

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)

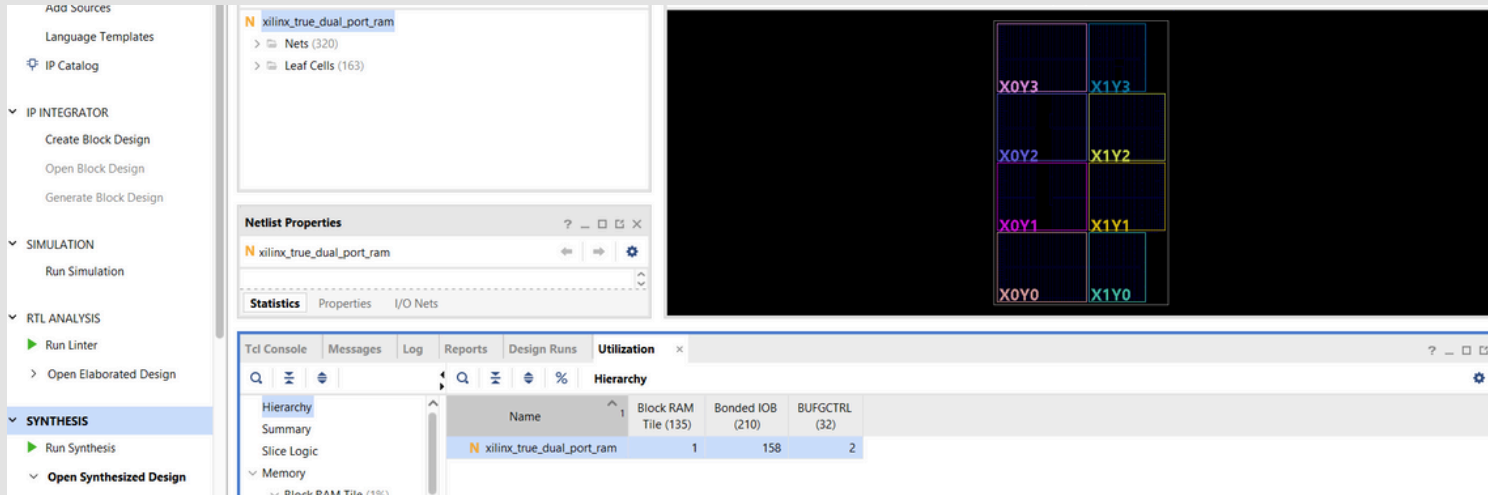
Table 1: BRAM Resource Utilization and Workflow Outcome

| 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% |

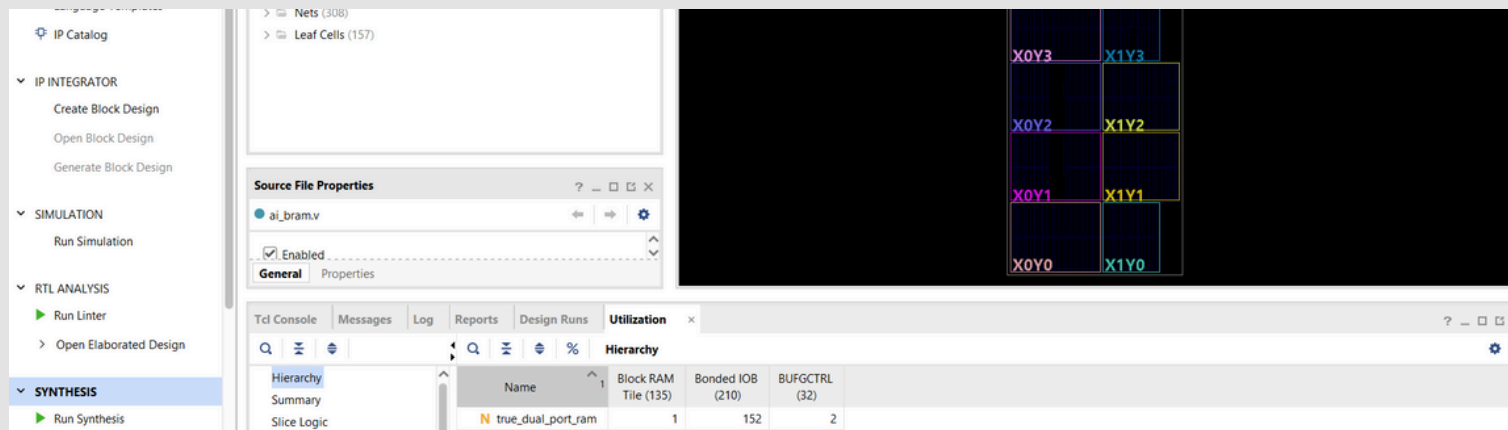
Analysis - Memory Inference (Experiment 1)



