Digital System Design and Fullchip Integration for Asynchronous Stochastic Neural Accelerator

Adhiraj Datar

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2021-161
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2021/EECS-2021-161.html

June 13, 2021

Digital System Design and Fullchip Integration for Asynchronous Stochastic Neural Accelerator

by Adhiraj Datar

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II**.

Approval for the Report and Comprehensive Examination:

Committee:

Professor Sayeef Salahuddin
Research Advisor

6/11/2021
(Date)

* * * * * * *

Professor Yakun Sophia Shao
Second Reader

5/13/2021
(Date)

Digital System Design and Fullchip Integration for Asynchronous Stochastic Neural Accelerator

Adhiraj Datar

May 2021

Abstract

Several NP-hard combinatorial optimization problems such as vehicle routing, optimal graph traversal and automatic ASIC place-and-route have direct practical applications. However, as the demand for highly scaled processing of these problems grows, traditional sequential and synchronous processor-based solutions incur exponential processing penalties and fail to keep up in performance. This project outlines the Parallel Asynchronous Stochastic Sampling Optimizer (PASSO) — a novel neural accelerator based on the Ising model that demonstrates a theoretical 250x power and 3x performance speedup over state-of-the-art systems on a 100-node Max-Cut problem. Specifically, this work highlights the design choices and implementation of the digital configuration, sampling and data streamout systems in the PASSO accelerator. In addition, the report covers the physical design and integration of the chip at the top level which was performed to submit the first PASSO design (PASSOv1) for tapeout in the GlobalFoundries 12LP process in April 2021.

Acknowledgements

I would like to acknowledge the entire project team including Saavan Patel, Philip Canoza and Steven Lu for their dedicated work in making this project a reality. Furthermore, I would like to thank Professor Sayeef Salahuddin for his sponsorship and excellent guidance for this project and my undergraduate research, and Professor Sophia Shao for her invaluable feedback and support to the entire project team.

In addition, I would like to thank Harrison Liew, Vighnesh Iyer, and Ioannis Karageorgos for their continued advice which was vital in making the PASSOv1 chip tapeout possible.

Furthermore, I would like to thank the Design Technology Architecture and Co-Optimization team at GlobalFoundries for providing essential standard cell and memory IP required to physically implement this project.

I would also like to thank Nirmaan Shanker and Suraj Cheema for supporting my undergraduate research.

Mostly, I would like to family and in particular my parents for supporting me throughout my education.

Contents

1 Introduction

1.1 Motivation

As the demand for data processing at high scales continues to increase, traditional computing architectures find it difficult to keep up with the growing complexity of the most difficult computing problems. Many optimization problems with practical applications such as vehicle routing, optimal traversal through a graph, or automatic ASIC place and route are considered NP-hard optimization problems. In other words, the computational difficulty of these problems increases exponentially with the scale of the problem to be optimized. Current computing heuristics used to solve these problems mostly rely on centralized, clock driven approaches where processors source instructions from memory and evaluate these sequentially to get an optimization to the presented problem. This approach scales poorly with size since it is limited by its clock driven sequential processing that gives up on much of the inherent parallelism often present in physical optimization problems, and is reliant on programmers to efficiently design algorithms that can be distributed across multiple von-neumann computing cores.

Recent work from the Salahuddin group has implemented decentralized, intrinsically parallel computing architectures to get around the von-neumann bottleneck. In these decentralized computing systems, each processing component has its own memory unit and performs computations stochastically using asynchronous, event-driven local interactions with other spatially correlated processing components. The most recent asynchronous stochastic processor designed by the group is based on a system known as the Ising model, which uses a distributed fabric of binary neurons with local neighbor-to-neighbor interaction strengths that optimizes problems by finding a minimum energy state. This processor - the "Parallel Asynchronous Stochastic Sampling Optimizer" (PASSO) - is a distributed, stochastic, asynchronous processor capable of efficient hardware based sampling and stochastic parallel updates and has applications in solving many NP-hard optimization problems such as the integer factorization, travelling salesman, boolean SAT, max-cut problems and many more.

This work focuses on the design and implementation of the synchronous digital systems used to configure and sample the PASSO processor as well as the back-end ASIC physical design and chip level integration of the PASSO processor performed to tape-out the first version of the PASSO processor (PASSOv1) in the GlobalFoundries GF14LPP/12LP process in April 2021.

1.2 Outline

First, this report briefly outlines some prior work performed in this area, including some earlier attempts at NP-hard optimization problem acceleration. Then, the report describes a summary of the PASSO architecture and provides a brief overview of the Ising model accelerator analog neuron architecture and design previously developed by other students in the Salahuddin group at Berkeley.

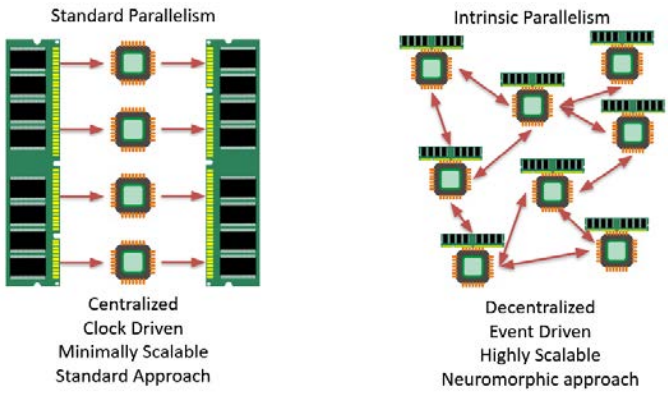


Figure 1: Potential advantages of an asynchronous decentralized compute fabric towards solving scaled optimization problems.

Next, the report describes in detail the digital functions and systems required to configure the PASSOv1 processor with local interaction weights and biases and to synchronously sample the updated states of the neurons generated through local interactions within the PASSOv1 processor. In this section, the report describes the architecture and circuit level design choices used to implement all of the required synchronous digital features of the PASSOv1 processor and includes functional verification of each of these digital systems.

Lastly, the report details the back-end physical ASIC design of the PASSOv1 processor. This includes the synthesis and place-and-route of the core, as well as the top level integration of the entire chip including timing closure, power planning, and interfacing with the IO. Finally, the report contains a brief plan for further work to be done on the PASSOv1 processor in the coming months.

2 Background

2.1 Ising Model

An Ising model system physically represents the behavior of magnetic objects in the field of statistical mechanics. Specifically, the model consists of a lattice of binary spins (neurons) that asynchronously update their state in parallel through local interactions between neighboring spins. This form of computing uses an interaction coefficient between connected spins that updates the spin state stochastically in an asynchronous, spatially correlated, and massively parallel manner.

