PROJECT IDEAS · M.TECH · VLSI · 2026-27

M.Tech VLSI projects, *with base papers and free tools*

These 30 M.Tech VLSI project ideas each start from an open-access IEEE paper published from 2023 to 2026, in five groups: RTL and FPGA, low-power arithmetic, hardware security and test, memory and analog, and ML for EDA. Every card gives **a route with free tools**, such as Vivado, Yosys, OpenROAD and SKY130, and one measured extension.

Last updated 2026-11-17

Every paper on this page was checked against its DOI record on Crossref on 17 November 2026, so the title, first author, journal and year are as IEEE registered them. All 30 are open access, in IEEE Access or the IEEE Open Journal of Circuits and Systems, and each card says only what the paper's abstract says. For our M.Tech service for ECE students, see M.Tech projects for ECE; for papers in wireless, antennas and signal processing, see the IEEE base papers for ECE.

KEY POINTS

- **30 M.Tech VLSI project ideas,** each built on an open-access IEEE paper from 2023 to 2026 and checked against its DOI record.
- **Five groups of six:** RTL, RISC-V and FPGA; low-power and approximate arithmetic; hardware security and testing; memory and analog; and ML for EDA and physical design.
- **Every card has a route with free tools:** Vivado ML Standard or Quartus Prime Lite for FPGA work, and Yosys, OpenROAD, Magic, Xschem and ngspice with the SkyWater SKY130 PDK for chip design. Cadence and Synopsys appear only as an option if your college lab has them.
- **Every card names one measured change** and the table that compares it with your rebuild of the base paper, in the same tools and the same process.
- **27 papers are in IEEE Access and 3 in the IEEE Open Journal of Circuits and Systems,** so you and your examiner can read every one free.

How do you choose an M.Tech VLSI project?

Decide three things, in this order: the kind of VLSI work you want to do, the tools you can actually use, and whether the work fits the two stages of your M.Tech.

Front-end, back-end, FPGA or analog?

VLSI projects fall into five tracks. Each produces a different kind of result, and each has a free route.

M.Tech VLSI project tracks: what each produces, the free tools for it and the licensed tools to use if your lab has them

TRACK	WHAT YOU PRODUCE	FREE TOOLS	IF YOUR COLLEGE LAB HAS IT
Front-end (RTL and synthesis)	Verified RTL, and synthesis reports for area, power and timing	Icarus Verilog or Verilator, GTKWave, Yosys, OpenSTA, SKY130 standard cells	Synopsys VCS and Design Compiler, or Cadence Xcelium and Genus

TRACK	WHAT YOU PRODUCE	FREE TOOLS	IF YOUR COLLEGE LAB HAS IT
FPGA	RTL running on a board, with utilisation, timing and power reports and measurements from the board	Vivado ML Standard for Artix-7 and Zynq-7000 boards, Quartus Prime Lite for MAX 10 and Cyclone V boards	Licensed editions of Vivado or Quartus for larger devices
Back-end (physical design)	A placed and routed layout (GDS) with post-layout timing and power, and clean DRC and LVS checks	OpenROAD, through the OpenROAD flow scripts or LibreLane (the successor to OpenLane), with Magic, KLayout and Netgen on the SKY130 or GF180MCU PDK	Cadence Innovus, or Synopsys IC Compiler II and PrimeTime
Analog, mixed-signal and memory	Schematic and layout of a cell or block, simulated at process corners and with Monte Carlo runs	Xschem, ngspice, Magic and Netgen on SKY130	Cadence Virtuoso and Spectre, or Synopsys HSPICE
ML for EDA	A model or agent that predicts or improves one step of the design flow, tested on designs it never saw	Python and PyTorch on a free Colab or Kaggle GPU, with OpenROAD to generate data	Commercial tools are optional

The free tools above install on a Linux computer; the IIC-OSIC-TOOLS Docker image bundles most of the chip design ones with the SKY130 and GF180MCU PDKs.

Front-end and FPGA projects suit students who enjoy Verilog and want working results early. Back-end projects suit those who want to learn placement, routing and timing closure, which take weeks to learn in any tool. Analog and memory projects need comfort with transistor-level circuits and SPICE, and give fewer but deeper results.

Which tools does your lab have?

- **Ask in the first week.** Find out from your guide which licences the department has, on which machines, and whether you can book time on them.
- **No licences?** Every idea on this page has a free route, from RTL or schematic to a layout.
- **Licences, but only on lab machines?** Develop and debug with the free tools at home, and run the final numbers in the lab.

- **Same flow for both designs.** Run the base design and yours through the same tools, library or PDK, corner and clock. If the paper used 28 nm or 65 nm, rebuild its design in your process rather than comparing your numbers with its figures.
- **Write the setup down.** Tool versions, PDK, corner, supply voltage and clock all go into the methodology chapter.

Does it fit Stage-I and Stage-II?

Most universities split the M.Tech project into two stages across the second year, each with a report and a review. Names and dates vary, so check your university's current regulations.

1. **Stage-I: base paper, tools and rebuild.** Choose the base paper (our guide on how to select a base paper covers the checks), install the tools, rebuild the base design with a testbench, and get first synthesis or simulation numbers. Fix the one change and the table you will fill in Stage-II, and present it; our base paper presentation guide covers the slides.
2. **Stage-II: extension, PPA tables and thesis.** Build the change, run the base design and yours through the same flow, fill the tables after synthesis and after layout, then write the thesis and, if your course asks for one, a paper.

Some tracks need an early start. Get a small design through the full OpenROAD or analog flow during Stage-I, because the first complete run takes longest. Order FPGA boards in Stage-I, and two or more of them for PUF work. For ML for EDA, generating your own data from tool runs can take weeks of computer time.

Which VLSI projects suit M.Tech in 2026-27?

Thirty ideas in five groups of six. Each card gives the problem and what you build and measure. Open "The base paper, the tools and the M.Tech extension" to see the paper, linked by its DOI, the free tools and one change to measure, with the table it produces. The track and difficulty on each card are our estimate for a two-stage M.Tech, not figures from the paper.

Topic ideas, not delivered projects

Each card is built on an open-access IEEE paper, checked against its DOI record on Crossref, and describes only what the paper's abstract states. The project around it is an idea to discuss with your guide. Our 8 delivered case studies are M.Tech and M.E. projects, each shown with its real paper pages, screens and results.

Digital design, RISC-V and FPGA accelerators

Write the RTL in Verilog or SystemVerilog, verify it in simulation, then measure it on a board. Vivado ML Standard and Quartus Prime Lite are free, and they cover low-cost Artix-7, Zynq-7000, MAX 10 and Cyclone V boards.

TOPIC IDEA · M.TECH · RTL AND FPGA

RISC-V core with packed-SIMD instructions for CNN inference on an FPGA

Base paper: IEEE Access, 2025

Problem: A small 32-bit processor in a low-cost camera or sensor node runs a CNN slowly, one operation at a time, and adding a separate AI chip raises the price of the product.

