CRI Compute Cluster (CCC) HDD

MAY 25, 2021

REV 1.0

Table of Contents

Revision History

Revision	Description	Date
1.0	First version	May 25, 2021
1.1	Updates	

Terminology

Term	Description
CCC	CRI Compute Cluster
CI	Command Interpreter
EEP	External Event Processor
IEP	Internal Event Processor
HP	HBM Processor
PFC	Pointer-FIFO Controller
NoC	Network on Chip
AXI	Advanced Extensible Interface
AHB	Advanced High-Performance Bus (Multiple Master support in each bus)
AHB-Lite	AHB with multiple lanes for multiple Master Nodes
Master/Initiator	NoC term for agent generating transaction
Slave/Target	NoC term for agent receiving transaction, and responding
AXI-Full	ARM protocol that supports independent Write/ Read/ Address channels, multiple beats, out of order execution using different transaction identifiers
AXI-Lite	Truncated AXI standard, with single beat transaction, and no ID support (implies no out of order)
HBM	High-Bandwidth Memory

1 Introduction

This is the CRI Compute Cluster (CCC) HDD.

CCC majorly consists of:

- CCC subsystems for core processing
- Inter-core and inter-FPGA event routing fabric

2 Scope

Key aspects covered in the CCC HDD are:

1. **CCC subsystems**
    a. Command Interpreter (CI)
    b. External Event Processor (EEP)
    c. Internal Event processor (IEP)
    d. HBM Processor (HP)
    e. Pointer-FIFO Controller (PFC)
    f. Routing Interface (RI)
2. **Hierarchical Event Routing (HiAER)**
    a. Inter-Core Network-On-Chip (NoC)
    b. Inter-FPGA Routing
    c. Inter-Server Routing
3. **Trace and Debug proposal** for CC and NoC
4. **FPGA Resources Summary**
    a. Synthesis Estimates
    b. Placement and Routing (PnR) Utilization
    c. Timing Optimizations

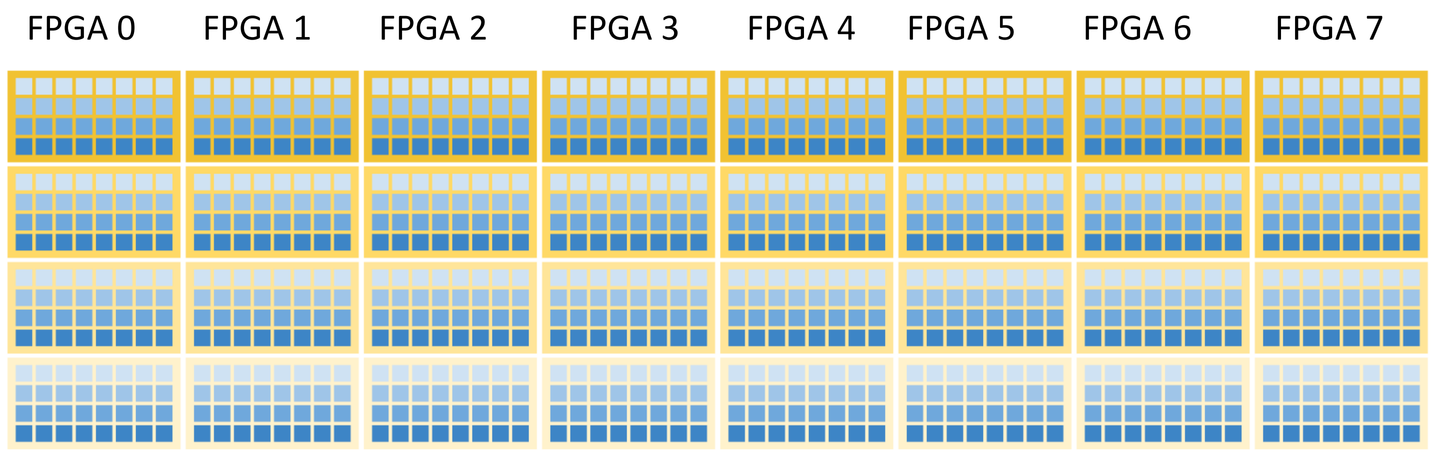
What is not covered:

1. Network Partitioning Algorithms, CRI Simulator

3 Key Opens (To-Do List)

S. No	Opens	Notes
1	Design RI Logic	Path to EEP Latch events from Multiple Hierarchies
2	AHB for Level 0 and Level 1 Communication	NoC Design for multicast
3	HBM Table Entries for External Events/Off-Core Messages	
4	HBM Table Entries for Relay Neurons	New FSM state and clock for zero-delay Neurons. Routing logic external messages to upper hierarchy.
5	Fixing Timing Issues	

