On Orders of Optimal Normal Basis Generators

Shuhong Gao
Department of Computer Science
University of Toronto
Toronto, Ontario, M5S 1A4, Canada
E-mail: sgao@cs.toronto.edu

Scott A. Vanstone
Department of Combinatorics and Optimization
University of Waterloo
Waterloo, Ontario, N2L 3G1, Canada
E-mail: savanstone@math.uwaterloo.ca

April 6, 1994

Abstract. In this paper we give some computational results on the multiplicative orders of optimal normal basis generators in F_{2^n} over F_2 for $n \leq 1200$ whenever the complete factorization of $2^n - 1$ is known. Our results show that a subclass of optimal normal basis generators always have very high multiplicative orders and are very often primitive. For a given optimal normal basis generator α in F_{2^n} and an arbitrary integer e, we show that α^e can be computed in $O(n \cdot v(e))$ bit operations, where $v(e)$ is the number of 1's in the binary representation of e.

For a prime power q and a positive integer n, let F_q and F_{q^n} be the finite fields of q and q^n elements, respectively. A normal basis N for F_{q^n} over F_q is a basis of the form $(\alpha, \alpha^q, \ldots, \alpha^{q^{n-1}})$ where $\alpha \in F_{q^n}$. In such case α is said to be a normal element or normal basis generator. The complexity of N, denoted by c_N, is defined to be the number of nonzero entries t_{ij} in the n expressions

$$\alpha \cdot \alpha^{q^i} = \sum_{j=0}^{n-1} t_{ij} \alpha^{q^j}, \quad 0 \leq i \leq n-1,$$

where $t_{ij} \in F_q$. It is easy to prove that $c_N \geq 2n - 1$. If $c_N = 2n - 1$, then N is called an optimal normal basis. Mullin, Onyszchuk, Vanstone and Wilson [11] constructed the following two families of optimal normal bases, which are essentially all the optimal normal bases in finite fields as shown by Gao and Lenstra [6].

Construction I. *Suppose $n+1$ is a prime and q is primitive in Z_{n+1}, where q is a prime or prime power. Then any primitive $(n+1)$th root of unity generates an optimal normal basis of F_{q^n} over F_q.*

Construction II. *Let $2n+1$ be a prime and assume that Z^*_{2n+1} is generated by 2 and -1. Then $\alpha = \gamma + \gamma^{-1}$ generates an optimal normal basis of F_{2^n} over F_2, where γ is a primitive $(2n+1)$th root of unity.*

Optimal normal bases have been successfully used in hardware implementation of large finite fields in order to construct cryptosystems which are secure and efficient (see for example [1, 2]). There are several reasons for the interest in optimal normal bases, and in normal bases in general. When elements of F_{q^n} are represented under a normal basis over F_q, taking a q-th power of an element in F_{q^n} is just a cyclic shift

of coordinates, so the cost is almost negligible. In practice $q = 2$, exponentiation by repeated square and multiply can be greatly expedited. Another nice property about normal bases is that their multiplication tables possess a lot of symmetry which is useful in practical implementation of finite fields. For optimal normal bases, they have the additional property of being self-dual (Construction II only) and having the lowest complexity, which makes it feasible for hardware realization of large finite fields.

In several cryptographic systems (including exponential pseudorandom number generators), a fixed element of a group needs to be repeatedly raised to many different large powers. To make such systems secure, the fixed element must have high order. While to implement these systems, there should be an efficient algorithm for computing large powers of the fixed element. In this paper, we show by computational results that the optimal normal basis generators given in Construction II have exactly this desired property: they have very high multiplicative orders, and large powers of them can be computed efficiently as indicated by the following result.

