

# Deterministic Differential Properties of the BMW Compression Function

Jian Guo<sup>1</sup> and Søren S. Thomsen<sup>2,\*</sup>

<sup>1</sup> Nanyang Technological University, Singapore

<sup>2</sup> DTU Mathematics, Technical University of Denmark

**Abstract.** In this paper, we give some deterministic differential properties for the compression function of SHA-3 candidate Blue Midnight Wish (tweaked version for round 2). The computational complexity is about  $2^0$  compression function calls. This applies to security parameters 0/16, 1/15, and 2/14. The efficient differentials can be used to find pseudo-preimages of the compression function with marginal gain over brute force. However, none of these attacks threaten the security of the BMW hash functions.

**Keywords:** Hash function cryptanalysis, Blue Midnight Wish, SHA-3, differential

## 1 Introduction

Blue Midnight Wish [3] (BMW) is one of the 14 second round candidates of NIST’s cryptographic hash algorithm competition [5]. It was tweaked after being selected for round 2, apparently in order to resist attacks by Thomsen [7]. Aumasson [1] and Nikolić et al. [6], independently of our work, found some distinguishers with data complexity  $2^{19}$ , and for a modified variant of BMW-512 with probability  $2^{-278.2}$ , respectively. In this paper, we give explicit constructions of message pairs, by tracing the propagation of the differences, to show some interesting behaviour on certain bits of the output with probability 1.

The paper is organised as follows. Section 2 gives a brief description of BMW. Then, we introduce some general observations in Section 3, which are further extended to differentials for BMW variants with security parameters 0/16, 1/15, 2/14, in Sections 4, 5, 6, respectively. A pseudo-preimage attack on the compression function using such efficient differentials is discussed in Section 7. Section 8 concludes the paper.

## 2 Description of BMW

BMW is a family of hash functions, containing four major instances, BMW- $n$ , with  $n \in \{224, 256, 384, 512\}$ , where  $n$  is the size of the hash output. It follows a tweaked Merkle-Damgård structure with double-pipe design, i.e., the size of the chaining value is twice the output size. Since our differentials concentrate on the compression function only, we refer to (tweaked for round 2) submission documents [3] for the descriptions of padding, finalisation, etc.

The compression function  $\text{bmw}_n$  of BMW- $n$  takes the chaining value  $H$  and a message block  $M$  as input, and produces the updated chaining value  $H^*$ . All  $H$ ,  $M$ , and  $H^*$  are of 16 words, where the size of a word is 32 bits for BMW-224/256, and 64 bits for BMW-384/512. We use  $X_i$  ( $i = 0, \dots, 15$ ) to denote the  $i$ -th word of  $X$ . The compression function comprises three functions, called  $f_0$ ,  $f_1$ , and  $f_2$ , in sequence. We introduce them here.

**The  $f_0$  function.** A temporary  $W$  is introduced as

$$\begin{aligned} W_i \leftarrow & \pm(M_{i+5} \oplus H_{i+5}) \pm (M_{i+7} \oplus H_{i+7}) \pm (M_{i+10} \oplus H_{i+10}) \\ & \pm(M_{i+13} \oplus H_{i+13}) \pm (M_{i+14} \oplus H_{i+14}) \end{aligned} \tag{1}$$

for  $i = 0, \dots, 15$ . By ‘ $\pm$ ’ we mean ‘+’ or ‘−’; which operator is used varies and does not seem to follow any simple pattern (see [3, Table 2.2] for details). Unless specified otherwise, all additions (and subtractions) are

---

\*Part of this work was carried out while the author was visiting Nanyang Technological University, by the support of the Singapore National Research Foundation under Research Grant NRF-CRP2-2007-03.

to be taken modulo  $2^w$  (where  $w$  is the word size) and all indices for  $H$  and  $M$  are modulo 16 throughout this paper. The outputs of  $f_0$  are  $Q_i$ ,  $i = 0, \dots, 15$ , which are computed as

$$Q_i \leftarrow s_{i \bmod 5}(W_i) + H_{i+1}, \quad (2)$$

where  $s_i$  are predefined bijective functions with  $i = 0, \dots, 4$ ; see Appendix A for the definitions of these. Note that without the feed-forward of  $H_{i+1}$ , the output of  $f_0$  would be a permutation of  $H \oplus M$ , since the computation of  $W$  corresponds to a multiplication by an invertible matrix.

**The  $f_1$  function.**  $f_1$  takes  $H$ ,  $M$ , and  $Q_0, \dots, Q_{15}$  (the output from  $f_0$ ) as input, and produces 16 new words  $Q_j$ , for  $j = 16, \dots, 31$ . The output words are computed one at a time through 16 rounds. There are two types of rounds,  $expand_1$  rounds and  $expand_2$  rounds. We denote the number of  $expand_1$  rounds by  $R$ , where  $R$  is a security parameter that can take any value between 0 and 16. There are  $16 - R$   $expand_2$  rounds. For the sake of clarity, we shall denote a specific choice of security parameter by  $R/(16 - R)$ ; the value of the security parameter suggested by the designers is  $2/14$  (in other words: 2  $expand_1$  rounds and 14  $expand_2$  rounds).

