

# ADD - ACCELERATOR DESIGN AND DEPLOY DOCUMENTATION

1.0

Thu Jun 29 2017 21:31:17



# Contents

<b>1</b>	<b>Hierarchical Index</b>	<b>1</b>
1.1	Class Hierarchy . . . . .	1
<b>2</b>	<b>Class Index</b>	<b>3</b>
2.1	Class List . . . . .	3
<b>3</b>	<b>Class Documentation</b>	<b>5</b>
3.1	add.dataflow.sync.ABS Class Reference . . . . .	5
3.1.1	Detailed Description . . . . .	5
3.1.2	Constructor & Destructor Documentation . . . . .	5
3.1.2.1	ABS . . . . .	6
3.1.3	Member Function Documentation . . . . .	6
3.1.3.1	Compute . . . . .	6
3.2	add.dataflow.sync.ACC_ADD Class Reference . . . . .	6
3.2.1	Detailed Description . . . . .	6
3.2.2	Constructor & Destructor Documentation . . . . .	7
3.2.2.1	ACC_ADD . . . . .	7
3.3	add.dataflow.sync.ACC_MAX Class Reference . . . . .	7
3.3.1	Detailed Description . . . . .	7
3.3.2	Constructor & Destructor Documentation . . . . .	8
3.3.2.1	ACC_MAX . . . . .	8
3.3.3	Member Function Documentation . . . . .	8
3.3.3.1	Accumulate . . . . .	8
3.3.3.2	Reseted . . . . .	8
3.4	add.dataflow.sync.ACC_MIN Class Reference . . . . .	8
3.4.1	Detailed Description . . . . .	9
3.4.2	Constructor & Destructor Documentation . . . . .	9
3.4.2.1	ACC_MIN . . . . .	9
3.4.3	Member Function Documentation . . . . .	9
3.4.3.1	Accumulate . . . . .	9
3.4.3.2	Reseted . . . . .	9
3.5	add.dataflow.sync.ACC_MUL Class Reference . . . . .	9

3.5.1	Detailed Description	10
3.5.2	Constructor & Destructor Documentation	10
3.5.2.1	ACC_MUL	10
3.5.3	Member Function Documentation	10
3.5.3.1	Accumulate	10
3.5.3.2	Reseted	10
3.6	add.dataflow.sync.ADD Class Reference	10
3.6.1	Detailed Description	11
3.6.2	Constructor & Destructor Documentation	11
3.6.2.1	ADD	11
3.6.3	Member Function Documentation	11
3.6.3.1	Compute	11
3.7	add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_FPGA Class Reference	12
3.7.1	Detailed Description	12
3.8	add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_HADES Class Reference	12
3.8.1	Detailed Description	12
3.9	add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_HADES_WITH_GENERATED_HDS Class Reference	13
3.9.1	Detailed Description	13
3.10	add.dataflow.sync.ADDI Class Reference	13
3.10.1	Detailed Description	13
3.10.2	Constructor & Destructor Documentation	14
3.10.2.1	ADDI	14
3.10.3	Member Function Documentation	14
3.10.3.1	Compute	14
3.11	add.dataflow.sync.AND Class Reference	14
3.11.1	Detailed Description	15
3.11.2	Constructor & Destructor Documentation	15
3.11.2.1	AND	15
3.11.3	Member Function Documentation	15
3.11.3.1	Compute	15
3.12	add.dataflow.sync.ANDI Class Reference	15
3.12.1	Detailed Description	16
3.12.2	Constructor & Destructor Documentation	16
3.12.2.1	ANDI	16
3.12.3	Member Function Documentation	16
3.12.3.1	Compute	16
3.13	add.dataflow.sync.BEQ Class Reference	17
3.13.1	Detailed Description	17
3.13.2	Constructor & Destructor Documentation	17

3.13.2.1	BEQ	17
3.13.3	Member Function Documentation	18
3.13.3.1	Compute	18
3.14	add.dataflow.sync.BEQI Class Reference	18
3.14.1	Detailed Description	18
3.14.2	Constructor & Destructor Documentation	19
3.14.2.1	BEQI	19
3.14.3	Member Function Documentation	19
3.14.3.1	Compute	19
3.15	add.dataflow.sync.BNE Class Reference	19
3.15.1	Detailed Description	19
3.15.2	Constructor & Destructor Documentation	20
3.15.2.1	BNE	20
3.15.3	Member Function Documentation	20
3.15.3.1	Compute	20
3.16	add.dataflow.sync.BNEI Class Reference	20
3.16.1	Detailed Description	21
3.16.2	Constructor & Destructor Documentation	21
3.16.2.1	BNEI	21
3.16.3	Member Function Documentation	21
3.16.3.1	Compute	21
3.17	add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_FPGA Class Reference	21
3.17.1	Detailed Description	22
3.18	add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_HADES Class Reference	22
3.18.1	Detailed Description	22
3.19	add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_HADES_WITH_GENERATED_HDS Class Reference	22
3.19.1	Detailed Description	23
3.20	add.util.CONF_READER Class Reference	23
3.20.1	Detailed Description	23
3.20.2	Member Function Documentation	23
3.20.2.1	ReadConfig	23
3.21	add.dataflow.DATAFLOW_SYNC_BASE_SIMUL Class Reference	24
3.21.1	Detailed Description	24
3.21.2	Member Function Documentation	24
3.21.2.1	exec_FPGA	24
3.21.2.2	exec_HADES	24
3.21.2.3	start_FPGA	24
3.21.2.4	start_FPGA	25
3.21.2.5	start_HADES	25

3.21.2.6	start_HADES	25
3.22	add.dataflow.sync.DIV Class Reference	25
3.22.1	Detailed Description	26
3.22.2	Constructor & Destructor Documentation	26
3.22.2.1	DIV	26
3.22.3	Member Function Documentation	26
3.22.3.1	Compute	26
3.23	add.dataflow.sync.DIVI Class Reference	27
3.23.1	Detailed Description	27
3.23.2	Constructor & Destructor Documentation	27
3.23.2.1	DIVI	27
3.23.3	Member Function Documentation	27
3.23.3.1	Compute	27
3.24	add.examples.dataflow_sync.FIR_4_DATAFLOW_FPGA Class Reference	28
3.24.1	Detailed Description	28
3.25	add.examples.dataflow_sync.FIR_4_DATAFLOW_HADES Class Reference	28
3.25.1	Detailed Description	28
3.26	add.examples.dataflow_sync.FIR_4_DATAFLOW_HADES_WITH_GENERATED_HDS Class Reference	29
3.26.1	Detailed Description	29
3.27	add.dataflow.sync.Generic_ACC Class Reference	29
3.27.1	Detailed Description	30
3.27.2	Constructor & Destructor Documentation	30
3.27.2.1	Generic_ACC	30
3.27.3	Member Function Documentation	30
3.27.3.1	Accumulate	30
3.27.3.2	evaluate	30
3.27.3.3	Reseted	31
3.28	add.dataflow.sync.Generic_Bin Class Reference	31
3.28.1	Detailed Description	32
3.28.2	Constructor & Destructor Documentation	32
3.28.2.1	Generic_Bin	32
3.28.3	Member Function Documentation	32
3.28.3.1	Compute	32
3.28.3.2	constructDynamicSymbol	33
3.28.3.3	constructPorts	33
3.28.3.4	evaluate	33
3.28.3.5	initialize	33
3.28.3.6	needsDynamicSymbol	33
3.28.3.7	NotCompute	33

3.28.3.8	Reseted	33
3.28.3.9	Set_Name	33
3.28.3.10	setString	33
3.28.3.11	setSymbol	34
3.28.3.12	Tick_Down	34
3.28.3.13	Tick_Up	34
3.28.3.14	write	34
3.29	add.dataflow.sync.Generic_Comp Class Reference	34
3.29.1	Detailed Description	35
3.29.2	Constructor & Destructor Documentation	35
3.29.2.1	Generic_Comp	35
3.29.3	Member Function Documentation	35
3.29.3.1	Compute	35
3.29.3.2	constructDynamicSymbol	35
3.29.3.3	constructPorts	36
3.29.3.4	evaluate	36
3.29.3.5	initialize	36
3.29.3.6	needsDynamicSymbol	36
3.29.3.7	Reseted	36
3.29.3.8	Set_Name	36
3.29.3.9	setString	36
3.29.3.10	setSymbol	36
3.29.3.11	Tick_Down	37
3.29.3.12	Tick_Up	37
3.29.3.13	write	37
3.30	add.dataflow.sync.Generic_Compl Class Reference	37
3.30.1	Detailed Description	38
3.30.2	Constructor & Destructor Documentation	38
3.30.2.1	Generic_Compl	38
3.30.3	Member Function Documentation	38
3.30.3.1	Compute	38
3.30.3.2	constructDynamicSymbol	38
3.30.3.3	constructPorts	38
3.30.3.4	evaluate	39
3.30.3.5	initialize	39
3.30.3.6	mousePressed	39
3.30.3.7	needsDynamicSymbol	39
3.30.3.8	Reseted	39
3.30.3.9	Set_Name	39
3.30.3.10	setString	39

3.30.3.11	setSymbol	40
3.30.3.12	Tick_Down	40
3.30.3.13	Tick_Up	40
3.30.3.14	write	40
3.31	add.dataflow.sync.Generic_I Class Reference	40
3.31.1	Detailed Description	41
3.31.2	Constructor & Destructor Documentation	42
3.31.2.1	Generic_I	42
3.31.3	Member Function Documentation	42
3.31.3.1	Compute	42
3.31.3.2	initialize	42
3.31.3.3	mousePressed	42
3.31.3.4	NotCompute	42
3.31.3.5	Reseted	43
3.31.3.6	write	43
3.32	add.dataflow.sync.Generic_IN Class Reference	43
3.32.1	Detailed Description	44
3.32.2	Constructor & Destructor Documentation	44
3.32.2.1	Generic_IN	44
3.32.2.2	Generic_IN	44
3.32.3	Member Function Documentation	44
3.32.3.1	constructDynamicSymbol	44
3.32.3.2	constructPorts	45
3.32.3.3	constructStandardValues	45
3.32.3.4	evaluate	45
3.32.3.5	getDelay	45
3.32.3.6	initialize	45
3.32.3.7	needsDynamicSymbol	45
3.32.3.8	Set_Name	45
3.32.3.9	setDelay	45
3.32.3.10	setDelay	46
3.32.3.11	setVectorIn	46
3.32.3.12	updateSymbol	46
3.32.3.13	wakeup	46
3.32.3.14	write	46
3.33	add.dataflow.sync.Generic_OUT Class Reference	46
3.33.1	Detailed Description	47
3.33.2	Constructor & Destructor Documentation	48
3.33.2.1	Generic_OUT	48
3.33.2.2	Generic_OUT	48



3.33.3	Member Function Documentation	48
3.33.3.1	constructDynamicSymbol	48
3.33.3.2	constructPorts	48
3.33.3.3	constructStandardValues	48
3.33.3.4	evaluate	48
3.33.3.5	getDelay	48
3.33.3.6	getDoneSignal	48
3.33.3.7	getVectorOut	49
3.33.3.8	initialize	49
3.33.3.9	needsDynamicSymbol	49
3.33.3.10	Set_Name	49
3.33.3.11	setDelay	49
3.33.3.12	setDelay	49
3.33.3.13	setVetor	49
3.33.3.14	updateSymbol	49
3.33.3.15	wakeup	50
3.33.3.16	write	50
3.34	add.dataflow.sync.Generic_Un Class Reference	50
3.34.1	Detailed Description	51
3.34.2	Constructor & Destructor Documentation	51
3.34.2.1	Generic_Un	51
3.34.3	Member Function Documentation	51
3.34.3.1	Compute	51
3.34.3.2	constructDynamicSymbol	51
3.34.3.3	constructPorts	52
3.34.3.4	evaluate	52
3.34.3.5	initialize	52
3.34.3.6	needsDynamicSymbol	52
3.34.3.7	NotCompute	52
3.34.3.8	Reseted	52
3.34.3.9	Set_Name	52
3.34.3.10	setString	52
3.34.3.11	setSymbol	53
3.34.3.12	Tick_Down	53
3.34.3.13	Tick_Up	53
3.34.3.14	write	53
3.35	add.dataflow.sync.IN_1 Class Reference	53
3.35.1	Detailed Description	53
3.35.2	Constructor & Destructor Documentation	54
3.35.2.1	IN_1	54

3.36	<a href="#">add.dataflow.sync.IN_16 Class Reference</a>	54
3.36.1	<a href="#">Detailed Description</a>	54
3.36.2	<a href="#">Constructor &amp; Destructor Documentation</a>	55
3.36.2.1	<a href="#">IN_16</a>	55
3.37	<a href="#">add.dataflow.sync.IN_2 Class Reference</a>	55
3.37.1	<a href="#">Detailed Description</a>	55
3.37.2	<a href="#">Constructor &amp; Destructor Documentation</a>	55
3.37.2.1	<a href="#">IN_2</a>	55
3.38	<a href="#">add.dataflow.sync.IN_32 Class Reference</a>	56
3.38.1	<a href="#">Detailed Description</a>	56
3.38.2	<a href="#">Constructor &amp; Destructor Documentation</a>	56
3.38.2.1	<a href="#">IN_32</a>	56
3.39	<a href="#">add.dataflow.sync.IN_4 Class Reference</a>	56
3.39.1	<a href="#">Detailed Description</a>	57
3.39.2	<a href="#">Constructor &amp; Destructor Documentation</a>	57
3.39.2.1	<a href="#">IN_4</a>	57
3.40	<a href="#">add.dataflow.sync.IN_8 Class Reference</a>	57
3.40.1	<a href="#">Detailed Description</a>	57
3.40.2	<a href="#">Constructor &amp; Destructor Documentation</a>	58
3.40.2.1	<a href="#">IN_8</a>	58
3.41	<a href="#">add.dataflow.sync.MAX Class Reference</a>	58
3.41.1	<a href="#">Detailed Description</a>	58
3.41.2	<a href="#">Constructor &amp; Destructor Documentation</a>	59
3.41.2.1	<a href="#">MAX</a>	59
3.41.3	<a href="#">Member Function Documentation</a>	59
3.41.3.1	<a href="#">Compute</a>	59
3.42	<a href="#">add.dataflow.sync.MERGE Class Reference</a>	59
3.42.1	<a href="#">Detailed Description</a>	59
3.42.2	<a href="#">Constructor &amp; Destructor Documentation</a>	60
3.42.2.1	<a href="#">MERGE</a>	60
3.42.3	<a href="#">Member Function Documentation</a>	60
3.42.3.1	<a href="#">evaluate</a>	60
3.43	<a href="#">add.dataflow.sync.MIN Class Reference</a>	60
3.43.1	<a href="#">Detailed Description</a>	60
3.43.2	<a href="#">Constructor &amp; Destructor Documentation</a>	61
3.43.2.1	<a href="#">MIN</a>	61
3.43.3	<a href="#">Member Function Documentation</a>	61
3.43.3.1	<a href="#">Compute</a>	61
3.44	<a href="#">add.dataflow.sync.MOD Class Reference</a>	61
3.44.1	<a href="#">Detailed Description</a>	62

3.44.2	Constructor & Destructor Documentation	62
3.44.2.1	MOD	62
3.44.3	Member Function Documentation	62
3.44.3.1	Compute	62
3.45	add.dataflow.sync.MODI Class Reference	62
3.45.1	Detailed Description	63
3.45.2	Constructor & Destructor Documentation	63
3.45.2.1	MODI	63
3.45.3	Member Function Documentation	63
3.45.3.1	Compute	63
3.46	add.dataflow.sync.MUL Class Reference	64
3.46.1	Detailed Description	64
3.46.2	Constructor & Destructor Documentation	64
3.46.2.1	MUL	64
3.46.3	Member Function Documentation	65
3.46.3.1	Compute	65
3.47	add.dataflow.sync.MULI Class Reference	66
3.47.1	Detailed Description	66
3.47.2	Constructor & Destructor Documentation	66
3.47.2.1	MULI	66
3.47.3	Member Function Documentation	67
3.47.3.1	Compute	67
3.48	add.dataflow.sync.NOT Class Reference	67
3.48.1	Detailed Description	67
3.48.2	Constructor & Destructor Documentation	68
3.48.2.1	NOT	68
3.48.3	Member Function Documentation	68
3.48.3.1	Compute	68
3.49	add.dataflow.sync.OR Class Reference	68
3.49.1	Detailed Description	68
3.49.2	Constructor & Destructor Documentation	69
3.49.2.1	OR	69
3.49.3	Member Function Documentation	69
3.49.3.1	Compute	69
3.50	add.dataflow.sync.ORI Class Reference	69
3.50.1	Detailed Description	70
3.50.2	Constructor & Destructor Documentation	70
3.50.2.1	ORI	70
3.50.3	Member Function Documentation	70
3.50.3.1	Compute	70

3.51	<a href="#">add.dataflow.sync.OUT_1 Class Reference</a>	70
3.51.1	<a href="#">Detailed Description</a>	71
3.51.2	<a href="#">Constructor &amp; Destructor Documentation</a>	71
3.51.2.1	<a href="#">OUT_1</a>	71
3.52	<a href="#">add.dataflow.sync.OUT_16 Class Reference</a>	71
3.52.1	<a href="#">Detailed Description</a>	71
3.52.2	<a href="#">Constructor &amp; Destructor Documentation</a>	72
3.52.2.1	<a href="#">OUT_16</a>	72
3.53	<a href="#">add.dataflow.sync.OUT_2 Class Reference</a>	72
3.53.1	<a href="#">Detailed Description</a>	72
3.53.2	<a href="#">Constructor &amp; Destructor Documentation</a>	73
3.53.2.1	<a href="#">OUT_2</a>	73
3.54	<a href="#">add.dataflow.sync.OUT_32 Class Reference</a>	73
3.54.1	<a href="#">Detailed Description</a>	73
3.54.2	<a href="#">Constructor &amp; Destructor Documentation</a>	73
3.54.2.1	<a href="#">OUT_32</a>	73
3.55	<a href="#">add.dataflow.sync.OUT_4 Class Reference</a>	74
3.55.1	<a href="#">Detailed Description</a>	74
3.55.2	<a href="#">Constructor &amp; Destructor Documentation</a>	74
3.55.2.1	<a href="#">OUT_4</a>	74
3.56	<a href="#">add.dataflow.sync.OUT_8 Class Reference</a>	74
3.56.1	<a href="#">Detailed Description</a>	75
3.56.2	<a href="#">Constructor &amp; Destructor Documentation</a>	75
3.56.2.1	<a href="#">OUT_8</a>	75
3.57	<a href="#">add.dataflow.sync.REG Class Reference</a>	75
3.57.1	<a href="#">Detailed Description</a>	75
3.57.2	<a href="#">Constructor &amp; Destructor Documentation</a>	76
3.57.2.1	<a href="#">REG</a>	76
3.58	<a href="#">add.dataflow.sync.SHL Class Reference</a>	76
3.58.1	<a href="#">Detailed Description</a>	76
3.58.2	<a href="#">Constructor &amp; Destructor Documentation</a>	77
3.58.2.1	<a href="#">SHL</a>	77
3.58.3	<a href="#">Member Function Documentation</a>	77
3.58.3.1	<a href="#">Compute</a>	77
3.59	<a href="#">add.dataflow.sync.SHLI Class Reference</a>	77
3.59.1	<a href="#">Detailed Description</a>	78
3.59.2	<a href="#">Constructor &amp; Destructor Documentation</a>	78
3.59.2.1	<a href="#">SHLI</a>	78
3.59.3	<a href="#">Member Function Documentation</a>	78
3.59.3.1	<a href="#">Compute</a>	78

