九齊科技股份有限公司
Nyquest Technology Co., Ltd.

使用手冊

NYC_NY8L

C Compiler for NY8L series

Version 1.1

Jul. 3, 2018

改 版 記 錄

版本	*日期*	*內 容 描 述*	*修正頁*
1.0	2018/05/31	新發佈。	-
1.1	2018/07/03	修改 ISR 使用方式。	7, 8

目　錄

1 簡介

NYC_NY8L 為針對九齊科技的 NY8L 系列 8 位元 MCU IC 而提供的 C 語言編譯器（Compiler）。它被上層開發工具軟體 *NYIDE* 所呼叫以編譯 C 程式，並內建獨立的組譯器（Assembler）進一步組譯及連結目的檔來產生.bin 檔，然後.bin 檔可下載到板子或燒錄到 OTP IC。

1.1 如何使用本手冊

1. 簡介

為何需要 *NYC_NY8L* 與安裝 *NYC_NY8L* 的基本需求。

2. 使用 *NYC_NY8L*

如何透過 *NYIDE* 使用 *NYC_NY8L*。

3. 語法與使用

介紹 *NYC_NY8L* 的語法與使用方式。

1.2 系統需求

底下列出使用 *NYC_NY8L* 的系統需求。

- Pentium 1.3GHz 或更高級處理器，Win7、Win8、Win10 作業系統。
- 至少 2G SDRAM。
- 至少 2G 硬碟空間。
- Visual C++ 2015 Redistributable（32bit）。

1.3 安裝 *NYC_NY8L*

請聯繫九齊科技來取得 *NYC_NY8L* 的安裝程式檔，雙擊執行後進入安裝程式嚮導，然後依照畫面指示將可輕易完成安裝流程。如果您的電腦沒有安裝 Visual C++ 2015 Redistributable（32bit），在安裝過程中會自動下載，此步驟需要網際網路連線。若您的環境沒有網際網路連線，請預先至微軟網站下載 Visual C++ 2015 Redistributable（32bit）並安裝。

2 使用 *NYC_NY8L*

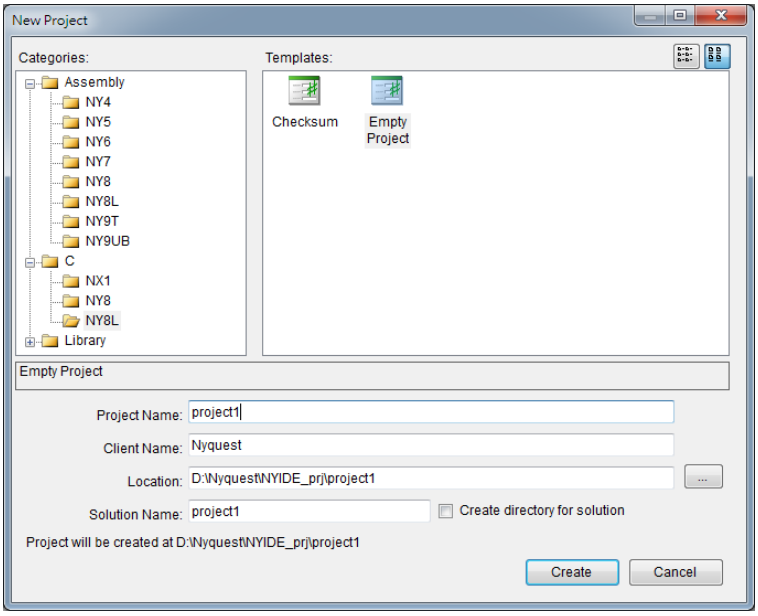
當使用者使用 NY8L 軟體開發工具 *NYIDE* 編寫 C 程式後，在 *NYIDE* 介面上按下 Build 時，軟體開發工具會自動尋找並使用已安裝於電腦上的 *NYC_NY8L* 作編譯。底下將說明透過 *NYIDE* 使用 *NYC_NY8L* 的流程。

2.1 透過 *NYIDE* 使用 *NYC_NY8L*

NYIDE 為九齊科技提供以開發 NY4 / 5 / 6 / 7 / 8 / 9T / 9UB / NX1 系列微控制器程式的整合性開發工具，主要目的為提供使用者以組合語言（Assembly）和 C 語言來編寫程式，並擁有建置和強大的除錯功能。當使用 *NYIDE* 開發 NY8L 專案，在建置和除錯時，*NYIDE* 會自動尋找並使用安裝於電腦上的 *NYC_NY8L* 工具鏈。底下簡單的介紹使用 *NYIDE* 開發 NY8L 專案。更詳細的操作方式，請參考 *NYIDE* 使用手冊。

