Quicker

1 标准C语言扩展

（）中的命令是该命令的短命令，所有的命令都能用F4（向前搜索）、F3（向后搜索）来或Ctrl Enter（向后）定位，为了方便最好能够设置si的search菜单下的search窗口(Ctrl f)选中select when found。

1.1 /* 自动生成/* */

该命令自动生成C语言的注释，它能自动换行对齐，在把文档中一长串注释拷贝过来时特别管用，不用自己去对齐了。还能自动识别中文和英文单词，对于中文不会把一个字分开，对于英文单词分开时会自动添加连字符，下面是一个注释的例子

abcdefghijk = abcd + cdefg + hijk /*该命令自动生成C语言的注释，它能自动换行对齐，在把文档中一长串注释拷贝过来时特别管用，不用自己去对齐了。*/

1.2 { 自动生成 { }

支持块命令插入

该命令自动生成

{

}

1.3 while (wh) 自动生成While语句

该命令自动生成如下格式

支持块命令插入

```
while ( # )
{
    #
}
```

1.1 if 自动生成if语句

支持块命令插入

if 生成if结构的语句

ife 对应生成if else结构的语句

ifs 对应生成if elseif else结构的f语句

该命令自动生成如下格式

```
if ( # )
{
    #
}
```

1.1 for 自动生成for语句

支持块命令插入

该命令自动生成如下格式

```
for ( #; #; # )
{
    #
}
```

1.1 fo 自动生成for语句

与前一条命令相比它直接会定义循环变量

```
UINT32 ulI = 0;

for ( ulI = 0;  ulI < #;  ulI++ )
{
    #
}
```

1.2 do 自动生成 do while语句

支持块命令插入

该命令自动生成如下格式

```
do
{
    #
```

} while (#);

1.1 #ifd 自动生成 #ifdef 命令

支持块命令插入

该命令生成如下格式

#ifdef UMSC

#endif /* UMSC */

UMSC为提示输入值

1.1 #ifn 自动生成 #ifndef 命令

支持块命令插入

该命令生成如下格式

#ifndef UMSC

#endif /* UMSC */

UMSC为提示输入值

1.1 #if 自动生成 #if 命令

支持块命令插入

该命令生成如下格式

#if (UMSC == 1)

#endif /*if (UMSC == 1) */

UMSC为提示输入值

1.1 cpp 自动生成适用于c++的c原型说明定义

该命令生成如下格式

#ifdef __cplusplus

#if __cplusplus

extern "C"{

#endif

#endif /* __cplusplus */

#ifdef __cplusplus

#if __cplusplus

}

#endif

#endif /* __cplusplus */

1.2 switch (sw) 自动生成switch语句

该命令将提示输入case的个数，生成如下格式

switch (#)

{

case #:

#

break;

default:

#

}

注意该命令还有一个重要的功能，在执行该命令之前如果先剪接一个case用的定义组

然后执行该命令，提示输入case个数时输入0，它能自动生成全部定义

具体如下：

```
typedef enum SAAL_EXTERNAL_EVENT
{
    AAL_ESTABLISH_INDICATION = 4,/*test*/
      /*test*/
   AAL_ESTABLISH_CONFIRM = 5,
   /*test*/ AAL_RELEASE_REQUEST = 6,
    AAL_RELEASE_INDICATION = 7, //test
//       AAL_RELEASE_INDICATION    = 20
   AAL_RELEASE_CONFIRM = 8,
    AAL_DATA_REQUEST = 9,
    AAL_DATA_INDICATION = 10,

} SAAL_EXTERNAL_EVENT_ENUM ;
```

或者是如下的宏定义形式

```
#define     AAL_ESTABLISH_INDICATION  4
/*test*/
#define     AAL_ESTABLISH_CONFIRM  5
#define     AAL_RELEASE_REQUEST  6
#define  /*test*/    AAL_RELEASE_INDICATION  7
#define     AAL_RELEASE_CONFIRM  8  //test
#define     AAL_DATA_REQUEST  9     /*test*/
#define     AAL_DATA_INDICATION  10
```