3.60	add.dataflow.sync.SHR Class Reference . . . . .	78
3.60.1	Detailed Description . . . . .	79
3.60.2	Constructor & Destructor Documentation . . . . .	79
3.60.2.1	SHR . . . . .	79
3.60.3	Member Function Documentation . . . . .	79
3.60.3.1	Compute . . . . .	79
3.61	add.dataflow.sync.SHRI Class Reference . . . . .	79
3.61.1	Detailed Description . . . . .	80
3.61.2	Constructor & Destructor Documentation . . . . .	80
3.61.2.1	SHRI . . . . .	80
3.61.3	Member Function Documentation . . . . .	80
3.61.3.1	Compute . . . . .	80
3.62	add.dataflow.sync.SLT Class Reference . . . . .	81
3.62.1	Detailed Description . . . . .	81
3.62.2	Constructor & Destructor Documentation . . . . .	81
3.62.2.1	SLT . . . . .	81
3.62.3	Member Function Documentation . . . . .	82
3.62.3.1	Compute . . . . .	82
3.63	add.dataflow.sync.SLTI Class Reference . . . . .	83
3.63.1	Detailed Description . . . . .	83
3.63.2	Constructor & Destructor Documentation . . . . .	84
3.63.2.1	SLTI . . . . .	84
3.63.3	Member Function Documentation . . . . .	84
3.63.3.1	Compute . . . . .	84
3.64	add.dataflow.sync.SUB Class Reference . . . . .	84
3.64.1	Detailed Description . . . . .	84
3.64.2	Constructor & Destructor Documentation . . . . .	85
3.64.2.1	SUB . . . . .	85
3.64.3	Member Function Documentation . . . . .	85
3.64.3.1	Compute . . . . .	85
3.65	add.dataflow.sync.SUBI Class Reference . . . . .	85
3.65.1	Detailed Description . . . . .	86
3.65.2	Constructor & Destructor Documentation . . . . .	86
3.65.2.1	SUBI . . . . .	86
3.65.3	Member Function Documentation . . . . .	86
3.65.3.1	Compute . . . . .	86
<b>Index</b>		<b>87</b>



# Chapter 1

## Hierarchical Index

### 1.1 Class Hierarchy

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

add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_FPGA . . . . .	12
add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_HADES . . . . .	12
add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_HADES_WITH_GENERATED_H- DS . . . . .	13
add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_FPGA . . . . .	21
add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_HADES . . . . .	22
add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_HADES_WITH_GENERATED_HDS . . . .	22
add.util.CONF_READER . . . . .	23
add.dataflow.DATAFLOW_SYNC_BASE_SIMUL . . . . .	24
add.examples.dataflow_sync.FIR_4_DATAFLOW_FPGA . . . . .	28
add.examples.dataflow_sync.FIR_4_DATAFLOW_HADES . . . . .	28
add.examples.dataflow_sync.FIR_4_DATAFLOW_HADES_WITH_GENERATED_HDS . . . . .	29
GenericRtlibObject	
add.dataflow.sync.Generic_Bin . . . . .	31
add.dataflow.sync.ADD . . . . .	10
add.dataflow.sync.AND . . . . .	14
add.dataflow.sync.DIV . . . . .	25
add.dataflow.sync.MAX . . . . .	58
add.dataflow.sync.MERGE . . . . .	59
add.dataflow.sync.MIN . . . . .	60
add.dataflow.sync.MOD . . . . .	61
add.dataflow.sync.MUL . . . . .	64
add.dataflow.sync.OR . . . . .	68
add.dataflow.sync.SHL . . . . .	76
add.dataflow.sync.SHR . . . . .	78
add.dataflow.sync.SLT . . . . .	81
add.dataflow.sync.SUB . . . . .	84
add.dataflow.sync.Generic_Comp . . . . .	34
add.dataflow.sync.BEQ . . . . .	17
add.dataflow.sync.BNE . . . . .	19
add.dataflow.sync.Generic_Compl . . . . .	37
add.dataflow.sync.BEQI . . . . .	18
add.dataflow.sync.BNEI . . . . .	20
add.dataflow.sync.Generic_Un . . . . .	50
add.dataflow.sync.ABS . . . . .	5
add.dataflow.sync.Generic_ACC . . . . .	29
add.dataflow.sync.ACC_ADD . . . . .	6
add.dataflow.sync.ACC_MAX . . . . .	7

add.dataflow.sync.ACC_MIN . . . . .	8
add.dataflow.sync.ACC_MUL . . . . .	9
add.dataflow.sync.Generic_I . . . . .	40
add.dataflow.sync.ADDI . . . . .	13
add.dataflow.sync.ANDI . . . . .	15
add.dataflow.sync.DIVI . . . . .	27
add.dataflow.sync.MODI . . . . .	62
add.dataflow.sync.MULI . . . . .	66
add.dataflow.sync.ORI . . . . .	69
add.dataflow.sync.SHLI . . . . .	77
add.dataflow.sync.SHRI . . . . .	79
add.dataflow.sync.SLTI . . . . .	83
add.dataflow.sync.SUBI . . . . .	85
add.dataflow.sync.NOT . . . . .	67
add.dataflow.sync.REG . . . . .	75
Serializable	
add.dataflow.sync.Generic_IN . . . . .	43
add.dataflow.sync.IN_1 . . . . .	53
add.dataflow.sync.IN_16 . . . . .	54
add.dataflow.sync.IN_2 . . . . .	55
add.dataflow.sync.IN_32 . . . . .	56
add.dataflow.sync.IN_4 . . . . .	56
add.dataflow.sync.IN_8 . . . . .	57
add.dataflow.sync.Generic_OUT . . . . .	46
add.dataflow.sync.OUT_1 . . . . .	70
add.dataflow.sync.OUT_16 . . . . .	71
add.dataflow.sync.OUT_2 . . . . .	72
add.dataflow.sync.OUT_32 . . . . .	73
add.dataflow.sync.OUT_4 . . . . .	74
add.dataflow.sync.OUT_8 . . . . .	74
SimObject	
add.dataflow.sync.Generic_IN . . . . .	43
add.dataflow.sync.Generic_OUT . . . . .	46
Simulatable	
add.dataflow.sync.Generic_IN . . . . .	43
add.dataflow.sync.Generic_OUT . . . . .	46
Wakeable	
add.dataflow.sync.Generic_IN . . . . .	43
add.dataflow.sync.Generic_OUT . . . . .	46



## Chapter 2

# Class Index

### 2.1 Class List

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

<a href="#">add.dataflow.sync.ABS</a>	5
<a href="#">add.dataflow.sync.ACC_ADD</a>	6
<a href="#">add.dataflow.sync.ACC_MAX</a>	7
<a href="#">add.dataflow.sync.ACC_MIN</a>	8
<a href="#">add.dataflow.sync.ACC_MUL</a>	9
<a href="#">add.dataflow.sync.ADD</a>	10
<a href="#">add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_FPGA</a>	12
<a href="#">add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_HADES</a>	12
<a href="#">add.examples.dataflow_sync.ADD_AB_HELLOWORLD_DATAFLOW_HADES_WITH_GENERATED_HDS</a>	13
<a href="#">add.dataflow.sync.ADDI</a>	13
<a href="#">add.dataflow.sync.AND</a>	14
<a href="#">add.dataflow.sync.ANDI</a>	15
<a href="#">add.dataflow.sync.BEQ</a>	17
<a href="#">add.dataflow.sync.BEQI</a>	18
<a href="#">add.dataflow.sync.BNE</a>	19
<a href="#">add.dataflow.sync.BNEI</a>	20
<a href="#">add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_FPGA</a>	21
<a href="#">add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_HADES</a>	22
<a href="#">add.examples.dataflow_sync.BRANCH_TEST_DATAFLOW_HADES_WITH_GENERATED_HDS</a>	22
<a href="#">add.util.CONF_READER</a>	23
<a href="#">add.dataflow.DATAFLOW_SYNC_BASE_SIMUL</a>	24
<a href="#">add.dataflow.sync.DIV</a>	25
<a href="#">add.dataflow.sync.DIVI</a>	27
<a href="#">add.examples.dataflow_sync.FIR_4_DATAFLOW_FPGA</a>	28
<a href="#">add.examples.dataflow_sync.FIR_4_DATAFLOW_HADES</a>	28
<a href="#">add.examples.dataflow_sync.FIR_4_DATAFLOW_HADES_WITH_GENERATED_HDS</a>	29
<a href="#">add.dataflow.sync.Generic_ACC</a>	29
<a href="#">add.dataflow.sync.Generic_Bin</a>	31
<a href="#">add.dataflow.sync.Generic_Comp</a>	34
<a href="#">add.dataflow.sync.Generic_Compl</a>	37
<a href="#">add.dataflow.sync.Generic_I</a>	40
<a href="#">add.dataflow.sync.Generic_IN</a>	43
<a href="#">add.dataflow.sync.Generic_OUT</a>	46
<a href="#">add.dataflow.sync.Generic_Un</a>	50
<a href="#">add.dataflow.sync.IN_1</a>	53
<a href="#">add.dataflow.sync.IN_16</a>	54
<a href="#">add.dataflow.sync.IN_2</a>	55

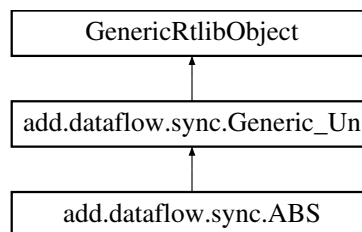
<a href="#">add.dataflow.sync.IN_32</a>	56
<a href="#">add.dataflow.sync.IN_4</a>	56
<a href="#">add.dataflow.sync.IN_8</a>	57
<a href="#">add.dataflow.sync.MAX</a>	58
<a href="#">add.dataflow.sync.MERGE</a>	59
<a href="#">add.dataflow.sync.MIN</a>	60
<a href="#">add.dataflow.sync.MOD</a>	61
<a href="#">add.dataflow.sync.MODI</a>	62
<a href="#">add.dataflow.sync.MUL</a>	64
<a href="#">add.dataflow.sync.MULI</a>	66
<a href="#">add.dataflow.sync.NOT</a>	67
<a href="#">add.dataflow.sync.OR</a>	68
<a href="#">add.dataflow.sync.ORI</a>	69
<a href="#">add.dataflow.sync.OUT_1</a>	70
<a href="#">add.dataflow.sync.OUT_16</a>	71
<a href="#">add.dataflow.sync.OUT_2</a>	72
<a href="#">add.dataflow.sync.OUT_32</a>	73
<a href="#">add.dataflow.sync.OUT_4</a>	74
<a href="#">add.dataflow.sync.OUT_8</a>	74
<a href="#">add.dataflow.sync.REG</a>	75
<a href="#">add.dataflow.sync.SHL</a>	76
<a href="#">add.dataflow.sync.SHLI</a>	77
<a href="#">add.dataflow.sync.SHR</a>	78
<a href="#">add.dataflow.sync.SHRI</a>	79
<a href="#">add.dataflow.sync.SLT</a>	81
<a href="#">add.dataflow.sync.SLTI</a>	83
<a href="#">add.dataflow.sync.SUB</a>	84
<a href="#">add.dataflow.sync.SUBI</a>	85

## Chapter 3

# Class Documentation

### 3.1 add.dataflow.sync.ABS Class Reference

Inheritance diagram for add.dataflow.sync.ABS:



#### Public Member Functions

- [ABS](#) ()
- int [Compute](#) (int data)

#### Additional Inherited Members

#### 3.1.1 Detailed Description

[ABS](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for delivering the absolute value of the input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

#### 3.1.2 Constructor & Destructor Documentation

### 3.1.2.1 add.dataflow.sync.ABS.ABS ( )

Object Constructor.

## 3.1.3 Member Function Documentation

### 3.1.3.1 int add.dataflow.sync.ABS.Compute ( int *data* )

Method responsible for the component computation.

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

#### Returns

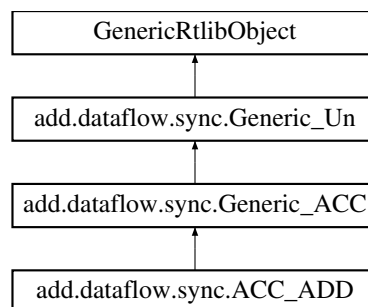
- Returns the result of the computation. In this case, returns the absolute value of the parameter.

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

- add/dataflow/sync/ABS.java

## 3.2 add.dataflow.sync.ACC\_ADD Class Reference

Inheritance diagram for add.dataflow.sync.ACC\_ADD:



### Public Member Functions

- [ACC\\_ADD](#) ()

### Additional Inherited Members

#### 3.2.1 Detailed Description

[ACC\\_ADD](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an adder accumulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

Version

1.0

### 3.2.2 Constructor & Destructor Documentation

#### 3.2.2.1 add.dataflow.sync.ACC\_ADD.ACC\_ADD ( )

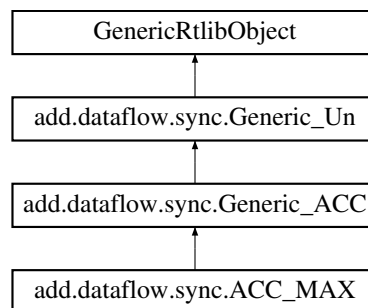
Object Constructor.

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

- add/dataflow/sync/ACC\_ADD.java

## 3.3 add.dataflow.sync.ACC\_MAX Class Reference

Inheritance diagram for add.dataflow.sync.ACC\_MAX:



### Public Member Functions

- [ACC\\_MAX](#) ()
- void [Reseted](#) ()

### Protected Member Functions

- void [Accumulate](#) (int data)

### Additional Inherited Members

#### 3.3.1 Detailed Description

[ACC\\_MAX](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements a store for the highest input value.

Universidade Federal de Viçosa - MG - Brasil.

Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

Version

1.0

### 3.3.2 Constructor & Destructor Documentation

#### 3.3.2.1 `add.dataflow.sync.ACC_MAX.ACC_MAX ( )`

Object Constructor.

### 3.3.3 Member Function Documentation

#### 3.3.3.1 `void add.dataflow.sync.ACC_MAX.Accumulate ( int data )` `[protected]`

Method that compares the parameter to the stored value. If the parameter is larger, it will override the stored value.

Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

#### 3.3.3.2 `void add.dataflow.sync.ACC_MAX.Reseted ( )`

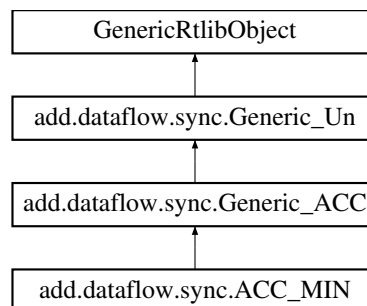
Method responsible for actions required when "Reset" occurs.

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

- `add/dataflow/sync/ACC_MAX.java`

## 3.4 `add.dataflow.sync.ACC_MIN` Class Reference

Inheritance diagram for `add.dataflow.sync.ACC_MIN`:



#### Public Member Functions

- [ACC\\_MIN](#) ()
- void [Reseted](#) ()

#### Protected Member Functions

- void [Accumulate](#) (int data)

#### Additional Inherited Members

### 3.4.1 Detailed Description

**ACC\_MIN** component for the UfV **ADD** synchronous data flow simulator.

The component implements a store for the lowest input value.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.4.2 Constructor & Destructor Documentation

#### 3.4.2.1 add.dataflow.sync.ACC\_MIN.ACC\_MIN ( )

Object Constructor.

### 3.4.3 Member Function Documentation

#### 3.4.3.1 void add.dataflow.sync.ACC\_MIN.Accumulate ( int *data* ) [protected]

Method that compares the parameter to the stored value. If the parameter is smaller, it will replace the stored value.

##### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

#### 3.4.3.2 void add.dataflow.sync.ACC\_MIN.Reseted ( )

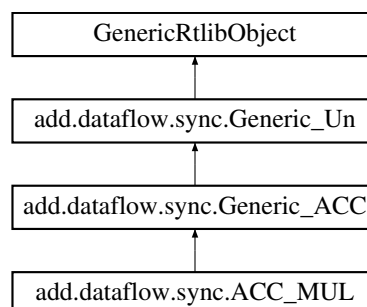
Method responsible for actions required when "Reset" occurs.

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

- add/dataflow/sync/ACC\_MIN.java

## 3.5 add.dataflow.sync.ACC\_MUL Class Reference

Inheritance diagram for add.dataflow.sync.ACC\_MUL:



## Public Member Functions

- [ACC\\_MUL](#) ()
- void [Reseted](#) ()

## Protected Member Functions

- void [Accumulate](#) (int data)

## Additional Inherited Members

### 3.5.1 Detailed Description

[ACC\\_MUL](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements a multiplication accumulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.5.2 Constructor & Destructor Documentation

#### 3.5.2.1 `add.dataflow.sync.ACC_MUL.ACC_MUL ( )`

Object Constructor.

### 3.5.3 Member Function Documentation

#### 3.5.3.1 `void add.dataflow.sync.ACC_MUL.Accumulate ( int data )` `[protected]`

Method that accumulates the input value with the stored. In this case, it multiplies the value stored by the input and stores it.

#### 3.5.3.2 `void add.dataflow.sync.ACC_MUL.Reseted ( )`

Method responsible for actions required when "Reset" occurs.

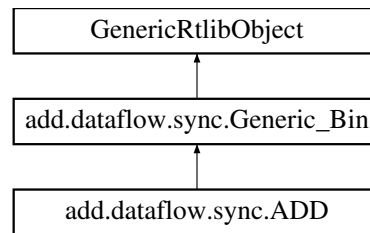
The documentation for this class was generated from the following file:

- `add/dataflow/sync/ACC_MUL.java`

## 3.6 `add.dataflow.sync.ADD` Class Reference

Inheritance diagram for `add.dataflow.sync.ADD`:





## Public Member Functions

- [ADD](#) ()
- int [Compute](#) (int data1, int data2)

## Additional Inherited Members

### 3.6.1 Detailed Description

[ADD](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for adding the inputs.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.6.2 Constructor & Destructor Documentation

#### 3.6.2.1 add.dataflow.sync.ADD.ADD ( )

Object Constructor.

### 3.6.3 Member Function Documentation

#### 3.6.3.1 int add.dataflow.sync.ADD.Compute ( int *data1*, int *data2* )

Method responsible for the component computation: in this case performs a addition of the parameters.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

#### Returns

- Returns the result of the computation. In this case the value of the addition of the parameters.

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

- add/dataflow/sync/ADD.java

### 3.7 add.examples.dataflow\_sync.ADD\_AB\_HELLOWORLD\_DATAFLOW\_FPGA Class Reference

#### Static Public Member Functions

- static void **main** (String argv[])

#### 3.7.1 Detailed Description

ADD\_AB data flow example in FPGA Board in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

\* 1.0

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

- add/examples/dataflow\_sync/ADD\_AB\_HELLOWORLD\_DATAFLOW\_FPGA.java

### 3.8 add.examples.dataflow\_sync.ADD\_AB\_HELLOWORLD\_DATAFLOW\_HADES Class Reference

#### Static Public Member Functions

- static void **main** (String argv[])

#### 3.8.1 Detailed Description

ADD\_AB data flow example in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

\* 1.0

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

- add/examples/dataflow\_sync/ADD\_AB\_HELLOWORLD\_DATAFLOW\_HADES.java

## 3.9 add.examples.dataflow\_sync.ADD\_AB\_HELLOWORLD\_DATAFLOW\_HADES\_WITH\_GENERATED\_HDS Class Reference

### Static Public Member Functions

- static void **main** (String argv[])

#### 3.9.1 Detailed Description

ADD\_AB data flow (Automatically generated) example in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

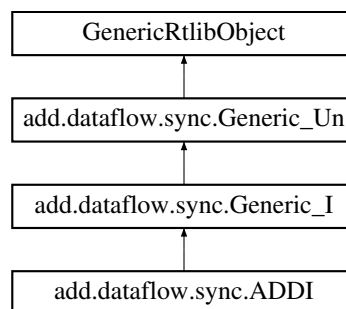
\* 1.0

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

- add/examples/dataflow\_sync/ADD\_AB\_HELLOWORLD\_DATAFLOW\_HADES\_WITH\_GENERATED\_HDS.java

## 3.10 add.dataflow.sync.ADDI Class Reference

Inheritance diagram for add.dataflow.sync.ADDI:



### Public Member Functions

- **ADDI** ()
- int **Compute** (int data)

### Additional Inherited Members

#### 3.10.1 Detailed Description

**ADDI** component for the UFV **ADD** synchronous data flow simulator.

The component is responsible for adding the input by a (Immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicos@gmail.com](mailto:cacauvicos@gmail.com)

**Version**

1.0

**3.10.2 Constructor & Destructor Documentation****3.10.2.1 add.dataflow.sync.ADDI.ADDI ( )**

Object Constructor.