This screenshot shows the Vivado IDE interface for a design named `xilinx_true_dual_port_ram`. The left sidebar displays the project hierarchy with `Nets (320)` and `Leaf Cells (163)`. The main window is divided into three panes:

- Netlist Properties:** Shows the selected component `xilinx_true_dual_port_ram` and tabs for `Statistics`, `Properties`, and `I/O Nets`.
- Utilization:** A table showing resource usage for the selected component.
- Hierarchy:** A diagram showing the component's internal structure with memory blocks labeled `X0Y0` through `X1Y3`.

| Name | Block RAM Tile (135) | Bonded IOB (210) | BUFGCTRL (32) |
|--|----------------------|------------------|---------------|
| <code>xilinx_true_dual_port_ram</code> | 1 | 158 | 2 |

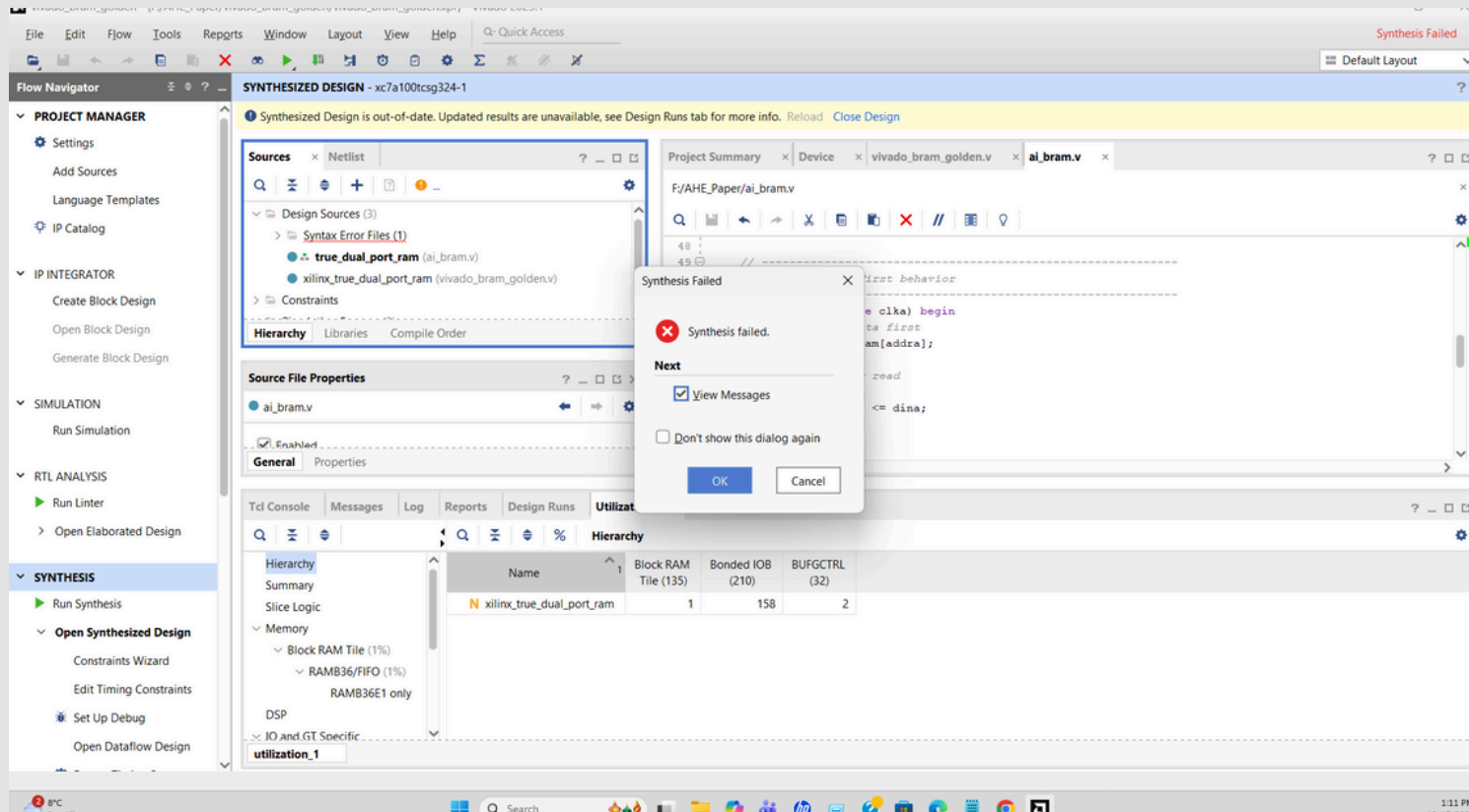


This screenshot shows the Vivado IDE interface for a design named `true_dual_port_ram`. The left sidebar displays the project hierarchy with `Nets (308)` and `Leaf Cells (157)`. The main window is divided into three panes:

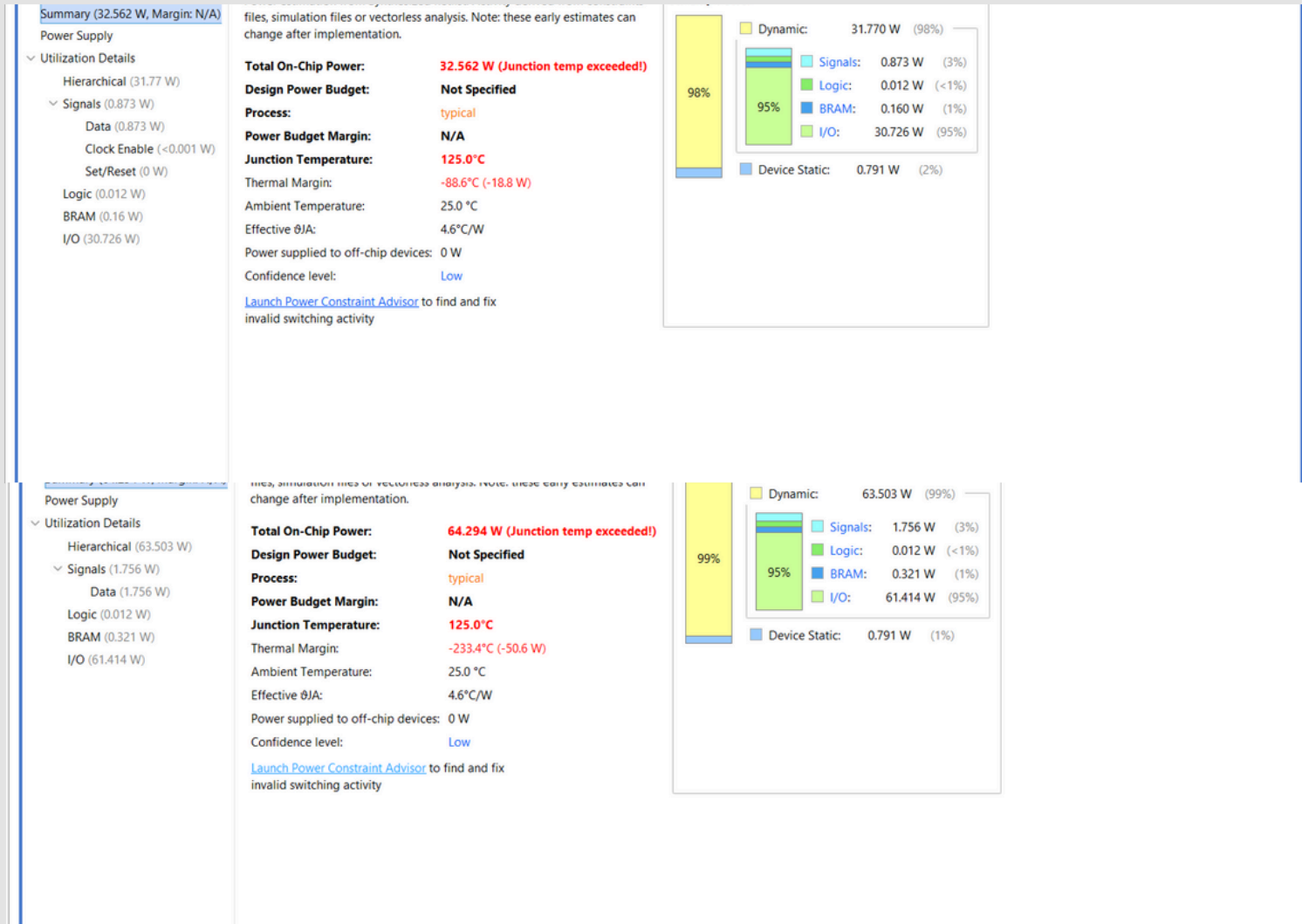
- Source File Properties:** Shows the selected component `ai_bram.v` and tabs for `General` and `Properties`.
- Utilization:** A table showing resource usage for the selected component.
- Hierarchy:** A diagram showing the component's internal structure with memory blocks labeled `X0Y0` through `X1Y3`.

