JOURNAL OF THE
CHUNGCHEONG MATHEMATICAL SOCIETY
Volume **25**, No. 4, November 2012

CLASSIFICATION OF BINARY FORMALLY SELF-DUAL EVEN CODES OF LENGTH 18

Sunghyu Han*

Abstract. We give the complete classification of binary formally self-dual even codes of length 18. There are exactly 26568 inequivalent such codes. This completes the classification of binary formally self-dual even codes of length up to 18.

1. Introduction

A binary linear $[n, k]$ code C is a k-dimensional vector subspace of $\mathbb{F}_2^n$, where $\mathbb{F}_2$ is the finite field of two elements, and the elements of C are called codewords. The weight $wt(\mathbf{c})$ of a codeword $\mathbf{c}$ is the number of non-zero coordinates, and the minimum weight of C is the smallest weight among all non-zero codewords of C. An $[n, k, d]$ code denotes an $[n, k]$ code with minimum weight d. Two codes C and C' are equivalent if one can be obtained from the other by permuting the coordinates. The automorphism group of C is the set of permutations of the coordinates which preserve C. The weight enumerator of C is $W_C(x, y) = \sum_{i=0}^{n} A_i x^{n-i} y^i$, where A_i is the number of codewords of weight i in C. We shall set $x = 1$ for writing a weight enumerator. The $n + 1$ tuple $(A_0, A_1, A_2, ..., A_n)$ is called the weight distribution of C.

The dual code $C^\perp$ of C is defined as $C^\perp = \{\mathbf{v} \in \mathbb{F}_2^n \mid \mathbf{u} \cdot \mathbf{v} = 0 \text{ for all } \mathbf{u} \in C\}$ where $\mathbf{u} \cdot \mathbf{v}$ denotes the standard inner product of $\mathbf{u}$ and $\mathbf{v}$. A code C is *self-dual* if $C = C^\perp$. A code C is *isodual* if C and $C^\perp$ are equivalent to each other. A code C is *formally self-dual* (f.s.d.) if C and $C^\perp$ have identical weight enumerators. By definition, if C is

Received April 23, 2012; Accepted October 10, 2012.
2010 Mathematics Subject Classification: Primary 94B60.
Key words and phrases: classification, formally self-dual codes, formally self-dual even codes.
*This work was supported by the Basic Science Research Program through the National Research Foundation of Korea (NRF), which is supported by the Ministry of Education, Science and Technology (2010-0007232).

self-dual then C is isodual, and if C is isodual then C is f.s.d.. A code is called *even* if the weights of all codewords are even, otherwise the code is called *odd*.

Self-dual codes have received an enormous research effort due to their close connections to other mathematical structures such as block designs, lattices, modular forms, and sphere packings. Of course they are also interesting subjects by themselves (c.f. [6]). The classification of binary self-dual codes has been done up to length 36. For length 36, there are exactly 519492 inequivalent binary self-dual codes [5].

Since an f.s.d. even code may have a larger minimum weight than a self-dual code of the same length, f.s.d. even codes are interesting codes. Another advantage of considering f.s.d. even codes is that we can obtain designs from vectors of a fixed weight in an extremal f.s.d. even code by the Assmus-Mattson theorem. The classification of f.s.d. even codes was done up to length 16. For lengths $2, 4, 6$, and 8, the classification was done in 1994 [8]. For lengths $10, 12, 14$, and 16, the classification was done in 2001 [2].

The purpose of this paper is to give a classification of f.s.d. even codes of length 18. Let C be an $[n, k, d]$ f.s.d. even code. Then $d \leq 2[\frac{n}{8}] + 2$ [7, p. 379]. So that for the code length 18, the possible minimum weight $d = 2, 4, 6$. It is known that there is a unique code with minimum weight 6 [8, 11]. The following theorem is our main result.

