

INTERFACING TECHNIQUES

In Digital Design

**With Emphasis On
Microprocessors**

Ronald L. Krutz

INTERFACING TECHNIQUES IN DIGITAL DESIGN WITH EMPHASIS ON MICROPROCESSORS

Ronald L. Krutz, Ph.D.

Director, Computer Engineering Center
Mellon Institute
Carnegie Mellon University



John Wiley & Sons

New York · Chichester · Brisbane · Toronto · Singapore

Copyright © 1988, by John Wiley & Sons, Inc.

All rights reserved. Published simultaneously in Canada.

Reproduction or translation of any part of this work beyond that permitted by Sections 107 and 108 of the 1976 United States Copyright Act without the permission of the copyright owner is unlawful. Requests for permission or further information should be addressed to the Permissions Department, John Wiley & Sons.

Library of Congress Cataloging in Publication Data:

Krutz, Ronald L., 1938—

Interfacing techniques in digital design with emphasis on microprocessors.

Includes index.

1. Interface circuits. 2. Microprocessors.
3. Digital electronics. I. Title.

TK7868.I58K78 1988

621.398

87-25227

Printed in Singapore

10 9 8 7 6 5 4 3 2 1

8967284

**INTERFACING TECHNIQUES
IN DIGITAL DESIGN
WITH EMPHASIS
ON MICROPROCESSORS**

Wiley & Sons

John Wiley & Sons, Inc. New York, N.Y.

To Hilda, Sheri, and Lisa

*The challenges are simpler—
the failures are fewer—
the rewards are greater*

..... because of you

ABOUT THE AUTHOR

Ronald L. Krutz is Associate Director of Mellon Institute, Carnegie Mellon University. He is also Director of the Mellon Institute Computer Engineering Center. Prior to this position, Dr. Krutz was on the faculty of the Carnegie Mellon Electrical Engineering Department and held senior management and research positions with the Singer Corporate Research & Development Laboratory and Gulf Research Laboratories. He is a Registered Professional Engineer in Pennsylvania and is the author of a videotape course on microprocessors (1984) as well as two other textbooks: *Microprocessors and Logic Design* (1980) and *Microprocessors for Managers* (1983).

PREFACE

Microprocessor interfacing technology is evolving as dynamically as the VLSI components it supports. The techniques of interfacing are based on fundamental electrical engineering principles, advanced microfabrication rules, and computer hardware and software design concepts. It is the aim of this book to tie together these different areas and develop interfacing skills based on practical design experience and meaningful, relevant examples.

Since an increasing number of microcomputer peripheral circuits are being integrated onto higher density chips, a complete coverage of interfacing should include an introduction to basic on-chip device and logic circuit design principles. This approach develops an understanding of the on- and off-chip drive requirements that have to be considered by the designer. Following a discussion of microprocessor interfacing methodologies, terms, and definitions in Chapter 1, the fundamentals of MOS and CMOS logic devices and their interfacing are presented in the first part of Chapter 2. Then transmission line principles and formulas are developed in detail and related to on- and off-chip line-driving applications. Useful transmission line design rules are derived and illustrated with examples using commonly available components. Chapter 2 concludes with a review of noise sources in digital circuits and presents the corresponding shielding and grounding techniques to minimize the associated problems.

Chapter 3 is devoted to the analysis of bus architectures and the interfacing to these buses. Specific sections include detailed coverage of dynamic RAM interface design, EEPROM system design, and HSCMOS/NMOS interfacing. Building on the detail of Chapter 3, the material in Chapter 4 covers serial and parallel interface standards and gives specifications for the most popular of these standards. Detailed examples using available components illustrate the application of serial and parallel interface standards.

As industry standards, the architecture of the IBM PC and the later architectures using the 80386 microprocessor are the main topics of Chapter 5. An introductory summary of the IBM Personal System/2 family is also provided. Through these architectural media, the interfacing material developed in the preceding chapter is illustrated, reviewed, and extended. Details of 80386 memory and I/O interfacing are specifically covered in depth. Included in this coverage are 80386 bus timing, pipelined and non-pipelined cycles, and cache memory interfacing. This approach provides valuable exposure to the student who will more than likely encounter these architectures in many applications.