The 16 output words  $Q_{16}, \dots, Q_{31}$  are computed as follows. An  $expand_1$  round computes

$$Q_{j+16} \leftarrow AddElement(j) + \sum_{i=0}^{15} s_{(i+1) \bmod 4}(Q_{i+j-16}). \quad (3)$$

Here,  $AddElement$  is defined as:

$$AddElement(j) \leftarrow (M_j^{\lll(j \bmod 16)+1} + M_{j+3}^{\lll(j+3 \bmod 16)+1} - M_{j+10}^{\lll(j+10 \bmod 16)+1} + K_j) \oplus H_{j+7}, \quad (4)$$

where  $X^{\lll n}$  denotes a left-rotation of register  $X$  by  $n$  positions (by left we mean towards the most significant bit). The words  $K_j$  are round constants equal to  $(j + 16) \cdot 0555555555555555_h$  for BMW-384/512 and  $(j + 16) \cdot 055555555_h$  for BMW-224/256. An  $expand_2$  round computes

$$\begin{aligned} Q_{j+16} \leftarrow & Q_j + r_1(Q_{j+1}) + Q_{j+2} + r_2(Q_{j+3}) + Q_{j+4} + r_3(Q_{j+5}) + Q_{j+6} + \\ & r_4(Q_{j+7}) + Q_{j+8} + r_5(Q_{j+9}) + Q_{j+10} + r_6(Q_{j+11}) + \\ & Q_{j+12} + r_7(Q_{j+13}) + s_4(Q_{j+14}) + s_5(Q_{j+15}) + AddElement(j). \end{aligned} \quad (5)$$

The functions  $r_i$  are rotation functions; see Appendix A for details.

**The  $f_2$  function.** We list the description of  $H_0^*$ , since our result concerns this word only.

$$H_0^* \leftarrow (XH^{\ll 5} \oplus Q_{16}^{\gg 5} \oplus M_0) + (XL \oplus Q_{24} \oplus Q_0), \quad (6)$$

where

$$\begin{aligned} XL &= Q_{16} \oplus \dots \oplus Q_{23}, \\ XH &= Q_{16} \oplus \dots \oplus Q_{31}. \end{aligned}$$

**Some notations.** The attacks described in this paper deal with input pairs for which there is a certain relation on the output pair. Hence, we shall be interested in how differences propagate through the BMW compression function. We use the following notation (apparently first introduced by De Cannière and Rechberger [2]) for the difference between two bits: ‘-’ means there is no difference with probability 1, ‘x’ means there is a difference with probability 1, and ‘?’ means there may or may not be a difference (the probability of a difference is not 0 or 1, but also may be bounded away from  $1/2$ ). When we talk about a difference in a word, e.g., in a 32-bit word, we write (for instance)

$$[????????????????x-----],$$

which means that the 14 least significant bits contain no difference, the 15th least significant bit contains a difference, and the 17 most significant bits may or may not contain a difference.

With the above descriptions, we are able to introduce our differentials starting with some important observations on the least significant bit (LSB) of  $H_0^*$ .

### 3 Observations

Let  $X[n]$  denote the  $n$ th bit of the word  $X$ , where the least significant bit is the 0th bit. Since an addition takes no carry into the least significant bit, we can state the following expression for the LSB  $H_0^*[0]$  of  $H_0^*$ :

$$H_0^*[0] = Q_{16}[5] \oplus M_0[0] \oplus XL[0] \oplus Q_{24}[0] \oplus Q_0[0].$$

Given the definition of  $XL$ , this expression can be restated as

$$H_0^*[0] = M_0[0] \oplus Q_0[0] \oplus Q_{16}[5] \oplus \bigoplus_{i=16}^{24} Q_i[0]. \quad (7)$$

Hence,  $H_0^*[0]$  does not depend on  $Q_{25}, \dots, Q_{31}$ . This means that if we can limit difference propagation through the first 9 rounds of  $f_1$  (where  $Q_{16}, \dots, Q_{24}$  are computed), and if we can still keep the difference on  $M_0[0]$  and  $Q_0[0]$  under our control, then the bit  $H_0^*[0]$  may be biased.

