A Variable-Radix Systolic Montgomery Multiplier

K.H. Tsoi, O.Y.H. Cheung and P.H.W. Leong
{khtsoi,yhcheung,phwl}@cse.cuhk.edu.hk
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, NT Hong Kong

Abstract

A variable radix systolic Montgomery multiplier, suitable for use in implementing the RSA public key cryptosystem is presented. Measurements of the effect of increasing radix on area and performance are given so that a designer can determine the optimal radix and its associated performance given the area requirements of the application. On a Xilinx XCV1000E-6 FPGA, a 1024 bit modular multiplication can be performed in $8.4\mu s$.

1 Introduction

RSA is the most widely used public key cryptosystem, and is based on modular exponentiation, i.e. the computation of $M^e \ mod \ N$ where *M* and *N* are numbers which are of the order of $n = 1024$ bits in length. Most implementations use Montgomery multiplication [1] to efficiently compute $a \times b \ mod \ N$. The Montgomery algorithm converts the input numbers to a special residual system in which computations are made modulo 2^n instead of modulo N, $N < 2^n$. Kornerup [2] proposed a systolic array for the computation of Montgomery multiplication and also extended it for higher radices [3]. Previous hardware implementations of Montgomery multipliers would select a small radix (usually either 2 or 4), and work with that radix. In this work, we present a module generator for a Montgomery multiplier in which the radix can be arbitrarily chosen. Such a module generator can exploit the reconfigurable nature of field programmable gate array (FPGA) devices and be used to generate the design which maximally utilizes the given device.

Montgomery multiplication can be performed using the below algorithm which computes $A \times B \ mod \ N$, where n is the number of bits, a_i is the ith digit of A, $R = 2^n$, and N' and R' are chosen such that $RR' = 1 \ mod \ N$ and $NN' - RR' = 1$

```
for i := 0 to n/log2k
  q := (S*N') mod k;          /* step1 */
  S := (S + qN) div k + ai B; /* step2 */
end for
```

Step 2 above can be modified to be

$$S = \lfloor \frac{S}{k} \rfloor + q \lfloor \frac{N}{k} \rfloor + a_i B + \lfloor \frac{(S \ mod \ k) + q(N \ mod \ k)}{k} \rfloor \quad (1)$$

which results in a simpler hardware design since it can be shown that *S* and *qN* always sum up to a multiple of k (i.e. the $log_2(k)$ least significant bits are always 0), and the last term (referred to as '*f*' in the rest of this paper) will be in the range $[0 \ldots k-1]$.

2 Design

The block diagram of an n-bit radix-k Montgomery multiplier is shown in Fig. 1. The s-cell computes the first three terms of Equation 1. The r-cell is identical, except that it has an additional input which is the last term of Equation 1 (generated by the f-cell). In Fig. 2, the internal structure of the r and s cells are shown. Let $j = log_2 k$. In each clock cycle, each cell computes *j* bits of *S*; passes its result to the cell on the right hand side; and passes the inputs, *A* and *Q*, to the cell on the left [3].

For an n-bit Montgomery multiplication, an $(n+2)$-bit multiplier is required to ensure that $0 \leq S \leq 2N$ at all times. Correspondingly, $\lceil \frac{n+2}{2k} \rceil$ systolic cells are required in our design. The number of cycles required for a radix-k, n-bit modular multiplication is $2(\lceil \frac{n+2}{k} \rceil + 1)$.

3 Results

The design was coded in VHDL and successfully tested on an Annapolis Wildstar FPGA board using a Xilinx Virtex XCV1000-6 FPGA. Figure 3(a) shows the number of

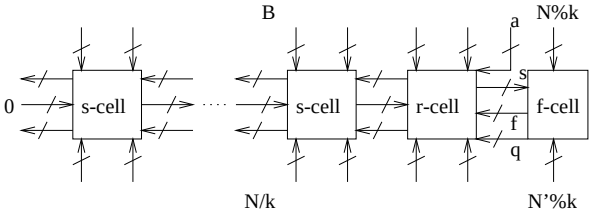

Figure 1: Top level overview of multiplier. All signals in this figure are $log_2(k)$-bits wide.

