zyl910

优化技巧、硬件体系、图像处理、图形学、游戏编程、国际化与文本信息处理。

[C] 让VC支持C99的整数类型V1.01。避免包含目录问题，更名auto_stdint.h、auto_inttypes.h（在VC6至VC2012、GCC、BCB等编译器下测试通过）

作者：zyl910

以前我曾为了让VC++等编译器支持C99的整数类型，编写了同名的stdint.h、inttypes.h来智能处理（http://www.cnblogs.com/zyl910/archive/2012/08/08/c99int.html）。现在将其升级到v1.01版。

一、改动说明

1.1 包含目录问题

在1.00版，我编写的头文件与系统头文件同名，利用“#include "XXX"”与“#include <XXX>”的区别，使其智能使用系统头文件。
这样做的优点是基本不需改动代码（只需将“#include <stdint.h>”改为“#include "stdint.h"”），而且易读性好，一看就知道是C99整数类型。

后来使用时发现该方案存在包含目录问题——我的stdint.h、inttypes.h不能放在项目include目录中。
这是因为“#include <XXX>”时会优先检查项目include目录，而后才是系统include目录。
如果将我的stdint.h、inttypes.h放在项目include目录中，会导致循环引用，无法定位到系统头文件。
所以，只能将我的stdint.h、inttypes.h放在项目src目录。
当项目较大时，会建立多级子目录来存放源码。这时只有将stdint.h、inttypes.h复制到各个子目录中，管理起来不方便。

于是我决定将我的stdint.h、inttypes.h分辨改名为auto_stdint.h、auto_inttypes.h，这样就可以放在项目include目录中了。缺点是使用时麻烦一点，要包含auto_stdint.h、auto_inttypes.h。

VC(39)
x86(25)
gcc(23)
Cpp(23)
Asm(18)
AVC(17)
H.264(17)
SSE(15)
cpu(15)
SIMD(14)
更多

积分与排名
积分 - 365375
排名 - 1557

随笔分类 (1164)
--- Best_重要的(32)
--- Bug_故障排除(1)
--- My_原创(152)
--- MyTip_小技巧(9)
--- Program_编程(93)
--- Public_发布(18)
--- Release_软件发布(3)
--- Tool_工具 (15)
--- Translation_翻译(1)
A00 Math_数学(2)
AA0 Algebra_代数(1)
AA2 Linear_线性代数(1)
AP0 Physical_物理(1)
APM Mechanics_力学(1)
B00 Optimization_优化技巧(17)
B40 Math_数学(4)
B44 Bit_位运算(8)
B60 SIMD(6)
C00 Language_语言(70)
C10 C系列(66)
C11 C(46)
C12 C++(23)
C13 C#(17)
C14 Java(6)

既然文件改名了，宏也要改名——
__AUTO_STDINT_H_INCLUDED（原_STDINT_H_ALL_）。
__AUTO_STDINT_H_USESYS（原_STDINT_H_SYS_）。
__AUTO_INTTYPES_H_INCLUDED（原_INTTYPES_H_ALL_）。
__AUTO_INTTYPES_H_USESYS（原_INTTYPES_H_SYS_）。

1.2 编译器兼容性

测试了 Visual C++ 2008。发现它果然不支持stdint.h与inttypes.h。
测试了 Visual C++ 2012。发现它支持stdint.h，仍不支持inttypes.h。

二、全部代码

文件清单——
auto_inttypes.h
auto_stdint.h
c99int.c
c99int.dsp
c99int.dsw
c99int_2003.sln
c99int_2003.vcproj
c99int_2005.sln
c99int_2005.vcproj
c99int_2008.sln
c99int_2008.vcproj
c99int_2010.sln
c99int_2010.vcxproj
c99int_2010.vcxproj.filters
c99int_2010.vcxproj.user
c99int_2012.sln
c99int_2012.vcxproj
c99int_2012.vcxproj.filters
c99int_bcb.bpf

C15 Objective-C(6)
C20 BASIC系列(4)
C22 VB(3)
C23 VBScript(1)
C50 Script0_脚本语言0(1)
C60 ScriptWeb_Web脚本语言(7)
C61 JavaScript(10)
CQ0 SQL_结构化查询语言(1)
CQ1 SQLServer(1)
D00 Platform_平台(45)
D20 DOS/BIOS(2)
D30 Windows(33)
D32 Win32 API(3)
D32. kernel32(2)
D33 COM(1)
D34 .Net(15)
更多

