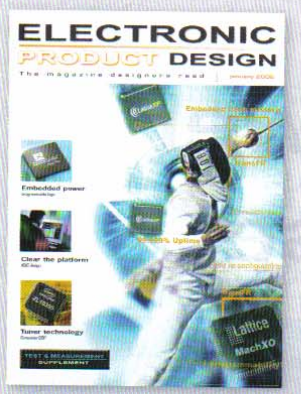
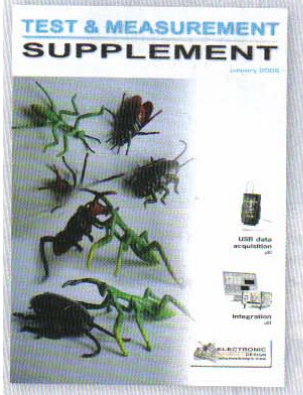
Electronic Product Design – January 2006



When executing in-field updates, Lattice Semiconductor explains how to eliminate downtime, particularly for non-redundant and mission-critical equipment



Bugs and glitches can delay a project, the right equipment to identify and eliminate is discussed in this month's supplement

- www.epdonthenet.net -

Editor Caroline Hayes
caroline.hayes@imlgroup.co.uk

Product Editor John Cornish
john.cornish@imlgroup.co.uk

Sub/production Editor Julie West

Designer Stuart Pritchard

Production Peter James
peter.james@imlgroup.co.uk

Circulation Manager Barbara Nye
barbara.nye@imlgroup.co.uk

Advertisement Manager Neal Henry
neal.henry@imlgroup.co.uk

Sales Executive Katherine Logan
katherine.logan@imlgroup.co.uk

Publisher Neil Whitaker
neil.whitaker@imlgroup.co.uk

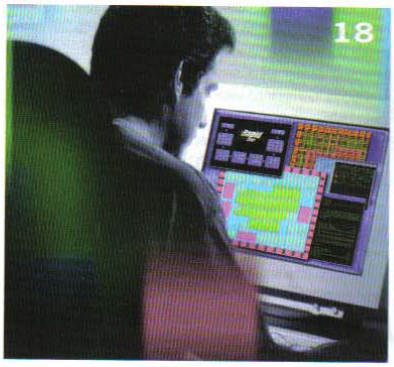
IML Group plc, Blair House, High Street,
Tonbridge, Kent TN9 1BQ
Tel 01732 359990 Fax 01732 770049

IML ABC BUSINESS PRESS BPA Applied for

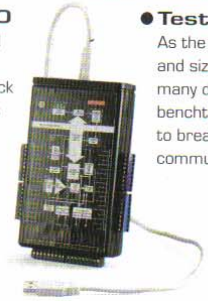
Electronic Product Design is published twelve times each year on a controlled circulation basis in the UK and certain areas of Northern Europe and Scandinavia. It is also available on subscription in the UK at £99.00 per annum; single copy price is £10.00; and at £126.00 per annum in Europe, or £183.00 per annum (airmail) in the rest of the world. ISSN No. 0263-1474

TECHNOLOGY FEATURES

NEWS ANALYSIS 4-6



TEST AND MEASUREMENT SUPPLEMENT 29-35



BUYERS' GUIDE 37-42

MAXIMISING UPTIME AT RECONFIGURATION

The increasing complexity of today's systems is an important factor to take into consideration when executing in-field updates. It is desirable to reduce or eliminate the downtime, particularly for non-redundant and mission-critical equipment

Unfortunately current FPGA update methods are extremely disruptive. Current programmable devices typically require the system to be offline when being re-written. Design engineers tackling such issues go to great lengths to circumvent these problems.

Clearly a new technology is needed and basic system requirements need to be examined to tackle this problem.

Re-configuration needs

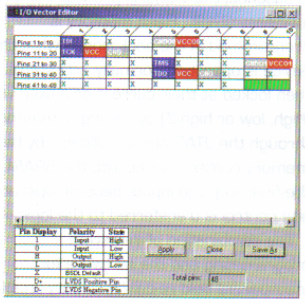
To re-configure a system in the field, basic functions need to be considered, critically that there is no disruption to system performance. The system must not crash, downtime by definition must be minimal and the new configuration must synchronise with the external logic before becoming active.

