Ternary Content-Addressable Memory (TCAM) Array with a Priority Encoder

Ivan Huang(qh229), Jiaying Zhang(jz2354), Kevin Wang(kw633), Qingmiao Xiao(qx99), Xiaoyu Liang(xl434)

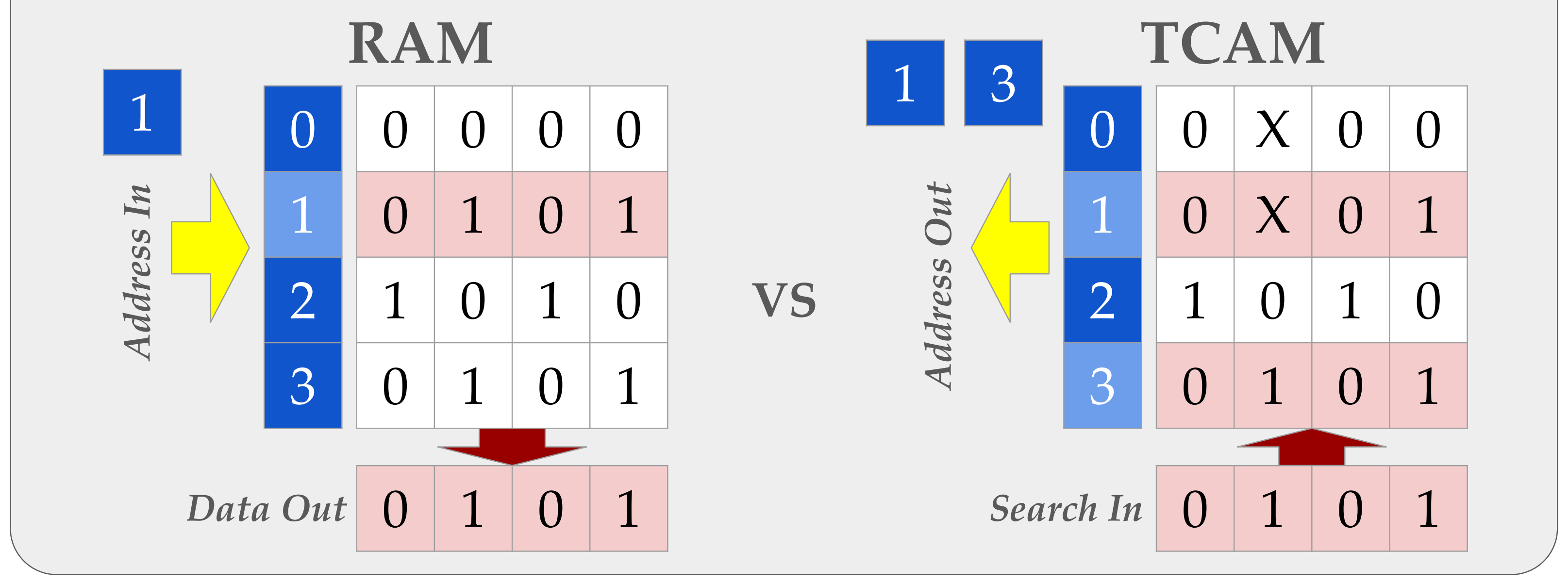
Abstract

This project aims to design, implement, and verify a **ternary content-addressable memory** (**TCAM**) with optimized speed and density. The system supports masked writes, ternary searches with "don't care" bits, and outputs the highest-index matching row. Functionality was verified through schematic-level simulation of individual cells, rows, and the integrated system.

Introduction & Motivation

Modern data processing and networking systems demand rapid associative lookups – eg. IP routing and access control filtering, where the address of the desired data is not known in advance. In conventional RAMs, data are fetched by supplying an address. Hence, performing an associative search requires iterating through all the addresses, making it unsuitable for such tasks.