随笔档案 (191)
2020年8月(1)
2020年7月(2)
2018年8月(1)
2018年5月(2)
2018年4月(1)
2018年3月(1)
2018年2月(2)
2018年1月(2)
2017年12月(1)
2017年11月(1)
2017年10月(2)
2017年9月(3)
2017年7月(1)
2017年3月(1)
2016年10月(1)
2016年2月(1)
2016年1月(1)
2015年8月(1)
2015年7月(2)
2015年5月(1)
2014年2月(2)
2013年12月(1)
2013年8月(3)

c99int_bcb.bpr
c99int_bcb.res
makefile

2.1 auto_stdint.h

全部代码——

```
//////////////////////////////////////////////////////////////
/*
auto_stdint.h: 兼容C99标准的stdint.h
Author: zyl910
Blog: http://www.cnblogs.com/zyl910
URL: http://www.cnblogs.com/zyl910/archive/2013/01/10/c99int_v101.html
Version: V1.01
Updata: 2013-01-10

测试过的编译器--
VC: 6, 2003, 2005, 2008, 2010, 2012.
BCB: 6.
GCC(Linux): 4.7.0(Fedora 17).
GCC(Mac): llvm-gcc-4.2(Mac OS X Lion 10.7.4, Xcode 4.4.1).
GCC(MinGW): 4.6.2(MinGW(20120426)), 4.7.1(TDM-GCC(MinGW-w64)).


Update
~~~~~~

[2013-01-01] V1.01
* 检查了对VC2008、VC2012的兼容性. 确认VC2008不支持stdint.h.
* 为了避免包含目录问题，更名auto_stdint.h（原stdint.h）.
* 更改宏名：__AUTO_STDINT_H_INCLUDED（原_STDINT_H_ALL_），__AUTO_STDINT_H_USESYS（原_STDINT_H_SYS_）.

[2012-08-08] V1.0
* V1.0发布.
* 参考了 msinttypes-r26. http://code.google.com/p/msinttypes/
* 修正了 VC6编译C++程序时wchar.h会报错 问题.
```

2013年7月(1)
2013年6月(3)
2013年4月(3)
2013年3月(1)
2013年1月(6)
2012年11月(5)
2012年10月(6)
2012年9月(6)
2012年8月(8)
2012年7月(7)
2012年6月(2)
2012年5月(9)
2012年4月(8)
2012年3月(9)
2012年2月(7)
2012年1月(7)
2011年12月(10)
更多

My

My.Blog.csdn
My.ebook.netyi
My.ebook.iask
My.Blog.cnblogs

最新评论

1. Re:[C] 跨平台使用TCHAR——让Linux等平台也支持tchar.h，解决跨平台时的格式控制字符问题，多国语言的同时显示（兼容vc/gcc/bcb，支持Windows/Linux/Mac)

真硬核啊，像现在调参侠满天飞的时代，还有人耐下新来做这样的code吗

--Bamboo123

2. Re:SIMD（MMX/SSE/AVX）变量命名规范心得

...

--marklove

```
*/
//////////////////////////////////////////////////////////////////////////

#ifndef __AUTO_STDINT_H_INCLUDED
#define __AUTO_STDINT_H_INCLUDED

// __AUTO_STDINT_H_USESYS: 编译器是否提供了<stdint.h>
#undef __AUTO_STDINT_H_USESYS
#if defined(__GNUC__)    // GCC.
    #define __AUTO_STDINT_H_USESYS
#elif defined(_MSC_VER)    // MSVC. VC6至VC2008均没有，从VC2010才支持的.
    #if _MSC_VER >=1600    // VC2010
        #define __AUTO_STDINT_H_USESYS
    #endif    // #if _MSC_VER >=1600    // VC2010
#elif defined(__BORLANDC__)    // BCB. BCB6是支持的.
    #if __BORLANDC__ >=0x0560    // BCB6
        #define __AUTO_STDINT_H_USESYS
    #endif    // #if __BORLANDC__ >=0x0560    // BCB6
#else
    #define _INTTYPES_H_SYS_    // 假设其他编译器支持C99.
#endif    // __AUTO_STDINT_H_USESYS


#ifdef __AUTO_STDINT_H_USESYS
// 使用编译器提供的<stdint.h>
#include <stdint.h>
#else
// 采用自定义的stdint.h. 参考了 msinttypes: http://code.google.com/p/msinttypes/
#ifndef _MSC_STDINT_H_ // [
#define _MSC_STDINT_H_

#include <limits.h>

// For Visual Studio 6 in C++ mode and for many Visual Studio versions when
// compiling for ARM we should wrap <wchar.h> include with 'extern "C++" {}'
// or compiler give many errors like this:

//   error C2733: second C linkage of overloaded function 'wmemchr' not allowed
//#ifdef __cplusplus
```

3. Re:全面解决.Net与Java互通时的RSA加解密问题，使用PEM格式的密钥文件

赞

--capital2012
4. Re:[C#] TestHttpPost：测试Http的POST方法的小工具
做了一点小改动以支持 json 数据

--xlog
5. Re:[C] 让VC、BCB支持C99的整数类型（stdint.h、inttypes.h）（兼容GCC）
You save my time,the issue was resolved ,good job!

--MonicaCQM
6. Re:[Java] zjdbcping：JDBC数据库连接测试工具
good，正好需要。thanks

--jiftle
7. Re:[Oracle] “表中有数据，但select count(*)的结果为0”问题的解决办法
记下来

--jiftle
8. Re:[C/C++] 各种C/C++编译器对UTF-8源码文件的兼容性测试（VC、GCC、BCB）
[VC2010, noBOM]len<1>=6,str=一瀛桦 // D2 BB E5 AD 97 41；“字A”的UTF-8编码为“E5 AD 97 41”，编译器将它们识别为GB2312编码的...

--Adano1
9. Re:[C#] 将NLog输出到RichTextBox，并在运行时动态修改日志级别过滤
@ ~雨落忧伤~nlog是 .NET的类库啊。官网是...

--zyl910

