1 INTRODUCTION TO DIGITAL SYSTEMS ENGINEERING

Digital systems are pervasive in modern society. We use computers for bookkeeping, engineering, publishing, and entertainment. Digital communications systems handle our telephone calls and enable our Web browsing sessions. Other uses of digital systems are less visible. Most consumer electronics products are largely digital and becoming more so. Music today is distributed digitally on compact optical disks, and video production is rapidly becoming a digital process. A typical appliance is controlled digitally by a microcomputer. As many as ten microcomputers can be found in the average car for controlling functions ranging from the sound system to the antilock brakes.

A *digital* system represents information with discrete symbols (of which digits are a special case) rather than with a continuously varying quantity, as in an *analog* system. Most systems use just two symbols, often denoted by the binary digits (or *bits*) 0 and 1, to represent all information. Simple truth propositions are

represented directly with a single bit, whereas strings of bits are used to represent more complex data.

In a digital system, noise below a given level can be completely rejected. Symbols are encoded into ranges of voltage or current level. If we add a small amount of voltage, V_N, to the nominal voltage for the 0 symbol, V_0, the resulting voltage, $V_0 + V_N$, will still be in the range for a 0 symbol and, more importantly, can be restored to the nominal level, V_0. This property allows us to process information through many noisy stages of logic with no accumulation of noise. In an analog system, in contrast, disturbing an information voltage, V_x, with noise gives a voltage, $V_y = V_x + V_N$, that represents a different piece of information. The analog signal cannot be restored and will be severely degraded after many stages of noisy processing.

All digital systems, whether used explicitly for computation or as part of an entertainment system, are constructed from three basic components: logic, memory, and communication channels. Logic operates on symbols, for example adding two numbers together or comparing two characters. Memory stores symbols, moving them in time so that information computed at one point in time can be recalled later. Communication channels, usually wires, move information in space so that values generated by logic in one location can be stored in memory, used by logic in a different location, or both.

The development of a digital system involves circuit and logic design, architecture, and systems engineering. Individual components such as logic gates and memory arrays are designed at the circuit level. At the logic level we compose these individual components into subsystems such as adders and finite-state machines. At a higher level the instruction-set and register-level organization of the system is governed by the principles of computer architecture.

This book addresses digital systems engineering. The systems-level engineering issues in a digital system include power distribution, noise management, signaling, timing, and synchronization. Power distribution deals with how to distribute a controlled DC voltage level throughout a system that draws considerable amounts of AC current. Transmitting digital symbols over wires at maximum speed and with minimum power is the challenge of signaling. Timing deals with how computations and communications are sequenced within and between modules. The design of signaling and timing conventions is dominated by considerations of noise or uncertainty; thus, the successful digital designer must understand noise sources and formulate methods for managing noise. Synchronization is required to coordinate the operation of two systems operating from separate time bases or to sample an input that may change asynchronously.

We address these issues from systems and circuits perspectives. For each topic, we start by giving the big picture with circuit details abstracted and then return to the topic and present detailed circuits.

1.1 WHY STUDY DIGITAL SYSTEMS ENGINEERING?

As technology has advanced, the systems-level engineering problems of digital systems have become more critical. At high signaling frequencies wires can no

longer be considered equipotential regions and must be modeled as transmission lines. On-chip wires are becoming more resistive, presenting a significant signaling challenge. The number and speed of gates on a chip have increased faster than the number and speed of pins, making interchip communication a system bottleneck and placing a premium on efficient signaling and timing conventions. With reduced supply voltages, higher currents, and thinner metal layers, power distribution is an increasingly challenging engineering problem. At high frequencies timing conventions must be carefully designed to prevent skew and jitter of clock signals from degrading system performance.

Careful attention to digital systems engineering issues makes an enormous difference in the performance, reliability, and power dissipation of a system. Too often, lack of attention to these issues results in a system that is unreliable or simply does not work. Recently, a major computer manufacturer delayed the release of a new system because several custom chips failed to operate at any speed owing to excessive clock skew. Another manufacturer had to revise several chips in a new system because noise problems led to unacceptable error rates on interchip signals.

Digital systems engineering has largely been igonored by the academic community. Most curricula include courses on circuit design of logic elements, logic design, register-transfer-level design, and high-level architecture. However, the engineering problems of composing circuits into systems are only briefly touched upon. A frequent misconception at the system level is that one must abstract away the electrical properties of the system and deal with discrete symbols at the logic or architectural level. To the contrary, careful management of electrical issues at the system level is critical to the success of any modern system.