Theorem 1.1. *There are exactly* 26568 *inequivalent formally self-dual even codes of length* 18, 2524 *of which are iso-dual and* 9 *of which are self-dual. There is exactly only one inequivalent formally self-dual even* $[18, 9, 6]$ *code which is iso-dual but not self-dual. There are exactly* 6819 *inequivalent formally self-dual even* $[18, 9, 4]$ *codes,* 743 *of which are iso-dual and* 2 *of which are self-dual. There are exactly* 19748 *inequivalent formally self-dual even* $[18, 9, 2]$ *codes,* 1780 *of which are iso-dual and* 7 *of which are self-dual.*

In Section 2, we give our classification algorithm. In Section 3, we describe the weight enumerators and the automorphism groups of the classified codes. All the computations are made using Magma [3]. The generator matrices of the classification can be found in [4].

2. Algorithm

In this section, we give our algorithm for a classification of binary f.s.d. even codes. Basically our algorithm use Recursive Build-up and Isomorph Rejection(RBIR) in [9, Section 3.1].

First we review the algorithm RBIR. This algorithm was used for a classification of linear codes [9]. Let $[I_k|A]$ be a parity check matrix of a $[2k, k]$ binary linear code, where I_k is the identity matrix of order k. If one column of the A part is deleted, we get a $[2k-1, k-1]$ code. Hence, all $[2k, k]$ codes can be obtained by starting from parity check matrices of the $[2k-1, k-1]$ codes, adding a new column in all possible ways, and removing equivalent forms of codes. This is done recursively, starting from the unique $[k, 0]$ code with parity check matrix $[I_k]$. Östergård noted that if we go through the candidates for a new column in lexicographic order, it is sufficient to test only candidates that are lexicographically bigger than the previous columns(that is, bigger than the last column).

We modify RBIR with two facts. First one is the following. Ostergård was only interested in construction of $[n, k]$ codes with minimum weight $d \geq 3$. Since we are also interested in the codes with minimum weight $d = 2$, we should test the candidate that is lexicographically equal to the previous column as well as ones that are lexicographically bigger than the previous columns.

For the second modification of RBIR, we need the following lemma.

LEMMA 2.1. *Let $H = [I|A]$ be a parity check matrix for a formally self-dual even code C (I is the identity matrix). Then each row of A and each column of A have odd weights.*

Proof. Note that $G = [A^T|I]$ is a generator matrix of the even code C. Each row of A^T (i.e., each column of A) has odd weight. Since H is a generator matrix of $C^\perp$ which is even, each row of A has odd weight. $\square$

By Lemma 2.1, we only have to consider odd weight candidates, when we add a new column in a parity check matrix. We define Modified Recursive Build-up and Isomorph Rejection(MRBIR) as the RBIR with above two modification. Now we describe our algorithm for the classification of $[2k, k]$ binary f.s.d. even codes.

Algorithm 1 (Main Algorithm)

- (Input) : k ($k \geq 2$)

- (Step 1): Find all inequivalent $[k+i, i]$ binary even codes using MRBIR for $1 \le i \le k-2$.
- (Step 2): For the parity check matrix of each inequivalent $[2k-2, k-2]$ binary even code, add all possible odd weight columns which are lexicographically equal to or bigger than the previous column, and then extend the $[2k-1, k-1]$ code to the $[2k, k]$ even code using the over all parity check, and then check whether the $[2k, k]$ even code is f.s.d., finally collect inequivalent $[2k, k]$ f.s.d. even codes.
- (Output): All inequivalent $[2k, k]$ binary f.s.d. even codes

If the n and k become large, then the running time grows very fast. The main reason is that the number of inequivalent codes becomes large if the n and k become large. So that if we have a new code then we have to test equivalence of the new code with the previously obtained many inequivalent codes. We use Magma built-in function "IsEquivalent(C_1, C_2)" for the equivalence test of two codes C_1 and C_2. IsEquivalent(C_1, C_2) returns true if C_1 and C_2 are equivalent, otherwise it returns false. To reduce the time of the equivalence test, we use the following lemma.

