

**A2B Synchronous System Bus
and
A2R Register/Peripheral Bus
in an
ARC based Multiprocessor System-on-chip**

Introduction

A2B is a high performance system bus designed for use in synthesizable designs. It is specifically developed to meet the challenges of multiprocessor and multiple DMA / IO processor, System-on-chip designs. A2B is designed to have the highest possible bus occupancy so that the sustainable bus bandwidth closely approaches the available peak bandwidth of a given configuration.

A2B is a fully synchronous system bus that is user configurable to provide a wide variety of performances to best match the system level requirements of a particular implementation. The user (system architect) can configure the system for address and data widths, may add special bus fields to transfer custom information and may select the arbitration algorithm. Bridges can also be included to allow different bus widths in separate sections so that a variety of devices can connect together. This allows, for example, a collection of CPUs to have a very wide bus connection to system memory and for slower I/O devices to have a narrower bus connecting through the bridge to the system memory.

A2R provides an interconnection mechanism between control registers in an ASIC design and any number of control devices; CPUs, debug ports etc. The bus is especially suited for synthesizable designs. It is specifically developed to meet the challenges of long interconnect delays in large System-on-chip designs and can be tailored to match system clock rates.

A2R is also a fully synchronous bus and again user configurable to provide a wide variety of performances to best match the system level requirements of a particular implementation. It may be configured for address and data widths, the addition of special bus fields to transfer custom information and the selection of the arbitration algorithm. The bus may also be segmented to allow concurrent access within segments.

The ARctangent-A4 microprocessor is a user-customizable 32-bit RISC core for ASIC, system-on-chip (SoC), and FPGA integration. It is synthesizable, configurable, and extendable - developers can readily modify and extend the architecture for specific applications. This unusual flexibility lets developers creatively tailor the hardware for the software, instead of bending the software to fit a rigid CPU architecture. Developers can choose to optimize the ARctangent-A4 processor for computational performance, digital signal processing, I/O throughput, application-specific algorithms, power consumption, silicon area, or cost.

