FSM IP Specification

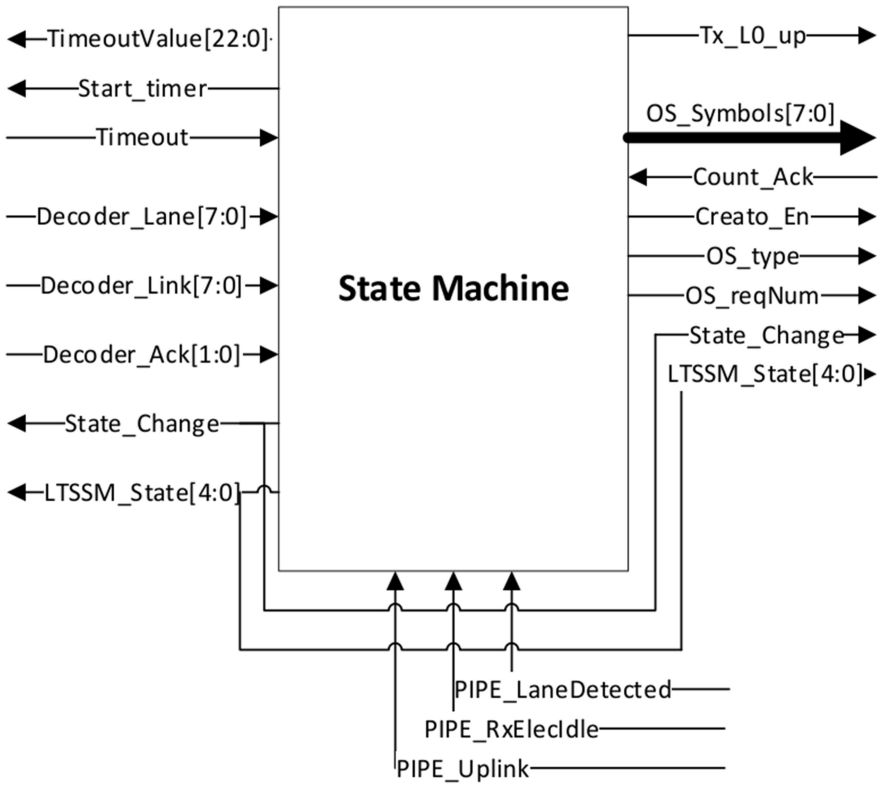
1. Introduction

1.1. Basic Operation

Link initialization and training is a hardware-based process controlled by the Physical Layer. The process configures and initializes a device's link and port so that normal packet traffic proceeds on the link. The full training process is automatically initiated by hardware after a reset and is managed by the LTSSM (Link Training and Status State Machine). The FSM sub-module of the LTSSM controls the flow of the link training process and the contents of the training sets used.

1.2. Block I/O Signals

Include a simple block diagram to show the IP I/O signals.



1.3. Interaction with Other Modules

The State machine functions as follows:

• Take the required signals from PIPE and Decoder for controlling the transitions between the states in the State machine module.

• Out the number of TS need to be sent to Order set creator block.

• Out the LTSSM states to PIPE interface, and get back the signals required for controlling the transitions between the states in the LTSSM block.

• After finishing the LTSSM training, L0_up signal indicates the normal operation.

2. I/O Signals Description

2.1. Inputs

Name	Width	Source	Description
CLK	1 bit	PIPE Clock	Global clock
i_resetn	1 bit	Synchronous reset	Global reset
i_timer_timeout	1 bit	**Timer**	Input from timer. • Timeout =1, time ≥ Timeout value • Timeout=0, otherwise.
i_LTSSM_UpDown	1 bit	PIPE	Indicates bit and symbol lock, ready to L0 state
i_OScreator_Ack	1 bit	Order set creator	To tell the State Machine the required number of ordered sets had been transmitted • OScreator_Ack = 1, Transmitted OS ≥ OS_reqNum • OScreator_Ack = 0, otherwise.
i_PHY_laneDetected	1 bit	PIPE	indicate successful detection of a receiver to LTSSM
i_PHY_RxElecIdle	1 bit	PIPE	Mirror the i_PHY_RxElecIdle coming from PHY to LTSSM
i_OSdecoder_Ack	2 bits	decoder	Acknowledgement signal on the number of TS order sets received to Control the transition between states.
i_OSdecoder_Link	8 bits	decoder	The link number decoded from TS1 and TS2
i_OSdecoder_Lane	8 bits	decoder	The lane number decoded from TS1 and TS2

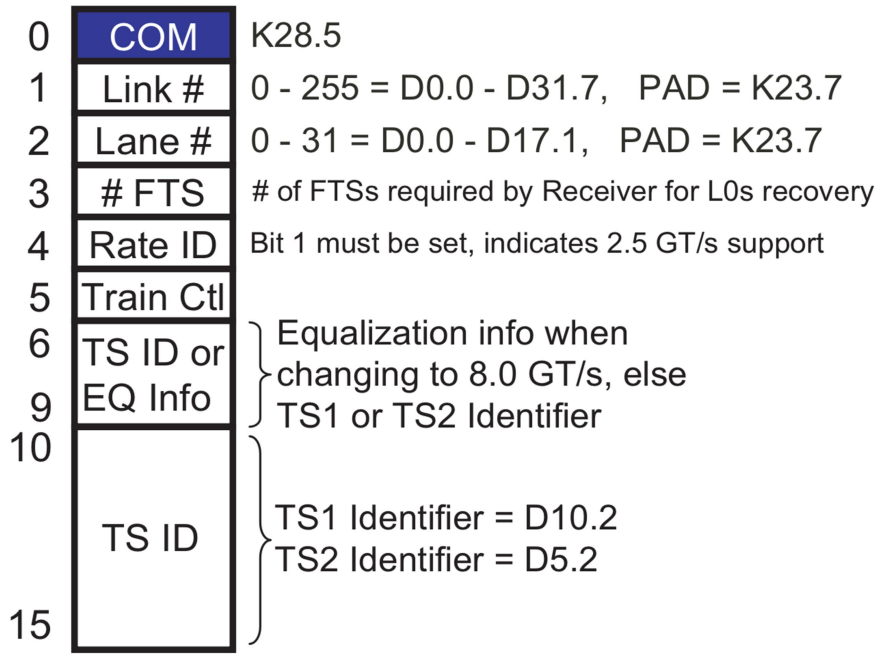
2.2. Outputs

Name	Width	Destination	Description
o_timer_timeoutValue	23 bits	Timer	Time needed by timer to timeout
o_timer_start	1 bit	Order set creator	start count the time: • timer_start=1, Start count • timer_start =0, Stop count and restart all the time data (when we start again, we will start from 0)
o_OScreator_Symbol0	8 bits	Order set creator	The order set symbol 0 data
o_OScreator_Symbol1	8 bits	Order set creator	The order set symbol 1 data
o_OScreator_Symbol2	8 bits	Order set creator	The order set symbol 2 data
o_OScreator_Symbol3	8 bits	Order set creator	The order set symbol 3 data
o_OScreator_Symbol4	8 bits	Order set creator	The order set symbol 4 data
o_OScreator_Symbol5	8 bits	Order set creator	The order set symbol 5 data
o_OScreator_Symbol6	8 bits	Order set creator	The order set symbol 6 data
o_OScreator_Symbol7	8 bits	Order set creator	The order set symbol 7 data
o_OScreator_Symbol8	8 bits	Order set creator	The order set symbol 8 data
o_OScreator_Symbol9	8 bits	Order set creator	The order set symbol 9 data
o_OScreator_Symbol10	8 bits	Order set creator	The order set symbol 10 data
o_OScreator_Symbol11	8 bits	Order set creator	The order set symbol 11 data
o_OScreator_Symbol12	8 bits	Order set creator	The order set symbol 12 data
o_OScreator_Symbol3	8 bits	Order set creator	The order set symbol 13 data
o_OScreator_Symbol14	8 bits	Order set creator	The order set symbol 14 data
o_OScreator_Symbol15	8 bits	Order set creator	The order set symbol 15 data
o_OScreator_enable	1 bit	Order set creator	To enable the creator to send the order set
o_OScreator_types	2 bits	Order set creator	Choose between TS creator or

			other order set creator.
o_OScreator_resetTScounter	1 bit	Order set creator	Number of repetitions of the order set
o_LTSSM_state	5 bits	PIPE & decoder	To update the current state
o_LTSSM_state_Change	1 bit	decoder	to update the decoder by the change in the state
o_LTSSM_L0_up	1 bit	Transmitter	To give the transmitter block permission to work in normal operation. o_LTSSM_L0_UP=1, indicate the L0 state.
o_OScreator_reqNum	11 bits	Order set creator	Number of repetitions of the order set.

3. Detailed Operation

The State machine module is responsible for determining the content of the training sets sent to the transmitter. The structure of the training sets is shown below.



There are two flavors of training sets, TS1 and TS2, the two flavors are used as a handshaking method two synchronize between the downstream and upstream ports of the link. Each training set consists of 16 8bit symbols sent in series.

1- **Symbol 0 (COM):** The K28.5 character identifies the start of an Ordered-Set. The receiver uses this character to achieve Bit Lock and Symbol Lock By locating this character on a synchronized transmission of TS1 or TS2 Ordered-Sets on a multi-Lane Link, the receiver can de-skew the Lanes.

2- **Symbol 1 (Link #):** In the early stages of Link training, when the TS1 and TS2 Ordered-Sets are exchanged, this field contains the PAD symbol (transmitted

as a null symbol). During the configuration state of the LTSSM, this field contains an assumed Link Number. TS1 and TS2 Ordered-Sets driven from different ports of a switch contain different Link Numbers.

3- **Symbol 2 (Lane #):** this field contains an assumed Lane Number for each Lane of a Link. The TS1 and TS2 Ordered-Sets driven on each Lane of a given link contain different numbers.

4- **Symbol 3 (N_FTS):** Contains the number of Fast Training Sequences. The exchange of FTS Ordered-Sets is used to achieve Bit Lock and Symbol Lock when exiting from the L0s to the L0 power state (not used in this implementation, hard wired to 256).

5- **Symbol 4 (Rate ID):** Each device informs its neighbor what data transfer rate it supports. A value of D2.0 indicates a 2.5Gbits/s transfer rate, while other values are currently reserved.

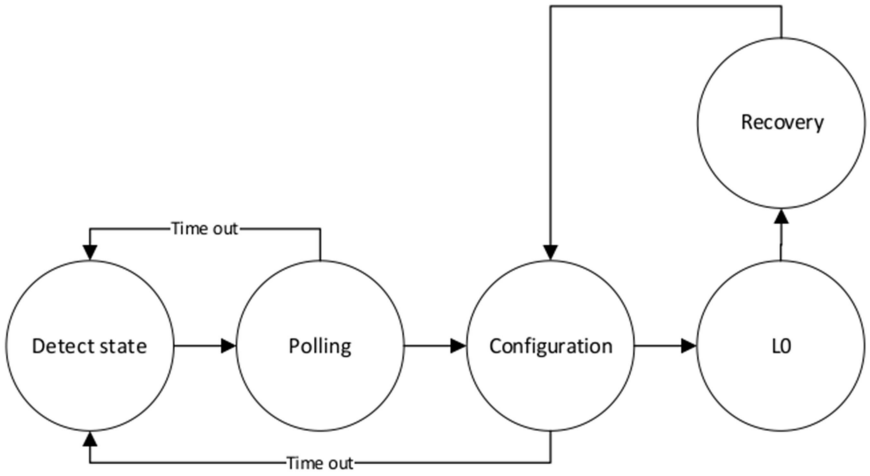
6- **Symbol 5 (Training Control):** A device that sends TS1 and TS2 Ordered-Sets uses this symbol to communicate additional information such as:

- Bit 0, when set, indicates Hot Reset.
- Bit 1, when set, indicates Link Disable.
- Bit 2, when set, indicates Enable Loopback.
- Bit 3, when set, indicates disable scrambling.
- The remainder of the bits are reserved.

Only one bit can be set in this field per Ordered-Set

7- Symbol 6-15 (Training Sequence ID): Driven with D10.2 for TS1 and D5.2 for TS2.

The figure below shows us the states of the link training and status state machine (LTSSM), each state consists of substates, the LTSSM consists mainly of 11 top-level states: Detect, Polling, Configuration, Recovery, L0, L0s, L1, L2, hot reset, Loopback and Disable. but in our design, we decide to implement only 5 states: Detect, Polling, Configuration, Recovery and L0. The normal flow of link training is Detect → Polling → Configuration → L0.

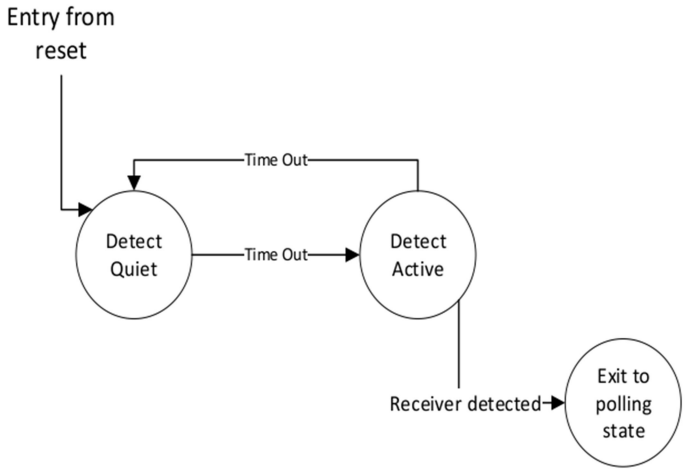


Detect state description:

The detect state is the first state after the reset, in this state a device is electrically detects a receiver. Also, we can enter this state from the polling and configuration state. The detect state composed of two substates shown in Figure 24: detect quiet and detect active.

