Topic 10
Bus Architecture & Interconnects

Peter Cheung
Department of Electrical & Electronic Engineering
Imperial College London

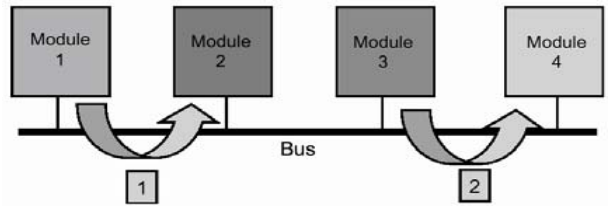
URL: www.ee.imperial.ac.uk/pcheung/
E-mail: p.cheung@imperial.ac.uk

Introductions & Sources

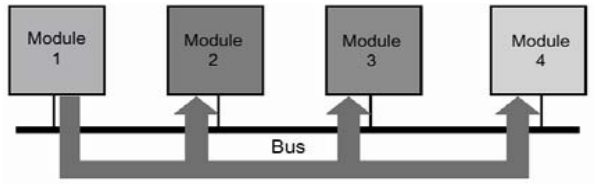
- We will consider a number of issues related to bus architectures in digital systems.
- Useful references:
  - "Bus Architecture of a System on a Chip with User-Configurable System Logic", Steven Winegarden, *IEEE JOURNAL OF SOLID-STATE CIRCUITS, VOL. 35, NO. 3, MARCH 2000, p425-433.*
  - *"AMBA: ENABLING REUSABLE ON-CHIP DESIGNS", David Flynn, IEEE Micro, July/August 1997.*
  - *AMBA™ Specification (Rev 2.0), ARM Ltd., 1999*
  - *The CoreConnect Bus Architecture, IBM,* http://www.chips.ibm.com/products/coreconnect
  - *VSI Alliance Architecture Document, version 1.0, 1997.*
  - Draft Chapter, *"System-on-Chip",* Flynn & Luk

Basic concepts: Bus basics: order and broadcast properties

- Communications on buses must be in strict order: serial nature of bus



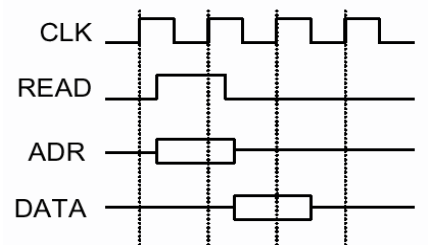
- It can broadcast a transaction – sending to multiple components simultaneously

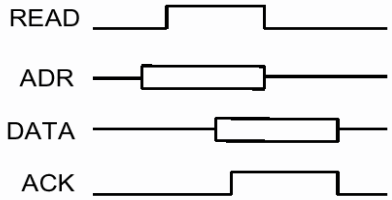


Basic concepts: Cycles, messages and transactions

- Buses operate in units of *cycle*s, *Messages* and *transaction*s.
  - *Cycle*s: A message requires a number of clock cycles to be sent from sender to receiver over the bus.
  - *Message*: These are logical unit of information. For example, a write message contains an address, control signals and the write data.
  - *Transaction*: A transaction consists of a sequence of messages which together form a transaction. For example, a memory read requires a memory read message and a reply with the requested data.