The optimization problem encoded in each Ising computer is to find the configuration of spinsin the lattice that minimizes the energy of the entire system. This is an instance of a combinatorial problem that the Ising system solves through distributed and highly parallel interaction/event-driven state updates.

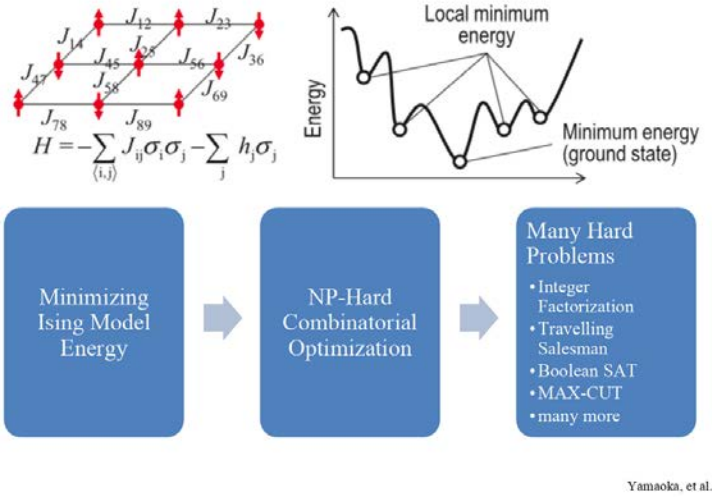


Figure 2: Schematic description and applications of a Ising model ground-state-search process.

As a result, the model lends itself very well to many NP-hard optimization problems such as the integer factorization, travelling salesman, boolean SAT, max-cut problems and many more [1].

In general, a two-step process is used to solve an NP-hard optimization problem on an Ising computer. The first step involves converting the optimization problem to an equivalent formulation the solution of which corresponds to finding the minimum energy state (ground-state) of the Ising machine used to solve the problem. This step generates a set of interaction weights and biases that encodes the optimization problem to be solved in the Ising machine used to solve the problem. The next step uses the Ising machine itself to find the minimum energy configuration of the neurons. Once the minimum energy configuration is achieved, the state encoded within the Ising computer gives the solution to the original NP-hard optimization problem.

2.2 Related Works

Several studies have proposed to accelerate optimization related calculations using the Ising system and other physical non von-neumann approaches. Some of these are implemented as digital asynchronous systems. Yamaoka, et al. [1] implemented an Ising machine chip that integrates 20K spin units and uses asynchronous random pulses applied to the spin states to escape local minima and converge to the global Ising machine energy minimum (ground-state search) in a parallel and asynchronous manner. Aramon, et al. [2] presents a digital annealer to solve fully connected optimization problems. This digital annealer performs a modified simulated annealing algorithm with parallel trials

run on custom CMOS hardware to achieve a two-orders-of-magnitude speedup for fully connected non-sparse spin glass problems over traditional single-core implementations of simulated annealing.

Some other approaches used to implement Ising machines use analog electronics to model neuron interactions and perform state updates. Wang, et al. [3] presents the realization of Ising machines using coupled nonlinear oscillators with logical values encoded in the phase states of the oscillators. This oscillator based machine is capable of solving ground state search optimization problems and has been demonstrated with max-cut and half adder problem solutions. Ahmed, et al. [4] report a probabilistic self annealing compute fabric consisting of a network of 560 spin units. These spin units are implemented as an array of 28x20 bistable ring oscillators that interact with 6 neighboring spin units in a hexagonal lattice to perform ground-state searches and solve NP-hard combinatorial optimization problems.

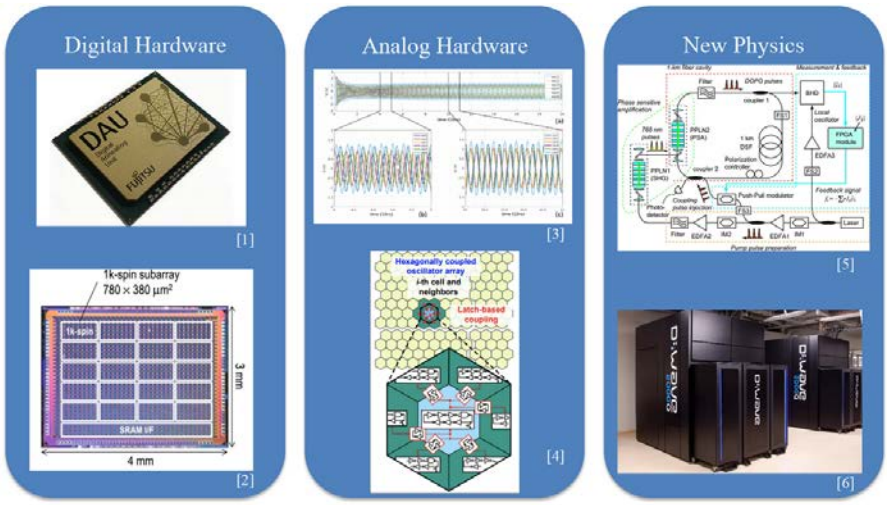
Another set of related works that implements Ising machines to solve NP-hard optimization problems relies on novel physics and devices. Inagaki, et a. [5] present an all-to-all coupled Ising model compute fabric with 2048 spin units. These spin units are realized using degenerate optical parametric oscillators (DOPOs) that encode states in a bistable 0 or π phase state.

While all of these related studies showcase implementations of NP-hard combinatorial optimization problem solving using Ising machines, the system described in this report is fundamentally different from each of them. The PASSO architecture outlined in this report features mixed signal neurons with intrinsic noise integration and stochastic parallel update capacity which allows for massively parallel, efficient hardware based sampling with a low memory overhead and an algorithmic guarantee of convergence to the optimization problem. This architecture is discussed in the following subsection.

2.3 Parallel Asynchronous Stochastic Sampling Optimizer (PASSO)

The PASSOv1 architecture allows for massively parallel asynchronous neuron state updates based solely on local interactions between neurons. The systolic array implemented in the PASSOv1 chip consists of 256 bistable mixed-signal neurons arranged in a 16x16 square grid. Each of these neurons is connected to local neighbors in a king's move configuration. In other words, each neuron's state is a stochastic function based on its interaction with the states of the 8 immediately adjacent neurons and the weights and biases that describe the interaction of each pair of connected neurons.