**3.10.3 Member Function Documentation****3.10.3.1 int add.dataflow.sync.ADDI.Compute ( int *data* )**

Method responsible for the component computation: in this case performs a addition of the parameter by an (immediate) constant.

**Parameters**

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

**Returns**

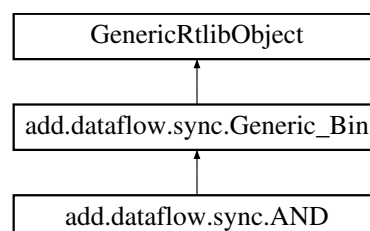
- Returns the result of the computation. In this case the value of the addition of the parameter by the constant.

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

- add/dataflow/sync/ADDI.java

**3.11 add.dataflow.sync.AND Class Reference**

Inheritance diagram for add.dataflow.sync.AND:

**Public Member Functions**

- [AND](#) ()
- int [Compute](#) (int data1, int data2)

## Additional Inherited Members

### 3.11.1 Detailed Description

**AND** component for the UFV **ADD** synchronous data flow simulator.

The component is responsible for the logical operation "AND" between the input

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.11.2 Constructor & Destructor Documentation

#### 3.11.2.1 add.dataflow.sync.AND.AND ( )

Object Constructor.

### 3.11.3 Member Function Documentation

#### 3.11.3.1 int add.dataflow.sync.AND.Compute ( int *data1*, int *data2* )

Method responsible for the component computation: in this case it performs the logical operation "AND" between the parameters.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

#### Returns

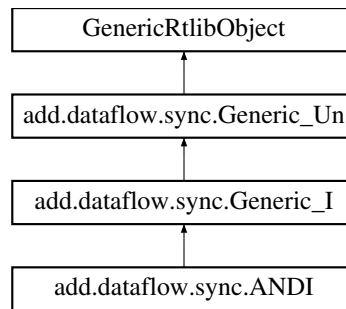
- Returns the result of the computation. In this case the result of the logical operation "AND" between the parameters.

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

- add/dataflow/sync/AND.java

## 3.12 add.dataflow.sync.ANDI Class Reference

Inheritance diagram for add.dataflow.sync.ANDI:



## Public Member Functions

- [ANDI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

### 3.12.1 Detailed Description

[ANDI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for the logical operation "AND" between the input and a constant (Immediate)

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.12.2 Constructor & Destructor Documentation

#### 3.12.2.1 `add.dataflow.sync.ANDI.ANDI ( )`

Object Constructor.

### 3.12.3 Member Function Documentation

#### 3.12.3.1 `int add.dataflow.sync.ANDI.Compute ( int data )`

Method responsible for the component computation: in this case it performs the logical operation "AND" between the parameter and the (immediate) constant.

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

**Returns**

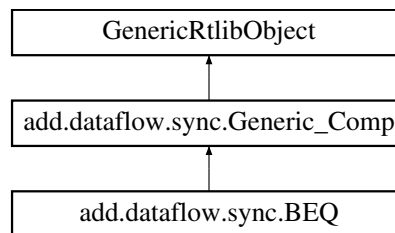
- Returns the result of the computation. In this case the result of the logical operation "AND" between the parameter and the constant.

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

- add/dataflow/sync/ANDI.java

### 3.13 add.dataflow.sync.BEQ Class Reference

Inheritance diagram for add.dataflow.sync.BEQ:

**Public Member Functions**

- [BEQ](#) ()
- int [Compute](#) (int data1, int data2)

**Additional Inherited Members**

#### 3.13.1 Detailed Description

[BEQ](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for comparing equality between the input. Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

**Version**

1.0

#### 3.13.2 Constructor & Destructor Documentation

##### 3.13.2.1 add.dataflow.sync.BEQ.BEQ ( )

Object Constructor.

### 3.13.3 Member Function Documentation

#### 3.13.3.1 `int add.dataflow.sync.BEQ.Compute ( int data1, int data2 )`

Method responsible for component computing: in this case performs a comparison of equality between the input. Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

##### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

##### Returns

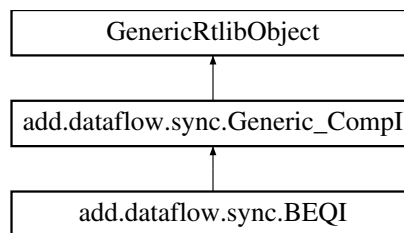
- Returns the result of the computation. In this case "1" if the parameters are equal or "0" if they are different.

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

- `add/dataflow/sync/BEQ.java`

## 3.14 `add.dataflow.sync.BEQI` Class Reference

Inheritance diagram for `add.dataflow.sync.BEQI`:



### Public Member Functions

- [BEQI](#) ()
- `int` [Compute](#) (int data)

### Additional Inherited Members

#### 3.14.1 Detailed Description

[BEQI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for comparing equality between the input and a constant (Immediate). Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

Universidade Federal de Viçosa - MG - Brasil.

##### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)



## Version

1.0

## 3.14.2 Constructor &amp; Destructor Documentation

## 3.14.2.1 add.dataflow.sync.BEQI.BEQI ( )

Object Constructor.

## 3.14.3 Member Function Documentation

3.14.3.1 int add.dataflow.sync.BEQI.Compute ( int *data* )

Method responsible for component computing: in this case performs a comparison of equality between the input and a constant. Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

## Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

## Returns

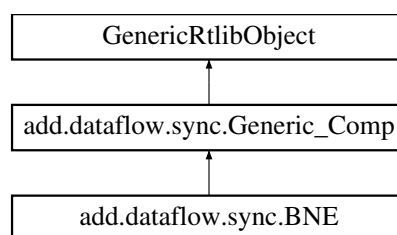
- Returns the result of the computation. In this case "1" if the parameter is equal to the constraint or "0" if they are different.

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

- add/dataflow/sync/BEQI.java

## 3.15 add.dataflow.sync.BNE Class Reference

Inheritance diagram for add.dataflow.sync.BNE:



## Public Member Functions

- [BNE](#) ()
- int [Compute](#) (int data1, int data2)

## Additional Inherited Members

## 3.15.1 Detailed Description

[BNE](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for comparing inequality between the input. Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.15.2 Constructor & Destructor Documentation

#### 3.15.2.1 `add.dataflow.sync.BNE.BNE ( )`

Object Constructor.

### 3.15.3 Member Function Documentation

#### 3.15.3.1 `int add.dataflow.sync.BNE.Compute ( int data1, int data2 )`

Method responsible for component computing: in this case performs a comparison of inequality between the input. Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

##### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

##### Returns

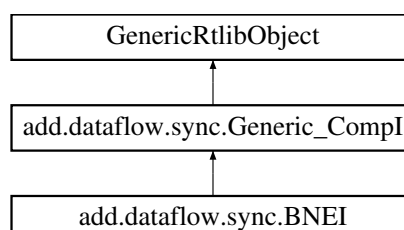
- Returns the result of the computation. In this case "1" if the parameters are different or "0" if they are equal.

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

- `add/dataflow/sync/BNE.java`

## 3.16 `add.dataflow.sync.BNEI` Class Reference

Inheritance diagram for `add.dataflow.sync.BNEI`:



## Public Member Functions

- [BNEI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

### 3.16.1 Detailed Description

[BEQI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for comparing inequality between the input and a constant (Immediate). Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.16.2 Constructor & Destructor Documentation

#### 3.16.2.1 add.dataflow.sync.BNEI.BNEI ( )

Object Constructor.

### 3.16.3 Member Function Documentation

#### 3.16.3.1 int add.dataflow.sync.BNEI.Compute ( int *data* )

Method responsible for component computing: in this case performs a comparison of inequality between the input and a constant. Depending on the result of the comparison, the "IF" output or the "ELSE" output will receive the value "1" while the other will receive the value "0".

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

#### Returns

- Returns the result of the computation. In this case "0" if the parameter is equal to the constraint or "1" if they are different.

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

- add/dataflow/sync/BNEI.java

## 3.17 add.examples.dataflow\_sync.BRANCH\_TEST\_DATAFLOW\_FPGA Class Reference

### Static Public Member Functions

- static void **main** (String argv[])

### 3.17.1 Detailed Description

Branch\_Test data flow example in FPGA Board in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

\* 1.0

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

- add/examples/dataflow\_sync/BRANCH\_TEST\_DATAFLOW\_FPGA.java

## 3.18 add.examples.dataflow\_sync.BRANCH\_TEST\_DATAFLOW\_HADES Class Reference

### Static Public Member Functions

- static void **main** (String argv[])

### 3.18.1 Detailed Description

Branch\_Test data flow example in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

\* 1.0

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

- add/examples/dataflow\_sync/BRANCH\_TEST\_DATAFLOW\_HADES.java

## 3.19 add.examples.dataflow\_sync.BRANCH\_TEST\_DATAFLOW\_HADES\_WITH\_GENERATED\_HDS Class Reference

### Static Public Member Functions

- static void **main** (String argv[])

### 3.19.1 Detailed Description

Branch\_Test data flow (Automatically generated) example in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

\* 1.0

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

- add/examples/dataflow\_sync/BRANCH\_TEST\_DATAFLOW\_HADES\_WITH\_GENERATED\_HDS.java

## 3.20 add.util.CONF\_READER Class Reference

### Public Member Functions

- [int\[\] ReadConfig](#) (File file)

### 3.20.1 Detailed Description

Class responsible for providing useful routines for the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.20.2 Member Function Documentation

#### 3.20.2.1 `int [] add.util.CONF_READER.ReadConfig ( File file )`

Method responsible for reading a configuration file and returning a vector with the values read.

#### Parameters

<i>file</i>	- File to read
-------------	----------------

#### Returns

- Returns a vector containing the values read in the file.

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

- add/util/CONF\_READER.java

### 3.21 add.dataflow.DATAFLOW\_SYNC\_BASE\_SIMUL Class Reference

#### Public Member Functions

- `int[] start_HADES (int[] CONF, String DESIGN)`
- `void start_HADES (String CONF, String DESIGN, String DESIRED_RETURN)`
- `int[] start_FPGA (int[] CONF, String QUARTUS)`
- `void start_FPGA (String CONF, String QUARTUS, String DESIRED_RETURN)`
- `int[] exec_HADES (int[] raw_data, String DESIGN)`
- `int[] exec_FPGA (int[] raw_data, String QUARTUS)`

#### 3.21.1 Detailed Description

Base class for executing algorithms in the UFV ADD synchronous data flow simulator or in the bundle with FPGA.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

#### 3.21.2 Member Function Documentation

3.21.2.1 `int[] add.dataflow.DATAFLOW_SYNC_BASE_SIMUL.exec_FPGA ( int[] raw_data, String QUARTUS )`

Execution in FPGA Boards

##### Parameters

<i>raw_data</i>	- Vector of data to be processed
<i>QUARTUS</i>	- Path to the quartus_stp application.

##### Returns

- Returns a vector with the processing results.

3.21.2.2 `int[] add.dataflow.DATAFLOW_SYNC_BASE_SIMUL.exec_HADES ( int[] raw_data, String DESIGN )`

Execution in HADES Simulator

##### Parameters

<i>raw_data</i>	- Vector of data to be processed
<i>DESIGN</i>	- Design to be used to run the simulator.

##### Returns

- Returns a vector with the processing results.

3.21.2.3 `int[] add.dataflow.DATAFLOW_SYNC_BASE_SIMUL.start_FPGA ( int[] CONF, String QUARTUS )`

Method responsible for running the algorithm on the FPGA board.

## Parameters

<i>CONF</i>	- Configuration vector and data to be executed.
<i>QUARTUS</i>	- Path to the quartus_stp application.

## Returns

- Returns a vector with the processing results.

3.21.2.4 void add.dataflow.DATAFLOW\_SYNC\_BASE\_SIMUL.start\_FPGA ( String *CONF*, String *QUARTUS*, String *DESIRED\_RETURN* )

Method responsible for running the algorithm on the FPGA board and display the output in the system default output.

## Parameters

<i>CONF</i>	- File containing the configuration and data to be processed.
<i>QUARTUS</i>	- Path to the quartus_stp application.
<i>DESIRED_RETURN</i>	- Expected outcome.

3.21.2.5 int [] add.dataflow.DATAFLOW\_SYNC\_BASE\_SIMUL.start\_HADES ( int[] *CONF*, String *DESIGN* )

Method responsible for executing the algorithm in the simulator.

## Parameters

<i>CONF</i>	- Configuration vector and data to be executed.
<i>DESIGN</i>	- Design to be used to run the simulator.

## Returns

- Returns a vector with the processing results.

3.21.2.6 void add.dataflow.DATAFLOW\_SYNC\_BASE\_SIMUL.start\_HADES ( String *CONF*, String *DESIGN*, String *DESIRED\_RETURN* )

Method responsible for executing the algorithm in the simulator and display the output in the system default output.

## Parameters

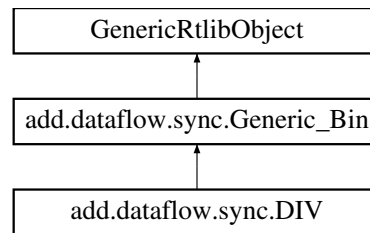
<i>CONF</i>	- File containing the configuration and data to be processed.
<i>DESIGN</i>	- Design to be used to run the simulator.
<i>DESIRED_RETURN</i>	- Expected outcome.

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

- add/dataflow/DATAFLOW\_SYNC\_BASE\_SIMUL.java

## 3.22 add.dataflow.sync.DIV Class Reference

Inheritance diagram for add.dataflow.sync.DIV:



## Public Member Functions

- [DIV](#) ()
- int [Compute](#) (int data1, int data2)

## Additional Inherited Members

### 3.22.1 Detailed Description

[DIV](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for dividing the inputs.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.22.2 Constructor & Destructor Documentation

#### 3.22.2.1 `add.dataflow.sync.DIV.DIV ( )`

Object Constructor.

### 3.22.3 Member Function Documentation

#### 3.22.3.1 `int add.dataflow.sync.DIV.Compute ( int data1, int data2 )`

Method responsible for the component computation: in this case performs a division of the parameters.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

#### Returns

- Returns the result of the computation. In this case the value of the division of the parameters.

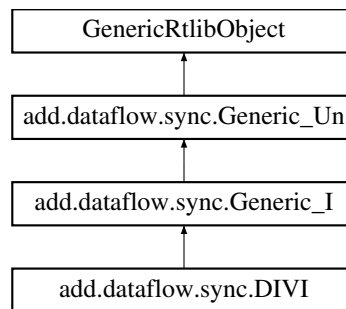
The documentation for this class was generated from the following file:

- `add/dataflow/sync/DIV.java`



## 3.23 add.dataflow.sync.DIVI Class Reference

Inheritance diagram for add.dataflow.sync.DIVI:



### Public Member Functions

- [DIVI](#) ()
- int [Compute](#) (int data)

### Additional Inherited Members

#### 3.23.1 Detailed Description

[DIVI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for dividing the input by a (immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.23.2 Constructor & Destructor Documentation

#### 3.23.2.1 add.dataflow.sync.DIVI.DIVI ( )

Object Constructor.

### 3.23.3 Member Function Documentation

#### 3.23.3.1 int add.dataflow.sync.DIVI.Compute ( int *data* )

Method responsible for the component computation: in this case performs a division of the parameter by an (immediate) constant.

## Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

## Returns

- Returns the result of the computation. In this case the value of the division of the parameter by the constant.

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

- add/dataflow/sync/DIVI.java

## 3.24 add.examples.dataflow\_sync.FIR\_4\_DATAFLOW\_FPGA Class Reference

### Static Public Member Functions

- static void **main** (String argv[])

#### 3.24.1 Detailed Description

FIR\_4 data flow example in FPGA Board in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

\* 1.0

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

- add/examples/dataflow\_sync/FIR\_4\_DATAFLOW\_FPGA.java

## 3.25 add.examples.dataflow\_sync.FIR\_4\_DATAFLOW\_HADES Class Reference

### Static Public Member Functions

- static void **main** (String argv[])

#### 3.25.1 Detailed Description

FIR\_4 data flow example in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

\* 1.0

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

- add/examples/dataflow\_sync/FIR\_4\_DATAFLOW\_HADES.java

### 3.26 add.examples.dataflow\_sync.FIR\_4\_DATAFLOW\_HADES\_WITH\_GENERATED\_HDS Class Reference

#### Static Public Member Functions

- static void **main** (String argv[])

#### 3.26.1 Detailed Description

FIR\_4 data flow (Automatically generated) example in the UFV ADD synchronous data flow simulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

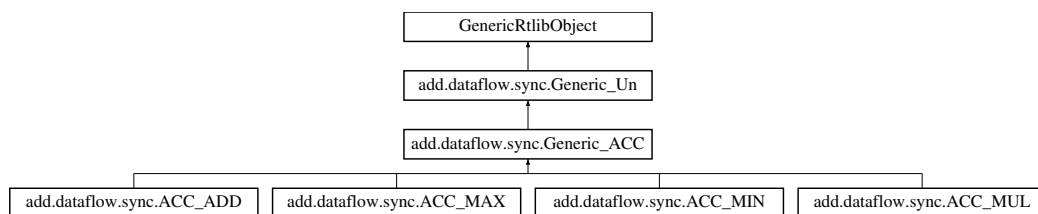
1.0

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

- add/examples/dataflow\_sync/FIR\_4\_DATAFLOW\_HADES\_WITH\_GENERATED\_HDS.java

### 3.27 add.dataflow.sync.Generic\_ACC Class Reference

Inheritance diagram for add.dataflow.sync.Generic\_ACC:



#### Public Member Functions

- [Generic\\_ACC](#) ()
- void [Reseted](#) ()
- void [evaluate](#) (Object arg)

## Protected Member Functions

- void [Accumulate](#) (int data)

## Protected Attributes

- int **acc**
- boolean **start**

### 3.27.1 Detailed Description

[Generic\\_ACC](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements a generic accumulator.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.27.2 Constructor & Destructor Documentation

#### 3.27.2.1 `add.dataflow.sync.Generic_ACC.Generic_ACC ( )`

Object Constructor.

### 3.27.3 Member Function Documentation

#### 3.27.3.1 `void add.dataflow.sync.Generic_ACC.Accumulate ( int data )` `[protected]`

Method responsible for performing the accumulation or not.

#### Parameters

<i>data</i>	- Value to be used for the computation.
-------------	---

#### 3.27.3.2 `void add.dataflow.sync.Generic_ACC.evaluate ( Object arg )`

[evaluate\(\)](#): called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked whether the ports are connected and will execute the `Compute (int data)` method if the `R_IN` input is high level. It will execute the [Reseted\(\)](#), [Tick\\_Up\(\)](#), and [Tick\\_Down\(\)](#) methods if their respective entries order it. It will update the output with the ACC value when the computation finishes.

