

## MAPLD2003

**Title: Improving FPGA application development using higher level tools.**

**Author: Eric Lord**

FPGAs have increased in size to the point where complete systems can be contained in possibly a single device and certainly in a group of devices. This is not only because of the size but also the 'extras' that are included such as memory, a variety of I/O including gigabit serial and most recently embedded processors. This leap in hardware capability has not been matched by similar dramatic evolutions in the development tools and all too often users are trying to make full use of the devices with the old tried and tested tools. These require lots of manual intervention and development times are long and expensive. Not the answer when time to market and competitiveness are the winners in present day industry.

This paper covers several developments that seek to improve on the above situation not just by making use of higher-level compilers, which are now becoming available, but also realising that systems require good communications. It is also about realising how a system should be developed and it is definitely not one depending solely on super-fast place and route tools where everything is poured into the melting pot and hopefully all is sorted out in the automatic process. The key is that a system is the sum of many parts and each can be developed separately and brought together in a controlled and orderly manner. This helps in the logistics because more resources can be employed in parallel thus reducing time. However this places more reliance on a good and agreed standard of communication between functions and also requires there to be a common database of the physical hardware and connections used and even better using an automated process to assign the physical system to the virtual one so that no manual errors can creep in.

This combination of techniques is exemplified in the paper by the use of tools provided by Nallatech in combination with their modular reconfigurable FPGA processing architecture DIME and DIME-II. FUSE is the application development software, which goes with the hardware and recently, additions of DIMETalk, a packet based point-to-point communications protocol, a Hardware Definition Database (HDD) and Cores for Interface Components (CIC) have been added. The cores are gradually being expanded to cover all DIME and DIME-II module definitions and components of DIMETalk, particularly the end-points. They are also being made available not just as components that are called from VHDL but can also be selected from higher-level compilers. The first target for this has been System Generator and plans are in place to include other tools such as AccelChip.

An example is given of a video communications link with input and output through A/D and D/A interfaces, two types of encryption and serialisation to convey the information over a laser link. Using System Generator, together with blocks for the interfaces and DIMETalk for the communications, it shows how rapidly the design can be developed and how quickly basic elements can be changed to address new requirements without requiring a change to the basic hardware.