A key feature of the PASSOv1 processor is that the neuron states are not sampled synchronously in subsets on a global system clock. Rather, the state sampling is a continuous and asynchronous process which intrinsically occurs for every neuron in the chip. As a result, the stochastic state updates too are asynchronous and intrinsically parallel since each neuron in the systolic array can update its state asynchronously and in parallel with all the neurons in the chip. The processor can be programmed with a stationary set of weights and biases



[1] Aramon, Maliheh, et al. *Frontiers in Physics* 7 (2019): 48.
[2] M. Yamaoka, et. al. IEEE Journal of Solid-StateCircuits51, 303–309 (2016).
[3] Wang, Tianshi, and Jaijeet Roychowdhury. *arXiv preprint* (2017)
[4] Ahmed, Ibrahim, *et al. Symposium on VLSI Circuits*. IEEE, 2020.
[5] Inagaki, Takahiro, et al. *Science* 354.6312 (2016): 603-606.
[6] Blais, A. M. Zagoskin, *Physical Review A* 61, 4 (2000).

Figure 3: Summary of related works, showing Digital Annealer from Fujitsu [1], Hitachi 1k spin subarray [2], oscillator based systems presented in [3] and [4], and "new physics" based systems such as the optical oscillator Ising machine [5] and the quantum annealer presented in [6].

for the neurons which allows for low static memory overhead at runtime. The structure of the neurons and sampling architecture also allows for multiplier-free synapses and fixed point computations, which eliminates one of the significant challenges faced by annealing algorithms which require high precision.

In the PASSOv1 chip, each neuron/spin unit contains an integrated noise source. The integration of noise into the spin unit itself allows local generation and usage of noise that does not rely on having a global noise path or a chip level input for noise unlike many of the related works referenced on Ising machine implementations. This form of noise generation with a dedicated, in-situ noise source for each neuron that has no dependence on global chip-level paths allows for highly distributed and asynchronous stochastic processing. One of the biggest advantages of this approach is that each neuron in the chip can update completely in parallel, independently of any global or synchronous paths in the chip. As a result, the optimization evaluation of the PASSOv1 chip is truly only limited by local interactions with neighboring neurons and the associated paths. This extremely high level of distributed parallelism and asynchronicity provides extremely large speedups (on the order of 1-2 magnitudes) over traditional synchronous optimization algorithms which update node states in a synchronous and sequential manner. Figure 4 shows a simulation of the parallel asynchronous processing architecture on a 150-node max cut problem. From the time series data, the PASSO asynchronous algorithm exhibits a 150 times speedup over the synchronous version since it can update several nodes in parallel and relies on

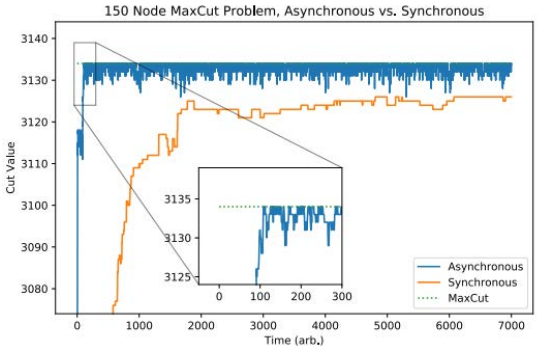


Figure 4: Simulated processing of 150-node max-cut problem on PASSOv1 shows an expected 150x performance speedup over a synchronous digital implementation.

distributed local interactions to converge to an energy minimum rather than using global paths and sequential clocked processing.

The PASSOv1 chip consists of 256 neurons arranged in a 16x16 grid with numbered rows and columns as shown in figure 5. At the top level, the PASSO chip has functionalities to program the chip with a set of weights and biases that encode a trained problem within the 256 neurons in the chip so that it can perform inference optimizations. The chip also implements a ready-valid handshake to begin sampling the asynchronously updated states on a synchronous sampling clock and stream the states out from an on-chip SRAM buffer. In addition, the chip has separate top-level reset signals for the neurons, the weight/bias configurations, and the digital sample streamout circuitry. Finally, the PASSOv1 chip also contains a small test cluster of 4 all-to-all connected neurons that will be used to perform direct programming, sampling and electrical measurements on the test cluster. The detailed implementation of each of these digital systems is the main work presented in this project report and is covered in detail in section 3 of the report. Figure 5 shows a top level view of the architecture and connectivity of the 256 neuron systolic array implemented on the PASSOv1 chip.

2.4 Mixed-Signal Neural Node Design

The PASSOv1 chip requires a node that can update asynchronously and in distributable, parallel fashion. The chip achieves this spin unit functionality with a mixed-signal neuron and synapse circuit. This node used in the processor has been developed by three students in the Salahuddin group—Saavan Patel, Steven Lu, and Philip Canoza—over the course of the PASSOv1 chip tapeout. The following section contains a very brief summary of the node circuit as background information for the project.

The node is a mixed-signal circuit that interacts with its 8 neighboring states

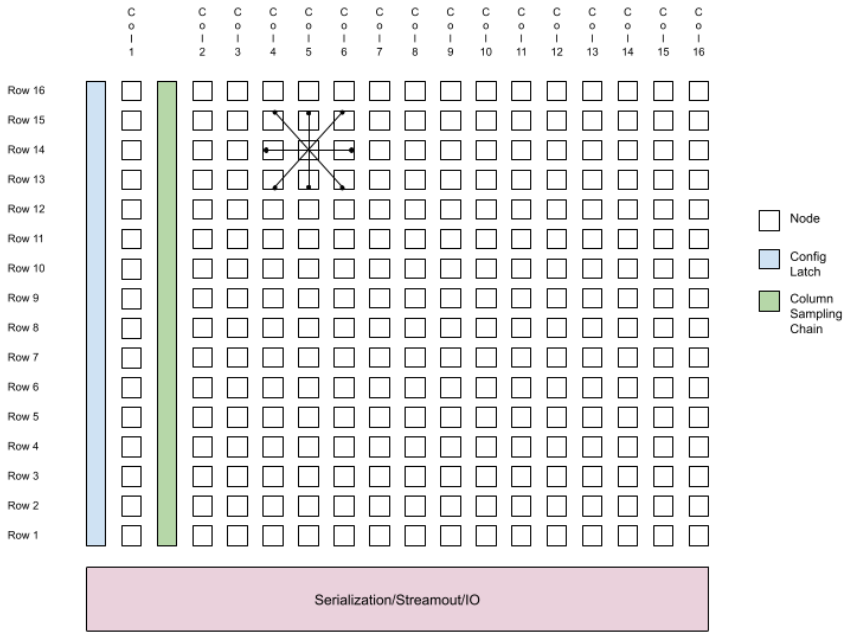


Figure 5: PASSOv1 architecture showing labeling of neuron rows and columns, relative location of interspersed digital configuration and sampling systems for column 1, and example of "king's move" connectivity of the neuron at row 14, column 5.

