和ARM体系结构相关的选项

这些是为ARM（Advanced RISC Machines）结构定义的“-m”开关：

-mapcs-frame	对所有函数都生成一个遵从ARM程序调用标准的堆栈帧，即使在正确执行代码无需严格这么做时。使用此开关时指定“-fomit-frame-pointer”将不产生叶函数的堆栈帧。缺省情况下是“-mno-apcs-frame”。
-mapcs	与“-mapcs-frame”相同。
mapcs-26	产生使用26比特程序计数器的处理器代码，遵从APCS 26比特选项的函数调用标准。此开关替代了编译器早期版本中的“-m2”和“-m3”开关。
-mapcs-32	产生使用32比特程序计数器的处理器代码，遵从APCS 32比特选项的函数调用标准。此开关替代了编译器早期版本中的“-m6”。
-mapcs-stack-check	产生检查每个函数入口可用的堆栈空间大小（实际使用了一些堆栈空间）。如果可用空间不足，将按照所需堆栈空间大小调用函数“__rt_stkovf_split_small”或者“__rt_stkovf_split_big”。运行时的系统需要提供这些函数。缺省情况下是“-mno-apcs-stack-check”，因为这样产生的代码较小。
-mapcs-float	使用浮点寄存器传递浮点参数。这是APCS的一个变种。如果目标硬件存在浮点单元，或者代码将执行较多的浮点计算，推荐使用此开关。缺省情况下是“-mno-apcs-float”，因为使用开关“-mapcs-float”将导致只有整数的代码大小稍有增加。
-mapcs-reentrant	产生可重入的位置独立代码。等价于“-fpic”开关。缺省情况下是“-mno-apcs-reentrant”。
-mthumb-interwork	产生支持ARM和THUMB指令集间调用的代码。不使用此开关，在一个程序里就不能可靠地使用两个指令集。缺省情况下是“-mno-thumb-interwork”，因为指定了“-mthumb-interwork”产生的代码稍微大一些。
-mno-sched-prolog	禁止对函数prolog里的指令重新排序，或者把这些指令与函数体里的指令进行合并。这就意味着所有函数都以一组可识别的指令开始（实际上是一小组不同函数开端中的一个），如果函数处于可执行代码段中，可以使用此信息定位函数的起始位置。缺省情况下是“-msched-prolog”。
-mhard-float	产生包含浮点指令的输出。这是缺省情况。
-msoft-float	产生包含浮点库调用的输出。注意：所需的库不是所有ARM目标机都提供的。一般使用机器惯用的C编译器的配置，但在交叉编译时无法直接

	获得，必须进行一些整理为交叉编译提供合适的库函数。“-msoft-float”改变了输出文件的调用规范，所以只有程序的所有部分都使用此开关编译才起作用。特别是需要使用“-msoft-float”开关编译与GNU CC一起获得的“libgcc.a”库才行。
-mlittle-endian	产生运行在little-endian模式下的处理器代码。这是所有标准配置的缺省情况。
-mbig-endian	产生运行在big-endian模式下的处理器代码；缺省情况是产生运行在little-endian模式下的处理器代码。
-mwords-little-endian	此开关只用于为big endian处理器产生代码。产生字序为little-endian但字节顺序为big-endian的代码。也就是说，字节顺序为“32107654”格式。注意：只有在要求与2.8之前版本编译器产生的big-endian ARM处理器代码兼容时才用到此开关。
-mshort-load-bytes	不试图通过从一个没有边界对齐的地址载入一个字的方式载入半个字（如“short”）。某些目标机的MMU被配置成用陷阱捕获没有边界对齐的载入；使用此开关产生的代码在这些环境下是安全的。
-mno-short-load-bytes	使用没有边界对齐的字载入半个字（如“short”）。此开关产生的代码效率较高，但MMU有时被配置成用陷阱捕获这些指令。
-mshort-load-words	与“-mno-short-load-bytes”相同。
-mno-short-load-words	与“-mshort-load-bytes”相同。
-mbsd	此开关只用于RISC iX。仿真原始的BSD模式编译器。这是没有指定“-ansi”时的缺省情况。
-mxopen	此开关只用于RISC iX。仿真原始的X/Open模式编译器。
-mno-symrename	此开关只用于RISC iX。不要在代码汇编后运行汇编程序随后的处理程序“symrename”。一般在准备与RISC iX C库连接时需要修改某些标准符号，此开关忽略这一过程。为交叉编译构建的编译器不运行汇编程序随后的处理程序。
-mcpu=	此开关指定目标ARM处理器的名称。GCC使用这个名称来确定产生汇编代码时可用的指令类型。可用的名称是： arm2、arm250、arm3、arm6、arm60、arm600、arm610、arm620、arm7、arm7m、arm7d、arm7dm、arm7di、arm7dmi、arm70、arm700、arm700i、arm710、arm710c、arm7100、arm7500、arm7500fe、arm7tdmi、arm8、strongarm和strongarm110。
-march=	此开关指定目标机ARM结构的名称。GCC使用这个名称来确定产生汇编代码时可用的指令类型。此开关可以同“-mcpu=”开关结合使用，也可以

	替代“-mcpu=”开关。可用的名称是：armv2、armv2a、armv3、armv3m、armv4和armv4t。
-mfpe=	此开关指定了目标机可用的浮点仿真版本。可用值为2和3。
-mstructure-size-boundary=	所有结构和联合的大小都处理成此开关设置比特数的倍数。可用值为8和32。不同的工具套件所使用的缺省值不同。对于COFF目标机使用的工具套件缺省值为8。指定较大的数字可以产生速度较快、效率较高的代码，但同时会增加程序的大小。若使用结构或联合交换信息，则使用某个值编译的代码可能不可以与使用另一个值编译的代码或库共同使用。鼓励程序员使用值32，因为工具套件未来版本可能会缺省使用值32。
-mthumb-interwork	产生支持THUMB和ARM指令集间调用的代码。不使用此开关，在一个程序里就不能可靠地使用两个指令集。缺省情况下是“-mno-thumb-interwork”，因为使用“-mthumb-interwork”开关产生的代码稍微大一些。
-mtpcs-frame	对所有非叶函数都生成一个遵从Thumb程序调用标准的堆栈帧。（叶函数是不调用任何其它函数的函数）。缺省情况下是“-mno-apcs-frame”。
-mtpcs-leaf-frame	对所有叶函数都生成一个遵从Thumb程序调用标准的堆栈帧。（叶函数是不调用任何其它函数的函数）。缺省情况下是“-mno-apcs-leaf-frame”。
-mlittle-endian	产生运行在little-endian模式下的处理器代码。这是所有标准配置的缺省情况。
-mbig-endian	产生运行在big-endian模式下的处理器代码。
-mstructure-size-boundary=	所有结构和联合的大小都处理成此开关设置比特数的倍数。可用值为8和32。不同的工具套件所使用的缺省值不同。对于COFF目标机使用的工具套件缺省值为8。指定较大的数字可以产生速度较快、效率较高的代码，但同时会增加程序的大小。这两个值可能不兼容。若使用结构或联合交换信息，则使用某个值编译的代码可能不可以与使用另一个值编译的代码或库共同使用。鼓励程序员使用值32，因为工具套件未来版本可能会缺省使用值32

Mark成浅蓝的为Thumb开关