Advanced Architectures

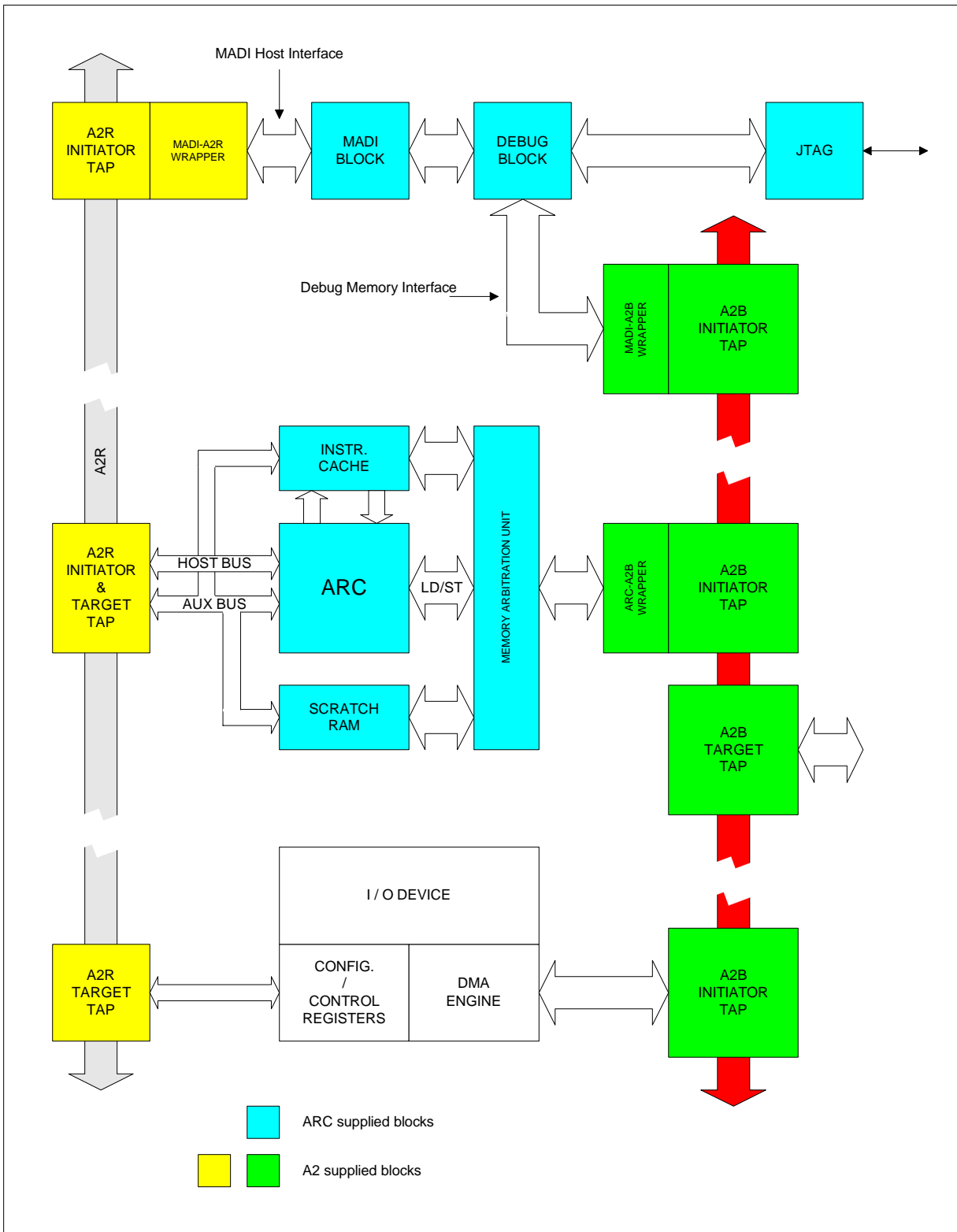


Figure 1: General Configuration for ARC in A2 Bus Environment

Advanced Architectures

ARCs and A2 Buses

The ARC system provides three primary interfaces to and from the CPU. The primary interface is to system memory and comes from the “Memory Arbitration Unit” which selects from a number of memory access request sources within an ARC structure that desire memory access and then forwards the request to System Memory. The other two interfaces are the Host Interface for debug operations and the Auxiliary Interface that allows the ARC CPU to read on-chip registers.

When multiple ARCs need to co-exist in a system many other requirements arise that often cause conflicts. For example, if the ARCs need to share a system memory amongst themselves and other on-chip devices such as DMA devices a more sophisticated mechanism is required to facilitate high-performance access to the common resource and ensure that sufficient bandwidth is provided. Also consider whether or not registers within an I/O device should be visible to only one ARC or multiple ARCs and indeed should the intimate registers of an ARC be visible to other ARCs, perhaps for debug in a deployed system when the “Host” debug is not present.

To address these issues Advanced Architectures’ system buses can be used. The A2B bus is a scalable mechanism that provides the highest possible performance; peak bus bandwidth can be sustained provided that initiators and targets can supply the traffic. This is achieved via its unique architecture