On Developing OpenCAPI Memory Host Agent (LPC) designs

Dimitris Syrivelis, IBM Research - Ireland
Ingredients List

Hardware checklist

- Power9 Processor server (e.g. Nimbus/Lagrange prod step DD2.1) (OCAPI NPU / OTL host)

- FPGA (e.g 9V3) attached on OpenCAPI Port. (AFU side DLx – TLx – AFU sotcore logic)

Software checklist

- Hostboot override files (to change OCAPI transceiver PLL config – useful for debugging)

- OPAL/Skiboot (PCI-scan + SerDES Link Training)

- Linux Kernel (Register OCAPI device – control all low-level details – PASID determination etc)

- LibOCXL (Application library to interact with the memory Host agent – Cacheable mapping of Host Memory agent)

Support included in the very latest releases only !!!!
H/w: Must read before diving into the verilog reference design code

OpenCAPI V3.0 Transaction Layer Specification:

https://opencapi.org/technical/specifications/
**H/w:** What one needs to support in AFU context

- Configuration Commands (PCI Configuration Space)
- Cfg space State Machines
- CFG space Table Module

**Diagram:**
- DLx
- TLx (Protocol Parser)
- AFU Context
**H/w:** What one needs to support in AFU context

- DLx
- TLx (Protocol Parser)
- Configuration Commands (PCI Configuration Space)
- MMIO Register Commands
- Cfg space State Machines
- MMIO cmd State Machines
- CFG space Table Module
- MMIO Register Module
**H/w**: What one needs to support in AFU context

- Configuration Commands (PCI Configuration Space)
- MMIO Register Commands
- Standard Memory IO Commands
- Cfg space State Machines
- MMIO cmd State Machines
- Memory IO State Machines
- CFG space Table Module
- MMIO Register Module
- Memory Technology Driver
**H/w: What one needs to support in AFU context**

- **DLx (Protocol Parser)**
  - Configuration Commands (PCI Configuration Space)
  - MMIO Register Commands
  - Standard Memory IO Commands
  - Command & Data Credit Management

- **TLx**
  - Configuration Commands (PCI Configuration Space)
  - MMIO cmd State Machines
  - Memory IO State Machines
  - Memory Technology Driver

- **CFG space Table Module**
  - Cfg space State Machines
  - MMIO Register Module
**H/w: What one needs to support in AFU context**

- Configuration Commands (PCI Configuration Space)
- MMIO Register Commands
- Standard Memory IO Commands
- Command & Data Credit Management
- Error Control
- Cfg space State Machines
- MMIO cmd State Machines
- Memory IO State Machines
- CFG space Table Module
- MMIO Register Module
- Memory Technology Driver

**TLx (Protocol Parser)**

**DLx**
H/w: New Functionality Focus!!!

DLx

TLx (Protocol Parser)

Configuration Commands (PCI Configuration Space)

MMIO Register Commands

Standard Memory IO Commands

Command & Data Credit Management

Error Control

CFG space State Machines

Memory IO State Machines

Memory Technology Driver

MMIO cmd State Machines

CFG space Table Module

AFU Context
**H/w**: OpenCAPI LPC Reference Design

- DLx
- TLx Protocol Framer
- Top_device
- AFU00 lpc_afu.v

Design clocks
H/w: OpenCAPI LPC Reference Design

DLx

TLx (Protocol Parser)

Configuration Commands (PCI Configuration Space)

