_t](#) > & [getInterThreadLatencies](#) () const
- [SimTime_t](#) [getInterThreadMinLatency](#) () const
- [uint64_t](#) [getTimeVortexMaxDepth](#) () const
- [uint64_t](#) [getTimeVortexCurrentDepth](#) () const
- [uint64_t](#) [getSyncQueueDataSize](#) () const
- [Statistics::StatisticProcessingEngine](#) * [getStatisticsProcessingEngine](#) (void) const
- [LinkMap](#) * [getComponentLinkMap](#) ([ComponentId_t](#) id) const
- const [ComponentInfoMap](#) & [getComponentInfoMap](#) (void)
- [BaseComponent](#) * [getComponent](#) (const [ComponentId_t](#) &id) const
- [ComponentInfo](#) * [getComponentInfo](#) (const [ComponentId_t](#) &id) const
- void [setOutputDirectory](#) (std::string &outDir)
- std::string & [getOutputDirectory](#) ()
- void [requireEvent](#) (std::string name)
- [SimTime_t](#) [getNextActivityTime](#) () const
- bool [isWireUpFinished](#) ()

Static Public Member Functions

- static [Simulation](#) * [createSimulation](#) ([Config](#) *config, [RankInfo](#) my_rank, [RankInfo](#) num_ranks, [SimTime_t](#) min_part)
- static void [shutdown](#) ()
- static [Simulation](#) * [getSimulation](#) ()
- static void [setSignal](#) (int signal)
- static [TimeConverter](#) * [getMinPartTC](#) ()
- static [TimeLord](#) * [getTimeLord](#) (void)
- static [Output](#) & [getSimulationOutput](#) ()
- static [SimTime_t](#) [getLocalMinimumNextActivityTime](#) ()
- static [SharedRegionManager](#) * [getSharedRegionManager](#) ()

Friends

- class **Link**
- class **Action**
- class **Output**
- class **SyncManager**
- void **wait_my_turn_start** ()
- void **wait_my_turn_end** ()

6.209.1 Detailed Description

Main control class for a SST [Simulation](#). Provides base features for managing the simulation

6.209.2 Member Typedef Documentation

6.209.2.1 `typedef std::map<std::pair<SimTime_t, int>, Clock*> SST::Simulation::clockMap_t`

Map of times to clocks

6.209.2.2 `typedef std::map<std::pair<SimTime_t, int>, OneShot*> SST::Simulation::oneShotMap_t`

Map of times to OneShots

6.209.3 Member Enumeration Documentation

6.209.3.1 `enum SST::Simulation::Mode_t`

Type of Run Modes

Enumerator

- UNKNOWN** Unknown mode - Invalid for running
- INIT** Initialize-only. Useful for debugging initialization and graph generation
- RUN** Run-only. Useful when restoring from a checkpoint
- BOTH** Default. Both initialize and Run the simulation

6.209.4 Member Function Documentation

6.209.4.1 `void SST::Simulation::complete ()`

Perform the [complete\(\)](#) phase of simulation

6.209.4.2 `Simulation * SST::Simulation::createSimulation (Config * config, RankInfo my_rank, RankInfo num_ranks, SimTime_t min_part) [static]`

Create new simulation

Parameters

<i>config</i>	- Configuration of the simulation
<i>my_rank</i>	- Parallel Rank of this simulation object
<i>num_ranks</i>	- How many Ranks are in the simulation

6.209.4.3 `BaseComponent* SST::Simulation::getComponent (const ComponentId_t & id) const` `[inline]`

returns the component with the given ID

6.209.4.4 `const ComponentInfoMap& SST::Simulation::getComponentInfoMap (void)` `[inline]`

Returns reference to the [Component](#) Map Returns reference to the [Component](#) ID Map

6.209.4.5 `LinkMap* SST::Simulation::getComponentLinkMap (ComponentId_t id) const` `[inline]`

Return pointer to map of links for a given component id

6.209.4.6 `int SST::Simulation::getCurrentPriority () const`

Return the current priority

6.209.4.7 `const SimTime_t & SST::Simulation::getCurrentSimCycle () const`

Return the current simulation time as a cycle count

6.209.4.8 `UnitAlgebra SST::Simulation::getElapsedSimTime () const`

Return the elapsed simulation time as a time

6.209.4.9 `SimTime_t SST::Simulation::getEndSimCycle () const`

Return the end simulation time as a cycle count

6.209.4.10 `Exit* SST::Simulation::getExit () const` `[inline]`

Return the exit event

6.209.4.11 `UnitAlgebra SST::Simulation::getFinalSimTime () const`

Return the end simulation time as a time

6.209.4.12 `SimTime_t SST::Simulation::getLocalMinimumNextActivityTime () [static]`

Gets the minimum next activity time across all TimeVortices in the Rank

6.209.4.13 `SimTime_t SST::Simulation::getNextActivityTime () const`

Returns the time of the next item to be executed that is in the TimeVortex of the [Simulation](#)

6.209.4.14 `Cycle_t SST::Simulation::getNextClockCycle (TimeConverter * tc, int priority = CLOCKPRIORITY)`

Returns the next Cycle that the TimeConverter would fire.

6.209.4.15 `RankInfo SST::Simulation::getNumRanks () const [inline]`

Get the number of parallel ranks in the simulation

6.209.4.16 `std::string& SST::Simulation::getOutputDirectory () [inline]`

Returns the output directory of the simulation

Returns

Directory in which simulation outputs are placed

6.209.4.17 `RankInfo SST::Simulation::getRank () const [inline]`

Get this instance's parallel rank

6.209.4.18 `static SharedRegionManager* SST::Simulation::getSharedRegionManager () [inline],[static]`

Returns the [Simulation's SharedRegionManager](#)

6.209.4.19 `static Simulation* SST::Simulation::getSimulation () [inline],[static]`

Return a pointer to the singleton instance of the [Simulation](#)

6.209.4.20 `Mode_t SST::Simulation::getSimulationMode () const [inline]`

Get the run mode of the simulation (e.g. init, run, both etc)

6.209.4.21 **static Output& SST::Simulation::getSimulationOutput ()** [inline],[static]

Return the base simulation [Output](#) class instance

6.209.4.22 **Statistics::StatisticProcessingEngine * SST::Simulation::getStatisticsProcessingEngine (void) const**

Return the Statistic Processing Engine associated with this [Simulation](#)

6.209.4.23 **static TimeLord* SST::Simulation::getTimeLord (void)** [inline],[static]

Return the [TimeLord](#) associated with this [Simulation](#)

6.209.4.24 **void SST::Simulation::initialize ()**

Perform the init() phase of simulation

6.209.4.25 **void SST::Simulation::insertActivity (SimTime_t time, Activity * ev)**

Insert an activity to fire at a specified time

6.209.4.26 **bool SST::Simulation::isWireUpFinished ()** [inline]

Returns true when the Wireup is finished.

6.209.4.27 **int SST::Simulation::performWireUp (ConfigGraph & graph, const RankInfo & myRank, SimTime_t min_part)**

Converts a [ConfigGraph](#) graph into actual set of links and components

6.209.4.28 **void SST::Simulation::printStatus (bool fullStatus)**

Causes the current status of the simulation to be printed to stderr.

Parameters

<i>fullStatus</i>	- if true, call printStatus() on all components as well as print the base Simulation 's status
-------------------	--

6.209.4.29 **void SST::Simulation::processGraphInfo (ConfigGraph & graph, const RankInfo & myRank, SimTime_t min_part)**

Processes the [ConfigGraph](#) to pull out any need information about relationships among the threads

6.209.4.30 **TimeConverter** * SST::Simulation::registerClock (const std::string & *freq*, Clock::HandlerBase * *handler*, int *priority*)

Register a handler to be called on a set frequency

6.209.4.31 **TimeConverter** * SST::Simulation::registerOneShot (std::string *timeDelay*, OneShot::HandlerBase * *handler*, int *priority*)

Register a [OneShot](#) event to be called after a time delay Note: [OneShot](#) cannot be canceled, and will always callback after the timedelay.

6.209.4.32 void SST::Simulation::requireEvent (std::string *name*)

Signifies that an event type is required for this simulation Causes to [Factory](#) to verify that the required event type can be found.

Parameters

<i>name</i>	fully qualified libraryName.EventName
-------------	---------------------------------------

6.209.4.33 **Cycle_t** SST::Simulation::reregisterClock (**TimeConverter** * *tc*, Clock::HandlerBase * *handler*, int *priority*)

Reactivate an existing clock and handler.

Returns

time when handler will next fire

6.209.4.34 void SST::Simulation::setOutputDirectory (std::string & *outDir*) [inline]

Set the output directory for this simulation

Parameters

<i>outDir</i>	Path of directory to place simulation outputs in
---------------	--

6.209.4.35 void SST::Simulation::setSignal (int *signal*) [static]

Sets an internal flag for signaling the simulation. Used internally

6.209.4.36 void SST::Simulation::setStopAtCycle (**Config** * *cfg*)

Set cycle count, which, if reached, will cause the simulation to halt.

6.209.4.37 void SST::Simulation::setup ()

Perform the [setup\(\)](#) and run phases of the simulation.

6.209.4.38 void SST::Simulation::shutdown () [static]

Used to signify the end of simulation. Cleans up any existing [Simulation](#) Objects

6.209.4.39 void SST::Simulation::unregisterClock (TimeConverter * tc, Clock::HandlerBase * handler, int priority)

Remove a clock handler from the list of active clock handlers

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

- sst/core/simulation.h
- sst/core/simulation.cc

6.210 SST::SimulatorHeartbeat Class Reference

```
#include <heartbeat.h>
```

Inheritance diagram for SST::SimulatorHeartbeat:

Collaboration diagram for SST::SimulatorHeartbeat:

Public Member Functions

- [SimulatorHeartbeat](#) (Config *cfg, int this_rank, [Simulation](#) *sim, [TimeConverter](#) *period)

Additional Inherited Members

6.210.1 Detailed Description

An optional heartbeat to show progress in a simulation

6.210.2 Constructor & Destructor Documentation

6.210.2.1 SST::SimulatorHeartbeat::SimulatorHeartbeat (Config * cfg, int this_rank, Simulation * sim, TimeConverter * period)

Create a new heartbeat object for the simulation core to show progress

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

- sst/core/heartbeat.h
- sst/core/heartbeat.cc

6.211 SST::ELI::SingleCtor< Base, Args > Struct Template Reference

Public Types

- template<class NewBase >
using **ChangeBase** = [SingleCtor](#)< NewBase, Args... >
- template<class NewCtor >
using **ExtendCtor** = [ExtendedCtor](#)< NewCtor, [SingleCtor](#)< Base, Args... >>

Static Public Member Functions

- template<class T >
static bool **add** ()

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

- sst/core/eli/elementbuilder.h

6.212 slist Struct Reference

Collaboration diagram for slist:

Public Attributes

- struct [slist](#) * **next**
- const void * **userdata**

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

- sst/core/libltdl/libltdl/slist.h

6.213 SST::SparseVectorMap< keyT, classT > Class Template Reference

Public Types

- typedef std::vector< classT >::iterator **iterator**
- typedef std::vector< classT >::const_iterator **const_iterator**

Public Member Functions

- void **push_back** (const classT &val)
- void **insert** (const classT &val)
- iterator **begin** ()
- iterator **end** ()
- const_iterator **begin** () const
- const_iterator **end** () const
- bool **contains** (keyT id) const
- classT & **operator[]** (keyT id)
- const classT & **operator[]** (keyT id) const
- void **clear** ()
- size_t **size** ()

Friends

- class **SST::Core::Serialization::serialize**< **SparseVectorMap**< keyT, classT > >
- class **ConfigGraph**

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

- sst/core/sparseVectorMap.h

6.214 SST::SparseVectorMap< keyT, keyT > Class Template Reference

Public Types

- typedef std::vector< keyT >::iterator **iterator**
- typedef std::vector< keyT >::const_iterator **const_iterator**

Public Member Functions

- void **push_back** (const keyT &val)
- void **insert** (const keyT &val)
- iterator **begin** ()
- iterator **end** ()
- const_iterator **begin** () const
- const_iterator **end** () const
- bool **contains** (keyT id)
- keyT & **operator[]** (keyT id)
- const keyT & **operator[]** (keyT id) const
- void **clear** ()
- size_t **size** ()

Friends

- class **SST::Core::Serialization::serialize**< **SparseVectorMap**< keyT, keyT > >
- class **ConfigGraph**

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

- sst/core/sparseVectorMap.h

6.215 SST::Core::ThreadSafe::Spinlock Class Reference

Public Member Functions

- void **lock** ()
- void **unlock** ()

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

- sst/core/threadsafe.h

6.216 SST::SST_ELI_element_version_extraction Struct Reference

Public Member Functions

- constexpr unsigned **getMajor** ()
- constexpr unsigned **getMinor** ()
- constexpr unsigned **getTertiary** ()

Public Attributes

- const unsigned **major**
- const unsigned **minor**
- const unsigned **tertiary**

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

- sst/core/eli/elementinfo.h

6.217 SST::RNG::SSTConstantDistribution Class Reference

```
#include "sst/core/rng/constant.h"
```

Inheritance diagram for SST::RNG::SSTConstantDistribution:

Collaboration diagram for SST::RNG::SSTConstantDistribution:

Public Member Functions

- [SSTConstantDistribution](#) (double v)
- [~SSTConstantDistribution](#) ()
- double [getNextDouble](#) ()
- double [getMean](#) ()

Protected Attributes

- double [mean](#)

6.217.1 Detailed Description

Implements a distribution which always returns a constant value (provided by the user). This can be used in situations where the user may not want to apply a distribution.

6.217.2 Constructor & Destructor Documentation

6.217.2.1 SST::RNG::SSTConstantDistribution::SSTConstantDistribution (double *v*) [\[inline\]](#)

Creates a constant distribution which returns a constant value.

Parameters

<i>v</i>	Is the constant value the user wants returned by the distribution
----------	---

6.217.2.2 SST::RNG::SSTConstantDistribution::~~SSTConstantDistribution () [\[inline\]](#)

Destroys the constant distribution

6.217.3 Member Function Documentation

6.217.3.1 double SST::RNG::SSTConstantDistribution::getMean () [\[inline\]](#)

Gets the constant value for the distribution

Returns

Constant value specified by the user when creating the class

6.217.3.2 double SST::RNG::SSTConstantDistribution::getNextDouble () [\[inline\]](#), [\[virtual\]](#)

Gets the next double for the distribution, in this case it will return the constant value specified by the user

Returns

Constant value specified by the user when creating the class

Implements [SST::RNG::SSTRandomDistribution](#).

6.217.4 Member Data Documentation

6.217.4.1 double SST::RNG::SSTConstantDistribution::mean [protected]

Describes the constant value to return from the distribution.

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

- sst/core/rng/constant.h

6.218 SST::RNG::SSTDiscreteDistribution Class Reference

```
#include "sst/core/rng/discrete.h"
```

Inheritance diagram for SST::RNG::SSTDiscreteDistribution:

Collaboration diagram for SST::RNG::SSTDiscreteDistribution:

Public Member Functions

- [SSTDiscreteDistribution](#) (const double *probs, const uint32_t probsCount)
- [SSTDiscreteDistribution](#) (const double *probs, const uint32_t probsCount, [SSTRandom](#) *baseDist)
- [~SSTDiscreteDistribution](#) ()
- double [getNextDouble](#) ()

Protected Attributes

- [SSTRandom](#) * [baseDistrib](#)
- bool [deleteDistrib](#)
- double * [probabilities](#)
- uint32_t [probCount](#)

6.218.1 Detailed Description

Creates a discrete distribution for use within SST. This distribution is the same across platforms and compilers.

6.218.2 Constructor & Destructor Documentation

6.218.2.1 SST::RNG::SSTDiscreteDistribution::SSTDiscreteDistribution (const double * *probs*, const uint32_t *probsCount*) [inline]

Creates an exponential distribution with a specific lambda

Parameters

<i>lambda</i>	The lambda of the exponential distribution
---------------	--

6.218.2.2 `SST::RNG::SSTDiscreteDistribution::SSTDiscreteDistribution (const double * probs, const uint32_t probsCount, SSTRandom * baseDist) [inline]`

Creates an exponential distribution with a specific lambda and a base random number generator

Parameters

<i>lambda</i>	The lambda of the exponential distribution
<i>baseDist</i>	The base random number generator to take the distribution from.

6.218.2.3 `SST::RNG::SSTDiscreteDistribution::~~SSTDiscreteDistribution () [inline]`

Destroys the exponential distribution

6.218.3 Member Function Documentation

6.218.3.1 `double SST::RNG::SSTDiscreteDistribution::getNextDouble () [inline], [virtual]`

Gets the next (random) double value in the distribution

Returns

The next random double from the discrete distribution, this is the double converted of the index where the probability is located

Implements [SST::RNG::SSTRandomDistribution](#).

6.218.4 Member Data Documentation

6.218.4.1 `SSTRandom* SST::RNG::SSTDiscreteDistribution::baseDistrib [protected]`

Sets the base random number generator for the distribution.

6.218.4.2 `bool SST::RNG::SSTDiscreteDistribution::deleteDistrib [protected]`

Controls whether the base distribution should be deleted when this class is destructed.

6.218.4.3 `double* SST::RNG::SSTDiscreteDistribution::probabilities [protected]`

The discrete probability list

6.218.4.4 `uint32_t SST::RNG::SSTDiscreteDistribution::probCount` `[protected]`

Count of discrete probabilities

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

- `sst/core/rng/discrete.h`

6.219 SST::SSTElementPythonModule Class Reference

```
#include <element_python.h>
```

Inheritance diagram for SST::SSTElementPythonModule:

Collaboration diagram for SST::SSTElementPythonModule:

Public Member Functions

- [SSTElementPythonModule](#) (std::string library)
Constructor for [SSTElementPythonModule](#). Must be called by derived class.
- `__attribute__ ((deprecated("Support for addPrimaryModule will be removed in version 9.0. Please use createPrimaryModule(.)."))))` void addPrimaryModule(char *file)
- [SSTElementPythonModuleCode](#) * [createPrimaryModule](#) (char *code=NULL, std::string filename="")

Protected Attributes

- std::string **library**
- std::string **pylibrary**
- std::string **sstlibrary**
- char * **primary_module**
- std::vector< std::pair< std::string, char * > > **sub_modules**
- [SSTElementPythonModuleCode](#) * **primary_code_module**

6.219.1 Detailed Description

Base class for python modules in element libraries.

Element libraries can include a class derived from this class and create a python module hierarchy.

6.219.2 Constructor & Destructor Documentation

6.219.2.1 `SST::SSTElementPythonModule::SSTElementPythonModule (std::string library)`

Constructor for [SSTElementPythonModule](#). Must be called by derived class.

Parameters

<i>library</i>	name of the element library the module is part of. Primary module name will be sst.library and submodules under this can also be created.
----------------	---

6.219.3 Member Function Documentation

6.219.3.1 SSTELEMENTPythonModuleCode * SST::SSTELEMENTPythonModule::createPrimaryModule (char * *code* = NULL, std::string *filename* = " ")

Create the top level python module (i.e. the one named sst.library) Python files will need to be turned into a char array (code parameter). One way to do this is to use the following in your Makefile:

```
%.inc: %.py
///      od -v -t x1 < $< | sed -e 's/^[^ ]*[ ]*/g' -e '/^\s*$$/d' -e 's/\([0-9a-f]*\)[ $$]*/0x\1,/g' > $@
```

Parameters

<i>code</i>	code to be compiled
<i>filename</i>	filename used when reporting errors

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

- sst/core/model/element_python.h
- sst/core/model/element_python.cc

6.220 SST::SSTELEMENTPythonModuleCode Class Reference

```
#include <element_python.h>
```

Public Member Functions

- [SSTELEMENTPythonModuleCode * addSubModule](#) (const std::string &module_name, char *code=NULL, std::string filename="")
- std::string [getFullModuleName](#) ()
Get the full name of the module.

Friends

- class **SSTELEMENTPythonModule**

6.220.1 Detailed Description

Class to represent the code that needs to be added to create the python module struture for the library.

6.220.2 Member Function Documentation

6.220.2.1 SSTElementPythonModuleCode * SST::SSTElementPythonModuleCode::addSubModule (const std::string & module_name, char * code = NULL, std::string filename = " ")

Add a submodule to the module

Python files will need to be turned into a char array (code parameter). One way to do this is to use the following in your Makefile:

```
%.inc: %.py
///      od -v -t x1 < $< | sed -e 's/^[^ ]*[ ]*//g' -e '/^\s*$$/d' -e 's/\([0-9a-f]*\)[ $$]*/0x\1,/g' > $@
```

Parameters

<i>module_name</i>	simple name of the module
<i>code</i>	code to be compiled
<i>filename</i>	filename used when reporting errors

6.220.2.2 std::string SST::SSTElementPythonModuleCode::getFullModuleName ()

Get the full name of the module.

Get the full name of the module formatted as parent_full_name.module_name.

Returns

full name of module as a string

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

- sst/core/model/element_python.h
- sst/core/model/element_python.cc

6.221 SST::SSTElementPythonModuleOldELI Class Reference

Inheritance diagram for SST::SSTElementPythonModuleOldELI:

Collaboration diagram for SST::SSTElementPythonModuleOldELI:

Public Member Functions

- **SSTElementPythonModuleOldELI** (const std::string &lib, genPythonModuleFunction func)
- void * **load** () override

Additional Inherited Members

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

- `sst/core/model/element_python.h`

6.222 SST::RNG::SSTExponentialDistribution Class Reference

```
#include "sst/core/rng/expon.h"
```

Inheritance diagram for SST::RNG::SSTExponentialDistribution:

Collaboration diagram for SST::RNG::SSTExponentialDistribution:

Public Member Functions

- [SSTExponentialDistribution](#) (const double mn)
- [SSTExponentialDistribution](#) (const double mn, [SSTRandom](#) *baseDist)
- [~SSTExponentialDistribution](#) ()
- double [getNextDouble](#) ()
- double [getLambda](#) ()

Protected Attributes

- double [lambda](#)
- [SSTRandom](#) * [baseDistrib](#)
- bool [deleteDistrib](#)

6.222.1 Detailed Description

Creates an exponential distribution for use within SST. This distribution is the same across platforms and compilers.

6.222.2 Constructor & Destructor Documentation

6.222.2.1 SST::RNG::SSTExponentialDistribution::SSTExponentialDistribution (const double *mn*) `[inline]`

Creates an exponential distribution with a specific lambda

Parameters

<i>mn</i>	The lambda of the exponential distribution
-----------	--

6.222.2.2 `SST::RNG::SSTExponentialDistribution::SSTExponentialDistribution (const double mn, SSTRandom * baseDist)`
`[inline]`

Creates an exponential distribution with a specific lambda and a base random number generator

Parameters

<i>mn</i>	The lambda of the exponential distribution
<i>baseDist</i>	The base random number generator to take the distribution from.

6.222.2.3 `SST::RNG::SSTExponentialDistribution::~~SSTExponentialDistribution ()` `[inline]`

Destroys the exponential distribution

6.222.3 Member Function Documentation

6.222.3.1 `double SST::RNG::SSTExponentialDistribution::getLambda ()` `[inline]`

Gets the lambda with which the distribution was created

Returns

The lambda which the user created the distribution with

6.222.3.2 `double SST::RNG::SSTExponentialDistribution::getNextDouble ()` `[inline]`, `[virtual]`

Gets the next (random) double value in the distribution

Returns

The next random double from the distribution

Implements [SST::RNG::SSTRandomDistribution](#).

6.222.4 Member Data Documentation

6.222.4.1 `SSTRandom* SST::RNG::SSTExponentialDistribution::baseDistrib` `[protected]`

Sets the base random number generator for the distribution.

6.222.4.2 `bool SST::RNG::SSTExponentialDistribution::deleteDistrib` `[protected]`

Controls whether the base distribution should be deleted when this class is destructed.

6.222.4.3 `double SST::RNG::SSTExponentialDistribution::lambda` `[protected]`

Sets the lambda of the exponential distribution.

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

- `sst/core/rng/expon.h`

6.223 SST::RNG::SSTGaussianDistribution Class Reference

```
#include "sst/core/rng/gaussian.h"
```

Inheritance diagram for SST::RNG::SSTGaussianDistribution:

Collaboration diagram for SST::RNG::SSTGaussianDistribution:

Public Member Functions

- [SSTGaussianDistribution](#) (double mn, double sd)
- [SSTGaussianDistribution](#) (double mn, double sd, [SSTRandom](#) *baseRNG)
- [~SSTGaussianDistribution](#) ()
- double [getNextDouble](#) ()
- double [getMean](#) ()
- double [getStandardDev](#) ()

Protected Attributes

- double [mean](#)
- double [stddev](#)
- [SSTRandom](#) * [baseDistrib](#)
- double [unusedPair](#)
- bool [usePair](#)
- bool [deleteDistrib](#)

6.223.1 Detailed Description

Creates a Gaussian (normal) distribution for which to sample

6.223.2 Constructor & Destructor Documentation

6.223.2.1 `SST::RNG::SSTGaussianDistribution::SSTGaussianDistribution (double mn, double sd)` `[inline]`

Creates a new distribution with a predefined random number generator with a specified mean and standard deviation.

Parameters

<i>mn</i>	The mean of the Gaussian distribution
<i>sd</i>	The standard deviation of the Gaussian distribution

6.223.2.2 `SST::RNG::SSTGaussianDistribution::SSTGaussianDistribution (double mn, double sd, SSTRandom * baseRNG) [inline]`

Creates a new distribution with a predefined random number generator with a specified mean and standard deviation.

Parameters

<i>mn</i>	The mean of the Gaussian distribution
<i>sd</i>	The standard deviation of the Gaussian distribution
<i>baseRNG</i>	The random number generator as the base of the distribution

6.223.2.3 `SST::RNG::SSTGaussianDistribution::~~SSTGaussianDistribution () [inline]`

Destroys the Gaussian distribution.

6.223.3 Member Function Documentation

6.223.3.1 `double SST::RNG::SSTGaussianDistribution::getMean () [inline]`

Gets the mean of the distribution

Returns

The mean of the Gaussian distribution

6.223.3.2 `double SST::RNG::SSTGaussianDistribution::getNextDouble () [inline],[virtual]`

Gets the next double value in the distribution

Returns

The next double value of the distribution (in this case a Gaussian distribution)

Implements [SST::RNG::SSTRandomDistribution](#).

6.223.3.3 `double SST::RNG::SSTGaussianDistribution::getStandardDev () [inline]`

Gets the standard deviation of the distribution

Returns

The standard deviation of the Gaussian distribution

6.223.4 Member Data Documentation

6.223.4.1 `SSTRandom*` `SST::RNG::SSTGaussianDistribution::baseDistrib` [protected]

The base random number generator for the distribution

6.223.4.2 `bool` `SST::RNG::SSTGaussianDistribution::deleteDistrib` [protected]

Controls whether the destructor deletes the distribution (we need to ensure we do this IF we created the distribution)

6.223.4.3 `double` `SST::RNG::SSTGaussianDistribution::mean` [protected]

The mean of the Gaussian distribution

6.223.4.4 `double` `SST::RNG::SSTGaussianDistribution::stddev` [protected]

The standard deviation of the Gaussian distribution

6.223.4.5 `double` `SST::RNG::SSTGaussianDistribution::unusedPair` [protected]

Random numbers for the distribution are read in pairs, this stores the second of the pair

6.223.4.6 `bool` `SST::RNG::SSTGaussianDistribution::usePair` [protected]

Random numbers for the distribution are read in pairs, this tells the code to use the second of the pair

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

- `sst/core/rng/gaussian.h`

6.224 SST::SSTInfoConfig Class Reference

```
#include <sstinfo.h>
```

Public Types

- `typedef std::multimap< std::string, std::string > FilterMap_t`

Public Member Functions

- [SSTInfoConfig](#) ()
- int [parseCmdLine](#) (int argc, char *argv[])
- std::set< std::string > [getElementsToProcessArray](#) ()
- FilterMap_t & [getFilterMap](#) ()
- unsigned int [getOptionBits](#) ()
- std::string & [getXMLFilePath](#) ()
- bool [debugEnabled](#) () const
- bool [processAllElements](#) () const
- bool [doVerbose](#) () const

6.224.1 Detailed Description

The SSTInfo Configuration class.

This class will parse the command line, and setup internal lists of elements and components to be processed.

6.224.2 Constructor & Destructor Documentation

6.224.2.1 SSTInfoConfig::SSTInfoConfig ()

Create a new SSTInfo configuration and parse the Command Line.

6.224.3 Member Function Documentation

6.224.3.1 bool SST::SSTInfoConfig::debugEnabled () const [inline]

Is debugging output enabled?

6.224.3.2 std::set<std::string> SST::SSTInfoConfig::getElementsToProcessArray () [inline]

Return the list of elements to be processed.

6.224.3.3 FilterMap_t& SST::SSTInfoConfig::getFilterMap () [inline]

Return the filter map

6.224.3.4 unsigned int SST::SSTInfoConfig::getOptionBits () [inline]

Return the bit field of various command line options enabled.

6.224.3.5 std::string& SST::SSTInfoConfig::getXMLFilePath () [inline]

Return the user defined path the XML File.

6.224.3.6 int SSTInfoConfig::parseCmdLine (int argc, char * argv[])

Parse the Command Line.

Parameters

<i>argc</i>	The number of arguments passed to the application
<i>argv</i>	The array of arguments

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

- sst/core/sstinfo.h
- sst/core/sstinfo.cc

6.225 SST::SSTLibraryInfo Class Reference

```
#include <sstinfo.h>
```

Public Member Functions

- [SSTLibraryInfo](#) (const std::string &name)
- std::string [getLibraryName](#) ()
- void [outputHumanReadable](#) (std::ostream &os, int LibIndex)
- void [outputXML](#) (int Index, [TiXmlNode](#) *XMLParentElement)
- template<class BaseType >
void **outputHumanReadable** (std::ostream &os, bool printAll)
- template<class BaseType >
void **outputXML** ([TiXmlElement](#) *node)
- std::string [getLibraryDescription](#) ()

6.225.1 Detailed Description

The SSTInfo representation of ElementLibraryInfo object.

This class is used internally by SSTInfo to load and process ElementLibraryInfo objects.

6.225.2 Constructor & Destructor Documentation

6.225.2.1 SST::SSTLibraryInfo::SSTLibraryInfo (const std::string & *name*) [inline]

Create a new SSTInfoElement_LibraryInfo object.

Parameters

<i>eli</i>	Pointer to an ElementLibraryInfo object.
------------	--

6.225.3 Member Function Documentation

6.225.3.1 `std::string SST::SSTLibraryInfo::getLibraryName () [inline]`

Return the Name of the Library.

6.225.3.2 `void SSTLibraryInfo::outputHumanReadable (std::ostream & os, int LibIndex)`

Output the Library Information.

Parameters

<i>LibIndex</i>	The Index of the Library.
-----------------	---------------------------

6.225.3.3 `void SSTLibraryInfo::outputXML (int Index, TiXmlNode * XMLParentElement)`

Create the formatted XML data of the Library.

Parameters

<i>LibIndex</i>	The Index of the Library.
<i>XMLParentElement</i>	The parent element to receive the XML data.

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

- sst/core/sstinfo.h
- sst/core/sstinfo.cc

6.226 SST::IMPL::Partition::SSTLinearPartition Class Reference

```
#include <linpart.h>
```

Inheritance diagram for SST::IMPL::Partition::SSTLinearPartition:

Collaboration diagram for SST::IMPL::Partition::SSTLinearPartition:

Public Member Functions

- [SSTLinearPartition](#) ([RankInfo](#) rankCount, [RankInfo](#) my_rank, int verbosity)
- void [performPartition](#) ([PartitionGraph](#) *graph) override
- void [performPartition](#) ([ConfigGraph](#) *graph) override
- bool [requiresConfigGraph](#) () override
- bool [spawnOnAllRanks](#) () override

Public Attributes

- SST_ELI_REGISTER_PARTITIONER([SSTLinearPartition](#),"sst","linear", SST_ELI_ELEMENT_VERSION(1, 0, 0),"Partitions components by dividing [Component](#) ID space into roughly equal portions. Components with sequential IDs will be placed close together.") protected [Output](#) * [partOutput](#)

6.226.1 Detailed Description

Performs a linear partition scheme of an SST simulation configuration. In this scheme a list of components (supplied as a graph) are grouped by slicing the list into approximately equal parts. A "part" is generated for each MPI rank performing the simulation. This means Components with sequential ids will be placed close together. In general this scheme provides a very effective partition for most parallel simulations which generate man similar components of interest close together in the input Python configuration. It is also very fast to compute a linear partition scheme. For more aggressive partition schemes users should try either a simple or Zoltan-based partitioner.

6.226.2 Constructor & Destructor Documentation

6.226.2.1 SSTLinearPartition::SSTLinearPartition (RankInfo rankCount, RankInfo my_rank, int verbosity)

Creates a new linear partition scheme.

Parameters

<i>mpiRankCount</i>	Number of MPI ranks in the simulation
<i>verbosity</i>	The level of information to output

6.226.3 Member Function Documentation

6.226.3.1 void SSTLinearPartition::performPartition (PartitionGraph * graph) [override],[virtual]

Performs a partition of an SST simulation configuration

Parameters

<i>graph</i>	The simulation configuration to partition
--------------	---

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.226.3.2 void SST::IMPL::Partition::SSTLinearPartition::performPartition (ConfigGraph * graph) [inline],[override],[virtual]

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [ConfigGraph](#) object. The consequence of using ConfigGraphs is that no-cut links are not supported.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.226.4 Member Data Documentation

- 6.226.4.1 SST_ELI_REGISTER_PARTITIONER (SSTLinearPartition, "sst", "linear", SST_ELI_ELEMENT_VERSION(1,0,0), "Partitions components by dividing Component ID space into roughly equal portions. Components with sequential IDs will be placed close together.") protected Output* SST::IMPL::Partition::SSTLinearPartition::partOutput

Number of ranks in the simulation [Output](#) object to print partitioning information

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

- sst/core/impl/partitioners/linpart.h
- sst/core/impl/partitioners/linpart.cc

6.227 SST::sstLongOpts_s Struct Reference

Public Attributes

- struct option **opt**
- const char * **argName**
- const char * **desc**
- Config::flagFunction **flagFunc**
- Config::argFunction **argFunc**

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

- sst/core/config.cc

6.228 SST::SSTModelDescription Class Reference

```
#include <sstmodel.h>
```

Public Member Functions

- virtual [ConfigGraph](#) * [createConfigGraph](#) ()=0

6.228.1 Detailed Description

Base class for Model Generation

6.228.2 Member Function Documentation

6.228.2.1 virtual `ConfigGraph*` `SST::SSTModelDescription::createConfigGraph ()` `[pure virtual]`

Create the [ConfigGraph](#)

This function should be overridden by subclasses.

This function is responsible for reading any configuration files and generating a [ConfigGraph](#) object.

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

- `sst/core/model/sstmodel.h`
- `sst/core/model/sstmodel.cc`

6.229 SST::Core::Interprocess::SSTMutex Class Reference

Public Member Functions

- void **processorPause** (int currentCount)
- void **lock** ()
- void **unlock** ()
- bool **try_lock** ()

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

- `sst/core/interprocess/sstmutex.h`

6.230 SST::Partition::SSTPartitioner Class Reference

```
#include <sstpart.h>
```

Inheritance diagram for `SST::Partition::SSTPartitioner`:

Public Member Functions

- virtual void [performPartition](#) ([PartitionGraph](#) *UNUSED(graph))
- virtual void [performPartition](#) ([ConfigGraph](#) *UNUSED(graph))
- virtual bool **requiresConfigGraph** ()
- virtual bool **spawnOnAllRanks** ()
- virtual void [performPartition](#) ([PartitionGraph](#) *graph)
- virtual void [performPartition](#) ([ConfigGraph](#) *graph)
- virtual bool **requiresConfigGraph** ()
- virtual bool **spawnOnAllRanks** ()

6.230.1 Detailed Description

Base class for Partitioning graphs

6.230.2 Member Function Documentation

6.230.2.1 `void SST::Partition::SSTPartitioner::performPartition (PartitionGraph * UNUSEDgraph) [inline],
[virtual]`

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [PartitionGraph](#) object.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented in [SST::IMPL::Partition::SSTSelfPartition](#).

6.230.2.2 `virtual void SST::Partition::SSTPartitioner::performPartition (PartitionGraph * graph) [virtual]`

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [PartitionGraph](#) object.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented in [SST::IMPL::Partition::SSTLinearPartition](#), [SST::IMPL::Partition::SimplePartitioner](#), [SST::IMPL::Partition::SSTSinglePartition](#), and [SST::IMPL::Partition::SSTRoundRobinPartition](#).

6.230.2.3 `void SST::Partition::SSTPartitioner::performPartition (ConfigGraph * UNUSEDgraph) [inline],
[virtual]`

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [ConfigGraph](#) object. The consequence of using ConfigGraphs is that no-cut links are not supported.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented in [SST::IMPL::Partition::SSTSelfPartition](#).

6.230.2.4 `virtual void SST::Partition::SSTPartitioner::performPartition (ConfigGraph * graph) [virtual]`

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [ConfigGraph](#) object. The consequence of using ConfigGraphs is that no-cut links are not supported.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented in [SST::IMPL::Partition::SSTLinearPartition](#), [SST::IMPL::Partition::SimplePartitioner](#), [SST::IMPL::Partition::SSTSinglePartition](#), and [SST::IMPL::Partition::SSTRoundRobinPartition](#).

