The GFP-128 Library version 1.0

Email: witeg@mail.com
Homepage: http://www.witeg.prv.pl

Introduction:

The GFP-128 provides big integers modular arithmetic for ECP library.

Exported functions:

The **addmod** function adds two big integers modulo constant:

VOID addmod(
[IN] **BYTE** **pbBigIntA*
[IN] **BYTE** **pbBigIntB*
[OUT] **BYTE** **pbBigIntC*
);

Parameters:
pbBigIntA
The address of big integer *A*.
pbBigIntB
The address of big integer *B*.
pbBigIntC
The address of the buffer to receive big integer *C=A+B mod const.*

Return Value:
This function does not return a value.

The **adduintmod** function adds unsigned integer (dword) to big integer modulo constant:

VOID adduintmod(
[IN] **BYTE** **pbBigIntA*
[IN] **UINT** *dwB*
[OUT] **BYTE** **pbBigIntC*
);

Parameters:
pbIntegerA
The address of big integer *A*.
dwB
Unsigned integer (dword) *B*
pbIntegerC
The address of the buffer to receive big integer *C=A+B mod const.*

Return Value:
This function does not return a value.

The **compare** function comapres two big integers:

VOID compare(
[IN] **BYTE** **pbBigIntA*
[IN] **BYTE** **pbBigIntB*
);

Parameters:
pbBigIntA
The address of big integer *A.*
pbBigIntB
The address of big integer *B.*

Return Value:
This function does not return a value. Comparision result is returned directly in EFLAG register.

The **comparezero** function compares a big integer with zero:

VOID comparezero(
[IN] **BYTE** **pbBigInt*
);

Parameters:
pbBigIntA
The address of big integer to be compared.

Return Value:
This function does not return a value. Comparision result is returned directly in EFLAG register.

The **compareone** function compares a big integer with one:

VOID compareone(
[IN] **BYTE** **pbBigInt*
);

Parameters:
pbBigInt
The address of big integer to be compared.

Return Value:
This function does not return a value. Comparision result is returned directly in EFLAG register.

The **converth2bmod** function converts hash value to big integer modulo constant:

VOID converth2bmod(
[IN] **BYTE** **pbHash*
[OUT] **BYTE** **pbBigInt*
);

Parameters:
pbHash
The address of a SHA1 hash (message digest)
pbBigInt
The address of the buffer to receive big integer

Return Value:
This function does not return a value.

The **copy** function copies one big integer to another:

VOID copy(
[IN] **BYTE** **pbBigIntA*
[OUT] **BYTE** ***pbBigIntB*
);

Parameters:
pbBigIntA
The address of big integer *A* to be copied..
pbBigIntB
The address of the buffer to receive big integer *A*.

Return Value:
This function does not return a value.

The **div2** function divides a big integer by two:

VOID div2(
[IN/OUT] **BYTE** **pbBigIntA*
);

Parameters:
pbBigIntA
The address of big integer *A* to be divided. On exit *A = A/2.*

Return Value:
This function does not return a value. The remainder is returned directly in the CPU carry flag (CF is set if remainder is equal to one).

The **div2mod** function divides a big integer by two modulo constant:

VOID div2mod(
[IN/OUT] **BYTE** **pbBigIntA*
);

Parameters:
pbBigIntA
The address of big integer *A* to be divided. On exit *A = A/2 mod const.*

Return Value:
This function does not return a value.

The **fixmod** function computes a value of 128bit big integer modulo constant:

VOID fixmod(
[IN/OUT] **BYTE** **pbBigIntA*
);

Parameters:
pbBigIntA
The address of big integer *A* to be reduced. On exit *A = A mod const..*

Return Value:
This function does not return a value.

The **invmod** function computes modular inverse of big integer modulo constant:

VOID invmod(
[IN/OUT] **BYTE** **pbBigIntA*
);

Parameters:
pbBigIntA
The address of big integer *A* to inverse. On exit *A = 1/A mod const.*

Return Value:
This function does not return a value.

The **modulo** function computes the remainder when a product of two big integers is divided by modulo constant:

VOID modulo(
[IN] **BYTE** **pbVBigInt*
[OUT] **BYTE** **pbBigInt*
);

Parameters:
pbVBigInt
The address of very big (256bit) integer to reduce.
pbBigInt
The address of the buffer to receive the remainder - big integer.

Return Value:
This function does not return a value.

The **mulmod** function multiply two big integers modulo constant:

VOID mulmod(
[IN] **BYTE** **pbBigIntA*
[IN] **BYTE** **pbBigIntB*
[OUT] **BYTE** **pbBigIntC*
);

Parameters:
pbBigIntA
The address of big integer *A*.
pbBigIntB
The address of big integer *B*.
pbBigIntC
The address of the buffer to receive big integer *C=A*B mod const.*

Return Value:
This function does not return a value.

The **multiply** function multiply two big integers:

VOID multiply(
[IN] **BYTE** **pbBigIntA*
[IN] **BYTE** **pbBigIntB*
[OUT] **BYTE** **pbVBigIntC*
);

Parameters:
pbBigIntA
The address of big integer *A*.
pbBigIntB
The address of big integer *B*.
pbBigIntC
The address of the buffer to receive very big (256bit) integer *C=A*B.*

Return Value:
This function does not return a value.

The **setmod** function sets modulus:

VOID setmod(
[IN] **BYTE** **pbBigIntMod*
);

Parameters:
pbBigIntMod
The address of modulus.

Return Value:
This function does not return a value.

The **submod** function subtracts two big integers modulo constant:

VOID submod(
[IN] **BYTE** **pbBigIntA*
[IN] **BYTE** **pbBigIntB*
[OUT] **BYTE** **pbBigIntC*
);

Parameters:
pbBigIntA
The address of big integer *A*.
pbBigIntB
The address of big integer *B*.
pbBigIntC
The address of the buffer to receive big integer *C=A-B mod const.*

Return Value:
This function does not return a value.

The **zero** function sets a big integer to zero:

VOID zero(
[OUT] **BYTE** **pbBigInt*
);

Parameters:
pbBigInt
The address of big integer.

Return Value:
This function does not return a value.

License :

"Software" means the program supplied by WiteG herewith.

Permission is hereby granted to any individual, organization or agency to use the Software for any legal NON-COMMERCIAL purpose, without any obligation to the author. You may distribute the Software freely, provided that the original distribution package (binaries and any other files included in it) is left intact. You may also disassemble, reverse engineer or modify the Software, but you MAY NOT distribute it in modified form.
Any COMMERCIAL use of the Software without commercial license obtained from the author is strictly prohibited.

Warranty:

This software is provided by WiteG as-is, without warranty of ANY KIND, either expressed or implied, including but not limited to the implied warranties of merchantability and/or fitness for a particular purpose. The author shall NOT be held liable for ANY damage to you, your computer, or to anyone or anything else, that may result from its use, or misuse. Use it at YOUR OWN RISK.
However, the author of this software does hereby declare that there are no hidden weaknesses or trapdoors inserted by him.

Please report any bugs, comments, questions or suggestions to me