Output: Build the paper's RV32IM core with the draft RISC-V P-extension, run its matrix multiplication, convolution, max-pooling and LeNet-5 tests on an FPGA, and measure cycles, speed-up, LUTs, flip-flops and power with and without the packed instructions.

− The base paper, the tools and the M.Tech extension

Base paper: P-CORE: Exploring RISC-V Packed-SIMD Extension for CNNs, M. Ali et al., IEEE Access, 2025
Tools and M.Tech extension: Tools: Verilog or SystemVerilog, Icarus Verilog or Verilator for simulation, a RISC-V GCC toolchain, and the free Vivado ML Standard with an Artix-7 or Zynq-7000 board such as the Basys 3 or PYNQ-Z2 (the paper used a Zynq UltraScale+ device). M.Tech extension: add one custom instruction of your own, such as a packed 8-bit multiply-accumulate that also adds the bias, and tabulate cycles per LeNet-5 layer, LUTs, flip-flops, DSPs, Fmax and power for RV32IM, RV32IM with the P-extension, and your extended core. Track: FPGA and RTL.
Difficulty: medium

TOPIC IDEA · M.TECH · RTL AND FPGA

Post-quantum key exchange (ML-KEM) on a small RISC-V core with custom instructions

Base paper: IEEE Access, 2024

Problem: Smart electricity meters, payment terminals and smart cards need key exchange that stays safe once large quantum computers arrive, and ML-KEM (Kyber), the algorithm NIST standardised for it, runs slowly in software on their small 32-bit processors.

Output: Rebuild the paper's Ibex RISC-V core with its Xkyber instructions for polynomial arithmetic, sampling and coefficient compression, run Kyber-512 encapsulation, and measure cycles, energy, code size and added area against the plain core.

− The base paper, the tools and the M.Tech extension

Base paper: Hardware Acceleration of Crystals-Kyber in Low-Complexity Embedded Systems With RISC-V Instruction Set Extensions, C. Gewehr et al., IEEE Access, 2024
Tools and M.Tech extension: Tools: the authors' RTL and software, which the abstract says are public on GitHub; Verilator for cycle counts; Yosys and OpenROAD with the SKY130 PDK for gate count and area; and an Artix-7 board with Vivado ML Standard to run the core on hardware. M.Tech extension: widen the polynomial unit so that it also handles the modulus of ML-DSA (Dilithium), the NIST post-quantum

Difficulty: medium.

(Dilithium), the NIST post-quantum signature, and tabulate cycles, code size and gate count for Kyber and Dilithium routines on the plain Ibex, with the paper's extension and with your shared unit. Track: front-end (RTL and synthesis). Difficulty: high.

TOPIC IDEA · M.TECH · RTL AND FPGA

Pipelined CNN accelerator with on-chip weights and line buffers

Base paper: IEEE Access, 2026

Problem: Inspection cameras on a production line and traffic cameras have to classify each frame in microseconds, and an accelerator that keeps fetching weights from external memory loses both time and power.

Output: Implement the paper's layer-wise streaming CNN in Verilog, with line buffers between layers and all weights stored on chip, verify each layer's timing in simulation, and report cycles per image, worst slack, resources and on-chip power after implementation.

The base paper, the tools and the M.Tech extension

Base paper: **A High-Speed, Low-Power Pipelined CNN Implementation on FPGA Using On-Chip Processing and Layer-Wise Streaming Control**, J. Pandey and A. R. Asati, IEEE Access, 2026
Tools and M.Tech extension: Tools: Python with TensorFlow or PyTorch to train the small CNN on the dataset the paper names, Verilog, and Vivado ML Standard with a Zynq-7000 board such as the PYNQ-Z2 (the paper used a ZCU104 and a signed 32-bit Q18.13 fixed-point format). M.Tech extension: cut the word length to 16-bit and then 8-bit fixed point, and tabulate classification accuracy, LUTs, flip-flops, DSPs, BRAMs, clock period,

TOPIC IDEA · M.TECH · RTL AND FPGA

Multiplication-free CNN accelerator for FPGAs with few DSP blocks

Base paper: IEEE Access, 2024

Problem: The low-cost FPGA boards most college labs can afford have few DSP multipliers, so a CNN accelerator that depends on them runs out of multipliers long before it runs out of logic.

Output: Reproduce the paper's lookup-based dot product with residual vector quantisation (RLDA), measure the accuracy it loses on a CNN, and build the table-lookup processing element in Verilog without DSP blocks.

The base paper, the tools and the M.Tech extension

Base paper: **Multiplication-Free Lookup-Based CNN Accelerator Using Residual Vector Quantization and Its FPGA Implementation**, H. Fuketa et al., IEEE Access, 2024
Tools and M.Tech extension: Tools: Python with PyTorch for the quantisation and accuracy tests (a small CNN on CIFAR-10 trains on a free Colab GPU; ResNet-18 on ImageNet, as in the paper, needs far more compute), Verilog, and Vivado ML Standard with an Artix-7 or Zynq-7000 board. M.Tech extension: build two versions of one convolution engine on the same low-cost board, an INT8 version on DSP blocks and an RLDA version on lookup tables, and tabulate accuracy, LUTs, DSPs, BRAMs, throughput and power for each. Track: FPGA. Difficulty: high.

microseconds per image and on-chip power for the 32-bit, 16-bit and 8-bit versions. Track: FPGA. Difficulty: medium.

TOPIC IDEA · M.TECH · RTL AND FPGA

Reconfigurable pipelined FFT processor for OFDM receivers

Base paper: IEEE Access, 2023

Problem: Wi-Fi, 4G and 5G receivers switch between FFT sizes, and a pipelined FFT built for the largest size leaves half its butterflies and multipliers idle on the smaller ones.

Output: Implement the paper's flexible-channel multiple-path delay feedback (FC-MDF) radix-2 FFT, which runs in single-channel mode for the larger size and in dual-channel mode for the smaller one, and compare utilisation, throughput and FPGA resources with a standard radix-2 single-path delay feedback (R2SDF) FFT.

The base paper, the tools and the M.Tech extension

Base paper: A Flexible-Channel MDF Architecture for Pipelined Radix-2 FFT, X. Zhou et al., IEEE Access, 2023
Tools and M.Tech extension: Tools: MATLAB or Python with NumPy for the fixed-point reference, Verilog, Icarus Verilog or the Vivado simulator, and Vivado ML Standard with any Artix-7 board. M.Tech extension: rebuild the design with a radix-2^2 decomposition, which needs fewer non-trivial twiddle-factor multiplications, and tabulate DSPs, LUTs, Fmax, throughput in each mode and signal-to-quantisation-noise ratio for R2SDF, the paper's FC-MDF and your version. Track: FPGA and RTL. Difficulty: medium.

