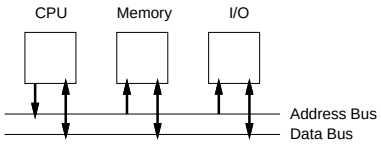
Microcomputer Architecture

This lecture gives an overview of the architecture of a simple microcomputer. It describes the operation of the CPU address and data buses during read and write cycles.
After this lecture you should be able to: (1) show how the following buses and signals are connected in a microcomputer system: power and ground, address and data buses, read and write strobes, and chip enables; (2) give the sequence of signals that must appear on the address, data, and control lines of a memory or I/O chip in order to read or write a particular data value to/from a particular address; (3) explain the purpose of these lines; and (4) compute the number of address lines required for a given memory size or vice-versa.

Microcomputer Components

A microcomputer is usually assembled from a microprocessor chip (CPU[1]) connected to memory chips and I/O (input/output) chips.



The different components are interconnected with *signal*s. A signal is a voltage or current that transfers information within a circuit.

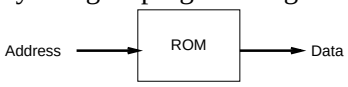
Memories

Memory chips store programs and data in microprocessor systems. The two main types of memory chips are ROMs and RAMs.

Read-Only Memory (ROM)

ROMs are used to store information that does not change. In an embedded computer control system this might include most of the software.

A ROM implements a lookup table. The input to the ROM is called the "address" and the output is called the "data." The information stored in the ROM is the data which is output for each possible address input. This data can be set when the ROM is manufactured or by using a "programming" device.



Both the address input and the data output of a ROM consists of multi-bit signals. The address is usually interpreted as a binary number.

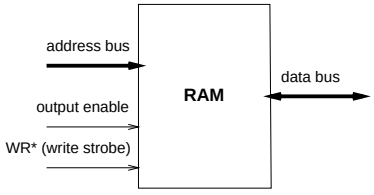
Exercise 1: How wide is the address bus on a 64 kByte (2^{16} byte) byte-wide (64k*times*8) ROM? How wide is the data bus?

There are many types of ROM chips. The most common are mask-programmed ROMs which come pre-programmed from the factory. There are also EPROMs (erasable programmable read-only memory) which can be erased by exposing them to strong ultra-violet light for several minutes and then re-written using a device programmer. EEPROMs (electrically erasable programmable read-only memory) are similar to EPROMs, but individual addresses can be erased (by writing to "magic" addresses) and then re-written. "Flash" EEPROMs are similar, but large parts of the device must be erased before they can be re-written.

Exercise 2: Could we implement the circuit described in Exercise 31 using a ROM? If so, state the size of ROM required (addresses$\times$bits) and draw a diagram showing how the signals a, b, c, x, and y would be connected to the ROM.

Read-Only Memory (RAM)

A RAM (random-access memory) is similar to a ROM except that the values stored in the RAM can be changed. The diagram below shows a typical RAM chip:



The address input selects the address that is to be read or written.

Unlike the ROM data bus, the RAM data bus is "bi-directional:" it can be an output (when the RAM

[1] Central Processing Unit

is being read) or an input (when the RAM is being written).

There are also two control signals: "output enable" and "write strobe." When the output enable signal is asserted (true), data bus is an output and the RAM acts like a ROM. A rising edge on the write strobe (similar to a flip-flop's clock input) causes the value on the data bus to be stored in the addressed memory location.

There are two main types of RAM chips. *Static* RAM (SRAM) chip storage their state in flip-flops while *dynamic* (DRAM) chips store their state as the charge in a capacitor. This makes DRAM chips simpler and cheaper than SRAM chips but DRAM memory contents need to re-written (refreshed) every few milliseconds since the change on the very small capacitors decays with time.

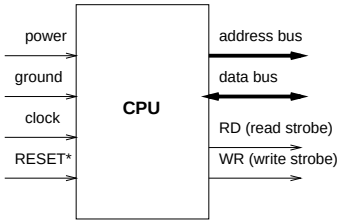
Note that the contents of a RAM are lost when power is removed. This is not the case for a ROM.

Exercise 3: Is a RAM chip a combinational circuit or a sequential circuit? How about a ROM chip?

