Міністерство освіти і науки України

Національний університет України

«Львівська політехніка»

МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ

ЛАБОРАТОРНИХ РОБІТ

З курсу «Мікроконтролери сімейства STMicroelectronics»

ДЛЯ СТУДЕНТІВ СПЕЦІАЛЬНОСТІ -5-84 –«ІНФОРМАЦІЙНО-УПРАВЛЯЮЧІСИСТЕМИТА ТЕХНОЛОГІЇ»

ЛЬВІВ-2016 р.

Міністерство освіти і науки України

Національний університет України

«Львівська політехніка»

МЕТОДИЧНІ ВКАЗІВКИ ДО ВИКОНАННЯ

ЛАБОРАТОРНИХ РОБІТ

З курсу «Мікроконтролери сімейства STMicroelectronics»

ДЛЯ СТУДЕНТІВ СПЕЦІАЛЬНОСТІ -5-84 –«ІНФОРМАЦІЙНО-УПРАВЛЯЮЧІСИСТЕМИТА ТЕХНОЛОГІЇ»

(Частина 1)

ЛЬВІВ-2016 р.

Методичні вказівки з курсу «Моделювання систем», «Основи системного аналізу комп’ютерних інформаційних систем», «Системне програмування» для студентів спеціальності 050804 – «Інформаційно-управляючі системи та технології»

Укладачі Начальник відділу ПКО «Політехніка» Гривна Юрій Теодорович

Пров.інженер ПКО «Політехніка» Рибак Ярослав Миколайович

Рецензент докт.техн.наук, проф.каф.АСУ.- Цмоць Іван Григорович

Зміст

Вступ

Лабораторна робота №1 5

Дослідження команд роботи процесора та режимів робити портів на

прикладі світлодіодної лінійки та пристрою «дві кнопки» 6

Лабораторна робота №2 10

Система переривань мікроконтролерів STM8, "сплячий режим" 10

Лабораторна робота №3 17

Дослідження внутрішніх таймерів та широтно-імпульсної модуляції (PWM) на

прикладі світлодіодної лінійки, пристрою «2 кнопки» 17

Лабораторна робота №4 25

АЦП мікроконтролерів STM8 .Робота з давачем аналогової напруги 25

Додаток А 32

Лабораторний стенд STM8 31

Додаток Б

Відналагодження програм у середовищі ST Visual Develop

Додаток В

Лабораторний стенд STM8 (керування кроковим двигуном), 58

Додаток Г 62

Керування двигуном постійного струму ІЧ-пультом 62

Додаток Д 71

Пасивний інфрачервоний давач руху 71

Вступ

Метою виконання лабораторних робіт є закріплення теоретичних знань основних принципів побудови і функціонування мікропроцесорних пристроїв на базі одно кристальних мікроконтролерів , використання набутих практичних навиків написання програм на асемблері та вивчення програмних та апаратних засобів від лагодження програм.

У даних методичних вказівках наведено варіанти завдань до 4 лабораторних робіт ,що охоплюють відомості по основним функціональним блокам мікроконтролерів сімейства STMicroelectronics, а саме портам введення та виведення інформації ,системи переривань, таймеру та АЦП.

В кожній роботі наведено теоретичні відомості ,приклад виконання та контрольні запитання.

В додатку приведена конфігурація лабораторного стенду ,приклад встановлення і роботи програмного забезпечення ,пристрої на базі базового модуля .а саме:

- керування кроковим двигуном;

- керування двигуном постійного струму ІЧ-пультом;

- пасивний інфрачервоний давач руху.

Матеріали даних методичних вказівок можуть бути використані при курсовому і дипломному проектуванні.

**Лабораторна робота №1**

**Дослідження команд роботи процесора та режимів роботи портів на прикладі**

**світлодіодної лінійки та пристрою "дві кнопки".**

**Завдання:**

Ініціалізувати порти, змінювати режим світіння світлових діодів після кожного

натискання на кнопки S1 і S2 згідно табл,1,1. Режим світіння світлових діодів задано в табл. 1.1.

Таблиця 1.1. Режими світіння

|  |  |
| --- | --- |
| Режим | Опис |
| 1 | Послідовно засвічувати світлові діоди так, щоб утворилася «крапка, що  Біжить». Час світіння кожного світлодіода дорівнює Т1. Напрямок  пересування: зліва направо. Після натискання кнопки S1 швидкість  пересування збільшується удвоє. |
| 2 | Послідовно засвічувати світлові діоди так, щоб утворилася «крапка, що  Біжить». Час світіння кожного світлодіода дорівнює Т1. Напрямок пересування:  справа наліво. Після натискання кнопки S1 змінити напрямок пересування. |
| 3 | Послідовно засвічувати та гасити кожен червоний та зелений світловий  діод. Час світіння кожного світлодіода дорівнює Т1. |
| 4 | Циклічно засвічувати та гасити зелені світлодіоди. Час світіння кожного  світлодіода дорівнює Т1. |
| 5 | Циклічно засвічувати та гасити червоні світлодіоди . Час світіння кожного світлодіода дорівнює Т1. Після натискання кнопки 51 час світіння кожного світлодіода дорівнює Т1/2. |
| 6 | Циклічно засвічувати та гасити зелені світлодіоди. Час світіння кожного  світлодіода дорівнює Т4 Після натискання кнопки S1 час світіння кожного  світлодіода дорівнює Т1/2. |