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

- `sst/core/part/sstpart.h`
- `sst/core/sstpart.cc`

6.231 SST::RNG::SSTPoissonDistribution Class Reference

```
#include "sst/core/rng/poisson.h"
```

Inheritance diagram for SST::RNG::SSTPoissonDistribution:

Collaboration diagram for SST::RNG::SSTPoissonDistribution:

Public Member Functions

- [SSTPoissonDistribution](#) (const double mn)
- [SSTPoissonDistribution](#) (const double mn, [SSTRandom](#) *baseDist)
- [~SSTPoissonDistribution](#) ()
- double [getNextDouble](#) ()
- double [getLambda](#) ()

Protected Attributes

- const double [lambda](#)
- [SSTRandom](#) * [baseDistrib](#)
- bool [deleteDistrib](#)

6.231.1 Detailed Description

Creates an Poisson distribution for use within SST. This distribution is the same across platforms and compilers.

6.231.2 Constructor & Destructor Documentation

6.231.2.1 `SST::RNG::SSTPoissonDistribution::SSTPoissonDistribution (const double mn) [inline]`

Creates an Poisson distribution with a specific lambda

Parameters

<i>mn</i>	The lambda of the Poisson distribution
-----------	--

6.231.2.2 `SST::RNG::SSTPoissonDistribution::SSTPoissonDistribution (const double mn, SSTRandom * baseDist) [inline]`

Creates an Poisson distribution with a specific lambda and a base random number generator

Parameters

<i>lambda</i>	The lambda of the Poisson distribution
<i>baseDist</i>	The base random number generator to take the distribution from.

6.231.2.3 SST::RNG::SSTPoissonDistribution::~~SSTPoissonDistribution () [inline]

Destroys the Poisson distribution

6.231.3 Member Function Documentation

6.231.3.1 double SST::RNG::SSTPoissonDistribution::getLambda () [inline]

Gets the lambda with which the distribution was created

Returns

The lambda which the user created the distribution with

6.231.3.2 double SST::RNG::SSTPoissonDistribution::getNextDouble () [inline],[virtual]

Gets the next (random) double value in the distribution

Returns

The next random double from the distribution

Implements [SST::RNG::SSTRandomDistribution](#).

6.231.4 Member Data Documentation

6.231.4.1 SSTRandom* SST::RNG::SSTPoissonDistribution::baseDistrib [protected]

Sets the base random number generator for the distribution.

6.231.4.2 bool SST::RNG::SSTPoissonDistribution::deleteDistrib [protected]

Controls whether the base distribution should be deleted when this class is destructed.

6.231.4.3 const double SST::RNG::SSTPoissonDistribution::lambda [protected]

Sets the lambda of the Poisson distribution.

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

- `sst/core/rng/poisson.h`

6.232 SST::RNG::SSTRandom Class Reference

```
#include "sst/core/rng/sstrng.h"
```

Inheritance diagram for SST::RNG::SSTRandom:

Public Member Functions

- virtual double [nextUniform](#) ()=0
- virtual uint32_t [generateNextUInt32](#) ()=0
- virtual uint64_t [generateNextUInt64](#) ()=0
- virtual int64_t [generateNextInt64](#) ()=0
- virtual int32_t [generateNextInt32](#) ()=0
- virtual [~SSTRandom](#) ()

6.232.1 Detailed Description

Implements the base class for random number generators for the SST core. This does not implement an actual RNG itself only the base class which describes the methods each class will implement.

6.232.2 Constructor & Destructor Documentation

6.232.2.1 virtual SST::RNG::SSTRandom::~~SSTRandom () [inline],[virtual]

Destroys the random number generator

6.232.3 Member Function Documentation

6.232.3.1 virtual int32_t SST::RNG::SSTRandom::generateNextInt32 () [pure virtual]

Generates the next random number as a signed 32-bit integer

Implemented in [SST::RNG::MarsagliaRNG](#), [SST::RNG::XORShiftRNG](#), and [SST::RNG::MersenneRNG](#).

6.232.3.2 virtual int64_t SST::RNG::SSTRandom::generateNextInt64 () [pure virtual]

Generates the next random number as a signed 64-bit integer.

Implemented in [SST::RNG::MarsagliaRNG](#), [SST::RNG::XORShiftRNG](#), and [SST::RNG::MersenneRNG](#).

6.232.3.3 virtual uint32_t SST::RNG::SSTRandom::generateNextUInt32 () [pure virtual]

Generates the next random number as an unsigned 32-bit integer.

Implemented in [SST::RNG::MarsagliaRNG](#), [SST::RNG::XORShiftRNG](#), and [SST::RNG::MersenneRNG](#).

6.232.3.4 `virtual uint64_t SST::RNG::SSTRandom::generateNextUInt64 () [pure virtual]`

Generates the next random number as an unsigned 64-bit integer.

Implemented in [SST::RNG::MarsagliaRNG](#), [SST::RNG::XORShiftRNG](#), and [SST::RNG::MersenneRNG](#).

6.232.3.5 `virtual double SST::RNG::SSTRandom::nextUniform () [pure virtual]`

Generates the next random number in the range 0 to 1.

Implemented in [SST::RNG::MarsagliaRNG](#), [SST::RNG::XORShiftRNG](#), and [SST::RNG::MersenneRNG](#).

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

- `sst/core/rng/ssrng.h`

6.233 SST::RNG::SSTRandomDistribution Class Reference

```
#include <distrib.h>
```

Inheritance diagram for SST::RNG::SSTRandomDistribution:

Public Member Functions

- virtual double [getNextDouble](#) ()=0
- virtual [~SSTRandomDistribution](#) ()
- [SSTRandomDistribution](#) ()

6.233.1 Detailed Description

Base class of statistical distributions in SST.

6.233.2 Constructor & Destructor Documentation

6.233.2.1 `virtual SST::RNG::SSTRandomDistribution::~~SSTRandomDistribution () [inline],[virtual]`

Destroys the distribution

6.233.2.2 `SST::RNG::SSTRandomDistribution::SSTRandomDistribution () [inline]`

Creates the base (abstract) class of a distribution

6.233.3 Member Function Documentation

6.233.3.1 virtual double SST::RNG::SSTRandomDistribution::getNextDouble () [pure virtual]

Obtains the next double from the distribution

Returns

The next double in the distribution being sampled

Implemented in [SST::RNG::SSTDiscreteDistribution](#), [SST::RNG::SSTGaussianDistribution](#), [SST::RNG::SSTUniformDistribution](#), [SST::RNG::SSTExponentialDistribution](#), [SST::RNG::SSTPoissonDistribution](#), and [SST::RNG::SSTConstantDistribution](#).

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

- `sst/core/rng/distrib.h`

6.234 SST::IMPL::Partition::SSTRoundRobinPartition Class Reference

Inheritance diagram for SST::IMPL::Partition::SSTRoundRobinPartition:

Collaboration diagram for SST::IMPL::Partition::SSTRoundRobinPartition:

Public Member Functions

- **SSTRoundRobinPartition** ([RankInfo](#) world_size, [RankInfo](#) my_rank, int verbosity)
- void [performPartition](#) ([PartitionGraph](#) *graph) override
- void [performPartition](#) ([ConfigGraph](#) *graph) override
- bool **requiresConfigGraph** () override
- bool **spawnOnAllRanks** () override

6.234.1 Member Function Documentation

6.234.1.1 void SST::IMPL::Partition::SSTRoundRobinPartition::performPartition ([PartitionGraph](#) * graph) [override], [virtual]

Performs a partition of an SST simulation configuration

Parameters

<i>graph</i>	The simulation configuration to partition
--------------	---

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.234.1.2 void SST::IMPL::Partition::SSTRoundRobinPartition::performPartition (ConfigGraph * *graph*) [inline],
[override], [virtual]

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [ConfigGraph](#) object. The consequence of using ConfigGraphs is that no-cut links are not supported.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented from [SST::Partition::SSTPartitioner](#).

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

- sst/core/impl/partitioners/rrobin.h
- sst/core/impl/partitioners/rrobin.cc

6.235 SST::IMPL::Partition::SSTSelfPartition Class Reference

```
#include <selfpart.h>
```

Inheritance diagram for SST::IMPL::Partition::SSTSelfPartition:

Collaboration diagram for SST::IMPL::Partition::SSTSelfPartition:

Public Member Functions

- [SST_ELI_REGISTER_PARTITIONER](#) (SSTSelfPartition, "sst", "self", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Used when partitioning is already specified in the configuration file.") SSTSelfPartition([RankInfo](#) UN←USED(total_ranks)
- [RankInfo](#) **UNUSED** (my_rank)
- [RankInfo](#) int **UNUSED** (verbosity)
- void [performPartition](#) ([ConfigGraph](#) *UNUSED(graph)) override
- void [performPartition](#) ([PartitionGraph](#) *UNUSED(graph)) override
- bool **requiresConfigGraph** () override
- bool **spawnOnAllRanks** () override

6.235.1 Detailed Description

Self partitioner actually does nothing. It is simply a pass through for graphs which have been partitioned during graph creation.

6.235.2 Member Function Documentation

6.235.2.1 void SST::IMPL::Partition::SSTSelfPartition::performPartition (ConfigGraph * *UNUSEDgraph*) [inline],
[override], [virtual]

Performs a partition of an SST simulation configuration

Parameters

<i>graph</i>	The simulation configuration to partition
--------------	---

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.235.2.2 `void SST::IMPL::Partition::SSTSelfPartition::performPartition (PartitionGraph * UNUSEDgraph) [inline], [override], [virtual]`

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [PartitionGraph](#) object.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.235.2.3 `SST::IMPL::Partition::SSTSelfPartition::SST_ELI_REGISTER_PARTITIONER (SSTSelfPartition , "sst" , "self" , SST_ELI_ELEMENT_VERSION(1, 0, 0) , "Used when partitioning is already specified in the configuration file.")`

Creates a new self partition scheme.

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

- `sst/core/impl/partitioners/selfpart.h`

6.236 SST::IMPL::Partition::SSTSinglePartition Class Reference

```
#include <singlepart.h>
```

Inheritance diagram for `SST::IMPL::Partition::SSTSinglePartition`:

Collaboration diagram for `SST::IMPL::Partition::SSTSinglePartition`:

Public Member Functions

- [SST_ELI_REGISTER_PARTITIONER](#) ([SSTSinglePartition](#), "sst", "single", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Allocates all components to rank 0. Automatically selected for serial jobs.") [SSTSinglePartition](#)([RankInfo](#) total_ranks)
- void [performPartition](#) ([PartitionGraph](#) *graph) override
- void [performPartition](#) ([ConfigGraph](#) *graph) override
- bool [requiresConfigGraph](#) () override
- bool [spawnOnAllRanks](#) () override

Public Attributes

- [RankInfo](#) **my_rank**
- [RankInfo](#) int **verbosity**

6.236.1 Detailed Description

Single partitioner is a virtual partitioner used for serial jobs. It simply ensures that all components are assigned to rank 0.

6.236.2 Member Function Documentation

6.236.2.1 `void SSTSinglePartition::performPartition (PartitionGraph * graph)` `[override],[virtual]`

Performs a partition of an SST simulation configuration

Parameters

<i>graph</i>	The simulation configuration to partition
--------------	---

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.236.2.2 `void SST::IMPL::Partition::SSTSinglePartition::performPartition (ConfigGraph * graph)` `[inline],[override],[virtual]`

Function to be overridden by subclasses

Performs the partitioning of the Graph using the [ConfigGraph](#) object. The consequence of using ConfigGraphs is that no-cut links are not supported.

Result of this function is that every [ConfigComponent](#) in graph has a Rank applied to it.

Reimplemented from [SST::Partition::SSTPartitioner](#).

6.236.2.3 `SST::IMPL::Partition::SSTSinglePartition::SST_ELI_REGISTER_PARTITIONER (SSTSinglePartition , "sst" , "single" , SST_ELI_ELEMENT_VERSION(1, 0, 0) , "Allocates all components to rank 0. Automatically selected for serial jobs.")`

Creates a new single partition scheme.

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

- `sst/core/impl/partitioners/singlepart.h`
- `sst/core/impl/partitioners/singlepart.cc`

6.237 SST::RNG::SSTUniformDistribution Class Reference

```
#include "sst/core/rng/uniform.h"
```

Inheritance diagram for SST::RNG::SSTUniformDistribution:

Collaboration diagram for SST::RNG::SSTUniformDistribution:

Public Member Functions

- [SSTUniformDistribution](#) (const uint32_t probsCount)
- [SSTUniformDistribution](#) (const uint32_t probsCount, [SSTRandom](#) *baseDist)
- [~SSTUniformDistribution](#) ()
- double [getNextDouble](#) ()

Protected Attributes

- [SSTRandom](#) * [baseDistrib](#)
- bool [deleteDistrib](#)
- uint32_t [probCount](#)

6.237.1 Detailed Description

Creates a Uniform distribution for use within SST. This distribution is the same across platforms and compilers.

6.237.2 Constructor & Destructor Documentation

6.237.2.1 `SST::RNG::SSTUniformDistribution::SSTUniformDistribution (const uint32_t probsCount)` `[inline]`

Creates an uniform distribution with a specific number of bins

Parameters

<i>probsCount</i>	Number of probability bins in this distribution
-------------------	---

6.237.2.2 `SST::RNG::SSTUniformDistribution::SSTUniformDistribution (const uint32_t probsCount, SSTRandom * baseDist)` `[inline]`

Creates a Uniform distribution with a specific number of bins and user supplied random number generator

Parameters

<i>probsCount</i>	Number of probability bins in the distribution
<i>baseDist</i>	The base random number generator to take the distribution from.

6.237.2.3 `SST::RNG::SSTUniformDistribution::~~SSTUniformDistribution ()` `[inline]`

Destroys the distribution and will delete locally allocated RNGs

6.237.3 Member Function Documentation

6.237.3.1 `double SST::RNG::SSTUniformDistribution::getNextDouble () [inline], [virtual]`

Gets the next (random) double value in the distribution

Returns

The next random double from the distribution, this is the double converted of the index where the probability is located

Implements [SST::RNG::SSTRandomDistribution](#).

6.237.4 Member Data Documentation

6.237.4.1 `SSTRandom* SST::RNG::SSTUniformDistribution::baseDistrib [protected]`

Sets the base random number generator for the distribution.

6.237.4.2 `bool SST::RNG::SSTUniformDistribution::deleteDistrib [protected]`

Controls whether the base distribution should be deleted when this class is destructed.

6.237.4.3 `uint32_t SST::RNG::SSTUniformDistribution::probCount [protected]`

Count of discrete probabilities

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

- `sst/core/rng/uniform.h`

6.238 StatGroupPy_t Struct Reference

Collaboration diagram for StatGroupPy_t:

Public Attributes

- `PyObject_HEAD SST::ConfigStatGroup * ptr`

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

- `sst/core/model/pymodel_statgroup.h`

6.239 SST::Core::Serialization::statics Class Reference

Public Types

- typedef void(* **clear_fxn**) (void)

Static Public Member Functions

- static void **register_finish** (clear_fxn fxn)
- static void **finish** ()

Static Protected Attributes

- static std::list< clear_fxn > * **fxns_** = 0

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

- sst/core/serialization/statics.h
- sst/core/serialization/statics.cc

6.240 SST::Statistics::Statistic< T > Class Template Reference

```
#include <statbase.h>
```

Inheritance diagram for SST::Statistics::Statistic< T >:

Collaboration diagram for SST::Statistics::Statistic< T >:

Public Member Functions

- **SST_ELI_DECLARE_INFO** ([ELI::ProvidesInterface](#), [ELI::ProvidesParams](#)) using Datum
- template<class... InArgs>
void [addData](#) (InArgs &&...args)

Static Public Member Functions

- static fieldType_t **fieldId** ()

Protected Member Functions

- [Statistic](#) ([BaseComponent](#) *comp, const std::string &statName, const std::string &statSubId, [Params](#) &stat←
Params)

Friends

- class **SST::Factory**
- class **SST::BaseComponent**

Additional Inherited Members

6.240.1 Detailed Description

```
template<typename T>
class SST::Statistics::Statistic< T >
```

Forms the template defined base class for statistics gathering within SST.

Template Parameters

<i>T</i>	A template for the basic numerical data stored by this Statistic
----------	--

6.240.2 Constructor & Destructor Documentation

6.240.2.1 `template<typename T> SST::Statistics::Statistic< T >::Statistic (BaseComponent * comp, const std::string & statName, const std::string & statSubId, Params & statParams)` `[inline]`, `[protected]`

Construct a [Statistic](#)

Parameters

<i>comp</i>	- Pointer to the parent constructor.
<i>statName</i>	- Name of the statistic to be registered. This name must match the name in the ElementInfoStatistic .
<i>statSubId</i>	- Additional name of the statistic
<i>statParams</i>	- The parameters for this statistic

6.240.3 Member Function Documentation

6.240.3.1 `template<typename T> template<class... InArgs> void SST::Statistics::Statistic< T >::addData (InArgs &&... args)` `[inline]`

Add data to the [Statistic](#) This will call the `addData_impl()` routine in the derived [Statistic](#).

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

- `sst/core/eli/elementinfo.h`
- `sst/core/statapi/statbase.h`

6.241 SST::Statistics::StatisticBase Class Reference

```
#include <statbase.h>
```

Inheritance diagram for SST::Statistics::StatisticBase:

Public Types

- enum [StatMode_t](#) { `STAT_MODE_UNDEFINED`, `STAT_MODE_COUNT`, `STAT_MODE_PERIODIC` }

Public Member Functions

- void [enable](#) ()
- void [disable](#) ()
- virtual void [clearStatisticData](#) ()
- virtual void [resetCollectionCount](#) ()
- virtual void [incrementCollectionCount](#) ()
- virtual void [setCollectionCount](#) (uint64_t newCount)
- virtual void [setCollectionCountLimit](#) (uint64_t newLimit)
- void [setFlagResetCountOnOutput](#) (bool flag)
- void [setFlagClearDataOnOutput](#) (bool flag)
- void [setFlagOutputAtEndOfSim](#) (bool flag)
- const std::string & [getCompName](#) () const
- const std::string & [getStatName](#) () const
- const std::string & [getStatSubId](#) () const
- const std::string & [getFullStatName](#) () const
- const std::string & [getStatTypeName](#) () const
- const StatisticFieldInfo::fieldType_t & [getStatDataType](#) () const
- const char * [getStatDataTypeShortName](#) () const
- const char * [getStatDataTypeFullName](#) () const
- [BaseComponent](#) * [getComponent](#) () const
- bool [isEnabled](#) () const
- bool [isOutputEnabled](#) () const
- uint64_t [getCollectionCountLimit](#) () const
- uint64_t [getCollectionCount](#) () const
- bool [getFlagResetCountOnOutput](#) () const
- bool [getFlagClearDataOnOutput](#) () const
- bool [getFlagOutputAtEndOfSim](#) () const
- [StatMode_t](#) [getRegisteredCollectionMode](#) () const
- void [delayOutput](#) (const char *delayTime)
- void [delayCollection](#) (const char *delayTime)
- virtual bool [isReady](#) () const
- virtual bool [isNullStatistic](#) () const

Static Public Member Functions

- static const std::vector< [ElementInfoParam](#) > & [ELI_getParams](#) ()

Protected Member Functions

- [StatisticBase](#) ([BaseComponent](#) *comp, const std::string &statName, const std::string &statSubId, [Params](#) &statParams)
- [Params](#) & [getParams](#) ()
- void [setStatisticDataType](#) (const StatisticFieldInfo::fieldType_t dataType)
- void [setStatisticTypeName](#) (const char *typeName)

Friends

- class [SST::Statistics::StatisticProcessingEngine](#)
- class [SST::Statistics::StatisticOutput](#)
- class [SST::Statistics::StatisticGroup](#)
- class [SST::BaseComponent](#)

6.241.1 Detailed Description

Forms the base class for statistics gathering within SST. Statistics are gathered and processed into various (extensible) output forms. Statistics are expected to be named so that they can be located in the simulation output files.

6.241.2 Member Enumeration Documentation

6.241.2.1 enum SST::Statistics::StatisticBase::StatMode_t

[Statistic](#) collection mode

6.241.3 Constructor & Destructor Documentation

6.241.3.1 SST::Statistics::StatisticBase::StatisticBase (BaseComponent * *comp*, const std::string & *statName*, const std::string & *statSubId*, Params & *statParams*) [protected]

Construct a [StatisticBase](#)

Parameters

<i>comp</i>	- Pointer to the parent constructor.
<i>statName</i>	- Name of the statistic to be registered. This name must match the name in the ElementInfoStatistic .
<i>statSubId</i>	- Additional name of the statistic
<i>statParams</i>	- The parameters for this statistic

6.241.4 Member Function Documentation

6.241.4.1 virtual void SST::Statistics::StatisticBase::clearStatisticData () [inline],[virtual]

Inform the [Statistic](#) to clear its data

Reimplemented in [SST::Statistics::AccumulatorStatistic< NumberBase >](#).

6.241.4.2 void SST::Statistics::StatisticBase::delayCollection (const char * *delayTime*)

Delay the statistic from collecting data for a specified delay time.

Parameters

<i>delayTime</i>	- Value in UnitAlgebra format for delay (i.e. 10ns).
------------------	--

6.241.4.3 void SST::Statistics::StatisticBase::delayOutput (const char * *delayTime*)

Delay the statistic from outputting data for a specified delay time

Parameters

<i>delayTime</i>	- Value in UnitAlgebra format for delay (i.e. 10ns).
------------------	--

6.241.4.4 void SST::Statistics::StatisticBase::disable () [inline]

Disable [Statistic](#) for collections

6.241.4.5 void SST::Statistics::StatisticBase::enable () [inline]

Enable [Statistic](#) for collections

6.241.4.6 uint64_t SST::Statistics::StatisticBase::getCollectionCount () const [inline]

Return the current collection count

6.241.4.7 uint64_t SST::Statistics::StatisticBase::getCollectionCountLimit () const [inline]

Return the collection count limit

6.241.4.8 const std::string & SST::Statistics::StatisticBase::getCompName () const

Return the [Component](#) Name

6.241.4.9 BaseComponent* SST::Statistics::StatisticBase::getComponent () const [inline]

Return a pointer to the parent [Component](#)

6.241.4.10 bool SST::Statistics::StatisticBase::getFlagClearDataOnOutput () const [inline]

Return the ClearDataOnOutput flag value

6.241.4.11 bool SST::Statistics::StatisticBase::getFlagOutputAtEndOfSim () const [inline]

Return the OutputAtEndOfSim flag value

6.241.4.12 `bool SST::Statistics::StatisticBase::getFlagResetCountOnOutput () const` `[inline]`

Return the ResetCountOnOutput flag value

6.241.4.13 `const std::string& SST::Statistics::StatisticBase::getFullStatName () const` `[inline]`

Return the full [Statistic](#) name of Component.StatName.SubId

6.241.4.14 `Params& SST::Statistics::StatisticBase::getParams ()` `[inline]`, `[protected]`

Return the [Statistic](#) Parameters

6.241.4.15 `StatMode_t SST::Statistics::StatisticBase::getRegisteredCollectionMode () const` `[inline]`

Return the collection mode that is registered

6.241.4.16 `const StatisticFieldInfo::fieldType_t& SST::Statistics::StatisticBase::getStatDataType () const` `[inline]`

Return the [Statistic](#) data type

6.241.4.17 `const char* SST::Statistics::StatisticBase::getStatDataTypeFullName () const` `[inline]`

Return the [Statistic](#) data type

6.241.4.18 `const char* SST::Statistics::StatisticBase::getStatDataTypeShortName () const` `[inline]`

Return the [Statistic](#) data type

6.241.4.19 `const std::string& SST::Statistics::StatisticBase::getStatName () const` `[inline]`

Return the [Statistic](#) Name

6.241.4.20 `const std::string& SST::Statistics::StatisticBase::getStatSubId () const` `[inline]`

Return the [Statistic](#) SubId

6.241.4.21 `const std::string& SST::Statistics::StatisticBase::getStatTypeName () const` `[inline]`

Return the [Statistic](#) type name

6.241.4.22 void SST::Statistics::StatisticBase::incrementCollectionCount () [virtual]

Increment current collection count

6.241.4.23 bool SST::Statistics::StatisticBase::isEnabled () const [inline]

Return the enable status of the [Statistic](#)

6.241.4.24 virtual bool SST::Statistics::StatisticBase::isNullStatistic () const [inline],[virtual]

Indicate if the [Statistic](#) is a [NullStatistic](#)

6.241.4.25 bool SST::Statistics::StatisticBase::isOutputEnabled () const [inline]

Return the enable status of the [Statistic](#)'s ability to output data

6.241.4.26 virtual bool SST::Statistics::StatisticBase::isReady () const [inline],[virtual]

Indicate that the [Statistic](#) is Ready to be used

6.241.4.27 void SST::Statistics::StatisticBase::resetCollectionCount () [virtual]

Set the current collection count to 0

6.241.4.28 void SST::Statistics::StatisticBase::setCollectionCount (uint64_t newCount) [virtual]

Set the current collection count to a defined value

6.241.4.29 void SST::Statistics::StatisticBase::setCollectionCountLimit (uint64_t newLimit) [virtual]

Set the collection count limit to a defined value

6.241.4.30 void SST::Statistics::StatisticBase::setFlagClearDataOnOutput (bool flag) [inline]

Set the Clear Data On [Output](#) flag. If Set, the data in the statistic will be cleared by calling [clearStatisticData\(\)](#).

6.241.4.31 void SST::Statistics::StatisticBase::setFlagOutputAtEndOfSim (bool flag) [inline]

Set the [Output](#) At End Of Sim flag. If Set, the statistic will perform an output at the end of simulation.

6.241.4.32 void SST::Statistics::StatisticBase::setFlagResetCountOnOutput (bool *flag*) [inline]

Set the Reset Count On [Output](#) flag. If Set, the collection count will be reset when statistic is output.

6.241.4.33 void SST::Statistics::StatisticBase::setStatisticDataType (const StatisticFieldInfo::fieldType_t *dataType*)
[inline], [protected]

Set the [Statistic](#) Data Type

6.241.4.34 void SST::Statistics::StatisticBase::setStatisticTypeName (const char * *typeName*) [inline],
[protected]

Set an optional [Statistic](#) Type Name

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

- sst/core/statapi/statbase.h
- sst/core/statapi/statbase.cc

6.242 SST::Statistics::StatisticCollector< T, F > Class Template Reference

```
#include <statbase.h>
```

6.242.1 Detailed Description

```
template<class T, bool F = std::is_fundamental<T>::value>
class SST::Statistics::StatisticCollector< T, F >
```

Base type that creates the virtual addData(...) interface Used for distinguishing fundamental types (collected by value) and composite struct types (collected by reference)

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

- sst/core/statapi/statbase.h

6.243 SST::Statistics::StatisticCollector< std::tuple< Args... >, false > Struct Template Reference

Public Member Functions

- virtual void **addData_impl** (Args...args)=0
- template<class... InArgs>
void **addData** (InArgs &&...args)

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

- sst/core/statapi/statbase.h

6.244 SST::Statistics::StatisticCollector< T, true > Struct Template Reference

Public Member Functions

- virtual void **addData_impl** (T data)=0

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

- sst/core/statapi/statbase.h

6.245 SST::Statistics::StatisticFieldInfo Class Reference

```
#include <statfieldinfo.h>
```

Public Types

- using **fieldType_t** = ::SST::Statistics::fieldType_t
- using **fieldHandle_t** = int32_t

Public Member Functions

- [StatisticFieldInfo](#) (const char *statName, const char *fieldName, fieldType_t fieldType)
- const std::string & [getStatName](#) () const
- const std::string & [getFieldName](#) () const
- fieldType_t [getFieldType](#) () const
- std::string [getFieldUniqueName](#) () const
- bool [operator==](#) ([StatisticFieldInfo](#) &FieldInfo1)
- void [setFieldHandle](#) (fieldHandle_t handle)
- fieldHandle_t [getFieldHandle](#) ()

Static Public Member Functions

- static const char * [getFieldTypeShortName](#) (fieldType_t type)
- static const char * [getFieldTypeFullName](#) (fieldType_t type)
- template<typename T >
static fieldType_t [getFieldTypeFromTemplate](#) ()

6.245.1 Detailed Description

The class for representing [Statistic Output Fields](#)

6.245.2 Constructor & Destructor Documentation

- 6.245.2.1 SST::Statistics::StatisticFieldInfo::StatisticFieldInfo (const char * *statName*, const char * *fieldName*, fieldType_t *fieldType*)

Construct a [StatisticFieldInfo](#)

Parameters

<i>statName</i>	- Name of the statistic registering this field.
<i>fieldName</i>	- Name of the Field to be assigned.
<i>fieldType</i>	- Data type of the field.

6.245.3 Member Function Documentation

6.245.3.1 `fieldHandle_t SST::Statistics::StatisticFieldInfo::getFieldHandle ()` `[inline]`

Get the field handle

Returns

The assigned field handle.

6.245.3.2 `const std::string& SST::Statistics::StatisticFieldInfo::getFieldName () const` `[inline]`

Return the field name related to this field info

6.245.3.3 `fieldType_t SST::Statistics::StatisticFieldInfo::getFieldType () const` `[inline]`

Return the field type related to this field info

6.245.3.4 `std::string SST::Statistics::StatisticFieldInfo::getFieldUniqueName () const`

Return the field type related to this field info

6.245.3.5 `const std::string& SST::Statistics::StatisticFieldInfo::getStatName () const` `[inline]`

Return the statistic name related to this field info

6.245.3.6 `bool SST::Statistics::StatisticFieldInfo::operator== (StatisticFieldInfo & FieldInfo1)`

Compare two field info structures

Parameters

<i>FieldInfo1</i>	- a FieldInfo to compare against.
-------------------	-----------------------------------

Returns

True if the Field Info structures are the same.

6.245.3.7 void SST::Statistics::StatisticFieldInfo::setFieldHandle (fieldHandle_t *handle*) [inline]

Set the field handle

Parameters

<i>handle</i>	- The assigned field handle for this FieldInfo
---------------	--

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

- sst/core/statapi/statfieldinfo.h
- sst/core/statapi/statfieldinfo.cc

6.246 SST::Statistics::StatisticFieldType< T > Class Template Reference

Inheritance diagram for SST::Statistics::StatisticFieldType< T >:

Collaboration diagram for SST::Statistics::StatisticFieldType< T >:

Public Member Functions

- **StatisticFieldType** (const char *name, const char *shortName)
- const char * **fieldName** () const override
- const char * **shortName** () const override

Static Public Member Functions

- static const char * **getFieldName** ()
- static const char * **getShortName** ()
- static fieldType_t **id** ()

Additional Inherited Members

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

- sst/core/statapi/statfieldinfo.h

6.247 SST::Statistics::StatisticFieldTypeBase Class Reference

Inheritance diagram for SST::Statistics::StatisticFieldTypeBase:

Public Member Functions

- virtual const char * **fieldName** () const =0
- virtual const char * **shortName** () const =0

Static Public Member Functions

- static [StatisticFieldTypeBase](#) * **getField** (fieldType_t fieldType)
- static void **checkRegisterConflict** (const char *oldName, const char *newName)
- static fieldType_t **allocateFieldEnum** ()

Static Protected Member Functions

- static void **addField** (fieldType_t id, [StatisticFieldTypeBase](#) *base)

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

- sst/core/statapi/statfieldinfo.h
- sst/core/statapi/statfieldinfo.cc

6.248 SST::Statistics::StatisticGroup Class Reference

Collaboration diagram for SST::Statistics::StatisticGroup:

Public Member Functions

- **StatisticGroup** (const [ConfigStatGroup](#) &csg)
- bool **containsStatistic** (const [StatisticBase](#) *stat) const
- bool **claimsStatistic** (const [StatisticBase](#) *stat) const
- void **addStatistic** ([StatisticBase](#) *stat)

Public Attributes

- bool **isDefault**
- std::string **name**
- [StatisticOutput](#) * **output**
- [UnitAlgebra](#) **outputFreq**
- std::vector< ComponentId_t > **components**
- std::vector< std::string > **statNames**
- std::vector< [StatisticBase](#) * > **stats**

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

- sst/core/statapi/statgroup.h
- sst/core/statapi/statgroup.cc

6.249 SST::Statistics::StatisticInfo Class Reference

Inheritance diagram for SST::Statistics::StatisticInfo:

Collaboration diagram for SST::Statistics::StatisticInfo:

Public Member Functions

- **StatisticInfo** (const std::string &name)
- **StatisticInfo** (const std::string &name, const [Params](#) ¶ms)
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override

Public Attributes

- std::string **name**
- [Params](#) **params**

Additional Inherited Members

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

- sst/core/statapi/statbase.h

6.250 SST::Statistics::StatisticOutput Class Reference

```
#include <statoutput.h>
```

Inheritance diagram for SST::Statistics::StatisticOutput:

Collaboration diagram for SST::Statistics::StatisticOutput:

Public Types

- using **fieldType_t** = StatisticFieldInfo::fieldType_t
- using **fieldHandle_t** = StatisticFieldInfo::fieldHandle_t
- using **FieldInfoArray_t** = std::vector< [StatisticFieldInfo](#) * >
- using **FieldNameMap_t** = std::unordered_map< std::string, fieldHandle_t >

Public Member Functions

- [StatisticOutput](#) ([Params](#) &outputParameters)
- std::string & [getStatisticOutputName](#) ()
- [Params](#) & [getOutputParameters](#) ()
- virtual bool [acceptsGroups](#) () const
- template<typename T >
fieldHandle_t [registerField](#) (const char *fieldName)
- [StatisticFieldInfo](#) * [getRegisteredField](#) (fieldHandle_t fieldHandle)
- template<typename T >
[StatisticFieldInfo](#) * [getRegisteredField](#) (const char *statisticName, const char *fieldName)
- [FieldInfoArray_t](#) & [getFieldInfoArray](#) ()
- virtual void [outputField](#) (fieldHandle_t fieldHandle, int32_t data)
- virtual void **outputField** (fieldHandle_t fieldHandle, uint32_t data)
- virtual void **outputField** (fieldHandle_t fieldHandle, int64_t data)
- virtual void **outputField** (fieldHandle_t fieldHandle, uint64_t data)
- virtual void **outputField** (fieldHandle_t fieldHandle, float data)
- virtual void **outputField** (fieldHandle_t fieldHandle, double data)
- const char * [getFieldTypeShortName](#) (fieldType_t type)

Static Public Member Functions

- static const std::vector< [SST::ElementInfoParam](#) > & [ELI_getParams](#) ()

Protected Member Functions

- virtual bool [checkOutputParameters](#) ()=0
- virtual void [printUsage](#) ()=0
- virtual void **implStartRegisterFields** ([StatisticBase](#) *UNUSED(statistic))
- virtual void **implRegisteredField** (fieldHandle_t UNUSED(fieldHandle))
- virtual void **implStopRegisterFields** ()
- virtual void [startOfSimulation](#) ()=0
- virtual void [endOfSimulation](#) ()=0
- virtual void [implStartOutputEntries](#) ([StatisticBase](#) *statistic)=0
- virtual void [implStopOutputEntries](#) ()=0
- virtual void **implStartRegisterGroup** ([StatisticGroup](#) *UNUSED(group))
- virtual void **implStopRegisterGroup** ()
- virtual void **implStartOutputGroup** ([StatisticGroup](#) *UNUSED(group))
- virtual void **implStopOutputGroup** ()
- void **setStatisticOutputName** (std::string name)
- void **lock** ()
- void **unlock** ()

Friends

- class **SST::Simulation**
- class **SST::Statistics::StatisticProcessingEngine**

6.250.1 Detailed Description

Forms the base class for statistics output generation within the SST core. Statistics are gathered by the statistic objects and then processed sent to the derived output object either periodically or by event and/or also at the end of the simulation. A single statistic output will be created by the simulation (per node) and will collect the data per its design.

6.250.2 Constructor & Destructor Documentation

6.250.2.1 SST::Statistics::StatisticOutput::StatisticOutput (Params & outputParameters)

Construct a base [StatisticOutput](#)

Parameters

<i>outputParameters</i>	- The parameters for the statistic Output .
-------------------------	---

6.250.3 Member Function Documentation

6.250.3.1 virtual bool SST::Statistics::StatisticOutput::acceptsGroups () const [inline],[virtual]

True if this StatOutput can handle StatisticGroups

Reimplemented in [SST::Statistics::StatisticOutputHDF5](#).

6.250.3.2 virtual bool SST::Statistics::StatisticOutput::checkOutputParameters () [protected],[pure virtual]

Have the [Statistic Output](#) check its parameters

Returns

True if all parameters are ok; False if a parameter is missing or incorrect.

Implemented in [SST::Statistics::StatisticOutputHDF5](#), [SST::Statistics::StatisticOutputCSV](#), [SST::Statistics::StatisticOutputTxt](#), [SST::Statistics::StatisticOutputConsole](#), and [SST::Statistics::StatisticOutputJSON](#).

6.250.3.3 virtual void SST::Statistics::StatisticOutput::endOfSimulation () [protected],[pure virtual]

Indicate to [Statistic Output](#) that simulation has ended. Allows object to perform any shutdown required.

Implemented in [SST::Statistics::StatisticOutputHDF5](#), [SST::Statistics::StatisticOutputCSV](#), [SST::Statistics::StatisticOutputTxt](#), [SST::Statistics::StatisticOutputConsole](#), and [SST::Statistics::StatisticOutputJSON](#).

6.250.3.4 FieldInfoArray_t& SST::Statistics::StatisticOutput::getFieldInfoArray () [inline]

Return the array of registered field infos.

