Emerging Technology...

System-on-chip (SOC) and its Effect on Microcomputer Bus Products

Wade D. Peterson, President & CEO
Silicore Corporation, Minneapolis, MN
TEL: (612) 722-3815  FAX: (612) 722-5841
peter299@maroon.tc.umn.edu  www.silicore.net

© 1999 Silicore Corporation  All rights reserved
What We'll Cover Today

- Short introduction to system-on-chip & IP core technologies.
- The impact of SOC on the µC board industry.
- The need for new VMEbus interface chips.
- A sneak preview of new VMEbus IP cores.
- The WISHBONE interconnection (a µC bus for SOC).
What the Heck is System-on-chip (SOC)?

- System-on-chip is exactly what it says...the ability to build entire systems on a single semiconductor device.
- This includes CPU, RAM, ROM, I/O, FIFO, network interface, bus interface and all of the ‘glue’ logic.
- It’s a lot like VMEbus or cPCI, except that the integration occurs at the chip level.
What’s this ‘IP Core’ Thing Anyway?

IP core: Intellectual Property Core

These are the building blocks for SOC.

They contain the functional elements such as CPU, network interfaces, bus interface etc.

There is currently a budding IP core industry.

Many new IP cores are being released almost daily.
So, What's all the Fuss About?

- SOC is a new paradigm in the semiconductor world.
- This is a juggernaut that can't be ignored!
- It alters the way we buy semiconductor technology.
- It's brought about by new chips and development tools.
- The full impact will be felt over the next 5-10 years.
Effects on the µC Board Industry

**SOC will not undermine the usefulness of µC boards:**
- There will always be a need for board level products.
- End users won’t want to tinker around with SOC.

**SOC will change:**
- Several business aspects of µC boards.
- How we design and deliver technology on µC boards.
Business Aspects

- SOC technology will allow µC board manufacturers to do a better job of differentiating their products.
- That's because SOC allows more choices for board builders.
- This will result in better profit margins for board manufacturers, and more choices for end users.
Technology Aspects

- Board makers and end users will also benefit, as it gives them a greater choice of technologies.

- Let’s take a look at some of these choices...
We are starting to see a trend where the latest technologies are available first as IP cores.

Great examples: NGIO, CPU and DSP cores.

Companies will need to adopt SOC technologies if they want to maintain a technological ‘edge’.

Small, innovative companies are also creating IP Cores.

That’s because they’re relatively cheap to develop.
The Big Ugly: Parts Obsolescence

- Parts obsolescence is a major concern in the industry.
- Soft IP cores are portable.
- As target devices (parts) become obsolete, they can be upgraded to the latest technology.
- This is really a return to the multiple vendor sources that we were accustomed to in the 1980’s.
- This means less dependence on single source vendors.
The Need to go Faster, Faster and still Faster

- SOC is an inherently high speed technology.
- On-chip capacitance and inductance is very low especially when compared to circuit board technologies.
- FPGA toggle rates are now approaching 1 GHz.
- SOC is a very attractive way to increase system speed.
Other Benefits...

- SOC solutions require less power.
- SOC solutions are very compact.
- Parts are available in a wide variety of costs, speeds, packages, voltages and temperature ranges.
- Radiation hardened parts are available (military!).
- JTAG support is very common, and easy to do on FPGAs.
Drawbacks of SOC

- The industry is in the ‘wildwest’ stage
- Some key, de facto standards haven’t developed yet
- Tool costs are high, but are coming down
- Lack of infrastructure among µC board manufacturers
- Substantial learning curves
- Lack of IP cores for VMEbus
- Lack of an interconnection standard
A Quick Overview of SOC integration

Since many system-on-chip technologies are unfamiliar to the audience, let’s take a look at how it’s done...
**SOC / System Integration Philosophy**

- Microcomputer bus systems are integrated at the board level.
- SOC systems are integrated at the IP core level.
Target Devices

- SOC is used on FPGA and ASIC target devices.

- FPGA: Field Programmable Gate Array.
  - Lower entry cost, higher production cost (<10K pcs/yr).
  - Shorter lead times.

- ASIC: Application Specific Integrated Circuit.
  - Higher entry cost, lower production cost (>10K pcs/yr).
  - Longer lead times.
Integration of Soft IP Cores on an FPGA

- **Commercial IP Core**
- **User IP Core**
- **User Logic**

**VHDL or Verilog Source**

**Vendor Specific I/O, RAM**

**Integration**

**Simulation**

**Synthesis**

**Place & Route**

**Completed Part**

**Typical Software Suppliers:**
- ModelSim
- Synopsis
- Synplicity
- Accolade

**Typical Vendors:**
- Actel
- Altera
- Lucent
- Quicklogic
- Xilinx

- **Vendor Specific**

- **Vendor Specific**

- **Vendor Specific**

- **Vendor Specific**
The Need For New VMEbus Semiconductors

- VMEbus interface chip technology is beginning to lag
- Semiconductor manufacturers aren’t keeping up anymore
  - New interface standards (VME320, T&M, hot-swap, etc.).
  - Wide temperature and radiation hardening
  - Packaging and voltages
  - JTAG support (especially for ball grid array packaging).
We decided to take advantage of this new situation by creating some new VMEbus interface chips. The new product is called VMEcore™.
VMEcore™ is a VHDL Soft IP core toolkit.

