Лабораторна робота № 5

Вивчення системи переривань мікроконтролерів PSoC 6 фірми Cypress

Мета роботи: Ознайомитися з архітектурою переривань мікроконтролерів PSoC 6, джерелами і типами переривань, налаштуванням переривань, алгоритмом опрацювання переривань.

Теоретичні відомості

Конфігурація переривань.

Розглянемо кроки, необхідні для налаштування переривань на мікроконтролері PSoC 6 без детального опису програмного забезпечення, що використовується для їх виконання. Ці кроки є загальними для ядер CM0+ і CM4, якщо не вказано інше, і повинні виконуватися для кожного ядра окремо.

1. Після скидання пристрою всі переривання заборонені (відключені), і пріоритети переривань встановлені в нуль.
2. Налаштувати рівень пріоритету необхідного IRQ в NVIC.
3. Налаштувати шлях переривання. Вибрати джерело переривання, яке підключено до потрібного IRQ ЦП. Для CM0+ вибрати відповідне периферійне переривання для підключення до ЦП. Для CM4 це неможливо налаштувати. Джерело переривання *n* завжди підключено до IRQn.
4. Налаштувати джерело переривання (периферійне) та ввімкнути (дозволити) це переривання.
5. Налаштувати векторну таблицю з адресою ISR (вектора). Векторна таблиця зберігає вхідні адреси кожного обробника винятків.
5. Необов’язково: очистити очікувані стани переривання в NVIC. Якщо дозволене попередньо заборонене переривання, то потрібно очистити стан очікування NVIC перед тим, як включити переривання. Це запобігає будь-якому помилковому запуску, викликаному попередніми перериваннями, які створили стан очікування.
7. Дозволити переривання в NVIC.
8. Дозволити глобальні переривання. Конфігурація переривання завершена.

Дозволене переривання спрацьовує, коли апаратний сигнал від джерела переривання активний і немає виконуваного переривання з більш високим пріоритетом. Коли це відбувається, виконання ЦП переходить до місця в його таблиці векторів, що відповідає спрацьованому перериванню. Це місце містить адресу ISR, пов'язану з цим перериванням.

ISR виконує завдання, необхідні для обробки переривання. Як правило, перше, що робить ISR, - це очистка джерела переривання, щоб уникнути повторного входу в ISR. Коли ISR припиняється, ЦП повертається до адреси, яку він виконував, до його переривання.

Розглянемо програмні засоби, доступні для виконання описаних вище кроків.

Конфігурація переривань.

Використання PDL (Peripheral Driver Library) – це набір засобів розробки програмного забезпечення (SDK - software development kit), який дозволяє розробляти мікропрограми (firmware) для пристроїв PSoC 6 MCU. Виклики функції PDL API використовуються для налаштування, ініціалізації, включення і використання периферійного драйвера. Одним з таких драйверів є системні переривання (SysInt). Системні переривання представляють структури та функції для налаштування і включення функції переривання. PDL також підтримують бібліотеки CMSIS-Core, які включають функції NVIC, використовувані для налаштування переривань.

Примітка: CY8C62x8/A підтримується лише в PDL v3.1.x та новіших версіях, що містять драйвер SysInt версії 1.20. Для розробки всіх пристроїв PSoC 6 рекомендується використовувати SysInt v1.20 або новішої версії.

В цих кроках використовуються API- інтерфейси PDL та NVIC для налаштування переривання для запуску сигналу від периферійного пристрою.

1. Налаштувати периферію для генерування переривання. Наприклад, для GPIO налаштувати режим драйвера (з резистивним підтягуванням вверх або вниз), перервати генерацію сигналу по зростаючому або спадному фронтах та зняти маску переривання.
2. Налаштувати переривання за допомогою структури, наданої API SysInt. Структура визначена у файлі драйверів PDL SysInt cy_sysint.h:

Ініціалізація структури конфігурації для одного каналу переривання.

```
* Initialization configuration structure for a single interrupt channel */
typedef struct {
    IRQn_Type       intrSrc;     /**< Interrupt source */
#if (CY_CPU_CORTEX_M0P)
    cy_en_intr_t    cm0pSrc;     /**< (CM0+ only) Maps cm0pSrc device interrupts
to intrSrc */
#endif
    uint32_t        intrPriority;   /**< Interrupt priority number (Refer to
__NVIC_PRIO_BITS) */
} cy_stc_sysint_t;
```

Ця структура використовується для конфігурації наступного:

а) Джерела переривань (intrSrc):

