Count the number of bit

```
//http://extensionlibrary.codeplex.com/
//The MIT License (MIT)
using System;
using System.Collections.Generic;
using System.Text;

namespace ExtensionLibrary.Tools
{
public static class BitOperator
{
#region Count the number of bit one

public static int GetCountOfBitOne(sbyte x)
{
int result = 0;
while (x != 0)
{
result++;
x &= (sbyte)(x – 1);
}
return result;
}

public static int GetCountOfBitOne(short x)
{
int result = 0;
while (x != 0)
{
result++;
x &= (short)(x – 1);
}
return result;
}

public static int GetCountOfBitOne(int x)
```

```
{
int result = 0;
while (x != 0)
{
result++;
x &= (x – 1);
}
return result;
}

public static int GetCountOfBitOne(long x)
{
int result = 0;
while (x!=0)
{
result++;
x &= (x – 1);
}
return result;
}

public static int GetCountOfBitOne(byte x)
{
int result = 0;
while (x != 0)
{
result++;
x &= (byte)(x – 1);
}
return result;
}

public static int GetCountOfBitOne(ushort x)
{
int result = 0;
while (x != 0)
{
result++;
```

```
x &= (ushort)(x – 1);
}
return result;
}

public static int GetCountOfBitOne(uint x)
{
int result = 0;
while (x != 0)
{
result++;
x &= (x – 1);
}
return result;
}

public static int GetCountOfBitOne(ulong x)
{
int result = 0;
while (x != 0)
{
result++;
x &= (x – 1);
}
return result;
}

#endregion

#region Count the number of bit zero

public static int GetCountOfBitZero(sbyte x)
{
return GetCountOfBitOne(~x);
}

public static int GetCountOfBitZero(short x)
{
return GetCountOfBitOne(~x);
```

```
}

public static int GetCountOfBitZero(int x)
{
return GetCountOfBitOne(~x);
}

public static int GetCountOfBitZero(long x)
{
return GetCountOfBitOne(~x);
}

public static int GetCountOfBitZero(byte x)
{
return GetCountOfBitOne(~x);
}

public static int GetCountOfBitZero(ushort x)
{
return GetCountOfBitOne(~x);
}

public static int GetCountOfBitZero(uint x)
{
return GetCountOfBitOne(~x);
}

public static int GetCountOfBitZero(ulong x)
{
return GetCountOfBitOne(~x);
}

#endregion

#region Get number of leading zero

public static int GetNumberOfLeadingZero(sbyte x)
{
int number = 8;
```

```
sbyte y = (sbyte)(x >> 4);
if (y != 0)
{
number -= 4;
x = y;
}

y = (sbyte)(x >> 2);
if (y != 0)
{
number -= 2;
x = y;
}

y = (sbyte)(x >> 1);
if (y != 0)
{
return number – 2;
}

return number – x;
}

public static int GetNumberOfLeadingZero(short x)
{
int number = 16;

short y = (short)(x >> 8);
if (y != 0)
{
number -= 8;
x = y;
}

y = (short)(x >> 4);
if (y != 0)
{
number -= 4;
x = y;
```

```
}

y = (short)(x >> 2);
if (y != 0)
{
number -= 2;
x = y;
}

y = (short)(x >> 1);
if (y != 0)
{
return number – 2;
}

return number – x;
}

public static int GetNumberOfLeadingZero(int x)
{
int number = 32;

int y = (x >> 16);
if (y != 0)
{
number -= 16;
x = y;
}

y = (x >> 8);
if (y != 0)
{
number -= 8;
x = y;
}

y = (x >> 4);
if (y != 0)
{
```

```
number -= 4;
x = y;
}

y = (x >> 2);
if (y != 0)
{
number -= 2;
x = y;
}

y = (x >> 1);
if (y != 0)
{
return number – 2;
}

return number – x;
}

public static int GetNumberOfLeadingZero(long x)
{
int number = 64;

long y = (x >> 32);
if (y != 0)
{
number -= 32;
x = y;
}

y = (x >> 16);
if (y != 0)
{
number -= 16;
x = y;
}

y = (x >> 8);
```

```
if (y != 0)
{
number -= 8;
x = y;
}

y = (x >> 4);
if (y != 0)
{
number -= 4;
x = y;
}

y = (x >> 2);
if (y != 0)
{
number -= 2;
x = y;
}

y = (x >> 1);
if (y != 0)
{
return number – 2;
}

return (int)(number – x);
}

public static int GetNumberOfLeadingZero(byte x)
{
int number = 8;

byte y = (byte)(x >> 4);
if (y != 0)
{
number -= 4;
x = y;
}
```

```
y = (byte)(x >> 2);
if (y != 0)
{
number -= 2;
x = y;
}

y = (byte)(x >> 1);
if (y != 0)
{
return number – 2;
}

return number – x;
}

public static int GetNumberOfLeadingZero(ushort x)
{
int number = 16;

ushort y = (ushort)(x >> 8);
if (y != 0)
{
number -= 8;
x = y;
}

y = (ushort)(x >> 4);
if (y != 0)
{
number -= 4;
x = y;
}

y = (ushort)(x >> 2);
if (y != 0)
{
number -= 2;
x = y;
```

```
}

y = (ushort)(x >> 1);
if (y != 0)
{
return number – 2;
}

return number – x;
}

public static int GetNumberOfLeadingZero(uint x)
{
int number = 32;

uint y = (x >> 16);
if (y != 0)
{
number -= 16;
x = y;
}

y = (x >> 8);
if (y != 0)
{
number -= 8;
x = y;
}

y = (x >> 4);
if (y != 0)
{
number -= 4;
x = y;
}

y = (x >> 2);
if (y != 0)
{
```

```
number -= 2;
x = y;
}

y = (x >> 1);
if (y != 0)
{
return number – 2;
}

return (int)(number – x);
}

public static int GetNumberOfLeadingZero(ulong x)
{
int number = 64;

ulong y = (x >> 32);
if (y != 0)
{
number -= 32;
x = y;
}

y = (x >> 16);
if (y != 0)
{
number -= 16;
x = y;
}

y = (x >> 8);
if (y != 0)
{
number -= 8;
x = y;
}

y = (x >> 4);
```

```
if (y != 0)
{
number -= 4;
x = y;
}

y = (x >> 2);
if (y != 0)
{
number -= 2;
x = y;
}

y = (x >> 1);
if (y != 0)
{
return number – 2;
}

return number – (int)x;
}

#endregion

#region Get number of leading one

public static int GetNumberOfLeadingOne(sbyte x)
{
return GetNumberOfLeadingZero(~x);
}

public static int GetNumberOfLeadingOne(short x)
{
return GetNumberOfLeadingZero(~x);
}

public static int GetNumberOfLeadingOne(int x)
{
return GetNumberOfLeadingZero(~x);
```

```
}

public static int GetNumberOfLeadingOne(long x)
{
return GetNumberOfLeadingZero(~x);
}

public static int GetNumberOfLeadingOne(byte x)
{
return GetNumberOfLeadingZero(~x);
}

public static int GetNumberOfLeadingOne(ushort x)
{
return GetNumberOfLeadingZero(~x);
}

public static int GetNumberOfLeadingOne(uint x)
{
return GetNumberOfLeadingZero(~x);
}

public static int GetNumberOfLeadingOne(ulong x)
{
return GetNumberOfLeadingZero(~x);
}

#endregion

#region Get number of tailing zero

public static int GetNumberOfTailingZero(sbyte x)
{
if (x == 0)
return 8;
int number = 7;
sbyte y = (sbyte)(x <
```