LEMMA 2.2. *Let C_1 and C_2 be $[n, k]$ binary linear codes. If C_1 and C_2 are equivalent then the followings are true. (i) The order of the automorphism group of C_1 and the order of the automorphism group of C_2 are the same. (ii) The weight distribution of C_1 and the weight distribution of C_2 are the same.*

Proof. It is obvious from the definition of equivalence. □

If we do not use Lemma 2.2, then we have the following equivalent test Algorithm 2. (In Algorithm 2, $InEqCodes$ is an array of the previously obtained inequivalent codes. C is a new code which we have to test equivalence with the previously obtained inequivalent codes. N is the size of $InEqCodes$.)

Algorithm 2

```
for i = 1 to N do
    if IsEquivalent(C, InEqCodes[i]) then
        break;
    end if;
end for;
```

TABLE 1. The number of inequivalent $[k+i,i]$ binary even codes ($1 \le i \le k-2$) and the number of inequivalent $[2k,k]$ f.s.d. even codes

$k \backslash i$	1	2	3	4	5	6	7	$k-1,k$	k-th Total Time Alg.1 with Alg.3	k-th Total Time Alg.1 with Alg.2
2								1	0.000	0.000
3	2							2	0.000	0.000
4	2	5						5	0.062	0.078
5	3	7	17					14	0.499	0.702
6	3	10	26	72				29	3.666	11.295
7	4	13	43	135	438			99	35.194	580.558
8	4	17	63	257	1031	4549		914	1060.495	68299.827
9	5	21	97	459	2479	15125	109261	26568	5146857.142	–

If we use Lemma 2.2, then we have the following equivalent test Algorithm 3. (In Algorithm 3, $\#AG(C)$ is the order of automorphism group of C and $WD(C)$ is the weight distribution of C.)

Algorithm 3

for $i = 1$ to N do
  if $(\#AG(C) = \#AG(InEqCodes[i]))$ and $(WD(C) = WD(InEqCodes[i]))$ then
    if IsEquivalent(C, $InEqCodes[i]$) then
      break;
    end if;
  end if;
end for;

Algorithm 3 is faster than Algorithm 2 in our case. The reason is the following. If we have a new code C then we have to test equivalence of C with the previously found inequivalent codes, $InEqCodes[i]$. For a given i, it is more probable that C and $InEqCodes[i]$ are not equivalent. In Algorithm 3, before we call $IsEquivalent(C, InEqCodes[i])$, it happens more frequently that $\#AG(C) \ne \#AG(InEqCodes[i])$ or $WD(C) \ne WD(InEqCodes[i])$. In this case, we can avoid the function call $IsEquivalent(C, InEqCodes[i])$ which is the most time consuming operation.

In Table 1, we give our computation results of Algorithm 1. We describe the number of inequivalent $[k+i,i]$ even codes $(1 \le i \le k-2)$ and the number of inequivalent $[2k,k]$ f.s.d. even codes. The first column represents k, the Input value. The second column to the eighth column

TABLE 2. Binary formally self-dual even codes of length $2 \leq n \leq 18$

n	$\#fsd$	$\#iso$	$\#sd$	d_{max}	$\#max, fsd$	$\#max, iso$	$\#max, sd$
2	1	1	1	2	1	1	1
4	1	1	1	2	1	1	1
6	2	2	1	2	2	2	1
8	5	5	2	4	1	1	1
10	14	10	2	4	1	1	0
12	29	23	3	4	3	3	1
14	99	71	4	4	10	10	1
16	914	338	7	4	144	68	3
18	26568	2524	9	6	1	1	0