TOPIC IDEA · M.TECH · RTL AND FPGA

Low-power lane detection on a DE1-SoC board, tested on Indian roads

Base paper: IEEE Access, 2024

Problem: Lane markings on many Indian roads are faded, broken or missing, and a driver-assistance camera has to find the lanes on a small, low-power board inside the vehicle.

Output: Rebuild the paper's pipeline, with grayscale conversion, Gaussian smoothing, Sobel, non-maximum suppression, hysteresis and the Hough transform, split between the FPGA fabric and the ARM processor of a DE1-SoC, and measure run time, clock cycles, power and energy on each side.

The base paper, the tools and the M.Tech extension

Base paper: Low-Power Lane Detection Unit With Sliding-Based Parallel Segment Detection Accelerator for FPGA, H. Yun and D. Park, IEEE Access, 2024
Tools and M.Tech extension: Tools: the free Quartus Prime Lite, a Terasic DE1-SoC board, VHDL or Verilog, C for the ARM side, and dash-camera clips you record on local roads and label for lanes. M.Tech extension: add edge thresholds that adapt to each frame's brightness and contrast, and tabulate lane detection rate, false detections, clock cycles per frame, logic use and power for fixed and adaptive thresholds on day, dusk and faded-marking clips. Track: FPGA

(hardware and software together). Difficulty: medium.

Low-power and approximate arithmetic

Report error metrics and application quality beside area, power and delay. An approximate circuit is worth something only if its output, such as an image or a classification, stays good enough.

TOPIC IDEA · M.TECH · LOW-POWER DESIGN

Approximate multiplier with configurable accuracy, on standard cells and on an FPGA

Base paper: IEEE Access, 2024

Problem: Phones and battery-powered cameras run smoothing, sharpening and JPEG on every frame, and an exact multiplier spends energy on low-order bits that the eye does not notice.

Output: Implement AMCAL in Verilog with its truncation and bit-manipulation scheme, compute its error metrics, compare delay, power and area with an exact Wallace multiplier and the DRUM approximate multiplier, and run the paper's smoothing, sharpening and JPEG tests.

The base paper, the tools and the M.Tech extension

Base paper: **AMCAL: Approximate Multiplier With the Configurable Accuracy Levels for Image Processing and Convolutional Neural Network**, R. Zendegani and S. Safari, IEEE Access, 2024
Tools and M.Tech extension: Tools: Verilog, Yosys and OpenROAD with the SKY130 PDK for area, power and delay (Cadence Genus or Synopsys Design Compiler if your college lab has it), Python for error metrics such as MRED and for the image tests, and Vivado ML Standard for the FPGA side. M.Tech extension: savings designed for standard cells may shrink on FPGA lookup tables, so map the exact, DRUM and AMCAL

TOPIC IDEA · M.TECH · LOW-POWER DESIGN

Low-power DNN accelerator built on an approximate signed multiplier

Base paper: IEEE Open Journal of Circuits and Systems, 2024

Problem: Edge AI cameras and sensors run neural networks on batteries, and the multipliers in the accelerator's processing elements take much of the power.

Output: Build the paper's 8-bit signed approximate multiplier, with its probability-optimised compressors, partial-product truncation and error compensation, use it as the processing element of a small systolic array, and measure power, area, delay and the accuracy lost on a CNN.

The base paper, the tools and the M.Tech extension

Base paper: **A Low-Power DNN Accelerator With Mean-Error-Minimized Approximate Signed Multiplier**, L. Du et al., IEEE Open Journal of Circuits and Systems, 2024
Tools and M.Tech extension: Tools: Verilog, Yosys and OpenROAD on SKY130 (Synopsys Design Compiler if your college lab has it), and Python with PyTorch to model the multiplier bit for bit inside a small VGG-style network on CIFAR-10, which trains on a free Colab GPU (the paper evaluated VGG-16). M.Tech extension: run a sensitivity sweep to find the layers that tolerate the approximate multiplier, use it only there, and tabulate accuracy, power,

designs to an Artix-7 as well, and tabulate SKY130 area, power and delay beside LUTs, Fmax and FPGA power for all three. Track: front-end (RTL and synthesis) and FPGA. Difficulty: medium.

area and energy per inference for an all-exact array, an all-approximate array and your mixed array. Track: front-end (RTL and synthesis). Difficulty: medium.

TOPIC IDEA · M.TECH · LOW-POWER DESIGN

Approximate recursive multipliers taken to layout with open-source tools

Base paper: IEEE Access, 2026

Problem: The paper synthesised its multipliers with Cadence Genus, which not every college lab has, and synthesis numbers alone do not show how much of the saving remains after placement and routing.

Output: Build the paper's four gate-level approximate building blocks and its 8×8 and 16×16 recursive multipliers in Verilog, take them and an exact multiplier through Yosys and OpenROAD on SKY130, and compare area, power and delay, with the paper's image smoothing, sharpening and multiplication tests.

The base paper, the tools and the M.Tech extension

Base paper: Efficient Approximate Recursive Multipliers (EARM) for Area and Power Optimized Image Processing Applications, K. Srinivas and G. K. Kumar, IEEE Access, 2026
Tools and M.Tech extension: Tools: Verilog, Icarus Verilog, Yosys, OpenROAD (directly, through the OpenROAD flow scripts, or through LibreLane, the successor to OpenLane), Magic or KLayout with SKY130, and Python for PSNR and SSIM; Cadence Genus and Innovus if your college lab has them. M.Tech extension: report every design twice, after synthesis and after place-and-route with extracted parasitics, and tabulate area, power, delay and

TOPIC IDEA · M.TECH · LOW-POWER DESIGN

Approximate floating-point adder with run-time precision control

Base paper: IEEE Access, 2025

Problem: Wearables and smart sensors run filters and small neural networks in floating point, and a full-precision adder uses the same power whether the application needs full accuracy or not.

Output: Implement the paper's adder, which skips the addition when the exponent difference passes a programmable threshold and rounds the mantissa to a tunable width, and measure power and mean relative error distance at each setting in FP32 and FP16.

The base paper, the tools and the M.Tech extension

Base paper: Novel Approximate Floating-Point Adder With Runtime Tunable Precision, L. Tegazzini et al., IEEE Access, 2025
Tools and M.Tech extension: Tools: SystemVerilog, Verilator with a C++ or Python reference model, and Yosys and OpenROAD on SKY130 for power, with switching activity taken from simulation (the paper used a 28 nm technology, so compare your own exact and approximate designs, not your numbers with the paper's). M.Tech extension: add a bfloat16 mode, the 16-bit format many ML accelerators use, and tabulate area, power, mean relative error distance and the error in a convolution filter and an LSTM test for FP16 and bfloat16 at each threshold setting.

power-delay product at both stages for the exact multiplier and the paper's S8–4 design, so the thesis shows how much of the saving survives layout. Track: back-end (RTL to layout). Difficulty: medium.