In the function  $f_1$ , differences may propagate only very slowly towards the LSB. Consider an *expand<sub>2</sub>* round:

$$\begin{aligned} Q_{j+16} \leftarrow & Q_j + r_1(Q_{j+1}) + Q_{j+2} + r_2(Q_{j+3}) + Q_{j+4} + r_3(Q_{j+5}) + Q_{j+6} + \\ & r_4(Q_{j+7}) + Q_{j+8} + r_5(Q_{j+9}) + Q_{j+10} + r_6(Q_{j+11}) + \\ & Q_{j+12} + r_7(Q_{j+13}) + s_4(Q_{j+14}) + s_5(Q_{j+15}) + \text{AddElement}(j). \end{aligned} \quad (8)$$

The function  $s_5$  is defined as

$$s_5(x) = x^{\gg 2} \oplus x.$$

Here  $x^{\gg 2}$  means a right-shift by two bit positions. Hence, if  $Q_{j+15}$  contains a difference in an *expand<sub>2</sub>* round, then the function  $s_5$  propagates this difference two positions down towards the LSB. For example, the difference

$$[?????????????????x\text{-----}]$$

would become

$$[?????????????????x\text{-----}].$$

### 4 The security parameter 0/16

Consider a variant of BMW with security parameter 0/16, meaning that all 16 rounds in  $f_1$  are of the *expand<sub>2</sub>* type. Consider an input pair to the compression function such that there is a difference in  $Q_0$  but in no word among  $Q_1, \dots, Q_{15}$ , nor in  $M_0, M_3, M_{10}$ , and  $H_7$ . This difference on  $Q_0$  will propagate to  $Q_{16}$ . Due to the additions, the difference may propagate towards the most significant bit (MSB), but never towards the LSB. Hence, if the  $t$  LSBs of  $Q_0$  contain no difference, then these bits also contain no difference in  $Q_{16}$ . As an example, the difference  $[\text{---}x\text{---}x\text{---}x\text{---}x\text{---}]$  in  $Q_0$  becomes  $[?????????????????x\text{-----}]$  in  $Q_{16}$ .

In the second round,  $Q_{16}$  will go through the function  $s_5$ , and the difference will be shifted two positions towards the least significant bit. In the example above, we would get  $[?????????????????x\text{-----}]$ . Hence, there will be no difference in the  $t - 2$  least significant bits of  $s_5(Q_{16})$ . The word  $Q_0$  no longer affects the function  $f_1$ , and if there is no difference in the  $t - 2$  least significant bits of  $\text{AddElement}(1)$ , then  $Q_{17}$  will contain no difference in the  $t - 2$  LSBs. In the following round (under some conditions on  $M$  and  $H$ ), the difference again propagates two positions towards the LSB, meaning that the  $t - 4$  LSBs contain no difference.

The condition that the only difference in the words  $Q_0, \dots, Q_{15}$  lies in  $Q_0$  can be enforced by having the same difference in  $H_1$  and in  $M_1$ , and no difference in all other words of  $H$  and  $M$ . This means that there is no difference in the permutation inside  $f_0$ , but the difference in  $H_1$  will be fed forward to  $Q_0$ . Denote by  $\Delta$  the difference on  $H_1$  and  $M_1$ . If  $\Delta$  has many trailing '0' bits, i.e., there is no difference in many LSBs of  $H_1$  and  $M_1$ , then the behaviour described above occurs.

The word  $M_1$  is involved in rounds 1, 7, and 14 of  $f_1$ , and  $H_1$  is involved in round 10. In rounds 1 and 7,  $M_1$  is rotated two positions left, and therefore, in order to keep differences out of the least significant bit positions, we need  $\Delta$  to have '0' bits in the two MSB positions. In rounds 9–15, we do not worry about difference propagation, since this will affect only the words  $Q_{25}, \dots, Q_{31}$ , which are not involved in the computation of  $H_0^*[0]$ .

The only remaining potential source of differences in the least significant bit positions are due to the rotation functions  $r_i$ . Looking closely at the effects of these functions one sees that they make no difference in the case of BMW-224/256, but they do have a significant effect in the case of BMW-384/512. On the other hand, in BMW-384/512, the “distance” to the LSB is greater, and therefore it is still possible to obtain interesting results as described now.

The difference  $\Delta$  with the maximum value of  $t$  fulfilling the mentioned requirements is  $\Delta = 2^{61}$  for BMW-384/512 (and  $\Delta = 2^{29}$  for BMW-224/256). Hence, we have the difference

$$[--x-----]$$

on  $H_1$  and  $M_1$ , which becomes

$$[??x-----]$$

in  $Q_0$  due to the feed forward of  $H_1$ . The 16 words computed in  $f_1$  will have the following differences:

$$\begin{aligned}\Delta Q_{16} &= [??x-----] \\ \Delta Q_{17} &= [????x-----] \\ \Delta Q_{18} &= [?????x-----] \\ \Delta Q_{19} &= [?????????x-----] \\ \Delta Q_{20} &= [?????????????x-----] \\ \Delta Q_{21} &= [?????????????????x-----] \\ \Delta Q_{22} &= [?????????????????????x-----] \\ \Delta Q_{23} &= [?????????????????????????x-----] \\ \Delta Q_{24} &= [?????????????????????????????x-----] \\ \Delta Q_{25} &= [?????????????????????????????????x-----] \\ \Delta Q_{26} &= [?????????????????????????????????????x-----] \\ \Delta Q_{27} &= [???x-----] \\ \Delta Q_{28} &= [???x-----] \\ \Delta Q_{29} &= [???x-----] \\ \Delta Q_{30} &= [???x-----] \\ \Delta Q_{31} &= [???x-----]\end{aligned}$$

The end result in the output word  $H_0^*$  is the difference (one can verify this by substituting all above differences to Eqn. (6))

$$[???-----].$$

Hence, there is no difference in the 5 LSBs with probability 1. In fact, there is also a strong bias in  $H_5^*$ , which has the difference

$$[???-----].$$

For BMW-224/256 one gets a similar behaviour; the difference on  $H_0^*$  is

$$[?????????????????????????????????????-----],$$

and the difference on  $H_5^*$  is  $[?????????????????????????????????x---]$ .