that eliminates “busy” cycles from the bus and organizes transactions based on target occupancy. This ensures that a correctly configured A2B bus is never a system bottleneck. The A2R bus is designed as a utility mechanism that provides debug and register level access to configuration and control modules in a system. These accesses are usually not time-critical and the full performance of the A2B bus is not necessary. By using the A2R to connect the ARCs Host and Auxiliary interfaces access can be provided to all system registers by any (or selected) bus masters including the ARC standard debug block for multiple ARC systems (MADI).

Figure 1 shows the general concept of connecting ARCs to the A2B and A2R buses. The ARC Host and Auxiliary buses tie directly to the A2R TAP and the ARC appears as an A2R target, via the Host Interface and as an initiator via the Auxiliary Interface. The MADI block connects to a standard A2R TAP via a small “Wrapper” that translates the MADI signals for the A2R. This mechanism is transparent to the debug system and user. The A2R TAPs can also be used to interface to control registers in other system level modules such as DMA controllers, small peripherals etc. The standard ARC memory debug port can be connected to the A2B bus via a “Wrapper” that allows debug access to the entire address space supported by the A2B bus. The ARC memory system can connect to the A2B via the Memory Arbitration Unit as shown in the figure. This provides a simple and quick connection to the buses that is often adequate for an application.