4 High level CCC Block Diagrams



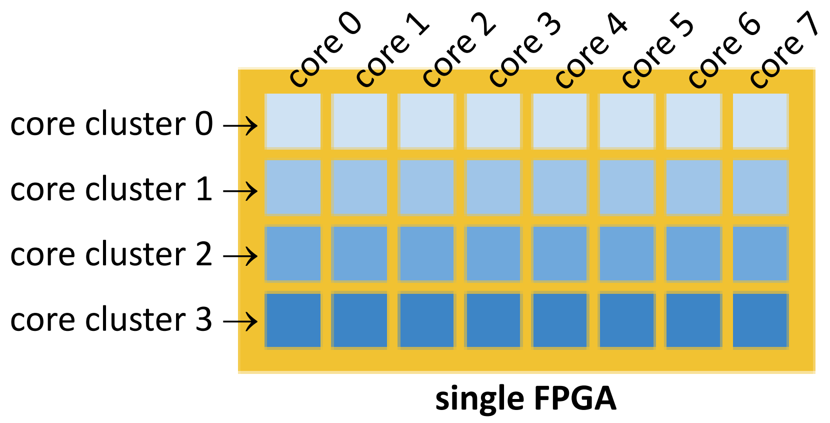


Fig:- FPGA Cluster

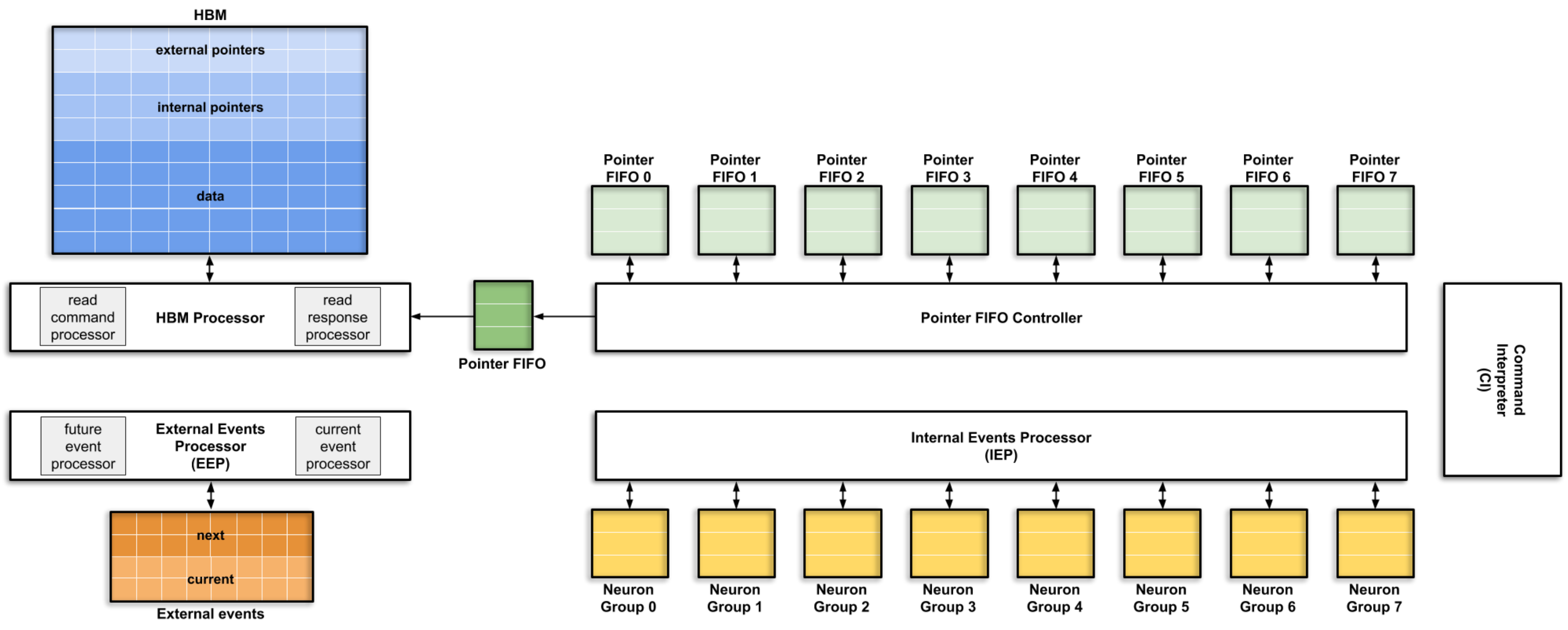


Fig:- Different Submodules inside each core

5 High-level Resource Estimates

Processors

- 32 HBM AXI interfaces → 32 parallel processors

Neurons

- 4M neurons / 32 processors = 128K neurons / processor

Weight memory

- 8 GB total HBM memory:

8 GB / 32 processors = 256 MB / processor
1MB for internal pointers and 1MB for external pointers
(How much for Relay Neurons)
256 MB / (128K × 256 synapses) = 64 bits / synapse

Neuron memory

- 960 URAM blocks:

960 blocks / 32 processors = 30 blocks / processor
30 blocks = 120K × 72 bits = 8.64 Mb / processor
8.64 Mb / 128K neurons = 67 bits / neuron