6.250.3.5 const char * SST::Statistics::StatisticOutput::getFieldTypeShortName (fieldType_t type)

[Output](#) field data.

Parameters

<i>type</i>	- The field type to get name of.
-------------	----------------------------------

Returns

String name of the field type.

6.250.3.6 Params& SST::Statistics::StatisticOutput::getOutputParameters () [inline]

Return the parameters for the [StatisticOutput](#)

6.250.3.7 StatisticFieldInfo * SST::Statistics::StatisticOutput::getRegisteredField (fieldHandle_t fieldHandle)

Adjust the hierarchy of the fields (FUTURE SUPPORT)

Parameters

<i>fieldHandle</i>	- The handle of the field to adjust.
<i>Level</i>	- The level of the field.
<i>parent</i>	- The parent field of the field. Return the information on a registered field via the field handle.
<i>fieldHandle</i>	- The handle of the registered field.

Returns

Pointer to the registered field info.

6.250.3.8 template<typename T> StatisticFieldInfo* SST::Statistics::StatisticOutput::getRegisteredField (const char * statisticName, const char * fieldName) [inline]

Return the information on a registered field via known names.

Parameters

<i>componentName</i>	- The name of the component.
<i>statisticName</i>	- The name of the statistic.
<i>fieldName</i>	- The name of the field .

Returns

Pointer to the registered field info.

6.250.3.9 std::string& SST::Statistics::StatisticOutput::getStatisticOutputName () [inline]

Return the [Statistic Output](#) name

6.250.3.10 `virtual void SST::Statistics::StatisticOutput::implStartOutputEntries (StatisticBase * statistic)`
`[protected],[pure virtual]`

Indicate to [Statistic Output](#) that a statistic is about to send data to be output Allows object to perform any initialization before output.

Implemented in [SST::Statistics::StatisticOutputHDF5](#), [SST::Statistics::StatisticOutputCSV](#), [SST::Statistics::StatisticOutputTxt](#), [SST::Statistics::StatisticOutputConsole](#), and [SST::Statistics::StatisticOutputJSON](#).

6.250.3.11 `virtual void SST::Statistics::StatisticOutput::implStopOutputEntries ()` `[protected],[pure virtual]`

Indicate to [Statistic Output](#) that a statistic is finished sending data to be output Allows object to perform any cleanup.

Implemented in [SST::Statistics::StatisticOutputHDF5](#), [SST::Statistics::StatisticOutputCSV](#), [SST::Statistics::StatisticOutputTxt](#), [SST::Statistics::StatisticOutputConsole](#), and [SST::Statistics::StatisticOutputJSON](#).

6.250.3.12 `virtual void SST::Statistics::StatisticOutput::outputField (fieldHandle_t fieldHandle, int32_t data)` `[virtual]`

[Output](#) field data.

Parameters

<i>fieldHandle</i>	- The handle of the registered field.
<i>data</i>	- The data to be output.

Reimplemented in [SST::Statistics::StatisticOutputHDF5](#), [SST::Statistics::StatisticOutputCSV](#), [SST::Statistics::StatisticOutputTxt](#), [SST::Statistics::StatisticOutputConsole](#), and [SST::Statistics::StatisticOutputJSON](#).

6.250.3.13 `virtual void SST::Statistics::StatisticOutput::printUsage ()` `[protected],[pure virtual]`

Have [Statistic](#) Object print out its usage and parameter info. Called when [checkOutputParameters\(\)](#) returns false

Implemented in [SST::Statistics::StatisticOutputHDF5](#), [SST::Statistics::StatisticOutputCSV](#), [SST::Statistics::StatisticOutputTxt](#), [SST::Statistics::StatisticOutputConsole](#), and [SST::Statistics::StatisticOutputJSON](#).

6.250.3.14 `template<typename T > fieldHandle_t SST::Statistics::StatisticOutput::registerField (const char * fieldName)`
`[inline]`

Register a field to be output (templated function)

Parameters

<i>fieldName</i>	- The name of the field.
------------------	--------------------------

Returns

The handle of the registered field or -1 if type is not supported. Note: Any field names (of the same data type) that are previously registered by a statistic will return the previously handle.

6.250.3.15 `virtual void SST::Statistics::StatisticOutput::startOfSimulation () [protected],[pure virtual]`

Indicate to [Statistic Output](#) that simulation has started. Allows object to perform any setup required.

Implemented in [SST::Statistics::StatisticOutputHDF5](#), [SST::Statistics::StatisticOutputCSV](#), [SST::Statistics::StatisticOutputTxt](#), [SST::Statistics::StatisticOutputConsole](#), and [SST::Statistics::StatisticOutputJSON](#).

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

- sst/core/statapi/statoutput.h
- sst/core/statapi/statoutput.cc

6.251 SST::Statistics::StatisticOutputConsole Class Reference

```
#include <statoutputconsole.h>
```

Inheritance diagram for SST::Statistics::StatisticOutputConsole:

Collaboration diagram for SST::Statistics::StatisticOutputConsole:

Public Member Functions

- [SST_ELI_REGISTER_DERIVED](#) ([StatisticOutput](#), [StatisticOutputConsole](#), "sst", "statoutputconsole", SST↵_ELI_ELEMENT_VERSION(1, 0, 0), "Output directly to console screen") [StatisticOutputConsole](#)([Params](#) &outputParameters)

Protected Member Functions

- bool [checkOutputParameters](#) () override
- void [printUsage](#) () override
- void [startOfSimulation](#) () override
- void [endOfSimulation](#) () override
- void [implStartOutputEntries](#) ([StatisticBase](#) *statistic) override
- void [implStopOutputEntries](#) () override
- void [outputField](#) (fieldHandle_t fieldHandle, int32_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, uint32_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, int64_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, uint64_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, float data) override
- void [outputField](#) (fieldHandle_t fieldHandle, double data) override

Additional Inherited Members

6.251.1 Detailed Description

The class for statistics output to the console. This will be the default statistic output in SST.

6.251.2 Member Function Documentation

6.251.2.1 `bool SST::Statistics::StatisticOutputConsole::checkOutputParameters ()` `[override]`, `[protected]`, `[virtual]`

Perform a check of provided parameters

Returns

True if all required parameters and options are acceptable

Implements [SST::Statistics::StatisticOutput](#).

6.251.2.2 `void SST::Statistics::StatisticOutputConsole::endOfSimulation ()` `[override]`, `[protected]`, `[virtual]`

Indicate to [Statistic Output](#) that simulation ended. [Statistic](#) output may perform any shutdown code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

6.251.2.3 `void SST::Statistics::StatisticOutputConsole::implStartOutputEntries (StatisticBase * statistic)` `[override]`, `[protected]`, `[virtual]`

Implementation function for the start of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is about to send data to the [Statistic Output](#) for processing.

Parameters

<i>statistic</i>	- Pointer to the statistic object than the output can retrieve data from.
------------------	---

Implements [SST::Statistics::StatisticOutput](#).

6.251.2.4 `void SST::Statistics::StatisticOutputConsole::implStopOutputEntries ()` `[override]`, `[protected]`, `[virtual]`

Implementation function for the end of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is finished sending data to the [Statistic Output](#) for processing. The [Statistic Output](#) can perform any output related functions here.

Implements [SST::Statistics::StatisticOutput](#).

6.251.2.5 void SST::Statistics::StatisticOutputConsole::outputField (fieldHandle_t *fieldHandle*, int32_t *data*)
 [override],[protected],[virtual]

Implementation functions for output. These will be called by the statistic to provide [Statistic](#) defined data to be output.

Parameters

<i>fieldHandle</i>	- The handle to the registered statistic field.
<i>data</i>	- The data related to the registered field to be output.

Reimplemented from [SST::Statistics::StatisticOutput](#).

6.251.2.6 void SST::Statistics::StatisticOutputConsole::printUsage () [override],[protected],[virtual]

Print out usage for this [Statistic Output](#)

Implements [SST::Statistics::StatisticOutput](#).

6.251.2.7 SST::Statistics::StatisticOutputConsole::SST_ELI_REGISTER_DERIVED ([StatisticOutput](#) , [StatisticOutputConsole](#) , "sst" , "statoutputconsole" , SST_ELI_ELEMENT_VERSION(1, 0, 0) , "Output directly to console screen")

Construct a StatOutputConsole

Parameters

<i>outputParameters</i>	- Parameters used for this Statistic Output
-------------------------	---

6.251.2.8 void SST::Statistics::StatisticOutputConsole::startOfSimulation () [override],[protected],[virtual]

Indicate to [Statistic Output](#) that simulation started. [Statistic](#) output may perform any startup code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

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

- sst/core/statapi/statoutputconsole.h
- sst/core/statapi/statoutputconsole.cc

6.252 SST::Statistics::StatisticOutputCSV Class Reference

```
#include <statoutputcsv.h>
```

Inheritance diagram for SST::Statistics::StatisticOutputCSV:

Collaboration diagram for SST::Statistics::StatisticOutputCSV:

Public Member Functions

- [SST_ELI_REGISTER_DERIVED](#) ([StatisticOutput](#), [StatisticOutputCSV](#), "sst", "statoutputcsv", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Output directly to console screen") [StatisticOutputCSV](#)([Params](#) &outputParameters)

Protected Member Functions

- bool [checkOutputParameters](#) () override
- void [printUsage](#) () override
- void [startOfSimulation](#) () override
- void [endOfSimulation](#) () override
- void [implStartOutputEntries](#) ([StatisticBase](#) *statistic) override
- void [implStopOutputEntries](#) () override
- void [outputField](#) (fieldHandle_t fieldHandle, int32_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, uint32_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, int64_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, uint64_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, float data) override
- void [outputField](#) (fieldHandle_t fieldHandle, double data) override

Additional Inherited Members

6.252.1 Detailed Description

The class for statistics output to a comma separated file.

6.252.2 Member Function Documentation

6.252.2.1 bool [SST::Statistics::StatisticOutputCSV::checkOutputParameters](#) () [[override](#)], [[protected](#)], [[virtual](#)]

Perform a check of provided parameters

Returns

True if all required parameters and options are acceptable

Implements [SST::Statistics::StatisticOutput](#).

6.252.2.2 void [SST::Statistics::StatisticOutputCSV::endOfSimulation](#) () [[override](#)], [[protected](#)], [[virtual](#)]

Indicate to [Statistic Output](#) that simulation ended. [Statistic](#) output may perform any shutdown code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

6.252.2.3 void [SST::Statistics::StatisticOutputCSV::implStartOutputEntries](#) ([StatisticBase](#) * *statistic*) [[override](#)], [[protected](#)], [[virtual](#)]

Implementation function for the start of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is about to send data to the [Statistic Output](#) for processing.

Parameters

<i>statistic</i>	- Pointer to the statistic object than the output can retrieve data from.
------------------	---

Implements [SST::Statistics::StatisticOutput](#).

6.252.2.4 `void SST::Statistics::StatisticOutputCSV::implStopOutputEntries () [override], [protected], [virtual]`

Implementation function for the end of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is finished sending data to the [Statistic Output](#) for processing. The [Statistic Output](#) can perform any output related functions here.

Implements [SST::Statistics::StatisticOutput](#).

6.252.2.5 `void SST::Statistics::StatisticOutputCSV::outputField (fieldHandle_t fieldHandle, int32_t data) [override], [protected], [virtual]`

Implementation functions for output. These will be called by the statistic to provide [Statistic](#) defined data to be output.

Parameters

<i>fieldHandle</i>	- The handle to the registered statistic field.
<i>data</i>	- The data related to the registered field to be output.

Reimplemented from [SST::Statistics::StatisticOutput](#).

6.252.2.6 `void SST::Statistics::StatisticOutputCSV::printUsage () [override], [protected], [virtual]`

Print out usage for this [Statistic Output](#)

Implements [SST::Statistics::StatisticOutput](#).

6.252.2.7 `SST::Statistics::StatisticOutputCSV::SST_ELI_REGISTER_DERIVED (StatisticOutput , StatisticOutputCSV , "sst" , "statoutputcsv" , SST_ELI_ELEMENT_VERSION(1, 0, 0) , "Output directly to console screen")`

Construct a StatOutputCSV

Parameters

<i>outputParameters</i>	- Parameters used for this Statistic Output
-------------------------	---

6.252.2.8 void SST::Statistics::StatisticOutputCSV::startOfSimulation () [override], [protected], [virtual]

Indicate to [Statistic Output](#) that simulation started. [Statistic](#) output may perform any startup code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

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

- sst/core/statapi/statoutputcsv.h
- sst/core/statapi/statoutputcsv.cc

6.253 SST::Statistics::StatisticOutputHDF5 Class Reference

```
#include <statoutputhdf5.h>
```

Inheritance diagram for SST::Statistics::StatisticOutputHDF5:

Collaboration diagram for SST::Statistics::StatisticOutputHDF5:

Public Member Functions

- [SST_ELI_REGISTER_DERIVED](#) ([StatisticOutput](#), [StatisticOutputHDF5](#), "sst", "statoutputhdf5", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Output to an HDF5 file") [StatisticOutputHDF5](#)([Params](#) &outputParameters)
- bool [acceptsGroups](#) () const override

Protected Member Functions

- bool [checkOutputParameters](#) () override
- void [printUsage](#) () override
- void [implStartRegisterFields](#) ([StatisticBase](#) *stat) override
- void [implRegisteredField](#) (fieldHandle_t fieldHandle) override
- void [implStopRegisterFields](#) () override
- void [implStartRegisterGroup](#) ([StatisticGroup](#) *group) override
- void [implStopRegisterGroup](#) () override
- void [startOfSimulation](#) () override
- void [endOfSimulation](#) () override
- void [implStartOutputEntries](#) ([StatisticBase](#) *statistic) override
- void [implStopOutputEntries](#) () override
- void [implStartOutputGroup](#) ([StatisticGroup](#) *group) override
- void [implStopOutputGroup](#) () override
- void [outputField](#) (fieldHandle_t fieldHandle, int32_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, uint32_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, int64_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, uint64_t data) override
- void [outputField](#) (fieldHandle_t fieldHandle, float data) override
- void [outputField](#) (fieldHandle_t fieldHandle, double data) override

Additional Inherited Members

6.253.1 Detailed Description

The class for statistics output to a comma separated file.

6.253.2 Member Function Documentation

6.253.2.1 `bool SST::Statistics::StatisticOutputHDF5::acceptsGroups () const` `[inline]`, `[override]`, `[virtual]`

True if this StatOutput can handle StatisticGroups

Reimplemented from [SST::Statistics::StatisticOutput](#).

6.253.2.2 `bool SST::Statistics::StatisticOutputHDF5::checkOutputParameters ()` `[override]`, `[protected]`, `[virtual]`

Perform a check of provided parameters

Returns

True if all required parameters and options are acceptable

Implements [SST::Statistics::StatisticOutput](#).

6.253.2.3 `void SST::Statistics::StatisticOutputHDF5::endOfSimulation ()` `[override]`, `[protected]`, `[virtual]`

Indicate to [Statistic Output](#) that simulation ended. [Statistic](#) output may perform any shutdown code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

6.253.2.4 `void SST::Statistics::StatisticOutputHDF5::implStartOutputEntries (StatisticBase * statistic)` `[override]`, `[protected]`, `[virtual]`

Implementation function for the start of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is about to send data to the [Statistic Output](#) for processing.

Parameters

<i>statistic</i>	- Pointer to the statistic object than the output can retrieve data from.
------------------	---

Implements [SST::Statistics::StatisticOutput](#).

6.253.2.5 void SST::Statistics::StatisticOutputHDF5::implStopOutputEntries () [override],[protected],[virtual]

Implementation function for the end of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is finished sending data to the [Statistic Output](#) for processing. The [Statistic Output](#) can perform any output related functions here.

Implements [SST::Statistics::StatisticOutput](#).

6.253.2.6 void SST::Statistics::StatisticOutputHDF5::outputField (fieldHandle_t *fieldHandle*, int32_t *data*) [override],[protected],[virtual]

Implementation functions for output. These will be called by the statistic to provide [Statistic](#) defined data to be output.

Parameters

<i>fieldHandle</i>	- The handle to the registered statistic field.
<i>data</i>	- The data related to the registered field to be output.

Reimplemented from [SST::Statistics::StatisticOutput](#).

6.253.2.7 void SST::Statistics::StatisticOutputHDF5::printUsage () [override],[protected],[virtual]

Print out usage for this [Statistic Output](#)

Implements [SST::Statistics::StatisticOutput](#).

6.253.2.8 SST::Statistics::StatisticOutputHDF5::SST_ELI_REGISTER_DERIVED (StatisticOutput , StatisticOutputHDF5 , "sst" , "statoutputhdf5" , SST_ELI_ELEMENT_VERSION(1, 0, 0) , "Output to an HDF5 file")

Construct a StatOutputHDF5

Parameters

<i>outputParameters</i>	- Parameters used for this Statistic Output
-------------------------	---

6.253.2.9 void SST::Statistics::StatisticOutputHDF5::startOfSimulation () [override],[protected],[virtual]

Indicate to [Statistic Output](#) that simulation started. [Statistic](#) output may perform any startup code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

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

- sst/core/statapi/statoutputhdf5.h
- sst/core/statapi/statoutputhdf5.cc

6.254 SST::Statistics::StatisticOutputJSON Class Reference

```
#include <statoutputjson.h>
```

Inheritance diagram for SST::Statistics::StatisticOutputJSON:

Collaboration diagram for SST::Statistics::StatisticOutputJSON:

Public Member Functions

- [SST_ELI_REGISTER_DERIVED](#) ([StatisticOutput](#), [StatisticOutputJSON](#), "sst", "statoutputjson", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Output to a JSON file") [StatisticOutputJSON](#)([Params](#) &outputParameters)

Protected Member Functions

- bool [checkOutputParameters](#) () override
- void [printUsage](#) () override
- void [startOfSimulation](#) () override
- void [endOfSimulation](#) () override
- void [implStartOutputEntries](#) ([StatisticBase](#) *statistic) override
- void [implStopOutputEntries](#) () override
- void [outputField](#) ([fieldHandle_t](#) fieldHandle, [int32_t](#) data) override
- void **outputField** ([fieldHandle_t](#) fieldHandle, [uint32_t](#) data) override
- void **outputField** ([fieldHandle_t](#) fieldHandle, [int64_t](#) data) override
- void **outputField** ([fieldHandle_t](#) fieldHandle, [uint64_t](#) data) override
- void **outputField** ([fieldHandle_t](#) fieldHandle, [float](#) data) override
- void **outputField** ([fieldHandle_t](#) fieldHandle, [double](#) data) override
- void [printIndent](#) ()

Additional Inherited Members

6.254.1 Detailed Description

The class for statistics output to a JSON formatted file

6.254.2 Member Function Documentation

6.254.2.1 `bool SST::Statistics::StatisticOutputJSON::checkOutputParameters () [override], [protected], [virtual]`

Perform a check of provided parameters

Returns

True if all required parameters and options are acceptable

Implements [SST::Statistics::StatisticOutput](#).

6.254.2.2 `void SST::Statistics::StatisticOutputJSON::endOfSimulation () [override],[protected],[virtual]`

Indicate to [Statistic Output](#) that simulation ended. [Statistic](#) output may perform any shutdown code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

6.254.2.3 `void SST::Statistics::StatisticOutputJSON::implStartOutputEntries (StatisticBase * statistic) [override],[protected],[virtual]`

Implementation function for the start of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is about to send data to the [Statistic Output](#) for processing.

Parameters

<i>statistic</i>	- Pointer to the statistic object than the output can retrieve data from.
------------------	---

Implements [SST::Statistics::StatisticOutput](#).

6.254.2.4 `void SST::Statistics::StatisticOutputJSON::implStopOutputEntries () [override],[protected],[virtual]`

Implementation function for the end of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is finished sending data to the [Statistic Output](#) for processing. The [Statistic Output](#) can perform any output related functions here.

Implements [SST::Statistics::StatisticOutput](#).

6.254.2.5 `void SST::Statistics::StatisticOutputJSON::outputField (fieldHandle_t fieldHandle, int32_t data) [override],[protected],[virtual]`

Implementation functions for output. These will be called by the statistic to provide [Statistic](#) defined data to be output.

Parameters

<i>fieldHandle</i>	- The handle to the registered statistic field.
<i>data</i>	- The data related to the registered field to be output.

Reimplemented from [SST::Statistics::StatisticOutput](#).

6.254.2.6 `void SST::Statistics::StatisticOutputJSON::printUsage () [override],[protected],[virtual]`

Print out usage for this [Statistic Output](#)

Implements [SST::Statistics::StatisticOutput](#).

6.254.2.7 SST::Statistics::StatisticOutputJSON::SST_ELI_REGISTER_DERIVED ([StatisticOutput](#) , [StatisticOutputJSON](#) , "sst" , "statoutputjson" , SST_ELI_ELEMENT_VERSION(1, 0, 0) , "Output to a JSON file")

Construct a StatOutputJSON

Parameters

<i>outputParameters</i>	- Parameters used for this Statistic Output
-------------------------	---

6.254.2.8 void SST::Statistics::StatisticOutputJSON::startOfSimulation () [override], [protected], [virtual]

Indicate to [Statistic Output](#) that simulation started. [Statistic](#) output may perform any startup code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

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

- sst/core/statapi/statoutputjson.h
- sst/core/statapi/statoutputjson.cc

6.255 SST::Statistics::StatisticOutputTxt Class Reference

```
#include <statoutputtxt.h>
```

Inheritance diagram for SST::Statistics::StatisticOutputTxt:

Collaboration diagram for SST::Statistics::StatisticOutputTxt:

Public Member Functions

- [SST_ELI_REGISTER_DERIVED](#) ([StatisticOutput](#), [StatisticOutputTxt](#), "sst", "statoutputtxt", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Output directly to console screen") [StatisticOutputTxt](#)([Params](#) &outputParameters)

Protected Member Functions

- bool [checkOutputParameters](#) () override
- void [printUsage](#) () override
- void [startOfSimulation](#) () override
- void [endOfSimulation](#) () override
- void [implStartOutputEntries](#) ([StatisticBase](#) *statistic) override
- void [implStopOutputEntries](#) () override
- void [outputField](#) (fieldHandle_t fieldHandle, int32_t data) override
- void **outputField** (fieldHandle_t fieldHandle, uint32_t data) override
- void **outputField** (fieldHandle_t fieldHandle, int64_t data) override
- void **outputField** (fieldHandle_t fieldHandle, uint64_t data) override
- void **outputField** (fieldHandle_t fieldHandle, float data) override
- void **outputField** (fieldHandle_t fieldHandle, double data) override

Additional Inherited Members

6.255.1 Detailed Description

The class for statistics output to a text file.

6.255.2 Member Function Documentation

6.255.2.1 `bool SST::Statistics::StatisticOutputTxt::checkOutputParameters () [override], [protected], [virtual]`

Perform a check of provided parameters

Returns

True if all required parameters and options are acceptable

Implements [SST::Statistics::StatisticOutput](#).

6.255.2.2 `void SST::Statistics::StatisticOutputTxt::endOfSimulation () [override], [protected], [virtual]`

Indicate to [Statistic Output](#) that simulation ended. [Statistic](#) output may perform any shutdown code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

6.255.2.3 `void SST::Statistics::StatisticOutputTxt::implStartOutputEntries (StatisticBase * statistic) [override], [protected], [virtual]`

Implementation function for the start of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is about to send data to the [Statistic Output](#) for processing.

Parameters

<i>statistic</i>	- Pointer to the statistic object than the output can retrieve data from.
------------------	---

Implements [SST::Statistics::StatisticOutput](#).

6.255.2.4 `void SST::Statistics::StatisticOutputTxt::implStopOutputEntries () [override], [protected], [virtual]`

Implementation function for the end of output. This will be called by the [Statistic](#) Processing Engine to indicate that a [Statistic](#) is finished sending data to the [Statistic Output](#) for processing. The [Statistic Output](#) can perform any output related functions here.

Implements [SST::Statistics::StatisticOutput](#).

6.255.2.5 void SST::Statistics::StatisticOutputTxt::outputField (fieldHandle_t *fieldHandle*, int32_t *data*) [override], [protected], [virtual]

Implementation functions for output. These will be called by the statistic to provide [Statistic](#) defined data to be output.

Parameters

<i>fieldHandle</i>	- The handle to the registered statistic field.
<i>data</i>	- The data related to the registered field to be output.

Reimplemented from [SST::Statistics::StatisticOutput](#).

6.255.2.6 void SST::Statistics::StatisticOutputTxt::printUsage () [override], [protected], [virtual]

Print out usage for this [Statistic Output](#)

Implements [SST::Statistics::StatisticOutput](#).

6.255.2.7 SST::Statistics::StatisticOutputTxt::SST_ELI_REGISTER_DERIVED (StatisticOutput , StatisticOutputTxt , "sst", "statoutputtxt", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Output directly to console screen")

Construct a StatOutputTxt

Parameters

<i>outputParameters</i>	- Parameters used for this Statistic Output
-------------------------	---

6.255.2.8 void SST::Statistics::StatisticOutputTxt::startOfSimulation () [override], [protected], [virtual]

Indicate to [Statistic Output](#) that simulation started. [Statistic](#) output may perform any startup code here as necessary.

Implements [SST::Statistics::StatisticOutput](#).

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

- sst/core/statapi/statoutputtxt.h
- sst/core/statapi/statoutputtxt.cc

6.256 SST::Statistics::StatisticProcessingEngine Class Reference

```
#include <statengine.h>
```

Public Member Functions

- void [performStatisticOutput](#) ([StatisticBase](#) *stat, bool endOfSimFlag=false)
- void [performGlobalStatisticOutput](#) (bool endOfSimFlag=false)
- template<class T >
[Statistic](#)< T > * **createStatistic** ([BaseComponent](#) *comp, const std::string &type, const std::string &statName, const std::string &statSubId, [Params](#) ¶ms)
- bool **registerStatisticWithEngine** ([StatisticBase](#) *stat, fieldType_t fieldType)
- [StatisticBase](#) * **isStatisticRegisteredWithEngine** (const std::string &compName, const ComponentId_t &compId, const std::string &statName, const std::string &statSubId, fieldType_t fieldId)
- const std::vector< [StatisticOutput](#) * > & **getStatOutputs** () const

Static Public Member Functions

- static [StatisticProcessingEngine](#) * **getInstance** ()

Friends

- class **SST::Simulation**

6.256.1 Detailed Description

An SST core component that handles timing and event processing informing all registered Statistics to generate their outputs at desired rates.

6.256.2 Member Function Documentation

6.256.2.1 void SST::Statistics::StatisticProcessingEngine::performGlobalStatisticOutput (bool *endOfSimFlag* = false)

Called by the Components and Subcomponent to perform a global statistic [Output](#). This routine will force ALL Components and Subcomponents to output their statistic information. This may lead to unexpected results if the statistic counts or data is reset on output.

Parameters

<i>endOfSimFlag</i>	- Indicates that the output is occurring at the end of simulation.
---------------------	--

6.256.2.2 void SST::Statistics::StatisticProcessingEngine::performStatisticOutput ([StatisticBase](#) * *stat*, bool *endOfSimFlag* = false)

Called by the Components and Subcomponent to perform a statistic [Output](#).

Parameters

<i>stat</i>	- Pointer to the statistic.
<i>EndOfSimFlag</i>	- Indicates that the output is occurring at the end of simulation.

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

- sst/core/statapi/statengine.h
- sst/core/statapi/statengine.cc

6.257 StatOutputPy_t Struct Reference

Collaboration diagram for StatOutputPy_t:

Public Attributes

- PyObject_HEAD size_t id
- [SST::ConfigStatOutput](#) * ptr

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

- sst/core/model/pymodel_statgroup.h

6.258 SST::StopAction Class Reference

```
#include <stopAction.h>
```

Inheritance diagram for SST::StopAction:

Collaboration diagram for SST::StopAction:

Public Member Functions

- [StopAction](#) (std::string msg)
- void [execute](#) () override
- void [print](#) (const std::string &header, [Output](#) &out) const override

Additional Inherited Members

6.258.1 Detailed Description

[Action](#) which causes the [Simulation](#) to end

6.258.2 Constructor & Destructor Documentation

6.258.2.1 SST::StopAction::StopAction (std::string msg) [inline]

Create a new [StopAction](#) which includes a message to be printed when it fires

6.258.3 Member Function Documentation

6.258.3.1 void SST::StopAction::execute () [inline],[override],[virtual]

Function which will be called when the time for this [Activity](#) comes to pass.

Implements [SST::Activity](#).

6.258.3.2 void SST::StopAction::print (const std::string & header, Output & out) const [inline],[override],[virtual]

Generic print-print function for this [Activity](#). Subclasses should override this function.

Reimplemented from [SST::Action](#).

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

- sst/core/stopAction.h

6.259 SST::Interfaces::StringEvent Class Reference

```
#include <stringEvent.h>
```

Inheritance diagram for SST::Interfaces::StringEvent:

Collaboration diagram for SST::Interfaces::StringEvent:

Public Member Functions

- [StringEvent](#) (const std::string &str)
- [StringEvent](#) (const [StringEvent](#) &me)
- [StringEvent](#) (const [StringEvent](#) *me)
- const std::string & [getString](#) (void)
- void **serialize_order** ([SST::Core::Serialization::serializer](#) &ser) override
- **ImplementSerializable** ([SST::Interfaces::StringEvent](#))

Additional Inherited Members

6.259.1 Detailed Description

Simple event to pass strings between components

6.259.2 Constructor & Destructor Documentation

6.259.2.1 SST::Interfaces::StringEvent::StringEvent (const std::string & str) [inline]

Create a new [StringEvent](#)

Parameters

<i>str</i>	- The String contents of this event
------------	-------------------------------------

6.259.2.2 SST::Interfaces::StringEvent::StringEvent (const StringEvent & *me*) [inline]

Copies an existing [StringEvent](#)

6.259.2.3 SST::Interfaces::StringEvent::StringEvent (const StringEvent * *me*) [inline]

Copies an existing [StringEvent](#)

6.259.3 Member Function Documentation

6.259.3.1 const std::string& SST::Interfaces::StringEvent::getString (void) [inline]

Returns the contents of this [Event](#)

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

- sst/core/interfaces/stringEvent.h

6.260 SST::SubComponent Class Reference

```
#include <subcomponent.h>
```

Inheritance diagram for SST::SubComponent:

Collaboration diagram for SST::SubComponent:

Public Member Functions

- **SST_ELI_DECLARE_INFO_EXTERN** (ELI::ProvidesParams, ELI::ProvidesSubComponentSlots, ELI::ProvidesPorts, ELI::ProvidesStats, ELI::ProvidesInterface) SubComponent(Component *parent)
- **SubComponent** (ComponentId_t id)
- virtual void [init](#) (unsigned int UNUSED(phase)) override
- virtual void [setup](#) () override
- virtual void [finish](#) () override

Protected Member Functions

- [Component](#) *const parent **__attribute__** ((deprecated("The parent data member will be removed in SST version 10.0. With the new subcomponent structure, direct access to your parent is not allowed.")))

Additional Inherited Members

6.260.1 Detailed Description

[SubComponent](#) is a class loadable through the factory which allows dynamic functionality to be added to a [Component](#). The [SubComponent](#) API is nearly identical to the [Component](#) API and all the calls are proxied to the parent [Component](#).

6.260.2 Member Function Documentation

6.260.2.1 `virtual void SST::SubComponent::finish () [inline],[override],[virtual]`

Called after simulation completes, but before objects are destroyed. A good place to print out statistics.

Reimplemented from [SST::BaseComponent](#).

Reimplemented in [SST::Interfaces::SimpleNetwork](#).

6.260.2.2 `virtual void SST::SubComponent::init (unsigned int UNUSEDphase) [inline],[override],[virtual]`

Used during the init phase. The method will be called each phase of initialization. Initialization ends when no components have sent any data.

Reimplemented from [SST::BaseComponent](#).

Reimplemented in [SST::Interfaces::SimpleNetwork](#).

6.260.2.3 `virtual void SST::SubComponent::setup () [inline],[override],[virtual]`

Called after all components have been constructed and initialization has completed, but before simulation time has begun.

Reimplemented from [SST::BaseComponent](#).

Reimplemented in [SST::Interfaces::SimpleNetwork](#).

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

- `sst/core/subcomponent.h`
- `sst/core/subcomponent.cc`

6.261 SST::SubComponentSlotInfo Class Reference

```
#include <baseComponent.h>
```

Collaboration diagram for `SST::SubComponentSlotInfo`:

Public Member Functions

- **SubComponentSlotInfo** ([BaseComponent](#) *comp, std::string slot_name)
- const std::string & **getSlotName** () const
- bool **isPopulated** (int slot_num) const
- bool **isAllPopulated** () const
- int **getMaxPopulatedSlotNumber** () const
- template<typename T >
 __attribute__((deprecated("This version of create will be removed in SST version 10.0. Please switch to the new user defined API, which includes the share flags."))) T *create(int slot_num)
- template<typename T >
 __attribute__((deprecated("This version of [createAll](#) will be removed in SST version 10.0. Please switch to the new user defined API, which includes the share flags and optional constructor arguments."))) void [createAll](#)([Params](#) ¶ms)
- std::vector< T * > bool class ARGS T * **create** (int slot_num, uint64_t share_flags, ARGS...args) const
- template<typename T , class... ARGS>
 void [createAll](#) (std::vector< T * > &vec, uint64_t share_flags, ARGS...args) const
- template<typename T , class... ARGS>
 void [createAllSparse](#) (std::vector< std::pair< int, T * > > &vec, uint64_t share_flags, ARGS...args) const
- template<typename T , class... ARGS>
 void [createAllSparse](#) (std::vector< T * > &vec, uint64_t share_flags, ARGS...args) const

Public Attributes

- [Params](#) ¶ms **const**
- std::vector< T * > & **vec**
- std::vector< T * > bool **insertNulls**

Protected Member Functions

- [SubComponent](#) * **protected_create** (int slot_num, [Params](#) ¶ms) const

6.261.1 Detailed Description

Used to load SubComponents when multiple SubComponents are loaded into a single slot (will also also work when a single [SubComponent](#) is loaded).

6.261.2 Member Function Documentation

6.261.2.1 template<typename T , class... ARGS> void SST::SubComponentSlotInfo::createAll (std::vector< T * > & *vec*, uint64_t *share_flags*, ARGS... *args*) const [inline]

Create all user defined subcomponents (defined in input file to SST run) for the slot.

Parameters

<i>vec</i>	Vector of T* that will hold the pointers to the new subcomponents. If an index is not occupied, a nullptr will be put in it's place. All components will be added to the end of the vector, so index N will be at vec.length() + N, where vec.length() is the length of the vector when it is passed to the call.
<i>share_flags</i>	Share flags to be used by subcomponent
<i>args</i>	Arguments to be passed to constructor. This signature is defined in the API definition

For ease in backward compatibility to old API, this call will try to load using new API and will fallback to old if unsuccessful.

6.261.2.2 `template<typename T, class... ARGS> void SST::SubComponentSlotInfo::createAllSparse (std::vector< std::pair< int, T * > > & vec, uint64_t share_flags, ARGS... args) const [inline]`

Create all user defined subcomponents (defined in input file to SST run) for the slot.

Parameters

<i>vec</i>	Vector of pair<int,T*> that will hold the pointers to the new subcomponents. The int will hold the index from which the subcomponent was loaded. Unoccupied indexes will be skipped. All components will be added to the end of the vector.
<i>share_flags</i>	Share flags to be used by subcomponent
<i>args</i>	Arguments to be passed to constructor. This signature is defined in the API definition

For ease in backward compatibility to old API, this call will try to load using new API and will fallback to old if unsuccessful.

6.261.2.3 `template<typename T, class... ARGS> void SST::SubComponentSlotInfo::createAllSparse (std::vector< T * > & vec, uint64_t share_flags, ARGS... args) const [inline]`

Create all user defined subcomponents (defined in input file to SST run) for the slot.

Parameters

<i>vec</i>	Vector of T* that will hold the pointers to the new subcomponents. Unoccupied indexes will be skipped. All components will be added to the end of the vector.
<i>share_flags</i>	Share flags to be used by subcomponent
<i>args</i>	Arguments to be passed to constructor. This signature is defined in the API definition

For ease in backward compatibility to old API, this call will try to load using new API and will fallback to old if unsuccessful.

