Bus Request-Response Trace for a SMART Interconnect System

Jeff Haight
Sonics, Inc.
Mountain View Cal.
jhaight@sonicsinc.com

Arcady Pann
Mobileye Vision Technologies LTD
Jerusalem, Israel
arkady.pann@mobileye.com

Dr. Neal Stollon
FS2 Division of MIPS Technologies Inc.
Mountain View, Calif.
neals@fs2.com

Introduction

This document discusses a socket-level traffic monitoring approach, called a Bus Request-Response Trace (Bus RRT[tm]) that uses On Chip Instrumentation (OCI[tm]) IP to provide information on system bus latencies and related measurements for a complex Systems on Chip (SoC) being developed by Mobileye Vision Technologies. In complex SoCs, which typically include an increasingly large number of embedded processors and other cores, the interconnect takes on a new level of complexity in order to enable the maximum performance of each processor. A comprehensive understanding of the interaction and performance of this interconnect is critical for real time performance and synergistic to the understanding of the overall processing operations and interactions that are best analyzed in a physical, rather than simulated environment (1).

There are a variety of reasons why new generations of interconnects and analysis tools to support them are becoming increasingly critical and important;

1) Growing levels of IC integration, with on-chip heterogeneous multiprocessing (HMP), require a clean way to efficiently handle complex data flow architectures with inter-communicating cores, often having a range of new requirements and features - different data feeds, operating speeds, types of data endianness, diverse and dynamic levels of security, and Quality of Service (QoS).
2) Growing awareness that flexible and rapid integration of IP from multiple external sources is key to reducing time to market, with concurrent requirements for integrating the test, verification, and simulation environments.
3) Growing sophistication of the processors' data flow requirements, requiring the ability to handle multi-processing and multi-threading in efficient, non-blocking manners. In particular, the multi-threading features of the MIPS32® 34Kf™ cores present an interconnection systems challenge to provide a system environment that allows full range of features to operate.
4) Growing appreciation for multi-generation design approaches that efficiently address product upgrades, market segmentation, and product differentiation while maintaining common design infrastructure to keep design efforts manageable. This requirement is especially apparent in automotive applications, where a common infrastructure is needed for differing platforms.
5) Supporting analysis IP provides a means of tying together pre-silicon and initial physical product verification by providing access and visibility to embedded operations (2). This analysis is key to in-depth understanding of the operational specifics of the design under different conditions.
6) All of the above factors can be addressed by the adoption of socket-based approaches, to minimize the effort of adding or replacing different blocks of IP. OCP architectures have pioneered this concept of socket based design, with other bus architectures adopting many of the same principles in order to provide the needed range of design tradeoffs and performance.

Mobileye EyeQ2[tm] System Overview

The initial architecture to integrate the RRT instrumentation is the EyeQ2[tm] processor, being developed by Mobileye Vision Technologies. The Mobileye EyeQ2 microprocessor is a complex 90 nm SoC, combining two hyper-thread MIPS32 34Kf soft cores, Mobileye VCE/VMP vector processors, 512KB on-chip ISRAM, 16 channels DMA and array of popular peripherals for external communication: 2xCAN, 2xUART, I2C, 32GPIO, FLASH Ctrl, two input video channels of high resolution (4000x2000 pix), one output video channel with capabilities of synthetic graphic integration, etc.

Mobile 166MHz 64bit DDR controller provides a reliable access to the 256MB memory space.

Each of 333MHz MIPS32 34Kf soft cores are supported by a 32/32KB data/instruction cache and 32/8 data/instruction scratchpad RAM. An Inter-Thread-Communication-Unit (ITU) block containing 32 entries is accessible from both MIPS CPUs in order to improve inter-thread and inter-CPU communication.