Theorem. *Let α be as in Construction II. Then, for any integer $0 \le e \le 2^n - 1$, α^e can be computed in $O(n \cdot v(e))$ bit operations, where $v(e)$ denotes the number of 1's in the binary representation of e.*

As $v(e) \le n$ for $0 \le e \le 2^n - 1$, α^e can be computed in $O(n^2)$ bit operations. In comparison, we should mention that for an arbitrary $\beta \in F_{2^n}$, if F_{2^n} is represented by an optimal normal basis, Stinson [13] and von zur Gathen [7] showed that β^e can be computed in about $O(n/\log_2 n)$ multiplications in F_{2^n}, and thus in $O(n^3/\log_2 n)$ bit operations where squaring is considered free and one multiplication in F_{2^n} under the normal basis needs $O(n^2)$ bit operations. If F_{2^n} is represented by a power basis (which is of the form $1, \xi, \cdots, \xi^{n-1}$), then, by using fast algorithms [12, 5] for multiplication, β^e can be computed by the square and multiply method in $O(n \log n \operatorname{loglog} n \ \log e)$, or $O(n^2 \log n \log\log n)$ bit operations. It is not known how to improve the time $O(n \cdot v(e))$ with precomputations as in [3]. The reason is that we do not have an $O(n)$ algorithm for computing the product of two arbitrary elements in F_{2^n}.

In the following, we assume that the conditions in Construction II are satisfied. Our goal is to determine the multiplicative order of $\alpha = \gamma + \gamma^{-1}$. We will compute in F_{2^n} represented under the optimal normal basis generated by α with elements ordered differently.

We use the standard algorithm in [9, page 87] for determining the multiplicative orders of elements in finite fields. To apply this algorithm to compute the multiplicative order of an element in F_{2^n}, one needs to know the complete factorization of the integer $2^n - 1$. Tables of factorizations of integers of the form $b^n \pm 1$ for small b and n are given in [4] and updated versions are available from the authors. In the following, we show how to efficiently compute α^e for an arbitrary integer e.

The optimal normal basis generated by α is $(\alpha, \alpha^2, \cdots, \alpha^{2^{n-1}})$. We will arrange the elements of the basis in a different order. For an integer i, define $\gamma_i = \gamma^i + \gamma^{-i}$. Obviously, $\gamma_0 = 0$ and $\gamma_1 = \alpha$. As the multiplicative order of γ is $2n + 1$, it is easy to check that $\gamma_i = \gamma_j$ if and only if $i \equiv \pm j \bmod 2n + 1$. So $\gamma_1, \gamma_2, \cdots, \gamma_n$ are all the distinct nonzero γ_i's. We claim that

$$\{\alpha, \alpha^2, \cdots, \alpha^{2^{n-1}}\} = \{\gamma_1, \gamma_2, \cdots, \gamma_n\}.$$

The reason is that for each $0 \le i \le n - 1$, $\alpha^{2^i} = \gamma^{2^i} + \gamma^{-2^i} = \gamma_{2^i}$ belongs to the set of the right hand side, while for each $1 \le i \le n$, since Z^*_{2n+1} is generated by 2 and -1, there is an integer k such that $i \equiv \pm 2^k \bmod 2n + 1$, and thus $\gamma_i = \alpha^{2^k}$ belongs to the set of the left hand side.

Therefore $\gamma_1, \gamma_2, \cdots, \gamma_n$ form a basis of F_{2^n} over F_2. To facilitate multiplication of elements represented under this basis, we define a new function from the set of integers to the set $\{0, 1, \cdots, n\}$. For any integer i, define $s(i)$ to be the unique integer such that

$$0 \le s(i) \le n, \quad \text{and} \quad i \equiv s(i) \bmod 2n + 1 \text{ or } i \equiv -s(i) \bmod 2n + 1.$$

Obviously, $s(0) = 0$, $s(i) = s(-i)$ and

$$\gamma_i = \gamma_{s(i)}, \quad \alpha^{2^i} = \gamma_{s(2^i)}, \quad \text{for all } i.$$

As $\gamma_i \cdot \gamma_j = \gamma_{i+j} + \gamma_{i-j}$ for all i, j, we have

$$\gamma_i \cdot \gamma_j = \gamma_{s(i+j)} + \gamma_{s(i-j)}, \quad 1 \le i, j \le n.$$