6.261.3 Member Data Documentation

6.261.3.1 Params& params SST::SubComponentSlotInfo::const

Initial value:

```
{
    return private_create<T>(slot_num,params)
```

6.261.3.2 std::vector<T*> bool SST::SubComponentSlotInfo::insertNulls

Initial value:

```

= true) const
{
    return private_createAll<T>(params, vec, insertNulls);
}

template <typename T>
__attribute__((deprecated("This version of createAll will be removed in SST version 10.0. Please
switch to the new user defined API, which includes the share flags and optional constructor arguments.")))
void createAll(std::vector<T*> &vec, bool insertNulls = true) const
{
    Params empty;
    return private_createAll<T>(empty, vec, insertNulls);
}

template <typename T>
T* create(int slot_num) const
{
    Params empty;
    return comp->loadUserSubComponentByIndex<T>(slot_name, slot_num, ComponentInfo::SHARE_NONE);
}

template <class T

```

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

- sst/core/baseComponent.h

6.262 symlist_chain Struct Reference

Collaboration diagram for symlist_chain:

Public Attributes

- struct [symlist_chain](#) * next
- const [lt_dlsymlist](#) * symlist

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

- sst/core/libltdl/loaders/preopen.c

6.263 SST::SyncBase Class Reference

```
#include <syncBase.h>
```

Collaboration diagram for SST::SyncBase:

Public Member Functions

- [SyncBase](#) ()
- virtual [ActivityQueue](#) * [registerLink](#) (const [RankInfo](#) &to_rank, const [RankInfo](#) &from_rank, LinkId_t link_id, [Link](#) *link)=0
- virtual int [exchangeLinkUntimedData](#) (int msg_count)=0
- virtual void [finalizeLinkConfigurations](#) ()=0
- virtual void [setExit](#) ([Exit](#) *ex)
- virtual void [setMaxPeriod](#) ([TimeConverter](#) *period)
- virtual uint64_t [getDataSize](#) () const =0
- virtual [Action](#) * [getSlaveAction](#) ()=0
- virtual [Action](#) * [getMasterAction](#) ()=0

Protected Member Functions

- void [sendUntimedData_sync](#) ([Link](#) *link, [Event](#) *data)
- void [finalizeConfiguration](#) ([Link](#) *link)

Protected Attributes

- [Exit](#) * [exit](#)
- [TimeConverter](#) * [max_period](#)

6.263.1 Detailed Description

[SyncBase](#) defines the API for Sync objects, which are used to synchronize between MPI ranks in a simulation. This is an internal class, and not a public-facing API.

6.263.2 Constructor & Destructor Documentation

6.263.2.1 SST::SyncBase::SyncBase () [inline]

Create a new Sync object which fires with a specified period

6.263.3 Member Function Documentation

6.263.3.1 virtual int SST::SyncBase::exchangeLinkUntimedData (int *msg_count*) [pure virtual]

Cause an exchange of Untimed Data to occur

6.263.3.2 virtual void SST::SyncBase::finalizeLinkConfigurations () [pure virtual]

Finish link configuration

6.263.3.3 `virtual ActivityQueue* SST::SyncBase::registerLink (const RankInfo & to_rank, const RankInfo & from_rank, LinkId_t link_id, Link * link) [pure virtual]`

Register a [Link](#) which this Sync Object is responsible for

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

- sst/core/syncBase.h
- sst/core/syncBase.cc

6.264 SST::SyncManager Class Reference

Inheritance diagram for SST::SyncManager:

Collaboration diagram for SST::SyncManager:

Public Member Functions

- **SyncManager** (const [RankInfo](#) &rank, const [RankInfo](#) &num_ranks, [TimeConverter](#) *minPartTC, SimTime_t min_part, const std::vector< SimTime_t > &interThreadLatencies)
- [ActivityQueue](#) * **registerLink** (const [RankInfo](#) &to_rank, const [RankInfo](#) &from_rank, LinkId_t link_id, [Link](#) *link)
- void **execute** (void) override
- void **exchangeLinkUntimedData** (std::atomic< int > &msg_count)
- void **finalizeLinkConfigurations** ()
- void **prepareForComplete** ()
- void **print** (const std::string &header, [Output](#) &out) const override
- uint64_t **getDataSize** () const

Additional Inherited Members

6.264.1 Member Function Documentation

6.264.1.1 `void SST::SyncManager::exchangeLinkUntimedData (std::atomic< int > & msg_count)`

Cause an exchange of Initialization Data to occur

Cause an exchange of Untimed Data to occur

6.264.1.2 `void SST::SyncManager::execute (void) [override],[virtual]`

Function which will be called when the time for this [Activity](#) comes to pass.

Implements [SST::Activity](#).

6.264.1.3 void SST::SyncManager::finalizeLinkConfigurations ()

Finish link configuration

6.264.1.4 void SST::SyncManager::prepareForComplete ()

Prepare for complete() phase

6.264.1.5 void SST::SyncManager::print (const std::string & header, Output & out) const [override],[virtual]

Generic print-print function for this [Activity](#). Subclasses should override this function.

Reimplemented from [SST::Action](#).

6.264.1.6 ActivityQueue * SST::SyncManager::registerLink (const RankInfo & to_rank, const RankInfo & from_rank, LinkId_t link_id, Link * link)

Register a [Link](#) which this Sync Object is responsible for

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

- sst/core/syncManager.h
- sst/core/syncManager.cc

6.265 SST::SyncQueue Class Reference

```
#include <syncQueue.h>
```

Inheritance diagram for SST::SyncQueue:

Collaboration diagram for SST::SyncQueue:

Classes

- struct [Header](#)

Public Member Functions

- bool [empty](#) () override
- int [size](#) () override
- void [insert](#) ([Activity](#) *activity) override
- [Activity](#) * [pop](#) () override
- [Activity](#) * [front](#) () override
- void [clear](#) ()
- char * [getData](#) ()
- uint64_t [getDataSize](#) ()

6.265.1 Detailed Description

Internal API

[Activity](#) Queue for use by Sync Objects

6.265.2 Member Function Documentation

6.265.2.1 void SST::SyncQueue::clear ()

Clear elements from the queue

6.265.2.2 bool SST::SyncQueue::empty () [override],[virtual]

Returns true if the queue is empty

Implements [SST::ActivityQueue](#).

6.265.2.3 Activity * SST::SyncQueue::front () [override],[virtual]

Returns the next activity

Implements [SST::ActivityQueue](#).

6.265.2.4 char * SST::SyncQueue::getData ()

Accessor method to the internal queue

6.265.2.5 void SST::SyncQueue::insert (Activity * activity) [override],[virtual]

Insert a new activity into the queue

Implements [SST::ActivityQueue](#).

6.265.2.6 Activity * SST::SyncQueue::pop () [override],[virtual]

Remove and return the next activity

Implements [SST::ActivityQueue](#).

6.265.2.7 `int SST::SyncQueue::size () [override],[virtual]`

Returns the number of activities in the queue

Implements [SST::ActivityQueue](#).

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

- `sst/core/syncQueue.h`
- `sst/core/syncQueue.cc`

6.266 SST::Interfaces::TestEvent Class Reference

```
#include <TestEvent.h>
```

Inheritance diagram for `SST::Interfaces::TestEvent`:

Collaboration diagram for `SST::Interfaces::TestEvent`:

Public Member Functions

- `void serialize_order (SST::Core::Serialization::serializer &ser) override`
- `ImplementSerializable (SST::Interfaces::TestEvent)`

Public Attributes

- `int count`
- `bool print_on_delete`

Additional Inherited Members

6.266.1 Detailed Description

Test [Event](#) Useful for early-testing of components.

6.266.2 Member Data Documentation

6.266.2.1 `int SST::Interfaces::TestEvent::count`

Unused

6.266.2.2 bool SST::Interfaces::TestEvent::print_on_delete

Prints a message to stdout when the message is deleted.

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

- sst/core/interfaces/TestEvent.h
- sst/core/interfaces/TestEvent.cc

6.267 SST::ThreadSync Class Reference

Inheritance diagram for SST::ThreadSync:

Collaboration diagram for SST::ThreadSync:

Public Member Functions

- [ThreadSync](#) (int num_threads, [Simulation](#) *sim)
- void **setMaxPeriod** ([TimeConverter](#) *period)
- void [registerLink](#) (LinkId_t link_id, [Link](#) *link)
- [ActivityQueue](#) * **getQueueForThread** (int tid)
- void [execute](#) (void) override
- void [processLinkUntimedData](#) ()
- void [finalizeLinkConfigurations](#) ()
- uint64_t **getDataSize** () const
- void [print](#) (const std::string &header, [Output](#) &out) const override

Static Public Member Functions

- static void **disable** ()

Additional Inherited Members

6.267.1 Constructor & Destructor Documentation

6.267.1.1 SST::ThreadSync::ThreadSync (int num_threads, Simulation * sim)

Create a new [ThreadSync](#) object

6.267.2 Member Function Documentation

6.267.2.1 void SST::ThreadSync::execute (void) [override],[virtual]

Function which will be called when the time for this [Activity](#) comes to pass.

Implements [SST::Activity](#).

6.267.2.2 void SST::ThreadSync::finalizeLinkConfigurations ()

Finish link configuration

6.267.2.3 void SST::ThreadSync::print (const std::string & header, Output & out) const [override],[virtual]

Generic print-print function for this [Activity](#). Subclasses should override this function.

Reimplemented from [SST::Action](#).

6.267.2.4 void SST::ThreadSync::processLinkUntimedData ()

Cause an exchange of Untimed Data to occur

6.267.2.5 void SST::ThreadSync::registerLink (LinkId_t link_id, Link * link)

Register a [Link](#) which this Sync Object is responsible for

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

- sst/core/threadSync.h
- sst/core/threadSync.cc

6.268 SST::ThreadSyncQueue Class Reference

```
#include <threadSyncQueue.h>
```

Inheritance diagram for SST::ThreadSyncQueue:

Collaboration diagram for SST::ThreadSyncQueue:

Public Member Functions

- bool [empty](#) () override
- int [size](#) () override
- [Activity](#) * [pop](#) () override
- void [insert](#) ([Activity](#) *activity) override
- [Activity](#) * [front](#) () override
- void [clear](#) ()
- std::vector< [Activity](#) * > & [getVector](#) ()

6.268.1 Detailed Description

Base Class for a queue of Activities

6.268.2 Member Function Documentation

6.268.2.1 `bool SST::ThreadSyncQueue::empty () [inline],[override],[virtual]`

Returns true if the queue is empty

Implements [SST::ActivityQueue](#).

6.268.2.2 `Activity* SST::ThreadSyncQueue::front () [inline],[override],[virtual]`

Not supported

Implements [SST::ActivityQueue](#).

6.268.2.3 `void SST::ThreadSyncQueue::insert (Activity * activity) [inline],[override],[virtual]`

Insert a new activity into the queue

Implements [SST::ActivityQueue](#).

6.268.2.4 `Activity* SST::ThreadSyncQueue::pop () [inline],[override],[virtual]`

Not supported

Implements [SST::ActivityQueue](#).

6.268.2.5 `int SST::ThreadSyncQueue::size () [inline],[override],[virtual]`

Returns the number of activities in the queue

Implements [SST::ActivityQueue](#).

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

- `sst/core/threadSyncQueue.h`

6.269 SST::ThreadSyncSimpleSkip Class Reference

Inheritance diagram for SST::ThreadSyncSimpleSkip:

Collaboration diagram for SST::ThreadSyncSimpleSkip:

Public Member Functions

- [ThreadSyncSimpleSkip](#) (int num_threads, int thread, [Simulation](#) *sim)
- void **setMaxPeriod** ([TimeConverter](#) *period)
- void **before** () override
- void **after** () override
- void **execute** (void) override
- void [processLinkUntimedData](#) () override
- void [finalizeLinkConfigurations](#) () override
- void **prepareForComplete** () override
- void [registerLink](#) (LinkId_t link_id, [Link](#) *link) override
- [ActivityQueue](#) * **getQueueForThread** (int tid) override
- uint64_t **getDataSize** () const

Additional Inherited Members

6.269.1 Constructor & Destructor Documentation

6.269.1.1 SST::ThreadSyncSimpleSkip::ThreadSyncSimpleSkip (int *num_threads*, int *thread*, [Simulation](#) * *sim*)

Create a new [ThreadSync](#) object

Create a new [ThreadSyncSimpleSkip](#) object

6.269.2 Member Function Documentation

6.269.2.1 void SST::ThreadSyncSimpleSkip::finalizeLinkConfigurations () [override],[virtual]

Finish link configuration

Implements [SST::NewThreadSync](#).

6.269.2.2 void SST::ThreadSyncSimpleSkip::processLinkUntimedData () [override],[virtual]

Cause an exchange of Untimed Data to occur

Implements [SST::NewThreadSync](#).

6.269.2.3 void SST::ThreadSyncSimpleSkip::registerLink (LinkId_t *link_id*, [Link](#) * *link*) [override],[virtual]

Register a [Link](#) which this Sync Object is responsible for

Implements [SST::NewThreadSync](#).

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

- sst/core/threadSyncSimpleSkip.h
- sst/core/threadSyncSimpleSkip.cc

6.270 SST::TimeConverter Class Reference

```
#include <timeConverter.h>
```

Public Member Functions

- SimTime_t [convertToCoreTime](#) (SimTime_t time)
- SimTime_t [convertFromCoreTime](#) (SimTime_t time)
- SimTime_t [getFactor](#) ()

Friends

- class **TimeLord**

6.270.1 Detailed Description

A class to convert between a component's view of time and the core's view of time.

6.270.2 Member Function Documentation

6.270.2.1 SimTime_t SST::TimeConverter::convertFromCoreTime (SimTime_t *time*) [inline]

Converts from the core's view to the components's view of time. The result is truncated, not rounded.

Parameters

<i>time</i>	time to convert from core time
-------------	--------------------------------

6.270.2.2 SimTime_t SST::TimeConverter::convertToCoreTime (SimTime_t *time*) [inline]

Converts from the component's view to the core's view of time.

Parameters

<i>time</i>	time to convert to core time
-------------	------------------------------

6.270.2.3 SimTime_t SST::TimeConverter::getFactor () [inline]

Return the factor used for conversions with Core Time

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

- sst/core/timeConverter.h

6.271 SST::TimeLord Class Reference

```
#include <timeLord.h>
```

Public Member Functions

- [TimeConverter](#) * [getTimeConverter](#) (std::string ts)
- [TimeConverter](#) * [getTimeConverter](#) (const [UnitAlgebra](#) &ts)
- SimTime_t [getSimCycles](#) (std::string timeString, std::string where)
- [UnitAlgebra](#) [getTimeBase](#) () const
- [TimeConverter](#) * [getNano](#) ()
- [TimeConverter](#) * [getMicro](#) ()
- [TimeConverter](#) * [getMilli](#) ()

Friends

- class [SST::Simulation](#)

6.271.1 Detailed Description

Class for creating and managing [TimeConverter](#) objects

6.271.2 Member Function Documentation

6.271.2.1 [TimeConverter](#)* SST::TimeLord::getMicro () [inline]

Return a [TimeConverter](#) which represents Microseconds

6.271.2.2 [TimeConverter](#)* SST::TimeLord::getMilli () [inline]

Return a [TimeConverter](#) which represents Milliseconds

6.271.2.3 [TimeConverter](#)* SST::TimeLord::getNano () [inline]

Return a [TimeConverter](#) which represents Nanoseconds

6.271.2.4 SimTime_t SST::TimeLord::getSimCycles (std::string *timeString*, std::string *where*)

Not a Public API. Returns the number of raw simulation cycles given by a specified time string

6.271.2.5 [UnitAlgebra](#) SST::TimeLord::getTimeBase () const [inline]

Return the Time Base of the [TimeLord](#)

6.271.2.6 [TimeConverter](#) * SST::TimeLord::getTimeConverter (std::string ts)

Create a new [TimeConverter](#) object using specified SI [Units](#). For example, "1 Ghz" (1 Gigahertz), "2.5 ns" (2.5 nanoseconds).

Parameters

<i>ts</i>	String indicating the base unit for this object. The string should be a floating point number followed by a prefix, and then frequency (i.e. Hz) or time unit (s). Allowable seconds prefixes are: 'f' (femto), 'p' (pico), 'n' (nano), 'u' (micro), 'm' (milli). Allowable frequency prefixes are 'k' (kilo), 'M' (mega), and 'G' (giga).
-----------	--

6.271.2.7 TimeConverter * SST::TimeLord::getTimeConverter (const UnitAlgebra & *ts*)

Create a new [TimeConverter](#) object using the specified units.

Parameters

<i>ts</i>	UnitAlgebra object indicating the base unit for this object.
-----------	--

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

- sst/core/timeLord.h
- sst/core/timeLord.cc

6.272 SST::TimeVortex Class Reference

```
#include <timeVortex.h>
```

Inheritance diagram for SST::TimeVortex:

Collaboration diagram for SST::TimeVortex:

Public Member Functions

- virtual bool [empty](#) () override=0
- virtual int [size](#) () override=0
- virtual void [insert](#) ([Activity](#) *activity) override=0
- virtual [Activity](#) * [pop](#) () override=0
- virtual [Activity](#) * [front](#) () override=0
- virtual void [print](#) ([Output](#) &out) const =0
- virtual uint64_t [getMaxDepth](#) () const
- virtual uint64_t [getCurrentDepth](#) () const =0

Protected Attributes

- uint64_t [max_depth](#)

6.272.1 Detailed Description

Primary [Event](#) Queue

6.272.2 Member Function Documentation

6.272.2.1 `virtual bool SST::TimeVortex::empty () [override],[pure virtual]`

Returns true if the queue is empty

Implements [SST::ActivityQueue](#).

Implemented in [SST::IMPL::TimeVortexPQ](#).

6.272.2.2 `virtual Activity* SST::TimeVortex::front () [override],[pure virtual]`

Returns the next activity

Implements [SST::ActivityQueue](#).

Implemented in [SST::IMPL::TimeVortexPQ](#).

6.272.2.3 `virtual void SST::TimeVortex::insert (Activity * activity) [override],[pure virtual]`

Insert a new activity into the queue

Implements [SST::ActivityQueue](#).

Implemented in [SST::IMPL::TimeVortexPQ](#).

6.272.2.4 `virtual Activity* SST::TimeVortex::pop () [override],[pure virtual]`

Remove and return the next activity

Implements [SST::ActivityQueue](#).

Implemented in [SST::IMPL::TimeVortexPQ](#).

6.272.2.5 `virtual void SST::TimeVortex::print (Output & out) const [pure virtual]`

Print the state of the [TimeVortex](#)

Implemented in [SST::IMPL::TimeVortexPQ](#).

6.272.2.6 `virtual int SST::TimeVortex::size () [override],[pure virtual]`

Returns the number of activities in the queue

Implements [SST::ActivityQueue](#).

Implemented in [SST::IMPL::TimeVortexPQ](#).

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

- `sst/core/timeVortex.h`

6.273 SST::IMPL::TimeVortexPQ Class Reference

```
#include <timeVortexPQ.h>
```

Inheritance diagram for SST::IMPL::TimeVortexPQ:

Collaboration diagram for SST::IMPL::TimeVortexPQ:

Public Member Functions

- bool [empty](#) () override
- int [size](#) () override
- void [insert](#) ([Activity](#) *activity) override
- [Activity](#) * [pop](#) () override
- [Activity](#) * [front](#) () override
- void [print](#) ([Output](#) &out) const override
- uint64_t [getCurrentDepth](#) () const override
- uint64_t [getMaxDepth](#) () const override

Additional Inherited Members

6.273.1 Detailed Description

Primary [Event](#) Queue

6.273.2 Member Function Documentation

6.273.2.1 bool SST::IMPL::TimeVortexPQ::empty () [override],[virtual]

Returns true if the queue is empty

Implements [SST::TimeVortex](#).

6.273.2.2 [Activity](#) * SST::IMPL::TimeVortexPQ::front () [override],[virtual]

Returns the next activity

Implements [SST::TimeVortex](#).

6.273.2.3 void SST::IMPL::TimeVortexPQ::insert ([Activity](#) * *activity*) [override],[virtual]

Insert a new activity into the queue

Implements [SST::TimeVortex](#).

6.273.2.4 **Activity** * `SST::IMPL::TimeVortexPQ::pop ()` `[override],[virtual]`

Remove and return the next activity

Implements [SST::TimeVortex](#).

6.273.2.5 `void SST::IMPL::TimeVortexPQ::print (Output & out) const` `[override],[virtual]`

Print the state of the [TimeVortex](#)

Implements [SST::TimeVortex](#).

6.273.2.6 `int SST::IMPL::TimeVortexPQ::size ()` `[override],[virtual]`

Returns the number of activities in the queue

Implements [SST::TimeVortex](#).

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

- `sst/core/impl/timevortex/timeVortexPQ.h`
- `sst/core/impl/timevortex/timeVortexPQ.cc`

6.274 TiXmlAttribute Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlAttribute:

Collaboration diagram for TiXmlAttribute:

Public Member Functions

- [TiXmlAttribute](#) ()
Construct an empty attribute.
- [TiXmlAttribute](#) (const char *_name, const char *_value)
Construct an attribute with a name and value.
- const char * [Name](#) () const
Return the name of this attribute.
- const char * [Value](#) () const
Return the value of this attribute.
- int [IntValue](#) () const
Return the value of this attribute, converted to an integer.
- double [DoubleValue](#) () const
Return the value of this attribute, converted to a double.
- const TIXML_STRING & [NameTStr](#) () const
- int [QueryIntValue](#) (int *_value) const
- int [QueryDoubleValue](#) (double *_value) const

- QueryDoubleValue* examines the value string. See [QueryIntValue\(\)](#).
- void [SetName](#) (const char *_name)
Set the name of this attribute.
- void [SetValue](#) (const char *_value)
Set the value.
- void [SetIntValue](#) (int _value)
Set the value from an integer.
- void [SetDoubleValue](#) (double _value)
Set the value from a double.
- const [TiXmlAttribute](#) * [Next](#) () const
Get the next sibling attribute in the DOM. Returns null at end.
- [TiXmlAttribute](#) * [Next](#) ()
- const [TiXmlAttribute](#) * [Previous](#) () const
Get the previous sibling attribute in the DOM. Returns null at beginning.
- [TiXmlAttribute](#) * [Previous](#) ()
- bool **operator==** (const [TiXmlAttribute](#) &rhs) const
- bool **operator<** (const [TiXmlAttribute](#) &rhs) const
- bool **operator>** (const [TiXmlAttribute](#) &rhs) const
- virtual const char * **Parse** (const char *p, [TiXmlParsingData](#) *data, [TiXmlEncoding](#) encoding)
- virtual void [Print](#) (FILE *cfile, int depth) const
- void **Print** (FILE *cfile, int depth, [TIXML_STRING](#) *str) const
- void **SetDocument** ([TiXmlDocument](#) *doc)

Friends

- class [TiXmlAttributeSet](#)

Additional Inherited Members

6.274.1 Detailed Description

An attribute is a name-value pair. Elements have an arbitrary number of attributes, each with a unique name.

Note

The attributes are not [TiXmlNode](#)s, since they are not part of the tinyXML document object model. There are other suggested ways to look at this problem.

6.274.2 Member Function Documentation

6.274.2.1 virtual void [TiXmlAttribute::Print](#) (FILE * *cfile*, int *depth*) const [inline], [virtual]

All [TinyXml](#) classes can print themselves to a filestream or the string class ([TiXmlString](#) in non-STL mode, `std::string` in STL mode.) Either or both *cfile* and *str* can be null.

This is a formatted print, and will insert tabs and newlines.

(For an unformatted stream, use the `<<` operator.)

Implements [TiXmlBase](#).

6.274.2.2 int TiXmlAttribute::QueryIntValue (int * _value) const

QueryIntValue examines the value string. It is an alternative to the [IntValue\(\)](#) method with richer error checking. If the value is an integer, it is stored in 'value' and the call returns TIXML_SUCCESS. If it is not an integer, it returns TIXML_WRONG_TYPE.

A specialized but useful call. Note that for success it returns 0, which is the opposite of almost all other TinyXml calls.

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp
- sst/core/tinyxml/tinyxmlparser.cpp

6.275 TiXmlAttributeSet Class Reference

Public Member Functions

- void **Add** ([TiXmlAttribute](#) *attribute)
- void **Remove** ([TiXmlAttribute](#) *attribute)
- const [TiXmlAttribute](#) * **First** () const
- [TiXmlAttribute](#) * **First** ()
- const [TiXmlAttribute](#) * **Last** () const
- [TiXmlAttribute](#) * **Last** ()
- [TiXmlAttribute](#) * **Find** (const char * _name) const
- [TiXmlAttribute](#) * **FindOrCreate** (const char * _name)

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp

6.276 TiXmlBase Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlBase:

Collaboration diagram for TiXmlBase:

Public Types

- enum {
TIXML_NO_ERROR = 0, TIXML_ERROR, TIXML_ERROR_OPENING_FILE, TIXML_ERROR_PARSING_ELEMENT,
TIXML_ERROR_FAILED_TO_READ_ELEMENT_NAME, TIXML_ERROR_READING_ELEMENT_VALUE,
TIXML_ERROR_READING_ATTRIBUTES, TIXML_ERROR_PARSING_EMPTY,
TIXML_ERROR_READING_END_TAG, TIXML_ERROR_PARSING_UNKNOWN, TIXML_ERROR_PARSING_COMMENT,
TIXML_ERROR_PARSING_DECLARATION,
TIXML_ERROR_DOCUMENT_EMPTY, TIXML_ERROR_EMBEDDED_NULL, TIXML_ERROR_PARSING_CDATA,
TIXML_ERROR_DOCUMENT_TOP_ONLY,
TIXML_ERROR_STRING_COUNT }

Public Member Functions

- virtual void [Print](#) (FILE *cfile, int depth) const =0
- int [Row](#) () const
- int [Column](#) () const
 - *See [Row\(\)](#)*
- void [SetUserData](#) (void *user)
 - *Set a pointer to arbitrary user data.*
- void * [GetUserData](#) ()
 - *Get a pointer to arbitrary user data.*
- const void * [GetUserData](#) () const
 - *Get a pointer to arbitrary user data.*
- virtual const char * **Parse** (const char *p, [TiXmlParsingData](#) *data, TiXmlEncoding encoding)=0

Static Public Member Functions

- static void [SetCondenseWhiteSpace](#) (bool condense)
- static bool [IsWhiteSpaceCondensed](#) ()
 - *Return the current white space setting.*
- static void [EncodeString](#) (const TIXML_STRING &str, TIXML_STRING *out)

Static Public Attributes

- static const int **utf8ByteTable** [256]

Static Protected Member Functions

- static const char * **SkipWhiteSpace** (const char *, TiXmlEncoding encoding)
- static bool **IsWhiteSpace** (char c)
- static bool **IsWhiteSpace** (int c)
- static const char * **ReadName** (const char *p, TIXML_STRING *name, TiXmlEncoding encoding)
- static const char * **ReadText** (const char *in, TIXML_STRING *text, bool ignoreWhiteSpace, const char *endTag, bool ignoreCase, TiXmlEncoding encoding)
- static const char * **GetEntity** (const char *in, char *value, int *length, TiXmlEncoding encoding)
- static const char * **GetChar** (const char *p, char *_value, int *length, TiXmlEncoding encoding)
- static bool **StringEqual** (const char *p, const char *endTag, bool ignoreCase, TiXmlEncoding encoding)
- static int **IsAlpha** (unsigned char anyByte, TiXmlEncoding encoding)
- static int **IsAlphaNum** (unsigned char anyByte, TiXmlEncoding encoding)
- static int **ToLower** (int v, TiXmlEncoding encoding)
- static void **ConvertUTF32ToUTF8** (unsigned long input, char *output, int *length)

Protected Attributes

- [TiXmlCursor](#) **location**
- void * [userData](#)
 - *Field containing a generic user pointer.*

Static Protected Attributes

- static const char * **errorString** [TIXML_ERROR_STRING_COUNT]

Friends

- class [TiXmlNode](#)
- class [TiXmlElement](#)
- class [TiXmlDocument](#)

6.276.1 Detailed Description

[TiXmlBase](#) is a base class for every class in TinyXml. It does little except to establish that TinyXml classes can be printed and provide some utility functions.

In XML, the document and elements can contain other elements and other types of nodes.

```
A Document can contain: Element (container or leaf)
                        Comment (leaf)
                        Unknown (leaf)
                        Declaration( leaf )
```

```
An Element can contain: Element (container or leaf)
                        Text (leaf)
                        Attributes (not on tree)
                        Comment (leaf)
                        Unknown (leaf)
```

```
A Decleration contains: Attributes (not on tree)
```

6.276.2 Member Function Documentation

6.276.2.1 `void TiXmlBase::EncodeString (const TIXML_STRING & str, TIXML_STRING * out)` [static]

Expands entities in a string. Note this should not contain the tag's '<', '>', etc, or they will be transformed into entities!

6.276.2.2 `virtual void TiXmlBase::Print (FILE * cfile, int depth) const` [pure virtual]

All TinyXml classes can print themselves to a filestream or the string class ([TiXmlString](#) in non-STL mode, `std::string` in STL mode.) Either or both *cfile* and *str* can be null.

This is a formatted print, and will insert tabs and newlines.

(For an unformatted stream, use the `<<` operator.)

Implemented in [TiXmlDocument](#), [TiXmlUnknown](#), [TiXmlDeclaration](#), [TiXmlText](#), [TiXmlComment](#), [TiXmlElement](#), and [TiXmlAttribute](#).

6.276.2.3 int TiXmlBase::Row () const [inline]

Return the position, in the original source file, of this node or attribute. The row and column are 1-based. (That is the first row and first column is 1,1). If the returns values are 0 or less, then the parser does not have a row and column value.

Generally, the row and column value will be set when the TiXmlDocument::Load(), [TiXmlDocument::LoadFile\(\)](#), or any TiXmlNode::Parse() is called. It will NOT be set when the DOM was created from operator>>.

The values reflect the initial load. Once the DOM is modified programmatically (by adding or changing nodes and attributes) the new values will NOT update to reflect changes in the document.

There is a minor performance cost to computing the row and column. Computation can be disabled if [TiXmlDocument::SetTabSize\(\)](#) is called with 0 as the value.

See also

[TiXmlDocument::SetTabSize\(\)](#)

6.276.2.4 static void TiXmlBase::SetCondenseWhiteSpace (bool *condense*) [inline],[static]

The world does not agree on whether white space should be kept or not. In order to make everyone happy, these global, static functions are provided to set whether or not TinyXml will condense all white space into a single space or not. The default is to condense. Note changing this value is not thread safe.

6.276.3 Member Data Documentation

6.276.3.1 const char * TiXmlBase::errorString [static],[protected]

Initial value:

```
=
{
    "No error",
    "Error",
    "Failed to open file",
    "Error parsing Element.",
    "Failed to read Element name",
    "Error reading Element value.",
    "Error reading Attributes.",
    "Error: empty tag.",
    "Error reading end tag.",
    "Error parsing Unknown.",
    "Error parsing Comment.",
    "Error parsing Declaration.",
    "Error document empty.",
    "Error null (0) or unexpected EOF found in input stream.",
    "Error parsing CDATA.",
    "Error when TiXmlDocument added to document, because TiXmlDocument can only be at the root.",
}
```

6.276.3.2 `const int TiXmlBase::utf8ByteTable` [static]

Initial value:

```
=
{
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,
    1,  1,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,
    2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,  2,
    3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,  3,
    4,  4,  4,  4,  4,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1,  1
}
```

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

- `sst/core/tinyxml/tinyxml.h`
- `sst/core/tinyxml/tinyxml.cpp`
- `sst/core/tinyxml/tinyxmlerror.cpp`
- `sst/core/tinyxml/tinyxmlparser.cpp`

6.277 TiXmlComment Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlComment:

Collaboration diagram for TiXmlComment:

Public Member Functions

- [TiXmlComment](#) ()
Constructs an empty comment.
- [TiXmlComment](#) (const char *_value)
Construct a comment from text.
- [TiXmlComment](#) (const [TiXmlComment](#) &)
- [TiXmlComment](#) & **operator=** (const [TiXmlComment](#) &base)
- virtual [TiXmlNode](#) * [Clone](#) () const
Returns a copy of this Comment.
- virtual void [Print](#) (FILE *cfile, int depth) const
- virtual const char * **Parse** (const char *p, [TiXmlParsingData](#) *data, [TiXmlEncoding](#) encoding)
- virtual const [TiXmlComment](#) * [ToComment](#) () const
Cast to a more defined type. Will return null not of the requested type.
- virtual [TiXmlComment](#) * [ToComment](#) ()
Cast to a more defined type. Will return null not of the requested type.
- virtual bool [Accept](#) ([TiXmlVisitor](#) *visitor) const

Protected Member Functions

- void **CopyTo** ([TiXmlComment](#) *target) const

Additional Inherited Members

6.277.1 Detailed Description

An XML comment.

6.277.2 Member Function Documentation

6.277.2.1 bool [TiXmlComment::Accept](#) ([TiXmlVisitor](#) * *visitor*) const [virtual]

Walk the XML tree visiting this node and all of its children.

Implements [TiXmlNode](#).

6.277.2.2 void [TiXmlComment::Print](#) (FILE * *cfile*, int *depth*) const [virtual]

All TinyXml classes can print themselves to a filestream or the string class ([TiXmlString](#) in non-STL mode, std::string in STL mode.) Either or both cfile and str can be null.

This is a formatted print, and will insert tabs and newlines.

(For an unformatted stream, use the << operator.)

Implements [TiXmlBase](#).

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp
- sst/core/tinyxml/tinyxmlparser.cpp

6.278 TiXmlCursor Struct Reference

Public Member Functions

- void **Clear** ()

Public Attributes

- int **row**
- int **col**

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

- sst/core/tinyxml/tinyxml.h

6.279 TiXmlDeclaration Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlDeclaration:

Collaboration diagram for TiXmlDeclaration:

Public Member Functions

- [TiXmlDeclaration](#) ()
Construct an empty declaration.
- [TiXmlDeclaration](#) (const char *_version, const char *_encoding, const char *_standalone)
Construct.
- [TiXmlDeclaration](#) (const [TiXmlDeclaration](#) ©)
- [TiXmlDeclaration](#) & **operator=** (const [TiXmlDeclaration](#) ©)
- const char * [Version](#) () const
Version. Will return an empty string if none was found.
- const char * [Encoding](#) () const
Encoding. Will return an empty string if none was found.
- const char * [Standalone](#) () const
Is this a standalone document?
- virtual [TiXmlNode](#) * [Clone](#) () const
Creates a copy of this Declaration and returns it.
- virtual void **Print** (FILE *cfile, int depth, TIXML_STRING *str) const
- virtual void [Print](#) (FILE *cfile, int depth) const
- virtual const char * **Parse** (const char *p, [TiXmlParsingData](#) *data, TiXmlEncoding encoding)
- virtual const [TiXmlDeclaration](#) * [ToDeclaration](#) () const
Cast to a more defined type. Will return null not of the requested type.
- virtual [TiXmlDeclaration](#) * [ToDeclaration](#) ()
Cast to a more defined type. Will return null not of the requested type.
- virtual bool [Accept](#) ([TiXmlVisitor](#) *visitor) const

Protected Member Functions

- void **CopyTo** ([TiXmlDeclaration](#) *target) const

Additional Inherited Members

6.279.1 Detailed Description

In correct XML the declaration is the first entry in the file.

```
<?xml version="1.0" standalone="yes"?>
```

TinyXml will happily read or write files without a declaration, however. There are 3 possible attributes to the declaration: version, encoding, and standalone.

Note: In this version of the code, the attributes are handled as special cases, not generic attributes, simply because there can only be at most 3 and they are always the same.

6.279.2 Member Function Documentation

6.279.2.1 bool TiXmlDeclaration::Accept (TiXmlVisitor * *visitor*) const [virtual]

Walk the XML tree visiting this node and all of its children.

Implements [TiXmlNode](#).

6.279.2.2 virtual void TiXmlDeclaration::Print (FILE * *cfile*, int *depth*) const [inline],[virtual]

All TinyXml classes can print themselves to a filestream or the string class ([TiXmlString](#) in non-STL mode, `std::string` in STL mode.) Either or both *cfile* and *str* can be null.

This is a formatted print, and will insert tabs and newlines.

(For an unformatted stream, use the `<<` operator.)

Implements [TiXmlBase](#).

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

- `sst/core/tinyxml/tinyxml.h`
- `sst/core/tinyxml/tinyxml.cpp`
- `sst/core/tinyxml/tinyxmlparser.cpp`

6.280 TiXmlDocument Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlDocument:

Collaboration diagram for TiXmlDocument:

Public Member Functions

- [TiXmlDocument](#) ()
Create an empty document, that has no name.
- [TiXmlDocument](#) (const char **documentName*)
Create a document with a name. The name of the document is also the filename of the xml.
- [TiXmlDocument](#) (const [TiXmlDocument](#) &*copy*)
- [TiXmlDocument](#) & **operator=** (const [TiXmlDocument](#) &*copy*)
- bool [LoadFile](#) (TiXmlEncoding *encoding*=TIXML_DEFAULT_ENCODING)
- bool [SaveFile](#) () const
Save a file using the current document value. Returns true if successful.
- bool [LoadFile](#) (const char **filename*, TiXmlEncoding *encoding*=TIXML_DEFAULT_ENCODING)
Load a file using the given filename. Returns true if successful.
- bool [SaveFile](#) (const char **filename*) const
Save a file using the given filename. Returns true if successful.
- bool [LoadFile](#) (FILE *, TiXmlEncoding *encoding*=TIXML_DEFAULT_ENCODING)

- bool [SaveFile](#) (FILE *) const
Save a file using the given FILE. Returns true if successful.*
- virtual const char * [Parse](#) (const char *p, [TiXmlParsingData](#) *data=0, [TiXmlEncoding](#) encoding=[TIXML_DEFAULT_ENCODING](#))
- const [TiXmlElement](#) * [RootElement](#) () const
- [TiXmlElement](#) * [RootElement](#) ()
- bool [Error](#) () const
- const char * [ErrorDesc](#) () const
Contains a textual (english) description of the error if one occurs.
- int [ErrorId](#) () const
- int [ErrorRow](#) () const
- int [ErrorCol](#) () const
The column where the error occurred. See [ErrorRow\(\)](#)
- void [SetTabSize](#) (int _tabsize)
- int [TabSize](#) () const
- void [ClearError](#) ()
- void [Print](#) () const
- virtual void [Print](#) (FILE *cfile, int depth=0) const
Print this Document to a FILE stream.
- void [SetError](#) (int err, const char *errorLocation, [TiXmlParsingData](#) *prevData, [TiXmlEncoding](#) encoding)
- virtual const [TiXmlDocument](#) * [ToDocument](#) () const
Cast to a more defined type. Will return null not of the requested type.
- virtual [TiXmlDocument](#) * [ToDocument](#) ()
Cast to a more defined type. Will return null not of the requested type.
- virtual bool [Accept](#) ([TiXmlVisitor](#) *content) const

Protected Member Functions

- virtual [TiXmlNode](#) * [Clone](#) () const

Additional Inherited Members

6.280.1 Detailed Description

Always the top level node. A document binds together all the XML pieces. It can be saved, loaded, and printed to the screen. The 'value' of a document node is the xml file name.