- Detect quiet is the initial substate, in this substate the transmitter in electrical idle, the intendent data rate 2.5 GT/s (Gen1 speed) and the Linkup=0 which mean the link is not operational.
- Detect active, this substate entered from the detect quiet, in this substate the transmitter tests if the receiver is connected or no. the next substate after the detect active, when receiver detected is the polling active substate.

The substates of the detect state are shown below.

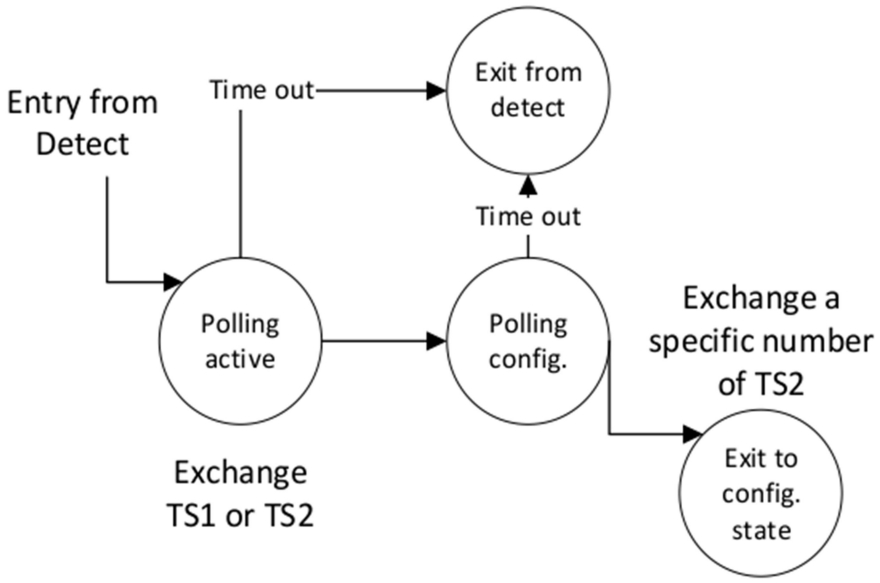


Polling state description

The polling state enters from the detect active after the receiver detection, in this state the transmitter starts to send TS1 and TS2.The goal of this state is to achieve the bit lock, symbol lock and learn the available lane data rate. This state composed of three substates but we implement only the first two: polling active and polling configuration.

- Polling active: in this substate, transmitter sends a minimum number of TS1s with pad lane and link number on all detected lanes.
- Polling configuration: in this substate, transmitter will stop sending TS1s and start sending TS2s with lane and link number (the purpose of sending TS2s is to advertise the link partner that this device is ready to proceed to the next state).

The substates of the polling state are shown below.



Configuration state description

The configuration state enters from polling configuration, or from the recovery state in this state the upstream and the downstream continues exchange TS1s and TS2s. the goal of this state is to determine the link width and to assign the lane numbers. This state composed of 6 substates: configuration linkwidth start, configuration linkwidth accept, configuration lanenum wait, configuration lanenum accept, configuration complete and configuration idle. In this state the downstream device differs from upstream device.

- **Configuration linkwidth start:**

Downstream device: The downstream now is the leader device, will initiate a link number and starts to sends TS1s with a non-PAD link number while the lane number remains PAD.

Upstream device: The upstream now is the follower and goes back to sending TS1s with mirrored link number and lane PAD.

• **Configuration linkwidth accept:**

Downstream device: The downstream now, will send TS1s with a non-PAD link number and Pad Lane number.

Upstream device: The upstream now send back TS1s with the saved link number selected and PAD lane number.

• **Configuration lanenum wait:**

Downstream device: The downstream will send TS1s with a non-PAD link number and non-Pad Lane number.

Upstream device: The upstream will sends TS1s with a non-PAD link number and mirrored Lane number.

• **Configuration lanenum Accept:**

Downstream device: The downstream will continue to sends TS1s with a non-PAD link number and non-Pad Lane number.

Upstream device: The upstream will sends TS1s with a non-PAD link number and non-Pad Lane number

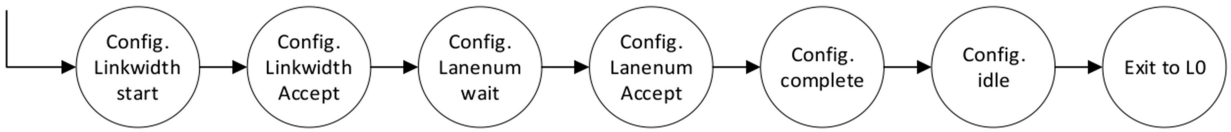
• **Configuration complete:**

Downstream device: The downstream starts to sends TS2s with a non-PAD link number and non-Pad Lane number.

Upstream device: The upstream will sends TS2s with a non-PAD link number and non-Pad Lane number.

• **Configuration idle:** during this substate the devices sends idle data and wait for minimum number of idle data after receiving it the link can transition to L0, where the LinkUp=1.

The substates of the configuration state are shown below.



L0 state description:

L0 state is the normal full-operational link state, during which Logical idle, TLPs and DLLPs are exchange between links. The physical layer notifies the upper layers that the link is ready for operation by setting LinkUp=1. During L0, there is a timer running with period of 590, when the timer triggers, a SKP ordered set is created and sent to the OS_buffer in the transmitter. These SKP ordered sets are added or removed by the receiver to compensate for any differences in the clock period between the transmitter and receiver.

The recovery state was not implemented in this design.

LTSSM states exit conditions:

According to the PCI-e standard, the exit condition for each state is as follows:

States	Substates:	Condition:
Detect	Detect.Quiet to Detect.Active	12 ms timeout or lane exit electrical idle
	Detect.Acive to Detect.Quiet	12 ms timeout, no lane detected
	Detect.Acive to Polling	Lane detected
Polling	Polling.Active to Detect.Quiet	24 ms timeout
	Polling.Active to Polling.Config	24 TS1 sent with training control=0, 8 received

	Polling.Config to Detect.Quiet	48 ms timeout
	Polling.Config to Configration	16 TS2 sent, 8 received

Starting from configuration state the exit condition will differ from downstream device Or Upstream device.

UP/Down		Exit condition
Downstream	Config.Linkwidth.start to Config.Linkwidth.Accept	Received 2 TS1 with non pad link number
	Config.Linkwidth.Accept to Config.Lanenum.wait	Received 2 TS1 with the link saved and Pad lane
	Config.Lanenum.wait to Config.Lanenum.Accept	Received 2 TS1 with the link saved and non-pad lane
	Config.Lanenum.Accept to Config.Complete	Received 2 TS1 with the link saved and lane saved
Upstream	Config.Linkwidth.start to Config.Linkwidth.Accept	Received 2 TS1 with non-pad link number
	Config.Linkwidth.Accept to Config.Lanenum.wait	Received 2 TS1 with the link saved and non-Pad lane
	Config.Lanenum.wait to Config.Lanenum.Accept	Received 2 TS2 with the link saved and Lane saved
	Config.Lanenum.Accept to Config.Complete	Received 2 TS2 with the link saved and Lane saved

The Config.complete and Config.Idle states have a special transition condition to ensure that there are enough ordered sets transmitted to the PIPE to allow for both the upstream and downstream ports to exit these states.

Both	Config.complete to Config.idle	Detect 1 TS2 with link and lane saved, then reset OS_creator counter. Then wait until another 8 TS2 packets are detected and the OS_creator has finished creating 16 TS2 packets.
	Config.idle to L0	Detect 1 IDLE packet, then reset OS_creator counter. Then wait until another 8 IDLE packets are detected and the OS_creator has finished creating 16 IDLE packets.

4. Configuration

4.1. Configuration Sequence

Only a reset sequence is needed to initialize the module, which resets to the "detect quite" of the FSM.

4.2. Latency

All outputs take only one clock cycle to react to the ACK signals coming from the other LTSSM (creator, decoder, timer, etc.) although these ACK signals themselves will have their own latency during normal operation depending on their source module.

4.3. Input Format and Constraints

There are no specific constraints for the input to this module.

5. Sequences and Waveforms

The following sequences demonstrate the different operating functions of the FSM. Throughout these sequences, the Encoding of the FSM states is as shown below:

State name:	Encoding (decimal)	Encoding (binary)
Detect_Quiet	0	5'b00000
Detect_Active	1	5'b00001
Polling_Active	2	5'b00010
Polling_configuration	3	5'b00011
Config_Linkwidth_start	4	5'b00100
Config_Linkwidth_Accept	5	5'b00101
Config_Lanenum_wait	6	5'b00110
Config_Lanenum_Accept	7	5'b00111
Config_complete	8	5'b01000
Config_idle	9	5'b01001
L0	10	5'b01010

Thoroughly explain all the sequences that can be inserted to the IP. Include all signal values (inputs and outputs) and waveforms.

5.1. Reset Sequence:

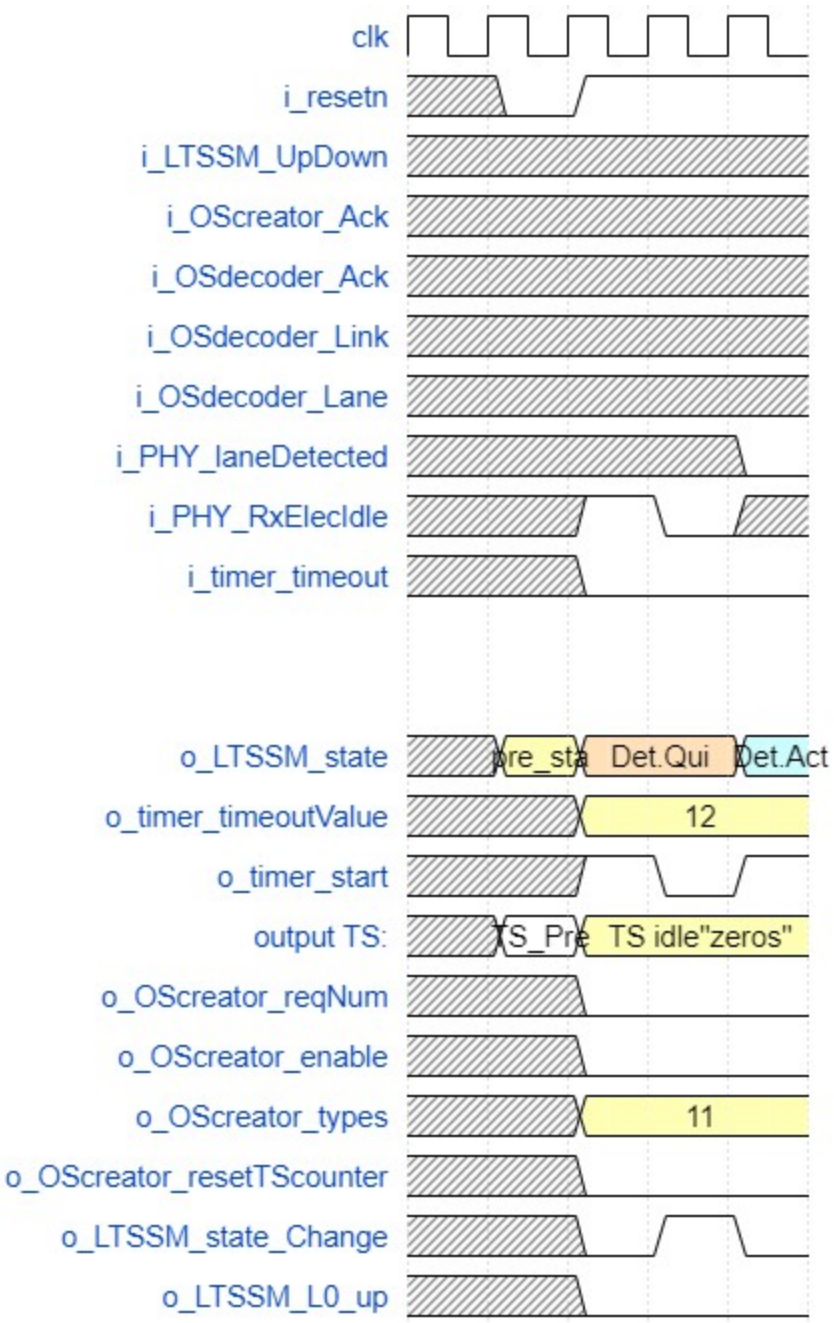
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4
i_resetn	0	1	1	1
i_timer_timeout	x	0	0	0
i_PHY_RxElecIdle	x	1	0	x
i_PHY_laneDetected	x	x	x	0