(denoted by the vector $\vec{h}$). The interaction is described by a set of weights and biases that are encoded into the configuration of every neuron. Specifically, the weights (denoted as the 8x8 matrix W) are a set of eight 8-bit vectors that describe the interaction of the neuron with each of its neighbors respectively, and the bias of each neuron (denoted as an 8-bit number b_v) represents the tendency of the neuron state to take on a binary value of 1. Finally, each neuron has 2 clamp bits which can be used to force the state of a particular neuron to a 0 or a 1. As a result, the complete configuration of each neuron can be encoded by a set of 74 bits: 64 bits for the interaction weights, 8 bits for the neuron bias, and 2 bits for the output clamp signals. Specifically, each neuron in the chip has a probably of assuming a state of 1 that is given by:

$$P(s_j = 1|s) = \sigma(\sum_{i \neq j}[W_{ij}s_i + b_j])$$

Through this relation, the node circuit is implemented as an asynchronous accumulation and activation operation. The accumulation function ($\sum_{i \neq j}[W_{ij}s_i + b_j]$)) is performed in the synapse subcircuit and the probabilistic activation function ($P(s_j = 1|s) = \sigma[\dots]$) is performed in the neuron subcircuit.

The neuron subcircuit is implemented using a noise generator and amplifier which takes shot noise from on-chip CMOS devices and amplifies it using a differential amplifier. This amplified noise is then compared with the output signal of the neuron's synapse in a sigmoidal comparator which results in a stochastic state output for the neuron.

The synapse circuit is implemented as a digital scale-and-accumulate adder block where the input neighboring states $\vec{h}$ are scaled by W_j^T and added to the bias b_j. Finally, the 7-bit digital state encoding is converted to an analog voltage that is passed into the sigmoidal comparator of the neuron.

Figure 6 shows a schematic representation of the node circuit showing the division of the stochastic node operation between the neuron and the synapse subcircuits. Figure 7 shows a stochastic time series of a simulated neuron where the three time series graphs represent 3 different input voltages V_{in} to the sigmoid comparator of the neuron. Figure 8 shows the layout of the node circuit. In this manner, the PASSOv1 processor implements the required functionality of the node circuit.

3 Digital Core-Logic Design

The main work presented in this project is the design and implementation of each of the digital systems required to configure the nodes of the chip, and the top-level integration of the entire PASSOv1 chip to get the design ready for the GF12LP multi-project wafer tapeout in April 2021. The following section details the digital functions required for the implementation of the chip and the design choices that were made in implementing the logic of these digital systems.

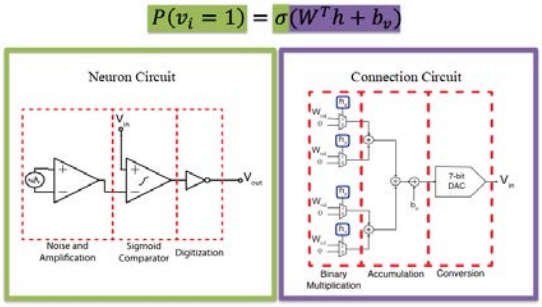


Figure 6: Schematic of node circuit showing division of stochastic operation between neuron and synapse subcircuits.

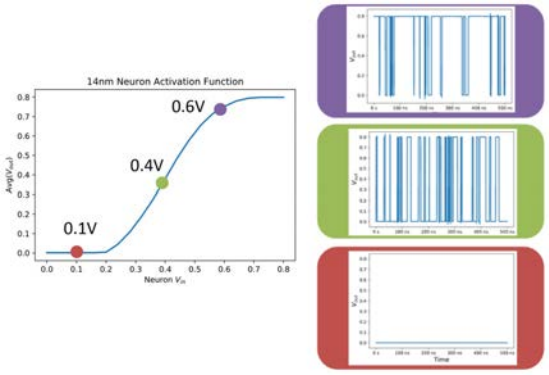


Figure 7: Time series of neuron switching activity with different synapse voltages.

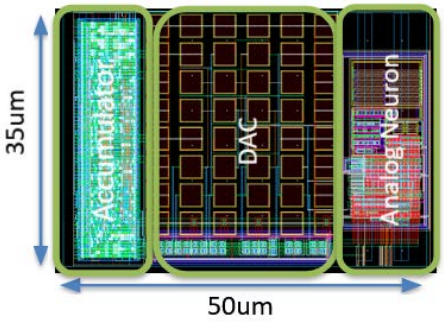


Figure 8: Node circuit layout in GF12LP.

3.1 Requirements and Specifications

The core of the PASSOv1 chip consists of 256 spin unit "nodes" arranged in a 16x16 square grid. Each of these nodes is connected to its 8 immediately adjacent nodes and updates its state asynchronously using a probabilistic function of the states of the adjacent nodes and precoded weights and biases that are programmed to the neuron for a corresponding combinatorial optimization problem.

In order to effectively use the PASSOv1 chip and interpret the inference data generated by the Ising machine, the following digital features and specifications are required on-chip to generate meaningful data and interface the chip with an external test board and computer so that the generated data can be effectively processed and analyzed:

- Chip configuration: In order to encode a combinatorial optimization problem in the processor and solve it, 74 aforementioned configuration bits (64 weights, 8 biases and 2 clamps) need to be encoded properly into every neuron in the chip. In addition, there are a total of 32 trimmable current biases on-chip that require a 7-bit configuration. Finally, there are 3 additional bits in the full-chip configuration that encode the number of neurons sampled on each sampling cycle and the frequency of neuron sampling for data streamout from the chip. The first digital function required in the processor is the implementation of a system that allows a user to configure the processor to solve a particular trained problem by loading all these configuration bits into the appropriate memory elements in the fabric so that the processor can use these to perform the ground-state search on the defined problem.

