AI for Automatic HDL Code Generation from High-Level Descriptions

Presented by:
Md Ruhul Kuddus Saleh

Supervisor: Prof. Dr.–Ing. Ali Hayek

Motivation

- Moore’s Law is slowing → GPUs, FPGAs, ASICs take the spotlight.
- HDL Development (Verilog/VHDL) needs expert, time & effort.
- LLMs offer hope by generating HDL directly from natural language.

High-Level Descriptions for FPGA Design

- Purpose: Bridge algorithm → hardware
- HLS (C/C++): Easy, but less control over memory & timing
- DSLs (Chisel, Bluespec, MyHDL): Hardware-oriented with more control, needs niche skills
- Goal: Efficient algorithm → hardware with minimal RTL coding

Traditional HLS Techniques & Tools

- **HLS Overview: High-level code → RTL, faster design, less manual coding**
- **Core Workflow:**

1. Schedule: Assign ops to clock cycles
2. Allocate: Map resources (ALUs, memory, BRAM)
3. Optimize: Pipelining & unrolling
4. Control: FSMs manage data flow

- **Tools: Xilinx Vivado, Intel HLS, Cadence Stratus, LegUp, ROCCC**
- **Limitations:**

1. Struggle to infer optimal hardware
2. Manual pragmas for performance
3. Inefficient for irregular control & dynamic memory
4. Long synthesis → delayed feedback

AI Techniques for HDL Generation

ML – Performance Prediction

- QoR: area, latency, power
- Fast feedback without full synthesis

DL – Structural Analysis

- GNNs on ASTs & Dataflow Graphs
- Transformers capture long-term dependencies

LLMs – HDL Generation

- NL → SystemVerilog (ChatGPT, CodeLlama, StarCoder)
- Supports code completion & refactoring
- Hallucinations & limited device awareness

Experimental Analysis: Methodology

- **Target Device: Xilinx Artix-7 FPGA.**
- **Tools: Vivado 2025.1.**
- **The Competitors:**

1. Professional "Golden Reference" (Xilinx templates/OpenTitan).
2. AI-Generated Code (ChatGPT-5.2 for memory; Claude Sonnet 4.5 for control logic).

Analysis - Memory Inference (Experiment 1)

Task: Generate a Dual-Port BRAM

Success:

Correctly inferred RAMB36E1 primitive

Failures:

- Syntax incompatibility: produced .v (Verilog-2001) file → Vivado rejected it
- Missing clock-enable signals (ena/enb) → memory always active → power doubled (+97%)

Analysis - Memory Inference (Experiment 1)

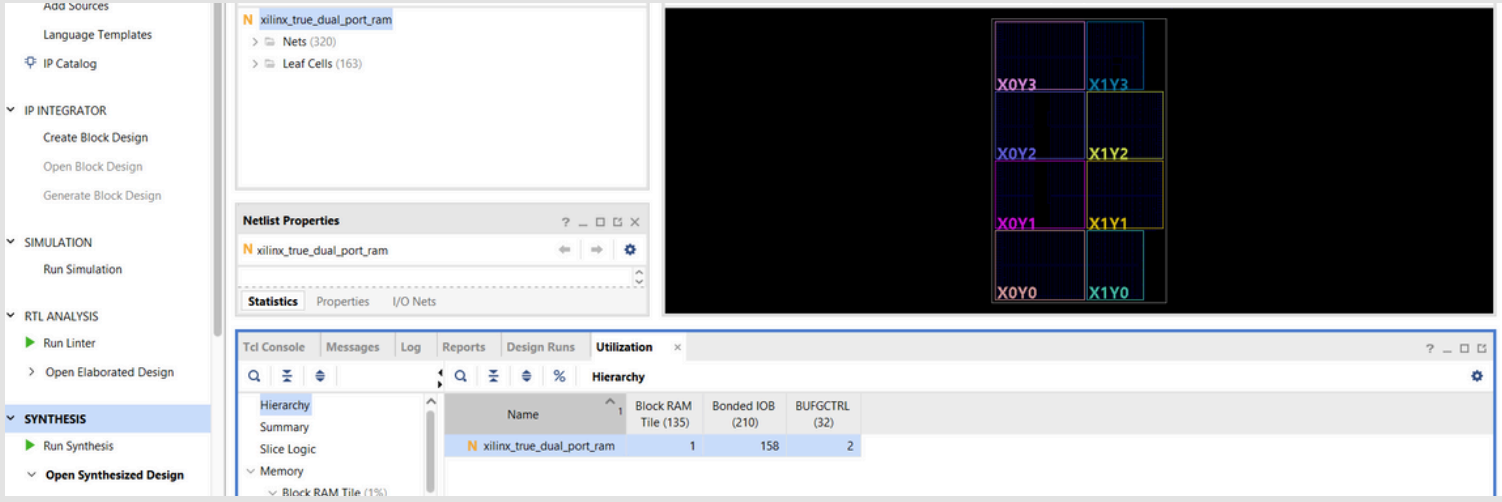
Table 1: BRAM Resource Utilization and Workflow Outcome

