Intel® Quartus® Prime Pro Edition User Guide

Power Analysis and Optimization

Updated for Intel® Quartus® Prime Design Suite: **21.4**

Online Version

Send Feedback

UG-20141

ID: 683174

Version: 2022.06.22

Contents

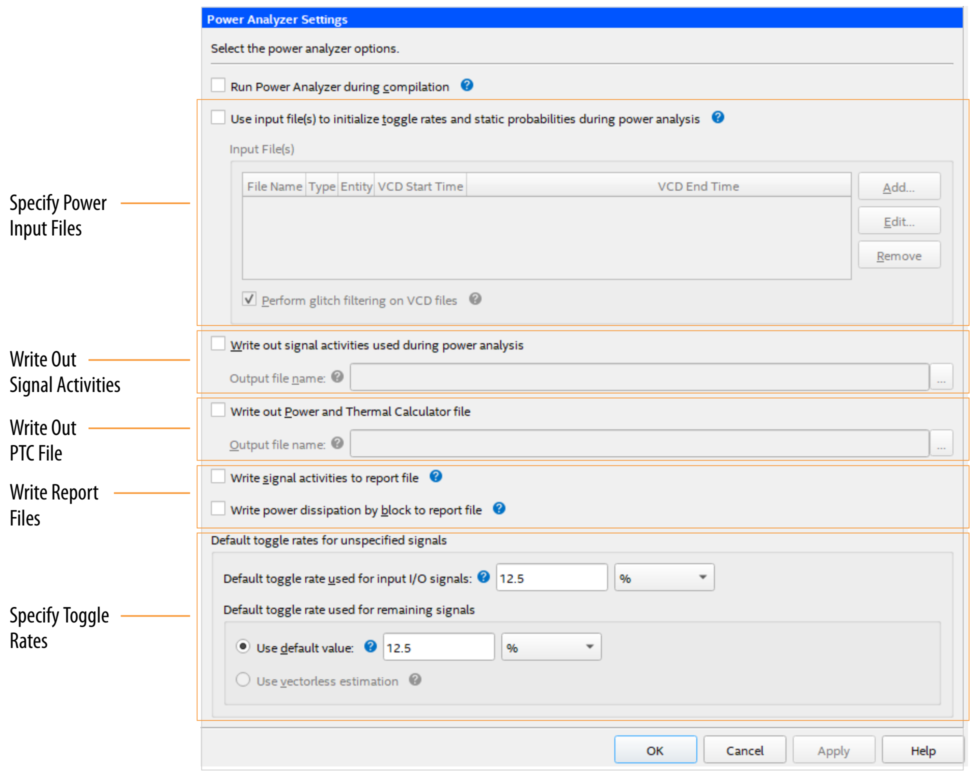
1. Power Analysis

Power consumption is a critical design consideration. When designing a PCB, you must determine the power consumption of the FPGA device to develop an accurate power budget, and to design the power supplies, voltage regulators, heat sink, and cooling system.

The Intel® Quartus® Prime Design Suite provides the Early Power Estimator (EPE) spreadsheet for Intel Arria® 10 devices, and the Intel FPGA Power and Thermal Calculator for Intel Stratix® 10 and Intel Agilex™ devices.

The Power Analyzer helps you to estimate the power consumption of your compiled design, for these three device families.

Figure 1. Power Analyzer Tool



Power Analyzer Settings

Select the power analyzer options.

- [ ] Run Power Analyzer during compilation
- [ ] Use input file(s) to initialize toggle rates and static probabilities during power analysis

Input File(s)

File Name	Type	Entity	VCD Start Time	VCD End Time

- [✓] Perform glitch filtering on VCD files
- [ ] Write out signal activities used during power analysis

Output file name:

- [ ] Write out Power and Thermal Calculator file

Output file name:

- [ ] Write signal activities to report file
- [ ] Write power dissipation by block to report file

Default toggle rates for unspecified signals

Default toggle rate used for input I/O signals: 12.5 %

Default toggle rate used for remaining signals

- ● Use default value: 12.5 %
- ○ Use vectorless estimation

Power estimation and analysis allows you to confirm that your design does not exceed thermal or power supply requirements throughout the design process:

- **Thermal**—Thermal power is the power that dissipates as heat from the FPGA. Devices use a heatsink or fan to act as a cooling solution. This cooling solution must be sufficient to dissipate the heat that the device generates. Additionally, the computed junction temperature must fall within normal device specifications.
- **Power supply**—Power supply is the power that the device needs to operate. Power supplies must provide adequate current to support device operation.

Note: Do not use the results of the Power Analyzer as design specifications. You must also verify the actual power during device operation to account for actual environmental operating conditions.

Related Information

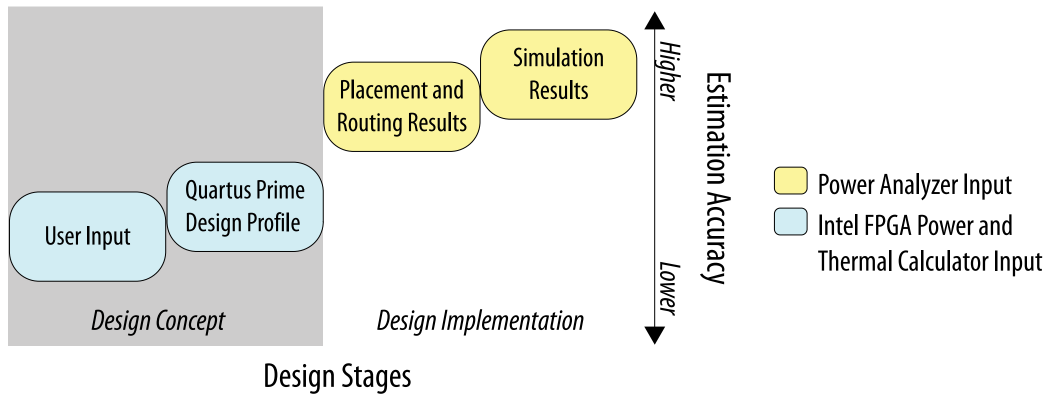
- Intel FPGA Power and Thermal Calculator (PTC) User Guide
- Power Analyzer Support Resources

1.1. Power Analysis Tools

The Intel Quartus Prime Design Suite provides tools to analyze the power consumption of your FPGA design at different stages of the design process.

- Intel FPGA Power and Thermal Calculator (PTC)—estimates power supply and system thermal requirements before compiling the design, or anytime during the design phase. Supports Intel Stratix 10 and Intel Agilex devices.
- Intel Quartus Prime Power Analyzer (QPA)—estimates power consumption for a post-fit design, allowing you establish guidelines for the power budget.
- Early Power Estimator (EPE) spreadsheet—estimates power consumption for power supply planning before compiling the design. Supports Intel Arria 10 and Intel Stratix 10 devices. (For versions of the Intel Quartus Prime software later than version 19.4, Intel Stratix 10 devices are supported in the Intel FPGA Power and Thermal Calculator.)

Figure 2. Estimation Accuracy for Different Inputs and Power Analysis Tools



The accuracy of the power model is determined on a per-power-rail basis for the Intel Quartus Prime Power Analyzer.

- For most Intel Stratix 10 designs, the Intel Quartus Prime Power Analyzer has the following accuracy, assuming final power models: Within 10% of silicon for the majority of power rails with higher power, assuming accurate inputs and toggle rates.
- For most Intel Agilex designs, the Intel Quartus Prime Power Analyzer has the following accuracy, assuming final power models: Within 10% of silicon for all power rails, assuming accurate inputs and toggle rates.

Table 1. Comparison of EPE/Intel FPGA PTC and Intel Quartus Prime Power Analyzer Capabilities

Characteristic	EPE / PTC	Intel Quartus Prime Power Analyzer
When to use	Any time *Note:* For post-fit power analysis, you get better results with the Intel Quartus Prime Power Analyzer.	Post-fit
Software requirements	EPE: Spreadsheet program. Intel FPGA PTC: Integrated into the Intel Quartus Prime software, and is also available as a standalone tool.	The Intel Quartus Prime software
Accuracy	Medium	Medium to very high
Data inputs	• Resource usage estimates • Clock requirements • Environmental conditions • Toggle rate	• Post-fit design • Clock requirements • Signal activity defaults • Environmental conditions • Register transfer level (RTL) simulation results (optional) • Post-fit simulation results (optional) • Signal activities per node or entity (optional)
Data outputs *Note:* The EPE and Power Analyzer outputs vary by device family.	• Total thermal power dissipation • Thermal static power • Thermal dynamic power • Off-chip power dissipation • Current drawn from voltage supplies	• Total thermal power • Thermal static power • Thermal dynamic power • Thermal I/O power • Thermal power by design hierarchy • Thermal power by block type • Thermal power dissipation by clock domain • Device supply currents
Estimation of transceiver power for dynamic reconfiguration features	Includes an estimation of the incremental power consumption by these features.	Not included

Note: The Intel Quartus Prime Power Analyzer does not support power analysis of the following Intel FPGA IP:

- Intel Stratix 10 HBM2 IP
- Intel Stratix 10 HPS IP
- Intel Arria 10 HPS IP

In versions of the Intel Quartus Prime software later than 19.4, you can obtain power estimations for the Intel Stratix 10 HBM2 IP and Intel Stratix 10 HPS IP using the Intel FPGA Power and Thermal Calculator (PTC).

For power estimation of Intel Arria 10 HPS IP, and for power estimation in the Intel Quartus Prime software version 19.4 or earlier, you can can obtain power estimations using the Early Power Estimator spreadsheet (EPE).

1.2. Running the Power Analyzer

Before running the Power Analyzer you must run full compilation of your design to generate the post-fit netlist. In addition, you must either provide timing assignments for all clocks in the design, or specify signal activity data for power analysis. You must specify the I/O standard on each device input and output, and the board trace model on each output in the design.

To run the Power Analyzer:

1. To specify device power characteristics, operating voltage, and temperature conditions for power analysis, click **Assignments ➤ Settings ➤ Operating Settings and Conditions**, as Settings for Power Analysis on page 8 describes.
2. To run full compilation of your design, click **Processing ➤ Start Compilation**.
3. Click **Assignments ➤ Settings ➤ Power Analyzer**.
4. Specify the source of signal activity data, as Generating Signal Activity Data for Power Analysis on page 12 describes.
5. To generate a Signal Activity (`.saf`) file during analysis, turn on **Write out signal activities used during power analysis**, and specify the file name.
6. To direct the Power Analyzer to generate an Early Power Estimation file, turn on **Write out Early Power Estimation file**, and specify the file name. The Early Power Estimation file summarizes the resource utilization and allows you to perform what-if analyses in EPE.
7. Specify the **Default toggle rates for unspecified signals**, as Specifying the Default Toggle Rate on page 17 describes.
8. To specify temperature range and cooling options, click **Cooling Solution and Temperature**.
9. To run full compilation of your design, click **Processing ➤ Start ➤ Start Power Analyzer**.
10. When power analysis is complete, click **Report** to open the Power Analyzer reports that Viewing Power Analysis Reports on page 18 describes.

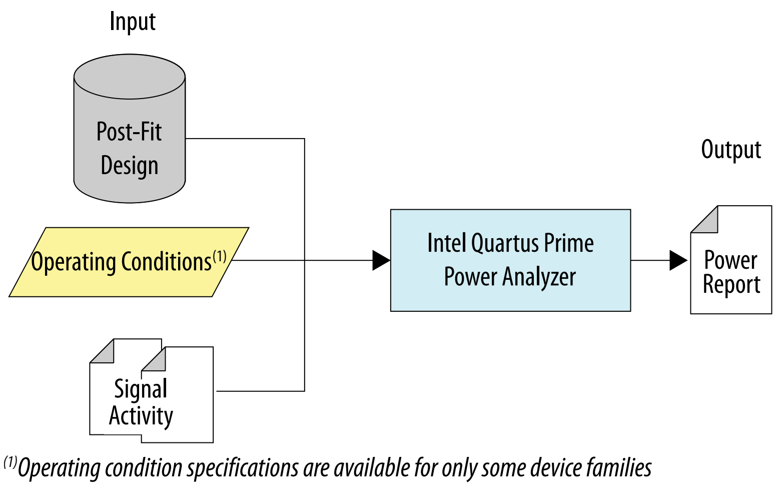
Related Information

Specifying Power Analyzer Input on page 8

1.3. Specifying Power Analyzer Input

The Power Analyzer accuracy is driven by design factors, operating conditions, and signal activity data that affect power consumption. The following figure shows how the Power Analyzer interprets these inputs and generates results in the Power Analysis report:

Figure 3. Power Analyzer High-Level Flow


Input
Post-Fit Design
Operating Conditions(1)
Signal Activity
Intel Quartus Prime Power Analyzer
Output
Power Report
(1)Operating condition specifications are available for only some device families

To obtain accurate I/O power estimates, the Power Analyzer requires full compilation of your design, in addition to specifying the following settings:

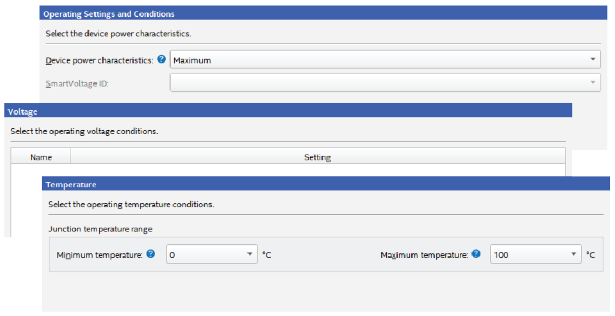
- The electrical standard on each I/O cell.
- The board trace model on each I/O standard in the design.
- Timing assignments for all the clocks in your design, or use a simulation-based flow to generate activity data.

Note: For accurate results, ensure that any .VCD file used with the Power Analyzer is the result of gate-level simulation.

1.3.1. Settings for Power Analysis

You can specify device power characteristics, operating voltage conditions, operating temperature conditions, Power Analyzer settings and thermal settings, in the **Operating Settings and Conditions**, **Power Analyzer Settings**, and **Thermal** pages of the **Settings** dialog box.

Figure 4. Operating Settings and Conditions



Operating Settings and Conditions

Select the device power characteristics.

Device power characteristics: Maximum

SmartVoltage ID:

Voltage

Select the operating voltage conditions.

Name	Setting

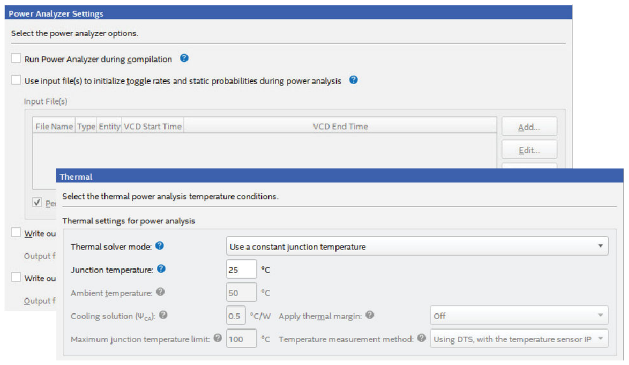
Temperature

Select the operating temperature conditions.

Junction temperature range

Minimum temperature: 0 °C Maximum temperature: 100 °C

Figure 5. Power Analyzer and Thermal Settings



Power Analyzer Settings

Select the power analyzer options.

- [ ] Run Power Analyzer during compilation
- [ ] Use input file(s) to initialize toggle rates and static probabilities during power analysis

Input File(s)

File Name	Type	Entity	VCD Start Time	VCD End Time

Thermal

Select the thermal power analysis temperature conditions.

Thermal settings for power analysis

Thermal solver mode: Use a constant junction temperature

Junction temperature: 25 °C

Ambient temperature: 50 °C

Cooling solution (Ψ_{CA}): 0.5 °C/W Apply thermal margin: Off

Maximum junction temperature limit: 100 °C Temperature measurement method: Using DTS, with the temperature sensor IP

The Power Analyzer reads the following settings to determine the operating conditions for power analysis:

Table 2. Device Operating Condition Settings

Option	Settings
Device power characteristics	• **Typical**—specifies average power consumed by typical silicon at nominal operating conditions. • **Maximum**—specifies maximum power consumed by worst-case device.
Voltage tab	Specifies the operating voltage conditions for each power rail in the device, and the supply voltages for power rails with selectable supply voltages.
Temperature tab	Specifies the minimum and maximum junction temperature range.

Table 3. Power Analyzer and Thermal Settings

Option	Settings
Power Analyzer Settings	Specifies the Power Analyzer options, including: • **Run Power Analyzer during compilation**—Check this box to turn on power analysis during compilation. • **Use input file(s) to initialize toggle rates and static probabilities during power analysis**—Check this box to use Signal Activity Files or VCD files to initialize toggle rates and static probabilities for power estimation. • **Write out signal activities used during power analysis**—Check this box to write the toggle rates and static probabilities used during power estimation to a file.

continued...

Option	Settings
	• **Write out Power and Thermal Calculator file**—Check this box to have the Power Analyzer export a design summary that you can import into the Intel FPGA Power and Thermal Calculator. • **Write signal activities to report file**—Check this box to have the Power Analyzer write a report file containing the signal activities used during power analysis. • **Write power dissipation by block to report file**—Check this box to have the Power Analyzer report the thermal power dissipation calculated during power analysis, in the **Thermal Power Dissipation by Block** report panel. • **Default toggle rate used for input I/O signals**—Specify a default toggle rate for use on input I/O pins during power estimation. Can be expressed as a percentage or in transitions/second. • **Default toggle rate used for remaining signals** — **Use default value**—Specify a default toggle rate for use during power estimation on all nodes except I/O pins. This value is used only if no toggle rate is specified through a Signal Activity File, VCD file, or user assignment. Can be expressed as a percentage or in transitions/second. — **Use vectorless estimation**—Turn on this control to use vectorless estimation to fill in undefined toggle rates and static probabilities. If this option is not available, the device family does not support vectorless estimation.
Thermal Settings	Specifies the thermal power analysis temperature conditions, including: • **Thermal Solver Mode**—Select the thermal solver mode to use during power estimation. • **Junction temperature**—Specifies the junction temperature, in °C, used during power estimation. • **Ambient temperature**—Specifies the ambient temperature, in °C, used during power estimation. • **Cooling solution**—Specifies the cooling solution case-to-ambient thermal resistance, in °C per watt. • **Maximum junction temperature limit**—Specifies the maximum junction temperature limit that no part of any die in the package should exceed. • **Apply additional margin**—Specifies, as a percentage, the amount of additional margin to apply to detailed thermal analysis results. Valid values are 0–25%. The default value is 0%. The recommended margin for Intel Agilex devices is 10%, and for Intel Stratix 10 devices, 25%. *Note:* For a design compiled in an earlier version of the Intel Quartus Prime software with the **Apply Recommended Margin** parameter set to *Yes*, the current version of Power Analyzer interprets this as an **Apply Additional Margin** setting of 25%. • **Temperature measurement method**—Select the method to use for reporting temperature sensors for thermal analysis.

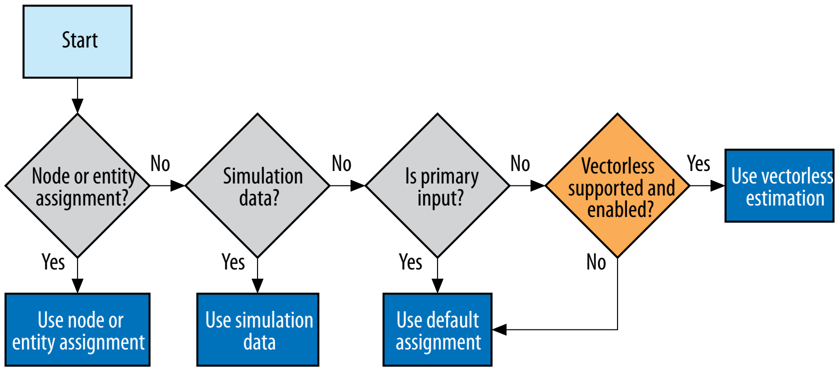
1.3.2. Specifying Signal Activity Data

The accuracy of the power estimation depends on how representative signal activity data is during power analysis. The Power Analyzer allows you to specify signal activity data from the following sources:

- `.vcd` files generated by simulation of the gate-level design netlist, by supported third-party simulators
- User-entered node, entity, and clock assignments
- User-entered default toggle rate assignment
- Vectorless estimation (selected devices)

You can mix and match the signal activity data sources on a signal-by-signal basis. The following figure shows the priority scheme applied to each signal.

