[DOCUMENT TITLE]

[Document subtitle]

Computer Organization and Operating System

Name: MD: Aynul Islam

Chinese Name: 叶子

Student Id: 4420190030

[DATE]

[COMPANY NAME]
[Company address]

Contents

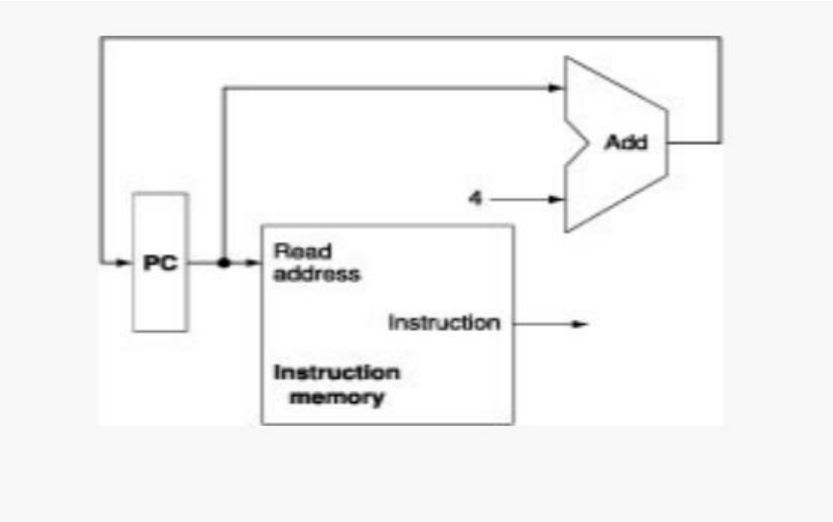
The Datapath Of Instructions

Introduction

A datapath is a collection of functional units such as arithmetic logic units or multipliers that perform data processing operations, registers, and buses.[1] Along with the control unit it composes the central processing unit (CPU). A larger datapath can be made by joining more than one datapaths using multiplexers. A datapath is a collection of functional units such as arithmetic logic units or multipliers that perform data processing operations, registers, and buses. Along with the control unit it composes the central processing unit (CPU). A larger datapath can be made by joining more than one datapaths using multiplexers.

Fetching Instructions

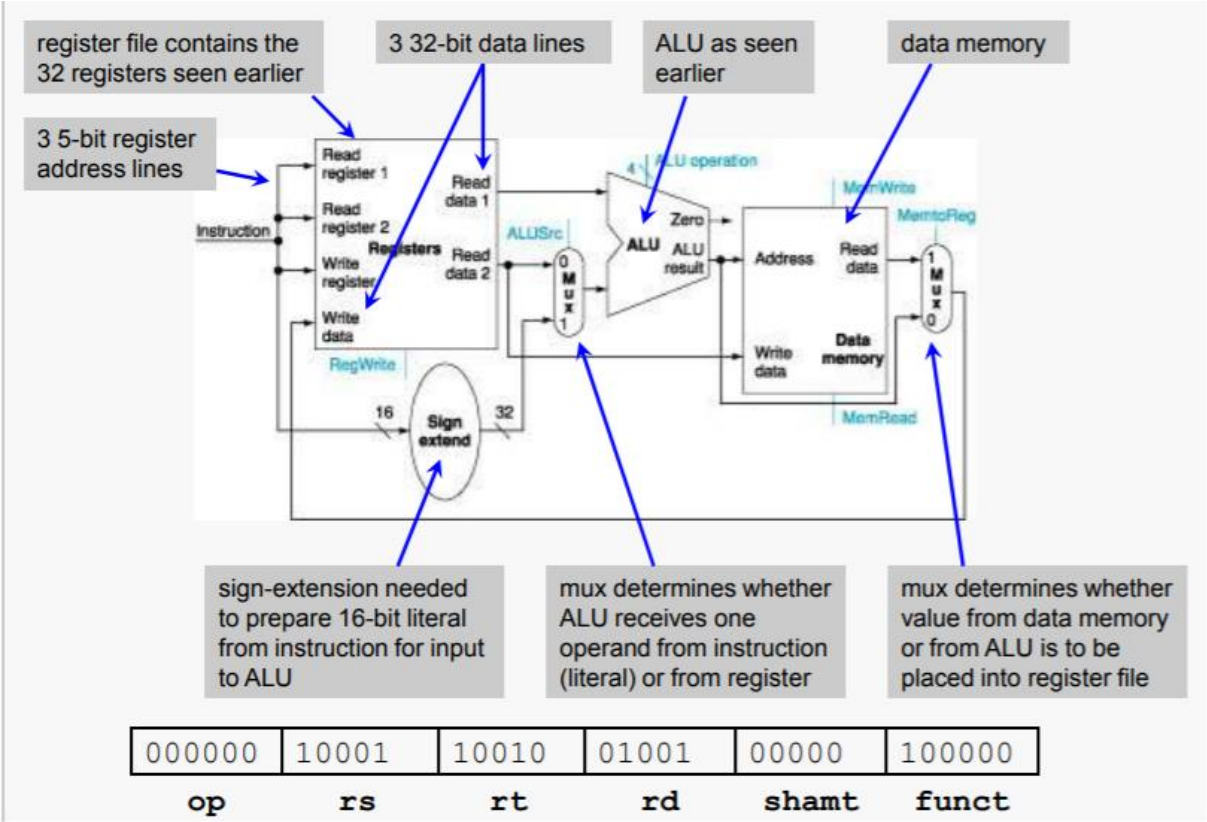
The basic steps are to send the address in the program counter (PC) to the instruction memory, obtain the specified instruction, and increment the value in the PC. For now, we assume sequential execution. Eventually the instruction memory will need write facilities (to load programs), but we ignore that for now. For now, the adder need only add the MIPS word size to the PC to prepare for loading the next instruction.