represent the number of inequivalent $[k+i, i]$ binary even codes($1 \leq i \leq k-2$) of Step 1. The ninth column represents the number of inequivalent $[2k, k]$ binary f.s.d. even codes of Step 2. The 10th column represents the total computing time in seconds for each k using Algorithm 3. The last column represents the total computing time in seconds for each k using Algorithm 2. We used notebook PC with 2GB RAM and 2.00 GHz. From the results, we know that Algorithm 3 is much faster than Algorithm 2. The total calculation time of the classification of binary f.s.d. even codes of length 18 is 5146857.142 seconds which is about two months. In Table 2, we give a complete classification of binary f.s.d. even codes up to length $n \leq 18$. The number of inequivalent binary f.s.d. even codes is listed under "#fsd". Among the f.s.d. even codes, the number of isodual codes is listed under "#iso". Among the isodual codes, the number of self-dual codes is listed under "#sd". In the table, "d_{max}" is the largest minimum weight for which a binary f.s.d. even code exists. The columns headed "#max,fsd", "#max,iso", and "#max,sd"are, respectively, the number of binary f.s.d. even codes with minimum weight d_{max}, the number of isodual binary f.s.d. even codes with minimum weight d_{max}, and the number of binary self-dual codes with minimum weight d_{max}.

TABLE 3. The weight enumerators of binary formally self-dual even codes of length 18

(α,β)	N_F	N_I	N_S	(α,β)	N_F	N_I	N_S	(α,β)	N_F	N_I	N_S
$(-9,-9)$	1	1	0	$(-9,-7)$	1	1	0	$(-9,-6)$	5	3	0
$(-9,-5)$	7	5	0	$(-9,-4)$	44	22	0	$(-9,-3)$	71	19	0
$(-9,-2)$	334	38	0	$(-9,-1)$	357	45	0	$(-9,0)$	1185	155	1
$(-9,1)$	756	68	0	$(-9,2)$	1566	78	0	$(-9,3)$	653	63	0
$(-9,4)$	1023	109	0	$(-9,5)$	245	35	0	$(-9,6)$	369	33	0
$(-9,7)$	42	14	0	$(-9,8)$	118	36	1	$(-9,9)$	6	6	0
$(-9,10)$	24	6	0	$(-9,12)$	9	5	0	$(-9,14)$	2	0	0
$(-9,16)$	1	1	0	$(-9,20)$	1	1	0	$(-8,-8)$	4	4	0
$(-8,-7)$	3	3	0	$(-8,-6)$	44	10	0	$(-8,-5)$	53	13	0
$(-8,-4)$	327	51	0	$(-8,-3)$	310	30	0	$(-8,-2)$	1166	50	0
$(-8,-1)$	839	49	0	$(-8,0)$	2213	169	1	$(-8,1)$	1044	54	0
$(-8,2)$	1883	67	0	$(-8,3)$	623	45	0	$(-8,4)$	977	87	0
$(-8,5)$	172	24	0	$(-8,6)$	272	22	0	$(-8,7)$	15	7	0
$(-8,8)$	92	22	0	$(-8,9)$	2	2	0	$(-8,10)$	12	2	0
$(-8,12)$	5	5	0	$(-8,16)$	6	6	2	$(-7,-10)$	1	1	0
$(-7,-9)$	1	1	0	$(-7,-8)$	13	7	0	$(-7,-7)$	6	2	0
$(-7,-6)$	80	8	0	$(-7,-5)$	57	11	0	$(-7,-4)$	268	32	0
$(-7,-3)$	222	20	0	$(-7,-2)$	745	33	0	$(-7,-1)$	437	27	0
$(-7,0)$	1048	110	1	$(-7,1)$	463	27	0	$(-7,2)$	847	35	0
$(-7,3)$	264	20	0	$(-7,4)$	377	39	0	$(-7,5)$	92	14	0
$(-7,6)$	130	10	0	$(-7,7)$	13	7	0	$(-7,8)$	28	12	0
$(-7,9)$	1	1	0	$(-7,10)$	11	3	0	$(-7,12)$	4	2	0
$(-7,16)$	1	1	0	$(-6,-12)$	2	2	0	$(-6,-10)$	3	1	0

3. Weight enumerators and automorphism groups

In this section, we describe the weight enumerators and the automorphism groups of the classified binary f.s.d. even codes of length 18.