## 5 The security parameter 1/15

When there is a single  $expand_1$  round in the beginning of  $f_1$ , followed by 15  $expand_2$  rounds, we can get a similar behaviour as described in the previous section if we can find a difference  $\Delta$  with many LSBs equal to 0, and such that  $s_1(\Delta)$  also has many LSBs equal to 0. We shall investigate this in a moment.

Now, in order to keep the difference  $\Delta$  from being changed by the feed-forward with  $H$  in  $f_0$ , we need a few more conditions on  $H$  and  $M$  compared to the security parameter 0/16. What we need is that  $s_0(W_0)$  contains ‘0’ bits in the positions where  $\Delta$  contains ‘1’ bits. An easy way to ensure this is by requiring that  $M_i$  and  $H_i$  are equal to zero for  $i \in \{5, 7, 10, 13, 14\}$ . Alternatively, without introducing any requirements, the condition is fulfilled with probability  $2^{-\|\Delta\|}$ , where  $\|\Delta\|$  is the Hamming weight of  $\Delta$  excluding the MSB.

### 5.1 Searching for good differences

In order to simplify the discussion we introduce the following function:

$$P(X) = \min\{i \mid \Delta X[i] \neq \text{'-'}\}.$$

In words,  $P(X)$  is the number of consecutive least significant bits of  $X$ , which *certainly* contain no difference. It is clear that  $P(X + Y) \geq \min(P(X), P(Y))$ , and  $P(X \gg \ell) = \max(P(X) - \ell, 0)$ . In the case of rotations, we have that if  $\ell \leq P(X)$ , then  $P(X \ggg \ell) = P(X) - \ell$ . For BMW-384/512, we have the following:

$$\begin{array}{ll} P(s_5(X)) = P(X) - 2, & \text{since } s_5(X) = X \ggg^2 \oplus X \\ P(s_4(X)) = P(X) - 1, & \text{since } s_4(X) = X \ggg^1 \oplus X \\ P(r_7(X)) = P(X) - 11, & \text{since } r_7(X) = X \lll^{53} = X \ggg^{11} \\ P(r_6(X)) = P(X) - 21, & \text{since } r_6(X) = X \lll^{43} = X \ggg^{21} \\ P(r_5(X)) = P(X) - 27, & \text{since } r_5(X) = X \lll^{37} = X \ggg^{27}. \end{array}$$

The last three identities are on the condition that  $\ell \leq P(X)$ , where  $\ell$  is the (right) rotation value.

As above, we assume that among  $\{Q_0, \dots, Q_{15}\}$ , only  $Q_0$  contains a difference, and among  $\{H_i\} \cup \{M_i\}$ , only  $H_1$  and  $M_1$  contain a difference. This happens if the differences in  $H_1$  and  $M_1$  are the same. Now we track the differences going into each of the first nine rounds of  $f_1$ . Below we have listed the (modified) input words that contain a difference in each round.

$$\begin{array}{ll} Q_{16} : & s_1(Q_0) \\ Q_{17} : & s_5(Q_{16}), M_1 \lll^2 \\ Q_{18} : & s_5(Q_{17}), s_4(Q_{16}) \\ Q_{19} : & s_5(Q_{18}), s_4(Q_{17}), r_7(Q_{16}) \\ Q_{20} : & s_5(Q_{19}), s_4(Q_{18}), r_7(Q_{17}), Q_{16} \\ Q_{21} : & s_5(Q_{20}), s_4(Q_{19}), r_7(Q_{18}), Q_{17}, r_6(Q_{16}) \\ Q_{22} : & s_5(Q_{21}), s_4(Q_{20}), r_7(Q_{19}), Q_{18}, r_6(Q_{17}), Q_{16} \\ Q_{23} : & s_5(Q_{22}), s_4(Q_{21}), r_7(Q_{20}), Q_{19}, r_6(Q_{18}), Q_{17}, r_5(Q_{16}), M_1 \lll^2 \\ Q_{24} : & s_5(Q_{23}), s_4(Q_{22}), r_7(Q_{21}), Q_{20}, r_6(Q_{19}), Q_{18}, r_5(Q_{17}), Q_{16} \end{array} \tag{9}$$

The goal is to find differences  $\Delta$  in  $Q_0$  such that the LSB of  $Q_i$ , for all  $i$ ,  $16 \leq i \leq 24$ , contains a strong bias. This bias is preferably in the form of a difference or no difference with probability 1. We now identify the minimum requirements on  $\Delta$  in order for this to happen. We assume the difference on  $H_1$  and  $M_1$  is also  $\Delta$ , i.e., that there is no propagation of bit differences in the feed forward of  $H_1$  in  $f_0$ .

