

Forum: [Hack Forums](#)

Thread: [Small collection of checksum algorithms \[Xor8, Adler, Fletcher, SysV32, Bsd32\]](#)

Barret - February 3, 2017, 8:48 am

These algorithms have been ported to vb.net and come from all over the net, mainly wiki, if you are looking for quick dirty checksum functions on a byte array.

XOR8

Code ''' <summary>

''' The longitudinal redundancy check (LRC) or horizontal redundancy check that is applied independently to each of a parallel group of bit streams

''' </summary>

''' <returns>UInt16</returns>

Public Shared Function Xor8(buffer() As Byte) As UInt16

Dim lrc As UInt16 = 0

For i As Integer = 0 To buffer.Length - 1

lrc = Convert.ToUInt16((lrc + buffer(i)) And &HFF)

Next

Return Convert.ToUInt16((((lrc Xor &HFF) + 1) And &HFF))

End Function

ADLER (16/32/64)

Code ''' <summary>

''' The Adler checksum algorithm was invented by Mark Adler in 1995 and is a modification of the Fletcher checksum.

''' </summary>

''' <returns>UInt16</returns>

Public Shared Function Adler16(buffer() As Byte) As UInt16

Dim a As UInt16 = 0, b As UInt16 = 0

For i As Integer = 0 To buffer.Length - 1

a = Convert.ToUInt16((a + buffer(i)) Mod &HFFF1)

b = Convert.ToUInt16((b + a) Mod &HFFF1)

Next

Return (b << 8) Or a

End Function

''' <summary>

''' The Adler checksum algorithm was invented by Mark Adler in 1995 and is a modification of the Fletcher checksum.

''' </summary>

''' <returns>UInt32</returns>

Public Shared Function Adler32(buffer() As Byte) As UInt32

Dim a As UInt32 = 0, b As UInt32 = 0

For i As Integer = 0 To buffer.Length - 1

a = Convert.ToUInt32((a + buffer(i)) Mod &HFFF1)

b = Convert.ToUInt32((b + a) Mod &HFFF1)

Next

Return (b << 16) Or a

End Function

```
''' <summary>
```

```
''' The Adler checksum algorithm was invented by Mark Adler in 1995 and is a modification of the Fletcher checksum.
```

```
''' </summary>
```

```
''' <returns>UInt64</returns>
```

```
Public Shared Function Adler64(buffer() As Byte) As UInt64
```

```
    Dim a As UInt64 = 0, b As UInt64 = 0
```

```
    For i As Integer = 0 To buffer.Length - 1
```

```
        a = Convert.ToUInt64((a + buffer(i)) Mod &HFFF1)
```

```
        b = Convert.ToUInt64((b + a) Mod &HFFF1)
```

```
    Next
```

```
    Return (b << 32) Or a
```

```
End Function
```

```
FLETCHER (16/32/64)
```

```
Code ''' <summary>
```

```
''' The Fletcher checksum is an algorithm for computing a position-dependent checksum devised by John G. Fletcher
```

```
''' </summary>
```

```
''' <returns>UInt16</returns>
```

```
Public Shared Function Fletcher16(buffer() As Byte) As UInt16
```

```
    Dim a As UInt16 = 0, b As UInt16 = 0
```

```
    For i As Integer = 0 To buffer.Length - 1
```

```
        a = Convert.ToUInt16((a + buffer(i)) Mod &HFF)
```

```
        b = Convert.ToUInt16((b + a) Mod &HFF)
```

```
    Next
```

```
    Return (b << 8) Or a
```

```
End Function
```

```
''' <summary>
```

```
''' The Fletcher checksum is an algorithm for computing a position-dependent checksum devised by John G. Fletcher
```

```
''' </summary>
```

```
''' <returns>UInt23</returns>
```

```
Public Shared Function Fletcher32(buffer() As Byte) As UInt32
```

```
    Dim a As UInt32 = 0, b As UInt32 = 0
```

```
    For i As Integer = 0 To buffer.Length - 1
```

```
        a = Convert.ToUInt32((a + buffer(i)) Mod &HFF)
```

```
        b = Convert.ToUInt32((b + a) Mod &HFF)
```

```
    Next
```

```
    Return (b << 16) Or a
```

```
End Function
```

```
''' <summary>
```

```
''' The Fletcher checksum is an algorithm for computing a position-dependent checksum devised by John G. Fletcher
```

```
''' </summary>
```

```
''' <returns>UInt64</returns>
```

```
Public Shared Function Fletcher64(buffer() As Byte) As UInt64
```

```
    Dim a As UInt64 = 0, b As UInt64 = 0
```

```
    For i As Integer = 0 To buffer.Length - 1
```

```
        a = Convert.ToUInt64((a + buffer(i)) Mod &HFF)
```

```
        b = Convert.ToUInt64((b + a) Mod &HFF)
```

```
    Next
```

```

    Return (b << 32) Or a
End Function
SYSV32
Code ''' <summary>
''' The SYSV checksum algorithm is a commonly used, legacy checksum algorithm.
''' </summary>
''' <returns>UInt32</returns>
Public Shared Function SysV32(buffer() As Byte) As UInt32
    Dim a As UInt32 = 0, b As UInt32 = 0
    Do While b <= buffer.Length - 1
        a = Convert.ToUInt32((a + buffer(Convert.ToInt32(b))) Mod &HFFFF)
        b = Convert.ToUInt32(b + 1)
    Loop
    Return (a << 32) Or a
End Function
BSD32
Code ''' <summary>
''' The BSD checksum algorithm is a commonly used, legacy checksum algorithm
''' </summary>
''' <returns>UInt32</returns>
Public Shared Function Bsd32(buffer() As Byte) As UInt32
    Dim sum As UInt32 = 0, count As UInt32 = 0
    Do While count <= buffer.Length - 1
        sum = Convert.ToUInt32((sum >> 1) + &H8000 * (sum And 1))
        sum = Convert.ToUInt32((sum + buffer(Convert.ToInt32(count))) And &HFFFF)
        count = Convert.ToUInt32(count + 1)
    Loop
    Return sum
End Function

```

Reason: After hours of f*%#\$ around with Mono.Cecil i was so fed up with not accomplishing anything, i had to make something that did work to feel better about myself. lol

fec-ri.muhayyel - March 8, 2019, 9:32 am

good subject but how i encrypt/decrypt words? how to use? give me a example please
thanks

gigajew - March 8, 2019, 3:24 pm

You don't encrypt/decrypt with these checksum calculators, it just gives you a unique number basically.

0xDEAD10CC - March 10, 2019, 11:58 am

(Mar 8, 2019 10:32 AM)fec-ri.muhayyel Wrote: [\[CITE\]](#) good subject but how i encrypt/decrypt words?
how to use? give me a example please
thanks

Quite a few of these, including pretty much all other checksums are only intended for integrity checks.