

**HY16F** 系列

IDE 軟體最佳化使用說明書



### 目錄

1	簡介	4
2	最佳化介紹與設置	4
3	使用優化設置需要注意的事項	5
3.1	調試的問題	6
3.2	查詢移去的 SECTIONS	6
3.3	避冤某些程式碼被 OPTIMIZE 的方法	9
3.4	部分代碼 OPTIMIZE 的方法	.10
3.4	<b>1</b> .1 單個文件設置優化:	.10
3.4	4.2 部分 code 設置優化	.12
3.4	4.3 單個函數設置優化	.12
3.5	Optimization 時,避冤 code 順序被改掉的方法	.13
3.6	其他需要注意的事項	.14
4	參考文件	.15
5	修訂記錄	.15



注意:

1、本說明書中的內容,隨著產品的改進,有可能不經過預告而更改。請客戶及時到本公司網站下載更新 http://www.hycontek.com

2、本規格書中的圖形、應用電路等,因第三方工業所有權引發的問題,本公司不承擔其責任。

**3**、本產品在單獨應用的情況下,本公司保證它的性能、典型應用和功能符合說明書中的條件。當使用在客戶 的產品或設備中,以上條件我們不作保證,建議客戶做充分的評估和測試。

4、請注意輸入電壓、輸出電壓、負載電流的使用條件,使IC內的功耗不超過封裝的容許功耗。對於客戶在超 出說明書中規定額定值使用產品,即使是瞬間的使用,由此所造成的損失,本公司不承擔任何責任。

5、本產品雖內置防靜電保護電路,但請不要施加超過保護電路性能的過大靜電。

6、本規格書中的產品,未經書面許可,不可使用在要求高可靠性的電路中。例如健康醫療器械、防災器械、 車輛器械、車載器械及航空器械等對人體產生影響的器械或裝置,不得作為其部件使用。

7、本公司一直致力於提高產品的品質和可靠度,但所有的半導體產品都有一定的失效概率,這些失效概率可 能會導致一些人身事故、火災事故等。當設計產品時,請充分留意冗餘設計並採用安全指標,這樣可以避免事 故的發生。

8、本規格書中內容,未經本公司許可,嚴禁用於其他目的之轉載或複製。



### 1 簡介

本文主要介紹 Andesight 的最佳化設置與其他相關事項

# 2 最佳化介紹與設置

設置步驟:打開 project 後,選擇菜單欄中的 project—>properties 後彈出下圖,在 settings—>optimization 中可選擇需要的優化級別

A Properties for HY16F	188_Style
type filter text	Settings $(\neg \neg \neg \neg \neg$
<ul> <li>Resource</li> <li>Builders</li> <li>C/C++ Build</li> <li>Discovery Options</li> <li>Environment</li> <li>Gozine</li> <li>Settings</li> <li>Target Configuration</li> <li>Tool Chain Editor</li> <li>C/C++ General</li> <li>Project References</li> <li>Run/Debug Settings</li> </ul>	Configuration: Bebug [ Active ]   Manage Configuration:  Tool Settings Build Steps Build Artifact B Binary Parsers E Error Parsers  Mades I Compiler  Proprocessor  Proprocessor  Defer optimization Level None (-00)  Defer optimization flags  Discouries  Defersing  Miscellaneous  Bandes I Linker  Stared
	高 1

Optimization level 的含義可參考如下信息:

Andesight 將代碼的佔用空間和執行速度優化級別各分為3個級別。

一般情況下程式的代碼空間的最佳化會選擇-Os3 和勾選 remove unused section;執行速度最佳化會選擇-O3 和加上-funroll-loops(可參考圖 2)。

(Optimize speed)-O/O1:GCC 將執行減少代碼尺寸和執行時間的優化,對於那些會嚴重影響編譯時間的優化選項,這個級別的優化並不會執行。

(Optimize speed more)-O2:-O的進階,在這一級別 GCC 將會提供所有支援的優化,但這其中並不包括以空間換時間的優化,例如編譯器不會使用循環展開和函數內聯。和-O相比,該選項進一步加快了編譯時間和生成代碼的性能。