6.280.2 Member Function Documentation

6.280.2.1 bool [TiXmlDocument::Accept](#) ([TiXmlVisitor](#) * *content*) const [virtual]

Walk the XML tree visiting this node and all of its children.

Implements [TiXmlNode](#).

6.280.2.2 void [TiXmlDocument::ClearError](#) () [inline]

If you have handled the error, it can be reset with this call. The error state is automatically cleared if you Parse a new XML block.

6.280.2.3 `TiXmlNode * TiXmlDocument::Clone () const` `[protected], [virtual]`

Create an exact duplicate of this node and return it. The memory must be deleted by the caller.

Implements [TiXmlNode](#).

6.280.2.4 `bool TiXmlDocument::Error () const` `[inline]`

If an error occurs, Error will be set to true. Also,

- The [ErrorId\(\)](#) will contain the integer identifier of the error (not generally useful)
- The [ErrorDesc\(\)](#) method will return the name of the error. (very useful)
- The [ErrorRow\(\)](#) and [ErrorCol\(\)](#) will return the location of the error (if known)

6.280.2.5 `int TiXmlDocument::ErrorId () const` `[inline]`

Generally, you probably want the error string ([ErrorDesc\(\)](#)). But if you prefer the ErrorId, this function will fetch it.

6.280.2.6 `int TiXmlDocument::ErrorRow () const` `[inline]`

Returns the location (if known) of the error. The first column is column 1, and the first row is row 1. A value of 0 means the row and column wasn't applicable (memory errors, for example, have no row/column) or the parser lost the error. (An error in the error reporting, in that case.)

See also

[SetTabSize](#), [Row](#), [Column](#)

6.280.2.7 `bool TiXmlDocument::LoadFile (TiXmlEncoding encoding = TIXML_DEFAULT_ENCODING)`

Load a file using the current document value. Returns true if successful. Will delete any existing document data before loading.

6.280.2.8 `bool TiXmlDocument::LoadFile (FILE * file, TiXmlEncoding encoding = TIXML_DEFAULT_ENCODING)`

Load a file using the given FILE*. Returns true if successful. Note that this method doesn't stream - the entire object pointed at by the FILE* will be interpreted as an XML file. TinyXML doesn't stream in XML from the current file location. Streaming may be added in the future.

6.280.2.9 `const char * TiXmlDocument::Parse (const char * p, TiXmlParsingData * data = 0, TiXmlEncoding encoding = TIXML_DEFAULT_ENCODING)` `[virtual]`

Parse the given null terminated block of xml data. Passing in an encoding to this method (either TIXML_ENCODING_LEGACY or TIXML_ENCODING_UTF8 will force TinyXml to use that encoding, regardless of what TinyXml might otherwise try to detect.

Implements [TiXmlBase](#).

6.280.2.10 `void TiXmlDocument::Print () const` `[inline]`

Write the document to standard out using formatted printing ("pretty print").

6.280.2.11 `const TiXmlElement* TiXmlDocument::RootElement () const` `[inline]`

Get the root element – the only top level element – of the document. In well formed XML, there should only be one. TinyXml is tolerant of multiple elements at the document level.

6.280.2.12 `void TiXmlDocument::SetTabSize (int _tabsize)` `[inline]`

[SetTabSize\(\)](#) allows the error reporting functions ([ErrorRow\(\)](#) and [ErrorCol\(\)](#)) to report the correct values for row and column. It does not change the output or input in any way.

By calling this method, with a tab size greater than 0, the row and column of each node and attribute is stored when the file is loaded. Very useful for tracking the DOM back in to the source file.

The tab size is required for calculating the location of nodes. If not set, the default of 4 is used. The tabsize is set per document. Setting the tabsize to 0 disables row/column tracking.

Note that row and column tracking is not supported when using operator<>>.

The tab size needs to be enabled before the parse or load. Correct usage:

```
TiXmlDocument doc;
doc.SetTabSize( 8 );
doc.Load( "myfile.xml" );
```

See also

[Row](#), [Column](#)

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp
- sst/core/tinyxml/tinyxmlparser.cpp

6.281 TiXmlElement Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlElement:

Collaboration diagram for TiXmlElement:

Public Member Functions

- [TiXmlElement](#) (const char *_in_value)
Construct an element.
- **TiXmlElement** (const [TiXmlElement](#) &)
- [TiXmlElement](#) & **operator=** (const [TiXmlElement](#) &base)
- const char * [Attribute](#) (const char *name) const
- const char * [Attribute](#) (const char *name, int *i) const
- const char * [Attribute](#) (const char *name, double *d) const
- int [QueryIntAttribute](#) (const char *name, int *_value) const
- int [QueryUnsignedAttribute](#) (const char *name, unsigned *_value) const
QueryUnsignedAttribute examines the attribute - see [QueryIntAttribute\(\)](#).
- int [QueryBoolAttribute](#) (const char *name, bool *_value) const
- int [QueryDoubleAttribute](#) (const char *name, double *_value) const
QueryDoubleAttribute examines the attribute - see [QueryIntAttribute\(\)](#).
- int [QueryFloatAttribute](#) (const char *name, float *_value) const
QueryFloatAttribute examines the attribute - see [QueryIntAttribute\(\)](#).
- void [SetAttribute](#) (const char *name, const char *_value)
- void [SetAttribute](#) (const char *name, int value)
- void [SetDoubleAttribute](#) (const char *name, double value)
- void [RemoveAttribute](#) (const char *name)
- const [TiXmlAttribute](#) * [FirstAttribute](#) () const
Access the first attribute in this element.
- [TiXmlAttribute](#) * **FirstAttribute** ()
- const [TiXmlAttribute](#) * [LastAttribute](#) () const
Access the last attribute in this element.
- [TiXmlAttribute](#) * **LastAttribute** ()
- const char * [GetText](#) () const
- virtual [TiXmlNode](#) * [Clone](#) () const
Creates a new Element and returns it - the returned element is a copy.
- virtual void [Print](#) (FILE *cfile, int depth) const
- virtual const char * **Parse** (const char *p, [TiXmlParsingData](#) *data, [TiXmlEncoding](#) encoding)
- virtual const [TiXmlElement](#) * [ToElement](#) () const
Cast to a more defined type. Will return null not of the requested type.
- virtual [TiXmlElement](#) * **ToElement** ()
Cast to a more defined type. Will return null not of the requested type.
- virtual bool [Accept](#) ([TiXmlVisitor](#) *visitor) const

Protected Member Functions

- void **CopyTo** ([TiXmlElement](#) *target) const
- void **ClearThis** ()
- const char * **ReadValue** (const char *in, [TiXmlParsingData](#) *prevData, [TiXmlEncoding](#) encoding)

Additional Inherited Members

6.281.1 Detailed Description

The element is a container class. It has a value, the element name, and can contain other elements, text, comments, and unknowns. Elements also contain an arbitrary number of attributes.

6.281.2 Member Function Documentation

6.281.2.1 `bool TiXmlElement::Accept (TiXmlVisitor * visitor) const` `[virtual]`

Walk the XML tree visiting this node and all of its children.

Implements [TiXmlNode](#).

6.281.2.2 `const char * TiXmlElement::Attribute (const char * name) const`

Given an attribute name, [Attribute\(\)](#) returns the value for the attribute of that name, or null if none exists.

6.281.2.3 `const char * TiXmlElement::Attribute (const char * name, int * i) const`

Given an attribute name, [Attribute\(\)](#) returns the value for the attribute of that name, or null if none exists. If the attribute exists and can be converted to an integer, the integer value will be put in the return 'i', if 'i' is non-null.

6.281.2.4 `const char * TiXmlElement::Attribute (const char * name, double * d) const`

Given an attribute name, [Attribute\(\)](#) returns the value for the attribute of that name, or null if none exists. If the attribute exists and can be converted to a double, the double value will be put in the return 'd', if 'd' is non-null.

6.281.2.5 `const char * TiXmlElement::GetText () const`

Convenience function for easy access to the text inside an element. Although easy and concise, [GetText\(\)](#) is limited compared to getting the [TiXmlText](#) child and accessing it directly.

If the first child of 'this' is a [TiXmlText](#), the [GetText\(\)](#) returns the character string of the Text node, else null is returned.

This is a convenient method for getting the text of simple contained text:

```
<foo>This is text</foo>
const char* str = fooElement->GetText();
```

'str' will be a pointer to "This is text".

Note that this function can be misleading. If the element foo was created from this XML:

```
<foo><b>This is text</b></foo>
```

then the value of str would be null. The first child node isn't a text node, it is another element. From this XML:

```
<foo>This is <b>text</b></foo>
```

[GetText\(\)](#) will return "This is ".

WARNING: [GetText\(\)](#) accesses a child node - don't become confused with the similarly named [TiXmlHandle::Text\(\)](#) and [TiXmlNode::ToText\(\)](#) which are safe type casts on the referenced node.

6.281.2.6 void TiXmlElement::Print (FILE * *cfile*, int *depth*) const [virtual]

All TinyXml classes can print themselves to a filestream or the string class ([TiXmlString](#) in non-STL mode, std::string in STL mode.) Either or both cfile and str can be null.

This is a formatted print, and will insert tabs and newlines.

(For an unformatted stream, use the << operator.)

Implements [TiXmlBase](#).

6.281.2.7 int TiXmlElement::QueryBoolAttribute (const char * *name*, bool * *_value*) const

QueryBoolAttribute examines the attribute - see [QueryIntAttribute\(\)](#). Note that '1', 'true', or 'yes' are considered true, while '0', 'false' and 'no' are considered false.

6.281.2.8 int TiXmlElement::QueryIntAttribute (const char * *name*, int * *_value*) const

QueryIntAttribute examines the attribute - it is an alternative to the [Attribute\(\)](#) method with richer error checking. If the attribute is an integer, it is stored in 'value' and the call returns TIXML_SUCCESS. If it is not an integer, it returns TIXML_WRONG_TYPE. If the attribute does not exist, then TIXML_NO_ATTRIBUTE is returned.

6.281.2.9 void TiXmlElement::RemoveAttribute (const char * *name*)

Deletes an attribute with the given name.

6.281.2.10 void TiXmlElement::SetAttribute (const char * *name*, const char * *_value*)

Sets an attribute of name to a given value. The attribute will be created if it does not exist, or changed if it does.

6.281.2.11 void TiXmlElement::SetAttribute (const char * *name*, int *value*)

Sets an attribute of name to a given value. The attribute will be created if it does not exist, or changed if it does.

6.281.2.12 void TiXmlElement::SetDoubleAttribute (const char * *name*, double *value*)

Sets an attribute of name to a given value. The attribute will be created if it does not exist, or changed if it does.

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp
- sst/core/tinyxml/tinyxmlparser.cpp

6.282 TiXmlHandle Class Reference

```
#include <tinyxml.h>
```

Public Member Functions

- [TiXmlHandle](#) ([TiXmlNode](#) * _node)
Create a handle from any node (at any depth of the tree.) This can be a null pointer.
- [TiXmlHandle](#) (const [TiXmlHandle](#) &ref)
Copy constructor.
- [TiXmlHandle](#) **operator=** (const [TiXmlHandle](#) &ref)
- [TiXmlHandle](#) [FirstChild](#) () const
Return a handle to the first child node.
- [TiXmlHandle](#) [FirstChild](#) (const char *value) const
Return a handle to the first child node with the given name.
- [TiXmlHandle](#) [FirstChildElement](#) () const
Return a handle to the first child element.
- [TiXmlHandle](#) [FirstChildElement](#) (const char *value) const
Return a handle to the first child element with the given name.
- [TiXmlHandle](#) [Child](#) (const char *value, int index) const
- [TiXmlHandle](#) [Child](#) (int index) const
- [TiXmlHandle](#) [ChildElement](#) (const char *value, int index) const
- [TiXmlHandle](#) [ChildElement](#) (int index) const
- [TiXmlNode](#) * [ToNode](#) () const
- [TiXmlElement](#) * [ToElement](#) () const
- [TiXmlText](#) * [ToText](#) () const
- [TiXmlUnknown](#) * [ToUnknown](#) () const
- [TiXmlNode](#) * [Node](#) () const
- [TiXmlElement](#) * [Element](#) () const
- [TiXmlText](#) * [Text](#) () const
- [TiXmlUnknown](#) * [Unknown](#) () const

6.282.1 Detailed Description

A [TiXmlHandle](#) is a class that wraps a node pointer with null checks; this is an incredibly useful thing. Note that [TiXmlHandle](#) is not part of the TinyXml DOM structure. It is a separate utility class.

Take an example:

```
<Document>
  <Element attributeA = "valueA">
    <Child attributeB = "value1" />
    <Child attributeB = "value2" />
  </Element>
</Document>
```

Assuming you want the value of "attributeB" in the 2nd "Child" element, it's very easy to write a *lot* of code that looks like:

```
TiXmlElement* root = document.FirstChildElement( "Document" );
if ( root )
{
    TiXmlElement* element = root->FirstChildElement( "Element" );
    if ( element )
    {
        TiXmlElement* child = element->FirstChildElement( "Child" );
        if ( child )
        {
            TiXmlElement* child2 = child->NextSiblingElement( "Child" );
            if ( child2 )
            {
                // Finally do something useful.
            }
        }
    }
}
```

And that doesn't even cover "else" cases. [TiXmlHandle](#) addresses the verbosity of such code. A [TiXmlHandle](#) checks for null pointers so it is perfectly safe and correct to use:

```
TiXmlHandle docHandle( &document );
TiXmlElement* child2 = docHandle.FirstChild( "Document" ).FirstChild( "Element" ).Child( "Child", 1 ).ToElement();
if ( child2 )
{
    // do something useful
}
```

Which is MUCH more concise and useful.

It is also safe to copy handles - internally they are nothing more than node pointers.

```
TiXmlHandle handleCopy = handle;
```

What they should not be used for is iteration:

```
int i=0;
while ( true )
{
    TiXmlElement* child = docHandle.FirstChild( "Document" ).FirstChild( "Element" ).Child( "Child", i ).ToElement();
    if ( !child )
        break;
    // do something
    ++i;
}
```

It seems reasonable, but it is in fact two embedded while loops. The Child method is a linear walk to find the element, so this code would iterate much more than it needs to. Instead, prefer:

```
TiXmlElement* child = docHandle.FirstChild( "Document" ).FirstChild( "Element" ).FirstChild( "Child" ).ToElement();
for( child; child; child=child->NextSiblingElement() )
{
    // do something
}
```

6.282.2 Member Function Documentation

6.282.2.1 TiXmlHandle TiXmlHandle::Child(const char * value, int index) const

Return a handle to the "index" child with the given name. The first child is 0, the second 1, etc.

6.282.2.2 `TiXmlHandle TiXmlHandle::Child (int index) const`

Return a handle to the "index" child. The first child is 0, the second 1, etc.

6.282.2.3 `TiXmlHandle TiXmlHandle::ChildElement (const char * value, int index) const`

Return a handle to the "index" child element with the given name. The first child element is 0, the second 1, etc. Note that only `TiXmlElement`s are indexed: other types are not counted.

6.282.2.4 `TiXmlHandle TiXmlHandle::ChildElement (int index) const`

Return a handle to the "index" child element. The first child element is 0, the second 1, etc. Note that only `TiXmlElement`s are indexed: other types are not counted.

6.282.2.5 `TiXmlElement* TiXmlHandle::Element () const` `[inline]`

Deprecated use `ToElement`. Return the handle as a `TiXmlElement`. This may return null.

6.282.2.6 `TiXmlNode* TiXmlHandle::Node () const` `[inline]`

Deprecated use `ToNode`. Return the handle as a `TiXmlNode`. This may return null.

6.282.2.7 `TiXmlText* TiXmlHandle::Text () const` `[inline]`

Deprecated use `ToText()` Return the handle as a `TiXmlText`. This may return null.

6.282.2.8 `TiXmlElement* TiXmlHandle::ToElement () const` `[inline]`

Return the handle as a `TiXmlElement`. This may return null.

6.282.2.9 `TiXmlNode* TiXmlHandle::ToNode () const` `[inline]`

Return the handle as a `TiXmlNode`. This may return null.

6.282.2.10 `TiXmlText* TiXmlHandle::ToText () const` `[inline]`

Return the handle as a `TiXmlText`. This may return null.

6.282.2.11 `TiXmlNode*` `TiXmlHandle::ToUnknown () const` `[inline]`

Return the handle as a [TiXmlNode](#). This may return null.

6.282.2.12 `TiXmlNode*` `TiXmlHandle::Unknown () const` `[inline]`

Deprecated use `ToUnknown()` Return the handle as a [TiXmlNode](#). This may return null.

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

- `sst/core/tinyxml/tinyxml.h`
- `sst/core/tinyxml/tinyxml.cpp`

6.283 TiXmlNode Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for `TiXmlNode`:

Collaboration diagram for `TiXmlNode`:

Public Types

- enum `NodeType` {
`TINYXML_DOCUMENT`, `TINYXML_ELEMENT`, `TINYXML_COMMENT`, `TINYXML_UNKNOWN`,
`TINYXML_TEXT`, `TINYXML_DECLARATION`, `TINYXML_TYPECOUNT` }

Public Member Functions

- `const char *` `Value () const`
- `const TIXML_STRING &` `ValueTStr () const`
- `void` `SetValue` (`const char *_value`)
- `void` `Clear ()`
Delete all the children of this node. Does not affect 'this'.
- `TiXmlNode *` `Parent ()`
One step up the DOM.
- `const TiXmlNode *` `Parent () const`
- `const TiXmlNode *` `FirstChild () const`
The first child of this node. Will be null if there are no children.
- `TiXmlNode *` `FirstChild ()`
- `const TiXmlNode *` `FirstChild` (`const char *value`) `const`
- `TiXmlNode *` `FirstChild` (`const char *_value`)
The first child of this node with the matching 'value'. Will be null if none found.
- `const TiXmlNode *` `LastChild () const`
- `TiXmlNode *` `LastChild ()`
The last child of this node. Will be null if there are no children.
- `const TiXmlNode *` `LastChild` (`const char *value`) `const`

- [TiXmlNode](#) * [LastChild](#) (const char *_value)
The last child of this node matching 'value'. Will be null if there are no children.
- const [TiXmlNode](#) * [IterateChildren](#) (const [TiXmlNode](#) *previous) const
- [TiXmlNode](#) * [IterateChildren](#) (const [TiXmlNode](#) *previous)
- const [TiXmlNode](#) * [IterateChildren](#) (const char *value, const [TiXmlNode](#) *previous) const
This flavor of IterateChildren searches for children with a particular 'value'.
- [TiXmlNode](#) * [IterateChildren](#) (const char *_value, const [TiXmlNode](#) *previous)
- [TiXmlNode](#) * [InsertEndChild](#) (const [TiXmlNode](#) &addThis)
- [TiXmlNode](#) * [LinkEndChild](#) ([TiXmlNode](#) *addThis)
- [TiXmlNode](#) * [InsertBeforeChild](#) ([TiXmlNode](#) *beforeThis, const [TiXmlNode](#) &addThis)
- [TiXmlNode](#) * [InsertAfterChild](#) ([TiXmlNode](#) *afterThis, const [TiXmlNode](#) &addThis)
- [TiXmlNode](#) * [ReplaceChild](#) ([TiXmlNode](#) *replaceThis, const [TiXmlNode](#) &withThis)
- bool [RemoveChild](#) ([TiXmlNode](#) *removeThis)
Delete a child of this node.
- const [TiXmlNode](#) * [PreviousSibling](#) () const
Navigate to a sibling node.
- [TiXmlNode](#) * [PreviousSibling](#) ()
- const [TiXmlNode](#) * [PreviousSibling](#) (const char *) const
Navigate to a sibling node.
- [TiXmlNode](#) * [PreviousSibling](#) (const char *_prev)
- const [TiXmlNode](#) * [NextSibling](#) () const
Navigate to a sibling node.
- [TiXmlNode](#) * [NextSibling](#) ()
- const [TiXmlNode](#) * [NextSibling](#) (const char *) const
Navigate to a sibling node with the given 'value'.
- [TiXmlNode](#) * [NextSibling](#) (const char *_next)
- const [TiXmlElement](#) * [NextSiblingElement](#) () const
- [TiXmlElement](#) * [NextSiblingElement](#) ()
- const [TiXmlElement](#) * [NextSiblingElement](#) (const char *) const
- [TiXmlElement](#) * [NextSiblingElement](#) (const char *_next)
- const [TiXmlElement](#) * [FirstChildElement](#) () const
Convenience function to get through elements.
- [TiXmlElement](#) * [FirstChildElement](#) ()
- const [TiXmlElement](#) * [FirstChildElement](#) (const char *_value) const
Convenience function to get through elements.
- [TiXmlElement](#) * [FirstChildElement](#) (const char *_value)
- int [Type](#) () const
- const [TiXmlDocument](#) * [GetDocument](#) () const
- [TiXmlDocument](#) * [GetDocument](#) ()
- bool [NoChildren](#) () const
Returns true if this node has no children.
- virtual const [TiXmlDocument](#) * [ToDocument](#) () const
Cast to a more defined type. Will return null if not of the requested type.
- virtual const [TiXmlElement](#) * [ToElement](#) () const
Cast to a more defined type. Will return null if not of the requested type.
- virtual const [TiXmlComment](#) * [ToComment](#) () const
Cast to a more defined type. Will return null if not of the requested type.
- virtual const [TiXmlUnknown](#) * [ToUnknown](#) () const
Cast to a more defined type. Will return null if not of the requested type.
- virtual const [TiXmlText](#) * [ToText](#) () const
Cast to a more defined type. Will return null if not of the requested type.
- virtual const [TiXmlDeclaration](#) * [ToDeclaration](#) () const

- Cast to a more defined type. Will return null if not of the requested type.*
- virtual [TiXmlDocument](#) * [ToDocument](#) ()
- Cast to a more defined type. Will return null if not of the requested type.*
- virtual [TiXmlElement](#) * [ToElement](#) ()
- Cast to a more defined type. Will return null if not of the requested type.*
- virtual [TiXmlComment](#) * [ToComment](#) ()
- Cast to a more defined type. Will return null if not of the requested type.*
- virtual [TiXmlUnknown](#) * [ToUnknown](#) ()
- Cast to a more defined type. Will return null if not of the requested type.*
- virtual [TiXmlText](#) * [ToText](#) ()
- Cast to a more defined type. Will return null if not of the requested type.*
- virtual [TiXmlDeclaration](#) * [ToDeclaration](#) ()
- Cast to a more defined type. Will return null if not of the requested type.*
- virtual [TiXmlNode](#) * [Clone](#) () const =0
- virtual bool [Accept](#) ([TiXmlVisitor](#) *visitor) const =0

Protected Member Functions

- [TiXmlNode](#) ([NodeType](#) _type)
- void [CopyTo](#) ([TiXmlNode](#) *target) const
- [TiXmlNode](#) * [Identify](#) (const char *start, [TiXmlEncoding](#) encoding)

Protected Attributes

- [TiXmlNode](#) * [parent](#)
- [NodeType](#) [type](#)
- [TiXmlNode](#) * [firstChild](#)
- [TiXmlNode](#) * [lastChild](#)
- [TIXML_STRING](#) [value](#)
- [TiXmlNode](#) * [prev](#)
- [TiXmlNode](#) * [next](#)

Friends

- class [TiXmlDocument](#)
- class [TiXmlElement](#)

Additional Inherited Members

6.283.1 Detailed Description

The parent class for everything in the Document Object Model. (Except for attributes). Nodes have siblings, a parent, and children. A node can be in a document, or stand on its own. The type of a [TiXmlNode](#) can be queried, and it can be cast to its more defined type.

6.283.2 Member Enumeration Documentation

6.283.2.1 enum `TiXmlNode::NodeType`

The types of XML nodes supported by TinyXml. (All the unsupported types are picked up by UNKNOWN.)

6.283.3 Member Function Documentation

6.283.3.1 `virtual bool TiXmlNode::Accept (TiXmlVisitor * visitor) const` [pure virtual]

Accept a hierarchical visit the nodes in the TinyXML DOM. Every node in the XML tree will be conditionally visited and the host will be called back via the [TiXmlVisitor](#) interface.

This is essentially a SAX interface for TinyXML. (Note however it doesn't re-parse the XML for the callbacks, so the performance of TinyXML is unchanged by using this interface versus any other.)

The interface has been based on ideas from:

- <http://www.saxproject.org/>
- <http://c2.com/cgi/wiki?HierarchicalVisitorPattern>

Which are both good references for "visiting".

An example of using [Accept\(\)](#):

```
TiXmlPrinter printer;
tinyxmlDoc.Accept( &printer );
const char* xmlcstr = printer.CStr();
```

Implemented in [TiXmlDocument](#), [TiXmlUnknown](#), [TiXmlDeclaration](#), [TiXmlText](#), [TiXmlComment](#), and [TiXmlElement](#).

6.283.3.2 `virtual TiXmlNode* TiXmlNode::Clone () const` [pure virtual]

Create an exact duplicate of this node and return it. The memory must be deleted by the caller.

Implemented in [TiXmlDocument](#), [TiXmlUnknown](#), [TiXmlDeclaration](#), [TiXmlText](#), [TiXmlComment](#), and [TiXmlElement](#).

6.283.3.3 `const TiXmlNode * TiXmlNode::FirstChild (const char * value) const`

The first child of this node with the matching 'value'. Will be null if none found.

6.283.3.4 `const TiXmlDocument * TiXmlNode::GetDocument () const`

Return a pointer to the Document this node lives in. Returns null if not in a document.

6.283.3.5 TiXmlNode * TiXmlNode::InsertAfterChild (TiXmlNode * *afterThis*, const TiXmlNode & *addThis*)

Add a new node related to this. Adds a child after the specified child. Returns a pointer to the new object or NULL if an error occurred.

6.283.3.6 TiXmlNode * TiXmlNode::InsertBeforeChild (TiXmlNode * *beforeThis*, const TiXmlNode & *addThis*)

Add a new node related to this. Adds a child before the specified child. Returns a pointer to the new object or NULL if an error occurred.

6.283.3.7 TiXmlNode * TiXmlNode::InsertEndChild (const TiXmlNode & *addThis*)

Add a new node related to this. Adds a child past the LastChild. Returns a pointer to the new object or NULL if an error occurred.

6.283.3.8 const TiXmlNode * TiXmlNode::IterateChildren (const TiXmlNode * *previous*) const

An alternate way to walk the children of a node. One way to iterate over nodes is:

```
for( child = parent->FirstChild(); child; child = child->NextSibling() )
```

IterateChildren does the same thing with the syntax:

```
child = 0;
while( child = parent->IterateChildren( child ) )
```

IterateChildren takes the previous child as input and finds the next one. If the previous child is null, it returns the first. IterateChildren will return null when done.

6.283.3.9 TiXmlNode * TiXmlNode::LinkEndChild (TiXmlNode * *addThis*)

Add a new node related to this. Adds a child past the LastChild.

NOTE: the node to be added is passed by pointer, and will be henceforth owned (and deleted) by tinyXml. This method is efficient and avoids an extra copy, but should be used with care as it uses a different memory model than the other insert functions.

See also

[InsertEndChild](#)

6.283.3.10 const TiXmlElement * TiXmlNode::NextSiblingElement () const

Convenience function to get through elements. Calls NextSibling and ToElement. Will skip all non-Element nodes. Returns 0 if there is not another element.

6.283.3.11 `const TiXmlElement * TiXmlNode::NextSiblingElement (const char * _value) const`

Convenience function to get through elements. Calls NextSibling and ToElement. Will skip all non-Element nodes. Returns 0 if there is not another element.

6.283.3.12 `TiXmlNode * TiXmlNode::ReplaceChild (TiXmlNode * replaceThis, const TiXmlNode & withThis)`

Replace a child of this node. Returns a pointer to the new object or NULL if an error occurred.

6.283.3.13 `void TiXmlNode::SetValue (const char * _value) [inline]`

Changes the value of the node. Defined as:

```
Document:  filename of the xml file
Element:   name of the element
Comment:   the comment text
Unknown:   the tag contents
Text:      the text string
```

6.283.3.14 `int TiXmlNode::Type () const [inline]`

Query the type (as an enumerated value, above) of this node. The possible types are: TINYXML_DOCUMENT, TINYXML_ELEMENT, TINYXML_COMMENT, TINYXML_UNKNOWN, TINYXML_TEXT, and TINYXML_DECLARATION.

6.283.3.15 `const char* TiXmlNode::Value () const [inline]`

The meaning of 'value' changes for the specific type of [TiXmlNode](#).

```
Document:  filename of the xml file
Element:   name of the element
Comment:   the comment text
Unknown:   the tag contents
Text:      the text string
```

The subclasses will wrap this function.

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp
- sst/core/tinyxml/tinyxmlparser.cpp

6.284 TiXmlOutputStream Class Reference

Inheritance diagram for TiXmlOutputStream:

Collaboration diagram for TiXmlOutputStream:

Public Member Functions

- [TiXmlOutputStream](#) & **operator**<< (const [TiXmlString](#) &in)
- [TiXmlOutputStream](#) & **operator**<< (const char *in)

Additional Inherited Members

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

- sst/core/tinyxml/tinystl.h

6.285 TiXmlParsingData Class Reference

Public Member Functions

- void **Stamp** (const char *now, TiXmlEncoding encoding)
- const [TiXmlCursor](#) & **Cursor** () const

Friends

- class **TiXmlDocument**

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

- sst/core/tinyxml/tinyxmlparser.cpp

6.286 TiXmlPrinter Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlPrinter:

Collaboration diagram for TiXmlPrinter:

Public Member Functions

- virtual bool [VisitEnter](#) (const [TiXmlDocument](#) &doc)
Visit a document.
- virtual bool [VisitExit](#) (const [TiXmlDocument](#) &doc)
Visit a document.
- virtual bool [VisitEnter](#) (const [TiXmlElement](#) &element, const [TiXmlAttribute](#) *firstAttribute)
Visit an element.
- virtual bool [VisitExit](#) (const [TiXmlElement](#) &element)
Visit an element.
- virtual bool [Visit](#) (const [TiXmlDeclaration](#) &declaration)
Visit a declaration.
- virtual bool [Visit](#) (const [TiXmlText](#) &text)
Visit a text node.
- virtual bool [Visit](#) (const [TiXmlComment](#) &comment)
Visit a comment node.
- virtual bool [Visit](#) (const [TiXmlUnknown](#) &unknown)
Visit an unknown node.
- void [SetIndent](#) (const char *_indent)
- const char * [Indent](#) ()
Query the indentation string.
- void [SetLineBreak](#) (const char *_lineBreak)
- const char * [LineBreak](#) ()
Query the current line breaking string.
- void [SetStreamPrinting](#) ()
- const char * [CStr](#) ()
Return the result.
- size_t [Size](#) ()
Return the length of the result string.

6.286.1 Detailed Description

Print to memory functionality. The [TiXmlPrinter](#) is useful when you need to:

1. Print to memory (especially in non-STL mode)
2. Control formatting (line endings, etc.)

When constructed, the [TiXmlPrinter](#) is in its default "pretty printing" mode. Before calling [Accept\(\)](#) you can call methods to control the printing of the XML document. After [TiXmlNode::Accept\(\)](#) is called, the printed document can be accessed via the [CStr\(\)](#), [Str\(\)](#), and [Size\(\)](#) methods.

[TiXmlPrinter](#) uses the Visitor API.

```
TiXmlPrinter printer;
printer.SetIndent( "\\t" );

doc.Accept( &printer );
fprintf( stdout, "%s", printer.CStr() );
```

6.286.2 Member Function Documentation

6.286.2.1 void TiXmlPrinter::SetIndent (const char * _indent) [inline]

Set the indent characters for printing. By default 4 spaces but tab () is also useful, or null/empty string for no indentation.

6.286.2.2 void TiXmlPrinter::SetLineBreak (const char * _lineBreak) [inline]

Set the line breaking string. By default set to newline (). Some operating systems prefer other characters, or can be set to the null/empty string for no indentation.

6.286.2.3 void TiXmlPrinter::SetStreamPrinting () [inline]

Switch over to "stream printing" which is the most dense formatting without linebreaks. Common when the XML is needed for network transmission.

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp

6.287 TiXmlString Class Reference

Inheritance diagram for TiXmlString:

Public Types

- typedef size_t **size_type**

Public Member Functions

- **TiXmlString** (const [TiXmlString](#) ©)
- **TIXML_EXPLICIT TiXmlString** (const char *copy)
- **TIXML_EXPLICIT TiXmlString** (const char *str, size_type len)
- [TiXmlString](#) & **operator=** (const char *copy)
- [TiXmlString](#) & **operator=** (const [TiXmlString](#) ©)
- [TiXmlString](#) & **operator+=** (const char *suffix)
- [TiXmlString](#) & **operator+=** (char single)
- [TiXmlString](#) & **operator+=** (const [TiXmlString](#) &suffix)
- const char * **c_str** () const
- const char * **data** () const
- size_type **length** () const
- size_type **size** () const
- bool **empty** () const
- size_type **capacity** () const
- const char & **at** (size_type index) const
- char & **operator[]** (size_type index) const
- size_type **find** (char lookup) const
- size_type **find** (char tofind, size_type offset) const
- void **clear** ()
- void **reserve** (size_type cap)
- [TiXmlString](#) & **assign** (const char *str, size_type len)
- [TiXmlString](#) & **append** (const char *str, size_type len)
- void **swap** ([TiXmlString](#) &other)

Static Public Attributes

- static const size_type **npos** = static_cast< TiXmlString::size_type >(-1)

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

- sst/core/tinyxml/tinyst.h
- sst/core/tinyxml/tinyst.cpp

6.288 TiXmlText Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlText:

Collaboration diagram for TiXmlText:

Public Member Functions

- [TiXmlText](#) (const char *initValue)
- **TiXmlText** (const [TiXmlText](#) ©)
- [TiXmlText](#) & **operator=** (const [TiXmlText](#) &base)
- virtual void **Print** (FILE *cfile, int depth) const
- bool **CDATA** () const
Queries whether this represents text using a CDATA section.
- void **SetCDATA** (bool _cdata)
Turns on or off a CDATA representation of text.
- virtual const char * **Parse** (const char *p, [TiXmlParsingData](#) *data, TiXmlEncoding encoding)
- virtual const [TiXmlText](#) * **ToText** () const
Cast to a more defined type. Will return null not of the requested type.
- virtual [TiXmlText](#) * **ToText** ()
Cast to a more defined type. Will return null not of the requested type.
- virtual bool **Accept** ([TiXmlVisitor](#) *content) const

Protected Member Functions

- virtual [TiXmlNode](#) * **Clone** () const
[internal use] Creates a new Element and returns it.
- void **CopyTo** ([TiXmlText](#) *target) const
- bool **Blank** () const

Friends

- class **TiXmlElement**

Additional Inherited Members

6.288.1 Detailed Description

XML text. A text node can have 2 ways to output the next. "normal" output and CDATA. It will default to the mode it was parsed from the XML file and you generally want to leave it alone, but you can change the output mode with [SetCDATA\(\)](#) and query it with [CDATA\(\)](#).

6.288.2 Constructor & Destructor Documentation

6.288.2.1 `TiXmlText::TiXmlText (const char * initValue) [inline]`

Constructor for text element. By default, it is treated as normal, encoded text. If you want it be output as a CDATA text element, set the parameter `_cdata` to 'true'

6.288.3 Member Function Documentation

6.288.3.1 `bool TiXmlText::Accept (TiXmlVisitor * content) const [virtual]`

Walk the XML tree visiting this node and all of its children.

Implements [TiXmlNode](#).

6.288.3.2 `void TiXmlText::Print (FILE * cfile, int depth) const [virtual]`

All TinyXml classes can print themselves to a filestream or the string class ([TiXmlString](#) in non-STL mode, `std::string` in STL mode.) Either or both `cfile` and `str` can be null.

This is a formatted print, and will insert tabs and newlines.

(For an unformatted stream, use the `<<` operator.)

Implements [TiXmlBase](#).

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

- `sst/core/tinyxml/tinyxml.h`
- `sst/core/tinyxml/tinyxml.cpp`
- `sst/core/tinyxml/tinyxmlparser.cpp`

6.289 TiXmlUnknown Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlUnknown:

Collaboration diagram for TiXmlUnknown:

Public Member Functions

- **TiXmlUnknown** (const [TiXmlUnknown](#) ©)
- [TiXmlUnknown](#) & **operator=** (const [TiXmlUnknown](#) ©)
- virtual [TiXmlNode](#) * **Clone** () const
Creates a copy of this Unknown and returns it.
- virtual void **Print** (FILE *cfile, int depth) const
- virtual const char * **Parse** (const char *p, [TiXmlParsingData](#) *data, TiXmlEncoding encoding)
- virtual const [TiXmlUnknown](#) * **ToUnknown** () const
Cast to a more defined type. Will return null not of the requested type.
- virtual [TiXmlUnknown](#) * **ToUnknown** ()
Cast to a more defined type. Will return null not of the requested type.
- virtual bool **Accept** ([TiXmlVisitor](#) *content) const

Protected Member Functions

- void **CopyTo** ([TiXmlUnknown](#) *target) const

Additional Inherited Members

6.289.1 Detailed Description

Any tag that tinyXml doesn't recognize is saved as an unknown. It is a tag of text, but should not be modified. It will be written back to the XML, unchanged, when the file is saved.