The combination of embedded flash with a parallel programming interface, JTAG I/O, and the right software utility, has opened up a whole set of new applications that previously required a complex solution. FPGA technology would seem to provide an ideal solution but significant issues with 'traditional' FPGAs need addressing before solving the basic problem.

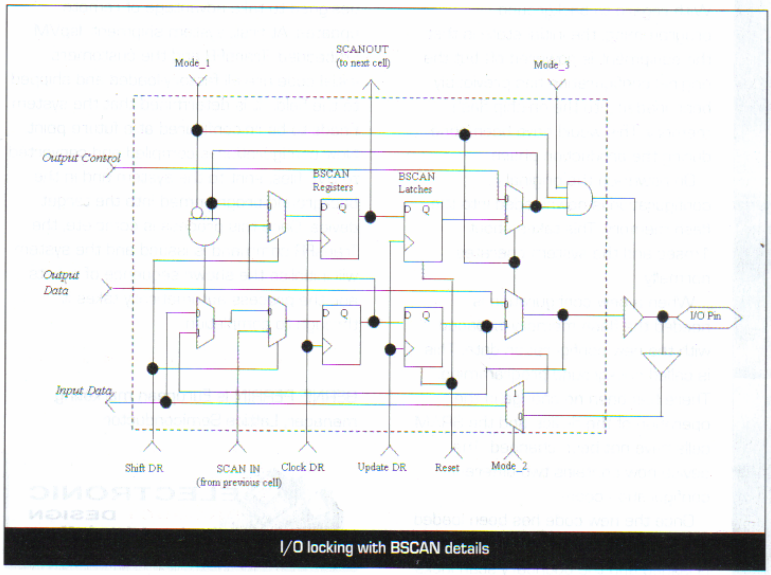
Traditional FPGAs are programmed serially, a relatively slow process because the FPGAs need to be erased before they can be programmed with new code. This raises difficult system issues as I/Os can become indeterminate while the FPGA is changing. Programming software also needs to be made available at a board level and most importantly a method of replacing the configuration data is needed.

The majority of FPGAs are SRAM-based and re-configurable, with the configuration code usually held in a separate PROM memory. This is a two-chip solution with a serial interface connecting the two chips together. The recent introduction of Flash-based FPGAs combining non-volatility with re-configurability creates a single chip solution. However, without further architectural considerations it is still not enough to enable a seamless re-configuration. New features are needed.

- Custom or Dynamic Mode
- Select
  - H – Drive Logic 1
  - L – Drive Logic 0
  - X – Leave Alone (Dynamic mode)
  - X – BSDL Default (Custom mode)
  - 1 – High Z (Input externally driven to 1)
  - 0 – High Z (Input externally driven to 0)



I/O Vector Editor, demonstrating how the states of I/O pins are specified



I/O locking with BSCAN details

Architecture

The architecture of the LatticeXP FPGAs incorporates features that enable field re-configuration to be carried out swiftly and seamlessly. The combination of SRAM and flash facilitates the implementation of Lattice's TransFR technology that enables field reconfiguration while the system operates. The flash memory is connected to the configuration bits through control logic. Programming is fast via a massively parallel data path and the download from flash SRAM takes less than 1msec, tackling the problem of slow serial programming.

There are two ports that can be used to programme the device, an 8bit parallel port called the sysCONFIG port and the JTAG port. The sysCONFIG port is an 8bit parallel interface that can also be used to programme the flash memory or the SRAM

continues on page 22

directly. The JTAG port, which can also be used to programme the device, conforms to the IEEE 1532/1149.1 programming specification, a key feature in the process.

The LatticeXP architecture has a rich set of IEEE 1149.1 boundary scan features, which enables the output circuitry to be sampled and pre-loaded with set configurations. They can be used to effectively 'pause' the FPGA during re-configuration. The BScan cell has three operational modes. Data output is routed so that the clock DR input and update DR registers and then captures the value. Mode_2 Mux controls the routing of the signal. The value of the current output is captured and the cell is in a known state. The device can then be re-programmed.