- це виділені номера переривань, визначені в заголовочному файлі пристрою (наприклад: cy8c6247bzi_d44.h).
- цей вибір залежить від того, якому ЦП хочемо назначити переривання.
- для ядра CM4 це число представляє як номер джерела переривання, так і номер IRQ ЦП. Виберіть номер переривання периферійного переривання, яке потрібно направити до ЦП. Наприклад, для маршрутизації переривання 0-го порту GPIO потрібно призначити значення ioss_interrupts_gpio_0_IRQn (= 0).
- для ядра CM0+ це число представляє один з 32 мультиплексорів, доступних для маршрутизації переривання на CM0+. Оскільки кожен мультиплексор підключений до виділеної лінії CM0+IRQ, використовуйте це для вибору

цільового номера CM0+IRQ. Наприклад, щоб використовувати мультиплексор №4 (CM0 + IRQ #4), потрібно вибрати ″NvicMux4_IRQn″ (= 4).

б) номер переривання ядра CM0+ (cm0pSrc):

- цей параметр використовується тільки для CM0+.
- він представляє номер джерела переривання, який повинен бути перенаправлений до логіки переривань мультиплексора /генератора CM0+, вибрану за допомогою параметра intrSrc. Виберіть номер переривання периферійного переривання, яке потрібно направити до центрального процесора; наприклад, для маршрутизації переривання порту 0 GPIO, призначте значення "ioss_interrupts_gpio_0_IRQn" (= 0).

в) пріоритет переривання (intrPriority):

- встановити пріоритет переривання. Для ядра CM4 підтримуються пріоритети від 0 до 7. Для ядра CM0+ підтримувані пріоритети -від 0 до 3.

Примітки:

- В ядрі CM0+ деякі IRQ зарезервовані для використання програмним забезпеченням і не доступні для користувача.
- В ядрі CM0+ пріоритет переривання 0 зарезервований для системних викликів.

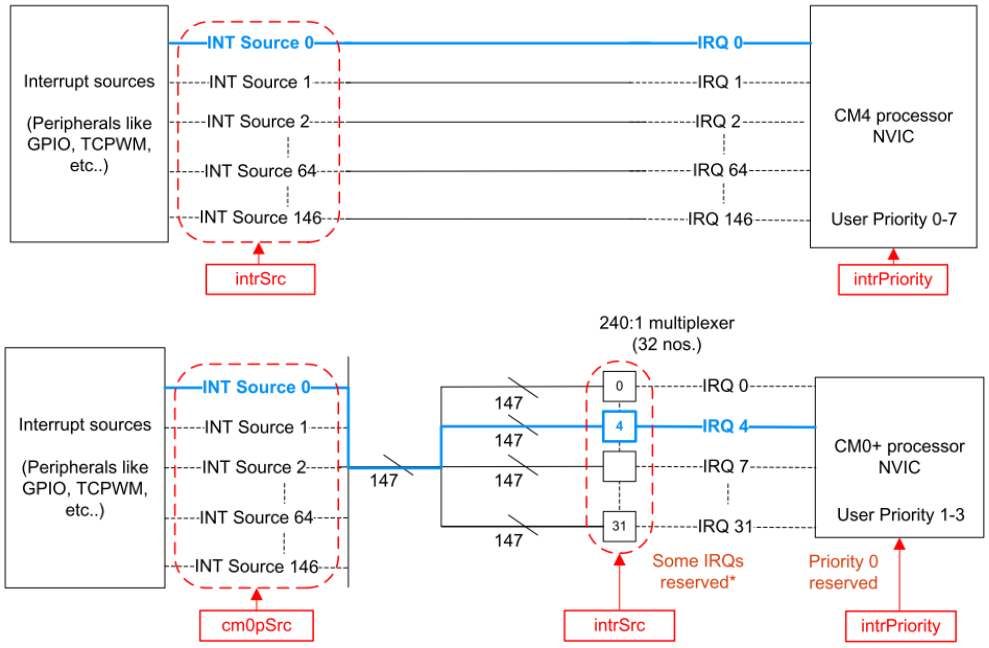

Рис. 5.2 Параметри структури PDL SysInt (виділені червоним кольором), які використовуються для конфігурації переривань.

A sample configured path is highlighted in blue.

Приклад налаштованого шляху виділений на рисунку синім кольором.

3. Виклик Cy_SysInt_Init(&SysInt_SW_cfg_1, ISR_1_handler).

