

HT32F125x 低功耗模式

文件編碼：HA0284T

概述

簡介

此應用範例用於幫助系統設計者對 HT32F125x 系列產品的低功耗模式在軟硬體實現方面有簡要地瞭解。用戶可以學習如何使用 HT32F125x 產品及如何合理安排低功耗模式，以實現 HT32F125x 在低功耗應用方面的最優化處理。

相關檔

- HT32F125x 規格書
- HT32F125x 使用手冊

功耗和喚醒時間測量

簡介

閱讀此節後，用戶可以瞭解 HT32F125x 低功耗模式功耗和喚醒時間的測量問題。

功耗測量

用戶可通過此應用範例附帶的 Zip 檔中提供的韌體來測量 HT32F125x 的功耗。該韌體存放在 CurrentMeasurements 文件夾中。

可在以下低功耗模式中進行功耗測量：

- 休眠模式
休眠模式下僅有 Cortex™-M3 CPU 內核的時鐘停止運行。此模式下的功耗可由時鐘源及有效的外設決定。為涵蓋此模式的所有功能，測量時用到不同時鐘源（HSI 和 HSE）、時鐘頻率（1MHz~72MHz）、APB 外設配置（所有外設時鐘開啟或所有外設時鐘關閉——包括 Flash 和 SRAM）。
- 深度休眠模式 1
深度休眠模式 1 中，VDD18 電源域的所有時鐘停止運行。高速振盪器（HSI、HSE）和 PLL 停止振盪，VDD18 電壓源為 LDO（處於低電流模式時）。
- 深度休眠模式 2
除了 VDD18 電壓源為 DMOS 外，深度休眠模式 2 與深度休眠模式 1 相同。
- 暫停模式：關閉 VDD18 電源域，保留 VDD33 和 VBAK 電源域。

注意：1. 不同低功耗模式的更多詳情請參考 HT32F125x 用戶手冊。

2. 處於休眠模式和深度休眠模式時，所有未使用的 I/O 腳都配置為輸入浮空狀態，斯密特觸發輸入除能，因此，這些 I/O 引腳的功耗為 0。

如何測量功耗

HT32F125x 開發板的功耗測量可通過兩種方式測得：可利用電錶代替跳線 J25，並且由一個外部供電給開發板；通過使用 USB 線。

韌體描述

頭文件 (main.h) 中，可通過選擇幾個#define 參數化範例。

- 取消相應行定義的注釋，選擇休眠模式所需的時鐘配置


```
#define HSE_PLL_ON
#define HSE_PLL_ON_72MHz
//define HSE_PLL_ON_36MHz
//define HSE_PLL_ON_18MHz

//define HSE_PLL_OFF
//define HSE_PLL_OFF_8MHz
//define HSE_PLL_OFF_1MHz

//define HSI_PLL_ON
//define HSI_PLL_ON_72MHz
//define HSI_PLL_ON_36MHz
//define HSI_PLL_ON_18MHz

//define HSI_PLL_OFF
//define HSI_PLL_OFF_8MHz
//define HSI_PLL_OFF_1MHz
```
- 取消相應行定義的注釋，選擇所需的低功耗模式


```
#define SLEEP
//define DEEP_SLEEP_1
//define DEEP_SLEEP_2
//define POWER_DOWN
```
- 取消相應行定義的注釋，選擇所需外設時鐘（門控時鐘）


```
//define SLEEP_ALLPERIPH_ENABLE
#define SLEEP_ALLPERIPH_DISABLE
```
- 休眠模式下，用戶可通過取消相應行定義的注釋來選擇所需的 FMC 時鐘選項


```
//define SLEEP_FMC_ENABLE
#define SLEEP_FMC_DISABLE
```
- 休眠模式下，用戶可通過取消相應行定義的注釋來選擇所需的 SRAM 時鐘選項


```
//define SLEEP_SRAM_ENABLE
//define SLEEP_SRAM_DISABLE
```