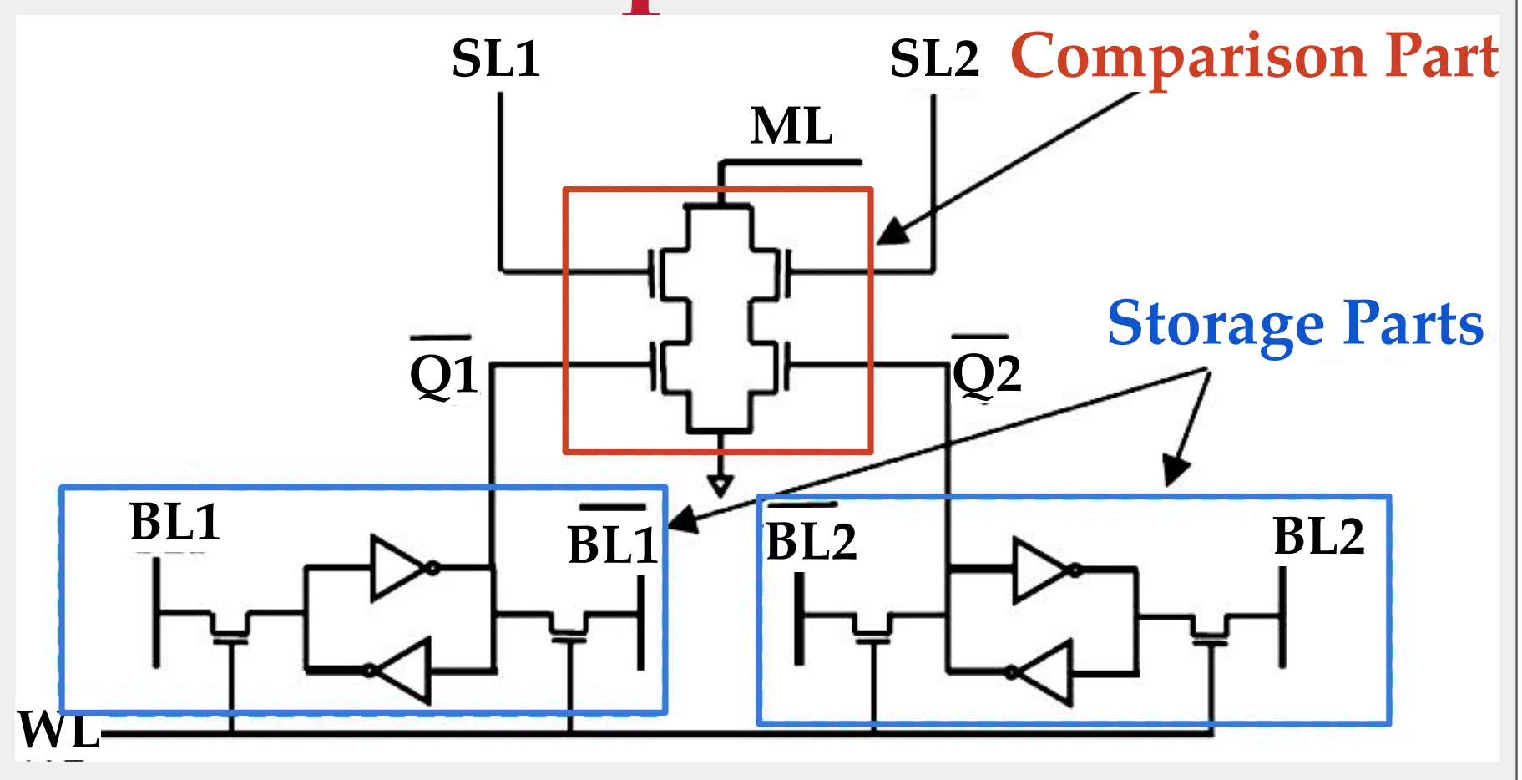
Content-Addressable Memories (CAMs) inverts this access model: the search data is compared to all stored words simultaneously, and any rows that match assert a hit line in the same clock cycle. Ternary CAMs (TCAMs) further extend this capability by allowing "don't care" bits**.** Our project implements a 32x32 TCAM array with an integrated priority encoder, with a focus on achieving high speed and compact layout.



Theory of Operation and Specifications

Each **TCAM cell** stores one bit in ternary form—0, 1, or X (don't care).

The storage blocks are two SRAM-like subcells, each made of cross-coupled inverters and pass transistors.



The comparison logic includes two NMOS transistors in series. The stored values (Q1, Q2) are compared with search lines SL1 and SL2.

Stored Data / Match Line / Select Line	0 (BL1=0, BL2=1)	1 (BL1=1, BL2=0)	X (BL1=1, BL2=1)
0(SL1=0, SL2=1)	1	0	1
1(SL1=1, SL2=0)	0	1	1
X(SL1=0, SL2=0)	1	1	1

Table.1 Truth Table of Match Line

The table above shows the truth table of matchline for a single TCAM cell. The matchline for a row outputs a high level only when all 32 TCAM cells in the row successfully match the input.

Specifications

- ***Search Delay***: $t_{PD} \sim t_{pre} + N \cdot t_{cell} + t_{encoder}$
- ***Search Power***: $P_{tot} \sim \alpha \cdot V_{dd}^2 \cdot f_{clk} \cdot (N \cdot C_{ml} + N \cdot C_{sl} + C_{encoder})$
- ***Area***: $A_{tot} \sim N \cdot N \cdot A_{cell} + A_{encoder}$