Тут SysInt_SW_cfg_1 – ім’я налаштованої структури. ISR_1_handler - ім'я обробника переривання, який виконується, при спрацюванні переривання. Ця функція виконує конфігурацію маршрутизації та пріоритету переривання, але не дозволяє її.

4. Виклик NVIC_ClearPendingIRQ(SysInt_SW_cfg_1.intrSrc), щоб очистити будь-які очікувані переривання.

5. Виклик NVIC_EnableIRQ(SysInt_SW_cfg_1.intrSrc), щоб дозволити переривання.

5. Виберіть функцію __enable_irq(), щоб увімкнути глобальні переривання. Це безпечно виконувати як перший крок, оскільки окремі переривання процесора ще не ввімкнено. Ви також можете виконати це пізніше, але переривання відключаються при запуску, якщо не викликано функцію глобального переривання.

Окрім драйвера PDL SysInt, API драйвера системних режимів живлення (SysPm) дозволяє функцію Sleep-on-Exit. Якщо в додатку поряд з перериваннями використовується режим ″сну″ або ″глибокого сну″, ця функція дозволяє внутрішньому програмному забезпеченню зберігати систему в режимі ″сну″ майже весь час, лише прокидається, щоб виконати переривання, а потім негайно повернутися до того ж режиму ″сну″. Програма не повертається до головної функції і залишається або в обробнику переривань, або в тому ж стані ″сну″, якщо функція *Sleep-on-Exit* знову не заборонена.

Cy_SysPm_SleepOnExit(true);

Конфігурування переривань з допомогою середовища розробки PSoC Creator 4.2.

Середовище PSoC Creator надає графічний інтерфейс для маршрутизації сигналів від периферійних пристроїв до лінії IRQ процесора. PSoC Creator забезпечує компонент переривання (SysInt). Цей компонент є елементом інтерфейсу користувача поверх драйвера SysInt PDL. Виходячи з конфігурації в компоненті, PSoC Creator генерує код для ініціалізації периферійних пристроїв, маршрутизації переривань та заповнення структури конфігурації переривання. Це зменшує об’єм коду, який ви повинні написати при налаштуванні переривань.

Використання графічного редактора (TopDesign).

Перетягніть Компонент із каталогу компонентів на TopDesign. Використовуйте TopDesign для розміщення та налаштування периферійних пристроїв, які забезпечують джерело переривання. Зверніться до опису (datasheet) компонент, щоб отримати інформацію про конфігурацію переривання певного периферійного пристрою. Деякі периферійні пристрої надають термінал переривання (наприклад, TCPWM). Помістіть екземпляр компонента SysInt і підключіть його до терміналу переривання периферійного пристрою.