The fetched instruction will be used by other portions of the datapath.

Arithmetic and Memory-access Instructions

The arithmetic instructions define the set of operations performed by the processor Arithmetic Logic Unit (ALU). The arithmetic instructions are further classified into binary, decimal, logical, shift/rotate, and bit/byte manipulation instructions.



Datapath Operation with an R-type Instruction

1. The instruction is fetched, the opcode in bits 31:26 is examined, revealing this is an

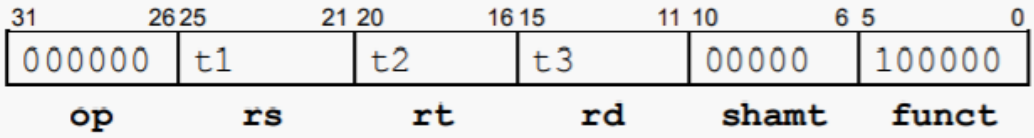
R-type instruction, and the PC is incremented accordingly

2. Data registers, specified by bits 25:21 and 20:16, are read from the register file and the main control unit sets its control lines

3. The ALU control determines the appropriate instruction from the funct field bits 5:0, and performs that operation on the data from the register file.

4. The result from the ALU is written into the register file at the destination specified by bits 15:11

Consider executing: `add $t1, $t2, $t3`



Datapath Operation with I-type Instruction

Now that we have a complete datapath for R-format instructions, let's add in support for I-format instructions. In our limited MIPS instruction set, these are lw, sw, and beq.

1. The instruction is fetched from memory, the opcode in bits 31:26 is examined revealing this is an load/store instruction, and the PC is incremented accordingly

2. Data register, specified by bits 25:21, is read from the register file

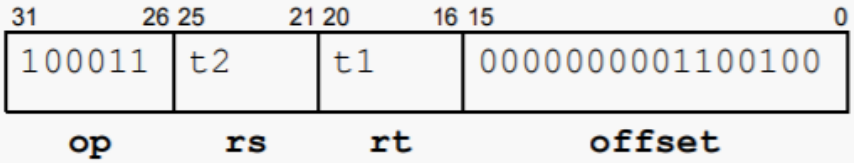
3. The ALU computes the sum of the retrieved register data and the sign-extended

immediate value in bits 15:0

4. The sum from the ALU is used as the address for the data memory

5. The data at the specified address is fetched from memory and written into the register file at the destination specified in bits 20:16 of the instruction.

Consider executing the instruction: `lw $t1, 100($t2)`



Datapath Operation with beq Instruction

The BEQ instruction branches the PC if the first source register's contents and the second source register's contents are equal.

1. The instruction is fetched, the opcode in bits 31:26 is examined, revealing this is a

Computer Science Dept Va Tech April 2006 Intro Computer Organization ©2006 McQuain WD

beq instruction, and the PC is incremented accordingly

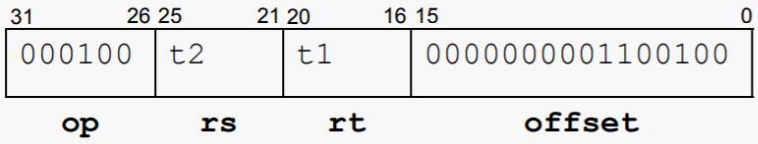
2. The data registers, specified by bits 25:21 and 20:16, are read from the register file

3. The ALU computes the difference of the two retrieved register data values; the value

of PC + 4 is added to the sign-extended value from bits 16:0, shifted left 2 bits

4. The Zero result from the ALU is used to decide which adder result to store in the PC

Consider executing the instruction: `beq $t1, $t2, offset`



Details for the Multi-cycle Datapath

The added elements are small in area compared to the ones that have been eliminated (2nd memory unit, adders), so this should be a cheaper design.