- Neuron state sampling: The 256 neurons in the PASSOv1 processor are capable of stochastically updating their state asynchronously and fully in parallel using only local interactions with neighboring node states. However, in order to transfer data off of the chip to a test board for the purposes of data collection, processing and analysis, the asynchronously generated neuron states need to be sampled on a sampling clock. The neuron state sampling system is responsible for collecting samples of a user defined subset of neurons in the chip at runtime in such a way that the samples can identifiably be streamed out of the chip. The ideal sampling frequency of the neurons is determined by the autocorrelation time of the node circuit, which is a timescale at which sequential samples obtained from a single neuron state still have a probabilistic correlation with each other. The nominal autocorrelation time for the node circuit described in section 2 has been determined to be roughly 300MHz. The ideal sampling scheme would be able to sample all neuron states continuously at a frequency of 300MHz and stream these from the PASSOv1 processor to a test board. However, because of I/O limitations, such a sampling scheme is unrealistic to implement. Instead, the sampling scheme implemented in this work consists of a configurable neuron state sampling scheme that can

sample a variable number of neurons at different frequencies, ranging from 16 neurons at a frequency of 300MHz to all 256 neurons at a frequency of 18.75MHz (with a fixed throughput of 4.8Gsamples/second).

- Neuron sample streamout: Finally, the digital systems on the chip need to transfer the collected samples off chip to the test board and external CPU for processing and analysis. Since the data transfer is limited by I/O pinout and speeds, the sampling frequency requirements are met by sampling the neuron states to get combinatorial optimization problem solution states in burst mode, writing the data to an SRAM buffer on a fast sampling clock (at a fixed throughput of 4.8Gsamples/second), and reading the data out of the SRAM buffer serially on a slower I/O clock that can meet pinout and I/O speed requirements.

To achieve this functionality under the given specifications, separate configuration, sampling and streamout systems are designed and implemented in RTL. The sample and streamout circuits operate together whereas the configuration circuitry operates completely independently of the sample and streamout systems. The following subsections describe the design choices and implementations of each of the three core digital systems that implement this functionality.

3.2 Configuration

The configuration system is responsible for programming a particular problem onto the PASSOv1 processor. Simulated training for a physical optimization problem generates a set of neuron interaction weights and biases that maps the problem onto the ground-state energy search of an Ising model system. These generated weights and biases then need to be written into the processor for the neuron interactions to occur in a manner that appropriately solve the problem.

The chip configuration system is implemented using a slave-serial bitstream input protocol that is commonly used as a device programming methodology on FPGAs and other reconfigurable fabrics. In this mode, the configuration is held in a shift chain of configuration latches which is supplied by an off-chip bitstream input and control signals that propagate the entire config bitstream through all the configuration latches in the chip one bit at a time per clock cycle. While a JTAG implementation was initially considered for the configuration system, it was removed from the design of the final chip since it would add additional complexity to the tapeout and there were no plans to use the additional features of the JTAG protocol during either chip testing or operation.

The configuration shift register chain consists of a total of 19171 bits. The breakdown of these configuration bits is as follows:

- Neuron configuration bits: 18944
- Current bias configuration bits: 224
- Sampling configuration bits: 3

The 74 configuration bits assigned to each neuron are encoded in the following order:

- config[73:10]: interaction weights (8x8 matrix)
- config[9:2]: neuron bias (8-bit number)
- config[1]: clamp bit to force output to logical 1
- config[0]: clamp bit to force output to logical 0

The processor configuration chain is formed by sequentially connecting configuration latches for neurons and current biases together with the intermediate nodes providing configuration values to the synapse and biasing circuits (creating a serial-input parallel-output shift register chain). The configuration shift register uses an off-chip active low shift enable signal and a configuration clock of 1MHz. In addition, the configuration system contains an echo of the shift chain output back to the test board for validation purposes. To ensure timing closure and skew-induced errors along the off-chip and on-chip configuration signal paths, the configuration shift register uses opposite edge capture as is standard with the slave-serial programming protocol. The microcontroller updates the configuration bitstream input on the negative edge of the configuration clock and the bitstream is captured and propagated on-chip on the positive edge of the configuration clock.

3.3 Neuron state sampling

The 256 neurons in the PASSOv1 processor are capable of stochastically updating their state asynchronously and fully in parallel using only local interactions with neighboring node states. However, in order to transfer data off of the chip to a test board for the purposes of data collection, processing and analysis, the asynchronously generated neuron states need to be sampled on a sampling clock. Ideally, the sampling system will be able to collect samples from each of the neurons at the autocorrelation frequency of the neuron (the timescale at which consecutive samples collected from the neuron have a probabilistic correlation). However, because of spatial and I/O bandwidth limitations, the PASSOv1 processor instead implements a configurable neuron state sampling scheme that outputs samples at a fixed rate of 4.8Gsamples/second.

The configurable sampling scheme uses a set of 3 config bits passed in through the configuration bitstream by the user to denote a specific subset of rows of the 256-neuron array that are then selected to be sampled by the sampling circuitry. Specifically, the sampling configuration bus causes the following rows to be sampled at the respective clock frequency based on their value.

- 3'b000: sample only row 1 at 300MHz
- 3'b001: sample rows 1-2 at 150MHz
- 3'b010: sample rows 1-4 at 75MHz

- 3'b011: sample rows 1-8 at 37.5MHz
- 3'b100: sample all rows (1-16) at 18.75MHz

Each of these configurations produces a fixed throughput of 16 samples at a frequency of 300MHz. To implement this behavior for sampling the neuron states, each column of neurons operates independently to generate one sample at a frequency of 300MHz that is one of the bits in a 16-bit sample out bus generated at the end of the sampling circuitry. As a result, we can describe the entirety of the sampling circuit by just looking at one column and how the samples along that column are generated. The clock used to sample the neuron states is a fixed 300MHz clock called "tclk".

The sampling circuitry of each column uses a modified scan flop architecture with a chain of length 16 featuring serial scan output and a parallel load option (16-bit PISO scan chain). It has a mux at the input of each register to determine whether the register chain should perform the parallel load or continue to shift the current samples on the chain. In order to sample the entire subset of selected neurons (refer to the config sampling map) on the same tclk positive edge, the parallel load enable signal is asserted with the periodicity of $\frac{300}{n}$MHz where n is the number of sampled rows. In other words, the signal that loads all selected neuron states into the scan chain of each column is raised high for 1 tclk cycle per n tclk cycles by the 3 sampling configuration bits in the chip configuration bitstream.

The samples are then shifted along the scan chain on the n intermediate tclk edges between consecutive parallel load signal assertions. In order to increase the margin of tolerance for race conditions between the load signal and the propagation of samples through the scan chain, the samples are shifted into the scan chain on the negative edge of tclk. Although this cuts the setup margin in half, the margin of 1.5ns is still more than enough to achieve setup timing closure, and this scenario prevents the possibility of a hold time violation causing a loaded sample to race through the scan chain of a neuron column. At the end of the scan chain, the samples are recaptured on the positive edge of tclk to be used in the streamout circuitry.