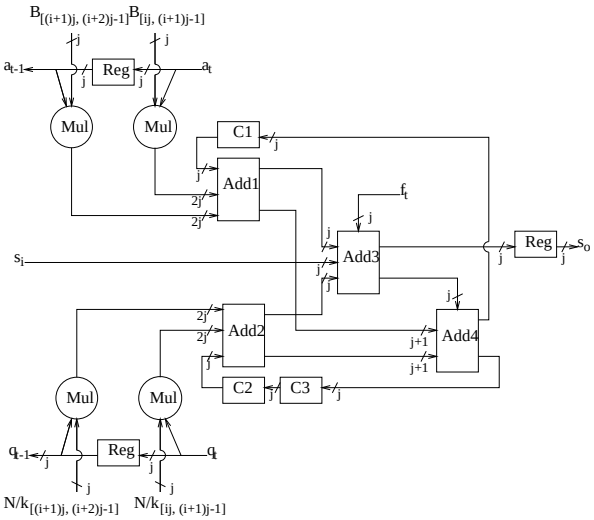

Figure 2: Circuit diagram showing the s-cell and r-cell which are identical except that the s-cell does not have the f_t input.

Virtex slices required for both n=512 and n=1024 as a function of the radix. The decrease in area with increasing radix at radix-2^3 occurs because although the area of each cell inceases slightly, the total number of cells required is smaller.

Figure 3(b) shows the time required for a multiplication with increasing radix. Note that the execution time does not improve linearly with k because the maximum clock frequency of the design (as reported by the Xilinx timing tools) decreases with increasing k since more levels of logic and routing are required.

From Figure 3(b), it can be seen that higher radices lead to better performance. Thus the highest performance that can be achieved for $n = 1024$ on a Xilinx XCV1000E device corresponds to the highest radix which lies under the horizontal line in Figure 3(a) which represents the number of slices in the XCV1000E, namely a radix-2^6 implementation ($8.4\mu s$).

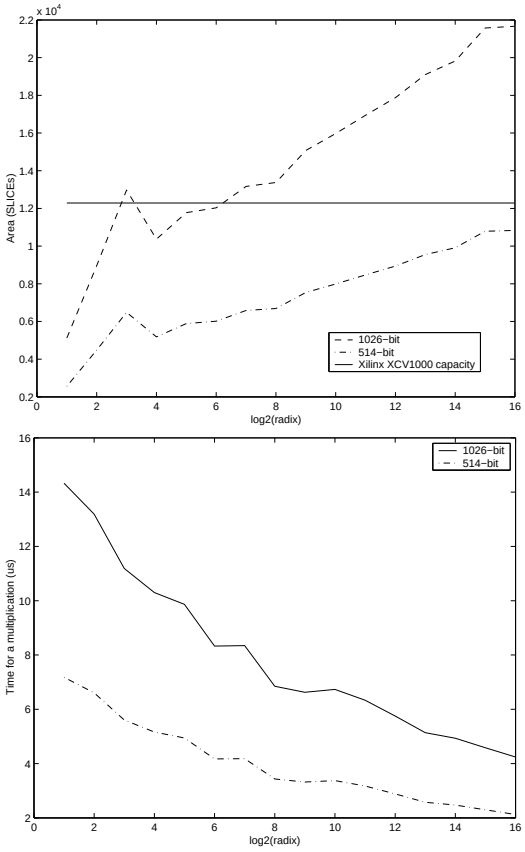

Figure 3: (a) Area in slices as a function of radix and (b) Multiplication time as a function of radix.

4 Conclusion

An FPGA based module generator for a systolic Montgomery multiplier was presented. This generator gives designers the flexibility to use multipliers of arbitrary radix, depending on their area and performance considerations.

References

[1] P. Montgomery, "Modular multiplication without trial division," in *Mathematics of Computation, vol. 44*, pp. 519–521, Apr 1985.

[2] P. Kornerup, "A systolic, linear-array multiplier for a class of right-shift algorithms," in *IEEE Transactions on computers, vol. 43*, pp. 892–898, Aug 1994.

[3] P. Kornerup, "High-radix modular multiplication for cryptosystems," in *Computer Arithmetic, 1993. Proceedings., 11th Symposium on*, pp. 277–283, Jul 1993.