Top Level

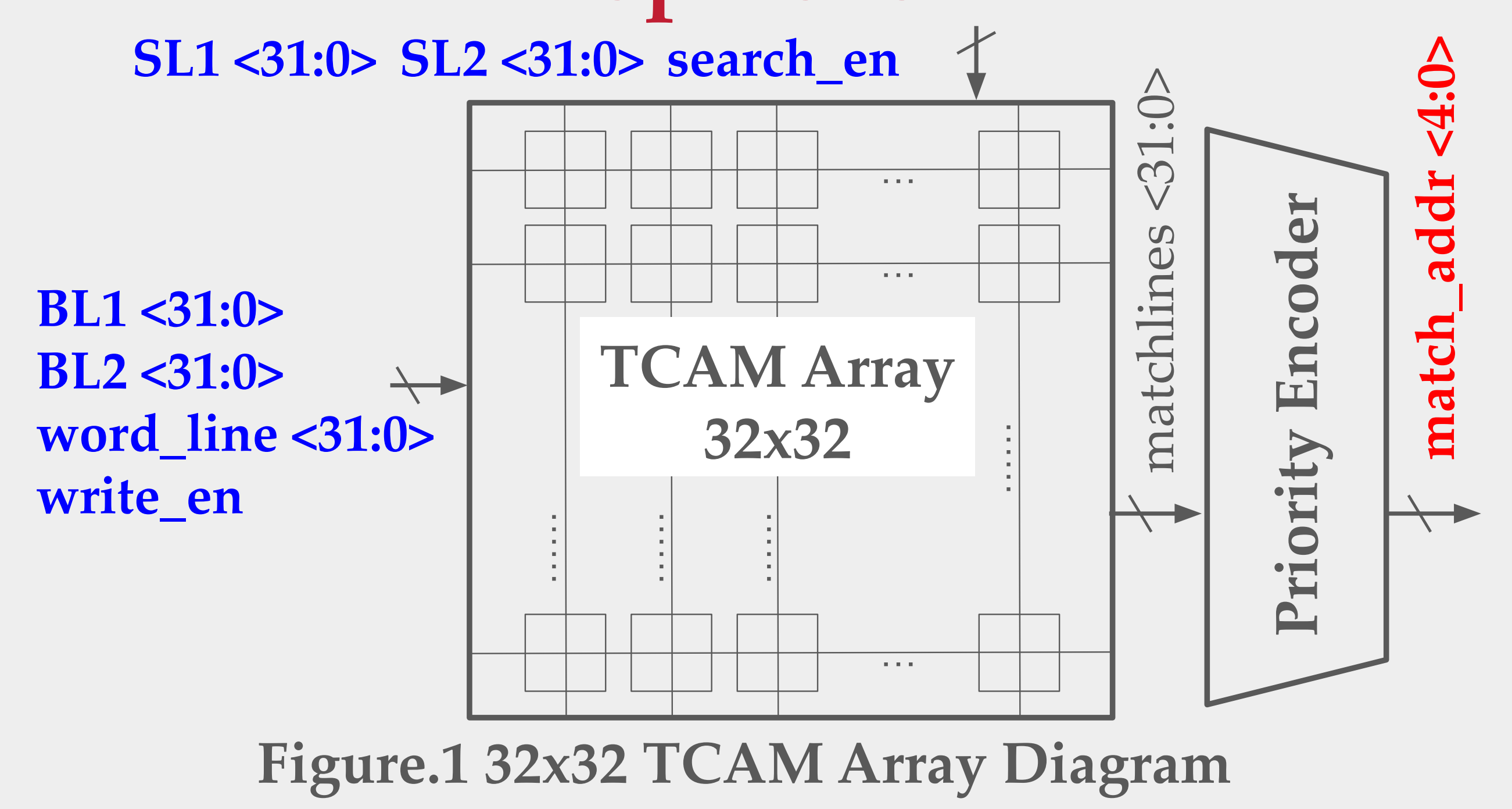


Figure.1 32x32 TCAM Array Diagram

Write operation: When *write_en* is asserted, *write_addr*[4:0] selects one of the 32 rows, and data is written using the *BL1*[31:0] and *BL2*[31:0].

Search operation: The circuit compares *SL1*[31:0] and *SL2*[31:0] (search inputs) with stored data. Matchlines are pre charged to high, and pulled low on mismatch.

Priority Encoder: The resulting 32-bit *matchlines*[31:0] feed directly into the Priority Encoder, which outputs *match_addr*[4:0] for the highest-indexed '1'