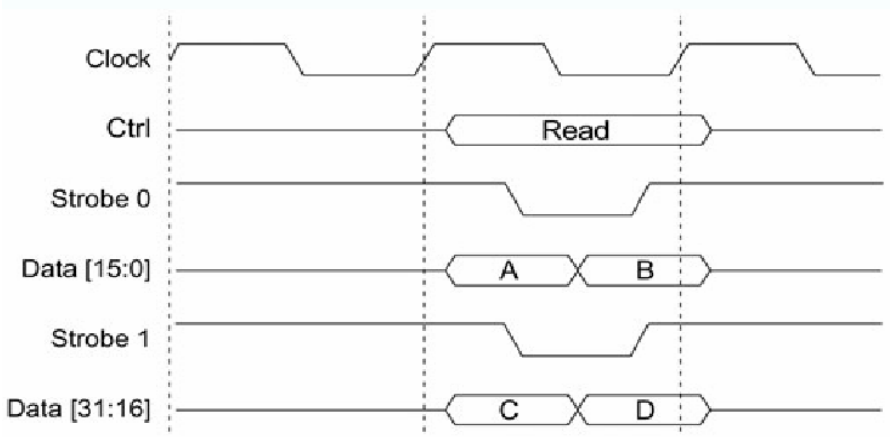
Synchronous vs Asynchronous





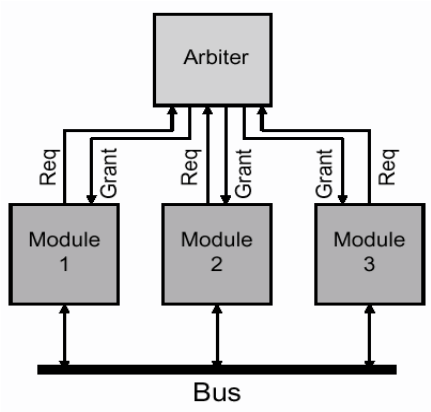
1. Master puts address on bus and asserts READ when address is stable
2. Memory puts data on bus and asserts ACK when data is stable
3. Master deasserts READ when data is read
4. Memory deasserts ACK

Basic concepts: Typical Source Synchronous Data Transfer



Basic concepts: Bus arbitration

- Only one bus master can control the us.
- Need some way of deciding who is master – may use a bus arbiter:



Basic concepts: Bus pipelining

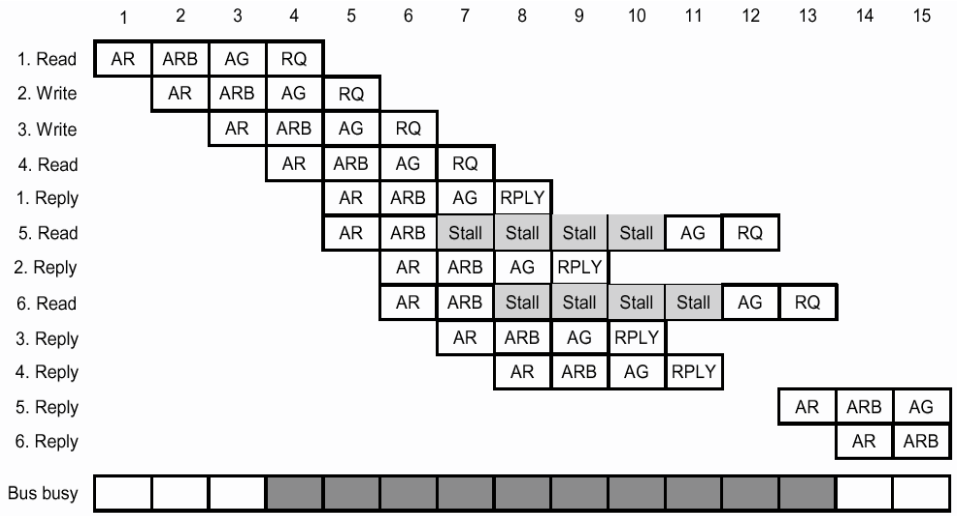
- A transaction may take multiple cycles
- Overlap multiple transaction through pipelining:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1. Read	AR	ARB	AG	RQ	P	RPLY									
2. Write		AR	ARB	AG	Stall	Stall	RQ	ACK							
3. Write			AR	ARB	Stall	Stall	AG	Stall	RQ	ACK					
4. Read				AR	Stall	Stall	ARB	Stall	AG	Stall	RQ	P	RPLY	RQ	
5. Read							AR	Stall	ARB	Stall	AG	RQ	P	RPLY	
6. Read									AR	Stall	ARB	AG	Stall	Stall	RQ
Bus busy															

