A Debug Probe for Concurrently Debugging Multiple Embedded Cores and Inter-Core Transactions in NoC-Based Systems

Shan Tang and Qiang Xu

**EDA&Testing Laboratory** 

1



The Chinese University of Hong Kong 香港中文大學

# Background

## Silicon Debug for Single Core

#### Run Control Interface (e.g., JTAG)

- Widely used in practice
- Not enough for tricky bugs
- Not applicable for certain real-time applications

#### □ Trace + Trigger

Effective in most cases

#### Challenging yet Well Studied Problem!

### Multi-Core Debug - Requirements

- Concurrent debug access to interacting cores and their transactions
- □ System-level triggering and trace
- Debug event synchronization for cores from multiple clock domains
- Limited DfD cost in terms of silicon area, routing and device pins.

#### Multi-Core Debug Architecture -ARM CoreSight



5

#### Multi-Core Debug Architecture – First Silicon



6

#### Network-on-Chip

- Most promising communication scheme for future giga-scale SoCs
- □ NoC generally contains:
  - Network interface
  - Router
  - Physical link
- Need debug support as a new design paradigm



### **NoC Monitoring Service**

- □ Ciordas et al. TODAES'05, IES'06
  - Monitoring probe attached at routers
  - Effective identifying bit-level errors
  - Costly in terms of NoC bandwidth at transaction level
  - Monitor instead of Debug

## Multi-Core Debug Platform for NoC-Based Systems

#### Rationale

- □ How to achieve concurrent debug access?
  - Reuse NoC to transfer debug data
  - Insert debug probe between core and NI
- How to monitor inter-core transactions in NoC-based systems
  - Not shared mechanism cannot simply listen
- How to deal with the latency problem?
  - Use QoS guaranteed service for debug connections
    - Two-pass debug strategy

#### **Proposed Platform**



■C port: Communication Port ■D port: Debug Port ■T port: Trace Port

■MSA: Monitor Service Access

- On-Chip Debug Architecture
- Off-Chip Debug Controller
- Supporting Debug Software

#### **On-Chip Debug Architecture**

- Core-Level Debug Probe
  - Between core and NI
  - Monitor transactions
  - Control/observe core's debug interface
- System-level Debug Agent
  - JTAG (+ trace port)
  - Controlled by off-chip debug controller

#### Debug Probe Design



13

#### Trigger & Trace Unit Design



## Debug Agent

- **TAP** controlled by off-chip debug controller
- Build debug connections between DA and DPs



# Control On-Chip Debug Registers through DA

| Off-Chip Debug Controller          | Debug Agent               |
|------------------------------------|---------------------------|
| Write 'DEBUG_REG' command          |                           |
| into IR;                           | Use DEBUG_REG_DATA        |
|                                    | register as DR;           |
| Write operation                    |                           |
| Shift in 'write' command to DR:    |                           |
| :WR:ADDR:DATA:-:                   | Write specified register; |
| Read operation                     |                           |
| Shift in 'read' command to DR:     |                           |
| :RD:ADDR:-:-:                      | Read specified register;  |
| Do {Shift out the contents of DR;} |                           |
| While (READ_VALID) != '1';         | Set READ_VALID when       |
|                                    | data is ready;            |
| The DATA field in RD is the        |                           |
| valid data;                        |                           |

## Supporting Debug Software

Provide GUI or command line interface

- □ 3-layer architecture:
  - Cross debugger
  - Core debugger and Transaction debugger
  - Multi-core debug driver (PC interfaces)

## **Off-Chip Debug Controller**

- Translation layer between debuggers and on-chip debug architecture
  - Schedule debug commands/data transfer
- All debug resources in DPs and CUDs are mapped into addressable registers

|        |        |                           |                       |            |                     | Expected                |
|--------|--------|---------------------------|-----------------------|------------|---------------------|-------------------------|
| JTAG   | Stop A | Stop B                    | Stop C                |            |                     | Trigger Time            |
|        |        |                           |                       |            |                     |                         |
| Core A |        | Sto                       | рА                    |            |                     |                         |
|        |        | ⊲T <sub>d_core_</sub> reç | _wrA •                |            | T <sub>delayA</sub> |                         |
| Core B |        |                           |                       | Stop B     |                     |                         |
| -      |        |                           | < T <sub>d_core</sub> | _reg_wrB   | ■ T <sub>del</sub>  | ayB                     |
| Core C |        |                           |                       |            | Stop                |                         |
|        |        |                           |                       | <b>∢</b> T | d_core_reg_wrC      | ► T <sub>delayC</sub> ► |