Figure 6. Signal Activity Data Source Priority Scheme

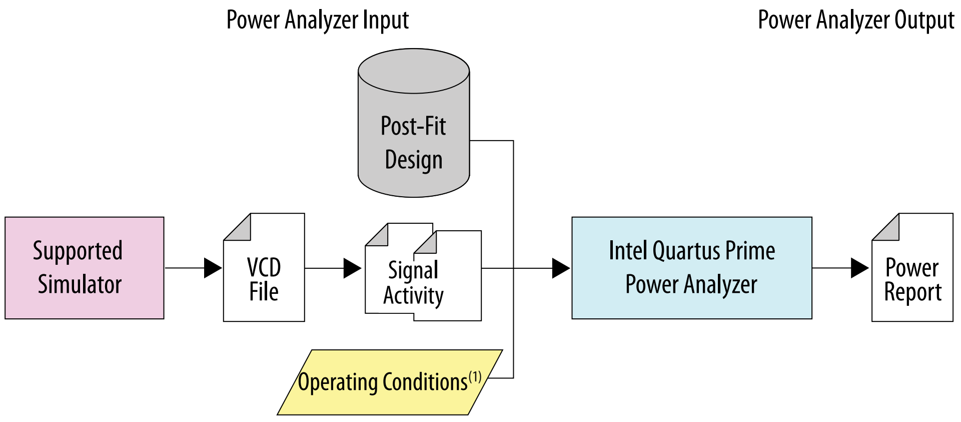

Start
Node or entity assignment?
No
Yes
Use node or entity assignment
Simulation data?
No
Yes
Use simulation data
Is primary input?
No
Yes
Use default assignment
Vectorless supported and enabled?
Yes
No
Use vectorless estimation

1.3.2.1. Using Simulation Signal Activity Data in Power Analysis

You can specify a Verilog Value Change Dump File (`.vcd`) generated by simulating a placed and routed gate-level netlist in a supported[1] simulator as the source of signal activity data for power analysis.

Third-party simulators can output a `.vcd` that contains signal activity and static probability information that inform the power analysis. The generated `.vcd` includes all of the routing resources and the exact logic array resource usage.

Figure 7. Using Simulation Signal Activity Data in Power Analysis



To improve accuracy of power analysis, you can generate a Standard Delay Output (`.sdo`) file that includes back-annotated delay estimates of the instances of core atoms for ModelSim simulation. ModelSim simulation can then output a more accurate `.vcd` for use as power analysis input. You must run the **Fitter (Finalize)** command before generating the `.sdo`. **Note:** To improve accuracy of power analysis, the Intel Quartus Prime EDA Netlist writer can generate a Standard Delay Output (`.sdo`) file that includes back-annotation of delays for a design's netlist for use during

[1] ModelSim*, ModelSim - Intel FPGA Edition, QuestaSim, Active-HDL, NCSim, VCS*, VCS MX, Riviera-PRO*

simulation in ModelSim. Although the `.sdo` only contains delay estimates and imprecise timing information, including the `.sdo` in simulation results in a more accurate output `.vcd` for power analysis.

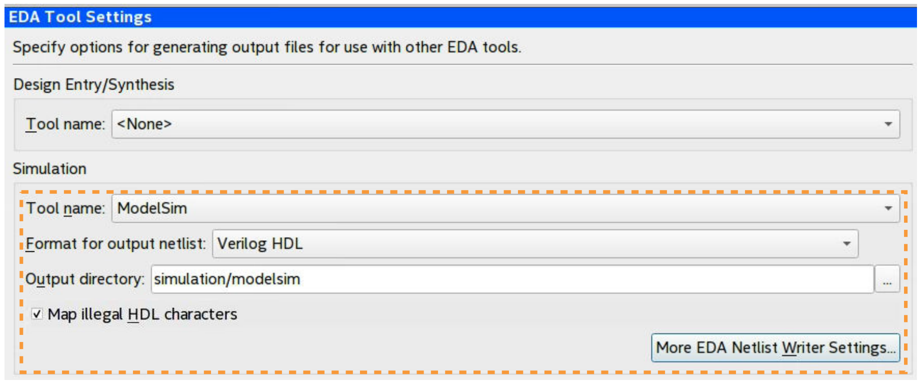
Note: The EDA Netlist Writer currently supports `.sdo` file generation only for Verilog `.vo` simulation in the ModelSim simulator (not ModelSim - Intel FPGA Edition) for Intel Stratix 10 designs. The EDA Netlist Writer does not currently support `.sdo` file generation for any other simulator or device family.

1.3.2.1.1. Generating Signal Activity Data for Power Analysis

Follow these steps to generate and use simulation signal activity data for power analysis:

1. To run full compilation on your design, click **Processing ➤ Start Compilation**.
2. To specify settings for output of simulation files, click **Assignments ➤ Settings ➤ EDA Tool Settings ➤ Simulation**. Select your simulator in **Tool name** and the **Format for output netlist** and **Output directory**.
3. Turn on **Map illegal HDL characters**. This setting directs the EDA Netlist Writer to map illegal characters for VHDL or Verilog HDL, and results in more accurate data for power analysis.

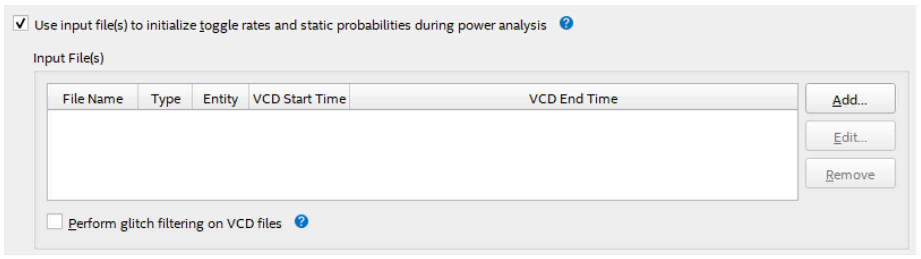
Figure 8. EDA Tool Settings for Simulation



EDA Tool Settings
Specify options for generating output files for use with other EDA tools.
Design Entry/Synthesis
Tool name: <None>
Simulation
Tool name: ModelSim
Format for output netlist: Verilog HDL
Output directory: simulation/modelsim
☑ Map illegal HDL characters
More EDA Netlist Writer Settings...

4. For Intel Stratix 10 designs, to generate a Standard Delay Output (`.sdo`) file that includes back-annotation of delays for power analysis, refer to Generating Standard Delay Output for Power Analysis on page 13.
5. In the Intel Quartus Prime software, click **Assignments ➤ Settings ➤ Power Analyzer Settings**.
6. Under **Input file**, turn on **Use input files to initialize toggle rates and static probabilities during power analysis**.

Figure 9. Specifying Power Analysis Input Files

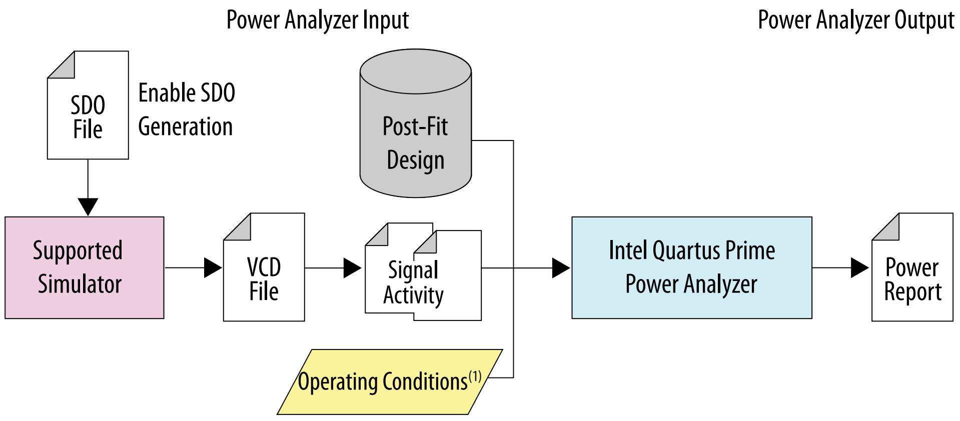

☑ Use input file(s) to initialize toggle rates and static probabilities during power analysis
Input File(s)
File Name
Type
Entity
VCD Start Time
VCD End Time
Add...
Edit...
Remove
☐ Perform glitch filtering on VCD files

7. To specify a `.vcd` for power analysis, click **Add** and specify the **File name**, **Entity**, and **Simulation period** for the `.vcd`, and click **OK**.
8. To enable glitch filtering during power analysis with the `.vcd` you generate, turn on **Perform glitch filtering on VCD files**.
9. To run the power analysis, click **Start** on the **Power Analysis** step in the Compilation Dashboard. View the toggle rates in the power analysis results.

1.3.2.1.2. Generating Standard Delay Output for Power Analysis

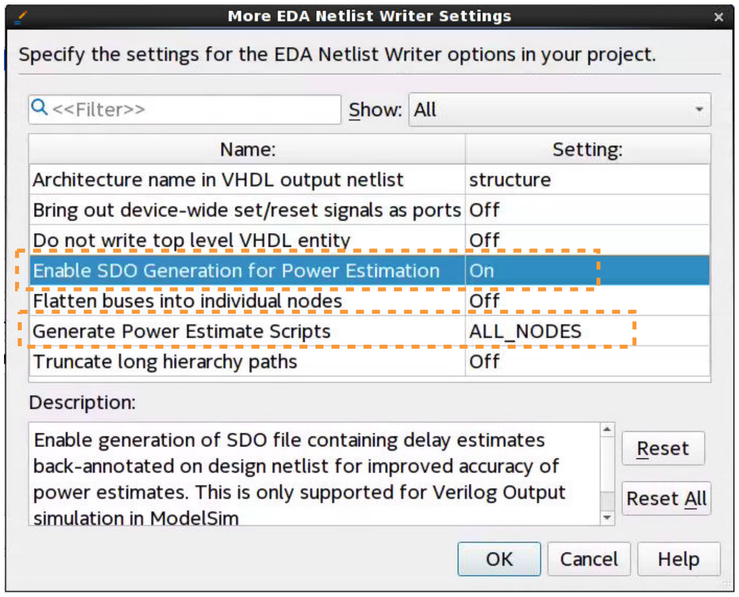
To improve accuracy of power analysis, you can generate a Standard Delay Output (`.sdo`) file that includes back-annotated delay estimates for ModelSim simulation. ModelSim simulation can then output a more accurate `.vcd` for use as power analysis input. You must run **Fitter (Finalize)** before generating the `.sdo`.

Figure 10. Using an SDO in Power Analysis



1. Click **Assignments ➤ Settings ➤ EDA Tool Settings ➤ Simulation**. In **Tool name** select **ModelSim** and **Verilog** for **Format for output netlist**.
2. Click **More EDA Netlist Writer Settings**. Set **Enable SDO Generation for Power Estimation** to **On**. Set **Generate Power Estimate Scripts** to **ALL_NODES**.

Figure 11. More EDA Netlist Writer Settings



Specify the settings for the EDA Netlist Writer options in your project.

Name:	Setting:
Architecture name in VHDL output netlist	structure
Bring out device-wide set/reset signals as ports	Off
Do not write top level VHDL entity	Off
Enable SDO Generation for Power Estimation	On
Flatten buses into individual nodes	Off
Generate Power Estimate Scripts	ALL_NODES
Truncate long hierarchy paths	Off

Description:

Enable generation of SDO file containing delay estimates back-annotated on design netlist for improved accuracy of power estimates. This is only supported for Verilog Output simulation in ModelSim

3. To run the Fitter, click **Processing ➤ Start ➤ Start Fitter (Finalize)**.
4. Create a representative testbench (`.vt`) that exercises the design functions appropriately.
5. To specify the appropriate hierarchy level for signals in the output `.vcd`, add the following line to the project `.qsf` file:

```
set_global_assignment -name EDA_TEST_BENCH_DESIGN_INSTANCE_NAME
     <DUT instance path> -section_id eda_simulation
```
(2)

6. After Fitter processing is complete, click **Processing ➤ Start ➤ Start EDA Netlist Writer**. EDA Netlist Writer generates the following files in `/<project>/simulation/modelsim/power/`:
   - `<project>.vo` (contains a reference to the `.sdo` file by default)
   - `<project>_dump_all_vcd_nodes.tcl`—specifies nodes to save in `.vcd`
   - `<project>_v.sdo`—back-annotated delay estimates
7. Create a ModelSim script (`.do`) to load the design and testbench, start ModelSim, and then source the `.do` script.
8. To specify the signals ModelSim includes in the `.vcd` file, source `*_dump_all_vcd_nodes.tcl` in ModelSim.
9. To generate the `.vcd` file, simulate the test bench and netlist in ModelSim. The `.vcd` file generates according to your specifications.
10. Specify the `.vcd` as an input to power analysis, as Generating Signal Activity Data for Power Analysis on page 12 describes.

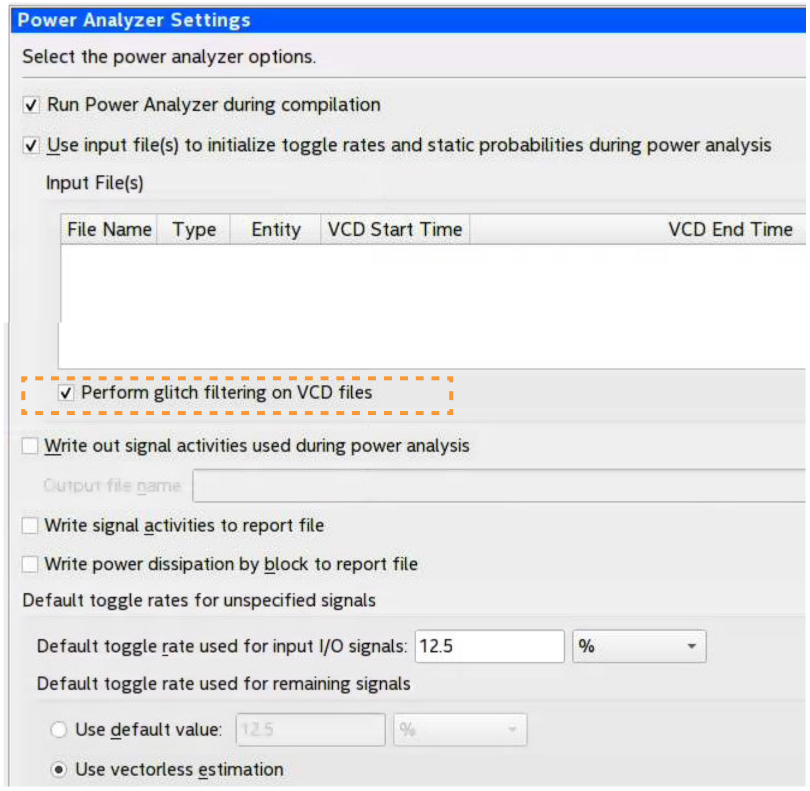
(2) Specify the full hierarchical path in the testbench, not just the instance name. For example, specify a|b|c, not just c.

Note: The EDA Netlist Writer currently supports `.sdo` file generation only for Verilog `.vo` simulation in the ModelSim simulator (not ModelSim - Intel FPGA Edition) for Intel Stratix 10 designs. The EDA Netlist Writer does not currently support `.sdo` file generation for any other simulator or device family.

1.3.2.1.3. Simulation Glitch Filtering

You can enable glitch filtering in the `.vcd` that you generate in a third-party simulator for use in power analysis by turning on the **Perform glitch filtering on VCD files** option.

Figure 12. Enabling Glitch Filtering for VCD


Power Analyzer Settings

Select the power analyzer options.

[✓] Run Power Analyzer during compilation

[✓] Use input file(s) to initialize toggle rates and static probabilities during power analysis

Input File(s)

File Name	Type	Entity	VCD Start Time	VCD End Time

[✓] Perform glitch filtering on VCD files

[ ] Write out signal activities used during power analysis

Output file name

[ ] Write signal activities to report file

[ ] Write power dissipation by block to report file

Default toggle rates for unspecified signals

Default toggle rate used for input I/O signals: 12.5 %

Default toggle rate used for remaining signals

○ Use default value: 12.5 %

● Use vectorless estimation

The Power Analyzer defines a glitch as two signal transitions so closely spaced in time that the pulse, or glitch, occurs faster than the logic and routing circuitry can respond. The output of a transport delay model simulator contains glitches for some signals. The logic and routing structures of the device form a low-pass filter that filters out glitches that are tens to hundreds of picoseconds long, depending on the device family.

Some third-party simulators use different models than the transport delay model as the default model. Different models cause differences in signal activity and power estimation. The inertial delay model, which is the ModelSim default model, filters out more glitches than the transport delay model and usually yields a lower power estimate.

Note: Intel FPGA recommends that you use the transport simulation model when using the Intel Quartus Prime software glitch filtering support with third-party simulators. Simulation glitch filtering has little effect if you use the inertial simulation model.

Glitch filtering in a simulator can also filter a glitch on one logic element (LE) (or other circuit element) output from propagating to downstream circuit elements to ensure that the glitch does not affect simulated results. Glitch filtering prevents a glitch on one signal from producing non-physical glitches on all downstream logic, which can result in a signal toggle rate and a power estimate that are too high. Circuit elements in which every input transition produces an output transition, including multipliers and logic cells configured to implement XOR functions, are especially prone to glitches. Therefore, circuits with such functions can have power estimates that are too high when glitch filtering is not used.

Note: Intel FPGA recommends that you use the glitch filtering feature to obtain the most accurate power estimates. For `.vcd` files, the Power Analyzer flows support two levels of glitch filtering.

The `.vcd` file reader performs glitch filtering that is complementary to simulation glitch filtering, but is often less precise. While the `.vcd` file reader has the ability to remove glitches on logic blocks, the file reader cannot determine how a given glitch potentially affects downstream logic and routing. Filtering the glitches during simulation avoids switching downstream routing and logic automatically.

Note: When running simulation for design verification (rather than to produce input to the Power Analyzer), Intel recommends that you turn off the glitch filtering option to produce the most rigorous and conservative simulation from a functionality viewpoint. When performing simulation to produce input for the Power Analyzer, Intel FPGA recommends that you turn on the glitch filtering to produce the most accurate power estimates.

1.3.2.2. Signal Activities from RTL (Functional) Simulation, Supplemented by Vectorless Estimation

In the functional simulation flow, simulation provides toggle rates and static probabilities for all pins and registers in your design. Vectorless estimation fills in the values for all the combinational nodes between pins and registers, giving good results. This flow usually provides a compilation time benefit when you use the third-party RTL simulator.

1.3.2.2.1. RTL Simulation Limitation

RTL simulation may not provide signal activities for all registers in the post-fitting netlist because synthesis loses some register names. For example, synthesis might automatically transform state machines and counters, thus changing the names of registers in those structures. As a result, a large number of nodes in the `.vcd` file may not match the nodes in your design netlist, which can result in the power analysis results being less accurate or of lower confidence.

1.3.2.3. Signal Activities from Vectorless Estimation and User-Supplied Input Pin Activities

The vectorless estimation flow provides a low level of accuracy, because vectorless estimation for registers is not entirely accurate.