**Порядок роботи з макетом:**

* Під'єднати до базової плати пристрій «2 кнопки».
* Під'єднати до базової плати адаптер живлення.
* Під'єднати до базової плати « Програматор ST-LINK», останній USB-кабелем А-В з персональним комп'ютером з інтегрованим середовищем розробки STVD.
* Процес інсталяції пакету ST Visual Develop (Версія 4.3.1) наведено в Додатку Б.

**Теоретичні відомості**

**Порти введення-виведення.** Мікроконтролер STM8 (в корпусі SDIP-32) має 22 ліній

вводу/виводу, з яких 12 ліній мають підвищену здатність до навантаження.

Кожен вивід порту може бути запрограмований на введення або виведення інформації.До того ж, окремі виводи мають декілька інших Функцій, як-то: зовнішнє переривання,дублювати сигнал введення/виведення для периферійного пристрою на кристалі або для аналогового введення.

Для керування лініями порт МК має по п'ять регістрів для кожного порту, це регістри даних порта (РхОDR) і (РхIDR), регістр напряму передачі даних (РхDDR), регістри опцій (РхСR1),(РхСR2).Ініціалізація портів полягає у запису даних в регістри напрямку та опцій, Скидання DDRx біта в О вибирає режим введення. Установка DDRx біта в 1 визначає режим виведення. Дія бітів регістру опцій DCRx в режимах введення та виведення відображена в табл. 1.2.

Таблиця 1.2. Спеціалізовані комірки пам'яті, що керують портами вводу-виводу.

|  |  |  |  |
| --- | --- | --- | --- |
| Адр. | Назва спеціалізованої комірки | | Значення біт для кожної ніжки |
| 00x0 | Px\_ODR | Port X Output Data Register  Вихідні дані порту | Регістр даних на виведення |
| 0x01 | Px\_IDR | Port X Input Data Register  Вхідні дані порту | Регістр даних на введення |
| 0x02 | Px\_DDR | Port X Data Direction Register  Біти визначають  напрямки сигналів порту для кожної ніжки | 0- вивід порту знаходиться в режимі вводу даних або очікування зовнішнього  напрямки сигналів порту переривання;  1 - вивід порту знаходиться в режимі виводу |
| 0x03 | Px\_CR1 | Port X Control Pegister 1 Визначає режим виводу  або наявність підтягуваль-  ного резистора до +Vdd  при вводі. | 0 при вводі - високоімпедансний стан;  1 при вводі - вмикає підтягувальний резистор;  0 при виводі - відкритий колектор;  1 при виводі - звичайний логічний вихід. |
| 0x04 | Px\_CR2 | Port X Control Pegister 2 Визначає яка ніжка при  вводі пов'язана  перериванням | 0 при вводі - переривання не задіяне;  1 при вводі - зміна стану викликає  вводі пов'язана з підпрограму-обробник переривання;  0 при виводі - швидкість даних до 2 МГЦ;  1 при виводі - швидкість даних до 10 МГЦ. |

Ініціалізація портів полягає у запису даних в регістри напрямку та опцій:

init\_port B: Id A,#INITPB\_DDR :0-вхід 1 - вихід відповідного біту регістра DDR

Id PB\_DDR,A

Id A,#INITPB,CR1 ;0 - для вхідного виводу - високоімпедансний

;вхід; для вихідного виводу - вихід за схемою а

;відкритим колектором;

;1 - для вхідного виводу - вхід з підтягувальним

;резистором; для вихідного виводу - вихід за

;двотактною схемою

Id PB\_CR1,A

ret ;повернення до головної програми

Після цього можна виконувати введення та виведення через порти даних. Підпрограма запису в парт має вигляд:

write-port B: Id A,port B\_TX ;Завантаження значення вмісту комірки пам'яті

;p0rtB\_TX в акумулятор

Id PB\_0DR,A ;завантаження вмісту регістра на виведення портуВ ;повернення до головної програми

;3 акумулятора

ret ;повернення до головної прогрвми

Підпрограма читання порту має вигляд:

read\_port B: Id A,PB\_|DR ;завантаження вмісту регістра IDR порту В в

;акумулятор

Id portB\_RX,A ;завантаження вмісту акумулятора у комірку

;пам'яті portB\_RX

Ret ; повернення до головної програми

**Приклад виконання лабораторного завдання**

Завдання:

Послідовно засвічувати світлові діоди так, щоб утворилася «крапка, що біжить». Час світіння кожного світлодіода дорівнює T1, Напрямок пересування: зліва направо. Принатисканні кнопки 51 (чорна кнопка) швидкість пересування збільшувати удвоє, а після третього натискання кнопки S1 повернутись до попередньої швидкості переміщення «крапки, що біжить».