Track: front-end (RTL and synthesis). Difficulty: medium.

TOPIC IDEA · M.TECH · LOW-POWER DESIGN

Stochastic computing multiply-accumulate unit for image processing on an FPGA

Base paper: IEEE Access, 2025

Problem: Simple vision tasks on tiny battery-powered devices, such as edge detection in a farm or factory sensor, need very little logic, and stochastic computing does arithmetic with single gates on random bit streams.

Output: Build the paper's multiplexer-based stochastic MAC with its auxiliary logic block for the select inputs, run grayscale conversion and Sobel edge detection on an FPGA, and compare resources and accuracy with AND- and XNOR-based stochastic multipliers.

The base paper, the tools and the M.Tech extension

Base paper: Toward Universal Multiplexer Multiply-Accumulate Architecture in Stochastic Computing, Y. Y. Lee et al., IEEE Access, 2025
Tools and M.Tech extension: Tools: Verilog, LFSR-based bit-stream generators, Vivado ML Standard with an Artix-7 board such as the Basys 3, and Python for PSNR against a floating-point reference. M.Tech extension: vary the bit-stream length, for example 64, 128 and 256 bits, compare with an 8-bit binary design, and tabulate the PSNR of the Sobel output, LUTs, flip-flops, cycles per pixel and energy per pixel for

TOPIC IDEA · M.TECH · LOW-POWER DESIGN

RTL power reduction for the MAC array of an FPGA object detector

Base paper: IEEE Access, 2026

Problem: Drones and smart cameras that run object detection on an FPGA are limited by battery power, and much of the dynamic power goes into multiply-accumulate units that switch when they do not need to.

Output: Apply the paper's RTL techniques to a CNN accelerator's MAC datapath, including change-detect enables, operand bypass, pipeline valid-gating and activity-based gating, and measure dynamic and total on-chip power, timing and LUTs before and after, with outputs checked bit for bit against the original.

The base paper, the tools and the M.Tech extension

Base paper: Ultra-Low-Power Multiply and Accumulate (MAC) Design on FPGA for Real-Time Edge AI Applications, A. Gundrapally et al., IEEE Access, 2026
Tools and M.Tech extension: Tools: Verilog, Vivado ML Standard with a switching activity file (SAIF) from simulation for the power report, and a Zynq-7000 board such as the PYNQ-Z2 running a small CNN accelerator (the paper used a Tiny-YOLOv4 accelerator built with Tensil on a ZCU104 and checked power with the board's INA226 current sensors). M.Tech extension: apply the techniques one at a time as well as together, and tabulate dynamic

each. Track: FPGA. Difficulty: low.

power, total on-chip power, LUTs, flip-flops, worst negative slack and frames per second at each step, so the thesis shows which technique saves the most. Track: FPGA. Difficulty: medium.

Hardware security and testing

A security result needs both sides: how much harder the attack becomes, and what the protection costs in area, power or speed. PUF work needs more than one board, so order them in Stage-I.

TOPIC IDEA · M.TECH · HARDWARE SECURITY

Graph neural network that finds hardware Trojans in FPGA netlists

Base paper: IEEE Access, 2026

Problem: Many chips designed in India are made at foundries abroad and use third-party IP blocks, so the design team cannot inspect every gate that reaches silicon, and a hidden Trojan can leak data or disable the chip on command.

Output: Synthesise Trust-Hub RTL benchmarks into gate-level netlists with Vivado, turn each netlist into a graph, train the paper's MultiSAINT model with focal loss, and report accuracy, F1 and recall per circuit against a TrojanSAINT baseline.

The base paper, the tools and the M.Tech extension

Base paper: MultiSAINT: Parallel Multi-Scale GNN for FPGA Hardware Trojan Detection, I. Yanti et al., IEEE Access, 2026
Tools and M.Tech extension: Tools: the Trust-Hub benchmarks, Vivado ML Standard for synthesis, Python with PyTorch Geometric, and a free Colab GPU. M.Tech extension: synthesise the same benchmarks a second way, with Yosys onto SKY130 standard cells, and tabulate recall, F1 and false positives when the model is trained on Vivado netlists and tested on Yosys netlists, before and after fine-tuning on a few Yosys netlists. Track: hardware security (machine learning on netlists). Difficulty: medium.

TOPIC IDEA · M.TECH · HARDWARE SECURITY

Machine learning that flags hardware Trojans in open-source RTL such as RISC-V cores

Base paper: IEEE Access, 2025

Problem: Students and start-ups reuse open-source cores and IP blocks from GitHub, and open source alone does not guarantee that nobody has hidden a Trojan in the RTL.

Output: Rebuild the paper's pipeline, in which RTL code is turned into TF-IDF features and classified by a decision tree, on Trojan-free and Trojan-inserted designs from the ISCAS and Trust-Hub benchmarks, and report accuracy, recall and false negatives.

The base paper, the tools and the M.Tech extension

Base paper: Hardware Trojan Detection in Open-Source Hardware Designs Using Machine Learning, V. T. Hayashi and W. V. Ruggiero, IEEE Access, 2025
Tools and M.Tech extension: Tools: Python with scikit-learn, the Trust-Hub and ISCAS benchmarks, open-source cores such as PicoRV32 or Ibex as test designs, and an open-weights LLM on a free Colab GPU if you add the paper's LLM step. M.Tech extension: insert your own Trojans into a RISC-V core that never appears in training, and tabulate accuracy, recall and false-positive rate on a random split and on this held-out core, for TF-IDF with a decision tree and for one code-

embedding model. Track: hardware security (machine learning on RTL). Difficulty: low.

TOPIC IDEA · M.TECH · HARDWARE SECURITY

Ring-oscillator PUF on an FPGA that resists machine-learning attacks

Base paper: IEEE Access, 2023

Problem: Cheap IoT devices and smart cards cannot afford secure memory for their keys; a physically unclonable function (PUF) derives a key from chip-to-chip variation instead, but simple PUFs can be cloned by machine-learning attacks.

Output: Implement a ring-oscillator PUF on an FPGA, add the paper's modulus step on the frequency-counter outputs, collect challenge-response pairs, run machine-learning attacks on both versions, and measure uniqueness, uniformity and reliability.

The base paper, the tools and the M.Tech extension

Base paper: A Design Strategy to Improve Machine Learning Resiliency for Ring Oscillator Physically Unclonable Function, Y. Jiang et al., IEEE Access, 2023
Tools and M.Tech extension: Tools: Verilog, Vivado ML Standard and at least two identical Artix-7 boards such as the Basys 3 (uniqueness cannot be measured on one chip), a UART link to log responses, and Python with scikit-learn for logistic regression and MLP attacks. M.Tech extension: compare the plain RO-PUF, the modulus RO-PUF and a two-input XOR RO-PUF on the same boards, and tabulate attack accuracy against the number of training pairs, uniqueness, uniformity,

TOPIC IDEA · M.TECH · HARDWARE SECURITY

True random number generator built from ring oscillators and an ADPLL

Base paper: IEEE Access, 2025

Problem: Encryption keys, nonces and one-time passwords in payment and IoT devices are only as safe as the random numbers behind them, and a software generator is predictable once its seed leaks.