#### Parameters

<i>arg</i>	an arbitrary object argument
------------	------------------------------

### 3.27.3.3 void add.dataflow.sync.Generic\_ACC.Reseted ( )

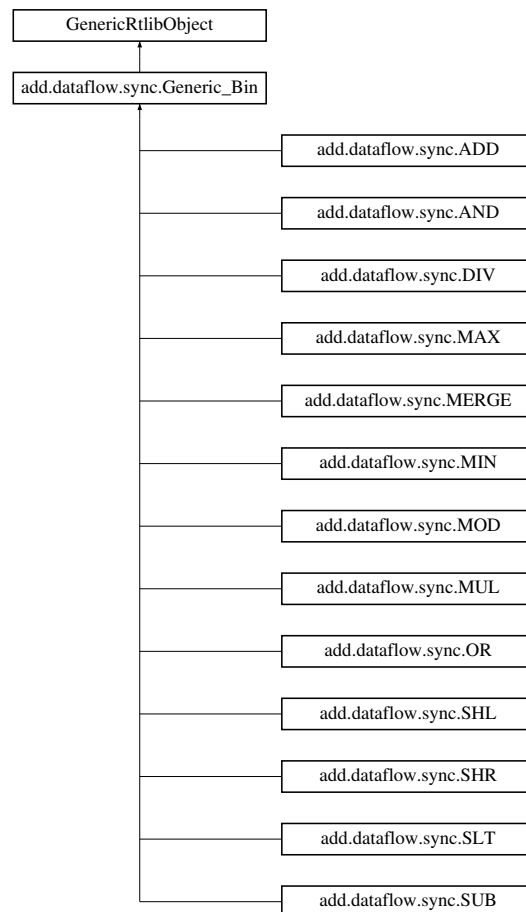
Method executed when the signal from the reset input goes to high logic level. In this case it clears the text displayed by the component and de accumulator.

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

- add/dataflow/sync/Generic\_ACC.java

## 3.28 add.dataflow.sync.Generic\_Bin Class Reference

Inheritance diagram for add.dataflow.sync.Generic\_Bin:



### Public Member Functions

- [Generic\\_Bin](#) ( )
- void [constructPorts](#) ( )
- void [setString](#) (String s)
- void [setSymbol](#) (Symbol s)
- int [Compute](#) (int data1, int data2)
- void [NotCompute](#) ( )

- void [Reseted](#) ()
- void [Tick\\_Up](#) ()
- void [Tick\\_Down](#) ()
- void [Set\\_Name](#) (String l)
- void [evaluate](#) (Object arg)
- boolean [needsDynamicSymbol](#) ()
- void [constructDynamicSymbol](#) ()
- void [write](#) (java.io.PrintWriter ps)
- boolean [initialize](#) (String s)

### Protected Attributes

- Label **stringLabel**
- String **l**
- PortStdLogic1164 **port\_CLK**
- PortStdLogicVector **port\_DATA\_IN1**

### 3.28.1 Detailed Description

[Generic\\_Bin](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component creates the basis for other components with two inputs.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.28.2 Constructor & Destructor Documentation

#### 3.28.2.1 `add.dataflow.sync.Generic_Bin.Generic_Bin ( )`

Object Constructor.

### 3.28.3 Member Function Documentation

#### 3.28.3.1 `int add.dataflow.sync.Generic_Bin.Compute ( int data1, int data2 )`

Method responsible for the computation of the output.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 1.

#### Returns

- Return of computation

## 3.28.3.2 void add.dataflow.sync.Generic\_Bin.constructDynamicSymbol ( )

Method responsible for dynamically constructing the component symbol.

## 3.28.3.3 void add.dataflow.sync.Generic\_Bin.constructPorts ( )

Method responsible for initializing the component input and output ports.

3.28.3.4 void add.dataflow.sync.Generic\_Bin.evaluate ( Object *arg* )

[evaluate\(\)](#): called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked whether the ports are connected and will execute the Compute (int data) method if the R\_IN (1 and 2) inputs are high level. It will execute the [Reseted\(\)](#), [Tick\\_Up\(\)](#), and [Tick\\_Down\(\)](#) methods if their respective entries order it. It will update the output with the Compute(int data) method result.

## Parameters

<i>arg</i>	an arbitrary object argument
------------	------------------------------

3.28.3.5 boolean add.dataflow.sync.Generic\_Bin.initialize ( String *s* )

Method responsible for reading the component settings in the file saved by the simulator.

## Parameters

<i>s</i>	- Settings for the component read from the file saved by the simulator.
----------	---

## Returns

- Returns true if the settings are read successfully.

## 3.28.3.6 boolean add.dataflow.sync.Generic\_Bin.needsDynamicSymbol ( )

Method responsible for indicating to the simulator that the component's symbol will be constructed dynamically by the [constructDynamicSymbol\(\)](#) method, or will be read from a file of the same name as the ".sym" extension.

## 3.28.3.7 void add.dataflow.sync.Generic\_Bin.NotCompute ( )

Method executed when computing is not performed. In this case it clears the text displayed by the component.

## 3.28.3.8 void add.dataflow.sync.Generic\_Bin.Reseted ( )

Method executed when the signal from the reset input goes to high logic level. In this case it clears the text displayed by the component.

3.28.3.9 void add.dataflow.sync.Generic\_Bin.Set\_Name ( String *l* )

Method responsible for changing the label that displays the name of the component.

3.28.3.10 void add.dataflow.sync.Generic\_Bin.setString ( String *s* )

Method responsible for updating the text displayed by the component.

## Parameters

<i>s</i>	- Text to be updated.
----------	-----------------------

3.28.3.11 void add.dataflow.sync.Generic\_Bin.setSymbol ( Symbol *s* )

Method responsible for updating the component symbol.

## 3.28.3.12 void add.dataflow.sync.Generic\_Bin.Tick\_Down ( )

Method executed when the clock signal goes to low logic level.

## 3.28.3.13 void add.dataflow.sync.Generic\_Bin.Tick\_Up ( )

Method executed when the clock signal goes to high logic level.

3.28.3.14 void add.dataflow.sync.Generic\_Bin.write ( java.io.PrintWriter *ps* )

Method responsible for writing component settings to the file saved by the simulator.

## Parameters

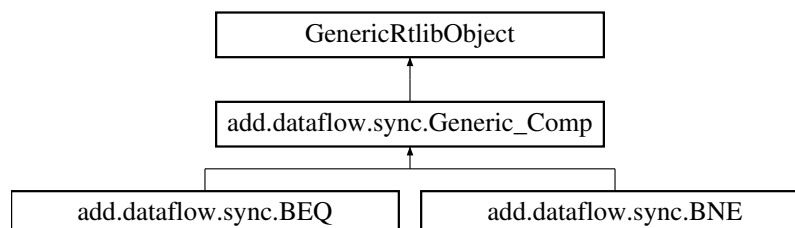
<i>ps</i>	-Simulator writing object.
-----------	----------------------------

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

- add/dataflow/sync/Generic\_Bin.java

## 3.29 add.dataflow.sync.Generic\_Comp Class Reference

Inheritance diagram for add.dataflow.sync.Generic\_Comp:



### Public Member Functions

- [Generic\\_Comp](#) ()
- void [constructPorts](#) ()
- void [setString](#) (String *s*)
- void [setSymbol](#) (Symbol *s*)
- int [Compute](#) (int data1, int data2)
- void [Reseted](#) ()
- void [Tick\\_Up](#) ()
- void [Tick\\_Down](#) ()
- void [Set\\_Name](#) (String *l*)
- void [evaluate](#) (Object arg)



- boolean [needsDynamicSymbol](#) ()
- void [constructDynamicSymbol](#) ()
- void [write](#) (java.io.PrintWriter ps)
- boolean [initialize](#) (String s)

### Protected Attributes

- Label **stringLabel**
- String **I**
- PortStdLogic1164 **port\_CLK**
- PortStdLogicVector **port\_DATA\_IN1**
- PortStdLogic1164 **port\_IF**

### 3.29.1 Detailed Description

[Generic\\_Comp](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component creates the basis for other components with an input and that make a comparison between the inputs.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.29.2 Constructor & Destructor Documentation

#### 3.29.2.1 `add.dataflow.sync.Generic_Comp.Generic_Comp ( )`

Object Constructor.

### 3.29.3 Member Function Documentation

#### 3.29.3.1 `int add.dataflow.sync.Generic_Comp.Compute ( int data1, int data2 )`

Method responsible for the computation of the output.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 1.

#### Returns

- Return of computation

#### 3.29.3.2 `void add.dataflow.sync.Generic_Comp.constructDynamicSymbol ( )`

Method responsible for dynamically constructing the component symbol.

### 3.29.3.3 void add.dataflow.sync.Generic\_Comp.constructPorts ( )

Method responsible for initializing the component input and output ports.

### 3.29.3.4 void add.dataflow.sync.Generic\_Comp.evaluate ( Object *arg* )

[evaluate\(\)](#): called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked whether the ports are connected and will execute the Compute (int data) method if the R\_IN (1 and 2) inputs are high level. It will execute the [Reseted\(\)](#), [Tick\\_Up\(\)](#), and [Tick\\_Down\(\)](#) methods if their respective entries order it. It will update the output with the Compute(int data) method result.

#### Parameters

<i>arg</i>	an arbitrary object argument
------------	------------------------------

### 3.29.3.5 boolean add.dataflow.sync.Generic\_Comp.initialize ( String *s* )

Method responsible for reading the component settings in the file saved by the simulator.

#### Parameters

<i>s</i>	- Settings for the component read from the file saved by the simulator.
----------	---

#### Returns

- Returns true if the settings are read successfully.

### 3.29.3.6 boolean add.dataflow.sync.Generic\_Comp.needsDynamicSymbol ( )

Method responsible for indicating to the simulator that the component's symbol will be constructed dynamically by the [constructDynamicSymbol\(\)](#) method, or will be read from a file of the same name as the ".sym" extension.

### 3.29.3.7 void add.dataflow.sync.Generic\_Comp.Reseted ( )

Method executed when the signal from the reset input goes to high logic level. In this case it clears the text displayed by the component.

### 3.29.3.8 void add.dataflow.sync.Generic\_Comp.Set\_Name ( String *I* )

Method responsible for changing the label that displays the name of the component.

### 3.29.3.9 void add.dataflow.sync.Generic\_Comp.setString ( String *s* )

Method responsible for updating the text displayed by the component.

#### Parameters

<i>s</i>	- Text to be updated.
----------	-----------------------

### 3.29.3.10 void add.dataflow.sync.Generic\_Comp.setSymbol ( Symbol *s* )

Method responsible for updating the component symbol.

3.29.3.11 void add.dataflow.sync.Generic\_Comp.Tick\_Down ( )

Method executed when the clock signal goes to low logic level.

3.29.3.12 void add.dataflow.sync.Generic\_Comp.Tick\_Up ( )

Method executed when the clock signal goes to high logic level.

3.29.3.13 void add.dataflow.sync.Generic\_Comp.write ( java.io.PrintWriter *ps* )

Method responsible for writing component settings to the file saved by the simulator.

Parameters

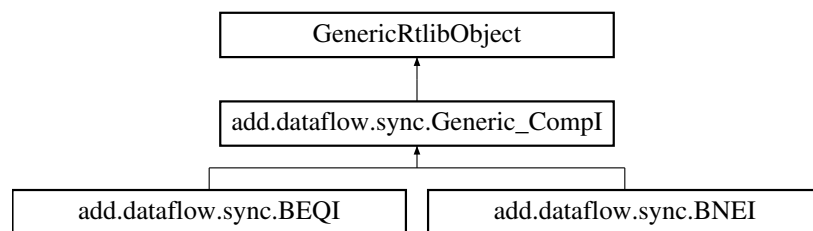
<i>ps</i>	-Simulator writing object.
-----------	----------------------------

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

- add/dataflow/sync/Generic\_Comp.java

## 3.30 add.dataflow.sync.Generic\_CompI Class Reference

Inheritance diagram for add.dataflow.sync.Generic\_CompI:



### Public Member Functions

- [Generic\\_CompI](#) ( )
- void [constructPorts](#) ( )
- void [setString](#) (String *s*)
- void [setSymbol](#) (Symbol *s*)
- int [Compute](#) (int *data*)
- void [Reseted](#) ( )
- void [Tick\\_Up](#) ( )
- void [Tick\\_Down](#) ( )
- void [Set\\_Name](#) (String *l*)
- void [evaluate](#) (Object *arg*)
- boolean [needsDynamicSymbol](#) ( )
- void [constructDynamicSymbol](#) ( )
- void [write](#) (java.io.PrintWriter *ps*)
- boolean [initialize](#) (String *s*)
- void [mousePressed](#) (java.awt.event.MouseEvent *me*)

## Protected Attributes

- Label **stringLabel**
- String **s**
- Rectangle **background**
- PortStdLogic1164 **port\_CLK**
- PortStdLogicVector **port\_DATA\_IN**
- int **constante**

### 3.30.1 Detailed Description

[Generic\\_Compl](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component creates the basis for other components with an input and that make a comparison with a (immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.30.2 Constructor & Destructor Documentation

#### 3.30.2.1 `add.dataflow.sync.Generic_Compl.Generic_Compl ( )`

Object Constructor.

### 3.30.3 Member Function Documentation

#### 3.30.3.1 `int add.dataflow.sync.Generic_Compl.Compute ( int data )`

Method responsible for the computation of the output.

##### Parameters

<i>data</i>	- Value to be used for the computation.
-------------	---

##### Returns

- Return of computation

#### 3.30.3.2 `void add.dataflow.sync.Generic_Compl.constructDynamicSymbol ( )`

Method responsible for dynamically constructing the component symbol.

#### 3.30.3.3 `void add.dataflow.sync.Generic_Compl.constructPorts ( )`

Method responsible for initializing the component input and output ports.

**3.30.3.4 void add.dataflow.sync.Generic\_Compl.evaluate ( Object *arg* )**

**evaluate()**: called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked whether the ports are connected and will execute the Compute (int data) method if the R\_IN input is high level. It will execute the [Reseted\(\)](#), [Tick\\_Up\(\)](#), and [Tick\\_Down\(\)](#) methods if their respective entries order it. It will update the output with the [Compute\(int data\)](#) method result.

**Parameters**

<i>arg</i>	an arbitrary object argument
------------	------------------------------

**3.30.3.5 boolean add.dataflow.sync.Generic\_Compl.initialize ( String *s* )**

Method responsible for reading the component settings in the file saved by the simulator.

**Parameters**

<i>s</i>	- Settings for the component read from the file saved by the simulator.
----------	---

**Returns**

- Returns true if the settings are read successfully.

**3.30.3.6 void add.dataflow.sync.Generic\_Compl.mousePressed ( java.awt.event.MouseEvent *me* )**

Method responsible for changing the value of the constant for more or less, depending on whether the mouse click is done by the right or left button respectively.

**Parameters**

<i>me</i>	- Object where the event occurred.
-----------	------------------------------------

**3.30.3.7 boolean add.dataflow.sync.Generic\_Compl.needsDynamicSymbol ( )**

Method responsible for indicating to the simulator that the component's symbol will be constructed dynamically by the [constructDynamicSymbol\(\)](#) method, or will be read from a file of the same name as the ".sym" extension.

**3.30.3.8 void add.dataflow.sync.Generic\_Compl.Reseted ( )**

Method executed when the signal from the reset input goes to high logic level. In this case it clears the text displayed by the component.

**3.30.3.9 void add.dataflow.sync.Generic\_Compl.Set\_Name ( String *l* )**

Method responsible for changing the label that displays the name of the component.

**3.30.3.10 void add.dataflow.sync.Generic\_Compl.setString ( String *s* )**

Method responsible for updating the text displayed by the component.

## Parameters

<i>s</i>	- Text to be updated.
----------	-----------------------

3.30.3.11 void add.dataflow.sync.Generic\_Compl.setSymbol ( Symbol *s* )

Method responsible for updating the component symbol.

3.30.3.12 void add.dataflow.sync.Generic\_Compl.Tick\_Down ( )

Method executed when the clock signal goes to low logic level.

3.30.3.13 void add.dataflow.sync.Generic\_Compl.Tick\_Up ( )

Method executed when the clock signal goes to high logic level.

3.30.3.14 void add.dataflow.sync.Generic\_Compl.write ( java.io.PrintWriter *ps* )

Method responsible for writing component settings to the file saved by the simulator.

## Parameters

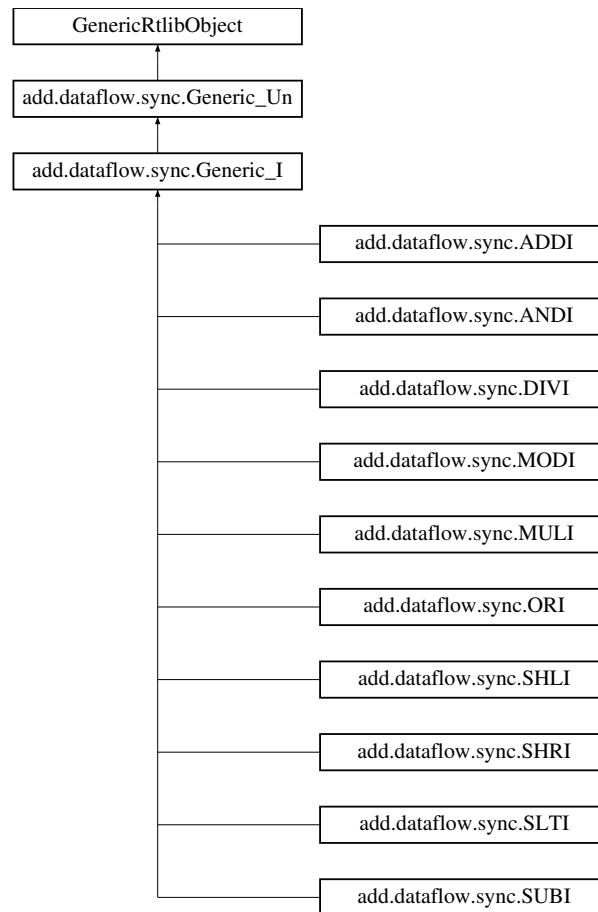
<i>ps</i>	-Simulator writing object.
-----------	----------------------------

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

- add/dataflow/sync/Generic\_Compl.java

## 3.31 add.dataflow.sync.Generic\_I Class Reference

Inheritance diagram for add.dataflow.sync.Generic\_I:



### Public Member Functions

- [Generic\\_I\(\)](#)
- int [Compute](#) (int data)
- void [NotCompute](#) ()
- void [Reseted](#) ()
- void [write](#) (java.io.PrintWriter ps)
- boolean [initialize](#) (String s)
- void [mousePressed](#) (java.awt.event.MouseEvent me)

### Protected Attributes

- int **constante**

#### 3.31.1 Detailed Description

[Generic\\_I](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component creates the basis for other components with an input and that perform the computation with an (immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

**3.31.2 Constructor & Destructor Documentation****3.31.2.1 add.dataflow.sync.Generic\_I.Generic\_I ( )**

Object Constructor.

**3.31.3 Member Function Documentation****3.31.3.1 int add.dataflow.sync.Generic\_I.Compute ( int *data* )**

Method responsible for the computation of the output and set the new text to be shown by the component. In this case the constant.

**Parameters**

<i>data</i>	- Value to be used for the computation.
-------------	---

**Returns**

- Return of computation

**3.31.3.2 boolean add.dataflow.sync.Generic\_I.initialize ( String *s* )**

Method responsible for reading the component settings in the file saved by the simulator.

**Parameters**

<i>s</i>	- Settings for the component read from the file saved by the simulator.
----------	---

**Returns**

- Returns true if the settings are read successfully.

**3.31.3.3 void add.dataflow.sync.Generic\_I.mousePressed ( java.awt.event.MouseEvent *me* )**

Method responsible for changing the value of the constant for more or less, depending on whether the mouse click is done by the right or left button respectively.

**Parameters**

<i>me</i>	- Object where the event occurred.
-----------	------------------------------------

**3.31.3.4 void add.dataflow.sync.Generic\_I.NotCompute ( )**

Method executed when computing is not performed.