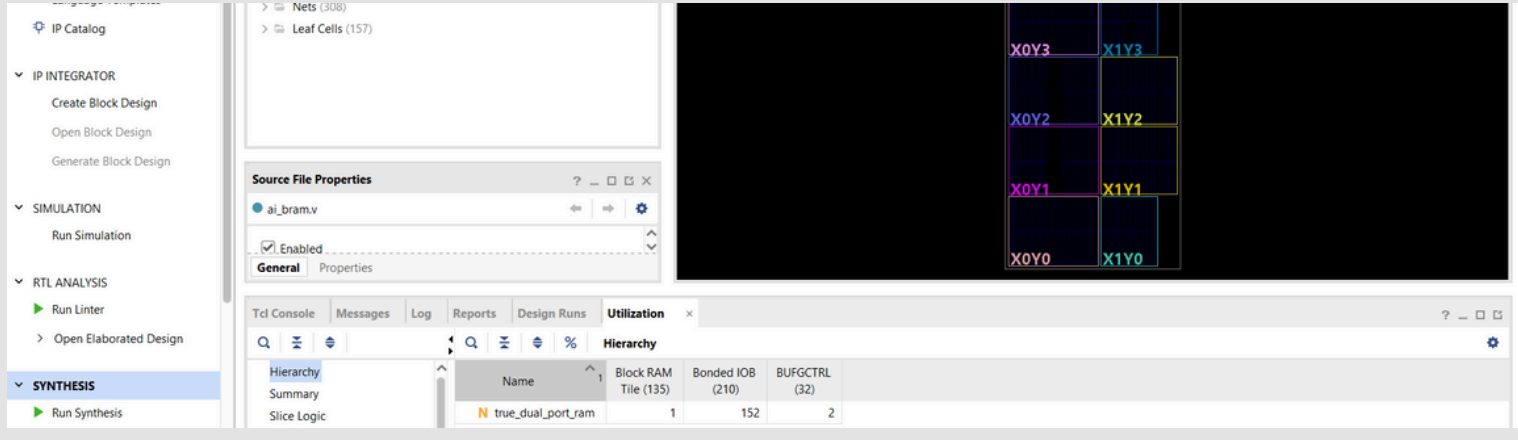
Metric	Professional (Xilinx Template)	AI-Generated (ChatGPT-5.2)	Difference
Block RAM Tiles	1 (1.00%)	1 (1.00%)	0%
Bonded IOB	158	152	−3.8%
Synthesis Status	**Pass**	**Fail**(Syntax Error)	**Workflow Failure**

