

Preface

Residue number systems (RNS) were invented by the third-century Chinese scholar Sun Tzu—a different Sun Tzu from the author of the famous *Art of War*. In Problem 26 of his *Suan Ching (Calculation Classic)*, our Sun Tzu posed a mathematical riddle:

We have things of which we do not know the number
If we count them by threes, we have two left over
If we count them by fives, we have three left over
If we count them by sevens, we have two left over
How many things are there?

In other words, “What number yields the remainders 2, 3, and 2 when divided by 3, 5, and 7, respectively?.” In modern terminology, 2, 3, and 2 are *residues*, and 3, 5, and 7, are *moduli*. Sun Tzu gave a rule, the *Tai-Yen (Great Generalization)* for the solution of his puzzle. In 1247, another Chinese mathematician, Qin Jiushao, generalized the Great Generalization into what we now call the *Chinese Remainder Theorem*, a mathematical jewel.

In the 1950s, RNS were rediscovered by computer scientists, who sought to put them to use in the implementation of fast arithmetic and fault-tolerant computing. Three properties of RNS make them well suited for these. The first is absence of carry-propagation in addition and multiplication, carry-propagation being the most significant speed-limiting factor in these operations. The second is that because the residue representations carry no weight-information, an error in any digit-position in a given representation does not affect other digit-positions. And the third is that there is no significance-ordering of digits in an RNS representation, which means that faulty digit-positions may be discarded with no effect other than a

reduction in dynamic range.

The new interest in RNS was not long-lived, for three main reasons: One, a complete arithmetic unit should be capable of at least addition, multiplication, division, square-root, and comparisons, but implementing the last three in RNS is not easy; two, computer technology became more reliable; and, three, converting from RNS notation to conventional notation, for “human consumption”, is difficult. Nevertheless, in recent years there has been renewed interest in RNS. There are several reasons for this new interest, including the following. A great deal of computing now takes place in embedded processors, such as those found in mobile devices, and for these high speed and low-power consumption are critical; the absence of carry-propagation facilitates the realization of high-speed, low-power arithmetic. Also, computer chips are now getting to be so dense that full testing will no longer be possible; so fault-tolerance and the general area of computational integrity have again become more important. Lastly, there has been progress in the implementation of the difficult arithmetic operations. True, that progress has not been of an order that would justify a deluge of letters home; but progress is progress, and the proper attitude should be gratitude for whatever we can get. In any case, RNS is extremely good for many applications—such as digital signal processing, communications engineering, computer security (cryptography), image processing, speech processing, and transforms—in which the critical arithmetic operations are addition and multiplication.

This book is targeted at advanced university students, researchers, and professional engineers interested in RNS and their applications. Both theoretical and practical aspects of RNS are discussed. Other than a basic knowledge of digital logic and some of that fuzzy quality known as “mathematical maturity”, no other background is assumed of the reader; whatever is required of conventional arithmetic has been included. It has not been our intention to give the last word on RNS—he or she who must know everything will find satisfaction in the many published papers that are referred to in the book—but, taken as a summary, the book takes the reader all the way from square-one right to the state-of-the-art.

Chapter 1 is an introduction to the basic concepts of RNS, arithmetic in RNS, and applications of RNS. Other conventional and unconventional number systems are also discussed, as, ultimately, it is these that form the basis of implementations of residue arithmetic.

Chapter 2 covers the mathematical fundamentals on which RNS are founded. The main subjects are the congruence relationship (which relates

numbers in modular arithmetic), the basic representation of numbers, an algebra of residues, the Chinese Remainder Theorem (very briefly), complex-number representation, and, also very briefly, the detection and correction of errors. The *Core Function*, a useful tool in dealing with the problematic (i.e. hard-to-implement) operations in residue number systems, is also discussed.

Forward conversion, the process of converting from conventional representations to RNS ones, is the main subject of Chapter 3. Algorithms and hardware architectures are given. The chapter is divided into two parts: one for arbitrary moduli-sets and one for “special” moduli-sets—those of the form $\{2^n - 1, 2^n, 2^n + 1\}$ —and extensions of these. The latter moduli-sets are of interest because with them implementations of almost all operations are relatively quite straightforward.

Chapters 4, 5, and 6 deal with addition/subtraction, multiplication, and division respectively. (Chapter 6 also includes a discussion of other operations that are closely related to division.) Each chapter has two main parts: one on the conventional version of the operation and one that shows how the conventional algorithms and hardware architectures can be modified to implement the RNS operation. The RNS part is again divided into one part for arbitrary moduli-sets and one part for special moduli-sets.

Reverse conversion, the process of converting back from RNS representations into conventional notations, is covered in Chapter 7. (The general structure of that chapter is similar to that of Chapter 3.) The main topics are the Chinese Remainder Theorem, *Mixed-Radix Conversion* (in which one views a residue number system as corresponding to a conventional system with multiple bases), and the Core Function.

The last chapter is a very brief introduction to some applications of RNS—digital signal processing, fault-tolerant computing, and communications. Lack of space has prevented us from giving an in-depth treatment of these topics. As there are more things in heaven and earth than are dreamt of in our philosophy, we leave it to the imaginative reader to dream up some more applications.

Acknowledgements

In all things, we thank and praise The Almighty. The love and joy I get from my family—Anne, Miles, and Micah—sustain me always. My contribution to this book has been made during my employment at Yonsei University, in a position funded by the Korean government’s Institute for Information Technology Assessment. I am indebted to both organizations.

Amos Omondi

I gratefully acknowledge the support rendered by Dr. Omondi, during the course of writing this book for his many insightful comments and suggestions. I also owe my gratitude to my colleagues at the School of Computer Engineering, Nanyang Technological University, Singapore. I am greatly indebted to my wife Manchala, our sons Josh and Brian and to my mother and my family for their love, endurance and patience. Finally, I would like thank my Lord Jesus Christ for His enduring love.

Benjamin Premkumar