ELectronics EXpress

LETTER

IEICE Electronics Express, Vol.10, No.18, 1–6

An improved quad Itoh-Tsujii algorithm for FPGAs

V. R. Venkatasubramani[a)], **N. Murali, and S. Rajaram**

Department of ECE, Thiagarajar college of Engineering, Madurai 625–015, India

a) *venthiru@tce.edu*

Abstract: In this paper, we propose an improved quad Itoh-Tsujii algorithm to compute multiplicative inverse efficiently on Field-programmable gate-arrays (FPGA) platforms for binary fields generated by irreducible trinomials. Efficiency is obtained by eliminating the precomputation steps required in conventional quad-ITA (QITA) scheme. Experimental results show that the proposed architecture improves the performance on FPGAs compared to existing techniques.

Keywords: Field-programmable gate-array (FPGA)-based designs, quad Itoh-Tsujii algorithm (QITA), Lookup Table (LUT)

Classification: Integrated circuits

References

[1] T. Itoh and S. Tsujii: Inf. Comput. **78** [3] (1988) 171.

[2] F. Rodríguez-Henríquez, N. A. Saqib, A. Diaz-Perez and C. K. Koc: *Cryptographic Algorithms on Reconfigurable Hardware* (Springer-Verlag, New York, 2006) 176.

[3] J. Guajardo and C. Paar: Des. Codes Cryptography **25** [2] (2002) 207.

[4] F. Rodríguez-Henríquez, N. A. Saqib and N. Cruz-Cortés: Proc. Int. Conf. Inf. Technol. Coding Comput. (ITCC'05) **1** (2005) 574.

[5] F. Rodríguez-Henríquez, G. Morales-Luna, N. A. Saqib and N. CruzCortés: Des. Codes Cryptography **45** [1] (2007) 19.

[6] C. Rebeiro, S. S. Roy and D. Mukhopadhyay: IEEE Trans. Very Large Scale Integr. (VLSI) Syst. **19** [8] (2011) 1508.

[7] D. E. Knuth: *The Art of Computer Programming Volume 2* (Addison-Wesley, Boston, 1981) 2nd ed. 444.

[8] C. Rebeiro and D. Mukhopadhyay: Proc. Int. Conf. Inf. VLSI Des (VLSID). (2008) 706.

[9] National Institute of Standards and Technology, Digital Signature Standards (DSS): FIPS 186-3 (2009) http://csrc.nist.gov/encryption

1 Introduction

The Itoh-Tsujii algorithm (ITA) [1] forms an integral and crucial component of elliptic curve cryptography. This algorithm can be used for efficient computation of multiplicative inverse [2]. The ITA was initially proposed to find the multiplicative inverse for normal basis representation

DOI: 10.1587/elex.10.20130612
Received August 02, 2013
Accepted August 19, 2013
Publicized September 02, 2013
Copyedited September 25, 2013

of elements in $GF(2^m)$. Later on, several works [3, 4, 5, 6] has been proposed to improve the original algorithm to make it feasible for polynomial basis representations. In [3], the ITA was generalized to compute multiplicative inverse efficiently by reducing clock cycles at the cost of increased hardware. In [4], efficient addition chains were used to compute the inversion of an element in $GF(2^{193})$. In [5], a parallel implementation of ITA takes 20 clock cycles to compute inverse of an element in $GF(2^{193})$. In [6], QITA was proposed for better utilization of FPGA resources with shorter addition chain to compute inverse of an element. In this express, we propose an improved Quad Itoh-Tsujii inverse algorithm for efficient implementation on FPGA platforms for binary fields generated by irreducible trinomials. Implementation of the proposed scheme demonstrates that the proposed QITA offer significant saving in execution delay against that in [6] and achieves better performance.

2 Quad Itoh-Tsujii algorithm

The ITA is based on *Fermat's little theorem*, by which the inverse of an element $a \in GF(2^m)$ is computed by $a^{-1} = a^{2^m-2}$. The m number of squarings hampers the performance of ITA. However, ITA efficiently uses addition chains to reduce the number of multiplications required. Exponentiation in the ITA is usually performed using squarer circuits [2]. In QITA [6], 2^2 circuits, also known as a quad circuit, are used for exponentiation in fields with irreducible trinomials. A quad circuit raises the input by a power of four. In FPGAs with 4 or 6 input LUTs, QITA offer the best LUT utilization and uses shorter addition chain. However, the overhead of the QITA is the need to precompute a^3. This normally takes two clock cycles if the multiplier used is purely a combinational circuit and produces the result in one clock cycle. Also there is a need to store this precomputed a^3 in intermediate register bank of m-bit size for use in further steps. Generally m is large in cryptographic applications. Minimizing the number of registers is important to realize a compact QITA architecture on hardware.