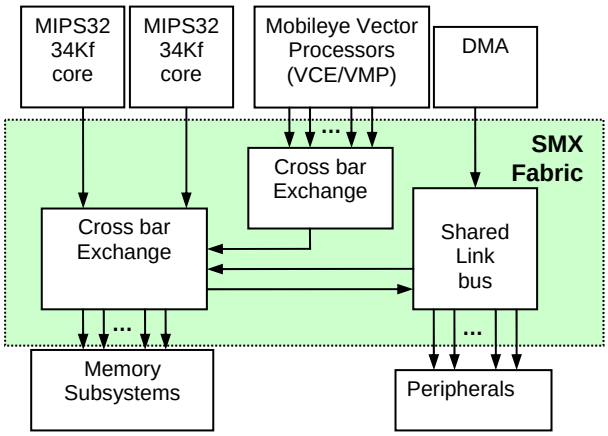

Figure 1 – Mobileye EyeQ2 SoC

Central to the SoC design are two instances of MIPS32 34Kf processor cores. The 34Kf core is a multi-threaded RISC architecture designed to exploit multi-threading in embedded applications by processing multiple software threads in parallel, while masking the effect of memory latency and allowing the user to allocate dedicated processing bandwidth to real-time tasks

The MIPS32 34Kf cores work in conjunction with an array of Mobileye VCE/VMP processors that are optimized to address automotive vision processing applications

Mobileye EyeQ2 uses two types of proprietary vector processors: VCE (Vision Computing Engine) and VMP (Vector Microcode Processor). VCE DSPs have a fixed logic architecture to address image preprocessing and search operations, including object classification, object tracking, disparity identification (for applications using stereo image) and filtering for vision analysis. The VMP engines provide further image and object processing using a high bandwidth architecture utilizing parallel vector, scalar and look-up table units.

Two MIPS 34Kfs, Mobileye VCE/VMP processors, the DMA, and peripheral cores are connected by the Sonics Multi-Service Exchange (SMX[tm]) fabric which enables the efficient functioning of these cores.

A Sonics 128bit 166MHz SMX (OCP) interconnect services the heavy duty traffic between these SoC components. An additional low bandwidth 32bit bus connects low bandwidth peripherals and VCE/VMP DMAs' configuration register via one of SMX ports.

The EyeQ2 architecture was designed for intensive parallel processing targeting the challenges of vehicle vision applications. EyeQ2 silicon systems available by the end of 2006 will be capable of concurrent detection of vehicles, motorcycles, pedestrians, traffic lights and signs, etc.

The Bus RRT system consists of both on-chip IP and analysis and software communicating over a high performance trace port (System Navigator Pro[tm] (SNP)) probe and was developed to provide visibility into the various interconnection points of the Mobileye EyeQ2 architecture.

The IP developed for the Bus RRT system is designed to record request and response bus events at the socket interface and measurement of one processor (bus master socket) with expandability to allow concurrent viewing of key parameters of all masters simultaneously. This system analysis implementation allows capture of information about core load/store operations and their latency for the different socket masters, and exports them over dual trace ports to the SNP probe, along other trace and analysis data, in particular MIPS EJTAG and PDtrace[tm] interfaces that are used in providing a complementary run control and trace analysis views of the MIPS processor operations.

While in this paper we present a specific architecture and its interfaces (to MIPS and VCE processor cores), both the on chip interconnect and analysis systems discussed can be applied to other processor cores or bus architectures under a similar generic scheme without any significant loss of generality.

SMX and Socket Based SoC Design

Figures 2 and 3 diagram demonstrate some of the features of a socket based design used in this design. The sockets accept initiators (masters) and targets (slaves), with the circuitry of the socket encompassing the necessary state machines, gating and muxing circuitry, and wiring to effect the desired data flow (QoS, multi-threaded non blocking communication, security features, dynamic power gating, etc.) characteristics.

Although optimal and assumed for OCP 2, SMX bridges are also available for ARM's AMBA AHB and AXI to simplify utilization of legacy hardware; bridges for other arbitrary existing interconnect structures can also be developed.

