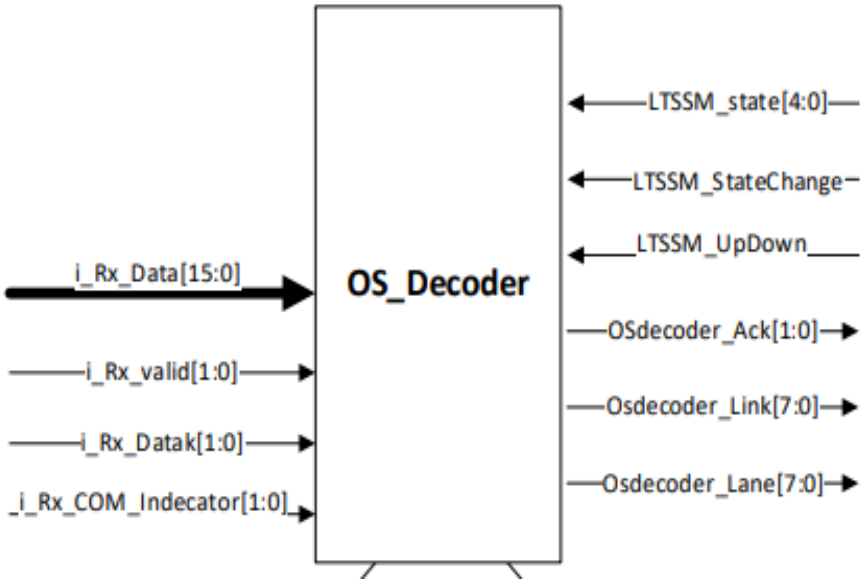
OS Decoder IP Specification

1. Introduction

1.1. Basic Operation

This IP receives data from the PCIe Rx and the state machine. Based on the data's content and the current state, it controls the state transitions during link training and initialization process.

1.2. Block I/O Signals



1.3. Interaction with Other Modules

1- PCIe Rx: the Decoder gets input data from the receiver. This data can be ordered sets sent from the other PCIe device on the link, and hence they are used to proceed throughout the link training and initialization process.

2- LTSSM State Machine: the Decoder gets input signals from the State Machine, which include information about the current state of the link, whenever the

state changes, and the type of the device (Upstream or Downstream). Also, the Decoder outputs signals to the State Machine, in which it can acknowledge that a state change should take place and the link and lane numbers of the received data from the receiver (useful in multi-lane communications).

2. I/O Signals Description

2.1. Inputs

Name	Width	Source	Description
clk	1 bit	Global	Clock signal
resetn	1 bit	Global	Synchronous low-level reset signal
i_Rx_Data	2 Bytes (16 bits)	PCIe Rx	Data from the receiver (sent from the other device on the link).
i_Rx_DataK	2 bits	PCIe Rx	Each bit corresponds to a byte in the Rx_Data signal. If the bit is 0, it corresponds to a data byte in the Rx_Data, and if it is 1, it corresponds to a control byte in the Rx_Data. Note: control bytes are COM, and bytes of Other OS (FTS, SKP, IDL).
i_Rx_valid	2 bits	PCIe Rx	Each bit indicates the validity of a byte in the Rx_Data signal.
i_Rx_COM_Indicator	2 bits	PCIe Rx	Indicates the beginning of a received ordered set in the Rx_Data signal. Note: 11 is an invalid case.
LTSSM_state	5 bits	State Machine	Encoding of the current state of the link.

LTSSM_StateChange	1 bit	State Machine	Indicates a change in the state of the link, so that the decoder should reset any counter of the received ordered sets to begin accounting only for those received in the new state.
LTSSM_UpDown	1 bit	State Machine	1 for Downstream and 0 for Upstream.