3 The proposed quad-ITA algorithm

The scheme for computing the multiplicative inverse for an odd m (which is generally the case) using the proposed QITA is shown in Algorithm 1. In the algorithm $\beta_k(a) = a^{(4^k-1)/3}$. For the input $a \in GF(2^m)$ and Brauer addition chain $u = \{1, 2, ...(m-1)/2\}$, the following algorithm computes $a^{-1} = \left[\beta_{(m-1)/2}(a)\right]^{2(2^n-1)}$.

Algorithm 1: Proposed Quad –ITA Algorithm: $(a,\ u,\ a^{-1})$

$l' = length\,(u)$
$\beta_{u_1}(a) = a$
$for\ each\ u_i\ \in\ (2 \le i \le l')\ do$
$\quad p = u_{i-1}$
$\quad q = u_i - u_{i-1}$
$\quad \beta_{u_i} = \beta_q * \beta_p^{4^q}$
End

DOI: 10.1587/elex.10.20130612
Received August 02, 2013
Accepted August 19, 2013
Publicized September 02, 2013
Copyedited September 25, 2013

$$return\ a^{-1} = \left[\beta_{l'}(a)\right]^{2(2^n-1)}$$

The merit of the proposed QITA is that $\beta_1(a) = a$ can be taken directly into the iteration steps of the algorithm and therefore it is not required to precompute a^3. This saves two clock cycles at this initial stage of the algorithm. However the final step takes two clock cycles thereby saving an overall one clock cycle compared to design in [6]. It is also not required to store the precomputed a^3 in intermediate register bank (generally of m-bit size). Therefore there is no need for intermediate register bank in the proposed design if Brauer addition chain [1, 7] is used. As we work with large numbers, the register file is the most critical component in the architecture. As the registers occupy more percentage of the gate area in a conventional architecture, reducing the number of the registers are very effective to minimize the total gate area. Again, note that reducing even one register decreases the total gate area considerably. Also the access to the register bank is not required. The proposed QITA can also be extended to use any higher power 2^n circuits based on [4]. Even for higher power circuits, precomputation of a^{2^n-1} is not needed. However in the final step, one need to calculate $a^{-1} = \left[\beta_{(m-1)/2}(a)\right]^{2(2^n-1)}$ that consists of Quads and squarings. At this step it is possible to perform Quad and multiplication operations in parallel to save clock cycles.

4 The proposed quad-ITA architecture

The proposed Quad-ITA architecture for a combinational multiplier is shown in Fig. 1. It generates the multiplicative inverse of the input $a \in GF(2^{233})$ by computing the steps in Table I.

The multiplier used is the hybrid Karatsuba algorithm [8] and quadblock consists of eight cascaded quad circuits for best performance [6].

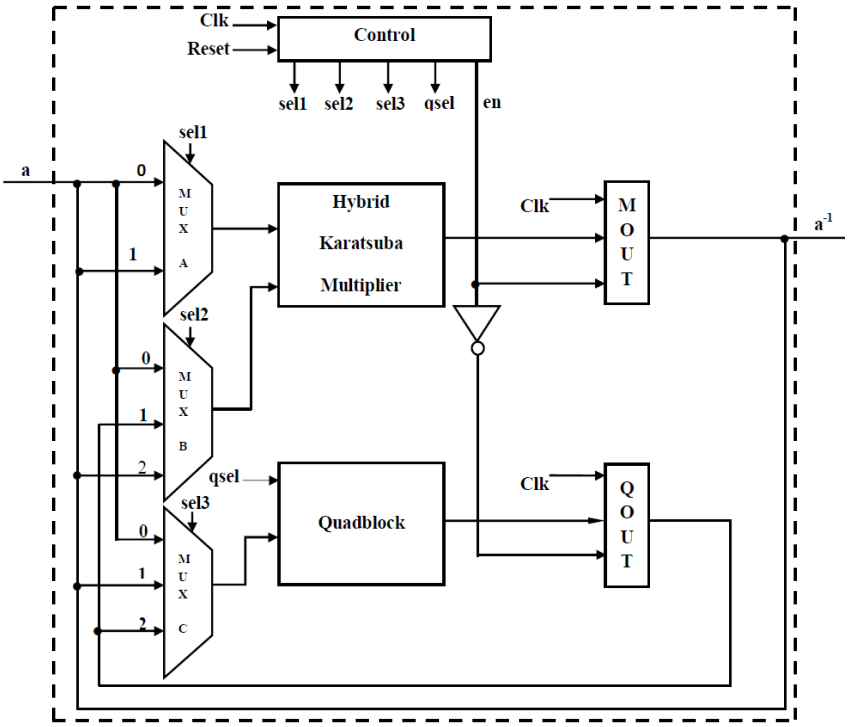

Fig. 1. Proposed Quad-ITA Architecture for GF (2^{233})

