PLUNIFY

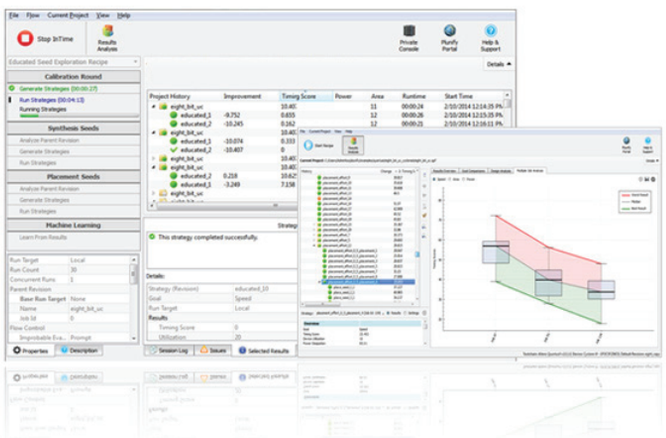
InTime

Datasheet

Overview

As an expert software that solves FPGA timing and optimization problems with massive compute power and machine learning, InTime has built-in intelligence to analyze an FPGA design and determine optimized strategies for synthesis and place-and-route. Based on knowledge of device, design and tool characteristics and proprietary algorithms, InTime generates settings and design constraints that have the highest chances of meeting design performance targets. InTime learns from every result and constantly improves its database.

By intelligently harnessing compute resources, InTime enables FPGA engineers to solve design problems and get to production silicon significantly faster than before.



Automatic generation of build strategies

InTime's signature database stores knowledge from over 40,000 CPU hours of benchmarking across multiple tool versions and device families. This enables InTime to tailor tool settings and constraints for specific design conditions.

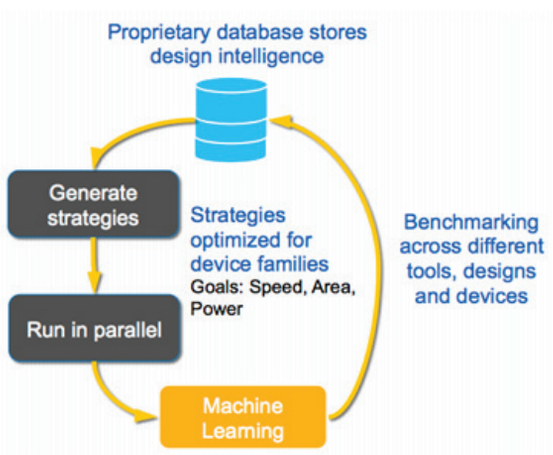


Figure 1: Signature database formation

Ease of use - just click "Start"

Just click "Start Recipe". InTime learns from your previous build results and improves them based on device, design and tool characteristics. With machine learning, the more builds InTime does, the smarter it gets.

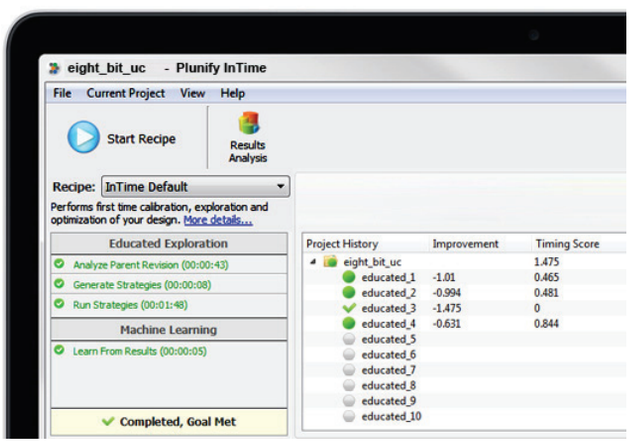


Figure 2: InTime Graphical User Interface

Automatically run builds in parallel

InTime can make use of available resources in different environments. The "Run Targets" setting enables builds to be run locally on a computer, distributed across multiple machines in your company's network. You can configure it to work with workload management software such as Load Sharing Facility (LSF) or Oracle/Sun Grid Engine (SGE).