6 Compute Flow

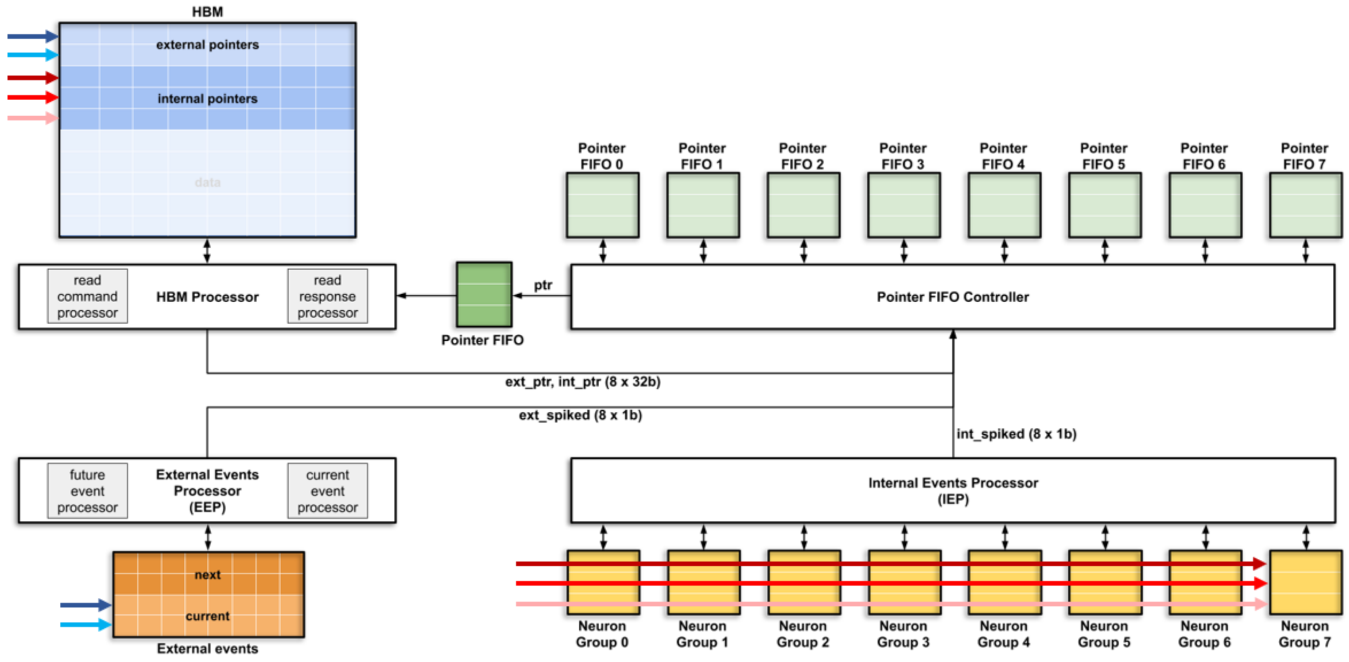
Compute in each core happens in 2 phases (Phase1 and Phase2 must complete within one biological clock (Delta_T).

Phase 1: Spike Decoding

- Analyze incoming external events for active inputs and internal neuron states for active ("spiked") neurons
- Translate incoming event and neuron addresses into pointers (from HBM pointer sections)
- Push pointers of active inputs/neurons into Pointer FIFO (to be used in 2nd phase)

Properties:

- Incoming external events (*next* and *current*) are stored in BRAM as a binary 2D array (17-bit address x 8-bit data)
- Neuron state variables are stored in 8 distinct URAMs (*neuron groups*) → 8 neuron states can be read in parallel
- Therefore, since at each clock cycle an HBM port produces 256 bits, and we require 8 pointers to be read simultaneously (because of the previous properties), each pointer uses 256/8 = 32 bits
- Pointer content: *ptr*[31:23] = read length (in # of rows), *ptr*[22:0] = base address