| Name | Block RAM Tile (135) | Bonded IOB (210) | BUFGCTRL (32) |
|---------------------------------|----------------------|------------------|---------------|
| <code>true_dual_port_ram</code> | 1 | 152 | 2 |

Analysis - Memory Inference (Experiment 1)



Analysis - Memory Inference (Experiment 1)



Summary (64.294 W, Margin: N/A)

Power Supply

- Utilization Details
 - Hierarchical (63.503 W)
 - Signals (1.756 W)
 - Data (1.756 W)
 - Clock Enable (<0.001 W)
 - Set/Reset (0 W)
 - Logic (0.012 W)
 - BRAM (0.321 W)
 - I/O (61.414 W)

files, simulation files or vectorless analysis. Note: these early estimates can change after implementation.

Total On-Chip Power: **64.294 W (Junction temp exceeded!)**

Design Power Budget: **Not Specified**

Process: typical

Power Budget Margin: N/A

Junction Temperature: **125.0°C**

Thermal Margin: -233.4°C (-50.6 W)

Ambient Temperature: 25.0 °C

Effective θ_{JA} : 4.6°C/W

Power supplied to off-chip devices: 0 W

Confidence level: Low

[Launch Power Constraint Advisor](#) to find and fix invalid switching activity

99%

Dynamic: 63.503 W (99%)

95%

Device Static: 0.791 W (1%)

| Category | Power (W) | Percentage |
|---------------|-----------|------------|
| Dynamic | 63.503 | 99% |
| Device Static | 0.791 | 1% |
| Signals | 1.756 | 3% |
| Logic | 0.012 | <1% |
| BRAM | 0.321 | 1% |
| I/O | 61.414 | 95% |

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)

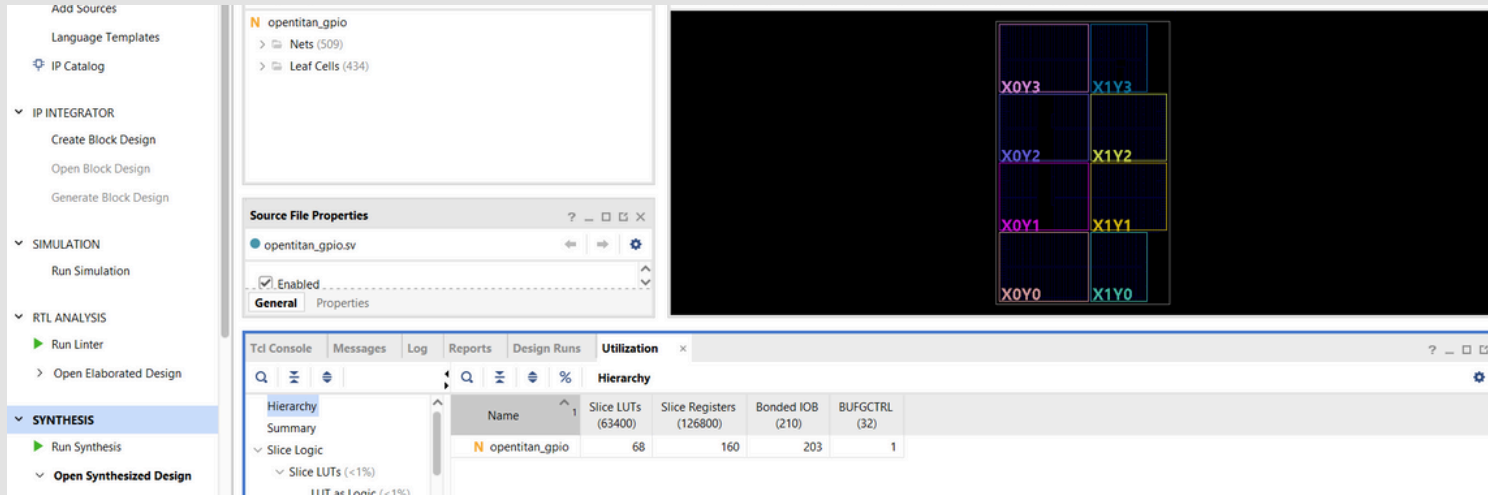
Table 3: Logic Resource and Physical Constraint Utilization