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4
o_LTSSM_state	State before Reset	Detect.quite	Detect.quite	Detect.Active
o_timer_timeoutValue	x	12	12	12
o_timer_start	x	1	0	1
o_LTSSM_L0_up	x	0	0	0
o_OScreator_reqNum	x	0	0	0
o_OScreator_types	x	3	3	3
o_OScreator_resetTScounter	x	0	0	0
o_LTSSM_stateChange	x	0	1	0
o_OScreator_enable	x	0	0	0
Output TS (Symbols)	Idle	Idle	Idle	Idle

Waveform(s):


clk
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
o_LTSSM_state
Det.Qui
Det.Act
o_timer_timeoutValue
12
o_timer_start
output TS:
TS_Pre
TS idle"zeros"
o_OScreator_reqNum
o_OScreator_enable
o_OScreator_types
11
o_OScreator_resetTScounter
o_LTSSM_state_Change
o_LTSSM_L0_up

5.2. Timeout Sequence:

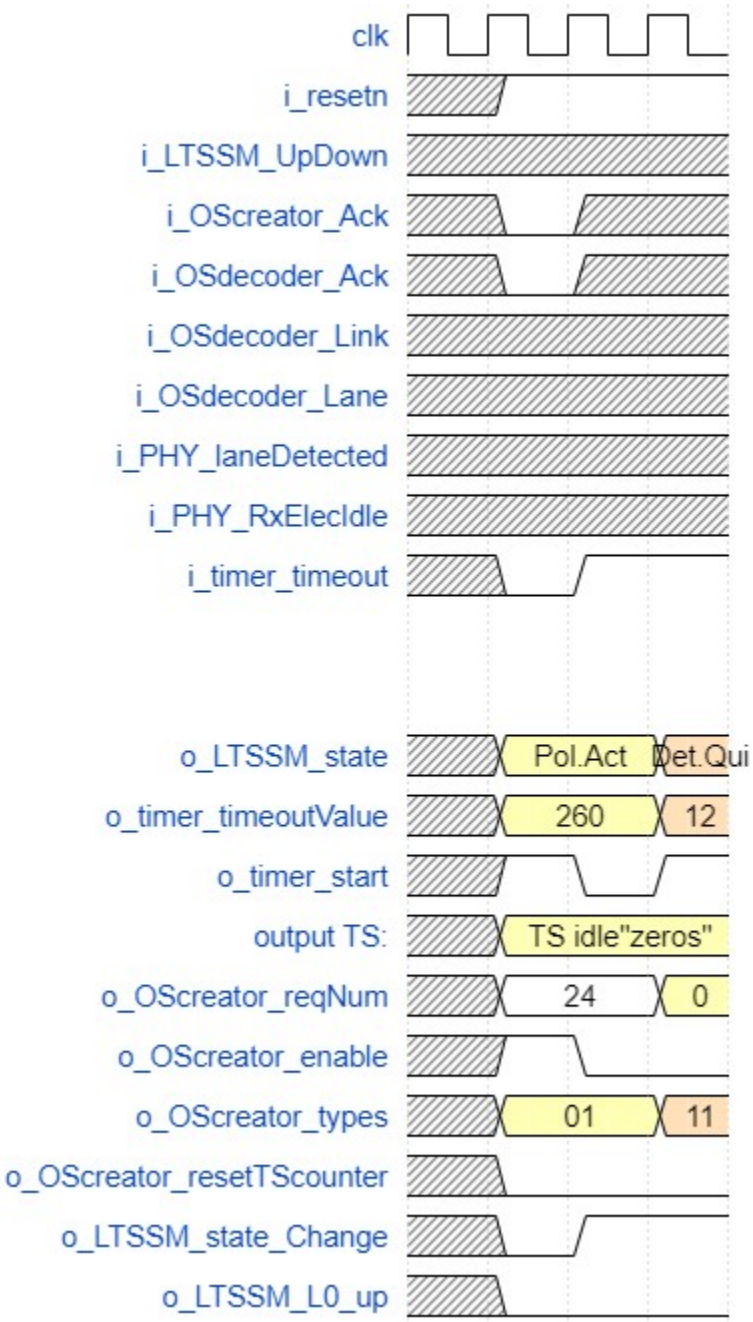
Inputs:

Signal:	Value 1	Value 2	Value 3
i_resetn	1	1	1
i_timer_timeout	0	1	1

Outputs:

Signal:	Value 1	Value 2	Value 3
o_LTSSM_state	Polling.active	Polling.active	Detect. quite
o_timer_timeoutValue	260	260	12
o_timer_start	1	0	1
o_LTSSM_L0_up	0	0	0
o_OScreator_reqNum	24	24	0
o_OScreator_types	1	1	3
o_OScreator_resetTScounter	0	0	0
o_LTSSM_stateChange	0	1	1
o_OScreator_enable	1	0	0
Output TS (Symbols)	TS1	TS1	Idle

Waveform(s):


clk
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
o_LTSSM_state
Pol.Act
Det.Qui
o_timer_timeoutValue
260
12
o_timer_start
output TS:
TS idle"zeros"
o_OScreator_reqNum
24
0
o_OScreator_enable
o_OScreator_types
01
11
o_OScreator_resetTScounter
o_LTSSM_state_Change
o_LTSSM_L0_up

5.3. Detect quiet Sequence:

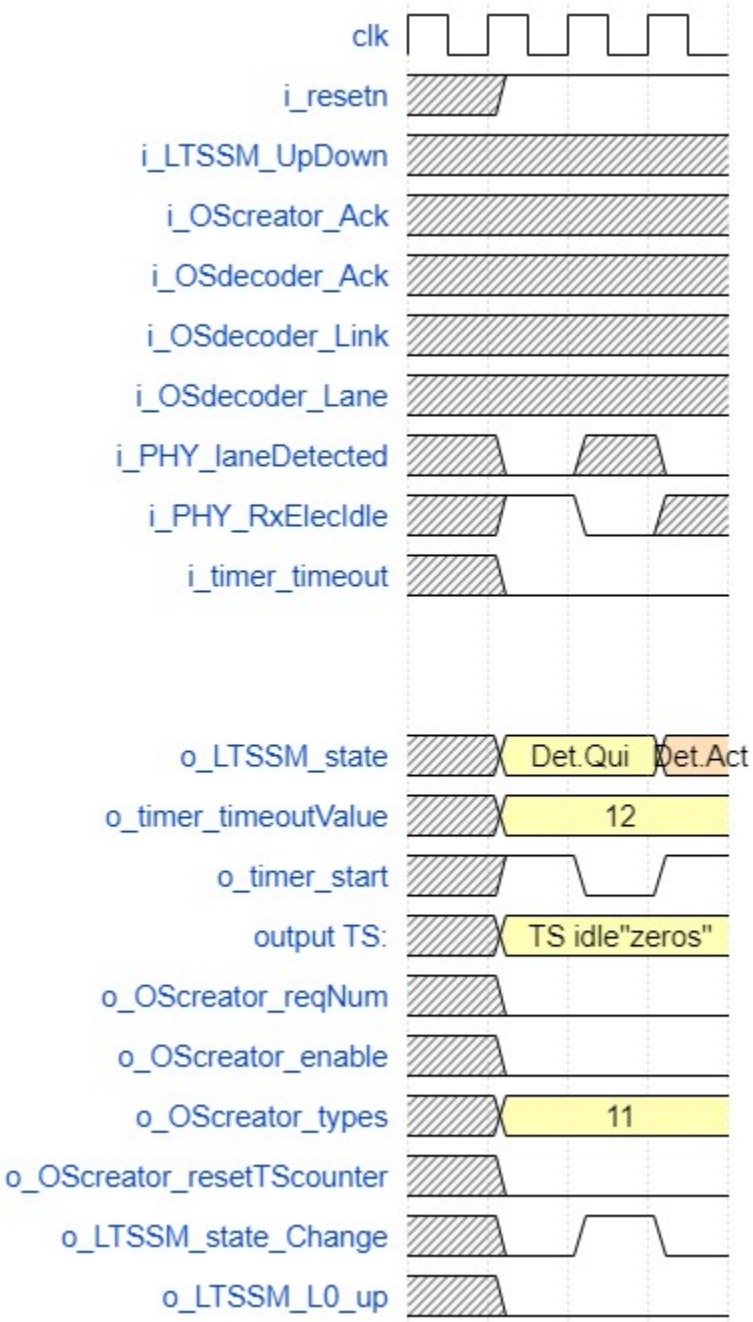
Inputs:

Signal:	Value 1	Value 2	Value 3
i_resetn	1	1	1
i_PHY_RxElecIdle	1	0	x
i_PHY_laneDetected	0	X	0

Outputs:

Signal:	Value 1	Value 2	Value 3
o_LTSSM_state	Detect.quite	Detect.quite	Detect.active
o_timer_timeoutValue	12	12	12
o_timer_start	1	0	1
o_LTSSM_L0_up	0	0	0
o_OScreator_reqNum	0	0	0
o_OScreator_types	3	3	3
o_OScreator_resetTScounter	0	0	0
o_LTSSM_stateChange	0	1	0
o_OScreator_enable	0	0	0
Output TS (Symbols)	Idle	Idle	Idle

Waveform(s):


clk
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
o_LTSSM_state
Det.Qui
Det.Act
o_timer_timeoutValue
12
o_timer_start
output TS:
TS idle"zeros"
o_OScreator_reqNum
o_OScreator_enable
o_OScreator_types
11
o_OScreator_resetTScounter
o_LTSSM_state_Change
o_LTSSM_L0_up

5.4. Detect active Sequence:

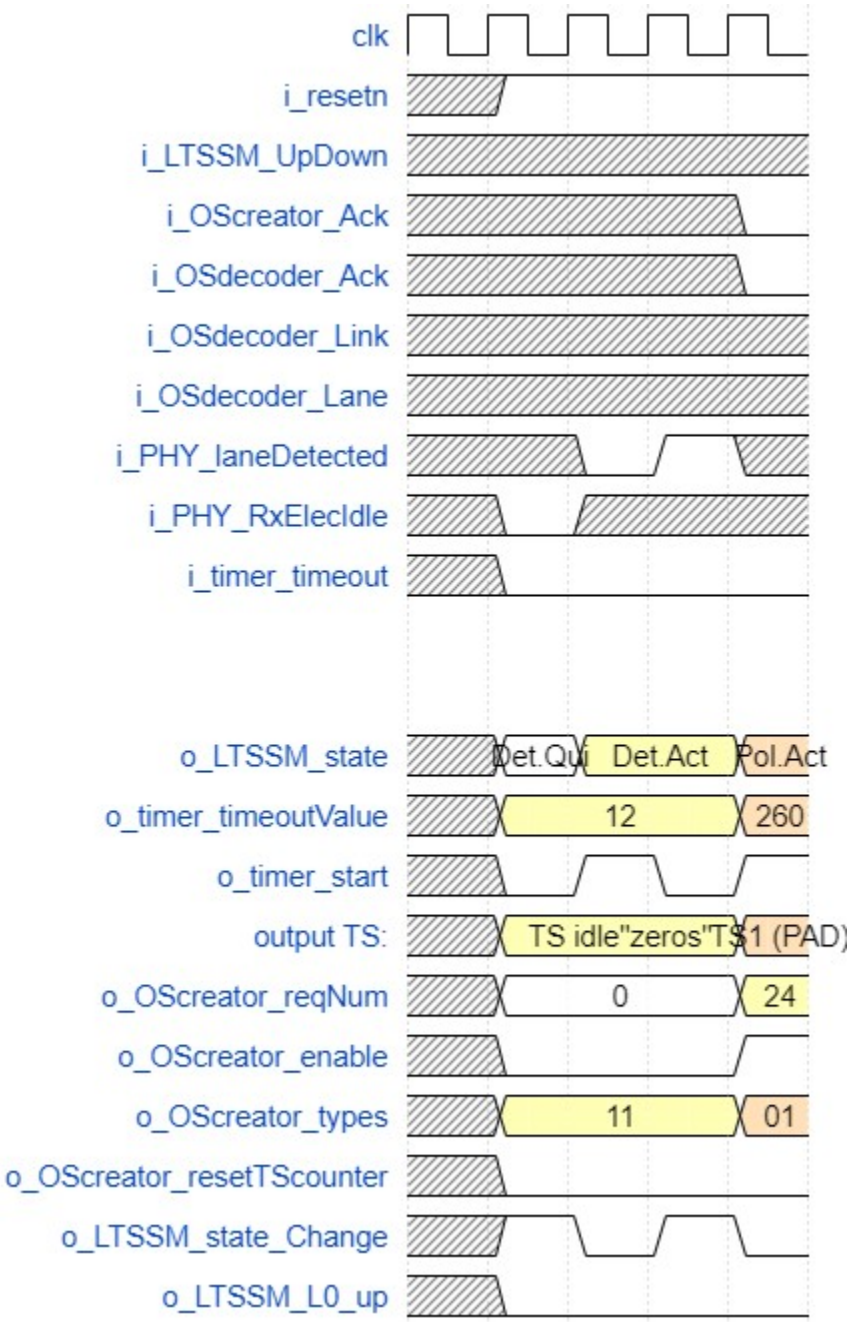
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4
i_resetn	1	1	1	1
i_PHY_RxElecIdle	0	x	X	x
i_PHY_laneDetected	x	0	1	x
i_OScreator_Ack	x	x	X	0
i_OSdecoder_Ack	x	x	X	00