Table 2: Power Consumption Comparison for BRAM Design

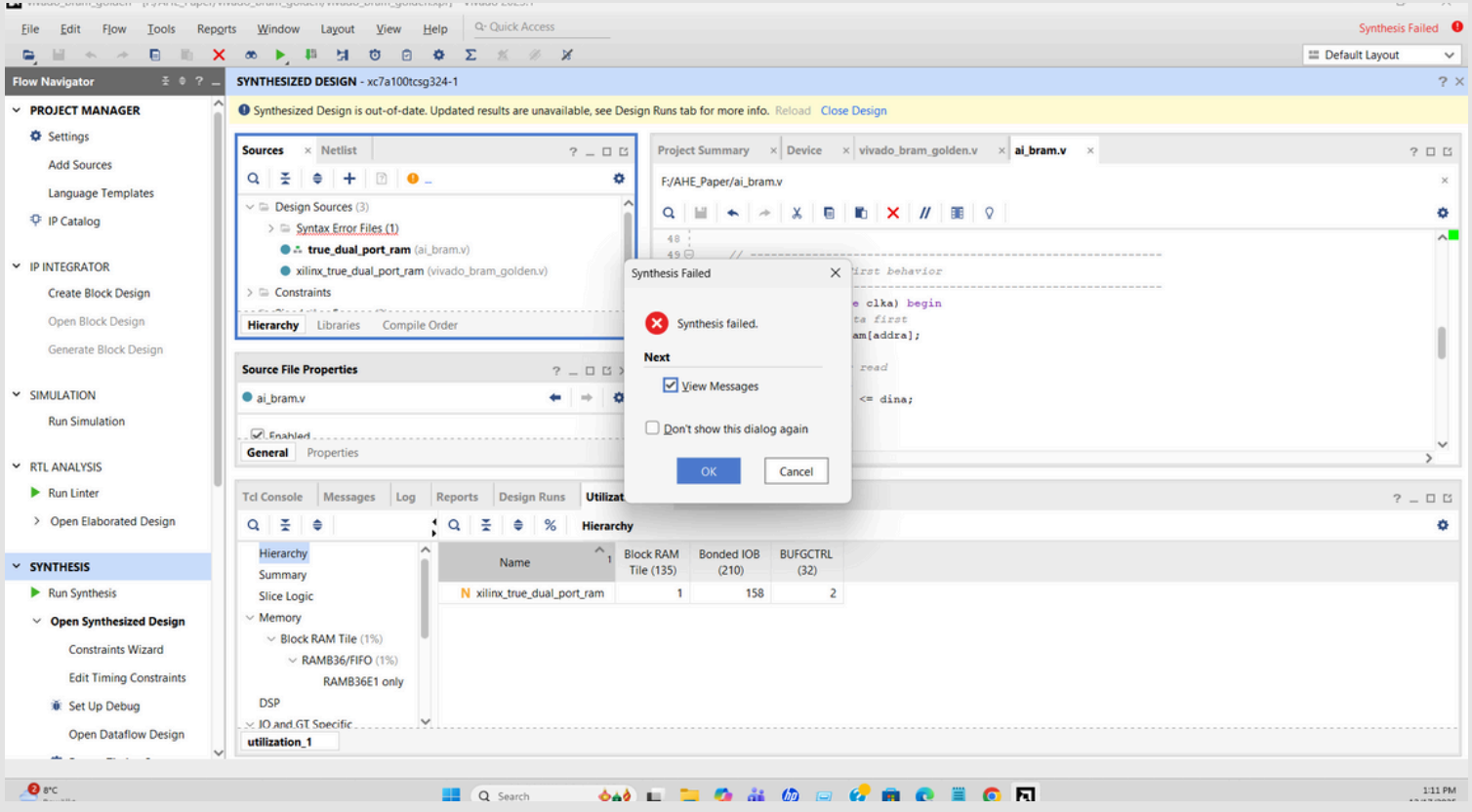
Power Component	Professional	AI-Generated	Increase
BRAM Power	0.160 W	0.321 W	+100% (2x)
I/O Power	30.726 W	61.414 W	+100% (2x)
Total On-Chip Power	**32.562 W**	**64.294 W**	**+97%**

Analysis - Memory Inference (Experiment 1)

