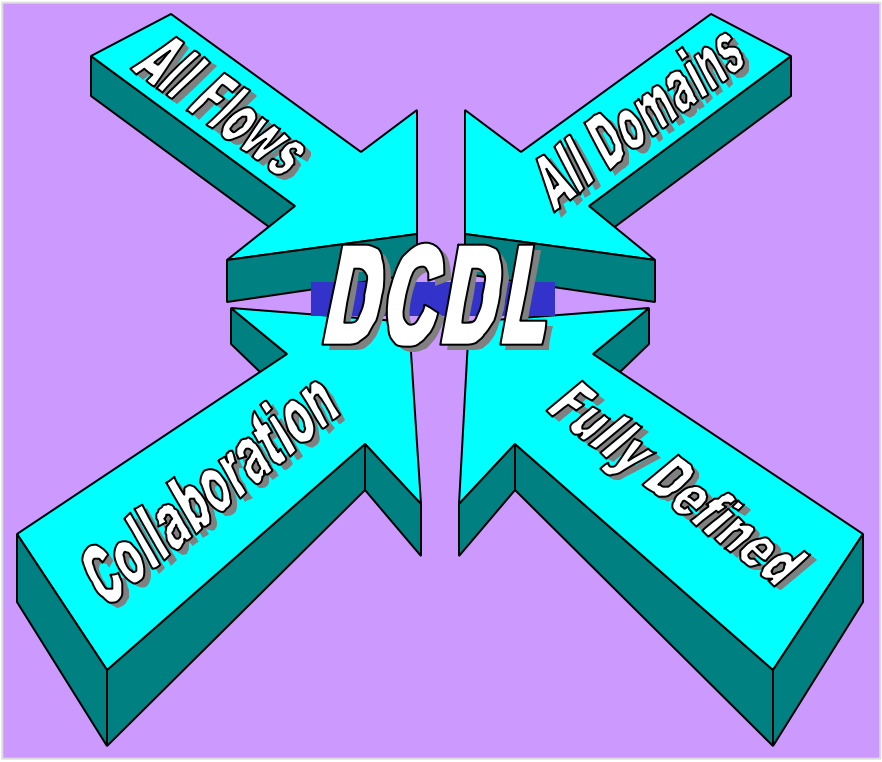
Design Constraints Description Language (DCDL)

DAC 2000 Edition

Design Intent - the Next Generation



DCDL- an open standard that provides an excellent method of specifying design intent for next generation, deep sub-micron design flows.

Major features:

- Open standard based on the continuous input of the industry
- Covers all constraint domains and design flows.
- Provides a complete syntax and semantic definition for true interoperability
- Covers the next generation of design constraints for deep sub-micron design
- Designed to fit seamlessly into Tcl environments
- Designed to work in embedded, proprietary scripting environments for initial entry or as a pure interchange language
- Free, open-source, reference parser being developed
- Free, browser-based getting started kit being developed

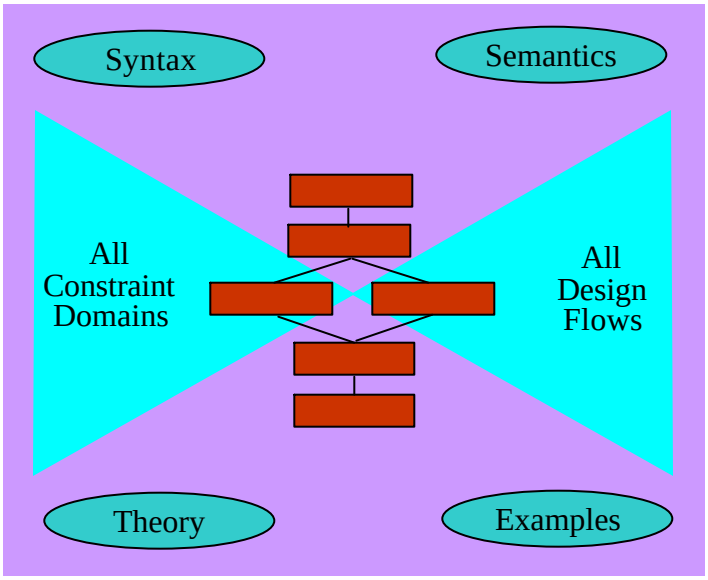
What is DCDL?

The Design Constraints Description Language (DCDL) is a set of commands that express design intent for all constraint domains within each electronic design flow. DCDL provides a syntactical and semantic definition that is language and tool independent.

Taking input from from designers, EDA vendors, and silicon vendors, the Accellera Working Group started with the timing domain and the related parasitic boundary and operating condition domains. Command syntax and precise semantics were defined with an eye toward deep sub-micron demands. Also, general language features and rules were defined and examples and theory were added.

Future constraint domains will include: power, signal integrity, area, logic architecture, physical, analog/mixed-signal, cost, reliability, manufacturability, test, and others.

