MICROPROCESSOR SYSTEM DESIGN

A Practical Introduction

Michael Spinks

Microprocessor System Design

A practical introduction

Michael J. Spinks

NEWNES

Newnes
An imprint of Butterworth-Heinemann Ltd
Linacre House, Jordan Hill, Oxford OX2 8DP

PART OF REED INTERNATIONAL BOOKS

OXFORD LONDON BOSTON
MUNICH NEW DELHI SINGAPORE SYDNEY
TOKYO TORONTO WELLINGTON

First published 1992

British Library Cataloguing in Publication Data

Spinks, Michael J.
Microprocessor system design: a practical introduction.
I. Title
004.256

ISBN 0 7506 0279 1

Library of Congress Cataloguing in Publication Data

Spinks, Michael J.
Microprocessor system design: a practical introduction/Michael J. Spinks
p. cm.
Includes bibliographical references and index.
ISBN 0 7506 0283 X
1. Microprocessors–Design and construction. 2. Electronic circuit design. I. Title.
TK7895.M5S65 1992
621.39'5–dc20 91-41074
CIP

Printed in Great Britain by
Thomson Litho Ltd, East Kilbride, Scotland
Composition by Scribe Design, Gillingham, Kent

Contents

Preface

This book has been written to introduce the reader to the concepts and techniques which go into the design of electronic circuits, especially microprocessor boards and their peripherals. No previous knowledge of electronics is assumed, and each time a new term or idea is introduced it is clearly explained. In addition, a glossary is provided for quick reference.

The book takes a different approach from that of many electronics books I have read, which take a very academic stand: they are full of unnecessary detail, jargon and often a lot of maths. It is easy to become lost or confused, and it is difficult to know which are the important points – if you understand them at all, that is! It is my experience that in reality design engineers rely on a few relatively simple techniques, and it is by combining such ideas that complex circuits are built up.

In my opinion an electronics book should foster the acquisition and understanding of these key techniques. I have written this text from a practical viewpoint to give the reader the essential knowledge that an engineer uses. I have kept the amount of maths to a minimum and have aimed to explain new concepts in plain English, rather than in terms of other pieces of jargon; this should make the contents accessible to readers of many backgrounds. Almost all the circuits and ideas have been used in real industrial situations, so the reader can study them in the knowledge that they will be of practical use, rather than of merely academic interest.

The book begins with the basic building blocks of electronic systems – digital and analog components – and is followed by the more advanced topics of operational amplifiers and programmable logic (PALs). The reader who masters these topics will be able to design many useful circuits and will have gained a solid grounding in electronics.

However, much of modern electronics is based on the flexibility of microprocessors, and anyone involved in electronics today must have a sound grasp of these devices. Here the 6809 is used to illustrate how a microprocessor works and the relationships between the hardware and software. The use of microprocessors is developed by considering how to expand a microprocessor system and the use of bus-based systems – *the* way to get a small control system working quickly. Some specialized circuits are then investigated – A/Ds, D/As, graphics and phase-locked loops. To conclude, some practical aspects of electronics design are examined.

Thus the book should be of interest to engineers and technicians who are involved in electronics, particularly microprocessor work. It should also provide useful reading for students on electronics engineering courses for HNC, HND and degree qualifications, and for anyone else who wishes to obtain a clear and concise working knowledge of electronics.

M.J.S.

Acknowledgements

I would like to thank the following for their assistance with this book:

Paul Cuthbert and Anthony Winter of Arcom Control Systems Ltd for permission to reproduce some of Arcom's circuit diagrams. I also thank the latter for many interesting discussions about various aspects of electronics.

One of the best ways to improve one's knowledge of electronics is to discuss ideas and circuits with fellow engineers. Amongst others the following have given unwitting tuition: Dave Albiston, Mike Cox, Paul Querelle and Peter Duffett-Smith.

Michael Tooley for his comments on the typescript.