This socket based interface simplifies addition, removal, or replacement of IP blocks, including development of test suites to address simulation, verification, and optimization of the design

The Intelligence is in the Agents

- Agents provide…
  - Protocol conversion
    - Agent adapts to IP core
  - Decoupling of IP cores from fabric
    - Provide local, isolated environment
  - Data flow services
- Agent data flow services
  - QoS-based arbitration
  - Power management
  - Access security
  - Error management
  - Burst, width, and command conversion

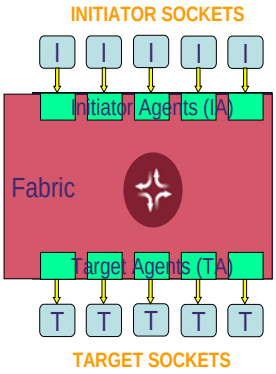

Figure 2 – Sonics Socket Based Agents

The complexities involved in communication centric designs and their flexibility and parameterization make embedded features for sophisticated latency and performance analyses an important consideration, not just to further optimize the current design, but to quickly and easily quantify requirements and enhancements for a systems optimization of the derivative designs.

These common sets of requirements for high performance interconnect networks to provide optimized system performance and the need to be able to monitor this performance transparently is addressed by the combination of the Sonics SMX Smart Interconnect with the FS2 RRT Monitoring IP and Analysis Tools.

Socket Monitoring via Smart Interconnects

Recognizing the need to monitor in-band and out-of-band data for values, latencies, and any other data that is potentially relevant for both debug and optimization, the SMX architecture includes features that enable data flow control, access, and monitoring of performance and optimization parameters. As in any design, there are always tradeoffs between the need to have visibility into the interconnect and communications operations and the resources that are required to be able to effectively and sufficiently monitor key information. These resources typically include some tradeoff of on-chip instrumentation resources, IO and trace buffering bandwidths, and the overall impact to the design. The specifics of the monitoring function in some cases vary with the size and performance or the interconnect structure.

In the SMX architecture, there are basically three classes of interconnect structures, Crossbar Exchanges, Shared Link Exchanges, and Extended Link exchanges, each with their own features and analysis requirements. SMX Crossbars allow the fastest unimpeded connectivity, while Shared Links require less overhead of additional gates and also support the optimal data flow by QoS selection. Exteneded Links support more widely separated cores. Related Sonics products also provide connectivity of slower peripheral or optimize scheduling to improve the efficiency of DRAM controllers and the effective bandwidth of external DRAM.

- Exchanges
  - Cross-bar (XB)
  - Shared Bus (SL)
  - Extender (EL)
- Pipelining options
  - Registering at socket interface
  - Register points (RP) at agent-fabric edge
  - Pipeline points (PP) between exchanges
- Register Target (RT) to access SMX services
- Multiple socket support

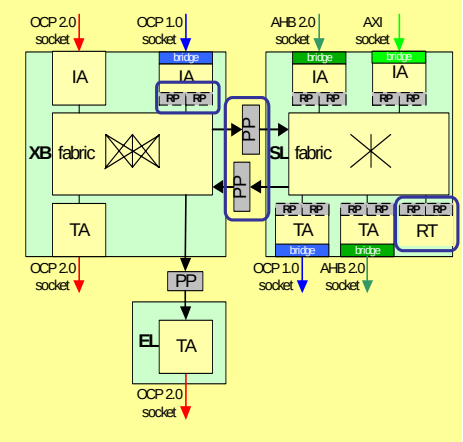

Figure 3 – SMX Internal Structure

SMX allows several methods of accessing performance and analysis information from the interconnect structure. Each type of interconnect segments allows integration of test ports at different segments internal to the interconnect to allow debug access. Signal information can also be accessed at the socket level which the implementation of the EyeQ2 Bus RRT.

An RRT Analysis Environment

The SMX Crossbar and Shared Link Exchanges connecting the processor masters (MIPS32 34Kf cores, VCE/VME cores and DMA) and slaves (off-chip DDR memory and on-chip ISRAM, etc.) in the EyeQ2 SoC present a complex data communications network. The data width of the Sonics SMX communication links can vary based on connection to specific cores, which include 32, 64, or 128 bit buses. The MIPS32 34Kf cores, as an example, operate at twice the system clock frequency and interfaces to the SMX over a 64 bit OCP2 bus. The SMX includes resources for resolving the mixing of bus widths and speeds across different blocks, which simplifies efficiency and optimization of performance with regards to different data rates, clock rates, etc. of the system cores.