1. Multi-cycle implementation : break up instructions into separate steps
    a. Each step takes a single clock cycle

    b. Each functional unit can be used more than once in an instruction ,as long as it is used in different clock cycles

    c. Reduces amount of hardware needed

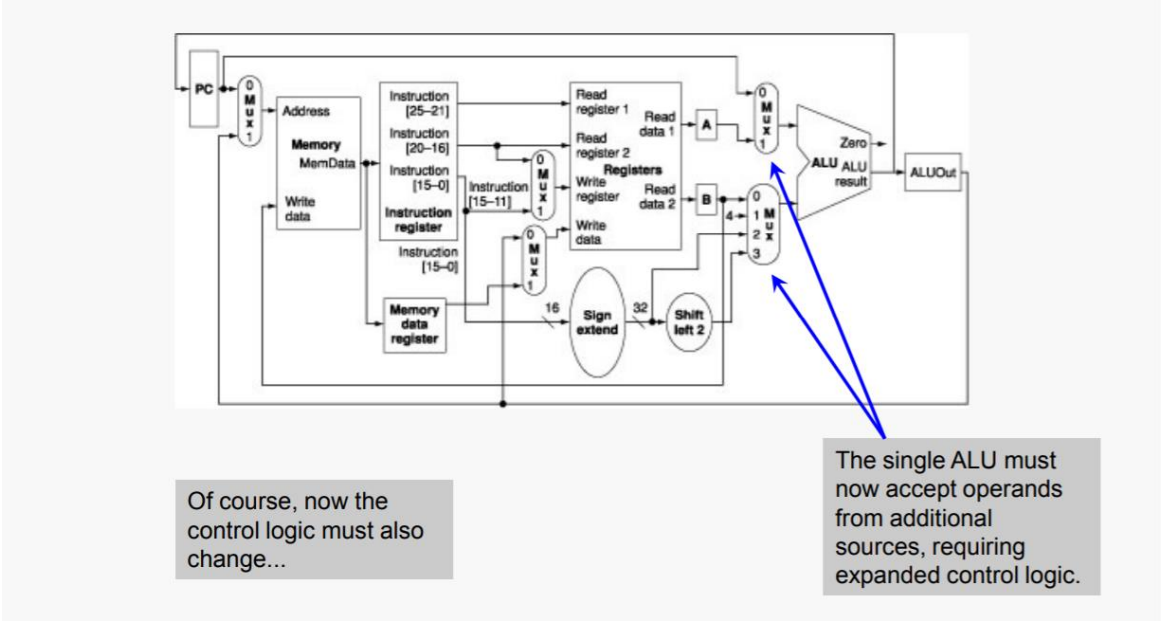
    d. Reduces average instruction time

2. Differences with single-cycle

    a. Single memory for instructions and data

    b. Single ALU (no separate adders for PC or branch calculation)

    c. Extra registers added after major functional units to hold results between clock cycles



The Hierarchical Structure Of The Memory

Introduction

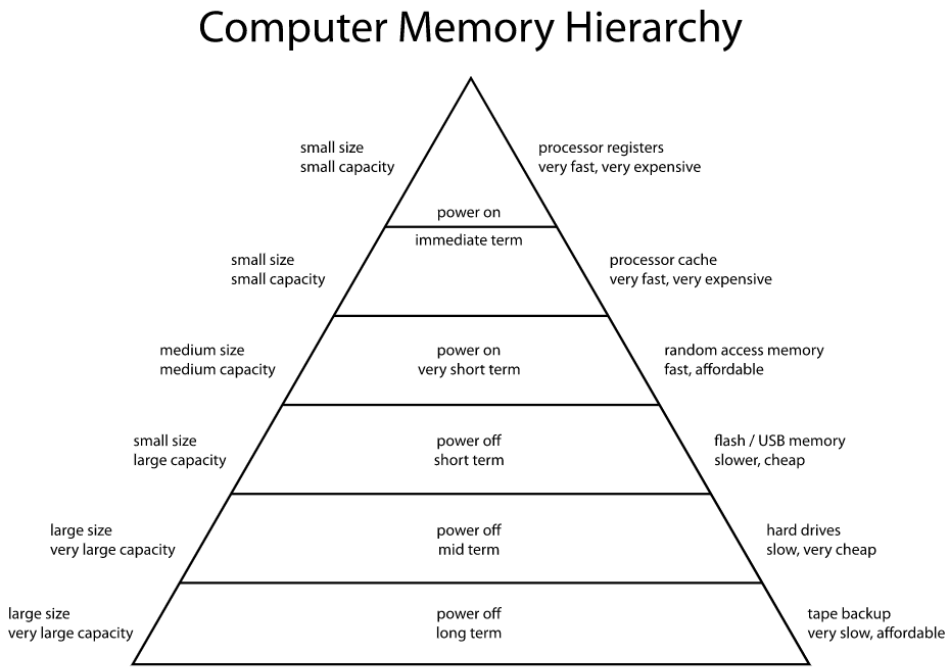
To this point in our study of systems, we have relied on a simple model of a computer system as a CPU that executes instructions and a memory system that holds instructions and data for the CPU. In our simple model, the memory system is a linear array of bytes, and the CPU can access each memory location in a constant amount of time. While this is an effective model as far as it goes, it does not reflect the way that modern systems really work.