3.1. Weight enumerators

The weight enumerator $W_C(1,y)$ of a binary f.s.d. even code C of length n is written using the Gleason's theorem (c.f. [10]) as

$$W_C(1,y) = \sum_{j=0}^{[n/8]} a_j(1+y^2)^{n/2-4j}(y^2(1-y^2)^2)^j$$

where a_j's are undetermined parameters. Thus the weight enumerators of an f.s.d. even code of length 18 is

$$W_C(1,y) = 1+(9+\alpha)y^2+(36+3\alpha+\beta)y^4+(84+\alpha-3\beta)y^6+(126-5\alpha+2\beta)y^8+\cdots,$$

where $1 = a_0, \alpha = a_1$, and $\beta = a_2$.

In Table 3 and Table 4, the ith column gives the values (α,β) in the weight enumerator $W_C(1,y)$, the $(i+1)$-st, $(i+2)$-nd, and $(i+3)$-rd column list the number N_F, N_I, and N_S of all the inequivalent f.s.d. even codes, all the isodual codes, and all the self-dual codes, respectively,

Table 4. The weight enumerators of binary formally self-dual even codes of length 18(continued)

(α,β)	N_F	N_I	N_S	(α,β)	N_F	N_I	N_S	(α,β)	N_F	N_I	N_S
$(-6,-9)$	4	2	0	$(-6,-8)$	39	7	0	$(-6,-7)$	10	4	0
$(-6,-6)$	80	6	0	$(-6,-5)$	58	6	0	$(-6,-4)$	248	28	0
$(-6,-3)$	133	13	0	$(-6,-2)$	367	21	0	$(-6,-1)$	184	14	0
$(-6,0)$	512	60	1	$(-6,1)$	160	12	0	$(-6,2)$	286	20	0
$(-6,3)$	83	11	0	$(-6,4)$	167	21	0	$(-6,5)$	27	7	0
$(-6,6)$	31	5	0	$(-6,7)$	4	2	0	$(-6,8)$	20	8	0
$(-6,12)$	1	1	0	$(-5,-12)$	1	1	0	$(-5,-10)$	6	2	0
$(-5,-8)$	23	7	0	$(-5,-7)$	8	4	0	$(-5,-6)$	53	3	0
$(-5,-5)$	38	6	0	$(-5,-4)$	119	9	0	$(-5,-3)$	68	4	0
$(-5,-2)$	217	13	0	$(-5,-1)$	91	7	0	$(-5,0)$	233	45	0
$(-5,1)$	78	8	0	$(-5,2)$	140	10	0	$(-5,3)$	38	6	0
$(-5,4)$	40	6	0	$(-5,5)$	18	4	0	$(-5,6)$	23	3	0
$(-5,8)$	6	2	0	$(-5,9)$	1	1	0	$(-5,10)$	1	1	0
$(-5,12)$	1	1	0	$(-4,-10)$	2	0	0	$(-4,-9)$	1	1	0
$(-4,-8)$	8	2	0	$(-4,-7)$	1	1	0	$(-4,-6)$	12	2	0
$(-4,-5)$	13	1	0	$(-4,-4)$	36	8	0	$(-4,-3)$	30	6	0