```
//extern "C" {
//#endif
//#  include <wchar.h>
//#ifdef __cplusplus
//}
//#endif
// <zyl910>: 在VC6下测试时，发现上面的方法会报告很多C2733错误．还是直接include算了．
#include <wchar.h>

// Define _W64 macros to mark types changing their size, like intptr_t.
#ifndef _W64
#  if !defined(__midl) && (defined(_X86_) || defined(_M_IX86)) && _MSC_VER >= 1300
#     define _W64 __w64
#  else
#     define _W64
#  endif
#endif


// 7.18.1 Integer types

// 7.18.1.1 Exact-width integer types

// Visual Studio 6 and Embedded Visual C++ 4 doesn't
// realize that, e.g. char has the same size as __int8
// so we give up on __intX for them.
#if (_MSC_VER < 1300)
   typedef signed char       int8_t;
   typedef signed short      int16_t;
   typedef signed int        int32_t;
   typedef unsigned char     uint8_t;
   typedef unsigned short    uint16_t;
   typedef unsigned int      uint32_t;
#else
   typedef signed __int8     int8_t;
   typedef signed __int16    int16_t;
   typedef signed __int32    int32_t;

   typedef unsigned __int8   uint8_t;
   typedef unsigned __int16  uint16_t;
```

10. Re:C#类与结构体究竟谁快——各种函数调用模式速度评测
现在7.2支持ref readonly 传结构体
--lindexi

```
   typedef unsigned __int32  uint32_t;
#endif
typedef signed __int64       int64_t;
typedef unsigned __int64     uint64_t;


// 7.18.1.2 Minimum-width integer types
typedef int8_t    int_least8_t;
typedef int16_t   int_least16_t;
typedef int32_t   int_least32_t;
typedef int64_t   int_least64_t;
typedef uint8_t   uint_least8_t;
typedef uint16_t  uint_least16_t;
typedef uint32_t  uint_least32_t;
typedef uint64_t  uint_least64_t;

// 7.18.1.3 Fastest minimum-width integer types
typedef int8_t    int_fast8_t;
typedef int16_t   int_fast16_t;
typedef int32_t   int_fast32_t;
typedef int64_t   int_fast64_t;
typedef uint8_t   uint_fast8_t;
typedef uint16_t  uint_fast16_t;
typedef uint32_t  uint_fast32_t;
typedef uint64_t  uint_fast64_t;

// 7.18.1.4 Integer types capable of holding object pointers
#ifdef _WIN64 // [
   typedef signed __int64    intptr_t;
   typedef unsigned __int64  uintptr_t;
#else // _WIN64 ][
   typedef _W64 signed int   intptr_t;
   typedef _W64 unsigned int uintptr_t;
#endif // _WIN64 ]

// 7.18.1.5 Greatest-width integer types
typedef int64_t   intmax_t;

typedef uint64_t  uintmax_t;
```

```
// 7.18.2 Limits of specified-width integer types

#if !defined(__cplusplus) || defined(__STDC_LIMIT_MACROS) // [   See footnote 220 at page 257 and footnote 221 at

// 7.18.2.1 Limits of exact-width integer types
#define INT8_MIN     ((int8_t)_I8_MIN)
#define INT8_MAX     _I8_MAX
#define INT16_MIN    ((int16_t)_I16_MIN)
#define INT16_MAX    _I16_MAX
#define INT32_MIN    ((int32_t)_I32_MIN)
#define INT32_MAX    _I32_MAX
#define INT64_MIN    ((int64_t)_I64_MIN)
#define INT64_MAX    _I64_MAX
#define UINT8_MAX    _UI8_MAX
#define UINT16_MAX   _UI16_MAX
#define UINT32_MAX   _UI32_MAX
#define UINT64_MAX   _UI64_MAX

// 7.18.2.2 Limits of minimum-width integer types
#define INT_LEAST8_MIN    INT8_MIN
#define INT_LEAST8_MAX    INT8_MAX
#define INT_LEAST16_MIN   INT16_MIN
#define INT_LEAST16_MAX   INT16_MAX
#define INT_LEAST32_MIN   INT32_MIN
#define INT_LEAST32_MAX   INT32_MAX
#define INT_LEAST64_MIN   INT64_MIN
#define INT_LEAST64_MAX   INT64_MAX
#define UINT_LEAST8_MAX   UINT8_MAX
#define UINT_LEAST16_MAX  UINT16_MAX
#define UINT_LEAST32_MAX  UINT32_MAX
#define UINT_LEAST64_MAX  UINT64_MAX