## 3.31.3.5 void add.dataflow.sync.Generic\_I.Reseted ( )

Method executed when the signal from the reset input goes to high logic level. It sets the new text to be shown by the component. In this case the constant.

## 3.31.3.6 void add.dataflow.sync.Generic\_I.write ( java.io.PrintWriter ps )

Method responsible for writing component settings to the file saved by the simulator.

## Parameters

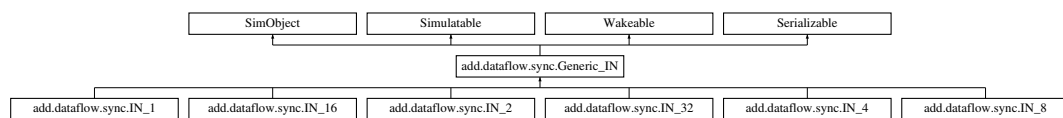
<i>ps</i>	-Simulator writing object.
-----------	----------------------------

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

- add/dataflow/sync/Generic\_I.java

## 3.32 add.dataflow.sync.Generic\_IN Class Reference

Inheritance diagram for add.dataflow.sync.Generic\_IN:



## Public Member Functions

- [Generic\\_IN](#) ( )
- [Generic\\_IN](#) (int QTDE\_PORTS)
- void [constructPorts](#) ( )
- void [setVectorIn](#) (int[] vector)
- void [Set\\_Name](#) (String l)
- void [evaluate](#) (Object arg)
- boolean [needsDynamicSymbol](#) ( )
- void [constructDynamicSymbol](#) ( )
- void [write](#) (java.io.PrintWriter ps)
- boolean [initialize](#) (String s)
- double [getDelay](#) ( )
- void [setDelay](#) (double \_delay)
- void [setDelay](#) (String s)
- void [wakeup](#) (Object arg)
- void [updateSymbol](#) ( )

## Protected Member Functions

- void [constructStandardValues](#) ( )

## Protected Attributes

- int **n\_bits** = 16
- StdLogicVector **vector**
- PortStdLogicVector **vectorOutputPort**
- double **delay**
- double **defaultdelay** = 10E-9
- boolean **enableAnimationFlag**
- ColoredValueLabel **valueLabel**
- FlexibleLabelFormatter **labelFormatter**
- PortStdLogic1164 **port\_CLK**
- PortStdLogicVector[] **port\_DATA\_OUT**
- PortStdLogic1164[] **port\_R\_OUT**
- int[] **vector\_in**
- int **idx\_data\_in**

### 3.32.1 Detailed Description

[Generic\\_IN](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component creates the basis for other components that implement input queues with 1, 2, 4, 8, 16, or 32 outputs.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.32.2 Constructor & Destructor Documentation

#### 3.32.2.1 `add.dataflow.sync.Generic_IN.Generic_IN ( )`

Object Constructor. By default, an input queue of an output is created.

#### 3.32.2.2 `add.dataflow.sync.Generic_IN.Generic_IN ( int QTDE_PORTS )`

Object Constructor. An input queue of N outputs is created.

#### Parameters

<i>QTDE_PORTS</i>	- Number of queue outputs to be created
-------------------	---

### 3.32.3 Member Function Documentation

#### 3.32.3.1 `void add.dataflow.sync.Generic_IN.constructDynamicSymbol ( )`

Method responsible for dynamically constructing the component symbol.

**3.32.3.2 void add.dataflow.sync.Generic\_IN.constructPorts ( )**

Method responsible for initializing the component input and output ports.

**3.32.3.3 void add.dataflow.sync.Generic\_IN.constructStandardValues ( ) [protected]**

Method responsible for creating some auxiliary variables for working with bit vectors.

**3.32.3.4 void add.dataflow.sync.Generic\_IN.evaluate ( Object *arg* )**

[evaluate\(\)](#): called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked whether the ports are connected. It Will pass the vector data to the outputs.

**Parameters**

<i>arg</i>	an arbitrary object argument
------------	------------------------------

**3.32.3.5 double add.dataflow.sync.Generic\_IN.getDelay ( )**

Method responsible for returning the value of the delay variable that contains the response delay time of the component.

**3.32.3.6 boolean add.dataflow.sync.Generic\_IN.initialize ( String *s* )**

Method responsible for reading the component settings in the file saved by the simulator.

**Parameters**

<i>s</i>	- Settings for the component read from the file saved by the simulator.
----------	---

**Returns**

- Returns true if the settings are read successfully.

**3.32.3.7 boolean add.dataflow.sync.Generic\_IN.needsDynamicSymbol ( )**

Method responsible for indicating to the simulator that the component's symbol will be constructed dynamically by the [constructDynamicSymbol\(\)](#) method, or will be read from a file of the same name as the ".sym" extension.

**3.32.3.8 void add.dataflow.sync.Generic\_IN.Set\_Name ( String *l* )**

Method responsible for changing the label that displays the name of the component.

**3.32.3.9 void add.dataflow.sync.Generic\_IN.setDelay ( double *\_delay* )**

Method responsible for changing the value of the delay variable that contains the response delay time of the component.

## Parameters

<i>_delay</i>	
---------------	--

## 3.32.3.10 void add.dataflow.sync.Generic\_IN.setDelay ( String s )

Method responsible for changing the value of the delay variable that contains the response delay time of the component.

## Parameters

<i>s</i>	
----------	--

## 3.32.3.11 void add.dataflow.sync.Generic\_IN.setVectorIn ( int[] vector )

Method responsible to set the data vector to be delivered to the outputs.

## Parameters

<i>vector</i>	- Vector that will be delivered to the outputs
---------------	--

## 3.32.3.12 void add.dataflow.sync.Generic\_IN.updateSymbol ( )

Method responsible for updating the component symbol.

## 3.32.3.13 void add.dataflow.sync.Generic\_IN.wakeup ( Object arg )

[wakeup\(\)](#): Called by the simulator as a reaction to our own scheduleWakeup()-calls. For RTLIB components, a [wakeup\(\)](#) is normally used to update the value label on its graphical symbol. A WakeupEvent for this purpose should have either 'null' or the current 'this' object as its payload.

A second use is to update our internal 'vector' variable at a specified simulation time, which is needed to implement the assign() method from interface `hades.simulator.Assignable`. A WakeupEvent for this purpose is expected to hold a `StdLogicVector` object (with the 'value' from the assign call) as its payload.

## 3.32.3.14 void add.dataflow.sync.Generic\_IN.write ( java.io.PrintWriter ps )

Method responsible for writing component settings to the file saved by the simulator.

## Parameters

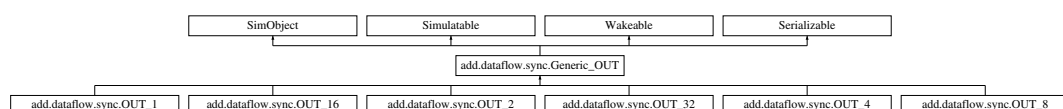
<i>ps</i>	-Simulator writing object.
-----------	----------------------------

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

- add/dataflow/sync/Generic\_IN.java

## 3.33 add.dataflow.sync.Generic\_OUT Class Reference

Inheritance diagram for `add.dataflow.sync.Generic_OUT`:



## Public Member Functions

- [Generic\\_OUT](#) ()
- [Generic\\_OUT](#) (int QTDE\_PORTS)
- void [constructPorts](#) ()
- boolean [getDoneSignal](#) ()
- void [setVetor](#) (int k)
- int[] [getVectorOut](#) ()
- void [Set\\_Name](#) (String l)
- void [evaluate](#) (Object arg)
- boolean [needsDynamicSymbol](#) ()
- void [constructDynamicSymbol](#) ()
- void [write](#) (java.io.PrintWriter ps)
- boolean [initialize](#) (String s)
- double [getDelay](#) ()
- void [setDelay](#) (double \_delay)
- void [setDelay](#) (String s)
- void [wakeUp](#) (Object arg)
- void [updateSymbol](#) ()

## Protected Member Functions

- void [constructStandardValues](#) ()

## Protected Attributes

- int **n\_bits** = 16
- StdLogicVector **vector**
- PortStdLogicVector **vectorOutputPort**
- double **delay**
- double **defaultdelay** = 10E-9
- boolean **enableAnimationFlag**
- ColoredValueLabel **valueLabel**
- FlexibleLabelFormatter **labelFormatter**
- PortStdLogic1164 **port\_CLK**
- PortStdLogicVector[] **port\_DATA\_IN**
- PortStdLogic1164[] **port\_R\_IN**
- int[] **vector\_out**
- int **idx\_data\_out**
- int **tamanho\_vetor\_saida**
- boolean **done** = false

### 3.33.1 Detailed Description

[Generic\\_OUT](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component creates the basis for other components that implement output queues with 1, 2, 4, 8, 16, or 32 inputs.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

## Version

1.0

**3.33.2 Constructor & Destructor Documentation****3.33.2.1 add.dataflow.sync.Generic\_OUT.Generic\_OUT ( )**

Object Constructor. By default, an input queue of an output is created.

**3.33.2.2 add.dataflow.sync.Generic\_OUT.Generic\_OUT ( int QTDE\_PORTS )**

Object Constructor. An output queue of N inputs is created.

## Parameters

<i>QTDE_PORTS</i>	- Number of queue inputs to be created
-------------------	--

**3.33.3 Member Function Documentation****3.33.3.1 void add.dataflow.sync.Generic\_OUT.constructDynamicSymbol ( )**

Method responsible for dynamically constructing the component symbol.

**3.33.3.2 void add.dataflow.sync.Generic\_OUT.constructPorts ( )**

Method responsible for initializing the component input and output ports.

**3.33.3.3 void add.dataflow.sync.Generic\_OUT.constructStandardValues ( ) [protected]**

Method responsible for creating some auxiliary variables for working with bit vectors.

**3.33.3.4 void add.dataflow.sync.Generic\_OUT.evaluate ( Object arg )**

**evaluate()**: called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked whether the ports are connected and if the R\_IN inputs are high level. It Will pass the data from the inputs to the vector.

## Parameters

<i>arg</i>	an arbitrary object argument
------------	------------------------------

**3.33.3.5 double add.dataflow.sync.Generic\_OUT.getDelay ( )**

Method responsible for returning the value of the delay variable that contains the response delay time of the component.

**3.33.3.6 boolean add.dataflow.sync.Generic\_OUT.getDoneSignal ( )**

Method responsible for returning end of data entry.

3.33.3.7 `int [] add.dataflow.sync.Generic_OUT.getVectorOut ( )`

Method responsible for returning the data vector received by the queue entries.

3.33.3.8 `boolean add.dataflow.sync.Generic_OUT.initialize ( String s )`

Method responsible for reading the component settings in the file saved by the simulator.

Parameters

<code>s</code>	- Settings for the component read from the file saved by the simulator.
----------------	---

Returns

- Returns true if the settings are read successfully.

3.33.3.9 `boolean add.dataflow.sync.Generic_OUT.needsDynamicSymbol ( )`

Method responsible for indicating to the simulator that the component's symbol will be constructed dynamically by the [constructDynamicSymbol\(\)](#) method, or will be read from a file of the same name as the ".sym" extension.

3.33.3.10 `void add.dataflow.sync.Generic_OUT.Set_Name ( String l )`

Method responsible for changing the label that displays the name of the component.

3.33.3.11 `void add.dataflow.sync.Generic_OUT.setDelay ( double _delay )`

Method responsible for changing the value of the delay variable that contains the response delay time of the component.

Parameters

<code>_delay</code>	
---------------------	--

3.33.3.12 `void add.dataflow.sync.Generic_OUT.setDelay ( String s )`

Method responsible for changing the value of the delay variable that contains the response delay time of the component.

Parameters

<code>s</code>	
----------------	--

3.33.3.13 `void add.dataflow.sync.Generic_OUT.setVetor ( int k )`

Method responsible for inserting elements into the vector.

3.33.3.14 `void add.dataflow.sync.Generic_OUT.updateSymbol ( )`

Method responsible for updating the component symbol.

### 3.33.3.15 void add.dataflow.sync.Generic\_OUT.wakeup ( Object arg )

[wakeup\(\)](#): Called by the simulator as a reaction to our own scheduleWakeup()-calls. For RTLIB components, a [wakeup\(\)](#) is normally used to update the value label on its graphical symbol. A WakeupEvent for this purpose should have either 'null' or the current 'this' object as its payload.

A second use is to update our internal 'vector' variable at a specified simulation time, which is needed to implement the assign() method from interface `hades.simulator.Assignable`. A WakeupEvent for this purpose is expected to hold a `StdLogicVector` object (with the 'value' from the assign call) as its payload.

### 3.33.3.16 void add.dataflow.sync.Generic\_OUT.write ( java.io.PrintWriter ps )

Method responsible for writing component settings to the file saved by the simulator.

#### Parameters

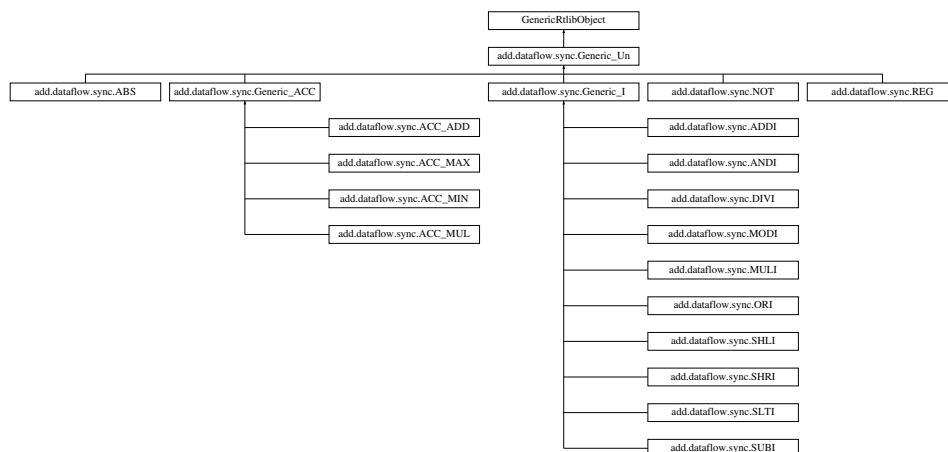
<i>ps</i>	-Simulator writing object.
-----------	----------------------------

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

- `add/dataflow/sync/Generic_OUT.java`

## 3.34 add.dataflow.sync.Generic\_Un Class Reference

Inheritance diagram for `add.dataflow.sync.Generic_Un`:



### Public Member Functions

- [Generic\\_Un](#) ()
- void [constructPorts](#) ()
- void [setString](#) (String s)
- void [setSymbol](#) (Symbol s)
- int [Compute](#) (int data)
- void [NotCompute](#) ()
- void [Reseted](#) ()
- void [Tick\\_Up](#) ()
- void [Tick\\_Down](#) ()
- void [Set\\_Name](#) (String l)
- void [evaluate](#) (Object arg)
- boolean [needsDynamicSymbol](#) ()



- void [constructDynamicSymbol](#) ()
- void [write](#) (java.io.PrintWriter ps)
- boolean [initialize](#) (String s)

### Protected Attributes

- Label **stringLabel**
- String **s**
- Rectangle **background**
- PortStdLogic1164 **port\_CLK**
- PortStdLogicVector **port\_DATA\_IN**

### 3.34.1 Detailed Description

[Generic\\_Un](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component creates the basis for other components with one input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.34.2 Constructor & Destructor Documentation

#### 3.34.2.1 `add.dataflow.sync.Generic_Un.Generic_Un ( )`

Object Constructor.

### 3.34.3 Member Function Documentation

#### 3.34.3.1 `int add.dataflow.sync.Generic_Un.Compute ( int data )`

Method responsible for the computation of the output.

#### Parameters

<i>data</i>	- Value to be used for the computation.
-------------	---

#### Returns

- Return of computation

#### 3.34.3.2 `void add.dataflow.sync.Generic_Un.constructDynamicSymbol ( )`

Method responsible for dynamically constructing the component symbol.

### 3.34.3.3 void add.dataflow.sync.Generic\_Un.constructPorts ( )

Method responsible for initializing the component input and output ports.

### 3.34.3.4 void add.dataflow.sync.Generic\_Un.evaluate ( Object *arg* )

[evaluate\(\)](#): called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked whether the ports are connected and will execute the [Compute \(int data\)](#) method if the R\_IN input is high level. It will execute the [Reseted\(\)](#), [Tick\\_Up\(\)](#), and [Tick\\_Down\(\)](#) methods if their respective entries order it. It will update the output with the [Compute\(int data\)](#) method result.

#### Parameters

<i>arg</i>	an arbitrary object argument
------------	------------------------------

### 3.34.3.5 boolean add.dataflow.sync.Generic\_Un.initialize ( String *s* )

Method responsible for reading the component settings in the file saved by the simulator.

#### Parameters

<i>s</i>	- Settings for the component read from the file saved by the simulator.
----------	---

#### Returns

- Returns true if the settings are read successfully.

### 3.34.3.6 boolean add.dataflow.sync.Generic\_Un.needsDynamicSymbol ( )

Method responsible for indicating to the simulator that the component's symbol will be constructed dynamically by the [constructDynamicSymbol\(\)](#) method, or will be read from a file of the same name as the ".sym" extension.

### 3.34.3.7 void add.dataflow.sync.Generic\_Un.NotCompute ( )

Method executed when computing is not performed. In this case it clears the text displayed by the component.

### 3.34.3.8 void add.dataflow.sync.Generic\_Un.Reseted ( )

Method executed when the signal from the reset input goes to high logic level. In this case it clears the text displayed by the component.

### 3.34.3.9 void add.dataflow.sync.Generic\_Un.Set\_Name ( String *I* )

Method responsible for changing the label that displays the name of the component.

### 3.34.3.10 void add.dataflow.sync.Generic\_Un.setString ( String *s* )

Method responsible for updating the text displayed by the component.

## Parameters

<i>s</i>	- Text to be updated.
----------	-----------------------

3.34.3.11 void add.dataflow.sync.Generic\_Un.setSymbol ( Symbol *s* )

Method responsible for updating the component symbol.

3.34.3.12 void add.dataflow.sync.Generic\_Un.Tick\_Down ( )

Method executed when the clock signal goes to low logic level.

3.34.3.13 void add.dataflow.sync.Generic\_Un.Tick\_Up ( )

Method executed when the clock signal goes to high logic level.

3.34.3.14 void add.dataflow.sync.Generic\_Un.write ( java.io.PrintWriter *ps* )

Method responsible for writing component settings to the file saved by the simulator.

## Parameters

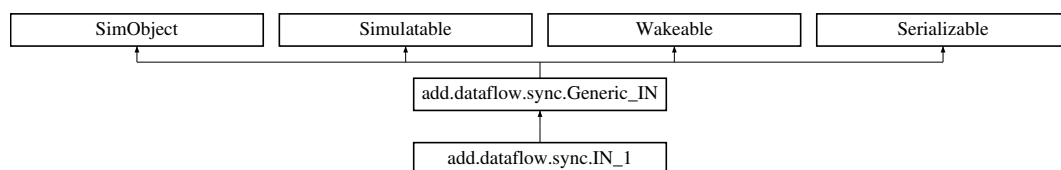
<i>ps</i>	-Simulator writing object.
-----------	----------------------------

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

- add/dataflow/sync/Generic\_Un.java

## 3.35 add.dataflow.sync.IN\_1 Class Reference

Inheritance diagram for add.dataflow.sync.IN\_1:



### Public Member Functions

- [IN\\_1](#) ()

### Additional Inherited Members

#### 3.35.1 Detailed Description

[IN\\_1](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an input queue with 1 output.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

**3.35.2 Constructor & Destructor Documentation****3.35.2.1 add.dataflow.sync.IN\_1.IN\_1 ( )**

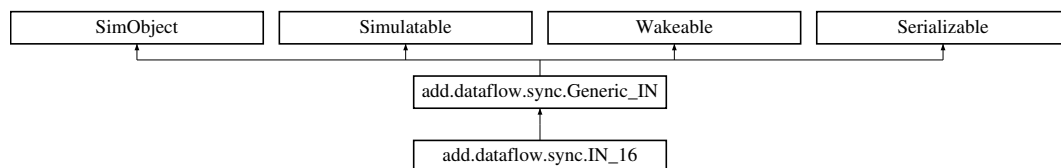
Object Constructor.

