rmNoto Sans

Comprehensive Report: The P-Bit Memristor Array System

Analysis of TSU Architecture Specification

November 5, 2025

Introduction

This report summarizes the structure, function, and development strategy of the **PBit Memristor Array System**, based on the provided hardware/software specification. This system represents a cutting-edge integration of physical neuromorphic hardware and a specialized software framework for energy-based, stochastic computing.

1 The Core Model and Physics: The P-Bit

The central element is the **PBitMemristorArray**, an energy-based machine learning model designed for **stochastic computing** (probabilistic sampling). The network consists of 256 p-bits (`p_0, ..., p_255`), where each p-bit's output is stochastic (0 or 1).

1.1 The Sigmoid Activation Function

The probability of a p-bit outputting 1 ($P(p_i = 1)$) is determined by the "tug-of-war" between the deterministic memristor current and the stochastic Gaussian noise. This physical competition yields the required **Sigmoid Activation Function**, which is derived from the complementary cumulative distribution function (Q-function) of the Gaussian distribution:

$$P(p_i = 1) = Q\left(\frac{V_{\text{TH}} - k \cdot V_{\text{READ}} \cdot G_M}{\sqrt{2} \cdot k \cdot \sigma_{\text{noise}}}\right) \tag{1}$$

The terms are defined as:

- G_M: The memristor's conductance, acting as the analog weight.
- $k \cdot V_{\text{READ}} \cdot G_M$: The deterministic **signal** current.
- $k \cdot \sigma_{\text{noise}}$: The stochastic **noise** (voltage fluctuation) from the TSU.

1.2 The Inverse Temperature (β)

The sensitivity, or gain, of the sigmoid curve is controlled by the **Inverse Temperature** (β), which is inversely proportional to the noise standard deviation (σ_{noise}):

$$\beta \propto \frac{1}{\sigma_{\text{noise}}} \tag{2}$$

This mechanism, tunable via the `noise_modulation` operation, allows the system to control the level of thermal fluctuation, which is essential for algorithms like **simulated annealing** in optimization tasks.

2 The Hardware Accelerator: The TSU

The **Thermodynamic Sampling Unit (TSU)** is a custom, specialized hardware component designed to perform the core stochastic sampling operation natively and efficiently.

2.1 TSU Architecture and Operation

- **Implementation:** The TSU is built upon the **1T1M memristor-based architecture** (`tsu_memristor_1`), where the memristor stores G_M and the sub-threshold transistor provides the physical thermal noise.
- **Native Sampling:** The key is the `tsu_pbit_native` operation, which performs the entire **threshold_noise** algorithm physically in one low-power step.
- **Efficiency:** The hardware targets high sample rates (10^9 samples/s) and uses an energy model based on kT (thermal energy), emphasizing low-power thermodynamic efficiency.
- **Kernel:** The `pbit_sigmoid_kernel` defines the internal firmware steps, including programming G_M, applying V_{READ}, injecting noise, and performing the final `threshold_compare` against V_{TH}.

3 The Software Framework and Validation Strategy

The model is managed by the specialized **THRML/JAX** software stack, which enables both control and rigorous validation of the physical hardware.

3.1 The THRML/JAX Software Stack

- **THRML (Thermodynamic ML):** The high-level library that defines the energy function (`pbit_threshold_energy`) and the sampling distribution (`sigmoid_gaussian`).
- **JAX:** The underlying numerical engine used for GPU execution (`gpu_a100` emulation), JIT compilation, and parallel processing.
- **Key Layers:** Includes the `pbit_sampling_layer` (using the TSU) and the `probability_layer` (used for calculating the theoretical probability on the GPU).

3.2 Multi-Target Development and Validation

The project uses a **Multi-Target** build strategy to ensure the TSU's physical output is mathematically accurate by validating it against a software reference.

3.2.1 The Three Build Targets

Table 1: Build Targets for the P-Bit System

Target ID	Hardware	Mode	Priority	Purpose
`tsu_production`	`tsu_memristor_1`	Production	High	Code for the actual 1T1M chip.
`gpu_emulation`	`gpu_a100`	Emulation	Medium	Digital emulation for testing large arrays.
`python_reference`	`cpu_x86`	Simulation	Fallback	**The Ground Truth:** Simple NumPy imple

3.2.2 The Validation Criterion

The core requirement is defined by the `BuildRule` which sets `validate_against = python_reference`. All other targets must satisfy the **probability_convergence** rule:

$$|\mathbf{P}_{\text{empirical}} - \mathbf{P}_{\text{theoretical}}| < 0.01 \tag{3}$$

This ensures that the physical, noise-driven output of the TSU hardware is tightly coupled to the underlying mathematical model, confirming its accuracy for scientific computing.