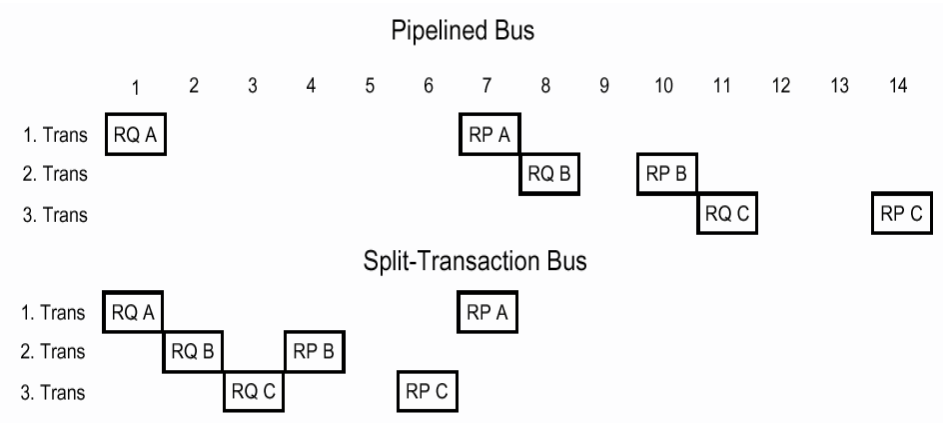
Basic concepts: Split-transaction bus

- A bus transaction can be divided into two or more phases, e.g.
  - "Request" phase
  - "Reply" phase
- These can be split into two separate sub-transactions, which may or may not happen consecutively. If split, these must compete for the bus by arbitration.