Key Schematics

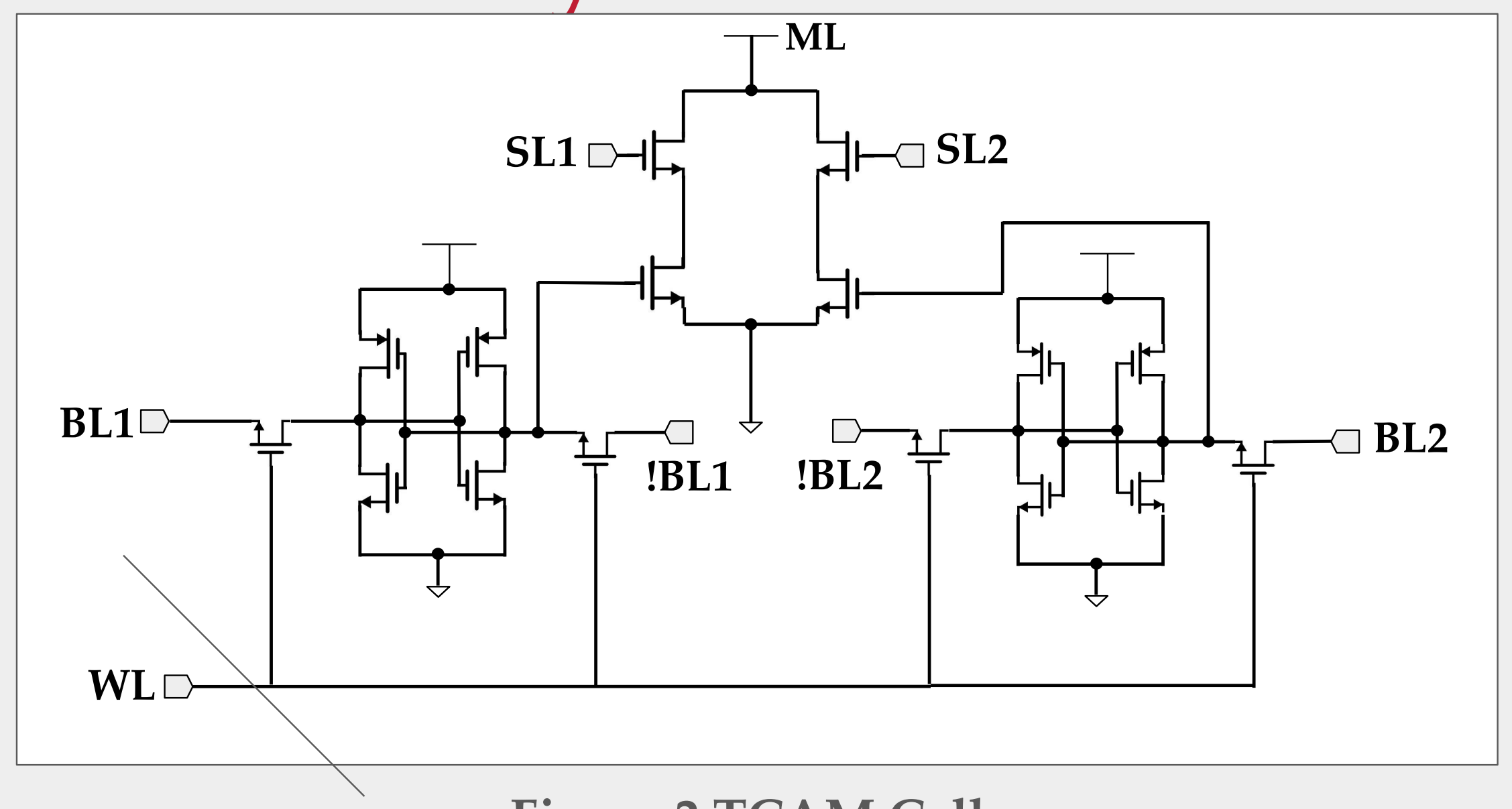


Figure.2 TCAM Cell

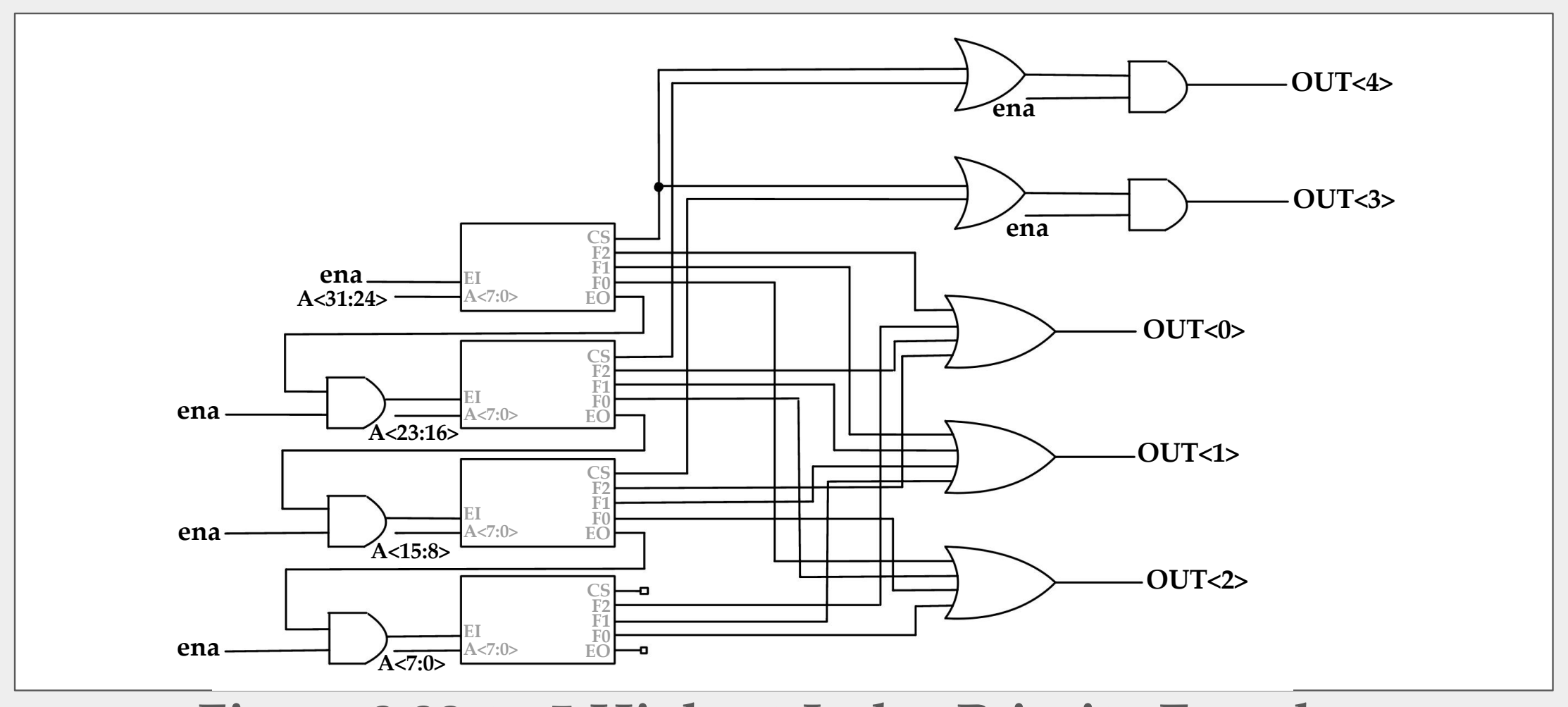


Figure.3 32-to-5 Highest-Index Priority Encoder

