前言

此應用筆記說明本晶片使用時所需之規範，包括各周邊使用時之注意事項及硬體設計時所要求的元件規格。

目錄

1 概述 7
1.1 開發環境 7
1.2 庫函數選擇 7
1.3 暫存器防寫保護 7
1.3.1 FC 防寫保護 7
1.3.2 IWDT 防寫保護 8
1.3.3 GPIO 防寫保護 8
1.4 暫存器防讀保護 8
1.4.1 IWDT 防讀保護 8
1.5 寫 1 清零(C_W1)暫存器 8
2 系統控制 9
2.1 系統時鐘 RCU 9
2.1.1 HOSC 時鐘（HOSCCLK) 9
2.1.2 HRC 時鐘（HRCCLK) 9
2.1.3 PLL 時鐘（PLLCLK) 10
2.1.4 LRC 時鐘（LRCCLK) 10
2.2 時鐘安全系統（CSS) 10
2.3 特殊區塊 IP 11
2.4 MCO 時鐘（MCOCLK) 11
2.5 低功耗喚醒流程 12
2.6 配置字 12
3 外設 13
3.1 CALC 模塊 13
3.1.1 概要說明 13
3.1.2 常用功能描述 13
3.1.3 配置說明 13
3.1.4 應用限制 13
3.1.5 實際應用建議 13
3.2 CRC 模塊 14
3.2.1 概要說明 14
3.2.2 常用功能描述 14
3.2.3 配置說明 14
3.2.4 應用限制 14
3.3 DMA 模塊 14
3.3.1 概要說明 14
3.3.2 常用功能描述 14
3.3.3 配置說明 14
3.3.4 應用限制 15
3.3.5 實際應用建議 15
3.4 FLASH 模塊 15
3.4.1 概要說明 15
3.4.2 常用功能描述 15
3.4.3 配置說明 15
3.4.4 讀保護（RP） 15
3.4.5 寫保護（WP） 15
3.4.6 使用者代碼讀保護（UCRP） 15
3.4.7 等待週期設定 16
3.5 EXTI 模塊 16

3.5.1 概要說明 16
3.5.2 常用功能描述 16
3.5.3 配置說明 16
3.5.4 應用限制 17
3.6 GPIO 模塊 17
3.6.1 概要說明 17
3.6.2 常用功能描述 17
3.6.3 配置說明 17
3.6.4 應用限制 17
3.6.5 實際應用建議 18
3.7 UART 模塊 18
3.7.1 概要說明 18
3.7.2 常用功能描述 18
3.7.3 配置說明 18
3.7.4 應用限制 18
3.7.5 中斷處理建議 19
3.7.6 實際應用建議 19
3.8 I2C 模塊 19
3.8.1 概要說明 19
3.8.2 常用功能描述 19
3.8.3 配置說明 19
3.8.4 應用限制 19
3.8.5 中斷處理建議 20
3.9 SPI 模塊 20
3.9.1 概要說明 20
3.9.2 常用功能描述 20
3.9.3 配置說明 20
3.9.4 中斷處理 21
3.9.5 模式故障處理 21
3.9.6 結束條件 21
3.10 ADC 模塊 22
3.10.1 概要說明 22
3.10.2 常用功能描述 22
3.10.3 配置說明 22
3.10.4 應用限制 22
3.10.5 VDD（V_{REFP}）計算 22
3.10.6 真實電壓轉換 23
3.10.7 溫度量測 23
3.10.8 ADC 取樣之時間限制 24
3.10.9 ADC 校準 24
3.10.10 校準注意事項 25
3.11 OPAMP 模塊 26
3.11.1 概要說明 26
3.11.2 常用功能描述 26
3.11.3 配置說明 26
3.11.4 應用限制 26
3.11.5 低功耗行為 26
3.12 TIMER 模塊 26
3.12.1 概要說明 26
3.12.2 常用功能描述 26

3.12.3 配置說明 26
3.12.4 應用限制 27
3.12.5 剎車功能 27
3.12.6 通道設置 27
3.12.7 鎖定等級（防寫功能） 27
3.12.8 蜂鳴器控制 28
3.13 IWDT 模塊 28
3.13.1 概要說明 28
3.13.2 常用功能描述 28
3.13.3 配置說明 28
3.13.4 超時時間計算 29
3.14 WWDT 模塊 29
3.14.1 概要說明 29
3.14.2 常用功能描述 29
3.14.3 配置說明 29
3.14.4 超時時間計算 29
3.14.5 中斷與復位行為 30
4 系統電路與版圖注意事項 31
4.1 最小系統電路 31
4.1.1 LQFP48 封裝芯片最小系統電路 31
4.1.2 QFN40 封裝芯片最小系統電路 31
4.2 外圍電路 32
4.2.1 外部晶振 HOSC 32
4.2.2 PTEM / PTEM-V1LITE 與調適接口 32
4.2.3 UART-BOOT 燒錄接口 32
4.2.4 上下電斜率 32
5 文件版本紀錄 33

表格目錄

表 1. HOSC 時鐘模式列表 9
表 2. 計算參數說明 10
表 3. Clock Source 11
表 4. 低功耗喚醒注意事項列表 12
表 5. 低功耗模式喚醒後系統執行狀態列表 12
表 6. 配置字使用注意事項列表 12
表 7. 讀保護層級說明 15
表 8. FLASH 等待時鐘數(Wait Cycle) 16
表 9. LOCKLVL 層級說明 27
表 10. 文件修訂版本記錄 33

圖形目錄

圖 1. RP 設定狀態圖.....16
圖 2. LQFP48 封裝芯片最小系統電路.....31
圖 3. QFN40 封裝芯片最小系統電路.....31

1 概述

1.1 開發環境

推薦用戶使用 Keil5、IAR8.11 進行固件開發。由於 Keil4 不支援 PACK 機制，故不推薦用戶使用 Keil4。

1.2 庫函數選擇

PT2E 系列芯片提供 2 種類型庫函數 HAL 和 LL：

- HAL

  提供較為完善的封裝，提供更為人性化的 API，適合大部分使用者(後續視需求開發)。

- LL

  基本上只提供暫存器位域級別的讀、寫介面，適合對晶元底層較為熟悉的使用者。

1.3 暫存器防寫保護

- 為避免程式的異常導致運行錯誤，晶片防寫暫存器用於防止被保護的暫存器被錯誤操作。
- 支援模組：FC、IWDT、GPIO。
- 防寫流程：1. 解鎖（允許寫入） 2. 設定或寫入 3. 鎖回（禁止寫入）
- 函數庫中均提供相應之巨集定義，可進行解除保護和啟用保護。

1.3.1 FC 防寫保護

FC 中對於 FC_CMD、FC_CTL.OPRLD 等暫存器寫訪問操作是受保護的狀態。用戶無法直接對閃存進行編程與擦除，為了避免閃存資料被意外擦除或修改。若用戶需對閃存執行編程或擦除功能時，必須先進行解鎖流程，解鎖後才可對其進行修改，修改完成後建議重新上鎖避免閃存資料被意外擦除或是覆蓋。

項目	說明
受保護之暫存器	FC_CMD、FC_CTL.OPRLD
保護之目的	避免閃存資料被誤擦除或覆寫
解鎖步驟	1. 檢查 FC_STA.CMDULK = 0（鎖定狀態） 2. 寫入 FC_UL = 0x00112233（第一組密鑰） 3. 寫入 FC_UL = 0x55667788（第二組密鑰） 4. 確認 FC_STA.CMDULK = 1（解鎖成功）
修改完成後建議重新上鎖	寫入 FC_UL = 0x00000000