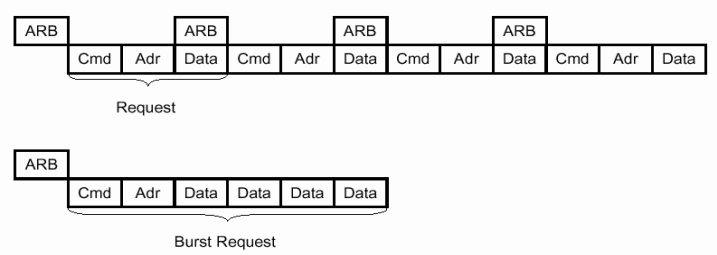
Basic concepts: Split-transaction bus

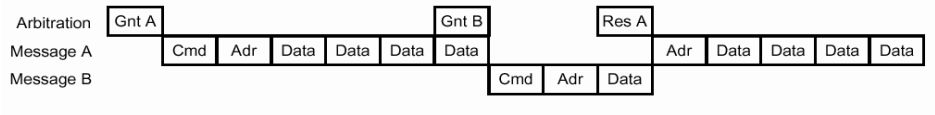


Basic concepts: Pipelined only bus vs split-transaction bus



Basic concepts: Burst transfer mode

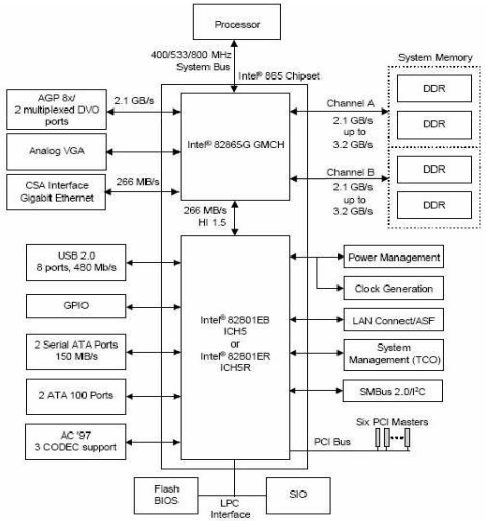




Bus bandwidth

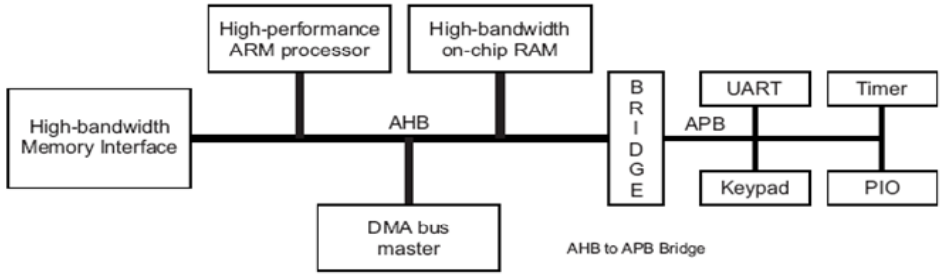
Bus	Width (bits)	Bus Speed (MHz)	Bus Bandwidth (MBytes/sec)
8-bit ISA	8	8.3	7.9
16-bit ISA	16	8.3	15.9
EISA	32	8.3	31.8
VLB	32	33	127.2
PCI	32	33	127.2
64-bit PCI 2.1	64	66	508.6
AGP	32	66	254.3
AGP (x2 mode)	32	66x2	508.6
AGP (x4 mode)	32	66x4	1,017.3

Bus hierarchy



AMBA bus

- Based around ARM processor
  - AHB – Advanced High-Performance Bus
    - ➤ Pipelining of Address / Data
    - ➤ Split Transactions
    - ➤ Multiple Masters
  - APB – Advanced Peripheral Bus
    - ➤ Low Power / Bandwidth Peripheral Bus

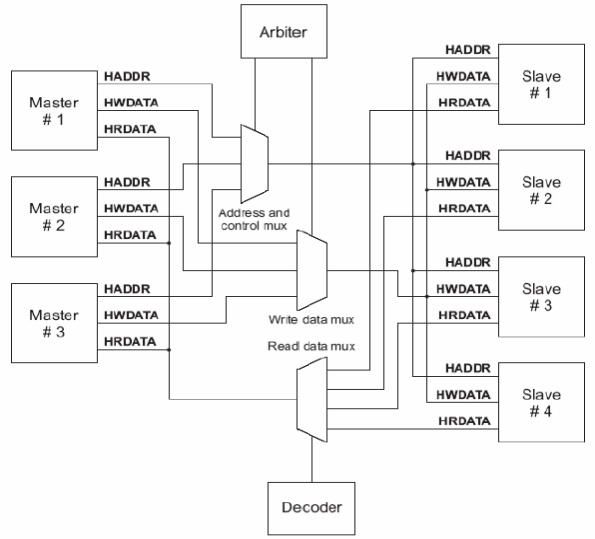


AMBA Bus Design Goals

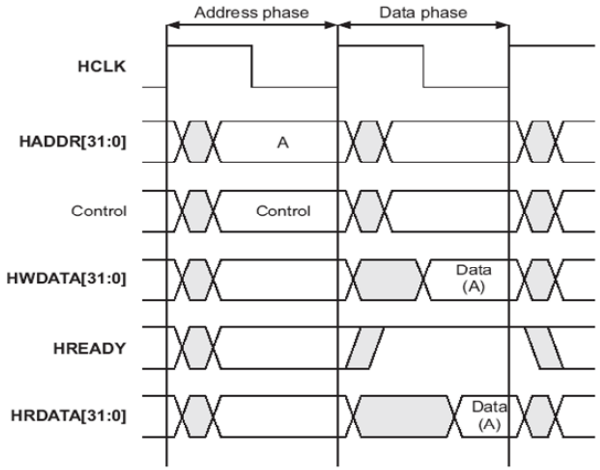
- Encourages modular design and design reuse
- Well defined interface protocol, clocking and reset
- Low-power support (helped by two-level partitioning)
- On-chip test access – built-in structure for testing modules connected on the bus