选中蓝色部分，将其拷贝到剪贴板(Ctrl C)，注意不要包含红色部分，然后执行sw命令提示输入case的个数这是填0，就会自动生成如下格式非常方便

```
switch( # )
{
    case AAL_ESTABLISH_INDICATION:
        #
        break;
    case AAL_ESTABLISH_CONFIRM:
        #
        break;
    case AAL_RELEASE_REQUEST:
        #
        break;
    case AAL_RELEASE_INDICATION:
        #
        break;
    case AAL_RELEASE_CONFIRM:
        #
        break;
    case AAL_DATA_REQUEST:
        #
        break;
    case AAL_DATA_INDICATION:
        #
        break;
    default:
        #
}
```

1.3 case (ca)自动生成case语句

支持块命令输入

该命令生成如下格式，用它可以避免遗漏break

```
case #:
    #
    break;
```

1.4 struct (st) 自动生成结构类型

该命令自动生成如下结构定义，它提示输入结构名，会自动转换成大写形式，并且自动在其后添加_STRU作为结构类型

typedef struct HELLO_TEST

{

#

}HELLO_TEST_STRU;

1.5 enum (en) 自动生成枚举类型

该命令自动生成如下结构定义，它提示输入结构名，会自动转换成大写形式，并且自动在其后添加_STRU作为结构类型

typedef enum HELLO_TEST

{

#

}HELLO_TEST_ENUM;

2 标准说明生成

2.1 config (co) 配置用户名和标题的说明语种

用户名就是前面自动生成的作者，修改者的名字，语种，就是前面提到的文件头说明和函数头说明所采用的的语言有中文和英文两种选择

2.2 file (fi) 生成标准的文件头

自动生成如下格式，它能自动取得文件名，作者名，时间，以及函数列表，而且在生成的过程中会主动弹出功能描述能的内容输入对话框（提示对话框只能输入256个字符），输入的内容能够自动换行对齐，这样既可以避免遗漏说明，而且一般在详细设计中对每个文件的主要功能都有详细的描述，只要把这些描述 拷贝过来即可，它能自动排列好， 非常方便。

2.3 func (fu) 生成标准的函数头说明

其自动生成如下函数头，该函数必须在该函数的前一行执行，它能自动取得函数名，能够自动取得函数的输入、输出参数，并且排列好，同时生成日期和作者。能够 提示输入功能描述，输入的内容能够自动换行对齐，这样既可以避免遗漏说明，而且一般在详细设计中对每个函数的主要功能都有详细的描述，只要把这些描述 拷贝过来即可，它能自动排列好， 非常方便。 （因为SI的宏功能有限，没能自动加入调用函数和被调函数的功能）

如果不是在已存在的函数前执行则提示输入函数名，提示输入函数描述，提示输入返回值类型，提示输入函数入口参数，输完后按Esc退出

```
/*****************************************************************************
****函 数 名   : afdaain
 功能描述   : 测试函数
 输入参数   : int d
                          int t
                          char t
 输出参数   : 无
 返 回 值   :    int
 调用函数   :
 被调函数   :

 修改历史        :
 1. 日期         : 2002.5.11
    作者         : lushengwen
    修改方式     : 新生成函数
*****************************************************************************
***/
```

2.4 hi 增加修改历史列表

自动在该行增加修改历史列表，用于文件头和函数头说明中的历史记录更新，其添加形式如下

```
2. 日      期     : 2002年5月17日
    作      者     : lushengwen
    修改内容     : lkasjdfkla
```

2.5 hd 自动生成函数头文件

该命令能够自动生成但前C文件的头文件定义，包括常用的宏定义，还有全部的函数原型定义。

2.6 Hdn 生成新的有文件

该命令能够自动生成指定的的头文件定义，包括常用的宏定义，提示输入函数原型的类型。

3 代码修改注释

3.1 pn 添加问题单号

在进行问题单修改时都要求在修改的地方注明问题单号和修改人以及修改时间，大部分一个问题单的修改都会涉及到几个地方，本功能提供自动取问题单号的功能，下面的几个命令所生成的问题单号就是有它提供，如果输入为#则不显示问题单号