In practice, a memory system is a hierarchy of storage devices with different capacities, costs, and access times. CPU registers hold the most frequently used data. Small, fast cache memories nearby the CPU act as staging areas for a subset of the data and instructions stored in the relatively slow main memory. The main memory stages data stored on large, slow disks, which in turn often serve as staging areas for data stored on the disks or tapes of other machines connected by networks.

Memory hierarchy

In computer architecture, the memory hierarchy separates computer storage into a hierarchy based on response time. Since response time, complexity, and capacity are related, the levels may also be distinguished by their performance and controlling technologies. Memory hierarchy affects performance in computer architectural design, algorithm predictions, and lower level programming constructs involving locality of reference.

Designing for high performance requires considering the restrictions of the memory hierarchy, i.e. the size and capabilities of each component. Each of the various components can be viewed as part of a hierarchy of memories (m1, m2, ..., mn) in which each member mi is typically smaller and faster than the next highest member mi+1 of the hierarchy. To limit waiting by higher levels, a lower level will respond by filling a buffer and then signaling for activating the transfer.



Storage Technologies

Much of the success of computer technology stems from the tremendous progress in storage technology. Early computers had a few kilobytes of random-access memory. The earliest IBM PCs didn't even have a hard disk. That changed with the introduction of the IBM PC-XT in 1982, with its 10-megabyte disk. By the year 2010, typical machines had 150,000 times as much disk storage, and the amount of storage was increasing by a factor of 2 every couple of years.

Random-Access Memory

Random-access memory (RAM) comes in two varieties—static and dynamic. Static RAM (SRAM) is faster and significantly more expensive than Dynamic RAM (DRAM). SRAM is used for cache memories, both on and off the CPU chip. DRAM is used for the main memory plus the frame buffer of a graphics system.

Typically, a desktop system will have no more than a few megabytes of SRAM, but hundreds or thousands of megabytes of DRAM.

Static RAM

SRAM stores each bit in a bistable memory cell. Each cell is implemented with a six-transistor circuit. This circuit has the property that it can stay indefinitely in either of two different voltage configurations, or states. Any other state will be unstable—starting from there, the circuit will quickly move toward one of the stable states. The pendulum is stable when it is tilted either all the way to the left or all the way to the right. In principle, the pendulum could also remain balanced in a vertical position indefinitely, but this state is metastable—the smallest disturbance would make it start to fall, and once it fell it would never return to the vertical position. Due to its bistable nature, an SRAM memory cell will retain its value indefinitely, as long as it is kept powered. Even when a disturbance, such

as electrical noise, perturbs the voltages, the circuit will return to the stable value when the disturbance is removed.

Registers

Usually, the register is a static RAM or SRAM in the processor of the computer which is used for holding the data word which is typically 64 or 128 bits. The program counter register is the most important as well as found in all the processors. Most of the processors use a status word register as well as an accumulator. A status word register is used for decision making, and the accumulator is used to store the data like mathematical operation. Usually, computers like complex instruction set computers have so many registers for accepting main memory, and RISC- reduced instruction set computers have more registers.

Cache Memory

Cache memory can also be found in the processor, however rarely it may be another IC (integrated circuit) which is separated into levels. The cache holds the chunk of data which are frequently used from main memory. When the processor has a single core then it will have two (or) more cache levels rarely. Present multi-core processors will be having three, 2-levels for each one core, and one level is shared.

Magnetic Disks

Magnetic DisksMagnThe magnetic disks in the computer are circular plates fabricated of plastic otherwise metal by magnetized material. Frequently, two faces of the disk are utilized as well as many disks may be stacked on one spindle

by read or write heads obtainable on every plane. All the disks in computer turn jointly at high speed. The tracks in the computer are nothing but bits which are stored within the magnetized plane in spots next to concentric circles. These are usually separated into sections which are named as sectors.etic Disks.

