2010-07-15

計算機アーキテクチャ 第一 (E)

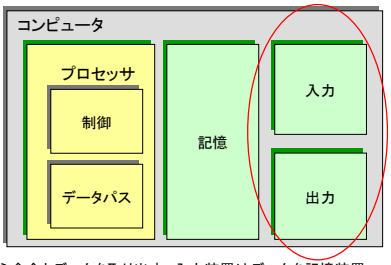
12. 入出力制御, 割り込み

吉瀬 謙二 計算工学専攻
kise_at_cs.titech.ac.jp
W641講義室 木曜日13:20 ー 14:50

Acknowledgement

- Lecture slides for Computer Organization and Design, Third Edition, courtesy of **Professor Mary Jane Irwin**, Penn State University
- Lecture slides for Computer Organization and Design, third edition, Chapters 1-9, courtesy of **Professor Tod Amon,** Southern Utah University.

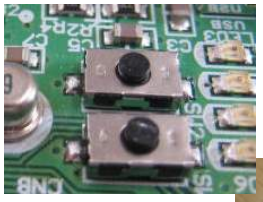
コンピュータ(ハードウェア)の古典的な要素



プロセッサは記憶装置から命令とデータを取り出す。入力装置はデータを記憶装置に書き込む。出力装置は記憶装置からデータを読みだす。制御装置は、データパス、記憶装置、入力装置、そして出力装置の動作を指定する信号を送る。

出典: パターソン & ヘネシー、コンピュータの構成と設計

Input and Output Devices（入出力装置）



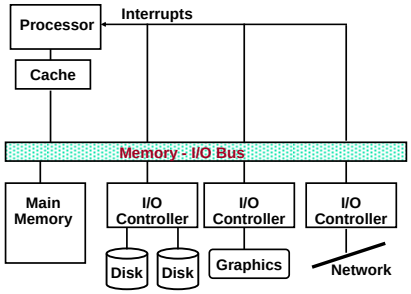
Input and Output Devices（入出力装置）

- I/O devices are **diverse** with respect to
  - **Behavior**（動作）– input, output or storage
  - **Partner**（相手）– human or machine
  - **Data rate**（転送速度）– the peak rate at which data can be transferred between the I/O device and the main memory or CPU

Device	Behavior	Partner	Data rate (Mb/s)
Keyboard	input	human	0.0001
Mouse	input	human	0.0038
Laser printer	output	human	3.2000
Graphics display	output	human	800.0000-8000.0000
Network/LAN	input or output	machine	100.0000-1000.0000
Magnetic disk	storage	machine	240.0000-2560.0000

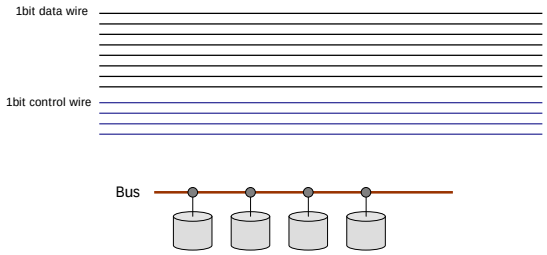
8 orders of magnitude range

A Typical I/O System（代表的な入出力装置）



Bus, I/O System Interconnect

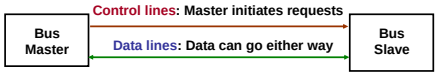
- A **bus**（バス）is a **shared** communication link (a single set of wires used to connect multiple subsystems)



Bus, I/O System Interconnect

- A **bus**（バス）is a shared communication link (a single set of wires used to connect multiple subsystems)
  - **Advantages**
    - **Low cost** – a single set of wires is shared in multiple ways
    - **Versatile**（**多目的**）– new devices can be added easily and can be moved between computer systems that use the same **bus standard**
  - **Disadvantages**
    - Creates a communication bottleneck – **bus bandwidth** limits the maximum **I/O throughput**
- The maximum bus speed is largely limited by
  - The **length** of the bus
  - The **number** of devices on the bus

Bus Characteristics

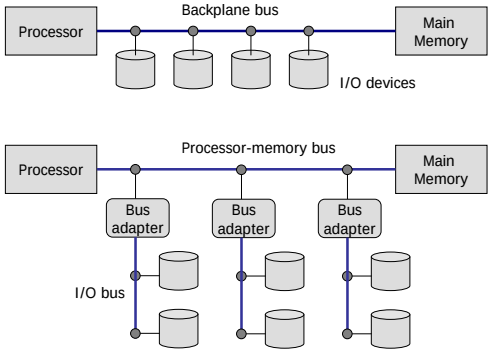


- **Control lines**
  - Signal requests and acknowledgments
  - Indicate what type of information is on the data lines
- **Data lines**
  - Data, addresses, and complex commands
- **Bus transaction** consists of
  - Master issuing the command (and address) – request
  - Slave receiving (or sending) the data – action
  - *Defined by what the transaction does to memory*
    - **Input** – inputs data from the I/O device to the memory
    - **Output** – outputs data from the memory to the I/O device