3.2 ap 添加问题单修改说明

该命令提示输入问题单号和修改原因，生成如下格式的说明

3.3 ab 添加开始说明

它能自动生成如下说明：

1.有问题单号的情况，其中问题单号是由ap命令加入的。

/*BEGIN: Added by lushengwen, 2002/5/13　　问题单号:D02556*/

2.没有问题单号的情况，

/*BEGIN: Added by lushengwen, 2002/5/13*/

3.4 ae 添加结束命令

该命令是as的对应命令，表示结束添加

支持块命令操作

3.5 abg 插入添加开始和结束说明

该命令是前两个命令的组合

支持块命令操作

3.6 db 删除开始命令

它能自动生成如下说明：

1.有问题单号的情况，其中问题单号是由ap命令加入的。

/*BEGIN: Deleted by lushengwen, 2002/5/13　　问题单号:D02556*/

2.没有问题单号的情况，

/*BEGIN: Deleted by lushengwen, 2002/5/13 */

3.7 de 删除结束命令

该命令是ds的对应命令，表示删除结束

3.8 dbg 插入删除开始和结束说明

该命令是前两个命令的组合

支持块命令操作

3.9 mb 修改开始命令

它能自动生成如下说明：

1.有问题单号的情况，其中问题单号是由ap命令加入的。

/*BEGIN: Modified by lushengwen, 2002/5/13　　问题单号:D02556*/

2.没有问题单号的情况，

/*BEGIN: Modified by lushengwen, 2002/5/13 */

3.10 me 修改结束命令

该命令是ms的对应命令，表示删除结束

3.11 mbg 插入修改开始和结束说明

该命令是前两个命令的组合

支持块命令操作

1 其余几个常用宏

1.1 将从C＋＋的 // 注释改为标准的C /* */注释

ComentCPPtoC()　文件中的//注释自动修改为/* */注释，因为很多C编译器不能很好的处理C＋＋风格的 // 注释，用该命令可以方便的将选中区域内的//注释进行修改，建议将该宏定义在菜单中。

1.2 能自动将Tab转换成空格（mstp_out.c）

ReplaceBufTab()　自动将单前文件中的Tab转换成空格

ReplaceTabInProj()　自动将工程中的文件中的Tab转换成空格

因为不同的编辑器对tab的长度定义不一至，造成代码格式紊乱，编程规范是不允许用tab键的，可以用该宏来将整个文件的tab进行替换，非常方便,建议将该宏定义在菜单中

1.3 插入当前的函数名

InsertFuncName ()　能自动的插入本函数名，现在的函数名一般较长，特别是在调试打印代码中为了显示出出错的函数经常需要输入本函数名，该宏能带来极大方便。建议定义为一个热键 ctrl 1

1.4 自动在函数入口、出口插入、删除打印函数

注意使用本功能时一定要求所有的语句符合公司编程规范，要求一条语句一行，因为各种编码情况很复杂，特别是老代码和有大量条件编译的情况下很难覆盖各种情况，最好在执行完后再检查一遍。不推荐使用工程内的插入和删除。

InsertTraceInfo()　能够自动在函数的出、入口首尾加入打印代码，即在光标处加入一个进入函数的打印，在函数的返回处加入一个出函数的打印，用于调试跟踪时很方便，建议定义为热键 Ctrl t

AutoInsertTraceInfoInBuf()能够在当前文件的函数出入口加入打印信息，定义菜单

AutoInsertTraceInfoInPrj()能够在当前工程的函数出入口加入打印信息，定义为菜单

RemoveTraceInfo() 删除InsertTraceInfo()添加的打印信息，定义为菜单

RemoveCurBufTraceInfo()删除文件中全部的InsertTraceInfo()添加的打印信息，定义为菜单

RemovePrjTraceInfo()删除但前工程中全部的InsertTraceInfo()添加的打印信息，定义为菜单

1.5 自动格式当前行

FormatLine()　能够自动将一行长的文字分成多行，并且从第二行开始，起始列为关标所在列，该宏是为了弥补因为对话框只能处理256个字符而编写的，当需要输入超过256个字符的说明时，就可以先把它拷贝到第一行，然后执行本宏，进行分行对齐。