Magnetic Tape

This tape is a normal magnetic recording which is designed with a slender magnetizable covering on an extended, plastic film of the thin strip. This is mainly used to back up huge data. Whenever the computer requires to access a strip, first it will mount to access the data. Once the data is allowed, then it will be unmounted. The access time of memory will be slower within magnetic strip as well as it will take a few minutes for accessing a strip.

Advantages of Memory Hierarchy

The need for a memory hierarchy includes the following.

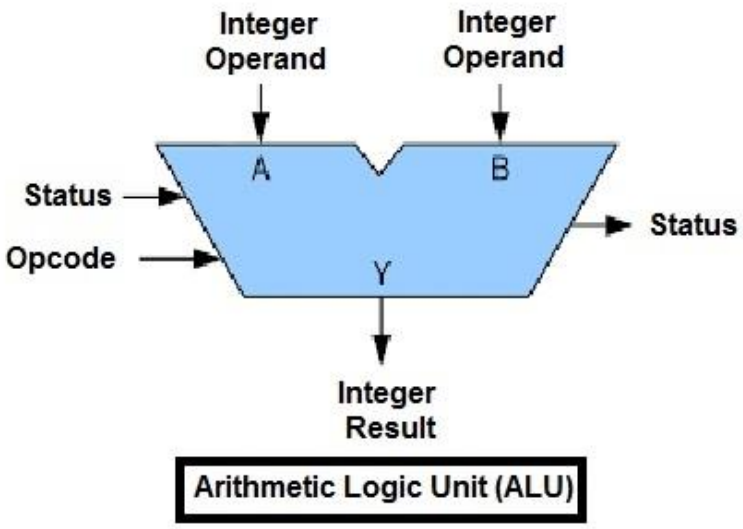
1. Memory distributing is simple and economical
2. Removes external destruction
3. Data can be spread all over
4. Permits demand paging & pre-paging
5. Swapping will be more proficient

Thus, this is all about memory hierarchy. From the above information, finally, we can conclude that it is mainly used to decrease the bit cost, access frequency, and to increase the capacity, access time. So it is up to the designer how much they need these characteristics for satisfying the necessities of their consumers.

The Operating Principle of ALU

Introduction

Inside a computer, there is an Arithmetic Logic Unit (ALU), which is capable of performing logical operations (e.g. AND, OR, Ex-OR, Invert etc.) in addition to the arithmetic operations (e.g. Addition, Subtraction etc.). The control unit supplies the data required by the ALU from memory, or from input devices, and directs the ALU to perform a specific operation based on the instruction fetched from the memory. ALU is the “calculator” portion of the computer.



An arithmetic logic unit(ALU) is a major component of the central processing unit of the a computer system. It does all processes related to arithmetic and logic operations that need to be done on instruction words. In some microprocessor architectures, the ALU is divided into the arithmetic unit (AU) and the logic unit (LU).

An ALU can be designed by engineers to calculate many different operations. When the operations become more and more complex, then the ALU will also become more and more expensive and also takes up more space in the CPU and

dissipates more heat. That is why engineers make the ALU powerful enough to ensure that the CPU is also powerful and fast, but not so complex as to become prohibitive in terms of cost and other disadvantages.

ALU is also known as an Integer Unit (IU). The arithmetic logic unit is that part of the CPU that handles all the calculations the CPU may need. Most of these operations are logical in nature. Depending on how the ALU is designed, it can make the CPU more powerful, but it also consumes more energy and creates more heat. Therefore, there must be a balance between how powerful and complex the ALU is and how expensive the whole unit becomes. This is why faster CPUs are more expensive, consume more power and dissipate more heat.

Operation Of ALU Categories

Different operation as carried out by ALU can be categorized as follows –

logical operations**:** These include operations like AND, OR, NOT, XOR, NOR, NAND, etc.

Bit-Shifting Operations : This pertains to shifting the positions of the bits by a certain number of places either towards the right or left, which is considered a multiplication or division operations.

Arithmetic operations − This refers to bit addition and subtraction. Although multiplication and division are sometimes used, these operations are more expensive to make. Multiplication and subtraction can also be done by repetitive additions and subtractions respectively.

Functions of the arithmetic logic unit (ALU)

An arithmetic logic unit (ALU) is a digital circuit used to perform arithmetic and logic operations. It represents the fundamental building block of the central

processing unit (CPU) of a computer. Modern CPUs contain very powerful and complex ALUs. In addition to ALUs, modern CPUs contain a control unit (CU).