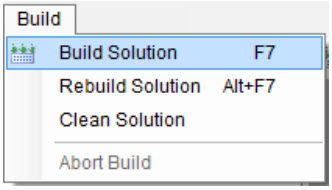
2.1.1 建立新專案

開啟 *NYIDE*，選擇建立新專案。在 New Project 視窗內，左邊的 Categories 選擇 C，然後選擇 NY8L。指定專案名稱及類型後，按下 Create。*NYIDE* 將會自動產生必要檔案，專案已於可建置狀態。



2.1.2 建置

在 *NYIDE* 主畫面上的功能表選擇 Build / Build Solution 進行建置（或按下快捷鍵 F7），即會呼叫 *NYC_NY8L* 執行建置動作。若建置成功會在專案目錄產生.bin 檔案，以進一步提供下載或燒錄。



3 語法與使用

NYC_NY8L 支援標準的 ANSI C89 語法，並且針對 NY8L 系列 IC 新增了一些特殊語法。

3.1 標準 C 語法

NYC_NY8L 支援標準的 ANSI C89 語法，有關詳細語言定義請參考：Standard ISO/IEC 9899 (http://www.open-std.org/jtc1/sc22/wg14/www/standards.html#9899)

3.1.1 註解

註解支援兩種格式，以雙斜線起頭的單行註解，以及/*起頭，至*/結尾的多行註解。

範例：

```
// single line comment

/*
Multi line comment
*/
```

3.1.2 資料型態

底下表格列出 *NYC_NY8L* 所使用的基本資料型態及允許的資料範圍。

型態	長度	範圍
char	1 byte	0 ~ 255
signed char	1 byte	-128 ~ 127
short	2 bytes	-32768 ~ 32767
unsigned short	2 bytes	0 ~ 65535 (0xFFFF)
int	2 bytes	-32768 ~ 32767
unsigned int	2 bytes	0 ~ 65535 (0xFFFF)
long	4 bytes	-2147483648 ~ 2147483647
unsigned long	4 bytes	0 ~ 4294967295 (0xFFFFFFFF)

3.2 擴充語法

3.2.1 保留字

底下列出所有保留字，使用者定義之符號不可和保留字相同。

__Pragma	__near__	else	near	unsigned
__AX__	asm	enum	register	void
__A__	auto	extern	restrict	volatile
__EAX__	break	far	return	while
__X__	case	fastcall	short	
__Y__	cdecl	float	signed	
__asm__	char	for	sizeof	
__attribute__	const	goto	static	
__cdecl__	continue	if	struct	
__far__	default	inline	switch	
__fastcall__	do	int	typedef	
__inline__	double	long	union	

3.2.2 中斷服務程式（ISR）

NY8L 系列有多個中斷，每個中斷都有一個 16 位元的中斷向量，用以存放中斷服務程式（ISR）的位址。當中斷發生時，需要暫停目前之程序，並保存當前的系統狀態，例如 X、Y、A 三個暫存器，然後根據中斷的種類查找中斷向量，取得相應的中斷服務程式的位址，並執行中斷服務程式，執行結束恢復狀態並以 `RTI` 指令返回中斷前位址。使用 NY8L C，只要在中斷服務程式標示屬性 `INTERRUPT_XXX`，*NYC_NY8L* 就會自動產生保存狀態及在中斷向量定義中斷服務程式位址等所必須的程式碼。下面的範例程式示範如何建立 TM1 中斷服務程式。

```
#include <ny8l.h>

void tm1_isr(void) INTERRUPT_TM1 {
    INTF = ~C_INTF_TM1_Flag;
}
```

可用的函數屬性清單如下，可以在安裝目錄的 `include/ny8l_common.h` 檔案找到到屬性的原始定義。

INTERRUPT_TM2	INTERRUPT_TM1	INTERRUPT_TM0	INTERRUPT_FT
INTERRUPT_ST	INTERRUPT_EXT	INTERRUPT_ADC	INTERRUPT_KSB
INTERRUPT_NMI	INTERRUPT_BRK		

NYC_NY8L 1.10 以上才提供中斷服務程式的函數屬性，在 1.10 以前的版本必須由使用者自行編寫內建的組合語言 vector.s 檔案來定義中斷向量。當舊版 *NYC_NY8L* 所開發的專案移至 1.10 版以上，如果有使用此函數屬性，必須自行刪除舊版專案中的 vector.s 檔案相應的定義，以免發生衝突，建議若有使用

函數屬性就將 vector.s 檔案刪除全部改用 C 的函數屬性。在 1.10 版本以上中斷向量可選擇使用匯編或C語言的函數屬性，但建議擇其一使用，不要兩個混合使用。