Flexible build environments

In a multi-machine configuration, a designated InTime "control" server allocates machines for every "job" (consisting of one or more builds). InTime communicates with the "worker" machines, encrypting and transferring files/results, enabling builds to be run in parallel, taking advantage of available compute resources.

InTime recipes

A "Recipe" is a method used by InTime to optimize a design. Recipes can be grouped into four categories: 'Learning', 'Last-Mile', 'General' and 'Advanced'.

Those in the 'Learning' category are the recipes that Plunify recommends when the user runs designs that are new to InTime. These recipes are intended to perform initial analysis and calibration of design settings and constraints. For example, the InTime Default recipe tweaks the value of all the available compiler settings based on learning done on past results. Meanwhile, recipes in the 'Last-Mile' category perform more specific optimizations when results have been sufficiently improved by the Learning recipes. For example, the Placement Seed Exploration recipe changes the placement seed of compiler settings only. Last but not least, recipes under 'General' and 'Advanced' are more for usability purposes.

	Recipes	Quartus	Vivado	ISE	Descriptions
Learning	InTime Default	Y	Y	Y	Performs first time calibration, exploration and optimization of your design
	InTime Default Extra	Y	Y		Performs actions in Intime Default recipe, and run additional optimizations
	Hot Start	Y	Y*		Generates initial strategies for your design by correlating it with other designs in the InTime database
	Deep Dive	Y	Y	Y	Performs deeper analysis of result of your design
	Explorer		Y		Explores different optimization in your design
Last-Mile	Auto Placement	Y			Performs self-guided placement adjustments to improve the worst timing failure paths in your design.
	Effort Level Exploration	Y			Explores different effort level
	Placement Seed Exploration	Y		Y	Explores the effect of placement seeds on the design goal
	Seeded Effort Level Exploration	Y			Performs exploration of seeds on best effort level results
	Router Effort Level Exploration	Y			Explores different routing effort level
	Extra Opt Exploration		Y		Explores optimization that focus on placement, physical optimization and routing
	Placement Exploration		Y		Explores the effect of placement adjustment on the design
	Map Seed Exploration			Y	Explores the effect of map seeds on the design goal
General	Just Compile My Design	Y	Y	Y	Compiles the active revision in your project
	Rerun Strategies	Y	Y	Y	Rerun all marked strategies
Advanced	Custom Flow	Y	Y	Y	Uses strategies specified by the user to compile

**Coming Soon*

Automate InTime recipe flow

InTime allows users to control the execution of InTime recipes using Tcl scripting. For example, you can now make InTime execute the InTime Default recipe followed by the Explorer recipe and finally one or more of the Last-Mile recipes. InTime sets the parent revision for each recipe according to metrics that you can specify, for example, TNS.

```
# get worst setup slack right after placement
open_design "calibrate_3_placed"
# terminate this strategy if the slack is too high
if { [get_slack] < -2 } {
  kill "calibrate_3"
}
close_design
# move on to the next strategy
```

Figure 3: Sample Tcl Script

Advanced result analysis

InTime's result analytics enables users to compare multiple builds at a glance to find similarities between passing builds and pinpoint causes of failure. InTime visualizes and tracks data such as total number of critical paths, interconnects, and relationships between various design goals like Total Negative Slack (TNS), Worst Slack (WS), Area and Power.