Most of the operations of a CPU are performed by one or more ALUs, which load data from input registers. A register is a small amount of storage available as part of a CPU. The control unit tells the ALU what operation to perform on that data and the ALU stores the result in an output register. The control unit moves the data between these registers, the ALU, and memory.

Circuit operation

An ALU is a combinational logic circuit, meaning that its outputs will change asynchronously in response to input changes. In normal operation, stable signals are applied to all of the ALU inputs and, when enough time (known as the "propagation delay") has passed for the signals to propagate through the ALU circuitry, the result of the ALU operation appears at the ALU outputs. The external circuitry connected to the ALU is responsible for ensuring the stability of ALU input signals throughout the operation, and for allowing sufficient time for the signals to propagate through the ALU before sampling the ALU result.

For example, a CPU begins an ALU addition operation by routing operands from their sources (which are usually registers) to the ALU's operand inputs, while the control unit simultaneously applies a value to the ALU's opcode input, configuring it to perform addition. At the same time, the CPU also routes the ALU result output to a destination register that will receive the sum. The ALU's input signals, which are held stable until the next clock, are allowed to propagate through the ALU and to the destination register while the CPU waits for the next clock. When the next clock arrives, the destination register stores the ALU result and, since the ALU operation has completed, the ALU inputs may be set up for the next ALU operation.

Functions

A number of basic arithmetic and bitwise logic functions are commonly supported by ALUs. Basic, general purpose ALUs typically include these operations in their repertoires

Arithmetic operations

a. Add: A and B are summed and the sum appears at Y and carry-out.
b. Add with carry: A, B and carry-in are summed and the sum appears at Y and carry-out.
c. Subtract: B is subtracted from A (or vice versa) and the difference appears at Y and carry-out. For this function, carry-out is effectively a "borrow" indicator. This operation may also be used to compare the magnitudes of A and B; in such cases the Y output may be ignored by the processor, which is only interested in the status bits (particularly zero and negative) that result from the operation.
d. Subtract with borrow: B is subtracted from A (or vice versa) with borrow (carry-in) and the difference appears at Y and carry-out (borrow out).
e. Two's complement (negate): A (or B) is subtracted from zero and the difference appears at Y.
f. Increment: A (or B) is increased by one and the resulting value appears at Y.
g. Decrement: A (or B) is decreased by one and the resulting value appears at Y.
h. Pass through: all bits of A (or B) appear unmodified at Y. This operation is typically used to determine the parity of the operand or whether it is zero or negative, or to load the operand into a processor register.

Bitwise logical operations

ALU shift operations cause operand A (or B) to shift left or right (depending on the opcode) and the shifted operand appears at Y. Simple ALUs typically can shift the

operand by only one bit position, whereas more complex ALUs employ barrel shifters that allow them to shift the operand by an arbitrary number of bits in one operation. In all single-bit shift operations, the bit shifted out of the operand appears on carry-out; the value of the bit shifted into the operand depends on the type of shift.

1. Arithmetic shift: the operand is treated as a two's complement integer, meaning that the most significant bit is a "sign" bit and is preserved.

2. Logical shift: a logic zero is shifted into the operand. This is used to shift unsigned integers.

3. Rotate: the operand is treated as a circular buffer of bits so its least and most significant bits are effectively adjacent.

4. Rotate through carry: the carry bit and operand are collectively treated as a circular buffer of bits.

I/O device

Introduction

In computing, input/output (I/O, or informally io or IO) is the communication between an information processing system, such as a computer, and the outside world, possibly a human or another information processing system. Inputs are the signals or data received by the system and outputs are the signals or data sent from it. The term can also be used as part of an action; to "perform I/O" is to perform an input or output operation.

I/O devices are the pieces of hardware used by a human (or other system) to communicate with a computer. For instance, a keyboard or computer mouse is an input device for a computer, while monitors and printers are output devices. Devices for communication between computers, such as modems and network cards, typically perform both input and output operations.

Input/output device

Alternatively referred to as an IO device, an input/output device is any hardware used by a human operator or other systems to communicate with a computer. As the name suggests, input/output devices are capable of sending data (output) to a computer and receiving data from a computer (input).

Examples of input/output devices

Input and output devices allow the computer system to interact with the outside world by moving data into and out of the system. An input device is used to bring data into the system. Some input devices are:

1. Keyboard
2. Mouse
3. Microphone
4. Bar code reader
5. Graphics tablet

An output device is used to send data out of the system. Some output devices are:

1. Monitor
2. Printer
3. Speaker

Input/output devices are usually called I/O devices. They are directly connected to an electronic module inside the systems unit called a device controller. For example, the speakers of a multimedia computer system are directly connected to a device controller called an audio card (such as a Soundblaster), which in turn is connected to the rest of the system.