- Transactions on AHB
  - Bus master obtain access to the bus
  - Bus master initiates transfer
  - Bus slave provides response

AMBA bus arbitration

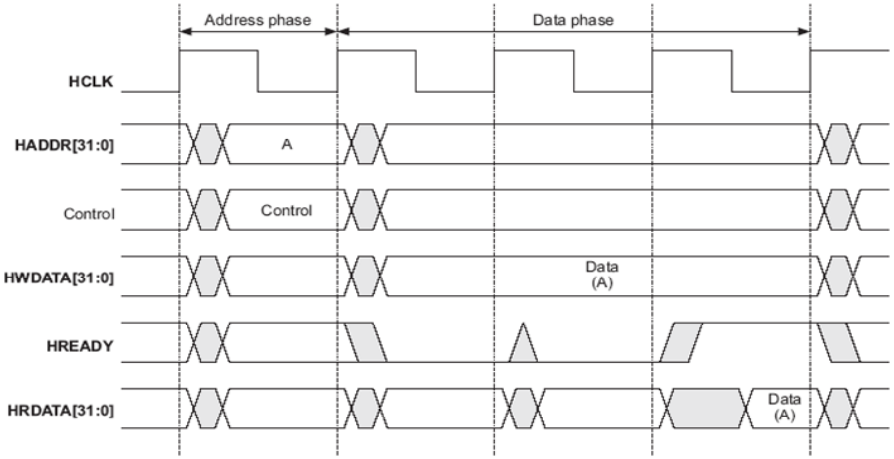


Simple AHB Transfer



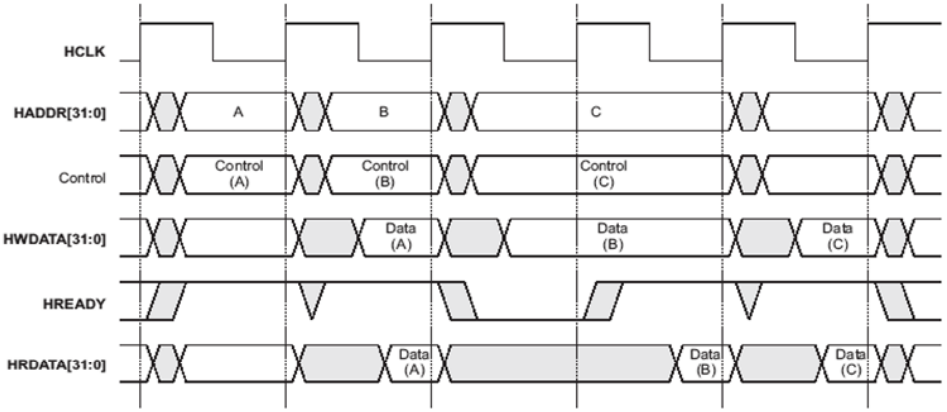
a) transfer without wait state

AHB Transfer with wait states

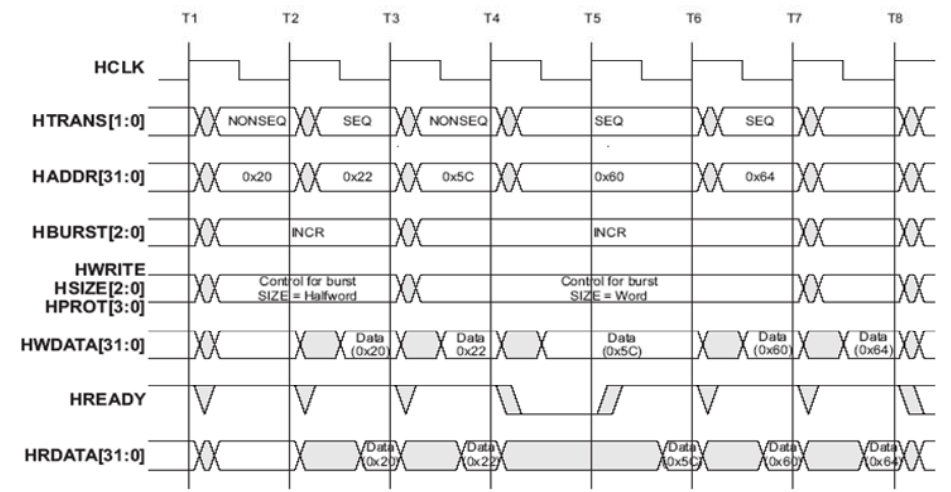


a) transfer with wait states

Multiple transfers with Pipelining



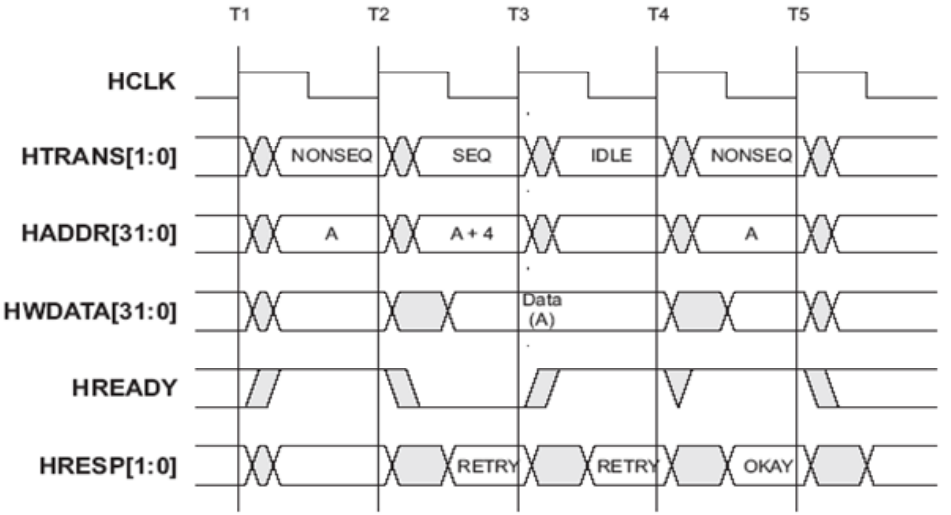