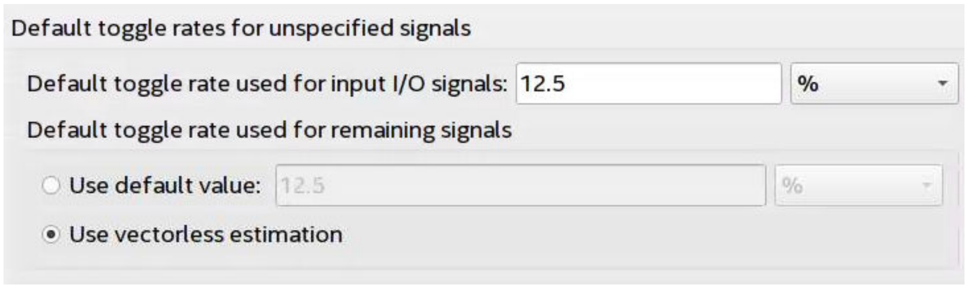
1.3.2.4. Signal Activities from User Defaults Only

The user defaults only flow provides the lowest degree of accuracy.

1.3.3. Specifying the Default Toggle Rate

You can specify the **Default toggle rates for unspecified signals** in your design for power analysis. The Power Analyzer uses the default toggle rate when no other method specifies the signal activity data.

Figure 13. Specifying the Default Toggle Rate



Default toggle rates for unspecified signals

Default toggle rate used for input I/O signals: 12.5 %

Default toggle rate used for remaining signals

○ Use default value: 12.5 %

● Use vectorless estimation

You specify the toggle rate in absolute terms (transitions per second), or as a fraction of the clock rate in effect for each node. The toggle rate for a clock derives from the timing settings for the clock. For example, if the Power Analyzer specifies a clock with an f_{MAX} constraint of 100 MHz and a default relative toggle rate of 20%, nodes in this clock domain transition in 20% of the clock periods, or 20 million transitions occur per second.

In some cases, the Power Analyzer cannot determine the clock domain for a node because the clock domain is ambiguous. For example, the Power Analyzer cannot determine a clock domain for a node unless you specify sufficient timing constraints for the clock domains. If the Power Analyzer cannot determine the clock domain for a node, the Power Analyzer substitutes and reports a toggle rate of zero.

Note: The transceiver I/O toggle rate is determined by the XCVR data rate value specified in your IP catalog settings. Do not include transceiver I/O toggle rate in the default toggle rates that you specify in the Power Analyzer.

Related Information

Toggle Rate on page 34

1.3.4. Specifying Toggle Rates for Specific Nodes

You can assign toggle rates and static probabilities to individual nodes in the design. These assignments have the highest priority, overriding data from all other signal activity sources.

You must use the Assignment Editor or Tcl commands to create the **Power Toggle Rate** and **Power Static Probability** assignments. You can specify the power toggle rate as an absolute toggle rate in transitions per second using the **Power Toggle Rate** assignment, or you can use the **Power Toggle Rate Percentage** assignment to specify a toggle rate relative to the clock domain of the assigned node for a more specific assignment made in terms of hierarchy level.

Note: If you use the **Power Toggle Rate Percentage** assignment, and the node does not have a clock domain, the Intel Quartus Prime software issues a warning and ignores the assignment.

Assigning toggle rates and static probabilities to individual nodes is appropriate for signals in which you have knowledge of the signal being analyzed. For example, if you know that a 100 MHz data bus or memory output produces data that is essentially random (uncorrelated in time), you can directly enter a 0.5 static probability and a toggle rate of 50 million transitions per second.

The Power Analyzer treats bidirectional I/O pins differently. The combinational input port and the output pad for a pin share the same name. However, those ports might not share the same signal activities. For reading signal activity assignments, the Power Analyzer creates a distinct name *<node_name~output>* when configuring the bidirectional signal as an output and *<node_name~result>* when configuring the signal as an input. For example, if a design has a bidirectional pin named `MYPIN`, assignments for the combinational input use the name `MYPIN~result`, and the assignments for the output pad use the name `MYPIN~output`.

Note: When you create the logic assignment in the Assignment Editor, you cannot find the `MYPIN~result` and `MYPIN~output` node names in the Node Finder. Therefore, to create the logic assignment, you must manually enter the two differentiating node names to create the assignment for the input and output port of the bidirectional pin.

1.3.4.1. Clock Node Toggle Rates

For clock nodes, the Power Analyzer uses timing requirements to derive the toggle rate when neither simulation data nor user-entered signal activity data is available. f_{MAX} requirements specify full cycles per second, but each cycle represents a rising transition and a falling transition. For example, a clock f_{MAX} requirement of 100 MHz corresponds to 200 million transitions per second for the clock node.

1.3.5. Avoiding Simulation Node Name Match

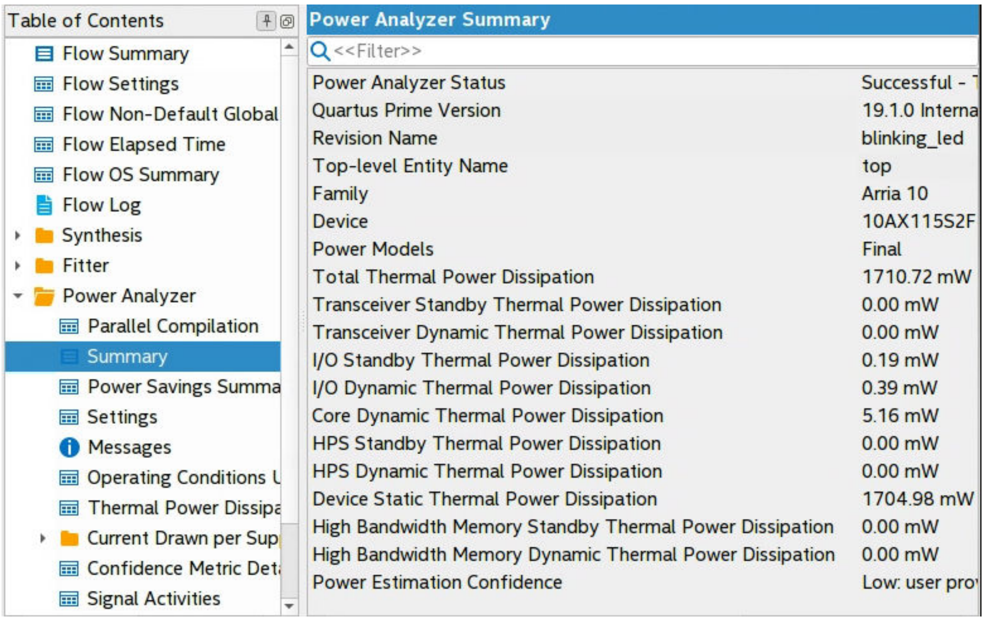
Node name mismatches happen when you have `.vcd` applied to entities other than the top-level entity. In a modular design flow, the gate-level simulation files created in different Intel Quartus Prime projects might not match their node names with the current Intel Quartus Prime project.

For example, you may have a file named `8b10b_enc.vcd`, which the Intel Quartus Prime software generates in a separate project called `8b10b_enc` while simulating the `8b10b` encoder. If you import the `.vcd` into another project called `Top`, you might encounter name mismatches when applying the `.vcd` to the `8b10b_enc` module in the `Top` project. This mismatch happens because the Intel Quartus Prime software might name all the combinational nodes in the `8b10b_enc.vcd` differently than in the `Top` project. To avoid such mismatches, Intel recommends using `.vcd` files generated from simulation of your top level project.

1.4. Viewing Power Analysis Reports

Following successful power analysis, click the **Power Analyzer** pulldown in the **Table of Contents** of the Compilation Report, to view the Power Analysis section of the report.

Figure 14. Power Analysis Reports



Power Analyzer Summary

Power Analyzer Status	Successful -
Quartus Prime Version	19.1.0 Interna
Revision Name	blinking_led
Top-level Entity Name	top
Family	Arria 10
Device	10AX115S2F
Power Models	Final
Total Thermal Power Dissipation	1710.72 mW
Transceiver Standby Thermal Power Dissipation	0.00 mW
Transceiver Dynamic Thermal Power Dissipation	0.00 mW
I/O Standby Thermal Power Dissipation	0.19 mW
I/O Dynamic Thermal Power Dissipation	0.39 mW
Core Dynamic Thermal Power Dissipation	5.16 mW
HPS Standby Thermal Power Dissipation	0.00 mW
HPS Dynamic Thermal Power Dissipation	0.00 mW
Device Static Thermal Power Dissipation	1704.98 mW
High Bandwidth Memory Standby Thermal Power Dissipation	0.00 mW
High Bandwidth Memory Dynamic Thermal Power Dissipation	0.00 mW
Power Estimation Confidence	Low: user pro

The Power Analysis reports contains the following sections:

Summary

The Summary section of the report shows the estimated total thermal power consumption of your design. This includes dynamic, static, and I/O thermal power consumption. The I/O thermal power includes the total I/O power drawn from the V_{CCIO} and V_{CCPD} power supplies and the power drawn from V_{CCINT} in the I/O subsystem including I/O buffers and I/O registers. The report also includes a confidence metric that reflects the overall quality of the data sources for the signal activities. For example, a **Low** power estimation confidence value reflects that you have provided insufficient toggle rate data, or most of the signal activity information used for power estimation is from default or vectorless estimation settings. For more information about the input data, refer to the Power Analyzer Confidence Metric report.

Power Savings Summary

Lists any savings (in mW) and the type of savings method, such as SmartVID Power Savings.

Parallel Compilation

When you enable parallel compilation, the Parallel Compilation report list the number of processors you use during Power Analysis

Settings

The Settings section of the report shows the Power Analyzer settings information of your design, including the default input toggle rates, operating conditions, and other relevant setting information.

Simulation Files Read

The Simulation Files Read section of the report lists the simulation output file that the `.vcd` used for power estimation. This section also includes the file ID, file type, entity, VCD start time, VCD end time, the unknown percentage, and the toggle percentage. The unknown percentage indicates the portion of the design module unused by the simulation vectors.

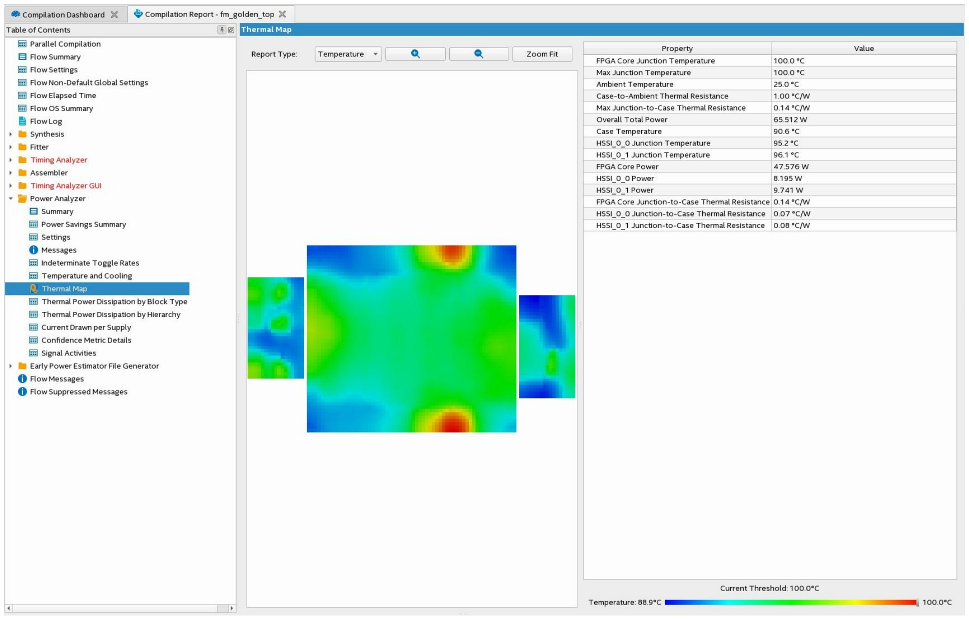
Operating Conditions Used

The Operating Conditions Used section of the report shows device characteristics, voltages, temperature, and cooling solution, if any, during the power estimation. This section also shows the entered junction temperature or auto-computed junction temperature during the power analysis.

Thermal Map Visualization

For Intel Agilex FPGA designs, the Power Analyzer provides a visualization of the expected thermal distribution on the core die and the transceiver dies. This data is available when you run the Power Analyzer on your compiled Intel Agilex FPGA design. After you run the Power Analyzer, select the **Thermal Map** section in the Power Analyzer report. You can set the threshold temperature you want to use, which is useful if you are making any what-if analyses based on your thermal design. You can adjust the threshold temperature in increments of 5°C, between the ambient temperature (or 50°C, whichever is lower), and an upper limit of 100°C.

Figure 15. Temperature View of the Thermal Map for an Intel Agilex FPGA Design



Report Type: Temperature

Property	Value
FPGA Core Junction Temperature	100.0 °C
Max Junction Temperature	100.0 °C
Ambient Temperature	25.0 °C
Case-to-Ambient Thermal Resistance	1.00 °C/W
Max Junction-to-Case Thermal Resistance	0.14 °C/W
Overall Total Power	65.512 W
Case Temperature	90.6 °C
HSSI_0_0 Junction Temperature	95.2 °C
HSSI_0_1 Junction Temperature	96.1 °C
FPGA Core Power	47.576 W
HSSI_0_0 Power	8.195 W
HSSI_0_1 Power	9.741 W
FPGA Core Junction-to-Case Thermal Resistance	0.14 °C/W
HSSI_0_0 Junction-to-Case Thermal Resistance	0.07 °C/W
HSSI_0_1 Junction-to-Case Thermal Resistance	0.08 °C/W

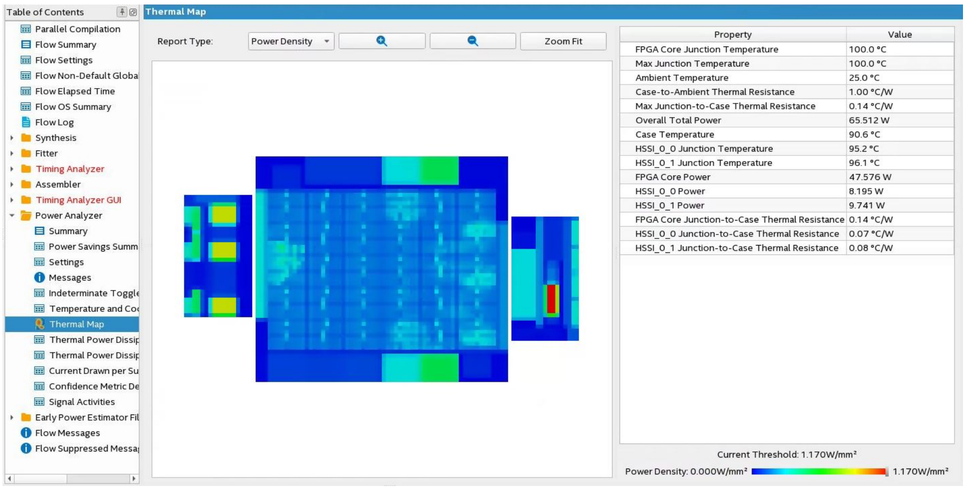
Current Threshold: 100.0°C

Temperature: 88.9°C 100.0°C

Knowing the locations of hot spots in your design can help you make modifications as necessary for proper operation of the system.

The thermal map report can show two different views—a temperature view and a power density view. You can choose the view from a pulldown selection in the GUI, when you open the thermal map in the Power Analyzer.

Figure 16. Power Density View of the Thermal Map for an Intel Agilex FPGA Design



Property	Value
FPGA Core Junction Temperature	100.0 °C
Max Junction Temperature	100.0 °C
Ambient Temperature	25.0 °C
Case-to-Ambient Thermal Resistance	1.00 °C/W
Max Junction-to-Case Thermal Resistance	0.14 °C/W
Overall Total Power	65.512 W
Case Temperature	90.6 °C
HSSI_0_0 Junction Temperature	95.2 °C
HSSI_0_1 Junction Temperature	96.1 °C
FPGA Core Power	47.576 W
HSSI_0_0 Power	8.195 W
HSSI_0_1 Power	9.741 W
FPGA Core Junction-to-Case Thermal Resistance	0.14 °C/W
HSSI_0_0 Junction-to-Case Thermal Resistance	0.07 °C/W
HSSI_0_1 Junction-to-Case Thermal Resistance	0.08 °C/W

Thermal Power Dissipated by Block

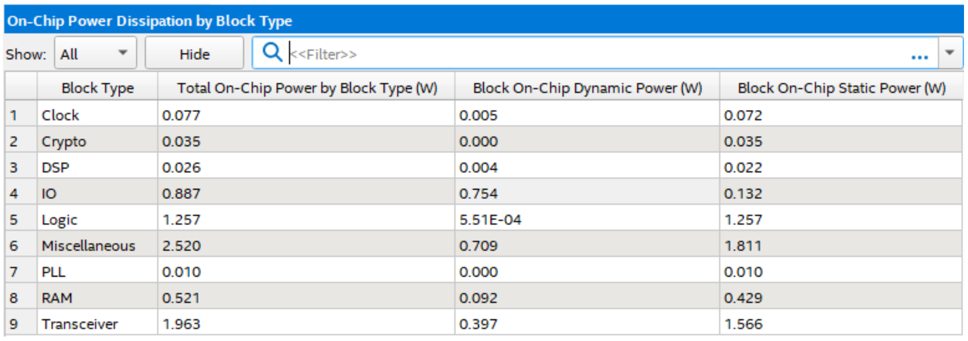
The Thermal Power Dissipated by Block section of the report shows estimated thermal dynamic power and thermal static power consumption categorized by atoms. This information provides you with estimated power consumption for each atom in your design.

By default, this section does not contain any data, but you can turn on the report with the **Write power dissipation by block to report file** option on the **Power Analyzer Settings** page.

On-Chip Power Dissipation by Block Type

The On-Chip Power Dissipation by Block Type section of the report shows the estimated total on-chip power consumption by block type, and estimated on-chip dynamic power and estimated on-chip static power, by block type.

Figure 17. On-Chip Power Dissipation by Block Type


On-Chip Power Dissipation by Block Type
Show: All
Hide
<<Filter>>

	Block Type	Total On-Chip Power by Block Type (W)	Block On-Chip Dynamic Power (W)	Block On-Chip Static Power (W)
1	Clock	0.077	0.005	0.072
2	Crypto	0.035	0.000	0.035
3	DSP	0.026	0.004	0.022
4	IO	0.887	0.754	0.132
5	Logic	1.257	5.51E-04	1.257
6	Miscellaneous	2.520	0.709	1.811
7	PLL	0.010	0.000	0.010
8	RAM	0.521	0.092	0.429
9	Transceiver	1.963	0.397	1.566

On-Chip Power Dissipation by Hierarchy

This On-Chip Power Dissipation by Hierarchy section of the report shows estimated total on-chip power consumption by hierarchy node, and both the cumulative and current-hierarchy-level dynamic, static, and routing power consumptions. This information is useful when locating modules with high power consumption in your design. (Available for Intel Agilex devices.)

Core Dynamic Thermal Power Dissipation by Clock Domain

The Core Dynamic Thermal Power Dissipation by Clock Domain section of the report shows the estimated total core dynamic power dissipation by each clock domain, which provides designs with estimated power consumption for each clock domain in the design. If the clock frequency for a domain is unspecified by a constraint, the clock frequency is listed as "unspecified." For all the combinational logic, the clock domain is listed as no clock with zero MHz.

Current Drawn per Supply

The Current Drawn per Supply section of the report lists the current drawn from each voltage supply. The V_{CCIO} and V_{CCPD} voltage supplies are further categorized by I/O bank and by voltage. This section also lists the minimum safe power supply size (current supply ability) for each supply voltage. Minimum current requirement can be higher than user mode current requirement in cases in which the supply has a specific power up current requirement that goes beyond user mode requirement.

The I/O thermal power dissipation on the summary page does not correlate directly to the power drawn from the V_{CCIO} and V_{CCPD} voltage supplies listed in this report. This is because the I/O thermal power dissipation value also includes portions of the V_{CCINT} power, such as the I/O element (IOE) registers, which are modeled as I/O power, but do not draw from the V_{CCIO} and V_{CCPD} supplies.