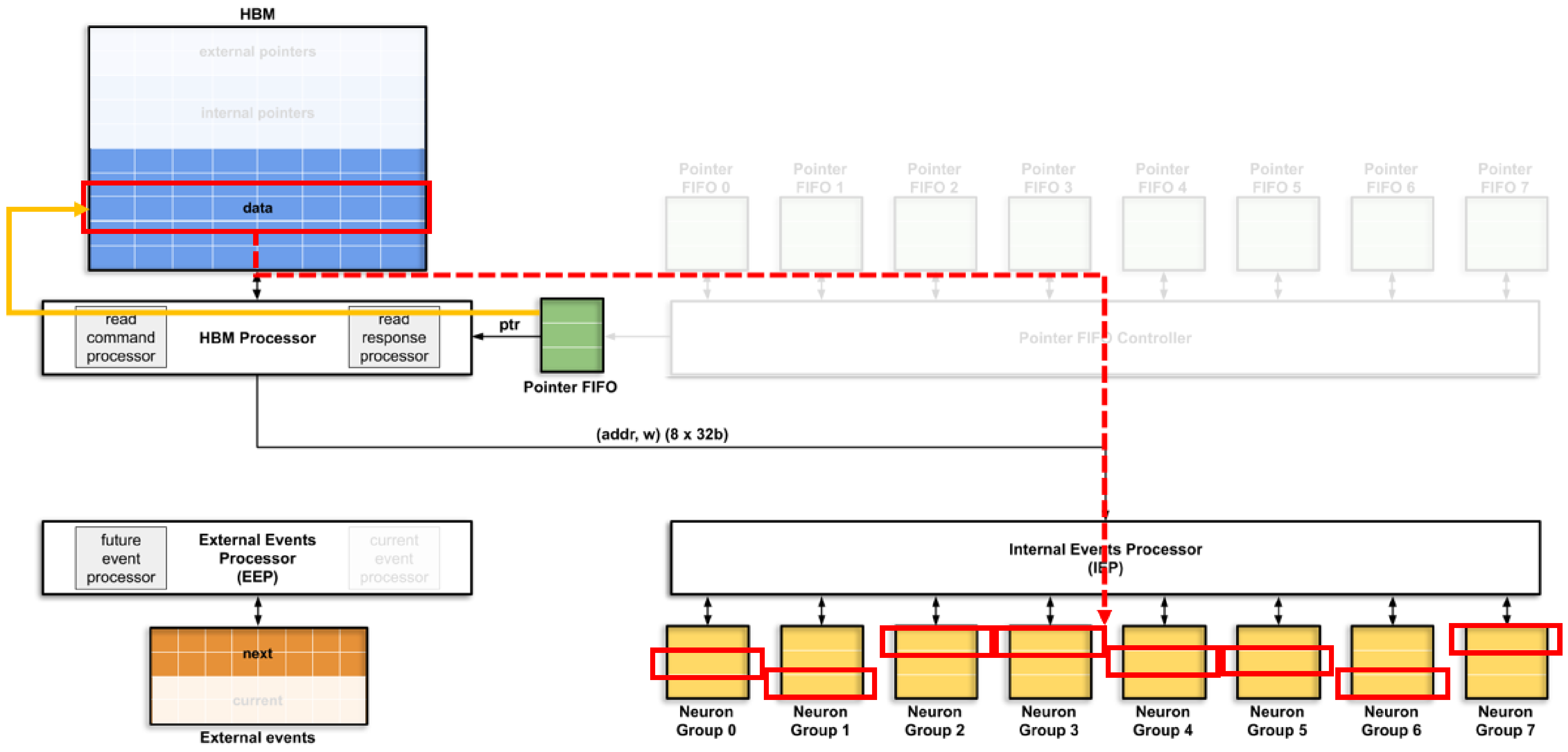


Phase 2: Spike processing and Neuron Dynamics

- Pop pointers from Pointer FIFO until empty
- Continuously (burst) read HBM data starting at each pointer base address
- Use the HBM data to update internal neuron state variables and send spike event information to off-core destination

Properties:

- Since there are 8 neuron groups, each synapse stored in the HBM data section consumes 256/8 = 32 bits
- The 8 synapses are aligned to the 8 neuron groups → 2^{17} neurons split into 8 groups requires 14 address bits
- Off-core events (intra-FPGA and inter-FPGA) are also stored using 32 bits.

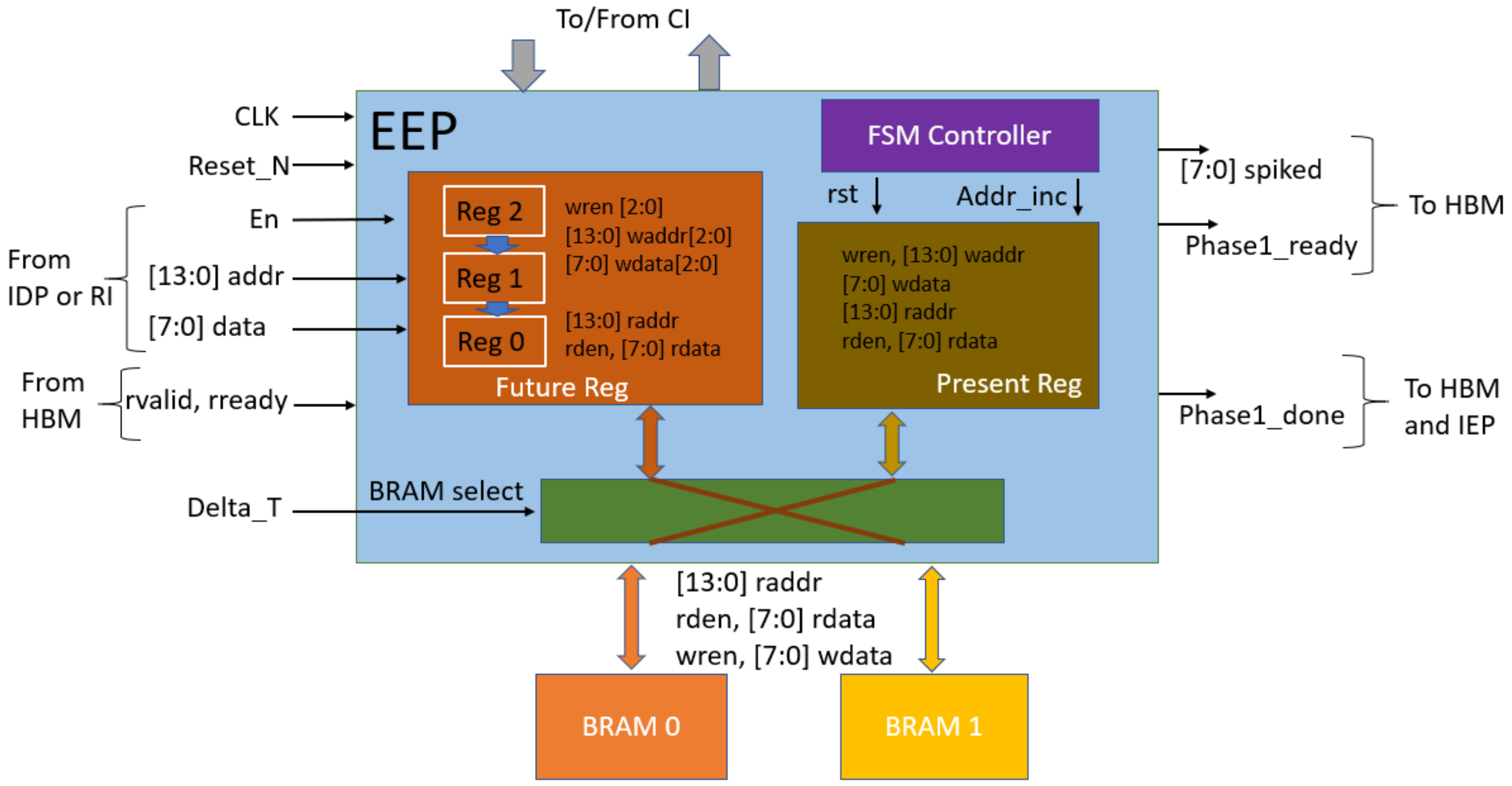


7 Command Interpreter (CI)

8 External Event processor (EEP)

EEP serves as the controller interface between the IDP or Router Interface and BRAMs used to store the external (axonal) inputs either from the host or from neighboring peer cores.