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

- add/dataflow/sync/IN\_1.java

**3.36 add.dataflow.sync.IN\_16 Class Reference**

Inheritance diagram for add.dataflow.sync.IN\_16:

**Public Member Functions**

- [IN\\_16 \(\)](#)

**Additional Inherited Members****3.36.1 Detailed Description**

[IN\\_16](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an input queue with 16 output.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

### 3.36.2 Constructor & Destructor Documentation

#### 3.36.2.1 add.dataflow.sync.IN\_16.IN\_16 ( )

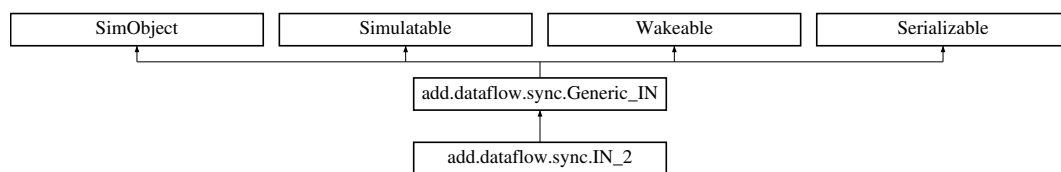
Object Constructor.

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

- add/dataflow/sync/IN\_16.java

## 3.37 add.dataflow.sync.IN\_2 Class Reference

Inheritance diagram for add.dataflow.sync.IN\_2:



### Public Member Functions

- [IN\\_2](#) ( )

### Additional Inherited Members

#### 3.37.1 Detailed Description

[IN\\_2](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an input queue with 2 output.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.37.2 Constructor & Destructor Documentation

#### 3.37.2.1 add.dataflow.sync.IN\_2.IN\_2 ( )

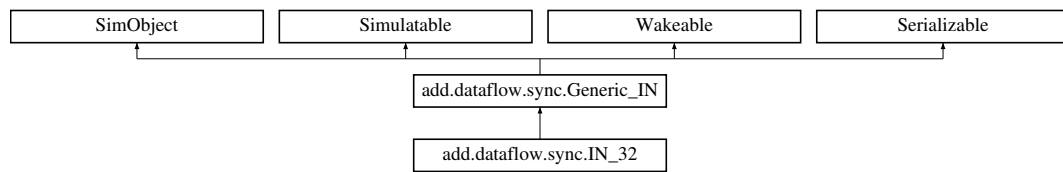
Object Constructor.

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

- add/dataflow/sync/IN\_2.java

### 3.38 add.dataflow.sync.IN\_32 Class Reference

Inheritance diagram for add.dataflow.sync.IN\_32:



#### Public Member Functions

- [IN\\_32](#) ()

#### Additional Inherited Members

##### 3.38.1 Detailed Description

[IN\\_32](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an input queue with 32 output.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

##### 3.38.2 Constructor & Destructor Documentation

###### 3.38.2.1 add.dataflow.sync.IN\_32.IN\_32 ( )

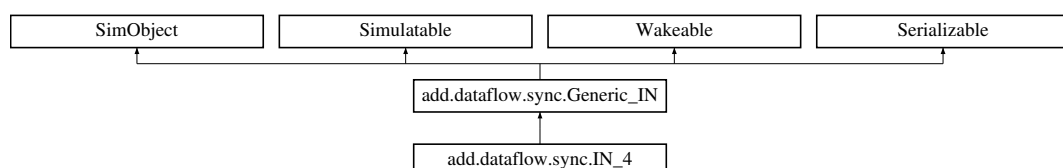
Object Constructor.

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

- `add/dataflow/sync/IN_32.java`

### 3.39 add.dataflow.sync.IN\_4 Class Reference

Inheritance diagram for add.dataflow.sync.IN\_4:



## Public Member Functions

- [IN\\_4](#) ()

## Additional Inherited Members

### 3.39.1 Detailed Description

[IN\\_4](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an input queue with 4 output.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.39.2 Constructor & Destructor Documentation

#### 3.39.2.1 add.dataflow.sync.IN\_4.IN\_4 ( )

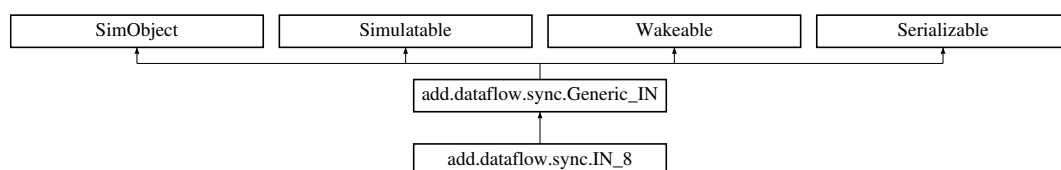
Object Constructor.

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

- add/dataflow/sync/IN\_4.java

## 3.40 add.dataflow.sync.IN\_8 Class Reference

Inheritance diagram for add.dataflow.sync.IN\_8:



## Public Member Functions

- [IN\\_8](#) ()

## Additional Inherited Members

### 3.40.1 Detailed Description

[IN\\_8](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an input queue with 8 output.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

**3.40.2 Constructor & Destructor Documentation****3.40.2.1 add.dataflow.sync.IN\_8.IN\_8 ( )**

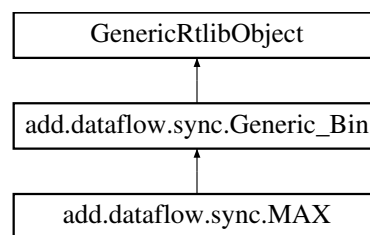
Object Constructor.

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

- add/dataflow/sync/IN\_8.java

**3.41 add.dataflow.sync.MAX Class Reference**

Inheritance diagram for add.dataflow.sync.MAX:

**Public Member Functions**

- [MAX](#) ()
- int [Compute](#) (int data1, int data2)

**Additional Inherited Members****3.41.1 Detailed Description**

[MAX](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for passing the output to the largest value input.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0



### 3.41.2 Constructor & Destructor Documentation

#### 3.41.2.1 add.dataflow.sync.MAX.MAX ( )

Object Constructor.

### 3.41.3 Member Function Documentation

#### 3.41.3.1 int add.dataflow.sync.MAX.Compute ( int *data1*, int *data2* )

Method responsible for the computation of components: in this case, it performs a comparison between the parameters and returns the largest between the two.

Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

Returns

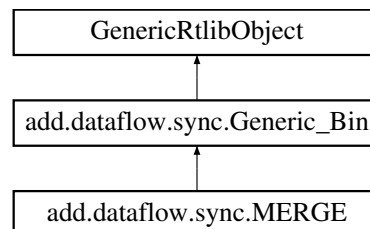
- Returns the result of the computation. In this case, the largest of the parameters.

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

- add/dataflow/sync/MAX.java

## 3.42 add.dataflow.sync.MERGE Class Reference

Inheritance diagram for add.dataflow.sync.MERGE:



### Public Member Functions

- [MERGE](#) ()
- void [evaluate](#) (Object arg)

### Additional Inherited Members

#### 3.42.1 Detailed Description

[MERGE](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for choosing which of the inputs to pass to the output depending on the value of R\_IN1 and R\_IN2.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

**3.42.2 Constructor & Destructor Documentation****3.42.2.1 add.dataflow.sync.MERGE.MERGE ( )**

Object Constructor.

**3.42.3 Member Function Documentation****3.42.3.1 void add.dataflow.sync.MERGE.evaluate ( Object *arg* )**

**evaluate()**: called by the simulation engine on all events that concern this object. The object is responsible for updating its internal state and for scheduling all pending output events. In this case, it will be checked if any of the R\_IN (1 or 2) inputs is at high level and put the respective input value in the output. If the two R\_IN signals are at high level, the value of input 1 will be set to the output. If both are 0, nothing will be done.

**Parameters**

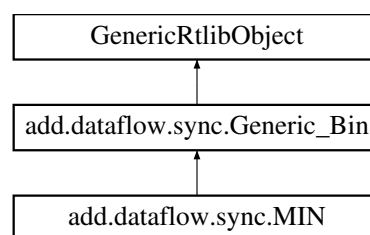
<i>arg</i>	an arbitrary object argument
------------	------------------------------

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

- add/dataflow/sync/MERGE.java

**3.43 add.dataflow.sync.MIN Class Reference**

Inheritance diagram for add.dataflow.sync.MIN:

**Public Member Functions**

- **MIN** ()
- int **Compute** (int data1, int data2)

**Additional Inherited Members****3.43.1 Detailed Description**

**MIN** component for the UFV **ADD** synchronous data flow simulator.

The component is responsible for passing the output to the lowest value input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.43.2 Constructor & Destructor Documentation

#### 3.43.2.1 add.dataflow.sync.MIN.MIN ( )

Object Constructor.

### 3.43.3 Member Function Documentation

#### 3.43.3.1 int add.dataflow.sync.MIN.Compute ( int *data1*, int *data2* )

Method responsible for the computation of components: in this case, it performs a comparison between the parameters and returns the smaller between the two.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

#### Returns

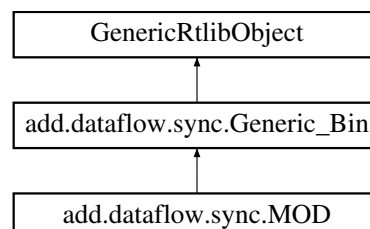
- Returns the result of the computation. In this case, the smallest of the parameters.

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

- add/dataflow/sync/MIN.java

## 3.44 add.dataflow.sync.MOD Class Reference

Inheritance diagram for add.dataflow.sync.MOD:



#### Public Member Functions

- [MOD](#) ()
- int [Compute](#) (int data1, int data2)

## Additional Inherited Members

### 3.44.1 Detailed Description

**MOD** component for the UFV **ADD** synchronous data flow simulator.

The component is responsible for calculating the rest of the integer division of the first input by the second one.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.44.2 Constructor & Destructor Documentation

#### 3.44.2.1 `add.dataflow.sync.MOD.MOD ( )`

Object Constructor.

### 3.44.3 Member Function Documentation

#### 3.44.3.1 `int add.dataflow.sync.MOD.Compute ( int data1, int data2 )`

Method responsible for the component computation: in this case, it returns the rest of the division between the parameters.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

#### Returns

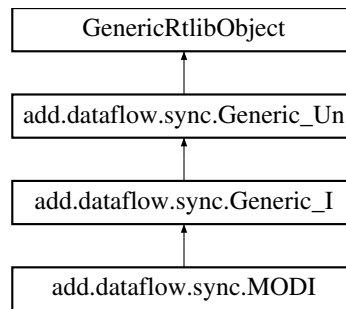
- Returns the result of the computation. In this case, it returns the rest of the division between the parameters.

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

- `add/dataflow/sync/MOD.java`

## 3.45 `add.dataflow.sync.MODI` Class Reference

Inheritance diagram for `add.dataflow.sync.MODI`:



## Public Member Functions

- [MODI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

### 3.45.1 Detailed Description

[MODI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for calculating the rest of the integer division of the input by a constant (Immediate).

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.45.2 Constructor & Destructor Documentation

#### 3.45.2.1 `add.dataflow.sync.MODI.MODI ( )`

Object Constructor.

### 3.45.3 Member Function Documentation

#### 3.45.3.1 `int add.dataflow.sync.MODI.Compute ( int data )`

Method responsible for the component computation: in this case, it returns the rest of the division of the parameter by the constant.

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

**Returns**

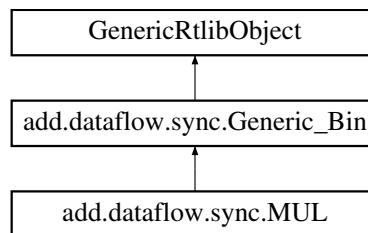
- Returns the result of the computation. In this case, it returns the rest of the division of the parameter by the constant.

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

- add/dataflow/sync/MODI.java

### 3.46 add.dataflow.sync.MUL Class Reference

Inheritance diagram for add.dataflow.sync.MUL:

**Public Member Functions**

- [MUL](#) ()
- int [Compute](#) (int data1, int data2)

**Additional Inherited Members**

#### 3.46.1 Detailed Description

[MUL](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for multiplying the inputs.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

**Version**

1.0

#### 3.46.2 Constructor & Destructor Documentation

##### 3.46.2.1 add.dataflow.sync.MUL.MUL ( )

Object Constructor.

### 3.46.3 Member Function Documentation

#### 3.46.3.1 int add.dataflow.sync.MUL.Compute ( int *data1*, int *data2* )

Method responsible for the component computation: in this case performs a multiplication of the parameters.

**Parameters**

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

**Returns**

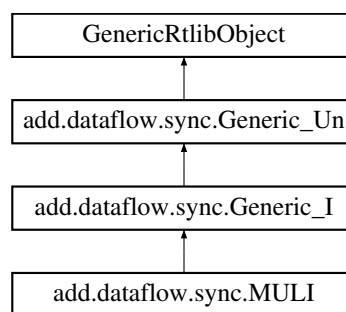
- Returns the result of the computation. In this case the value of the multiplication of the parameters.

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

- add/dataflow/sync/MUL.java

### 3.47 add.dataflow.sync.MULI Class Reference

Inheritance diagram for add.dataflow.sync.MULI:

**Public Member Functions**

- [MULI](#) ()
- int [Compute](#) (int data)

**Additional Inherited Members**

#### 3.47.1 Detailed Description

[MULI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for multiplying the input by a (Immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

**Version**

1.0

#### 3.47.2 Constructor & Destructor Documentation

##### 3.47.2.1 add.dataflow.sync.MULI.MULI ( )

Object Constructor.



### 3.47.3 Member Function Documentation

#### 3.47.3.1 int add.dataflow.sync.MULI.Compute ( int *data* )

Method responsible for the component computation: in this case performs a multiplying of the parameter by an (immediate) constant.

##### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

##### Returns

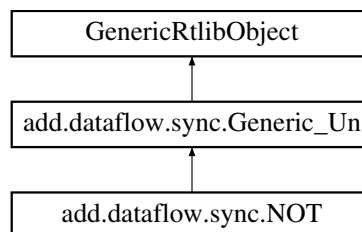
- Returns the result of the computation. In this case the value of the multiplication of the parameter by the constant.

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

- add/dataflow/sync/MULI.java

## 3.48 add.dataflow.sync.NOT Class Reference

Inheritance diagram for add.dataflow.sync.NOT:



### Public Member Functions

- [NOT](#) ()
- int [Compute](#) (int data)

### Additional Inherited Members

#### 3.48.1 Detailed Description

[NOT](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for the bitwise inversion of the input.

Universidade Federal de Viçosa - MG - Brasil.

##### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

##### Version

1.0

### 3.48.2 Constructor & Destructor Documentation

#### 3.48.2.1 `add.dataflow.sync.NOT.NOT ( )`

Object Constructor.

### 3.48.3 Member Function Documentation

#### 3.48.3.1 `int add.dataflow.sync.NOT.Compute ( int data )`

Method responsible for the component computation: in this case performs a bitwise inversion of the parameter.

Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

Returns

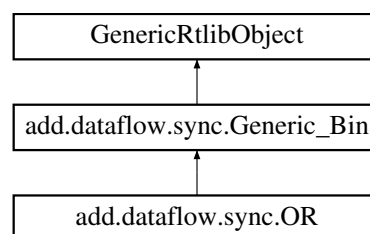
- Returns the result of the computation. In this case the value of the bitwise inversion of the parameter.

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

- `add/dataflow/sync/NOT.java`

## 3.49 `add.dataflow.sync.OR` Class Reference

Inheritance diagram for `add.dataflow.sync.OR`:



### Public Member Functions

- [OR](#) ()
- `int Compute (int data1, int data2)`

### Additional Inherited Members

#### 3.49.1 Detailed Description

[OR](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for the logical operation "OR" between the input

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicos@gmail.com](mailto:cacauvicos@gmail.com)

**Version**

1.0

**3.49.2 Constructor & Destructor Documentation****3.49.2.1 add.dataflow.sync.OR.OR ( )**

Object Constructor.

**3.49.3 Member Function Documentation****3.49.3.1 int add.dataflow.sync.OR.Compute ( int *data1*, int *data2* )**

Method responsible for the component computation: in this case it performs the logical operation "OR" between the parameters.

**Parameters**

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

**Returns**

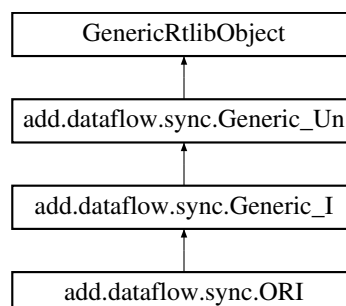
- Returns the result of the computation. In this case the result of the logical operation "OR" between the parameters.

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

- add/dataflow/sync/OR.java

**3.50 add.dataflow.sync.ORI Class Reference**

Inheritance diagram for add.dataflow.sync.ORI:

**Public Member Functions**

- [ORI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

### 3.50.1 Detailed Description

**ORI** component for the UFV **ADD** synchronous data flow simulator.

The component is responsible for the logical operation "OR" between the input and a constant (Immediate)

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.50.2 Constructor & Destructor Documentation

#### 3.50.2.1 `add.dataflow.sync.ORI.ORI ( )`

Object Constructor.

### 3.50.3 Member Function Documentation

#### 3.50.3.1 `int add.dataflow.sync.ORI.Compute ( int data )`

Method responsible for the component computation: in this case it performs the logical operation "OR" between the parameter and the (immediate) constant.

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

#### Returns

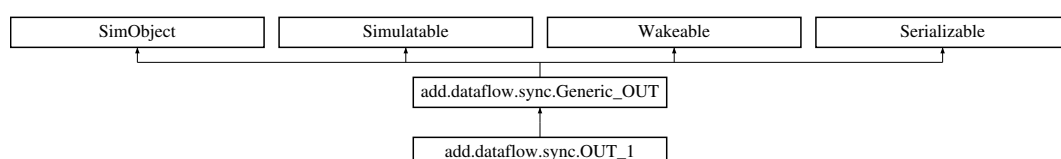
- Returns the result of the computation. In this case the result of the logical operation "OR" between the parameter and the constant.

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

- `add/dataflow/sync/ORI.java`

## 3.51 `add.dataflow.sync.OUT_1` Class Reference

Inheritance diagram for `add.dataflow.sync.OUT_1`:



## Public Member Functions

- [OUT\\_1\(\)](#)

## Additional Inherited Members

### 3.51.1 Detailed Description

[OUT\\_1](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an output queue with 1 input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.51.2 Constructor & Destructor Documentation

#### 3.51.2.1 add.dataflow.sync.OUT\_1.OUT\_1( )

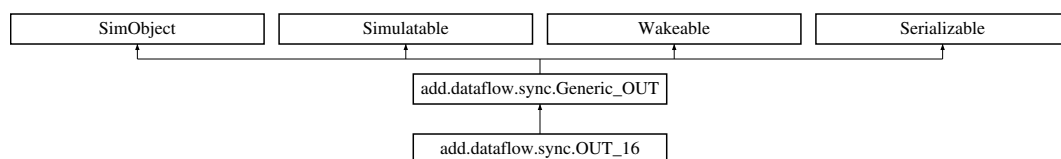
Object Constructor.