Output: Build the paper's TRNG, with ten ring oscillators built from ADPLL-based digitally controlled oscillators, sampling flip-flops, an XOR combiner and an XOR-corrector post-processing circuit, and measure throughput, LUTs, flip-flops, energy per bit and the NIST SP 800-22 results.

The base paper, the tools and the M.Tech extension

Base paper: Design and Implementation of Multiple Ring Oscillator-Based TRNG Architecture by Using ADPLL, H. Bharat Meitei and M. Kumar, IEEE Access, 2025
Tools and M.Tech extension: Tools: VHDL or Verilog, Vivado ML Standard with an Artix-7 board such as the Basys 3 (the paper used Artix-7, Kintex-7 and Zynq-7000 devices), a UART or USB link to stream bits to a PC, and the free NIST SP 800-22 statistical test suite. M.Tech extension: add the on-line health tests of NIST SP 800-90B, the repetition count test and the adaptive proportion test, in hardware, and tabulate throughput,

reliability at room temperature and with the board warmed, and LUTs. Track: hardware security on FPGA. Difficulty: medium.

LUTs, flip-flops, energy per bit and NIST pass rates without and with them, at room temperature and with the board warmed. Track: hardware security on FPGA. Difficulty: low.

TOPIC IDEA · M.TECH · HARDWARE SECURITY

Power side-channel attack and defence on a RISC-V core with AES instructions

Base paper: IEEE Access, 2024

Problem: Smart cards, payment terminals and IoT nodes run AES on small processors, and an attacker who records the chip's power draw while it encrypts can recover the secret key.

Output: Build the paper's 32-bit RISC-V cores with custom AES instructions, a 3-stage core with a software pause and a 4-stage core with a hardware pause, generate power traces during encryption, run a power analysis attack such as correlation power analysis, and compare the two cores on the paper's security metrics.

The base paper, the tools and the M.Tech extension

Base paper: Power Side-Channel Attacks on Crypto-Core Based on RISC-V ISA for High-Security Applications, T. M. Ignatius et al., IEEE Access, 2024
Tools and M.Tech extension: Tools: Verilog, Vivado ML Standard and a Basys 3 board as in the paper, Icarus Verilog to dump switching activity from a gate-level simulation, and Python for a Hamming-distance power model and the attack, in place of the Synopsys PrimeTime PX traces the paper used (use PrimeTime PX if your college lab has it). M.Tech extension: add a countermeasure, such as random dummy cycles or a

TOPIC IDEA · M.TECH · HARDWARE SECURITY

Secure scan chain with key-based obfuscation against scan attacks

Base paper: IEEE Access, 2024

Problem: Scan chains are added so that the factory can test every chip, but the same chains let an attacker shift out the internal state of a crypto chip and read its key.

Output: Insert a scan chain into an AES core, add the paper's key-controlled obfuscation cells between the scan flip-flops and its controller at the scan-out port, and measure area overhead, fault coverage and whether a scan-based key attack still succeeds.

The base paper, the tools and the M.Tech extension

Base paper: A Low-Overhead and High-Security Scan Design Based on Scan Obfuscation, W. Wang et al., IEEE Access, 2024
Tools and M.Tech extension: Tools: Verilog, Yosys, and an open-source flow for scan insertion and test generation, such as the Fault toolchain or the DFT commands in OpenROAD, on SKY130 (Synopsys DFT Compiler and TetraMAX, or Cadence Modus, if your college lab has them). M.Tech extension: apply the same scheme to a lightweight cipher such as PRESENT, whose scan chain is much shorter, and tabulate area overhead, fault coverage, number of test patterns and test time for plain and obfuscated scan on AES and on PRESENT. Track: design for testability.

masked S-box, and tabulate the number of traces needed to recover the key, area, power and cycles per encryption for the unprotected core and the protected one. Track: hardware security (RTL and simulation). Difficulty: high.

Difficulty: medium.

Memory, analog and mixed-signal circuits

Full-custom work with Xschem, ngspice and Magic on the free SkyWater SKY130 PDK. Report results at process corners and from Monte Carlo runs, before and after layout extraction, and do not compare your 130 nm numbers directly with a paper's 28 nm or 65 nm ones.

TOPIC IDEA · M.TECH · ANALOG AND MEMORY

SRAM cell sizing for noise margin, speed and leakage with the SKY130 PDK

Base paper: IEEE Access, 2025

Problem: Every chip with a processor is full of SRAM, and the sizes of the cell's transistors decide how stable it is, how fast it reads and writes, and how much it leaks while the device sleeps.

Output: Apply the paper's transistor-sizing algorithm to an SRAM cell with its precharge and write-driver circuits, and measure read, write and hold static noise margins, access times and leakage, with Monte Carlo runs across process, voltage and temperature.

The base paper, the tools and the M.Tech extension

Base paper: **A Novel Algorithm for Aspect Ratio Estimation in SRAM Design to Achieve High SNM, High Speed, and Low Leakage Power**, S. M. Mantrashetti et al., IEEE Access, 2025
Tools and M.Tech extension: Tools: Xschem for the schematic, ngspice with the SKY130 models (the paper used 45 nm at 1.2 V), Magic for the layout, and Python to sweep sizes and plot butterfly curves; Cadence Virtuoso and Spectre if your college lab has them. M.Tech extension: port the algorithm to SKY130, and tabulate read, write and hold SNM, read and write access times, leakage power and cell area for a cell sized by textbook ratios and one sized by the

TOPIC IDEA · M.TECH · ANALOG AND MEMORY

Ultralow-voltage retention SRAM cell with Schmitt-trigger inverters

Base paper: IEEE Open Journal of Circuits and Systems, 2025

Problem: Wearables and IoT nodes sleep most of the time, and the SRAM that holds their data during sleep keeps leaking, so the battery drains even when the device does nothing.

Output: Design the paper's 8T cell with its new Schmitt-trigger inverters, find how low the supply can go while the cell still holds its data, and measure hold noise margin and standby leakage against a standard 6T cell.

The base paper, the tools and the M.Tech extension

Base paper: **A New Ultralow-Voltage Retention SRAM Cell Enhancing Noise Immunity**, K. Ito et al., IEEE Open Journal of Circuits and Systems, 2025
Tools and M.Tech extension: Tools: Xschem and ngspice with the SKY130 PDK and its Monte Carlo mismatch models, Magic for the layout, and Python to extract noise margins from the butterfly curves. M.Tech extension: find the lowest retention voltage in SKY130 for the 6T cell and for the paper's 8T cell, and tabulate hold noise margin (the worst case over the Monte Carlo runs), standby leakage per bit and cell area at each supply voltage in your sweep. Track: memory (full-custom). Difficulty: medium.

algorithm, at three process corners and from 500 Monte Carlo runs. Track: memory (full-custom). Difficulty: medium.