Нижче приведено текст програми.

start: ;основна програма

sim ;заборона переривань

call init\_port ;ініціалізація портів

time.b

clrw Y

ldw X,#$ffff ;встановлення T1

start1:

Id A,#$fe

Id PD\_ODR,A ;вмикаємо red №1

call knopka ;аналіз стану кнопок

call red\_off ;вимикаємо гесі №1

Id A,#$ef

Id PC\_ODR,A ;вмикаємо green №1

call knopka

саll green\_off ;вимикаємо green №1

Id A,#$fd

ld PD\_ODR,A ;вмикаємо red №2

call knopka

call red\_off, ;вимикаємо red №2

Id A,#$df

ld PC\_ODR,A ;вмикаємо green №2

call knopka

call green\_off ;вимикаємо green №2

Id A,#$fb

Id PDfODR,A ;вмикаємо red №3

call knopka

caII red\_off ;вимикаємо гесі №3

Id A,#$bf

ld PC\_0DR,A ;вмикаємо green №3

call knopka

call green\_off ;вимикаємо green №3

Id A,#$f7

Id PD\_ODR,A ;вмикаємо red №4

call knopka

call red\_off ;вимикаемо red N94

Id А,#$7f

Id PC\_ODR,A ;вмикаємо green №4

call knopka

call green\_off ;вимикаємо green №4

call knopka

cpw Y,#3

jrult start1 ;зациклюємо програму

jpf start

init\_port: ;підпрограма ініціалізації портів

Id A,#%00001111

Id PD\_DDR,A ;порт D керує red - світлодіодами

Id PD\_CR1,A ;DO — D3 B режимі виводу

Id PD\_CR2,A

Id A,#%11110000

ld PC\_DDR,A ;порт С керує green - світдодіодами

Id PC\_CR1,A ;C4‘—.C7 в режимі виводу

Id A,#% 00000000

ld PB\_DDR,A ;порт PB в режимі вводу стану

ret ;кнопок S1 і S2

red\_off: ; підпрограма вимкнення гесі - світлодіодів

Id A,#$ff

Id PD\_ODR,A

ret

green\_off: ; підпрограма вимкнення green — світлодіодів

Id A,#$ff

Id PC\_ODR,A

ret

delay: ; підпрограма затримки на 0,5сек.

clrw X

loop: ;коеф. зміни Т1 в комірці time

incw X

cpw X,time

jrult loop

ret

knopka: ;підпрограма аналізу стану кнопок

btjt PB\_lDR,#0,knopka1

call delay ;затримка після натискання кнопки

call delay

Idw X,time ;якщо кнопка 51 натиснута, то змінити

srlw X ;час Т1 на Т1/2

Idw time,X

incw Y

knopka1:

call delay

ret

**Контрольні запитання**

1. Дайте характеристику МК сімейства STM8.
2. Як відбувається звернення до портів введення-виведення?
3. Які можливості конфігурації мають порти введення-виведення?
4. Назвіть режими введення
5. Назвіть режими виведення.

**Лабораторна робота N92**

**Система переривань мікроконтролерів SТМ8, "сплячий режим".**

**Завдання:**

Написати програму керування заданими світлодіодами за допомогою кнопок S1 та S2

згідно заданому у варіанті способу (таблиця 2.1). Для цього ініціалізувати систему переривань для роботи у заданому режимі та організувати обробку відповідних переривань.

Таблиця 2.1

|  |  |
| --- | --- |
| Режим | Опис |
| 1. | Засвічувати та гасити зелені світлодіоди (3 рази з періодом 0,5сек.) по передньому та низькому рівню при натисканні кнопки S1. Циклічно з періодом 1сек. засвічувати та гасити червоні світлодіоди. |
| 2. | Засвічувати та гасити червоні світлодіоди (3 рази з періодом 0,5сек.) по передньому танизькому рівню при натисканні кнопки S2. Циклічно з періодом 1сек. засвічувати та гасити зелені світлодіоди . |
| 3. | Засвічувати та гасити заданий світлодюд заданою кнопкою по задньому фронту. Після кожного третього натискання кнопки эасвічувати всі світлодіоди. |
| 4. | Засвічувати та гасити заданий світлодіод заданою кнопкою по передньому та задньому Фронту. Кожне третє натискання кнопки ігнорувати. |

Порядок роботи з макетом:

* Під'єднати до базової плати пристрій «2 кнопки».
* Під'єднати до базової плати адаптер живлення.
* Під'єднати до базової плати «Програматор ST-LINK», останній USB-кабелем А-В з персональним комп'ютером з інтегрованим середовищем розробки STVD.
* Процес інсталяції пакету ST Visual Developer (Версія 4.3.1) наведено в Додатку Б.

