High-Performance Pipelined Architecture of Elliptic Curve Scalar Multiplication Over GF(2^m)

Abstract:

This paper proposes an efficient pipelined architecture of elliptic curve scalar multiplication (ECSM) over GF(2^m). The architecture uses a bit-parallel finite field (FF) multiplier accumulator (MAC) based on the Karatsuba–Ofman algorithm. The Montgomery ladder algorithm is modified for better sharing of execution paths. The data path in the architecture is well designed, so that the critical path contains few extra logic primitives apart from the FF MAC. In order to find the optimal number of pipeline stages, scheduling schemes with different pipeline stages are proposed and the ideal placement of pipeline registers is thoroughly analyzed. We implement ECSM over the five binary fields recommended by the National Institute of Standard and Technology on Xilinx Virtex-4 and Virtex-5 field-programmable gate arrays. The three-stage pipelined architecture is shown to have the best performance, which achieves a scalar multiplication over GF(2^{163}) in 6.1μs using 7354 Slices on Virtex-4. Using Virtex-5, the scalar multiplication form=163, 233, 283, 409, and 571 can be achieved in 4.6, 7.9, 10.9, 19.4, and 36.5 μs, respectively, which are faster than previous results. The proposed architecture of this paper analysis the logic size, area and power consumption using Xilinx 14.2.

Enhancement of the project:

Existing System:

Elliptic curve scalar multiplication (ECSM) is the key operation, which dominates the performance of ECC cryptosystem. Various architectures have been proposed to speed up ECSM. Most of them explore pipeline and parallelism to improve the working frequency and to
reduce the required number of clock cycles in ECSM. Leong and Leung developed a microcoded elliptic curve processor, supporting ECSM over GF(2m) for arbitrary m. Sakiyama et al. proposed a superscalar coprocessor and accelerated ECSM by exploiting instruction-level parallelism (ILP) dynamically. A pipelined application specific instruction set processor for ECC was proposed, which performed ECSM over GF(2163) in 19.55 μs on Xilinx XC4VLX200. Designs implemented high-speed scalar multiplication over a special class of curves, such as Koblitz curves, binary Edwards curves, and Hessian curves. In this paper, we focus on optimizing ECSM over generic curves in GF(2m).

Some designs duplicate arithmetic blocks to maximize the parallelism in ECSM. For GF(2163), Kim et al. used three Gaussian normal basis multipliers to achieve ECSM in 10 μs on Xilinx XC4VLX80. Zhang et al. developed three finite-field (FF) cores and a main controller to achieve ECSM in 7.7 μs on Xilinx XC4VLX80. The best design in performed ECSM in 5.5 μs on Xilinx Virtex-5 using three digit-serial FF multipliers and one FF divider. Despite high speed, these designs require massive logic resources, and thus, they are not practical for FPGA implementation. Considering the tradeoff between area and speed, many designs use word-serial or digit-serial FF multipliers to implement ECSM. These designs usually require a large number of clock cycles for a scalar multiplication. Ansari and Hasan proposed an efficient scheme, which kept the pseudopipelined word-serial FF multiplier working without idle cycles. A scalar multiplication over GF(2163) costs 4050 clock cycles and 21 μs on Xilinx XC4VLX200. FF multipliers with different word sizes (w) were developed, and the best design with w = 55 performed ECSM over GF(2163) in 2751 clock cycles and 9.6 μs on Xilinx XC4VLX200.

Disadvantages:

- Area coverage is high
- Performance speed is slow

Proposed System:
Data Dependence Analysis of ECSM

The modified Montgomery ladder scalar multiplication totally takes \(m(6M + 5S + 3A) + (11M + 5A + I)\) operations, where \(M, S, A,\) and \(I\) denote multiplication, square, addition, and inversion in \(GF(2^m)\), respectively, and \(m\) is the dimension of the binary field \(GF(2^m)\). The original Montgomery ladder scalar multiplication requires \((m - 1)(6M + 5S + 3A) + (10M + 7A + 3S + I)\) operations. The increased operations are due to the merged initialization and the modified postprocess for better sharing the data path with the main loop. As square and addition are much cheaper than multiplication, and inversion occurs only once, we can see that optimizing operations in the main loop, especially the FF multiplication, is the key to realize high-performance ECSM.

Fig. 1. Data dependence graph of (a) point addition and (b) point doubling in the Montgomery ladder algorithm.
Each iteration in the main loop performs point addition and point doubling, which take $6M + 5S + 3A$ together. The data dependence of point addition and doubling in the Montgomery ladder algorithm is shown in Fig. 1. The critical path lies in calculating the $X$-coordinate of point addition, which takes $2M + 1S + 2A$, as is shaded in Fig. 1. Thus, it may use at most three FF multipliers to achieve maximum parallelism in scalar multiplication.

PROPOSED ARCHITECTURE OF ELLIPTIC CURVE SCALAR MULTIPLICATION:

we propose the high-performance architecture based on the improved Montgomery ladder scalar multiplication algorithm, as shown in Fig. 2.

Fig. 2. Proposed architecture of ECSM.

The proposed ECSM architecture consists of one bit-parallel FF MAC, one FF squarer, a register bank, a finite-state machine, and a $6 \times 18$ control ROM. The FF MAC is implemented using the Karatsuba–Ofman algorithm, and is well pipelined. The $n$-stage pipelined FF MAC takes $n$ clock cycles to finish one multiplication. The FF squarer is not pipelined, and one clock cycle is required to finish one square. The inputs to FF MAC, A, B, and C, and the input to FF squarer, S,
are all registered. Another four registers T1, T2, T3, and T4 are used in the data path for data caching.

Fig. 3. Data path of ECSM using a three-stage pipelined FF MAC.
The data path of ECSM using a three-stage pipelined FF MAC is given for example in Fig. 6. The terms X1, X2, Z1, and Z2 are not presented, because they are the intermediate results of the FF MAC or FF Squarer. The bold dashed line in Fig. 6 shows the critical path of the three-stage pipelined architecture, which consists of a pipelined FF MAC, an addition (XOR), and a 4:1 MUX. Data paths with other pipeline stages are similar to Fig. 6 except for different data connections. Control signals stored in the control ROM are also different. But, the critical path delay remains unchanged.

**Advantages:**

- Area reduction
- Speed is increased

**Software implementation:**

- Modelsim
- Xilinx ISE