1.3.2 IWDT 防寫保護

IWDT 中部分暫存器寫訪問操作是受保護的狀態，必須透過以下流程進行解鎖及狀態檢查後方可進行修改。

項目	說明
受保護之暫存器	IWDT_BKPR、IWDT_BKRLR、IWDT_BKWINR
解鎖步驟	1. 對 IWDT_BKKR 寫入 0x00005555 2. 由於 IWDT 暫存器，存放在備用暫存器中，寫入後須等待 33 個 PCLK 時鐘更新才完成，可透過 IWDT_BKFR.BUSY 檢查狀態
重新上鎖	寫入任意非密鑰值

1.3.3 GPIO 防寫保護

透過 GPIOx_LCK 暫存器，經由 IO 鎖定流程，可鎖定暫存器寫訪問功能，此功能開啟後無法取消，只允許系統復位清除。

項目	說明
受保護之暫存器	GPIOx_MOD、GPIOx_OT、GPIOx_PUD、GPIOx_AFL、GPIOx_AFH
IO 鎖定流程	1. 連續兩次寫入相同 32 位值至 GPIOx_LCK（寫入的 GPIOx_LCK[31:16] = ~GPIOx_LCK[15:0] ） 2. 讀取 GPIOx_LCK.LCKK = 1 表示鎖定成功

1.4 暫存器防讀保護

1.4.1 IWDT 防讀保護

若需讀取 IWDT_BKPR.PR、IWDT_BKRLR.RL、IWDT_BKWINR.WIN、IWDT_BKSR.CNT 等暫存器，需依下列流程進行：

1. 讀取前須先解鎖: 向 IWDT_BKKR 寫入 0x00005555
2. 解鎖後根據預讀取之暫存器設定不同操作

讀取目標	操作步驟
IWDT_BKPR.PR	設定 IWDT_BKPR.R_PR=1 → 等待 IWDT_BKFR.BUSY=0 → 讀取 IWDT_BKPR.PR
IWDT_BKRLR.RL	設定 IWDT_BKRLR.R_RL=1 → 等待 IWDT_BKFR.BUSY=0 → 讀取 IWDT_BKRLR.RL
IWDT_BKWINR.WIN	設定 IWDT_BKWINR.R_WIN=1 → 等待 IWDT_BKFR.BUSY=0 → 讀取 IWDT_BKWINR.WIN
IWDT_BKSR.CNT	設定 IWDT_BKSR.R_CNT=1 → 等待 IWDT_BKFR.BUSY=0 → 讀取 IWDT_BKSR.CNT

1.5 寫 1 清零(C_W1)暫存器

- 當中斷、喚醒或復位等事件發生時，對應的事件狀態暫存器旗標會由硬體自動置為 1。其清除方式則需透過對應的事件清除暫存器寫入 1 ，由硬體自動完成旗標的清除。
- 若直接對事件狀態暫存器寫入 0，或以其他方式嘗試清除旗標，皆不會產生任何效果。

2 系統控制

2.1 系統時鐘 RCU

系統時鐘（SYSCLK）可由以下來源提供:

- HRC (High Speed Internal RC Oscillator) - 內部高速 16 MHz RC 振盪器
- HOSC (High Speed External Oscillator) - 外部高速時鐘振盪器
- PLL (Phase Locked Loop) - 鎖相環
- LRC (Low Speed Internal RC Oscillator) - 內部低速 32 kHz RC 振盪器

系統重置後，預設使用內部 16 MHz HRC 作為系統時鐘。可透過 RCU_CFG.SW 暫存器切換系統時鐘來源；切換前必須確保目標時鐘源已開啟並穩定。切換後可讀取 RCU_CFG.SWS 以確認是否切換完成。

Note:

1. 無法透過 RCU_CON 關閉當前正在使用的系統時鐘源，錯誤操作時，暫存器無反應。
2. 當 PLL 為系統時鐘時，作為其輸入的 HRC/HOSC 同樣不可關閉。
3. 每個時鐘源在未使用時皆可獨立關閉以降低功耗。
4. 周邊狀態時鐘依總線時鐘而定，即 AHB/APB，時鐘最高可配置為 80 MHz。
   - AHB：主要系統總線時鐘（HCLK）
   - APB：外設總線時鐘（PCLK）

2.1.1 HOSC 時鐘 (HOSCCLK)

表 1. HOSC 時鐘模式列表

模式	說明
外部時鐘源 (HOSC Bypass)	由外部提供有效的時鐘源，並透過 HOSCI 引腳輸入。設定暫存器 RCU_CON.HOSCBYP 與 RCU_CON.HOSCON 2 個位元啟用功能。
外部晶體振盪器 (HOSC Crystal)	透過 RCU_CON.HOSCON 控制開關。RCU_CON.HOSCRDY = 1 表示時鐘穩定。

注意事項：當開啟 RCU_CON.HOSCON 後，等待 RCU_CON.HOSCRDY 期間應加入 1 ms 超時機制。若逾時未達穩定狀態，須關閉後再重新啟用，直到標誌位 = 1。

2.1.2 HRC 時鐘 (HRCCLK)

- HRC 時鐘訊號是由內部高速 16 MHz RC 振盪器產生。
- 配置 RCU_CON.HRCON 位元控制 HRC 開啟與關閉。
- HRC 時鐘狀態可藉由 RCU_CON.HRCRDY 標誌位確認，當 HRC 時鐘穩定時，RCU_CON.HRCRDY = 1。

2.1.3 PLL 時鐘（PLLCLK）

PLL 為非整數鎖相迴路（Fractional-N PLL）。輸入時鐘頻率範圍為 3~16 MHz，可由 RCU_PLLCFG1.PLLSRC 選擇 HRC 或 HOSC。透過 RCU_PLLCFG1.PREDIV 進行預除頻（建議維持輸入約 4 MHz）。輸出範圍為 4~80 MHz，由壓控振盪器（VCO）提供。

輸出時鐘頻率計算：

$$f_{VCO} = f_{PLLIN} \times \left(FN + \frac{FK}{2^{19}}\right), 192MHz \le f_{VCO} \le 384MHz$$

$$f_{PLL} = \frac{f_{VCO}}{FM}, FM = 2, 4, 6, 8, 12, 16, 24, 32, 48$$

表 2. 計算參數說明

參數	說明
FN	整數倍數控制（RCU_PLLCFG2.FN)
FK	小數倍數控制（RCU_PLLCFG2.FK)
FM	除頻控制，可設為 2、4、6、8、12、16、24、32、48 (RCU_PLLCFG2.FM)

PLL 配置流程：

1. 關閉 PLL：RCU_CON.PLLON = 0
2. 確認 RCU_CON.PLLRDY = 0
3. 設定預除頻與倍率參數
4. 啟用 PLL：RCU_CON.PLLON = 1
5. 等待 RCU_CON.PLLRDY = 1，確認穩定

Note:

1. PLL 的配置（包括輸入時鐘源的選擇、預除頻和倍率設定）必須在 PLL 開啟之前完成，開啟 PLL 後便無法修改配置，必須關閉 PLL 才可修改配置。
2. RCU 沒有針對 PLL 的輸入時鐘源做硬體檢查，因此使用者在開啟 PLL 之前，需確保選擇的輸入時鐘源已經開啟且穩定。