3.2.3 中斷服務程式 （ISR）使用組合語言

在 C 語言專案中可以混合使用副檔名為.s 的組合語言檔案，也可以使用組合語言來撰寫中斷服務程式。底下提供 TM2 中斷服務程式的組合語言範例。

```
    .export Vector_tm2
    .import ___exit_isr_pop_axy
.segment "CODE"
Vector_tm2:
    phy
    phx
    pha
    lda     #$FE
    sta     _INTF
    jmp     ___exit_isr_pop_axy
```

在上面的範例中，匯出符號 Vector_tm2 為內建函數庫所規定的中斷服務程式名稱，若名稱不符合，編譯器將無法把中斷服務程式地址定義到中斷向量。中斷服務程式名稱清單如下：

Vector_tm2	Vector_tm1	Vector_tm0	Vector_ft
Vector_st	Vector_ext	Vector_adc	Vector_ksb
Vector_nmi	Vector_brk		

而匯入符號 `___exit_isr_pop_axy` 為內建函數庫所提供的函數，將系統狀態復原並返回中斷前的位址，內容如下：

```
___exit_isr_pop_axy:
        pla
        plx
        ply
        rti
```

3.2.4 暫存器位址定義

所有 NY8L IC 的特殊暫存器都已經被定義在"NY8L.h"的檔案中，並置放於程式安裝資料夾的 include 目錄下。建議使用者直接使用該標頭檔，不須自行定義特殊暫存器。

3.2.5 讀寫指定位址

讀取絕對位址記憶體的方式是將所要讀取的位址作為立即值轉型為指標再取值。例如以下範例讀取於絕對位址 0x7F0 及 0x7F1 的 2 bytes int。

```
uint16_t value = (*(uint16_t*)0x7f0);
```

而寫入絕對位址暫存器方法相似，以下範例將記憶體位址 0x80 寫入值 0x12。

```
*(uint8_t*)0x80 = 0x12;
```

指標的型態將決定所要存取的單位大小。

3.2.6 內嵌組合語言（Inline assembly）

在 C 語言之中可以內嵌組合語言，使用__asm__關鍵字即可插入組合語言程式。

範例：

```
__asm__(“nop”);
```

如果需要在組合語言之中使用 C 語言定義的變數，可以使用%v 格式化文字。

範例：

```
__asm__(“lda %v”, my_var);
```

Inline assembly 跟一般 C 語言產生的組合語言一樣會受到最佳化的影響。目前沒有辦法單獨將 inline assembly 程式片段排除在最佳化之外，只能選擇整個函數是否要開啟最佳化。如果最佳化會造成您的困擾，可以使用 volatile 關鍵字讓包含__asm__的函數不被最佳化。

範例：

```
__asm__ volatile("lda #0");
```

3.3 系統標頭檔

在 NYC_NY8L 安裝目錄中的 include 資料夾有所有 C 語言使用的標頭檔（header file），本章節介紹這些標頭檔的內容及使用方法。

3.3.1 特殊指令巨集

ny8l_common.h 檔案定義了常用的特殊組合語言指令巨集，以較為低階的方式有效率地控制 IC 行為，使用者可在適當的時機呼叫這些巨集。

巨集	說明
CLI()	開啟中斷功能
SEI()	關閉中斷功能
CLRWDT()	清除 watch dog 計時器
SLEEP()	進入睡眠模式
ROM_BANK(addr)	取得(addr >> 16) & 0xFF 的值(bank address)

3.3.2 特殊功能暫存器（SFR）

ny8l_common.h 檔案定義了特殊功能暫存器（SFR）的名稱，使用者可以藉由這些定義存取 SFR。但是不建議使用者直接引用 ny8l_common.h，建議只引用標頭檔 NY8L.h。使用者不應該更改 *NYC_NY8L* 檔案，因為此檔案必須與內建函數庫中的命名對應，更改此檔案將造成連結失敗。

