2018 24th IEEE International Symposium on Asynchronous Circuits and Systems

Loihi Asynchronous Neuromorphic Research Chip

Andrew Lines, Prasad Joshi, Ruokun Liu, Steve McCoy, Jonathan Tse, Yi-Hsin Weng, Mike Davies
Intel Labs
Hillsboro, Oregon 97124
andrew.lines@intel.com

***Abstract*—Intel's "Loihi" neuromorphic research chip implements spiking neural networks on 128 custom cores with 1024 neurons each. It supports a wide variety of algorithms inspired by computational neuroscience, notably on-chip learning. Loihi's design is specified in the Communicating Sequential Processes (CSP) language and implemented by an automated flow that generates two-phase bundled-data (BD) asynchronous pipelines with pulsed latch datapaths. This optimizes area and energy while using a mostly standard cell library and simplifies integration with synchronous collateral. The pre-silicon design was verified by static timing analysis, back-annotated gate-level simulation, and FPGA emulation. Tunable delay lines provide sufficient timing margin in extreme corners such as near-threshold-voltage. Loihi was manufactured in Intel's 14nm ASIC process in Q4 2017 and is functional from 0.55V to 1.25V.**

***Index Terms*—neuromorphic, asynchronous, bundled-data**

I. Introduction

Loihi implements spiking neural networks with a "neuromorphic" approach that emulates biological behavior by sending spikes between neurons which excite or inhibit downstream neurons to spike. Algorithms and applications are covered in [1]. Loihi incorporates on-chip learning and per-synapse delays, advances beyond IBM's TrueNorth [2].

Loihi has an 8×4 mesh fabric where each router connects to 4 local neuron cores. Links on the south edge of the mesh connect off-chip and to 3 embedded x86 CPUs (the only synchronous units). The fabric has two parallel meshes, one for read/write requests and another for read responses. Embedded or off-chip CPUs can configure and inspect the state of all neuron cores. This fabric architecture descends from the NanoMesh [3]. The key new message is a spike, which lets a neuron core indicate when it spikes to a fanout neuron core. Spikes may travel through either mesh.

The neuron core is mostly a linear pipeline, following the life cycle of a spike. Each incoming spike encodes an axon index, which points to a variable length list of synapse entries (with many formats, potentially encoding a weight, delay, fanout neuron compartment, and tag). The list is walked and the synapse weight is accumulated for each fanout. Simultaneously, the core iterates through its neurons, inspecting the accumulated weights for the previous timestep, and computing whether the neuron should fire and updating its state for the next timestep. If a neuron fires, the end of the pipeline walks a variable length list of destination cores and axons that it sends spikes to. Learning occurs between N timesteps, when it replays recently active input spikes, and updates the synapse state according to microcode programmable learning rules, such as STDP or supervised learning.

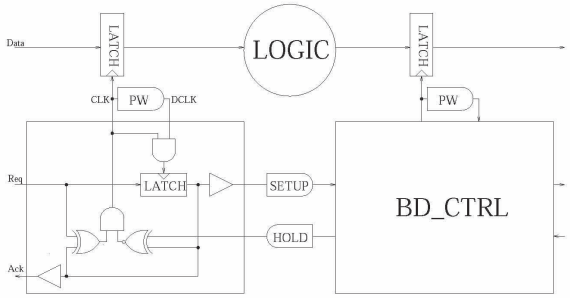

Fig. 1. Two stages of asynchronous 2-phase BD pipeline with TDLs for pulse-width, setup, and hold timing.

II. CSP to RTL

Cells written in CSP are automatically translated into Verilog RTL with the addition of bundled-data pipeline controllers, C-elements, state loops, and conditional communication interfaces. This uses an enhancement of the Proteus flow previously used for quasi-delay-insensitive (QDI) designs [4]. The topology of the circuit is derived from CSP, where each cell has one stage of logic. Additional pipelining can be added to ports or inside the combinational logic by retiming. State variables loop back through two pipeline stages.

The post-synthesis netlist is simulated in Verilog. Alternate versions of the pipeline controllers wait for a posedge clock, so they can be mapped to an FPGA for emulation. The FPGA uses the same netlist as the ASIC to capture all logical features including slack; however it differs in precise timing.

Synthesis targets Intel's standard cell library plus 2 and 3 input C-elements with combinational keepers. We added tunable delay line (TDL) cells, implemented as current-starved inverter chains, where the inverter power supplies are connected to Vdd and GND through either 1 or 3 transistors in series, selectable by a configuration bit. A QDI shift register chain initializes these configuration bits and other pseudo-static settings.

The most complex new cells are BD controllers, built from smaller standard cells but routed as placeable hard macros; this is necessary to alter their timing model to hide combinational loops. To pipeline, we chose a new controller design because we needed to tune the pulse width of the generated clock. It is similar to Click [5] but replaces the toggle flop with a latch. Although larger than Mousetrap [6], the clock is normally low

978-1-5386-5883-3/18/$31.00 ©2018 IEEE
DOI 10.1109/ASYNC.2018.00018

so glitches don't propagate. The datapath uses latches. See Figure 1.

Additional controllers implement CSP's conditional send/receive operations, by skipping either the input or output handshake based on a control bit from the datapath. Other controllers must be instantiated as subcells, such as metastable arbiters, a crossbar controller, and a controller to clock a standard synchronous SRAM (modified to expose internal signals to match delay better).