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4
o_LTSSM_state	Detect.quite	Detect.active	Detect.active	Polling.Active
o_timer_timeoutValue	12	12	12	260
o_timer_start	0	1	0	1
o_LTSSM_L0_up	0	0	0	0
o_OScreator_reqNum	0	0	0	24
o_OScreator_types	3	3	3	1
o_OScreator_resetTScounter	0	0	0	0
o_LTSSM_stateChange	1	0	1	0
o_OScreator_enable	0	0	0	1
Output TS (Symbols)	Idle	Idle	Idle	TS1(PAD, PAD)

Waveform(s):


clk
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
o_LTSSM_state
Det.Act
Pol.Act
o_timer_timeoutValue
12
260
o_timer_start
output TS:
TS idle"zeros"TS1 (PAD)
o_OScreator_reqNum
0
24
o_OScreator_enable
o_OScreator_types
11
01
o_OScreator_resetTScounter
o_LTSSM_state_Change
o_LTSSM_L0_up

5.5. Polling active Sequence:

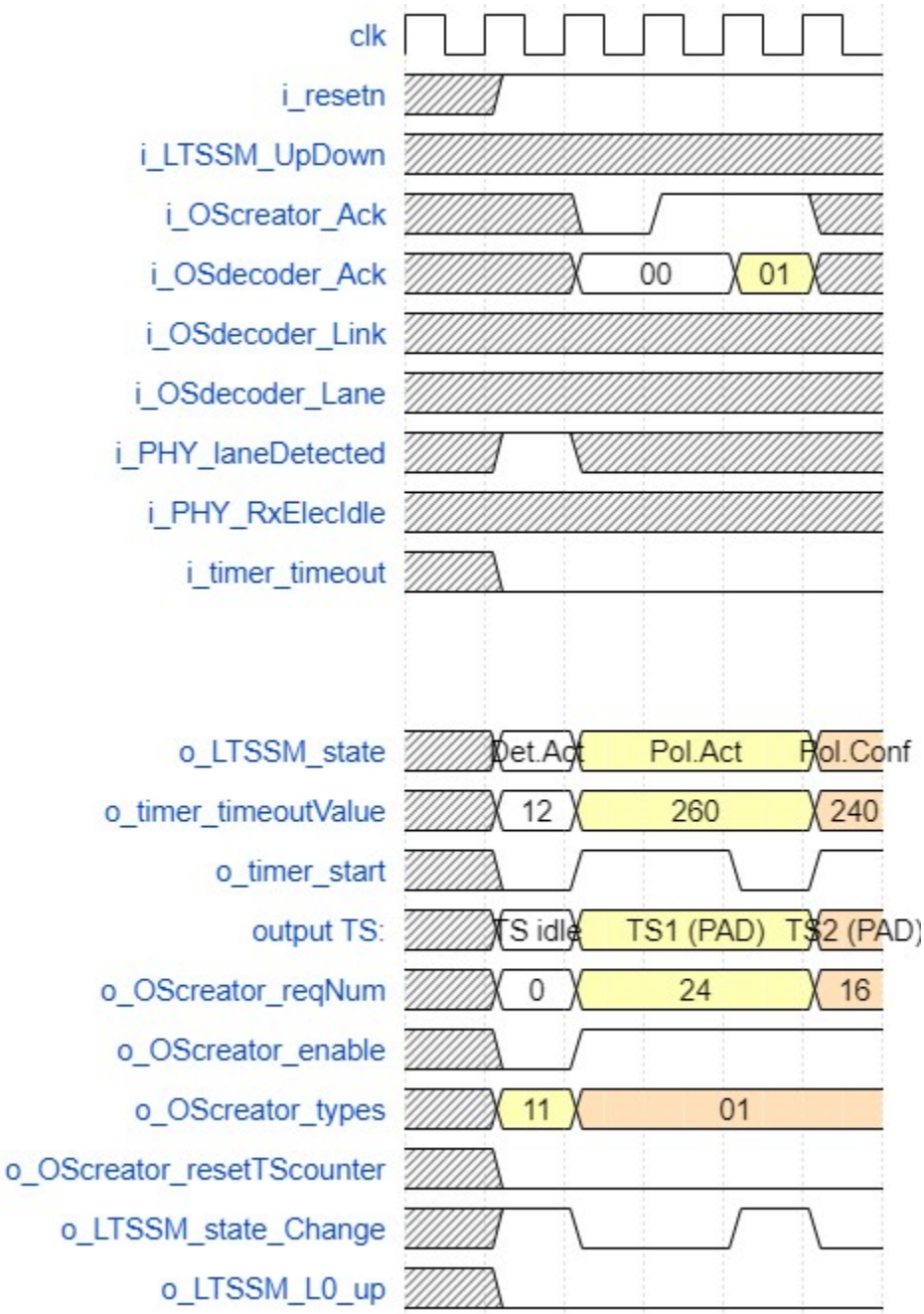
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_resetn	1	1	1	1	1
i_PHY_laneDetected	1	x	x	x	X
i_OScreator_Ack	x	0	1	1	X
i_OSdecoder_Ack	x	00	00	01	X

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	Det.Active	Pol.Active	Pol.Active	Pol.Active	Pol_config
o_timer_timeoutValue	12	260	260	260	240
o_timer_start	0	1	1	0	1
o_LTSSM_L0_up	0	0	0	0	0
o_OScreator_reqNum	0	24	24	24	16
o_OScreator_types	3	1	1	1	1
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0
o_OScreator_enable	0	1	1	1	1
Output TS (Symbols)	Idle	TS1(PAD, PAD)	TS1(PAD, PAD)	TS1(PAD, PAD)	TS2(PAD, PAD)

Waveform(s):


clk
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
00
01
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
o_LTSSM_state
Det.Act
Pol.Act
Pol.Conf
o_timer_timeoutValue
12
260
240
o_timer_start
output TS:
TS idle
TS1 (PAD)
TS2 (PAD)
o_OScreator_reqNum
0
24
16
o_OScreator_enable
o_OScreator_types
11
01
o_OScreator_resetTScounter
o_LTSSM_state_Change
o_LTSSM_L0_up

5.6. Polling configuration Sequence:

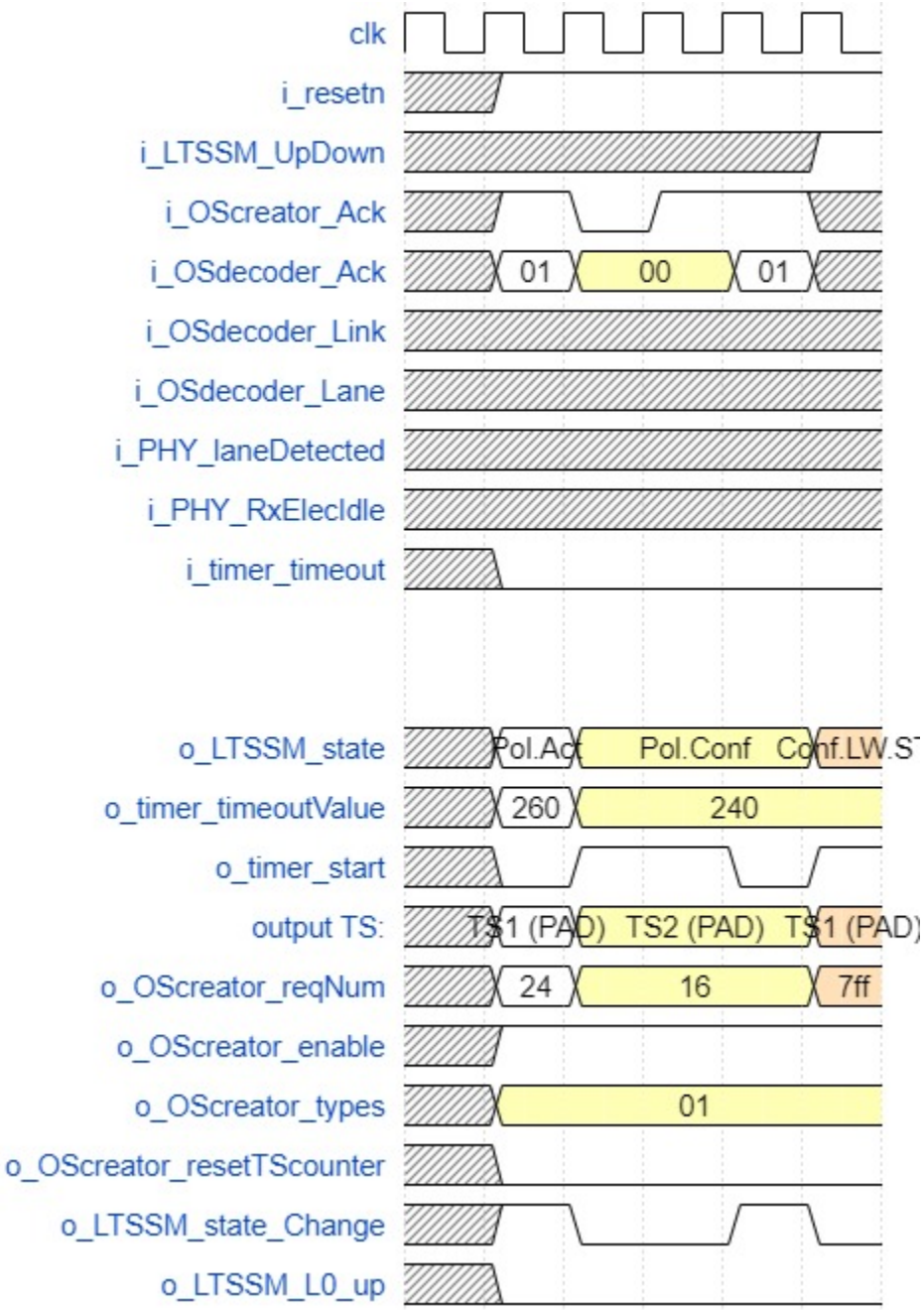
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_resetn	1	1	1	1	1
i_OScreator_Ack	1	0	1	1	X
i_OSdecoder_Ack	01	00	00	01	X
I_LTSSM_UpDown	x	x	x	x	1

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	Pol.Active	Pol_config	Pol_config	Pol_config	conf.LW.ST
o_timer_timeoutValue	260	240	240	240	240
o_timer_start	0	1	1	0	1
o_LTSSM_L0_up	0	0	0	0	0
o_OScreator_reqNum	24	16	16	26	11'h7ff
o_OScreator_types	1	1	1	1	1
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0
o_OScreator_enable	1	1	1	1	1
Output TS (Symbols)	TS1(PAD, PAD)	TS2(PAD, PAD)	TS2(PAD, PAD)	TS2(PAD, PAD)	TS1(Link#,PAD)

Waveform(s):


clk
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
01
00
01
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
o_LTSSM_state
Pol.Act
Pol.Conf
Conf.LW.ST
o_timer_timeoutValue
260
240
o_timer_start
output TS:
TS1 (PAD)
TS2 (PAD)
TS1 (PAD)
o_OScreator_reqNum
24
16
7ff
o_OScreator_enable
o_OScreator_types
01
o_OScreator_resetTScounter
o_LTSSM_state_Change
o_LTSSM_L0_up

5.7. Configuration linkwidth start (downstream) Sequence:

Description:

This sequence shows the transition process into the Config.linkwidth.start state, the operation within it, and the transition process out of it (for downstream ports).