Realizing that the "real world" is analog in nature, the conversion and interfacing of the analog signals to digital form is explained in detail in Chapter 6. The reverse technology, digital-to-analog conversion, is also developed.

In light of the emphasis on Computer Integrated Manufacturing (CIM) and Computer-Aided Engineering (CAE) in most industries, the topic of local area networks (LAN's) in Chapter 7 is especially relevant. In particular, the ISO-OSI standards and the related MAP standard for manufacturing system communication as presented in Chapter 7 are essential knowledge for the student now and in the future. Also, other token passing and CSMA/CD LAN standards are defined and discussed in depth.

Chapter 8 includes material not usually associated with interfacing—software. In most actual design and interfacing problems, however, software development is the most critical and costly area to be addressed. In particular, software engineering techniques and languages such as C and PL/M-86 that are used to develop interfaces are summarized and relevant examples are presented.

RONALD L. KRUTZ

CONTENTS

| | | |
|----------|---|----------|
| 1 | MICROPROCESSOR INTERFACING: AN OVERVIEW | 1 |
| 1.1 | Interfacing Layers | 1 |
| 1.1.1 | Other Considerations | 2 |
| 1.2 | Definitions | 2 |
| 1.2.1 | Buses | 3 |
| 1.2.2 | Memory Characteristics | 4 |
| 1.3 | Costs | 5 |
| 1.4 | Interfacing | 6 |
| 2 | CIRCUIT AND ELECTRICAL INTERFACING CONSIDERATIONS | 7 |
| 2.1 | Internal Characteristics of MOSFET Digital Systems | 7 |
| 2.1.1 | MOSFET Circuits | 9 |
| | NOR/NAND Circuits | 11 |
| | Pass Transistor | 11 |
| | CMOS Structure | 13 |
| | Interface Between the Real World and Digital Systems | 13 |
| 2.2 | Transmission Line Considerations | 14 |
| 2.2.1 | Effects of Characteristic Impedance on Signal Propagation | 18 |
| 2.2.2 | Application of Transmission Line Characteristics to MOS Circuits | 25 |
| 2.3 | MOS Memory as Transmission Line Load | 28 |
| 2.3.1 | Other Methods of Avoiding Oscillation Problems | 29 |
| 2.4 | Noise | 31 |
| 2.4.1 | Noise Sources | 31 |
| 2.4.2 | Shielding Techniques | 32 |
| 2.4.3 | Summary of Noise Sources, Effects, and Noise Reduction Techniques | 34 |
| 2.4.4 | Interconnection Wiring Noise Reduction | 34 |
| 2.4.5 | Grounding Connections | 38 |

3 BUS INTERFACING 41

- 3.1 Delays 41
- 3.2 Asynchronous Buses 43
 - 3.2.1 Asynchronous Bus Read 43
 - 3.2.2 Asynchronous Bus Write 45
- 3.3 Synchronous Buses 45
 - 3.3.1 Synchronous Bus Read and Write 45
 - 3.3.2 Synchronous Bus Utilizing WAIT State (Semisynchronous) 46
 - 3.3.3 Thirty-Two-Bit Bus Standards 48
- 3.4 Memory and Peripheral Interfacing 49
 - 3.4.1 Dynamic RAM 49
 - Power Supply 50
 - Memory Cycles 54
 - Timing Estimates 56
 - Detailed Dynamic RAM Design Example 57
 - 3.4.2 High Speed CMOS Logic Interfacing 72
 - NMOS to HSCMOS 72
 - HSCMOS to NMOS 73
 - Noise Margin 74
 - Microprocessor Bus Interfacing Example 74
 - Timing Characteristics 76
 - 3.4.3 Electrically Erasable Programmable Read Only Memory (EEPROM) 79
 - Processor Interface 79
 - Read Mode 79
 - Byte Write/Erase 80
 - Chip Erase 82
 - 8298 Integrated Controller 83

4 COMMUNICATIONS AND DATA TRANSFER 86

- 4.1 Serial Communications 86
 - 4.1.1 Asynchronous Transmission 86
 - 20 Milliampere Current Loop 88
 - RS-232-C 88
 - RS-232-C Specification Limits 92
 - Typical RS-232-C Connection and Timing 93
 - RS-422-A, RS-423-A, RS-449 95
 - RS-422-A 95
 - RS-423-A 96
 - RS-449 97
 - RS-485 99