注意：無論時鐘源來自 HSI 還是 HSE，若系統時鐘頻率等於或小於 8MHz，PLL 都將關閉。

執行低功耗範例後，若用戶想重新載入 Flash 記憶體的內容，需將啟動模式由主 Flash 切換到 SRAM，並且要按下重置鍵。此舉是因為當 HT32F125x 處於低功耗模式時，調試器不能連接到 HT32F125x。啟動模式需重新配置到主 Flash。而後，通過重啟目標板開始進行測量。

測量結果

測量結果如表 1 和表 2 所示。

表 1 休眠模式下功耗測量結果

條件	f _{HCLK}	所有 APB 外設使能	所有 APB 外設除能
運行 HSE。 AHB 預除頻器用於減小頻率， 休眠模式下 FMC 和 SRAM 時鐘關閉	72MHz	26.42mA	6.76mA
	36MHz	15.64mA	5.25mA
	18MHz	10.1mA	4.49mA
	8MHz	4.39mA	1.43mA
	1MHz	2.21mA	1.13mA
運行 HSE。 AHB 預除頻器用於減小頻率， 休眠模式下 FMC 和 SRAM 時鐘開啟	72MHz	28.27mA	8.67mA
	36MHz	16.57mA	6.21mA
	18MHz	10.57mA	4.98mA
	8MHz	4.6mA	1.64mA
	1MHz	2.24mA	1.16mA
運行 HSI。 AHB 預除頻器用於減小頻率， 休眠模式下 FMC 和 SRAM 時鐘關閉	72MHz	26.39mA	6.63mA
	36MHz	15.5mA	5.12mA
	18MHz	9.97mA	4.36mA
	8MHz	4.33mA	1.36mA
	1MHz	2.15mA	1.06mA

表 2 深度休眠模式和暫停模式下功耗測量

模式	條件	V _{DD33} /V _{BAT} = 3.3V
深度休眠 1 模式	V _{DD18} 來自 LDO，LDO 處於低功耗模式， RTC on，使用 LSI	67uA
深度休眠 2 模式	V _{DD18} 來自 DMOS，LDO Off，RTC on，使用 LSI	11.5uA
暫停模式	V _{DD18} Off，RTC on，使用 LSI	4.5uA

喚醒時間測量

此節描述了如何測量 HT32F125x 由不同低功耗模式下喚醒的時間。此應用範例提供了相關的韌體，韌體位於提供的 Zip 壓縮包內的 WakeUpTiming 文件夾下。

喚醒時間定義

- 休眠模式和深度休眠模式
喚醒時間起始於 RTCOUT (PB10) 的上升緣，結束於 WFE 後執行完第一條指令。
- 暫停模式
從暫停模式中喚醒後，與遇到重定條件相同，程式重新執行。暫停模式的喚醒時間介於 RTCOUT(PB10)的上升緣與執行完第一條指令之間。

如何測量喚醒時間

系統由低功耗模式喚醒後，PA5 腳被設定以測量喚醒時間，示波器需連接在 PA5 和 PB10 之間。喚醒時間介於 PB10 (RTCOUT) 的上升緣和 PA5 的上升緣之間。

韌體描述

頭文件 (main.h) 中，可通過選擇幾個 #define 參數化範例。

- 取消相應行定義的注釋，選擇所需的低功耗模式

```

// #define SLEEP
// #define DEEP_SLEEP_1
// #define DEEP_SLEEP_2
// #define POWER_DOWN
    
```

- 取消相應行的注釋，選擇所需的系統時鐘源

```

/* Define the system clock */
#define HCLK_HSI
// #define HCLK_HSI_PLL
// #define HCLK_HSE
// #define HCLK_HSE_PLL
    
```

在主文件 (main.c) 中。進入低功耗模式前，PA5 配置為輸出推挽式，重定到低電壓狀態。

由低功耗模式喚醒後：

- 如果從休眠模式和深度休眠模式喚醒，則執行直接寫入 GPIOA_SRR (輸出設定重置控制暫存器) 以設定 PA5 為高準位。
- 如果從暫停模式喚醒，則在代碼啟動時 PA5 腳應被設定 (此部分代碼位元於啟動檔中，如下所示)。