2.1.4 LRC 時鐘（LRCCLK）

- LRC 為內部低速 RC 振盪器，約 32 kHz，支援在低功耗模式下維持 IWDT 運作。
- 配置 RCU_LCON.LRCON 位元控制 LRC 開啟與關閉。
- LRC 時鐘狀態可藉由 RCU_LCON.LRCRDY 標誌位確認，當 LRC 時鐘穩定時，RCU_LCON.LRCRDY = 1。

2.2 時鐘安全系統（CSS）

- HOSC 時鐘安全系統（Clock Security System，CSS）是一個系統安全機制。
- 當系統時鐘選用 HOSC 時，為了防止 HOSC 發生故障，而導致系統死機，避免應用陷入不安全的狀態。可以透過 RCU_CON.CSSON 開啟或關閉時鐘安全系統。

- 只有當 HOSC 時鐘開啟且訊號穩定才能開啟時鐘安全系統，並且在偵測到 HOSC 停止時關閉，並自動切換成 HRC。

2.3 特殊區塊 IP

以下這些特殊區塊只能支援特定時鐘源

表 3. Clock Source

Source / IP	SYS	AHB	APB	HRC	HOSC	PLL	LRC
UART 時鐘 (UARTKCLK)	•		•	•			•
ADC 時鐘 (ADCCLK)	•	•				•	
I2C 時鐘 (I2CCLK)	•		•	•			
IWDT 時鐘 (IWDTCLK)							•
System Tick 時鐘 (STCLK)		•					•

2.4 MCO 時鐘（MCOCLK)

- 微控制器時鐘輸出（MCO）功能可將系統時鐘輸出至外部引腳，供其他裝置使用或進行除頻測試。
- 透過配置 RCU_CFG.MSW 選擇輸出時鐘，並依需求可配置 RCU_CFG.MPRE 將微控制器時鐘輸出進行除頻（可以除頻 2、4、8、16、32、64 或 128 倍)。
- 支援的輸出來源：
  - LRC 時鐘
  - HRC 時鐘
  - HOSC 時鐘
  - PLL 時鐘
  - 系統時鐘（SYSCLK)
  - AHB 時鐘
  - APB 時鐘
  - ADC 核心時鐘
  - I2C 核心時鐘
  - UART 核心時鐘

2.5 低功耗喚醒流程

表 4. 低功耗喚醒注意事項列表

注意事項	說明
從低功耗模式喚醒後， 必須檢查/清除喚醒旗標。	1. 當經由引腳 WKUPx 喚醒後，必須清除喚醒旗標 SYSCFG_WKSR.WKCLR。 2. 需檢查喚醒旗標 SYSCFG_WKSR.FLAG 是否已清除。 3. 由於 UART 預設引腳設置於 WKUPx 引腳上，因此在尚未清除旗標前無法使用 UART 功能。
從低功耗模式 (STOP0/STOP1) 喚醒後，需檢查系統時鐘來源是否穩定。	1. 低功耗模式 (STOP0/STOP1)喚醒後，系統時鐘配置保持睡前狀態 2. 為了確保時鐘穩定，請務必確認時鐘來源的 RDY 旗標 3. 可由 RCU_CFG.SWS 取得當前的時鐘源，在判定相對應的穩定旗標 RDY。

表 5. 低功耗模式喚醒後系統執行狀態列表

低功耗模式	系統流程
SLEEP	原流程繼續往下執行
STOP0	原流程繼續往下執行
STOP1	原流程繼續往下執行
STANDBY	系統重啟，流程重頭開始執行
SHUTDOWN	系統重啟，流程重頭開始執行

2.6 配置字

表 6. 配置字使用注意事項列表

注意事項	說明
配置字修改與重置	當使用 PTEM/PTEM-V1LITE，修改芯片配置字之後，需要斷電後重新上電，才能正常調試。
BOR 配置字	BOR 電壓等級 (Level)，必須設定高於 MCU 工作電壓；電壓等級數值，請參考 PT2E01_Datasheet.pdf
IWDT 配置字	IWDT 配置字開啟後，軟件無法關閉，預設 1 秒發生 IWDT 復位，支持軟件修改復位時間。

3 外設

3.1 CALC 模塊

3.1.1 概要說明

CALC 模組提供整數除法與平方根運算功能，可在硬體層加速運算過程，降低 CPU 負擔。

3.1.2 常用功能描述

- 支援除法與平方根兩種計算模式。
- 支援有號與無號運算。
- 運算結果自動更新於結果暫存器中。

3.1.3 配置說明

運算觸發方式由 CALC_DIVCON.TRM 位元控制：

觸發條件	描述
TRM = 0	寫入被除數後觸發
TRM = 1	寫入除數後觸發

平方根運算在寫入被開方數後即自動開始。

3.1.4 應用限制

- 當 CALC_STAT.BUSY = 1 時，禁止軟體讀取商數及餘數，直至該標誌值為 0 才會返回商數及餘數。
- 若在 BUSY 狀態下重新觸發，先前結果將被覆蓋。
- 有號除法若發生溢位，結果不可靠（硬體不提供溢位標示）。
- 當除數為 0 時，CALC_STAT.DZ = 1，結果無效。
- 平方根運算會捨去小數部分。

3.1.5 實際應用建議

當被開方數非常小時，硬體計算誤差較大。

可先將輸入值左移 << 2n，計算後再將結果右移 >> n，以提高精度。

（注意：左移時需避免 Overflow。）

3.2 CRC 模塊

3.2.1 概要說明

CRC（Cyclic Redundancy Check）模組提供硬體加速的循環冗餘檢查功能，用於資料完整性驗證。

3.2.2 常用功能描述

- 支援 8/16/32-bit 資料寬度。
- 可設定多項式、初始值與輸入反轉。
- 運算結果自動保留於硬體暫存器。

3.2.3 配置說明

1. 啟用 RCU 內的 CRC 時鐘。
2. 重設 CRC 控制暫存器（CRC_CON.RESET = 1）。
3. 將待校驗資料寫入 CRC_DATA 暫存器，即自動開始運算，暫存器在 CRC 計算期間被鎖住。

3.2.4 應用限制

- 使用前需重設 CRC 模組，以避免殘留上一次的計算結果。
- 若對 32-bit 資料進行 8-bit CRC，可直接填入 32-bit 資料，或拆分為連續 8-bit 寫入。

3.3 DMA 模塊

3.3.1 概要說明

DMA（Direct Memory Access）模組可在不經 CPU 介入的情況下，直接在記憶體與外設間搬移資料，以提升傳輸效率。

3.3.2 常用功能描述

- 支援記憶體與外設之間的雙向傳輸。
- 可設定通道優先權與仲裁率。
- 支援多種資料寬度（8/16/32-bit）。

3.3.3 配置說明

1. 開啟對應外設與 DMA 控制器時鐘。
2. 配置 DMA 通道來源與目的位址、資料長度、傳輸方向。
3. 設定通道優先權與觸發來源。
4. 啟用 DMA 通道（DMA_CHEN）。

3.3.4 應用限制

- DMA 與 CPU 共用 AHB 總線，若同時存取同一地址，CPU 存取會被暫停數個總線週期。
- 資料來源與目的地寬度應一致，以避免資料錯位或浪費空間。
- DMA 控制暫存器寫入後，必須確認讀回值一致才視為設定成功。
- CPU 若需操作相同外設，應先確認 DMA 傳輸已完成。
- 非循環模式可透過 DMA_NDTn.NRDT 確認是否傳輸完成。