DOI: 10.1587/elex.10.20130612
Received August 02, 2013
Accepted August 19, 2013
Publicized September 02, 2013
Copyedited September 25, 2013

Table I. Inverse of $a \in GF(2^{233})$ using proposed QITA with control word

	$\beta_{u_i}(a)$	$\beta_{u_j+u_k}(a)$	Exponentiation	Clk	sel1	sel2	sel3	qsel	en
	$\beta_1(a)$	--	a	--	--	--	--	--	--
1	$\beta_2(a)$	$\beta_{1+1}(a)$	$(\beta_1)^{4^1}\beta_1 = a^{(4^2-1)/3}$	1 2	-- 0	-- 1	2 --	1 --	0 1
2	$\beta_3(a)$	$\beta_{2+1}(a)$	$(\beta_2)^{4^1}\beta_2 = a^{(4^3-1)/3}$	3 4	-- 0	-- 1	0 --	1 --	0 1
3	$\beta_6(a)$	$\beta_{3+3}(a)$	$(\beta_3)^{4^3}\beta_3 = a^{(4^6-1)/3}$	5 6	-- 1	-- 1	0 --	3 --	0 1
4	$\beta_7(a)$	$\beta_{6+1}(a)$	$(\beta_6)^{4^1}\beta_1 = a^{(4^7-1)/3}$	7 8	-- 0	-- 1	0 --	1 --	0 1
5	$\beta_{14}(a)$	$\beta_{7+7}(a)$	$(\beta_7)^{4^7}\beta_7 = a^{(4^{14}-1)/3}$	9 10	-- 1	-- 1	0 --	7 --	0 1
6	$\beta_{28}(a)$	$\beta_{14+14}(a)$	$(\beta_{14})^{4^{14}}\beta_{14} = a^{(4^{28}-1)/3}$	11 12 13	-- -- 1	-- -- 1	0 1 --	8 6 --	0 0 1
7	$\beta_{29}(a)$	$\beta_{28+1}(a)$	$(\beta_{28})^{4^1}\beta_1 = a^{(4^{29}-1)/3}$	14 15	-- 0	-- 1	0 --	1 --	0 1
8	$\beta_{58}(a)$	$\beta_{29+29}(a)$	$(\beta_{29})^{4^{29}}\beta_{29} = a^{(4^{58}-1)/3}$	16 17 18 19 20	-- -- -- -- 1	-- -- -- -- 1	0 1 1 1 --	8 8 8 5 --	0 0 0 0 1
9	$\beta_{116}(a)$	$\beta_{58+58}(a)$	$(\beta_{58})^{4^{58}}\beta_{58} = a^{(4^{116}-1)/3}$	21 22 23 24 25 26 27 28 29	-- -- -- -- -- -- -- -- 1	-- -- -- -- -- -- -- -- 1	0 1 1 1 1 1 1 1 --	8 8 8 8 8 8 8 2 --	0 0 0 0 0 0 0 0 1
10	a^{-1}	--	$(\beta_{116})^4(\beta_{116})^2 = a^{2^{232}-1}$	30 31	1 1	2 1	1 --	1 --	0 1

Here the quadblock and multiplier delay are consider to be about the same. The input qin fed to the quadblock generates qin^4, qin^{4^2}..... qin^{4^8}. The select lines $qsel$ of the multiplexer in the quadblock determines which of the eight powers gets passed on to the output. The MOUT and QOUT buffers store the output of the multiplier and the quadblock, respectively. At every clock cycle, either the MOUT or QOUT is considered depending on the signal en, except for the final step. The controller is a finite state machine whose design is based on the Brauer addition chain and the number of cascaded quad circuits in the quadblock. At every clock cycle, control signals are generated for the multiplexer selection lines and enable to the buffers. Table I shows the steps to compute Inverse of $a \in GF(2^{233})$ using proposed QITA with generated control signals. The element $\beta_{u_1}(a) = a$ is taken directly into the iteration steps of the algorithm. Note that precomputation of a^3 and subsequent storage in register bank is not needed in our architecture.