DTD tags get thrown into TiXmlUnknowns.

6.289.2 Member Function Documentation

6.289.2.1 bool TiXmlUnknown::Accept ([TiXmlVisitor](#) * *content*) const [virtual]

Walk the XML tree visiting this node and all of its children.

Implements [TiXmlNode](#).

6.289.2.2 void TiXmlUnknown::Print (FILE * *cfile*, int *depth*) const [virtual]

All TinyXml classes can print themselves to a filestream or the string class ([TiXmlString](#) in non-STL mode, std::string in STL mode.) Either or both cfile and str can be null.

This is a formatted print, and will insert tabs and newlines.

(For an unformatted stream, use the << operator.)

Implements [TiXmlBase](#).

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

- sst/core/tinyxml/tinyxml.h
- sst/core/tinyxml/tinyxml.cpp
- sst/core/tinyxml/tinyxmlparser.cpp

6.290 TiXmlVisitor Class Reference

```
#include <tinyxml.h>
```

Inheritance diagram for TiXmlVisitor:

Public Member Functions

- virtual bool [VisitEnter](#) (const [TiXmlDocument](#) &)
Visit a document.
- virtual bool [VisitExit](#) (const [TiXmlDocument](#) &)
Visit a document.
- virtual bool [VisitEnter](#) (const [TiXmlElement](#) &, const [TiXmlAttribute](#) *)
Visit an element.
- virtual bool [VisitExit](#) (const [TiXmlElement](#) &)
Visit an element.
- virtual bool [Visit](#) (const [TiXmlDeclaration](#) &)
Visit a declaration.
- virtual bool [Visit](#) (const [TiXmlText](#) &)
Visit a text node.
- virtual bool [Visit](#) (const [TiXmlComment](#) &)
Visit a comment node.
- virtual bool [Visit](#) (const [TiXmlUnknown](#) &)
Visit an unknown node.

6.290.1 Detailed Description

Implements the interface to the "Visitor pattern" (see the [Accept\(\)](#) method.) If you call the [Accept\(\)](#) method, it requires being passed a [TiXmlVisitor](#) class to handle callbacks. For nodes that contain other nodes (Document, Element) you will get called with a [VisitEnter](#)/[VisitExit](#) pair. Nodes that are always leaves are simply called with [Visit\(\)](#).

If you return 'true' from a Visit method, recursive parsing will continue. If you return false, **no children of this node or its siblings** will be Visited.

All flavors of Visit methods have a default implementation that returns 'true' (continue visiting). You need to only override methods that are interesting to you.

Generally [Accept\(\)](#) is called on the [TiXmlDocument](#), although all nodes support Visiting.

You should never change the document from a callback.

See also

[TiXmlNode::Accept\(\)](#)

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

- `sst/core/tinyxml/tinyxml.h`

6.291 SST::Tokenizer< TokenizerFunc > Class Template Reference

Public Types

- typedef iter **iterator**
- typedef iter **const_iterator**
- typedef std::string **value_type**

Public Member Functions

- iter **begin** ()
- iter **end** ()
- **Tokenizer** (const std::string &s, const TokenizerFunc &f=TokenizerFunc())

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

- sst/core/stringize.h

6.292 SST::TraceFunction Class Reference

Public Member Functions

- **TraceFunction** (uint32_t line, const char *file, const char *func, bool print_sim_info=true)
- [Output](#) & **getOutput** ()
- void [output](#) (const char *format,...) const `__attribute__((format(printf`

6.292.1 Member Function Documentation

6.292.1.1 void SST::TraceFunction::output (const char * *format*, ...) const

[Output](#) the message with formatting as specified by the format parameter.

Parameters

<i>format</i>	Format string. All valid formats for printf are available.
...	Arguments for format.

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

- sst/core/output.h
- sst/core/output.cc

6.293 SST::Core::ThreadSafe::UnboundedQueue< T > Class Template Reference

Public Member Functions

- void **insert** (const T &t)
- bool **try_remove** (T &result)
- T **remove** ()

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

- sst/core/threadsafe.h

6.294 SST::UninitializedQueue Class Reference

Used for debugging, and preventing accidentally sending messages into an incorrect queue.

```
#include <uninitializedQueue.h>
```

Inheritance diagram for SST::UninitializedQueue:

Collaboration diagram for SST::UninitializedQueue:

Public Member Functions

- [UninitializedQueue](#) (std::string message)
- bool [empty](#) () override
- int [size](#) () override
- void [insert](#) ([Activity](#) *activity) override
- [Activity](#) * [pop](#) () override
- [Activity](#) * [front](#) () override

6.294.1 Detailed Description

Used for debugging, and preventing accidentally sending messages into an incorrect queue.

Always uninitialized queue

6.294.2 Constructor & Destructor Documentation

6.294.2.1 SST::UninitializedQueue::UninitializedQueue (std::string *message*)

Create a new Queue

Parameters

<i>message</i>	- Message to print when something attempts to use this Queue
----------------	--

6.294.3 Member Function Documentation

6.294.3.1 `bool SST::UninitializedQueue::empty () [override],[virtual]`

Returns true if the queue is empty

Implements [SST::ActivityQueue](#).

6.294.3.2 `Activity * SST::UninitializedQueue::front () [override],[virtual]`

Returns the next activity

Implements [SST::ActivityQueue](#).

6.294.3.3 `void SST::UninitializedQueue::insert (Activity * activity) [override],[virtual]`

Insert a new activity into the queue

Implements [SST::ActivityQueue](#).

6.294.3.4 `Activity * SST::UninitializedQueue::pop () [override],[virtual]`

Remove and return the next activity

Implements [SST::ActivityQueue](#).

6.294.3.5 `int SST::UninitializedQueue::size () [override],[virtual]`

Returns the number of activities in the queue

Implements [SST::ActivityQueue](#).

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

- `sst/core/uninitializedQueue.h`
- `sst/core/uninitializedQueue.cc`

6.295 SST::Statistics::UniqueCountStatistic< T > Class Template Reference

```
#include <statuniquecount.h>
```

Inheritance diagram for `SST::Statistics::UniqueCountStatistic< T >`:

Collaboration diagram for `SST::Statistics::UniqueCountStatistic< T >`:

Public Member Functions

- **SST_ELI_DECLARE_STATISTIC_TEMPLATE** ([UniqueCountStatistic](#), "sst", "[UniqueCountStatistic](#)", SST_ELI_ELEMENT_VERSION(1, 0, 0), "Track unique occurrences of statistic", "SST::Statistic<T>") [UniqueCountStatistic](#)([BaseComponent](#) *comp

Public Attributes

- const std::string & **statName**
- const std::string const std::string & **statSubId**
- const std::string const std::string [Params](#) & **statParams**: [Statistic](#)<T>(comp
- const std::string const std::string [Params](#) **statName**
- const std::string const std::string [Params](#) **statSubId**
- const std::string const std::string [Params](#) **statParams**

Protected Member Functions

- void [addData_impl](#) (T data) override

Additional Inherited Members

6.295.1 Detailed Description

```
template<typename T>
class SST::Statistics::UniqueCountStatistic< T >
```

Creates a [Statistic](#) which counts unique values provided to it.

Template Parameters

<i>T</i>	A template for holding the main data type of this statistic
----------	---

6.295.2 Member Function Documentation

6.295.2.1 `template<typename T> void SST::Statistics::UniqueCountStatistic< T >::addData_impl (T data)`
`[inline], [override], [protected]`

Present a new value to the [Statistic](#) to be included in the unique set

Parameters

<i>data</i>	New data item to be included in the unique set
-------------	--

6.295.3 Member Data Documentation

6.295.3.1 `template<typename T> const std::string const std::string Params SST::Statistics::UniqueCountStatistic<T>::statParams`

Initial value:

```
{
    this->setStatisticTypeName("UniqueCount")
}
```

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

- `sst/core/statapi/statuniquecount.h`

6.296 SST::UnitAlgebra Class Reference

```
#include <unitAlgebra.h>
```

Inheritance diagram for SST::UnitAlgebra:

Collaboration diagram for SST::UnitAlgebra:

Public Member Functions

- [UnitAlgebra](#) (std::string val)
- void [print](#) (std::ostream &stream)
- void [printWithBestSI](#) (std::ostream &stream)
- std::string [toString](#) () const
- std::string [toStringBestSI](#) () const
- [UnitAlgebra](#) & **operator=** (const std::string &v)
- [UnitAlgebra](#) & **operator*=** (const [UnitAlgebra](#) &v)
- template<typename T>
[UnitAlgebra](#) & **operator*=** (const T &v)
- [UnitAlgebra](#) & **operator/=** (const [UnitAlgebra](#) &v)
- template<typename T>
[UnitAlgebra](#) & **operator/=** (const T &v)
- [UnitAlgebra](#) & **operator+=** (const [UnitAlgebra](#) &v)
- template<typename T>
[UnitAlgebra](#) & **operator+=** (const T &v)
- [UnitAlgebra](#) & **operator-=** (const [UnitAlgebra](#) &v)
- template<typename T>
[UnitAlgebra](#) & **operator-=** (const T &v)
- bool **operator>** (const [UnitAlgebra](#) &v) const
- bool **operator>=** (const [UnitAlgebra](#) &v) const
- bool **operator<** (const [UnitAlgebra](#) &v) const
- bool **operator<=** (const [UnitAlgebra](#) &v) const
- [UnitAlgebra](#) & **invert** ()
- bool [hasUnits](#) (std::string u) const
- [sst_big_num](#) [getValue](#) () const
- int64_t [getRoundedValue](#) () const
- void **serialize_order** (SST::Core::Serialization::serializer &ser) override

Additional Inherited Members

6.296.1 Detailed Description

Performs Unit math in full precision

Allows operations such as multiplying a frequency by 2.

6.296.2 Constructor & Destructor Documentation

6.296.2.1 UnitAlgebra::UnitAlgebra (std::string val)

Create a new [UnitAlgebra](#) instance, and pre-populate with a parsed value.

Parameters

<i>val</i>	Value to parse. It is of the following format: <pre>val := NUMBER()?UNITS NUMBER := (-)?[0-9]+([0-9]+)? UNITS := UNITGROUP(/UNITGROUP) UNITGROUP := UNIT(-UNIT)* UNIT := (SIPREFIX)?(BASEUNIT COMPUNIT) SIPREFIX := {a,f,p,n,u,m,[kKMGTPE]i?} BASEUNIT := {s,B,b,events} COMPUNIT := {Hz,hz,Bps,bps,event}</pre>
------------	--

6.296.3 Member Function Documentation

6.296.3.1 int64_t UnitAlgebra::getRoundedValue () const

Return the rounded value as a 64bit integer

6.296.3.2 sst_big_num SST::UnitAlgebra::getValue () const [inline]

Return the raw value

6.296.3.3 bool UnitAlgebra::hasUnits (std::string u) const

Returns true if the units in the parameter string are found in this object.

6.296.3.4 UnitAlgebra & UnitAlgebra::invert ()

Apply a reciprocal operation to the object

6.296.3.5 UnitAlgebra & UnitAlgebra::operator*=(const UnitAlgebra & v)

Multiply by an argument;

6.296.3.6 `template<typename T> UnitAlgebra& SST::UnitAlgebra::operator*=(const T & v)` `[inline]`

Multiply by an argument;

6.296.3.7 `UnitAlgebra & UnitAlgebra::operator+=(const UnitAlgebra & v)`

Add an argument;

6.296.3.8 `template<typename T> UnitAlgebra& SST::UnitAlgebra::operator+=(const T & v)` `[inline]`

Multiply by an argument;

6.296.3.9 `UnitAlgebra & UnitAlgebra::operator-=(const UnitAlgebra & v)`

Subtract an argument;

6.296.3.10 `template<typename T> UnitAlgebra& SST::UnitAlgebra::operator-=(const T & v)` `[inline]`

Divide by an argument;

6.296.3.11 `UnitAlgebra & UnitAlgebra::operator/=(const UnitAlgebra & v)`

Divide by an argument;

6.296.3.12 `template<typename T> UnitAlgebra& SST::UnitAlgebra::operator/=(const T & v)` `[inline]`

Divide by an argument;

6.296.3.13 `bool UnitAlgebra::operator< (const UnitAlgebra & v) const`

Compare if this object is less than the argument

6.296.3.14 `bool UnitAlgebra::operator<= (const UnitAlgebra & v) const`

Compare if this object is less than, or equal to, the argument

6.296.3.15 `bool UnitAlgebra::operator> (const UnitAlgebra & v) const`

Compare if this object is greater than the argument

6.296.3.16 `bool UnitAlgebra::operator>= (const UnitAlgebra & v) const`

Compare if this object is greater than, or equal to, the argument

6.296.3.17 `void UnitAlgebra::print (std::ostream & stream)`

Print to an ostream the value

6.296.3.18 `void UnitAlgebra::printWithBestSI (std::ostream & stream)`

Print to an ostream the value Formats the number using SI-prefixes

6.296.3.19 `string UnitAlgebra::toString () const`

Return a string representation of this value

6.296.3.20 `string UnitAlgebra::toStringBestSI () const`

Return a string representation of this value Formats the number using SI-prefixes

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

- sst/core/unitAlgebra.h
- sst/core/unitAlgebra.cc

6.297 SST::Units Class Reference

```
#include <unitAlgebra.h>
```

Public Member Functions

- [Units](#) (std::string units, [sst_big_num](#) &multiplier)
- [Units](#) & [operator=](#) (const [Units](#) &v)
- [Units](#) & [operator*=](#) (const [Units](#) &v)
- [Units](#) & [operator/=](#) (const [Units](#) &v)
- bool [operator==](#) (const [Units](#) &lhs) const
- bool [operator!=](#) (const [Units](#) &lhs) const
- [Units](#) & [invert](#) ()
- std::string [toString](#) () const

Static Public Member Functions

- static void [registerBaseUnit](#) (std::string u)
- static void [registerCompoundUnit](#) (std::string u, std::string v)

Friends

- class **UnitAlgebra**

6.297.1 Detailed Description

Helper class internal to [UnitAlgebra](#).

Contains information on valid units

6.297.2 Constructor & Destructor Documentation

6.297.2.1 Units::Units (std::string *units*, sst_big_num & *multiplier*)

Create a new instantiation of a [Units](#) with a base unit string, and multiplier

Parameters

<i>units</i>	String representing the new unit
<i>multiplier</i>	Value by which to multiply to get to this unit

6.297.3 Member Function Documentation

6.297.3.1 Units & Units::invert ()

Perform a reciprocal operation. Numerator and Denominator swap.

6.297.3.2 bool SST::Units::operator!= (const Units & *lhs*) const [inline]

Inequality Operator

6.297.3.3 Units & Units::operator*= (const Units & *v*)

Self-multiplication operator

6.297.3.4 Units & Units::operator/= (const Units & *v*)

Self-division operator

6.297.3.5 Units & Units::operator= (const Units & *v*)

Assignment operator

6.297.3.6 `bool Units::operator==(const Units & lhs) const`

Equality Operator

6.297.3.7 `void Units::registerBaseUnit(std::string u) [static]`

Create a new Base Unit type

6.297.3.8 `void Units::registerCompoundUnit(std::string u, std::string v) [static]`

Create a new Compound Unit type

6.297.3.9 `string Units::toString() const`

Return a String representation if this Unit

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

- sst/core/unitAlgebra.h
- sst/core/unitAlgebra.cc

6.298 SST::Core::XMLConfigGraphOutput Class Reference

Inheritance diagram for SST::Core::XMLConfigGraphOutput:

Collaboration diagram for SST::Core::XMLConfigGraphOutput:

Public Member Functions

- **XMLConfigGraphOutput** (const char *path)
- virtual void **generate** (const [Config](#) *cfg, [ConfigGraph](#) *graph) override

Protected Member Functions

- void **generateXML** (const std::string indent, const [ConfigComponent](#) &comp, const [ConfigLinkMap_t](#) &link↔ Map) const
- void **generateXML** (const std::string indent, const [ConfigLink](#) &link, const [ConfigComponentMap_t](#) &comp↔ Map) const

Additional Inherited Members

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

- sst/core/cfgoutput/xmlConfigOutput.h
- sst/core/cfgoutput/xmlConfigOutput.cc

6.299 SST::RNG::XORShiftRNG Class Reference

```
#include "sst/core/rng/xorshift.h"
```

Inheritance diagram for SST::RNG::XORShiftRNG:

Collaboration diagram for SST::RNG::XORShiftRNG:

Public Member Functions

- [XORShiftRNG](#) (unsigned int [seed](#))
- [XORShiftRNG](#) ()
- double [nextUniform](#) () override
- uint32_t [generateNextUInt32](#) () override
- uint64_t [generateNextUInt64](#) () override
- int64_t [generateNextInt64](#) () override
- int32_t [generateNextInt32](#) () override
- void [seed](#) (uint64_t newSeed)
- [~XORShiftRNG](#) ()

Protected Attributes

- uint32_t [x](#)
- uint32_t [y](#)
- uint32_t [z](#)
- uint32_t [w](#)

6.299.1 Detailed Description

Implements a lightweight RNG based on XOR-shift operations. We utilize the XORSHIFT algorithm from: <http://en.wikipedia.org/wiki/Xorshift>. This is a very lightweight and inexpensive RNG.

6.299.2 Constructor & Destructor Documentation

6.299.2.1 XORShiftRNG::XORShiftRNG (unsigned int *seed*)

Create a new Mersenne RNG with a specified seed

Parameters

in	<i>seed</i>	The seed for this RNG
----	-------------	-----------------------

6.299.2.2 XORShiftRNG::XORShiftRNG ()

Creates a new Mersenne using a random seed which is obtained from the system clock. Note this will give different results on different platforms and between runs.

6.299.2.3 XORShiftRNG::~~XORShiftRNG ()

Destructor for Mersenne

6.299.3 Member Function Documentation

6.299.3.1 int32_t XORShiftRNG::generateNextInt32 () [override],[virtual]

Generates the next random number as a signed 32-bit integer

Implements [SST::RNG::SSTRandom](#).

6.299.3.2 int64_t XORShiftRNG::generateNextInt64 () [override],[virtual]

Generates the next random number as a signed 64-bit integer

Implements [SST::RNG::SSTRandom](#).

6.299.3.3 uint32_t XORShiftRNG::generateNextUInt32 () [override],[virtual]

Generates the next random number as an unsigned 32-bit integer

Implements [SST::RNG::SSTRandom](#).

6.299.3.4 uint64_t XORShiftRNG::generateNextUInt64 () [override],[virtual]

Generates the next random number as an unsigned 64-bit integer

Implements [SST::RNG::SSTRandom](#).

6.299.3.5 double XORShiftRNG::nextUniform () [override],[virtual]

Generates the next random number as a double value between 0 and 1.

Implements [SST::RNG::SSTRandom](#).

6.299.3.6 void XORShiftRNG::seed (uint64_t newSeed)

Seed the XOR RNG

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

- sst/core/rng/xorshift.h
- sst/core/rng/xorshift.cc

Index

- ~IPCTunnel
 - SST::Core::Interprocess::IPCTunnel, [116](#)
- ~MersenneRNG
 - SST::RNG::MersenneRNG, [136](#)
- ~SSTConstantDistribution
 - SST::RNG::SSTConstantDistribution, [224](#)
- ~SSTDDiscreteDistribution
 - SST::RNG::SSTDDiscreteDistribution, [226](#)
- ~SSTExponentialDistribution
 - SST::RNG::SSTExponentialDistribution, [231](#)
- ~SSTGaussianDistribution
 - SST::RNG::SSTGaussianDistribution, [233](#)
- ~SSTPoissonDistribution
 - SST::RNG::SSTPoissonDistribution, [243](#)
- ~SSTRandom
 - SST::RNG::SSTRandom, [244](#)
- ~SSTRandomDistribution
 - SST::RNG::SSTRandomDistribution, [245](#)
- ~SSTUniformDistribution
 - SST::RNG::SSTUniformDistribution, [250](#)
- ~XORShiftRNG
 - SST::RNG::XORShiftRNG, [346](#)
- Accept
 - TiXmlComment, [311](#)
 - TiXmlDeclaration, [313](#)
 - TiXmlDocument, [314](#)
 - TiXmlElement, [318](#)
 - TiXmlNode, [326](#)
 - TiXmlText, [333](#)
 - TiXmlUnknown, [334](#)
- acceptsGroups
 - SST::Statistics::StatisticOutput, [266](#)
 - SST::Statistics::StatisticOutputHDF5, [275](#)
- add_builder
 - SST::Core::Serialization::serializable_factory, [190](#)
- addComponent
 - SST::ConfigGraph, [61](#)
- addData
 - SST::Statistics::Statistic, [253](#)
- addData_impl
 - SST::Statistics::AccumulatorStatistic, [22](#)
 - SST::Statistics::HistogramStatistic, [110](#)
 - SST::Statistics::UniqueCountStatistic, [339](#)
- addLink
 - SST::ConfigGraph, [61](#)
- addLoader
 - SST::ELI::LoadedLibraries, [128](#)
- addRecvLatency
 - SST::Link, [122](#)
- addSelfPort
 - SST::LinkMap, [126](#)
- addStatisticOutputParameter
 - SST::ConfigGraph, [61](#)
- addStatisticParameterForComponentName
 - SST::ConfigGraph, [61](#)
- addSubModule
 - SST::SSTElementPythonModuleCode, [229](#)
- Addr
 - SST::Interfaces::SimpleMem, [205](#)
- addr
 - SST::Interfaces::SimpleMem::Request, [180](#)
- addrs
 - SST::Interfaces::SimpleMem::Request, [180](#)
- advise_handle
 - SST::LoaderData, [129](#)
- afterInitQueue
 - SST::Link, [124](#)
- afterRunQueue
 - SST::Link, [124](#)
- allow_adaptive
 - SST::Interfaces::SimpleNetwork::Request, [183](#)
- Attribute
 - TiXmlElement, [318](#)
- BOTH
 - SST::Simulation, [215](#)
- baseDistrib
 - SST::RNG::SSTDDiscreteDistribution, [226](#)
 - SST::RNG::SSTExponentialDistribution, [231](#)
 - SST::RNG::SSTGaussianDistribution, [234](#)
 - SST::RNG::SSTPoissonDistribution, [243](#)
 - SST::RNG::SSTUniformDistribution, [251](#)
- checkForStructuralErrors
 - SST::ConfigGraph, [61](#)
- checkOutputParameters
 - SST::Statistics::StatisticOutput, [266](#)
 - SST::Statistics::StatisticOutputCSV, [272](#)
 - SST::Statistics::StatisticOutputConsole, [270](#)
 - SST::Statistics::StatisticOutputHDF5, [275](#)
 - SST::Statistics::StatisticOutputJSON, [277](#)
 - SST::Statistics::StatisticOutputTxt, [280](#)
- checkRanks
 - SST::ConfigGraph, [61](#)
- Child
 - TiXmlHandle, [321](#)
- ChildElement
 - TiXmlHandle, [322](#)
- clear

- SST::Params, 157
- SST::SyncQueue, 293
- clearBuffer
 - SST::Core::Interprocess::IPCTunnel, 116
- ClearError
 - TiXmlDocument, 314
- clearStatisticData
 - SST::Statistics::AccumulatorStatistic, 22
 - SST::Statistics::StatisticBase, 255
- Clock
 - SST::Clock, 46
- clockMap_t
 - SST::Simulation, 215
- Clone
 - TiXmlDocument, 314
 - TiXmlNode, 326
- clone
 - SST::Event, 86
- cls_id_
 - SST::Core::Serialization::serializable_builder_impl, 189
- cmd
 - SST::Interfaces::SimpleMem::Request, 180
- Command
 - SST::Interfaces::SimpleMem::Request, 177
- complete
 - SST::BaseComponent, 32
 - SST::Interfaces::SimpleNetwork, 208
 - SST::Simulation, 215
- component
 - SST::ConfigLink, 65
- ComponentHolder, 49
- ComponentPy_t, 51
- Config
 - SST::Config, 53
- ConfigComponent
 - SST::ConfigComponent, 58
- configFile
 - SST::Config, 54
- configureLink
 - SST::BaseComponent, 32, 33
- configureSelfLink
 - SST::BaseComponent, 33, 34
- configuredQueue
 - SST::Link, 124
- const
 - SST::SubComponentSlotInfo, 288
- contains
 - SST::Params, 157
- containsComponentInRank
 - SST::ConfigGraph, 61
- convert_to
 - SST::decimal_fixedpoint, 71
- convertFromCoreTime
 - SST::TimeConverter, 299
- convertToCoreTime
 - SST::TimeConverter, 299
- count
 - SST::Interfaces::TestEvent, 294
 - SST::Params, 157
- Create
 - SST::Factory, 90
- createAll
 - SST::SubComponentSlotInfo, 287
- createAllSparse
 - SST::SubComponentSlotInfo, 288
- CreateComponent
 - SST::Factory, 90
- createConfigGraph
 - SST::SSTModelDescription, 240
- CreateModule
 - SST::Factory, 90
- CreateModuleWithComponent
 - SST::Factory, 90
- CreatePartitioner
 - SST::Factory, 91
- createPrimaryModule
 - SST::SSTElementPythonModule, 228
- createSimulation
 - SST::Simulation, 215
- CreateStatistic
 - SST::Factory, 91
- CreateSubComponent
 - SST::Factory, 91
- current_ref
 - SST::ConfigLink, 65
- custOpc
 - SST::Interfaces::SimpleMem::Request, 180
- CustomCmd
 - SST::Interfaces::SimpleMem::Request, 177
- data
 - SST::Interfaces::SimpleMem::Request, 180
- dataVec
 - SST::Interfaces::SimpleMem::Request, 177
- debug
 - SST::Output, 148
- debugEnabled
 - SST::SSTInfoConfig, 235
- debugFile
 - SST::Config, 54
- debugPrefix
 - SST::Output, 148
- decimal_fixedpoint
 - SST::decimal_fixedpoint, 69, 70
- defaultTimeBase
 - SST::Link, 125
- defaultValue
 - SST::ElementInfoParam, 77
- delayCollection
 - SST::Statistics::StatisticBase, 255
- delayOutput
 - SST::Statistics::StatisticBase, 255
- deleteDistrib
 - SST::RNG::SSTDiscreteDistribution, 226
 - SST::RNG::SSTExponentialDistribution, 231
 - SST::RNG::SSTGaussianDistribution, 234

- SST::RNG::SSTPoissonDistribution, 243
- SST::RNG::SSTUniformDistribution, 251
- deliverEvent
 - SST::Link, 122
- delivery_link
 - SST::Event, 87
- description
 - SST::ElementInfoParam, 77
 - SST::ElementInfoPort, 77
 - SST::ElementInfoPort2, 78
 - SST::ElementInfoStatistic, 79
- dest
 - SST::Interfaces::SimpleNetwork::Request, 183
- disable
 - SST::Statistics::StatisticBase, 256
- DoesComponentInfoStatisticNameExist
 - SST::Factory, 92
- DoesSubComponentSlotExist
 - SST::Factory, 92
- dump_component_graph_file
 - SST::Config, 54
- ElemLoader
 - SST::ElemLoader, 81
- Element
 - TiXmlHandle, 322
- emergencyShutdown
 - SST::BaseComponent, 34
- empty
 - SST::ActivityQueue, 28
 - SST::IMPL::TimeVortexPQ, 303
 - SST::InitQueue, 114
 - SST::LinkMap, 126
 - SST::Params, 157
 - SST::PollingLinkQueue, 163
 - SST::SyncQueue, 293
 - SST::ThreadSyncQueue, 297
 - SST::TimeVortex, 302
 - SST::UninitializedQueue, 338
- enable
 - SST::Statistics::StatisticBase, 256
- enable_sig_handling
 - SST::Config, 55
- enableLevel
 - SST::ElementInfoStatistic, 79
- enableStatisticForComponentName
 - SST::ConfigGraph, 62
- enableVerify
 - SST::Params, 157
- enabledStatistics
 - SST::ConfigComponent, 58
- EncodeString
 - TiXmlBase, 308
- endOfSimulation
 - SST::Statistics::StatisticOutput, 266
 - SST::Statistics::StatisticOutputCSV, 272
 - SST::Statistics::StatisticOutputConsole, 270
 - SST::Statistics::StatisticOutputHDF5, 275
 - SST::Statistics::StatisticOutputJSON, 277
- SST::Statistics::StatisticOutputTxt, 280
- endSimulation
 - SST::Action, 25
- Error
 - TiXmlDocument, 315
- ErrorId
 - TiXmlDocument, 315
- ErrorRow
 - TiXmlDocument, 315
- errorString
 - TiXmlBase, 309
- exchangeLinkUntimedData
 - SST::RankSyncParallelSkip, 172
 - SST::RankSyncSerialSkip, 173
 - SST::SyncBase, 290
 - SST::SyncManager, 291
- execute
 - SST::Activity, 26
 - SST::Event, 86
 - SST::Exit, 88
 - SST::NullEvent, 141
 - SST::StopAction, 284
 - SST::SyncManager, 291
 - SST::ThreadSync, 295
- Exit
 - SST::Exit, 87
- F_FLUSH_SUCCESS
 - SST::Interfaces::SimpleMem::Request, 177
- F_LOCKED
 - SST::Interfaces::SimpleMem::Request, 177
- F_NONCACHEABLE
 - SST::Interfaces::SimpleMem::Request, 177
- FILE
 - SST::Output, 147
- FULL
 - SST::Interfaces::SimpleNetwork::Request, 182
- fatal
 - SST::Output, 149
- file
 - SST::Output, 153
- finalizeLinkConfigurations
 - SST::RankSyncParallelSkip, 172
 - SST::RankSyncSerialSkip, 173
 - SST::SyncBase, 290
 - SST::SyncManager, 291
 - SST::ThreadSync, 295
 - SST::ThreadSyncSimpleSkip, 298
- find
 - SST::Params, 157–159
- find_array
 - SST::Params, 160
- find_prefix_params
 - SST::Params, 160
- finish
 - SST::BaseComponent, 34
 - SST::Interfaces::SimpleNetwork, 208
 - SST::SubComponent, 286
- FirstChild

- TiXmlNode, 326
- Flags
 - SST::Interfaces::SimpleMem::Request, 177
- flags
 - SST::Interfaces::SimpleMem::Request, 180
- flags_t
 - SST::Interfaces::SimpleMem::Request, 177
- flush
 - SST::Output, 149
- FlushLine
 - SST::Interfaces::SimpleMem::Request, 177
- FlushLineInv
 - SST::Interfaces::SimpleMem::Request, 177
- FlushLineResp
 - SST::Interfaces::SimpleMem::Request, 177
- free
 - SST::Core::MemPool, 134
- front
 - SST::ActivityQueue, 28
 - SST::IMPL::TimeVortexPQ, 303
 - SST::InitQueue, 114
 - SST::PollingLinkQueue, 163
 - SST::SyncQueue, 293
 - SST::ThreadSyncQueue, 297
 - SST::TimeVortex, 302
 - SST::UninitializedQueue, 338
- generateNextInt32
 - SST::RNG::MarsagliaRNG, 132
 - SST::RNG::MersenneRNG, 136
 - SST::RNG::SSTRandom, 244
 - SST::RNG::XORShiftRNG, 347
- generateNextInt64
 - SST::RNG::MarsagliaRNG, 132
 - SST::RNG::MersenneRNG, 136
 - SST::RNG::SSTRandom, 244
 - SST::RNG::XORShiftRNG, 347
- generateNextUInt32
 - SST::RNG::MarsagliaRNG, 133
 - SST::RNG::MersenneRNG, 136
 - SST::RNG::SSTRandom, 244
 - SST::RNG::XORShiftRNG, 347
- generateNextUInt64
 - SST::RNG::MarsagliaRNG, 133
 - SST::RNG::MersenneRNG, 136
 - SST::RNG::SSTRandom, 244
 - SST::RNG::XORShiftRNG, 347
- generateUniqueld
 - SST::Event, 86
- getArithmeticMean
 - SST::Statistics::AccumulatorStatistic, 22
- getBytesMemUsed
 - SST::Core::MemPool, 134
- getCollectionCount
 - SST::Statistics::StatisticBase, 256
- getCollectionCountLimit
 - SST::Statistics::StatisticBase, 256
- getCompName
 - SST::Statistics::StatisticBase, 256
- getComponent
 - SST::Simulation, 216
 - SST::Statistics::StatisticBase, 256
- getComponentInfoMap
 - SST::Simulation, 216
- GetComponentInfoStatisticEnableLevel
 - SST::Factory, 92
- GetComponentInfoStatisticUnits
 - SST::Factory, 93
- getComponentLinkMap
 - SST::Simulation, 216
- getComponentMap
 - SST::ConfigGraph, 62
- getCoordinates
 - SST::BaseComponent, 34
- getCount
 - SST::Statistics::AccumulatorStatistic, 22
- getCurrentPriority
 - SST::Simulation, 216
- getCurrentSimCycle
 - SST::Simulation, 216
- getCurrentSimTime
 - SST::BaseComponent, 34, 35
- getCurrentSimTimeMicro
 - SST::BaseComponent, 35
- getCurrentSimTimeMilli
 - SST::BaseComponent, 35
- getCurrentSimTimeNano
 - SST::BaseComponent, 35
- getCustomOpc
 - SST::Interfaces::SimpleMem::Request, 178
- getData
 - SST::SyncQueue, 293
- getDefaultTimeBase
 - SST::Link, 122
- getDeliveryLink
 - SST::Event, 86
- getDeliveryTime
 - SST::Activity, 26
- GetDocument
 - TiXmlNode, 326
- getElapsedSimTime
 - SST::Simulation, 216
- getElementsToProcessArray
 - SST::SSTInfoConfig, 235
- getEndSimCycle
 - SST::Simulation, 216
- getEndpointID
 - SST::Interfaces::SimpleNetwork, 208
- getExit
 - SST::Simulation, 216
- getFactor
 - SST::TimeConverter, 299
- getFieldHandle
 - SST::Statistics::StatisticFieldInfo, 261
- getFieldInfoArray
 - SST::Statistics::StatisticOutput, 266
- getFieldName

- SST::Statistics::StatisticFieldInfo, 261
- getFieldType
 - SST::Statistics::StatisticFieldInfo, 261
- getFieldTypeShortName
 - SST::Statistics::StatisticOutput, 266
- getFieldUniqueName
 - SST::Statistics::StatisticFieldInfo, 261
- getFilterMap
 - SST::SSTInfoConfig, 235
- getFinalSimTime
 - SST::Simulation, 216
- getFlagClearDataOnOutput
 - SST::Statistics::StatisticBase, 256
- getFlagOutputAtEndOfSim
 - SST::Statistics::StatisticBase, 256
- getFlagResetCountOnOutput
 - SST::Statistics::StatisticBase, 256
- getFlags
 - SST::Interfaces::SimpleMem::Request, 178
- getFractionWords
 - SST::decimal_fixedpoint, 71
- getFullModuleName
 - SST::SSTElementPythonModuleCode, 229
- getFullStatName
 - SST::Statistics::StatisticBase, 257
- getId
 - SST::BaseComponent, 35
 - SST::Link, 122
 - SST::LinkPair, 127
- getInstructionPointer
 - SST::Interfaces::SimpleMem::Request, 178
- getLambda
 - SST::RNG::SSTExponentialDistribution, 231
 - SST::RNG::SSTPoissonDistribution, 243
- getLeft
 - SST::LinkPair, 127
- getLibPath
 - SST::Config, 53
- getLibraryName
 - SST::SSTLibraryInfo, 236
- getLink
 - SST::Interfaces::SimpleMem, 205
 - SST::LinkMap, 126
- getLinkBW
 - SST::Interfaces::SimpleNetwork, 208
- getLinkId
 - SST::Event, 86
- getLinkMap
 - SST::ConfigGraph, 62
 - SST::LinkMap, 126
- getLocalMinimumNextActivityTime
 - SST::Simulation, 216
- getLocalShareID
 - SST::SharedRegion, 200
- getLocalSharedRegion
 - SST::BaseComponent, 35
- getMax
 - SST::Statistics::AccumulatorStatistic, 23
- getMean
 - SST::RNG::SSTConstantDistribution, 224
 - SST::RNG::SSTGaussianDistribution, 233
- getMemFlags
 - SST::Interfaces::SimpleMem::Request, 178
- getMergeInfo
 - SST::RegionInfo, 175
- getMicro
 - SST::TimeLord, 300
- getMilli
 - SST::TimeLord, 300
- getMin
 - SST::Statistics::AccumulatorStatistic, 23
- getMinLatency
 - SST::ConfigLink, 64
 - SST::PartitionLink, 163
- getName
 - SST::BaseComponent, 35
- getNano
 - SST::TimeLord, 300
- getNextActivityTime
 - SST::Simulation, 217
- getNextClockCycle
 - SST::BaseComponent, 35
 - SST::Simulation, 217
- getNextCycle
 - SST::Clock, 46
- getNextDouble
 - SST::RNG::SSTConstantDistribution, 224
 - SST::RNG::SSTDiscreteDistribution, 226
 - SST::RNG::SSTExponentialDistribution, 231
 - SST::RNG::SSTGaussianDistribution, 233
 - SST::RNG::SSTPoissonDistribution, 243
 - SST::RNG::SSTRandomDistribution, 246
 - SST::RNG::SSTUniformDistribution, 250
- getNumRanks
 - SST::Simulation, 217
- getOptionBits
 - SST::SSTInfoConfig, 235
- getOutputDirectory
 - SST::Simulation, 217
- getOutputLocation
 - SST::Output, 149
- getOutputParameters
 - SST::Statistics::StatisticOutput, 267
- getParamName
 - SST::Params, 160
- getParamNames
 - SST::Factory, 93
- getParams
 - SST::Statistics::StatisticBase, 257
- getParent
 - SST::BaseComponent, 35
- getPotentialElements
 - SST::ElemLoader, 81
- getPrefix
 - SST::Output, 149
- getPriority