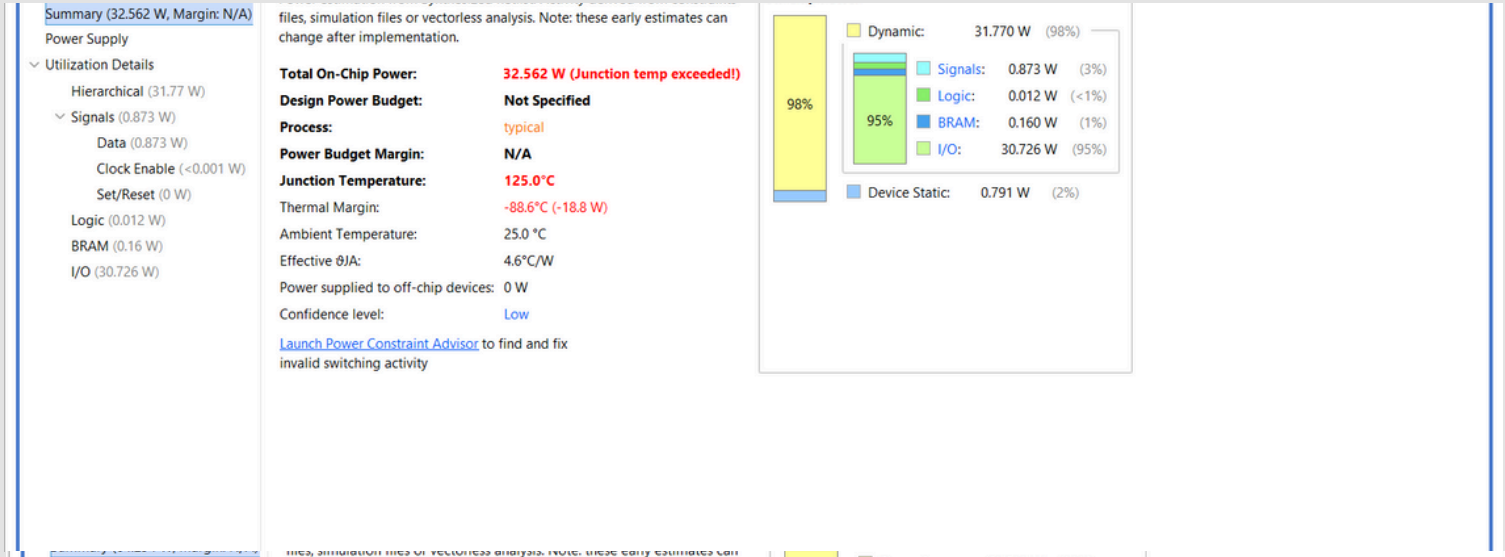


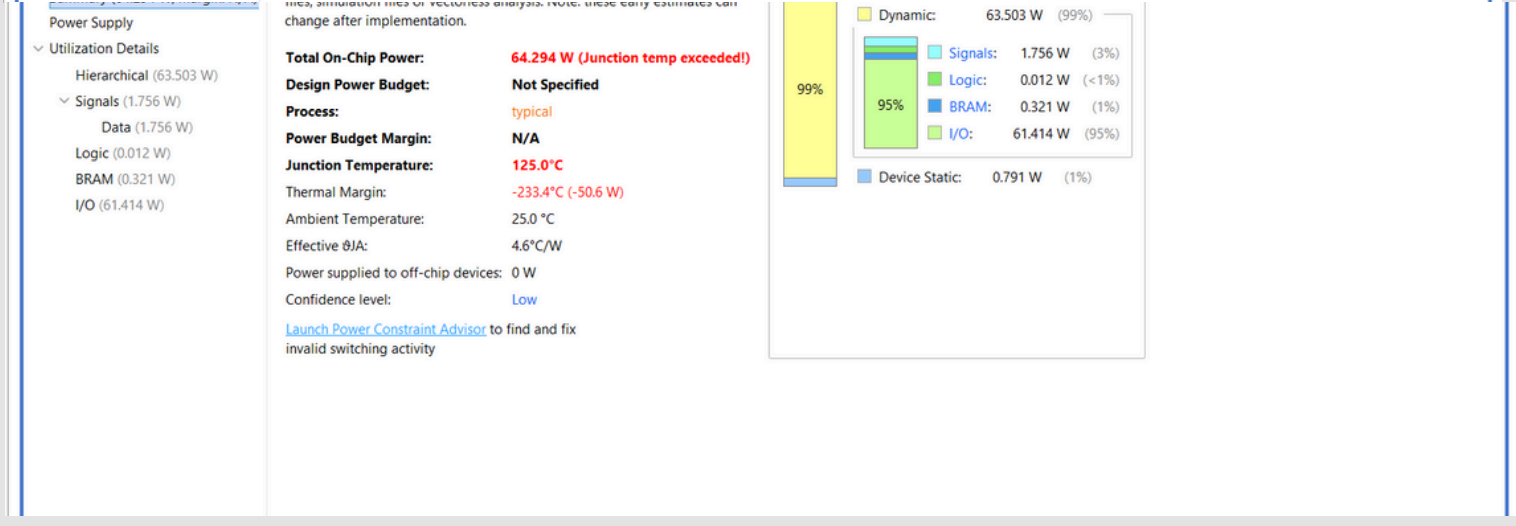


Analysis - Memory Inference (Experiment 1)