The EyeQ2 system analysis environment consists of two major subsystems

1. PDtrace for each 34Kf core. PDtrace interfaces support an aggregated processor trace port for both of the 34Kf core trace outputs. The core and system clock speed is reduced so that the PDtrace port can sustain the required trace bandwidth.

2. Bus Request-Response Trace (RRT) allows the trace of one single bus socket, all masters in the system simultaneously or a selection of masters. The RRT trace buffers each request-response output and includes a trace "funnel" to route the buffered outputs to the off-chip trace port. Like for the PDtrace, system clock speed is reduced so that the RRT port can sustain the required maximum trace bandwidth.

RRT and PDtrace data are sent off-chip over a dedicated 16 DDR channel trace port. Both PDtrace and RRT trace port interfaces are supported via a single FS2 System Navigator Pro probe, using two Mictor38 connector interfaces, each with its own independent clock source. The probe combines the trace inputs from the two sources and records them in a common memory buffer.

The PDtrace Mictor includes a common JTAG connection and PDtrace trigger pins for trigger and trigger acknowledge. All applicable features of RRT and PDtrace can be configured via this JTAG port.

RRT Operations

The RRT provides for capture and collection of the following information of the EyeQ2 operations. All Capture is done on chip at the RRT agents and exported via the RRT port:

a) Recording of specifics of master-slave socket transactions and the number of clocks of delay between each request and response.

b) Captured timing and latency of read cycles. Burst reads are reported on arrival of the first requested word or on the arrival of the last word of the burst.

c) Transactions between one (selected) master and all slaves it transacts with, or several masters at the same time. These masters may include any of - the two 34Kf cores, one active VCE/VME channel (selected as output of the Crossbar) and one active channel of the DMA.

Trace collection allows overall capture for an extended (for example. at least one video frame) processing period using the Memory buffer in the probe. Concatenation of multiple frames may be done as a post processing stage on exported RRT trace files.

Post-trace software provides most significant post processing and views of transactions and delay times over varying periods of time for both single and multiple cores. RRT data is correlated and used in conjunction with PDtrace data to provide a picture of system operation.

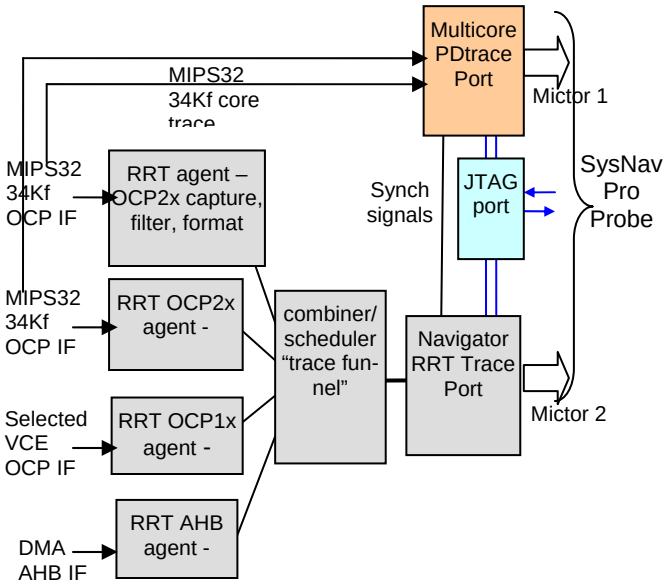

Figure 4 - RRT and PDtrace subsystems

RRT Implementation

The on chip component of RRT consists of 3 primary On-Chip Instrumentation (OCI) IP blocks, all of which are implemented in synthesizable Verilog code.

a) RRT agents, specific to processor or core level interface to capture and buffer relevant trace information based on system operations and trace configuration.

b) The RRT "trace funnel", which provides the aggregation of trace information from all RRT agents and combines and schedules the trace information for export, and

c) The RRT Navigator Trace Port, which handles communications with the off chip probe.

Configuration of each block is performed via JTAG, over a common JTAG chain.