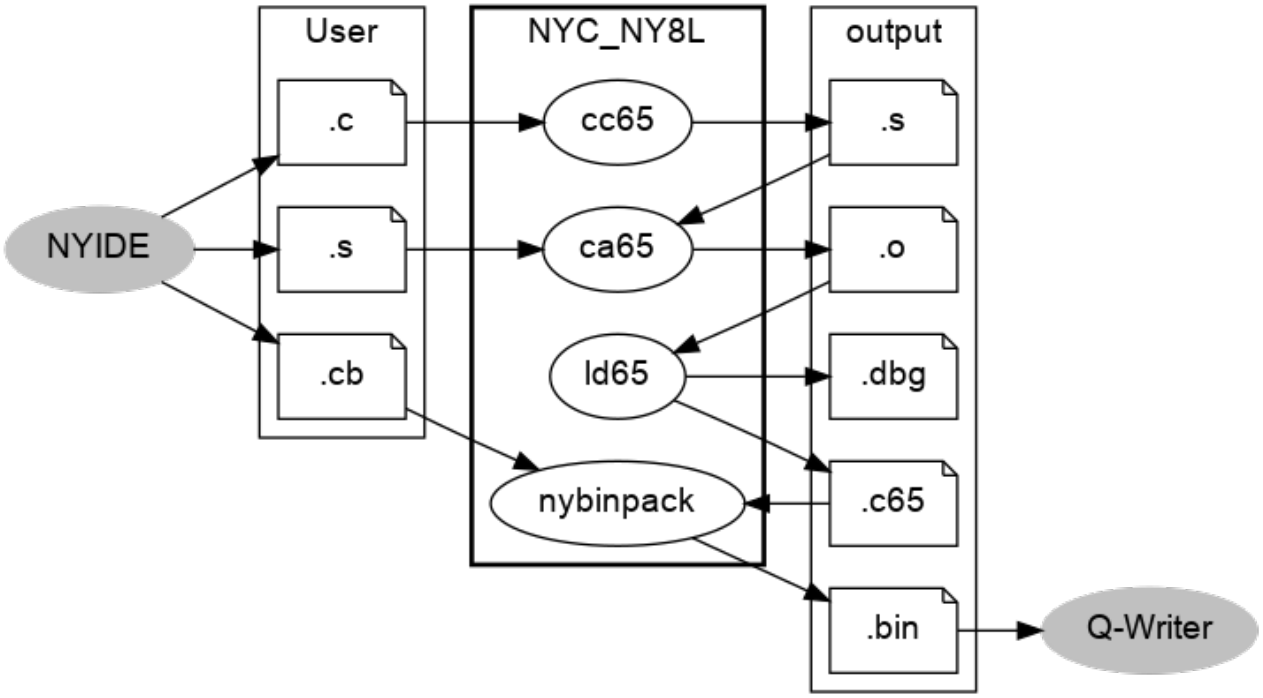
3.4 選項

使用 *NYIDE* 開發 C 語言專案，可以設定若干專案建置選項。這些選項可以控制編譯器、組譯器、連結器的行為，點選功能表的專案（Project）/專案設定（Project Settings）可開啟設定界面。

- 產生組合語言列表檔（Generate ASM listing file）：組譯完成產生列表檔，檔名為 *.lst。不勾選此選項可加快編譯速度。
- 產生連結列表檔（Generate listing file）：連結後產生列表檔，檔名為 *.link.lst。此檔為最終.bin 檔案的反組譯結果，不勾選此選項可以加快編譯速度。
- 產生位址對應檔（Generate map file）：連結後產生位址對應檔，檔名為 *.map。此檔包含位址分配資訊，不勾選此選項可加快編譯速度。
- 最佳化（Optimization）：最佳化所產生的程式碼。C 語言編譯為組合語言的最佳化，可以產生較為精簡的程式碼。
- 開機清空記憶體（Clear RAM to zero on startup）：在 IC 開機尚未進入使用者 main 函數前將所有記憶體設為 0。
- 中繼檔案目錄（Intermediate Directory）：編譯過程中的中繼檔案存放目錄。
- 組合語言引用路徑（ASM include path）：指定組合語言.s 檔的引用搜尋路徑。預設為專案根目錄及 NYC_NY8L 安裝目錄的 asminc 資料夾。使用者可以增加自定義的路徑。
- 引用檔案路徑（Include path）：設定 C 語言 include 關鍵字引用標頭檔的搜尋路徑。預設的路徑為專案根目錄及 NYC_NY8L 安裝目錄的 include 資料夾。使用者可以增加自定義的路徑。

3.5 開發流程

使用 *NYIDE* 編寫 C 語言程式，並設定專案所需的組態檔.cb，*NYIDE* 建置時自動呼叫 *NYC_NY8L* 之中的 *cc65.exe* 產生組合語言檔案.s，*ca65.exe* 組譯組合語言檔案，ld65.exe 連結所有目的檔案，nybinpack.exe 合併組態檔與程式檔，產生最終.bin 檔。最後可以藉由 *Q-Writer* 將.bin 檔燒錄至 IC。



3.6 使用建議

底下提出一些開發 C 語言專案的建議。

- 盡量使用無符號（unsigned）變數，在部份的運算不用判斷正負號執行速度會比較快。
- 運算式之中不要交互使用常數與變數，將常數集中才能有效的最佳化。
  例如 1 + a + 2 就是不好的寫法，1 跟 2 無法在編譯時期計算。建議寫成 a+1+2，如此一來 1+2 可在編譯時期計算，執行期間只需要計算 a+3 即可。