| 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 |

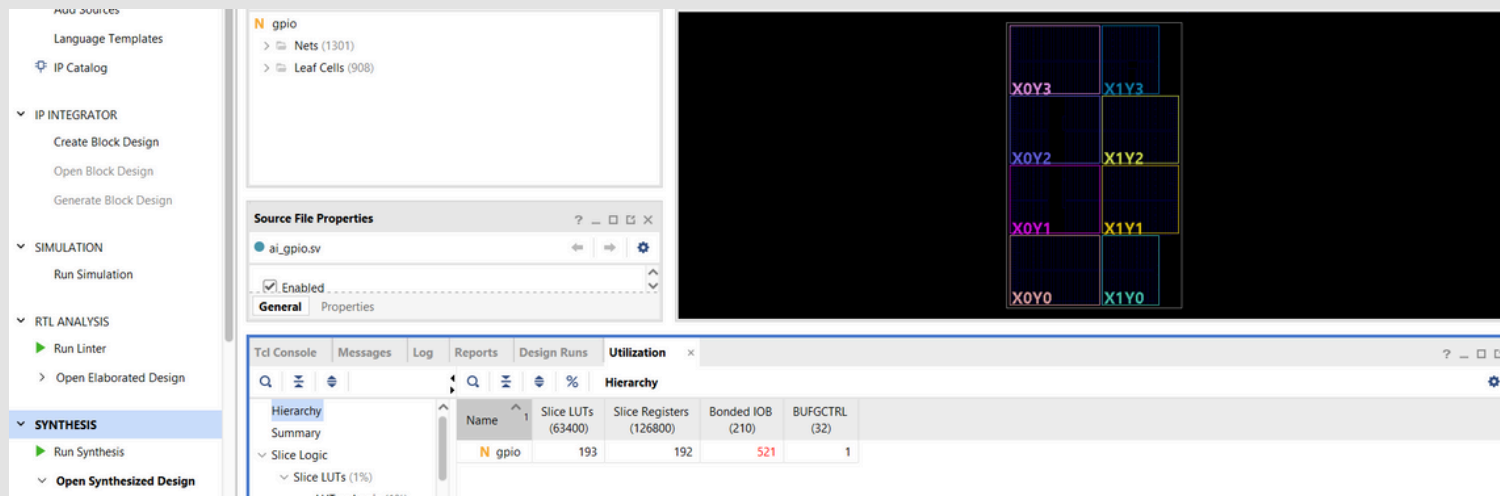
Analysis - Memory Inference (Experiment 1)



The screenshot shows the Vivado IDE interface for the 'opentitan_gpio' project. The left sidebar displays the project hierarchy with 'SYNTHESIS' selected. The main window shows the 'Utilization' report for the 'opentitan_gpio' design.

| Name | Slice LUTs (63400) | Slice Registers (126800) | Bonded IOB (210) | BUFCTRL (32) |
|----------------|--------------------|--------------------------|------------------|--------------|
| opentitan_gpio | 68 | 160 | 203 | 1 |

The top right window displays a logic diagram with a 4x2 grid of cells labeled X0Y3, X1Y3, X0Y2, X1Y2, X0Y1, X1Y1, X0Y0, and X1Y0.