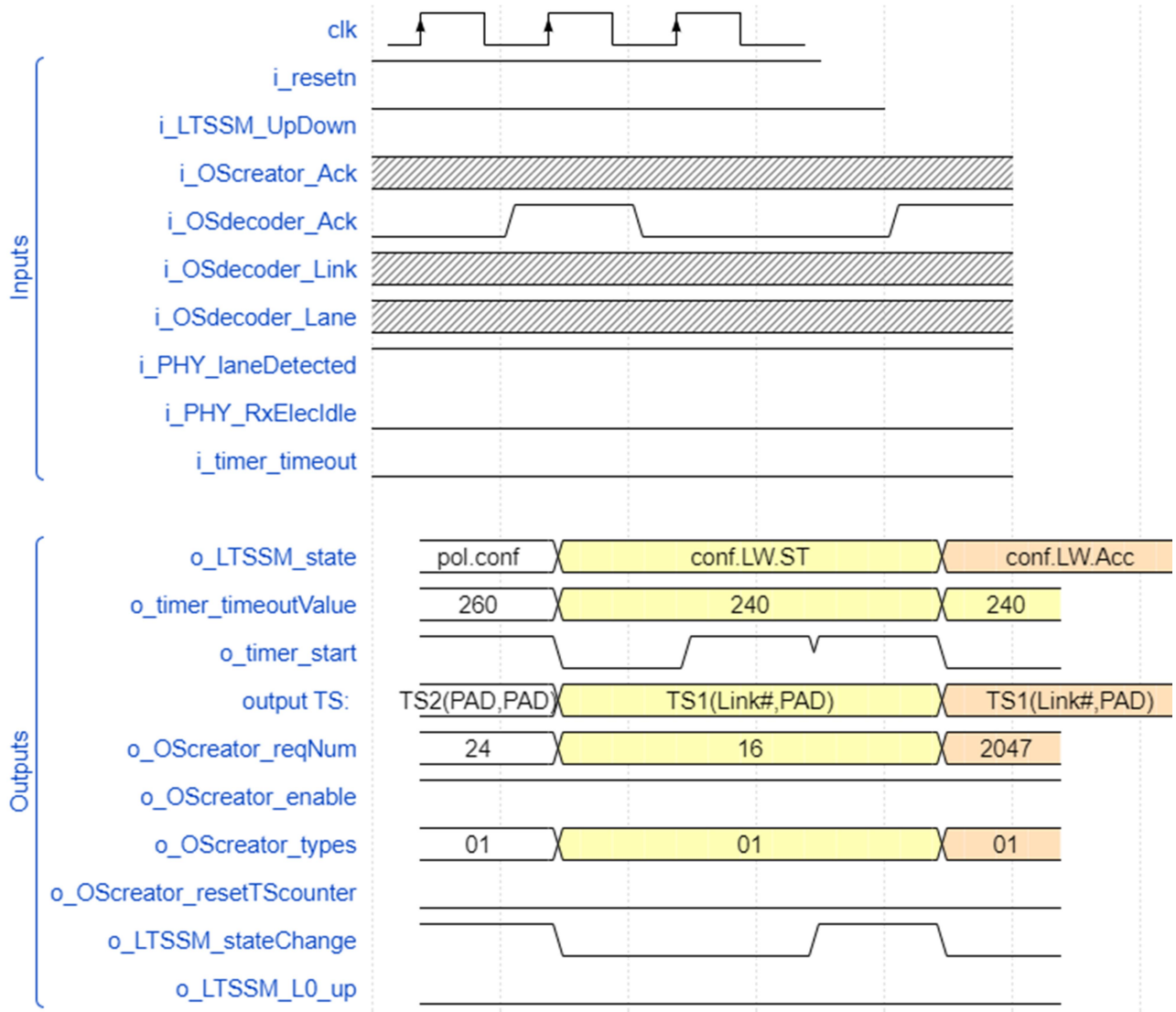
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	X	x	x	x	x
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	X	x	x	x	x
i_OSdecoder_Lane	X	x	x	x	x

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	pol.conf	conf.LW.ST	conf.LW.ST	conf.LW.ST	conf.LW.Acc
o_timer_timeoutValue	260	240	240	240	240
o_timer_start	1	0	1	1	0
output TS:	TS2(PAD,PAD)	TS1(Link#,PAD)	TS1(Link#,PAD)	TS1(Link#,PAD)	TS1(Link#,PAD)
o_OScreator_reqNum	24	16	16	16	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Inputs
o_LTSSM_state
pol.conf
conf.LW.ST
conf.LW.Acc
o_timer_timeoutValue
260
240
240
o_timer_start
output TS:
TS2(PAD,PAD)
TS1(Link#,PAD)
TS1(Link#,PAD)
o_OScreator_reqNum
24
16
2047
o_OScreator_enable
o_OScreator_types
01
01
01
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up
Outputs

5.8. Configuration linkwidth start (upstream) Sequence:

Description:

This sequence shows the transition process into the Config.linkwidth.start state, the operation within it, and the transition process out of it (for upstream ports)..

Inputs:

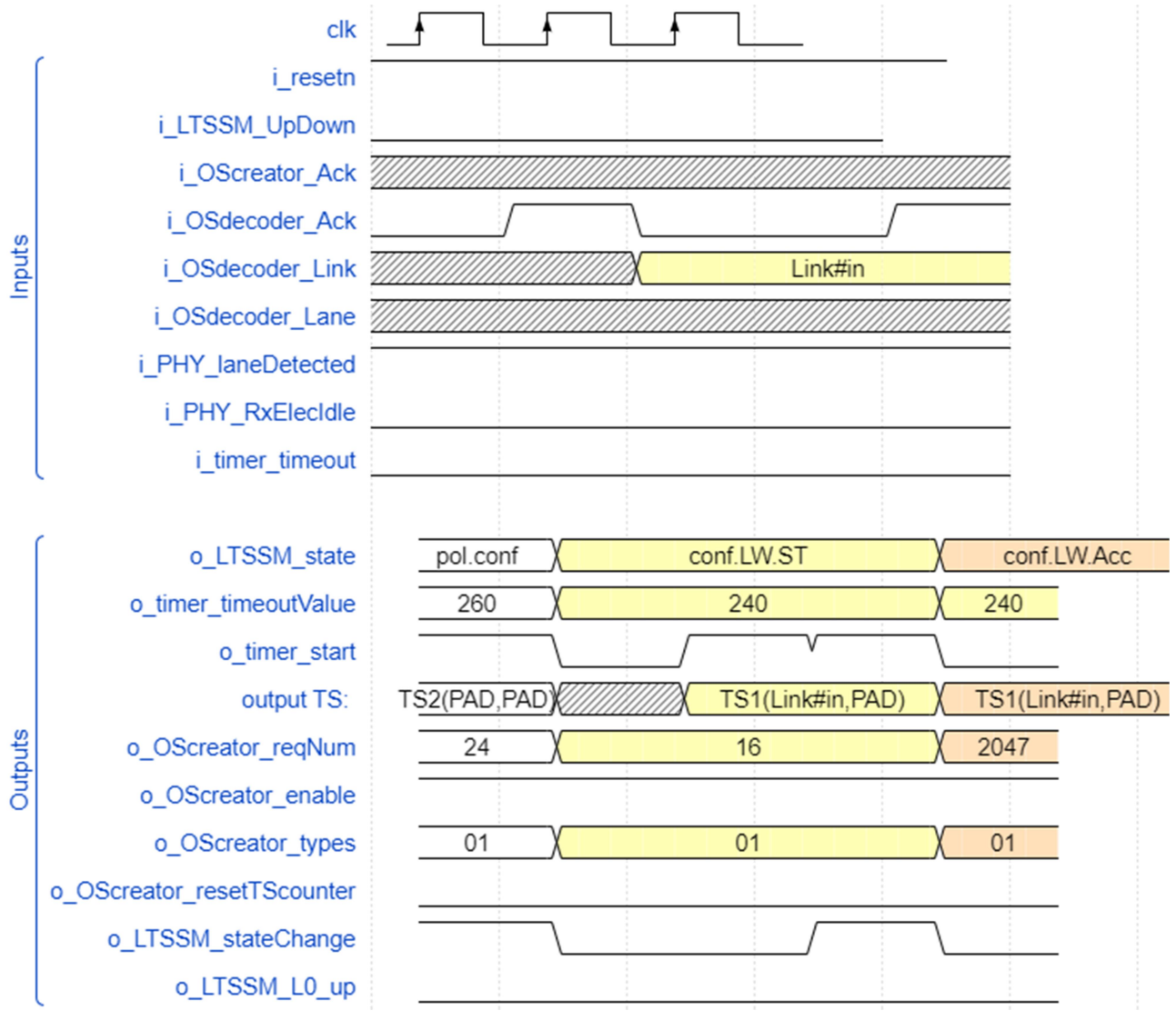
Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	X	x	x	x	X
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	X	x	Link#in	Link#in	Link#in
i_OSdecoder_Lane	X	x	x	x	x

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	pol.conf	conf.LW.ST	conf.LW.ST	conf.LW.ST	conf.LW.Acc
o_timer_timeoutValue	260	240	240	240	240
o_timer_start	1	0	1	1	0
output TS:	TS2(PAD,PAD)	x	TS1(Link#,PAD)	TS1(Link#,PAD)	TS1(Link#,PAD)
o_OScreator_reqNum	24	16	16	16	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateCh	1	0	0	1	0

ange					

Waveform(s):



5.9. Configuration linkwidth accept (downstream) Sequence:

Description:

This sequence shows the transition process into the Config.linkwidth.accept state, the operation within it, and the transition process out of it (for downstream ports).

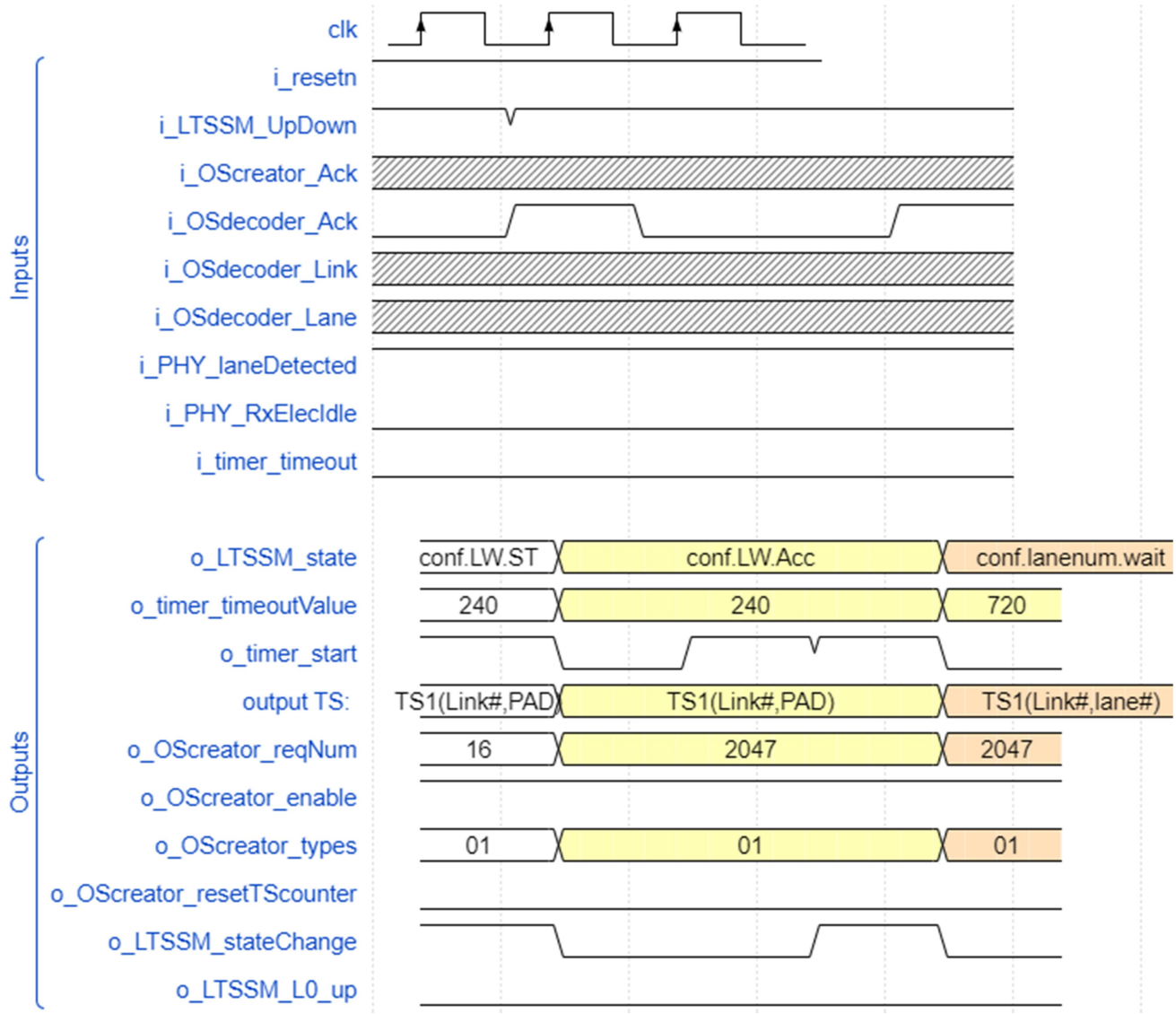
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	X	x	x	x	x
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	X	x	x	x	x
i_OSdecoder_Lane	X	x	x	x	x

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.LW.ST	conf.LW.Acc	conf.LW.Acc	conf.LW.Acc	conf.lane#.wait
o_timer_timeoutValue	240	240	240	240	720
o_timer_start	1	0	1	1	0
output TS:	TS1(Link#,PAD)	TS1(Link#,PAD)	TS1(Link#,PAD)	TS1(Link#,PAD)	TS1(link#,lane#)
o_OScreator_reqNum	16	2047	2047	2047	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.LW.ST
conf.LW.Acc
conf.lanenum.wait
o_timer_timeoutValue
240
240
720
o_timer_start
output TS:
TS1(Link#,PAD)
TS1(Link#,PAD)
TS1(Link#,lane#)
o_OScreator_reqNum
16
2047
2047
o_OScreator_enable
o_OScreator_types
01
01
01
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.10. Configuration linkwidth accept (upstream) Sequence:

Description:

This sequence shows the transition process into the Config.linkwidth.accept state, the operation within it, and the transition process out of it (for upstream ports).