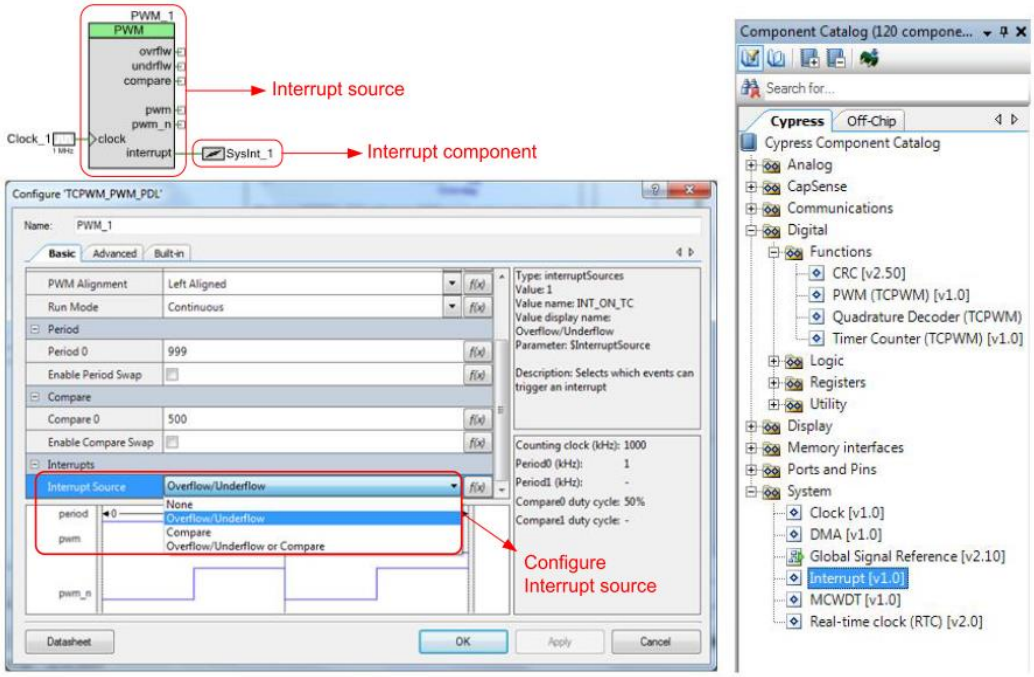

Рис. 5.3. TopDesign з компонентою переривань

Деякі периферійні пристрої не мають зовнішнього терміналу переривання (наприклад, SCB має вбудовані переривання) або можуть мати можливість його відкрити (наприклад, UART).

Компонента переривання має дві налаштовувані опції, які налаштовуються (рис. 5.4).

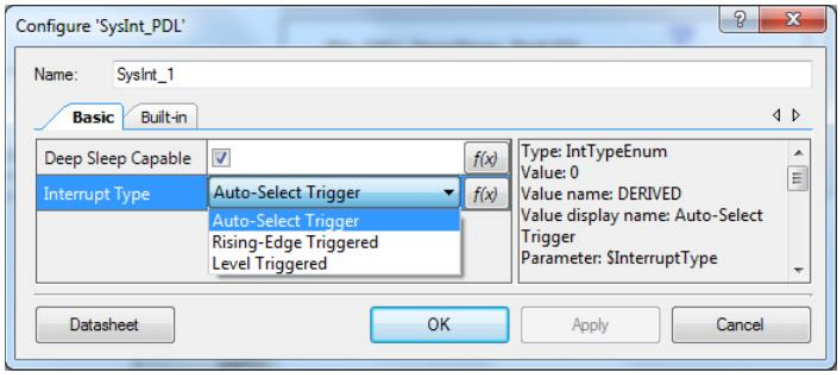

Рис. 5.4. Вигляд вікна конфігурування переривань SysInt

Deep Sleep Capable

Встановіть цей прапорець, якщо хочете, щоб переривання було призначено лінії IRQ ЦП, яка підтримує функцію "глибокого сну". Ви повинні переконатися, що джерело переривання також активне і здатне подавати сигнал переривання під час "глибокого сну", у разі відсутності якого середовище PSoC Creator видасть помилку під час зборки проекту. Ця опція є важливою лише у випадку, якщо переривання призначено для ядра CM0+, у якого є 8 (IRQ 0-7) слотів режиму "глибокого сну" для маршрутизації. Цей прапорець призначений тільки для вказівки щодо автоматичного призначення IRQ для переривання і може бути відмінено ручним призначенням у вікні CyDWR. Для ядра CM4, якщо джерело переривання підтримує режим "глибокого сну" (IRQ 0-40), відключення прапорця (заборона) не впливає на функцію "глибокого сну" переривання.

Тип переривання.

У конфігурації компонента переривання доступні три параметри для типу переривання: переривання з автоматичним вибором, переривання по передньому (додатному) фронту та переривання по рівню. Вибір конкретного параметра залежить від джерела переривання (фіксована функція або UDB/ DSI) та вимог додатку. У більшості випадків залиште опцію автоматичний вибір (Auto-Select), щоб дозволити середовищу PSoC Creator отримувати тип переривання від природи джерела переривання.

Виберіть тільки переривання по рівню для джерел переривань з фіксованою функцією. Виберіть тільки переривання по рівню та переривання по передньому фронті для джерел UDB.

Використання згенерованого коду середовищем PSoC Creator та PDL.

Збірка проекту генерує код для використання в додатку. Папка Pins and Interrupts містить файли з кодом, сформованим за допомогою інформації, що вводиться на вкладці Interrupts в CyDWR.

cyfitter_sysint.h містить макроси з інформацією про номер переривання, призначенні його процесору та пріоритет.

cyfitter_sysint_cfg.c/h оголошує та попередньо заповнює екземпляри структури конфігурації SysInt PDL, використовуючи інформацію CyDWR.

Структура конфігурації для кожного переривання визначається умовно на основі призначення ЦП.