$(-4,-2)$	62	6	0	$(-4,-1)$	33	5	0	$(-4,0)$	98	22	1
$(-4,1)$	22	2	0	$(-4,2)$	42	6	0	$(-4,3)$	17	3	0
$(-4,4)$	28	8	0	$(-4,5)$	8	2	0	$(-4,6)$	10	2	0
$(-4,7)$	1	1	0	$(-4,8)$	3	1	0	$(-3,-16)$	1	1	0
$(-3,-12)$	6	2	0	$(-3,-10)$	9	3	0	$(-3,-9)$	1	1	0
$(-3,-8)$	8	0	0	$(-3,-7)$	3	1	0	$(-3,-6)$	10	0	0
$(-3,-5)$	11	3	0	$(-3,-4)$	24	6	0	$(-3,-3)$	9	1	0
$(-3,-2)$	31	3	0	$(-3,-1)$	21	1	0	$(-3,0)$	35	13	0
$(-3,1)$	9	3	0	$(-3,2)$	10	6	0	$(-3,3)$	7	1	0
$(-3,4)$	13	1	0	$(-3,5)$	2	0	0	$(-3,7)$	1	1	0
$(-3,9)$	1	1	0	$(-3,10)$	1	1	0	$(-3,12)$	2	2	0
$(-2,-12)$	1	1	0	$(-2,-10)$	2	0	0	$(-2,-8)$	13	3	0
$(-2,-7)$	1	1	0	$(-2,-6)$	5	1	0	$(-2,-5)$	4	2	0
$(-2,-4)$	11	3	0	$(-2,-3)$	11	1	0	$(-2,-2)$	26	2	0
$(-2,-1)$	13	1	0	$(-2,0)$	33	7	0	$(-2,1)$	5	3	0
$(-2,2)$	9	1	0	$(-2,4)$	9	5	0	$(-2,5)$	3	1	0
$(-2,6)$	1	1	0	$(-2,7)$	2	2	0	$(-1,-10)$	2	0	0
$(-1,-7)$	1	1	0	$(-1,-6)$	10	2	0	$(-1,-5)$	3	1	0
$(-1,-3)$	4	0	0	$(-1,-2)$	9	3	0	$(-1,-1)$	4	0	0
$(-1,0)$	15	9	0	$(-1,1)$	5	1	0	$(-1,2)$	5	1	0
$(-1,3)$	2	2	0	$(-1,6)$	1	1	0	$(0,-9)$	1	1	0
$(0,-6)$	2	0	0	$(0,-4)$	5	1	0	$(0,-3)$	1	1	0
$(0,-2)$	3	1	0	$(0,0)$	9	5	1	$(0,1)$	2	0	0
$(0,2)$	6	2	0	$(0,3)$	1	1	0	$(0,4)$	2	0	0
$(0,8)$	1	1	0	$(1,-20)$	1	1	0	$(1,-12)$	1	1	0
$(1,-9)$	1	1	0	$(1,-8)$	2	0	0	$(1,-5)$	1	1	0
$(1,-2)$	5	1	0	$(1,-1)$	2	0	0	$(1,0)$	6	2	0
$(1,1)$	2	2	0	$(1,2)$	1	1	0	$(1,3)$	1	1	0
$(2,-7)$	1	1	0	$(2,-4)$	5	3	0	$(2,0)$	5	3	0
$(3,-10)$	1	1	0	$(3,-5)$	1	1	0	$(3,-2)$	1	1	0
$(3,1)$	5	1	0	$(3,2)$	4	0	0	$(3,9)$	1	1	0
$(4,0)$	1	1	0	$(4,2)$	3	1	0	$(4,3)$	1	1	0
$(4,4)$	3	1	0	$(5,0)$	2	2	0	$(5,2)$	1	1	0
$(6,-12)$	1	1	0	$(7,-7)$	1	1	0	$(7,9)$	1	1	0
$(7,10)$	1	1	0	$(8,-2)$	1	1	0	$(9,0)$	2	2	0
$(9,3)$	1	1	0	$(13,-5)$	1	1	0	$(14,0)$	1	1	0
$(20,2)$	1	1	0	$(27,9)$	1	1	0				

TABLE 5. Automorphism groups of binary formally self-dual even codes of length 18