(Optimize speed most)-O3:除了-O2 提供的優化選項外,還指定了-finline-functions,-funswitch-loops 和-fgcse-afer-reload 等選項,-O3 優化級別提升生成的可執行檔的速度,但也可能增加它的大小。在有些 情況下也可能會使得程式運行減慢。

(Optimize size most)-Os3 (-Os):這個選項是用來優化代碼尺寸,-Os 打開了所有-O2 級別中不會顯著 增長代碼尺寸的優化選項,同時-Os 還會執行更加優化程式空間占用的選項。因此,有時-Os 可能會影響 程式性能,可以選擇其他較低的優化級別代替-Os

(Optimize size more)-Os2:相比-Os 減少使用-mifc 等選項,只打開部分的代碼優化選項.

(Optimize size)-Os1:代碼空間優化深度較小,對代碼的影響也較小,只打開部分代碼優化選項

圖 2 中的 Remove unused function sections(-ffunction-sections)和 Remove unused data sections (-fdata-sections)其功能是將沒有使用到的函數和數據刪除,以減少代碼佔用空間,如有需要請勾選。在不 使用這兩項功能情況下,程式中調用到的頭文件(如#include "DrvI2C.h"),頭文件裏沒用到的函數同樣會被 編譯佔用空間。

-funroll-loops 可以添加到 other optimization flags 如圖 2,作用是僅對循環次數能夠在編譯時或運行時確 定的循環進行展開,生成的代碼尺寸將變大,對於不同的代碼執行速度可能變快也可能變慢。



各優化級別之間更詳細的功能介紹可參考:

https://gcc.gnu.org/onlinedocs/gcc-4.4.6/gcc/Optimize-Options.html#Optimize-Options

Andes\_Programming\_Guide 文檔

## 3 使用優化設置需要注意的事項



### 3.1 調試的問題

使用以上的優化功能時,由於程式碼被優化,所以在 debug 時需要注意可能會遇到 debug 的指標沒 有對齊到正確的位置,如下圖紅色框部分,由於事先 MCUSTATUSbits.b\_TMAdone 的值已經預設為 0, 所以紅色框部分的程式屬於冗餘會被優化掉,雖然 debug 的指標對齊到紅色框部分,但是裡面的指令可以 通過參考下圖的 dissassembly 彙編指令的執行情況得知沒有被執行,即調試運行的結果是正確的。



## 3.2 查詢移去的 sections

在-ffunction-section enable 時,要注意是否有用的 section 被 optimize 移除,通常是像 if/else 的 condition 可以在 compiling time 推算出來,所以被 gcc 當成 dead code 優化掉,查詢移去 些 sections 的方法:

#### 1).在 Andes C Compiler/Miscellaneous 裡加上 -ffunction-sections -fdata-sections



Settings				
Configuration: Debug [Active]				
🛞 Tool Settings 🎤 Build Steps 🖤 Build Artifact 🖬 Binary Parsers 📀 Error Parsers				
🖉 nds32le-elf-newlib-v2 Configurations 🛛 Other flags 🔤 -c -fmessage-length=0 -ffunction-sections -fdata-sections				
□ □ Werbose (-v)				
- Bupport ( NSI programs (ansi)				
- Support ANSI programs (ansi)				
Gcov (-fprofile-arcs -ftest-coverage)				
- Optimization				
Debugging				
Warnings				
Miscellaneous				

2). 取消勾選 Remove unused sections (因為把原本的選項拆開來設定 Remove unused sections 即為 -ffunction-sections -fdata-sections 及--gc-sections)

A Properties for hello2	
type filter text      Resource     Builders     C/C++ Build     Discovery Options     Environment     Logging     Settings     Target Configuration     Tool Chain Editor     C/C++ General     Project References     Run/Debug Settings	Settings Configuration: Debug [Active]  Tool Settings Puild Steps Puild Artifact Parsers Parsers Tool Settings Puild Steps Puild Artifact Parsers Parsers Tool Settings Perprocessor Perpro

3). 在 Andes C Linker/Muscellaneous 裡加上這 2 個 option :--gc-sections, --print-gc-sections (按下緑色的十字 button 可增加)