#### **Debug Access Delay**



19

## Experimental Results

#### A Multi-Core Debug Example

DP detects a transaction trigger and stop three interacting cores

1. set DP trigger condition;

- 2. start DP and cores at the same time;
- 3. while (!triggered) check trigger status;
- 4. read out timestamp of the trigger event;
- 5. calculate relative trigger time for each core;
- 6. reset debug environment;
- 7. start DP and cores at the same time;
- 8. stop cores at trigger time;
- 9. hand over the control to the debuggers;

#### Simulation Environment



#### Simulation Results – Multi-core Concurrent Debug

| - | Clk                 |       |          |           |         |          |         |        |          |         |          |         |         |          |          |          |          |              |        |        |   |
|---|---------------------|-------|----------|-----------|---------|----------|---------|--------|----------|---------|----------|---------|---------|----------|----------|----------|----------|--------------|--------|--------|---|
| • | DA OCP MCmd(write)  | 1     |          |           |         |          | debu    | g acce | ss on ç  | hip-lev | /el debu | g port  |         |          |          |          |          |              |        |        | _ |
| • | DP1 0CP MCmd(write) | Ŀ     | setting  | DP1's d   | elay co | ounter   |         | _      |          |         | F        | DP1 rec | eived ' | write' o | peratio  | n        |          |              |        |        | _ |
| • | DP1 Delay Counter   | _)000 | 000087 p | ore-calcu | lated o | lelay va | lue     |        |          |         |          |         |         | )        | )        |          |          |              | 00000  |        | _ |
| • | CUD1 JTAG Reg Write |       |          |           | *       |          |         |        |          |         |          |         | delay   | counte   | r count  | ting dov | wn       |              |        |        | l |
| • | DP2 0CP MCmd(write) |       | NoC linl | k delay ′ |         | – settin | g DP2's | delay  | counte   | -       |          |         |         |          | ←DP2     | receive  | d 'write | e' opera     | tion   |        | _ |
| 4 | DP2 Delay Counter   | 00000 | 000      | -         | )(      | 0000044  |         | - 3    |          |         | 1        | 2       |         |          |          |          |          |              | 00000  |        | _ |
| 4 | CUD2 JTAG Reg Write |       |          |           |         |          |         |        |          |         |          |         |         |          |          |          |          |              |        |        | l |
| 4 | DP3 0CP MCmd(write) |       |          | se        | tting D | P3's del | ay coun | ter →  |          |         |          |         | DP      | 2 receiv | ved 'wri | ite' ope | ration – | +            |        |        | _ |
| 4 | DP3 Delay Counter   | 00000 | 000      |           |         |          | -       | _      | )(000000 | 001     |          | -       |         |          |          |          |          | <b>#0000</b> | 00000  |        | _ |
| • | CUD3 JTAG Reg Write |       |          |           |         |          |         |        |          |         |          |         | s       | etting 3 | CUDs     | debug    | registe  | rs at th     | e same | time → | l |