The reported current drawn from the I/O Voltage Supplies (ICCIO and ICCPD) as reported in the Power Analyzer report includes any current drawn through the I/O into off-chip termination resistors. This can result in ICCIO and ICCPD values that are higher than the reported I/O thermal power, because this off-chip current dissipates as heat elsewhere and does not factor in the calculation of device temperature. Therefore, total I/O thermal power does not equal the sum of current drawn from each V_{CCIO} and V_{CCPD} supply multiplied by V_{CCIO} and V_{CCPD} voltage.

For SoC devices, there is no standalone ICC_AUX_SHARED current drawn information. The ICC_AUX_SHARED is reported together with ICC_AUX.

Confidence Metric Details

The Confidence Metric is defined in terms of the total weight of signal activity data sources for both combinational and registered signals. Each signal has two data sources allocated to it; a toggle rate source and a static probability source.

The Confidence Metric Details section also indicates the quality of the signal toggle rate data to compute a power estimate. The confidence metric is low if the signal toggle rate data comes from poor predictors of real signal toggle rates in the device during an operation. Toggle rate data that comes from simulation, user-entered assignments on specific signals or entities are reliable. Toggle rate data from default toggle rates (for example, 12.5% of the clock period) or vectorless estimation are relatively inaccurate. This section gives an overall confidence rating in the toggle rate data, from low to high. This section also summarizes how many pins, registers, and combinational nodes obtained their toggle rates from each of simulation, user entry, vectorless estimation, or default toggle rate estimations. This detailed information helps you understand how to increase the confidence metric, letting you determine your own confidence in the toggle rate data.

Signal Activities

The Signal Activities section lists toggle rates and static probabilities assumed by power analysis for all signals with fan-out and pins. This section also lists the signal type (pin, registered, or combinational) and the data source for the toggle rate and static probability. By default, this section does not contain any data, but you can turn on the report with the **Write signal activities to report file** option on the **Power Analyzer Settings** page.

Intel recommends that you keep the **Write signal activities to report file** option turned off for a large design because of the large number of signals present. You can use the Assignment Editor to specify that activities for individual nodes or entities are reported by assigning an on value to those nodes for the **Power Report Signal Activities** assignment.

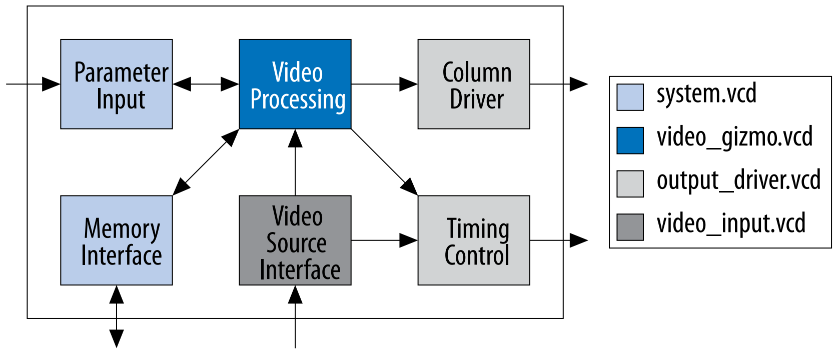
Messages

The Messages section lists the messages that the Intel Quartus Prime software generates during the analysis.

1.5. Power Analysis in Modular Design Flows

In modular or hierarchical design flows you develop each design block separately, and then instantiate these blocks into a higher-level design to form a complete design. The Intel Quartus Prime software supports simulation and power analysis of the top-level design or individual blocks with the design.

Figure 18. Modular Simulation Flow

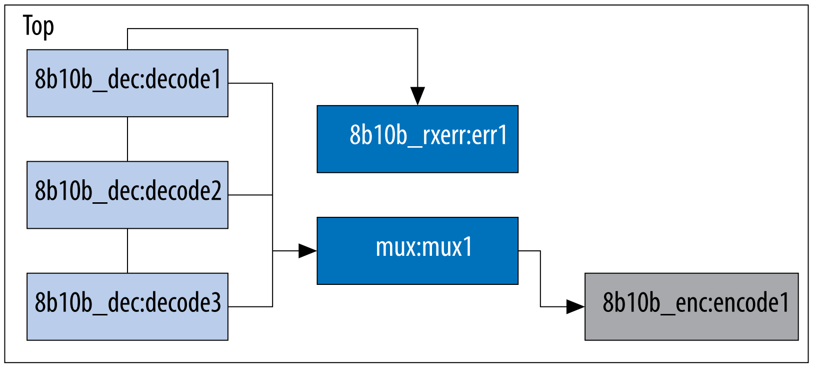


You can associate multiple .vcd simulation output files with specific node names, enabling the integration of partial design simulations into a complete design power analysis. When specifying multiple .vcd files for a node, more than one simulation file can contain signal activity information for the same signal. In those cases, the Power Analyzer follows these rules:

- When you apply multiple .vcd files to the same design node, the Power Analyzer calculates the signal activity as the equal-weight arithmetic average of each .vcd.
- When you apply multiple simulation files to design nodes at different levels in the design hierarchy, the signal activity in the power analysis derives from the simulation file that applies to the most specific design node.

The following figure shows an example of a hierarchical design:

Figure 19. Example Hierarchical Design



The top-level module of the design, called Top, consists of three 8b/10b decoders, followed by a mux. The software encodes the output of the mux to produce the final output of the top-level module. An error-handling module handles any 8b/10b decoding errors. The Top module contains the top-level entity of the design and any logic not defined as part of another module. The design file for the top-level module can be a wrapper for the hierarchical entities or can contain its own logic.

The following usage scenarios show common ways that you can simulate the design and import the .vcd into the Power Analyzer:

1.5.1. Complete Design Simulation Power Analysis Flow

You can simulate the entire gate-level design and generate a .vcd from a third-party simulator. The Power Analyzer can then import the .vcd (specifying the top-level design). The resulting power analysis uses the signal activities information from the generated .vcd, including those that apply to submodules, such as decode [1-3], err1, mux1, and encode1.

1.5.2. Modular Design Simulation Power Analysis Flow

You can independently simulate the top-level design, and then import all the resulting .vcd files into the Power Analyzer. For example, you can simulate the 8b10b_dec independent of the entire design and mux, 8b10b_rxerr, and 8b10b_enc. You can then import the .vcd files generated from each simulation by specifying the appropriate instance name. For example, if the files produced by the simulations are `8b10b_dec.vcd`, `8b10b_enc.vcd`, `8b10b_rxerr.vcd`, and `mux.vcd`, you can use the import specifications in the following table:

Table 4. Import Specifications

File Name	Entity
8b10b_dec.vcd	Top|8b10b_dec:decode1
8b10b_dec.vcd	Top|8b10b_dec:decode2
8b10b_dec.vcd	Top|8b10b_dec:decode3
8b10b_rxerr.vcd	Top|8b10b_rxerr:err1
8b10b_enc.vcd	Top|8b10b_enc:encode1
mux.vcd	Top|mux:mux1

The resulting power analysis applies the simulation vectors in each file to the assigned instance. Simulation provides signal activities for the pins and for the outputs of functional blocks. If the inputs to an instance are input pins for the entire design, the simulation file associated with that instance does not provide signal activities for the inputs of that instance. For example, an input to an instance such as `mux1` has its signal activity specified at the output of one of the decode instances.

1.5.3. Multiple Simulation Power Analysis Flow

You can perform multiple simulations of an entire design or specific modules of a design. For example, in the process of verifying the top-level design, you can have three different simulation testbenches: one for normal operation, and two for corner cases. Each of these simulations produces a separate `.vcd`. In this case, apply the different `.vcd` file names to the same top-level entity, as shown in the following table.

Table 5. Multiple Simulation File Names and Entities

File Name	Entity
normal.vcd	Top
corner1.vcd	Top
corner2.vcd	Top

The resulting power analysis uses an arithmetic average of the signal activities calculated from each simulation file to obtain the final signal activities used. If a signal `err_out` has a toggle rate of zero transition per second in `normal.vcd`, 50 transitions per second in `corner1.vcd`, and 70 transitions per second in `corner2.vcd`, the final toggle rate in the power analysis is 40 transitions per second.

If you do not want the Power Analyzer to read information from multiple instances and take an arithmetic average of the signal activities, use a `.vcd` that includes only signals from the instance that you care about.

1.5.4. Overlapping Simulation Power Analysis Flow

You can perform a simulation on the entire design, and more exhaustive simulations on a submodule, such as `8b10b_rxerr`. The following table lists the import specification for overlapping simulations:

Table 6. Overlapping Simulation Import Specifications

File Name	Entity
`full_design.vcd`	`Top`
`error_cases.vcd`	`Top|8b10b_rxerr:err1`

In this case, the software uses signal activities from `error_cases.vcd` for all the nodes in the generated `.vcd` and uses signal activities from `full_design.vcd` for only those nodes that do not overlap with nodes in `error_cases.vcd`. In general, the more specific hierarchy (the most bottom-level module) derives signal activities for overlapping nodes.

1.5.5. Partial Design Simulation Power Analysis Flow

You can perform a simulation in which the entire simulation time is not applicable to signal activity calculation. For example, if you run a simulation for 10,000 clock cycles and reset the chip for the first 2,000 clock cycles. If the Power Analyzer performs the signal activity calculation over all 10,000 cycles, the toggle rates are only 80% of their steady state value (because the chip is in reset for the first 20% of the simulation). In this case, you must specify the useful parts of the `.vcd` for power analysis. The **Limit VCD Period** option enables you to specify a start and end time when performing signal activity calculations.

1.5.5.1. Specifying Start and End Time for Signal Activity Calculations

To specify a start and end time for signal activity calculations using the **Limit VCD period** option, follow these steps:

1. In the Intel Quartus Prime software, click **Assignments ➤ Settings**.
2. Under the **Category** list, click **Power Analyzer Settings**.
3. Turn on the **Use input file(s) to initialize toggle rates and static probabilities during power analysis** option.
4. Click **Add**.
5. In the **File name** and **Entity** fields, browse to the necessary files.
6. Under **Simulation period**, turn on **VCD file** and **Limit VCD period** options.
7. In the **Start time** and **End time** fields, specify the desired start and end time.
8. Click **OK**.

You can also use the following Tcl or `.qsf` assignment to specify `.vcd` files:

```
set_global_assignment -name POWER_INPUT_FILE_NAME "test.vcd" -section_id
test.vcd
set_global_assignment -name POWER_VCD_FILE_START_TIME "10 ns" -section_id
test.vcd
set_global_assignment -name POWER_VCD_FILE_END_TIME "1000 ns" -section_id
test.vcd
set_instance_assignment -name POWER_READ_INPUT_FILE test.vcd -to test_design
```

1.5.6. Vectorless Estimation Power Analysis Flow

For some device families, the Power Analyzer automatically derives estimates for signal activity on nodes with no simulation or user-entered signal activity data.

Vectorless estimation statistically estimates the signal activity of a node based on the signal activities of nodes feeding that node, and on the actual logic function that the node implements. Vectorless estimation cannot derive signal activities for primary inputs. Vectorless estimation is accurate for combinational nodes, but not for registered nodes. Therefore, the Power Analyzer requires simulation data for at least the registered nodes and I/O nodes for accuracy.

1.6. Scripting Support

You can run procedures and create settings described in this chapter in a Tcl script. Alternatively, you can run procedures at a command prompt. For more information about scripting command options, refer to the Intel Quartus Prime Command-Line and Tcl API Help browser. To run the Help browser, type the following command at the command prompt:

```
quartus_sh --qhelp
```

Related Information

Intel Quartus Prime Pro Edition Settings File Reference Manual

1.6.1. Running the Power Analyzer from the Command–Line

The executable to run the Power Analyzer is `quartus_pow`. For a complete listing of all command–line options supported by `quartus_pow`, type the following command at a system command prompt:

```
quartus_pow --help
```

or

```
quartus_sh --qhelp
```

The following lists the examples of using the `quartus_pow` executable. Type the command listed in the following section at a system command prompt:

Note: These examples assume that operations are performed on Intel Quartus Prime project called *sample*.

- To instruct the Power Analyzer to generate a EPE File:

```
quartus_pow sample --output_epe=sample.csv ←
```

- To instruct the Power Analyzer to generate a EPE File without performing the power estimate:

```
quartus_pow sample --output_epe=sample.csv --estimate_power=off  ←
```

- To instruct the Power Analyzer to generate an export file for the Power and Thermal Calculator:

```
quartus_pow sample   --output_ptc=sample.qptc  ←
```

- To instruct the Power Analyzer to generate an export file for the Power and Thermal Calculator without performing the power estimate:

```
quartus_pow sample --output_ptc=sample.qptc --estimate_power=off  ←
```

- To instruct the Power Analyzer to use a `.vcd` as input (`sample.vcd`):

```
quartus_pow sample --input_vcd=sample.vcd ←
```

- To instruct the Power Analyzer to use two `.vcd` files as input files (`sample1.vcd` and `sample2.vcd`), perform glitch filtering on the `.vcd` and use a default input I/O toggle rate of 10,000 transitions per second:

```
quartus_pow sample --input_vcd=sample1.vcd --input_vcd=sample2.vcd \
--vcd_filter_glitches=on --\
default_input_io_toggle_rate=10000transitions/s
```

- To instruct the Power Analyzer not to use an input file, specify a default input I/O toggle rate of 60%, with vectorless estimation off, and a default toggle rate of 20% on all remaining signals:

```
quartus_pow sample --no_input_file --default_input_io_toggle_rate=60% \
--use_vectorless_estimation=off --default_toggle_rate=20%
```

Note: No command–line options are available to specify the information found on the **Operating Settings and Conditions** and **Power Analyzer Settings ➤ Thermal** pages. Use the Intel Quartus Prime GUI to specify these options.

The `quartus_pow` executable creates a report file, *<revision name>*`.pow.rpt`. You can locate the report file in the main project directory. The report file contains the same information that the Power Analyzer Compilation Report.

Related Information

Viewing Power Analysis Reports on page 18

1.7. Power Analysis Revision History

The following revision history applies to this chapter:

Document Version	Intel Quartus Prime Version	Changes
2022.06.22	21.4	In the *Power Analysis Tools* topic, added statements about Intel Quartus Prime Power Analyzer accuracy for Intel Stratix 10 and Intel Agilex designs.
2021.12.13	21.4	• In the *Settings for Power Analysis* topic, modified a *Thermal Settings* entry in the *Power Analyzer and Thermal Settings* table. Specifically, changed *Apply Recommended Margin* to *Apply Additional Margin*, and modified the description accordingly. • In the *Viewing Power Analysis Reports* topic, added the *Crypto* block type to the *On-Chip Power Dissipation by Block Type* section.
2021.10.04	21.3	Recast the note in the *Power Analysis Tools* topic for greater clarity.

continued...

Document Version	Intel Quartus Prime Version	Changes
2021.03.29	21.1	• In the *Power Analysis* topic, updated the figure. • In the *Running the Power Analyzer* topic, modified step 3 and removed the figure following step 9. • Changed the title of the *Device Operating Condition Settings for Power Analysis* topic to *Settings for Power Analysis*, updated the existing figure and added an additional figure, recast the existing table and added an additional table. • In the *Generating Signal Activity Data for Power Analysis* topic, modified step 5 and the figure within step 6. • In the *Viewing Power Analysis Reports* topic, made minor changes to the first paragraph of the *Thermal Map Visualization* section. • In the *Running the Power Analyzer from the Command–Line* topic, modified the note at the bottom of the topic.
2020.12.07	20.3.0	Added note to the *Specifying the Default Toggle Rate* topic.
2020.10.05	20.3.0	• Added mention of gate-level simulation to the *Specifying Power Analyzer Input*, *Specifying Signal Activity Data*, *Using Simulation Signal Activity Data in Power Analysis*, and *Complete Design Simulation Power Analysis Flow* topics, • Added a sentence to the *RTL Simulation Limitation* and *Avoiding Simulation Node Name Match* topics. • Added a *Thermal Map Visualization* section to the *Viewing Power Analysis Reports* topic.
2020.04.13	20.1.0	Added information about the Intel FPGA Power and Thermal Calculator to the following topics: • *Power Analysis* • *Power Analysis Tools* • *Running the Power Analyzer from the Command–Line*
2019.12.04	19.1.0	• Removed references to entity-specific toggle rates in "Specifying Toggle Rates for Specific Nodes." Toggle rates must be either global or node specific.
2019.08.02	19.1.0	• Clarified wording of statements about .vcd files in "Simulation Glitch Filtering" topic. • Corrected typo in "Specifying the Default Toggle Rate" topic. • Corrected typo in "Running the Power Analyzer from the Command Line" topic. • Improved explanation in "Generating Standard Delay Output for Power Analysis" topic.
2019.07.03	19.1.0	• Corrected broken links to Help.
2019.04.01	19.1.0	• Described new support for generation of SDO for use in power analysis. • Retitled some topic headings for greater clarity. • Changed the order of some topics for improved flow of information. • Added descriptions of Power Savings Summary and Parallel Compilation power analysis reports. • Added new Power Analysis flow diagrams.
2018.09.24	18.1.0	• General chapter reorganization. • Moved *Factors Affecting Power Consumption* to chapter: *Power Optimization*. • Updated figure: *Power Analyzer High-level Flow*.

continued...

Document Version	Intel Quartus Prime Version	Changes
		• Divided topic: *Types of Power Analysis* into two topics: *Power Estimations and Design Requirements* and *Design Activity and Power Analysis*. • Updated figure: *Power Analysis Tools from Design Concept through Design Implementation* and renamed to: *Estimation Accuracy for Different Inputs and Power Analysis Tools* • Removed content referring to device families not supported in *Intel Quartus Prime Pro Edition*.
2018.06.11	18.0.0	• In *Comparison of the EPE and the Intel Quartus Prime Power Analyzer*, updated the data output types that the Power Analyzer supports. • In *Comparison of the EPE and the Intel Quartus Prime Power Analyzer*, added row about estimation of transceiver power for features that you enable only through dynamic reconfiguration. • Specified features not supported by the Power Analyzer.
2017.05.08	17.0.0	Removed references to PowerPlay name. Power analysis occurs in the Intel Quartus Prime Power Analyzer.
2016.10.31	16.1.0	• Implemented Intel rebranding. • Removed support for `.vcd` generation by the Compiler. Generate `.vcd` files for power estimation in your EDA simulator.
2015.11.02	15.1.0	Changed instances of *Quartus II* to *Intel Quartus Prime*.
2014.12.15	14.1.0	• Removed Signal Activities from Full Post-fit Netlist (Timing) Simulation and Signal Activities from Full Post-fit Netlist (Zero Delay) Simulation sections as these are no longer supported. • Updated location of Fitter Settings, Analysis & Synthesis Settings, and Physical Synthesis Optimizations to Compiler Settings.
2014.08.18	14.0a10.0	Updated "Current Drawn from Voltage Supplies" to clarify that for SoC devices or for Arria V SoC and Cyclone V SoC devices, there is no standalone ICC_AUX_SHARED current drawn information. The ICC_AUX_SHARED is reported together with ICC_AUX.
November 2012	12.1.0	• Updated "Types of Power Analyses" on page 8–2, and "Confidence Metric Details" on page 8–23. • Added "Importance of .vcd" on page 8–20, and "Avoiding Power Estimation and Hardware Measurement Mismatch" on page 8–24
June 2012	12.0.0	• Updated "Current Drawn from Voltage Supplies" on page 8–22. • Added "Using the HPS Power Calculator" on page 8–7.
November 2011	10.1.1	• Template update. • Minor editorial updates.
December 2010	10.1.0	• Added links to Quartus II Help, removed redundant material. • Moved "Creating PowerPlay EPE Spreadsheets" to page 8–6. • Minor edits.
July 2010	10.0.0	• Removed references to the Quartus II Simulator. • Updated Table 8–1 on page 8–6, Table 8–2 on page 8–13, and Table 8–3 on page 8–14. • Updated Figure 8–3 on page 8–9, Figure 8–4 on page 8–10, and Figure 8–5 on page 8–12.

continued...