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

- `add/dataflow/sync/OUT_1.java`

## 3.52 add.dataflow.sync.OUT\_16 Class Reference

Inheritance diagram for add.dataflow.sync.OUT\_16:



## Public Member Functions

- [OUT\\_16\(\)](#)

## Additional Inherited Members

### 3.52.1 Detailed Description

[OUT\\_16](#) component for the UFV [ADD](#) synchronous data flow simulator.

O implements an input queue with 16 output.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

**3.52.2 Constructor & Destructor Documentation****3.52.2.1 add.dataflow.sync.OUT\_16.OUT\_16 ( )**

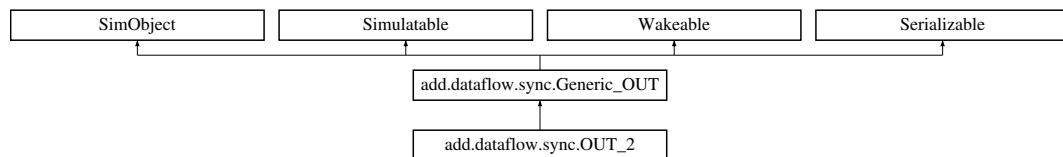
Object Constructor.

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

- add/dataflow/sync/OUT\_16.java

**3.53 add.dataflow.sync.OUT\_2 Class Reference**

Inheritance diagram for add.dataflow.sync.OUT\_2:

**Public Member Functions**

- [OUT\\_2 \(\)](#)

**Additional Inherited Members****3.53.1 Detailed Description**

[OUT\\_2](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an output queue with 2 input..

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

### 3.53.2 Constructor & Destructor Documentation

#### 3.53.2.1 add.dataflow.sync.OUT\_2.OUT\_2 ( )

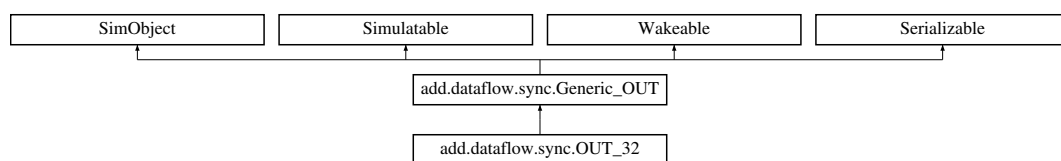
Object Constructor.

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

- add/dataflow/sync/OUT\_2.java

## 3.54 add.dataflow.sync.OUT\_32 Class Reference

Inheritance diagram for add.dataflow.sync.OUT\_32:



### Public Member Functions

- [OUT\\_32 \(\)](#)

### Additional Inherited Members

#### 3.54.1 Detailed Description

[OUT\\_32](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an output queue with 32 input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.54.2 Constructor & Destructor Documentation

#### 3.54.2.1 add.dataflow.sync.OUT\_32.OUT\_32 ( )

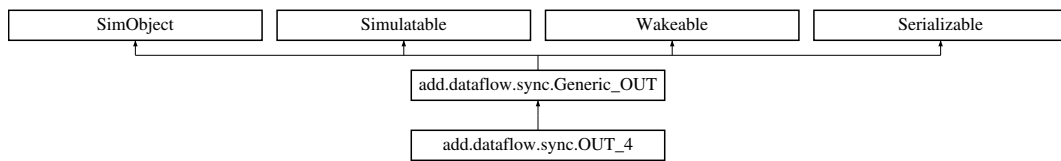
Object Constructor.

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

- add/dataflow/sync/OUT\_32.java

### 3.55 add.dataflow.sync.OUT\_4 Class Reference

Inheritance diagram for add.dataflow.sync.OUT\_4:



#### Public Member Functions

- [OUT\\_4\(\)](#)

#### Additional Inherited Members

##### 3.55.1 Detailed Description

[OUT\\_4](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an output queue with 4 input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

##### 3.55.2 Constructor & Destructor Documentation

###### 3.55.2.1 add.dataflow.sync.OUT\_4.OUT\_4( )

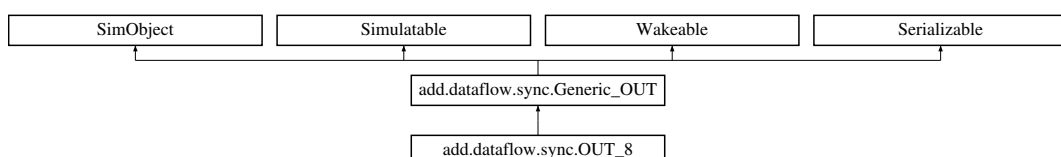
Object Constructor.

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

- `add/dataflow/sync/OUT_4.java`

### 3.56 add.dataflow.sync.OUT\_8 Class Reference

Inheritance diagram for add.dataflow.sync.OUT\_8:





## Public Member Functions

- [OUT\\_8](#) ()

## Additional Inherited Members

### 3.56.1 Detailed Description

[OUT\\_8](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component implements an output queue with 8 input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.56.2 Constructor & Destructor Documentation

#### 3.56.2.1 add.dataflow.sync.OUT\_8.OUT\_8 ( )

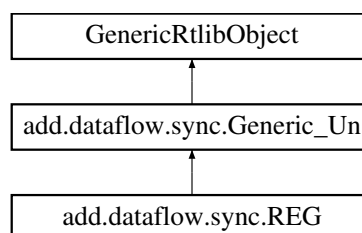
Object Constructor.

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

- add/dataflow/sync/OUT\_8.java

## 3.57 add.dataflow.sync.REG Class Reference

Inheritance diagram for add.dataflow.sync.REG:



## Public Member Functions

- [REG](#) ()

## Additional Inherited Members

### 3.57.1 Detailed Description

[REG](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for pass the input to the output when a clock pulse occurs.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.57.2 Constructor & Destructor Documentation

#### 3.57.2.1 add.dataflow.sync.REG.REG ( )

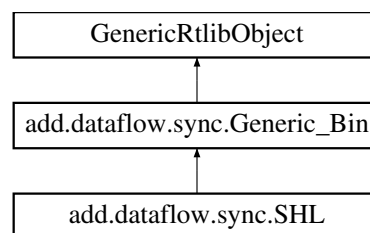
Object Constructor.

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

- add/dataflow/sync/REG.java

### 3.58 add.dataflow.sync.SHL Class Reference

Inheritance diagram for add.dataflow.sync.SHL:



#### Public Member Functions

- [SHL \(\)](#)
- [int Compute](#) (int data1, int data2)

#### Additional Inherited Members

#### 3.58.1 Detailed Description

[SHL](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for moving all the bits of the first input to the left N times, where N is equal to the value of the second input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

## Version

1.0

## 3.58.2 Constructor &amp; Destructor Documentation

## 3.58.2.1 add.dataflow.sync.SHL.SHL ( )

Object Constructor.

## 3.58.3 Member Function Documentation

3.58.3.1 int add.dataflow.sync.SHL.Compute ( int *data1*, int *data2* )

Method responsible for the component computation: in this case, it moves all the bits of the first parameter to the left N times, where N is equal to the value of the second parameter.

## Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

## Returns

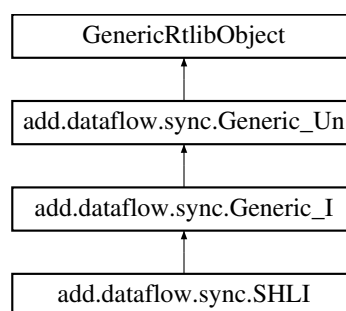
- Returns the result of the computation. In this case, it moves all the bits of the first parameter to the left N times, where N is equal to the value of second parameter.

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

- add/dataflow/sync/SHL.java

## 3.59 add.dataflow.sync.SHLI Class Reference

Inheritance diagram for add.dataflow.sync.SHLI:



## Public Member Functions

- [SHLI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

### 3.59.1 Detailed Description

**SHLI** component for the UFV **ADD** synchronous data flow simulator.

The component is responsible for moving all the bits of the input to the left N times, where N is equal to the value of a (immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.59.2 Constructor & Destructor Documentation

#### 3.59.2.1 `add.dataflow.sync.SHLI.SHLI ( )`

Object Constructor.

### 3.59.3 Member Function Documentation

#### 3.59.3.1 `int add.dataflow.sync.SHLI.Compute ( int data )`

Method responsible for the component computation: in this case, it moves all the bits of the parameter to the left N times, where N is equal to the value of a (immediate) constant.

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

#### Returns

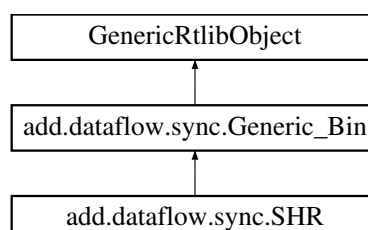
- Returns the result of the computation. In this case, it moves all the bits of the parameter to the left N times, where N is equal to the value of a (immediate) constant.

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

- `add/dataflow/sync/SHLI.java`

## 3.60 `add.dataflow.sync.SHR` Class Reference

Inheritance diagram for `add.dataflow.sync.SHR`:



## Public Member Functions

- [SHR](#) ()
- int [Compute](#) (int data1, int data2)

## Additional Inherited Members

### 3.60.1 Detailed Description

[SHR](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for moving all the bits of the first input to the right N times, where N is equal to the value of the value from the second input.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.60.2 Constructor & Destructor Documentation

#### 3.60.2.1 add.dataflow.sync.SHR.SHR ( )

Object Constructor.

### 3.60.3 Member Function Documentation

#### 3.60.3.1 int add.dataflow.sync.SHR.Compute ( int *data1*, int *data2* )

Method responsible for the component computation: in this case, it moves all the bits of the first parameter to the right N times, where N is equal to the value of the second parameter.

#### Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

#### Returns

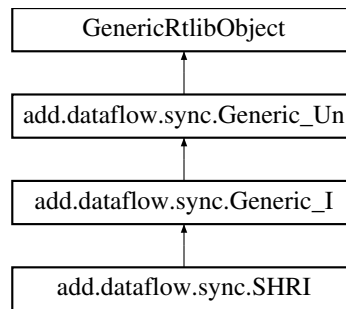
- Returns the result of the computation. In this case, it moves all the bits of the first parameter to the right N times, where N is equal to the value of the second parameter.

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

- add/dataflow/sync/SHR.java

## 3.61 add.dataflow.sync.SHRI Class Reference

Inheritance diagram for add.dataflow.sync.SHRI:



## Public Member Functions

- [SHRI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

### 3.61.1 Detailed Description

[SHRI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for moving all the bits of the input to the right N times, where N is equal to the value of a (immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.61.2 Constructor & Destructor Documentation

#### 3.61.2.1 `add.dataflow.sync.SHRI.SHRI ( )`

Object Constructor.

### 3.61.3 Member Function Documentation

#### 3.61.3.1 `int add.dataflow.sync.SHRI.Compute ( int data )`

Method responsible for the component computation: in this case, it moves all the bits of the parameter to the right N times, where N is equal to the value of a (immediate) constant.

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

**Returns**

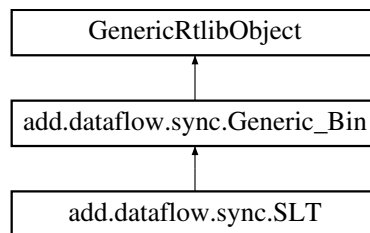
- Returns the result of the computation. In this case, it moves all the bits of the parameter to the right N times, where N is equal to the value of a (immediate) constant.

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

- add/dataflow/sync/SHRI.java

## 3.62 add.dataflow.sync.SLT Class Reference

Inheritance diagram for add.dataflow.sync.SLT:

**Public Member Functions**

- [SLT](#) ()
- int [Compute](#) (int data1, int data2)

**Additional Inherited Members**

### 3.62.1 Detailed Description

[SLT](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for returning the value 1 if the first input is less than the second one.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

**Version**

1.0

### 3.62.2 Constructor & Destructor Documentation

#### 3.62.2.1 add.dataflow.sync.SLT.SLT ( )

Object Constructor.

### 3.62.3 Member Function Documentation

#### 3.62.3.1 `int add.dataflow.sync.SLT.Compute ( int data1, int data2 )`

Method responsible for the component computation: in this case performs a comparison if the first parameter is less than the other one.



## Parameters

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

## Returns

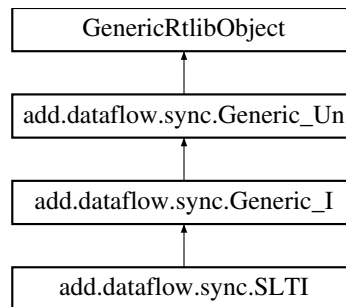
- Returns the result of the computation. In this case 1 or 0 depending on the comparison between the parameters.

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

- add/dataflow/sync/SLT.java

### 3.63 add.dataflow.sync.SLTI Class Reference

Inheritance diagram for add.dataflow.sync.SLTI:



## Public Member Functions

- [SLTI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

#### 3.63.1 Detailed Description

[SLTI](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for returning the value 1 if the input is less than the (immediate) constant.

Universidade Federal de Viçosa - MG - Brasil.

## Author

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

## Version

1.0

### 3.63.2 Constructor & Destructor Documentation

#### 3.63.2.1 `add.dataflow.sync.SLTi.SLTi ( )`

Object Constructor.

### 3.63.3 Member Function Documentation

#### 3.63.3.1 `int add.dataflow.sync.SLTi.Compute ( int data )`

Method responsible for the component computation: in this case performs a comparison if parameter is less than the (immediate) constant.

Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

Returns

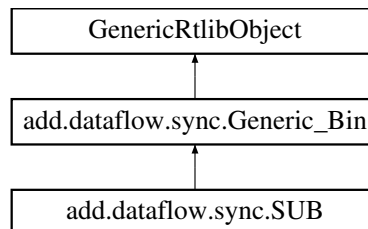
- Returns the result of the computation. In this case 1 or 0 depending on the comparison between the parameter and the constant.

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

- `add/dataflow/sync/SLTi.java`

## 3.64 `add.dataflow.sync.SUB` Class Reference

Inheritance diagram for `add.dataflow.sync.SUB`:



### Public Member Functions

- [SUB \(\)](#)
- `int Compute (int data1, int data2)`

### Additional Inherited Members

#### 3.64.1 Detailed Description

[SUB](#) component for the UFV [ADD](#) synchronous data flow simulator.

The component is responsible for subtracting the inputs.

Universidade Federal de Viçosa - MG - Brasil.

**Author**

Jeronimo Costa Penha - [jeronomopenha@gmail.com](mailto:jeronomopenha@gmail.com)  
 Ricardo Santos Ferreira - [cacauvicsa@gmail.com](mailto:cacauvicsa@gmail.com)

**Version**

1.0

**3.64.2 Constructor & Destructor Documentation****3.64.2.1 add.dataflow.sync.SUB.SUB ( )**

Object Constructor.

**3.64.3 Member Function Documentation****3.64.3.1 int add.dataflow.sync.SUB.Compute ( int *data1*, int *data2* )**

Method responsible for the component computation: in this case performs a subtraction of the parameters.

**Parameters**

<i>data1</i>	- Value to be used for the computation related to input 1.
<i>data2</i>	- Value to be used for the computation related to input 2.

**Returns**

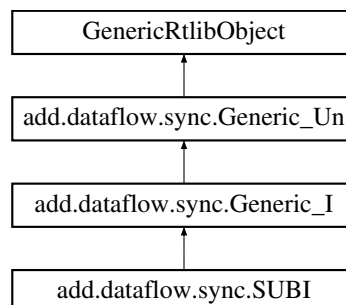
- Returns the result of the computation. In this case the value of the subtraction of the parameters.

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

- add/dataflow/sync/SUB.java

**3.65 add.dataflow.sync.SUBI Class Reference**

Inheritance diagram for add.dataflow.sync.SUBI:

**Public Member Functions**

- [SUBI](#) ()
- int [Compute](#) (int data)

## Additional Inherited Members

### 3.65.1 Detailed Description

**SUBI** component for the UFV **ADD** synchronous data flow simulator.

The component is responsible for subtracting the input by a constant (Immediate).

Universidade Federal de Viçosa - MG - Brasil.

#### Author

Jeronimo Costa Penha - [jeronimopenha@gmail.com](mailto:jeronimopenha@gmail.com)

Ricardo Santos Ferreira - [cacauvicosa@gmail.com](mailto:cacauvicosa@gmail.com)

#### Version

1.0

### 3.65.2 Constructor & Destructor Documentation

#### 3.65.2.1 `add.dataflow.sync.SUBI.SUBI ( )`

Object Constructor.

### 3.65.3 Member Function Documentation

#### 3.65.3.1 `int add.dataflow.sync.SUBI.Compute ( int data )`

Method responsible for the component computation: in this case performs a subtraction of the parameter by an (immediate) constant.

#### Parameters

<i>data</i>	- Value to be used for computing.
-------------	-----------------------------------

#### Returns

- Returns the result of the computation. In this case the value of the subtraction of the parameter by the constant.

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

- `add/dataflow/sync/SUBI.java`

# Index

ABS  
    add::dataflow::sync::ABS, 5  
ACC\_ADD  
    add::dataflow::sync::ACC\_ADD, 7  
ACC\_MAX  
    add::dataflow::sync::ACC\_MAX, 8  
ACC\_MIN  
    add::dataflow::sync::ACC\_MIN, 9  
ACC\_MUL  
    add::dataflow::sync::ACC\_MUL, 10  
ADD  
    add::dataflow::sync::ADD, 11  
ADDI  
    add::dataflow::sync::ADDI, 14  
AND  
    add::dataflow::sync::AND, 15  
ANDI  
    add::dataflow::sync::ANDI, 16  
Accumulate  
    add::dataflow::sync::ACC\_MAX, 8  
    add::dataflow::sync::ACC\_MIN, 9  
    add::dataflow::sync::ACC\_MUL, 10  
    add::dataflow::sync::Generic\_ACC, 30  
add.dataflow.sync.ABS, 5  
add.dataflow.sync.ACC\_ADD, 6  
add.dataflow.sync.ACC\_MAX, 7  
add.dataflow.sync.ACC\_MIN, 8  
add.dataflow.sync.ACC\_MUL, 9  
add.dataflow.sync.ADD, 10  
add.dataflow.sync.ADDI, 13  
add.dataflow.sync.AND, 14  
add.dataflow.sync.ANDI, 15  
add.dataflow.sync.BEQ, 17  
add.dataflow.sync.BEQI, 18  
add.dataflow.sync.BNE, 19  
add.dataflow.sync.BNEI, 20  
add.dataflow.sync.DIV, 25  
add.dataflow.sync.DIVI, 27  
add.dataflow.sync.Generic\_ACC, 29  
add.dataflow.sync.Generic\_Bin, 31  
add.dataflow.sync.Generic\_Comp, 34  
add.dataflow.sync.Generic\_Compl, 37  
add.dataflow.sync.Generic\_I, 40  
add.dataflow.sync.Generic\_IN, 43  
add.dataflow.sync.Generic\_OUT, 46  
add.dataflow.sync.Generic\_Un, 50  
add.dataflow.sync.IN\_1, 53  
add.dataflow.sync.IN\_16, 54  
add.dataflow.sync.IN\_2, 55  
add.dataflow.sync.IN\_32, 56  
add.dataflow.sync.IN\_4, 56  
add.dataflow.sync.IN\_8, 57  
add.dataflow.sync.MAX, 58  
add.dataflow.sync.MERGE, 59  
add.dataflow.sync.MIN, 60  
add.dataflow.sync.MOD, 61  
add.dataflow.sync.MODI, 62  
add.dataflow.sync.MUL, 64  
add.dataflow.sync.MULI, 66  
add.dataflow.sync.NOT, 67  
add.dataflow.sync.OR, 68  
add.dataflow.sync.ORI, 69  
add.dataflow.sync.OUT\_1, 70  
add.dataflow.sync.OUT\_16, 71  
add.dataflow.sync.OUT\_2, 72  
add.dataflow.sync.OUT\_32, 73  
add.dataflow.sync.OUT\_4, 74  
add.dataflow.sync.OUT\_8, 74  
add.dataflow.sync.REG, 75  
add.dataflow.sync.SHL, 76  
add.dataflow.sync.SHLI, 77  
add.dataflow.sync.SHR, 78  
add.dataflow.sync.SHRI, 79  
add.dataflow.sync.SLT, 81  
add.dataflow.sync.SLTI, 83  
add.dataflow.sync.SUB, 84  
add.dataflow.sync.SUBI, 85  
add.util.CONF\_READER, 23  
add::dataflow::sync::ABS  
    ABS, 5  
    Compute, 6  
add::dataflow::sync::ACC\_ADD  
    ACC\_ADD, 7  
add::dataflow::sync::ACC\_MAX  
    ACC\_MAX, 8  
    Accumulate, 8  
    Reseted, 8  
add::dataflow::sync::ACC\_MIN  
    ACC\_MIN, 9  
    Accumulate, 9  
    Reseted, 9  
add::dataflow::sync::ACC\_MUL  
    ACC\_MUL, 10  
    Accumulate, 10  
    Reseted, 10  
add::dataflow::sync::ADD  
    ADD, 11  
    Compute, 11