We first find the bare requirements on  $Q_{16}$  in order to reach our goal. The round in which the  $P$ -value of  $Q_{16}$  drops the most is round 7 (computing  $Q_{23}$ ), in which  $r_5$  is computed on  $Q_{16}$ . This yields the requirement  $P(Q_{16}) \geq 27$ .

The requirements on  $Q_{17}$  are similarly found to be  $P(Q_{17}) \geq 27$ . This “updates” the requirement on  $Q_{16}$  due to the dependence of  $Q_{17}$  on  $Q_{16}$ , which means that we get  $P(Q_{16}) \geq 29$ .

If we continue like this, we find requirements on subsequent words of  $Q$ , which may iteratively require updates to requirements on previous words. The end result is that the requirement on  $Q_{16}$  becomes  $P(Q_{16}) \geq 32$  and the requirement on  $M_1$  is  $P(M_1) \geq 25$  combined with the requirement that there is no difference in

The function  $s_1$  can be described as a matrix multiplication over  $\mathbb{F}_2$ . The matrix  $S_1$  has 64 rows and columns, and the input  $x$  is viewed as a 64-bit column vector. Then we have  $s_1(x) = S_1 \cdot x$ . Searching for a good difference  $\Delta$  corresponds to finding the kernel of a submatrix  $\hat{S}_1$  of  $S_1$ , in which rows  $0, \dots, 31$  and columns  $0, 1$ , and  $39, \dots, 63$  are removed. Hence, we keep the columns corresponding to input bits that may contain a difference, and we keep the rows corresponding to output bits which must contain no difference. See Fig. 1.

Figure 1 shows a 20x20 grid representing the matrix  $\hat{S}_1$ . The grid is mostly filled with dots, with a diagonal line of '1's from the top-left to the bottom-right. The '1's are located at positions (i, i) for i from 1 to 20. The matrix is symmetric, as the '1's are also present at (i, i) for i from 20 down to 1.

The kernel of  $\hat{S}_1$  has dimension 5 and hence contains  $2^5 - 1 = 31$  non-zero vectors. Five basis vectors of the kernel correspond to the 64-bit words  $0204800008000000_h$ ,  $0102400004000000_h$ ,  $1004000040000000_h$ ,  $0081200002000000_h$ , and  $2401000090000000_h$ , and so any linear combination of these (except 0) can be used as a feed for  $\Delta$ . As an example, if we choose  $\Delta = 1004000040000000_h$  (and assuming  $\Delta$  is not changed by the feed-forward in  $f_0$ ), we have the following differences with probability 1 (the words  $Q_i$  for  $1 \leq i < 16$

contain no difference):

$$\begin{aligned}
\Delta Q_0 &= [---x-----x-----x-----] \\
\Delta Q_{16} &= [????????????????????x-----] \\
\Delta Q_{17} &= [????????????????????x-----] \\
\Delta Q_{18} &= [????????????????????x-----] \\
\Delta Q_{19} &= [????????????????????x-----] \\
\Delta Q_{20} &= [????????????????????x-----] \\
\Delta Q_{21} &= [????????????????????x-----] \\
\Delta Q_{22} &= [????????????????????x-----] \\
\Delta Q_{23} &= [????????????????????x-----] \\
\Delta Q_{24} &= [????????????????????x-----]
\end{aligned}$$

Hence,  $XL$  will be

$$[????????????????????x-----],$$

and from (7) we see that  $H_0^*[0]$  will contain no difference with probability 1.

For BMW-224/256, a similar investigation results in a solution space for  $\Delta$  of dimension 2, parametrised by the vectors  $08901000_h$  and  $20404000_h$ . As an example, with  $\Delta = 20404000_h$  we have the following differences with probability 1:

$$\begin{aligned}
\Delta Q_0 &= [--x-----x-----x-----] \\
\Delta Q_{16} &= [????????????????x-----] \\
\Delta Q_{17} &= [????????????????x-----] \\
\Delta Q_{18} &= [????????????????x-----] \\
\Delta Q_{19} &= [????????????????x-----] \\
\Delta Q_{20} &= [????????????????x-----] \\
\Delta Q_{21} &= [????????????????x-----] \\
\Delta Q_{22} &= [????????????????x-----] \\
\Delta Q_{23} &= [????????????????x--] \\
\Delta Q_{24} &= [????????????????x]
\end{aligned}$$

Hence,  $XL$  will be  $[????????????????x--]$ , and  $H_0^*[0]$  will contain a difference with probability 1. If we instead take  $\Delta$  to be the xor of the two basis vectors, then  $H_0^*[0]$  will contain *no* difference with probability 1.