**Теоретичні відомості**

В попередній роботі для Формування затримки в програмі миготіння світлодіода

використовувався 16- розрядний лічильник з використанням регістра Х. При формуванні затримки процесор ЗЭЙНЯТИЙ добавлянням ОДИНИЧКИ В ЛІЧИЛЬНИК, Даремно витрачаючи електроенергію Для вирішення цієї проблеми в контролері STM8 передбачений пристрій AWU - Auto Wake Up. Цей пристрій виводить процесор із сплячого режиму через вказаний час. Блок - схема пристрою AWU наведена на рис.2.2. У контролера є внутрішній RC генератор, його частота ділиться на деяке число і подається на лічильник, який теж рахує до певного числа. Після завершення рахунку генерується переривання, яке виводить процесор із сплячки. Час, через який потрібно вивести процесор із стану "сну", задається базовим часовим регістром AWU\_TBR і в середині діапазону значенням регістра подільника AWU\_APR. Формат регістру AWU\_TBR подано на рис.2.1.

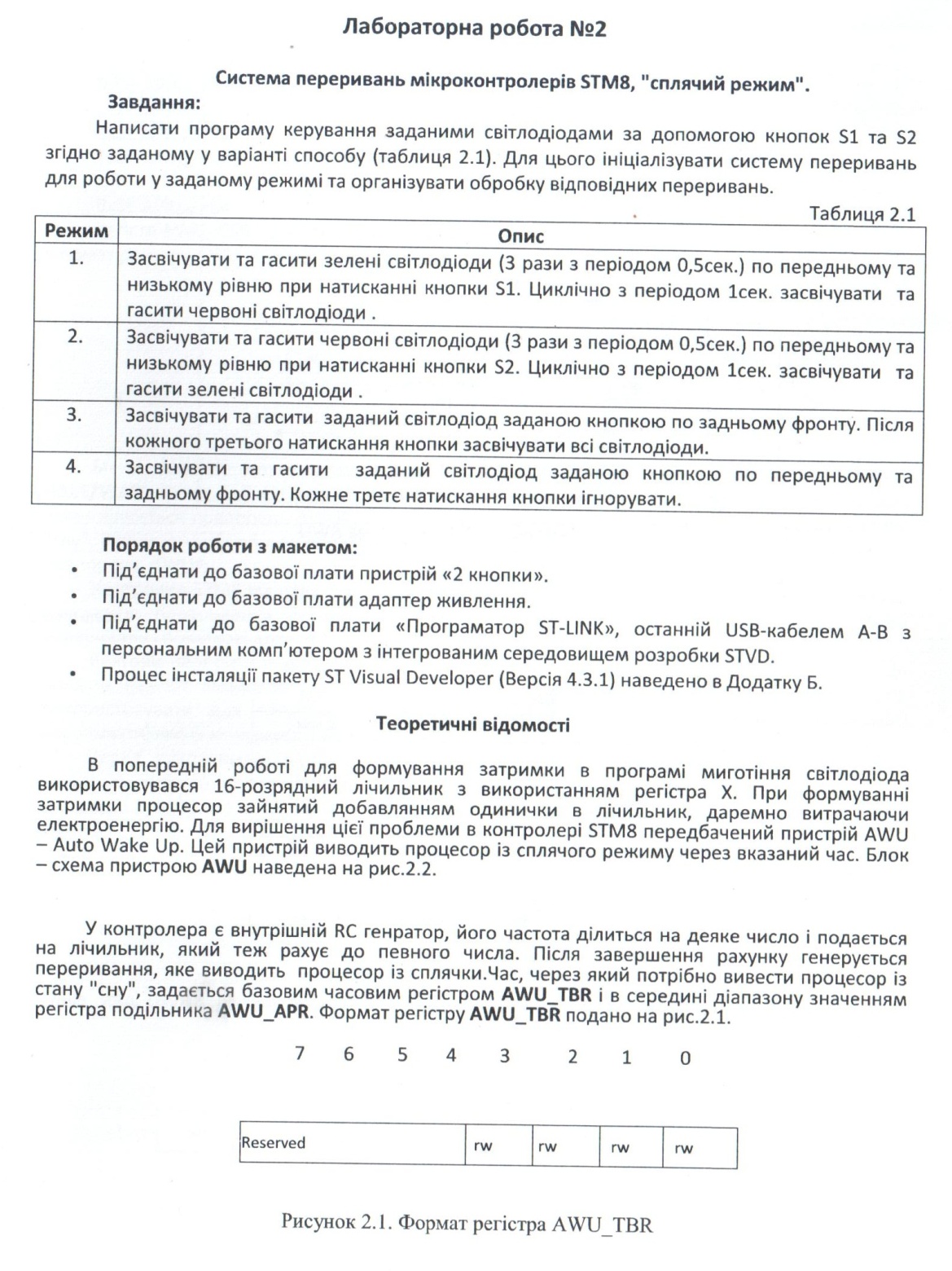


Рисунок 2.1 Формат регістра AWU\_TBR

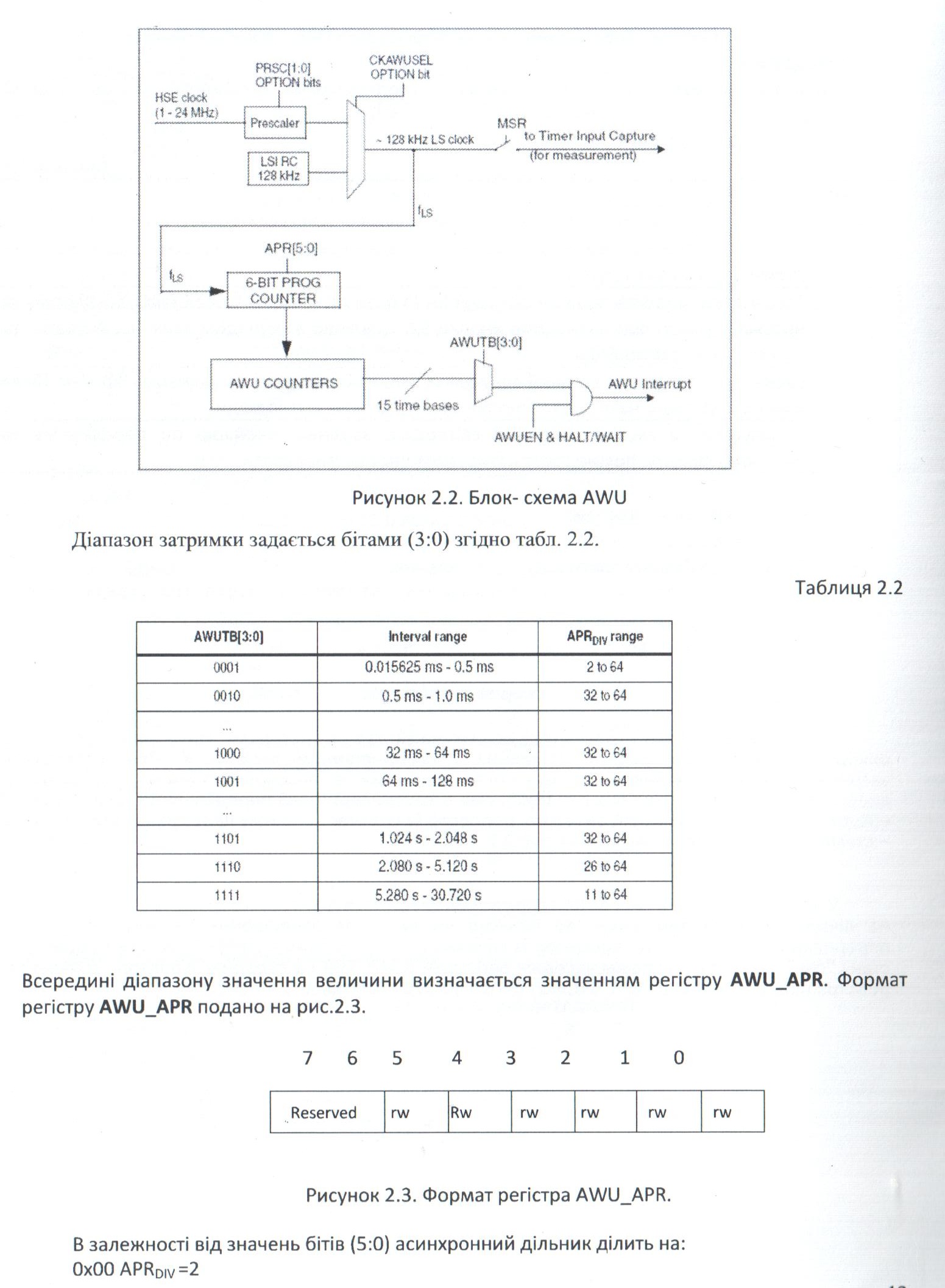
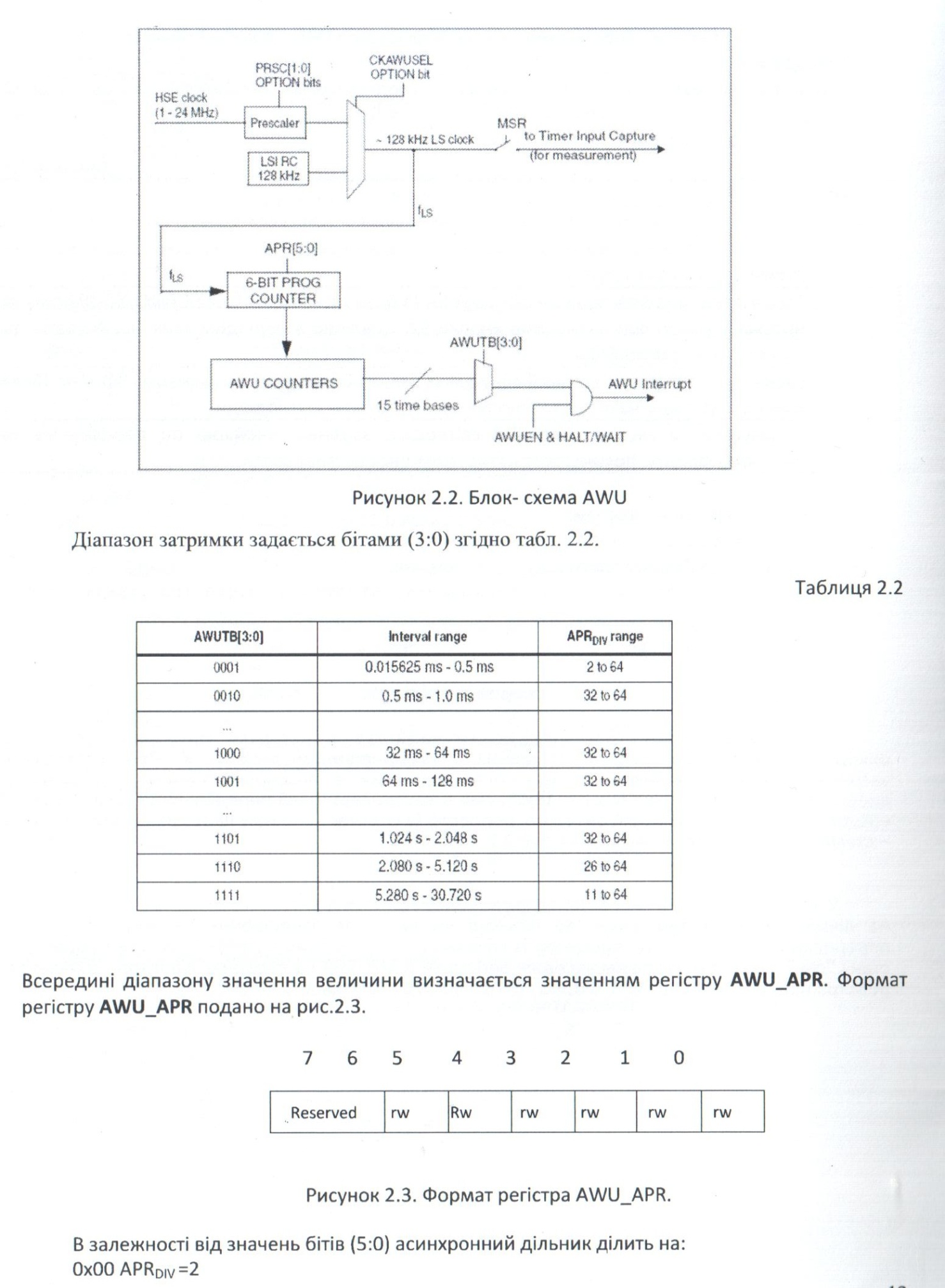


