# Arbitrary-Precision Arithmetic Library Comparison

September 9, 2011

## 1 BeeCrypt

http://beecrypt.sourceforge.net/

```
struct _mpnumber
#endif
{
        size_t  size;
        mpw*    data;
}

#if (MP_WBITS == 64)
typedef uint64_t        mpw;
typedef uint32_t        mphw;
#elif (MP_WBITS == 32)
# if HAVE_UINT64_T
#  define HAVE_MPDW 1
typedef uint64_t        mpdw;
# endif
typedef uint32_t        mpw;
typedef uint16_t        mphw;
#else
# error
#endif
```

## 2 GMP

http://gmplib.org/

```
struct
{
  int _mp_alloc;    /* Number of *limbs* allocated and pointed
```

```
                      to by the _mp_d field.  */
  int _mp_size;     /* abs(_mp_size) is the number of limbs the
                      last field points to.  If _mp_size is
                      negative this is a negative number.  */
  mp_limb_t *_mp_d; /* Pointer to the limbs.  */
} __mpz_struct;

typedef __mpz_struct mpz_t[1];

#ifdef __GMP_SHORT_LIMB
typedef unsigned int            mp_limb_t;
typedef int                     mp_limb_signed_t;
#else
#ifdef _LONG_LONG_LIMB
typedef unsigned long long int  mp_limb_t;
typedef long long int           mp_limb_signed_t;
#else
typedef unsigned long int       mp_limb_t;
typedef long int                mp_limb_signed_t;
#endif
#endif
```

# 3 IMath

http://spinning-yarns.org/michael/sw/imath/

```
typedef struct mpz {
  mp_digit    single;
  mp_digit   *digits;
  mp_size     alloc;
  mp_size     used;
  mp_sign     sign;
} mpz_t, *mp_int;

typedef unsigned char      mp_sign;
typedef unsigned int       mp_size;
typedef int                mp_result;
typedef long               mp_small;  /* must be a signed type */
typedef unsigned long      mp_usmall; /* must be an unsigned type */
#ifdef USE_LONG_LONG
typedef unsigned int       mp_digit;
typedef unsigned long long mp_word;
#else
typedef unsigned short     mp_digit;
typedef unsigned int       mp_word;
#endif
```

# 4 libgcrypt

http://www.gnupg.org/

```
#ifndef BITS_PER_MPI_LIMB
#if BYTES_PER_MPI_LIMB == SIZEOF_UNSIGNED_INT
  typedef unsigned int mpi_limb_t;
  typedef   signed int mpi_limb_signed_t;
#elif BYTES_PER_MPI_LIMB == SIZEOF_UNSIGNED_LONG
  typedef unsigned long int mpi_limb_t;
  typedef   signed long int mpi_limb_signed_t;
#elif BYTES_PER_MPI_LIMB == SIZEOF_UNSIGNED_LONG_LONG
  typedef unsigned long long int mpi_limb_t;
  typedef   signed long long int mpi_limb_signed_t;
#elif BYTES_PER_MPI_LIMB == SIZEOF_UNSIGNED_SHORT
  typedef unsigned short int mpi_limb_t;
  typedef   signed short int mpi_limb_signed_t;
#else
#error BYTES_PER_MPI_LIMB does not match any C type
#endif
#define BITS_PER_MPI_LIMB    (8*BYTES_PER_MPI_LIMB)
#endif /*BITS_PER_MPI_LIMB*/

struct gcry_mpi
{
  int alloced;        /* Array size (# of allocated limbs). */
  int nlimbs;         /* Number of valid limbs. */
  int sign;           /* Indicates a negative number and is also used
                         for opaque MPIs to store the length.  */
  unsigned int flags; /* Bit 0: Array to be allocated in secure memory space.*/
                      /* Bit 2: the limb is a pointer to some m_alloced data.*/
  mpi_limb_t *d;      /* Array with the limbs */
};
```

# 5 libtommath

https://github.com/libtom/libtommath

```
/* NB: mp_digit type definition is too cumbersome */

/* the infamous mp_int structure */
typedef struct {
    int used, alloc, sign;
    mp_digit *dp;
} mp_int;
```

# 6 LiDIA

http://www.cdc.informatik.tu-darmstadt.de/TI/LiDIA/

```
typedef unsigned long   DigitType;
#define BitsPerDigit    (sizeof(DigitType)*8)

typedef struct {
    DigitType   *vec;
    int         maxlength, length, sign;
}               Integer, *pInteger;
```

# 7 MAMP

http://www.tc.umn.edu/ ringx004/mapm-main.html

```
typedef struct  {
    UCHAR   *m_apm_data;
    long    m_apm_id;
    int     m_apm_refcount;       /* <- used only by C++ MAPM class */
    int m_apm_malloclength;
    int m_apm_datalength;
    int m_apm_exponent;
    int m_apm_sign;
} M_APM_struct;

typedef M_APM_struct *M_APM;
```

# 8 MPI

http://spinning-yarns.org/michael/mpi/

```
typedef char                mp_sign;
typedef unsigned short      mp_digit;  /* 2 byte type */
typedef unsigned int        mp_word;   /* 4 byte type */
typedef unsigned int        mp_size;

typedef struct {
  mp_sign       sign;    /* sign of this quantity      */
  mp_size       alloc;   /* how many digits allocated  */
  mp_size       used;    /* how many digits used       */
  mp_digit     *dp;      /* the digits themselves      */
} mp_int;
```