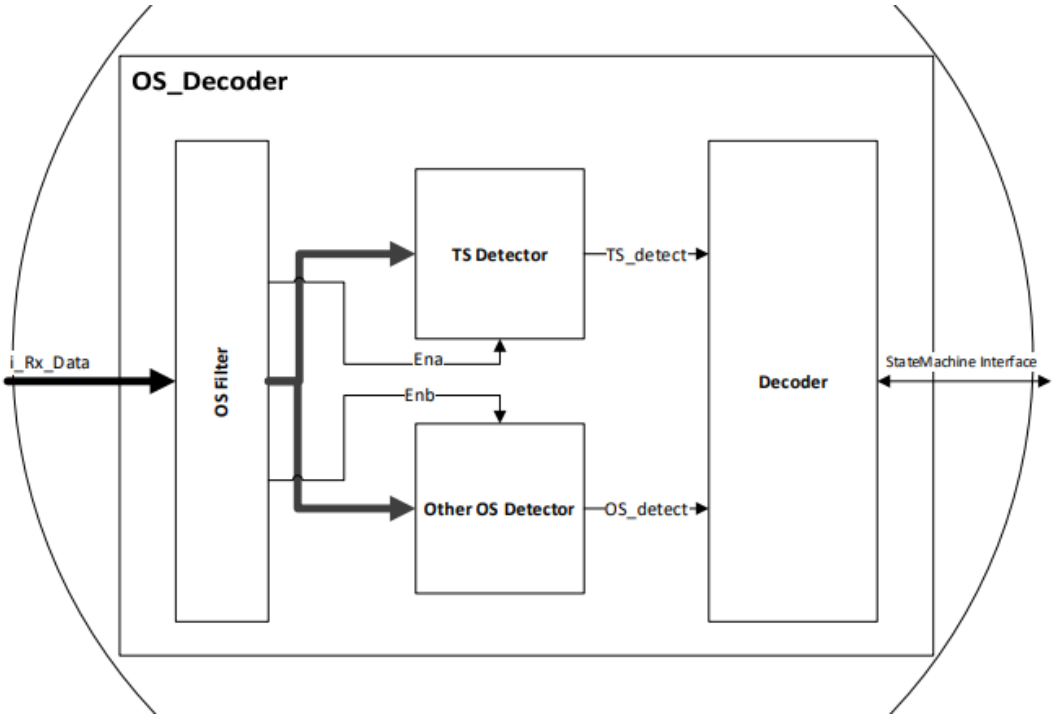
2.2. Outputs

Name	Width	Destination	Description
OSdecoder_Ack	2 bits	State Machine	Acknowledgement signal on the number of TS received to control the transition between the states. Note: for states requiring two conditions to occur, this signal is 01 when the first one occurs and 10 when the second one occurs. (e.g. Config_complete requires receiving one then eight TS2/Link#/Lane#.
OSdecoder_Lane	8 bits	State Machine	The lane number decoded from TS1 and TS2.
OSdecoder_Link	8 bits	State Machine	The link number decoded from TS1 and TS2.

3. Detailed Operation

This IP consists of 4 sub-blocks: filter, TS Detector, Other OS Detector, and the Decoder Main Block. The filter checks if the received data from the PCIe Rx is TS or Other OS, and accordingly it enables TS Detector or Other OS Detector or neither of them (if data received is not TS nor Other OS). TS Detector stores the content of the TS two byte-by-two byte till the whole 16 bytes are stored, then it outputs a detect signal to the Decoder Main Block. Similarly, Other OS Detector outputs detect and type signals when it receives the 4 bytes of an OS (SKP, FTS, IDL, EIE). Finally, the Decoder Main Block keeps counting whenever a TS is detected only if the TS content is matching with the required in the current state (e.g. TS2/PAD/PAD in Polling.Config state). When the required number of TS is received, it outputs an acknowledgement signal to the state machine to change the state.

Note: TS2/PAD/PAD means TS2 with Link and Lane number characters set to PAD.



4. Configuration

4.1. Configuration Sequence

1- Not TS Nor OS: received data has no COM character and it is not logical IDLE.

2- TS2 in Config.complete State: this state requires receiving one TS2/Link#/Lane#, then eight TS2/Link#/Lane#.

3- Logical IDLE in Config.idle State: this state requires receiving one Logical IDLE, then eight Logical IDLE's.

4.2. Latency

The IP consists of four pipelined sequential sub-blocks. As the IP receives Rx data two bytes in a clock cycle, it takes either 8 or 9 cycles to store and detect a single TS. The 9 cycles' case occurs when the COM is received in the second byte in the first cycle (i_Rx_COM_Indicator = 2'b10). Based on that, it would take multiples of either 8 or 9 or both to receive the needed number of matching TS based on the current state plus any in-between cycles in which TS are not received. Also, an additional cycle is needed in which the last TS is checked, and Ack signal is driven high.