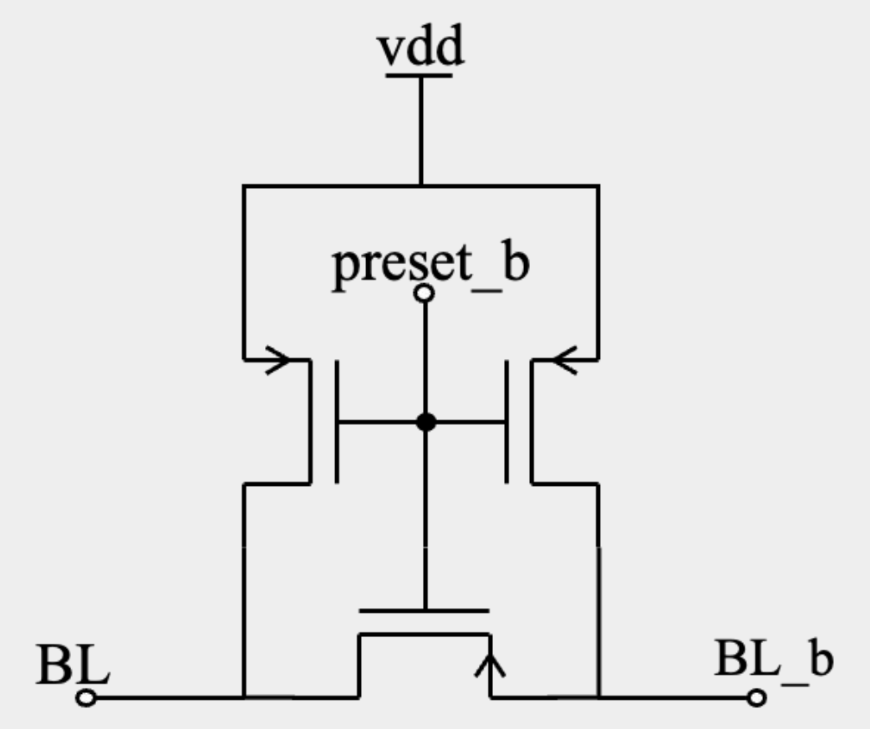


Figure.4 Bitline Precharge

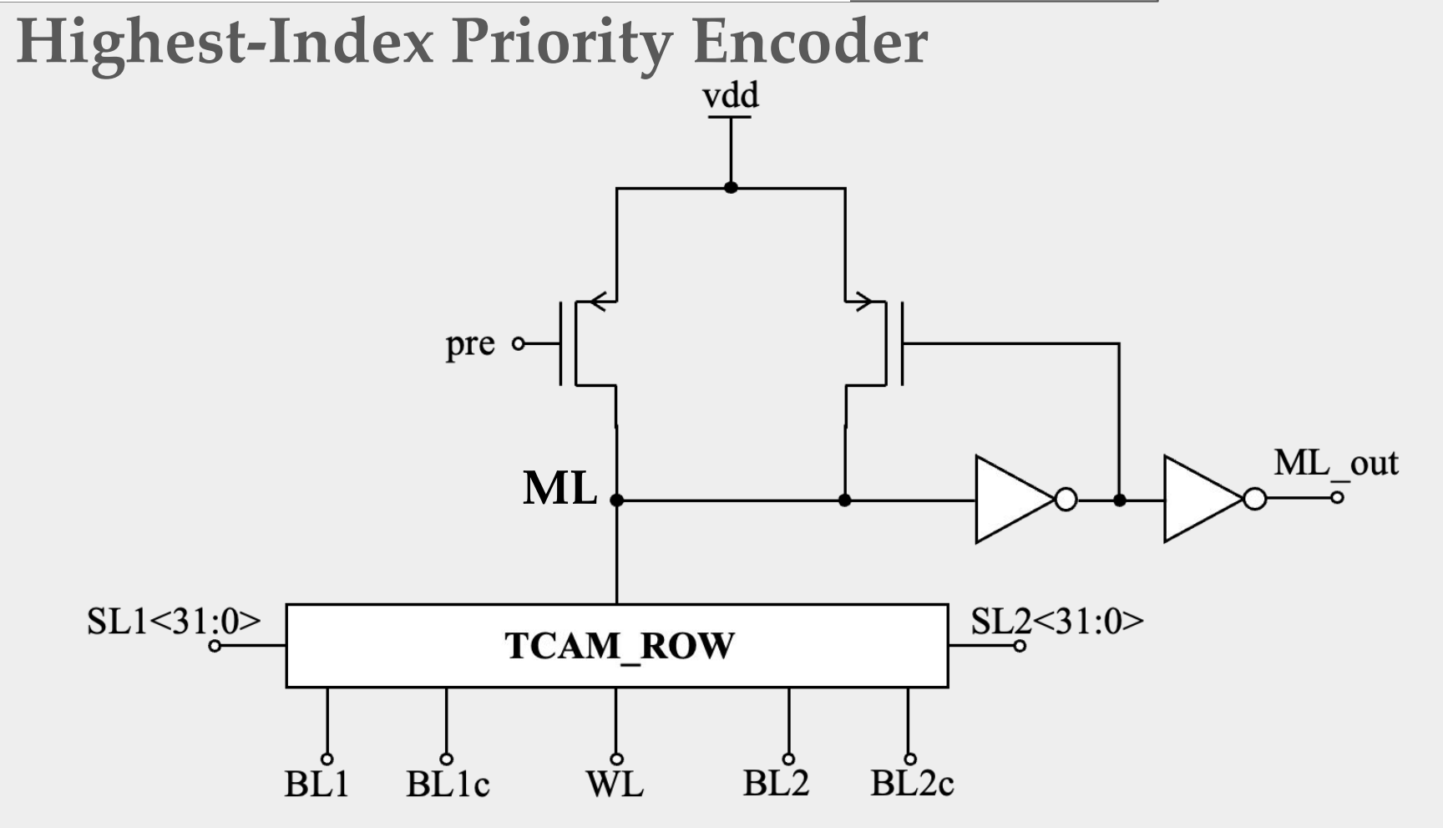


Figure.6 TCAM Row w/ Matchline Precharge