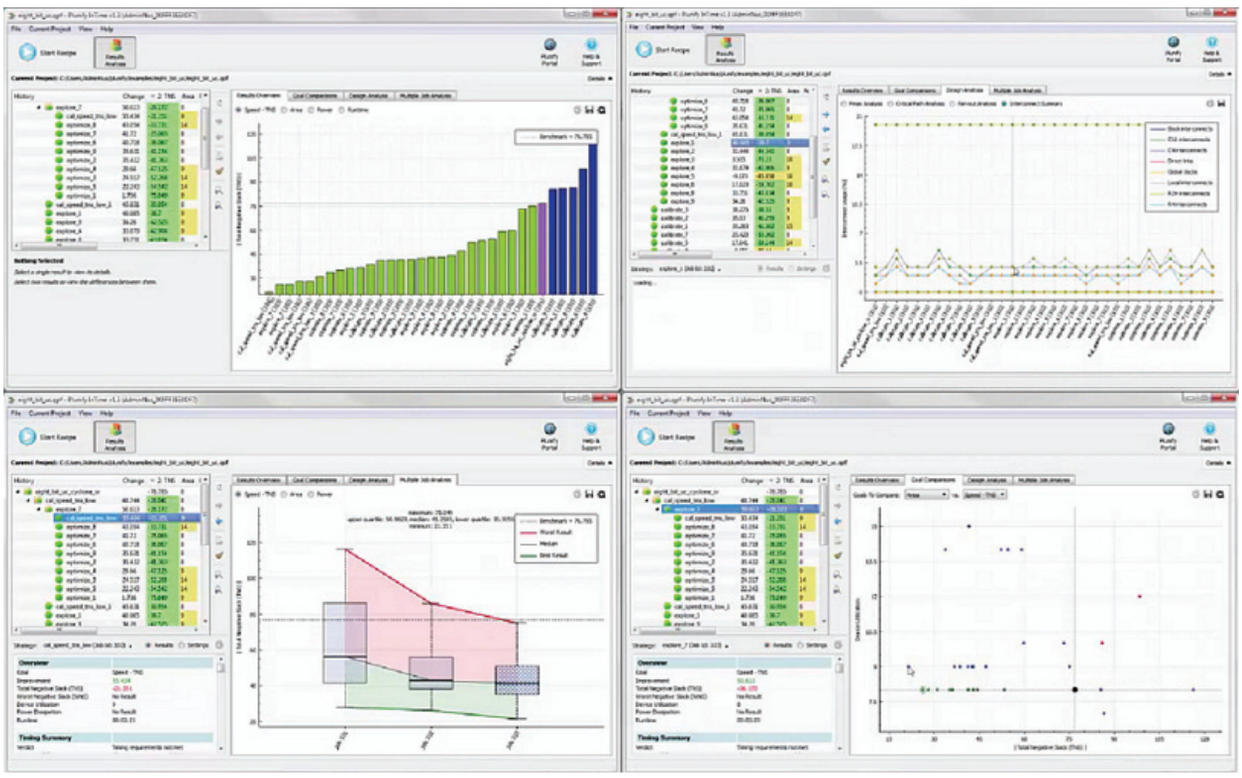


Figure 4: (clockwise from top left) TNS of different compilations, interconnects chart, machine learning improvements, TNS/WS distribution

System Requirements

InTime Server (License)	InTime Client
Minimum 2GB RAM At least 10GB of free disk space	Minimum 1GB RAM, with 4 GB+ virtual memory At least 200MB free disk space for InTime software At least 10GB free disk space for build files
Processor: Intel i3 CPU or similar	
OS: 64-bit Windows / Ubuntu 12.04 / RedHat Enterprise 5+ / other compatible Linux distributions	
Java: Java Runtime Environment (JRE 1.6 and above)	
Networking: Available network port e.g. 39940, 33941,33942 (For multiple machines)	

Licensing

- Annual license priced according to the maximum number of concurrent builds

Supported FPGA Tools

- Altera Quartus-II 13.0 onwards, Quartus Prime Standard/Pro Edition 15.1 onwards
- Xilinx ISE version 11.1 onwards, Vivado 2014.2 onwards

About Plunify

Plunify helps chip design companies optimize FPGA designs with big data and machine learning. Plunify is based in Singapore and in the United States.

Plunify Pte Ltd
Email: tellus@plunify.com

Singapore
82, Lorong 23 Geylang,
Atrix Building, #05-14,
Singapore (388409)

United States
4962 El Camino Real #225
Los Altos, CA 94022
USA

China
Room 1737, Level 17, Raffles City Tower 2, No. 3 Section 4,
South Renmin Road, Wuhou District, Chengdu, 610041,
China