A user defined set of Bus RRT fields are captured, including:

- the master ID (only required if multiple masters are being recorded at one time)
- Slave ID based on unique address bits that identify one slave from another. Hardware in the agents can recog-

nize the memory mapped areas and encode them into the slave ID field.

- protocol and traced bits that determine the alignment of a read response cycle to its "parent" request cycle
- request and/or response cycle type (or encoded in other fields)
- Cycle type – read vs. write, single access vs. burst.
- buffer overflow indicator bit, indicating if the RRT record has lost synchronization with the processor operations
- Trace of upper address bits to determine code vs. data memory mapped regions. Two defined modes: Fast (partial) address field and Full (complete address field) are a user selectable options.
- trigger signal – a 1 bit signal to allow on-chip subsystems to send a trigger signal to the probe

Bus RRT records are further broken down into two mode – Fast and Full. Fast mode is limited to a single cycle frame and includes socket level control signals characterizing the bus transfer along with buffer overflow and/or trigger indicators. Full mode includes control signals as well as full address trace, based on a memory map of necessary upper addresses, and typically is transmitted over multiple trace clock cycles

The capture of this data via RRT allows the following to be performed during chip level operation

a. Measurement of a processing loop such as frame time. The SysNavPro trace depth handles 2 Gigbyte of trace data (sufficient for example for the trace of at least one video frame).

b. Capturing available information for aligning socket measurements with core processor execution to correlate cause-effect of code execution to socket traffic based on coordinated recording of trace from both sources.

c. Capturing available information on aligning socket measurements to correlate each hardware thread to the data transfers that each processor generates.

d. Extraction of thread information extractable from socket address bits traced. Post-trace software can display per-thread socket transaction information providing valuable information to users on the density of transactions over time and the delays associated with those memory accesses, generated for each hardware thread.

e. Post-processing of the trace matches up requests and responses (using the socket protocol and possibly ID bits), and calculate the delay between them based on timestamp values stored along with trace. Sysnav Pro supports a trace timestamp that provides an accurate timeline of each request-response frame.

A RRT triggering system is implemented within the Probe (off-chip) and includes event monitoring of all captured control and address signals to control start/stop pf capture of trace information in the probe. This trigger may also be used to put one or more cores in Debug mode and to communicate with the processor and PDtrace subsystems. On-chip trigger output pins indicate to the probe status of the processor cores.

The probe and on chip logic have a common triggering communication to allow the probe to enable and disable/stall RRT operations in conjunction with PDtrace operations. The triggering scheme also communicates stalling of the trace capture based on processor status.

Post-Trace Analysis Tools for RRT and PDtrace

RRT is supported by a set of control and display views and utilities to support analysis of RRT and PDtrace data. Additional visualization can be supported via export of trace to third-party tools. Control setup includes the setting of master trace priorities and selecting which masters are to be in the trace; trigger setup for precise post-trigger positioning and reading trace and formatting data for additional analysis views. These additional views may include

Raw State View for RRT - basic acquisition is displayed as a state display that shows one line per trace frame with columns correspond to the trace fields: transaction type (read/write, request or response), master name, slave name, transaction ID or outstanding request count, buffer overflow and probe generated trace timestamp values.

Aligned State View for RRT –alignment *concatenates* two frames – a request cycle and its matching response cycle and a delta timestamp between current and next transactions.

Correlated view of RRT and PDtrace - allows viewing of common PDtrace and RRT data captured at a common timestamp with a known or defined offset. It also allows RRT and PDtrace data to be locally correlated based on address values, or common triggers, markers and instruction (read/write/burst) types captured in both the PDtrace and RRT. Correlating socket traffic with instructions defines a processor to bus level relationship, such as determining which thread caused a read or write socket cycle

Graphical Display - FS2 trace solutions are supported by a multi-view (Navigator) GUI, which is customized for RRT data display as captured by the probe. The Analyzer GUI allows complex triggering of capture and display of RRT information as waveform and state views the. The GUI includes utilities for control of bus event monitoring and

template based triggering based on captured trace information.