All the staff at Butterworth-Heinemann who have worked on this book.

This page intentionally left blank

1 Digital logic

Logic levels

In digital electronics we are concerned with electrical signals which can have one of two values. These values are most easily referred to as one of the following pairs of names – HI and LO, TRUE and FALSE, or 1 and 0. Since confusion can arise as to whether a 'TRUE' signal is represented by a high or low electrical signal, here we shall use the HI and LO pair, and use them simply as a reflection of whether an electrical signal is in the high or low state. The exact voltage levels corresponding to the HI and LO states depend on the logic family (this book looks at TTL and CMOS) being used.

TTL

TTL (Transistor–Transistor Logic) operates off a +5 V supply and a TTL input will recognize a signal as a logic HI if the input voltage exceeds about 2.4 V. It will recognize a signal as a LO logic level if the input voltage is less than about 0.8 V. If the input voltage is between these figures, say 1.5 V, then the output is undefined (it could be in either the LO or the HI state); such a state will obviously occur momentarily when an input changes state between HI and LO or vice versa. Besides, if an input level is sitting at such a voltage then it means there is something wrong with your circuit, for example you might be trying to drive too many inputs from just one output. The output of a TTL gate is designed to satisfy another's input requirements, so if an output is in the HI state then it is guaranteed to have a voltage of at least 2.4 V (typically 3.4 V), and in the LO state a maximum voltage of 0.4 V (typically 0.2 V).

TTL comes in many varieties and the differences (mostly speed) are looked at later. The type in most common use at present is the 74LS series (LS stands for low-power Schottky) which can operate up to around 25 MHz. The 74 is the start of the part number and indicates the standard commercial series which operates over the temperature range 0 to 70°C, compared with the 54LS military series which can work from –55 to 125°C.

CMOS

The other logic family we shall look at is called CMOS (Complementary Metal Oxide Semiconductor). CMOS uses much less power than TTL (the power consumption increasing with frequency), so it is especially suitable for battery-powered operation. CMOS comes in two main varieties. The older 40xx series operates off a range of power supplies from 3 to 15 V (at 5 V it is nearly compatible with TTL), but is limited to speeds of a few megahertz; the higher the supply voltage the faster it can run. The newer high-speed CMOS families (74HC and 74AC), operating off 2 to 6 V, are pin compatible with the 74 TTL series and should be considered for new designs. For direct interfacing to TTL the 74HCT and 74ACT series are available, and can be used to replace TTL devices in existing circuits to reduce power consumption.

The output of a CMOS gate swings right to the supply rails and since the logic switching point is roughly midway between the supply rails CMOS has higher *noise immunity* than TTL. One disadvantage of CMOS chips is that they are susceptible to damage from static electricity before being inserted into a circuit, so correct handling procedures need to be adopted: do not touch the pins of ICs, work on a conducting surface and wear an earthed wrist-strap.

Simple gates and truth tables

In this section we shall look at simple logic gates and at how *truth tables* can be drawn up to show the relationship between the inputs and outputs of a logic circuit.