TOPIC IDEA · M.TECH · ANALOG AND MEMORY

8T SRAM compute-in-memory column with a current mirror for accurate MAC

Base paper: IEEE Access, 2024

Problem: Edge AI chips spend much of their energy moving weights between memory and the processor; computing inside the SRAM array avoids that, but bit-line currents add up non-linearly.

Output: Design an 8T SRAM column that multiplies 4-bit inputs by stored 4-bit weights in the current domain, add the paper's cascaded current mirror and its read word-line pulses timed in the middle of each slot, and measure MAC linearity, energy per operation and throughput.

The base paper, the tools and the M.Tech extension

Base paper: 8T-SRAM Based Process-In-Memory (PIM) System With Current Mirror for Accurate MAC Operation, K.-Y. Chung et al., IEEE Access, 2024
Tools and M.Tech extension: Tools: Xschem and ngspice with the SKY130 PDK, Magic for the layout, and Python to compare the column's output with the ideal dot product and to map a small MNIST network onto it (the paper's chip was made in a 65 nm process). M.Tech extension: simulate the same column with and without the cascaded current mirror across process corners and temperature, and tabulate MAC integral non-linearity in output steps, energy per MAC and the MNIST

TOPIC IDEA · M.TECH · ANALOG AND MEMORY

Dynamic comparators for low-power SAR ADCs, compared in SKY130

Base paper: IEEE Open Journal of Circuits and Systems, 2025

Problem: Medical patches and sensor nodes mostly use SAR ADCs, and the comparator sets much of the ADC's power, speed and accuracy.

Output: Rebuild the five voltage-domain dynamic comparators that the paper studies, and measure comparison time, power, offset, kickback and noise for each as the supply voltage is reduced.

The base paper, the tools and the M.Tech extension

Base paper: A Comparative Study of Dynamic Comparators for Low-Power Successive Approximation ADC, F. Yuan, IEEE Open Journal of Circuits and Systems, 2025
Tools and M.Tech extension: Tools: Xschem and ngspice with the SKY130 PDK, the same 130 nm node as the paper but a different foundry, Monte Carlo runs for offset, transient noise simulation for input-referred noise, and Python for the plots; Cadence Spectre if your college lab has it. M.Tech extension: put each comparator inside an 8-bit SAR ADC with a capacitive DAC in ngspice, and tabulate comparison time, energy per comparison, offset sigma, kickback and the ADC's ENOB for each comparator. Track: analog and mixed-signal. Difficulty: medium.

accuracy each version gives when its errors are fed into the network. Track: memory and mixed-signal. Difficulty: high.

TOPIC IDEA · M.TECH · ANALOG AND MEMORY

Compact 5-bit SAR ADC for on-chip temperature sensors

Base paper: IEEE Access, 2026

Problem: Processors and memories place temperature sensors across the die to catch hot spots, and each sensor needs an ADC small enough to repeat many times.

Output: Design the paper's 5-bit SAR ADC, in which voltage interpolating comparison removes the passive components of the reference DAC, and measure DNL, INL, SNDR, power and layout area.

The base paper, the tools and the M.Tech extension

Base paper: Design of a Compact 5-bit SAR ADC With Voltage Interpolating Comparison for On-Chip Temperature Sensors, C.-K. Kwon and Y.-J. Min, IEEE Access, 2026
Tools and M.Tech extension: Tools: Xschem and ngspice with the SKY130 PDK, Magic for the layout and parasitic extraction, and Python for the FFT and for DNL and INL from a ramp test (the paper used 28 nm CMOS, so compare designs you build in the same process). M.Tech extension: design a conventional 5-bit SAR ADC with a capacitive DAC in the same process, and tabulate layout area, power, DNL, INL, SNDR and Walden figure of merit for both, before and after parasitic extraction. Track: analog and mixed-signal. Difficulty: high.

TOPIC IDEA · M.TECH · ANALOG AND MEMORY

Bandgap reference sized by particle swarm optimisation for high PSRR

Base paper: IEEE Access, 2023

Problem: Every ADC, regulator and sensor chip needs a reference voltage that stays steady as temperature changes and as ripple appears on the supply, and tuning a bandgap circuit by hand takes many simulation runs.

Output: Design the paper's bandgap reference with its modified op-amp, drive ngspice from a particle swarm optimisation loop in Python, compare the paper's inertia-weight strategies, and measure temperature coefficient, PSRR at 1 kHz and the number of simulations each strategy needs.

The base paper, the tools and the M.Tech extension

Base paper: Novel Methods for Improved Particle Swarm Optimization in Designing the Bandgap Reference Circuit, T. Hoang et al., IEEE Access, 2023
Tools and M.Tech extension: Tools: Xschem and ngspice with the SKY130 PDK (the paper used 65 nm CMOS), Python with NumPy for the optimisation loop, and Magic for the final layout. M.Tech extension: add the slow and fast process corners to the cost function, so the circuit is optimised for all corners at once, and tabulate temperature coefficient, PSRR at 1 kHz, power and number of simulations at each corner for nominal-only and corner-aware

optimisation. Track: analog.
Difficulty: Medium.

Machine learning for EDA and physical design

These projects need a GPU more than a licence, and a free Colab or Kaggle GPU is enough for most of them. The open OpenROAD flow gives you real layouts, reports and logs to train and test on.

TOPIC IDEA · M.TECH · EDA AND PHYSICAL DESIGN

Early routing congestion prediction with a U-Net and a vision transformer

Base paper: IEEE Access, 2026

Problem: Routing congestion usually shows up only at the end of place-and-route, so a design team, or a student sharing a lab server, loses days to repeated runs.

Output: Train the paper's residual U-Net with depthwise separable convolutions and a vision transformer encoder to predict congestion maps from placement features, and report prediction accuracy and structural similarity on designs left out of training.

The base paper, the tools and the M.Tech extension

Base paper: A Depthwise Separable Residual U-Net With Vision Transformer Encoding for IC Routing Congestion Prediction, R. Jenila and K. J. Naidu, IEEE Access, 2026
Tools and M.Tech extension: Tools: Python with PyTorch on a free Colab or Kaggle GPU, the public CircuitNet dataset of placement features and congestion maps, and OpenROAD with its SKY130 and Nangate45 platforms to make maps of your own; Cadence Innovus if your college lab has it, since the paper compared the model with its early global route. M.Tech extension: generate congestion maps from your own OpenROAD runs, compare the model with OpenROAD's global-route congestion report, and tabulate NRMSE, SSIM, hotspot

TOPIC IDEA · M.TECH · EDA AND PHYSICAL DESIGN

Reinforcement learning that sizes analog circuits from specification to layout in SKY130

Base paper: IEEE Access, 2024

Problem: Sizing an analog circuit by hand takes weeks of simulation, and the commercial optimisers that speed it up are out of reach for students without Cadence or Synopsys licences.