MMIO Register Commands

MMIO cmd State Machines

CFG space Table Module

MMIO Register Module

Memory IO State Machines

Memory Technology Driver

lpc_mmio_regs.v

SAM + Block RAM

lpc_afu_credit_manager.v

lpc_afu.v

Command & Data Credit Management

Error Control
**H/w:** OpenCAPI LPC Reference Design (PCI-Config)

```c
cfg_descriptor DESC (  
    .clock  
    , .reset  
    // READ ONLY field inputs  

    //, .ro_name_space  
    //, .ro_name_space  
    accordingly with NULLs  
    , .ro_afu_version_major  
    , .ro_afu_version_minor  
    , .ro_afuc_type  
    data  
    , .ro_afum_type  
    , .ro_profile  
    //, .ro_global_mmio_offset  
    be h0000  
    , .ro_global_mmio_bar  
    , .ro_global_mmio_size  
    , .ro_cmd_flag_x1_supported  
    , .ro_cmd_flag_x3_supported  
    , .ro_atc_2M_page_supported  
    , .ro_atc_64K_page_supported  
    , .ro_max_host_tag_size  
    , .ro_per_pasid_mmio_offset  
    be h0000  
    , .ro_per_pasid_mmio_bar  
    , .ro_per_pasid_mmio_stride  
    //, .ro_mem_size  
    //, .ro_mem_size  
    //, .ro_mem_start_addr  
    //, .ro_nao_wvid  
    //)

    ( clock  
    ( reset  
    // (positive active)

    // 22222111111110000000000  
    // 43210887545421987654321  
    // Keep string exactly 24 characters long  
    // 'IBM',LPC000000000000000000 = Keep string exactly 24 characters long  
    // 'IBM',LPC000000000000000000 = Keep string exactly 24 characters long  
    // 'IBM',LPC000000000000000000 = Keep string exactly 24 characters long

    ( "IBM",LPC000000000000000000  
    ("IBM\RMEM","16[8'h00]" ) )  
    ) // '.' is an illegal character in the name  
    ) // String must contain EXACTLY 24 characters, so pad  

    ( 'AFU_VERSION_MAJOR  
    ( 'AFU_VERSION_MINOR  
    ( 3'b001  
    // Type C1 issues commands to the host but does not cache host  
    ) // Type M1 contains host mapped addresses (i.e. MMIO or memory)  
    ) // Device Interface Class  
    ) // MMIO space start offset from BAR 0 addr ([15:0] assumed to  

    ( 3'b000  
    ( 32'h0000,0000  
    ( 1'h0  
    ( 1'h0  
    ( 1'h0  
    ( 1'h0  
    ( 5'h00000  
    ( 48'h0000,0000,0000  
    // MMIO space is contained in BAR0  
    ) // LPC MMIO size is 1 MB, but Global MMIO section is 512 KB  
    ) // cmd_flag x1 is not supported  
    ) // cmd_flag x3 is not supported  
    ) // Address Translation Cache page size of 2MB is not supported  
    ) // Address Translation Cache page size of 64KB is not supported  
    ) // Caching is not supported  
    ) // PASID space start at BAR 0+512KB address ([15:0] assumed to  

    ( 3'b000  
    ( 16'h001  
    ( 8'h14  
    ( 8'h2A  
    ( 64'h0000,0000,0000,0000  
    ( 128'h0000,0000,0000,0000,0000,0000,0000,0000 ) // LPC has no WWID  

    lpc_afu.v
```
H/w: OpenCAPI LPC Reference Design
Command Input/Output

LPC Main Functionality (P9 NPU masters commands towards AFU)

Used for Error Control (AFU masters commands towards P9 NPU)

8 Channels: Think Parallel!

TLx -> AFU Cmd
TLx -> AFU CmdData
AFU -> TLx Resp
AFU -> TLx RespData

AFU -> TLx Cmd
AFU -> TLx Data
TLx -> AFU Resp
TLx -> AFU Data

lpc_afu.v
**H/w:** OpenCAPI LPC Reference Design Command Input/Output – AFU -> TLx credit management

Independent credit manager per outgoing channel (i.e. AFU -> TLx channels)

Current support allows
Only one command to claim
Credits at any given time

```v

Credit Mgr

Credit Mgr

Credit Mgr

Credit Mgr

```

```v

lpc_afu.v

```
H/w: OpenCAPI LPC Reference Design Command Input/Output – AFU -> TLx credit management

Keep Count Of the available resources at the other side !!!

Current support allows
Only one command to claim Credits at any given time

Independent credit manager per outgoing channel (i.e. AFU -> TLx channels)

```
lpclpc_afu.credit_mgr #C,MBC3) TARC {
.clock
.reset
        | reset
        | .resync_credits
        | .tlx_initial_credit
        | .tlx_afu_resp_initial_credit
        | .tlx_afuResp.credit
        | .tlx.credit
        | .afu_consume_credit_0 (d'bo)
        | .afu_consume_credit_1 (d'bo)
        | .afu_consume_credit_2 (d'bo)
        | .afu_consume_credit_3 (d'bo)
        | .afu_consume_credit_4 (d'bo)
        | .afu_consume_credit_5 (d'bo)
        | .afu_consume_credit_6 (d'bo)
        | .afu_consume_credit_7 (d'bo)
        | .afu_reclaim_credit_0 (d'bo)
        | .afu_reclaim_credit_1 (d'bo)
        | .afu_reclaim_credit_2 (d'bo)
        | .afu_reclaim_credit_3 (d'bo)
        | .afu_reclaim_credit_4 (d'bo)
        | .afu_reclaim_credit_5 (d'bo)
        | .afu_reclaim_credit_6 (d'bo)
        | .afu_reclaim_credit_7 (d'bo)
        | .afu_reclaim信貸_0 (d'bo)
        | .afu_reclaim信貸_1 (d'bo)
        | .afu_reclaim信貸_2 (d'bo)
        | .afu_reclaim信貸_3 (d'bo)
        | .afu_reclaim信貸_4 (d'bo)
        | .afu_reclaim信貸_5 (d'bo)
        | .afu_reclaim信貸_6 (d'bo)
        | .afu_reclaim信貸_7 (d'bo)
        | .creditoverflow (d'bo)
        | .creditunderflow (d'bo)

// Number of starting credits from TLX for AFU->TLX resp
// But credit return is an individual signal for each operation type
```