Example: Polling.active state requires receiving either 8 TS1/PAD/PAD or 8 TS2/PAD/PAD. Assuming TS are received sequentially, and the COM of the first one is the first byte, the whole operation till the output of Ack signal would take (8*8) + 1 = 65 clock cycles.

4.3. Input Format and Constraints

- i_Rx_COM_Indicator cannot be 2'b11.
- LTSSM_state should not have different encodings than the following:
    a. Detect_Quiet = 5'b00000
    b. Detect_Active = 5'b00001

c. Polling_Active = 5'b00010
d. Polling_configuration = 5'b00011
e. Config_Linkwidth_start = 5'b00100
f. Config_Linkwidth_Accept = 5'b00101
g. Config_Lanenum_wait = 5'b00110
h. Config_Lanenum_Accept = 5'b00111
i. Config_complete = 5'b01000
j. Config_idle = 5'b01001
k. L0 = 5'b01010

- Special byte encodings in i_Rx_Data:
  a. PAD = 8'hF7.
  b. TS1 = 8'h4A. (last 10 bytes of TS1)
  c. TS2 = 8'h45. (last 10 bytes of TS2)
  d. COM = 8'hBC.
  e. SKP = 8'h1C. (the 3 bytes of SKP OS)
  f. FTS = 8'h3C. (the 3 bytes of FTS OS)
  g. IDL = 8'h7C. (the 3 bytes of IDL OS)
  h. EIE = 8'hFC. (the 3 bytes of EIE OS)
- To test the functionality of the block in any state requiring either Link or Lane number to be set to a specific value rather than PAD (states of configuration and L0), input sequences must start from Config_Linkwidth_Start state.

5. Sequences and Waveforms

5.1. Sequence 1: Not OS Nor TS

Inputs:

i_Rx_Data → don't care.

i_Rx_DataK → don't care.

i_Rx_valid → 2'b11.

i_Rx_COM_Indicator → 2'b00.

LTSSM_state → don't care.

LTSSM_StateChange → 1'b0.

LTSSM_UpDown → don't care.

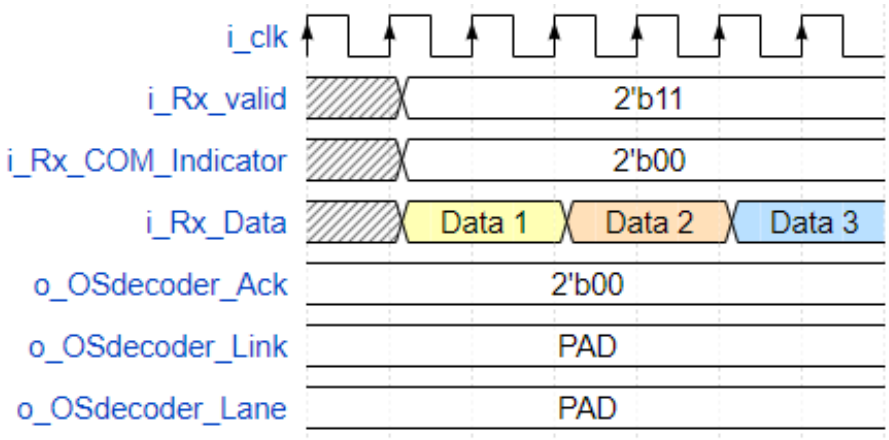
Outputs:

OSdecoder_Ack → 2'b00.

OSdecoder_Lane → default value (PAD = 8'hF7).

OSdecoder_Link → default value (PAD = 8'hF7).

Waveform(s):



5.2. Sequence 2: TS2 in Config.complete State

Inputs:

i_Rx_Data → subsequent 9 TS2/Link#/Lane#. Let Link# = Lane# = 8'b1, then TS2 would be inserted as follows:

I. 16'h01BC (Link#, COM)
II. 16'h0F01 (N_FTS, Lane#). Note: N_FTS is useless as FTS is not supported.
III. 16'h0002 (Training Ctrl, Rate ID).
IV. 16'h4545 (TS2 ID, TS2 ID). This one is repeated 5 times.

i_Rx_DataK → 2'b01 in the first two bytes of any TS2 (COM, Link#), 2'b00 otherwise.

i_Rx_valid → 2'b11.

i_Rx_COM_Indicator → 2'b01 in the first two bytes of any TS2 (COM, Link#), 2'b00 otherwise.

LTSSM_state → 5'b01000.

LTSSM_StateChange → 1'b0.

