BOOK REVIEW:

DIGITAL SYSTEM IMPLEMENTATION

Gerrit A. Blaauw

Prentice Hall, Series in Automatic Computation 1976.
384 p.

In the introduction to this book Prof. G. Blaauw describes the 3 levels of designing a digital system:
Architecture
Implementation and
Realization.

His declared intention for this book is the description of implementation methods seen as a reflexion of the architectural description. And the table of contents seems to promise a conventional book:

Chapter 1 introduces basic notation and facts about computers. The basic elements of computer architecture (operations, data, operators, instructions) are introduced.

Addition (chapter 2), Multiplication (chapter 3) and Division (chapter 4) follow. Different implementations are demonstrated for each topic.

Chapter 5 discusses an often neglected topic: 'Data Path'. It discusses how the elements described in chapter 2 through 4 are interconnnected, various alternatives are listed and described.

In chapter 6 ('Internal Control') various methods of controlling a computer (both via microprogramming and hard-wired logic) are described, followed by chapter 7 ('System Control') discussing the flow of control on the architectural level (instruction fetch and decoding, branching, instructions, interrrupts).

In chapter 8 ('Storage') various methods of organizing data storage are explained, including address translation and physical storage media.

Chapter 9 ('Communications') discusses many aspects of input/output plus communication with peripherals together with communication protocols.

Four appendices ('Summary of APL', 'IBM System/360 and INTEL 8080 Instructions', 'Machine Arithmetic' and 'Switching Algebra') complete the book.

However, soon after chapter 1 it becomes obvious that this book differs from previous ones:

In order to define the architecture of the adder (chapter 2) a diagram is given which is immediately translated into a APL Program (which makes it necessary to interrupt the flow of presentation by a 6 page introduction to APL). From the very general APL definition of addition several variants of adders ('bit adder', 'nand circuit for a bit adder', 'series adder' 'series-parallel adder', 'four bit ripple adder', etc.) are derived. All this variants are described by APL programs!

The same approach is taken in all following chapters. Blaauw even manages to describe both microprogrammed and hard-wired control by APL definitions.

Storage (chapter 8) and I/O (chapter 9) seem to be less amenable to formal description in general and by APL in particular.

What are the outstanding merits of this book?

The listing of different choices for adders, multiplication, division and data-paths is practically complete. Especially laudable is chapter 5 ('data paths') which is usually omitted in standard books, also the two appendices on machine arithmetic and on switching algebra.

For the formal definition of the various implementation elements most features of APL are used. Since all APL-features are profoundly explained, a reader gets an introductory course into APL at the same time.

But there is a second side to the coin:

The application of APL makes it difficult at times to understand some otherwise simple situations, the browsing and isolated reading of sections is very cumbersome. Blaauw himself mentions this dilemma in the introduction by noting that the price for avoiding ambiguity of prose description is either an abundancy of words, a highly stylized text, or a

formal definition with the problems of unfamiliarity of symbols and difficulty of being understood.

However, even stronger criticism must be made:

Throughout the book Blaauw's intimate knowledge of IBM System/360 is felt, in many places to the point of ignoring alternative choices. In chapter 1 the architecture of System/360 is presented without even mentioning the difficult and sometimes almost arbitrary decisions described by Amdahl, Blaauw and Brooks in 'The Architecture of the System /360'

This strong bias persists throughout the book, /360 machine architecture sometimes being taken almost as a dogma. Possible trade-offs and alternative approaches are not described, e.g. 2's complement versus 1's complement notation is missing, semaphores are not mentioned, no discussion of various alternatives of choosing the bit-size of a byte (6, 7, or 8) is found, register versus stack machine is ignored.

In some instances the reader should have some extra knowledge of System/360 to be able to understand the text, especially when PSW's and interrupts are discussed without really explaining the concept.

There are also a few subjects for which the reviewer would have liked to see either more detail and facts or a better description: In the historical introduction Konrad Zuse is not mentioned, references to recent developments in LSI (chapter 1) would not have hurt, an explanation of carry predict adder (chapter 2) the difference between digital and analog storage techiques, paging, non-destructive memory read-out, associative memories and their consequences (chapter 8), channels and I/O interfaces (chapter 9) would have deserved more attention.

Summing up, the book is certainly worthwhile reading, especially for its strong use of formal description methods, for the parallel teaching of implementation and APL and also for bridging the hardware/software gap by showing the formal equivalence between a (software) program (given in APL) and a (hardware) circuit (described by a gate diagram). The exercises following each chapter make it a valuable candidate for course adoption.

G. Chroust