Sometimes secondary memory devices like the hard disk are called I/O devices (because they move data in and out of main memory.) What counts as an I/O device depends on context. To a user, an I/O device is something outside of the system box. To a programmer, everything outside of the processor and main memory looks like an I/O devices. To an engineer working on the design of a processor, everything outside of the processor is an I/O device.

How Five Parts Are Connected Together And Worked As A Whole

Introduction

Von Neumann computer systems contain three main building blocks:

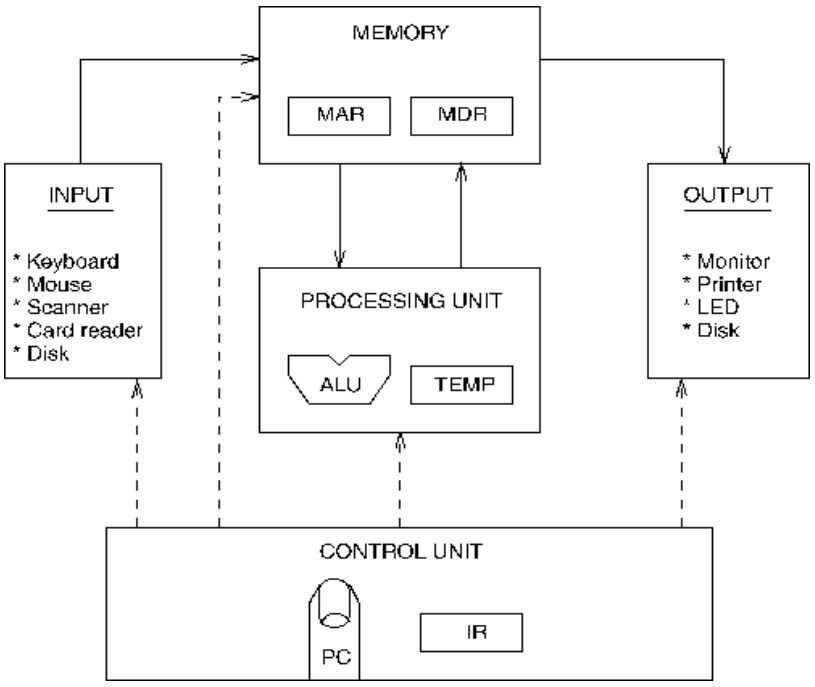
the central processing unit (CPU),memory,and input/output devices (I/O).

These three components are connected together using the system bus.

The most prominent items within the CPU are the registers: they can be manipulated directly by a computer program.

Components of the Von Neumann Model

1. Memory: Storage of information (data/program)
2. Processing Unit: Computation/Processing of Information
3. Input: Means of getting information into the computer. e.g. keyboard, mouse
4. Output: Means of getting information out of the computer. e.g. printer, monitor
5. Control Unit: Makes sure that all the other parts perform their tasks correctly and at the correct time.



Communication Between Memory and Processing Unit

Communication between memory and processing unit consists of two registers:

1. Memory Address Register (MAR).
2. Memory Data Register (MDR).

To Read,

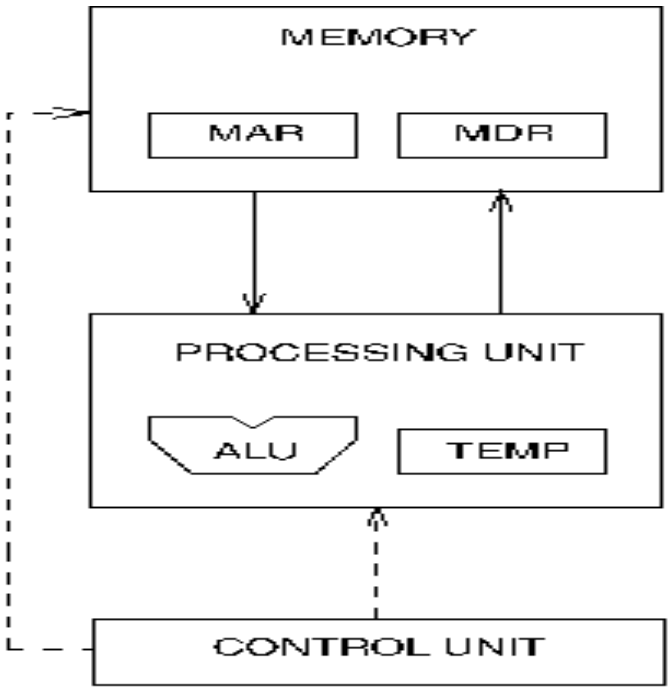
The address of the location is put in MAR.