3.3.5 實際應用建議

多通道同時使用時，請依資料重要性調整優先權；可搭配仲裁設定以平衡效能與穩定性。

3.4 FLASH 模塊

3.4.1 概要說明

FLASH 模組負責儲存程式與使用者資料，支援多層級保護機制及等待週期調整。

3.4.2 常用功能描述

- 提供三種保護機制：讀保護（RP）、寫保護（WP）、使用者代碼讀保護（UCRP）。
- 支援重新上電後保護設定重載。

3.4.3 配置說明

- 所有保護設定均需在重新上電或從 STANDBY / SHUTDOWN 模式恢復後生效。
- 解鎖與保護設定需透過配置字重載完成。操作方式可參考 FC 防寫保護。

3.4.4 讀保護（RP）

表 7. 讀保護層級說明

等級	描述
Level 0	不保護（預設）
Level 1	禁止程式下載，但內部程式可自我修改。若降級至 Level 0，會觸發全擦除並需將燒入器重新上電後才可再進行燒入。
Level 2	禁止程式下載與調試，且不可再修改等級。

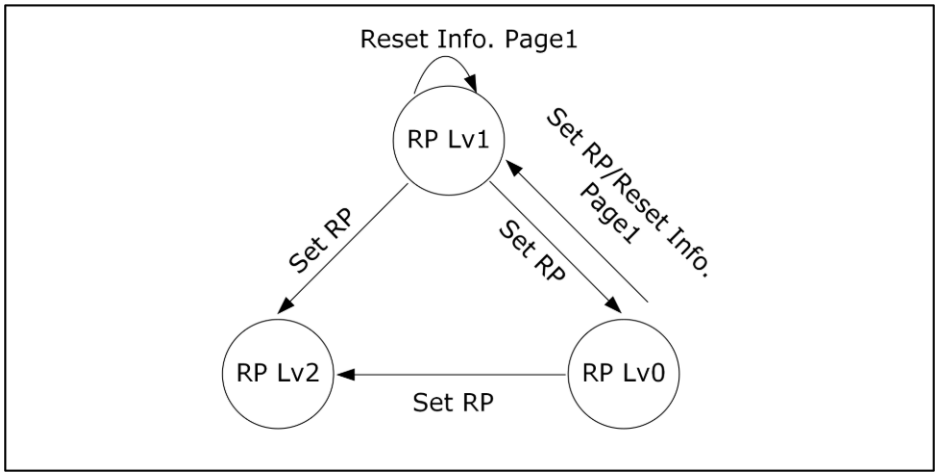
3.4.5 寫保護（WP）

- 可設定兩個區塊（ WP0、WP1）。
- 啟用後禁止擦除與寫入，需透過外部工具解除保護。

3.4.6 使用者代碼讀保護（UCRP）

- 可設定兩個區塊（UCRP0、UCRP1）。
- 保護後禁止讀取、修改與覆寫。

圖 1. RP 設定狀態圖



3.4.7 等待週期設定

系統頻率變化時，必須確保有足夠的 FLASH 等待周期 FC_CTL.WAIT，支持最多 3 個等待週期，如下表。

表 8. FLASH 等待時鐘數(Wait Cycle)

系統時鐘	FC_CTL.WAIT
> 72Mhz	3
72Mhz ≥ f_{SYSCLK} > 48Mhz	2
48Mhz ≥ f_{SYSCLK} > 24Mhz	1
24Mhz ≥ f_{SYSCLK}	0(無等待)

3.5 EXTI 模塊

3.5.1 概要說明

EXTI（External Interrupt/Event Controller）模組負責偵測外部事件，產生中斷或事件觸發，用於喚醒系統或驅動外設。

3.5.2 常用功能描述

- ◆ 支援 GPIO 腳位外部中斷。
- ◆ 可用於 ADC 觸發、低功耗喚醒等外部事件來源。
- ◆ 提供上升沿、下降沿或雙邊沿觸發模式。

3.5.3 配置說明

1. 配置對應 GPIO 腳位為輸入模式。
2. 設定 EXTI 腳位對應暫存器：
   - ■ ICFG1 控制 GPIO0 ~ GPIO7。
   - ■ ICFG2 控制 GPIO8 ~ GPIO15。

3. 設定觸發條件（上升沿、下降沿或雙邊沿）。
4. 啟用對應中斷或事件。

3.5.4 應用限制

- 若用於 ADC 外部觸發或低功耗喚醒，需確保對應 GPIO 已配置中斷模式。
- 使用 EXTI 觸發 CMP 中斷前，請確認已開啟並設定對應的 CMP 通道。
- 使用 EXTI 時必須先設定 EXTI 的 IO，再設定 EXTI_RTS 或 EXTI_FTS。

3.6 GPIO 模塊

3.6.1 概要說明

GPIO（General Purpose Input/Output）模組提供通用輸入與輸出控制，可配置為數位輸入、輸出或多功能復用模式。

3.6.2 常用功能描述

- 支援輸入、輸出、開漏、推挽等多種模式。
- 可作為外設功能的復用腳位。
- 提供上拉、下拉電阻控制。
- 支援低功耗狀態下的引腳保持與喚醒功能。

3.6.3 配置說明

1. 設定 GPIOx_MOD 指定腳位模式（輸入、輸出或復用）。
2. 若使用外設功能，先設定 GPIOx_AFx 選擇復用種類，再將 GPIOx_MOD 設為復用模式 (10b)。
3. 若為輸入模式，需設定 GPIOx_PUD 為上拉或下拉。

3.6.4 應用限制

- 未使用的 GPIO：建議設為類比模式（11b）並懸空，以避免漏電。
- 低功耗喚醒腳位：PB00 ~ PB07 支援喚醒功能，需設定 AF9 並開啟 SYSCFG_WKUP.WKEN，並選擇喚醒事件 SYSCFG_WKUP.WKEG 是上升緣或下降緣模式。
- 低功耗模式保持狀態：在 STANDBY/SHUTDOWN 模式下，PB00~PB07 會維持進入前的模式設定（輸入/輸出、PUD、OT、DS），但暫存器值不保留。
- 在進入 STANDBY/SHUTDOWN 模式前，須先完成喚醒引腳（WKUPx）配置，因進入後該引腳會被自動鎖定。系統喚醒後，需將 SYSCFG_WKSR.WKCLR 設為 1 以清除喚醒旗標，喚醒流程結束後引腳鎖定會自動解除。若 WKUPx 被設定為 UART、I2C、SPI 等複用功能，應在喚醒流程完成後再啟用通訊。
- 部分 I/O 腳位於出廠時已具備固定上拉或下拉設定，使用者在使用時須注意設置：

- SWD CLK (PC00) ： 預設下拉
- SWD DATA (PC01) ： 預設上拉
- NRST ： 預設上拉

3.6.5 實際應用建議

進入低功耗模式前，應確認輸出腳位不處於浮動狀態，以降低功耗與電氣干擾。

3.7 UART 模塊

3.7.1 概要說明

UART（Universal Asynchronous Receiver/Transmitter）模組用於異步序列通訊，支援可調波特率、自動波特率偵測與中斷控制。