Рисунок 2.2. Блок-схема AWU

Діапазон затримки задається бітами (3:0) згідно табл. 2.2.

Таблиця 2.2.



Всередині діапазону значення величини визначається значенням регістру AWU\_APR. Формат регістру AWU\_APR подано на рис.2.3.

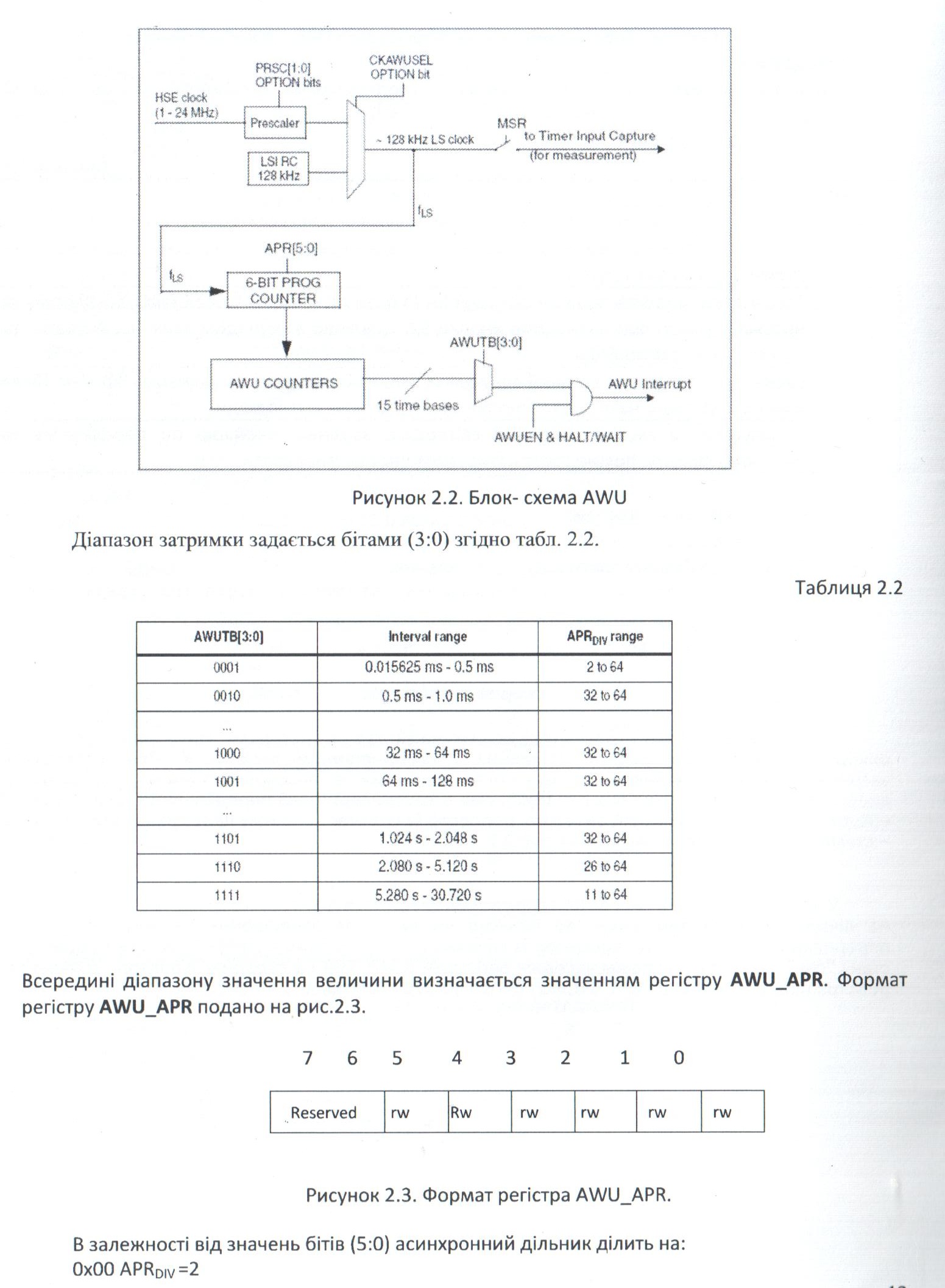


Рисунок 2.3. Формат регістра AWU\_APR.

B залежності від значень бітів (5:0) асинхронний дільник ділить на:

0x00 =2

0x01 =3

……

OxOE =8

……

OxOE =16

OxOF =17

……

ОХЗЕ =64

Регістр **AWU\_CSR** - регістр стану/керування.

Формат регістру AWU\_CSR подано на рис.2.4.