It consists of two ping-pong buffers (BRAM0 and BRAM1) switching between future inputs and present inputs at each Delta-T cycle. This helps avoiding any conflict between write from RI or IDP and read to HBM. At every algo time step, future register is used to fill input events into either BRAM 0 or BRAM 1 while Present register is used to read input events from the other BRAM. In next timestamp, they are swapped. (Each of the BRAM here are of size 128Kx1 (2D array of 16Kx8 and single-port read-first configured.)



Each BRAM has a pipeline depth of 3. Thus, it is expected to fill the 3-stage pipeline before going into read cycle stage in Present register. Present BRAM read advances synchronously with HBM read commands. The detailed block diagram is shown below, and transition between FSM states are discussed.

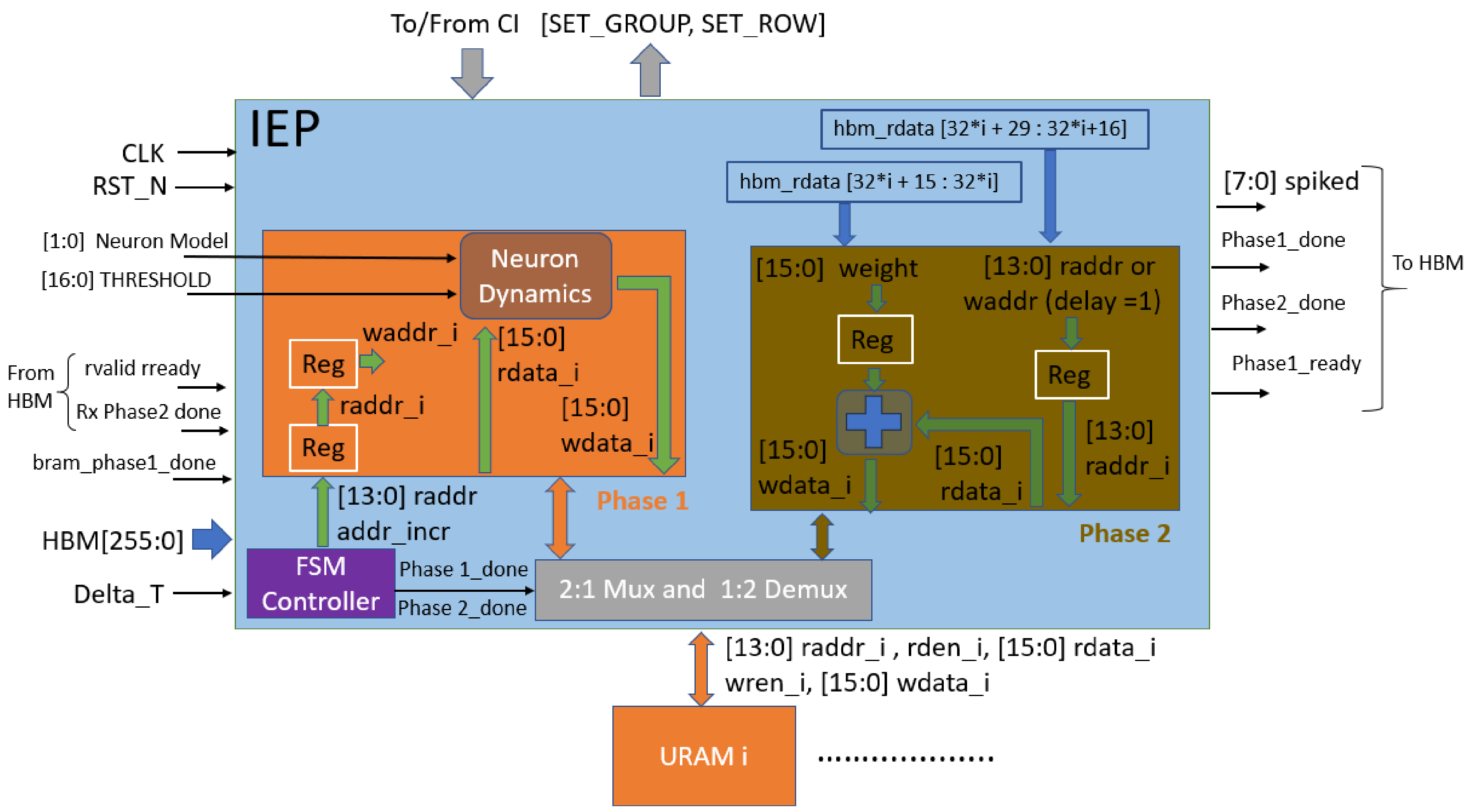
As BRAM has a read delay of 3 cycles, write address, enable and data in the Future Register have 3 pipeline stages.

In the READ INPUTS state, Read Address is incremented by 1, if HBM responds with a rready and rvalid for current input address. Phase 1 is done after Write Address (which is Read Address – PIPE DEPTH) reaches the maximum limit. After every Read, the same input address is cleared through writing zero.

FSM State	Functionality and Condition	Next State and Exit Condition
RESET	Addr Reset = 1, Read En = 0, Addr Increment = 0	IDLE
IDLE	Addr Reset = 1	FILL PIPE (if next algo timestep, Delta T =1)
FILL PIPE	Read En = 1, Addr Increment = 1 (If Read Address < PIPE DEPTH)	READ INPUTS (when Read Address = PIPE DEPTH)
READ INPUTS	Read En = 1, Addr Increment = 1 (If HBM rvalid, rready = 1)	PHASE1 DONE (when Write Address = BRAM ADDR LIMIT)
PHASE1 DONE		IDLE
DEFAULT		RESET

9 Internal Event processor (IEP)

IEP handles the internal events in Phase 1 to detect which of the internal neurons have fires and sends the spike information to HBM processor. In Phase 2, It receives the destination neuron and synaptic weight to update the membrane potential.