// 7.18.2.3 Limits of fastest minimum-width integer types
#define INT_FAST8_MIN    INT8_MIN
#define INT_FAST8_MAX    INT8_MAX
#define INT_FAST16_MIN   INT16_MIN

#define INT_FAST16_MAX   INT16_MAX
#define INT_FAST32_MIN   INT32_MIN
```

```
#define INT_FAST32_MAX   INT32_MAX
#define INT_FAST64_MIN   INT64_MIN
#define INT_FAST64_MAX   INT64_MAX
#define UINT_FAST8_MAX   UINT8_MAX
#define UINT_FAST16_MAX  UINT16_MAX
#define UINT_FAST32_MAX  UINT32_MAX
#define UINT_FAST64_MAX  UINT64_MAX

// 7.18.2.4 Limits of integer types capable of holding object pointers
#ifdef _WIN64 // [
#  define INTPTR_MIN   INT64_MIN
#  define INTPTR_MAX   INT64_MAX
#  define UINTPTR_MAX  UINT64_MAX
#else // _WIN64 ][
#  define INTPTR_MIN   INT32_MIN
#  define INTPTR_MAX   INT32_MAX
#  define UINTPTR_MAX  UINT32_MAX
#endif // _WIN64 ]

// 7.18.2.5 Limits of greatest-width integer types
#define INTMAX_MIN   INT64_MIN
#define INTMAX_MAX   INT64_MAX
#define UINTMAX_MAX  UINT64_MAX

// 7.18.3 Limits of other integer types

#ifdef _WIN64 // [
#  define PTRDIFF_MIN  _I64_MIN
#  define PTRDIFF_MAX  _I64_MAX
#else  // _WIN64 ][
#  define PTRDIFF_MIN  _I32_MIN
#  define PTRDIFF_MAX  _I32_MAX
#endif  // _WIN64 ]

#define SIG_ATOMIC_MIN  INT_MIN
#define SIG_ATOMIC_MAX  INT_MAX


#ifndef SIZE_MAX // [
#  ifdef _WIN64 // [
```

```
#     define SIZE_MAX  _UI64_MAX
#  else // _WIN64 ][
#     define SIZE_MAX  _UI32_MAX
#  endif // _WIN64 ]
#endif // SIZE_MAX ]

// WCHAR_MIN and WCHAR_MAX are also defined in <wchar.h>
#ifndef WCHAR_MIN // [
#  define WCHAR_MIN  0
#endif  // WCHAR_MIN ]
#ifndef WCHAR_MAX // [
#  define WCHAR_MAX  _UI16_MAX
#endif  // WCHAR_MAX ]

#define WINT_MIN  0
#define WINT_MAX  _UI16_MAX

#endif // __STDC_LIMIT_MACROS ]


// 7.18.4 Limits of other integer types

#if !defined(__cplusplus) || defined(__STDC_CONSTANT_MACROS) // [   See footnote 224 at page 260

// 7.18.4.1 Macros for minimum-width integer constants

#define INT8_C(val)  val##i8
#define INT16_C(val) val##i16
#define INT32_C(val) val##i32
#define INT64_C(val) val##i64

#define UINT8_C(val)  val##ui8
#define UINT16_C(val) val##ui16
#define UINT32_C(val) val##ui32
#define UINT64_C(val) val##ui64

// 7.18.4.2 Macros for greatest-width integer constants

#define INTMAX_C   INT64_C
#define UINTMAX_C  UINT64_C
```

```
#endif // __STDC_CONSTANT_MACROS ]


#endif // _MSC_STDINT_H_ ]

#endif // #ifdef __AUTO_STDINT_H_USESYS

#endif // #ifndef __AUTO_STDINT_H_INCLUDED
```

2.2 auto_inttypes.h

全部代码——

```
//////////////////////////////////////////////////////////////////////
/*
auto_inttypes.h: 兼容C99标准的inttypes.h
Author: zyl910
Blog: http://www.cnblogs.com/zyl910
URL: http://www.cnblogs.com/zyl910/archive/2013/01/10/c99int_v101.html
Version: V1.01
Updata: 2013-01-10

测试过的编译器--
VC: 6, 2003, 2005, 2008, 2010, 2012.
BCB: 6.
GCC(Linux): 4.7.0(Fedora 17).
GCC(Mac): llvm-gcc-4.2(Mac OS X Lion 10.7.4, Xcode 4.4.1).
GCC(MinGW): 4.6.2(MinGW(20120426)), 4.7.1(TDM-GCC(MinGW-w64)).


