Advanced Flash Programming using Processor Assisted Methods

Presenter: Jan Heiber (j.heiber@goepel.com)
Presentation Contents

1. Status on Flash Programming
2. Intro - μP Assisted Procedures
3. Available System Solutions
4. Summary and Outlook
Facts on Flash Programming using BScan

Background Information

- Boundary Scan is well proven for programming of Flash components
- Access to all Flash signals from Boundary Scan side needed
- Automated Flash programming tools are state of the art
- External toggling of WE pin can expedite programming procedure
- Use of special library models to define Flash programming algorithms
- Flash banks can typically be programmed like a single device
Flash Technology Changes by OCF ...

- Rapid growing number of Micro Control Units (MCU) with OCF
- Continuously growing Flash size
- Standard Boundary Scan can’t program On-Chip Flash
Impact by Serial and GBit Flash...

1. Programming Serial Flash through BScan
   - Flash size: 100Kb, 1Mb, 10Mb, 100Mb
   - Prog time: Minutes, Hours
   - Notes:
     - Large BScan cell overhead
     - Increasing Flash size (NOR)
     - BScan programming not efficient

2. Programming GBit Flash through BScan
   - Flash size: 10Mb, 100Mb, 1Gb, 10Gb
   - Prog time: Seconds, Minutes, Hours
   - Notes:
     - Size is doubling every 12 months
     - Increasing use of NAND Flash
     - BScan programming not efficient
Some Thoughts ...

**Conclusions**

- BScan programming becomes inefficient due to On-Chip Flash and serial/GBit Flash
- Long BScan registers in µP and FPGA causing a large overhead
- Key challenge is to find better technologies but keeping the benefits of ISP methodology
Intro - μP Assisted Procedures
Boundary Scan vs. Processor Emulation

**Boundary Scan I/F**

- **Boundary Scan TAP**
  - TAP Controller
    - Instruction Register
    - Standard Data Registers
  - BScan Register
  - Pin Interface

  - Static pin electronics
  - BScan cells define vectors
  - Serially controlled pin interface
  - Scalable number of pins
  - Arbitrary static signal timing
  - Arbitrary vector definition per pin

**μP Emulation I/F**

- **JTAG Emulation TAP**
  - TAP Controller
    - Instruction Register
    - Standard Data Registers
    - Emulation Registers
  - I/O Controller (μP Core)
  - Pin Interface (System Bus)

  - Dynamic pin electronics
  - μP defines vectors
  - Parallel controlled pin interface
  - Fixed number of pins
  - Rigid dynamic signal timing
  - just Address / Data bus controllable
Interconnection of μP Core and Flash

**Pro’s**
- Programming access ready by design
- No special DfP (Design for Programming) rules needed
- μP Core acts as the high speed programming engine
- In-System Programming (ISP) methodology like BScan
- Support for any serial or parallel Flash incl. NAND

Control System

JTAG TAP

On-Chip Resources
- Embedded Flash
- Complex Bus I/F
- Legacy Bus I/F
- Legacy I/O Ports

System Bus I/F

External Bus Devices
- Flash Components
  - SRAM / DRAM
  - Peripheral I/O Ports
  - Peripheral Bus I/F
  - Aux Resources

Board Level
Flash Programming Method

System tools
- Programming vectors
- Programming status

Unit Under Test (UUT) with embedded controller
- Embedded programming agent
- Dynamic executed programming
- Dynamic sampled programming response
- System bus connected
- Flash programming target

TAP Controller
- Instruction Register
- Standard Data Registers
- Emulation Registers

JTAG debug interface

I/O Controller (µP Core)

Pin Interface (System Bus)
Progress with Handicaps ...

Conclusions

- μP assisted programming can cover OCF, serial/Gbit Flash and continues ISP philosophy
- Core allows high speed programming but image transfer via JTAG can be a bottleneck
- Need for System solutions to support both BScan and μP assisted programming
Available System Solutions
Solution #1: Standard Programming

Script development with native s/w tool chain

Executable

Download image into µP core environment

Programming execution by µP core

Chip Level

µP Core

On-Chip Resources
- Embedded Flash
- Complex Bus I/F
- Legacy Bus I/F
- Legacy I/O Ports

System Bus I/F

External Bus Devices
- Flash Components
  - SRAM / DRAM
  - Peripheral I/O Ports
  - Peripheral Bus I/F
  - Aux Resources

Board Level

Control System
Solution #2: In-Application Programming

Embedded firmware

Image download via high speed I/F

Launching ROM/Flash resident IAP routine

Programming execution by µP core

Chip Level

µP Core

On-Chip Resources
- Embedded Flash
- High speed I/F
- Legacy Bus I/F
- Legacy I/O Ports

System Bus I/F

External Bus Devices

Flash Components
- SRAM / DRAM
- Peripheral I/O Ports
- Peripheral Bus I/F
- Aux Resources

Board Level
Advanced Programming

Control System

Script development with native s/w tool chain

Executable

Image download via high speed I/F

Programming execution by µP core

Chip Level

On-Chip Resources
- Embedded Flash
- High speed I/F

System Bus I/F
- Legacy Bus I/F
- Legacy I/O Ports

External Bus Devices
- Flash Components
  - SRAM / DRAM
  - Peripheral I/O Ports
  - Peripheral Bus I/F
  - Aux Resources

Board Level

JTAG TAP

µP Core
# Comparison of System Solutions

## Features

<table>
<thead>
<tr>
<th>Features</th>
<th>Standard</th>
<th>In-Application</th>
<th>Advanced</th>
</tr>
</thead>
<tbody>
<tr>
<td>Automated Flash program scripting</td>
<td>yes</td>
<td>no</td>
<td>Yes</td>
</tr>
<tr>
<td>Pre-programmed firmware needed</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Flash image transfer channel</td>
<td>Debug Port</td>
<td>COM I/F</td>
<td>COM I/F</td>
</tr>
<tr>
<td>Booting before programming needed</td>
<td>no</td>
<td>yes</td>
<td>no</td>
</tr>
<tr>
<td>Access/Equipment for debug port needed</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Support for In-line production</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>Relative image transfer time</td>
<td>high</td>
<td>low</td>
<td>low</td>
</tr>
<tr>
<td>Total programming time (with booting)</td>
<td>medium</td>
<td>medium</td>
<td>best</td>
</tr>
<tr>
<td>Best suited Flash image size [MB]</td>
<td>[x] range</td>
<td>[xx] range</td>
<td>[xx] range</td>
</tr>
</tbody>
</table>
Summary and Outlook
Flash Programming Solutions Overview

- **On Chip Flash**
  - Flash Size [Mbyte]
  - State of the Art for embedded flash

- **External Flash**
  - Boundary Scan Flash Programming
  - FPGA Accelerated Flash Programming
  - Advanced µP assisted Flash Programming
  - Standard µP assisted Flash Programming

- **Chip embedded Programmer**

- Largest project requests for external flash devices so far

- State of the Art for embedded flash
## Conclusions and forecast

- **BScan is not efficient for programming On-chip Flash and serial/Gbit Flash.**
- **μP assisted programming is an excellent solution.** Key is the used image transfer link.
- **New standards like IEEE1687 enables further high speed programming techniques.**
More Information...

Further readings and references


Thank you for your attention.

Any questions?

For further information please use following contact information

Jan Heiber  j.heiber@goepel.com
Website     www.goepel.com
Scandinavia +45-8748-0608
Germany      +49-3641-6896-0