Properties for hello2			*
type filter text	Settings		
Resource     Builders     CIC++ Build     Build Yariables     Discovery Options     Environment     Logging     Settings     Target Configuration     Tool Chain Editor     CIC++ General     Project References     Run/Debug Settings	Configuration: Debug [Active] Tool Settings Puild Steps Puild A Symbols Compiler Preprocessor Symbols Directories Directories Maniellaneous Miscellaneous Sharde Linker Schward Linkers Shard Linkers Shard Linkers Shard Linkers	Artifact Binary Parsers S Error Parsers Linker flags Other options (-Xlinker [option])  recorrectionsprint-gc-sections	Mar D



4). 我的原始碼

```
#include <stdio.h>
#include <stdlib.h>
int int1=34;
int int2=56;
void foo(void);
void foo2(void);
int main(void)
  {
  puts("!!!Hello World!!!"); /* prints !!!Hello World!!! */
  return EXIT SUCCESS;
}
void foo()
{
  puts("foo"); /* prints !!!Hello World!!! */
}
void foo2()
{
  puts("foo2"); /* prints !!!Hello World!!! */
}
```

5).Build code,原本的 int1 int2 foo1 foo2 都被刪掉了如下

```
Set Terminal If Target Manager: Local Targets Console CO
CDT Build Console [hello2]
nds321e-elf-gcc -static -Xlinker --gc-sections -Xlinker --print-gc-sections -00 -
e:/andestech/andesight200mcu/toolchains/nds321e-elf-newlib-v2/bin/../lib/gcc/nds3
e: Removing unused section '.data.int1' in file './src/hello2.o'
e:/andestech/andesight200mcu/toolchains/nds321e-elf-newlib-v2/bin/../lib/gcc/nds3
e: Removing unused section '.data.int2' in file './src/hello2.o'
e:/andestech/andesight200mcu/toolchains/nds321e-elf-newlib-v2/bin/../lib/gcc/nds3
e: Removing unused section '.data.int2' in file './src/hello2.o'
e:/andestech/andesight200mcu/toolchains/nds321e-elf-newlib-v2/bin/../lib/gcc/nds3
e: Removing unused section '.text.foo' in file './src/hello2.o'
e:/andestech/andesight200mcu/toolchains/nds321e-elf-newlib-v2/bin/../lib/gcc/nds3
e: Removing unused section '.text.foo' in file './src/hello2.o'
e:/andestech/andesight200mcu/toolchains/nds321e-elf-newlib-v2/bin/../lib/gcc/nds3
e: Removing unused section '.text.foo2' in file './src/hello2.o'
```

6).在 project 出現紅色 x 不用理會它。只是因為有輸出 message。 它還是能正常產生結果。有.adx 檔輸出。



🖹 💕 hello2
🗊 🐝 Binaries
🖃 👘 Includes
🚊 🔁 src
😟 🔂 hello2.c
🖻 👝 Debug
🗈 👝 output
🗊 🗁 src
🛓 救 hello2.adx - [Andes/le]
扄 makefile
扄 objects.mk
🔤 🚡 sources.mk
👄 L.NO

# 3.3 避冤某些程式碼被 optimize 的方法

參考語法: <u>http://gcc.gnu.org/onlinedocs/gcc/Funct ... agmas.html</u>

下面提供 2 個寫法,其中寫法 2 會有 warning 出現,說這個語法在這個 machine 不支援,其實它的結果 是正確的。

備註:關於設置單個函數的優化級別 optimize ("Ox")是不支持"Os1" and "Os2",因為"Os1" and "Os2" 是由 Andes 定義的,不是標準的 GCC 優化級別

寫法 1: 只針對 1 個 function ( 即 add() )。

```
___attribute___((optimize("O0"))) //設置優化級別屬性為不優化
int add (int a, int b )
{
    int x = a;
    int y = b;
    return x + y;
    }
    int main ()
    {
        int r = 1;
        int a = r;
        int b = r;
        func ();
        return 0;
    }
```