3.7.2 常用功能描述

- 支援全雙工傳輸。
- 提供自動波特率偵測功能。
- 支援多種中斷事件（接收、傳送、錯誤）。

3.7.3 配置說明

1. 啟用對應 UART 時鐘。
2. 設定波特率、資料位元、停止位元、奇偶校驗。
3. 啟用中斷（若需）。
4. 開啟 UART 傳輸（UART_CON1.UEN = 1）。

3.7.4 應用限制

- 若發生自動波特率偵測超時（UART_RIF.ABTO），UART_MCON.ABREN 會自動清除。建議同時啟用 UART_MCON.ABRREPT 重複偵測功能。
- 若自動波特率成功但接收資料異常，需重新啟用 ABREN。
- 超時後無法收發資料的問題與修正方式：

在部分應用中，若發生自動波特率偵測超時（ABTO=1）後，UART 模組可能暫時無法再接收或傳送資料。此現象通常是由內部波特率產生邏輯未重新載入所造成。

修正方式（Workaround）

當偵測到超時情況時，可透過重新更新 UART_BRR 暫存器的內容來觸發硬體重新載入設定，使 UART 模組恢復正常通訊。軟體中採以下兩種方式進行：

1. Polling 模式
   - 程式主迴圈中輪詢 UART_RIF.ABTO 標誌。

- 當偵測到 ABTO=1 時，重新寫入一次 UART_BRR 暫存器（即使內容相同）。

2. Interrupt 模式
   - 啟用 UART_IER.ABTO 中斷。
   - 在中斷服務程式 (ISR) 中判斷 UART_IFM.ABTO。
   - 重新寫入 UART_BRR 暫存器以更新波特率設定。

   重新設定 UART_BRR 前必須確保 uart tx /rx enable 為關閉狀態，否則無法成功設定 UART_BRR

3.7.5 中斷處理建議

- 傳送中斷 (TXE) 發生時，應先清除中斷旗標再寫入新資料至 UART_TXDATA。
- 第一筆傳輸的空中斷與後續中斷間隔時間極短，若未先清除中斷可能造成資料錯送。

3.7.6 實際應用建議

傳送流程建議採：

清除中斷 → 寫入 UART_TXDATA → 等待 TXE 中斷清除 → 準備下一筆

可避免資料競爭。

3.8 I2C 模塊

3.8.1 概要說明

I2C（Inter-Integrated Circuit）模組提供主從雙向通訊功能，支援 7-bit 與 10-bit 位址模式，常用於感測器或 EEPROM 通訊。

3.8.2 常用功能描述

- 支援主機與從機模式。
- 支援自動停止與重新載入傳輸模式。
- 支援多重中斷事件（TX、RX、錯誤、重啟）。

3.8.3 配置說明

1. 啟用 RCU 中的 I2C 時鐘。
2. 設定 I2C_CON1 與 I2C_CON2 控制暫存器。
3. 配置位址模式（7-bit 或 10-bit）。
4. 設定資料長度 (NBYTES) 與是否啟用自動結束 (AUTOEND)。
5. 啟用 I2C 模組（I2C_CON1.PE = 1）。

3.8.4 應用限制

- 當 SDA 線被外部設備異常拉低，可能導致誤入從機模式。

  建議在 START 之後啟動軟體超時計時器，若超時則執行：

I2C_CON1.PE = 0 → 重新配置 → I2C_CON1.PE = 1

- 關閉重載模式時最大支援 65535 Bytes。
- 啟用重載模式後，傳輸結束時會自動拉低 SCL，可重新設定下一段資料長度。
- 接收模式下最大傳輸長度為 255 Bytes，超過需使用重載機制。
- I2C_CON2 必須按字（32 位）訪問。
- 在 Master 模式下，若設定為非 AUTOEND 且需要連續傳送資料，在重新設定 NBYTES 及 START bit 後，必須等待 START bit 由高轉低，再開始進行資料傳送，以避免資料遺漏。

3.8.5 中斷處理建議

- 發送空中斷時（TXE），I2C_TXDATA 寫入後會立即被搬移至 TX 移位暫存器。
- 中斷流程建議：

  先清除中斷 → 再填寫 I2C_TXDATA

  可避免中斷密集觸發導致的競態。

- 位址設定

模式	填入方式	範例
7-bit	位址左移一位	0x50 << 1
10-bit	直接填入	0x50

3.9 SPI 模塊

3.9.1 概要說明

SPI（Serial Peripheral Interface）模組提供全雙工高速同步序列通訊，支援主從模式與多種 FIFO 緩衝中斷。

3.9.2 常用功能描述

- 支援主機與從機模式。
- 提供發送與接收 FIFO。
- 支援模式故障檢測。
- 支援雙邊沿採樣。

3.9.3 配置說明

1. 啟用 SPI 時鐘並配置 SPI_CON1。
2. 設定時鐘極性、相位與資料寬度。
3. 配置主/從模式與 NSS 管理方式。
4. 啟用中斷（若使用 FIFO）。

SPI 作為從機時，時鐘信號由主機提供，所以建議先使能從機，然後主機再發送時鐘，否則數據傳輸可能不正常。

3.9.4 中斷處理

- 發送 FIFO 中斷
  - 中斷事件：
    - FIFO 空中斷（TXE）
    - FIFO 低於閥值中斷（TXTH）
  - 中斷流程： 優先清除中斷旗標 → 檢查 FIFO 狀態 → 填寫 SPI_DATA
- 接收 FIFO 中斷
  - 事件：
    - FIFO 非空（RXNE）
    - FIFO 滿（RXF）
    - FIFO 超過閥值（RXTH）
  - 建議優先讀取 SPI_DATA，避免 FIFO 滿溢導致資料遺失。

3.9.5 模式故障處理

若在主機模式下發生以下任一情況：

- NSS 腳位被外部拉低
- SPI_CON1.SSOUT = 0

則會觸發模式故障（SPI_IER.MODF = 1），同時：

- SPI_CON1.SPIEN = 0（關閉輸出）
- SPI_CON1.MSTREN = 0（強制進入從機模式）

恢復方式：

1. 將 NSS 拉高。
2. 清除 SPI_RIF.MODF。
3. 重新設定 SPIEN 與 MSTREN。

3.9.6 結束條件

在關閉 SPI 前，應確保：

模式	條件
全雙工	RXFLV = 0 且 SPI_STAT.BUSY = 0
單工 / 半雙工（發送）	TXE = 1 且 SPI_STAT.BUSY = 0
單工 / 半雙工（接收）	等待 RXNE = 1 後再關閉

3.10 ADC 模塊

3.10.1 概要說明

ADC（Analog-to-Digital Converter）模組將類比訊號轉換為數位值，用於感測器輸入、電壓監測與溫度量測等應用。

3.10.2 常用功能描述

- 支援多通道、單次或連續轉換模式。
- 提供自動注入通道、掃描與中斷功能。
- 內建溫度感測器與參考電壓通道。

3.10.3 配置說明

1. 在 ADCEN = 0 狀態下設定 RCU 與低功耗開關。
2. 啟用 ADC 時鐘並配置取樣時間、觸發方式。
3. 在 ADCEN = 1 且 ADCDIS = 0 時，設定轉換暫存器與觸發。
4. 啟用中斷或 DMA（視應用需求）。

3.10.4 應用限制

- 僅在 ADC 關閉狀態（ADCEN = 0）可更改時鐘與低功耗配置。
- 設定自動注入模式時，不可啟用非連續模式（DISCEN=0、JDISCEN=0）。
- 硬體並沒有強制禁止寫入 ADC 暫存器，然而錯誤的暫存器寫入可能導致 ADC 進入未知狀態，需重新關閉 ADC 再開啟。

