The Basics of Microcontrollers

An Online Continuing Education Course for Engineers

Course Number: E-3053

Credit: 3 Hours / 3 PDH / 3 CPD

The Basics of Microcontrollers

Mark A. Strain, P.E.

Introduction

Microcontrollers and microprocessors whether seen or unnoticed are an integral part of everyday life from the time you get up in the morning to the time you retire at night. You quite possibly encounter hundreds every day.

Think about your day. Your alarm clock that woke you up contains a processor that controls the display, the clock and the radio. Your coffee maker that came on at a preset time contains a processor that controls the clock as well as your electric toothbrush that alerted you when to recharge the battery. The smart phone that you picked up to check your email and messages contains a handful of processors: one to control the display, keypad and user applications, one to control the GPS, one or two to control the phone radio system and one to control the Bluetooth transceiver. Your television that you turned on to get the news contains processors to control the display, receiver and remote as well as one for the cable box. After getting dressed and getting into the car you encountered a couple more: one in your remote garage door opener and one in the alarm system of your house. Your car in which you drove into work contains a dozen or so processors for the engine electronic control system, GPS, radio system, electronic compass, etc. Needless to say these tiny little black silicon brains have embedded themselves as an inseparable part of everyday life. Seemingly complicated technology within the tiny black square, when broken down into its individual subsystems, the tiny device becomes easily understood.

Architecture

Microcontrollers are composed of the following main subsystems: the central processing unit (CPU) or core, a memory interface (for data memory and program memory) and peripherals. There are two main categories of memory: nonvolatile and volatile. Nonvolatile memory is that which retains its memory after power is removed. Disk storage and flash memory are examples of nonvolatile memory. Volatile memory is that which is erased during a power cycle. Random access memory (RAM) is an example of volatile memory.

Program memory is memory where the program code or set of instructions for the machine resides. The program is usually written in a higher level language which is compiled and translated into machine code to be interpreted by the central processing unit (CPU). Program memory is permanent storage. It is nonvolatile; the data is not persistent over power cycles. Data memory is memory that the machine uses to temporarily store data and to store variables during execution. Sometimes the entire computer program is copied from program memory into data memory and execution begins in the area of data memory where the program is copied. Data memory is temporary storage. It is volatile; the data does not exist if power is removed. Three types of processor architectures will be discussed in this course: the Harvard architecture, von Neumann architecture and the modified Harvard architecture. The differences in the three designs focus on the processor's access to program memory and data memory.

Harvard Architecture

The Harvard architecture is a microprocessor design with separate memory and memory busses for program and data memory. A machine with a Harvard architecture has separate data and address busses for program and data [4].

The advantage of a machine with a Harvard architecture is that simultaneous access to program and data is possible. Many microcontrollers use this architecture to speed processing by simultaneous access to program code and data. Microcontrollers typically have small amounts of program (usually flash memory) and data (RAM) memory. Examples include the Atmel AVR and Microchip PIC microcontrollers.

In a machine incorporating a Harvard architecture program and data memory can have different timing and bus width. Instruction fetches and data access do not share the same path from the CPU to memory, so there is no collision of data (or bus contention) and a data access operation does not interfere with an instruction fetch. This feature makes a microprocessor designed with a Harvard architecture faster than one with a single bus from the CPU to memory.

In a Harvard architecture the data address 0x00000000 is not the same as program address 0x00000000. Program address 0x00000000 is typically where a processor begins execution out of reset. Figure 1 illustrates the Harvard architecture.

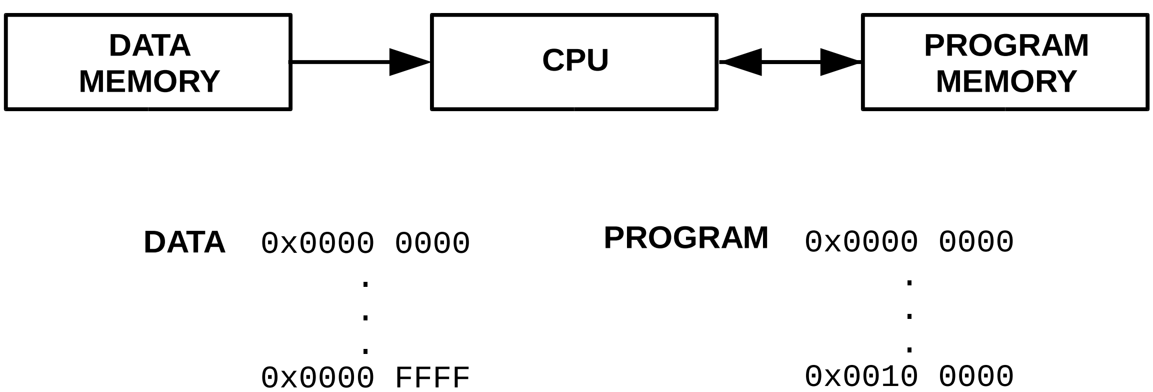

Figure 1 - Harvard Architecture

The Harvard architecture contains a program address 0x00000000 and a data address 0x00000000.