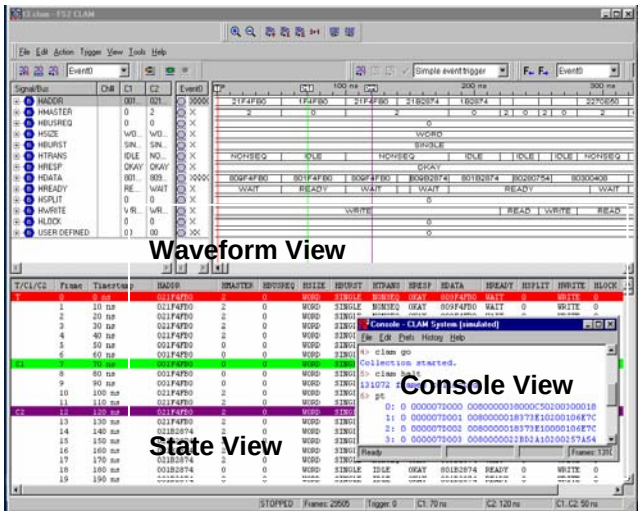

Figure 5 – RRT Navigator Graphical Bus Trace View

RRT Application in the Mobileye EYEQ Development

Mobileye's new generation EyeQ2 SoC targets compute-intensive vision processing applications operating in real-time environment of a vehicle. These applications place a high priority on both efficient and reliable communications between processing resources.

Internal buses traffic visibility and measurement capabilities are an important component of this SoC toolkit. In complex SoC environment, software performance can be dramatically impacted by a range of problems caused by poor bus traffic balancing, bad QoS management, unprotected concurrent memory access, incorrect cache coherency management. etc.

Poor on -bus traffic balancing causes latencies that can seriously impact the real-time capabilities for processing at video frame rates. Mobileye algorithm timing is a data dependent, since, in the real time environment surrounding a vehicle, each video frame is unique, based on the road situation. The need for optimal on chip data access is one of major drivers for a high performance bus traffic monitoring tool enabling statistical measures of bus latencies. The Bus RRT approach provides this by combining flexibility of fast (partial)/full bus tracing at almost real-time rates with powerful post-trace analysis correlating on-bus and on-CPU events.

An important feature of RRT is ability to correlate and resolve operations in the OCP fabric and MIPS32 34Kf architectures and processes to provide understanding of system operations. A bus event that can be matched per CPU and even per TC thread significantly simplifies an analysis by a developer.

RRT allows this with zero-impact on processing timing, selective tracing, trace data compression, and concurrent tracing of the SMX and two MIPS32 34Kf core operations by a FS2 Navigator probe. Together with PDtrace, RRT gives a serious advantage for system analysis compared with other monitoring alternatives.

Productivity of this solution was also an important criteria for focusing on trace based system analysis. Mobileye estimation for RRT productivity compared with a SystemC model alternative has showed up to 100 times speed advantage. From a business model, RRT is much simpler than SystemC model based analysis approaches, once third party IP integration overhead and all expenses of licenses for external developers are factored in .

Comprehensive RRT control by a flexible triggering system is an additional "plus". EyeQ2/RRT tracing control enables optimal utilization of on-probe memory buffer and conditional debugging control.

The RRT system is a result of close cooperation between Mobileye and FS2 and the support of MIPS Technologies and Sonics, as IP providers. The system provides a full, cheap and efficient tracing/debugging solution for a complex dual-MIPS32 34Kf SoC environment. It will be an important component in the EyeQ2 tool chain enabling Mobileye and customer development of high-quality products meeting requirements of automotive industry.

References:

(1) "On-chip instrumentation aids OCP debugging" EE Times 09/26/2005 http://www.eetimes.com/showArticle.jhtml?articleID=171100311

(2) "Integrating On Chip Debug Instrumentation and EDA Verification Tools" DesignCon East 2005 www.fs2.com/pdfs/DesignCon%20East%202005_FS2_ID1202_final.pdf

(3) "Interconnect Design for Advanced SoC and NoC," Jari Nurmi, Hannu Tenhunen et al, eds. Ch. 14, "Socket-Based Design Techniques using Decoupled Interconnects." Drew Wingard, PhD. CTO, Sonics Inc. Kluwer Academic Publishers, 2004.