OPEN POSSIBILITIES.

DC-SCM 2.0 LVDS Tunneling Protocol & Interface (LTPI) Introduction [On Demand Virtual Session]
DC-SCM 2.0 LVDS Tunneling Protocol & Interface (LTPI) Introduction

Kasper Wszolek, Principal Engineer, Intel
Yi (Roy) Zeng, Principal Engineer, Intel
Agenda

• Background
• LTPI Introduction
• Channels Concept
• Working Principle
• Frames definition
• Electrical I/O Specification
• Link training and interface bring-up
• Implementation example
• Call to Action
Background

- DC-SCM modular designs moves substantial part of server platform on a module (BMC, Root Of Trust, Front Panel, SPI Flashes & TPM)
- DC-SCM 1.0/2.0 SFF-TA1002 4C+ connector is limited to 168 pins and cannot accommodate all low speed and high pin count interfaces
- 2 x Serial GPIO (9 pins) interfaces are defined in DC-SCM 1.0 to tunnel GPIO signals between HPM platform and DC-SCM Module
LTPI Introduction

DC-SCI 2.0 LTPI:
- DC-SCI 2.0 introduces high-speed and scalable Low-voltage differential signaling Tunneling Protocol & Interface (LTPI)
- LTPI uses 4 differential links (8 pins) in place of 2 Serial GPIO interfaces from DC-SCI 1.0 and can be used to tunnel more than just GPIOs making room for DC-SCI evolution
LTPI Overview

BMC
- GPIOs
- I2Cs
- UARTs
- OEM
- Config

SCM
- SCM CPLD
- LVDS Channels
- LVDS – TX DAT, TX CLK, RX DAT, RX CLK

HPM
- HPM FPGA
- HPM UART Interfaces
- I2C/Smbus/PMbus Slave Devices
- GPIO
- I2C
- UART
- OEM Intf.
- Config Intf.
- Interface to LVDS CSR Space

CPU Alerts, VR WARNINGS, Power Status, I2C/PMBus Alerts, SCM Status & Control, Mux Selects, Power Throttle Requests, Power Throttle Controls
LTPI Channels

- Each channel is mapped to a physical interface that is serialized through LTPI i.e., GPIO, I2C/SMBus, UART, OEM Interfaces and Data Interfaces
- Time-Division Multiplexing (TDM) method is used to tunnel each channel through LVDS
Working Principle

DC-SCM

SCM CPLD

I2C

Relay

FIFO

Comma Symbol

+ 

8b/10b Encode

Link Manage

Frame Generation + CRC Generation

Comma Symbol + 8b/10b Encode

Comma Detection + 8b/10b Decode

OEM

DATA INTF.

CSR INTF.

UART ctrl

LTPI Controller

LVDS TX PHY

TX DATA

TX CLK

LVDS RX PHY

RX DATA

RX CLK

FIFO

DC-SCM

HPM

HPM FPGA

I2C Relay

GPIO ctrl

UART ctrl

OEM

DATA INTF.

DATA INTF.

OEM INTF.

CSR INTF.

UART

Data MM

COMMA DETECTION

+ 

8b/10b Decode

Frame Parsing + CRC Check

Comma Symbol + 8b/10b Encode

Frame Generation + CRC Generation

LTPI Controller

LVDS TX PHY

TX DATA

TX CLK

LVDS RX PHY

RX DATA

RX CLK

OEM INTF.

DATA INTF.

OEM INTF.

DATA INTF.

CSR INTF.

UART

Data MM

COMMA DETECTION

+ 

8b/10b Decode

Frame Parsing + CRC Check

Comma Symbol + 8b/10b Encode

Frame Generation + CRC Generation

LTPI Controller

LVDS TX PHY

TX DATA

TX CLK

LVDS RX PHY

RX DATA

RX CLK

OEM INTF.

DATA INTF.

OEM INTF.

DATA INTF.

CSR INTF.

UART

Data MM

COMMA DETECTION

+ 

8b/10b Decode

Frame Parsing + CRC Check

Comma Symbol + 8b/10b Encode

Frame Generation + CRC Generation

LTPI Controller

LVDS TX PHY

TX DATA

TX CLK

LVDS RX PHY

RX DATA

RX CLK

OEM INTF.

DATA INTF.

OEM INTF.

DATA INTF.

CSR INTF.