lpclpc_afu.v
**H/w: OpenCAPI LPC Reference Design Command Input/Output – TLx -> AFU credit management**

```verilog
// Return credit in the same cycle
tlx_afu_cmd_credit = 1;

always @(*)
  if (tlx_afu_cmd_valid == 1)
    begin
      // Return credit in the same cycle
      tlx_afu_cmd_credit = 1;
    end
```

Credit Mgr

- TLx -> AFU Cmd
- TLx -> AFU CmdData
- AFU -> TLx Resp
- AFU -> TLx RespData
- AFU -> TLx Cmd
- AFU -> TLx Data
- TLx -> AFU Resp
- TLx -> AFU Data

lpc_afu.v
**H/w:** OpenCAPI LPC Reference Design Command

**Input/Output – TLx -> AFU credit management**

```
tlx_afu_cmd_initial_credit = 1;
always @(*)
    if (tlx_afu_cmd_valid == 1)
        begin
            // Return credit in the same cycle
            tlx_afu_cmd_credit = 1;
        end

    // In the command SM context eg Full Memory Read:
    if(tlx_afu_cmd_dl == 2’b01)
        tlx_afu_cmd_rd_cnt = 3’b001; // Ask 1 data flit
    else if (tlx_afu_cmd_dl == 2’b10)
        tlx_afu_cmd_rd_cnt = 3’b010; // Ask 2 data flits
    else tlx_afu_cmd_rd_cnt = 3’b100; // Ask 4 data flits
```

```lpc_afu.v```
**H/w:** OpenCAPI LPC Reference Design Command Input/Output – TLx -> AFU credit management

Requested Dataflits will be arriving in every cycle

```plaintext
TLx -> AFU Cmd
TLx -> AFU CmdData
AFU -> TLx Resp
AFU -> TLx RespData
AFU -> TLx Cmd
AFU -> TLx Data
TLx -> AFU Resp
TLx -> AFU Data
```
**H/w:** OpenCAPI LPC Reference Design Command Input/Output

- Stream semantics for Tlx-AFU channels

```
“READY”: (”Have FIFO space to Receive command”)
assign VALID = afu_tlx_cmd_valid
```

`see: lpc_cmdfifo.v`

Diagram:
```
TLx -> AFU Cmd
TLx -> AFU CmdData
AFU -> TLx Resp
AFU -> TLx RespData
AFU -> TLx Cmd
AFU -> TLx Data
TLx -> AFU Resp
TLx -> AFU Data
```

`lpc_afu.v`
**H/w:** OpenCAPI LPC Reference Design Command Input/Output – Stream semantics for Tlx-AFU channels

```
TLx -> AFU Cmd
TLx -> AFU CmdData
AFU -> TLx Resp
AFU -> TLx RespData
AFU -> TLx Cmd
AFU -> TLx Data
TLx -> AFU Resp
TLx -> AFU Data
```

READY = 1 (Data flits controlled by `tlx_afu_cmd_rd_cnt`)

assign VALID = `afu_tlx_data_valid`

`lpc_afu.v`
**H/w:** OpenCAPI LPC Reference Design Command Input/Output – Stream push semantics

assign READY = (afu_tlx_resp_credit > 0)
assign VALID = afu_tlx_resp_valid

Credit Mgr
**H/w: OpenCAPI LPC Reference Design Commands**

rd_mem, pr_rd_mem, write_mem, pr_wr_mem, write_mem.be

**NOTES:**
These commands use Physical Addresses but not Real
(Real Host Address - Memory IO BAR value)
Command opcodes are carefully selected so you can
Determine Read or Write Families by inspecting a single bit

*tlx -> AFU Cmd*
*tlx -> AFU CmdData*
*afu -> tlx Resp*
*afu -> tlx RespData*
*afu -> tlx Cmd*
*afu -> tlx Data*
*tlx -> AFU Resp*
*tlx -> AFU Data*
H/w: OpenCAPI LPC Reference Design Commands

Mem_rd_response, mem_rd_fail, mem_wr_response, mem_wr_fail

TLx -> AFU Cmd
TLx -> AFU CmdData
AFU -> TLx Resp
AFU -> TLx RespData
AFU -> TLx Cmd
AFU -> TLx Data
TLx -> AFU Resp
TLx -> AFU Data

lpc_afu.v
H/w: OpenCAPI LPC Reference Design Commands

Assign_actag, intrp_req

- TLx -> AFU Cmd
- TLx -> AFU CmdData
- AFU -> TLx Resp
- AFU -> TLx RespData
- AFU -> TLx Cmd
- AFU -> TLx Data
- TLx -> AFU Resp
- TLx -> AFU Data
- lpc_afu.v
**H/w:** OpenCAPI LPC Reference Design Commands