Document Version	Intel Quartus Prime Version	Changes
November 2009	9.1.0	• Updated “Creating PowerPlay EPE Spreadsheets” on page 8–6 and “Simulation Results” on page 8–10. • Added “Signal Activities from Full Post-fit Netlist (Zero Delay) Simulation” on page 8–19 and “Generating a .vcd from Full Post-fit Netlist (Zero Delay) Simulation” on page 8–21. • Minor changes to “Generating a .vcd from ModelSim Software” on page 8–21. • Updated Figure 11–8 on page 11–24.
March 2009	9.0.0	• This chapter was chapter 11 in version 8.1. • Removed Figures 11-10, 11-11, 11-13, 11-14, and 11-17 from 8.1 version.
November 2008	8.1.0	• Updated for the Quartus II software version 8.1. • Replaced Figure 11-3. • Replaced Figure 11-14.
May 2008	8.0.0	• Updated Figure 11–5. • Updated “Types of Power Analyses” on page 11–5. • Updated “Operating Conditions” on page 11–9. • Updated “PowerPlay Power Analyzer Compilation Report” on page 11–31. • Updated “Current Drawn from Voltage Supplies” on page 11–32.

2. Power Optimization

The Intel Quartus Prime software offers power-driven compilation to fully optimize device power consumption. Power-driven compilation focuses on reducing the design's total power consumption in synthesis and place-and-route stages.

This chapter focuses on design optimization options and techniques that help reduce core dynamic power and I/O power. In addition to these techniques, there are additional power optimization techniques available for specific devices, including Programmable Power Technology and Device Speed Grade Selection.

Related Information

- Power Analysis on page 4
- AN 711: Power Reduction Features in Intel Arria 10 Devices
- Intel FPGA Literature and Technical Documentation

2.1. Factors Affecting Power Consumption

Understanding the following factors that affect power consumption allows you to use the Power Analyzer and interpret its results effectively:

Design Activity and Power Analysis on page 32

Device Selection on page 32

Environmental Conditions on page 33

Device Resource Usage on page 33

Signal Activity on page 34

2.1.1. Design Activity and Power Analysis

Power consumption of a device also depends on the design's activity over time. Static power (P_{STATIC}) is the thermal power that a chip dissipates independent of user clocks. P_{STATIC} includes leakage power from all FPGA functional blocks, except for I/O DC bias power and transceiver DC bias power, which are accounted for in the I/O and transceiver sections. Dynamic power is the additional power consumption of a device due to signal activity or switching.

2.1.2. Device Selection

Device families have different power characteristics. Many parameters affect the device family power consumption, including choice of process technology, supply voltage, electrical design, and device architecture.

Power consumption also varies in a single device family. A larger device with more transistors consumes more static power than a smaller device in the same family. In devices that employ global routing architectures, dynamic power can also increase with device size.

The choice of device package also affects the ability of the device to dissipate heat, and you may need to use a different cooling solution to comply with junction temperature constraints.

Process variation can affect power consumption. Process variation primarily impacts static power, because sub-threshold leakage current varies exponentially with changes in transistor threshold voltage. Therefore, you must consult device specifications for static power, and not rely on empirical observation. Process variation has a weak effect on dynamic power.

2.1.3. Environmental Conditions

The main environmental parameters affecting junction temperature are operating temperature and the cooling solution. Operating temperature primarily affects device static power consumption. Higher junction temperatures result in higher static power consumption. The device thermal power and cooling solution that you use must keep the device junction temperature within the maximum operating range for the device.

The following table lists the environmental conditions that influence power consumption.

Table 7. Environmental Conditions that Affect Power Consumption

Environmental Condition	Description
Airflow	Measures how quickly the device replaces heated air from the vicinity of the device with air at ambient temperature. You can either specify airflow as “still air” when you are not using a fan, or as the linear feet per minute rating of the fan in the system. Higher airflow decreases thermal resistance.
Heat Sink and Thermal Compound	A heat sink allows more efficient heat transfer from the device to the surrounding area because of its large surface area exposed to the air. The thermal compound that interfaces the heat sink to the device also influences the rate of heat dissipation. The case-to-ambient thermal resistance (θ_{CA}) parameter describes the cooling capacity of the heat sink and thermal compound employed at a given airflow. Larger heat sinks and more effective thermal compounds reduce θ_{CA}.
Junction Temperature	The junction temperature of a device is equal to: $T_{Junction}=T_{Ambient}+P_{Thermal}\cdot\theta_{JA}$ in which θ_{JA} is the total thermal resistance from the device transistors to the environment, in degrees Celsius per watt. The value θ_{JA} is equal to the sum of the junction-to-case (package) thermal resistance (θ_{JC}), and the case-to-ambient thermal resistance (θ_{CA}) of the cooling solution.
Board Thermal Model	The junction-to-board thermal resistance (θ_{JB}) is the thermal resistance of the path through the board, in degrees Celsius per watt. To compute junction temperature, you can use this board thermal model along with the board temperature, the top-of-chip θ_{JA} and ambient temperatures.

2.1.4. Device Resource Usage

Power consumption depends on the number and types of device resources that a design uses.

2.1.4.1. Number, Type, and Loading of I/O Pins

Output pins drive off-chip components, resulting in high-load capacitance that leads to a high-dynamic power per transition. Terminated I/O standards require external resistors that draw constant (static) power from the output pin.

2.1.4.2. Number and Type of Hard Logic Blocks

A design with more logic elements (LEs), multiplier elements, memory blocks, transceiver blocks, or HPS system tends to consume more power than a design with fewer circuit elements. The operating mode of each circuit element also affects its power consumption.

For example, a DSP block performing 18 × 18 multiplications and a DSP block performing multiply-accumulate operations consume different amounts of dynamic power, because of different amounts of charging internal capacitance on each transition. The operating mode of a circuit element also affects static power.

2.1.4.3. Number and Type of Global Signals

Global signal networks span large portions of the device and have high capacitance, resulting in significant dynamic power consumption. The type of global signal is important as well. Global clocks cover the entire device, whereas quadrant clocks only span one-fourth of the device. Clock networks that span smaller regions have lower capacitance and tend to consume less power. The location of the logic array blocks (LABs) driven by the clock network can also have an impact because the Intel Quartus Prime software automatically disables unused branches of a clock.

2.1.5. Signal Activity

The behavior of each signal in the design is an important factor in estimating power consumption. To get accurate results from the power analysis, the signal activity must represent the actual operating behavior of the design.

The two most important behaviors of a signal are toggle rate and static probability.

2.1.5.1. Toggle Rate

The toggle rate of a signal is the average number of times that the signal changes value per unit of time. The units for toggle rate are transitions per second, and a transition is a change from `1` to `0`, or `0` to `1`.

Note: Inaccurate signal toggle rate data is the largest source of power estimation error.

Dynamic power increases linearly with the toggle rate as you charge the board trace model more frequently for logic and routing. The Intel Quartus Prime software models full rail-to-rail switching. For high toggle rates, especially on circuit output I/O pins, the circuit can transition before fully charging the downstream capacitance. The result is a slightly conservative prediction of power by the Power Analyzer.

Note: The transceiver I/O toggle rate is determined by the XCVR data rate value specified in your IP catalog settings. Do not include transceiver I/O toggle rate in the default toggle rates that you specify in the Power Analyzer.

Related Information

Specifying the Default Toggle Rate on page 17

2.1.5.2. Static Probability

The static probability of a signal is the fraction of time that the signal is logic 1 during device operation. Static probability ranges from 0 (always at ground) to 1 (always at logic-high).

The static probability of input signals impacts the design's static power consumption, due to state-dependent leakage in routing and logic. This effect becomes more important for smaller geometries. In output I/O standards that drive termination resistors, the static power also depends on the static probability on I/O pins.

2.2. Design Space Explorer II for Power-Driven Optimization

The Design Space Explorer II (DSE II) tool allows you to find and implement the project settings that result in best power behavior.

The DSE II offers two options in **Exploration mode** that target power optimization: **Power (High Effort)** and **Power (Aggressive)**. In both cases, the target is an overall improvement in the design's power; specifically, reducing the total thermal power in the design.

When the optimization targets power, the DSE II runs the Intel Quartus Prime Power Analyzer for every group of settings. The resultant reports help you debug the design and determine trade-offs between power requirements and performance optimization.

Related Information

- Design Space Explorer II
  In *Intel Quartus Prime Pro Edition User Guide: Design Optimization*
- Launch Design Space Explorer Command (Tools Menu)
  In *Intel Quartus Prime Help*

2.3. Power-Driven Compilation

The standard Intel Quartus Prime compilation flow consists of Analysis and Synthesis, placement and routing, Assembly, and Timing Analysis. Power-driven compilation takes place at the Analysis and Synthesis and Place-and-Route stages.

Intel Quartus Prime software settings that control power-driven compilation are located in the **Power optimization during synthesis** list in the **Advanced Settings (Synthesis)** dialog box, and the **Power optimization during fitting** list on the **Advanced Fitter Settings** dialog box. The following sections describes these power optimization options at the Analysis and Synthesis and Fitter levels.

2.3.1. Power-Driven Synthesis

Synthesis netlist optimization occurs during the synthesis stage of the compilation flow. You can apply these settings on a project or entity level.

The **Power Optimization During Synthesis** logic option determines how aggressively Analysis & Synthesis optimizes the design for power. To access this option at a project level, click **Assignments ➤ Settings ➤ Compiler Settings ➤ Advanced Settings (Synthesis)**.

Table 8. Power Optimization During Synthesis Options

Settings	Description	Optimization Techniques Included
Off	The Compiler does not perform netlist, placement, or routing optimizations to minimize power.	-
Normal compilation (Default)	The Compiler applies low compute effort algorithms to minimize power through netlist optimizations that do not reduce design performance.	• Memory block optimization • Power-aware logic mapping
Extra effort	Besides the techniques in the **Normal compilation** setting, the Compiler applies high-compute-effort algorithms to minimize power through netlist optimizations. Selecting this option might impact performance.	• Memory block optimization • Power-aware logic mapping • Power-aware memory balance

You can also control memory optimization options from the Intel Quartus Prime **Settings** dialog box. The **Default Parameters** page allows you to edit the **Low_Power_Mode** parameter. The settings for this parameter are equivalent to the values of the **Power Optimization During Synthesis** logic options. The **Low_Power_Mode** parameter always takes precedence over the **Optimize Power for Synthesis** option for power optimization on memory.

Table 9. Low Power Mode Parameter Options

Parameter Value	Equivalent Setting in Power Optimization During Synthesis Logic Option
None	**Off**
Auto	**Normal compilation**
All	**Extra effort**

Related Information

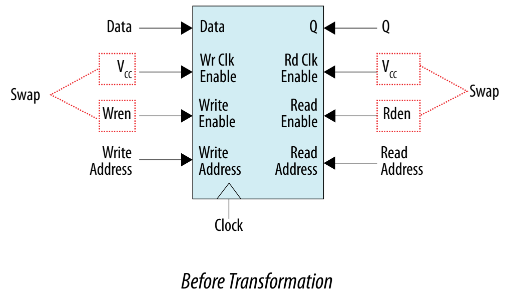
- Clock Enable in Memory Blocks on page 42
- Intel Quartus Prime Compiler Settings on page 39

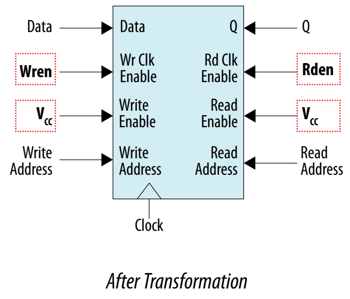
2.3.1.1. Memory Block Optimization

Memory optimization involves moving user-defined read/write enable signals to associated read-and-write clock enable signals for all memory types.

Memory blocks can represent a large fraction of total design dynamic power. Minimizing the number of memory blocks accessed during each clock cycle can significantly reduce memory power.

Figure 20. Memory Block Transformation





In the default implementation of a simple dual-port memory block, write-clock enable signals and read-clock enable signals connect to V_{CC}, making both read and write memory ports active during each clock cycle.

Memory transformation moves the read-enable and write-enable signals to the respective read-clock enable and write-clock enable signals. This technique reduces the design's memory power consumption, because memory ports are shut down when they are not accessed.

2.3.1.2. Power-Aware Logic Mapping

Power-aware logic mapping reduces power by rearranging the logic during synthesis to eliminate nets with high switching rates.

2.3.1.3. Power-Aware Memory Balancing

Power-aware memory balancing chooses the best configuration for a memory implementation and provides optimal power saving by determining the required number of memory blocks, decoder, and multiplexer circuits. When the design does not specify target-embedded memory blocks for the design's memory functions, the power-aware balancer automatically selects them during memory implementation.

The Compiler includes this optimization technique when the **Power Optimization During Synthesis** logic option is set to **Extra effort**.

There is a trade-off between power saved by accessing fewer memories and power consumed by the extra decoder and multiplexor logic. The Intel Quartus Prime software automatically balances the power savings against the costs to choose the lowest power configuration for each logical RAM. The benchmark data shows that the power-driven synthesis can reduce memory power consumption by as much as 60% in Stratix devices.

You can also set the **MAXIMUM_DEPTH** parameter manually to configure the memory for low power optimization. This technique is the same as the power-aware memory balancer, but it is manual rather than automatic like the **Extra effort** setting in the **Power optimization** list. The **MAXIMUM_DEPTH** parameter always takes precedence over the **Optimize Power for Synthesis** options for power optimization on memory optimization. You can set the **MAXIMUM_DEPTH** parameter for memory modules manually in the Intel FPGA IP instantiation or in the IP Catalog.

Related Information

- RAM and ROM Parameter Settings
  In *Intel Stratix 10 Embedded Memory User Guide*

- Maximum Block Depth Configuration
  In *Embedded Memory (RAM: 1-PORT, RAM: 2-PORT, ROM: 1-PORT, and ROM: 2-PORT) User Guide*

2.3.2. Power-Driven Fitter

The Intel Quartus Prime software allows you to control the power-driven compilation setting of the Fitter on a project-wide basis. The **Advanced Fitter Settings** dialog box page provides the **Power optimization during Fitting** logic option, that determines how aggressively the Fitter optimizes the design for power.

Table 10. Power-Driven Fitter Option

Option	Description
Off	The Fitter does not perform optimizations to minimize power.
Normal compilation (Default)	The Fitter applies low compute effort algorithms to minimize power through placement and routing optimizations. These techniques do not reduce design performance. Includes DSP optimizations that create power-efficient DSP block configurations for DSP functions.
Extra effort	Besides the optimization techniques of the **Normal Compilation** option, the Fitter applies high compute effort algorithms to minimize power through placement and routing optimizations. These techniques might impact performance. The **Extra effort** setting for the Fitter requires extensive effort to optimize the design for power and can increase compilation time.

The **Extra effort** setting the Fitter works to minimize power even after the design meets timing requirements by moving the logic closer during placement to localize high-toggling nets and choosing routes with low capacitance.

The **Extra effort** setting uses a Value Change Dump (`.vcd`) file that guides the Fitter to fully optimize the design for power, based on the signal activity of the design. The best power optimization during fitting results from using the most accurate signal activity information. If there is no `.vcd` file, the Intel Quartus Prime software estimates the signal activities from the settings in the **Power Analyzer Settings** page in the **Settings** dialog box, such as assignments, clock assignments, and vectorless estimation values.

Related Information

Assignment Editor Options on page 40

2.3.3. Area-Driven Synthesis

Using area optimization rather than timing or delay optimization during synthesis saves power because you use fewer logic blocks. Using less logic usually means less switching activity.

The Intel Quartus Prime software provides **Speed**, **Balanced**, or **Area** for the **Optimization Technique** option. You can also specify this logic option for specific modules in your design with the Assignment Editor in cases where you want to reduce area using the **Area** setting (potentially at the expense of register-to-register timing performance) while leaving the default **Optimization Technique** setting at **Balanced** (for the best trade-off between area and speed for certain device families). The **Speed Optimization Technique** can increase the resource usage of your design if the constraints are too aggressive and can also result in increased power consumption.

Related Information

Assignment Editor Options on page 40

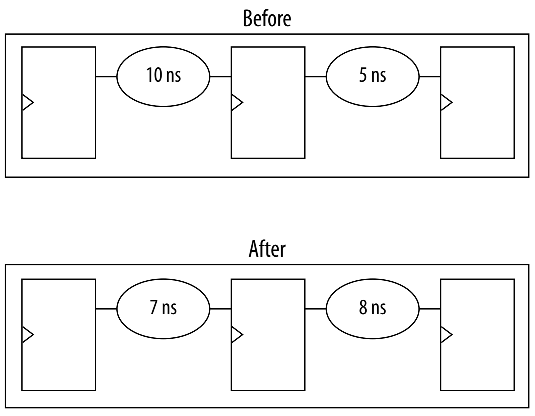
2.3.4. Gate-Level Register Retiming

You can also use gate-level register retiming to reduce circuit switching activity. Retiming shuffles registers across combinational blocks without changing design functionality.

The **Perform gate-level register retiming** option in the Intel Quartus Prime software enables the movement of registers across combinational logic to balance timing, allowing the software to trade off the delay between critical and noncritical paths.

Retiming uses fewer registers than pipelining. In this example of gate-level register retiming, the 10 ns critical delay is reduced by moving the register relative to the combinational logic, resulting in the reduction of data depth and switching activity.

Figure 21. Gate-Level Register Retiming



Gate-level register retiming makes changes at the gate level. If you are using an atom netlist from a third-party synthesis tool, you must also select the **Perform WYSIWYG primitive resynthesis** option to undo the atom primitives to gates mapping (so that register retiming can be performed), and then to remap gates to Intel primitives.

Related Information

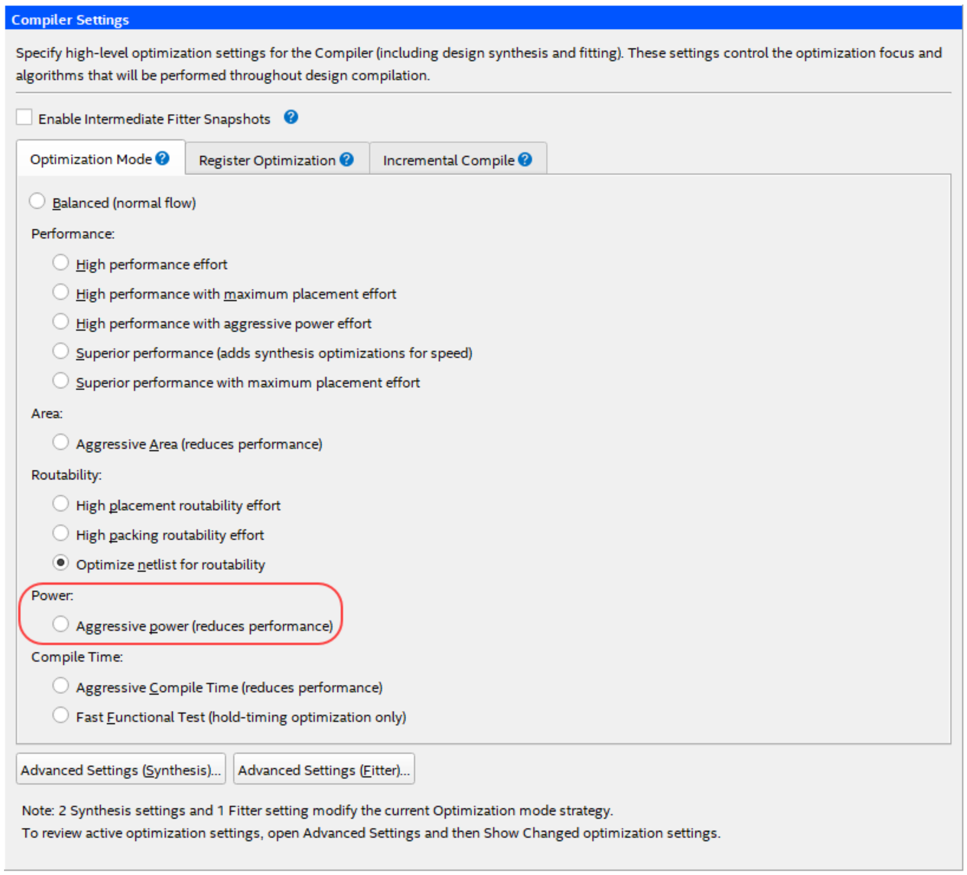
Netlist Optimizations and Physical Synthesis
In *Intel Quartus Prime Pro Edition User Guide: Design Optimization*

2.3.5. Intel Quartus Prime Compiler Settings

The Intel Quartus Prime software provides settings that optimize power for the full design.