This "sampling column circuit" is reproduced across each column in the chip, from columns 1 through 16. Once the sampling process is started, the 16-bit sample bus formed by the serial scan chain outputs of each of the sample columns will be populated with the samples from each row one at a time, starting with row 1, then moving up to row n and finally resetting back to row 1, restarting the sampling sequence. Each set of samples produced from rows 1 through n will be captured on a sampling register at the exact same tclk positive edge and as a result, a full set of samples from rows 1 through n will be temporally correlated with each other.

The generation of parallel sample load signal is implemented with a series of decentralized counters with one counter present at each neuron. Therefore, we want to make sure that each of these counters is in phase and receives the reset signal at the same time. In order to effectively achieve this, the sampling system builds a reset tree that pipelines the reset to each row with a delay of 1 tclk

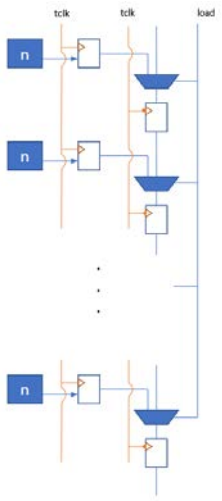


Figure 9: Modified parallel-in/serial-out scan chain architecture with periodic load operation based on sampling configuration used to implement the sampling column circuit. The sampling system is created by reproducing this chain across 16 neuron columns and a fingerprint generator.

cycle. In order to offset the determinate, synchronous delay in reset reception by each row, the load signal generator for each row is initialized to a different value (15 for row 1, down to 1 for row 15) so that they each count down to 0 on the same tclk cycle after initialization and stay in phase for subsequent sampling cycles. The topmost row (row 16) does not have a counter and a scan flop since there is no shift input to load into this register. In this architecture, If the sampling system is set to sample a smaller subset of rows than all 16, the samples present on any scan chain nodes further above the topmost sampled will be flushed out since the parallel load operation will be triggered before these samples can propagate down towards the output net of the sample column.

In order to deterministically map the 16-bit sample to a specific row and account for off-by-one misalignments incurrent during startup, an identifying signal is generated in parallel with the samples that encodes which row is currently being sampled at the output of the sample tile. This signal is called the "fingerprint" and it uniquely identifies to which row a set of 16 samples belongs.

In this setup, the fingerprint and the sample signals will always be parallel with each other, forming a parallel 17-bit bus. The fingerprint signal follows an identical datapath to each sample column but it's inputs are always hard-coded in the fixed pattern of "01 000 111 0000 1111" starting from row 1 up to row 16. For each subset of sampling (1 row, 2 rows, 4 rows, 8 rows, 16 rows), only the first n digits of the fingerprint signal will repeat every sampling cycle just as the first n rows are sampled from depending on the 3 sampling configuration bits. The key feature of the fingerprint signal is that it forms a unique pattern for any

power of 2. Hence, each k-th bit of the 16-bit fingerprint signal is associated with the k-th row. To identify which row a set of 16 samples is from, the position of the fingerprint signal bit (out of the 16 bits in the fingerprint pattern) in parallel with the 16 samples needs to be determined. If the current value of the fingerprint bit is found to be in the k-th position (out of 16) of the fingerprint pattern, the corresponding set of 16 samples will be from the k-th row of the chip.

3.4 Neuron sample streamout

On each tclk positive edge (300MHz frequency), the sampling system generates a set of 16 samples and a fingerprint signal corresponding to those 16 samples. The streamout system consists of an SRAM buffer that holds samples from the sampling system and an arbitrating FSM that asserts the control signals for the SRAM and performs clock domain crossings between the faster write clock (tclk, 300MHz) and the slower IO clock (ioclk, 20MHz). In addition, the streamout system also implements a 16-bit SPI transmitter to serialize and output the collected samples as a serial bitstream (since the number of IO bumps disallows 16-bit parallel output). Finally, the streamout tile also implements a ready-valid handshake between the test board and the PASSOv1 chip that begins a cycle of neuron state sampling and streamout in burst-mode operation, generating a total of 128kBits (corresponding to 27μ s asynchronous evaluation time) of neuron state data to analyze per sample and streamout cycle.

Using the SRAM buffer, the processor can operate in a burst-mode format where the external control signals (chip-level inputs) can begin a sampling and streamout cycle. This cycle collects samples at a 4.8Gsample/second throughput until the SRAM is completely filled with samples, and then sequentially reads SRAM addresses at a much slower clock that can easily be supported by the used IO libraries and IP. Since the 16-bit sample bus is serialized before transmission out of the chip at a rate of 20MHz, the samples need to be read out of the SRAM at a frequency of $\frac{20}{16} = 1.25$ MHz. This 1.25MHz clock used to read from the SRAM is generated by dividing ioclk (20MHz) using a clock divider and is named ioclk_d16. The streamout algorithm is as follows:

Algorithm 1: Burst-Mode Streamout Cycle

```
Initialize SRAM write and read addresses to 0;
Wait for receipt of early receiver-ready signal from μC on test board;
while SRAM write address < 8192 do
    On tclk (300MHz) positive edge, write 16-bit sample bus generated
     by sample tile and 1-bit fingerprint to SRAM;
    Increment SRAM write address by 1;
end
Assert streamout data transmitter-valid signal;
while SRAM read address < 8192 do
    On ioclk_d16 (1.25MHz) positive edge, read 17-bit
     sample+fingerprint signal from SRAM;
    Serialize and output parallel signal extracted from SRAM using
     on-chip SPI transmitter;
    Hold streamout data transmitter-valid signal high;
    Increment SRAM read address by 1;
end
De-assert streamout data transmitter-valid signal;
Reset counters and addresses to prepare for next streamout cycle.
```

Looking at the streamout algorithm, it is apparent that the execution of a single burst-mode sample and streamout cycle involves a ready-valid handshake and clock domain crossings between the sample write clock (tclk) and the sample read clock (ioclk_d16). In order to realize this in practice, an arbiter FSM is designed and implemented that is responsible for making sure that the ready-valid handshake is executed properly between the off-chip test controller and the PASSOv1 processor. In addition, the arbiter FSM also handles all the required clock domain crossings between tclk and ioclk_d16 and ensures that at all times, the appropriate signals are applied to the SRAM to make sure that the correct operation necessitated by the streamout algorithm is performed. The SRAM used in the chip is a true dual-port SRAM configured as a simple dual-port SRAM generated using the GF12LP ARM SRAM memor compiler where port A is tied to tclk and exclusively used for writing to the SRAM and port B is tied to ioclk_d16 and exclusively used for reading samples from the SRAM.