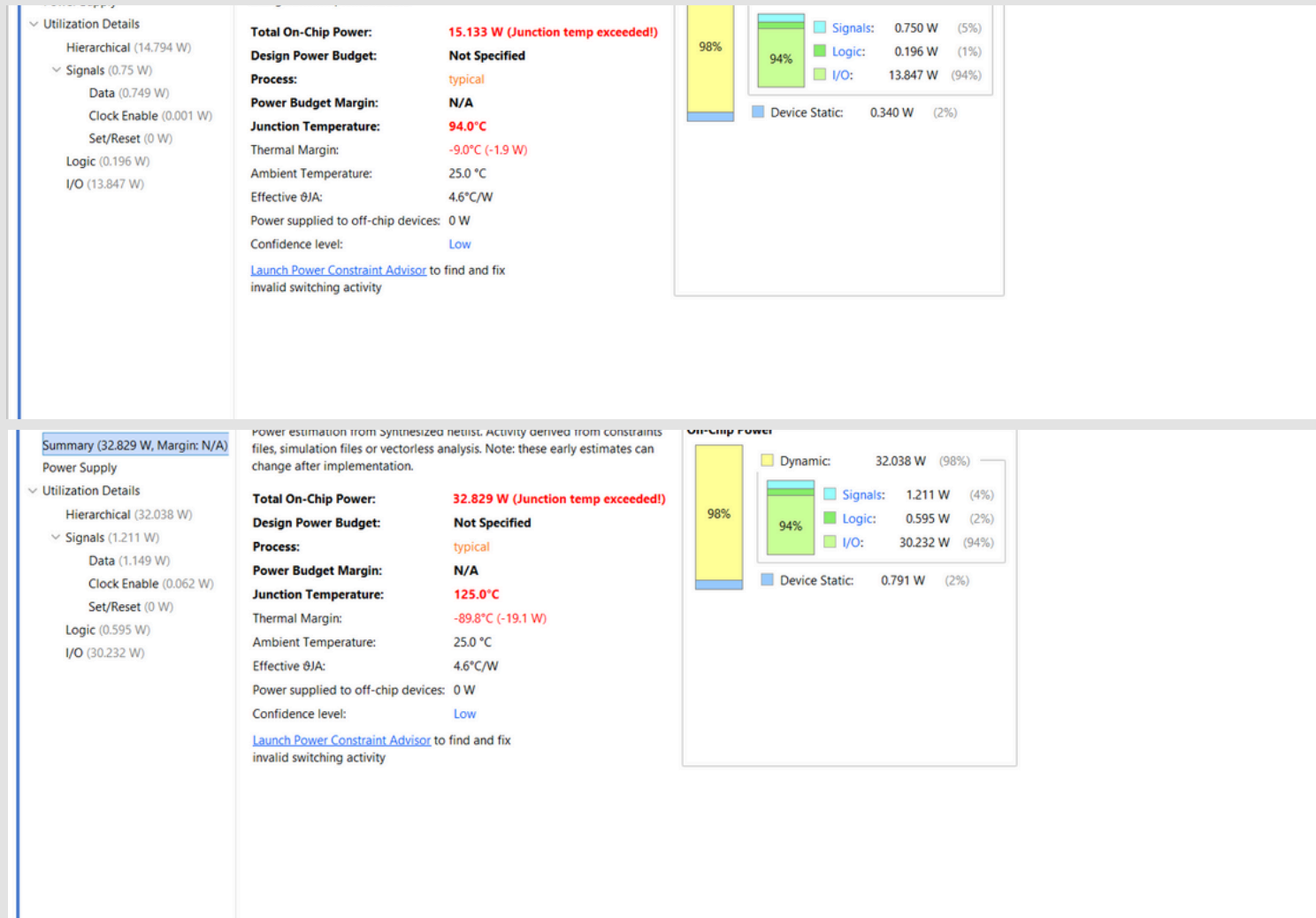


The screenshot shows the Vivado IDE interface for the 'gpio' project. The left sidebar displays the project hierarchy with 'SYNTHESIS' selected. The main window shows the 'Utilization' report for the 'gpio' design.

| Name | Slice LUTs (63400) | Slice Registers (126800) | Bonded IOB (210) | BUFCTRL (32) |
|------|--------------------|--------------------------|------------------|--------------|
| gpio | 193 | 192 | 521 | 1 |

The top right window displays a logic diagram with a 4x2 grid of cells labeled X0Y3, X1Y3, X0Y2, X1Y2, X0Y1, X1Y1, X0Y0, and X1Y0.

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

