TinyTools

A graphical programming environment for fast workflow and optimised DSP usage in *TinyCore*

Overview

The ***TinyTools*** for programming Oxford Digital's ***TinyCore*** has been developed to achieve rapid prototyping and development and highly efficient implementation of DSP algorithms.

TinyTools is centred on a graphical programming environment and the ***TinyDraw*** front-end allows DSP engineers to visually create algorithms in a fast, intuitive manner.

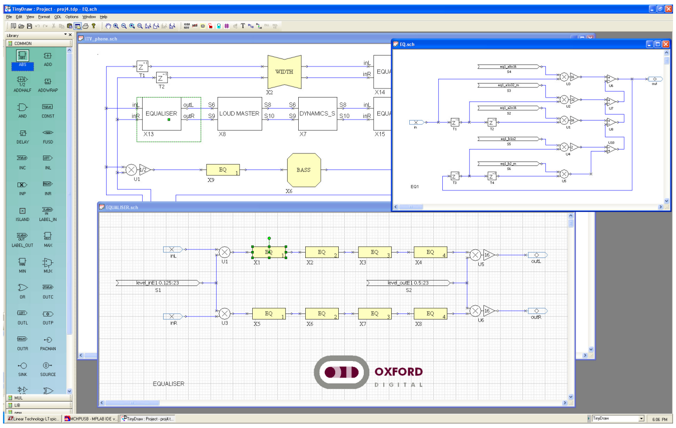
Once entered, the DSP design is then optimised using ***TinyOpt***. This ensures the maximum processing can be squeezed into any given ***TinyCore***, or alternatively will allow the minimum specification ***TinyCore*** to be used in order to save gate count.

TinyTools then applies the assembler, ***TinyAsm***, to create code that is ready to be loaded and executed directly on ***TinyCore*** development hardware within seconds of completing a DSP schematic.

Once the DSP is running on ***TinyCore***, coefficients in the DSP design can be adjusted using ***TinyGcon***. Real-time control ensures easy and precise fine-tunings can be made to the algorithm, with the results immediately observable by listening and/or measurement.

When fast turnarounds are needed and tight schedules must be met, ***TinyTools*** will ensure the fastest delivery of DSP algorithms, from concept right through to completion. Recent examples are the development of a demonstrable AEC algorithm from scratch within three days and the delivery of DSP code for an application together with HDL for a custom core in six days.

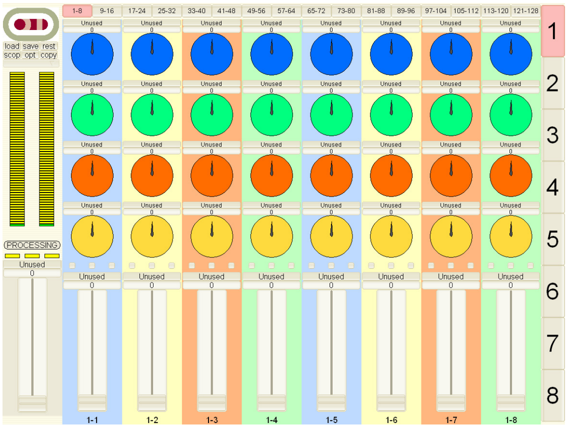
Features



TinyDraw

- ***TinyDraw*:** A graphical programming schematic capture package. Each of ***TinyCore***'s instructions is represented graphically, and elements are wired together to intuitively develop DSP algorithms. ***TinyDraw*** is fully hierarchical, allowing functional blocks to be developed and saved as library items, which leads to extensive software re-use.
- ***TinyOpt*:** an optimising compiler which takes a graphical DSP design from ***TinyDraw*** (or other schematic capture package, such as MathWorks® *Simulink*®) and converts it into assembler. ***TinyOpt*** utilises a 23-pass optimisation to remove all unnecessary instructions to the point where it beats even Oxford Digital's best assembly level experts by ~10%. The compiler automatically builds a DSP design for any given ***TinyCore*** (with enough DSP resource for the design) so that a design can be automatically reused on another ***TinyCore*** design with different parameters such as bit width, number of instructions per sample, audio I/O or RAM specification.
- ***TinyAsm*:** assembler for ***TinyCore*** that converts ***TinyOpt***'s output into machine code.