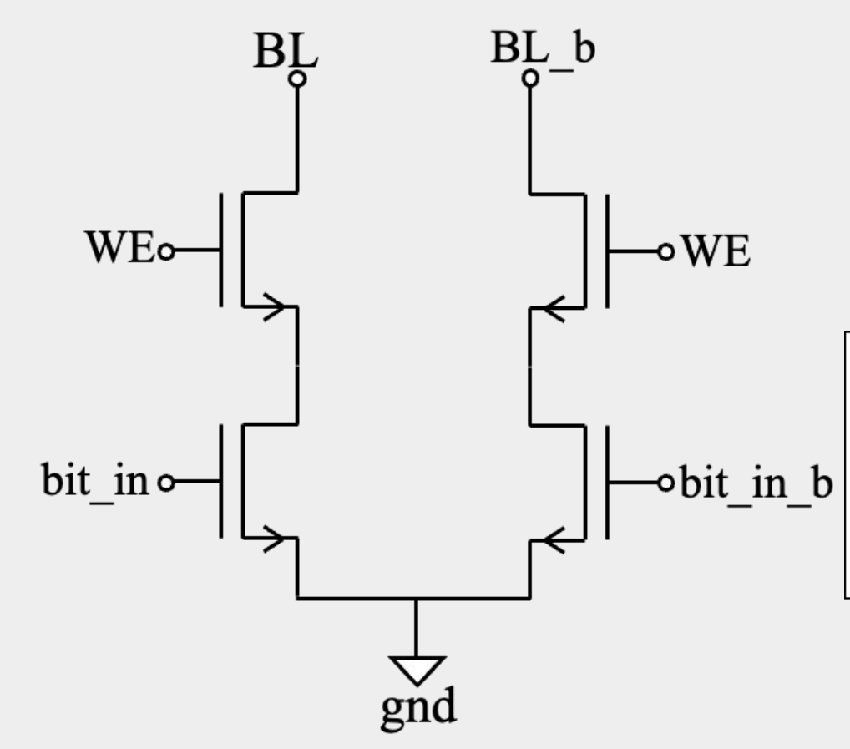


Figure.5 Write Cell

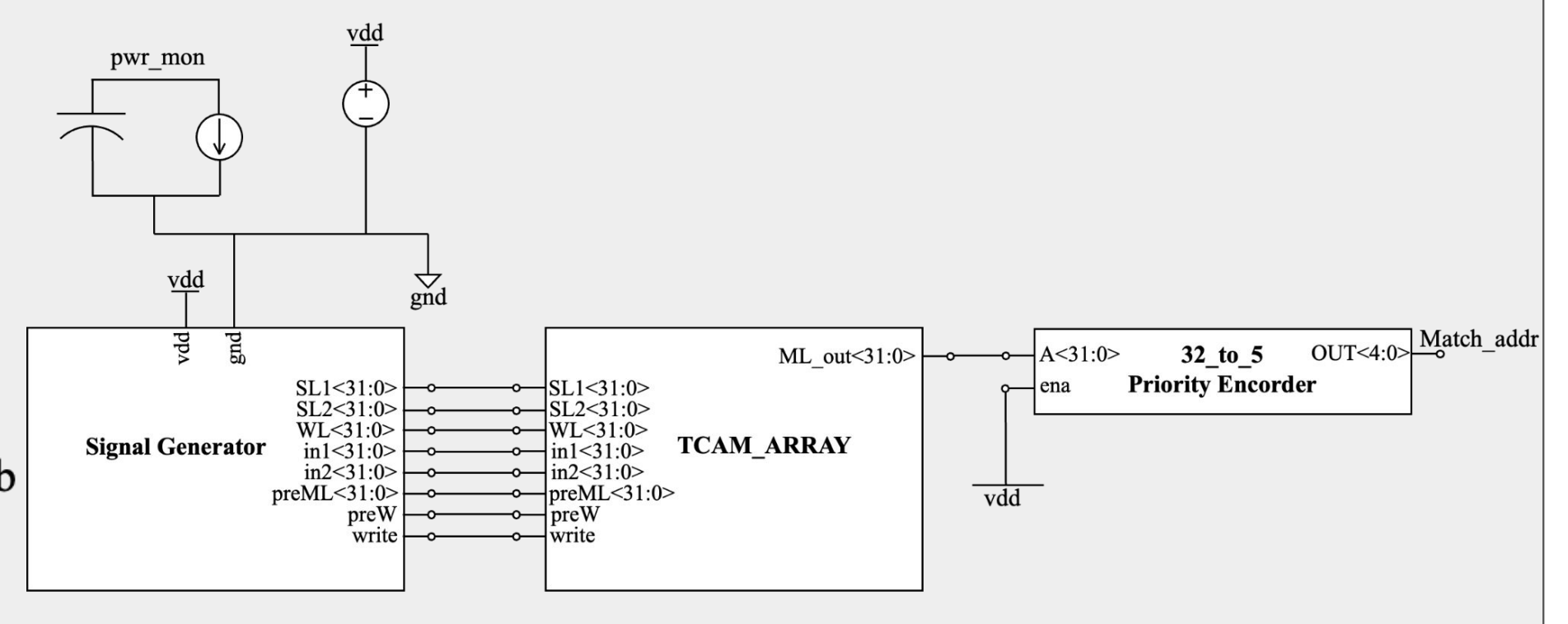


Figure.7 Full TCAM array + PE with test signal generator implemented in Verilog-A