In fact, to do a credible job of computer architecture one must have an intimate understanding of system-level engineering issues. Architecture deals with organizing a system and defining interfaces (e.g., instruction sets and channel protocols) to achieve cost and performance goals. System-level engineering constrains what the architect can do and is a major determinant of the cost and performance of the resulting system. It is far too easy for one to abstract these issues and architect a system that, owing to bandwidth limitations, is unrealizable or suboptimal.

Many in industry have taken ad hoc appoaches to systems engineering problems. Clocks are distributed by open-loop trees that are tuned until the circuit works. Power is distributed on wide metal lines. Full-swing signals are used for signaling on and between chips. Typically, a company addresses a systems engineering problem using the same approach taken with its last system. This often leads to a false sense of security. As technology advances, ad hoc approaches that worked in the past become suboptimal and often fail. Many companies have been caught unawares by excessive power supply noise, unreliable signaling, and badly skewed clocks. In one classic example, manufacturers designed standard logic components using the high-inductance corner pins of their packages for power and ground. This was originally done to simplify routing of printed circuit (PC) boards without power planes. Over the years, standard logic parts increased in speed until their switching current induced so much noise across the supply

inductors that the output could not reliably be detected. Only after many parts failed in systems did the manufacturers relocate the pins to the center of the package.

In this book we will take an engineering science approach to systems engineering problems; that is, we will study the underlying physical and electrical mechanisms that relate to power distribution and signaling and timing conventions. We will develop models for these mechanisms and use these models to analyze the performance of potential solutions. We will present specific solutions to these problems that work under today's constraints, and, more importantly, we will give you the tools to understand why these solutions work and to evaluate whether they will continue to work under future constraints.

Standards, both published and de facto, have historically played a large role in digital engineering. The use of full-swing underterminated voltage-mode signaling and edge-triggered synchronous timing is almost universal even though, as we will see, there are far better signaling and timing alternatives. The popularity of the prevailing approaches derives from the many catalog parts available using these conventions and to a lesser extent to the lack of understanding about alternatives and the criteria for evaluating them.

The trend toward building digital systems from custom or semicustom very large scale integrated (VLSI) components such as application specific integrated circuits (ASICs) has lessened the importance of catalog parts, except for memories, and hence enables the system designer to control signaling and timing conventions. Because these ASICs largely communicate with one another, the system designer is free to choose any convention she wishes for communication between these components. The designer is also free to choose the conventions employed by and between the components on a single chip. Thus, at the same time that continued advances in VLSI fabrication technology drive many of the challenging systems problems, VLSI also offers an opportunity to solve these problems by adding degrees of freedom in the choice of signaling and timing conventions.

1.2 AN ENGINEERING VIEW OF A DIGITAL SYSTEM

An engineer views a digital system in terms of information flow, power flow, and timing. For example, Figure 1-1 is an engineering view of a node of a modern multicomputer. The arrows in the figure illustrate the information flow and are annotated with the signaling convention employed, and the individual components are annotated with their power and timing information. As we shall see, these are not independent variables. The choice of a signaling convention, for example, can determine both the available bandwidth from a component and the power it requires.

The heart of the system is a central processing unit (CPU) operating at 500 MHz and dissipating 80 W of power from a 2.5-V supply. The CPU is connected to a set of cache chips via data and address lines and to a controller chip via two unidirectional channels. The controller in turn connects via channels to a router,

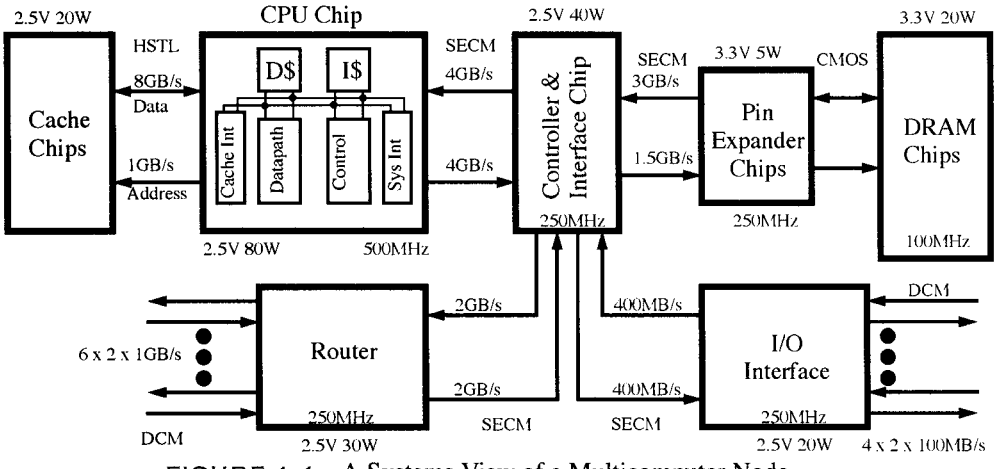

FIGURE 1-1 A Systems View of a Multicomputer Node

an I/O interface, and, via some pin expanders[1] to a set of dynamic random access memory (DRAM) chips. The router has six bidirectional channels that connect to the routers on other nodes, and the I/O interface has four channels that connect to peripherals.