Next we show how to compute the product $\gamma_i \cdot A$ where $1 \le i \le n$ and A is an arbitrary element in F_{2^n}. Suppose that $A = \sum_{k=1}^{n} a_k \gamma_k$ where $a_k \in F_2$. Then

$$\gamma_i \cdot A = \sum_{k=1}^{n} a_k \, \gamma_i \cdot \gamma_k = \sum_{k=1}^{n} a_k (\gamma_{s(k+i)} + \gamma_{s(k-i)}).$$

Note that

$$\begin{aligned}
\sum_{k=1}^{n} a_k \gamma_{s(k+i)} &= \sum_{k=1}^{n-i} a_k \gamma_{k+i} + \sum_{k=n+1-i}^{n} a_k \gamma_{2n+1-(k+i)} \\
&= \sum_{k=i+1}^{n} a_{k-i} \gamma_k + \sum_{k=n+1-i}^{n} a_{2n+1-(k+i)} \gamma_k \\
&= \sum_{k=i+1}^{n} a_{s(k-i)} \gamma_k + \sum_{k=n+1-i}^{n} a_{s(k+i)} \gamma_k, \\
\sum_{k=1}^{n} a_k \gamma_{s(k-i)} &= \sum_{k=1}^{i} a_k \gamma_{i-k} + \sum_{k=i+1}^{n} a_k \gamma_{k-i} \\
&= \sum_{k=1}^{i} a_{i-k} \gamma_k + \sum_{k=1}^{n-i} a_{n+k} \gamma_k \\
&= \sum_{k=1}^{i} a_{s(k-i)} \gamma_k + \sum_{k=1}^{n-i} a_{s(k+i)} \gamma_k,
\end{aligned}$$

where and hereafter we assume that $a_0 = 0$. We see that

$$\begin{aligned}
\gamma_i \cdot A &= \sum_{k=1}^{n} (a_{s(k-i)} + a_{s(k+i)}) \gamma_k \\
&= \sum_{k=1}^{c} (a_{i-k} + a_{k+i}) \gamma_k + \sum_{k=c+1}^{d} f(k) \gamma_k + \sum_{k=d+1}^{n} (a_{k-i} + a_{2n+1-(k+i)}) \gamma_k,
\end{aligned}$$

where $c = \min\{i, n-i\}$, $d = \max\{i, n-i\} = n - c$ and

$$f(k) = \begin{cases} a_{i-k} + a_{2n+1-(k+i)}, & \text{if } i > n-i, \\ a_{k-i} + a_{k+i}, & \text{if } i < n-i. \end{cases}$$

This shows that $\gamma_i \cdot A$ can be computed in $O(n)$ bit operations.

Now, to compute α^e we can assume that $0 \le e < 2^n - 1$, as $\alpha^{2^n - 1} = 1$. Write $e = \sum_{k=0}^{n-1} e_k 2^k$ where $e_k \in \{0, 1\}$. Then

$$\alpha^e = \prod_{k=0}^{n-1} (\alpha^{2^k})^{e_k} = \prod_{k=0}^{n-1} (\gamma_{s(2^k)})^{e_k}.$$

This suggests that α^e can be computed iteratively as follows.

Algorithm

Input: An integer e with $0 \le e \le 2^n - 1$.

Output: α^e represented in the basis $(\gamma_1, \cdots, \gamma_n)$.

Step 1. Set $A := 1 = \sum_{k=1}^{n} \gamma_k$ and compute the binary representation: $e = \sum_{k=0}^{n-1} e_k 2^k$;

Step 2. For k from 0 to $n-1$, if $e_k = 1$ then set $A := \gamma_{s(2^k)} \cdot A$;

Step 3. Return A;

End.

The correctness of the algorithm is obvious. The major cost is incurred at Step 2 where $v(e)$ products of the form $\gamma_k \cdot A$ are computed. Since we have shown that each such product can be computed in $O(n)$ bit operations, the total cost is $O(n \cdot v(e))$ bit operations. Therefore α^e can be computed in $O(n \cdot v(e))$ bit operations, as claimed by the theorem above.