為測量暫停模式的喚醒時間，需先配置和設定 PA5 腳。為此，添加如下彙編代碼到工具啟動檔：

```

/* Enable peripheral clocks of AFIO and GPIOA */
LDR R0, = 0x4008802C
LDR R1, = 0x00014000
STR R1, [R0]
/* PA5 output high */
LDR R0, = 0x4001A000
LDR R1, = 0x0020
STR R1, [R0, #0x24]
STR R1, [R0]
    
```

注意：執行低功耗範例後，若用戶想重新載入 Flash 記憶體的內容，需將啟動模式由主 Flash 切換到 SRAM，並且要按下重置鍵。此舉是因為當 HT32F125x 處於低功耗模式時，調試器不能連接到 HT32F125x。啟動模式需重新配置到主 Flash。而後，通過重啟目標板開始進行測量。

測量結果

休眠模式、深度休眠模式和暫停模式的喚醒時間測量結果如表 3 所示。

表 3 喚醒時間測量結果

符號	參數	條件	典型值
tWUSLEEP	從休眠模式中喚醒	在 HSI 時鐘下喚醒	1.04us
tWUDS1	從深度休眠模式 1 中喚醒		17.4us
tWUDS2	從深度休眠模式 2 中喚醒		17.4us
tWUPD	從暫停模式中喚醒		118us
tWUSLEEP	從休眠模式中喚醒	在 HSE 時鐘下喚醒	1.04us
tWUDS1	從深度休眠模式 1 中喚醒		5.04ms
tWUDS2	從深度休眠模式 2 中喚醒		5.04ms
tWUSLEEP	從休眠模式中喚醒	在 PLL 時鐘下喚醒 PLL 使用 HSI	0.2us
tWUDS1	從深度休眠模式 1 中喚醒		146us
tWUDS2	從深度休眠模式 2 中喚醒		146us
tWUSLEEP	從休眠模式中喚醒	在 PLL 時鐘下喚醒 PLL 使用 HSE	0.2us
tWUDS1	從深度休眠模式 1 中喚醒		5.16ms
tWUDS2	從深度休眠模式 2 中喚醒		5.16ms

結論

根據不同的結果，用戶可以在功耗和喚醒時間之間作一權衡，前提條件是：功耗更低、喚醒時間更長。

由於 HSE 和 PLL 需更長的準備時間，故當時鐘源為 HSE 或 PLL 時，深度休眠模式喚醒時間更長。為縮短深度休眠模式的喚醒時間，用戶可以先使用 HSI，然後切換到其他外設時鐘源（系統時鐘、HSE 或 PLL）。

根據應用的限制，用戶應發現最好的權衡結果。

功耗優化

簡介

事實上：微控制器的功耗隨著時鐘頻率的增大而增加。所以，用戶須尋找到最優的功耗/性能比。很多應用中，可通過調整系統/外設頻率達到所需的性能而降低功耗。若系統/外設工作無特別需求，可使用 HT32F125x 的低功耗模式。

應用中使用時鐘配置

本節介紹如何使用 HT32F125x 的時鐘配置，所使用的韌體位於本應用範例的 Zip 壓縮包內的 Run_Mode 文件夾中。

該程式使用 HT32F125x 的 USART 從 RTC 傳送時間。

- 首先，用戶必須使用超級終端調整時間。
- 時間顯示在超級終端上，並且每秒刷新一次。RTC 設定為每秒產生一次中斷。
- 當中斷發生時，捕捉 RTC 計數器的值；計算時間，並使用 USART 傳送出去。

硬體環境

在 HT32F125x 開發板上使用此範例：請參考 HT32F125x 開發板的用戶手冊

- 使用 HT32F125x 開發板封裝中的 USART 線
- DB9 連接器和 PC 串口間必須使用一個零數據機兩端均為母端子的 RS232 線連接。
- 由一個電錶代替跳線 J25 測量功耗。