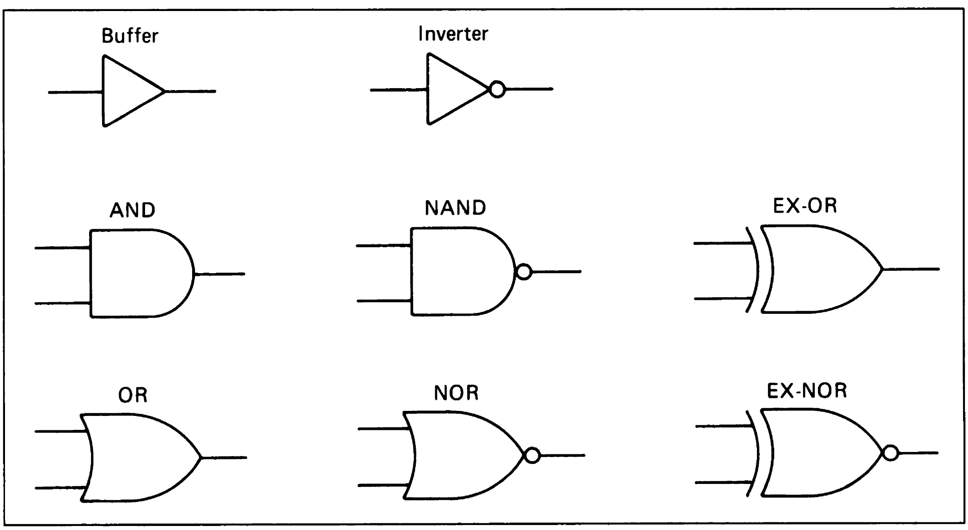

Figure 1.1 *The basic logic gate symbols*

The buffer

The simplest logic building block or gate is the *buffer*. The output of a buffer chip is simply a copy of its input. This is more useful than you might think – the output could have more drive than the input or a different type of driving circuitry, such as tri-state or open-collector, as we shall see later. The symbol for the buffer is shown in Figure 1.1. The logic equation for a buffer is written as Y = A, where Y is the output and A the input; both can have the value HI or LO.

The inverter and truth tables

The other single input gate is known as the *inverter*, see Figure 1.1. Here the output is the inverse, or opposite, of the input. The 74LS04 is an example of an inverter and has six inverters in one 14 pin package. In logic terminology the output is the negation or NOT of the input. The equation for the inverter is Y = /A where the '/' means NOT. This can be written as a truth table as follows.

Input (A)	Output (Y)
LO	HI
HI	LO

A truth table contains all possible inputs of a logic circuit and the corresponding outputs arranged in a table. The name arises from the alternative labelling of the logic levels as TRUE and FALSE. Truth tables can be useful when trying to simplify logic problems, to minimize the number of chips required or for producing a list of inputs and outputs for coding into a programmable logic device (see Chapter 4).

The AND gate

The next simplest logic gates are those which have two inputs and produce one output. Firstly the AND gate (74LS08): here the output is HI if and only if the two inputs (A and B) are both HI. The symbol for an AND gate is shown in Figure 1.1. The logic equation (also known as the Boolean equation) for an AND gate is Y = A & B, or alternatively A.B or A*B, and the truth table is given below.

A	B	Y
LO	LO	LO
LO	HI	LO
HI	LO	LO
HI	HI	HI

The OR gate

Secondly, the OR gate (74LS32; four in a package): the output is HI if either of the two inputs is HI. The symbol for an OR gate is given in Figure 1.1. Its equation is Y = A + B and its truth table is as follows.

A	B	Y
LO	LO	LO
HI	LO	HI
LO	HI	HI
HI	HI	HI

The NAND gate

The NAND gate (74LS00), see Figure 1.1, is the negation of the AND gate and has the equation Y = /(A & B). Its truth table is shown below.

A	B	Y
LO	LO	HI
HI	LO	HI
LO	HI	HI
HI	HI	LO

De Morgan's theorem

The previous truth table is an example of *de Morgan's theorem* which states that /(A & B) = /A + /B, and that /(A + B) = /A & /B. In words: if you negate a logic term, then this is equal to the individual terms negated and with AND substituted for OR and vice versa. We can draw up a truth table to show this formula, as below.

A	/A	B	/B	A & B	/A + /B
LO	HI	LO	HI	LO	HI
LO	HI	HI	LO	LO	HI
HI	LO	LO	HI	LO	HI
HI	LO	HI	LO	HI	LO

In the days before programmable logic (PALs, EPLDs, etc.) de Morgan's theorem was quite important as a means of rewriting logic equations to minimize the number of gates required in a circuit. Nowadays gates are cheap and if you are using programmable logic the computer program which processes your equations implements de Morgan's theorem for you.