By using the algorithms described above, we have computed the multiplicative order of α for $n \le 1200$ where the conditions of Construction II are satisfied and the complete factorization of $2^n - 1$ is known. The results are summarized in Table 1. The index of α in F_{2^n} is defined to be $(2^n - 1)/e$ where e is the multiplicative order of α. Thus index 1 in the table means that the corresponding α is a primitive element. The "?" in the table means that the complete factorization of the corresponding number $2^n - 1$ is not known yet, and thus the index computed from the partial factorization may not be the true index.

Table 1 indicates that the multiplicative order of α is at least $O((2^n - 1)/n)$. This means that α always has very high multiplicative order. The last two values of n in the table are Mersenne primes; the corresponding optimal normal basis generators α are automatically primitive. Note that α is frequently primitive. In particular, one can check that if n is prime then α is primitive in the table. We conjecture that this is always so, as stated below.

Conjecture. *Suppose that n and $2n+1$ are both primes. Then, for any primitive $(2n+1)$th root γ of unity in $F_{2^{2n}}$, $\alpha = \gamma + \gamma^{-1}$ is a primitive element in F_{2^n}.*

Note that when n and $2n+1$ are both primes, Z^*_{2n+1} is always generated by 2 and -1. So the conditions in Construction II are satisfied and α generates an optimal normal basis for F_{2^n} over F_2, which is easily seen to be a self-dual basis. This means that if the conjecture is true then α generates a self-dual, primitive optimal normal basis for F_{2^n} over F_2. Also note that it is not known if there are infinitely many integers n such that both n and $2n+1$ are primes, though it is conjectured so. It is interesting to note that D.H. Lehmer [8] found several chains of primes with each member one more than twice the previous one (such chains are called Cuningham chains), as listed in columns of Table 2.

Finally, we remark that the inverse of α is easy to compute. Actually, if $\alpha^{-1} = \sum_{i=1}^{n} a_k \gamma_k$, then

$$a_k = 1 \text{ if } \ell \text{ is odd, and } 0 \text{ if } \ell \text{ is even,}$$

where $\ell = k/2$ if k is even and $\ell = n - (k-1)/2$ if k is odd. So α^{-1} can be computed in $O(n)$ bit operations.

Acknowledgement. The authors would like to thank Joachim von zur Gathen for his helpful comments on earlier draft of the paper. Thanks also go to all the people contributed to the Cuningham project [4]. In particular, we would like to thank S.S. Wagstaff for collecting the new factors and making them electrically available. Also we used MAPLE in our computing. Part of Table 1 has appeared in [10].

n	Index	n	Index	n	Index	n	Index	n	Index
2	1	183	1	413	1	683	1?	950	3
3	1	186	3	414	3	686	3	953	1?
5	1	189	1	419	1	690	151	965	1
6	1	191	1	426	1	713	1?	974	3
9	1	194	3	429	1	719	1?	975	1
11	1	209	1	431	1	723	1	986	3
14	1	210	1	438	3	725	1?	989	1?
18	3	221	1	441	1	726	1	993	1
23	1	230	1	443	1	741	7	998	1
26	1	231	1	453	1	743	1?	1013	1?
29	1	233	1	470	1	746	1	1014	7
30	1	239	1	473	1	749	1?	1019	1?
33	1	243	1	483	1	755	1	1026	7
35	1	245	1	491	1	761	1?	1031	1?
39	1	251	1	495	1	765	1	1034	3
41	1	254	1	509	1	771	1	1041	1
50	3	261	1	515	1	774	1	1043	1?
51	1	270	7	519	1	779	1?	1049	1
53	1	273	1	530	1	783	7?	1055	1?
65	1	278	3	531	1	785	1?	1065	1?
69	1	281	1	543	1?	791	1?	1070	1
74	1	293	1	545	1	803	23?	1103	1
81	1	299	1	554	1	809	1?	1106	381
83	1	303	1	558	1	810	1	1110	9
86	1	306	1	561	1	818	1	1118	1?
89	1	309	1	575	1	831	1	1119	1
90	1	323	1	585	1	833	1?	1121	1
95	1	326	1	593	1?	834	1	1133	1?
98	3	329	1	606	9	846	1	1134	3
99	7	330	1	611	1?	866	1	1146	1
105	1	338	3	614	3	870	1	1154	1
113	1	350	3	615	1	873	1	1155	1
119	1	354	3	618	1	879	1	1166	1
131	1	359	1	629	1?	891	1	1169	1
134	3	371	1	638	1	893	1?	1178	3?
135	1	375	1	639	1	911	1?	1185	1
146	1	378	3	641	1?	923	1?	1194	1?
155	1	386	1	645	7	930	3	1199	1?
158	1	393	7	650	3	933	1?	1211	1?
173	1	398	1	651	1	935	1?	1218	1?
174	3	410	11	653	1	938	1	9689	1
179	1	411	1	659	1?	939	1?	21701	1