- add::dataflow::sync::ADDI
  - ADDI, [14](#)
  - Compute, [14](#)
- add::dataflow::sync::AND
  - AND, [15](#)
  - Compute, [15](#)
- add::dataflow::sync::ANDI
  - ANDI, [16](#)
  - Compute, [16](#)
- add::dataflow::sync::BEQ
  - BEQ, [17](#)
  - Compute, [18](#)
- add::dataflow::sync::BEQI
  - BEQI, [19](#)
  - Compute, [19](#)
- add::dataflow::sync::BNE
  - BNE, [20](#)
  - Compute, [20](#)
- add::dataflow::sync::BNEI
  - BNEI, [21](#)
  - Compute, [21](#)
- add::dataflow::sync::DIV
  - Compute, [26](#)
  - DIV, [26](#)
- add::dataflow::sync::DIVI
  - Compute, [27](#)
  - DIVI, [27](#)
- add::dataflow::sync::Generic\_ACC
  - Accumulate, [30](#)
  - evaluate, [30](#)
  - Generic\_ACC, [30](#)
  - Reseted, [31](#)
- add::dataflow::sync::Generic\_Bin
  - Compute, [32](#)
  - constructDynamicSymbol, [32](#)
  - constructPorts, [33](#)
  - evaluate, [33](#)
  - Generic\_Bin, [32](#)
  - initialize, [33](#)
  - needsDynamicSymbol, [33](#)
  - NotCompute, [33](#)
  - Reseted, [33](#)
  - Set\_Name, [33](#)
  - setString, [33](#)
  - setSymbol, [34](#)
  - Tick\_Down, [34](#)
  - Tick\_Up, [34](#)
  - write, [34](#)
- add::dataflow::sync::Generic\_Comp
  - Compute, [35](#)
  - constructDynamicSymbol, [35](#)
  - constructPorts, [35](#)
  - evaluate, [36](#)
  - Generic\_Comp, [35](#)
  - initialize, [36](#)
  - needsDynamicSymbol, [36](#)
  - Reseted, [36](#)
  - Set\_Name, [36](#)
  - setString, [36](#)
  - setSymbol, [36](#)
  - Tick\_Down, [36](#)
  - Tick\_Up, [37](#)
  - write, [37](#)
- add::dataflow::sync::Generic\_Compl
  - Compute, [38](#)
  - constructDynamicSymbol, [38](#)
  - constructPorts, [38](#)
  - evaluate, [38](#)
  - Generic\_Compl, [38](#)
  - initialize, [39](#)
  - mousePressed, [39](#)
  - needsDynamicSymbol, [39](#)
  - Reseted, [39](#)
  - Set\_Name, [39](#)
  - setString, [39](#)
  - setSymbol, [40](#)
  - Tick\_Down, [40](#)
  - Tick\_Up, [40](#)
  - write, [40](#)
- add::dataflow::sync::Generic\_I
  - Compute, [42](#)
  - Generic\_I, [42](#)
  - initialize, [42](#)
  - mousePressed, [42](#)
  - NotCompute, [42](#)
  - Reseted, [42](#)
  - write, [43](#)
- add::dataflow::sync::Generic\_IN
  - constructDynamicSymbol, [44](#)
  - constructPorts, [44](#)
  - constructStandardValues, [45](#)
  - evaluate, [45](#)
  - Generic\_IN, [44](#)
  - getDelay, [45](#)
  - initialize, [45](#)
  - needsDynamicSymbol, [45](#)
  - Set\_Name, [45](#)
  - setDelay, [45, 46](#)
  - setVectorIn, [46](#)
  - updateSymbol, [46](#)
  - wakeup, [46](#)
  - write, [46](#)
- add::dataflow::sync::Generic\_OUT
  - constructDynamicSymbol, [48](#)
  - constructPorts, [48](#)
  - constructStandardValues, [48](#)
  - evaluate, [48](#)
  - Generic\_OUT, [48](#)
  - getDelay, [48](#)
  - getDoneSignal, [48](#)
  - getVectorOut, [48](#)
  - initialize, [49](#)
  - needsDynamicSymbol, [49](#)
  - Set\_Name, [49](#)
  - setDelay, [49](#)
  - setVetor, [49](#)

- updateSymbol, [49](#)
- wakeup, [49](#)
- write, [50](#)
- add::dataflow::sync::Generic\_Un
  - Compute, [51](#)
  - constructDynamicSymbol, [51](#)
  - constructPorts, [51](#)
  - evaluate, [52](#)
  - Generic\_Un, [51](#)
  - initialize, [52](#)
  - needsDynamicSymbol, [52](#)
  - NotCompute, [52](#)
  - Reseted, [52](#)
  - Set\_Name, [52](#)
  - setString, [52](#)
  - setSymbol, [53](#)
  - Tick\_Down, [53](#)
  - Tick\_Up, [53](#)
  - write, [53](#)
- add::dataflow::sync::IN\_1
  - IN\_1, [54](#)
- add::dataflow::sync::IN\_16
  - IN\_16, [55](#)
- add::dataflow::sync::IN\_2
  - IN\_2, [55](#)
- add::dataflow::sync::IN\_32
  - IN\_32, [56](#)
- add::dataflow::sync::IN\_4
  - IN\_4, [57](#)
- add::dataflow::sync::IN\_8
  - IN\_8, [58](#)
- add::dataflow::sync::MAX
  - Compute, [59](#)
  - MAX, [59](#)
- add::dataflow::sync::MERGE
  - evaluate, [60](#)
  - MERGE, [60](#)
- add::dataflow::sync::MIN
  - Compute, [61](#)
  - MIN, [61](#)
- add::dataflow::sync::MOD
  - Compute, [62](#)
  - MOD, [62](#)
- add::dataflow::sync::MODI
  - Compute, [63](#)
  - MODI, [63](#)
- add::dataflow::sync::MUL
  - Compute, [65](#)
  - MUL, [64](#)
- add::dataflow::sync::MULI
  - Compute, [67](#)
  - MULI, [66](#)
- add::dataflow::sync::NOT
  - Compute, [68](#)
  - NOT, [68](#)
- add::dataflow::sync::OR
  - Compute, [69](#)
  - OR, [69](#)
- add::dataflow::sync::ORI
  - Compute, [70](#)
  - ORI, [70](#)
- add::dataflow::sync::OUT\_1
  - OUT\_1, [71](#)
- add::dataflow::sync::OUT\_16
  - OUT\_16, [72](#)
- add::dataflow::sync::OUT\_2
  - OUT\_2, [73](#)
- add::dataflow::sync::OUT\_32
  - OUT\_32, [73](#)
- add::dataflow::sync::OUT\_4
  - OUT\_4, [74](#)
- add::dataflow::sync::OUT\_8
  - OUT\_8, [75](#)
- add::dataflow::sync::REG
  - REG, [76](#)
- add::dataflow::sync::SHL
  - Compute, [77](#)
  - SHL, [77](#)
- add::dataflow::sync::SHLI
  - Compute, [78](#)
  - SHLI, [78](#)
- add::dataflow::sync::SHR
  - Compute, [79](#)
  - SHR, [79](#)
- add::dataflow::sync::SHRI
  - Compute, [80](#)
  - SHRI, [80](#)
- add::dataflow::sync::SLT
  - Compute, [82](#)
  - SLT, [81](#)
- add::dataflow::sync::SLTI
  - Compute, [84](#)
  - SLTI, [84](#)
- add::dataflow::sync::SUB
  - Compute, [85](#)
  - SUB, [85](#)
- add::dataflow::sync::SUBI
  - Compute, [86](#)
  - SUBI, [86](#)
- add::util::CONF\_READER
  - ReadConfig, [23](#)
- BEQ
  - add::dataflow::sync::BEQ, [17](#)
- BEQI
  - add::dataflow::sync::BEQI, [19](#)
- BNE
  - add::dataflow::sync::BNE, [20](#)
- BNEI
  - add::dataflow::sync::BNEI, [21](#)
- Compute
  - add::dataflow::sync::ABS, [6](#)
  - add::dataflow::sync::ADD, [11](#)
  - add::dataflow::sync::ADDI, [14](#)
  - add::dataflow::sync::AND, [15](#)
  - add::dataflow::sync::ANDI, [16](#)

- add::dataflow::sync::BEQ, 18
- add::dataflow::sync::BEQI, 19
- add::dataflow::sync::BNE, 20
- add::dataflow::sync::BNEI, 21
- add::dataflow::sync::DIV, 26
- add::dataflow::sync::DIVI, 27
- add::dataflow::sync::Generic\_Bin, 32
- add::dataflow::sync::Generic\_Comp, 35
- add::dataflow::sync::Generic\_Compl, 38
- add::dataflow::sync::Generic\_I, 42
- add::dataflow::sync::Generic\_Un, 51
- add::dataflow::sync::MAX, 59
- add::dataflow::sync::MIN, 61
- add::dataflow::sync::MOD, 62
- add::dataflow::sync::MODI, 63
- add::dataflow::sync::MUL, 65
- add::dataflow::sync::MULI, 67
- add::dataflow::sync::NOT, 68
- add::dataflow::sync::OR, 69
- add::dataflow::sync::ORI, 70
- add::dataflow::sync::SHL, 77
- add::dataflow::sync::SHLI, 78
- add::dataflow::sync::SHR, 79
- add::dataflow::sync::SHRI, 80
- add::dataflow::sync::SLT, 82
- add::dataflow::sync::SLTI, 84
- add::dataflow::sync::SUB, 85
- add::dataflow::sync::SUBI, 86
- constructDynamicSymbol
  - add::dataflow::sync::Generic\_Bin, 32
  - add::dataflow::sync::Generic\_Comp, 35
  - add::dataflow::sync::Generic\_Compl, 38
  - add::dataflow::sync::Generic\_IN, 44
  - add::dataflow::sync::Generic\_OUT, 48
  - add::dataflow::sync::Generic\_Un, 51
- constructPorts
  - add::dataflow::sync::Generic\_Bin, 33
  - add::dataflow::sync::Generic\_Comp, 35
  - add::dataflow::sync::Generic\_Compl, 38
  - add::dataflow::sync::Generic\_IN, 44
  - add::dataflow::sync::Generic\_OUT, 48
  - add::dataflow::sync::Generic\_Un, 51
- constructStandardValues
  - add::dataflow::sync::Generic\_IN, 45
  - add::dataflow::sync::Generic\_OUT, 48
- DIV
  - add::dataflow::sync::DIV, 26
- DIVI
  - add::dataflow::sync::DIVI, 27
- evaluate
  - add::dataflow::sync::Generic\_ACC, 30
  - add::dataflow::sync::Generic\_Bin, 33
  - add::dataflow::sync::Generic\_Comp, 36
  - add::dataflow::sync::Generic\_Compl, 38
  - add::dataflow::sync::Generic\_IN, 45
  - add::dataflow::sync::Generic\_OUT, 48
  - add::dataflow::sync::Generic\_Un, 52
- add::dataflow::sync::MERGE, 60
- exec\_FPGA
  - add::dataflow::DATAFLOW\_SYNC\_BASE\_SIMUL, 24
- Generic\_ACC
  - add::dataflow::sync::Generic\_ACC, 30
- Generic\_Bin
  - add::dataflow::sync::Generic\_Bin, 32
- Generic\_Comp
  - add::dataflow::sync::Generic\_Comp, 35
- Generic\_Compl
  - add::dataflow::sync::Generic\_Compl, 38
- Generic\_I
  - add::dataflow::sync::Generic\_I, 42
- Generic\_IN
  - add::dataflow::sync::Generic\_IN, 44
- Generic\_OUT
  - add::dataflow::sync::Generic\_OUT, 48
- Generic\_Un
  - add::dataflow::sync::Generic\_Un, 51
- getDelay
  - add::dataflow::sync::Generic\_IN, 45
  - add::dataflow::sync::Generic\_OUT, 48
- getDoneSignal
  - add::dataflow::sync::Generic\_OUT, 48
- getVectorOut
  - add::dataflow::sync::Generic\_OUT, 48
- IN\_1
  - add::dataflow::sync::IN\_1, 54
- IN\_16
  - add::dataflow::sync::IN\_16, 55
- IN\_2
  - add::dataflow::sync::IN\_2, 55
- IN\_32
  - add::dataflow::sync::IN\_32, 56
- IN\_4
  - add::dataflow::sync::IN\_4, 57
- IN\_8
  - add::dataflow::sync::IN\_8, 58
- initialize
  - add::dataflow::sync::Generic\_Bin, 33
  - add::dataflow::sync::Generic\_Comp, 36
  - add::dataflow::sync::Generic\_Compl, 39
  - add::dataflow::sync::Generic\_I, 42
  - add::dataflow::sync::Generic\_IN, 45
  - add::dataflow::sync::Generic\_OUT, 49
  - add::dataflow::sync::Generic\_Un, 52
- MAX
  - add::dataflow::sync::MAX, 59
- MERGE
  - add::dataflow::sync::MERGE, 60
- MIN
  - add::dataflow::sync::MIN, 61
- MOD
  - add::dataflow::sync::MOD, 62
- MODI



- add::dataflow::sync::MODI, [63](#)
- MUL
  - add::dataflow::sync::MUL, [64](#)
- MULI
  - add::dataflow::sync::MULI, [66](#)
- mousePressed
  - add::dataflow::sync::Generic\_CompI, [39](#)
  - add::dataflow::sync::Generic\_I, [42](#)
- NOT
  - add::dataflow::sync::NOT, [68](#)
- needsDynamicSymbol
  - add::dataflow::sync::Generic\_Bin, [33](#)
  - add::dataflow::sync::Generic\_Comp, [36](#)
  - add::dataflow::sync::Generic\_CompI, [39](#)
  - add::dataflow::sync::Generic\_IN, [45](#)
  - add::dataflow::sync::Generic\_OUT, [49](#)
  - add::dataflow::sync::Generic\_Un, [52](#)
- NotCompute
  - add::dataflow::sync::Generic\_Bin, [33](#)
  - add::dataflow::sync::Generic\_I, [42](#)
  - add::dataflow::sync::Generic\_Un, [52](#)
- OR
  - add::dataflow::sync::OR, [69](#)
- ORI
  - add::dataflow::sync::ORI, [70](#)
- OUT\_1
  - add::dataflow::sync::OUT\_1, [71](#)
- OUT\_16
  - add::dataflow::sync::OUT\_16, [72](#)
- OUT\_2
  - add::dataflow::sync::OUT\_2, [73](#)
- OUT\_32
  - add::dataflow::sync::OUT\_32, [73](#)
- OUT\_4
  - add::dataflow::sync::OUT\_4, [74](#)
- OUT\_8
  - add::dataflow::sync::OUT\_8, [75](#)
- REG
  - add::dataflow::sync::REG, [76](#)
- ReadConfig
  - add::util::CONF\_READER, [23](#)
- Reseted
  - add::dataflow::sync::ACC\_MAX, [8](#)
  - add::dataflow::sync::ACC\_MIN, [9](#)
  - add::dataflow::sync::ACC\_MUL, [10](#)
  - add::dataflow::sync::Generic\_ACC, [31](#)
  - add::dataflow::sync::Generic\_Bin, [33](#)
  - add::dataflow::sync::Generic\_Comp, [36](#)
  - add::dataflow::sync::Generic\_CompI, [39](#)
  - add::dataflow::sync::Generic\_I, [42](#)
  - add::dataflow::sync::Generic\_Un, [52](#)
- SHL
  - add::dataflow::sync::SHL, [77](#)
- SHLI
  - add::dataflow::sync::SHLI, [78](#)
- SHR
  - add::dataflow::sync::SHR, [79](#)
- SHRI
  - add::dataflow::sync::SHRI, [80](#)
- SLT
  - add::dataflow::sync::SLT, [81](#)
- SLTI
  - add::dataflow::sync::SLTI, [84](#)
- SUB
  - add::dataflow::sync::SUB, [85](#)
- SUBI
  - add::dataflow::sync::SUBI, [86](#)
- Set\_Name
  - add::dataflow::sync::Generic\_Bin, [33](#)
  - add::dataflow::sync::Generic\_Comp, [36](#)
  - add::dataflow::sync::Generic\_CompI, [39](#)
  - add::dataflow::sync::Generic\_IN, [45](#)
  - add::dataflow::sync::Generic\_OUT, [49](#)
  - add::dataflow::sync::Generic\_Un, [52](#)
- setDelay
  - add::dataflow::sync::Generic\_IN, [45](#), [46](#)
  - add::dataflow::sync::Generic\_OUT, [49](#)
- setString
  - add::dataflow::sync::Generic\_Bin, [33](#)
  - add::dataflow::sync::Generic\_Comp, [36](#)
  - add::dataflow::sync::Generic\_CompI, [39](#)
  - add::dataflow::sync::Generic\_Un, [52](#)
- setSymbol
  - add::dataflow::sync::Generic\_Bin, [34](#)
  - add::dataflow::sync::Generic\_Comp, [36](#)
  - add::dataflow::sync::Generic\_CompI, [40](#)
  - add::dataflow::sync::Generic\_Un, [53](#)
- setVectorIn
  - add::dataflow::sync::Generic\_IN, [46](#)
- setVetor
  - add::dataflow::sync::Generic\_OUT, [49](#)
- start\_FPGA
  - add::dataflow::DATAFLOW\_SYNC\_BASE\_SIMUL, [24](#), [25](#)
- Tick\_Down
  - add::dataflow::sync::Generic\_Bin, [34](#)
  - add::dataflow::sync::Generic\_Comp, [36](#)
  - add::dataflow::sync::Generic\_CompI, [40](#)
  - add::dataflow::sync::Generic\_Un, [53](#)
- Tick\_Up
  - add::dataflow::sync::Generic\_Bin, [34](#)
  - add::dataflow::sync::Generic\_Comp, [37](#)
  - add::dataflow::sync::Generic\_CompI, [40](#)
  - add::dataflow::sync::Generic\_Un, [53](#)
- updateSymbol
  - add::dataflow::sync::Generic\_IN, [46](#)
  - add::dataflow::sync::Generic\_OUT, [49](#)
- wakeup
  - add::dataflow::sync::Generic\_IN, [46](#)
  - add::dataflow::sync::Generic\_OUT, [49](#)
- write

add::dataflow::sync::Generic\_Bin, [34](#)  
add::dataflow::sync::Generic\_Comp, [37](#)  
add::dataflow::sync::Generic\_Compl, [40](#)  
add::dataflow::sync::Generic\_I, [43](#)  
add::dataflow::sync::Generic\_IN, [46](#)  
add::dataflow::sync::Generic\_OUT, [50](#)  
add::dataflow::sync::Generic\_Un, [53](#)