Burst mode transfer (undefined length)



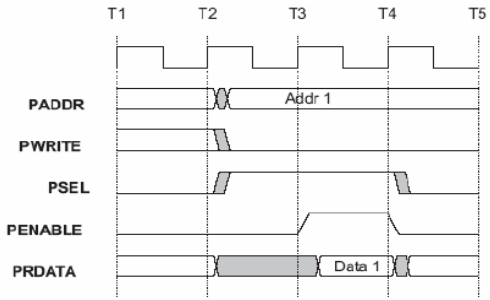
Slave Transfer Responses

- 1. Complete transfer immediately (single cycle transfer)
- 2. Insert one or more wait states to allow completion
- 3. Signal error to indicate transfer failed
- 4. Back of from the bus, try later (RE-TRY or SPLIT responses)

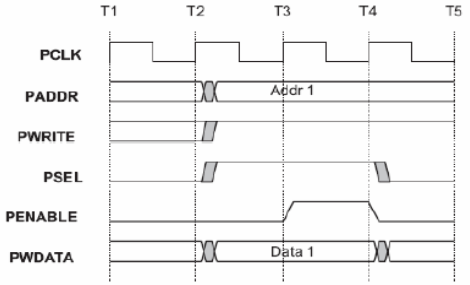
Retry Respnses on the AHB bus



Advanced Peripheral Bus (APB)