RAMs are used to store information that needs to be changed. This includes variables and programs when they need to be loaded from external storage such as disks.

CPU

The following diagram shows the input and output signals on a typical CPU:



As with any other digital logic chip, the CPU chip needs pins to supply power and ground. The CPU also needs a clock signal (a signal that periodically switches from high to low). This clock input is used by the processor to synchronize its internal operations. The processor is a (very complicated) state machine and the clock is used to sequence between the processor's states. The clock speed typically ranges from 32768 (2^{15}) Hz (for slow, typically battery operated applications) to several hundred MHz (for fast, typically computationally-intensive applications).

A *bus* is a group of related signals. The CPU has two buses: the address bus and the data bus. These two buses are used together by the CPU to: (a) transfer the computer instructions ("opcodes") making up the program from memory chips to the CPU and (b) transfer data to/from the CPU to/from memory or I/O chips. The thick lines in the diagram above indicates the buses carry multiple signals and the arrows indicate whether the bus/signal is an input, output or, as in the case of the data bus, if it can alternate between the two directions.

An address bus of N bits can be used to select (address) one of 2^N bytes in the microcomputer's memory. Typical address bus sizes ("widths") are 16 bits (most 8-bit microprocessors), 20 bits (the 8088 CPU used in the original IBM PC), 24 bits (the 68000 CPU used in the lab computer) and 32 bits (the chips used in modern microprocessors). The individual signals in the bus are usually given the labels $A_0, A_1, A_2, \ldots, A_{N-1}$.

Exercise 4: Approximately how many bytes can each of the above processors address? *Hint: Use the approximation that* 2^{10} *is defined as "1k" and* 2^{20} *is "1M."*

The data bus is used to transfer data between memory or I/O peripherals and the CPU. The data bus width in modern computers is always an power of 2 times 8 bits (8, 16, 32, 64). For simplicity we will only consider CPUs with an 8-bit (1-byte) data bus. The individual signals in the data bus are usually given the labels $D_0, D_1, \ldots, D_7$.

The address and data buses connect the CPU to the memory and I/O chips. Multiple memory devices can be connected in parallel to these buses but additional logic circuitry ("address decoders") are required to ensure that only one device has its output enable or write strobe asserted at any given time.

Finally, there are a number of control signals. We will only consider three of these: the read and write strobe outputs (RD and WR), and the RESET input.

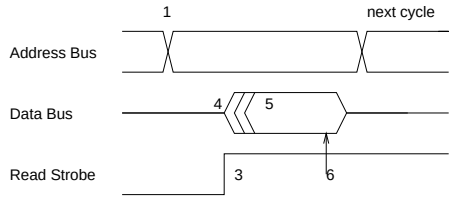
The RD and WR outputs are used to control memory devices. If the RD output is high, the CPU is reading from memory (a "read" cycle); if the WR output is high the CPU is writing to memory (a "write" cycle). These signals are often active-low (e.g. RD*) and/or are combined into two other signals (R/W* and a data transfer strobe, DS*, signal).

An active-low signal is “true” when it is at the low-voltage level.

The RESET input pin resets the processor to a known initial state. This is usually done when power is first applied or if the processor gets stuck while executing a buggy program. When the RESET pin is brought high the processor stops executing the current instruction sequence and restarts execution at an address that contains a program to restart the computer.

Read and Write Cycles

The following diagram shows the signals on a ROM or RAM during a read cycle.



The following operations take place during a read operation (“cycle”):

- the CPU puts the address of the desired memory location on the address bus (1)
- the CPU turns its data bus into an input (2)
- the CPU asserts (brings high) the RD signal line (3)
- the RAM detects the high signal on the RD line and turns its data bus into an output (4)
- the RAM looks up the value current value stored for that memory location and, after a short delay (the access time), outputs it on the data bus (5)
- the CPU read the value from the memory (6)

The following table shows the values of the different signals over time during a read cycle where the CPU reads the value 0x32 from address 0x105:

address bus	data bus	RD	WR
0x105	X	L	L
0x105	X	H	L
0x105	0x32	H	L

The following operations take place during a write cycle:

- the CPU puts the address of the desired memory location on the address bus
- the CPU turns its data bus into an output
- the CPU puts the value to be stored on the data bus
- the CPU asserts (brings high) the WR signal line
- the RAM detects the high signal on the WR line and turns its data bus into an input
- the RAM stores the value currently on the data bus into the desired memory location inside the RAM

Exercise 5: Draw a timing diagram and write out a table similar to those above showing the values appearing on the two strobes and the address and data buses when the value 0x33 is written to address 0x1200.