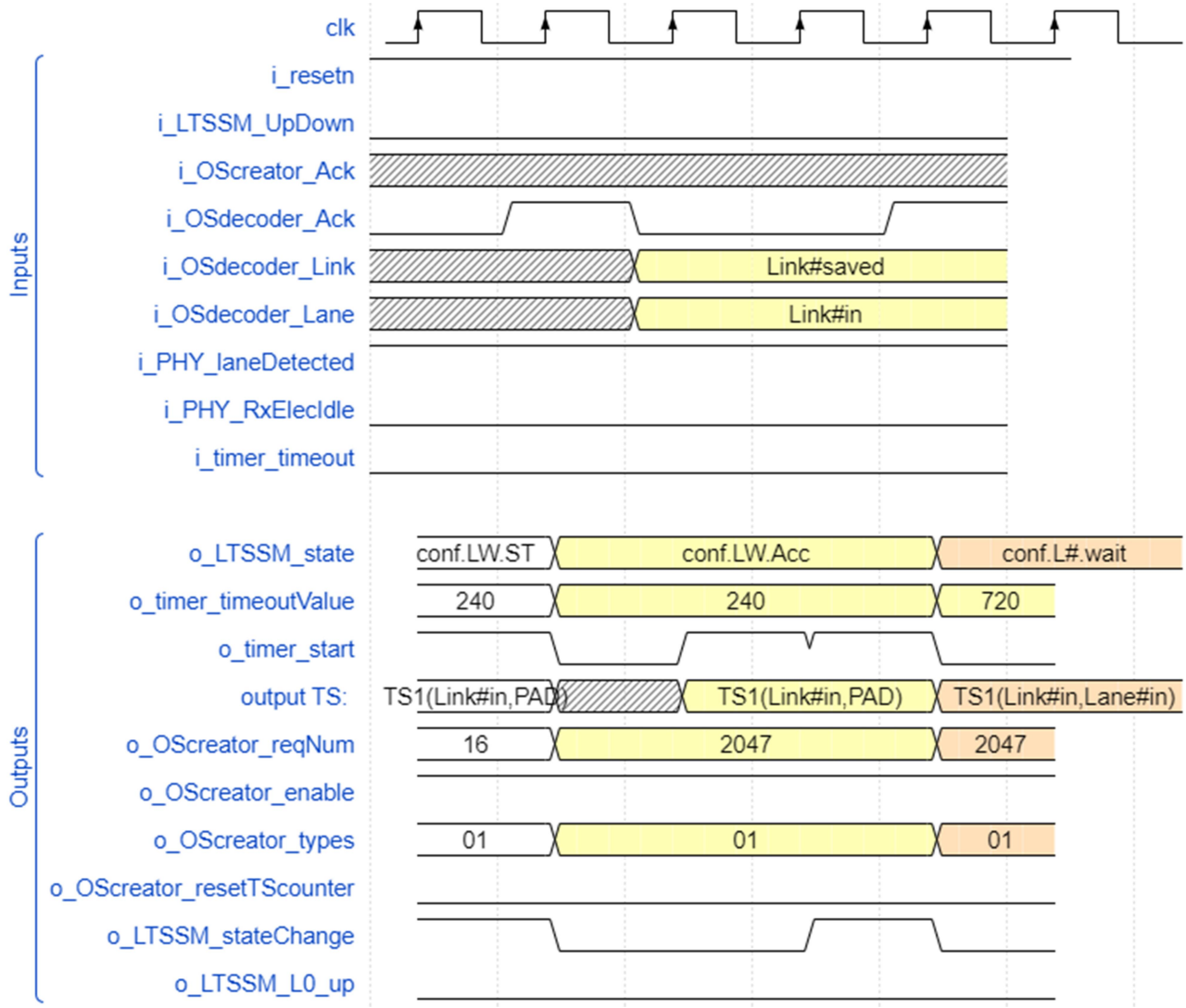
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	x	x	x	x	X
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	x	x	Link#saved	Link#saved	Link#saved
i_OSdecoder_Lane	x	x	Lane#in	Lane#in	Lane#in

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.LW.ST	conf.LW.Acc	conf.LW.Acc	conf.LW.Acc	conf.lane#.wait
o_timer_timeoutValue	240	240	240	240	720
o_timer_start	1	0	1	1	0
output TS:	TS1(Lnkin,PAD)	TS1(Lnkin,PAD)	TS1(Lnkin,PAD)	TS1(Lnkin,PAD)	TS1(lnkin,lane_in)
o_OScreator_reqNum	16	2047	2047	2047	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
Link#saved
i_OSdecoder_Lane
Link#in
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.LW.ST
conf.LW.Acc
conf.L#.wait
o_timer_timeoutValue
240
240
720
o_timer_start
output TS:
TS1(Link#in,PAD)
TS1(Link#in,PAD)
TS1(Link#in,Lane#in)
o_OScreator_reqNum
16
2047
2047
o_OScreator_enable
o_OScreator_types
01
01
01
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.11. Configuration lanenum wait (downstream) Sequence:

Description:

This sequence shows the transition process into the Config.lanenum.wait state, the operation within it, and the transition process out of it (for downstream ports).

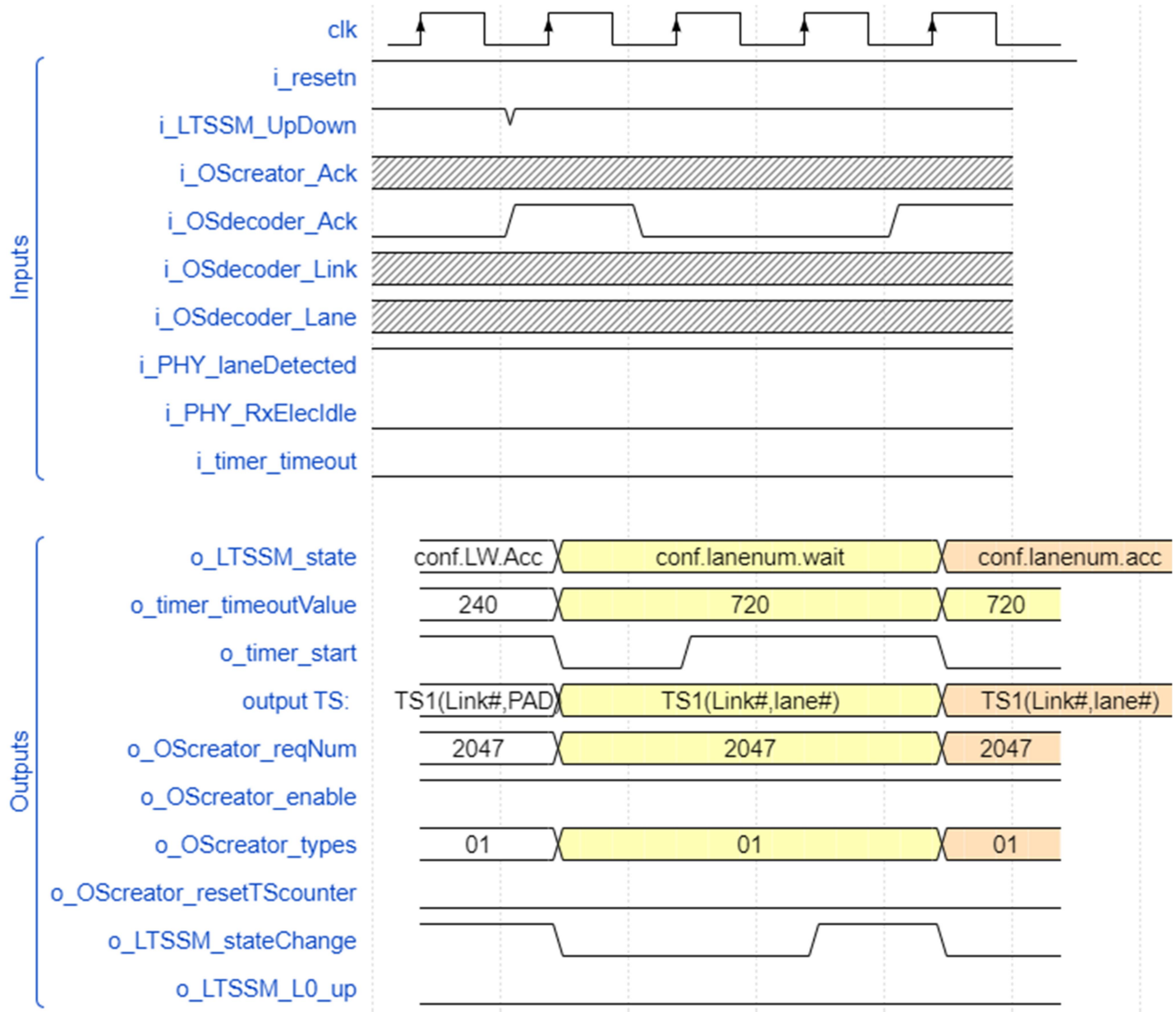
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	x	x	x	x	x
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	x	x	x	x	x
i_OSdecoder_Lane	x	x	x	x	x

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.LW.Acc	conf.lane#.wait	conf.lane#.wait	conf.lane#.wait	conf.lane#.acc
o_timer_timeoutValue	240	720	720	720	720
o_timer_start	1	0	1	1	0
output TS:	TS1(Link#,PAD)	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)
o_OScreator_reqNum	2047	2047	2047	2047	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.LW.Acc
conf.lanenum.wait
conf.lanenum.acc
o_timer_timeoutValue
240
720
720
o_timer_start
output TS:
TS1(Link#,PAD)
TS1(Link#,lane#)
TS1(Link#,lane#)
o_OScreator_reqNum
2047
2047
2047
o_OScreator_enable
o_OScreator_types
01
01
01
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.12. Configuration lanenum wait (upstream) Sequence:

Description:

This sequence shows the transition process into the Config.lanenum.wait state, the operation within it, and the transition process out of it (for downstream ports).

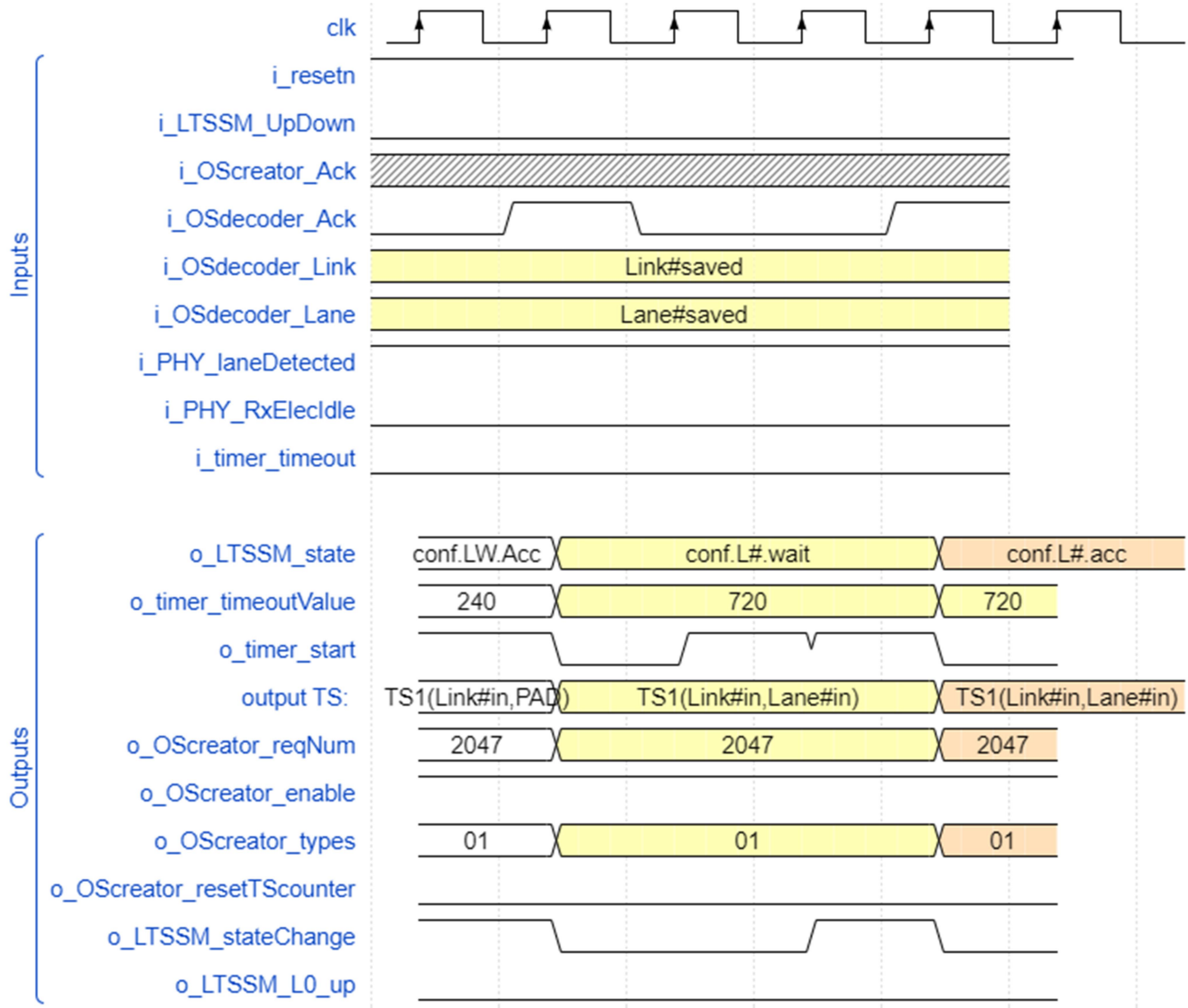
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	X	x	x	x	X
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	Link#saved	Link#saved	Link#saved	Link#saved	Link#saved
i_OSdecoder_Lane	Lane#saved	Lane#saved	Lane#saved	Lane#saved	Lane#saved

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.LW.Acc	conf.lane#.wait	conf.lane#.wait	conf.lane#.wait	conf.lane#.acc
o_timer_timeoutValue	240	720	720	720	720
o_timer_start	1	0	1	1	0
output TS:	TS1(Link#,PAD)	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)
o_OScreator_reqNum	2047	2047	2047	2047	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
Link#saved
i_OSdecoder_Lane
Lane#saved
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.LW.Acc
conf.L#.wait
conf.L#.acc
o_timer_timeoutValue
240
720
720
o_timer_start
output TS:
TS1(Link#in,PAD)
TS1(Link#in,Lane#in)
TS1(Link#in,Lane#in)
o_OScreator_reqNum
2047
2047
2047
o_OScreator_enable
o_OScreator_types
01
01
01
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.13. Configuration lanenum Accept (downstream) Sequence:

Description:

This sequence shows the transition process into the Config.lanenum.accept state, the operation within it, and the transition process out of it (for downstream ports).