UART

Data MM

COMMA DETECTION

+ 

8b/10b Decode

Frame Parsing + CRC Check

Comma Symbol + 8b/10b Encode

Frame Generation + CRC Generation

LTPI Controller

LVDS TX PHY

TX DATA

TX CLK

LVDS RX PHY

RX DATA

RX CLK

OEM INTF.

DATA INTF.

OEM INTF.

DATA INTF.

CSR INTF.
Working Principle

- Traffic on supported interfaces is generated/consumed by devices connected on DC-SCM (e.g., BMC) and HPM (e.g., CPU)
Working Principle

- LTPI captures/generates the traffic from supported interfaces through dedicated interface controllers
Working Principle

- LTPI Controller generates a frame with each interface state encoded on bits dedicated for given channel
Working Principle
- Each channel is encoded on predefined bits within LTPI Frame
Working Principle

- The LTPI Frame is encoded using 10b encoding and Comma Symbol is added to the frame.
The frame is serialized through LVDS link with dedicated synchronized clock.
Working Principle

- On RX side the LVDS PHY is used to capture the frame from LVDS serial link
Working Principle

- Frame is detected based on Comm Symbol and deserialized
Working Principle

- Frame is validated against CRC and decomposed into channels
Working Principle

- Respective channel states are passed to interface controllers
Working Principle

- Traffic is re-generated on supported interfaces connected to physical devices e.g. CPU or BMC
Working Principle

- The other direction from HPM to DC-SCM follows exactly same principle
LTPI frames definition

LTPI Frames are fixed size frames
- Default size is 160 bits : 16 symbols x 10bits

<table>
<thead>
<tr>
<th>Sequence</th>
<th>Symbol</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Comma Symbol</td>
</tr>
<tr>
<td>1</td>
<td>Frame Payload</td>
</tr>
<tr>
<td>2</td>
<td>Frame Payload</td>
</tr>
<tr>
<td>14</td>
<td>Frame Payload</td>
</tr>
<tr>
<td>15</td>
<td>CRC</td>
</tr>
</tbody>
</table>

LTPI Frames Types are identified by different Comma Symbols:

<table>
<thead>
<tr>
<th>Comma Symbol</th>
<th>Frame Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comma Symbol</td>
<td>Link Detect and Speed Selection</td>
<td>Initial frame transmitted during LTPI bring-up flow to provide information about supported LTPI speed capabilities. Frame is also used to achieve DC balance on the link and select target link speed.</td>
</tr>
<tr>
<td>Symbol</td>
<td>Capabilities Advertise and Configure/Accept</td>
<td>Frame is used to advertise supported capabilities (Channels and features supported by LTPI) and configure enabled capabilities supported by DC:SCM and HPM LTPI endpoints.</td>
</tr>
<tr>
<td>Symbol</td>
<td>LTPI Operational Frame</td>
<td>Frame use during normal LTPI operation after interface initialization and configuration is completed. Frame carries all the channels supported by given LTPI implementation.</td>
</tr>
</tbody>
</table>
Electrical I/O Specification

Measured at DC-SCM CONN Side

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Minimum</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vod (Differential Voltage Swing)</td>
<td>100</td>
<td>800</td>
<td>mV</td>
</tr>
</tbody>
</table>

Common Voltage is VCCIO_RX/2 where VCCIO_RX is the Voltage at RX Buffer

Parameter | Min. | Max  | Unit |
----------|------|------|------|
VID (Input Differential Voltage Swing) | 100   | 800  | mV   |
## Electrical I/O – LVDS Standards

### Typical Common Voltage and Vod/Vid for various LVDS Standards

<table>
<thead>
<tr>
<th>Voltage</th>
<th>VCCIO RX</th>
<th>VCCIO TX</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>3.3V</strong></td>
<td>3.3V</td>
<td>3.3V</td>
</tr>
<tr>
<td><strong>2.5V</strong> (True LVDS)</td>
<td>2.5V (LVDS)</td>
<td>2.5V (Sub-LVDS)</td>
</tr>
<tr>
<td><strong>2.5V</strong> (Emulated LVDS)</td>
<td>2.5V</td>
<td>2.5V</td>
</tr>
<tr>
<td><strong>1.8V</strong> (True Sub-LVDS)</td>
<td>1.8V</td>
<td>1.8V</td>
</tr>
<tr>
<td><strong>1.8V</strong> (Emulated Sub-LVDS)</td>
<td>1.8V</td>
<td>1.8V</td>
</tr>
</tbody>
</table>