1. The memory is enabled for a read.
2. The value is put in MDR by the memory

To Write,

The address of the location is put in MAR.

1. The data is put in MDR.
2. The Write Enable signal is asserted.
3. The value in MDR is written to the location specified.



CPU data-path

- Hardware units like ALU's, registers, memory, etc., are linked together into a data-path.
- The flow of bits around the data-path is controlled by the "gates" which allow the bits to flow (on) or not flow (off) through the data-path.
- The binary instructions (1 = on; 0 = off) that control the flow are called micro-instructions.

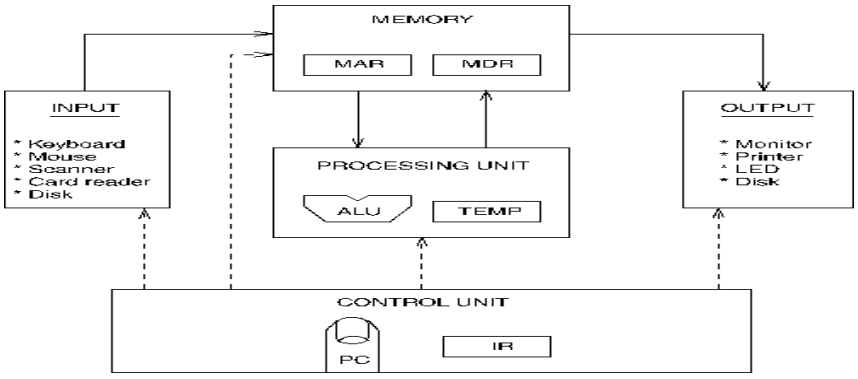
ALU, the Processing Unit

- Processing unit is hardware that implements Arithmetic and Logical Operations.
- ALU stands for Arithmetic and Logic Unit, capable of performing
  ADD, SUBTRACT, AND, OR, and NOT operations.
- The size of input quantities of ALU is often referred to as word length of the computer.
- Many processors today have word length of 32 and 64 bit.

- ❖ Processing unit also includes a set of registers for temporary storage of data and memory addressing.

Control Unit

- ✓ Manages the Precessing Unit.
- ✓ Implemented as FSM.
- ✓ FSM directs all activity.
- ✓ Clock-based step-by-step precessing, cycle-by-cycle.
- ✓ FSM is controlled by the
  - ➢ Clock signal
  - ➢ Instruction Register
  - ➢ Reset signal



Input/Output

- ➢ I/O controller provides the necessary interface to I/O devices.
- ➢ Takes care of low-level, device-dependent details.
- ➢ Provides necessary electrical signal interface.

conclusion

During execution of a program, an interrupt or an exception may cause the processor to temporarily , suspend the program in order to service the needs of a particular event. The event may be external to the processor or may be internal. Interrupts are used for handling asynchronous external events, such as when an external device wants to perform an I/O operation. Exceptions are used for handling synchronous internal events that occur due to instruction faults, such as divide by zero.

Interrupts are caused by asynchronous signals applied to certain input lines, called interrupt request lines, of a processor. When an external event triggers an interrupt during execution of a program, the processor suspends the execution of the program, determines the source of the interrupt, acknowledges the interrupt, saves the state of the program (such as program counter and registers) in a stack, loads the address of the proper interrupt service routine into the program counter, and then processes the interrupt by executing the interrupt service routine. After finishing the processing of the interrupt, it resumes the interrupted program. Often, interrupts are classified into different levels of priorities. For example, disk drives are given higher priorities then keyboards. Thus, if an interrupt is being processed and a higher-priority interrupt arrives, the process of the former interrupt will be suspended and the latter interrupt will be serviced. In general, the priority levels are divided into two groups: maskable and nonmaskable. A maskable interrupt can be handled or delayed by the processor. However, nonmaskable interrupts have very high priority. Often, when a nonmaskable interrupt routine is started, it cannot be interrupted by any other interrupt. They are usually intended for catastrophic events, such as memory error detection or power failure. A typical use of a nonmaskable interrupt would be to start a power failure routine when the power goes down.

Reference:

N.B: I followed those website while i was writing the course paper.

1. The Datapath Of Instructions:
    1. Link 1.
    2. Link 2.
    3. Link 3.
2. The Hierarchical Structure Of The Memory:
    1. Link 1.
    2. Link 2.
    3. Link 3.
3. The Operating Principle of ALU
    1. Link 1.
    2. Link 2.
    3. Link 3.
4. I/O device
    1. Link 1.
    2. Link 2.
    3. Link 3.
5. How Five Parts Are Connected Together And Worked As A Whole
    1. Link 1.
    2. Link 2.
    3. Link 3.