Analysis - Memory Inference (Experiment 1)





Analysis - Control Logic (Experiment 2)

Task: Generate GPIO Controller with masked writes

Result:

Physical hazard

Issues:

- IO Usage: 521 Bonded IOBs → 148% of device capacity → unimplementable
- Logic Density: 2.1× more Leaf Cells than professional design
- Software Mindset: AI used software-style loops → massive combinational logic instead of bitwise masking

Analysis - Memory Inference (Experiment 1)

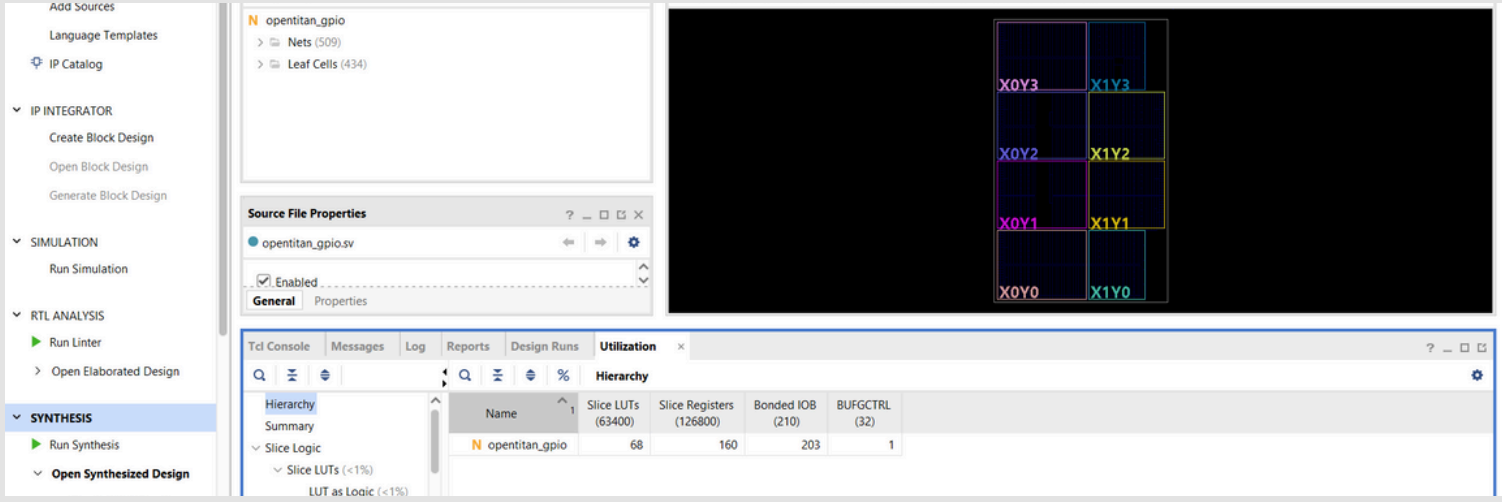
Table 3: Logic Resource and Physical Constraint Utilization