1.2.1 Feeds and Speeds

Much of the art of digital systems design is managing the movement of information. A good design balances demands for bandwidth against the constraints of limited numbers of chip and module pins and limited amounts of wiring area on chips and boards. The designer controls the partitioning of the system and the topology of connections to strike the appropriate balance. This is a critical task. The efficient use of available bandwidth is probably the single most important determinant of overall system performance.

A key tool in the process of balancing a system is a diagram, like Figure 1-1, that shows the *feeds*, where information flows, and the *speeds*, how fast (in bits/s) the information flows along each path. This information is used in two ways. First, together with the signaling rate, the combined speeds into and out of each component or module indicate how many pins are required. Second, system performance can be predicted from the bandwidths at various points in the system in conjunction with an architectural model (usually a simulator).

For example, suppose our architectural model indicates that the bandwidth from the CPU to the second-level cache should equal the bandwidth from the CPU to the rest of the system. Our packaging allows at most 300 signal pins on

[1] A pin-expander chip takes a few pins that use a high-speed signaling convention, like single-ended current mode (SECM) at 500 Mb/s, and converts them to a larger number of pins in a slower signaling convention such as full-swing complementary metal-oxide semiconductor (CMOS) at 100 Mb/s.

TABLE 1-1 **Pin Counts for System of Figure 1-1**

Chip	Signal	Speed (GB/s)	Rate (Mb/s-pin)	Pins
CPU	Cache data	8	500	128
	Cache address	1	500	16
	Channels	8	500	128
	Total			272
Controller	CPU channels	8	500	128
	Router channels	4	500	64
	Memory channels	4.5	500	72
	I/O channels	0.8	500	14
	Total			278
Router	Controller channels	4	500	64
	Router channels	12	250	384
	Total			448

a component. Our signaling convention dictates that all signals must be point-to-point. Signals on the board operate single-ended at 500 Mb/s-pin, and signals that go off the board (router and I/O outputs) operate at differentially 500 Mb/s over a pair of pins for 250 Mb/s-pin. Using these numbers, Table 1-1 shows the number of pins required by the three major chips in the system.

The table shows that the CPU and controller chip are within the pin budget, and a few pins are left over. The router, however, is way over the pin budget, and thus the system as currently sketched cannot be implemented with the planned packaging. At this point the designer needs to repartition the system (see Exercise 1.1) to eliminate this bottleneck, reduce the router bandwidth, with possible performance penalties, or attempt to remove the constraint by getting a package with more pins or using a faster signaling convention.

1.2.2 Signaling Conventions

A signaling convention is the method used to encode symbols (usually 1 and 0) into physical quantities (e.g., voltage and current). An efficient signaling convention is one that maximizes the bandwidth per pin, minimizes power dissipation, and provides good noise immunity. Such a signaling convention can dramatically increase available bandwidth and hence system performance. Suppose, for example, the entire system in Figure 1-1 were implemented using a full-swing CMOS signaling convention that was limited to 100 Mb/s-pin. The CPU and controller chips would require about 1,400 signal pins! With CMOS signaling, a system of this level of performance simply could not be implemented.

Most of the channels between chips on the same PC board in Figure 1-1 use single-ended current-mode (SECM) signaling. This convention encodes the symbol 1 as a +2.5-mA current and a 0 as a −2.5-mA current. The wire over

which a signal travels is designed to be a 50-Ω transmission line and is parallel-terminated with a 50-Ω resistor on the receiving chip into a midrail (1.25-V) supply. This gives a voltage swing of 250 mV, from 1.125 to 1.375 V. The receivers detect the arriving signal by measuring the voltage across the termination resistor with a sensitive amplifier.

In contrast, most systems today use a form of full-swing signaling. For example, the CMOS signaling convention, as employed by the DRAMs in Figure 1-1, encodes 1 as the positive supply voltage (3.3 V) and 0 as the negative supply voltage (0 V). The impedance of the wires is uncontrolled, and the lines are unterminated. A CMOS inverter is used to detect the received voltage in the reference frame of the receiver power supplies.