Update
```

```
~~~~~~

[2013-01-10] V1.01
* 检查了对VC2008、VC2012的兼容性. 确认VC2012仍不支持inttypes.h.
* 为了避免包含目录问题，更名auto_stdint.h（原stdint.h）.
* 更改宏名：__AUTO_INTTYPES_H_INCLUDED（原_INTTYPES_H_ALL_），__AUTO_INTTYPES_H_USESYS（原_INTTYPES_H_SYS_）.

[2012-08-08] V1.00
* V1.0发布.
* 参考了 msinttypes-r26. http://code.google.com/p/msinttypes/
* 修正VC6不支持I32问题.


*/
////////////////////////////////////////////////////////////////

#ifndef __AUTO_INTTYPES_H_INCLUDED
#define __AUTO_INTTYPES_H_INCLUDED

// __AUTO_INTTYPES_H_USESYS: 编译器是否提供了<inttypes.h>
#undef __AUTO_INTTYPES_H_USESYS
#if defined(__GNUC__)    // GCC.
    #define __AUTO_INTTYPES_H_USESYS
#elif defined(_MSC_VER)    // MSVC. VC2012仍不支持.
#elif defined(__BORLANDC__)    // BCB. BCB6仍不支持.
#else
    #define __AUTO_INTTYPES_H_USESYS    // 假设其他编译器支持C99.
#endif    // __AUTO_INTTYPES_H_USESYS


#ifdef __AUTO_INTTYPES_H_USESYS
// 使用编译器提供的<inttypes.h>
#include <inttypes.h>
#else
// 采用自定义的inttypes.h. 参考了 msinttypes: http://code.google.com/p/msinttypes/

#ifndef _MSC_INTTYPES_H_ // [
#define _MSC_INTTYPES_H_
```

```
//#include "stdint.h"
#include "auto_stdint.h"

// 7.8 Format conversion of integer types

typedef struct {
   intmax_t quot;
   intmax_t rem;
} imaxdiv_t;

// 7.8.1 Macros for format specifiers

#if !defined(__cplusplus) || defined(__STDC_FORMAT_MACROS) // [   See footnote 185 at page 198

// The fprintf macros for signed integers are:
#define PRId8       "d"
#define PRIi8       "i"
#define PRIdLEAST8  "d"
#define PRIiLEAST8  "i"
#define PRIdFAST8   "d"
#define PRIiFAST8   "i"

#define PRId16       "hd"
#define PRIi16       "hi"
#define PRIdLEAST16  "hd"
#define PRIiLEAST16  "hi"
#define PRIdFAST16   "hd"
#define PRIiFAST16   "hi"

#if defined(_MSC_VER) && _MSC_VER<=1200    // VC6
#define PRId32       "d"
#define PRIi32       "i"
#define PRIdLEAST32  "d"
#define PRIiLEAST32  "i"
#define PRIdFAST32   "d"
#define PRIiFAST32   "i"
#else
#define PRId32       "I32d"
#define PRIi32       "I32i"
```

```
#define PRIdLEAST32  "I32d"
#define PRIiLEAST32  "I32i"
#define PRIdFAST32   "I32d"
#define PRIiFAST32   "I32i"
#endif

#define PRId64       "I64d"
#define PRIi64       "I64i"
#define PRIdLEAST64  "I64d"
#define PRIiLEAST64  "I64i"
#define PRIdFAST64   "I64d"
#define PRIiFAST64   "I64i"

#define PRIdMAX     "I64d"
#define PRIiMAX     "I64i"

#define PRIdPTR     "Id"
#define PRIiPTR     "Ii"

// The fprintf macros for unsigned integers are:
#define PRIo8       "o"
#define PRIu8       "u"
#define PRIx8       "x"
#define PRIX8       "X"
#define PRIoLEAST8  "o"
#define PRIuLEAST8  "u"
#define PRIxLEAST8  "x"
#define PRIXLEAST8  "X"
#define PRIoFAST8   "o"
#define PRIuFAST8   "u"
#define PRIxFAST8   "x"
#define PRIXFAST8   "X"

#define PRIo16       "ho"
#define PRIu16       "hu"
#define PRIx16       "hx"
#define PRIX16       "hX"
#define PRIoLEAST16  "ho"
#define PRIuLEAST16  "hu"
```

```
#define PRIxLEAST16  "hx"
#define PRIXLEAST16  "hX"
#define PRIoFAST16   "ho"
#define PRIuFAST16   "hu"
#define PRIxFAST16   "hx"
#define PRIXFAST16   "hX"

#if defined(_MSC_VER) && _MSC_VER<=1200    // VC6
#define PRIo32       "o"
#define PRIu32       "u"
#define PRIx32       "x"
#define PRIX32       "X"
#define PRIoLEAST32  "o"
#define PRIuLEAST32  "u"
#define PRIxLEAST32  "x"
#define PRIXLEAST32  "X"
#define PRIoFAST32   "o"
#define PRIuFAST32   "u"
#define PRIxFAST32   "x"
#define PRIXFAST32   "X"
#else
#define PRIo32       "I32o"
#define PRIu32       "I32u"
#define PRIx32       "I32x"
#define PRIX32       "I32X"
#define PRIoLEAST32  "I32o"
#define PRIuLEAST32  "I32u"
#define PRIxLEAST32  "I32x"
#define PRIXLEAST32  "I32X"
#define PRIoFAST32   "I32o"
#define PRIuFAST32   "I32u"
#define PRIxFAST32   "I32x"
#define PRIXFAST32   "I32X"
#endif

#define PRIo64       "I64o"
#define PRIu64       "I64u"
#define PRIx64       "I64x"
#define PRIX64       "I64X"
```

```
#define PRIoLEAST64  "I64o"
#define PRIuLEAST64  "I64u"
#define PRIxLEAST64  "I64x"
#define PRIXLEAST64  "I64X"
#define PRIoFAST64   "I64o"