If higher performance is required the A2B can be connected directly to special ARC caches (not standard ARC caches) that can accept and provide data at the full rate of the A2B. **Figure 2** shows a 128-bit wide A2B bus that connects directly to caches and still provides 32-bit access for non-cacheable accesses. The caches are configured such that a natural burst size on the bus matches the cache line size to optimize transfers over the A2B bus. Backdoor accesses to the caches, for debug and configuration operations are still available via the ARC Auxiliary bus. Non-cacheable accesses still use the ARC DMP and MAU and a 32-bit wide path is provided by the A2B wrapper to allow direct access to system resources that are not cacheable. This mechanism can also be used to provide interlocked or atomic operations to system memories and devices for secure semaphore signaling between multiple processors. Advanced Architectures also has in development a queuing memory

Advanced Architectures

block that attaches to the A2B bus and provides secure access to a configurable number of system level queues to enable multiple processors to intercommunicate with minimal overhead.

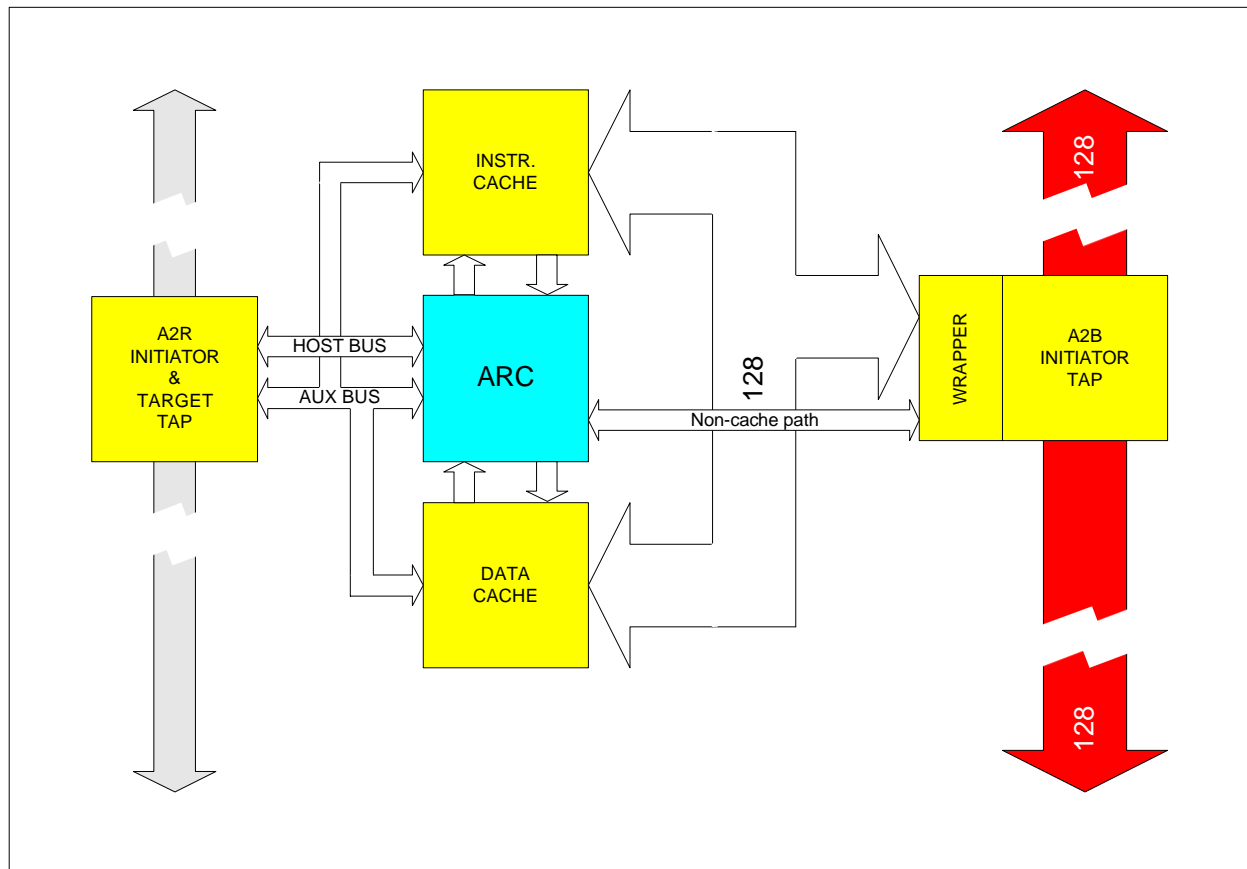


Figure 2: ARC with high performance caches tuned to A2B width

The A2B bus can also support cache coherency protocols for situations that need to share cacheable data. Cache coherency via SNOOP is available over the A2B bus and can be configured for Snoop Write Hit Invalidate or Snoop Write Hit Update and Snoop Read Hit Retry or Snoop Read Hit & Supply. Coherency protocols can be avoided if all the software specifies shared data structures as "volatile". This will cause the caches to be bypassed and access will be made directly to the A2B bus. The system architect must judge between the added complexities of cache coherency against the performance of all shared data being non-cacheable.

The A2B bus can also be used to create hierarchies of buses that can more easily span a System-on-Chip design. **Figure 3** shows an example where three different sizes of buses are used to interconnect various on-chip modules and then bridges are used to connect the buses together. The bridges provide buffering and bus width re-sizing and may be configured to be isolating or non-isolating mechanisms so that address spaces may be optimized. Care must be taken to ensure that any cache coherency protocols are congruent to ensure that there are no stale data issues. It is recommended that SNOOP protocols are only used within a single A2B segment and if cache coherency is required above this level that directory-based mechanisms are used. The A2B may be configured to support any of these protocols by carrying the physical layer within its fabric. It does not, however, provide the logic and control mechanisms that are required.