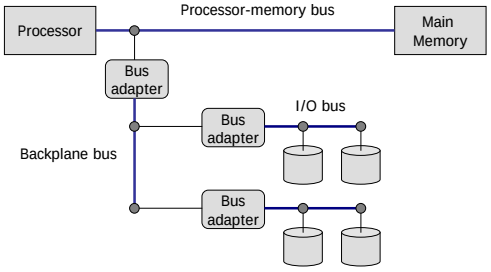
Types of Buses

- **Processor-memory bus**
  - Short and high speed
  - Matched to the memory system to maximize the memory-processor bandwidth
  - Optimized for cache block transfers
- **I/O bus** (industry standard, e.g., SCSI, USB, Firewire)
  - Usually is lengthy and slower
  - Needs to accommodate a wide range of I/O devices
  - Connects to the processor-memory bus or backplane bus
- **Backplane bus** (industry standard, e.g., ATA, PCIexpress)
  - The backplane is an interconnection structure within the chassis
  - Used as an intermediary bus connecting I/O busses to the processor-memory bus

Types of Buses



Types of Buses

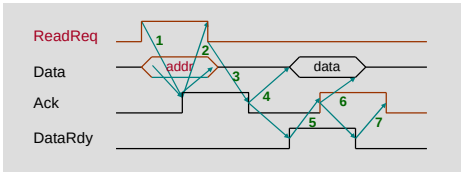


Synchronous（同期式）, Asynchronous（非同期式）Buses

- **Synchronous bus** (e.g., processor-memory buses)
  - Includes a clock in the control lines and has a fixed protocol for communication that is **relative** to the clock
  - **Advantage**: involves very little logic and can run very fast
  - **Disadvantages**:
    - Every device communicating on the bus must use same clock rate
    - To avoid **clock skew**, they cannot be long if they are fast
- **Asynchronous bus** (e.g., I/O buses)
  - It is not clocked, so requires a **handshaking protocol** and additional control lines (**ReadReq, Ack, DataRdy**)
  - **Advantages**:
    - Can accommodate a wide range of devices and device speeds
    - Can be lengthened without worrying about clock skew or synchronization problems
  - **Disadvantage**: slow

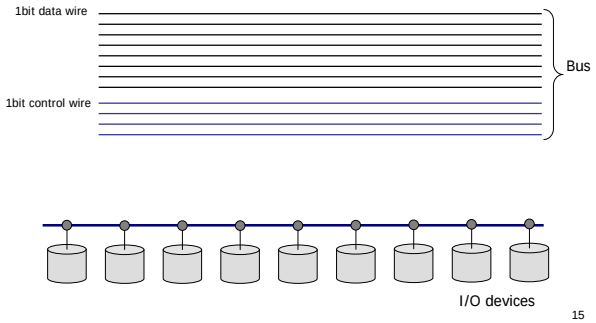
Asynchronous Bus **Handshaking Protocol**

An I/O device reads data from memory.



1. Memory sees **ReadReq**, reads **addr** from data lines, and raises **Ack**
2. I/O device sees **Ack** and releases the **ReadReq** and data lines
3. Memory sees **ReadReq** go low and drops Ack
4. When memory has data ready, it places it on data lines and raises **DataRdy**
5. I/O device sees **DataRdy**, reads the data from data lines, and raises **Ack**
6. Memory sees **Ack**, releases the data lines, and drops **DataRdy**
7. I/O device sees **DataRdy** go low and drops **Ack**

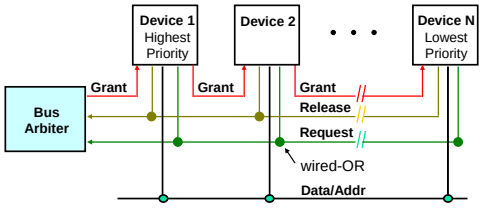
The Need for Bus **Arbitration**（調停）



The Need for Bus **Arbitration**（調停）

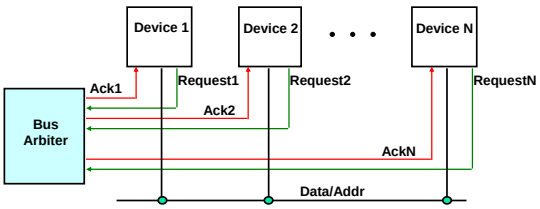
- Multiple devices may need to use the bus **at the same time**
- **Bus arbitration schemes** usually try to balance:
  - **Bus priority** – the highest priority device should be serviced first
  - **Fairness** – even the lowest priority device should never be completely locked out from the bus
- **Bus arbitration schemes** can be divided into four classes
  - **Daisy chain arbitration**
  - **Centralized, parallel arbitration**
  - Distributed arbitration by collision detection
    - device uses the bus when its not busy and if a collision happens (because some other device also decides to use the bus) then the device tries again later (Ethernet)
  - Distributed arbitration by self-selection