韌體描述

- 在 PC 上配置超級終端
 - 字長 = 8 bits
 - 一個 Stop 位元
 - 無奇偶校驗
 - 串列傳輸速率 = 115200
 - 流程控制：無
- 配置韌體

頭文件 (main.h) 中，可通過選擇幾個 #define 參數化範例。

 - 定義外設選擇 (門控時鐘)


```

                    // #define ALL_PERIPHERIALS_ENABLE
                    // #define ONLY_USART_RTC_ENABLE
                    
```
 - 定義頻率選擇


```

                    // #define HCLK_72MHz
                    // #define HCLK_8MHz
                    
```
 - 定義應用程式等待 RTC 中斷時切換到休眠模式


```

                    // #define SLEEP_WFI_ON
                    
```

注意：執行低功耗範例後，若用戶想重新載入 Flash 記憶體的內容，需將啟動模式由主 Flash 切換到 SRAM，並且要按下重置鍵。此舉是因為當 HT32F125x 處於低功耗模式時，調試器不能連接到 HT32F125x。啟動模式需重新配置到主 Flash。而後，通過重啟目標板開始進行測量。

測量結果

表 4 在 25°C 時運行模式測量舉例

外設時鐘	頻率	休眠	典型功耗
ALL On	72MHz	No	51.8mA
僅 RTC On	72MHz	No	37.4mA
僅 RTC On	72MHz	Yes	12.7mA
僅 RTC On	8MHz	No	11.1mA
僅 RTC On	8MHz	Yes	2.05mA

結論

為減少功耗，HT32F125x 必須根據用戶需求以最優化配置進行初始化。用戶必須著眼於應用的要求，並配置相應的 HT32F125x。從這個範例中，用戶可以借鑒可能的 HT32F125x 時鐘配置和優化應用程式的功耗。

配置描述如下：

- 系統和外設頻率
若應用程式無需最大頻率下運行，用戶可通過使用 PLL 或預除頻器除頻以減小 HCLK。
- 門控時鐘
為優化功耗，用戶應通過 HT32F125x 門控時鐘選項除能未使用的外設。
- 休眠模式
減少功耗的另一個方法是當應用程式等待事件或者中斷時切換到 HT32F125x 的休眠模式。

在電池產品應用中使用深度休眠模式和暫停模式

簡介

由電池供電的一些應用程式不是一直處於運行中的。此種應用中，微控制器等待外部事件，且需要在暫停運行期間減少功耗。

此應用範例提供了兩個在電池供電應用中如何使用 HT32F125x 的範例（深度休眠模式和暫停模式）。這些範例中，一旦應用程式無需處理，HT32F125x 就切換到低功耗模式。這兩個低功耗模式基於 Cortex-M3 內核的深度睡眠功能和 WFE 指令。

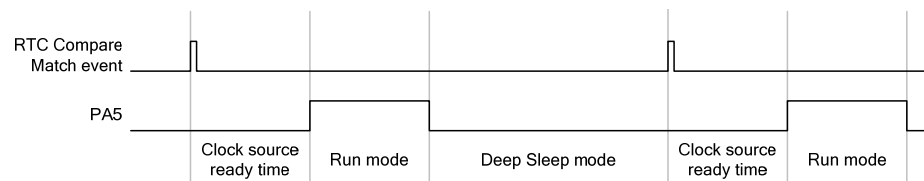
深度休眠模式

本節介紹如何使用 HT32F125x 的 WFE 指令和深度休眠模式，所使用的韌體位於本應用範例的 Zip 壓縮包內的 DeepSleepMode 文件夾中。

該範例執行定期 ADC 轉換並將轉換結果記憶在 RAM 緩衝中。它採用 RTC 比較匹配事件以自動喚醒深度休眠模式。

- 首先，配置備份域、GPIO 和 ADC。
- RTC 正在運行，且由主迴圈中的 RTC 比較匹配觸發每次 ADC 轉換和 ADC 轉換結果的記憶。
- RTC 比較匹配事件將 HT32F125x 由深度休眠模式中喚醒。
- PA5 I/O 顯示了 ADC 轉換和 RTC_CMP 暫存器重新載入所需的時間。