Both Phase 1 and Phase 2 operates on 8 neurons in parallel (one from each neuron group), thus neuron groups are address aligned to [13:0]. In Phase 1, Read and Write Address is controlled by the FSM to increment the read address sequentially and send the spikes if neurons have exceeded membrane threshold.

In Phase 2, while popping out the Pointer FIFOs filled from Phase 1, IEP receives a 256-bit data packet meant for 8 neurons. If the MSb ([31])of 32-bit packet is 1’b0, then the packet is understood as a message intended for internal neuron. 13-bits ([29:16]) of the message denotes the address of the destination neuron within the neuron group and lowest 16-bit ([16:0]) denotes the synaptic weight associated with it. If the MSb ([31])of 32-bit packet is 1’b1, then the 32-bit packet is sent off-core via RI.

URAM has always one cycle read latency. Thus, Write Address (waddr) is a single-clock delayed version of Read Address (raddr). Write Data (wdata) for both Phase 1 and Phase 2 is calculated as a function of Read data (rdata).

These are the following membrane potential updates (wdata) in Phase 1 and Phase 2.

Phase 1:-

```
if (uram_rdata_i > threshold)         uram_wdata_i = 16'd0;
        else begin if (neuron_model==2'd0) uram_wdata_i = 16'd0;
              else if (neuron_model==2'd1) uram_wdata_i = uram_rdata_i + i+1;
              else if (neuron_model==2'd2) uram_wdata_i = uram_rdata_i - (uram_rdata_i >>> 3);
              else if (neuron_model==2'd3) uram_wdata_i = uram_rdata_i; end
```

Phase 2:-

```
uram_raddr_i <= exec_hbm_rdata[32*i+29:32*i+16];
if (exec_hbm_rdata[32*i+031]) exec_hbm_rdata_reg_7 <= 16'd0;
      else                    exec_hbm_rdata_reg_i <= exec_hbm_rdata[015+32*i:32*i];
uram_wdata_i = uram_rdata_i + exec_hbm_rdata_reg_i;
```

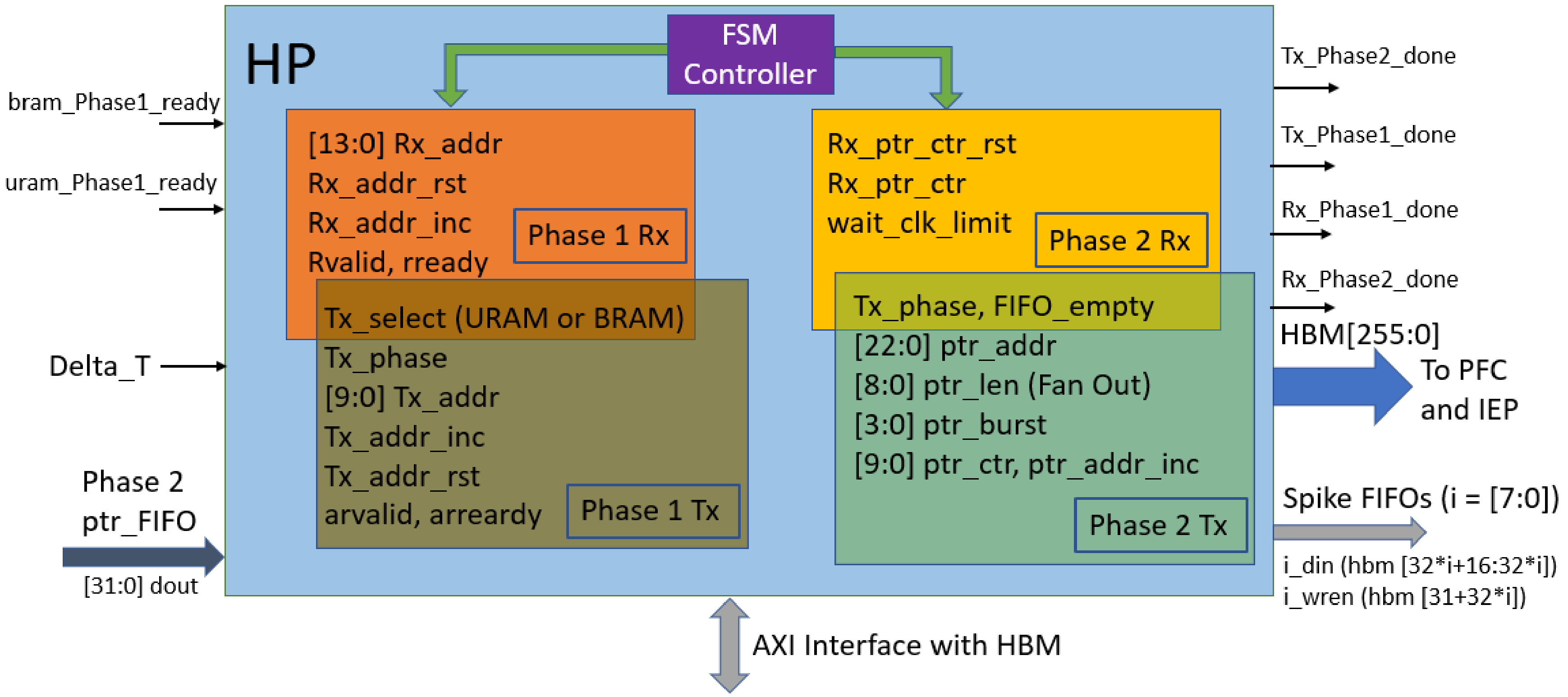
If there are multiple writes to same neuron address consecutively (two rvalid), we need to bypass the read to that neuron address as the previous write would not have executed yet. In this case, a delayed version of previous wdata is used to calculate the immediate next wdata. This will be discussed in a separate timing diagram.