Suitable for FPGA or ASIC.

Supports all interfaces in ANSI/VITA 1-1994 (VME64).

Insures compliance with the VMEbus standard.
Local WISHBONE interconnect.

Full VMEbus bandwidth to 40/80 Mbyte/sec.

Incorporates third generation synchronous technology.

Clever design eliminates common interface problems.
The full toolkit will support:

- Master
- Slave
- Location monitor
- Interrupter
- Handler
- Bus arbiter
A general purpose VMEbus IP core is very difficult.

There are $10^{22}$ combinations of VMEbus interface!

VMEcore™ supports all of these combinations!
A unique software package called the VMEbus Interface Writer™ creates a VHDL soft IP core to the user’s specifications.

VMEbus interface options are entered into the software, and it generates a VHDL soft IP core!

A complete test bench (with test vectors) is also created!

Here’s a demonstration..
There is no standard interconnection architecture on SOC, at least nothing similar to a microcomputer bus. This makes it difficult to integrate systems. We have created the WISHBONE interconnection to solve this problem. WISHBONE is very similar to other µC buses. It's tailored toward SOC applications.
WISHBONE Architecture

Simple, compact, logical IP core hardware interface

It's independent of:

- Hardware technology (FPGA, ASIC, etc).
- IP core delivery method (soft, firm or hard).
- Hardware description languages (VHDL, Verilog®, etc).
- Synthesis, router or layout tools.
WISHBONE Specification

- Informal, open standard
- No royalties or licensing required
- The bus specification can be downloaded for free
- See our website at www.silicore.net
WISHBONE Features

- Synchronous protocol.
- One data transfer per clock cycle.
- Variable speed handshaking protocol.
- Speed is limited only by the target semiconductor technology.
WISHBONE Features

- MASTER/SLAVE architecture
- Up to 64-bit address and data paths (expandable)
- BIG ENDIAN or LITTLE ENDIAN byte lane ordering
- Multiprocessing support
- Memory mapped, FIFO and crossbar configurations
- Various interconnection methods (dual I/O, three-state)
WISHBONE Bus Cycles

- Full set of popular data transfer protocols, including:
  - READ/WRITE cycle
  - BLOCK transfer cycle
  - RMW cycle
  - EVENT cycle
Typical WISHBONE READ Cycle

Here's a typical READ Cycle:

- **CLK_I**: Clock signal for the master.
- **ADR_O**: Address output for the slave.
- **DAT_I**: Data input for the master.
- **DAT_O**: Data output for the slave.
- **WE_O**: Write Enable output for the slave.
- **SEL_O**: Selection output for the slave.
- **STB_O**: Start of Transfer output for the slave.
- **ACK_O**: Acknowledge output for the slave.
- **CYC_O**: Cycle output for the slave.

**WISHBONE MASTER**

- **CLK_I**: Clock input for the master.
- **ADR_I**: Address input for the master.
- **DAT_I**: Data input for the master.
- **DAT_O**: Data output for the master.
- **WE_I**: Write Enable input for the master.
- **SEL_I**: Selection input for the master.
- **STB_I**: Start of Transfer input for the master.
- **ACK_I**: Acknowledge input for the master.
- **CYC_I**: Cycle input for the master.
- **TAGN_O**: Tag input for the master.

**WISHBONE SLAVE**

- **CLK_I**: Clock output for the slave.
- **ADR_O**: Address output for the slave.
- **DAT_I**: Data input for the slave.
- **DAT_O**: Data output for the slave.
- **WE_O**: Write Enable output for the slave.
- **SEL_O**: Selection output for the slave.
- **STB_O**: Start of Transfer output for the slave.
- **ACK_O**: Acknowledge output for the slave.
- **CYC_O**: Cycle output for the slave.
- **TAGN_O**: Tag input for the slave.

**MASTER SIGNALS**

- **CLK_I**: Clock signal for the master.
- **ADR_O**: Address output for the slave.
- **DAT_I**: Data input for the master.
- **DAT_O**: Data output for the slave.
- **WE_O**: Write Enable output for the slave.
- **SEL_O**: Selection output for the slave.
- **STB_O**: Start of Transfer output for the slave.
- **ACK_O**: Acknowledge output for the slave.
- **CYC_O**: Cycle output for the slave.