Daisy Chain Bus Arbitration（デイジーチェイン方式）



- **Advantage:** simple
- **Disadvantages**:
  - Cannot assure fairness – a low-priority device may be locked out
  - Slower – the daisy chain grant signal limits the bus speed

Centralized Parallel Arbitration（集中並列方式）



- **Advantages**: flexible, can assure fairness
- **Disadvantages**: more complicated arbiter hardware
- Used in essentially all processor-memory buses and in high-speed I/O buses

The Need for Bus **Arbitration**（調停）

- Multiple devices may need to use the bus **at the same time**
- **Bus arbitration schemes** usually try to balance:
  - Bus priority – the highest priority device should be serviced first
  - Fairness – even the lowest priority device should never be completely locked out from the bus
- **Bus arbitration schemes** can be divided into four classes
  - **Daisy chain arbitration**
  - **Centralized, parallel arbitration**
  - Distributed arbitration by collision detection（分散衝突検出方式）
    - device uses the bus when its not busy and if a collision happens (because some other device also decides to use the bus) then the device tries again later (**Ethernet**)
  - Distributed arbitration by self-selection（分散型自己判定方式）

19

Buses in Transition

- From synchronous, parallel, *wide* buses to asynchronous *narrow* buses
  - Reflection on wires and clock skew makes it difficult to use 16 to 64 parallel wires running at a high clock rate (e.g., ~400 MHz) so companies are transitioning to buses with a few one-way wires running at a very high "clock" rate (~2 GHz)

	PCI	PCIexpress	ATA	Serial ATA
Total # wires	120	36	80	7
# data wires	32 – 64 (2-way)	2 x 4 (1-way)	16 (2-way)	2 x 2 (1-way)
Clock (MHz)	33 – 133	635	50	150
Peak BW (MB/s)	128 – 1064	300	100	375 (3 Gbps)

20

ATA Cable Sizes

- **Serial ATA** cables (**red**) are much thinner than **parallel ATA** cables (**green**)

21

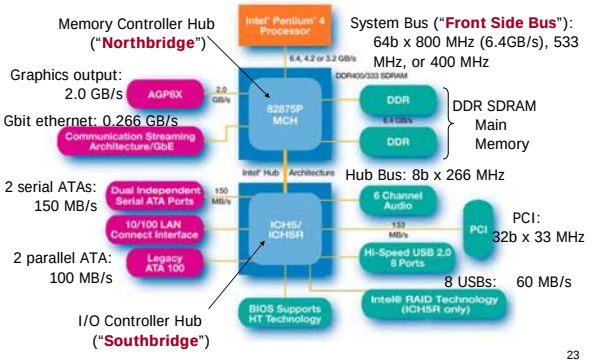
Bus Bandwidth Determinates

- The bandwidth of a bus is determined by
  - Whether its is synchronous or asynchronous and the timing characteristics of the protocol used
  - The data bus width
  - Whether the bus supports block transfers or only word transfers

	Firewire	USB 2.0
Type	I/O	I/O
Data lines	4	2
Clocking	Asynchronous	Synchronous
Max # devices	63	127
Max length	4.5 meters	5 meters
Peak bandwidth	50 MB/s (400 Mbps) 100 MB/s (800 Mbps)	0.2 MB/s (low) 1.5 MB/s (full) 60 MB/s (high)

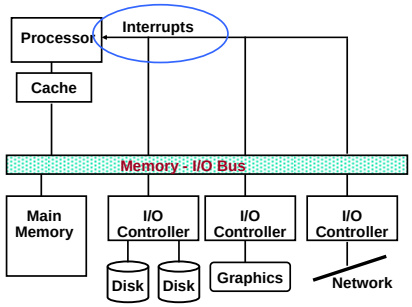
22

Example: The Pentium 4's Buses



23

I/O Systemの利用方法と割り込み



24

Communication of I/O Devices and Processor

- How the processor directs the I/O devices
  - **Memory-mapped I/O**
    - Portions of the high-order memory address space are assigned to each I/O device
    - Read and writes to those memory addresses are interpreted as commands to the I/O devices
    - Load/stores to the I/O address space can only be done by the OS
  - **Special I/O instructions**