While working toward an approved Accellera standard, the Working Group is also getting started with the IEEE standardization process.



DCDL provides a robust set of commands with well defined semantics to cover all constraint domains and design flows.

www.eda.org/dcdl

accellera SLDL VSI Alliance

Why is DCDL Important?

While there are many standard ways to describe a design, VHDL and Verilog for example, there was no standard method to describe a design's intent - the explicit objectives that the design must attain or avoid. DCDL fills this void by creating an open constraint language that seamlessly fits into today's design flows and the design flows of the future.

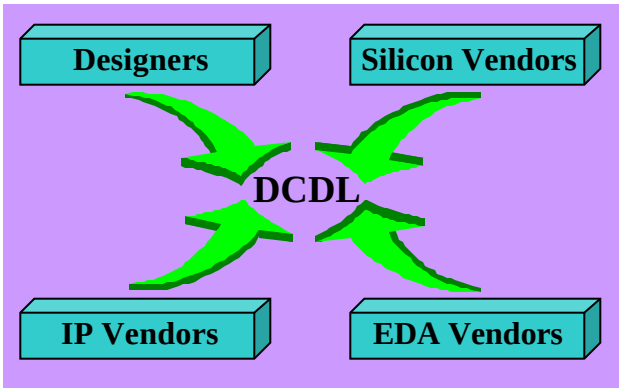
In deep sub-micron designs (DSM) it is crucial to be able to describe all the constraints at the block or design level, optimize based on these constraints to satisfy multiple conflicting objectives, and then verify that the constraints have been met.

As new tools appear at higher levels in the flow and as new tools are added to address DSM affects, proprietary constraint formats will no longer be practical - as they already do not provide the semantic definition required to for true, algorithmic behavior matching between tools; and they do not cover nearly enough domains. Proprietary formats are dependent on a particular vendor to implement change and are not driven by open, industry input - thus formats fall behind technically.

Who Benefits from DCDL?

DCDL is somewhat unique in that it serves several audiences and provides them with these primary uses:

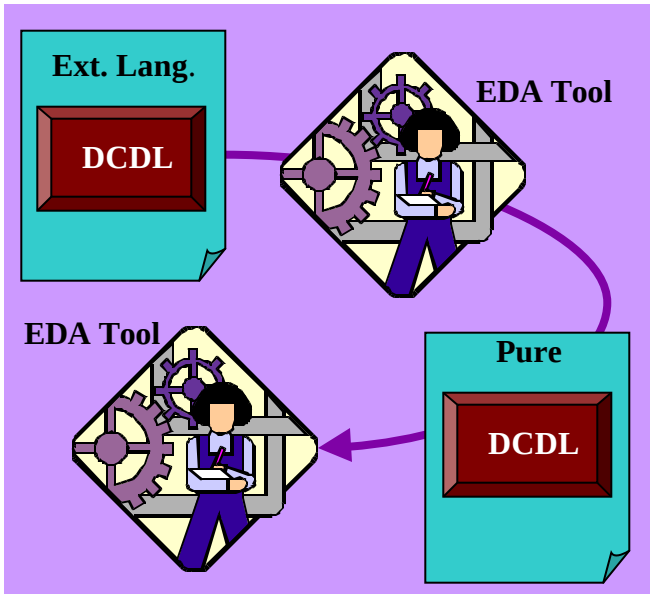
- **Designers**: as a consistent means to specify initial constraints and to pass those constraints to each tool in the design flow.
- **Silicon vendors**: as an efficient method of qualifying tools and to support a single method of passing constraints to tools from any EDA tool vendor.
- **IP providers**: as a vendor-neutral, single way to package design intent with soft and firm core data.
- **EDA vendors**: as a single method to enter constraints - allowing reader and writer code reuse and an easier fit into existing customer design flows. The EDA vendor can guarantee the behavior of the supported set of DCDL commands within a particular tool.



How is DCDL Used?

From the designer's perspective, initial design constraints are typically entered using a graphical user interface or by using a text editor. The designer can create a "pure" DCDL description or embed DCDL commands within a particular tool's extension language. For example, DCDL was designed to fit seamlessly into a Tcl environment.

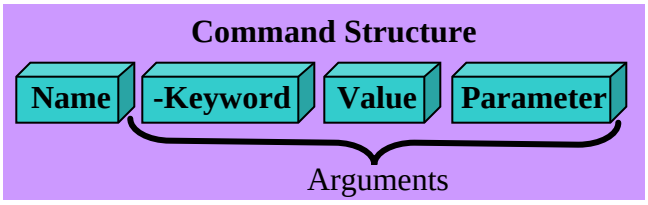
The designer performs tasks and/or analysis within a particular tool, and then writes out the constraints in pure DCDL format. The next tool in the design flow then reads this constraint file, such that the designer does not have to re-enter this information.