#define PRIuFAST64   "I64u"
#define PRIxFAST64   "I64x"
#define PRIXFAST64   "I64X"

#define PRIoMAX     "I64o"
#define PRIuMAX     "I64u"
#define PRIxMAX     "I64x"
#define PRIXMAX     "I64X"

#define PRIoPTR     "Io"
#define PRIuPTR     "Iu"
#define PRIxPTR     "Ix"
#define PRIXPTR     "IX"

// The fscanf macros for signed integers are:
#define SCNd8       "d"
#define SCNi8       "i"
#define SCNdLEAST8  "d"
#define SCNiLEAST8  "i"
#define SCNdFAST8   "d"
#define SCNiFAST8   "i"

#define SCNd16       "hd"
#define SCNi16       "hi"
#define SCNdLEAST16  "hd"
#define SCNiLEAST16  "hi"
#define SCNdFAST16   "hd"
#define SCNiFAST16   "hi"

#define SCNd32       "ld"
#define SCNi32       "li"
#define SCNdLEAST32  "ld"
#define SCNiLEAST32  "li"
#define SCNdFAST32   "ld"
```

```
#define SCNiFAST32   "li"

#define SCNd64       "I64d"
#define SCNi64       "I64i"
#define SCNdLEAST64  "I64d"
#define SCNiLEAST64  "I64i"
#define SCNdFAST64   "I64d"
#define SCNiFAST64   "I64i"

#define SCNdMAX     "I64d"
#define SCNiMAX     "I64i"

#ifdef _WIN64 // [
#  define SCNdPTR     "I64d"
#  define SCNiPTR     "I64i"
#else  // _WIN64 ][
#  define SCNdPTR     "ld"
#  define SCNiPTR     "li"
#endif  // _WIN64 ]

// The fscanf macros for unsigned integers are:
#define SCNo8       "o"
#define SCNu8       "u"
#define SCNx8       "x"
#define SCNX8       "X"
#define SCNoLEAST8  "o"
#define SCNuLEAST8  "u"
#define SCNxLEAST8  "x"
#define SCNXLEAST8  "X"
#define SCNoFAST8   "o"
#define SCNuFAST8   "u"
#define SCNxFAST8   "x"
#define SCNXFAST8   "X"

#define SCNo16       "ho"
#define SCNu16       "hu"
#define SCNx16       "hx"
#define SCNX16       "hX"
#define SCNoLEAST16  "ho"
```

```
#define SCNuLEAST16  "hu"
#define SCNxLEAST16  "hx"
#define SCNXLEAST16  "hX"
#define SCNoFAST16   "ho"
#define SCNuFAST16   "hu"
#define SCNxFAST16   "hx"
#define SCNXFAST16   "hX"

#define SCNo32       "lo"
#define SCNu32       "lu"
#define SCNx32       "lx"
#define SCNX32       "lX"
#define SCNoLEAST32  "lo"
#define SCNuLEAST32  "lu"
#define SCNxLEAST32  "lx"
#define SCNXLEAST32  "lX"
#define SCNoFAST32   "lo"
#define SCNuFAST32   "lu"
#define SCNxFAST32   "lx"
#define SCNXFAST32   "lX"

#define SCNo64       "I64o"
#define SCNu64       "I64u"
#define SCNx64       "I64x"
#define SCNX64       "I64X"
#define SCNoLEAST64  "I64o"
#define SCNuLEAST64  "I64u"
#define SCNxLEAST64  "I64x"
#define SCNXLEAST64  "I64X"
#define SCNoFAST64   "I64o"
#define SCNuFAST64   "I64u"
#define SCNxFAST64   "I64x"
#define SCNXFAST64   "I64X"

#define SCNoMAX     "I64o"
#define SCNuMAX     "I64u"
#define SCNxMAX     "I64x"
#define SCNXMAX     "I64X"
```

```
#ifdef _WIN64 // [
#  define SCNoPTR     "I64o"
#  define SCNuPTR     "I64u"
#  define SCNxPTR     "I64x"
#  define SCNXPTR     "I64X"
#else  // _WIN64 ][
#  define SCNoPTR     "lo"
#  define SCNuPTR     "lu"
#  define SCNxPTR     "lx"
#  define SCNXPTR     "lX"
#endif  // _WIN64 ]

#endif // __STDC_FORMAT_MACROS ]

// 7.8.2 Functions for greatest-width integer types

// 7.8.2.1 The imaxabs function
#define imaxabs _abs64

// 7.8.2.2 The imaxdiv function

#ifdef _MSC_VER
// This is modified version of div() function from Microsoft's div.c found
// in %MSVC.NET%\crt\src\div.c
#ifdef STATIC_IMAXDIV // [
static
#else // STATIC_IMAXDIV ][
_inline
#endif // STATIC_IMAXDIV ]
imaxdiv_t __cdecl imaxdiv(intmax_t numer, intmax_t denom)
{
   imaxdiv_t result;

   result.quot = numer / denom;
   result.rem = numer % denom;

   if (numer < 0 && result.rem > 0) {
      // did division wrong; must fix up
      ++result.quot;
```

```
      result.rem -= denom;
   }

   return result;
}
#endif  // #ifdef _MSC_VER

// 7.8.2.3 The strtoimax and strtoumax functions
#define strtoimax _strtoi64
#define strtoumax _strtoui64

// 7.8.2.4 The wcstoimax and wcstoumax functions
#define wcstoimax _wcstoi64
#define wcstoumax _wcstoui64


#endif // _MSC_INTTYPES_H_ ]

#endif // #ifdef __AUTO_INTTYPES_H_USESYS

#endif // #ifndef __AUTO_INTTYPES_H_INCLUDED
```