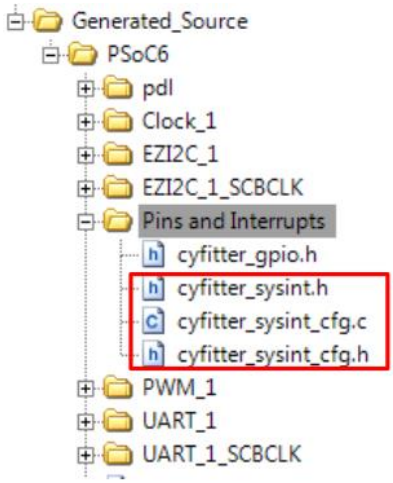

Рис. 5.5. Згенеровані файли

Кроки для дозволу переривань в мікропрограмі подібні до описаних раніше (PDL), але їх менше.

1. Виклик __enable_irq() API для дозволу глобальних переривань.
2. Виклик Cy_SysInt_Init(&SysInt_1_cfg, ISR_1_handler).

- де SysInt_1_cfg - ім'я автоматично сформованої структури з файлу cyfitter_sysint_cfg.c. ISR_1_handler - ім'я обробника переривання, який виконується, коли запускається переривання. Функція обробника може знаходитись у відповідній головній програмі main.c ЦП, якому призначено переривання. Якщо обробник існує поза main.c, цей файл повинен бути скомпільований і зв'язаний з виконуваним файлом для ЦП, який обробляє ISR.

- Цей крок налаштовує переривання (маршрутизація, пріоритет та призначення обробника переривання), але не дозволяє його.

3. Виклик NVIC_ClearPendingIRQ(SysInt_1_cfg.intrSrc), щоб очистити будь-які очікувані переривання.

4. Виклик NVIC_EnableIRQ(SysInt_1_cfg.intrSrc), щоб дозволити переривання.

Розглянемо приклад проекту, який демонструє використання GPIO, налаштованого як вхідний вивід, для генерування переривань на CM4 ЦП в мікроконтролері PSoC 5. Сигнал GPIO перериває програму, виконувану ЦП, і виконує визначену користувачем програму обробки переривання (ISR - Interrupt Service Routine). Переривання GPIO виступає джерелом пробудження, щоб розбудити ЦП з ″глибокого сну″.

Операції:

1. Підключіть комплект до USB-порту комп'ютера.
2. Створіть проект і запрограмуйте його для мікроконтролера PSoC 5. Виберіть Debug>Program.
3. Переконайтесь, що світлодіод блимає чотири рази, а потім вимикається, вказуючи на те, що процесор перейшов у режим глибокого сну.
4. Натисніть перемикач користувача (SW2), підключений до порту P0[4], щоб викликати перервання. Це повинно спричинити пробудження пристрою, що призведе до того, що світлодіод знову почне блимати з новою частотою (більша частота блимання свідчить про те, що ISR виконано). Світлодіод блимає чотири рази, і пристрій знову переходить у режим ″глибокого сну″.
5. Повторне натискання перемикача повторює цикл пробудження, і світлодіод продовжує блимати з початковою частотою 1 Гц. З кожним перериванням і виконанням ISR частота блимання чергується між 1 Гц і 2 Гц.

Проектування та впровадження.

Мікроконтролер PSoC - це мікроконтролер з архітектурою з двома ядрами Arm Cortex M0+ (CM0+) і Arm Cortex M4 (CM4). Процесор CM0+ включає процесор CM4 при перезавантаженні пристрою. В цьому проекті використовується переривання GPIO для пробудження процесора CM4 з ″глибокого сну″. До вихідного виводу підключається світлодіод, який використовується для індикації поточного стану процесора. Блимаючий індикатор вказує на те, що центральний процесор активний. Після чотирьох послідовних блимань ЦП отримує команду перейти в режим ″глибокого сну″. Оскільки стан GPIO зберігається під час ″глибокого сну″, то світлодіод перестає блимати та вимкнеться, щоб вказати, що процесор знаходиться в режимі ″глибокого сну″.

Вхідний вивод, під’єднаний ззовні до перемикача, сконфігурований для генерації переривання при натисканні перемикача. Переривання викликає наступні дії:

1. Генерує сигнал, який виводить процесор з режиму ″глибокого сну″.
2. Виконує програму обробки переривання (ISR).

Коли виконується ISR, оновлюється прапорець, який використовується для зміни швидкості блимання світлодіода. З кожним натисканням перемикача світлодіод по черзі блимає з періодами 500 мсек та 1 сек.

На рис. 5.5 показано схему цього проекту в PSoC Creator. Проект використовує компоненти GPIO, Global Signal Reference та SysInt.