The first step in the computation of a^{-1} is the determination of $(\beta_{u_2}) = a^5$ that takes two clock cycles. The first clock uses a as the input to the quadblock to compute $(\beta_{u_1})^{4^1}$. In the next clock, this is multiplied with a to produce the required output. In general, computing any step

DOI: 10.1587/elex.10.20130612
Received August 02, 2013
Accepted August 19, 2013
Publicized September 02, 2013
Copyedited September 25, 2013

$\beta_{u_i}(a) = \beta_{u_j+u_k}(a)$ takes $\left\lceil \frac{u_j}{8} \right\rceil + 1$ clock cycles. Out of this, one clock cycle is used by the multipliers and quadblock requires $\left\lceil \frac{u_j}{8} \right\rceil$ clock cycles. The final step takes two clock cycles. Both multiplication and single power quad operations are done in the first clock cycle to compute $(\beta_{116})^4$. In the second clock, this is multiplied with $(\beta_{116})^2$ to produce the required a^{-1} output. At the end of each step, the result is present in MOUT. Note that removal of precomputing a^3 saved two clock cycles. However, computation of final step in Algorithm 1 requires one additional clock cycle compared to the design in [6]. Overall one clock cycle is saved in our design.

5 Experimental results

In order to compare the two algorithms, the results were taken on Virtex 4 (contains four-input LUTs) and Virtex 5 (contains six-input LUTs) FPGA platforms using synthesis tool of Xilinx ISE 11.1 design software. The Table II compares the performance on the binary field with irreducible trinomials with specified NIST digital signature standard [9]. The table shows a significant reduction in LUTs and computation time T = (Delay × Clock cycles) resulting in increase of overall performance P = 1/(No. of LUTs × T). Generally, trade-off exists between number of LUTs and delay. Also, about m number of slice registers S gets reduced in the proposed scheme. This is significant because m is a large number in cryptographic applications.

Table II. Comparison Results on FPGA Platforms over NIST Binary Fields with Irreducible Trinomials

Field	Algorithm	S	LUTs	Delay (ns)	Clock cycle	T (ns)	P
VIRTEX-4 DEVICE: XC4VFX140							
GF(2^{233})	QITA	731	26122	10.3	30	309	123.9
GF(2^{233})	Proposed QITA	466	24713	10.06	29	290	139.5
GF(2^{409})	QITA	1281	60644	15.8	32	505.6	32.6
GF(2^{409})	Proposed QITA	818	57539	16.27	31	502.2	34.6
VIRTEX-5 DEVICE: XC5VLX220							
GF(2^{233})	QITA	704	20950	7.79	30	233.7	204.2
GF(2^{233})	Proposed QITA	481	18151	7.44	29	215.7	255.3
GF(2^{409})	QITA	1233	44948	9.2	35	322	69.1
GF(2^{409})	Proposed QITA	827	44740	8.61	34	292.4	76.4

Table III compares the proposed work with [4, 5, 6] for the same FPGA platform, finite field and addition chain. The proposed implementation improves resource utilization and computation time comparatively. Our design requires no BRAMs and only one global clock (GCLK). This contributes to the reduction of area and power consumption. However, our design uses distributed RAM that needs more slices.

Table IV compares the proposed work with [6] in terms of power consumption using Xilinx Power Analysis (XPA) tool. The result shows that our design has better power consumption comparatively. The proposed design saves 16 mW of total power.

DOI: 10.1587/elex.10.20130612
Received August 02, 2013
Accepted August 19, 2013
Publicized September 02, 2013
Copyedited September 25, 2013

Table III. Comparison for Inversion in GF (2^{193}) on XCV3200EFG1156

Implementation	Resources Utilized (Slices, BRAMs, GCLK)	Freq in MHz (f)	Clock cycles (c)	Time in μs (c/f)	P
Sequential[4]	10065, 12, 3	21.2	28	1.32	75.2
Parallel[5]	11081, 12, 2	21.2	20	0.94	95.7
Quad-ITA[6]	10420, 0, 2	35	21	0.60	160
Proposed QITA	10190, 0, 1	37	20	0.54	181

Table IV. Xilinx Power Analysis Comparison Result on XC4VFX140

Field	Algorithm	Total Power (mW)
GF(2^{233})	QITA [6]	1458
GF(2^{233})	Proposed QITA	1442

6 Conclusions

This paper proposes an improvement in Quad Itoh-Tsujii inverse algorithm for an efficient implementation on FPGA platforms for fields generated by irreducible trinomials. It requires no register bank and less clock cycle. The implementation of the proposed Quad-ITA in FPGA platform results in advantages such as superior computation time, less register storage, reduced power consumption and better performance over existing techniques. The design is scalable and is suitable for use in resource constrained cryptographic applications.

DOI: 10.1587/elex.10.20130612
Received August 02, 2013
Accepted August 19, 2013
Publicized September 02, 2013
Copyedited September 25, 2013