2.3 c99int.c

全部代码——

```
//////////////////////////////////////////////////////////////
/*
c99int.c: 测试C99整数类型.
Author: zyl910
Blog: http://www.cnblogs.com/zyl910
URL: http://www.cnblogs.com/zyl910/archive/2013/01/10/c99int_v101.html
Version: V1.01
```

```
Updata: 2013-01-01



Update
~~~~~~

[2013-01-10] V1.01
* 使用V1.01版的auto_stdint.h、auto_inttypes.h。

[2012-08-08] V1.0
* V1.0发布.


*/
//////////////////////////////////////////////////////////////

#define __STDC_LIMIT_MACROS
#define __STDC_CONSTANT_MACROS
#define __STDC_FORMAT_MACROS

#include <stdio.h>

#include <wchar.h>

#include "auto_stdint.h"
#include "auto_inttypes.h"


int main(int argc, char* argv[])
{
    uint8_t i8 = (uint8_t)INT8_C(-1);
    uint16_t i16 = (uint16_t)INT16_C(-1);
    uint32_t i32 = (uint32_t)INT32_C(-1);
    uint64_t i64 = (uint64_t)INT64_C(-1);

    printf("c99int:\t%" PRIu8 ", %" PRIu16 ", %" PRIu32 ", %" PRIu64 "\n", i8, i16, i32, i64);
    return 0;
}
```

2.4 makefile

全部代码——

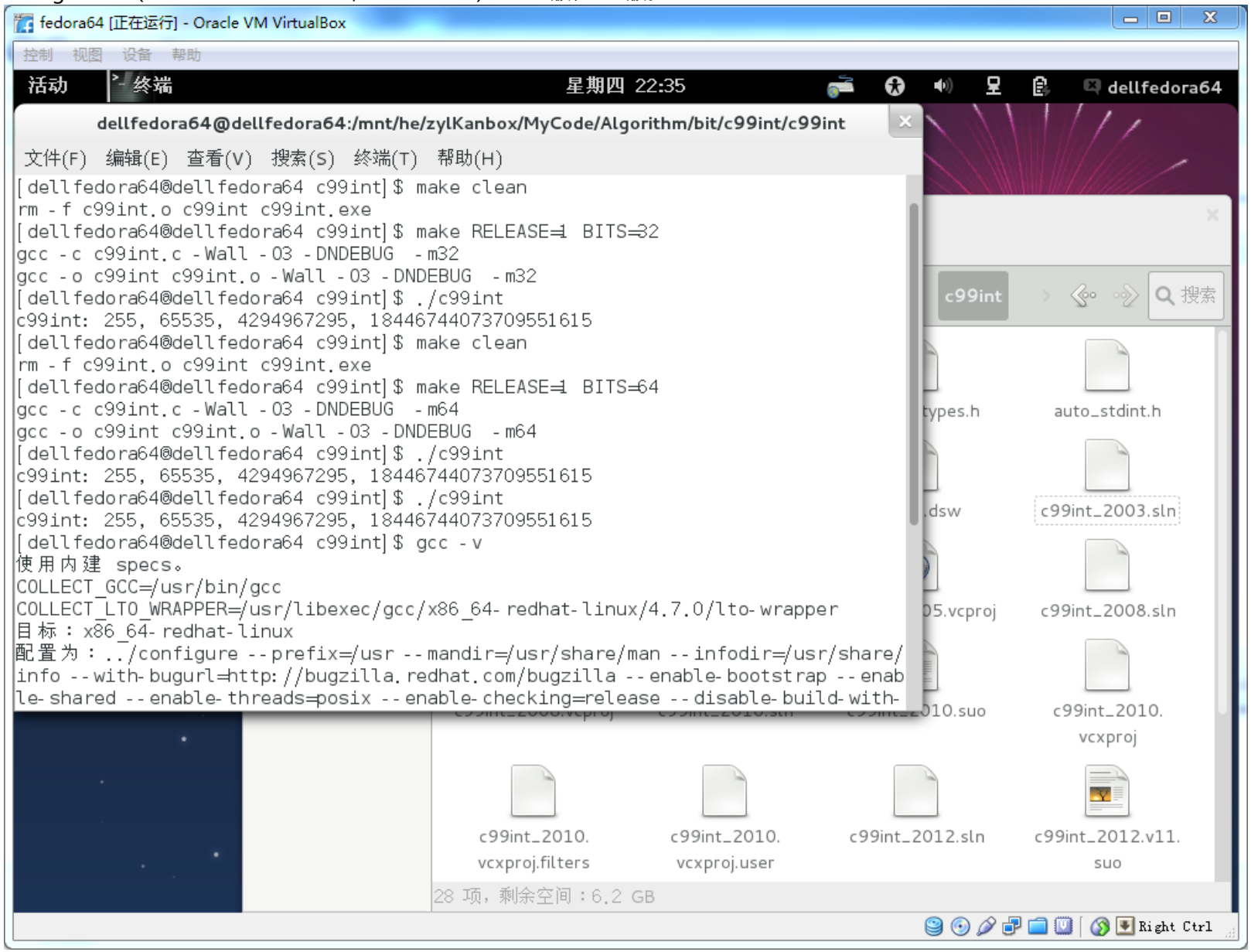
```
# flags
CC = gcc
CFS = -Wall