Simulation Results

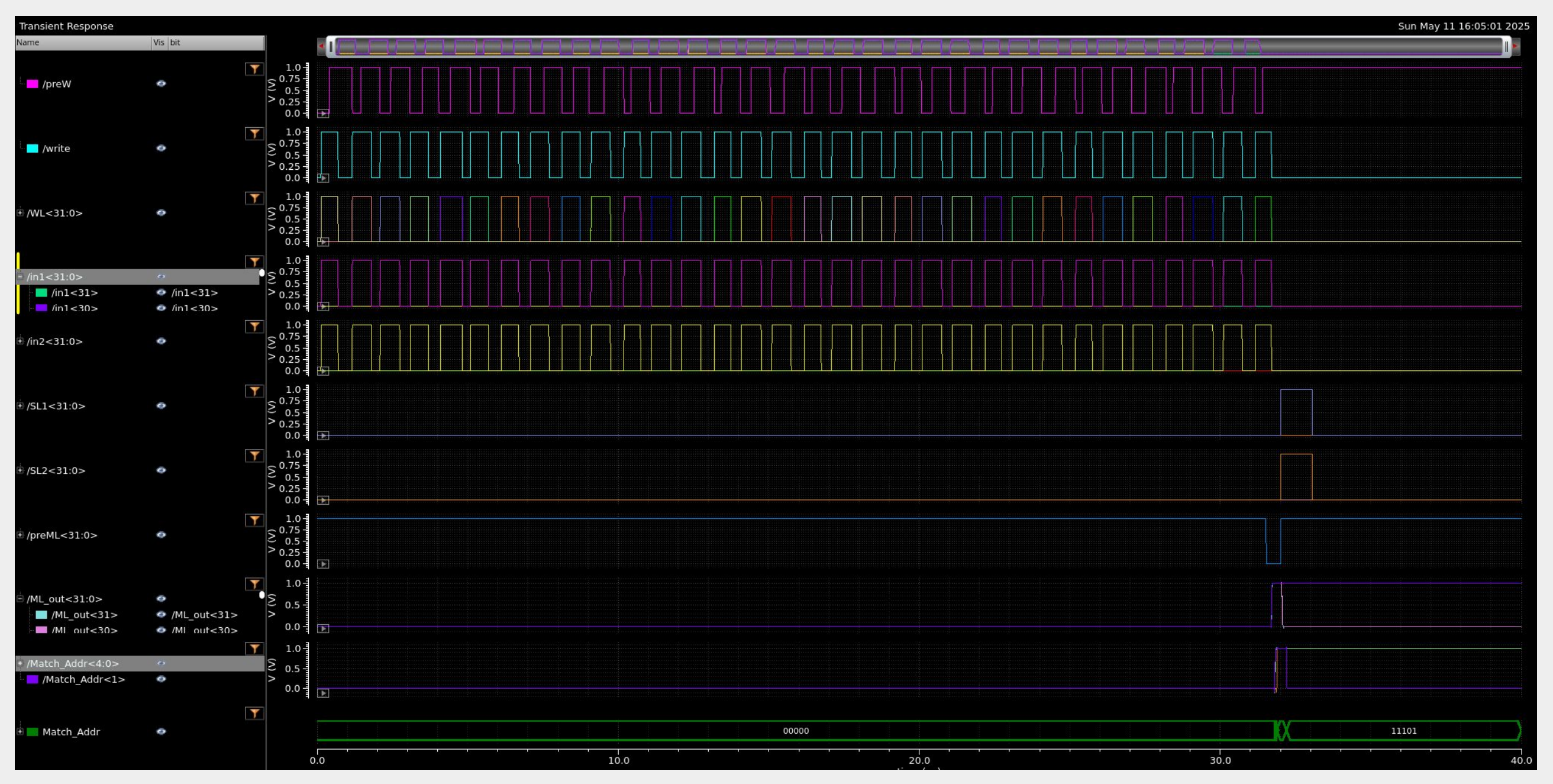


Figure.8 Array+PE Simulation Waveform

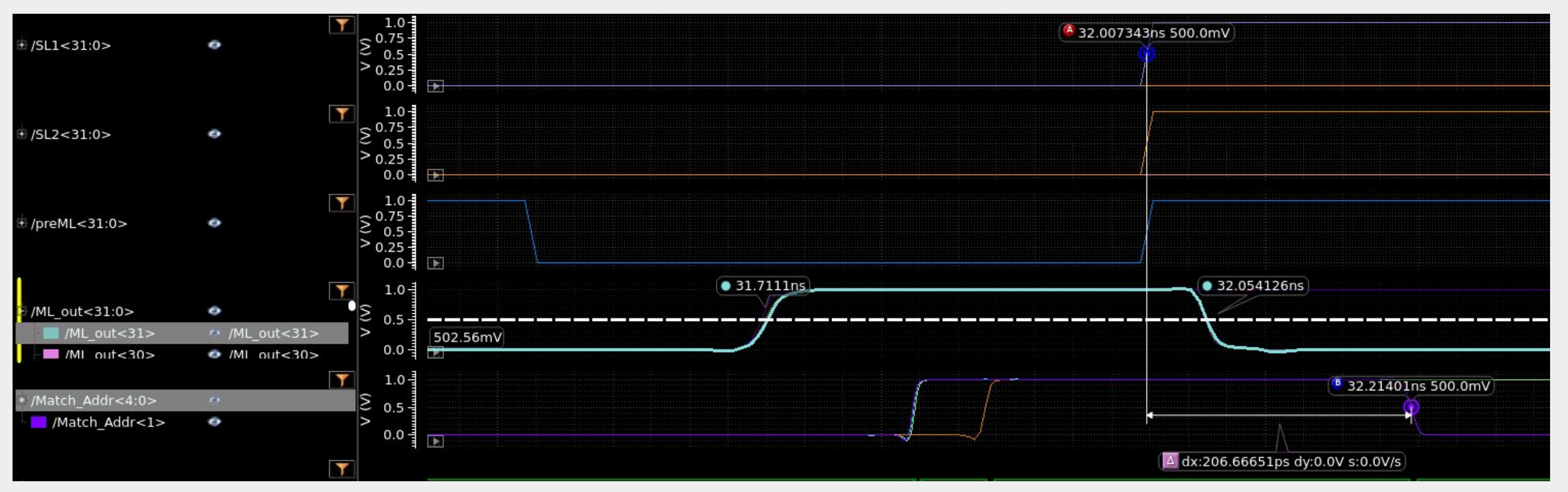


Figure.9 Searchline to matchline worst delay

To test the full array, Verilog-A is used to generate all the input data and control signals. We write data into all 32 rows of the TCAM, input the search data, and check the PE output *Match_Addr[4:0]*. Figure.8 shows the full waveform. After the write operations, *preML<31:0>* are asserted low to precharge the matchlines, and the search signals pull down mismatched lines. In this case we are writing 32'0F0F0F0F to the first 30 rows and 32'0F0F0F0E to the last two, and searching for 32'0F0F0F0F. The output is at address 29 (11101), which is correct. Our worst-delay from the search line to the matchlines is **~47ps**, and **~200ps** through priority encoder.

$T_{sl\text{->}ml}$(ps)	47.154	$T_{sl\text{->}pe}$(ps)	200.076	Power(pj)	55.478

Table.2 Performance of TCAM

Post-layout verification plan

We will complete layout for the TCAM cell, row, and array. Post-extraction, we expect worse much worse performance compared to the schematic, as our array is quite big. Monte-Carlo simulation is appropriate similar to Lab4.