- 4.1.2 Synchronous Transmission 102
 - HDLC/SDLC Data Link Control 102
- 4.2 IEEE Standard Digital Interface for Programmable Instrumentation, IEEE STD 488-1978, Parallel/Asynchronous 115
 - 4.2.1 Terminology and Definitions 116
 - 4.2.2 Logic Levels 118
 - 4.2.3 Drivers and Receivers 118
 - 4.2.4 Example Circuits 119
 - 4.2.5 Handshake Timing 122
 - 4.2.6 General Interface Management Bus 130
- 4.3 Microprocessor Parallel Communications 132
 - 4.3.1 Addressing 133
 - 4.3.2 Basic I/O Circuit 134
 - 4.3.3 Strobed I/O Circuits 135
 - 4.3.4 Pseudobidirectional I/O Port 137

5 IBM PERSONAL COMPUTER AND 80386 PROCESSOR INTERFACING, ASPECTS OF IBM PERSONAL SYSTEM/2 FAMILY 139

- 5.1 IBM PC 139
 - 5.1.1 System Board/RAM Interface 140
 - 5.1.2 I/O Channel 143
 - General Interface 143
 - 5.1.3 SDLC Controller (8273) as Used on SDLC Communications Adapter 153
 - 5.1.4 Printer Adapter 156
 - 5.1.5 General RAM Interface 158
 - 5.1.6 General ROM Interface 160
 - 5.1.7 IBM PC Summary 160
- 5.2 IBM Personal System/2 Summary 160
- 5.3 80386 Microprocessor 162
 - 5.3.1 80386 General Description 163
 - 5.3.2 80386 Structure 163
- 5.4 80386 Bus Interface 164
 - 5.4.1 Bus Timing 165
 - 5.4.2 Non-Pipelined Read Cycle 165
 - 5.4.3 Non-Pipelined Write Cycle 167
 - 5.4.4 Pipelined Cycles 168
 - 5.4.5 Interrupt Acknowledge Cycle 169

| | | |
|-------|--------------------------------|-----|
| 5.4.6 | Timing Considerations | 172 |
| | READY# Timing | 172 |
| | READ Timing | 172 |
| | WRITE Timing | 172 |
| 5.4.7 | LOCK Cycle | 173 |
| 5.4.8 | Reset Timing | 173 |
| 5.5 | Interfacing to Memory | 175 |
| 5.5.1 | Dynamic RAM Subsystem Design | 176 |
| 5.5.2 | Dynamic Memory Refresh | 177 |
| 5.6 | 80386 Cache Memory Interfacing | 179 |
| 5.6.1 | Organization of Cache Memory | 180 |
| 5.6.2 | Cache Data Integrity | 183 |
| 5.6.3 | Typical Cache Performance | 183 |
| 5.6.4 | Example Design | 183 |
| 5.7 | I/O Interfacing | 186 |
| 5.7.1 | General I/O Interface | 187 |

6 ANALOG INTERFACING TO MICROCOMPUTERS 192

| | | |
|-------|--|-----|
| 6.1 | Analog-to-Digital Converters | 192 |
| 6.1.1 | Approach | 192 |
| 6.1.2 | Errors and Accuracy | 193 |
| 6.1.3 | Temperature Effects | 198 |
| 6.1.4 | Methods of A/D Conversion | 200 |
| | Counter Converter | 200 |
| | Timing | 200 |
| | Successive Approximation A/D Converter | 201 |
| | Flash Converter | 202 |
| | Integrating A/D Converters | 205 |
| | Single-Slope (Single-Ramp) | 205 |
| | Dual Ramp (Dual Slope) Converter | 206 |
| 6.1.5 | Example Error Calculations | 209 |
| 6.2 | Digital-To-Analog (D/A) Converters | 209 |
| 6.2.1 | Weighted Current Source | 210 |
| 6.2.2 | R-2R Network D/A Converter | 211 |
| 6.3 | Interfacing A/D and D/A Converters | 213 |
| 6.3.1 | Data Acquisition Systems | 213 |
| 6.3.2 | D/A Converter Interface | 215 |
| 6.3.3 | Additional A/D Interface Example | 216 |