3.10.5 VDD（V_{REFP}）計算

由於供應電壓可能變動，建議利用內建 V_{REFINT} 通道計算實際 VDD（即 V_{REFP}）：

$$V_{REFP} = FULL_SCALE \times \frac{ADC_{VREFINT_CAL}}{ADC_{VREFINT}}$$

其中:

參數	說明
$ADC_{VREFINT_CAL}$	晶片在生產驗證過程中會在 VDDA = 5V 的狀態下，利用校準後的 ADC 取樣 V_{REFINT} 電壓並紀錄在配置字中，可利用 ll_syscfg_get_adc_vref(SYSCFG) 讀取電壓值。
$ADC_{VREFINT}$	當下 ADC 取樣 V_{REFINT} 所得 ADC 數值，可藉由 ADC_IN[18] 內部通道量測此電壓。
FULL_SCALE	ADC 輸出最大可表示的數值，12 位元分辨率，FULL_SCALE 為 4095。

範例程式

可利用以下函式給入 V_{REFINT} 作為引數計算 VDD（V_{REFP}）。

```
float ll_adc_vdd_calculation(uint16_t adc_vrefint)
{ return (float)ll_syscfg_get_adc_vref(SYSCFG)/adc_vrefint*4095;
 }
```

Note:

1. 需先執行下列程式碼打開 V_{REFINT} 開關才可讀取到正確電壓值。

   ll_adc_enable_vrefinten(ADC12_COMM)

2. ADC Sample Time 建議設置較長的時間以取得較為穩定的電壓數值。

3.10.6 真實電壓轉換

當取得 VDD（V_{REFP}） 之後可根據以下公式對數值換算以取得精確電壓值。

$$V_{AIN} = ADC_{AIN}\frac{V_{REFP}}{FULL_SCALE}$$

3.10.7 溫度量測

晶片內建溫度感測器可透過 V_{TSENSE} 經過進一步計算後取得晶片溫度。

$$V_{TSENSE} = ADC_{TSENSE}\frac{V_{REFP}}{FULL_SCALE}$$

當取得 V_{TSENSE} 後代入以下公式即可推算出實際溫度。

$$Temparature(°C) = \frac{V_{TSENSE} - V_{30}}{SLOPE} + 30$$

$$SLOPE = \frac{V_{85} - V_{30}}{55}$$

其中:

參數	說明
V_{AIN}	輸入類比電壓
ADC_{AIN}	ADC 對輸入通道進行取樣後輸出的數位值
ADC_{TSENSE}	ADC 量取溫度感測電壓所得數值，可藉由 ADC_IN[17] 內部通道量測電壓。
V_{30}	晶片在生產驗證過程中會在控溫 30°C 的狀態下，利用校準後的 ADC 取樣 V_{TSENSE} 電壓並紀錄在配置字中，可利用 ll_syscfg_get_adc_temp_low(SYSCFG) 讀取電壓值。
V_{85}	晶片在生產驗證過程中會在控溫 85°C 的狀態下，利用校準後的 ADC 取樣 V_{TSENSE} 電壓並紀錄在配置字中，可利用 ll_syscfg_get_adc_temp_high(SYSCFG) 讀取電壓值。

Note:

1. 需先執行下列程式碼打開 V_{TSENSE} 開關才可讀取到正確電壓值。

   ll_adc_enable_vtsen (ADC12_COMM)

2. ADC Sample Time 建議設置較長的時間以取得較為穩定的電壓數值。
3. 實際使用範例可參考 ADC_TemperatureSensor 範例程式

3.10.8 ADC 取樣之時間限制

ADC 內部電壓取樣（V_{TSENSE}, V_{REFINT}, V_{RES}）需要一定的取樣時間，以下為各內部電壓至少需等待的取樣時間：

- V_{REFINT}：3 μs
- V_{RES}：0.8 μs
- V_{TSENSE}：3 μs

時間計算方式：

$$t_{sample} = \frac{3.5 + T_{SMP}}{f_{ADC}}$$

其中:

參數	名稱	說明
t_{sample}	總採樣時間	單次 ADC 取樣所需的總時間（單位：秒）。包含固定的前置採樣時間與由 SMP 設定決定的可變採樣時間。
3.5	固定取樣週期	ADC 內部為建立採樣電容連接所需的固定週期，單位為 ADC 時鐘週期。
T_{SMP}	SMP 設定採樣週期	由暫存器 SMPx[7:0] 設定的採樣時間對應週期數。

3.10.9 ADC 校準

當系統重新上電後，ADC 的量測精度可能因 V_{REFP} 電壓變動而受到影響。為了提升量測準確度，可透過下列線性補償公式計算補償參數 ：

$$y = Ax + B$$

其中，A 為增益 (Gain)，B 為偏移量 (Offset)。

為了推導出 A 與 B，需要以兩組高、低電壓值{x1,x2}、{y1,y2}代入上述公式運算，求得 Gain(A) 與 Offset(B)。後續使用時將 ADC 轉換結果(y)減去 Offset 之後除以 Gain 即可回推準確值(x)。

◆ 由於硬體支援此補償機制，所以在算出 Gain 與 Offset 可填入相關暫存器中並開啟硬體補償模式，在使用 ADC 時即可取得補償後之轉換值。

◆ SDK 所提供的驅動程式(ll_adc.c)內的 adc_calibration() 函數有進行此校正補償運算。

在 adc_calibration() 程序中，

利用晶片出廠時量測 $\frac{1}{32} \times V_{RES1}$與 $\frac{31}{32} \times V_{RES1}$ 兩點（做為 {x1, x2}），

與目前系統狀態下量測的 $\frac{1}{32} \times V_{RES2}$與 $\frac{31}{32} \times V_{RES2}$ 數值(做為 {y1, y2})，

代入上述公式運算，求得 Gain(A) 與 Offset(B)。

算出 Gain(A) 與 Offset(B) 後，程序會將數值填入硬體相關暫存器並開啟硬體補償。

參數	說明
$\frac{1}{32} \times V_{RES1}$	晶片出廠時量測量測 5V 供電下 $\frac{1}{32} \times V_{DDA}$ 的電壓值，並且記錄於配置字中。
$\frac{31}{32} \times V_{RES1}$	晶片出廠時量測量測 5V 供電下 $\frac{31}{32} \times V_{DDA}$ 的電壓值，並且記錄於配置字中。

參數	說明
$\frac{1}{32} \times V_{RES2}$	目前系統狀態下量測的 $\frac{1}{32} \times V_{DDA}$ 的電壓值。
$\frac{31}{32} \times V_{RES2}$	目前系統狀態下量測的 $\frac{31}{32} \times V_{DDA}$ 的電壓值。

3.10.10 校準注意事項

- 由於硬體會對於 Offset 數值飽和進行判斷，當量測電壓值接近 V_{REFP} 時可能會導致誤判，因此補償方式改為硬體與軟體配合進行。

  在使用上仍然藉由 adc_calibration() 程序中求得 Gain 與 Offset，只填入 Gain 值藉由硬體協助運算，而 Offset 補償則由取得 ADC 轉換數值後藉由 SDK 所提供的 Driver(ll_adc.c)內 ll_adc_read() 進行。