To set the optimization mode on the Intel Quartus Prime software, click **Assignments ➤ Settings ➤ Compiler Settings**.

Figure 22. Compiler Settings



Compiler Settings

Specify high-level optimization settings for the Compiler (including design synthesis and fitting). These settings control the optimization focus and algorithms that will be performed throughout design compilation.

☐ Enable Intermediate Fitter Snapshots

Optimization Mode | Register Optimization | Incremental Compile

○ Balanced (normal flow)

Performance:

- ○ High performance effort
- ○ High performance with maximum placement effort
- ○ High performance with aggressive power effort
- ○ Superior performance (adds synthesis optimizations for speed)
- ○ Superior performance with maximum placement effort

Area:

- ○ Aggressive Area (reduces performance)

Routability:

- ○ High placement routability effort
- ○ High packing routability effort
- ● Optimize netlist for routability

Power:

- ○ Aggressive power (reduces performance)

Compile Time:

- ○ Aggressive Compile Time (reduces performance)
- ○ Fast Functional Test (hold-timing optimization only)

Advanced Settings (Synthesis)... | Advanced Settings (Fitter)...

Note: 2 Synthesis settings and 1 Fitter setting modify the current Optimization mode strategy.
To review active optimization settings, open Advanced Settings and then Show Changed optimization settings.

Aggressive Power (reduces performance)

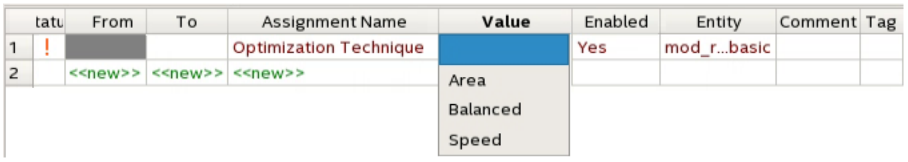
Makes aggressive effort to optimize synthesis for low power. The Compiler further reduces the routing usage of signals with the highest specified or estimated toggle rates, saving additional dynamic power but potentially affecting performance.

2.3.6. Assignment Editor Options

The Assignment Editor allows you to select Optimization Technique & Synthesis Power Optimization for individual modules. With this feature, you can focus on the parts of the design that require more work.

The **Optimization Technique** logic option specifies the overall optimization goal for Analysis & Synthesis: attempt to maximize performance or minimize logic usage.

Figure 23. Optimization Technique Options

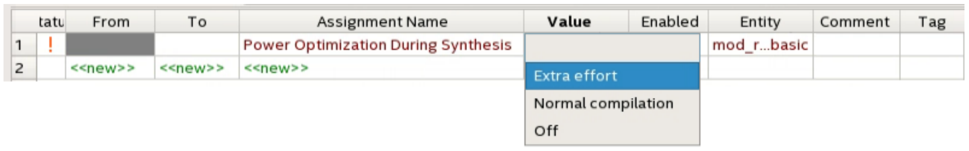


	tatu	From	To	Assignment Name	Value	Enabled	Entity	Comment	Tag
1	!			Optimization Technique		Yes	mod_r...basic		
2		<<new>>	<<new>>	<<new>>					

Area
Balanced
Speed

The **Power Optimization During Synthesis** logic option determines how aggressively Analysis & Synthesis optimizes the design for power.

Figure 24. Power Optimization During Synthesis Options



tatu	From	To	Assignment Name	Value	Enabled	Entity	Comment	Tag
1 !			Power Optimization During Synthesis			mod_r...basic		
2	<<new>>	<<new>>	<<new>>					

Table 11. Power Optimization During Synthesis Options

Settings	Description	Optimization Techniques Included
Off	The Compiler does not perform netlist, placement, or routing optimizations to minimize power.	-
Normal compilation (Default)	The Compiler applies low compute effort algorithms to minimize power through netlist optimizations that do not reduce design performance.	• Memory block optimization • Power-aware logic mapping
Extra effort	Besides the techniques in the **Normal compilation** setting, the Compiler applies high-compute-effort algorithms to minimize power through netlist optimizations. Selecting this option might impact performance.	• Memory block optimization • Power-aware logic mapping • Power-aware memory balance

Related Information

- Area-Driven Synthesis on page 38
- Power-Driven Fitter on page 38

2.4. Design Guidelines

During FPGA design implementation, you can apply the following design techniques to reduce power consumption. The results of these techniques are different from design to design.

2.4.1. Clock Power Management

Clocks represent a significant portion of dynamic power consumption due to their high switching activity and long paths. Actual clock-related power consumption is higher, because the power consumption of a block includes local clock distribution within logic, memory, and DSP or multiplier blocks.

The Intel Quartus Prime software optimizes clock routing power automatically, enabling only those portions of the clock network that are necessary to feed downstream registers.

2.4.1.1. Clock Gating

For designs containing logic that is not operational 100% of the time, you can reduce power consumption by gating (disabling) the clock that feeds that logic.

This solution requires that the logic in question have its own dedicated clock source. Clock duplication (such as introducing a duplicate PLL clock output) is necessary if the logic in question does not have its own dedicated clock source.

The following topics describe methods for gating clock networks.

2.4.1.1.1. Root Clock Gate

You can gate each clock network dynamically at the root level, using the Clock Control Intel FPGA IP Core.

Refer to the *Root Clock Gate* section of the *Clocking and PLL User Guide* for your Intel device.

2.4.1.1.2. Sector Clock Gate

You can gate each clock network dynamically at the clock sector level using the Clock Control Intel FPGA IP Core.

Refer to the *Sector Clock Gate* section of the *Clocking and PLL User Guide* for your Intel device.

2.4.1.1.3. I/O PLL Clock Gate

You can dynamically gate each of the device's I/O PLL output counters, using I/O PLL Reconfiguration.

Refer to the *I/O PLL Clock Gate* section of the *Clocking and PLL User Guide* for your Intel device.

2.4.1.2. Clock Enable in Memory Blocks

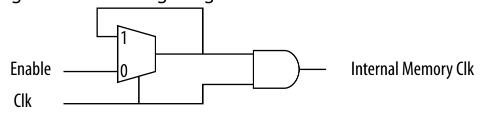
In memory blocks, power consumption is tied to the clock rate, and is insensitive to the toggle rate on the data and address lines. Memory consumes approximately 20% of the core dynamic power in typical designs.

When a memory block is clocked, a sequence of timed events occur within the block to execute a read or write. The circuitry that the clock controls consumes the same amount of power, independent of changes in address or data from one cycle to the next. Thus, the toggle rate of input data and the address bus have no impact on memory power consumption.

The key to reducing memory power consumption is to reduce the number of memory clocking events. You can achieve this reduction through network-wide clock gating, or on a per-memory basis through use of the clock enable signals on the memory ports.

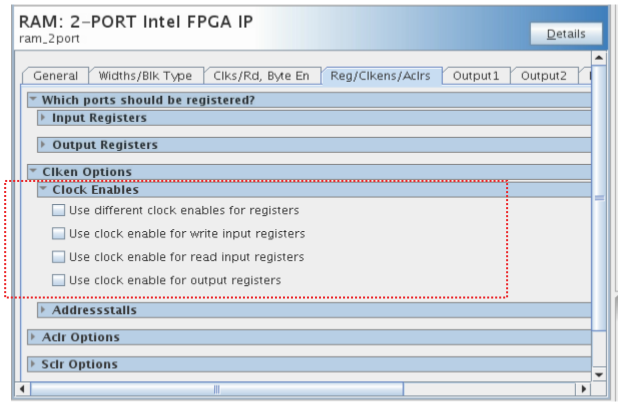
Figure 25. Memory Clock Enable Signal

Logical view of the internal clock of the memory block. Use the appropriate enable signals on the memory to make use of the clock enable signal instead of gating the clock.



The clock enable signal enables the memory only when necessary, and shuts down for the rest of the time, reducing the overall memory power consumption. You include these enable signals when generating the memory block function.

Figure 26. Clock Enable in RAM 2-Port



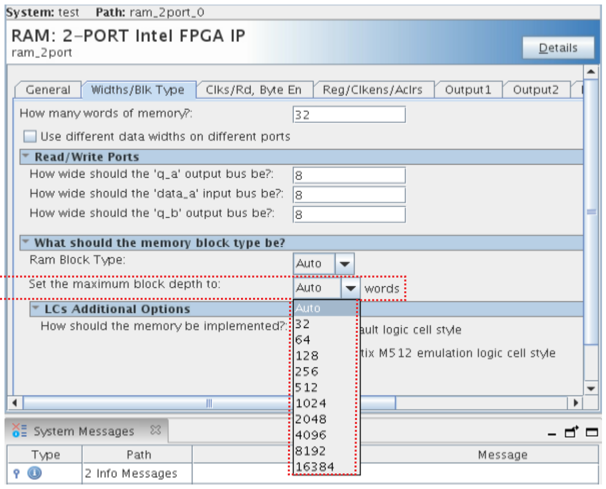
RAM: 2-PORT Intel FPGA IP
ram_2port
Details

General | Widths/Blk Type | Clks/Rd, Byte En | Reg/Clkens/Aclrs | Output1 | Output2

Which ports should be registered?
Input Registers
Output Registers
Clken Options
Clock Enables
☐ Use different clock enables for registers
☐ Use clock enable for write input registers
☐ Use clock enable for read input registers
☐ Use clock enable for output registers
Addressstalls
Aclr Options
Sclr Options

The Intel Quartus Prime software automatically chooses the best design memory configuration for optimal power. However, you can set the **MAXIMUM_DEPTH** parameter for memory modules during the IP core instantiation.

Figure 27. RAM 2-Port Maximum Depth



System: test Path: ram_2port_0
RAM: 2-PORT Intel FPGA IP
ram_2port
Details

General | Widths/Blk Type | Clks/Rd, Byte En | Reg/Clkens/Aclrs | Output1 | Output2

How many words of memory?: 32
☐ Use different data widths on different ports
Read/Write Ports
How wide should the 'q_a' output bus be?: 8
How wide should the 'data_a' input bus be?: 8
How wide should the 'q_b' output bus be?: 8
What should the memory block type be?
Ram Block Type: Auto
Set the maximum block depth to: Auto words
Auto / 32 / 64 / 128 / 256 / 512 / 1024 / 2048 / 4096 / 8192 / 16384
LCs Additional Options
How should the memory be implemented?: ...ault logic cell style / ...tix M512 emulation logic cell style

System Messages
Type | Path | Message
2 Info Messages

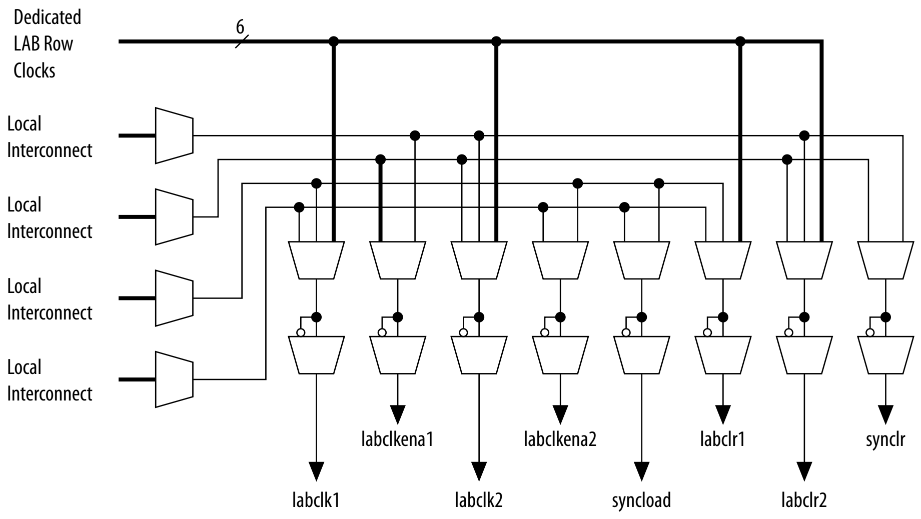
Related Information

- Power-Driven Compilation on page 35
- Clock Power Management on page 41
- Clocking Modes and Clock Enable
  In *Embedded Memory (RAM: 1-PORT, RAM: 2-PORT, ROM: 1-PORT, and ROM: 2-PORT) User Guide*

2.4.1.3. LAB Clock Power

Another contributor to clock power consumption are LAB clocks, which distribute clock to the registers within a LAB. LAB clock power can be the dominant contributor to overall clock power.

Figure 28. LAB-Wide Control Signals



To reduce LAB-wide clock power consumption without disabling the entire clock tree, use the LAB-wide clock enable to gate the LAB-wide clock. The Intel Quartus Prime software automatically promotes register-level clock enable signals to the LAB-level. A shared gated clock controls all registers within an LAB that share a common clock and clock enable. To take advantage of these clock enables, use a clock enable construct in the relevant HDL code for the registered logic.

2.4.1.3.1. LAB-Wide Clock Enable Example

This VHDL code makes use of a LAB-wide clock enable. This clock-gating logic is automatically turned into an LAB-level clock enable signal.

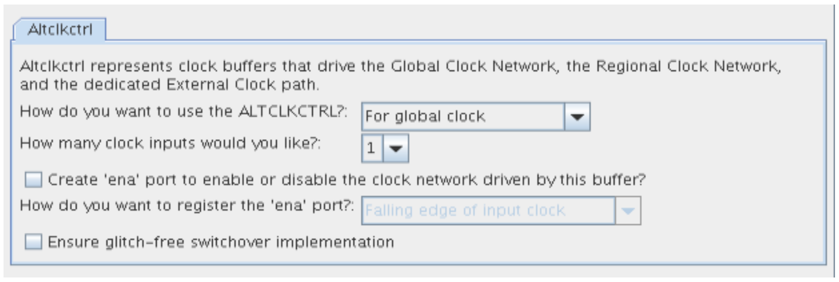
```
IF clk'event AND clock = '1' THEN
    IF logic_is_enabled = '1' THEN
        reg <= value;
    ELSE
        reg <= reg;
    END IF;
END IF;
```

2.4.1.4. Clock Enables

Use clock enables instead of gated clocks:

```
assign clk_gate = clk1 & gateA & gateB;
always @ (posedge clk_gate) begin
   sr[N-1:1] <= sr[N-2:0];
   sr[0]<=din1;
end
```

```
assign enable = gateA & gateB;
always @(posedge clk2) begin
   if (enable) begin
      sr[N-1:1] <= sr[N-2:0];
      sr[0]<=din2;
   end
end
```


Altclkctrl

Altclkctrl represents clock buffers that drive the Global Clock Network, the Regional Clock Network, and the dedicated External Clock path.

How do you want to use the ALTCLKCTRL?: For global clock

How many clock inputs would you like?: 1

☐ Create 'ena' port to enable or disable the clock network driven by this buffer?

How do you want to register the 'ena' port?: Falling edge of input clock

☐ Ensure glitch-free switchover implementation

Reduce LAB-wide clock power consumption without disabling the entire clock tree, use the LAB-wide clock enable to gate the LAB-wide clock.

```
always @(posedge clk)
begin
   if (ena)
      temp <= dataa;
   else
      temp <= temp;
   end
end
```

2.4.1.5. Global Signals

Intel FPGAs have different kinds of global signal resources available. Global signals can span the entire chip or smaller regions. Choose the clock networks that can cover all the fanout on a specific domain. For example, you can reduce clock power by switching from a clock network that spans the entire chip to one quarter of the chip, provided all the fanout for that clock is within that region of the chip.

2.4.1.5.1. Viewing Clock Details in the Chip Planner

1. Open the Chip Planner (**Tools ➤ Chip Planner**).
2. In the **Task** pane, under **Clock Reports**, double-click **Report Clock Details**.

Figure 31. Chip Planner Task Pane

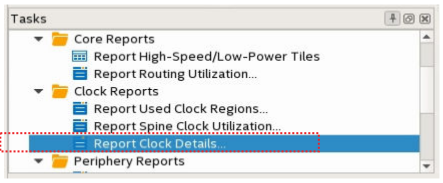
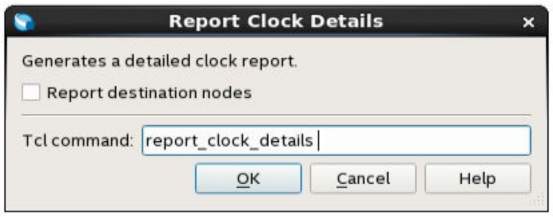

Tasks
Core Reports
Report High-Speed/Low-Power Tiles
Report Routing Utilization...
Clock Reports
Report Used Clock Regions...
Report Spine Clock Utilization...
Report Clock Details...
Periphery Reports

Figure 32. Report Clock Details


Report Clock Details

Generates a detailed clock report.

☐ Report destination nodes

Tcl command: report_clock_details

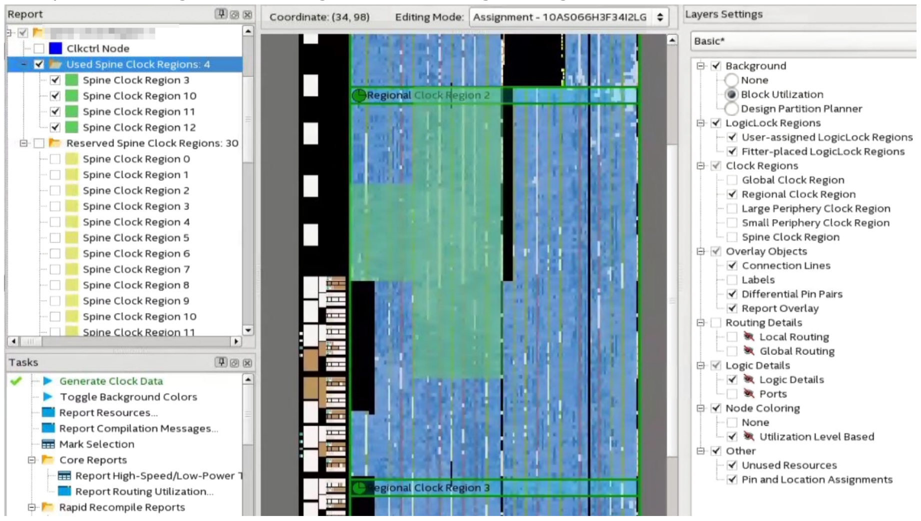
3. Click **OK**.

The **Report** pane generates the **Clock** folder.

4. Expand the **Clock** folder and select **Used spine clock regions** to highlight on the Chip planner.
5. In the **Layers Settings** pane, turn on **Regional/Periphery clock region** to see whether used spine clock regions are within.

Figure 33. Clock Highlight in Chip Planner

This example uses a Regional clock Region instead of a global signal.



2.4.1.6. Merge Clocks

Evaluate the possibility of merging clocks and PLLs in the design.