Why is current-mode signaling preferable to CMOS signaling? Current-mode signaling offers lower power, faster operation, and better noise immunity. The only advantage of CMOS signaling is that it is used by the vast majority of standard parts. To interface to such parts (e.g., the DRAMs in Figure 1-1) one must use this signaling convention.

1.2.2.1 Signaling Speed

Current-mode signaling offers faster operation because it employs incident wave signaling for minimum delay and uses a terminated transmission line to minimize intersymbol interference. Suppose the average on-board wire is 30 cm long. The current-mode signaling system injects a 250-mV signal at one end of this line, and 2 ns later the wave carrying this signal arrives at the termination and is absorbed into the terminator. No residual energy is left on the line to corrupt subsequent bits. Thus, the line can immediately be used to send a second bit. In fact, several bits can be pipelined on a long wire. By getting the symbol on and off the line quickly, current-mode signaling enables fast signaling rates.

In contrast, the average CMOS driver, with an output impedance of 200 Ω cannot drive the 50-Ω line through 3.3 V in a single step. To do so would require a very low output impedance and enormous current (66 mA). Instead this driver injects a 660-mV (13-mA) signal into the line. After 2 ns, this wave reflects off the receiving end of the line, giving 1.3 V. Only after the wave reflects off the source and then the receiver for a second time is the line voltage sufficiently high to be detected as a logic 1. Even after 10 ns (2.5 round trips) the line is still over a half volt short of 3.3 V. This residual state leads to intersymbol interference. Because CMOS signaling takes a long time to get a symbol on the line, it is limited to signaling periods that are many times the delay of the line.

1.2.2.2 Signaling Power

By operating with a small signal swing, current-mode signaling dissipates much less power. The current-mode signaling convention draws 2.5 mA from a 2.5-V supply for a power dissipation of 6.2 mW. The actual signal power is ten times less, 620 μW (2.5 mA times the 250-mV signal swing). Operating at 500 Mb/s, the signal energy per bit is 1.25 pJ (2 ns times 620 μW), and the power supply energy per bit is 12.5 pJ.

In contrast, the CMOS system operates with a peak current of 13 mA and an average current of 9.4 mA (assuming the line toggles each cycle) from a 3.3-V supply for a power dissipation of 31 mW. Operating at 100 Mb/s, for the reasons described above, the signal energy per bit is 310 pJ or about 250 times that of the current-mode system![2] The power supply energy is the same here as the signal energy (about 25 times that of the current-mode system).

1.2.2.3 Signal Integrity

At this point you might think that with such a large signal swing the CMOS system must be much more reliable in the presence of noise. In fact the opposite is true. As we shall see in Chapter 7, the current-mode system has better noise immunity because it isolates the signal from external noise sources, such as power supply noise, and reduces internal noise sources, such as receiver sensitivity and intersymbol interference.

1.2.2.4 Other Signaling Conventions

The same arguments apply to signaling conventions on-chip. For example, using full-swing signaling to communicate between the instruction cache, I$, and control unit in Figure 1-1, would waste power. However, the constraints for on-chip signaling are different, and hence different solutions are appropriate.

Two other off-chip signaling conventions appear in the channel labels of Figure 1-1. A differential current-mode signaling convention (DCM) is used for all signals that leave the PC board to avoid the problems of signal-return cross talk that would otherwise occur owing to the large return impedance of typical off-board cables and connectors. The cache static random access memory (SRAM) chips are shown using HSTL,[3] an underterminated signaling convention (Section 7.3.4) that is just beginning to be offered in standard parts such as SRAMs.

1.2.3 Timing and Synchronization

A timing convention governs the sequencing of data through logic and across channels. The convention governs when signals can change and when they are sampled.

1.2.3.1 Synchronous Timing

Within a *clock domain*, an area where all signals have their timing referenced to a single clock signal, the convention may be built around edge-triggered flip-flops or level-sensitive latches. Some conventions, such as using multiple nonoverlapping clock phases, are very tolerant of clock skew, whereas others require very careful clock distribution to minimize skew.

[2] The fact that the CMOS system does not reach 3.3 V after 10 ns works in its favor here. The energy to charge a 30-cm, 50-pF line to 3.3 V is 545 pJ ($3.3^2 \times 50$ pF).

[3] HSTL is an acronym for high-speed transceiver logic.

For example, the system of Figure 1-1 employs a separate clock domain for the core logic of each chip. Most of the chips operate at 250 MHz, whereas the CPU operates at 500 MHz. The internal logic in each clock domain uses level-sensitive latches and a single-phase clock. Minimum delay constraints are used to relax clock skew tolerances.