7 Local Area Networks 217

- 7.1 Topology 217
 - 7.1.1 Star 218
 - 7.1.2 Ring 218
 - 7.1.3 Bus 219
- 7.2 Related Communications Terminology 219
 - 7.2.1 Frequency Division Multiplexing 220
 - 7.2.2 Time Division Multiplexing 220
 - 7.2.3 Space Division Multiplexing 220
 - 7.2.4 Broadband and Baseband Approaches 221
- 7.3 Architecture 222
 - 7.3.1 International Standards Organization OSI Model 223
 - 7.3.2 Medium Access Options 227
 - Token Passing 227
 - Token Ring 228
 - Token Bus 229
 - MAP 233
 - TOP 240
 - CSMA/CD 241
 - Carrier Sensing 242
 - Detection of Collision 243
 - Retransmission 243
 - Ethernet Message Format 244
 - Preamble 245
 - Destination Address 245
 - Source Address 245
 - Type Field 245
 - Data Field 245
 - Frame (Packet) Check Sequence 245
 - Bit Encoding 246
 - Typical Implementation 246
 - Some Products 251

8 SOFTWARE INTERFACING 252

- 8.1 Modular Programming 252
- 8.2 Structured Programming 254
- 8.3 Programming Style 255
- 8.4 BASIC 257
- 8.5 FORTRAN 259

| | | |
|--------|--|-----|
| 8.6 | PASCAL | 261 |
| 8.7 | Ada | 262 |
| 8.8 | LISP | 264 |
| 8.8.1 | Definitions | 265 |
| 8.8.2 | Examples | 265 |
| 8.8.3 | Control | 267 |
| 8.9 | PL/M-86 | 268 |
| 8.9.1 | Background | 268 |
| 8.9.2 | Description | 268 |
| 8.9.3 | Identifiers | 268 |
| 8.9.4 | Data Types | 269 |
| 8.9.5 | Operators | 271 |
| 8.9.6 | Declarations | 273 |
| 8.9.7 | Initialization | 274 |
| 8.9.8 | Utilization of Data and Initial Attributes | 274 |
| 8.9.9 | Procedures | 275 |
| 8.9.10 | Attributes | 277 |
| 8.9.11 | PL/M-86 Flow Control | 281 |
| 8.9.12 | PL/M-86 Built-In Procedures | 288 |
| 8.9.13 | Examples | 291 |
| 8.9.14 | Conclusion | 297 |
| 8.10 | The C Programming Language | 297 |
| 8.10.1 | General Description | 298 |
| 8.10.2 | Variables | 298 |
| 8.10.3 | Preprocessor | 300 |
| 8.10.4 | Operators | 301 |
| 8.10.5 | Program Structure and Control | 301 |
| 8.10.6 | Library Functions | 302 |
| 8.10.7 | Reference | 303 |
| 8.10.8 | Interfacing Example | 303 |
| 8.10.9 | Conclusion | 322 |

Appendix A ASCII CHARACTER SET 323

Appendix B INTERFACE-RELATED READINGS 324

| | |
|---|-----|
| Real-Time Operating System Needs | 324 |
| DIM: A Network Compatible Intermediate Interface Standard | 332 |
| VME Bus Interfacing: A Case Study | 359 |

INDEX 379

1

MICROPROCESSOR INTERFACING: AN OVERVIEW

Interfacing is a term that applies across a broad range of electronic implementations. It relates to systems as well as to individual transistors. Because of this breadth, interfacing involves many technologies that are encompassed by the disciplines of electrical engineering and computer engineering. Since these technologies are usually separated into subdisciplines with their own areas of emphasis, their integration into a reasonably concise and unified treatment is desirable. The treatment of interfacing in this book is directed toward the integration of these various subdisciplines.

1.1

INTERFACING LAYERS

Interfacing usually involves effectively traversing a boundary from one entity to another. In the field of electronics, the entities can be viewed in a hierarchical fashion from a system, subsystem, component, and transistor level. Boundaries have to be traversed in all these levels. In doing so, opposite ends of the spectrum, such as the effects of electrons in motion and instruction execution times, may have to be considered.