The streamout control FSM consists of two coupled FSMs, one of which updates state on tclk and handles the SRAM write-related signals and the other which updates on ioclk_d16 and handles the SRAM read-related signals. All clock domain crossings are handled by using synchronized state pointers of the coupled FSMs to make decisions regarding FSM state updates and assigning the SRAM signals based on the FSM states. Both FSMs have a 2-bit state with the state mapping with 4 states: IDLE (00), START (01), RUN (11) and STOP (10). The FSM transition sequence is described as follows:

Algorithm 2: Streamout Control FSM State Transition

1 Initialize read and write FSM states to IDLE. Reset read and write addresses to 13'b0;
2 On receiving the rx-ready signal, transition write FSM to START state;
3 Transition write FSM to RUN state;
4 During write RUN state: assert port A SRAM write enable, +1 write address per tclk cycle until write address = 8192;
5 Transition write FSM to STOP state. De-assert SRAM write enable;
6 Transition read FSM to START state;
7 Transition read FSM to RUN state;
8 During read RUN state: assert port B SRAM read enable, +1 read address per ioclk_d16 cycle until read address = 8192;
9 Transition read FSM to STOP state. De-assert SRAM read enable;
10 Reset write FSM to IDLE state;
11 Reset read FSM to IDLE state;

All state pointers are evaluated as a 1-bit boolean signal and synchronized before being referred to across clock domains. Similarly, the ready valid signals are synchronized to tclk and ioclk_d16 as inputs and synchronized to ioclk while being output from the chip.

From the output of the SRAM, a 16-bit SPI transmitter serializes the 16-bit sample bus read from the SRAM on ioclk_d16 to a serial bus on ioclk and drives the bus out of the chip. The fingerprint signal in common with each of these 16 samples is held at its constant value over the period of the serialization and output from the chip as a top-level pinout. In this manner, the streamout system is designed to transfer neuron state samples in bursts of 128kBit sets at a constant sampling frequency of 4.8Gsamples/second.

4 Physical Implementation

4.1 Overview

The PASSOv1 processor consists of a binary spin network of 256 neurons. On the top level, the main output generated by the PASSOv1 chip is a distribution of sampled values of the states of a subset of the neurons at period points in time. In application, the top level inputs to the processor are designed to be programmed in by an FPGA that interfaces with a host computer. The FPGA to chip interface is established through a simple PCB and the inputs to the chip are applied directly. After the application of the proper input signals, the host computer will receive a probability distribution of states that can be analyzed to obtain a solution to the optimization problem programmed into the chip.

The chip is implemented in the GlobalFoundries GF12LP process with a die area of $4mm^2$ (2mm x 2mm). The chip interfaces with the test board socket using C4 (flip chip) SNAG180 IO bumps. Under the manufacturing and

Figure 10: Schematic representation of the streamout system containing the control FSM, SRAM buffer and sample output serializer.

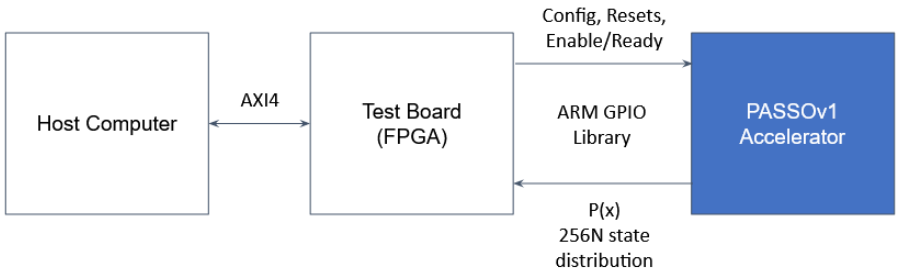


Figure 11: Planned evaluation setup of PASSOv1 processor.

packaging constraints, 64 IO bumps arranged in a 8x8 with a pitch of 230μm were able to be instantiated over the chip area.

In order to use the PASSOv1 processor, the chip is first reset by applying the analog, configuration and digital system reset signals from the controller. Next, the streamout receiver ready signal is raised, which starts the sampling and streamout algorithm described in the previous section. When the streamout transmitter data-valid signal is detected to go high at the test board, the test board begins writing the stream of samples coming from the chip to a DRAM memory on the FPGA. After the cycle is finished, the collected sample data is transmitted from the FPGA's on-board memory to the host computer through an AXI4 host interface. After this data transfer, the collected probability density distribution can be analyzed and used to infer solutions to the combinatorial optimization problem programmed into the chip.

4.2 Physical Design Considerations

The chip is implemented on a 2mm x 2mm die ($4mm^2$ area). Figure 13 depicts the floorplan of the PASSOv1 chip as implemented. The core of the chip consists of the 256 neuron grid where each neuron has an area of approximately 50μm $\times 35\mu$m. The configuration latches are interspersed among the 256 neuron core as are the sampling column circuits which sample the neuron states. The SRAM buffer and streamout logic are located at towards the bottom section of the chip. The test cluster containing 4 neurons and 2 current biases is located towards the bottom-left of the chip. The C4 IO bumps are arranged in a 8x8 grid over the chip area as showed in figure 12.

The chip was integrated in a mixed top-down and bottom-up approach. First, the synapse circuit was hardened as a macro using digital logic synthesis using Cadence Genus and automatic place-and-route using Cadence Innovus. The hardened synapse macro was integrated with the full-custom neuron circuit to develop the node or full neuron that formed the building block of the systolic array presented in this work. The 256 neuron core along with all of the digital and test systems and the IO drivers and bumps were integrated full flat in a digital-on-top flow which resulted in the final chip.

In order to preserve neuron signal integrity, rigid constraints of delay, transition, and routing capacitance were included to constrain inter-neuron routing and asynchronous behavior. The neuron state was synchronized to a sampling clock with a series of 3 synchronizing registers and the transition at the asynchronous node was constrained to be at least 2 orders of magnitude faster than the highest expected switching frequency of a neuron. This arrangement deterministically prevents metastable neuron states from propagating through the sampling circuitry and also highly reduces the changes of a state being corrupted due to synchronous sampling of an asynchronous signal.