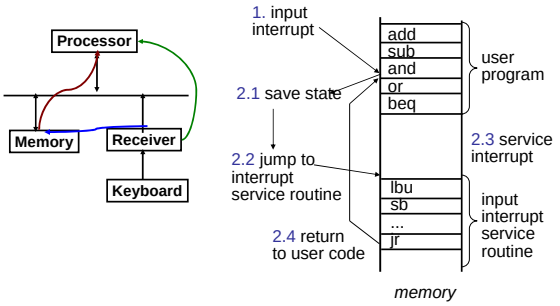
25

Communication of I/O Devices and Processor

- How the I/O device communicates with the processor
  - **Polling** – the processor periodically checks the status of an I/O device to determine its need for service
    - Processor is totally in control – but does **all** the work
    - Can waste a lot of processor time due to speed differences
  - **Interrupt-driven I/O – the I/O device issues an interrupts to the processor to indicate that it needs attention**

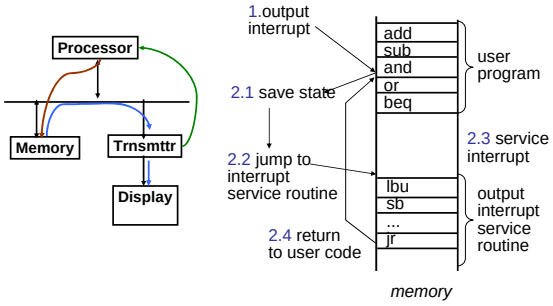
26

Interrupt-Driven Input



27

Interrupt-Driven Output



28

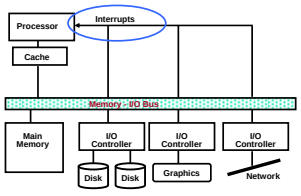
Interrupt-Driven I/O

- An I/O interrupt is **asynchronous**
  - Is not associated with any instruction so doesn't prevent any instruction from completing
    - You can pick your own convenient point to handle the interrupt
- With I/O interrupts
  - Need a way to identify the device generating the interrupt
  - Can have different urgencies (so may need to be prioritized)
- **Advantages** of using interrupts
  - No need to continuously poll for an I/O event; user program progress is only suspended during the actual transfer of I/O data to/from user memory space
- **Disadvantage** – special hardware is needed to
  - Cause an interrupt (I/O device) and detect an interrupt and save the necessary information to resume normal processing after servicing the interrupt (processor)

29

Direct Memory Access (DMA)

- For high-bandwidth devices (like disks) **interrupt-driven I/O** would consume a *lot* of processor cycles
- **DMA** – the I/O controller has the ability to transfer data **directly** to/from the memory without involving the processor
- There may be multiple DMA devices in one system



30

Direct Memory Access (DMA) how to?

1. The processor initiates the DMA transfer by supplying the I/O device address, the operation to be performed, the memory address destination/source, the number of bytes to transfer
2. The I/O DMA controller manages the entire transfer (possibly thousand of bytes in length), arbitrating for the bus
3. When the DMA transfer is complete, the I/O controller interrupts the processor to let it know that the transfer is complete

31

I/O and the **Operating System**

- The operating system acts as the interface between the I/O hardware and the program requesting I/O
  - To protect the **shared I/O resources**, the user program is not allowed to communicate directly with the I/O device
- Thus **OS** must be able to give commands to I/O devices, handle interrupts generated by I/O devices, provide fair access to the shared I/O resources, and schedule I/O requests to enhance system throughput
  - I/O interrupts result in a transfer of processor control to the **supervisor (OS) process**

32

Computer Architecture & Design

33

参考書

- **コンピュータの構成と設計 第3版**、パターソン＆ヘネシー（成田光彰　訳）、日経BP社、2006
- コンピュータアーキテクチャ 定量的アプローチ 第4版
  翔泳社, 2008
- コンピュータアーキテクチャ,
  村岡 洋一 著, 近代科学社, 1989
- 計算機システム工学,
  富田 真治, 村上 和彰 著, 昭晃堂, 1988
- コンピュータハードウエア,
  富田 真治, 中島 浩 著, 昭晃堂, 1995
- 計算機アーキテクチャ,
  橋本 昭洋 著, 昭晃堂, 1995

34

参考書

- **コンピュータの構成と設計 第3版**、パターソン＆ヘネシー（成田光彰　訳）、日経BP社、2006
- コンピュータアーキテクチャ 定量的アプローチ 第4版
  翔泳社, 2008
- コンピュータアーキテクチャ,
  村岡 洋一 著, 近代科学社, 1989
- 計算機システム工学,
  富田 真治, 村上 和彰 著, 昭晃堂, 1988
- コンピュータハードウエア,
  富田 真治, 中島 浩 著, 昭晃堂, 1995
- 計算機アーキテクチャ,
  橋本 昭洋 著, 昭晃堂, 1995

35

期末試験

- 期末試験
  - 2010年07月29日(木)　W641講義室, 5,6時限

36