1.6 更新函数列表

UpdateFunctionList() 能够自动在光标所在行重新生成函数列表，用于函数头说明的函数列表更新。

1.7 复合语句删除

DelCompoundStatement()能自动删除复合语句，定义热键 Ctrl D

对于如下语句，如果我想删除条件 ulCount　> 0，只需将光标放在if语句这一行（蓝色行），执行 Ctrl D 即可

```
   stSubsystemRec.hwRBSubSystemCpuAveUsageLimit                    =
CPU_RESTORE_THRESHOLD;
```

```
if( ulCount > 0 )
{
    stSubsystemRec.hwRBSubSystemAdminStatus = ADSTATUS_DOWN;
    stSubsystemRec.hwRBSubSystemCpuMaxUsageLimit = CPU_ALARM;
    stSubsystemRec.hwRBSubSystemDSPAveUsageLimit = DSP_ALARM;
    stSubsystemRec.hwRBSubSystemDSPMaxUsageLimit = DSP_RESTORE;
    stSubsystemRec.hwRBSubSystemOperStatus = OPSTATUS_OTHER;
}
stSubsystemRec.hwRBSubSystemIndex = 0;
stSubsystemRec.hwRBSubSystemIpAddress = ulIpAddr;
```

执行结果

```
    stSubsystemRec.hwRBSubSystemCpuAveUsageLimit                    =
CPU_RESTORE_THRESHOLD;
    stSubsystemRec.hwRBSubSystemAdminStatus = ADSTATUS_DOWN;
    stSubsystemRec.hwRBSubSystemCpuMaxUsageLimit = CPU_ALARM;
    stSubsystemRec.hwRBSubSystemDSPAveUsageLimit = DSP_ALARM;
    stSubsystemRec.hwRBSubSystemDSPMaxUsageLimit = DSP_RESTORE;
    stSubsystemRec.hwRBSubSystemOperStatus = OPSTATUS_OTHER;
    stSubsystemRec.hwRBSubSystemIndex = 0;
    stSubsystemRec.hwRBSubSystemIpAddress = ulIpAddr;
```

1.8 其它跟扩展命令对应的宏

下面宏可以根据需要定义为热键

ClearPrombleNo 清除问题单 Alt Del

ExpandBraceLarge 加入{} /*支持块输入*/ Ctrl]

ExpandBraceLittle 加入() /*支持块输入*/ Ctrl Shift 9

ExpandBraceMid 加入[] /*支持块输入*/ Alt [

FileHeaderCreate 生成文件头说明 Ctrl Shift i

FunctionHeaderCreate 生成函数头说明 Ctrl Shift u

HeaderFileCreate 生成头文件 Ctrl Shift d

InsertCase 插入case语句 Ctrl Alt c

InsertDo 插入do语句 /*支持块输入*/ Ctrl Alt d

InsertElse 插入else语句 /*支持块输入*/ Ctrl Alt e

InsertFor 插入for语句 /*支持块输入*/ Ctrl Alt f

InsertIf 插入if语句 /*支持块输入*/ Ctrl Alt I

InsertSwitch 插入switch语句 Ctrl Alt s

InsertWhile	插入while语句 /*支持块输入*/	Ctrl Alt w
InsIfdef	插入#ifdef语句 /*支持块输入*/	Ctrl 3
PredefIfStr	插入#if语句 /*支持块输入*/	Alt 3
InsertReviseAdd	插入添加说明 /*支持块输入*/	Ctrl Shift a
InsertReviseDel	插入删除说明 /*支持块输入*/	Ctrl Shift r
InsertReviseMod	插入修改说明 /*支持块输入*/	Ctrl Shift m