#### Vod/Vid

- **Vod**: Vod 350mV, Vid 350mV
- **Vid**: Vod 350mV, Vid 150mV

**LVDS I/O Voltage Swing Requirements**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Minimum</th>
<th>Max</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vod/Vid (Differential Voltage Swing)</td>
<td>100</td>
<td>800</td>
<td>mV</td>
</tr>
</tbody>
</table>
Electrical I/O Interoperability

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>DC Blocking Capacity</td>
<td>0.22 uF</td>
</tr>
<tr>
<td>LVDS PCB Impedance</td>
<td>85 Ohm</td>
</tr>
<tr>
<td>LVDS Termination Resistor</td>
<td>100 Ohm</td>
</tr>
<tr>
<td>DC Blocking Caps Placement</td>
<td>DC-SCM Module</td>
</tr>
</tbody>
</table>
Link training and interface bring-up
Implementation example

Summary

<table>
<thead>
<tr>
<th>Device</th>
<th>LVDS Link Speed</th>
<th>Normal Latency GPIOs</th>
<th>Low Latency GPIOs</th>
<th>I2C Links 400kHz</th>
<th>UART Links 115200</th>
<th>Data Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel MAX10 10M25 CPLD</td>
<td>200 MHz</td>
<td>112</td>
<td>4</td>
<td>4</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Latency/Bandwidth

<table>
<thead>
<tr>
<th>Normal Latency GPIOs</th>
<th>Low Latency GPIOs</th>
<th>I2C Links</th>
<th>UART Links</th>
<th>Data Channels</th>
</tr>
</thead>
<tbody>
<tr>
<td>~9.4us</td>
<td>~1.3us</td>
<td>185 kHz</td>
<td>115200</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Resource utilization

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Utilization</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total Logic Elements</td>
<td>~2,200</td>
<td>~8.8%</td>
</tr>
<tr>
<td>PLL</td>
<td>2</td>
<td>50%</td>
</tr>
<tr>
<td>GPIOS/UARTs</td>
<td>~1,000</td>
<td></td>
</tr>
<tr>
<td>I2C</td>
<td>~1,200 (300 LE per link)</td>
<td></td>
</tr>
</tbody>
</table>
### Key features Re-cap

<table>
<thead>
<tr>
<th>Key Features</th>
<th>Benefits and Value Add</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>LVDS I/O (Low-Voltage Differential Signaling)</strong></td>
<td>- Improved Signal Integrity and Bandwidth comparing to Single-ended</td>
</tr>
<tr>
<td></td>
<td>- Supported by most of CPLD &amp; FPGA</td>
</tr>
<tr>
<td><strong>AC-coupling</strong></td>
<td>- Allows for multiple LVDS voltage standard use</td>
</tr>
<tr>
<td></td>
<td>- Improved CPLD/FPGA interoperability</td>
</tr>
<tr>
<td><strong>Multiple interfaces tunneling</strong></td>
<td>- Supports tunneling of more interfaces than just GPIO: SMBus/I2C, UART, Data Channel</td>
</tr>
<tr>
<td><strong>High bandwidth capabilities</strong></td>
<td>- Reference designs running at 200Mbps</td>
</tr>
<tr>
<td></td>
<td>- Latest FPGAs support LVDS at 1600 Mbps</td>
</tr>
<tr>
<td><strong>Scalability &amp; Flexibility</strong></td>
<td>- Specific designs can choose which interfaces are tunneled with LTPI</td>
</tr>
<tr>
<td></td>
<td>- OEM Extensions can be added to LTPI as OEM channels</td>
</tr>
<tr>
<td></td>
<td>- Data Channel allows for random access and data exchange between DC-SCM CPLD and HPM CPLD</td>
</tr>
<tr>
<td><strong>Interoperability</strong></td>
<td>- Link training and capabilities exchange protocol defined</td>
</tr>
<tr>
<td></td>
<td>- BMC controls the process of LTPI bring-up and configuration</td>
</tr>
</tbody>
</table>
Call to Action

• Join us in Experience Center to see the Live Demo of LTPI interface:
  • Intel Demo of LTPI Implementation on Intel Max10 CPLDs
  • Lattice Demo of LTPI Implementation on Lattice MachXO3 FPGAs
• Provide feedback to OCP HW Management Module Subproject

Project Wiki with latest specification:  
https://www.opencompute.org/wiki/Hardware_Management/Hardware_Management_Module

Mailing list: OCP-HWMgt-Module@OCP-All.groups.io
Thank you!