- SST::Activity, [27](#)
- getPtr
 - SST::SharedRegion, [200](#)
- getPythonModule
 - SST::Factory, [93](#)
- getRank
 - SST::Simulation, [217](#)
- getRawPtr
 - SST::SharedRegion, [200](#)
- getRegisteredCollectionMode
 - SST::Statistics::StatisticBase, [257](#)
- getRegisteredField
 - SST::Statistics::StatisticOutput, [267](#)
- getRight
 - SST::LinkPair, [127](#)
- getRoundedValue
 - SST::UnitAlgebra, [341](#)
- getSharedData
 - SST::Core::Interprocess::IPCTunnel, [116](#)
- getSharedRegionManager
 - SST::Simulation, [217](#)
- getSimCycles
 - SST::TimeLord, [300](#)
- getSimulation
 - SST::Simulation, [217](#)
- getSimulationMode
 - SST::Simulation, [217](#)
- getSimulationOutput
 - SST::Simulation, [217](#)
- getSize
 - SST::SharedRegion, [200](#)
- getStandardDev
 - SST::RNG::SSTGaussianDistribution, [233](#)
- getStandardDeviation
 - SST::Statistics::AccumulatorStatistic, [23](#)
- getStatDataType
 - SST::Statistics::StatisticBase, [257](#)
- getStatDataTypeFullName
 - SST::Statistics::StatisticBase, [257](#)
- getStatDataTypeShortName
 - SST::Statistics::StatisticBase, [257](#)
- getStatName
 - SST::Statistics::StatisticBase, [257](#)
 - SST::Statistics::StatisticFieldInfo, [261](#)
- getStatSubId
 - SST::Statistics::StatisticBase, [257](#)
- getStatTypeName
 - SST::Statistics::StatisticBase, [257](#)
- getStatisticOutputName
 - SST::Statistics::StatisticOutput, [267](#)
- getStatisticsProcessingEngine
 - SST::Simulation, [218](#)
- getString
 - SST::Interfaces::StringEvent, [285](#)
- getSum
 - SST::Statistics::AccumulatorStatistic, [23](#)
- getSumSquared
 - SST::Statistics::AccumulatorStatistic, [23](#)
- GetText
 - TiXmlElement, [318](#)
- getTimeBase
 - SST::TimeLord, [300](#)
- getTimeConverter
 - SST::TimeLord, [300](#), [301](#)
- getTimeLord
 - SST::Simulation, [218](#)
- getValue
 - SST::UnitAlgebra, [341](#)
- getVariance
 - SST::Statistics::AccumulatorStatistic, [24](#)
- getVerboseLevel
 - SST::Config, [53](#)
 - SST::Output, [149](#)
- getVerboseMask
 - SST::Output, [149](#)
- getVirtualAddress
 - SST::Interfaces::SimpleMem::Request, [179](#)
- getWholeWords
 - SST::decimal_fixedpoint, [71](#)
- getXMLFilePath
 - SST::SSTInfoConfig, [235](#)
- givePayload
 - SST::Interfaces::SimpleNetwork::Request, [182](#)
- Handler
 - SST::Clock::Handler, [95](#)
 - SST::Clock::Handler< classT, void >, [101](#)
 - SST::Event::Handler, [98](#)
 - SST::Event::Handler< classT, void >, [103](#)
 - SST::Interfaces::SimpleMem::Handler, [99](#)
 - SST::Interfaces::SimpleMem::Handler< classT, void >, [104](#)
 - SST::Interfaces::SimpleNetwork::Handler, [100](#)
 - SST::Interfaces::SimpleNetwork::Handler< classT, void >, [105](#)
 - SST::OneShot::Handler, [97](#)
 - SST::OneShot::Handler< classT, void >, [102](#)
- hasLibrary
 - SST::Factory, [93](#)
- hasUnits
 - SST::UnitAlgebra, [341](#)
- head
 - SST::Interfaces::SimpleNetwork::Request, [183](#)
- heartbeatPeriod
 - SST::Config, [55](#)
- INIT
 - SST::Simulation, [215](#)
- IPCTunnel
 - SST::Core::Interprocess::IPCTunnel, [116](#)
- id
 - SST::ConfigComponent, [58](#)
 - SST::ConfigLink, [65](#)
 - SST::Interfaces::SimpleMem::Request, [180](#)
- id_t
 - SST::Interfaces::SimpleMem::Request, [177](#)
- id_type

- SST::Event, 86
- implStartOutputEntries
 - SST::Statistics::StatisticOutput, 267
 - SST::Statistics::StatisticOutputCSV, 272
 - SST::Statistics::StatisticOutputConsole, 270
 - SST::Statistics::StatisticOutputHDF5, 275
 - SST::Statistics::StatisticOutputJSON, 278
 - SST::Statistics::StatisticOutputTxt, 280
- implStopOutputEntries
 - SST::Statistics::StatisticOutput, 268
 - SST::Statistics::StatisticOutputCSV, 273
 - SST::Statistics::StatisticOutputConsole, 270
 - SST::Statistics::StatisticOutputHDF5, 275
 - SST::Statistics::StatisticOutputJSON, 278
 - SST::Statistics::StatisticOutputTxt, 280
- inRange
 - SST::RankInfo, 171
- incrementCollectionCount
 - SST::Statistics::StatisticBase, 257
- init
 - SST::BaseComponent, 36
 - SST::Interfaces::SimpleNetwork, 209
 - SST::Output, 150
 - SST::SubComponent, 286
- initialize
 - SST::Interfaces::SimpleMem, 205
 - SST::Interfaces::SimpleNetwork, 209
 - SST::Interfaces::SimpleNetwork::NetworkInspector, 138
 - SST::Simulation, 218
- insert
 - SST::ActivityQueue, 28
 - SST::IMPL::TimeVortexPQ, 303
 - SST::InitQueue, 114
 - SST::Params, 160
 - SST::PollingLinkQueue, 164
 - SST::SyncQueue, 293
 - SST::ThreadSyncQueue, 297
 - SST::TimeVortex, 302
 - SST::UninitializedQueue, 338
- insertActivity
 - SST::Simulation, 218
- InsertAfterChild
 - TiXmlNode, 326
- InsertBeforeChild
 - TiXmlNode, 327
- InsertEndChild
 - TiXmlNode, 327
- insertLink
 - SST::LinkMap, 126
- insertNulls
 - SST::SubComponentSlotInfo, 288
- inspectPayload
 - SST::Interfaces::SimpleNetwork::Request, 183
- instrPtr
 - SST::Interfaces::SimpleMem::Request, 180
- inverse
 - SST::decimal_fixedpoint, 71
- invert
 - SST::UnitAlgebra, 341
 - SST::Units, 344
- isEnabled
 - SST::Statistics::StatisticBase, 258
- isNetworkInitialized
 - SST::Interfaces::SimpleNetwork, 209
- isNullStatistic
 - SST::Statistics::StatisticBase, 258
- isOutputEnabled
 - SST::Statistics::StatisticBase, 258
- isPortConnected
 - SST::BaseComponent, 36
- isPortNameValid
 - SST::Factory, 93
- isReady
 - SST::SharedRegion, 200
 - SST::Statistics::StatisticBase, 258
- isScheduled
 - SST::OneShot, 145
- isSubComponentLoadableUsingAPI
 - SST::Factory, 94
- isWireUpFinished
 - SST::Simulation, 218
- IterateChildren
 - TiXmlNode, 327
- key_type
 - SST::Params, 156
- KeySet_t
 - SST::Params, 156
- lambda
 - SST::RNG::SSTExponentialDistribution, 231
 - SST::RNG::SSTPoissonDistribution, 243
- latency
 - SST::ConfigLink, 65
 - SST::Link, 125
- latency_str
 - SST::ConfigLink, 65
- line
 - SST::Output, 154
- Link
 - SST::Link, 122
- LinkEndChild
 - TiXmlNode, 327
- LinkPair
 - SST::LinkPair, 127
- LinkPy_t, 128
- links
 - SST::ConfigComponent, 59
- LoadFile
 - TiXmlDocument, 315
- loadLibrary
 - SST::ElemLoader, 81
- loadModule
 - SST::BaseComponent, 36
- loadModuleWithComponent
 - SST::BaseComponent, 36

- loadUserSubComponentByIndex< T, ARGS... >
 - SST::BaseComponent, 39
- lt__advise, 129
- lt__handle, 129
- lt__interface_id, 130
- lt_dlinfo, 130
- lt_dlsymlist, 131
- lt_dlvtable, 131
- lt_interface_data, 131
- malloc
 - SST::Core::MemPool, 134
- MarsagliaRNG
 - SST::RNG::MarsagliaRNG, 132
- mean
 - SST::RNG::SSTConstantDistribution, 225
 - SST::RNG::SSTGaussianDistribution, 234
- memFlags
 - SST::Interfaces::SimpleMem::Request, 181
- MemPool
 - SST::Core::MemPool, 134
- merge
 - SST::SharedRegionInitializedMerger, 202
 - SST::SharedRegionMerger, 204
- MersenneRNG
 - SST::RNG::MersenneRNG, 135, 136
- Mode_t
 - SST::Simulation, 215
- model_options
 - SST::Config, 55
- modifyRegion
 - SST::SharedRegion, 201
- ModuleLoaderPy_t, 137
- NO_ID
 - SST::Event, 87
- NONE
 - SST::Interfaces::SimpleNetwork::Request, 182
 - SST::Output, 147
- name
 - SST::ConfigComponent, 59
 - SST::ConfigLink, 65
 - SST::ElementInfoParam, 77
 - SST::ElementInfoPort, 77
 - SST::ElementInfoPort2, 78
 - SST::ElementInfoStatistic, 79
- negate
 - SST::decimal_fixedpoint, 71
- NextSiblingElement
 - TiXmlNode, 327
- nextSubID
 - SST::ConfigComponent, 59
- nextUniform
 - SST::RNG::MarsagliaRNG, 133
 - SST::RNG::MersenneRNG, 136
 - SST::RNG::SSTRandom, 245
 - SST::RNG::XORShiftRNG, 347
- nid_t
 - SST::Interfaces::SimpleNetwork, 208
- no_cut
 - SST::ConfigLink, 65
- no_env_config
 - SST::Config, 55
- Node
 - TiXmlHandle, 322
- NodeType
 - TiXmlNode, 326
- numAlloc
 - SST::Core::MemPool, 135
- numFree
 - SST::Core::MemPool, 135
- OneShot
 - SST::OneShot, 144
- oneShotMap_t
 - SST::Simulation, 215
- operator!=
 - SST::Units, 344
 - SST::decimal_fixedpoint, 71
- operator<
 - SST::UnitAlgebra, 342
 - SST::decimal_fixedpoint, 72
- operator<=
 - SST::UnitAlgebra, 342
 - SST::decimal_fixedpoint, 73
- operator>
 - SST::UnitAlgebra, 342
 - SST::decimal_fixedpoint, 74
- operator>=
 - SST::UnitAlgebra, 342
 - SST::decimal_fixedpoint, 74
- operator*=
 - SST::UnitAlgebra, 341
 - SST::Units, 344
 - SST::decimal_fixedpoint, 72
- operator()
 - SST::Activity::less_time, 119
 - SST::Activity::less_time_priority, 119
 - SST::Activity::less_time_priority_order, 120
 - SST::Activity::pq_less_time_priority, 165
 - SST::Activity::pq_less_time_priority_order, 165
 - SST::Clock::Handler, 96
 - SST::Clock::Handler< classT, void >, 101
 - SST::Clock::HandlerBase, 105
 - SST::Event::Handler, 98
 - SST::Event::Handler< classT, void >, 103
 - SST::Event::HandlerBase, 107
 - SST::Interfaces::SimpleMem::Handler, 99
 - SST::Interfaces::SimpleMem::Handler< classT, void >, 104
 - SST::Interfaces::SimpleMem::HandlerBase, 107
 - SST::OneShot::Handler, 97
 - SST::OneShot::Handler< classT, void >, 102
 - SST::OneShot::HandlerBase, 106
 - SST::char_delimiter, 45
 - SST::escaped_list_separator, 84
- operator+=
 - SST::UnitAlgebra, 342

- SST::decimal_fixedpoint, 72
- operator==
 - SST::UnitAlgebra, 342
 - SST::decimal_fixedpoint, 72
- operator/=
 - SST::UnitAlgebra, 342
 - SST::Units, 344
 - SST::decimal_fixedpoint, 72
- operator=
 - SST::Params, 160
 - SST::Units, 344
 - SST::decimal_fixedpoint, 73
- operator==
 - SST::Statistics::StatisticFieldInfo, 261
 - SST::Units, 344
 - SST::decimal_fixedpoint, 73
- Output
 - SST::Output, 147, 148
- output
 - SST::Output, 150, 151
 - SST::TraceFunction, 336
- output_config_graph
 - SST::Config, 55
- output_core_prefix
 - SST::Config, 55
- output_directory
 - SST::Config, 55
- output_dot
 - SST::Config, 55
- output_json
 - SST::Config, 55
- output_location_t
 - SST::Output, 147
- output_xml
 - SST::Config, 55
- outputField
 - SST::Statistics::StatisticOutput, 268
 - SST::Statistics::StatisticOutputCSV, 273
 - SST::Statistics::StatisticOutputConsole, 270
 - SST::Statistics::StatisticOutputHDF5, 276
 - SST::Statistics::StatisticOutputJSON, 278
 - SST::Statistics::StatisticOutputTxt, 280
- outputHumanReadable
 - SST::SSTLibraryInfo, 237
- outputXML
 - SST::SSTLibraryInfo, 237
- OverallOutputter, 154
- pack_buffer
 - SST::Core::Serialization::pvt::ser_packer, 186
- pair_link
 - SST::Link, 125
- Params
 - SST::Params, 156
- params
 - SST::ConfigComponent, 59
- Parse
 - TiXmlDocument, 315
- parseCmdLine
 - SST::Config, 54
 - SST::SSTInfoConfig, 235
- partOutput
 - SST::IMPL::Partition::SSTLinearPartition, 239
- partitioner
 - SST::Config, 56
- performGlobalStatisticOutput
 - SST::Statistics::StatisticProcessingEngine, 282
- performPartition
 - SST::IMPL::Partition::SSTLinearPartition, 238
 - SST::IMPL::Partition::SSTRoundRobinPartition, 246
 - SST::IMPL::Partition::SSTSelfPartition, 247, 248
 - SST::IMPL::Partition::SSTSinglePartition, 249
 - SST::IMPL::Partition::SimplePartitioner, 212
 - SST::Partition::SSTPartitioner, 241
- performStatisticOutput
 - SST::Statistics::StatisticProcessingEngine, 282
- performWireUp
 - SST::Simulation, 218
- pop
 - SST::ActivityQueue, 28
 - SST::IMPL::TimeVortexPQ, 303
 - SST::InitQueue, 114
 - SST::PollingLinkQueue, 164
 - SST::SyncQueue, 293
 - SST::ThreadSyncQueue, 297
 - SST::TimeVortex, 302
 - SST::UninitializedQueue, 338
- popAllowedKeys
 - SST::Params, 161
- port
 - SST::ConfigLink, 65
- postCreationCleanup
 - SST::ConfigGraph, 62
- prepareForComplete
 - SST::RankSyncParallelSkip, 172
 - SST::RankSyncSerialSkip, 173
 - SST::SyncManager, 292
- primaryComponentDoNotEndSim
 - SST::Component, 47
- primaryComponentOKToEndSim
 - SST::Component, 47
- Print
 - SST::Config, 54
 - TiXmlAttribute, 305
 - TiXmlBase, 308
 - TiXmlComment, 311
 - TiXmlDeclaration, 313
 - TiXmlDocument, 315
 - TiXmlElement, 318
 - TiXmlText, 333
 - TiXmlUnknown, 334
- print
 - SST::Action, 25
 - SST::Activity, 27
 - SST::Clock, 46
 - SST::Config, 54

- SST::ConfigComponent, 58
- SST::ConfigGraph, 62
- SST::ConfigLink, 64
- SST::Event, 86
- SST::Exit, 88
- SST::IMPL::TimeVortexPQ, 304
- SST::NullEvent, 141
- SST::OneShot, 145
- SST::PartitionGraph, 162
- SST::PartitionLink, 163
- SST::StopAction, 284
- SST::SyncManager, 292
- SST::ThreadSync, 296
- SST::TimeVortex, 302
- SST::UnitAlgebra, 343
- print_all_params
 - SST::Params, 161
- print_env
 - SST::Config, 56
- print_on_delete
 - SST::Interfaces::TestEvent, 294
- print_timing
 - SST::Config, 56
- printStatus
 - SST::BaseComponent, 36
 - SST::Simulation, 218
- printTimingInfo
 - SST::Config, 54
- printUsage
 - SST::Statistics::StatisticOutput, 268
 - SST::Statistics::StatisticOutputCSV, 273
 - SST::Statistics::StatisticOutputConsole, 271
 - SST::Statistics::StatisticOutputHDF5, 276
 - SST::Statistics::StatisticOutputJSON, 278
 - SST::Statistics::StatisticOutputTxt, 281
- printWithBestSI
 - SST::UnitAlgebra, 343
- probCount
 - SST::RNG::SSTDiscreteDistribution, 226
 - SST::RNG::SSTUniformDistribution, 251
- probabilities
 - SST::RNG::SSTDiscreteDistribution, 226
- processGraphInfo
 - SST::Simulation, 218
- processLinkUntimedData
 - SST::ThreadSync, 296
 - SST::ThreadSyncSimpleSkip, 298
- publish
 - SST::SharedRegion, 201
- pushAllowedKeys
 - SST::Params, 161
- PyComponent, 169
- PySubComponent, 169
- QueryBoolAttribute
 - TiXmlElement, 319
- QueryIntAttribute
 - TiXmlElement, 319
- QueryIntValue
 - TiXmlElement, 305
- rFunctor
 - SST::Link, 125
- ROUTE
 - SST::Interfaces::SimpleNetwork::Request, 182
- RUN
 - SST::Simulation, 215
- rank
 - SST::ConfigComponent, 59
- RankSyncParallelSkip
 - SST::RankSyncParallelSkip, 172
- RankSyncSerialSkip
 - SST::RankSyncSerialSkip, 173
- Read
 - SST::Interfaces::SimpleMem::Request, 177
- readMessage
 - SST::Core::Interprocess::IPCTunnel, 116
- readMessageNB
 - SST::Core::Interprocess::IPCTunnel, 117
- ReadResp
 - SST::Interfaces::SimpleMem::Request, 177
- recv
 - SST::Interfaces::SimpleNetwork, 209
 - SST::Link, 122
- recvInitData
 - SST::Interfaces::SimpleMem, 206
 - SST::Interfaces::SimpleNetwork, 210
 - SST::Link, 123
- recvQueue
 - SST::Link, 125
- recvResponse
 - SST::Interfaces::SimpleMem, 206
- recvUntimedData
 - SST::Interfaces::SimpleNetwork, 210
 - SST::Link, 123
- refDec
 - SST::Exit, 88
- refInc
 - SST::Exit, 88
- registerAsPrimaryComponent
 - SST::Component, 48
- registerBaseUnit
 - SST::Units, 345
- registerClock
 - SST::BaseComponent, 38
 - SST::Simulation, 218
- registerCompoundUnit
 - SST::Units, 345
- registerExit
 - SST::Component, 48
- registerField
 - SST::Statistics::StatisticOutput, 268
- registerHandler
 - SST::Clock, 46
 - SST::OneShot, 145
- registerLink
 - SST::EmptyRankSync, 82
 - SST::EmptyThreadSync, 83

- SST::NewRankSync, 139
- SST::NewThreadSync, 140
- SST::RankSyncParallelSkip, 172
- SST::RankSyncSerialSkip, 173
- SST::SyncBase, 290
- SST::SyncManager, 292
- SST::ThreadSync, 296
- SST::ThreadSyncSimpleSkip, 298
- registerOneShot
 - SST::BaseComponent, 38
 - SST::Simulation, 219
- registerOutputFields
 - SST::Statistics::AccumulatorStatistic, 24
- registerStatistic
 - SST::BaseComponent, 38
- registerTimeBase
 - SST::BaseComponent, 38
- RemoveAttribute
 - TiXmlElement, 319
- ReplaceChild
 - TiXmlNode, 328
- Request
 - SST::Interfaces::SimpleMem::Request, 178
 - SST::Interfaces::SimpleNetwork::Request, 182
- requestToReceive
 - SST::Interfaces::SimpleNetwork, 210
- RequireEvent
 - SST::Factory, 94
- requireEvent
 - SST::Simulation, 219
- reregisterClock
 - SST::BaseComponent, 39
 - SST::Simulation, 219
- resetCollectionCount
 - SST::Statistics::StatisticBase, 258
- resize
 - SST::Core::ThreadSafe::Barrier, 29
- restart
 - SST::RNG::MarsagliaRNG, 133
- RootElement
 - TiXmlDocument, 316
- Row
 - TiXmlBase, 308
- runMode
 - SST::Config, 56
- SST::Action, 25
 - endSimulation, 25
 - print, 25
- SST::Activity, 26
 - execute, 26
 - getDeliveryTime, 26
 - getPriority, 27
 - print, 27
 - setDeliveryTime, 27
 - setPriority, 27
 - setQueueOrder, 27
- SST::Activity::less_time, 118
 - operator(), 119
- SST::Activity::less_time_priority, 119
 - operator(), 119
- SST::Activity::less_time_priority_order, 120
 - operator(), 120
- SST::Activity::pq_less_time_priority, 164
 - operator(), 165
- SST::Activity::pq_less_time_priority_order, 165
 - operator(), 165
- SST::ActivityQueue, 27
 - empty, 28
 - front, 28
 - insert, 28
 - pop, 28
 - size, 28
- SST::BaseComponent, 30
 - complete, 32
 - configureLink, 32, 33
 - configureSelfLink, 33, 34
 - emergencyShutdown, 34
 - finish, 34
 - getCoordinates, 34
 - getCurrentSimTime, 34, 35
 - getCurrentSimTimeMicro, 35
 - getCurrentSimTimeMilli, 35
 - getCurrentSimTimeNano, 35
 - getId, 35
 - getLocalSharedRegion, 35
 - getName, 35
 - getNextClockCycle, 35
 - getParent, 35
 - init, 36
 - isPortConnected, 36
 - loadModule, 36
 - loadModuleWithComponent, 36
 - loadUserSubComponentByIndex< T, ARGS... >, 39
 - printStatus, 36
 - registerClock, 38
 - registerOneShot, 38
 - registerStatistic, 38
 - registerTimeBase, 38
 - reregisterClock, 39
 - setDefaultTimeBase, 39
 - setup, 39
 - Status, 39
 - unregisterClock, 39
 - wasLoadedWithLegacyAPI, 39
- SST::ChangeSet, 43
- SST::Clock, 45
 - Clock, 46
 - getNextCycle, 46
 - print, 46
 - registerHandler, 46
 - schedule, 46
 - unregisterHandler, 46
- SST::Clock::Handler
 - Handler, 95
 - operator(), 96

- SST::Clock::Handler< classT, argT >, 95
- SST::Clock::Handler< classT, void >, 100
 - Handler, 101
 - operator(), 101
- SST::Clock::HandlerBase, 105
 - operator(), 105
- SST::Component, 47
 - primaryComponentDoNotEndSim, 47
 - primaryComponentOKToEndSim, 47
 - registerAsPrimaryComponent, 48
 - registerExit, 48
 - SST_ELI_DECLARE_INFO_EXTERN, 48
 - unregisterExit, 48
- SST::ComponentExtension, 49
- SST::ComponentInfo, 50
 - statEnableList_t, 51
- SST::ComponentInfo::EqualsID, 84
- SST::ComponentInfo::EqualsName, 84
- SST::ComponentInfo::HashID, 108
- SST::ComponentInfo::HashName, 108
- SST::ComponentInfoMap, 51
- SST::Config, 52
 - Config, 53
 - configFile, 54
 - debugFile, 54
 - dump_component_graph_file, 54
 - enable_sig_handling, 55
 - getLibPath, 53
 - getVerboseLevel, 53
 - heartbeatPeriod, 55
 - model_options, 55
 - no_env_config, 55
 - output_config_graph, 55
 - output_core_prefix, 55
 - output_directory, 55
 - output_dot, 55
 - output_json, 55
 - output_xml, 55
 - parseCmdLine, 54
 - partitioner, 56
 - Print, 54
 - print, 54
 - print_env, 56
 - print_timing, 56
 - printTimingInfo, 54
 - runMode, 56
 - setConfigEntryFromModel, 54
 - setStopAt, 54
 - setTimeBase, 54
 - stopAfterSec, 56
 - stopAtCycle, 56
 - timeBase, 56
 - timeVortex, 56
 - verbose, 56
 - world_size, 56
- SST::ConfigComponent, 57
 - ConfigComponent, 58
 - enabledStatistics, 58
 - id, 58
 - links, 59
 - name, 59
 - nextSubID, 59
 - params, 59
 - print, 58
 - rank, 59
 - slot_num, 59
 - subComponents, 59
 - type, 59
 - ultimate_parent, 59
 - weight, 59
- SST::ConfigGraph, 60
 - addComponent, 61
 - addLink, 61
 - addStatisticOutputParameter, 61
 - addStatisticParameterForComponentName, 61
 - checkForStructuralErrors, 61
 - checkRanks, 61
 - containsComponentInRank, 61
 - enableStatisticForComponentName, 62
 - getComponentMap, 62
 - getLinkMap, 62
 - postCreationCleanup, 62
 - print, 62
 - setComponentRanks, 62
 - setLinkNoCut, 62
 - setStatisticLoadLevel, 62
 - setStatisticOutput, 62
 - setStatisticOutputParams, 62
- SST::ConfigLink, 64
 - component, 65
 - current_ref, 65
 - getMinLatency, 64
 - id, 65
 - latency, 65
 - latency_str, 65
 - name, 65
 - no_cut, 65
 - port, 65
 - print, 64
- SST::ConfigStatGroup, 65
 - verifyStatsAndComponents, 66
- SST::ConfigStatOutput, 66
- SST::Core::ConfigGraphOutput, 63
- SST::Core::ConfigGraphOutputException, 63
- SST::Core::DotConfigGraphOutput, 76
- SST::Core::Environment::EnvironmentConfigGroup, 83
- SST::Core::Environment::EnvironmentConfiguration, 83
- SST::Core::Interprocess::CircularBuffer< T >, 45
- SST::Core::Interprocess::IPCTunnel
 - ~IPCTunnel, 116
 - clearBuffer, 116
 - getSharedData, 116
 - IPCTunnel, 116
 - readMessage, 116
 - readMessageNB, 117
 - sharedData, 117

- shutdown, [117](#)
- writeMessage, [117](#)
- SST::Core::Interprocess::IPTunnel< ShareDataType, MsgType >, [115](#)
- SST::Core::Interprocess::SSTMutex, [240](#)
- SST::Core::JSONConfigGraphOutput, [118](#)
- SST::Core::MemPool, [134](#)
 - free, [134](#)
 - getBytesMemUsed, [134](#)
 - malloc, [134](#)
 - MemPool, [134](#)
 - numAlloc, [135](#)
 - numFree, [135](#)
- SST::Core::PythonConfigGraphOutput, [170](#)
- SST::Core::Serialization::need_delete_statics< T >, [138](#)
- SST::Core::Serialization::pvt::raw_ptr_wrapper< TPtr >, [174](#)
- SST::Core::Serialization::pvt::ser_array_wrapper< TPtr, IntType >, [184](#)
- SST::Core::Serialization::pvt::ser_buffer_accessor, [185](#)
- SST::Core::Serialization::pvt::ser_buffer_overrun, [185](#)
- SST::Core::Serialization::pvt::ser_packer, [185](#)
 - pack_buffer, [186](#)
- SST::Core::Serialization::pvt::ser_sizer, [186](#)
- SST::Core::Serialization::pvt::ser_unpacker, [187](#)
 - unpack_buffer, [187](#)
- SST::Core::Serialization::serializable, [187](#)
- SST::Core::Serialization::serializable_builder, [188](#)
- SST::Core::Serialization::serializable_builder_impl< cls_id, [189](#)
- SST::Core::Serialization::serializable_builder_impl< T >, [188](#)
- SST::Core::Serialization::serializable_factory, [189](#)
 - add_builder, [190](#)
- SST::Core::Serialization::serializable_type< T >, [190](#)
- SST::Core::Serialization::serialize< bool >, [191](#)
- SST::Core::Serialization::serialize< pvt::raw_ptr_↵ wrapper< TPtr > >, [191](#)
- SST::Core::Serialization::serialize< pvt::ser_array_↵ _wrapper< T, IntType >, typename std::enable_if< std::is_fundamental< T >↵ ::value || std::is_enum< T >::value >::type >, [192](#)
- SST::Core::Serialization::serialize< pvt::ser_array_↵ _wrapper< T, IntType >, typename std::enable_if< !std::is_fundamental< T >::value && !std::is_enum< T >::value >::type >, [192](#)
- SST::Core::Serialization::serialize< pvt::ser_array_↵ wrapper< void, IntType > >, [193](#)
- SST::Core::Serialization::serialize< SST::Sparse↵ VectorMap< keyT, classT > >, [194](#)
- SST::Core::Serialization::serialize< serializable * >, [193](#)
- SST::Core::Serialization::serialize< std::deque< T > >, [194](#)
- SST::Core::Serialization::serialize< std::list< T > >, [194](#)
- SST::Core::Serialization::serialize< std::map< Key, Value > >, [194](#)
- SST::Core::Serialization::serialize< std::pair< U, V > >, [195](#)
- SST::Core::Serialization::serialize< std::set< T > >, [195](#)
- SST::Core::Serialization::serialize< std::string >, [195](#)
- SST::Core::Serialization::serialize< std::vector< T > >, [196](#)
- SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type >, [196](#)
- SST::Core::Serialization::serialize< T *, typename std::enable_if< std::is_fundamental< T >↵ ::value || std::is_enum< T >::value >::type >, [196](#)
- SST::Core::Serialization::serialize< T, Enable >, [190](#)
- SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_base_of< SST::Core::Serialization::serializable, T >::value >::type >, [197](#)
- SST::Core::Serialization::serialize< T, typename std::enable_if< std::is_fundamental< T >↵ ::value || std::is_enum< T >::value >::type >, [197](#)
- SST::Core::Serialization::serialize< T[N], typename std::enable_if< std::is_fundamental< T >↵ ::value || std::is_enum< T >::value >::type >, [197](#)
- SST::Core::Serialization::serialize< T[N], typename std::enable_if< !std::is_fundamental< T >↵ ::value && !std::is_enum< T >::value >::type >, [198](#)
- SST::Core::Serialization::serializer, [198](#)
- SST::Core::Serialization::statics, [252](#)
- SST::Core::ThreadSafe::Barrier, [29](#)
 - resize, [29](#)
 - wait, [29](#)
- SST::Core::ThreadSafe::BoundedQueue< T >, [40](#)
- SST::Core::ThreadSafe::Spinlock, [223](#)
- SST::Core::ThreadSafe::UnboundedQueue< T >, [336](#)
- SST::Core::XMLConfigGraphOutput, [345](#)
- SST::ELI::Allocator< Base, T, Enable >, [29](#)
- SST::ELI::Allocator< SSTELEMENTPythonModule, T >, [29](#)
- SST::ELI::Builder< Base, Args >, [40](#)
- SST::ELI::BuilderDatabase, [41](#)
- SST::ELI::BuilderInfoImpl< Policy, Policies >, [41](#)
- SST::ELI::BuilderInfoImpl< void >, [41](#)
- SST::ELI::BuilderLibrary< Base, CtorArgs >, [42](#)
- SST::ELI::BuilderLibraryDatabase< Base, CtorArgs >, [42](#)
- SST::ELI::CachedAllocator< Base, T >, [43](#)
- SST::ELI::CtorList< Base, Ctor, Ctors >, [67](#)
- SST::ELI::CtorList< Base, void >, [67](#)
- SST::ELI::DataBase< T >, [68](#)
- SST::ELI::DerivedBuilder< SSTELEMENTPythonModule,

- SSTElementPythonModuleOldELI, const
std::string & >, 75
- SST::ELI::DerivedBuilder< T, Base, Args >, 75
- SST::ELI::ElementsBuilder< Base, CtorTuple >, 80
- SST::ELI::ElementsBuilder< Base, std::tuple< Args...
> >, 80
- SST::ELI::ElementsInfo< Base >, 80
- SST::ELI::ExtendedCtor< NewCtor, OldCtor >, 88
- SST::ELI::GetParams< T, Enable >, 94
- SST::ELI::GetParams< T, typename MethodDetect<
decltype(T::ELI_getParams())>::type >, 95
- SST::ELI::InfoDatabase, 111
- SST::ELI::InfoLibrary< Base >, 111
- SST::ELI::InfoLibraryDatabase< Base >, 111
- SST::ELI::InfoPorts< T, Enable >, 112
- SST::ELI::InfoPorts< T, typename MethodDetect<
decltype(T::ELI_getPorts())>::type >, 112
- SST::ELI::InfoStats< T, Enable >, 112
- SST::ELI::InfoStats< T, typename MethodDetect<
decltype(T::ELI_getStatistics())>::type >, 112
- SST::ELI::InfoSubs< T, Enable >, 113
- SST::ELI::InfoSubs< T, typename MethodDetect<
decltype(T::ELI_getSubComponentSlots())>←
::type >, 113
- SST::ELI::InstantiateBuilder< Base, T >, 114
- SST::ELI::InstantiateBuilderInfo< Base, T >, 115
- SST::ELI::LoadedLibraries, 128
addLoader, 128
- SST::ELI::MethodDetect< T >, 137
- SST::ELI::NoValidConstructorsForDerivedType< 0 >,
141
- SST::ELI::NoValidConstructorsForDerivedType<
NumValid >, 140
- SST::ELI::ProvidesCategory, 166
- SST::ELI::ProvidesDefaultInfo, 166
- SST::ELI::ProvidesInterface, 167
- SST::ELI::ProvidesParams, 167
- SST::ELI::ProvidesPorts, 168
- SST::ELI::ProvidesStats, 168
- SST::ELI::ProvidesSubComponentSlots, 168
- SST::ELI::SingleCtor< Base, Args >, 221
- SST::ELI::is_tuple_constructible< T, std::tuple< Args...
> >, 118
- SST::ELI::is_tuple_constructible< T, U >, 117
- SST::ElemLoader, 80
ElemLoader, 81
getPotentialElements, 81
loadLibrary, 81
- SST::ElementInfoParam, 76
defaultValue, 77
description, 77
name, 77
- SST::ElementInfoPort, 77
description, 77
name, 77
validEvents, 77
- SST::ElementInfoPort2, 78
description, 78
- name, 78
validEvents, 78
- SST::ElementInfoStatistic, 79
description, 79
enableLevel, 79
name, 79
units, 79
- SST::ElementInfoSubComponentSlot, 79
- SST::EmptyRankSync, 81
registerLink, 82
- SST::EmptyThreadSync, 82
registerLink, 83
- SST::Event, 85
clone, 86
delivery_link, 87
execute, 86
generateUniqueld, 86
getDeliveryLink, 86
getLinkId, 86
id_type, 86
NO_ID, 87
print, 86
setDeliveryLink, 86
setRemoteEvent, 86
- SST::Event::Handler
Handler, 98
operator(), 98
- SST::Event::Handler< classT, argT >, 97
- SST::Event::Handler< classT, void >, 102
Handler, 103
operator(), 103
- SST::Event::HandlerBase, 106
operator(), 107
- SST::Exit, 87
execute, 88
Exit, 87
print, 88
refDec, 88
refInc, 88
- SST::Factory, 89
Create, 90
CreateComponent, 90
CreateModule, 90
CreateModuleWithComponent, 90
CreatePartitioner, 91
CreateStatistic, 91
CreateSubComponent, 91
DoesComponentInfoStatisticNameExist, 92
DoesSubComponentSlotExist, 92
GetComponentInfoStatisticEnableLevel, 92
GetComponentInfoStatisticUnits, 93
getParamNames, 93
getPythonModule, 93
hasLibrary, 93
isPortNameValid, 93
isSubComponentLoadableUsingAPI, 94
RequireEvent, 94
- SST::IMPL::Partition::SSTLinearPartition, 237