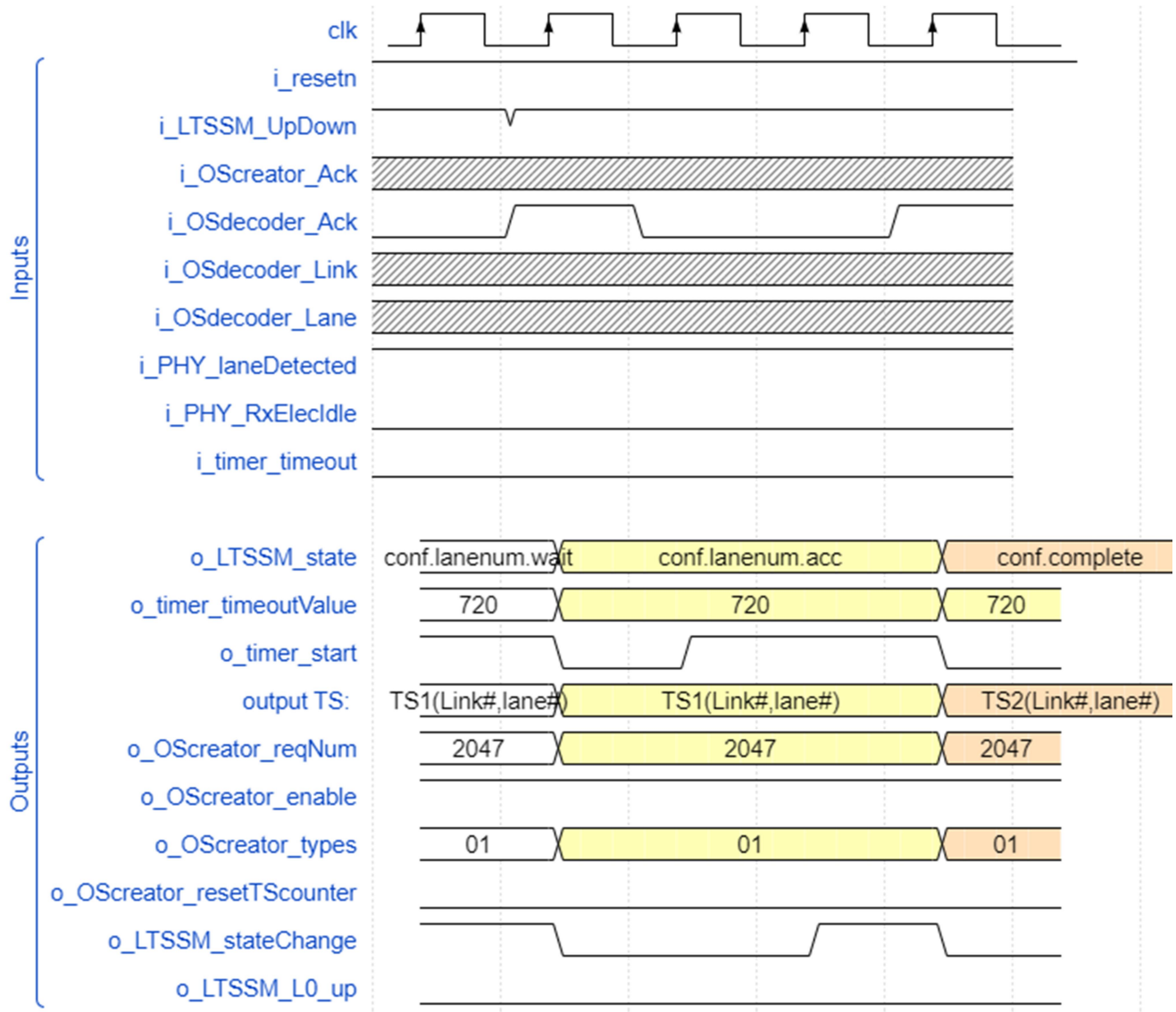
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	X	x	x	x	x
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	X	x	x	x	x
i_OSdecoder_Lane	X	x	x	x	x

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.lane#.wait	conf.lane#.acc	conf.lane#.acc	conf.lane#.acc	conf.complete
o_timer_timeoutValue	720	720	720	720	720
o_timer_start	1	0	1	1	0
output TS:	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)	TS2(link#,lane#)
o_OScreator_reqNum	2047	2047	2047	2047	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.lanenum.wait
conf.lanenum.acc
conf.complete
o_timer_timeoutValue
720
720
720
o_timer_start
output TS:
TS1(Link#,lane#)
TS1(Link#,lane#)
TS2(Link#,lane#)
o_OScreator_reqNum
2047
2047
2047
o_OScreator_enable
o_OScreator_types
01
01
01
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.14. Configuration.lanenum.Accept (upstream) Sequence:

Description:

This sequence shows the transition process into the Config.lanenum.accept state, the operation within it, and the transition process out of it (for upstream ports).

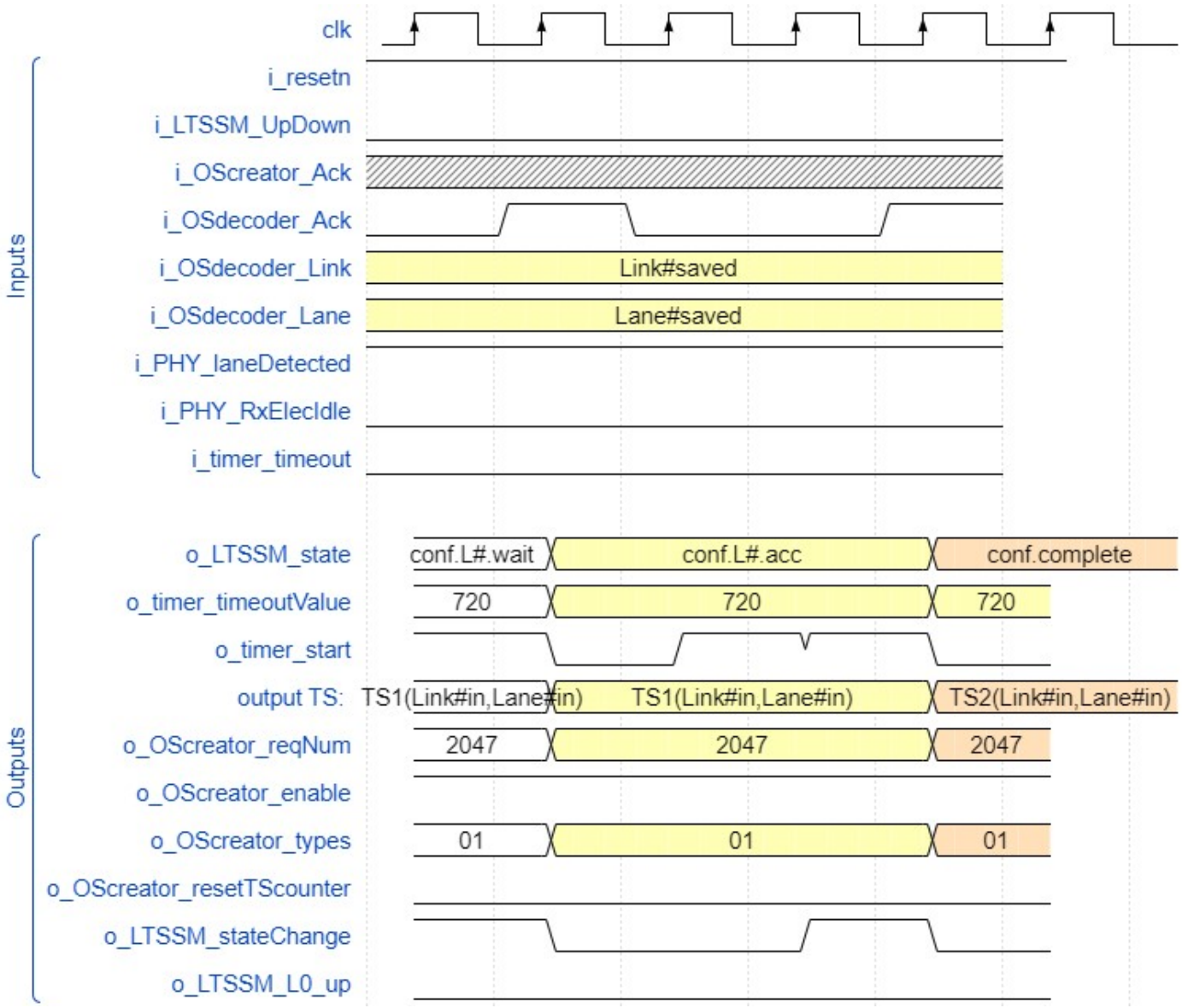
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	X	x	x	x	X
i_OSdecoder_Ack	00	01	00	00	01
i_OSdecoder_Link	Link#saved	Link#saved	Link#saved	Link#saved	Link#saved
i_OSdecoder_Lane	Lane#saved	Lane#saved	Lane#saved	Lane#saved	Lane#saved

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.lane#.wait	conf.lane#.acc	conf.lane#.acc	conf.lane#.acc	conf.complete
o_timer_timeoutValue	720	720	720	720	720
o_timer_start	1	0	1	1	0
output TS:	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)	TS1(link#,lane#)	TS2(link#,lane#)
o_OScreator_reqNum	2047	2047	2047	2047	2047
o_OScreator_types	01	01	01	01	01
o_OScreator_resetTScounter	0	0	0	0	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
i_OSdecoder_Link
Link#saved
i_OSdecoder_Lane
Lane#saved
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.L#.wait
conf.L#.acc
conf.complete
o_timer_timeoutValue
720
720
720
o_timer_start
output TS:
TS1(Link#in,Lane#in)
TS1(Link#in,Lane#in)
TS2(Link#in,Lane#in)
o_OScreator_reqNum
2047
2047
2047
o_OScreator_enable
o_OScreator_types
01
01
01
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.15. Configuration.complete Sequence:

Description:

This sequence shows the transition process into the Config.complete state, the operation within it, and the transition process out of it.

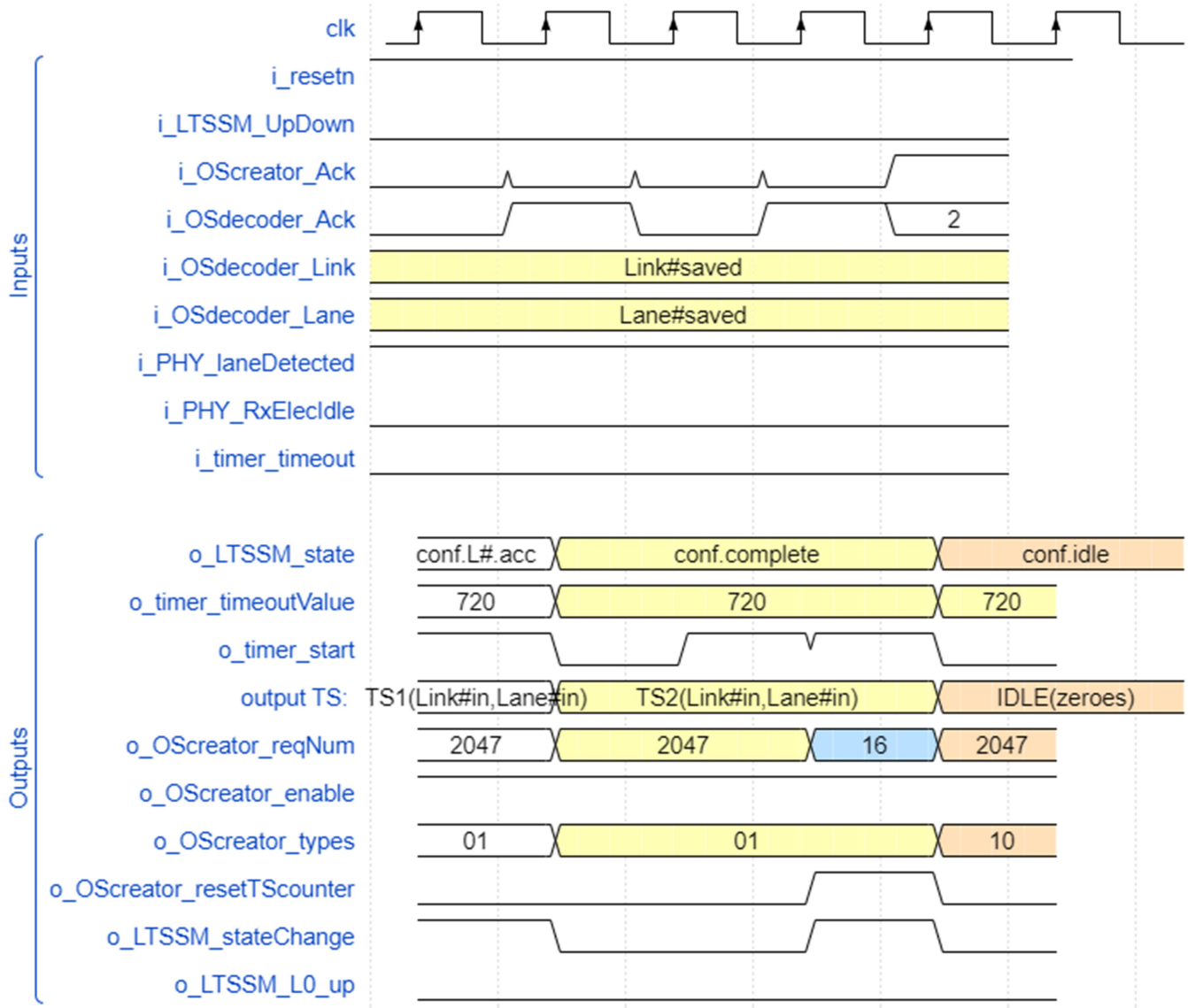
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	0	0	0	0	1
i_OSdecoder_Ack	00	01	00	01	2 (10)
i_OSdecoder_Link	Link#saved	Link#saved	Link#saved	Link#saved	Link#saved
i_OSdecoder_Lane	Lane#saved	Lane#saved	Lane#saved	Lane#saved	Lane#saved