Output: Run the paper's open-source framework, which uses a heterogeneous graph neural network inside a reinforcement learning agent, to size a latched comparator in SKY130 at schematic level and after layout, and measure how many simulations it needs to meet each target specification.

The base paper, the tools and the M.Tech extension

Base paper: An Open-Source AMS Circuit Optimization Framework Based on Reinforcement Learning—From Specifications to Layouts, Z. Li and A. Chan Carusone, IEEE Access, 2024
Tools and M.Tech extension: Tools: the authors' open-source framework, ngspice, Magic and Netgen with the SKY130 and GF180MCU PDKs, and Python with PyTorch. M.Tech extension: apply the agent to a second circuit, such as a two-stage op-amp, compare it with random search and Bayesian optimisation, and tabulate the simulations needed to meet the specification and the gain,

recall and seconds per prediction on designs the model never saw. Track: ML for EDA (physical design). Difficulty: medium.

bandwidth, phase margin and power before and after layout for each method. Track: ML for EDA (analog). Difficulty: high.

TOPIC IDEA · M.TECH · EDA AND PHYSICAL DESIGN

Fine-tuning a small LLM to write synthesisable Verilog from a text specification

Base paper: IEEE Access, 2026

Problem: Students and junior engineers spend long hours writing and debugging Verilog for standard blocks such as counters, FIFOs and state machines, and general chatbots often produce code that does not synthesise.

Output: Fine-tune an open code model with LoRA or QLoRA on pairs of specifications and code, as in the paper, pass every answer through compilation, simulation and a synthesis check, and report pass@k on the VerilogEval benchmark.

− The base paper, the tools and the M.Tech extension

Base paper: Generation of Synthesizable Verilog Code From Natural Language Specifications, D. S. Yashchenko et al., IEEE Access, 2026

Tools and M.Tech extension: Tools: Python with Hugging Face Transformers and PEFT, a free Colab or Kaggle GPU, the public VerilogEval benchmark, Icarus Verilog for simulation and Yosys for the synthesis check. M.Tech extension: write a test set of 30 to 50 problems from your college's digital design lab, each with a testbench, and tabulate pass@1, pass@3, the share of answers that synthesise in Yosys and their cell count against hand-written solutions, for the base

TOPIC IDEA · M.TECH · EDA AND PHYSICAL DESIGN

LLM-assisted design of a single-cycle RISC-V processor on a MAX 10 FPGA

Base paper: IEEE Access, 2026

Problem: Writing a full processor in Verilog is a long job for one student, and large language models write small modules well but often fail when the modules are joined into one design.

Output: Repeat the paper's structured prompting workflow (CLST) to generate the submodules and the top level of an RV32I processor, test each with a testbench, and compare pass rates, generation time, LUTs and clock frequency on a MAX 10 FPGA with a hand-written reference core.

− The base paper, the tools and the M.Tech extension

Base paper: LLM-Aided Design and Evaluation of a RISC-V Processor, T. El-Helwani and M. A. S. Khalid, IEEE Access, 2026

Tools and M.Tech extension: Tools: the free Quartus Prime Lite with a MAX 10 board such as the Terasic DE10-Lite, Icarus Verilog for the testbenches, the RISC-V architectural tests, and an LLM (the paper used ChatGPT with GPT-5.1). M.Tech extension: run the same workflow with an open-weights code model on a free Colab GPU, and tabulate pass@1 and pass@5 for each submodule and the top level, generation time, LUTs and Fmax for a commercial chatbot if you have

model and your fine-tuned one. Track: ML for EDA (RTL generation). Difficulty: medium.

access, the open model and the hand-written reference. Track: ML for EDA (RTL generation) and FPGA. Difficulty: medium.

TOPIC IDEA · M.TECH · EDA AND PHYSICAL DESIGN

LLM agent that diagnoses and repairs failed RTL-to-GDS runs in open-source flows

Base paper: IEEE Access, 2026

Problem: Students running OpenROAD or OpenLane for the first time lose days to failures in constraints, floorplans, placement density or routing, buried in long tool logs.

Output: Rebuild the paper's agent, with stage-aware log parsing, retrieval of EDA knowledge, patch generation and tool reruns that decide whether a fix is accepted, and measure repair success against an LLM that only reads retrieved documents.

The base paper, the tools and the M.Tech extension

Base paper: FlowFixer-EDA: A Tool-Feedback-Augmented Large Language Model Agent for Automated Diagnosis and Repair of Open-Source RTL-to-GDS Design Flow Failures, K. Dong et al., IEEE Access, 2026
Tools and M.Tech extension: Tools: the OpenROAD flow scripts or LibreLane with the SKY130 and Nangate45 platforms, Python, and an open-weights LLM on a free Colab GPU or a local machine; check whether the authors have released their EDA-FlowBench benchmark before you plan around it. M.Tech extension: build a benchmark of 50 or more failures by breaking working runs on purpose, with a clock period

TOPIC IDEA · M.TECH · EDA AND PHYSICAL DESIGN

Post-quantum Saber core taken to layout with an open-source flow and with Cadence

Base paper: IEEE Access, 2025

Problem: Not every college lab has a Cadence or Synopsys licence, and students want to know how far an open-source RTL-to-GDS flow can take a real crypto chip.

Output: Implement the paper's Saber core with its striding Toom-Cook polynomial multiplier, take it through an open-source flow on the SkyWater 130 nm process to a full layout, and report area, timing and clock cycles for key generation, encryption and decryption.

The base paper, the tools and the M.Tech extension

Base paper: Saber With Hybrid Striding Toom Cook-Based Multiplier: Implementation Using Open-Source Tool Flow and Industry Standard Chip Design Tools, M. Naveed Abbasi et al., IEEE Access, 2025
Tools and M.Tech extension: Tools: Verilog, Verilator, and Yosys with OpenROAD (through the OpenROAD flow scripts or LibreLane), Magic and KLayout on SKY130; Cadence Genus and Innovus if your college lab has them, since the paper ran both flows. M.Tech extension: build the core twice, once with a plain schoolbook multiplier and once with the Toom-Cook multiplier, take both through the same open flow on SKY130, and tabulate post-layout area, Fmax,

that is too tight, a placement density that is too high or missing pin constraints, and tabulate repair success, harmful repairs and tool reruns per task for a retrieval-only LLM and for the tool-feedback agent on the RTL-to-GDS flow. Difficulty: medium.

power and cycles for key generation, encryption and decryption. Track: back-end (RTL to GDS). Difficulty: high.