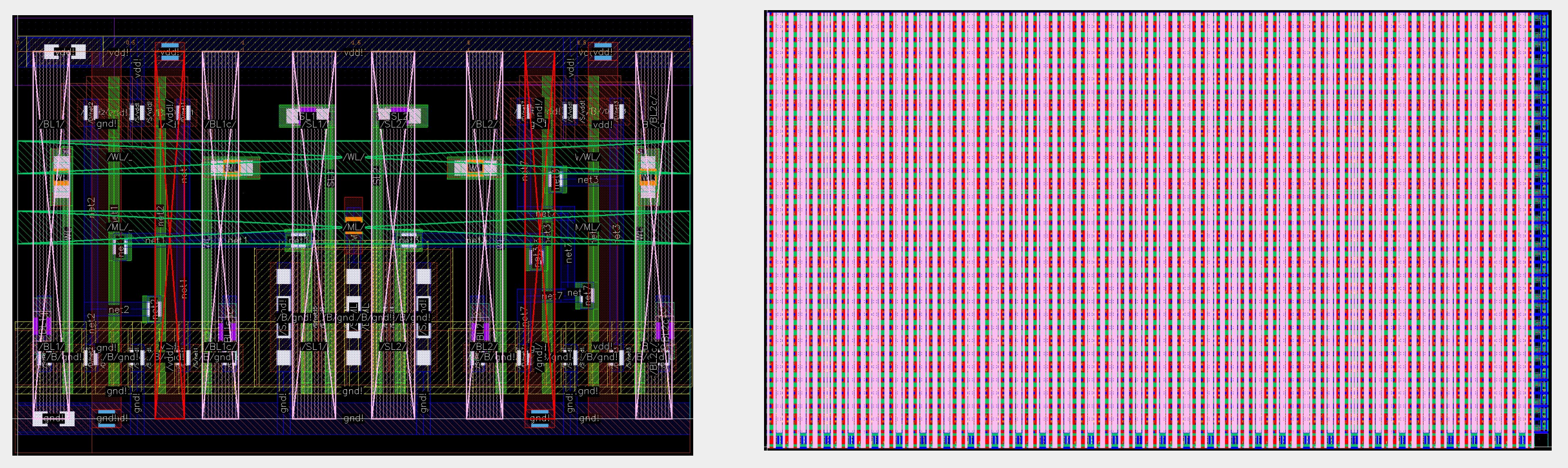


Figure.10 Layout of TCAM cell and array (from left to right)

Discussion

The Verilog-A testbench was tricky to setup: we could not find a way to generate signals with loops. We utilized Python (!) scripts to generate the ~5000 lines of Verilog-A code needed to write full 32-bit values to the 32 entries.. On the schematic side, we had to experiment sizings for the ML and SL transistors. The size of our array meant that these transistors have to be strong in order to pull to the correct value. For example, inefficient size in the pull-down NMOS will result in the matchline staying high when only one bit misses.

References

- S. Kumar, A. Noor, B. K. Kaushik and B. Kumar, "Design of Ternary Content Addressable Memory (TCAM) with 180 nm," 2011 International Conference on Devices and Communications (ICDeCom), Mesra, India, 2011, pp. 1-5, doi: 10.1109/ICDECOM.2011.5738528.
- K. Pagiamtzis and A. Sheikholeslami, "Content-addressable memory (CAM) circuits and architectures: a tutorial and survey," in IEEE Journal of Solid-State Circuits, vol. 41, no. 3, pp. 712-727, March 2006, doi: 10.1109/JSSC.2005.864128.