- ***TinyGcon*:** a customisable UI featuring 128 strips of knobs, buttons and faders, for controlling the coefficients of a DSP algorithm in real-time. ***TinyGcon*** also provides metering, and allows scenes to be set up so that different control settings can be stored in one session.



TinyGcon

Applications

- Rapid implementation of known algorithms
- Algorithm development and prototyping
- Inclusion of Tiny Audio Effects library elements in custom designs
- Inclusion of branded effects such as SRS Labs Inc. StudioSound HD™ in custom designs
- Optimisation of ***TinyCore*** implementations by “What if” investigations for bit-width and other parameters.

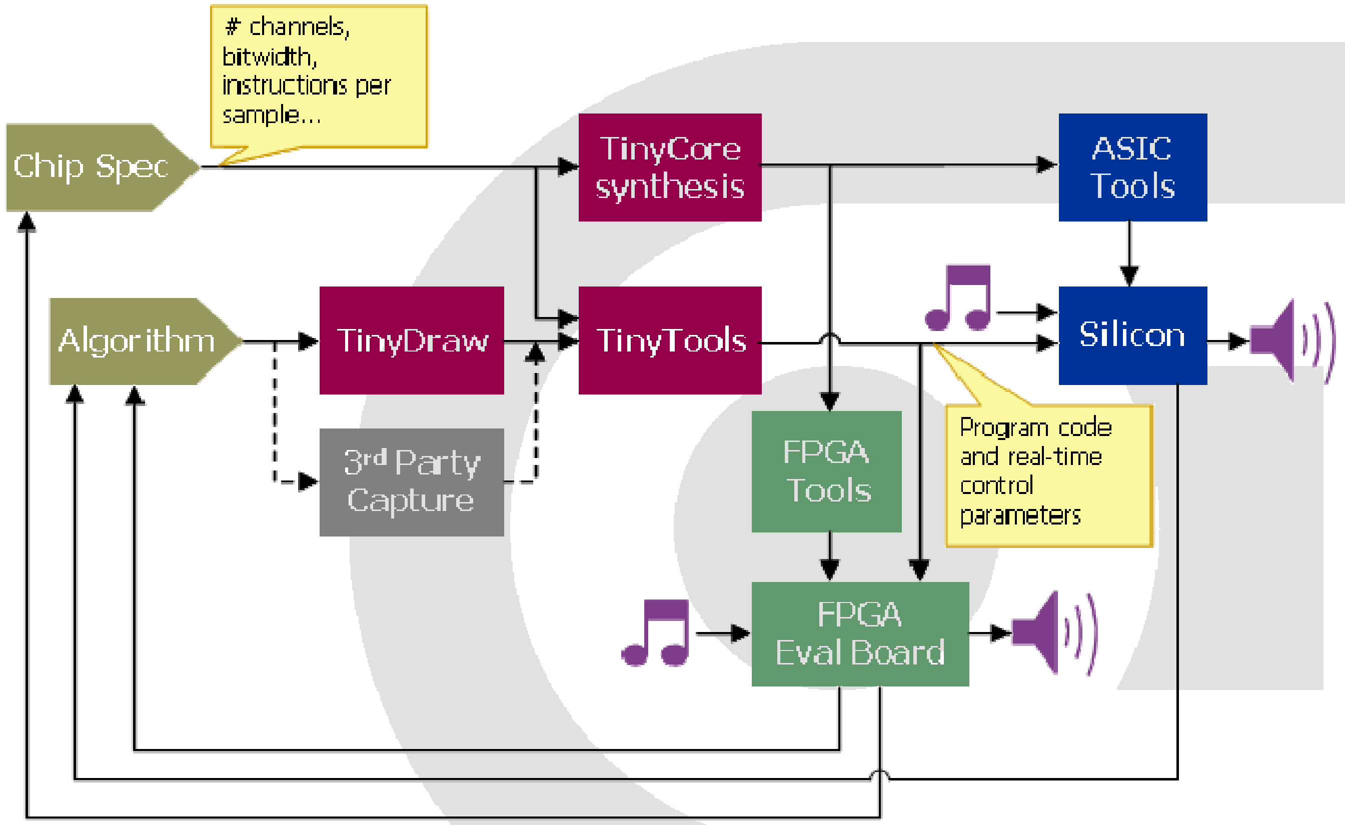


Diagram showing the relationship of the ***TinyTools*** together with recursive paths for modifying specification and algorithm for fine tuning the implementation