What results must an M.Tech VLSI thesis report?

Your panel looks for a fair comparison: the base design and yours, built with the same tools, the same library or PDK, the same corner and the same clock. These are the results to report.

Results an M.Tech VLSI thesis reports, how to state them and where they come from with free tools

RESULT	WHAT TO REPORT	WHERE IT COMES FROM
Area	Cell area in µm² and, after layout, core area and utilisation (chip design); LUTs, flip-flops, DSPs and BRAMs (FPGA); layout area (custom cells)	Yosys and OpenROAD reports, the Vivado or Quartus utilisation report, Magic
Power	Dynamic and leakage power at a stated voltage, clock and corner, using switching activity from simulation (a VCD or SAIF file), not default toggle rates	OpenSTA or OpenROAD with the activity file, the Vivado power report with SAIF, ngspice for custom cells; board measurements where you can
Timing	Critical path delay or Fmax, worst and total negative slack, latency in cycles and throughput	OpenSTA, OpenROAD, the Vivado timing summary or the Quartus Timing Analyzer
Combined figures	Power-delay product, area-delay product and energy per operation	Worked out from the three rows above
Quality	For approximate designs, error metrics such as MED, MRED and NMED, with PSNR, SSIM or accuracy in the application; detection rates for security work; pass@k for LLM work	Python, against an exact or floating-point reference
Analog and memory figures	Gain, bandwidth, phase margin, PSRR, temperature coefficient, offset and noise; DNL, INL, SNDR, ENOB and figure of merit for an ADC; noise margins, access time and leakage for SRAM	ngspice at each corner and from Monte Carlo runs, with Python for the analysis

Post-synthesis or post-layout?

- **Post-synthesis** numbers come from the gate-level netlist before placement. There are no real wires yet, so delay and power are optimistic, and area is cell area only.
- **Post-layout** numbers come after placement, clock tree synthesis and routing, with parasitics extracted from the layout. They include wire delay, the clock tree and buffers, and the core area at your chosen utilisation.
- **On an FPGA,** the same split is post-synthesis against post-implementation in Vivado or Quartus. Quote post-implementation utilisation and timing.
- **For analog and memory cells,** pre-layout means simulating the schematic, and post-layout means simulating with the parasitics Magic extracts from your layout.
- **Report both where you can,** and label which one each table shows. If a saving shrinks after layout, say so: that is a result too.

Build your main results table like this one, with your own numbers in it.

A template for the PPA comparison table in an M.Tech VLSI thesis

DESIGN	AREA	POWER	DELAY OR FMAX	POWER-DELAY PRODUCT	QUALITY METRIC
Base paper, as published	Paper's figure, with its process	Paper's figure	Paper's figure	Paper's figure	Paper's figure
Base design, your rebuild	Your number	Your number	Your number	Your number	Your number
Your design	Your number	Your number	Your number	Your number	Your number
Change against your rebuild	%	%	%	%	%

State the tool versions, PDK or FPGA device, corner, supply voltage, clock and whether the numbers are post-synthesis or post-layout in the caption. Measure the change against your own rebuild; the paper's figures are there for reference.

Before you submit, check that:

- Every number in a table names its tool, library or device, corner and clock.
- The base design and yours were run through the same flow with the same settings.
- Power uses switching activity from simulation, and the testbench that produced it is in the appendix.
- Each table says whether it shows post-synthesis or post-layout results.
- Approximate designs report an application-level quality figure, not only error metrics.
- Analog results cover the process corners and a Monte Carlo run, not only the typical corner.

Our **M.Tech thesis format guide** shows where these tables go in the results chapter, and our guide on **how to implement a base paper** covers what to do when your rebuild does not match the paper's numbers.

How can we help with an M.Tech VLSI project?

We check the base paper first: whether its design can be rebuilt with the tools or boards you have, and whether the rebuild and one measured extension fit before your reviews. Then we rebuild it, plan the extension with you, run both designs through the same flow, and support the Stage-I and Stage-II reports, the thesis and, if you need one, a paper and viva preparation.

See how an ECE project runs, track by track, on our **M.Tech projects for ECE** page, which covers VLSI, embedded systems and signal processing. For more ECE papers, see the **IEEE base papers for ECE**; for the M.Tech service across branches, see **M.Tech projects**.

We build, guide, help with the writing and explain every result. The thesis you submit must follow your university's rules and cite the base paper.

More M.Tech domains, each built on verified IEEE base papers: M.Tech projects in embedded systems and IoT, M.Tech projects in machine learning, M.Tech projects in image processing and M.Tech projects in cyber security.

Frequently asked questions

Which is the best M.Tech project in VLSI?

There is no single best one. A good M.Tech VLSI project rebuilds a recent IEEE paper with tools you can use, then makes one change that you measure in area, power, timing or accuracy against your own rebuild. Pick the track your lab supports: FPGA, front-end, back-end or analog. Our guide to selecting a base paper covers the checks.

Can I do an M.Tech VLSI project without Cadence or Synopsys?

Yes. FPGA projects run on the free Vivado ML Standard or Quartus Prime Lite. Chip design projects can use Yosys, OpenROAD, Magic, Xschem and ngspice with the free SkyWater SKY130 PDK, from RTL or schematic to a full layout. Name the tools, the PDK and the corner in your thesis, and run the base design and yours through the same flow.

Which is better for M.Tech: front-end, back-end or FPGA?

It depends on the work you want to do and the tools you have. Front-end means RTL, verification and synthesis. Back-end means placement, routing and timing closure up to a layout. FPGA projects end with a design running on a board. Analog work means circuits at transistor level. Each idea on this page names its track.

What results should an M.Tech VLSI thesis report?

Area, power and timing for the base design and yours, built with the same tools, library and clock, plus utilisation for FPGA designs and quality figures such as PSNR or accuracy where a design is approximate. Say whether each number is post-

synthesis or post-layout. Our M.Tech thesis format guide shows where the results chapter goes.

Do I need an FPGA board for an M.Tech VLSI project?

Only for an FPGA project, where results measured on a board are much stronger than simulation alone. Front-end, back-end, analog and ML for EDA projects need only a computer. If you buy a board, check that the free edition of Vivado or Quartus supports its device, and order it during Stage-I.

Can The Ultimate Project World help with an M.Tech VLSI project?

Yes. Send the DOI of your base paper, the tools your lab has and your stage review dates in the free consultation. We tell you which parts we can take on, such as the rebuild, the extension and its PPA tables, the thesis, an IEEE-format paper and viva preparation. Our M.Tech projects for ECE page explains how it runs.

RELATED SERVICES

From a topic to a project you can present

M.Tech projects

A base paper, one measured extension, a paper and the thesis.

IEEE projects

Start from a published base paper and extend it.

Research papers

A paper in IEEE, Springer or journal format, and a journal shortlist.