LTSSM_UpDown → don't care.

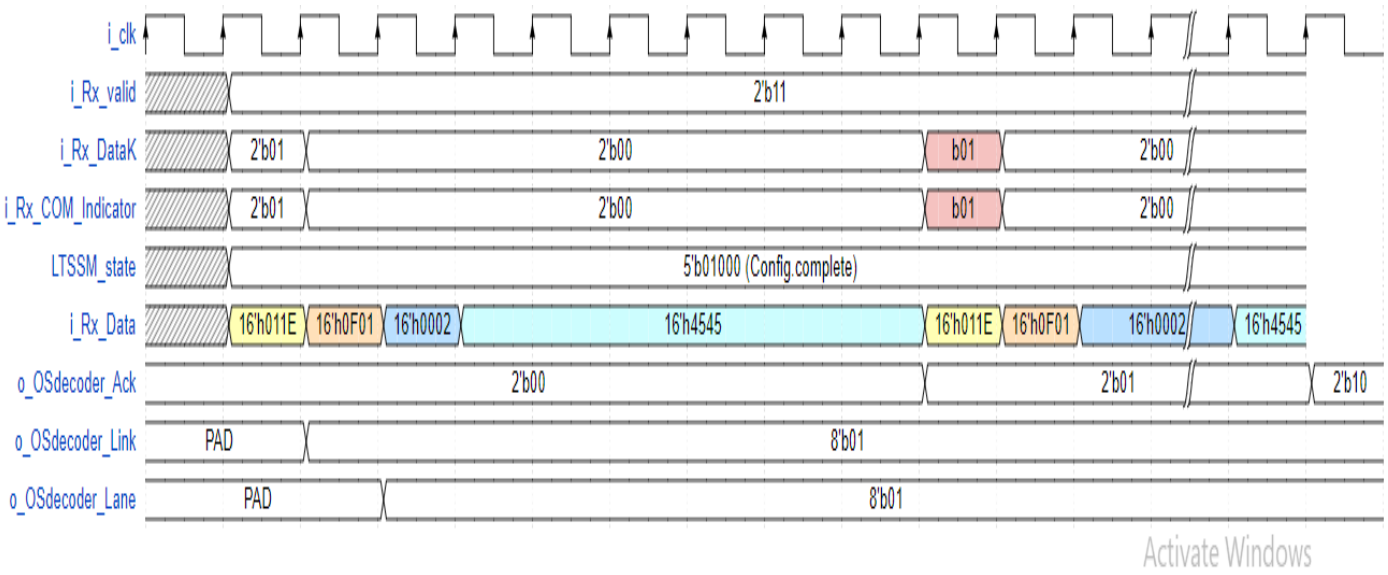
Outputs:

OSdecoder_Ack → 2'b00, then 2'b01 after receiving the first TS2, then 2'b10 after receiving the other 8 TS2.

OSdecoder_Lane → PAD, then 8'b01 after receiving the second two bytes of TS2.

OSdecoder_Link → PAD, then 8'b01 after receiving the first two bytes of TS2.

Waveform(s):


i_clk
i_Rx_valid
2'b11
i_Rx_DataK
2'b01
2'b00
b01
2'b00
i_Rx_COM_Indicator
2'b01
2'b00
b01
2'b00
LTSSM_state
5'b01000 (Config.complete)
i_Rx_Data
16'h011E
16'h0F01
16'h0002
16'h4545
16'h011E
16'h0F01
16'h0002
16'h4545
o_OSdecoder_Ack
2'b00
2'b01
2'b10
o_OSdecoder_Link
PAD
8'b01
o_OSdecoder_Lane
PAD
8'b01
Activate Windows

5.3. Sequence 3: Logical IDLE in Config.idle State

Inputs:

i_Rx_Data → 16'h0000.

i_Rx_DataK → 2'b00.

i_Rx_valid → 2'b11.

i_Rx_COM_Indicator → 2'b00.

LTSSM_state → 5'b01001.

LTSSM_StateChange → 1'b0.

LTSSM_UpDown → don't care.

Outputs:

OSdecoder_Ack → 2'b00, then 2'b01 after receiving the first IDLE bytes, then 2'b10 after receiving the other 8 IDLE bytes.

OSdecoder_Lane → should have the value set in previous states (let it be 8'h01).

OSdecoder_Link → should have the value set in previous states (let it be 8'h01).

Waveform(s):