FSM State	Functionality and Condition	Next State and Exit Condition
RESET	Addr Reset = 1, Read En = 0, Addr Increment = 0	IDLE
IDLE	Addr Reset = 1 (if Delta T =1)	FILL PIPE (if next algo timestep, Delta T =1) READ URAM0 or WRITE URAM0 (if phase 2 done and ci2iep not empty)
READ URAM0	URAM Read En = 1	READ URAM1
READ URAM1	URAM Read En = 1 iep2ci_wren =1 and ci2iep_rden=1 (if iep2ci not full)	IDLE (if iep2ci not full)
WRITE URAM	ci2iep_rden = 1	IDLE
FILL PIPE PHASE1	URAM Read En = 1, Addr Increment=1	WAIT BRAM PHASE1 DONE (if URAM RADDR >= 14’d1)
WAIT BRAM PHASE1 DONE	Wait	PUSH PTR FIFO
PUSH PTR FIFO	URAM Read En = 1, Addr Increment=1 ((If HBM rvalid, rready = 1)	PHASE1 DONE (when Write Address = BRAM ADDR LIMIT)
PHASE1 DONE		POP PTR FIFO
POP PTR FIFO	URAM Read En = 1 ((If HBM rvalid, rready = 1)	PHASE 2 DONE (When HBM_rx_phase2_done = 1)
PHASE2 DONE		IDLE

10 HBM processor (HP)

In Phase 1, HP reads the external and internal pointers sequentially (in burst mode) based on the Rx_addr (Receive Command address) increment from FSM and provides the 256-bit packet to PFC (which later determines which of the 32b packets to push to pointer FIFO based on the spike events). Rx_addr denotes the actual neuron address corresponding to the pointer location while Tx_addr (Transmit command address) stores the HBM address sent as a command to HBM.

10-bit Tx-addr is translated to **hbm_araddr = {5'd0, {8'd0, tx_select, tx_addr, 4'd0}, 5'd0}.** It is incremented by 1 (after 16 bursts) when **HBM responds with arready = 1.** Rx_addr is incremented when one 256-bit packet read is complete **(HBM responds with rvalid = 1).** Phase 1 is complete when **Rx_addr** reaches the **ADDRESS LIMIT.**



In Phase 2, Pointers are popped out from the FIFO and sent to HBM for fetching post synaptic connections. In the 32-bit pointer, lowest 22-bits denotes the ptr_addr while upper 9-bits denotes the ptr_len (Fan out of the Neuron denoting which other neurons the spike to be sent). 22-bit ptr_addr is translated to **hbm_araddr = {5'd0, ptr_addr, 5'd0}.**

In each HBM read command, a ptr_ctr is used to keep track of ptr_len. Maximum ptr_burst being 16, the ptr_ctr and ptr_addr is incremented by ptr_burst+1 after every burst read command until **ptr_ctr == ptr_len**. Phase 2 is done when **rx_ptr_ctr == tx_ptr_ctr** which means all the pointers has been served. There is also a safe **wait clock limit of 15** to make sure we wait until the PFC circles through the **round robin arbitration** from all 8 pointer FIFOs.

FSM State	Functionality and Condition	Next State and Exit Condition
RESET		IDLE

IDLE	tx_addr_rst = 1
	tx_done_rst = 1 (after Delta T)
		SEND INPUT READ CMD
		(if next algo timestep, Delta T =1)

SEND INPUT READ CMD	Arvalid = 1 Tx_addr_inc (when Arready = 1)	SEND OUTPUT READ CMD (with tx_addr_rst = 1, tx_select_inc = 1) (when Tx_addr = INPUT_ADDR_LIMIT)
SEND OUTPUT READ CMD	Arvalid = 1 Tx_addr_inc (when Arready = 1)	PHASE1 DONE (when Tx_addr = OUTPUT_ADDR_LIMIT)

PHASE1 DONE	tx_addr_rst = 1, tx_select_inc = 1
	tx_phase_inc = 1, tx_ptr_ctr_rst = 1
		POP PTR FIFO

POP PTR FIFO	ptr_addr_set = 1, ptr_FIFO_rden = 1 (if ptr_FIFO_empty = 0)	SEND PTR READ CMD (if ptr_FIFO_empty = 0) PHASE2 DONE (wait_cnt = wait_limit)
SEND PTR READ CMD	Arvalid = 1 ptr_addr_inc (when Arready = 1)	POP PTR FIFO if (ptr_ctr[8:4] == ptr_len[8:4])
PHASE 2 DONE	tx_phase_inc = 1	IDLE

Fig:- Transmit Command (To HBM) State Machines and their transitions

Read data[255:0] is sent to IEP and Spike FIFO controller (which eventually pushes the off-core events into FIFO if the MSb is 1 in a 32-b packet, Lowest 17-bit in the external message denotes the destination neuron ID).
Similarly, IEP decodes the HBM packet and updates the membrane potential for the internal destination neurons.