- partOutput, 239
- performPartition, 238
- SSTLinearPartition, 238
- SST::IMPL::Partition::SSTRoundRobinPartition, 246
 - performPartition, 246
- SST::IMPL::Partition::SSTSelfPartition, 247
 - performPartition, 247, 248
 - SST_ELI_REGISTER_PARTITIONER, 248
- SST::IMPL::Partition::SSTSinglePartition, 248
 - performPartition, 249
 - SST_ELI_REGISTER_PARTITIONER, 249
- SST::IMPL::Partition::SimplePartitioner, 212
 - performPartition, 212
- SST::IMPL::TimeVortexPQ, 303
 - empty, 303
 - front, 303
 - insert, 303
 - pop, 303
 - print, 304
 - size, 304
- SST::InitQueue, 113
 - empty, 114
 - front, 114
 - insert, 114
 - pop, 114
 - size, 114
- SST::Interfaces::SimpleMem, 204
 - Addr, 205
 - getLink, 205
 - initialize, 205
 - recvInitData, 206
 - recvResponse, 206
 - sendInitData, 206
 - sendRequest, 206
 - SimpleMem, 205
- SST::Interfaces::SimpleMem::Handler
 - Handler, 99
 - operator(), 99
- SST::Interfaces::SimpleMem::Handler< classT, argT >, 98
- SST::Interfaces::SimpleMem::Handler< classT, void >, 103
 - Handler, 104
 - operator(), 104
- SST::Interfaces::SimpleMem::HandlerBase, 107
 - operator(), 107
- SST::Interfaces::SimpleMem::Request, 175
 - addr, 180
 - addrs, 180
 - cmd, 180
 - Command, 177
 - custOpc, 180
 - CustomCmd, 177
 - data, 180
 - dataVec, 177
 - F_FLUSH_SUCCESS, 177
 - F_LOCKED, 177
 - F_NONCACHEABLE, 177
 - Flags, 177
 - flags, 180
 - flags_t, 177
 - FlushLine, 177
 - FlushLineInv, 177
 - FlushLineResp, 177
 - getCustomOpc, 178
 - getFlags, 178
 - getInstructionPointer, 178
 - getMemFlags, 178
 - getVirtualAddress, 179
 - id, 180
 - id_t, 177
 - instrPtr, 180
 - memFlags, 181
 - Read, 177
 - ReadResp, 177
 - Request, 178
 - setFlags, 179
 - setInstructionPointer, 179
 - setMemFlags, 179
 - setPayload, 179
 - setVirtualAddress, 180
 - size, 181
 - TxBegin, 177
 - TxEnd, 177
 - virtualAddr, 181
 - Write, 177
 - WriteResp, 177
- SST::Interfaces::SimpleNetwork, 207
 - complete, 208
 - finish, 208
 - getEndpointID, 208
 - getLinkBW, 208
 - init, 209
 - initialize, 209
 - isNetworkInitialized, 209
 - nid_t, 208
 - recv, 209
 - recvInitData, 210
 - recvUntimedData, 210
 - requestToReceive, 210
 - send, 210
 - sendInitData, 210
 - sendUntimedData, 211
 - setNotifyOnReceive, 211
 - setNotifyOnSend, 211
 - setup, 211
 - SimpleNetwork, 208
 - spaceToSend, 211
- SST::Interfaces::SimpleNetwork::Handler
 - Handler, 100
- SST::Interfaces::SimpleNetwork::Handler< classT, argT >, 99
- SST::Interfaces::SimpleNetwork::Handler< classT, void >, 104
 - Handler, 105
- SST::Interfaces::SimpleNetwork::HandlerBase, 107

- SST::Interfaces::SimpleNetwork::NetworkInspector, 138
 - initialize, 138
- SST::Interfaces::SimpleNetwork::Request, 181
 - allow_adaptive, 183
 - dest, 183
 - FULL, 182
 - givePayload, 182
 - head, 183
 - inspectPayload, 183
 - NONE, 182
 - ROUTE, 182
 - Request, 182
 - size_in_bits, 183
 - src, 183
 - tail, 183
 - takePayload, 183
 - TraceType, 182
 - vn, 184
- SST::Interfaces::StringEvent, 284
 - getString, 285
 - StringEvent, 284, 285
- SST::Interfaces::TestEvent, 294
 - count, 294
 - print_on_delete, 294
- SST::Link, 120
 - addRecvLatency, 122
 - afterInitQueue, 124
 - afterRunQueue, 124
 - configuredQueue, 124
 - defaultTimeBase, 125
 - deliverEvent, 122
 - getDefaultTimeBase, 122
 - getId, 122
 - latency, 125
 - Link, 122
 - pair_link, 125
 - rFunctor, 125
 - recv, 122
 - recvInitData, 123
 - recvQueue, 125
 - recvUntimedData, 123
 - send, 123
 - sendInitData, 123
 - sendUntimedData, 124
 - setDefaultTimeBase, 124
 - setFunctor, 124
 - setLatency, 124
 - setPolling, 124
 - uninitQueue, 125
 - untimedQueue, 125
- SST::LinkMap, 125
 - addSelfPort, 126
 - empty, 126
 - getLink, 126
 - getLinkMap, 126
 - insertLink, 126
- SST::LinkPair, 127
 - getId, 127
 - getLeft, 127
 - getRight, 127
 - LinkPair, 127
- SST::LoaderData, 128
 - advise_handle, 129
- SST::Module, 137
- SST::NewRankSync, 138
 - registerLink, 139
- SST::NewThreadSync, 139
 - registerLink, 140
- SST::NullEvent, 141
 - execute, 141
 - print, 141
- SST::OneShot, 144
 - isScheduled, 145
 - OneShot, 144
 - print, 145
 - registerHandler, 145
- SST::OneShot::Handler
 - Handler, 97
 - operator(), 97
- SST::OneShot::Handler< classT, argT >, 96
- SST::OneShot::Handler< classT, void >, 101
 - Handler, 102
 - operator(), 102
- SST::OneShot::HandlerBase, 106
 - operator(), 106
- SST::Output, 145
 - debug, 148
 - debugPrefix, 148
 - FILE, 147
 - fatal, 149
 - file, 153
 - flush, 149
 - getOutputLocation, 149
 - getPrefix, 149
 - getVerboseLevel, 149
 - getVerboseMask, 149
 - init, 150
 - line, 154
 - NONE, 147
 - Output, 147, 148
 - output, 150, 151
 - output_location_t, 147
 - STDERR, 147
 - STDOUT, 147
 - setFileName, 151
 - setOutputLocation, 151
 - setPrefix, 151
 - setVerboseLevel, 152
 - setVerboseMask, 152
 - verbose, 152
 - verbosePrefix, 153
- SST::Params, 154
 - clear, 157
 - contains, 157
 - count, 157
 - empty, 157

- enableVerify, 157
- find, 157–159
- find_array, 160
- find_prefix_params, 160
- getParamName, 160
- insert, 160
- key_type, 156
- KeySet_t, 156
- operator=, 160
- Params, 156
- popAllowedKeys, 161
- print_all_params, 161
- pushAllowedKeys, 161
- size, 161
- verifyParam, 161
- SST::Partition::SSTPartitioner, 240
 - performPartition, 241
- SST::PartitionComponent, 161
- SST::PartitionGraph, 162
 - print, 162
- SST::PartitionLink, 162
 - getMinLatency, 163
 - print, 163
- SST::PollingLinkQueue, 163
 - empty, 163
 - front, 163
 - insert, 164
 - pop, 164
 - size, 164
- SST::RNG::MarsagliaRNG, 131
 - generateNextInt32, 132
 - generateNextInt64, 132
 - generateNextUInt32, 133
 - generateNextUInt64, 133
 - MarsagliaRNG, 132
 - nextUniform, 133
 - restart, 133
 - seed, 133
- SST::RNG::MersenneRNG, 135
 - ~MersenneRNG, 136
 - generateNextInt32, 136
 - generateNextInt64, 136
 - generateNextUInt32, 136
 - generateNextUInt64, 136
 - MersenneRNG, 135, 136
 - nextUniform, 136
 - seed, 136
- SST::RNG::SSTConstantDistribution, 223
 - ~SSTConstantDistribution, 224
 - getMean, 224
 - getNextDouble, 224
 - mean, 225
 - SSTConstantDistribution, 224
- SST::RNG::SSTDiscreteDistribution, 225
 - ~SSTDiscreteDistribution, 226
 - baseDistrib, 226
 - deleteDistrib, 226
 - getNextDouble, 226
 - probCount, 226
 - probabilities, 226
 - SSTDiscreteDistribution, 225, 226
- SST::RNG::SSTExponentialDistribution, 230
 - ~SSTExponentialDistribution, 231
 - baseDistrib, 231
 - deleteDistrib, 231
 - getLambda, 231
 - getNextDouble, 231
 - lambda, 231
 - SSTExponentialDistribution, 230
- SST::RNG::SSTGaussianDistribution, 232
 - ~SSTGaussianDistribution, 233
 - baseDistrib, 234
 - deleteDistrib, 234
 - getMean, 233
 - getNextDouble, 233
 - getStandardDev, 233
 - mean, 234
 - SSTGaussianDistribution, 232, 233
 - stddev, 234
 - unusedPair, 234
 - usePair, 234
- SST::RNG::SSTPoissonDistribution, 242
 - ~SSTPoissonDistribution, 243
 - baseDistrib, 243
 - deleteDistrib, 243
 - getLambda, 243
 - getNextDouble, 243
 - lambda, 243
 - SSTPoissonDistribution, 242
- SST::RNG::SSTRandom, 244
 - ~SSTRandom, 244
 - generateNextInt32, 244
 - generateNextInt64, 244
 - generateNextUInt32, 244
 - generateNextUInt64, 244
 - nextUniform, 245
- SST::RNG::SSTRandomDistribution, 245
 - ~SSTRandomDistribution, 245
 - getNextDouble, 246
 - SSTRandomDistribution, 245
- SST::RNG::SSTUniformDistribution, 249
 - ~SSTUniformDistribution, 250
 - baseDistrib, 251
 - deleteDistrib, 251
 - getNextDouble, 250
 - probCount, 251
 - SSTUniformDistribution, 250
- SST::RNG::XORShiftRNG, 346
 - ~XORShiftRNG, 346
 - generateNextInt32, 347
 - generateNextInt64, 347
 - generateNextUInt32, 347
 - generateNextUInt64, 347
 - nextUniform, 347
 - seed, 347
 - XORShiftRNG, 346

- SST::RankInfo, 171
 - inRange, 171
- SST::RankSyncParallelSkip, 171
 - exchangeLinkUntimedData, 172
 - finalizeLinkConfigurations, 172
 - prepareForComplete, 172
 - RankSyncParallelSkip, 172
 - registerLink, 172
- SST::RankSyncSerialSkip, 173
 - exchangeLinkUntimedData, 173
 - finalizeLinkConfigurations, 173
 - prepareForComplete, 173
 - RankSyncSerialSkip, 173
 - registerLink, 173
- SST::RegionInfo, 174
 - getMergeInfo, 175
- SST::RegionInfo::BulkMergeInfo, 42
- SST::RegionInfo::ChangeSetMergeInfo, 44
- SST::RegionInfo::RegionMergeInfo, 175
- SST::SST_ELI_element_version_extraction, 223
- SST::SSTElementPythonModule, 227
 - createPrimaryModule, 228
 - SSTElementPythonModule, 227
- SST::SSTElementPythonModuleCode, 228
 - addSubModule, 229
 - getFullModuleName, 229
- SST::SSTElementPythonModuleOldELI, 229
- SST::SSTInfoConfig, 234
 - debugEnabled, 235
 - getElementsToProcessArray, 235
 - getFilterMap, 235
 - getOptionBits, 235
 - getXMLFilePath, 235
 - parseCmdLine, 235
 - SSTInfoConfig, 235
- SST::SSTLibraryInfo, 236
 - getLibraryName, 236
 - outputHumanReadable, 237
 - outputXML, 237
 - SSTLibraryInfo, 236
- SST::SSTModelDescription, 239
 - createConfigGraph, 240
- SST::SelfLink, 184
- SST::SharedRegion, 199
 - getLocalShareID, 200
 - getPtr, 200
 - getRawPtr, 200
 - getSize, 200
 - isReady, 200
 - modifyRegion, 201
 - publish, 201
- SST::SharedRegionImpl, 201
- SST::SharedRegionInitializedMerger, 201
 - merge, 202
- SST::SharedRegionManager, 202
- SST::SharedRegionManagerImpl, 203
- SST::SharedRegionMerger, 203
 - merge, 204
- SST::Simulation, 213
 - BOTH, 215
 - clockMap_t, 215
 - complete, 215
 - createSimulation, 215
 - getComponent, 216
 - getComponentInfoMap, 216
 - getComponentLinkMap, 216
 - getCurrentPriority, 216
 - getCurrentSimCycle, 216
 - getElapsedSimTime, 216
 - getEndSimCycle, 216
 - getExit, 216
 - getFinalSimTime, 216
 - getLocalMinimumNextActivityTime, 216
 - getNextActivityTime, 217
 - getNextClockCycle, 217
 - getNumRanks, 217
 - getOutputDirectory, 217
 - getRank, 217
 - getSharedRegionManager, 217
 - getSimulation, 217
 - getSimulationMode, 217
 - getSimulationOutput, 217
 - getStatisticsProcessingEngine, 218
 - getTimeLord, 218
 - INIT, 215
 - initialize, 218
 - insertActivity, 218
 - isWireUpFinished, 218
 - Mode_t, 215
 - oneShotMap_t, 215
 - performWireUp, 218
 - printStatus, 218
 - processGraphInfo, 218
 - RUN, 215
 - registerClock, 218
 - registerOneShot, 219
 - requireEvent, 219
 - reregisterClock, 219
 - setOutputDirectory, 219
 - setSignal, 219
 - setStopAtCycle, 219
 - setup, 219
 - shutdown, 220
 - UNKNOWN, 215
 - unregisterClock, 220
- SST::SimulatorHeartbeat, 220
 - SimulatorHeartbeat, 220
- SST::SparseVectorMap< keyT, classT >, 221
- SST::SparseVectorMap< keyT, keyT >, 222
- SST::Statistics::AccumulatorStatistic
 - addData_impl, 22
 - clearStatisticData, 22
 - getArithmeticMean, 22
 - getCount, 22
 - getMax, 23
 - getMin, 23

- getStandardDeviation, [23](#)
- getSum, [23](#)
- getSumSquared, [23](#)
- getVariance, [24](#)
- registerOutputFields, [24](#)
- statParams, [24](#)
- SST::Statistics::AccumulatorStatistic< NumberBase >, [21](#)
- SST::Statistics::HistogramStatistic
 - addData_impl, [110](#)
 - statParams, [110](#)
- SST::Statistics::HistogramStatistic< BinDataType >, [109](#)
- SST::Statistics::ImplementsStatFields, [110](#)
- SST::Statistics::NullStatistic< T >, [142](#)
- SST::Statistics::NullStatisticBase< std::tuple< Args... >, false >, [143](#)
- SST::Statistics::NullStatisticBase< T, B >, [142](#)
- SST::Statistics::NullStatisticBase< T, false >, [143](#)
- SST::Statistics::NullStatisticBase< T, true >, [143](#)
- SST::Statistics::Statistic
 - addData, [253](#)
 - Statistic, [253](#)
- SST::Statistics::Statistic< T >, [252](#)
- SST::Statistics::StatisticBase, [253](#)
 - clearStatisticData, [255](#)
 - delayCollection, [255](#)
 - delayOutput, [255](#)
 - disable, [256](#)
 - enable, [256](#)
 - getCollectionCount, [256](#)
 - getCollectionCountLimit, [256](#)
 - getCompName, [256](#)
 - getComponent, [256](#)
 - getFlagClearDataOnOutput, [256](#)
 - getFlagOutputAtEndOfSim, [256](#)
 - getFlagResetCountOnOutput, [256](#)
 - getFullStatName, [257](#)
 - getParams, [257](#)
 - getRegisteredCollectionMode, [257](#)
 - getStatDataType, [257](#)
 - getStatDataTypeFullName, [257](#)
 - getStatDataTypeShortName, [257](#)
 - getStatName, [257](#)
 - getStatSubId, [257](#)
 - getStatTypeName, [257](#)
 - incrementCollectionCount, [257](#)
 - isEnabled, [258](#)
 - isNullStatistic, [258](#)
 - isOutputEnabled, [258](#)
 - isReady, [258](#)
 - resetCollectionCount, [258](#)
 - setCollectionCount, [258](#)
 - setCollectionCountLimit, [258](#)
 - setFlagClearDataOnOutput, [258](#)
 - setFlagOutputAtEndOfSim, [258](#)
 - setFlagResetCountOnOutput, [258](#)
 - setStatisticData, [259](#)
 - setStatisticTypeName, [259](#)
 - StatMode_t, [255](#)
 - StatisticBase, [255](#)
- SST::Statistics::StatisticCollector< std::tuple< Args... >, false >, [259](#)
- SST::Statistics::StatisticCollector< T, F >, [259](#)
- SST::Statistics::StatisticCollector< T, true >, [260](#)
- SST::Statistics::StatisticFieldInfo, [260](#)
 - getFieldHandle, [261](#)
 - getFieldName, [261](#)
 - getFieldType, [261](#)
 - getFieldUniqueName, [261](#)
 - getStatName, [261](#)
 - operator==, [261](#)
 - setFieldHandle, [262](#)
 - StatisticFieldInfo, [260](#)
- SST::Statistics::StatisticFieldType< T >, [262](#)
- SST::Statistics::StatisticFieldTypeBase, [262](#)
- SST::Statistics::StatisticGroup, [263](#)
- SST::Statistics::StatisticInfo, [264](#)
- SST::Statistics::StatisticOutput, [264](#)
 - acceptsGroups, [266](#)
 - checkOutputParameters, [266](#)
 - endOfSimulation, [266](#)
 - getFieldInfoArray, [266](#)
 - getFieldTypeShortName, [266](#)
 - getOutputParameters, [267](#)
 - getRegisteredField, [267](#)
 - getStatisticOutputName, [267](#)
 - implStartOutputEntries, [267](#)
 - implStopOutputEntries, [268](#)
 - outputField, [268](#)
 - printUsage, [268](#)
 - registerField, [268](#)
 - startOfSimulation, [269](#)
 - StatisticOutput, [266](#)
- SST::Statistics::StatisticOutputCSV, [271](#)
 - checkOutputParameters, [272](#)
 - endOfSimulation, [272](#)
 - implStartOutputEntries, [272](#)
 - implStopOutputEntries, [273](#)
 - outputField, [273](#)
 - printUsage, [273](#)
 - SST_ELI_REGISTER_DERIVED, [273](#)
 - startOfSimulation, [273](#)
- SST::Statistics::StatisticOutputConsole, [269](#)
 - checkOutputParameters, [270](#)
 - endOfSimulation, [270](#)
 - implStartOutputEntries, [270](#)
 - implStopOutputEntries, [270](#)
 - outputField, [270](#)
 - printUsage, [271](#)
 - SST_ELI_REGISTER_DERIVED, [271](#)
 - startOfSimulation, [271](#)
- SST::Statistics::StatisticOutputHDF5, [274](#)
 - acceptsGroups, [275](#)
 - checkOutputParameters, [275](#)
 - endOfSimulation, [275](#)

- implStartOutputEntries, 275
- implStopOutputEntries, 275
- outputField, 276
- printUsage, 276
- SST_ELI_REGISTER_DERIVED, 276
- startOfSimulation, 276
- SST::Statistics::StatisticOutputJSON, 277
 - checkOutputParameters, 277
 - endOfSimulation, 277
 - implStartOutputEntries, 278
 - implStopOutputEntries, 278
 - outputField, 278
 - printUsage, 278
 - SST_ELI_REGISTER_DERIVED, 278
 - startOfSimulation, 279
- SST::Statistics::StatisticOutputTxt, 279
 - checkOutputParameters, 280
 - endOfSimulation, 280
 - implStartOutputEntries, 280
 - implStopOutputEntries, 280
 - outputField, 280
 - printUsage, 281
 - SST_ELI_REGISTER_DERIVED, 281
 - startOfSimulation, 281
- SST::Statistics::StatisticProcessingEngine, 281
 - performGlobalStatisticOutput, 282
 - performStatisticOutput, 282
- SST::Statistics::UniqueCountStatistic
 - addData_impl, 339
 - statParams, 339
- SST::Statistics::UniqueCountStatistic< T >, 338
- SST::StopAction, 283
 - execute, 284
 - print, 284
 - StopAction, 283
- SST::SubComponent, 285
 - finish, 286
 - init, 286
 - setup, 286
- SST::SubComponentSlotInfo, 286
 - const, 288
 - createAll, 287
 - createAllSparse, 288
 - insertNulls, 288
- SST::SyncBase, 289
 - exchangeLinkUntimedData, 290
 - finalizeLinkConfigurations, 290
 - registerLink, 290
 - SyncBase, 290
- SST::SyncManager, 291
 - exchangeLinkUntimedData, 291
 - execute, 291
 - finalizeLinkConfigurations, 291
 - prepareForComplete, 292
 - print, 292
 - registerLink, 292
- SST::SyncQueue, 292
 - clear, 293
 - empty, 293
 - front, 293
 - getData, 293
 - insert, 293
 - pop, 293
 - size, 293
- SST::SyncQueue::Header, 108
- SST::ThreadSync, 295
 - execute, 295
 - finalizeLinkConfigurations, 295
 - print, 296
 - processLinkUntimedData, 296
 - registerLink, 296
 - ThreadSync, 295
- SST::ThreadSyncQueue, 296
 - empty, 297
 - front, 297
 - insert, 297
 - pop, 297
 - size, 297
- SST::ThreadSyncSimpleSkip, 297
 - finalizeLinkConfigurations, 298
 - processLinkUntimedData, 298
 - registerLink, 298
 - ThreadSyncSimpleSkip, 298
- SST::TimeConverter, 299
 - convertFromCoreTime, 299
 - convertToCoreTime, 299
 - getFactor, 299
- SST::TimeLord, 300
 - getMicro, 300
 - getMilli, 300
 - getNano, 300
 - getSimCycles, 300
 - getTimeBase, 300
 - getTimeConverter, 300, 301
- SST::TimeVortex, 301
 - empty, 302
 - front, 302
 - insert, 302
 - pop, 302
 - print, 302
 - size, 302
- SST::Tokenizer< TokenizerFunc >, 336
- SST::TraceFunction, 336
 - output, 336
- SST::UninitializedQueue, 337
 - empty, 338
 - front, 338
 - insert, 338
 - pop, 338
 - size, 338
 - UninitializedQueue, 337
- SST::UnitAlgebra, 340
 - getRoundedValue, 341
 - getValue, 341
 - hasUnits, 341
 - invert, 341

- operator<, 342
- operator<=, 342
- operator>, 342
- operator>=, 342
- operator*=: 341
- operator+=, 342
- operator-=, 342
- operator/=: 342
- print, 343
- printWithBestSI, 343
- toString, 343
- toStringBestSI, 343
- UnitAlgebra, 341
- SST::Units, 343
 - invert, 344
 - operator!=, 344
 - operator*=: 344
 - operator/=: 344
 - operator=, 344
 - operator==, 344
 - registerBaseUnit, 345
 - registerCompoundUnit, 345
 - toString, 345
 - Units, 344
- SST::char_delimiter, 44
 - operator(), 45
- SST::decimal_fixedpoint
 - convert_to, 71
 - decimal_fixedpoint, 69, 70
 - getFractionWords, 71
 - getWholeWords, 71
 - inverse, 71
 - negate, 71
 - operator!=, 71
 - operator<, 72
 - operator<=, 73
 - operator>, 74
 - operator>=, 74
 - operator*=: 72
 - operator+=, 72
 - operator-=, 72
 - operator/=: 72
 - operator=, 73
 - operator==, 73
 - toDouble, 74
 - toLong, 74
 - toString, 74
 - toUnsignedLong, 74
- SST::decimal_fixedpoint< whole_words, fraction_words
>, 68
- SST::escaped_list_separator, 84
 - operator(), 84
- SST::sstLongOpts_s, 239
- SST_ELI_DECLARE_INFO_EXTERN
 - SST::Component, 48
- SST_ELI_REGISTER_DERIVED
 - SST::Statistics::StatisticOutputCSV, 273
 - SST::Statistics::StatisticOutputConsole, 271
 - SST::Statistics::StatisticOutputHDF5, 276
 - SST::Statistics::StatisticOutputJSON, 278
 - SST::Statistics::StatisticOutputTxt, 281
- SST_ELI_REGISTER_PARTITIONER
 - SST::IMPL::Partition::SSTSelfPartition, 248
 - SST::IMPL::Partition::SSTSinglePartition, 249
- SSTConstantDistribution
 - SST::RNG::SSTConstantDistribution, 224
- SSTDiscreteDistribution
 - SST::RNG::SSTDiscreteDistribution, 225, 226
- SSTElementPythonModule
 - SST::SSTElementPythonModule, 227
- SSTExponentialDistribution
 - SST::RNG::SSTExponentialDistribution, 230
- SSTGaussianDistribution
 - SST::RNG::SSTGaussianDistribution, 232, 233
- SSTInfoConfig
 - SST::SSTInfoConfig, 235
- SSTLibraryInfo
 - SST::SSTLibraryInfo, 236
- SSTLinearPartition
 - SST::IMPL::Partition::SSTLinearPartition, 238
- SSTPoissonDistribution
 - SST::RNG::SSTPoissonDistribution, 242
- SSTRandomDistribution
 - SST::RNG::SSTRandomDistribution, 245
- SSTUniformDistribution
 - SST::RNG::SSTUniformDistribution, 250
- STDERR
 - SST::Output, 147
- STDOUT
 - SST::Output, 147
- schedule
 - SST::Clock, 46
- seed
 - SST::RNG::MarsagliaRNG, 133
 - SST::RNG::MersenneRNG, 136
 - SST::RNG::XORShiftRNG, 347
- send
 - SST::Interfaces::SimpleNetwork, 210
 - SST::Link, 123
- sendInitData
 - SST::Interfaces::SimpleMem, 206
 - SST::Interfaces::SimpleNetwork, 210
 - SST::Link, 123
- sendRequest
 - SST::Interfaces::SimpleMem, 206
- sendUntimedData
 - SST::Interfaces::SimpleNetwork, 211
 - SST::Link, 124
- SetAttribute
 - TiXmlElement, 319
- setCollectionCount
 - SST::Statistics::StatisticBase, 258
- setCollectionCountLimit
 - SST::Statistics::StatisticBase, 258
- setComponentRanks
 - SST::ConfigGraph, 62

- SetCondenseWhiteSpace
 - TiXmlBase, 309
- setConfigEntryFromModel
 - SST::Config, 54
- setDefaultTimeBase
 - SST::BaseComponent, 39
 - SST::Link, 124
- setDeliveryLink
 - SST::Event, 86
- setDeliveryTime
 - SST::Activity, 27
- SetDoubleAttribute
 - TiXmlElement, 319
- setFieldHandle
 - SST::Statistics::StatisticFieldInfo, 262
- setFileName
 - SST::Output, 151
- setFlagClearDataOnOutput
 - SST::Statistics::StatisticBase, 258
- setFlagOutputAtEndOfSim
 - SST::Statistics::StatisticBase, 258
- setFlagResetCountOnOutput
 - SST::Statistics::StatisticBase, 258
- setFlags
 - SST::Interfaces::SimpleMem::Request, 179
- setFunctor
 - SST::Link, 124
- SetIndent
 - TiXmlPrinter, 331
- setInstructionPointer
 - SST::Interfaces::SimpleMem::Request, 179
- setLatency
 - SST::Link, 124
- SetLineBreak
 - TiXmlPrinter, 331
- setLinkNoCut
 - SST::ConfigGraph, 62
- setMemFlags
 - SST::Interfaces::SimpleMem::Request, 179
- setNotifyOnReceive
 - SST::Interfaces::SimpleNetwork, 211
- setNotifyOnSend
 - SST::Interfaces::SimpleNetwork, 211
- setOutputDirectory
 - SST::Simulation, 219
- setOutputLocation
 - SST::Output, 151
- setPayload
 - SST::Interfaces::SimpleMem::Request, 179
- setPolling
 - SST::Link, 124
- setPrefix
 - SST::Output, 151
- setPriority
 - SST::Activity, 27
- setQueueOrder
 - SST::Activity, 27
- setRemoteEvent
 - SST::Event, 86
- setSignal
 - SST::Simulation, 219
- setStatisticDataType
 - SST::Statistics::StatisticBase, 259
- setStatisticLoadLevel
 - SST::ConfigGraph, 62
- setStatisticOutput
 - SST::ConfigGraph, 62
- setStatisticOutputParams
 - SST::ConfigGraph, 62
- setStatisticTypeName
 - SST::Statistics::StatisticBase, 259
- setStopAt
 - SST::Config, 54
- setStopAtCycle
 - SST::Simulation, 219
- SetStreamPrinting
 - TiXmlPrinter, 331
- SetTabSize
 - TiXmlDocument, 316
- setTimeBase
 - SST::Config, 54
- SetValue
 - TiXmlNode, 328
- setVerboseLevel
 - SST::Output, 152
- setVerboseMask
 - SST::Output, 152
- setVirtualAddress
 - SST::Interfaces::SimpleMem::Request, 180
- setup
 - SST::BaseComponent, 39
 - SST::Interfaces::SimpleNetwork, 211
 - SST::Simulation, 219
 - SST::SubComponent, 286
- sharedData
 - SST::Core::Interprocess::IPCTunnel, 117
- shutdown
 - SST::Core::Interprocess::IPCTunnel, 117
 - SST::Simulation, 220
- SimThreadInfo_t, 213
- SimpleMem
 - SST::Interfaces::SimpleMem, 205
- SimpleNetwork
 - SST::Interfaces::SimpleNetwork, 208
- SimulatorHeartbeat
 - SST::SimulatorHeartbeat, 220
- size
 - SST::ActivityQueue, 28
 - SST::IMPL::TimeVortexPQ, 304
 - SST::InitQueue, 114
 - SST::Interfaces::SimpleMem::Request, 181
 - SST::Params, 161
 - SST::PollingLinkQueue, 164
 - SST::SyncQueue, 293
 - SST::ThreadSyncQueue, 297
 - SST::TimeVortex, 302

- SST::UninitializedQueue, 338
- size_in_bits
 - SST::Interfaces::SimpleNetwork::Request, 183
- slist, 221
- slot_num
 - SST::ConfigComponent, 59
- spaceToSend
 - SST::Interfaces::SimpleNetwork, 211
- sprockit::serializable_ptr_type, 190
- src
 - SST::Interfaces::SimpleNetwork::Request, 183
- startOfSimulation
 - SST::Statistics::StatisticOutput, 269
 - SST::Statistics::StatisticOutputCSV, 273
 - SST::Statistics::StatisticOutputConsole, 271
 - SST::Statistics::StatisticOutputHDF5, 276
 - SST::Statistics::StatisticOutputJSON, 279
 - SST::Statistics::StatisticOutputTxt, 281
- statEnableList_t
 - SST::ComponentInfo, 51
- StatGroupPy_t, 251
- StatMode_t
 - SST::Statistics::StatisticBase, 255
- StatOutputPy_t, 283
- statParams
 - SST::Statistics::AccumulatorStatistic, 24
 - SST::Statistics::HistogramStatistic, 110
 - SST::Statistics::UniqueCountStatistic, 339
- Statistic
 - SST::Statistics::Statistic, 253
- StatisticBase
 - SST::Statistics::StatisticBase, 255
- StatisticFieldInfo
 - SST::Statistics::StatisticFieldInfo, 260
- StatisticOutput
 - SST::Statistics::StatisticOutput, 266
- Status
 - SST::BaseComponent, 39
- stddev
 - SST::RNG::SSTGaussianDistribution, 234
- StopAction
 - SST::StopAction, 283
- stopAfterSec
 - SST::Config, 56
- stopAtCycle
 - SST::Config, 56
- StringEvent
 - SST::Interfaces::StringEvent, 284, 285
- subComponents
 - SST::ConfigComponent, 59
- symlist_chain, 289
- SyncBase
 - SST::SyncBase, 290
- tail
 - SST::Interfaces::SimpleNetwork::Request, 183
- takePayload
 - SST::Interfaces::SimpleNetwork::Request, 183
- Text
 - TiXmlHandle, 322
- ThreadSync
 - SST::ThreadSync, 295
- ThreadSyncSimpleSkip
 - SST::ThreadSyncSimpleSkip, 298
- TiXmlAttribute, 304
 - Print, 305
 - QueryIntValue, 305
- TiXmlAttributeSet, 306
- TiXmlBase, 306
 - EncodeString, 308
 - errorString, 309
 - Print, 308
 - Row, 308
 - SetCondenseWhiteSpace, 309
 - utf8ByteTable, 309
- TiXmlComment, 310
 - Accept, 311
 - Print, 311
- TiXmlCursor, 311
- TiXmlDeclaration, 312
 - Accept, 313
 - Print, 313
- TiXmlDocument, 313
 - Accept, 314
 - ClearError, 314
 - Clone, 314
 - Error, 315
 - ErrorId, 315
 - ErrorRow, 315
 - LoadFile, 315
 - Parse, 315
 - Print, 315
 - RootElement, 316
 - SetTabSize, 316
- TiXmlElement, 316
 - Accept, 318
 - Attribute, 318
 - GetText, 318
 - Print, 318
 - QueryBoolAttribute, 319
 - QueryIntAttribute, 319
 - RemoveAttribute, 319
 - SetAttribute, 319
 - SetDoubleAttribute, 319
- TiXmlHandle, 320
 - Child, 321
 - ChildElement, 322
 - Element, 322
 - Node, 322
 - Text, 322
 - ToElement, 322
 - ToNode, 322
 - ToText, 322
 - ToUnknown, 322
 - Unknown, 323
- TiXmlNode, 323
 - Accept, 326

- Clone, [326](#)
- FirstChild, [326](#)
- GetDocument, [326](#)
- InsertAfterChild, [326](#)
- InsertBeforeChild, [327](#)
- InsertEndChild, [327](#)
- IterateChildren, [327](#)
- LinkEndChild, [327](#)
- NextSiblingElement, [327](#)
- NodeType, [326](#)
- ReplaceChild, [328](#)
- SetValue, [328](#)
- Type, [328](#)
- Value, [328](#)
- TiXmlOutStream, [328](#)
- TiXmlParsingData, [329](#)
- TiXmlPrinter, [329](#)
 - SetIndent, [331](#)
 - SetLineBreak, [331](#)
 - SetStreamPrinting, [331](#)
- TiXmlString, [331](#)
- TiXmlText, [332](#)
 - Accept, [333](#)
 - Print, [333](#)
 - TiXmlText, [333](#)
- TiXmlUnknown, [333](#)
 - Accept, [334](#)
 - Print, [334](#)
- TiXmlVisitor, [335](#)
- timeBase
 - SST::Config, [56](#)
- timeVortex
 - SST::Config, [56](#)
- toDouble
 - SST::decimal_fixedpoint, [74](#)
- ToElement
 - TiXmlHandle, [322](#)
- toLong
 - SST::decimal_fixedpoint, [74](#)
- ToNode
 - TiXmlHandle, [322](#)
- toString
 - SST::UnitAlgebra, [343](#)
 - SST::Units, [345](#)
 - SST::decimal_fixedpoint, [74](#)
- toStringBestSI
 - SST::UnitAlgebra, [343](#)
- ToText
 - TiXmlHandle, [322](#)
- ToUnknown
 - TiXmlHandle, [322](#)
- toUnsignedLong
 - SST::decimal_fixedpoint, [74](#)
- TraceType
 - SST::Interfaces::SimpleNetwork::Request, [182](#)
- TxBegin
 - SST::Interfaces::SimpleMem::Request, [177](#)
- TxEnd
 - SST::Interfaces::SimpleMem::Request, [177](#)
- Type
 - TiXmlNode, [328](#)
- type
 - SST::ConfigComponent, [59](#)
- UNKNOWN
 - SST::Simulation, [215](#)
- ultimate_parent
 - SST::ConfigComponent, [59](#)
- uninitQueue
 - SST::Link, [125](#)
- UninitializedQueue
 - SST::UninitializedQueue, [337](#)
- UnitAlgebra
 - SST::UnitAlgebra, [341](#)
- Units
 - SST::Units, [344](#)
- units
 - SST::ElementInfoStatistic, [79](#)
- Unknown
 - TiXmlHandle, [323](#)
- unpack_buffer
 - SST::Core::Serialization::pvt::ser_unpacker, [187](#)
- unregisterClock
 - SST::BaseComponent, [39](#)
 - SST::Simulation, [220](#)
- unregisterExit
 - SST::Component, [48](#)
- unregisterHandler
 - SST::Clock, [46](#)
- untimedQueue
 - SST::Link, [125](#)
- unusedPair
 - SST::RNG::SSTGaussianDistribution, [234](#)
- usePair
 - SST::RNG::SSTGaussianDistribution, [234](#)
- utf8ByteTable
 - TiXmlBase, [309](#)
- validEvents
 - SST::ElementInfoPort, [77](#)
 - SST::ElementInfoPort2, [78](#)
- Value
 - TiXmlNode, [328](#)
- verbose
 - SST::Config, [56](#)
 - SST::Output, [152](#)
- verbosePrefix
 - SST::Output, [153](#)
- verifyParam
 - SST::Params, [161](#)
- verifyStatsAndComponents
 - SST::ConfigStatGroup, [66](#)
- virtualAddr
 - SST::Interfaces::SimpleMem::Request, [181](#)
- vn
 - SST::Interfaces::SimpleNetwork::Request, [184](#)

wait
 SST::Core::ThreadSafe::Barrier, [29](#)
wasLoadedWithLegacyAPI
 SST::BaseComponent, [39](#)
weight
 SST::ConfigComponent, [59](#)
world_size
 SST::Config, [56](#)
Write
 SST::Interfaces::SimpleMem::Request, [177](#)
writeMessage
 SST::Core::Interprocess::IPCTunnel, [117](#)
WriteResp
 SST::Interfaces::SimpleMem::Request, [177](#)

XORShiftRNG
 SST::RNG::XORShiftRNG, [346](#)