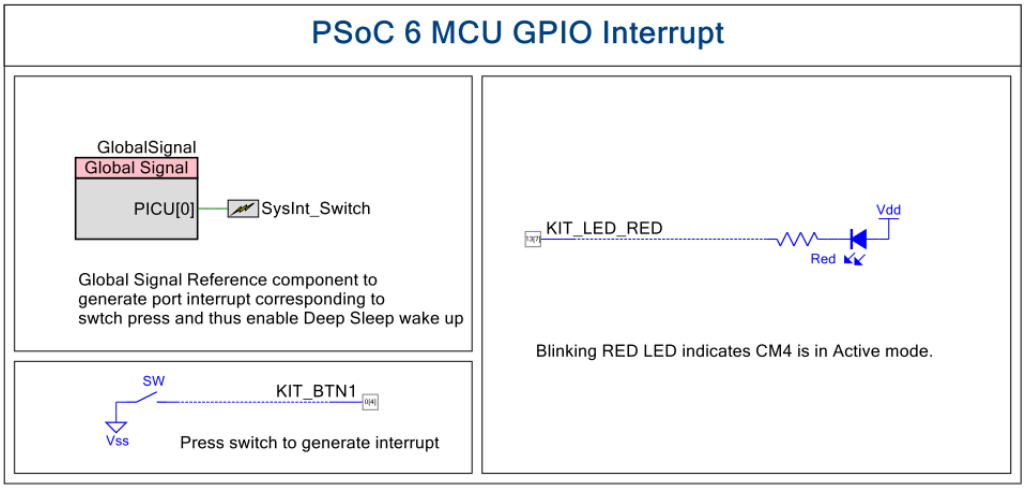

Рис. 5.5. Схема проекту для переривань GPIO в PSoC Creator 4.2

Компонента Global Signal Reference використовується для маршрутизації переривання порту з вхідного виводу. Переривання портів доступні в режимі "живлення глибокого сну" і, таким чином, можуть розбудити процесор з режиму "глибокого сну". Компоненту SysInt також можна підключити безпосередньо до виводу, як показано на рис. 5.5. Це призводить до того, що контакт використовується як джерело переривання UDB. Однак цей сигнал переривання направляється через цифрове системне з'єднання (DSI) і недоступний під час "глибокого сну" і не може розбудити процесор.

Рис. 5.5. Пряме під'єднання вхідного виводу до компоненти SysInt

На рис. 5.7 … Рис. 5.9 зображені параметри, які не використовуються за замовчуванням.