Table 1: Indices of optimal normal basis generators in F_{2^n}.

1122659	2164229	2329469	10257809	10309889
2245319	4328459	4658939	20515619	20619779
4490639	8656919	9317879	41031239	41239559
8981279	17313839	18635759	82062479	82479119
17962559	34627679	37271519	164124959	164958239
35925119	69255359	74543039	328249919	329916479
71850239	138510719	149086079	656499839	659832959

Table 2: Cunningham Chains of Primes

References

[1] G.B. Agnew, R.C. Mullin, I.M. Onyszchuk and S.A. Vanstone, "An implementation for a fast public key cryptosystem", *J. of Cryptology* **3** (1991), 63-79.

[2] G.B. Agnew, R.C. Mullin and S.A. Vanstone, "An implementation of elliptic curve cryptosystems over $F_{2^{155}}$", *IEEE J. on Selected Areas in Communications* **11** (1993), 804–813.

[3] E.F. Brickell, D.M. Gordon, K.S. McCurley and D.B. Wilson, "Fast exponentiation with precomputation", preprint, 1992.

[4] J. Brillhart, D.H. Lehmer, J.L. Selfridge, B. Tuckerman and S.S. Wagstaff, "Factorizations of $b^n \pm 1$, $b = 2, 3, 5, 6, 7, 10, 11, 12$ Up to High Powers", Vol. **22** of *Contemporary Mathematics*, AMS, 1988, 2nd edition.

[5] D.G. Cantor and E. Kaltofen, "On fast multiplication of polynomials over arbitrary algebra", *Acta. Inform.* **28** (1991), 693-701.

[6] S. Gao and H.W. Lenstra, Jr., "Optimal normal bases", *Designs, Codes and Cryptography* **2** (1992), 315-323.

[7] J. von zur Gathen, "Efficient and optimal exponentiation in finite fields", *Computational Complexity* **1** (1991), 360–394.

[8] D.H. Lehmer, "On certain chains of primes", *Proc. London Math. Soc.* **14A** (Littlewood 80 volume, 1965), 183-186.

[9] R. Lidl and H. Niederreiter, *Finite Fields*, Addison-Wesley, Reading, MA, 1983. (Now distributed by Cambridge University Press.)

[10] A.J. Menezes, I.F. Blake, X. Gao, R.C. Mullin, S.A. Vanstone and T. Yaghoobian, *Applications of Finite Fields*, Kluwer Academic Publishers, Boston-Dordrecht-Lancaster, 1993.

[11] R.C. Mullin, I.M. Onyszchuk, S.A. Vanstone and R.M. Wilson, "Optimal normal bases in $GF(p^n)$", *Discrete Applied Math.* **22** (1988/1989), 149-161.

[12] A. Schönhage and V. Strassen, "Schnelle Multiplikation großer Zahlen", *Computing* **7** (1971), 281-292.

[13] D.H. Stinson, "Some observations on parallel algorithms for fast exponentiation in $GF(2^n)$", *SIAM J. Computing* **19** (1990), 711-717.