Pre-calculated delay can be inserted
Multiple cores can be concurrently debugged

# Simulation Results – Transaction Trace

| 🔷 Clk                       | 100 | ก่านขามขางบน |          | לירונות המתרחות לי |               |          |          | התהתהתהה   |         | ההההההקר | תתתתתחת      | הכתותותה | ממסמתיומה     | ท่องขางกากบ |          | התתתתת היות | nonnoni  | ההרונורונורונו | N  |
|-----------------------------|-----|--------------|----------|--------------------|---------------|----------|----------|------------|---------|----------|--------------|----------|---------------|-------------|----------|-------------|----------|----------------|----|
| 🔷 Register Write Enable     |     | <u>л</u>     |          |                    | сс<br>        | onfigure | the trig | ger and    | d trace | module   |              |          |               | R           |          |             |          |                |    |
| 🔶 Register Write Address    | 0   | <b>X</b> 1   |          |                    |               |          | _        | <u>)</u> 2 |         |          |              |          |               | Х3          |          |             | _        |                | _  |
| 🔷 Register Write Data       | E   | 000008       | 10       |                    |               |          |          | )00000     | 008     | -        |              |          |               | )FFFFF      | FOF      |             |          |                | -  |
| Detector Hit                |     |              |          |                    |               |          |          |            |         |          |              |          |               |             | det      | ector hi    |          |                |    |
| Transaction Analyzer Hit    |     |              |          |                    |               |          |          |            |         | _        |              |          |               |             | transa   | ction an    | alyzer h | it →           |    |
| 🔷 Trace Buffer Write Enable |     |              |          |                    |               | OCP tra  | ansactio | ons        |         |          | _            |          |               | tra         | insactio | n record    | genera   | ted →          | 0  |
| ♦ Trace Buffer Write Data   | 00  | 0000000      | 00000000 | 0000000            | 0000000       | 000000   | 0000000  | 0000000    | 000000  | 000000   | 0000000      | 0000000  | 0000000       | 0000000     | 0000000  | 00000000    | 0000000  | 0000 XX        | Ż  |
| OCP Master Address          | 00  | 0            | XXX0000  | ))000              | 100           |          | 0 ))(0   | 00 )       | 000     | 000      | X000         | ))(000   | XX000         | ))000       | XX000    | ))(000      | XOC      | 0 ))(00        | 10 |
| OCP Master Command          | 00  | )1           | 0000     | ))(000             | × ))(00)      | 0))(00   | 0 ))(0   | 00 )       | 000     | 000      | ))(000       | ))(000   | ))(000        | ))(000      | ))(000   | ))(000      | 100      | 0))(00         | 0  |
| OCP Response                | 00  | )            |          | ))(00              | <u>))</u> (00 | ))(00    | Xoc      | X          | 00      | 000      | <u>))</u> 00 | 000      | <u>))</u> (00 | ))(00       | ))(00    | ))(00       | ))(00    | )))00          |    |

 Configurable trigger and trace conditions
Transactions are recorded when trigger event happens

#### DfD Area Cost – Debug Probe

| Reference                         | Area (µm <sup>2</sup> ) | Percentage |
|-----------------------------------|-------------------------|------------|
| Debug Probe Top                   | 188059.0                | 100.0%     |
| Transaction Trace Module          |                         |            |
| Trigger and Trace Top             | 19692.1                 | 10.5%      |
| Input Switch                      | 481.8                   | 0.3%       |
| Detector                          | 543.3                   | 0.3%       |
| Transaction Analyzer              | 364.4                   | 0.2%       |
| MUX                               | 14.3                    | 0.0%       |
| Record Generation                 | 4629.7                  | 2.5%       |
| Shadow Buffer                     | 9197.9                  | 4.9%       |
| Trigger and Trace Control         | 4456.3                  | 2.4%       |
| Transaction Trace Buffer (32x128) | 120689.9                | 64.2%      |
| Debug Access Module               |                         |            |
| Delay Control                     | 5384.8                  | 2.9%       |
| OCP Slave INF                     | 2020.7                  | 1.1%       |
| Core Debug Module                 |                         |            |
| JTAG INF                          | 3912.9                  | 2.1%       |
| Core Trace Module                 |                         |            |
| Core Trace INF                    | 3099.6                  | 1.6%       |
| Core Trace Buffer                 | 31225.6                 | 16.6%      |
| OCP Slave INF                     | 2024.0                  | 1.1%       |

#### Future Work

Verify the proposed debug platform in-field

- Introduce DfD units inside NoC to locate the exact NoC error
- NoC without QoS connections for debug?

## Conclusion