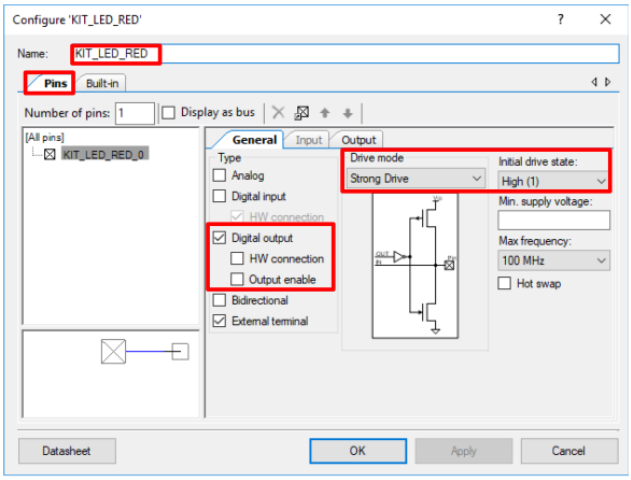

Рис. 5.7. Конфігурація виводу GPIO для LED

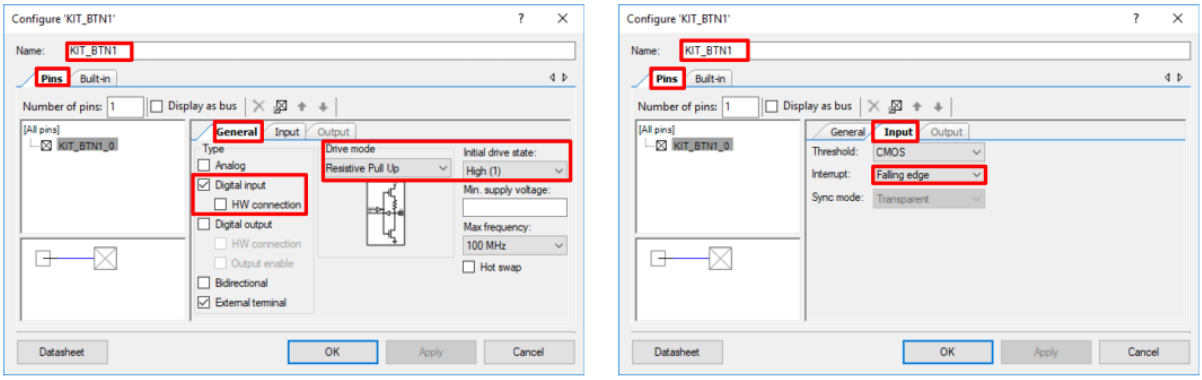

Рис. 5.8. Конфігурація виводу GPIO для входу перемикача

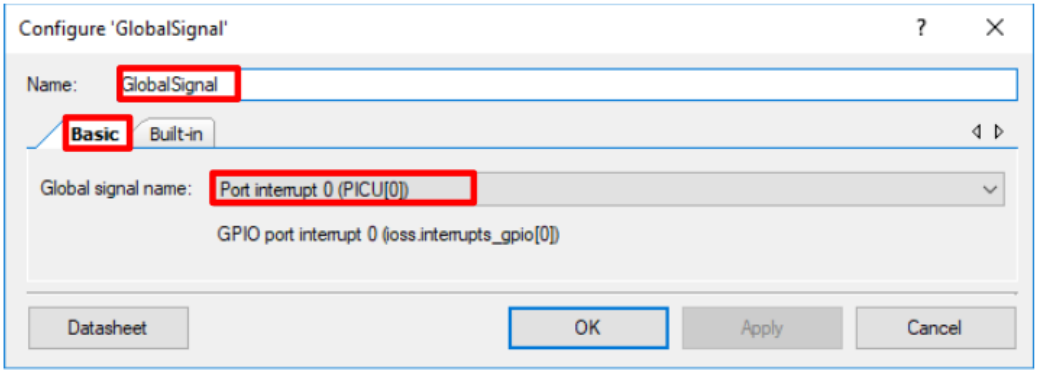

Рис. 5.9. Конфігурація Global Signal Reference

Ресурси проекту та глобальні ресурси.

Призначення виводів для перемикача SW2 та світлодіодів стенду CY8CKIT-062 BLE:

Digital Input Pin (SW2) - KIT_BTN1 P0[4]
Digital Output Pin (LED) - KIT_LED_RED P13[7

На рис. 5.10 показана конфігурація переривання для цього проекту. Відмітимо, що компонента SysInt призначена ядру CM4 через прапорець. Пріоритет переривань залишається за замовчуванням, оскільки немає інших переривань.

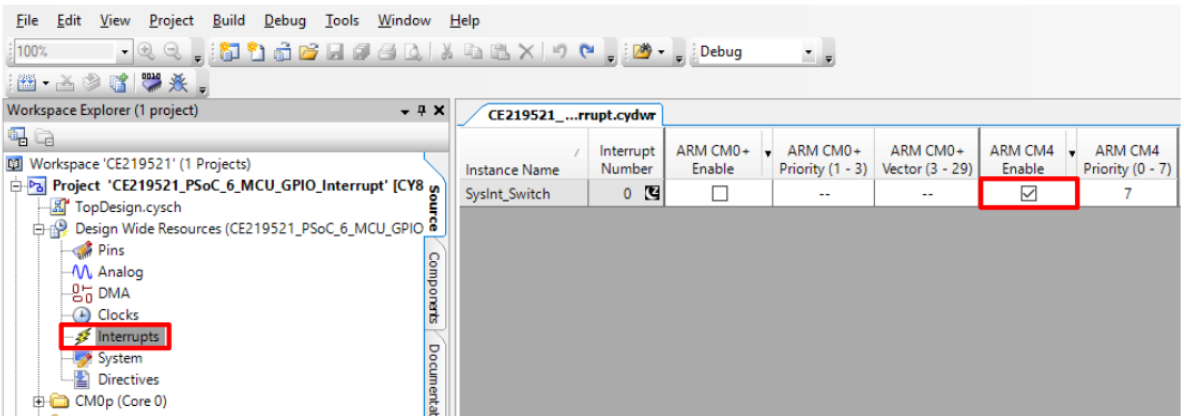

Рис. 5.10. Конфігурація системи переривань

Алгоритм програми:

```
#include "project.h"

/* Макроси для налаштування функціональності проекту */
#define DELAY_SHORT                (500)     /* msec */
#define DELAY_LONG                 (1000)    /* msec */

#define LED_BLINK_COUNT            (4)

#define LED_ON                     (0)
#define LED_OFF                    (1)

#define SWITCH_INTR_PRIORITY       (3u)

/* Global variable */
uint32_t interrupt_flag = false;
```