寫法 2:包含在裡面的 code 都不會被 optimize。 注意!這個寫法不一定要以 function 為範圍, 可以任意選取一段 code。

```
#pragma GCC push_options
#pragma GCC optimize ("O0")
                                 //優化級別為"O0"
int add (int a, int b)
{
int x = a;
int y = b;
return x + y;
}
#pragma GCC pop_options
int main ()
{
int r = 1;
int a = r;
int b = r;
func ();
return 0;
}
備註:#pragma GCC push_options
      #pragma GCC pop_options
      這 2 行是表示會先把原來的 option push 進去,
      例如是-Os,後面再把-Os pop回來,恢復原來的 optimize 設定。
```

# **3.4** 部分代碼 optimize 的方法

**3.4.1** 單個文件設置優化: 鼠標右擊源代碼文件後,參照下圖進行優化級別設置







#### 3.4.2 部分 code 設置優化級別

參考語法:

http://gcc.gnu.org/onlinedocs/gcc/Funct ... agmas.html

代碼如下:

#pragma GCC push\_options

#pragma GCC optimize ("O0") //優化級別為"O0",可以設置為其他級別除了 Os1 和 Os2

// code ...

//程式碼

#pragma GCC pop\_options

備註: #pragma GCC push\_options

#pragma GCC pop\_options

這 2 行是表示會先把原來的 option push 進去,

例如是-Os,後面再把-Os pop 回來,恢復原來的 optimize 設定。



3.4.3 單個函數設置優化級別

```
對單個函數的優化級別進行設置,以下代碼是設置成"Os"
void __attribute__ ((optimize ("Os"))) __cpu_init()
{
unsigned int tmp;
/* turn on BTB */
tmp = 0x0;
__nds32__mtsr(tmp, NDS32_SR_MISC_CTL);
/* Set PSW INTL to 0 */
tmp = __nds32__mfsr(NDS32_SR_PSW);
tmp = tmp & 0xffffff9;
```

/\* ....\*/

return;

}

關於設置單個函數的優化級別 optimize ("Ox")是不支持"Os1" and "Os2",因為"Os1" and "Os2"是由 Andes 定義的,不是標準的 GCC 優化級別

# 3.5 Optimization 時,避免 code 順序被改掉的方法

#### 如果不要讓 optimization 改變程式碼的順序,

可以加2個 options: -fno-schedule -insns -fno-schedule-insns2

Settings	
Configuration: Debug [Active]	M
<ul> <li>Tool Settings Puild Steps Puild Artif</li> <li>ands32le-elf-mculib-v2j Configurations</li> <li>Andes C Compiler</li> <li>Preprocessor</li> <li>Symbols</li> <li>Directories</li> <li>Optimization</li> <li>Debugging</li> <li>Warnings</li> <li>Miscellaneous</li> </ul>	act       Binary Parsers       Servor Parsers         Other flags       -c -fmessage-length=1 -fno-schedule-insms -fno-schedule-insms2         Verbose (-v)       Support ANSI programs (-ansi)         Gcov (-fprofile-arcs -ftest-coverage)



### 3.5 其他需要注意的事項

(1)和硬體相關的變數與 Register 要加上 volatile.加上 volatile 這個關鍵字避兒變數被最佳化,例如以下程式:

XBYTE[2]=0x55;

XBYTE[2]=0x56;

XBYTE[2]=0x57;

XBYTE[2]=0x58;

編譯器優化功能會認為只有XBYTE[2]=0x58(即忽略前三條語句,合併為一條語句)。如果鍵入volatile, 則編譯器會逐一的進行編譯並產生四條代碼相應的機器代碼。

(2)在 c code 宣告變數有加 volatile,在 extern 時變數也是要加上 volatile

(3)中斷服務程式中修改供其他函數使用的變量建議加上 volatile,因為開啓優化功能後,對變量的訪問可能借用暫存器,而不是直接訪問變量地址,結果可能導致變量地址的值改變沒有反映到暫存器上面

(4)多線程應用中被幾個任務共享的變量也建議加上 volatile



4 參考文件

Andes\_Programming\_Guide\_v1.5\_PG009\_V2.1 部分内容摘自 <u>http://forum.andestech.com</u>

# 5 修訂記錄

以下描述本檔差異較大的地方,而標點符號與字形的改變不在此描述範圍。

版本	頁次	變更摘要	日期
V01	All	初版發行	2015/03/11