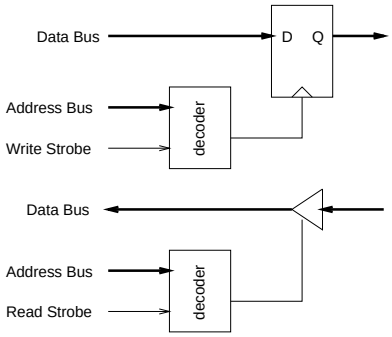
I/O Chips

I/O chips are used to allow the CPU to interface with peripherals (keyboards, printers, etc).

An output I/O chip consists of D flip-flops which are loaded during a write cycle. The flip-flops’ inputs (D) are connected to the data bus and the outputs (Q) are connected to the peripheral. An address decoder is connected to the flip-flop’s clock input. When the CPU does a write cycle to the memory location that the address decoder is designed for, a rising edge is seen by the clock signal of the flip-flop. This loads the flip-flop with the data on the data bus and “stores” them on the output pins.

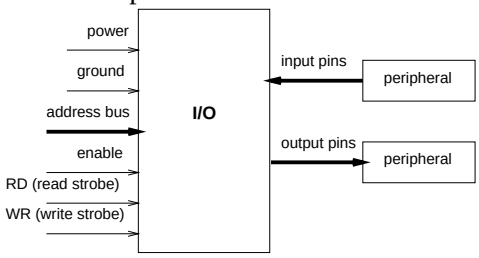
An input chip is a buffer that allows the value currently on the input pins to be placed onto the data bus. When the CPU does a read operation the buffer is enabled and the value on the input pins is read into the CPU.

The following diagram shows the internal structure of a simple parallel i/o chip:



There are many different I/O chips available. They usually include additional logic circuits to make it easier for the CPU to deal with specific peripherals such as modems or hard disks. Later in the course we will look at a few common interface chips.

The following diagram shows the external interface of an i/o chip:

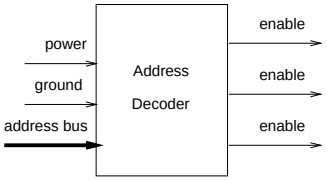


Address Decoders

A microcomputer often uses several memory and I/O chips. Typically each chip can hold an amount which is smaller than the total amount of memory that the microprocessor (CPU) can address. For example, a CPU with with a 16-bit address bus can address a 64 kBytes of memory but may be used in a system with a 16 kByte RAM chip, a 32 kByte EPROM chip and a 4-byte I/O chip.

Exercise 6: How many address lines are required by each of the above chips?

The purpose of the address decoder is to look at the address output by the CPU and enable individual memory or I/O chips. The following diagram shows the inputs and outputs of an address decoder:



Exercise 7: Draw a diagram showing how a CPU with an 8-bit data bus and a 20-bit address bus, two 8k by 8 RAMs, a 64k by 8 EPROM, an I/O chip with 4 internal one-byte ports and various address decoders would be connected to build a microcomputer. Show the connections of the data and address buses and the read and write strobes. Use arrows at each chip to indicate whether a particular signal is an input or an output. Indicate the width of each bus and the range of the address bus signals used by each chip.

Microcontrollers

A microcontroller is a single-chip microcomputer. The single chip includes the CPU, a RAM, an I/O chip and an EPROM or EEPROM. This allows all of the pins on the chip to be used for I/O. For example the Atmel 89C1051 is a typical microcontroller. It is a 20-pin chip which contains a clone of an Intel 8051 CPU, 64 bytes of RAM, 1kB of flash EEPROM for program storage and 12 pins which can be used for I/O. Current microcontrollers range from 8-pin devices that sell for less than $1 to chips with hundreds of pins that include Intel 486 processor ‘cores.’