圖 1 深度休眠模式舉例



韌體描述

頭文件 (main.h) 中，可通過選擇幾個 #define 參數化範例。

- 取消相應模式定義的注釋，選擇所需的低功耗模式


```

                //#define DEEP_SLEEP_1
                //#define DEEP_SLEEP_2
            
```
- 選擇迴圈時序，取消相應行的注釋


```

                //#define LOOP_50mS
                //#define LOOP_500mS
                //#define LOOP_1S
                //#define LOOP_3S
                //#define LOOP_5S
            
```

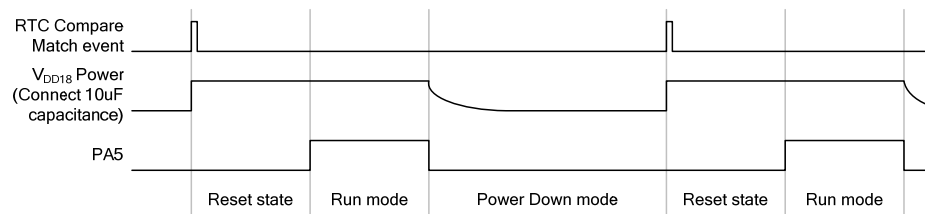
注意：執行低功耗範例後，若用戶想重新載入 Flash 記憶體的內容，需將啟動模式由主 Flash 切換到 SRAM，並且要按下重置鍵。此舉是因為當 HT32F125x 處於低功耗模式時，調試器不能連接到 HT32F125x。啟動模式需重新配置到主 Flash。而後，通過重啟目標板開始進行測量。

暫停模式

本節介紹如何使用 HT32F125x 的暫停模式，所使用的韌體位於本應用範例的 Zip 壓縮包內的 PowerDownMode 文件夾中。

該範例執行定期 ADC 轉換並將轉換結果記憶在 RAM 緩衝中。它採用 RTC 比較匹配事件以自動喚醒暫停模式。

- 首先，配置備份域、GPIO 和 ADC。
- RTC 運行且 RTC 比較匹配事件將 HT32F125x 由暫停模式中喚醒。
- 每次由暫停模式中喚醒後，HT32F125x 從重定模式重啟，且在 HT32F125x 已配置後執行每次 ADC 轉換。
- PA5 I/O 顯示了程式運行所需的時間。



韌體描述

頭文件 (main.h) 中，可通過選擇幾個 #define 參數化範例。

- 選擇迴圈時序，取消相應行的注釋


```

                //#define LOOP_50mS
                //#define LOOP_500mS
                //#define LOOP_1S
                //#define LOOP_3S
                //#define LOOP_5S
            
```

注意：執行低功耗範例後，若用戶想重新載入 Flash 記憶體的內容，需將啟動模式由主 Flash 切換到 SRAM，並且要按下重置鍵。此舉是因為當 HT32F125x 處於低功耗模式時，調試器不能連接到 HT32F125x。啟動模式需重新配置到主 Flash。而後，通過重啟目標板開始進行測量。

如何測量電流消耗

HT32F125x 開發板的功耗測量可通過兩種方式測得：可利用電錶代替跳線 J25，並且由一個外部供電給開發板；通過使用 USB 線。

測量結果

測量結果如表 5 所示。

表 5 在 25°C 下低功耗模式測量結果舉例

低功耗模式	50ms	500ms	1s	3s	5s
深度休眠模式 1	77.2uA	69.3uA	69.2uA	68.6uA	68.5uA
深度休眠模式 2	23.4uA	14.3uA	13.8uA	13.4uA	13.4uA
暫停模式	363.2uA	46.2uA	26.4uA	10.7uA	8.8uA

結論

由於在 Cortex- M3 CPU 內核集成了用於低功耗應用（休眠模式和深度休眠模式）的高效核心級指令，同時結合 HT32F125x 的低功耗特性，使用戶可以根據應用的需求來優化系統功耗。

測量表明，必須考慮喚醒時間和功耗之間的最優化。