Von Neumann Architecture

The von Neumann architecture is named after John von Neumann (a renowned mathematician in the 1940s). The von Neumann architecture is a microprocessor design with the same memory bus for program and data memory. A machine with a von Neumann architecture stores data and program instructions in the same memory area – maybe even the same memory device [10].

The von Neumann architecture is called a stored-program architecture. This means that the machine's program instruction is read from some form of nonvolatile storage (program memory) such as a disk drive or flash memory device and written into volatile RAM (data memory). Once the program is written to RAM execution is transferred to the program in RAM and the program runs from RAM. Here the program instructions and the data are located in the same memory device. Code can be treated as data and data can be treated as code. Figure 2 illustrates the von Neumann architecture.

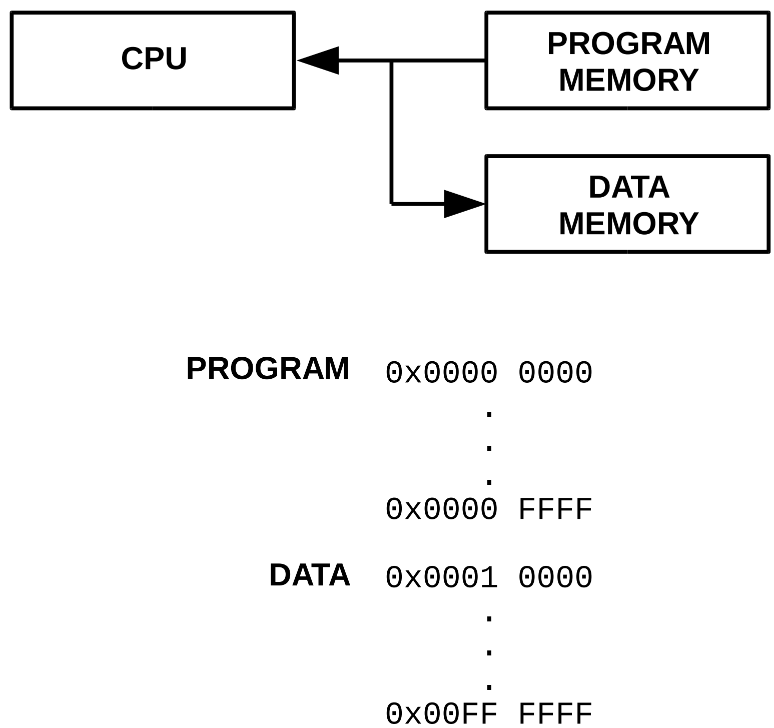

Figure 2 - von Neumann Architecture

The von Neumann architecture contains only one address 0x00000000.

Modified Harvard Architecture

The modified Harvard architecture is a design based on a combination of both the Harvard and von Neumann architectures. The modified Harvard architecture allows for the contents of program memory to be accessed as data memory. It allows concurrent access to both memory busses, thus relaxing the strict separation between instructions and data as in the Harvard architecture.

The following three characteristics describe processors based on the modified Harvard architecture [6]:

1. Program and data memories occupy different address spaces so that there is only one address 0x00000000.

An address space defines every possible memory location for the processor. For example, the reset vector (or the first instruction that the processor executes out of reset) may be located at address 0x00000000, the first data memory address may be located at 0x00010000, and a peripheral like Timer1 may be located at address 0xF0010000. Therefore, program memory will have different address locations than data memory.

Processors based on von Neumann and modified Harvard architectures have a single address space.

2. Program and data memories have separate busses to the CPU.

   The separate busses allow program instructions and data memory to be accessed simultaneously, thus improving throughput. Pure Harvard machines have separate pathways for data and program memory. Modified Harvard machines have memory caches or other tightly coupled memory with separate paths for program instructions and data memories.