#Aut(C)	#*codes*	#Aut(C)	#*codes*	#Aut(C)	#*codes*	#Aut(C)	#*codes*
1	2327	768	262	27648	28	373248	2
2	1473	864	10	30720	2	387072	3
3	6	960	1	31104	2	414720	1
4	4795	1024	101	32768	1	451584	1
6	27	1152	184	36864	13	460800	2
8	2918	1296	2	41472	12	518400	1
9	1	1344	3	46080	2	552960	6
12	40	1536	120	49152	1	663552	2
16	4550	1680	1	51840	1	691200	1
18	2	1728	19	55296	28	829440	1
24	92	2048	26	57600	2	921600	1
28	1	2304	169	61440	1	967680	1
32	2047	2448	1	62208	4	995328	4
36	24	2688	5	69120	1	1036800	2
40	1	3072	79	73728	4	1105920	4
48	305	3456	130	82944	15	1290240	1
56	2	4096	10	86016	1	1327104	1
64	2202	4320	1	92160	1	1548288	1
72	46	4608	59	96768	2	1658880	5
80	2	5184	21	110592	13	1679616	1
96	387	5376	4	112896	1	2073600	1
108	1	5760	2	115200	2	2211840	3
112	1	6144	32	122880	1	2322432	2
128	930	6912	47	124416	4	2654208	2
144	239	7680	1	129024	1	4147200	1
160	2	8064	2	138240	1	5160960	1
192	493	8192	1	147456	1	5529600	1
216	6	9216	72	165888	6	7225344	1
240	1	10368	24	172032	1	8294400	2
256	514	10752	4	186624	1	10321920	1
288	159	12288	9	193536	1	12441600	1
320	2	13824	26	230400	2	18662400	2
360	1	15360	1	245760	1	19353600	1
384	647	16128	6	276480	5	24883200	1
432	8	16384	1	279936	1	101606400	1
512	204	18432	38	290304	2	185794560	1
576	366	20736	16	294912	2	203212800	1
640	1	21504	1	322560	1	3251404800	1
672	2	24192	6	331776	4	131681894400	1
720	1	24576	6	345600	1		

with the weight enumerator corresponding to (α, β), where $i = 1, 5, 9, 13$. From Table 3 and Table 4, we know that there are 227 different weight enumerators.

3.2. Automorphism groups

Another property of a code is its automorphism group. The order of the automorphism groups of the classified codes are displayed in Table 5. In Table 5, $\#Aut(C)$ is the order of the automorphism group and $\#codes$ is the number of codes whose automorphism group order is $\#Aut(C)$.

From Table 5, we know that the total number of the different orders of the automorphism groups is 159.

References

[1] P. Monk, *An iterative finite element method for approximating the biharmonic equation,* Math. Comp. **151** (1988), 451-476.

[2] K. Betsumiya K and M. Harada, *Classification of formally self-dual even codes of lengths up to 16,* Des. Codes Cryptogr. **23** (2001), 325–332.

[3] W. Bosma, J. Cannon, and C. Playoust, *The Magma algebra system. I. The user language,* J. Symbolic Comput. **24** (1997), 235-265.

[4] S. Han, *online available at http://kutacc.kut.ac.kr/˜sunghyu/data/bfsd.htm*

[5] M. Harada and A. Munemasa, *Classification of Self-Dual Codes of Length 36,* Advances Math. Communications, to appear.

[6] W. C. Huffman, *On the classification and enumeration of self-dual codes,* Finite Fields Appl. **11** (2005), 451–490.

[7] W. C. Huffman and V. S. Pless, *Fundamentals of Error-correcting Codes,* Cambridge University Press, Cambridge (2003).

[8] G. T. Kennedy and V. Pless, *On designs and formally self-dual codes,* Des. Codes Cryptogr. **4** (1994), 43–55.

[9] P. R. J. Östergård, *Classifying subspaces of hamming spaces,* Des. Codes Cryptogr. **27** (2002), 297–305.

[10] E. Rains and N. J. A. Sloane, *Self-dual codes Handbook of Coding Theory (V. S. Pless and W. C. Huffman, eds.),* Elsevier, Amsterdam, (1998) 177–294.

[11] J. Simonis, *The [18,9,6] code is unique,* Discrete Math. **106/107** (1992), 439–448.

$*$

School of Liberal Arts
Korea University of Technology and Education
Cheonan 330-708, Republic of Korea
E-mail: `sunghyu@koreatech.ac.kr`