Design	2clks & 2PLLs	1 Clk & 1 PLL
Oc_dma_stamp25	6.079W	5.46W

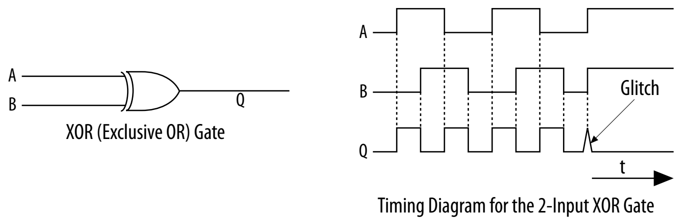
- 2clks & 2PLLs
  Clk1:350Mhz, Fanout 46788
  Clk2: 365Mhz, Fanout 2450
- 1Clk & 1PLL
  Merge clks
  clk: 365Mhz, Fanout 51277

2.4.2. Pipelining and Retiming

Glitches are unnecessary and unpredictable temporary logic switches at the output of combinational logic. Designs with glitches consume more power, because of faster switching activity. A glitch usually occurs when there is a mismatch in input signal timing, leading to unequal propagation delay.

For example, consider a 2-input XOR gate where one input changes from `1` to `0`, and moments later the other input changes from `0` to `1`. For a short time, both inputs become `1` (high), resulting in `0` (low) at the output of the XOR gate. Then, when the second input transition takes place, the XOR gate output becomes `1` (high). Therefore, before the output becomes stable, the input delay produces a glitch in the output.

Figure 34. XOR Gate Showing Glitch at the Output

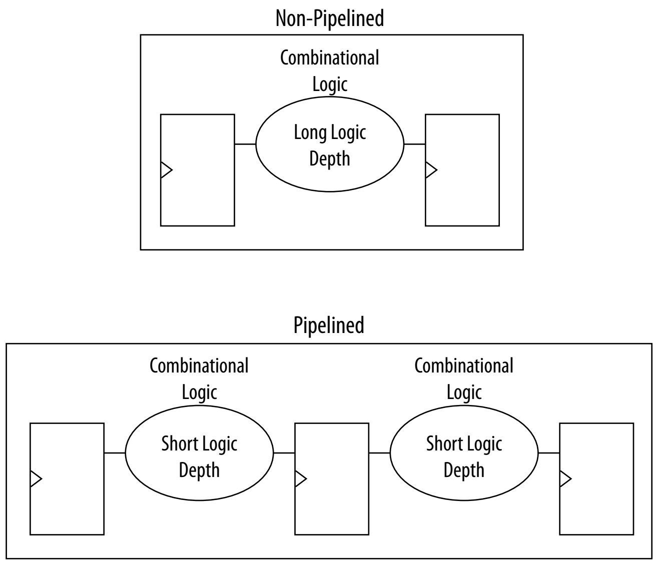


A glitch can propagate to subsequent logic and create unnecessary switching activity, increasing power consumption. Circuits with many XOR functions, such as arithmetic circuits or cyclic redundancy check (CRC) circuits, tend to have many glitches if there are several levels of combinational logic between registers.

Registers stop glitches from propagating through combinational paths. Pipelining is a technique that breaks combinational paths by inserting registers. By reducing logic-level numbers between registers, pipelining can result in higher clock speed operations. However, pipelining increases the latency of a circuit in terms of the number of clock cycles to a first result.

The following figure shows how pipelining breaks a long combinational path.

Figure 35. Pipelining Example



This reduction in switching activity lowers power dissipation in combinational logic. However, for designs with few glitches, pipelining can increase power consumption by adding unnecessary registers. Pipelining can also increase resource utilization.

2.4.3. Architectural Optimization

Design-level architectural optimizations allow you to take advantage of device architecture features. These features include dedicated memory, DSPs, or multiplier blocks that can perform memory or arithmetic-related functions. You can reduce power consumption by choosing blocks in place of LUTs. For example, you can build large shift registers from RAM-based FIFO buffers instead of building the shift registers from the LE registers.

Related Information

Timing Closure and Optimization
In *Intel Quartus Prime Pro Edition User Guide: Design Optimization*

2.4.4. I/O Power Guidelines

The Power Analyzer calculates I/O power using the default capacitive load set for the I/O standard in the **Capacitive Loading** page of the **Device and Pin Options** dialog box. Any other components defined in the board trace model are not taken into account for the power measurement.

Nonterminated I/O Standards

Nonterminated I/O standards such as LVTTL and LVCMOS have a rail-to-rail output swing. The voltage difference between logic-high and logic-low signals at the output pin is equal to the V_{CCIO} supply voltage. If the capacitive loading at the output pin is known, the following expression determines the dynamic power consumed in the I/O buffer:

$$P = \frac{F \cdot C \cdot V^{2}}{2}$$

where:

- F is the output transition frequency
- C is the total load capacitance being switched
- V is equal to V_{CCIO} supply voltage

Because of the quadratic dependence on V_{CCIO}, lower voltage standards consume significantly less dynamic power.

Transistor-to-transistor logic (TTL) I/O buffers consume very little static power. As a result, the total power that a LVTTL or LVCMOS output consumes is highly dependent on load and switching frequency.

Resistively Terminated I/O Standards

In resistively terminated I/O standards like SSTL and HSTL, the output load voltage swings by a small amount around a bias point. The dynamic power equation above is valid as well, but V is the actual load voltage swing. This voltage is much smaller than V_{CCIO}, resulting in lower dynamic power when comparing to nonterminated I/O under similar conditions.

Resistively terminated I/O standards dissipate significant static (frequency-independent) power, because the I/O buffer is constantly driving current into the resistive termination network. However, the lower dynamic power of these I/O standards means they often have lower total power than LVCMOS or LVTTL for high-frequency applications. As a best practice, when using resistively terminated standards choose the lowest drive strength I/O setting that meets the speed and waveform requirements to minimize I/O power.

You can save a small amount of static power by connecting unused I/O banks to the lowest possible V_{CCIO} voltage.

Related Information

Managing Device I/O Pins
In *Intel Quartus Prime Pro Edition User Guide: Design Constraints*

2.4.5. Dynamically Controlled On-Chip Terminations (OCT)

Dynamic OCT enables series termination (RS) and parallel termination (RT) to dynamically turn on/off during the data transfer. This feature is especially useful in FPGAs with external memory interfaces, such as interfacing with DDR memories.

Dynamic OCT eliminates the constant DC power that parallel termination consumes when transmitting data, reducing power consumption when compared to conventional termination. Parallel termination is extremely useful for applications that interface with external memories where I/O standards, such as HSTL and SSTL, are used. Parallel termination supports dynamic OCT, which is useful for bidirectional interfaces.

For more information about dynamic OCT in specific devices, refer to the *Intel Stratix 10 General Purpose I/O User Guide* or the *Intel Arria 10 Core Fabric and General Purpose I/O Handbook*.

Example 1. Example: Power Saving for a DDR3 Interface with OCT

The static current consumed by parallel OCT is equal to the V_{CCIO} voltage divided by 100 W. For DDR3 interfaces with SSTL-15, the static current per pin is:

$$\frac{1.5V}{100W} = 15mA$$

Therefore, the static power is:

$$1.5V \times 15mA = 22.5mW$$

For an interface with 72 DQ and 18 DQS pins, the static power is:

$$90pins \times 2.25mW = 2.025W$$

Dynamic parallel OCT disables parallel termination during write operations, so if writing occurs 50% of the time, the power saved by dynamic parallel OCT is:

$$50\% \times 2.025W = 1.0125W$$

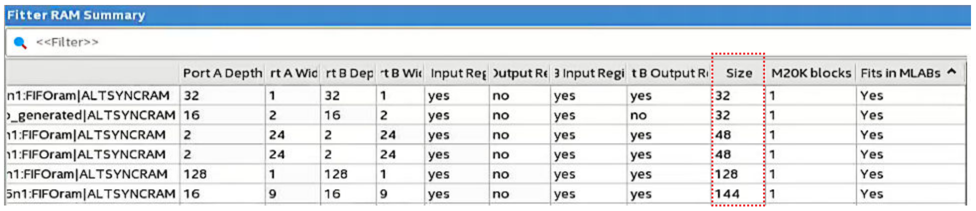
For more information about dynamic OCT in Stratix IV devices, refer to the chapter in the *Stratix IV Device Handbook*.

Related Information

- Dynamic OCT
  In *Intel Arria 10 Core Fabric and General Purpose I/O Handbook*
- Dynamic OCT
  In *Intel Stratix 10 General Purpose I/O User Guide*

2.4.6. Memory Optimization (M20K/MLAB)

M20K memory blocks represent a big part of the power consumption in a design. The Fitter RAM Summary Report displays the utilization of the memory blocks in different parts of the design.



Fitter RAM Summary

<<Filter>>

	Port A Depth	rt A Wid	rt B Dep	rt B Wid	Input Reg	Output Re	B Input Regi	t B Output R	Size	M20K blocks	Fits in MLABs
n1:FIFOram|ALTSYNCRAM	32	1	32	1	yes	no	yes	yes	32	1	Yes
_generated|ALTSYNCRAM	16	2	16	2	yes	no	yes	no	32	1	Yes
1:FIFOram|ALTSYNCRAM	2	24	2	24	yes	no	yes	yes	48	1	Yes
1:FIFOram|ALTSYNCRAM	2	24	2	24	yes	no	yes	yes	48	1	Yes
1:FIFOram|ALTSYNCRAM	128	1	128	1	yes	no	yes	yes	128	1	Yes
5n1:FIFOram|ALTSYNCRAM	16	9	16	9	yes	no	yes	yes	144	1	Yes

Figure 36. Fitter RAM Summary Report

Some guidelines to optimize the use of memories are:

- Port shallow memories from M20K to MLAB.

  For example, implement in HDL with `ramstyle` attribute:

  ```
  (* ramstyle = "MLAB" *) reg [0:7] my_ram[0:63];
  ```

- Avoid read-during-write behavior and set to **Don't care** (at the HDL level) wherever possible.

  Read-during-write behavior impact the power of single-port and bidirectional dual-port RAMs. **Don't care** allows an optimization that sets the read-enable signal to the inversion of the existing write-enable signal (if one exists). This allows the core of the RAM to shut down, which prevents switching, saving a significant amount of power.

- Pack input/output registers in M20K.

2.4.6.1. Implementation

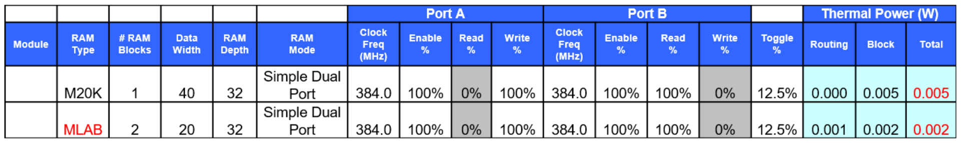
Table 12. Single-port Embedded Memory Configurations for Intel Arria 10 Devices

This table lists the maximum configurations that single-port RAM and ROM modes support.

Memory Block	Depth (bits)	Programmable Width
MLAB	32	x16, x18, or x20
	64 [3]	x8, x9, x10
M20K	512	x40, x32
	1K	x20, x16
	2K	x10, x8
	4K	x5, x4
	8K	x2
	16K	x1

[3] Supported through software emulation and consumes additional MLAB blocks.

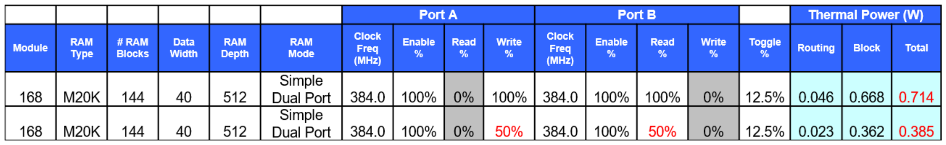
Figure 37. Power numbers from EPE



						Port A				Port B					Thermal Power (W)		
Module	RAM Type	# RAM Blocks	Data Width	RAM Depth	RAM Mode	Clock Freq (MHz)	Enable %	Read %	Write %	Clock Freq (MHz)	Enable %	Read %	Write %	Toggle %	Routing	Block	Total
	M20K	1	40	32	Simple Dual Port	384.0	100%	0%	100%	384.0	100%	100%	0%	12.5%	0.000	0.005	0.005
	MLAB	2	20	32	Simple Dual Port	384.0	100%	0%	100%	384.0	100%	100%	0%	12.5%	0.001	0.002	0.002

2.4.6.2. Rd/Wr Enables

Dedicated RAM blocks dissipate most energy whenever the RAM is accessed for a read or write cycle. You can save power by adding Read/Write enable.



						Port A				Port B					Thermal Power (W)		
Module	RAM Type	# RAM Blocks	Data Width	RAM Depth	RAM Mode	Clock Freq (MHz)	Enable %	Read %	Write %	Clock Freq (MHz)	Enable %	Read %	Write %	Toggle %	Routing	Block	Total
168	M20K	144	40	512	Simple Dual Port	384.0	100%	0%	100%	384.0	100%	100%	0%	12.5%	0.046	0.668	0.714
168	M20K	144	40	512	Simple Dual Port	384.0	100%	0%	50%	384.0	100%	50%	0%	12.5%	0.023	0.362	0.385

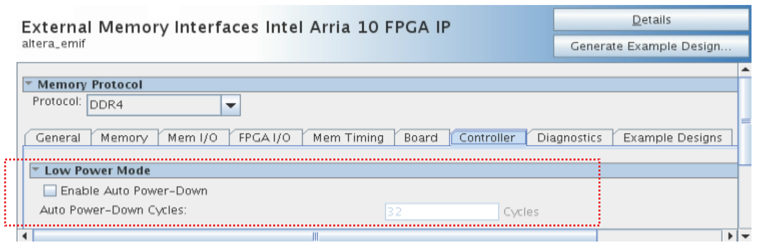
2.4.7. DDR Memory Controller Settings

The External Memory Interfaces Intel Arria 10 FPGA IP provides low power mode settings. These settings put DDR memory in power saving mode when the controller is idle, providing power savings on external memory DDR. The **Enable Auto Power-Down** and **Auto Power-Down Cycles** settings enable this capability.

Low Power Mode Settings

- **Enable Auto Power-Down**—directs the controller to place the memory device in power-down mode after a specific number of idle controller clock cycles. You can configure the idle wait time. All ranks must be idle to enter auto power-down.
- **Auto Power-Down Cycles**—specifies the number of cycles the controller must be IDLE before entering the power down state. You determine the number based on the traffic pattern. If the number is too small, the control enters power down too frequently, affecting efficiency. The Intel Arria 10 device family supports from 1 to 65534 cycles.

Figure 38. Intel Arria 10 EMIF Controller Parameters



External Memory Interfaces Intel Arria 10 FPGA IP
altera_emif

Memory Protocol
Protocol: DDR4

Low Power Mode
Enable Auto Power-Down
Auto Power-Down Cycles: 32 Cycles

Related Information

Intel Arria 10 EMIF IP DDR3 Parameters: Controller
In *External Memory Interfaces Intel Arria 10 FPGA IP User Guide*

2.4.8. DSP Implementation

When you maximize the packing of DSP blocks, you reduce Logic Utilization, power consumption, and increase efficiency. The HDL coding style grants you control of the DSP resources available in the FPGA.

Example 2. Implement Multiplier + Accumulator in 1 DSP

```
always @ (posedge clk)
begin
   if (ena)
   begin
      dataout <= dataa * datab + datac * datad;
   end
end
```

Example 3. Implement multiplication in 2 DSPs and the adder in LABs

```
always @ (posedge clk)
begin
   if (ena)
   begin
      mult1 <= dataa * datab;
      mult2 <= datac * datad;
   end
end
always @(posedge clk)
begin
   if (ena)
   begin
      dataout <= mult1 + mult2
   end
end
```

Related Information

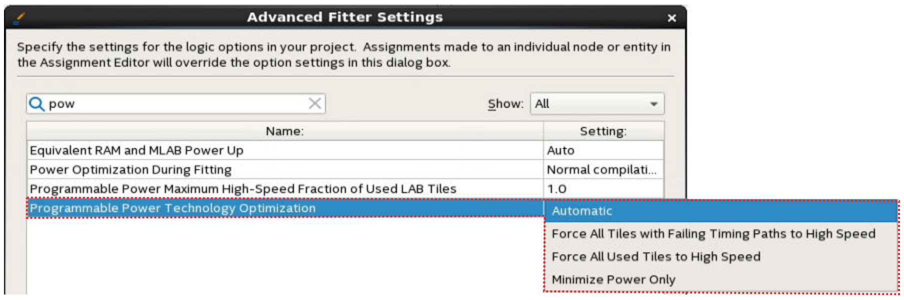
Inferring Multipliers and DSP Functions
In *Intel Quartus Prime Pro Edition User Guide: Design Recommendations*

2.4.9. Reducing High-Speed Tile (HST) Usage

High-Speed tiles are available in the Intel Arria 10 design family.

1. In the **Advanced Fitter Settings** pane, The **Programmable Power Technology Optimization** logic option controls how the fitter configures tiles to operate in high-speed mode or low-power mode. Select **Minimize Power Only**.

Figure 39. Programmable Power Technology Optimization


Advanced Fitter Settings

Specify the settings for the logic options in your project. Assignments made to an individual node or entity in the Assignment Editor will override the option settings in this dialog box.

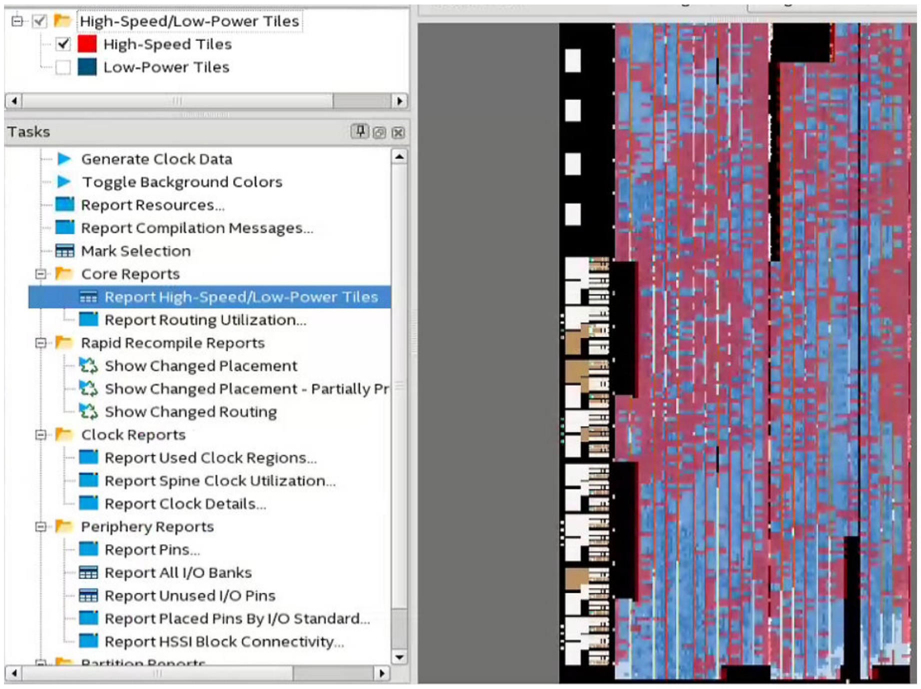
pow   Show: All

Name:	Setting:
Equivalent RAM and MLAB Power Up	Auto
Power Optimization During Fitting	Normal compilati...
Programmable Power Maximum High-Speed Fraction of Used LAB Tiles	1.0
Programmable Power Technology Optimization	Automatic

Automatic
Force All Tiles with Failing Timing Paths to High Speed
Force All Used Tiles to High Speed
Minimize Power Only

2. Identify entity modules that use HST by plotting entity modules and HST heatmap on the Chip Planner and modify the floorplan to reduce usage.

Figure 40. Entity Modules and HST Heatmap on the Chip Planner



2.4.10. Unused Transceiver Channels

Transceivers in the device degrade over time unless you preserve them. The Intel Quartus Prime software generates a warning message if a design contains unused XCVRs.

You do not need to preserve transceivers under 8Gbps. For transceivers over 8Gbps, the best practice is to preserve if there is a possibility for future usage. Otherwise, you can turn the transceivers off. You enable unused transceivers through dynamic reconfiguration or a new device programming file.