What Does DCDL Look Like?

Because DCDL is not an extension language, there are no programming structures such as control loops, variables, or operators. DCDL consists of a set of commands with a particular syntax and semantic definition.

Commands can be specified in any order. However, commands that can affect other commands must be defined first. The command name appears first, followed by arguments: keywords and keyword values and a positional parameter that is not associated with a particular keyword. Arguments can appear in any order and not all commands contain all types of arguments.



The following portion of DCDL code provides a look at several typical commands used to define operating conditions, clock definition, timing boundary constraints, and timing exceptions. *The syntax is subject to change as the specification is still under development.*

```
#Universal commands
version “1.0”
design_name_space -verilog “1995”
units -time 1E-9 -capacitance 1E12

#Operating conditions
operating_process -best -value 1.2
operating_temperature -best \
 -value 20.0
operating_voltage -best -value 2.5

#Clock commands
waveform -name clk -period 5.0 \
 -edges {0 2.5}
clock -waveform clk -ports {ck1}

#Boundary commands
data_arrival_time -waveform clk \
 -ports {ina[*] reset set} 2.0
data_required_time -waveform clk \
 -ports {outa[*] carry1 sim}

#Timing exception commands
borrow_limit -instance {inst1} 2.5
false_path -from {reset set}
disable -cell and2 -from {a}
multi_cycle_path -source \
 -to {memwrite} {1 2}

#Parasitic boundary commands
driver_resistance -ports \
 {halt resume din[*] bistm} 0.2
port_capacitance -ports \
 {clk_out test_err} 2
fanout_load -ports {out[*]} 4

#More universal commands

constant -pins {grnd} 0
extend_dcdl my_cmd -arguments \
 “-ports {p1 p2} 1.10”
funtional_mode -mode_name write \
 {ramu1}
include “/net/td1/common.dcdl”
```

What are the Deliverables?

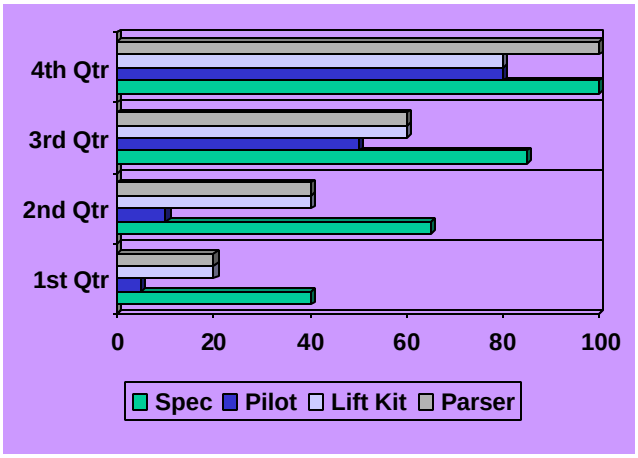
The Working Group plans on delivering several items for the DCDL standard effort:

- **V.10 specification**: a document that provides general language information, syntax, semantics, examples, and theory for the timing, operating conditions, and parasitic boundary domains.
- **Quick reference guide**: a small document that provides an overview of each DCDL command. This document is automatically generated from the specification source.
- **Golden syntax parser**: an open-source, Tcl-based parser that checks DCDL syntax. This parser can be included into a Tcl-based tool or used as an example.
- **Pilot project**: a “proof of concept” project that involves driving an ASIC and FPGA design flow with a real-world design and DCDL.
- **DCDL Lift Kit**: a browser-based package of materials that allows a user to quickly get started with DCDL. The materials include demonstrations, documents, training, application notes, and parser integration materials. The Lift Kit can also be used as a standalone website to share with others.

The DCDL Lift Kit provides a convenient method of housing all the deliverables plus extra materials to get started using DCDL quickly.

What are the Milestones?

The Working Group is focusing on the completion of V1.0 of the specification. However, the other deliverables are also considered important for the advancement of the standard and are being worked on in parallel.



The planned milestones in terms of percent complete per quarter for the year 2000.

How can you Help?

There are many ways to become involved with the DCDL standard effort:

- **Suggest**: if after reading about DCDL you feel it is a good fit for your company, learn more about the standard and request support for DCDL within your company or from your vendors.
- **Monitor**: by signing up on the email reflector. Send email to majordomo@ eda.org. Enter no subject and the text “subscribe dcwg” in the message body.
- **Contribute**: attend the weekly phone conference and contribute your ideas. Refer to the website for information.

Continue your Discovery of DCDL by Visiting the Website

Enter for a chance to win a Palm™ V

Follow the special DAC give-away link at:

www.eda.org/dcwg

A drawing for this Palm™ V will be held June 16, 2000. See the website for details.

Main website: www.eda.org/ dcwg

Sponsor websites:

- Accellera: www.accellera.org
- SLDL: www.inmet.com/SLDL
- VSIA: www.vsi.org