- 因為此 Offset 補償數值極小，為了效率考量，SDK 範例中預設只靠硬體進行 Gain 補償，若使用者需要精確的量測值則使用 ll_adc_read() 額外補償 Offset。

  軟體 Offset 運算後的數值可能產生 OverFlow 與 UnderFlow 情況，使用時需特別注意。

- 補償時在下列應用中須特別注意:
  - OverSampling: 根據 OverSampling(ADC_CFG2.OVSRATIO) 的次數與 OverSamplingShift(ADC_CFG2.OVSSHIFT) 的配置，進行 offset 數值倍率的計算。
    - ex1:
      OverSampling = 8x
      OverSamplingShift = No Shifts
      則需要加上 8 倍 Offset
    - ex2:
      OverSampling = 8x
      OverSamplingShift = Shift 2-bits
      則需要加上 2 倍 Offset
  - Compensation：各通道可獨立設定 Offset Compensation，若再搭配 ADC Calibration 的 Offset 設定一同使用，則轉換結果已由硬體進行 Offset Compensation，只需另外進行 ADC Calibration Offset
    - 正常情況下只有在開啟 Compensation 時才會有轉換值為負值的情況。使用者在單獨使用 ADC Calibration 的 Offset 時若發現轉換值與其相減後為負值可以自行根據應用判斷是否為 0
  - ADC Watch Dog：此看門狗的門檻數值是直接對 ADC 轉換值進行判定，所以在靠近門檻邊界時可能造成誤判，可藉由將門檻值加上 Offset 避免誤判

3.11 OPAMP 模塊

3.11.1 概要說明

OPAMP（Operational Amplifier）模組提供可程式化的運算放大器，適用於訊號緩衝、放大與濾波等應用。

3.11.2 常用功能描述

- 支援單端、差動與緩衝模式。
- 可與 ADC 模組直接連結以擷取放大後訊號。
- 支援低功耗運作與暫存器保持。

3.11.3 配置說明

- 啟用對應的 OPAMP 時鐘。
- 設定輸入端與增益模式（外部分壓或內部回授）。
- 設定輸出端連結至目標模組（如 ADC）。

3.11.4 應用限制

- 輸入阻抗應盡量小以避免失真。
- 使用獨立模式時需配置外部分壓電阻。

3.11.5 低功耗行為

模式	暫存器行為
SLEEP	無影響
STOP0 / STOP1	無影響，保持原設定
STANDBY / SHUTDOWN	OPAMP 暫存器斷電，退出後需重新配置

3.12 TIMER 模塊

3.12.1 概要說明

TIMER 模組提供計數、PWM 產生、輸入捕獲與輸出比較等功能，是 MCU 時序控制的核心元件。

3.12.2 常用功能描述

- 支援基本計數、比較、捕獲、PWM、編碼器模式。
- 支援主從觸發、死區時間、互補輸出。
- 具備剎車（Brake）與輸出安全保護功能。

3.12.3 配置說明

1. 設定計數方向、預分頻器與自動重載值。
2. 選擇觸發來源與更新事件（UERSEL）。

3. 設定輸出比較模式（含極性與死區時間）。
4. 若啟用剎車功能，配置 BKIN 腳位與保護參數。

3.12.4 應用限制

- 當 UERSEL = 1 時，僅上溢或下溢才產生更新事件。

  例: UERSEL=1 時，Slave Mode Reset 僅初始化計數器，不產生更新事件
- CHn 與 CHnN 互補輸出不得同時為有效電平。
- 若死區時間大於脈衝寬度，對應輸出脈衝不會產生。

3.12.5 剎車功能

- 剎車輸入為電平有效，當保持有效時 GOEN 不能設為 1，此外，BRK 中斷狀態也無法被清除。
- 當準位配置錯誤時（例如配置成高電平有效），為了安全目的無法支持雙向剎車輸入模式。
- 雙向剎車模式需設定 E16C6T_BDCFG.BKDID = 1 並設 BKIN 為開漏。
- 剎車事件發生後，GOEN 會自動清除為 0。
  - OFF 狀態設定:
  - OFFSSI = 0 輸出高阻態，不受控制。
  - OFFSSI = 1
    - 單通道輸出時：煞車事件後，通道輸出腳會依 OISSn 設定維持於預設的空閒狀態。
    - 雙通道輸出（主輸出與互補輸出）時：通道輸出會先輸出無效狀態（取決於極性設定），經過死區時間後回到空閒電平。系統會確保主、互補兩腳不會同時為有效，以防止功率元件誤導通。

3.12.6 通道設置

當通道配置為輸入模式時:

CCnNPOL	CCnPOL	模式	說明
0	0	非反相 / 上升沿	在門控或編碼器模式下，輸入訊號不反相；在復位、外部時鐘或觸發模式下，偵測上升沿有效。
0	1	反相 / 下降沿	在門控或編碼器模式下，輸入訊號先反相；在復位、外部時鐘或觸發模式下，偵測下降沿有效。
1	0	保留	不可使用。
1	1	非反相 / 上升沿 + 下降沿	在門控、編碼器、復位、外部時鐘或觸發模式下與 00 設定相同，但在輸入捕獲模式下，偵測雙邊沿有效。

3.12.7 鎖定等級（防寫功能）

表 9. LOCKLVL 層級說明

LOCKLVL	描述
00	無保護
01	鎖定部分暫存器（例：E16C6T_CON2 中之 OISS、DTPEN、BRKEN...等）

10	鎖定級別 1 + OFFSSR/ OFFSSI/ CCnPOL/ CCnNPOL
11	鎖定級別 2 + CHnMOD/ CHnPEN（通道配置為輸出模式）

3.12.8 蜂鳴器控制

首先藉由軟體控制 2 組 Timer 產生具有高頻載波之 PWM 調變訊號，接著利用 IR Out 將兩個訊號進行邏輯運算 AND 進而驅動蜂鳴器，並且其中一個 Timer (A32C2T2) 由另外一個 Timer (A32C2T1) 的更新事件觸發 Counter 計數，藉此達成訊號同步，提升聲音輸出品質。

可由範例程式 **TIMER_sound_generator_two_pwm** 所提供函數控制蜂鳴器

- **BuzzerConfig(uint16_t frequency, uint8_t volume, uint8_t carrier_frequency)**

  使用時依序給予音調頻率(Hz)、音量比例、載波頻率(kHz)作為引數，函數內會自動進行運算並設定 Timer

  Note: 若調整了音調頻率或是載波頻率，需要同步對音量比例進行調整

- **BuzzerStop(void)**

  關閉蜂鳴器輸出

- **BuzzerVoiceFadeAway(void)**

  控制蜂鳴器音量平滑縮減至無聲

3.13 IWDT 模塊

3.13.1 概要說明

IWDT（Independent Watchdog Timer）模組獨立於系統主時鐘，用於偵測軟體失控並觸發系統重啟。

3.13.2 常用功能描述

- 採用 LRC 作為時鐘來源。
- 具可調預分頻與重載值設定。
- 超時未餵狗將導致 MCU 復位。
- 提供「窗口時間」機制，監測餵狗時機。
- IWDT 啟用後便無法停止。

3.13.3 配置說明

1. 對 IWDT_BKKR 寫入 0x0000CCCC 啟用 IWDT
2. 配置寫入解鎖，參考 IWDT 防寫保護
3. 設定預分頻值 (PR) 與重載值 (RL[11:0])。
4. 配置 IWDT 重載計數器數值。
5. 根據啟用窗口與否進行不同設定:

- 不啟用窗口設定：將 IWDT_BKRLR 的值刷新到看門狗計時器 (IWDT_BKKR=0x0000AAAA)。
- 啟用窗口設定：配置窗口暫存器 IWDT_BKWINR。此時會將 IWDT_BKRLR 的值刷新到看門狗計時器。

每次解鎖及寫入設定都需要等待 IWDT_BKFR.BUSY 數值更新為 0

3.13.4 超時時間計算

超時計算公式如下：

$$t_{IWDT} = t_{LRC} \times PR \times RL$$

其中：

參數	說明
t_{IWDT}	IWDT 超時時間
t_{LRC}	LRC 時鐘週期，固定為 $\frac{1}{32768}$≈30.5 μs
PR	預分頻值，由分頻器設定決定
RL	重載計數值（0x000 ~ 0xFFF），預設為 0x0FFF

Note:
1. 詳細設定可參考 IWDT 範例程式
2. 當系統因 Watchdog 復位循環重啟，可使用 NRST 引腳拉低一段時間進行系統復位。

3.14 WWDT 模塊

3.14.1 概要說明

WWDT（Window Watchdog Timer）模組監控程式運行時間窗口，防止過早或過晚餵狗，確保系統穩定運行。

3.14.2 常用功能描述

- 提供「窗口時間」機制，監測餵狗時機。
- 餵狗過早或過晚皆會觸發復位。
- 支援中斷模式，用於資料保存或安全處理。

3.14.3 配置說明

1. 設定分頻器 (WDGTB) 與窗口值 (WWDT_CFG.W)。
2. 啟用 WWDT 並寫入初始計數值。
3. 在允許窗口內餵狗，若超過則觸發復位。

3.14.4 超時時間計算

超時計算公式如下：

$$t_{WWDT} = t_{PCLK} \times 4096 \times 2^{WDGTB} \times (t[5:0] + 1)\ (ms)$$

其中：

參數	說明
t_{WWDT}	WWDT 超時時間
t_{PCLK}	APB1 以 ms 為單位的時鐘週期
t[5:0]	計數值

3.14.5 中斷與復位行為

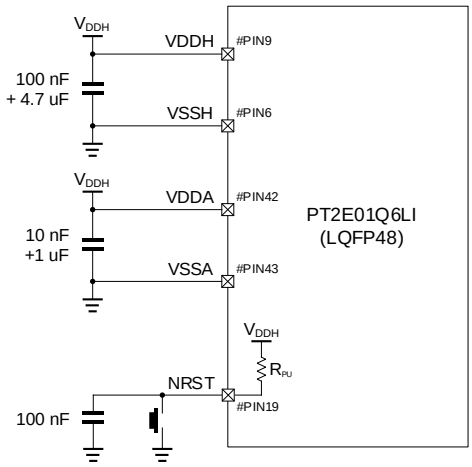
- 計數從 0x7F 遞減至 0x3F 時觸發復位。
- 若啟用中斷，當計數至 0x40 會產生中斷事件，可於中斷中保存資料或執行安全操作。

4 系統電路與版圖注意事項

4.1 最小系統電路

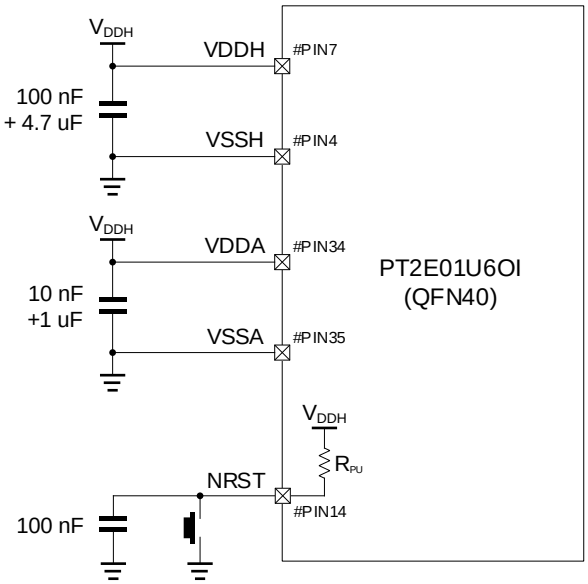
4.1.1 LQFP48 封裝芯片最小系統電路

圖 2. LQFP48 封裝芯片最小系統電路



4.1.2 QFN40 封裝芯片最小系統電路

圖 3. QFN40 封裝芯片最小系統電路



Note:

1. 電源設計：芯片電源配置必須依照上圖建議，以確保在各種應用場景下皆能穩定運行。
2. 去耦電容配置：每組電源引腳需連接陶瓷耦合電容（如圖所示），並應儘可能靠近對應管腳，以降低電源雜訊、提高芯片穩定性。
3. 復位電路設計：NRST 引腳採用 RC 復位，內部支援上拉電阻（上拉值請參考數據手冊電氣特性章節），電容建議使用 100nF。

4.2 外圍電路

4.2.1 外部晶振 HOSC

注意事項	說明
外部陶瓷電容	電容建議數值，請參閱 PT2E01 數據手冊，電器特性章節。
版圖參考原則	外部晶振應儘量靠近 IC，提高穩定度。

4.2.2 PTEM / PTEM-V1LITE 與調適接口

注意事項	說明
電路建議	• PC00 與 PC01 為 PTEM/PTEM-V1LITE 調適接口。 ○ PC00（SWCLK）：預設為輸入下拉模式。 ○ PC01（SWDIO）：預設為輸入上拉模式。 • 若應用程式將此兩腳配置為一般 IO 功能，需預留 NRST 引腳。 當 NRST = 0 時，系統可自動恢復 SWCLK 與 SWDIO 功能，用於燒錄或除錯。
Socket 燒錄時的電源配置	使用 Socket 進行離線燒錄時，請確保芯片電源依照 圖 2. LQFP48 封裝芯片最小系統電路 的電源配置建議，否則可能造成燒錄異常或啟動失敗。

4.2.3 UART-BOOT 燒錄接口

注意事項	說明
開機程序預設選擇	系統開機流程中，提供 BootRom 與 Main Flash 兩種選擇，PT2E01 預設為 Main Flash。此配置可透過修改 配置字 更換預設值。
UART-BOOT 接口選用	當配置字 BOOT BYPASS 選項為 BootRom 時： • 系統開機從 BootRom 開始執行。 • PB00 與 PB01 引腳預設為 UART1 通訊接口： ○ PB01 → UART1_TX（輸出） ○ PB00 → UART1_RX（輸入） • 可透過 UART 通訊接口搭配 UART-BOOT 燒錄工具進行程式下載或更新。
	當配置字 BOOT BYPASS 選項為 Main Flash 時： • 系統開機將直接跳轉至 Main Flash 區域執行。 • PB00 與 PB01 引腳預設為 GPIO。 • 不具備 UART-BOOT 燒錄功能。

4.2.4 上下電斜率

上電斜率需小於於 1.2s/V，下電斜率需大於 140μs/V，才能正常啟動

5 文件版本紀錄

表 10. 文件修訂版本記錄

日期	版本	修訂內容
2025-12-03	PRE1.0	初版
2026-07-08	PRE2.0	1. I2C 章節新增 Master 模式下連續傳送注意事項 2. 4.1.2 封裝信息更新为 QFN40。