Background programming

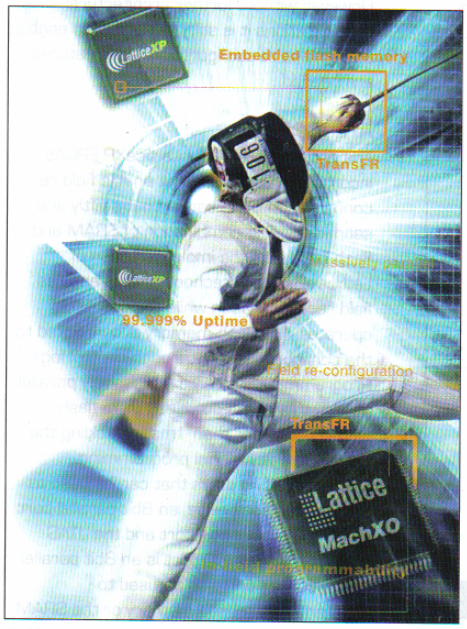
In essence, the TransFR process comprises four easy steps. Flash memory is programmed in the background while the device is operating normally. I/O states are then locked at their current or a fixed value (high, low or high-Z) by issuing commands through the JTAG device, followed by flash memory content loaded into the SRAM. In the final step the inputs become operational and control is transferred to the logic.

The combination of in-system programming and the software for transparent field re-configurability enables the system to swiftly re-configure in an efficient manner. The necessary functions are contained within Lattice's ispVM programming utility, specifically a subset called ispVM Embedded and a command designated TransFR.

Flash-based, non-volatile FPGA devices

IspVM supports both the IEEE 1532 programming standard and the IEEE1149.1 JTAG boundary scan test standard. The TransFR function is a software utility embedded in ispVM that simplifies the whole process for the design engineer to specify such things as Pin assignments. IspVM Embedded is a simplified version of ispVM and as the name implies it is designed to be embedded within the target application so that when the TransFR command is sent, the ispVM embedded machine re-configures the target device.

The combination of in-system programming and the software for transparent field re-configurability enables the system to swiftly re-configure in an efficient manner



Loading reconfigurations

With regard to background programming, the initial state is that the equipment is powered off but the original configuration has previously been loaded into the on-chip flash memory. This would have been done during the production phase.

On power-up the original configuration is downloaded into the flash memory. This takes about 1msec and the system operates normally.

When a new configuration is needed the flash memory is loaded with the new configuration data. This is called background programming. There has been no change in the operation of the device and the SRAM cells have not been changed. The device now contains two different configuration codes.

Once the new code has been loaded into the flash, the TransFR command is issued and the boundary scan locks the I/O states. These states will have been specified by the design engineer and are set up using Lattice's ispVM software. These can be set into high, low, high-Z or leave alone, i.e. its current state.

When the I/Os have been locked, the new configuration can be downloaded into the SRAM. The user logic becomes operational and responsive to inputs, and the PLLs re-lock, which takes less than 150µsec. I/O control is transferred back to the user logic. A seamless transfer has been made to the new configuration.

Once the new configuration has been loaded into the device using the IEEE 1532 programming standard, the system needs to be returned to normal operation. In this case, the Mode_2_Mux allows the input data to be driven from point A internal or B external. This allows the clocks to run and the device to synchronise both internally and with the rest of the system. When this is done, the lock is removed from the outputs and they can be driven normally. The key is that the final three steps of the process are initiated as a single software command.

By using ispVM, all the elements are now in place to complete an update from a remote location. The system has been designed to take advantage of remote updates. At first, system shipment, IspVM Embedded, TransFR and the customers initial code are all factory-loaded and shipped to the field. It is determined that the system needs to be re-configured at a future point. New configuration is compiled and converted to SVF files, sent to the system and in the background programmed into the target device. Once this process is complete, the TransFR command is issued and the system will initialise the shown sequence of events and the process automatically takes it through to completion.

BERNIE PERRIN is European marketing manager, Lattice Semiconductor