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.lane#.acc	conf.complete	conf.complete	conf.complete	conf.idle
o_timer_timeoutValue	720	720	720	720	240
o_timer_start	1	0	1	1	0
output TS:	TS1(link#,lane#)	TS2(link#,lane#)	TS2(link#,lane#)	TS2(link#,lane#)	Idle (zeroes)
o_OScreator_reqNum	16	2047	2047	16	2047
o_OScreator_types	01	01	01	01	10
o_OScreator_resetTScounter	0	0	0	1	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
2
i_OSdecoder_Link
Link#saved
i_OSdecoder_Lane
Lane#saved
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.L#.acc
conf.complete
conf.idle
o_timer_timeoutValue
720
720
720
o_timer_start
output TS:
TS1(Link#in,Lane#in)
TS2(Link#in,Lane#in)
IDLE(zeroes)
o_OScreator_reqNum
2047
2047
16
2047
o_OScreator_enable
o_OScreator_types
01
01
10
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.16. Configuration.Idle Sequence:

Description:

This sequence shows the transition process into the Config.idle state, the operation within it, and the transition process out of it.

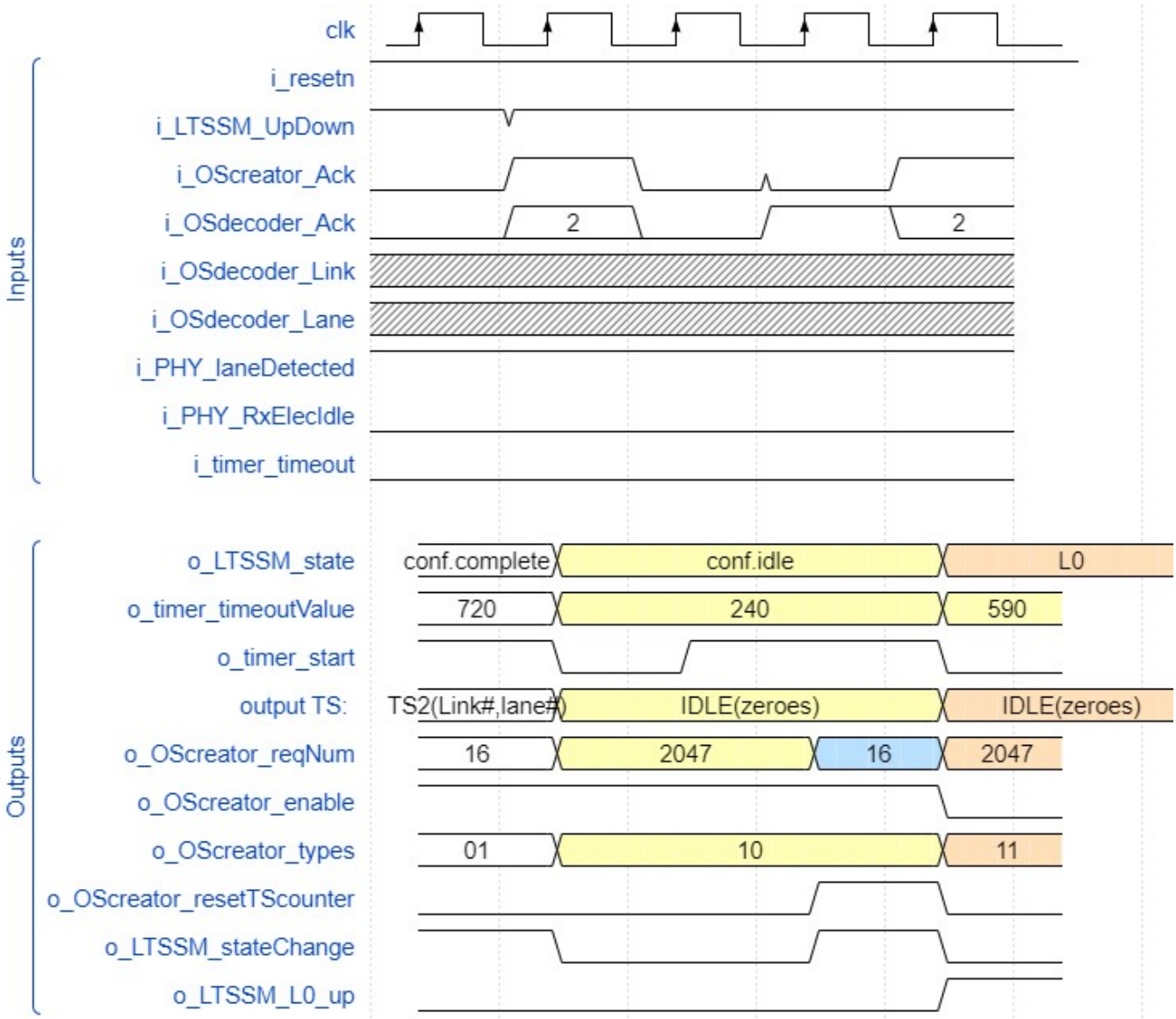
Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_LTSSM_UpDown	1	1	1	1	1
i_OScreator_Ack	0	0	0	0	1
i_OSdecoder_Ack	00	01	00	01	2 (10)
i_OSdecoder_Link	X	x	x	x	x
i_OSdecoder_Lane	X	x	x	x	x

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.complete	conf.idle	conf.idle	conf.idle	L0
o_timer_timeoutValue	720	240	240	240	590
o_timer_start	1	0	1	1	0
output TS:	TS2(link#,lane#)	Idle (zeroes)	Idle (zeroes)	Idle (zeroes)	Idle (zeroes)
o_OScreator_reqNum	16	2047	2047	16	2047
o_OScreator_types	01	10	10	10	11
o_OScreator_resetTScounter	0	0	0	1	0
o_LTSSM_stateChange	1	0	0	1	0

Waveform(s):


clk
Inputs
i_resetn
i_LTSSM_UpDown
i_OScreator_Ack
i_OSdecoder_Ack
2
2
i_OSdecoder_Link
i_OSdecoder_Lane
i_PHY_laneDetected
i_PHY_RxElecIdle
i_timer_timeout
Outputs
o_LTSSM_state
conf.complete
conf.idle
L0
o_timer_timeoutValue
720
240
590
o_timer_start
output TS:
TS2(Link#,lane#)
IDLE(zeroes)
IDLE(zeroes)
o_OScreator_reqNum
16
2047
16
2047
o_OScreator_enable
o_OScreator_types
01
10
11
o_OScreator_resetTScounter
o_LTSSM_stateChange
o_LTSSM_L0_up

5.17. L0 with SKP Sequence:

Description:

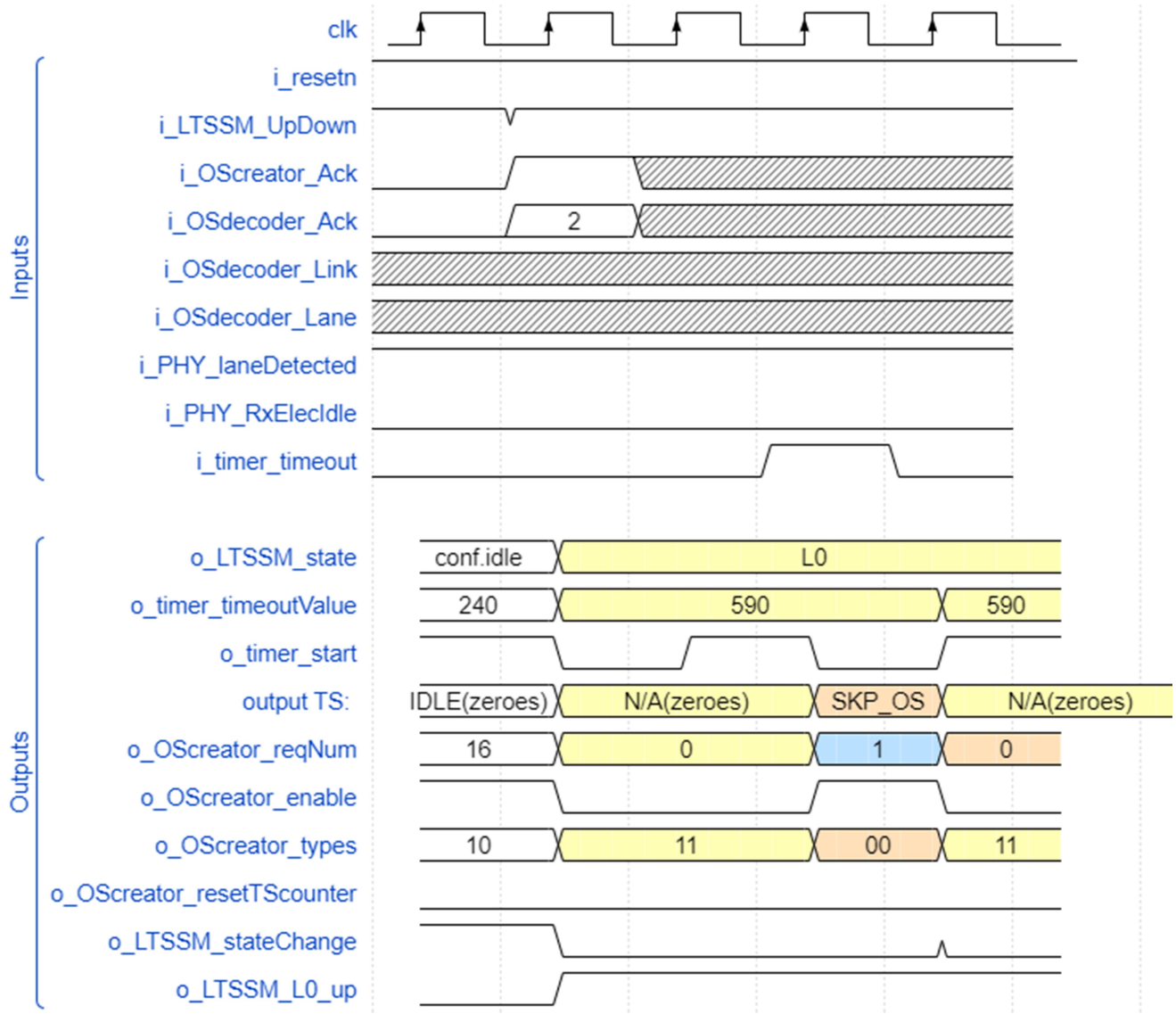
This sequence shows the transition process into the L0 state, and the operation within it. It also shows the periodic transmition of SKP order sets every 590 cycles during this state.

Inputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
i_OScreator_Ack	0	1	x	x	X
i_OSdecoder_Ack	00	2 (10)	x	x	x
i_OSdecoder_Link	X	x	x	X	x
i_OSdecoder_Lane	X	x	x	X	x
i_timer_timeout	0	0	0	1	0

Outputs:

Signal:	Value 1	Value 2	Value 3	Value 4	Value 5
o_LTSSM_state	conf.complete	conf.idle	conf.idle	conf.idle	L0
o_timer_timeout Value	240	590	590	590	590
o_timer_start	1	0	1	0	1
output TS:	Idle (zeroes)	Data	Data	SKP_OS	Data
o_OScreator_req Num	16	0	0	1	0
o_OScreator_typ es	10	11	11	00	11
o_OScreator_res etTScounter	0	0	0	0	0

o_LTSSM_state Change	1	0	0	0	0

Waveform(s):