```
TLx -> AFU Cmd
TLx -> AFU CmdData
AFU -> TLx Resp
AFU -> TLx RespData
AFU -> TLx Cmd
AFU -> TLx Data
TLx -> AFU Resp
TLx -> AFU Data
```

`int_resp`

`lpc_afu.v`
H/w: OpenCAPI LPC Reference Design Error Control

1. SUE (Unrecoverable part of the OpenCAPI transaction layer spec)
   - Embedded in command responses.
     (e.g. cmd flit issued and followup dataflit fails checksum)

2. Design detected errors (e.g. Fifo/credit overflows, bad command, memory h/w errors etc):
   - Define a memory-mapped 64-bit registers and assign bits
   - Trigger interrupt SM to send an interrupt command to host (AFU -> TLx cmd channel)
**H/w:** The OpenCAPI LPC Reference Design Handling Memory I/O commands

- TLx -> AFU Cmd
- TLx -> AFU CmdData
- AFU -> TLx Resp
- AFU -> TLx RespData
- AFU -> TLx Cmd
- AFU -> TLx Data
- TLx -> AFU Resp
- TLx -> AFU Data

Where PA belongs?

- lpc_cmd_fifo
- resp_fifo
- Mem I/O SM
- pipeline is empty

- pr_rd_mem or pr_wr_mem

- MMIO R SM
- MMIO Range
- Mem Range

IBM Research
**H/w:** The OpenCAPI LPC Reference Design Handling Memory I/O commands

- **TLx -> AFU Cmd**
- **TLx -> AFU CmdData**
- **AFU -> TLx Resp**
- **AFU -> TLx RespData**
- **AFU -> TLx Cmd**
- **AFU -> TLx Data**
- **TLx -> AFU Resp**
- **TLx -> AFU Data**

```
pr_rd_mem or pr_wr_mem
```

Where PA belongs?

```
lpc_cmd_fifo
```

```
MMIO Range
```

```
Mem I/O Range
```

```
Pipeline is empty
```

```
resp_fifo
```

```
“Steal” commands here and handle in a dataflow HLS example
```

```
lpc_afu.v
```

IBM Research
H/w: HLS dummy example dataflow design

![Diagram of dataflow design](image.png)
Building / Debugging the design

- **Aim for 400Mhz** – design runs in the transceiver clock domain
  - 200Mhz synchronized phase clock available but not meaningful for LPC
  - Use latest vivado – spreadlogic placement/routing policies
  - DLx/TLx are already floorplanned, floorplan in the same spirit.

- Possibility to operate OpenCAPI transceivers at 20G (clock drops to 312 Mhz)
  - Primarily used for debugging with ILA – diminishes performance advantage
  - Requires ring override files to be burned to boot flash
Switch OpenCAPI transceivers to operate at 20G
Example on Zaius (Nimbus/LaGrange DD2.1)

- OpenBMC side. burn attribute override info to flash:
  
  
  #20G
  pflash -e -f -p ATTR_TMP.obus.20.0 -P ATTR_TMP

  You will need to reboot afterwards.

- Design side need to configure the design DLx_phy.xci (Ultrascale Transceivers)
S/w: Starting up

1. Burn the right OPAL/Skiboot version

2. Reconfigure FPGA

3. During OPAL bootstrap you should get the OCAPI serDES link(s) trained
S/w: Starting up

1. Use Recent Kernel Version

2. After login, your AFU should be visible at /sys/class/ocxl/:

```
root@arlab119:/home/jsyr# ls /sys/class/ocxl/
IBM,LPC.0005:00:00.1.0
root@arlab119:/home/jsyr# 
```
Questions?

Dimitris.syrivelis@ibm.com

http://www.research.ibm.com/labs/ireland