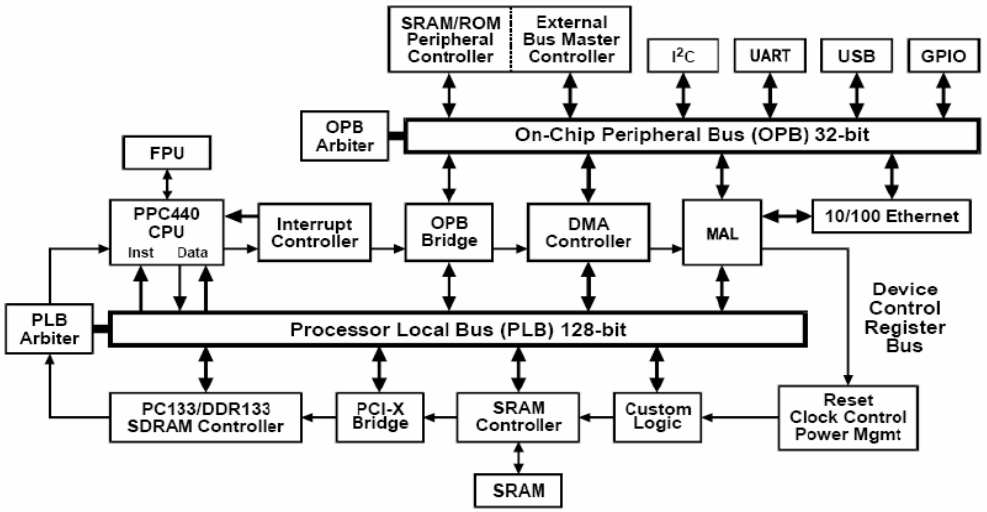


a) Read operation



b) Write operation

IBM CoreConnect Bus

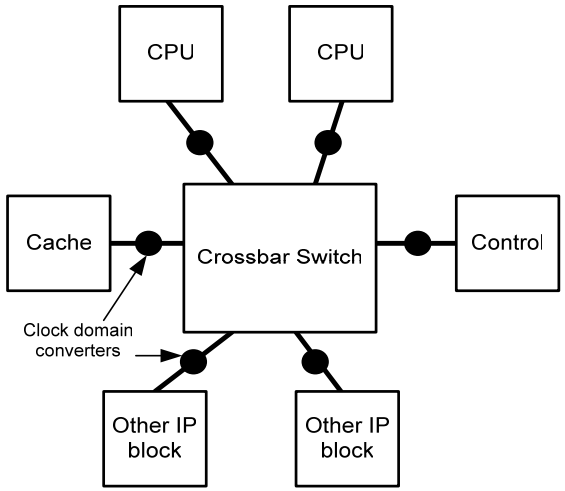


CoreConnect vs AMBA

	IBM CoreConnect Processor Local Bus	ARM AMBA 2.0 AMBA High-performance Bus
Bus Architecture	32-, 64-, and 128-bits Extendable to 256-bits	32-, 64-, and 128-bits
Data Buses	Separate Read and Write	Separate Read and Write
Key Capabilities	Multiple Bus Masters 4 Deep Read Pipelining 2 Deep Write Pipelining Split Transactions Burst Transfers Line Transfers	Multiple Bus Masters Pipelining Split Transactions Burst Transfers Line Transfers
	On-Chip Peripheral Bus	**AMBA Advanced Peripheral Bus**
Masters Supported	Supports Multiple Masters	Single Master: The APB Bridge
Bridge Function	Master on PLB or OPB	APB Master Only
Data Buses	Separate Read and Write	Separate or 3-state

Crossbar Switch Approach

- Uses asynchronous channels
- Different modules can run at different clock frequency
- Globally Asychronous, Locally Synchronous (GALS) system



Network-on-chip approach

- Array of tiles
- Each tile contains client logic and router logic
- 2-D mesh topology
- Uses data packets, not wires, for communication
- Predictable delay, and noise