# args
RELEASE =0
UNICODE =0
BITS =
CFLAGS =

# [args] 生成模式. 0代表debug模式, 1代表release模式. make RELEASE=1.
ifeq ($(RELEASE),0)
    # debug
    CFS += -g
else
    # release
    CFS += -O3 -DNDEBUG
    //CFS += -O3 -g -DNDEBUG
endif

# [args] UNICODE模式. 0代表ansi模式, 1代表unicode模式. make UNICODE=1.
ifeq ($(UNICODE),0)
    # ansi
    CFS +=
else
    # unicode
    CFS += -D_UNICODE -DUNICODE
endif
```

```
# [args] 程序位数. 32代表32位程序, 64代表64位程序, 其他默认. make BITS=32.
ifeq ($(BITS),32)
    CFS += -m32
else
    ifeq ($(BITS),64)
        CFS += -m64
    else
    endif
endif

# [args] 使用 CFLAGS 添加新的参数. make CFLAGS="-mavx".
CFS += $(CFLAGS)


.PHONY : all clean

# files
TARGETS = c99int
OBJS = c99int.o

all : $(TARGETS)

c99int : $(OBJS)
    $(CC) -o $@ $^ $(CFS)


c99int.o : c99int.c
    $(CC) -c $< $(CFS)


clean :
    rm -f $(OBJS) $(TARGETS) $(addsuffix .exe,$(TARGETS))
```

三、测试结果

在以下编译器中成功编译——
VC6：x86版。
VC2003：x86版。
VC2005：x86版、x64版。
VC2008：x86版。
VC2010：x86版、x64版。
VC2012：x86版、x64版。
BCB6：x86版。
GCC 4.6.2（MinGW (20120426)）：x86版。
GCC 4.7.1（TDM-GCC(MinGW-w64)）：x86版、x64版。
GCC 4.7.0（Fedora 17）：x86版、x64版。

llvm-gcc-4.2（Mac OS X Lion 10.7.4, Xcode 4.4.1）：x86版、x64版。



参考文献——

《ISO/IEC 9899:1999 (C99)》。ISO/IEC，1999。www.open-std.org/jtc1/sc22/wg14/www/docs/n1124.pdf

《C99标准》。yourtommy。http://blog.csdn.net/yourtommy/article/details/7495033
msinttypes-r26. http://code.google.com/p/msinttypes/
《VC 里边怎么用C99》. http://hi.baidu.com/419836321/blog/item/bf643830976204b15edf0e3a.html
《[C/C++] 显示各种C/C++编译器的预定义宏（C11标准、C++11标准、VC、BCB、Intel、GCC）》.
http://www.cnblogs.com/zyl910/archive/2012/08/02/printmacro.html
《[C] 让VC、BCB支持C99的整数类型（stdint.h、inttypes.h）（兼容GCC）》.
http://www.cnblogs.com/zyl910/archive/2012/08/08/c99int.html

源码下载——
http://files.cnblogs.com/zyl910/c99int_v101.rar

分类: --- My_原创, --- Program_编程, C00 Language_语言, C10 C系列, C11 C, D00 Platform_平台, D30 Windows, D40 Linux, D50 Mac

标签: Int, VC, c99, c99int, gcc, bcb

« 上一篇： [VBScript] allfiles.vbs: 显示子目录下的所有文件的修改时间、大小、全限定名等信息
» 下一篇： [C/C++] VC2012编译的程序在WinXP下报告“指定的可执行文件不是有效的 Win32 应用程序”错误