Question 1

Define the following acronyms as they apply to digital logic circuits:

- ASIC
- PAL
- PLA
- PLD
- CPLD
- FPGA

file 03041

Answer 1

- ASIC: *Application-Specific Integrated Circuit*
- PAL: *Programmable Array Logic*
- PLA: *Programmable Logic Array*
- PLD: *Programmable Logic Device*
- CPLD: *Complex Programmable Logic Device*
- FPGA: *Field-Programmable Gate Array*

Follow-up question: now, comment on what each of these acronyms actually means, going beyond a mere recitation of the definition.

Notes 1

There is a veritable "alphabet soup" of acronyms in the world of programmable digital logic, and these are just a few. Going into the precise meaning of each acronym may not be the best use of time in answering this question, as there is little context in which to understand the meanings. Please do not attempt to do what so many technical courses do, and that is stuff students' heads with acronym definitions to the neglect of actually *understanding* the various technologies. This question is intended only as an opening to an in-depth discussion of programmable logic, and not an end in itself!

Question 2

Why would anyone use programmable logic devices (PLD, PAL, PLA, CPLD, FPGA, etc.) in place of traditional "hard-wired" logic such as NAND, NOR, AND, and OR gates? Are there any applications where hard-wired logic would do a better job than a programmable device?

file 03048

Answer 2

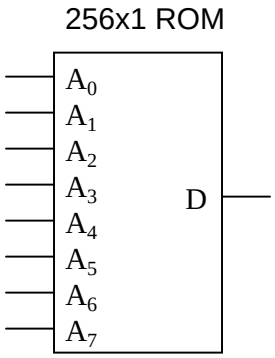
I'll let you do the research on this one!

Notes 2

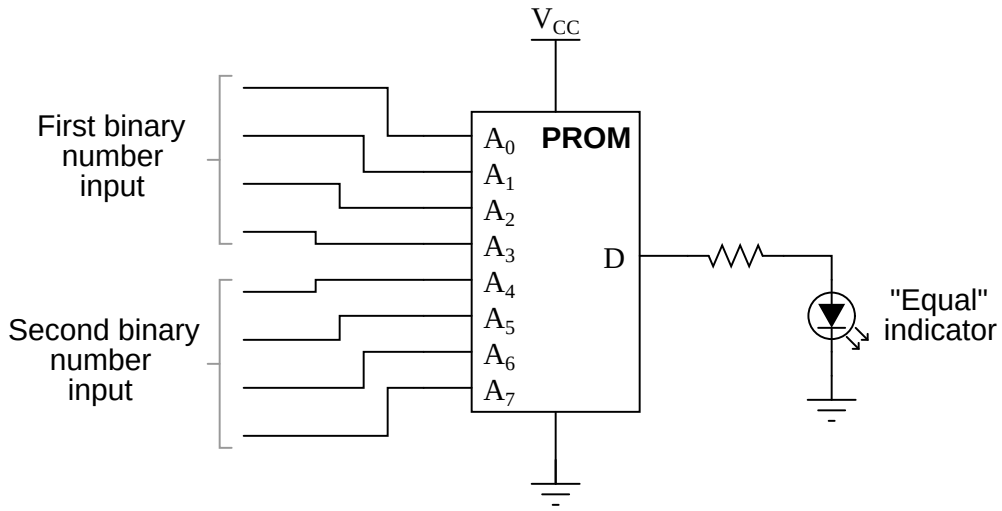
Ask your students to share where they found their information on programmable devices, and how they determined the advantages and disadvantages of this technology compared to hard-wired logic.

Question 3

Perhaps the simplest form of programmable logic is a PROM integrated circuit, programmed with a specific truth table. Take for instance this example of a 256 × 1 PROM:



Suppose we wished to program this memory IC to act as a digital comparator, outputting a logical "high" state only when two four-bit binary numbers are equal:



Describe what the truth table would look like for the data we must program into this memory chip. How many rows would the truth table have? Could you briefly describe the content pattern of the data without having to complete the entire truth table?

file 03043

Answer 3

Here's a clue: the truth table would only have sixteen rows with a "1" output. All other rows will be programmed with "0" outputs!

Notes 3

This is an example of a *look-up table*, whereby arbitrary data programmed into a memory circuit fulfills a logic function. If time permits, discuss with your students what other sorts of useful logic functions might be programmed into a PROM chip such as this.

Question 4

Microcontrollers are single-chip microcomputers, containing a microprocessor core, memory, I/O control, and other associated components necessary to make the system self-contained. Simply put, a microcontroller follows sequential instructions that someone enters into its memory.

Programmable logic devices, however, are fundamentally different from microcontrollers both in how they are programmed and how they function after being programmed. Explain what some of these differences are.

file 03044

Answer 4

Unlike microcontrollers, programmable logic devices are not (necessarily) sequential devices: the latter act as a collection of logic gates and other "primitive" logic elements to directly implement certain logic functions.

Notes 4

Discuss with your students how programmable logic devices are more primitive and direct devices than microcontrollers, which are more abstract by comparison. Perhaps the easiest distinction to understand is in terms of gate connections. In a microcontroller, the connections between its constituent gates are fixed; only the software (bits stored in memory) ever change. In a programmable logic device, it is as though you are directly forging connections between its constituent gates (as many or as few as needed), creating a hard-wired circuit by specifying connections in a "hardware description language" (HDL).

