ELEC2041
Microprocessors and Interfacing
Lectures 30: Memory and Bus Organisation - I
http://webct.edtec.unsw.edu.au/
May 2006
Saeid Nooshabadi
saeid@unsw.edu.au

Overview

° Memory Interfacing
  • Memory Type
  • Memory Decoding
  • D-RAM Access
  • Making DRAM Access fast

Review: Buses in a PC: Connect a few devices

CPU Memory bus
PCI: Internal (Backplane) I/O bus
Memory PCI Interface
SCSI: External I/O bus Ethernet Interface
External SCSI Interface
Ethernet Interface

° Data rates
  • Memory: 133 MHz, 8 bytes \( \Rightarrow 1064 \text{ MB/s (peak)} \)
  • PCI: 33 MHz, 8 bytes wide \( \Rightarrow 264 \text{ MB/s (peak)} \)
  • SCSI: “Ultra3” (80 MHz), “Wide” (2 bytes) \( \Rightarrow 160 \text{ MB/s (peak)} \)
  • Ethernet: 12.5 \text{ MB/s (peak)}

Review: Computers with Memory Mapped I/O

I/O devices Accessed like memory

device controller 1
device controller 2
device 1
device 2
Big Picture: A System on a Chip

Integration of Core Processor and many subsystem micro-cells
- ARM7TDMI core
- Cache RAM
- Embedded Co-processors
- External Memory Interface
- Low bandwidth I/O devices
- Timers
- I/O ports

ARM System Architecture

Need a Mechanism to access various memory units and I/O devices, uniquely, to avoid access conflicts

ARM System Architecture with Multiple Masters

Need a Mechanism to allow various Processing units to access the Memory Bus without causing conflict

ARM Core Interface Signals

Clock control
IRQ/FIQ

Memory Interface
**ARM Core Memory Interface Signals**

- **clock control**: mclk, wait, eclk
- **32 bit address**: A[31:0]
- **Separate Data in and out**: Din[31:0] & Dout[31:0]
- **Bidirectional Data bus**: D[31:0]
- **nmreq and seq for requesting memory access**
- **nr/w for read/write indication**
- **mas[1:0] for data size identification**: word 10, half-word 01 and byte 00.
- **All activities controlled by mclk**.

*Internal clock is mclk AND wait*

**Simple Memory Interface**

- **4 SRAMs**: write enabled separately
- **Read enabled together**
- **4 ROMs**: No write enable
  - Read enabled together

**Simple Memory Decoder Control**

- **Controls the Activation of RAM and ROM**
  - a[31]: 0 → ROM
  - a[31]: 1 → RAM
- **It controls the byte write enables during write**
  - mas[1:0]: 00 Byte, 01 H-word, 10 Word
- **It ensures that data is ready before processor continues.**

**SRAM/ROM Memory Timing**

- **Address should be stable during the falling edge**
- **SRAM is fast, ROM is slow**
  - ROM needs more time. Slows the system
  - **Solutions?**
    - Slow down the MCLK clock; loose performance
    - Use Wait states; more complex control
**ROM Wait Control State Transition**

- ROM access requires 4 clock cycles
- RAM access is fast

![Diagram of ROM Wait Control State Transition](image)

**Timing Diagram for for ROM Wait States**

- mclk
- A[31:0]
- wait
- ROM0e

![Timing Diagram for ROM Wait States](image)

**Improving Performance**

- Processor internal operations cycles do not need access to memory
  - Mem. Access is much slower than internal operations.
  - Use wait states for mem Accesses
- mreq = 1 internal operation
- mreq = 0 memory access

![Diagram of Improving Performance](image)

**DRAM Interface**

- Dynamic RAM Features:
  - much cheaper than SRAM
  - more capacity than SRAM
  - slower than SRAM
- Widely used in Computer Systems
DRAM Organisation

- Two dimensional matrix
- Bits are accessed by:
  - Accepting row and column addresses down the same multiplexed address bus
  - First Row address is presented and latched by ras signal
  - Next column address is presented and latched by cas signal

Making DRAM Access Fast

- Accessing data in the same row using cas-only access is 2 – 3 times faster
  - cas-only access does not activate the cell matrix
  - If next accesses is within the same row, a new column address may be presented just by applying a cas-only access.
- Fact: Most processor addresses are sequential (75%)
- If we had a way of knowing that the next address is sequential with respect with the current address (current address + 4), then we could only assert cas and make DRAM access fast
- Difficulty?
  - Detecting early in memory access cycle that the next address is in the same row.

ARM Solution to cas-only Access

- ARM address register Instruction:
  - 75% of next addresses are current address +4.
  - Sequential addresses flagged by seq signal
  - The external mem device checks previous address and row boundaries to issue cas only or ras-cas

Revised State Transition Diagram

- seq = 1: sequential address
- seq = 0: non-sequential
- mreq = 1 internal operation
- mreg = 0 memory access
Support for OS: Memory Protection

- Control unit can provide protection to certain areas in user mode:
  - ntran: Processor in USER (=1) or Privileged mode (=0)
  - nopc: memory access is for instruction (=1) or for data (0)
  - abort: caused pre-fetch abort exception

ARM Processor Bus Interface

- Arm Processor is optimised for high speed on-chip cache memory Interfacing
- It is a sub-system embedded in a larger system
- We need some interfacing rules and protocols to allow interfacing to other sub systems
  - Each sub-systems should follow these rules in order for the system to work properly.
- Options:
  - Making an ad hoc choice in every design
  - Use an established standard
- ARM provides Advanced Micro controller Bus Architecture (AMBA)
  - ARM processor uses AMBA to interface to the System Bus

AMBA Based System

- ASB: Advanced System Bus: To connect High Performance modules
- APB: Advanced Peripheral Bus: Simpler interface for low performance peripherals

ARM Core AMBA Interface

- ARM core cannot understand AMBA signaling standards directly.
  - It needs an interface unit for decoding and translation to AMBA signals
  - Some signals are just renamed
Reading Material


Conclusion

- Memory interfacing can degrade performance
- Can improve performance by increasing the clock frequency and allocating differing clock cycles for each memory access type
- Cas-only accesses in DRAM are 2 to 3 times faster than ras – cas accesses.
- Control unit can provide protection to certain areas in user mode
- ARM processor uses AMBA to interface to the System Bus