## 6 The security parameter 2/14

The results described above cannot be directly extended to the security parameter 2/14. The reason is that the difference in  $Q_{16}$  goes through  $s_0$  instead of  $s_5$  in round 1.  $s_0$  is much more effective in spreading differences than  $s_5$ .

However, we observe that it is still possible if we are lucky (as attacker) enough to get the differences in some LSBs cancelled. Note that when the security parameter is 2/14 instead of 1/15, we have the same dependencies (see (9)) except that  $Q_{17}$  depends on  $s_0(Q_{16})$  instead of on  $s_5(Q_{16})$ . Hence, we may investigate whether the requirement  $P(Q_{17}) \geq 27$  that we found above holds for some  $\Delta$  among the 31 candidates mentioned above. Unfortunately, this is not the case.

Instead, we may allow differences in the 25 LSBs of  $Q_0$  and hope that the modular addition cancels the differences in the 27 LSBs of  $s_0(s_1(Q_0))$  and  $M_1^{\lll 2}$ , which are the only terms in the computation of  $Q_{17}$  that contain differences. We still need  $s_1(Q_0)$  to contain no difference in the 32 LSBs, and we also need  $M_1$  to have no difference in the two MSBs. So we search for  $\Delta$  so that  $s_0(s_1(\Delta))$  and  $\Delta^{\lll 2}$  agree in the 27 LSBs, and so that  $s_1(\Delta)$  has ‘0’ bits in the 32 LSBs and  $\Delta$  has ‘0’ bits in the two MSBs.

Let  $S_0$  and  $S_1$  denote the bit matrices corresponding to the functions  $s_0$  and  $s_1$ , and let  $R_2$  denote the bit matrix corresponding to the operation  $x^{\lll 2}$ . Let  $\Delta = s_1(\Delta)$ ; this means that we are interested in  $\Delta$  having 32 trailing ‘0’ bits, and such that  $S_0 \cdot \Delta$  and  $R_2 \cdot S_1^{-1} \cdot \Delta$  agree in the 27 LSBs (where  $\Delta$  in this case is viewed as a 64-bit column vector). Hence, similar to the situation above for the security parameter  $1/15$ , we are in fact interested in the kernel of a submatrix of  $S_0 - R_2 \cdot S_1^{-1}$ . The submatrix is the  $27 \times 32$  matrix where the last 32 columns and the first 37 columns are removed. Moreover, we need  $\Delta$  to be such that  $s_1^{-1}(\Delta)$  has ‘0’ bits in the two MSBs.

It turns out that the kernel of this submatrix has dimension 5 and is parametrised by the vectors that can be found in the table below, where also the corresponding  $\Delta$ s are listed.

$\Delta$	$\Delta = s_1^{-1}(\Delta)$
80D2227300000000 <sub>h</sub>	2B0D8FF05891139A <sub>h</sub>
48002F6000000000 <sub>h</sub>	29A78CAE96017B01 <sub>h</sub>
22C4DC6100000000 <sub>h</sub>	89ABBD3D9226E308 <sub>h</sub>
10D27CB300000000 <sub>h</sub>	784296AD7493E598 <sub>h</sub>
01201CFD00000000 <sub>h</sub>	28E58FDD2900E7E8 <sub>h</sub>

Clearly, there are 7 (non-zero) linear combinations that contain only ‘0’ bits in the two MSB positions and therefore admit a bias of the type ‘-’ or ‘x’ in  $H_0^*[0]$ . One of these ( $\Delta = 28E58FDD2900E7E8_h$ ) also admits this type of bias in  $H_0^*[1]$ . Moreover, among the remaining 24 non-zero linear combinations, there are 16 which admit a weaker bias in the sense that  $H_0^*[0]$  contains a difference with probability about  $3/8$  or  $5/8$  (i.e., a bias  $1/8$ , estimated from many experiments). Note that a difference in the two MSBs of  $M_1$  is no longer a problem in round 1, since we obtain the required difference in round 1 by having the differences in the 27 LSBs of  $s_0(Q_{16})$  and  $M_1^{\lll 2}$  cancel. This can be ensured through simple message modifications, as explained in the following.

First, we choose  $H_1 = M_1 = 0$ . Then we choose  $H_i$  and  $M_i$  at random,  $i \in \{0, 2, 3, \dots, 15\}$ . We then correct  $M_5$  such that  $Q_0 = 0$ . Hence,  $Q_0 \oplus \Delta = \Delta$ , and so all bit differences in  $Q_0$  are of the form  $0 \rightarrow 1$ . We then correct  $Q_8$  (through proper choice of  $H_9$  and  $M_9$ , without affecting other words) such that  $Q_{16} = 0$ . This ensures that there is no carry propagation after adding the difference  $\Delta$  on  $s_1(Q_0)$ . Hence, the difference on  $Q_{16}$  will be  $\Delta$  as required. This, in turn, means that  $s_0(Q_{16})$  will result in a difference that is the same as the difference on  $M_1^{\lll 2}$  in the 27 LSB positions. All bit differences in  $s_0(Q_{16})$  will be of the form  $0 \rightarrow 1$ . We can make the difference on  $M_1^{\lll 2}$  cancel the difference on  $s_0(Q_{16})$  (in the 27 LSBs) by making sure that all bit differences on  $M_1^{\lll 2}$  are of the form  $1 \rightarrow 0$ . This is ensured by correcting  $M_{11}$  so that  $AddElement(1) = 0$  and by choosing  $H_8 = \text{FFFFFFFFFFFFFFFF}_h$ . Note that this can be done in the very beginning, since these values do not depend on any values of  $Q$ . There are still many degrees of freedom left in the attack.