4.3 Chip Parameters

After integrating the chip in a digital-on-top flow with Cadence Innovus standard cell driven place-and-route, the chip was DRC cleaned and optimized for congestion, routability, and power consumption. In addition, the chip was make compliant with all level 1 pattern matching, metrology, MAS and MOB related GlobalFoundries 12LP rules, along with making sure that the design was passing signoff level timing closure and power checks. Table 1 shows a brief summary of the most significant chip metrics from the final implementation of the chip.

5 Conclusion and Future Work

The PASSOv1 processor implements a stochastic mixed-signal neuron capable of updating its state asynchronously and massively in parallel constrained only by local neighbor-to-neighbor interactions with complete independence from

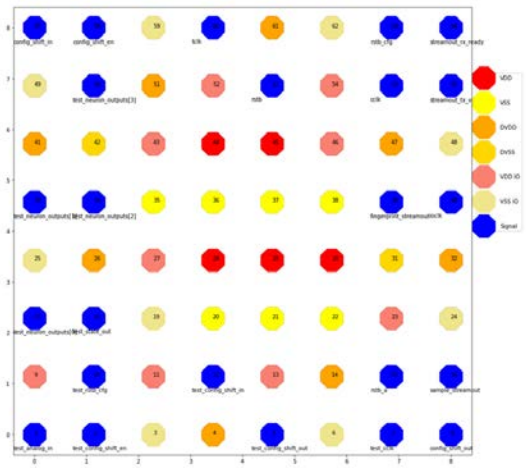


Figure 12: PASSOv1 processor C4 bump assignment. More than half of the available bumps are conserved for power/ground supply.

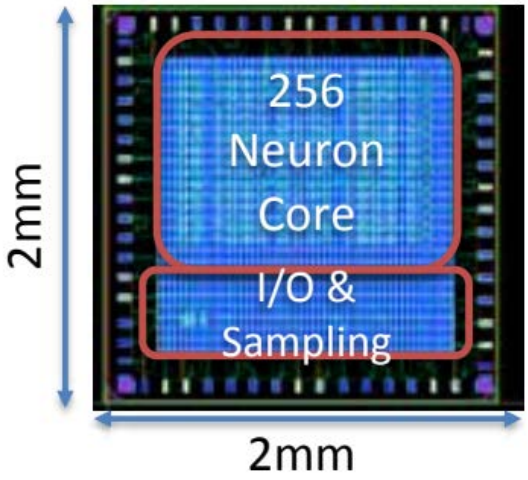


Figure 13: Die shot of final PASSOv1 processor design submitted to tapeout in April 2021.

PASSOv1 Chip Parameters	
Chip Parameter	Value
Number of Nodes	256
Chip Area	2mm x 2mm = $4mm^2$
Core Area	0.8mm x 1.3mm = $1.04mm^2$
Power (est)	22 mW
Nominal Auto-correlation Frequency	300 MHz
Intrinsic Frequency	170 MHz
Configuration Time (est)	25 ms
Synapse Speed	2 GHz
Sample Cycle Duration (est)	$\approx$1 ms
Connectivity	King's Move

Table 1: Key PASSOv1 Chip Metrics After Physical Implementation

any global chip-level synchronous signals, memories or timing paths. This distributed processing fabric has been demonstrated to hold a 250x power and 3x performance speedup over state of the art systems in the 100 node max-cut problem. The work presented in this report begins by outlining the requirements and specifications of the digital support circuitry that is required to sample the neurons in parallel on a sampling clock and stream the generated samples to an external test board connected to a host interface where the probability distribution of neuron states can be processed and analyzed to solve the optimization problem configured into the PASSOv1 processor.

At the end of this project in April 2021, the final design of the PASSOv1 processor was made compliant with all GF12LP tapeout rules and the design was submitted to GlobalFoundries as part of a multi-project wafer. The full process of testing, validating and benchmarking the PASSOv1 processor will continue on and occur after the termination of this MS degree. Specifically, the PASSOv1 wafers will be fabricated in Fall 2021 and the resulting dies belonging to the team will be packaged using a third party vendor. In parallel, a test board will be designed to hold a packaged PASSOv1 processor and interface with an FPGA and host microprocessor that will be able to run algorithms on the fabricated PASSOv1 processor and analyze the retrieved data. This process will continue on into the better part of the year and possibly into the year 2022 as well. Furthermore, the team within the Salahuddin group has plans for next generation chips based on the PASSO architecture that will aim to achieve more complex systems with additional performance boosts over the PASSOv1 chip. This project and the work contained therein demonstrates a promising starting point for a project and architecture that demonstrates promise to significantly accelerate the processing of NP-hard combinatorial optimization problems.

References

[1] M. Hayashi, M. Yamaoka, C. Yoshimura, T. Okuyama, H. Aoki and H. Mizuno, "An Accelerator Chip for Ground-State Searches of the Ising Model with Asynchronous Random Pulse Distribution," 2015 Third International Symposium on Computing and Networking (CANDAR), 2015, pp. 542-546, doi: 10.1109/CANDAR.2015.64.

[2] M. Aramon, G. Rosenberg, E. Valiante, T. Miyazawa, H. Tamura and H. Katzgraber, "Physics-Inspired Optimization for Quadratic Unconstrained Problems Using a Digital Annealer," Frontiers in Physics (Vol. 7), 2019, pp. 48, doi: 10.3389/fphy.2019.00048

[3] T. Wang and J. Roychowdhury, "Oscillator-based Ising Machine," arXiv preprint, arXiv:1709.08102

[4] I. Ahmed, P. -W. Chiu and C. H. Kim, "A Probabilistic Self-Annealing Compute Fabric Based on 560 Hexagonally Coupled Ring Oscillators for Solving Combinatorial Optimization Problems," 2020 IEEE Symposium on VLSI Circuits, 2020, pp. 1-2, doi: 10.1109/VLSICircuits18222.2020.9162869.

[5] T. Inagaki, Y. Haribara, K. Igarashi, T. Sonobe, S. Tamate, T. Honjo, A. Marandi, P. McMahon, T. Umeki, K. Enbutsu, O. Tadanaga, H. Takenouchi, K. Aihara, K. Kawarabayashi, K. Inoue, S. Utsunomiya and H. Takesue, "A coherent Ising machine for 2000-node optimization problems," Science 04 Nov 2016, Vol. 354, Issue 6312, pp. 603-606, doi: 10.1126/science.aah4243

[6] M. Nielson, Xilinx 7 series FPGA Application Note, https://www.xilinx.com/support/documentation/application_notes/xapp583-fpga-configuration.pdf