Most state is stored in ECC protected 6T SRAM in each core. Most are implemented with multiple banks, and many perform a read-modify-write operation with a multi-cycle delay until the write is performed. If the SRAM reads a recently written address, it must flush the pipeline, but this rarely causes stalls.

III. Place & Route & Static Timing

From RTL, the flow is more standard, using Synopsys tools for SPR, STA, FEV, DRC, LVS, SPICE, SDF simulation, and IR drop. Cadence Virtuoso was used for custom cells, floorplanning, and chip assembly.

The primary novelty is how we define the timing arcs of the controllers and the timing constraints such that commercial tools can handle them. All timing arcs of the controllers are characterized by Nanotime, but we post-process the LIB to cut arcs that would form combinational loops, yielding different LIB's for each timing mode.

First we satisfy minimum pulse-width of the generated clock. BD controllers potentially have problems with the clock pulse dispersing as it propagates through the clock tree. Our TCL scripts propagate rising and falling clock edges, checking for insufficient voltage swing at all stages, and evaluating the min_pulse_width_high constraint at the latches. If this fails, the script adds delay to stretch the pulse.

A setup check sets the delay on the request wires. Starting from the output clock on the pipeline controller, the short path goes to the local datapath, through the combinational logic, and ends on the D pin of a downstream latch. The long path goes through the pulse-width delay, forward through the request TDL, through the next controller to the CLK pin of the downstream latch.

A hold check sets the delay on the acknowledge wires. Starting from a virtual clock pin inside the controller, the short path goes through the falling clock to the local datapath latches. The long path goes backward through the acknowledge to the upstream controller, through its rising clock, through its datapath latches, and back through the combinational logic to the D pin of the latches. Although BD handshakes typically have good hold margin, latch placement and clock trees could still cause failures.

Other timing constraints limit the maximum local cycle time between controllers, check hold races that arise on the request/acknowledge, limit slew rates, etc.

IV. Results

Loihi is 60mm^2 in 14nm (Figure 2), with 53mm^2 in neuron cores and fabric. Loihi has 2.1B transistors, 85% in 33MB of SRAM. At nominal 0.75V, the frequencies of units vary from 860MHz in mesh routers, to 100MHz in neuron state updates. Many anticipated optimizations are not yet done, such as optimal slack-matching. Loihi uses very conservative timing margins. Loihi is functional from 0.55V to 1.25V; slower TDLs are needed below 0.65V, and the CPUs fail below 0.55V. Figure 3 graphs speed and energy over voltage for a benchmark network.

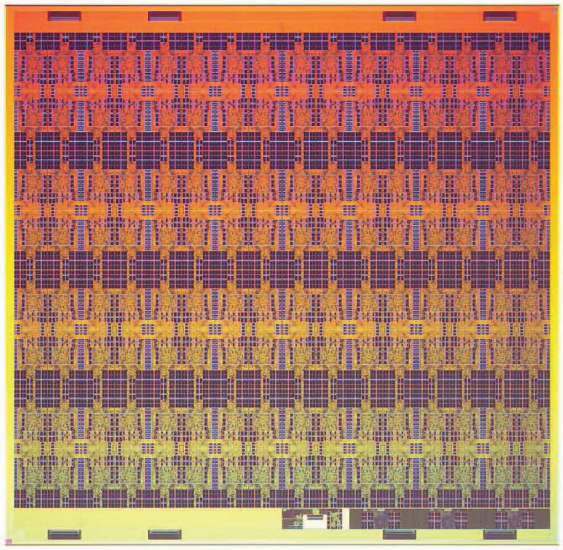

Fig. 2. Die photo.

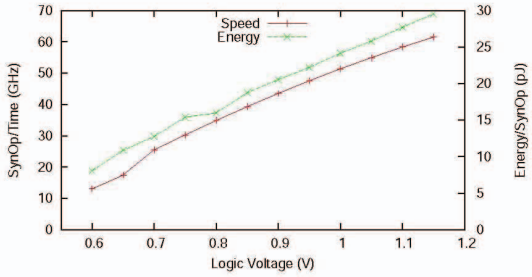

Fig. 3. Speed and energy of synaptic operations for 128K neurons, spiking each timestep with fanout to 1K 1-bit synapses.

References

[1] M. Davies et al, *Loihi: A Neuromorphic Manycore Processor with On-Chip Learning*, IEEE Micro, Jan 2018.

[2] P.A. Merolla et al, *A million spiking-neuron integrated circuit with a scalable communication network and interface*, Science #6197, Aug 2014.

[3] J. Tse, A. Lines, *NanoMesh: An Asynchronous Many-Core System-on-Chip*, IEEE ASYNC, 2011.

[4] P. Beerel, G. Dimou, A. Lines, *Proteus: An ASIC flow for GHz asynchronous designs*, IEEE Design & Test, Nov 2011.

[5] A. Peeters, M. Wit, W. Mallon, *Click Elements An Implementation Style for Data-Driven Compilation*, IEEE ASYNC, 2010.

[6] M. Singh, S. Nowick, *MOUSETRAP: High-Speed Transition-Signaling Asynchronous Pipelines*, IEEE VLSI, June 2007.