Advanced Architectures

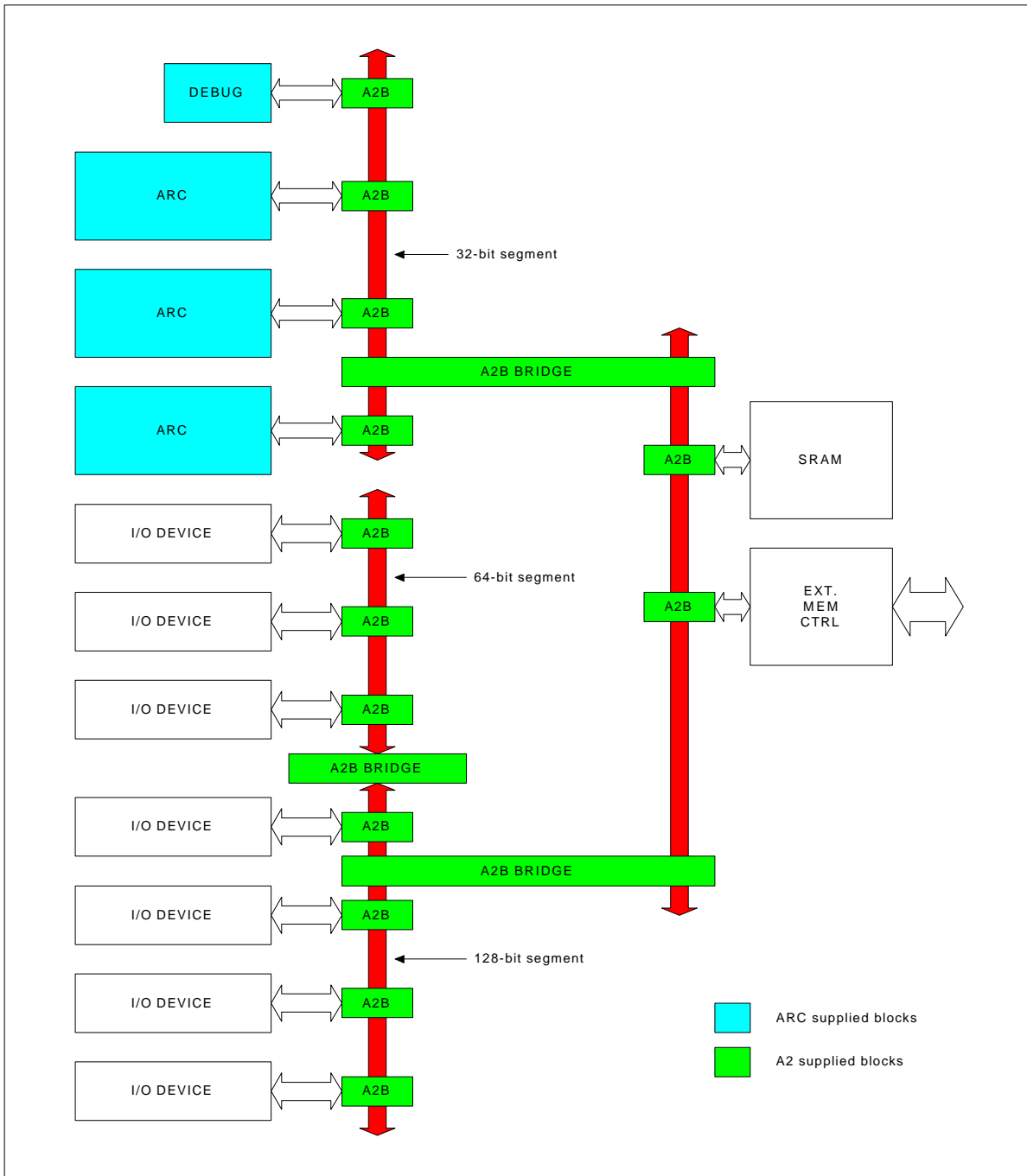


Figure 3: Hierarchy example of A2B buses

In the example shown in **figure 3**; note that there is a bridge between the 64-bit segment and the left-hand 128-bit segment that then itself bridges across to the rightmost 128-bit segment. Any configuration of bridges is possible as long as there is only one path between any two segments. It would not be appropriate, for example, to put a bridge between the 32-bit and 64-bit segments, as there is already a path via the 128-bit segments. Care must also be exercised in creating too many levels in the hierarchy as latency goes up for every bridge traversed even though there is no loss of bandwidth.

Advanced Architectures

Figure 4 shows an example of a hierarchy of A2R buses. The A2R does not use a bridge, per se; rather it uses what is called a gate. In order to access through the gate an initiator must assert a higher-level request. When this request is granted the initiator has access to the higher level and can send its access upwards. This mechanism allows the initiators in separate segments to access within their segment at any time and only at a higher level through a higher-level arbitration. The A2R vertical segment to the left in the drawing performs arbitration amongst the three gates and the TAP to the ARC MADi block. Once a request is granted the initiator will send an access to across this bus to another TAP or GATE.