For BMW-224/256, we get the following three solutions:

$\Delta$	$\Delta = s_1^{-1}(\Delta)$
99108000 <sub>h</sub>	5CD58223 <sub>h</sub>
54E68000 <sub>h</sub>	6A2F79CC <sub>h</sub>
245B0000 <sub>h</sub>	872008B6 <sub>h</sub>

Only the xor of the first two basis vectors fulfils the requirement that the two MSBs of  $\Delta$  are ‘0’ bits. Using this value of  $\Delta$  (and with a similar message modification as above), one gets that the LSB of  $H^*[0]$  is always ‘-’. Four out of the remaining six non-zero linear combinations yield a difference in the same bit with probability  $3/8$  or  $5/8$  (again an estimate based on experiments).

**C program.** The differential properties described in this section are demonstrated in a C program available for download [4].

## 7 Potential Applications

In this section, we show how to convert the efficient differentials into pseudo-preimages of the compression function. To describe the attack, we consider a small ideal case: assume we have a set of differences  $D_1, D_2, D_3$  such that the differentials give  $[-x]$ ,  $[x-]$ , and  $[xx]$  on two output bits, respectively. Given any target  $T$ , we perform the pseudo-preimage attack as follows.



1. Randomly choose  $(H, M)$  from the set of inputs that fulfil the requirements for the differentials. Compute  $H^* = \text{bmw}_n(H, M)$ .
2. Compare  $H^*$  with  $T$  for the two bits.
3. If it gives  $[-]$ , further compare others bits;
4. else if it gives  $[-x]$ , compare  $\text{bmw}_n(H \oplus D_1, M \oplus D_1)$  with  $T$ ;
5. else if it gives  $[x-]$ , compare  $\text{bmw}_n(H \oplus D_2, M \oplus D_2)$  with  $T$ ;
6. else if it gives  $[xx]$ , compare  $\text{bmw}_n(H \oplus D_3, M \oplus D_3)$  with  $T$ .
7. Repeat steps 1-6 until a full match is found.

Note, steps 2-5 each gives a full match with probability  $2^{2-n'}$  (with  $n'$  the size of the chaining value). Hence, the expected time complexity is  $2^{n'-2} \times (1 + 3/4) \simeq 2^{n'-1.2}$ , with negligible memory requirements. More generally, if there are  $2^k - 1$  differences giving all possible  $2^k - 1$  probability 1 differentials on  $k$  output bits of the compression function, then the pseudo-preimage takes time about  $2^{n'-k} \cdot (2 - 2^{-k}) \simeq 2^{n'-k+1}$ .

In the case of BMW-512, we only have differences giving differentials on the 2 LSBs of  $H_0^*$  with  $[x-]$ ,  $[?x]$ , and  $[x?]$ . This can be converted into a pseudo-preimage of  $\text{bmw}_{512}$  in time  $2^{1023.2}$ .

An interesting problem here is to find more such differentials, such that the complexity could be further reduced. Moreover, if the differentials work on the lower half of the output bits (those to be taken as the output of the hash function), then the pseudo-preimage on the compression function can be further extended to a pseudo-preimage attack on the hash function.

## 8 Conclusion

We have described some deterministic differential properties for the BMW compression function with security parameters 0/16, 1/15 and 2/14: by choosing a certain xor difference in two input words to the compression function (and with conditions on absolute values of a few other words), a single (or a few) output bits of the compression function contain a difference with probability 0 or 1.

The differentials work for the compression function only, and do not affect the security of the hash function because of the additional blank invocation of the compression function before returning the hash output. Moreover,  $H_0^*$  is discarded in the final hash output, and only the least significant half (or less) bits of  $H^*$  of the final compression are taken.

Combining with more sophisticated message modification techniques, the differentials might be further extended to higher security parameters, hence increasing security parameter might not be enough to resist them. Tweaking the rotation values for the  $s_i$  and  $r_i$  functions may work, under the condition that the tweak does not affect other security properties.

Another interesting problem to consider is to devise differentials on other output words than merely  $H_0^*$ . In particular, a bias on one of the output words  $H_8^*, \dots, H_{15}^*$  would be interesting.

We note that tracing the propagation of differences, as done in this paper, might help to explain the distinguisher found by Aumasson [1].

**Acknowledgements.** We would like to thank Jean-Philippe Aumasson for his comments and help. The work in this paper is supported in part by the Singapore Ministry of Education under Research Grant T206B2204.