2.4.11. Periphery Power reduction XCVR Settings

2.4.11.1. Transceiver Settings

- Use min VCCR/T possible (depending on data rate).
- Certain devices have DFE ON by default. If possible, turn off the channel, This depends on the how lossy is the channel.
- Turn off PDN compensation.
  This setting induces jitter, which is necessary to check system tolerance.
- Use one equalizer stage.

DFE	Adaptation	Equalizer Stage	Transmitter High-Speed Compensation
Disabled	Disabled	Non-S1 Mode	Disabled
Disabled	Disabled	Non-S1 Mode	Enabled
Disabled	Disabled	N/A	Enabled

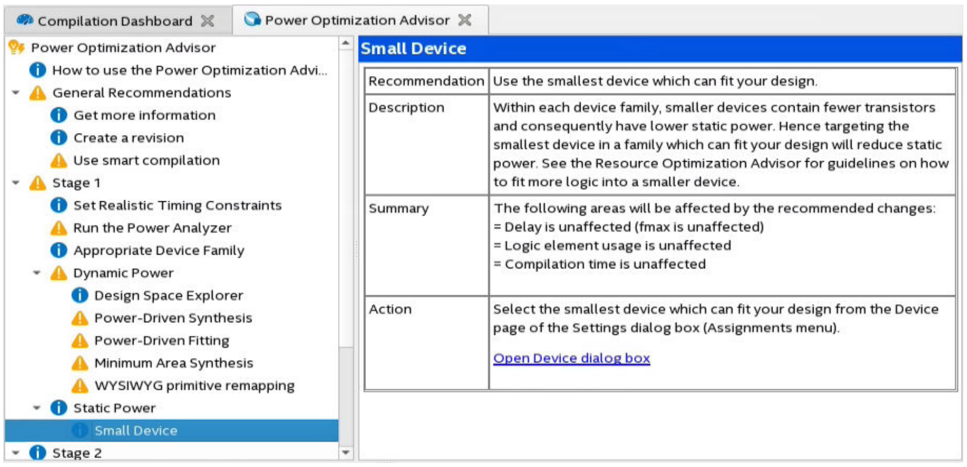
2.4.11.2. I/O Current Strength

As a best practice, choose a low voltage I/O standard and the lowest drive strength that meets the speed requirements.

2.5. Power Optimization Advisor

The Intel Quartus Prime Power Optimization Advisor provides advice and recommendations based on the current design project settings and assignments. You run the Advisor after the Power Analyzer.

Figure 41. Power Optimization Advisor



Small Device

Recommendation	Use the smallest device which can fit your design.
Description	Within each device family, smaller devices contain fewer transistors and consequently have lower static power. Hence targeting the smallest device in a family which can fit your design will reduce static power. See the Resource Optimization Advisor for guidelines on how to fit more logic into a smaller device.
Summary	The following areas will be affected by the recommended changes: = Delay is unaffected (fmax is unaffected) = Logic element usage is unaffected = Compilation time is unaffected
Action	Select the smallest device which can fit your design from the Device page of the Settings dialog box (Assignments menu). Open Device dialog box

The Power Optimization Advisor organizes the recommendations into stages that suggest the implementation order. Each recommendation includes a description, summary of the effect of the recommendation, and the action required to make the appropriate setting.

An icon indicates whether each recommended setting is made in the current project. Checkmark icons appear next to recommendations that are already implemented, warning icons appear next to recommendations that are not followed for this compilation. Information icons indicate general suggestions.

Recommendations include a link to the location in the Intel Quartus Prime GUI where you can change the setting. After implementing the recommended changes, recompile your design. You can verify power results with the Power Analyzer.

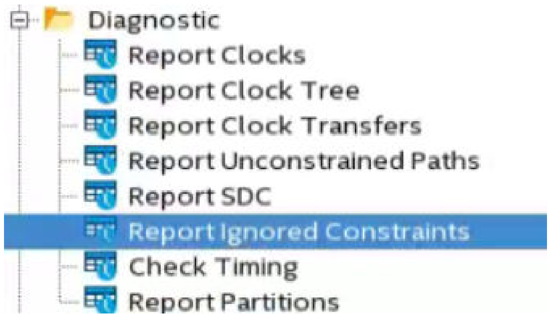
2.5.1. Set Realistic Timing Constraints

Timing requirements are too high, the Compiler increases HST Usage. In addition, the Fitter efforts focus more in timing than power optimization.

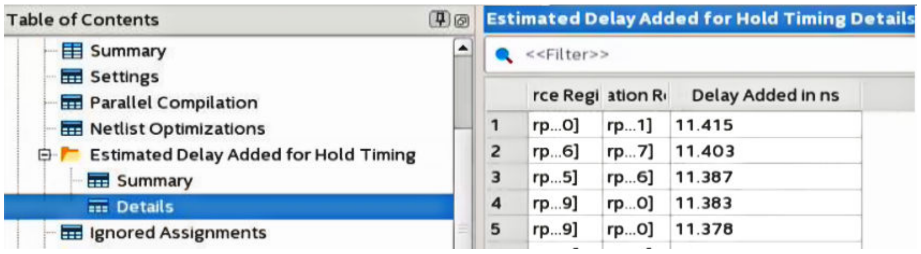
2.5.1.1. Find Timing Information

- To find False or Multi-Cycle Paths, click **Report Ignored Constraints** in the Timing Analyzer **Tasks** pane.

Figure 42. Report Ignored Constraints



- To see a list of the 10 paths with highest delay in the design, in the **Reports** pane find **Fitter Summary Report ➤ Estimate Delay Added for Hold Timing ➤ Details**.



	rce Regi	ation R	Delay Added in ns
1	rp...0]	rp...1]	11.415
2	rp...6]	rp...7]	11.403
3	rp...5]	rp...6]	11.387
4	rp...9]	rp...0]	11.383
5	rp...9]	rp...0]	11.378

2.5.2. Appropriate Device Family

Choose a device family with the dynamic and static power characteristics best suited to your application.

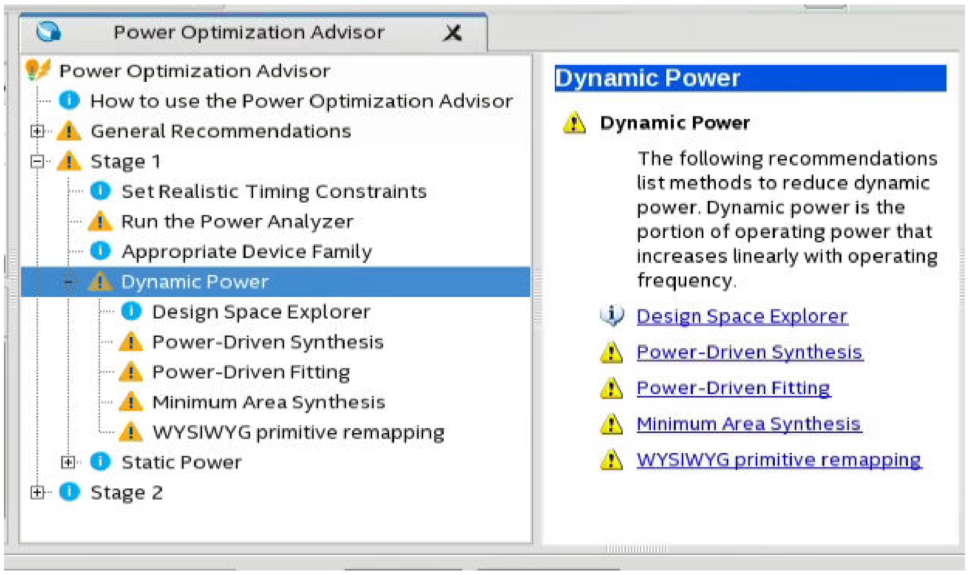
Related Information

Device Selection on page 32

2.5.3. Dynamic Power

The recommendations in this section can reduce dynamic power.

Figure 43. Dynamic Power Recommendations in the Power Optimization Advisor



The following recommendations list methods to reduce dynamic power. Dynamic power is the portion of operating power that increases linearly with operating frequency.

Related Information

- Design Space Explorer II for Power-Driven Optimization on page 35
- Power-Driven Synthesis on page 35
- Power-Driven Fitter on page 38
- Area-Driven Synthesis on page 38

2.5.4. Static Power

The recommendations in this section can reduce static power dissipation. Static power is the frequency independent power that a design dissipates, even when the design clocks are stopped.

Small Device

Use the smallest device which can fit your design.

Related Information

Device Selection on page 32

2.5.5. Appropriate I/O Standards

Choose appropriate I/O Standards to minimize design power.

Related Information

I/O Power Guidelines on page 48

2.5.6. Use RAM Blocks

Implement RAMs and medium to large shift registers in RAM blocks instead of logic cell registers.

Related Information

Memory Optimization (M20K/MLAB) on page 50

2.5.7. Shut Down RAM Blocks

Use the clock enable, read enable and write enable ports on RAM blocks to shut them down during cycles in which the RAM is not read or written. If your design does not depend on a specific read result when reading and writing the same address, then specify "don't care" for the read-during-write parameter in the RAM IP Catalog.

Related Information

- Clock Enable in Memory Blocks on page 42
- Memory Optimization (M20K/MLAB) on page 50

2.5.8. Clock Enables on Logic

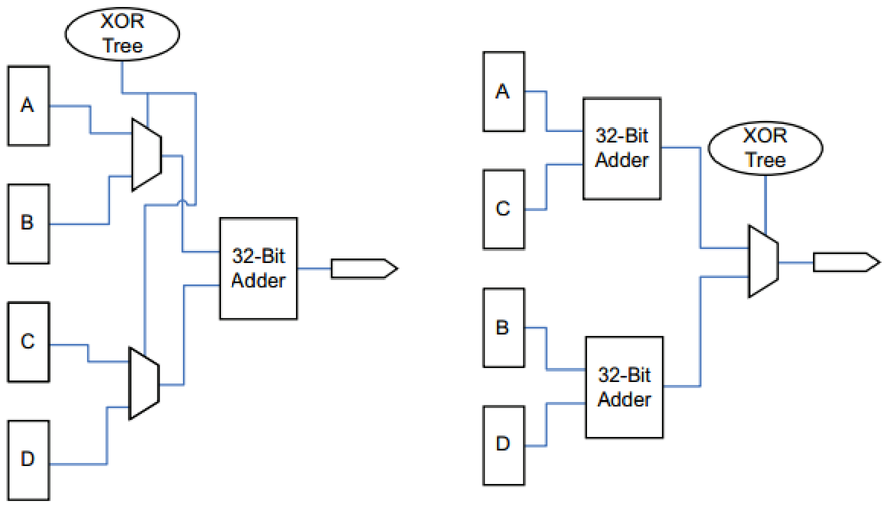
Another technique for power reduction is gating clocks when the logic does not require them. Even though you can build clock-gating logic, this approach can generate clock glitches in FPGAs using ALMs or LEs.

2.5.9. Pipeline Logic to Reduce Glitching

Long chains of cascaded logic blocks can create glitches due to path delay differences between the input signals. Inserting Flip-Flops to cut these long chains terminates the propagation of glitches to consecutive logic cells.

Circuits that heavily use of XIO functions (for example, Cyclic redundancy check) tend to glitch significantly when cascaded. Add pipeline registers or re-architect to reduce signal toggling.

Example 4. Glitch Prone Design



Related Information

Pipelining and Retiming on page 46

2.6. Power Optimization Revision History

The following revision history applies to this chapter:

Table 13. Document Revision History

Document Version	Intel Quartus Prime Version	Changes
2022.01.17	21.4	Added *Clock Gating* topics in the *Design Guidelines* section.
2021.10.04	21.3	Modified the *Compiler Settings* topic.
2020.12.07	18.1.0	Added note to the *Toggle Rate* topic.
2019.08.02	18.1.0	• Corrected typo in "Viewing Clock Details in the Chip Planner" topic. • Corrected typo in "Pipelining and Retiming" topic. • Corrected typo in "Implementation" topic. • Updated "DDR Memory Controller Settings" topic for latest IP name and to correct typos. • Corrected typo in "Pipeline Logic to Reduce Glitching" topic.
2018.09.24	18.1.0	• Added topic: *Factors Affecting Power Consumption*, moved from chapter: *Power Analysis* • Extended content about Power Optimization Advisor with a description of recommendations. • Added design guidelines: *Memories (M20K/MLAB)*, *DDR Memory Controller Settings*, *DSP Implementation*, *Reducing High-Speed Tile (HST) Usage*, *Unused Transceiver Channels*, *Periphery Power reduction XCVR Settings* • Removed content referring to device families not supported in *Intel Quartus Prime Pro Edition*.
2018.06.11	18.0.0	• Moved general information about the Design Space Explorer (DSE II) to the *Design Optimization User Guide*, left a section about using DSE II for Power-Driven Optimization.
2018.05.07	18.0.0	• Moved general information about the Design Space Explorer (DSE II) to the *Design Optimization User Guide*, left a section about using DSE II for Power-Driven Optimization.
2016.10.31	16.1.0	• Implemented Intel rebranding. • Removed statement of support for gate-level timing simulation.
2015.11.02	15.1.0	Changed instances of *Quartus II* to *Intel Quartus Prime*. • Updated screenshot for DSE II GUI. • Added information about remote hosts for DSE II.
2014.12.15	14.1.0	• Updated location of Fitter Settings, Analysis & Synthesis Settings, and Physical Synthesis Optimizations to Compiler Settings. • Updated DSE II GUI and optimization settings.
2014.06.30	14.0.0	Updated the format.
May 2013	13.0.0	Added a note to "Memory Power Reduction Example" on Qsys and SOPC Builder power savings limitation for on-chip memory block.
June 2012	12.0.0	Removed survey link.
November 2011	10.0.2	Template update.
December 2010	10.0.1	Template update.
July 2010	10.0.0	• Was chapter 11 in the 9.1.0 release • Updated Figures 14-2, 14-3, 14-6, 14-18, 14-19, and 14-20 • Updated device support • Minor editorial updates

continued...

Document Version	Intel Quartus Prime Version	Changes
November 2009	9.1.0	• Updated Figure 11-1 and associated references • Updated device support • Minor editorial update
March 2009	9.0.0	• Was chapter 9 in the 8.1.0 release • Updated for the Quartus II software release • Added benchmark results • Removed several sections • Updated Figure 13–1, Figure 13–17, and Figure 13–18
November 2008	8.1.0	• Changed to 8½” × 11” page size • Changed references to altsyncram to RAM • Minor editorial updates
May 2008	8.0.0	• Added support for Stratix IV devices • Updated Table 9–1 and 9–9 • Updated “Architectural Optimization” on page 9–22 • Added “Dynamically-Controlled On-Chip Terminations” on page 9–26 • Updated “Referenced Documents” on page 9–29 • Updated references

3. Power Analysis and Optimization Document Archive

For the latest and previous versions of this user guide, refer to *Intel Quartus Prime Pro Edition User Guide: Power Analysis and Optimization*. If a software version is not listed, the guide for the previous software version applies.

A. Intel Quartus Prime Pro Edition User Guides

Refer to the following user guides for comprehensive information on all phases of the Intel Quartus Prime Pro Edition FPGA design flow.

Related Information

- Intel Quartus Prime Pro Edition User Guide: Getting Started
  Introduces the basic features, files, and design flow of the Intel Quartus Prime Pro Edition software, including managing Intel Quartus Prime Pro Edition projects and IP, initial design planning considerations, and project migration from previous software versions.
- Intel Quartus Prime Pro Edition User Guide: Platform Designer
  Describes creating and optimizing systems using Platform Designer, a system integration tool that simplifies integrating customized IP cores in your project. Platform Designer automatically generates interconnect logic to connect intellectual property (IP) functions and subsystems.
- Intel Quartus Prime Pro Edition User Guide: Design Recommendations
  Describes best design practices for designing FPGAs with the Intel Quartus Prime Pro Edition software. HDL coding styles and synchronous design practices can significantly impact design performance. Following recommended HDL coding styles ensures that Intel Quartus Prime Pro Edition synthesis optimally implements your design in hardware.
- Intel Quartus Prime Pro Edition User Guide: Design Compilation
  Describes set up, running, and optimization for all stages of the Intel Quartus Prime Pro Edition Compiler. The Compiler synthesizes, places, and routes your design before generating a device programming file.
- Intel Quartus Prime Pro Edition User Guide: Design Optimization
  Describes Intel Quartus Prime Pro Edition settings, tools, and techniques that you can use to achieve the highest design performance in Intel FPGAs. Techniques include optimizing the design netlist, addressing critical chains that limit retiming and timing closure, optimizing device resource usage, device floorplanning, and implementing engineering change orders (ECOs).
- Intel Quartus Prime Pro Edition User Guide: Programmer
  Describes operation of the Intel Quartus Prime Pro Edition Programmer, which allows you to configure Intel FPGA devices, and program CPLD and configuration devices, via connection with an Intel FPGA download cable.
- Intel Quartus Prime Pro Edition User Guide: Block-Based Design
  Describes block-based design flows, also known as modular or hierarchical design flows. These advanced flows enable preservation of design blocks (or logic that comprises a hierarchical design instance) within a project, and reuse of design blocks in other projects.

- Intel Quartus Prime Pro Edition User Guide: Partial Reconfiguration
  Describes Partial Reconfiguration, an advanced design flow that allows you to reconfigure a portion of the FPGA dynamically, while the remaining FPGA design continues to function. Define multiple personas for a particular design region, without impacting operation in other areas.
- Intel Quartus Prime Pro Edition User Guide: Third-party Simulation
  Describes RTL- and gate-level design simulation support for third-party simulation tools by Aldec*, Cadence*, Siemens EDA, and Synopsys* that allow you to verify design behavior before device programming. Includes simulator support, simulation flows, and simulating Intel FPGA IP.
- Intel Quartus Prime Pro Edition User Guide: Third-party Synthesis
  Describes support for optional synthesis of your design in third-party synthesis tools by Siemens EDA, and Synopsys*. Includes design flow steps, generated file descriptions, and synthesis guidelines.
- Intel Quartus Prime Pro Edition User Guide: Third-party Logic Equivalence Checking Tools
  Describes support for optional logic equivalence checking (LEC) of your design in third-party LEC tools by OneSpin*.
- Intel Quartus Prime Pro Edition User Guide: Debug Tools
  Describes a portfolio of Intel Quartus Prime Pro Edition in-system design debugging tools for real-time verification of your design. These tools provide visibility by routing (or "tapping") signals in your design to debugging logic. These tools include System Console, Signal Tap logic analyzer, system debugging toolkits, In-System Memory Content Editor, and In-System Sources and Probes Editor.
- Intel Quartus Prime Pro Edition User Guide: Timing Analyzer
  Explains basic static timing analysis principals and use of the Intel Quartus Prime Pro Edition Timing Analyzer, a powerful ASIC-style timing analysis tool that validates the timing performance of all logic in your design using an industry-standard constraint, analysis, and reporting methodology.
- Intel Quartus Prime Pro Edition User Guide: Power Analysis and Optimization
  Describes the Intel Quartus Prime Pro Edition Power Analysis tools that allow accurate estimation of device power consumption. Estimate the power consumption of a device to develop power budgets and design power supplies, voltage regulators, heat sink, and cooling systems.
- Intel Quartus Prime Pro Edition User Guide: Design Constraints
  Describes timing and logic constraints that influence how the Compiler implements your design, such as pin assignments, device options, logic options, and timing constraints. Use the Interface Planner to prototype interface implementations, plan clocks, and quickly define a legal device floorplan. Use the Pin Planner to visualize, modify, and validate all I/O assignments in a graphical representation of the target device.
- Intel Quartus Prime Pro Edition User Guide: PCB Design Tools
  Describes support for optional third-party PCB design tools by Siemens EDA and Cadence*. Also includes information about signal integrity analysis and simulations with HSPICE and IBIS Models.
- Intel Quartus Prime Pro Edition User Guide: Scripting
  Describes use of Tcl and command line scripts to control the Intel Quartus Prime Pro Edition software and to perform a wide range of functions, such as managing projects, specifying constraints, running compilation or timing analysis, or generating reports.