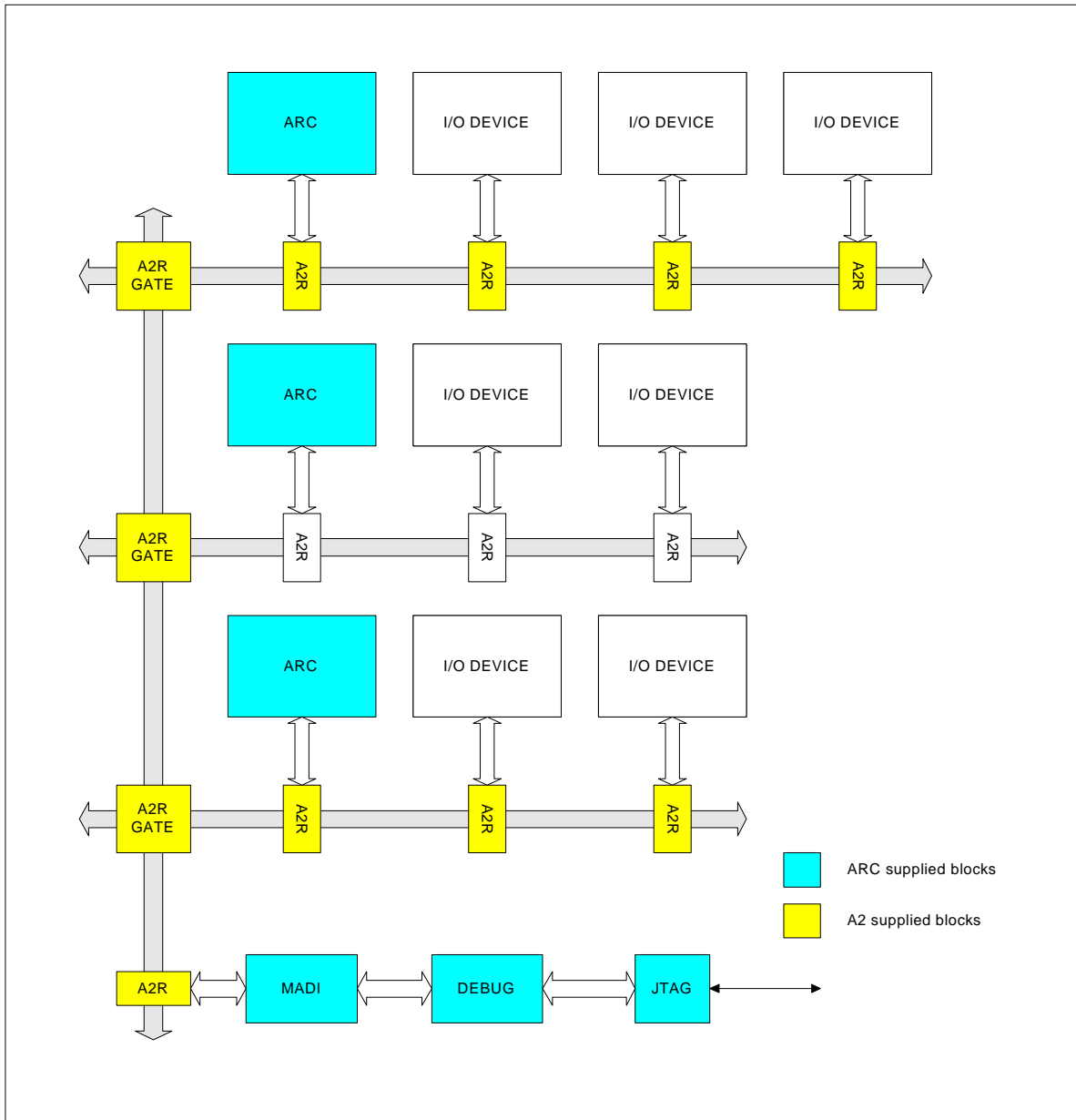


Figure 4: Hierarchy example of A2R buses

Advanced Architectures

A request arriving at a destination gate will assert a high priority request so that it gets the next transaction on the downstream segment. In the upper right-hand side of the figure is a gate that only has target devices below it. This target gate will only open when an access to the address space below is received. This is used primarily to reduce the loading of a segment and if a serial point-to-point fabric is selected then it will also reduce access times when not accessing the devices in the lower segment.

The A2R can be configured as either a parallel OR fabric or as a point-to-point fabric. The point-to-point fabric has several advantages as all bus links have a single fan-out and fan-in and may also be pipelined. This eases the constraints required for the bus in the physical design process as the bus can be tuned to achieve a specific clock speed rather easily and long lines across a chip can be accommodated. In **figure 4**, for example, the leftmost bus could well be a physically long bus but by pipelining it at one or more A2R GATES it can be broken up into different clock cycles so that timing can be met. As the accesses over this higher level are less frequent then the added access times are not usually an issue.