## References

1. J.-P. Aumasson. Practical distinguisher for the compression function of Blue Midnight Wish. Comment on the NIST Hash Competition, February 2010. Available: <http://131002.net/data/papers/Aum10.pdf>.
2. C. De Cannière and C. Rechberger. Finding SHA-1 Characteristics: General Results and Applications. In X. Lai and K. Chen, editors, *Advances in Cryptology – ASIACRYPT 2006, Proceedings*, volume 4284 of *Lecture Notes in Computer Science*, pages 1–20. Springer, 2006.
3. D. Gligoroski, V. Klíma, S. J. Knapskog, M. El-Hadedy, J. Amundsen, and S. F. Mjøl̂snes. Cryptographic hash function BLUE MIDNIGHT WISH. Submission to NIST (Round 2). Available: [http://people.item.ntnu.no/~danilog/Hash/BMW-SecondRound/Supporting\\_Documentation/BlueMidnightWishDocumentation.pdf](http://people.item.ntnu.no/~danilog/Hash/BMW-SecondRound/Supporting_Documentation/BlueMidnightWishDocumentation.pdf) (2010/03/22), September 2009.

4. J. Guo and S. S. Thomsen. C program that demonstrates the distinguisher. Available: <http://www2.mat.dtu.dk/people/S.Thomsen/bmw/bmw-distinguisher.zip>.
5. National Institute of Standards and Technology. Announcing Request for Candidate Algorithm Nominations for a New Cryptographic Hash Algorithm (SHA-3) Family. *Federal Register*, 27(212):62212–62220, November 2007. Available: [http://csrc.nist.gov/groups/ST/hash/documents/FR\\_Notice\\_Nov07.pdf](http://csrc.nist.gov/groups/ST/hash/documents/FR_Notice_Nov07.pdf) (2009/04/07).
6. I. Nikolić, J. Pieprzyk, P. Sokołowski, and R. Steinfeld. Rotational Cryptanalysis of (Modified) Versions of BMW and SIMD. Comment on the NIST Hash Competition, March 2010. Available: [https://cryptolux.org/mediawiki/uploads/0/07/Rotational\\_distinguishers\\_%28Nikolic%2C\\_Pieprzyk%2C\\_Sokolowski%2C\\_Steinfeld%29.pdf](https://cryptolux.org/mediawiki/uploads/0/07/Rotational_distinguishers_%28Nikolic%2C_Pieprzyk%2C_Sokolowski%2C_Steinfeld%29.pdf) (2010/03/22).
7. S. S. Thomsen. Pseudo-cryptanalysis of the Original Blue Midnight Wish. In S. Hong and T. Iwata, editors, *Fast Software Encryption*, LNCS, Seoul, South Korea, 2010. Springer. To appear.

## A Sub-functions used in $f_0$ and $f_1$

The sub-functions  $s_i$ ,  $0 \leq i \leq 4$ , and  $r_i$ ,  $1 \leq i \leq 7$ , used in  $f_0$  and  $f_1$  are defined as follows.

BMW-224/256	BMW-384/512
$s_0(x) = x \gg^1 \oplus x \ll^3 \oplus x \ll^4 \oplus x \ll^{19}$	$s_0(x) = x \gg^1 \oplus x \ll^3 \oplus x \ll^4 \oplus x \ll^{37}$
$s_1(x) = x \gg^1 \oplus x \ll^2 \oplus x \ll^8 \oplus x \ll^{23}$	$s_1(x) = x \gg^1 \oplus x \ll^2 \oplus x \ll^{13} \oplus x \ll^{43}$
$s_2(x) = x \gg^2 \oplus x \ll^1 \oplus x \ll^{12} \oplus x \ll^{25}$	$s_2(x) = x \gg^2 \oplus x \ll^1 \oplus x \ll^{19} \oplus x \ll^{53}$
$s_3(x) = x \gg^2 \oplus x \ll^2 \oplus x \ll^{15} \oplus x \ll^{29}$	$s_3(x) = x \gg^2 \oplus x \ll^2 \oplus x \ll^{28} \oplus x \ll^{59}$
$s_4(x) = x \gg^1 \oplus x$	$s_4(x) = x \gg^1 \oplus x$
$s_5(x) = x \gg^2 \oplus x$	$s_5(x) = x \gg^2 \oplus x$
$r_1(x) = x \ll^3$	$r_1(x) = x \ll^5$
$r_2(x) = x \ll^7$	$r_2(x) = x \ll^{11}$
$r_3(x) = x \ll^{13}$	$r_3(x) = x \ll^{27}$
$r_4(x) = x \ll^{16}$	$r_4(x) = x \ll^{32}$
$r_5(x) = x \ll^{19}$	$r_5(x) = x \ll^{37}$
$r_6(x) = x \ll^{23}$	$r_6(x) = x \ll^{43}$
$r_7(x) = x \ll^{27}$	$r_7(x) = x \ll^{53}$