In a microprocessor-based system, a number of levels can be defined arbitrarily to attempt to structure the interfacing hierarchy. One possible layering is

- electrical(physical)
- signal
- logic
- protocol
- code
- algorithmic/heuristic

Note that this model includes software implementations as part of the hierarchy. In microprocessor-based systems, especially real-time data acquisition and control applications, the software “interface” to the equipment to be controlled is a costly and critical portion of the total implementation. This aspect, denoted by the code and algorithmic/heuristic levels, is treated in Chapter 8.

Protocol interfacing is studied through communications and local area network standards and implementations. Interfacing at the logic level is addressed through circuit applications, bus interfacing, and data transfer. Signal interfacing techniques are developed in electrical interfacing, bus interfacing, and data transfer techniques. The physical interfacing techniques are developed through the electrical and analog interfacing coverages. All the levels are summarized and illustrated by means of the IBM Personal Computer family and 80386 microprocessor-based material.

1.1.1 OTHER CONSIDERATIONS

In the majority of interfacing applications, timing considerations are critical and must be incorporated into the overall design. In many real-time control systems, the exact response times of all the elements to be controlled may not be available prior to system installation. This unavailability may be due to the sheer number of controlled elements or to their inability to be measured until the total system is assembled and tuned. In such cases, hardware and software accommodations must be made during the specification, design, and implementation phases of the project to handle these timing variables. This “handling” must include the ability to make some changes in the software without requiring radical rewrites or time-consuming modifications of the interface.

Theoretically, since software is more easily changed than hardware, required adaptations to the control system should be made through code changes. If the software was designed using modern software engineering principles, this approach is feasible. Even assembly language coding that is required for time-critical portions of the software can be designed and implemented in a structured manner. Examples of good real-time software implementations for the C and PL/M 86 languages are given in Chapter 8.

Before proceeding to the specifics of interfacing, some fundamental microprocessor-related definitions, characteristics, and economics are provided.

1.2

DEFINITIONS

Strictly speaking, a *microprocessor* is composed of an arithmetic logic unit (ALU), control circuitry, a small amount of scratch pad memory, and, in some cases, timing (clock) circuits. A block diagram of a microprocessor is given in

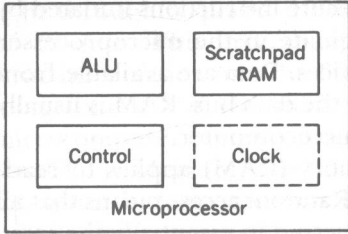


FIGURE 1.1 Microprocessor block diagram.

Figure 1.1. On the other hand, a *microcomputer* is made up of a microprocessor, program memory that is usually a type of read only memory (ROM), read/write random access memory (RAM), and a peripheral interface that provides communication to external devices. If the timing (clock) circuits are not contained in the microprocessor, then they must also be provided for in the microcomputer. A general microcomputer configuration is shown in Figure 1.2.

1.2.1 BUSES

The component elements of the microcomputer are interconnected by wiring paths known as *buses*. There are three types of buses: *address*, *data*, and *control*. The address bus carries binary-coded patterns (addresses) from the microprocessor to the ROM, RAM, and peripheral interface. The addresses identify unique locations that will be the source or destination of information transferred to or from the microprocessor. This information travels over the bidirectional data bus. The information falls into the four general categories of instructions, control words, status indicators, and data. Instructions contained in ROM are executed by the microprocessor, but intelligent peripheral

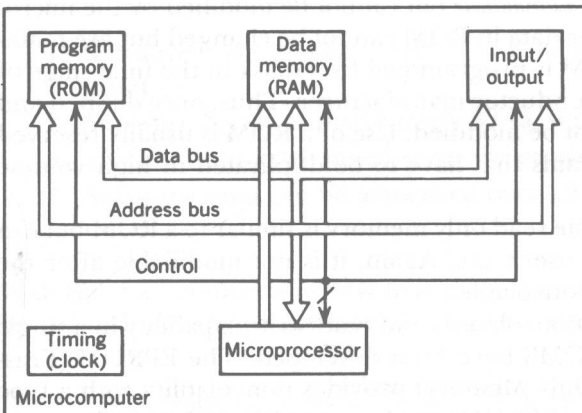


FIGURE 1.2 General microcomputer configuration.