Advanced Architectures

A2B Feature List

Data Paths

- 8, 16, 32 ... (any power of 2)-bit Write data bus
- 8, 16, 32 ... (any power of 2)-bit Read data bus
- 8, 16, 32 ... 128-bit Address bus
- Custom buses

Protocols

- True split transaction reads
- Split read and write buses
- Clock rate independent protocols
- Multiple byte and burst modes
- Tagged transactions for secure operation
- Single protocol for all performance levels

Addressing

- Customizable assignment of address space
- Big-endian and little-endian support

Devices

- Any number of transaction initiators
- Any number of transaction targets

Arbitration

- User configurable/customizable
- Base algorithms provided
 - Simple priority
 - Round-robin
 - Multiple levels

Extensions

- Parity protection on any and all buses
- Virtual addressing and translation
- Central address translation support
- Address burst and complex addressing
- Multiprocessor protocols
 - Interlock / Atomic operations
 - Cache coherency via SNOOP
 - Cache coherency via Directory
- Error and Retry protocols
- User-defined bus fields
- Bus monitoring

Bridges

- On-chip between segments
 - supports different bus widths
- Off-chip
 - supports direct chip-to-chip
 - supports tri-state bus

A2R Feature List

Data Paths

- Multiplexed address and data bus for minimum – interconnect
- 8, 16, 32 ... or any width of address and data bus
- Custom bus widths may be specified

Protocols

- Simple master and target interfaces
- Clock rate independent protocols
- Single protocol for all performance levels
- Multiple master and target capability
- Multi-segment capability
- Direct attach of ARC host and auxiliary buses

Addressing

- Customizable assignment of address space
- ARC debug compatible

Devices

- Any number of transaction initiators
- Any number of transaction targets

Arbitration

- User configurable/customizable
- Base algorithms provided
 - Simple priority
 - Round-robin
 - Multiple levels

Extensions

- Parity protection
- Error protocol
- User-defined bus fields
- Bus monitoring

Bridges

- On-chip between segments

Advanced Architectures

About Advanced Architectures

Advanced Architectures (A2) designs high-performance computer systems, subsystems and components. A2's strengths include the conceptual design and architecture of complete systems including custom DSPs and CPUs. Advanced Architectures is unique as a design house by being able to provide a leadership role in the development of complete systems from concept through manufacturing. A2's proficient interfacing with sales, marketing, manufacturing, and finance ensures corporate success. A2's design experience includes implementing structured design methodologies, performing HDL modeling simulation, performance analysis, logical design, packaging and detailed implementation.

Advanced Architectures has extensive knowledge and experience in the development and application of high performance computing systems. Designs have ranged from complete supercomputer designs to customized DSP solutions with microprocessor based control and custom compute engines for compression and decompression algorithms.

We are an ARC Certified Design Center and have completed several ARC based systems and are in the final stages of the development of SIMD extensions to the ARC that we have named A2MP. These extensions provide a framework for SIMD processing within the ARC environment, providing memory addressing and other infrastructure and yet still retains the flexibility of the ARC in that custom instructions and functions can be added to the A2MP to tailor the system to an application.

Disclaimer: Information furnished by Advanced Architectures is believed accurate and reliable. Advanced Architectures reserves the right to change specifications detailed in this Application Note at any time, without notice, in order to improve reliability, function or design and assumes no responsibility for any errors within this document. Advanced Architectures does not make any commitment to update this information. Advanced Architectures assumes no obligation to correct errors contained herein or to advise any user of this text of any correction, if such be made, nor does Advanced Architectures assume responsibility for the function of un-described features or parameters.

No license is granted by implication or otherwise under any patent or patent rights of Advanced Architectures or ARC Cores Ltd.

Copyright © 2001, Advanced Architectures.

Advanced Architectures contact info:

www.a-2.com

Advanced Architectures
19421 Sierra Lago Road
Irvine CA 92612-3812
+1 949 412 3486

info@a-2.com