3. Instruction and data memory may be accessed in different ways.

   Program instructions may be stored in nonvolatile flash memory and the data may be located in some form of RAM memory device, but the access to each is uniform, i.e. the same width, 16 bits for example.

Examples of processors incorporating the modified Harvard architecture include the x86, ARM, MIPS, Blackfin and PowerPC [6].

Central Processing Unit

The CPU is sometimes called the "core". It is called the core because it is the center or heart of the computational system. The CPU performs all of the calculations that occur and is the primary component carrying out the microcontroller's or microprocessor's functions.

The CPU is composed of the arithmetic logic unit, the control unit and registers.

Arithmetic Logic Unit

Almost all microcontrollers contain an arithmetic logic unit (ALU). It is a fundamental building block of the CPU. The ALU is a digital circuit that performs arithmetic and logical operations. The arithmetic operations include addition, subtraction, multiplication and division. These are integer operations, i.e., the inputs (or operands) are integers as well as the output. Some of the more complex processors will contain a floating point unit that handles fractional numbers in addition to integers. The logic operations include AND, OR, NOT and XOR as well as logical comparisons and bit shifting left and right.

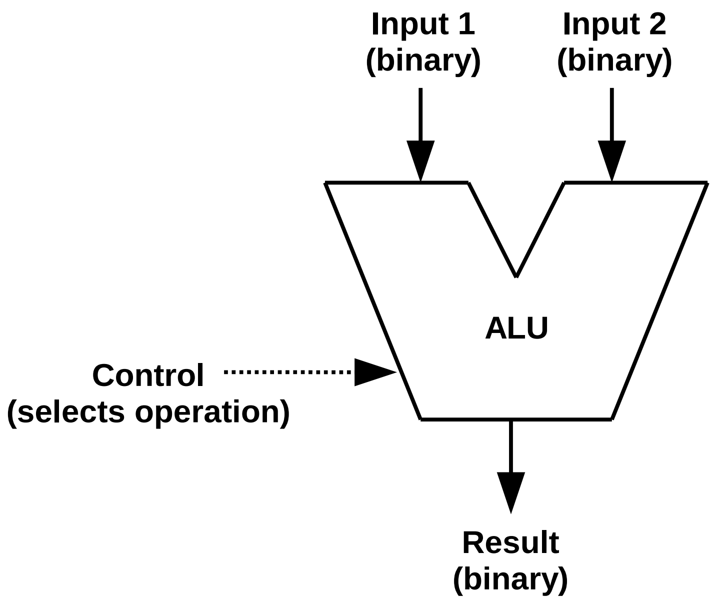

Figure 3 - Arithmetic Logic Unit

The ALU performs arithmetic and logic operations on two binary numbers resulting in another binary number. The numbers (the operands and the result) are represented using two's complement format. Two's complement allows subtraction to be performed by adding the complement of a number instead of subtracting the number. The utilization of two's complement greatly simplifies the ALU circuit since addition is a fundamental operation.

The following is an example of a two's complement operation (subtracting 109 from 202):

1100 1010 (202)
0110 1101 (109)

Take the two's complement of –109:
0110 1101 (109)

Invert all bits
1001 0010

Add one (this is the two's complement of –109)
1001 0011

Now add the two numbers:
1100 1010
1001 0011

0101 1101
The result is 93.

Multiplication and division are performed by bit shifting. Multiplication is performed by bit shifting to the left and division is performed by bit shifting to the right. A shift left by one bit is equivalent to multiplying the number by 2:

58 x 2 = 116
0011 1010 (58)

Shift the number left by one bit:
0111 0100 (116)

A shift left by two bits is equivalent to multiplying by 4:
58 x 4 = 232
0011 1010 (58)

Shift the number left by two bits:
1110 1000 (232)

Similarly, a shift right by one bit is equivalent to dividing by 2:
172 / 2 = 86
1010 1100 (172)

Shift the number right by one bit:
0101 0110 (86)

A shift right by two bits is equivalent to ... by 4:

The inputs to the ALU ... d the control lines to the ALU. Figure 3 show ... lines pass operation codes to the ALU which ... The output of the ALU is the result of the comp ... egister or a memory location. The result will ... tion.

Control Unit

The control unit function ... ike the ALU. It contains digital circuitry ... n instructions. The control unit configures th ... e memory busses for a read or write operati ... write operation. The control unit directs th ... code and execute. The fetch-decode-execute cycl ... ber crunching factory. The control unit is the facilitator of this ... unit fetches instructions from program memory. It then decomposes (or decodes) the instruction into its basic elements (opcode and operands).