Question 5

The simplest types of programmable logic ICs are called PLDs (Programmable Logic Devices), PALs (Programmable Array Logic), PLAs (Programmable Logic Array), and GALs (Generic Array Logic). While each acronym represents a slightly different internal design architecture, these devices share a common feature of using inverters, AND gates, and OR gates to implement any desired combinational logic function.

Explain how it is possible to generate any arbitrary logic function with just these gate types (inverter, AND, OR), without any others. What principle or convention of Boolean algebra is used by these devices to do this?

file 03049

Answer 5

With a sufficient number of AND, OR, and inverter gates, any SOP or POS expression may be generated.

Notes 5

This question requires students to review the principles of how SOP and POS expressions relate to truth tables, and in doing so explain how any arbitrary truth table may be fulfilled.

Question 6

Some programmable logic devices (and PROM memory devices as well) use tiny fuses which are intentionally "blown" in specific patterns to represent the desired program. Programming a device by blowing tiny fuses inside of it carries certain advantages and disadvantages – describe what some of these are.

file 03050

Answer 6

Certainly, the stored program will be nonvolatile, but it will also be read-only. This is why fuse-programmed devices are sometimes called "OTP". (I'll let you research what that acronym means.)

Notes 6

It is interesting to mention that some programmable devices (Texas Instruments' TIBPAL series, for example) are built with a "security fuse" inside which prevents anyone from reverse-engineering a programmed chip!

Question 7

A common term used to describe the internal workings of a programmable logic device is a *macrocell*. What, exactly, is a macrocell?

file 03051

Answer 7

A macrocell is a collection of logic gates and a flip-flop, lumped together in one unit. PLDs usually have many macrocells, which may be interconnected to form a variety of synchronous logic functions.

Notes 7

Ask your students to show you where they found their information, and if they were able to determine how many macrocells are in a typical PLD.

Question 8

The similarities and differences between microcontroller (microprocessor) systems and programmable logic devices may be illuminated by analogy. Read the following scenarios where two different solutions are employed to solve common problems. For each scenario, determine which solution is analogous to a microcontroller and which solution is analogous to a programmable logic device:

A business manager must make a hiring decision: either hire several specialty-skilled employees to perform various tasks (one task per person), or hire a few broadly-skilled people who can be given new instructions and/or training to switch between different tasks as needed.

Two tinkerers are modifying pianos to play short songs automatically (with no human operator). The first decides to build a "tape reader" device similar to that of an old mechanical player-piano, where a paper scroll bearing punched holes "tells" the piano keys when to strike and in what order. The second decides to build a much simpler sequencing mechanism, where each key on the piano from left to right is struck in sequence, the proper ordering of notes in the song being arranged by re-connecting the keys to different hammers inside the piano.

file 03045

Answer 8

First scenario: broadly-skilled employees = microcontroller; specialty-skilled employees = programmable logic.

Second scenario: tape reader = microcontroller; re-linking keys to hammers = programmable logic.

Notes 8

Understanding the distinction between microcontrollers and programmable logic devices can be difficult, especially if one has limited experience with both (as most students do). Questions such as this, which ask students to examine opposing analogies, teaches some of the distinguishing principles without becoming mired in technical detail.

The fundamental principle I want students to see from these analogies is that microcontrollers and microprocessors are re-programmed by changing a *sequence* of fixed operations, while programmable logic systems are re-programmed by changing *associations* between fixed elements.

Question 9

Most microcomputers can only perform one task (operation) at a time. They achieve the illusion of "multi-tasking" by alternately devoting time to one of several tasks in a rapid fashion – a sort of multiplexed computation. Most programmable logic devices, on the other hand, are easily able to perform multiple logic operations in a truly simultaneous manner. Explain how this is possible, whereas a microprocessor can only do one thing at a time.

file 03046

Answer 9

The secret is in the programming: programmable logic devices are literally "wired" by the programs you write for them, with thousands of logic elements available to be connected in almost any way you desire. Microprocessors, on the other hand, have fixed wiring that responds to sequences of steps, the program merely specifying those sequence of those steps.

Notes 9

Understanding the distinction between microcontrollers and programmable logic devices can be difficult, especially if one has limited experience with both (as most students do). The purpose of this question is to shed some more light on this often misunderstood subject, while simultaneously highlighting an important feature of programmable logic: true simultaneity.

The fundamental principle I want students to see from these analogies is that microcontrollers and microprocessors are re-programmed by changing a *sequence* of fixed operations, while programmable logic systems are re-programmed by changing *associations* between fixed elements.

Question 10

Verilog and *VHDL* are two popular examples of a *hardware description language*, used when working with programmable logic. Explain the purpose of such a "language." What does it mean for a technician or engineer to "speak" this language, and how is it "spoken" to an actual programmable chip?

file 03047

Answer 10

A hardware description language (HDL) is a textual convention for specifying the interconnections of a programmable logic device. Text files are written by a human programmer, then "compiled" into a form that the programmable logic device can directly accept and use.

Notes 10

If time permits, you may want to compare and contrast fully-featured languages such as Verilog and VHDL with more primitive hardware description languages such as ABEL. In either case, though, files written in an HDL are intended to describe the interconnections of available logic elements inside a programmable logic device.