Функція: `void Switch_ISR(void).` Ця функція виконується, коли спрацьовує переривання GPIO.

```
void Isr_switch(void)
{
     /* Очищення спрацьованого переривання виводу */
      Cy_GPIO_ClearInterrupt(KIT_BTN1_PORT, KIT_BTN1_NUM);
      NVIC_ClearPendingIRQ(SysInt_Switch_cfg.intrSrc);

     /* Встановити прапорець переривання */
      interrupt_flag = true;
}

int main(void)
{
      uint32_t count = 0;
      uint32_t delayMs = DELAY_LONG;

    /* Глобальний дозвіл переривань */
    __enable_irq();

    /* Ініціалізація та дозвіл переривань GPIO */
      Cy_SysInt_Init(&SysInt_Switch_cfg, Isr_switch);
      NVIC_ClearPendingIRQ(SysInt_Switch_cfg.intrSrc);
      NVIC_EnableIRQ(SysInt_Switch_cfg.intrSrc);

      for(;;)
      {
        /* Зміна статусу переривання */
            if(true == interrupt_flag)
            {
                  interrupt_flag = false;

            /* Оновлення частоти блимання LED */
                  if(DELAY_LONG == delayMs)
                  {
                        delayMs = DELAY_SHORT;
                  }
                  else
                  {
                        delayMs = DELAY_LONG;
```

```
                }
            }

        /* Цикл блимання світлодіодом LED LED_BLINK_COUNT разів*/
            for (count = 0; count < LED_BLINK_COUNT; count++)
            {
                Cy_GPIO_Write(KIT_LED_RED_PORT, KIT_LED_RED_NUM, LED_ON);
                Cy_SysLib_Delay(delayMs);
                Cy_GPIO_Write(KIT_LED_RED_PORT, KIT_LED_RED_NUM, LED_OFF);
                Cy_SysLib_Delay(delayMs);
            }
            /* Перехід в режим "глибокого сну" */
            Cy_SysPm_DeepSleep(CY_SYSPM_WAIT_FOR_INTERRUPT);
    }
}
```

Функції, що використовуються в цьому проекті.

`void NVIC_ClearPendingIRQ (IRQn_Type IRQn);`

Ця функція видаляє стан очікування заданого переривання IRQn. IRQn не може бути від'ємним числом.

Параметри:

[in] IRQn Interrupt number

Зауваження:

- регістри, які керують станом переривань: SETPEND, CLRPEND.
- переривання може мати статус очікування, хоча воно є неактивним.

`void NVIC_EnableIRQ (IRQn_Type IRQn);`

Ця функція дозволяє вказане переривання IRQn для конкретного пристрою. IRQn не може бути від'ємним числом.

Параметри:

[in] IRQn Interrupt number

Зауваження:

- регістри, які керують дозволом та забороною переривань: SETENA, CLRENA.
- кількість підтримуваних переривань залежить від реалізації проекту кристалу і може бути зчитана з регістру типів переривань (ICTR - Interrupt Controller Type Register) з дискретністю 32:

ICTR[4:0]=

- 0 – підтримується 32 переривання;
- 1 – підтримується 64 переривання.
- 2 – …

IRA - interrup: request acknowledge (підтвердження запиту на переривання);

IRQ - interrupt request (запит на переривання);

ISR - interrupt service routine (програма обробки переривання);

IVR - interrupt vector read (читання вектора переривань);

Опис літератури:

CE219521 – PSoC 6 MCU - GPIO Interrupt. Електронний ресурс. Режим доступу: https://www.cypress.com/file/385796/download

Завдання.

1. Реалізувати проект засвічування світлодіодів LED8 та LED9 стенду PSoC 6 BLE PIONER KIT. з використанням переривань GPIO, до якого під’єднано перемикач SW_2. Алгоритм засвічування світлодіодів: при включенні живлення стенду світлодіод LED8 блимає з частотою 1 Гц 8 разів. Після цього процесор переходить в режим ″глибокого сну″. З цього режиму його виводить натискання на кнопку SW_2. При настанні переривання починає блимати 4 рази світлодіод LED9 з частотою 2 Гц.

Світлодіоди LED8 і LED9 можуть працювати у всьому діапазоні робочих напруг PSoC 6 MCU. Світлодіоди засвічуються низьким рівнем сигналу ″0″. Тому їх виводи повинні бути ″заземлені″ для того, щоб їх засвітити.