1.2.3.2 Pipelined Timing

With all signals referenced to a single clock, the maximum operating frequency depends on the propagation delay through the logic or across the channel. In situations where data flow primarily in one direction and delays are predictable (e.g., most channels), substantially higher throughput can be realized by using *pipelined timing*. With pipelined timing, each pipeline stage operates in its own clock domain, and the phase of the clock is delayed to match the delay of the data. Therefore, the maximum throughput is independent of propagation delay and is limited only by the uncertainty in the delay: skew and jitter. In Figure 1-1, the channels between the controller and the other components all operate using pipelined timing, allowing operation with clock periods much shorter than the delay of the wires. Pipeline timing is also occasionally used to advantage in the design of pipelined arithmetic units. In this context, it is sometimes referred to as *wave-pipelining*.

1.2.3.3 Closed-Loop Timing

The static portion of delay uncertainty, skew, can be eliminated by using a control loop that measures the relative delay or phase of two signals and adjusts a variable delay or frequency element to match delays. This form of closed-loop timing allows operation at very high frequencies without requiring very tight tolerances or matching of wire and logic delays. The dynamic portion of uncertainty, jitter, can be reduced by using a phase-lock loop that tracks the low-frequency variations in an incoming signal while rejecting high-frequency jitter.

1.2.3.4 Clock Distribution

Distributing a clock over a large clock domain with low skew and controlled duty factor is a challenging engineering problem. Off-chip, clock trees with matched transmission lines and buffers are typically used. On-chip, the problem is more difficult, for the lossy nature of on-chip wires leads to large diffusive delays, and power supply variations modulate the delay of clock buffers. The CPU chip in Figure 1-1, for example, uses a six-level clock tree with the leaves shorted in a low-resistance, two-dimensional grid to control clock skew across the chip to within 300 ps. This skew is large compared with the typical gate delay (100 ps). However, by using a timing convention that employs a two-phase nonoverlapping clock driving transparent latches and that constrains the minimum delay between latches, the design easily tolerates the 300-ps skew without increasing the cycle time and without any danger of hold-time violations. In designing the timing of a system, an engineer chooses between methods to control skew and methods that tolerate skew. A well-designed system usually balances the two approaches.

1.2.3.5 Synchronization

Before signals from different clock domains can be combined, they must be *synchronized* to a common clock. For example, the receiving side of each channel in the controller chip of Figure 1-1 operates in its own clock domain as a result of the pipelined timing employed. Before these signals can be combined in the core of the controller, they must be synchronized to the core clock. In general, synchronizing a signal to a clock requires a delay as the synchronizer waits for metastable states to decay. However, the synchronization being performed in Figure 1-1, and in most digital systems, is a special case in that the events being synchronized are periodic. Thus, future transitions can be predicted in advance and the signals synchronized without delay.

1.2.4 Power Distribution

A digital system requires a stable DC supply voltage, to within a few hundred millivolts, to ensure proper operation of logic and communication circuits. The power distribution system must provide this steady voltage in the presence of very large AC current demands. The resistive nature of on-chip wires and the inductance inherent in most packaging elements make this a difficult problem.

A modern CMOS circuit can vary its current draw from 0 to a maximum value in a fraction of a clock cycle. In the system of Figure 1-1, for example, the CPU dissipates 80 W at 2.5 V for an average current of 32 A. The peak current is easily twice this amount, 64 A, and current demand can readily change from zero to this maximum value in half a clock cycle, 1 ns, for a peak derivative of $di/dt = 64$ GA/s. Even with hundreds of supply pins (of a few nH each), this current transient induces an unacceptable voltage transient across the parasitic inductance of the package. On-chip bypass capacitors and perhaps regulators are required to manage this transient.

A typical power distribution system is a hierarchy. Small local elements, like on-chip bypass capacitors and regulators, provide small amounts of energy to local regions and handle the high-frequency components of transients. Larger elements supply larger regions and handle lower-frequency components of the transients. Because of their physical distance from the point of use, and the inductance that implies, they are not able to manage the high-frequency transients. At higher levels of the hierarchy, the supply voltage is usually raised to allow distribution to be performed with lower currents and hence smaller and less expensive bus-bars and cables.

1.2.5 Noise

Noise is a major concern in the engineering of digital systems. It corrupts signals on channels between modules, disturbs the state of logic networks and memory cells, and adds jitter and skew to the timing of signals. Signaling and timing conventions are designed around the constraints of system noise. For example, both the amplitude of signal swing and the clock cycle of a pipelined timing system are determined primarily by noise constraints.