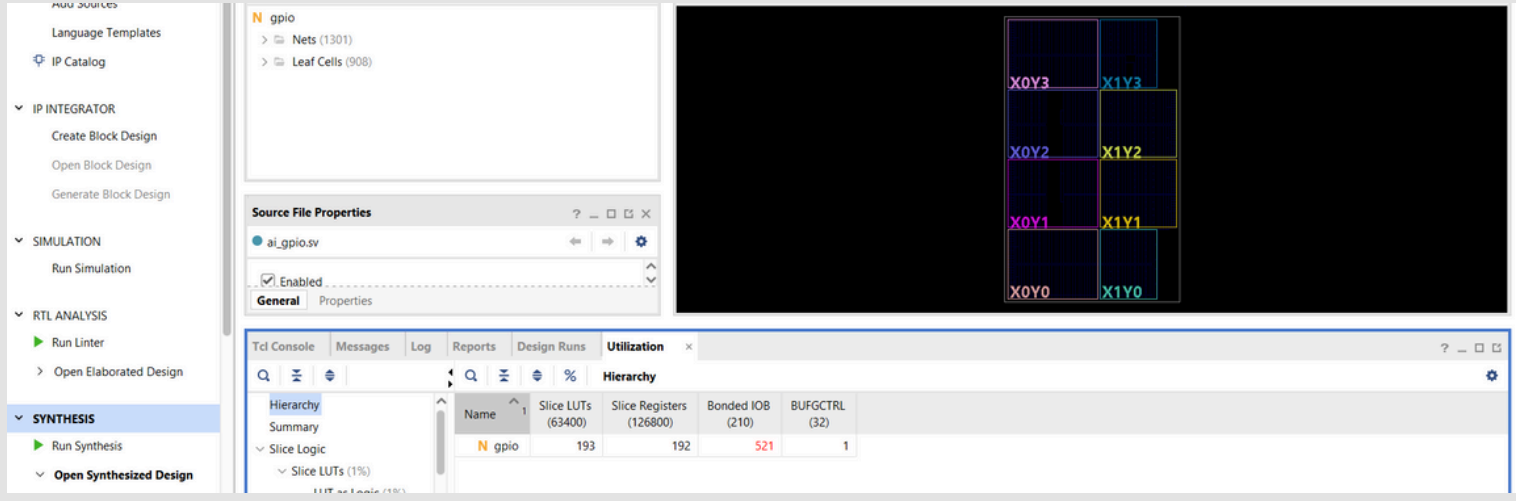
Metric	Professional (OpenTitan)	AI-Generated (Claude 4.5)	Status
Slice LUTs	68	193	2.8× Increase
Slice Registers	160	192	1.2× Increase
Bonded IOB	203	521	**FAILED (148%)**
Max IOB Available	210	210	—
Total Leaf Cells	434	908	**2.1x Area**

Table 4: Power and Thermal Analysis Results

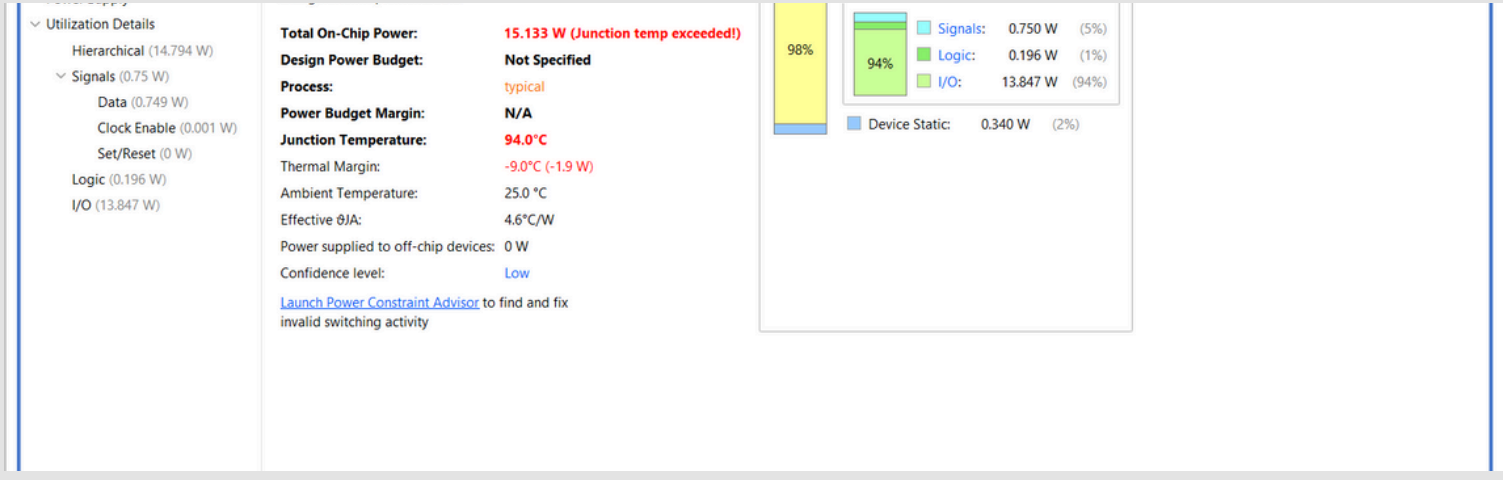
Metric	Professional (OpenTitan)	AI-Generated Code	Difference / Status
Logic Power	0.196 W	0.595 W	+203% (3×)
I/O Power	13.847 W	30.232 W	+118%
Total On-Chip Power	15.133 W	32.829 W	+117%
Junction Temp	**94.0 °C (Safe)**	**125.0 °C (Failed)**	**DESTRUCTIVE**