FSM State	Functionality and Condition	Next State and Exit Condition
RESET		IDLE

IDLE	rx_addr_rst = 1
	rx_done_rst = 1 (after Delta T)
		WAIT BRAM PIPE
		(if next algo timestep, Delta T =1)

WAIT BRAM PIPE	rx_addr_rst = 1	READ INPUT PTR (if bram_phase1_ready =1)

READ INPUT PTR	rready = 1
	rx_addr_inc =1 (when rvalid = 1)
		WAIT URAM PIPE
		(when rx_addr = {ADDR_LIMIT, ADDR_MOD})

WAIT URAM PIPE	rx_addr_rst = 1	READ OUTPUT PTR (if uram_phase1_ready =1)

READ OUTPUT PTR	rready = 1
	rx_addr_inc =1 (when rvalid = 1)
		PHASE1 DONE
		(when rx_addr = {ADDR_LIMIT, ADDR_MOD})

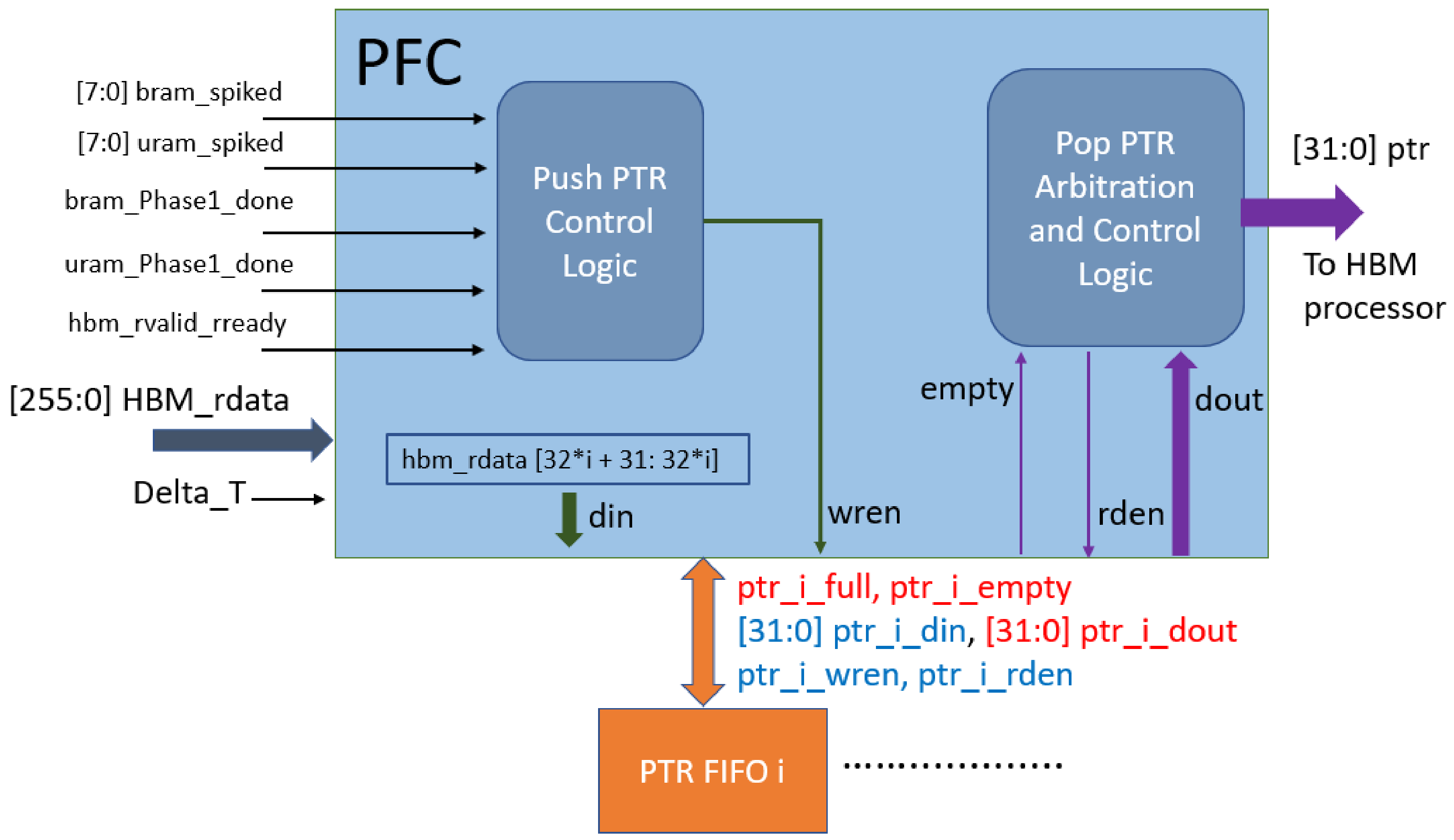
PHASE1 DONE	rx_ptr_ctr_rst = 1	READ SYNAPSE DATA
READ SYNAPSE DATA	rready = 1	PHASE2 DONE If Tx_phase2_done and (rx_ptr_ctr == tx_ptr_ctr)
PHASE2 DONE		IDLE

Fig:- Receive Command (from HBM) State Machines and their transitions

11 Pointer-FIFO Controller (PFC)

PFC intakes 8-bit spike events from IEP and EEP and done events from both. After 32b pointers corresponding to each neuron address are read from HBM, the external or internal pointers with spike event on are pushed into one of the 8 pointer FIFOs.

During Phase 2, a round robin arbitration logic is followed to pop pointers and sent to HBM for synapse read data.


PFC
[7:0] bram_spiked
[7:0] uram_spiked
bram_Phase1_done
uram_Phase1_done
hbm_rvalid_rready
[255:0] HBM_rdata
Delta_T
Push PTR Control Logic
hbm_rdata [32*i + 31: 32*i]
din
wren
Pop PTR Arbitration and Control Logic
empty
rden
dout
[31:0] ptr
To HBM processor
ptr_i_full, ptr_i_empty
[31:0] ptr_i_din, [31:0] ptr_i_dout
ptr_i_wren, ptr_i_rden
PTR FIFO i
…………………