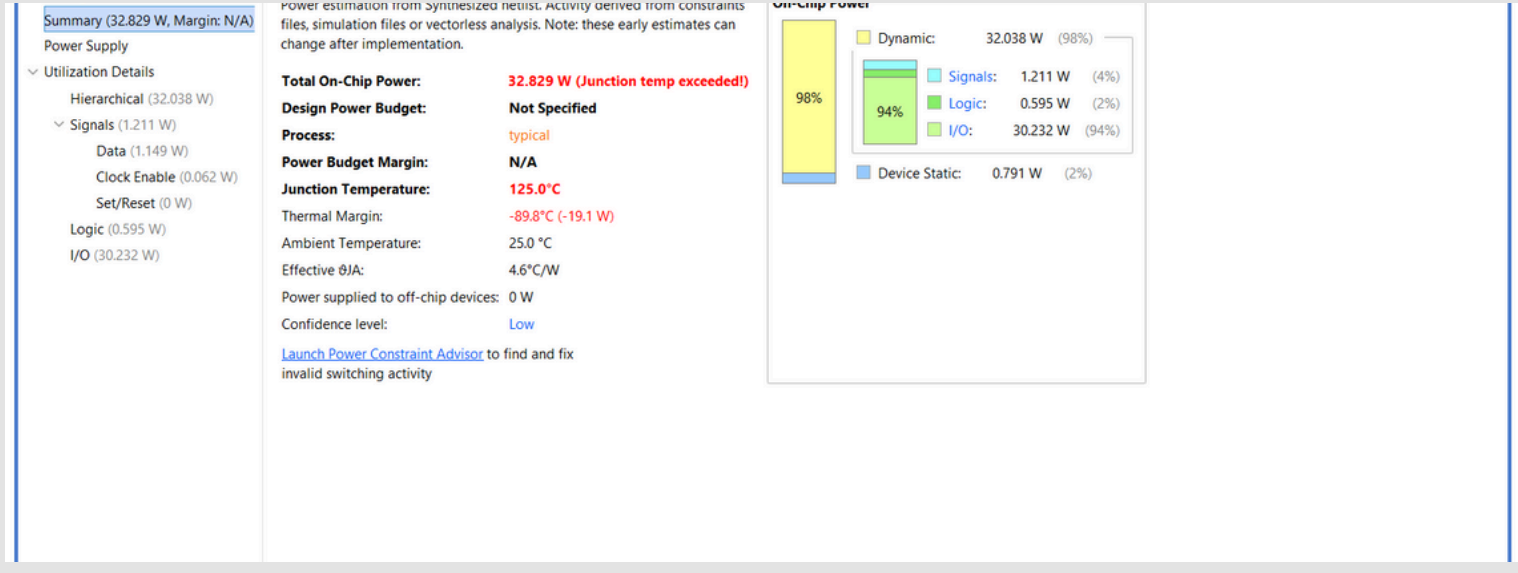
Analysis - Memory Inference (Experiment 1)





Analysis - Memory Inference (Experiment 1)





Analysis - Power & Thermal Failure

The Data:

- **Logic Power: 0.196 W (Pro) vs. 0.595 W (AI) — a 3x increase.**
- **Junction Temp: 94°C (Safe) vs. 125°C (Destructive).**

Critical Finding: AI-generated hardware can reach temperatures that trigger thermal shutdown or permanent silicon damage.

Analysis - Power & Thermal Failure

The Data:

- **Logic Power: 0.196 W (Pro) vs. 0.595 W (AI) — a 3x increase.**
- **Junction Temp: 94°C (Safe) vs. 125°C (Destructive).**

Critical Finding: AI-generated hardware can reach temperatures that trigger thermal shutdown or permanent silicon damage.

Conclusion

- AI is an HDL assistant, not an autonomous designer
- Excels at rapid first drafts
- Falls short at physical implementation & optimization
- Device-Aware Models: Understand FPGA architectures & constraints
- Power-Aware Design: Learn clock gating and hardware-specific power optimizations
- Robust Verification: Human oversight remains essential, especially for safety-critical designs

Thank You for Your Attention