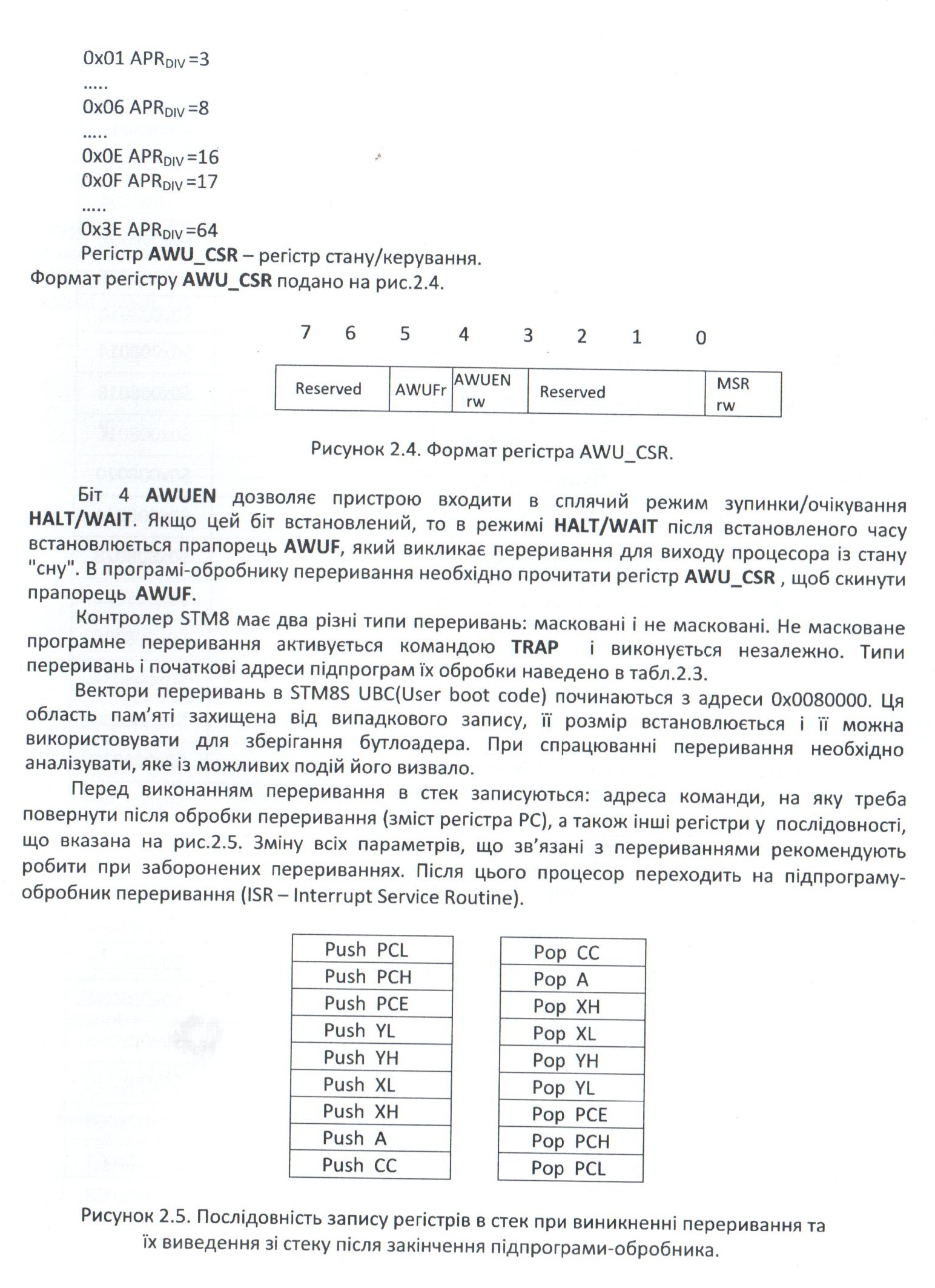


Рисунок 2.4. Формат регістра AWU\_CSR.

Біт 4 AWUEN дозволяє пристрою входити в сплячий режим зупинки/очікування

НАLT/МІАІІ Якщо цей біт встановлений, то в режимі HALT/WAIT після встановленого часу встановлюється прапорець AWUF, який викликає переривання для виходу процесора із стану "сну". В програмі-обробнику переривання необхідно прочитати регістр AWU\_CSR , щоб скинути прапорець AWUF.

Контролер STM8 має два різні типи переривань: масковані і не масковані. Не масковане програмне переривання активується командою TRAP i виконується незалежно. Типи переривань і початкові адреси підпрограм їх обробки наведено в табл.2.4.3.

Вектори переривань в STM8$ UBC(User boot code) починаються з адреси 0x0080000. Ця область пам'яті захищена від випадкового запису, її розмір встановлюється і її можна використовувати для зберігання бутлоадера. При спрацюванні переривання необхідно аналізувати, яке із можливих подій його визвало.

Перед виконанням переривання в стек записуються: адреса команди, на яку треба повернути після обробки переривання (зміст регістра РС), а також інші регістри у послідовності, що вказана на рис.2.5. Зміну всіх параметрів, що зв'язані з перериваннями рекомендують робити при заборонених перериваннях. Після цього процесор переходить на підпрограму- обробник переривання (ISR— Interrupt Service Routine).

|  |
| --- |
| Push PCL |
| Push РСН |
| Push PCE |
| Push YL |
| Push YH |
| Push XL |
| Push XH |
| Push A |
| Push CC |

|  |
| --- |
| C |
| А |
| XH |
| XL |
| YH |
| YL |
| PCE |
| РСН |
| PCL |

Рисунок 2.4.5. Послідовнісгь запису регістрів в стек при виникненні переривання та їх виведення зі стеку після закінчення підпрограми-обробника.

Таблиця 2.3. Джерела переривань

|  |  |  |  |
| --- | --- | --- | --- |
|  | Джерело о | Опис | Адреси  підпрограми  (вектор |
| 1 | RESET | Скидання | $0х008000 |
| 2 | TRAP | Програмне переривання | $0x008004 |
| 3 | TLI | Задає пріоритет переривань | $0><00801C |
| 4 | AWU | Автопробудження від «сплячки» | $Ох008018 |
| 5 | CLK | Сигнали від генератора | $Ох008014 |
| 6 | ЕХТЮ | Зміна станів ніжок порту А | $Ox008010 |
| 7 | EXTIl | Зміна станів ніжок порту В | $Ох00800С |
| 8 | EXTIZ | Зміна станів ніжок порту С | $0х008008 |
| 9 | ЕХТІЗ | Зміна станів ніжок порту D | $0х00803С |
| 10 | ЕХТИ | Зміна станів ніжок порту Е | $0х008038 |
| 11 | CAN | Переривання від RX | $0х008034 |
| 12 | CAN | Переривання від TX/ER/SC | $0x008030 |
| 13 | SPl | Закінчення передачі по SPІ | SOXOOSOZC |
| 14 | TIM1 | Оновлення / Переповнення / Зупинка | $0x008028 |
| 15 | ТіМ1 | Захоплення / Успішне порівняння | $0х008024 |
| 16 | ТіМ2 | Оновлення / Переповнення | $0x008020 |
| 17 | TIM2 | Захоплення / Успішне порівняння | $Ox008040 |
| 18 | TIM3 | Оновлення / Переповнення | $0х008044 |
| 19 | ТІМЗ | Захоплення /Успішне порівняння | $0х008048 |
| 20 | UART1 | Завершення Тх UART1 | $0x00804C |
| 21 | UART1 | PeriCTp даних UART1 заповнений | $0x008068 |
| 22 | l2C | Переривання після передачі / прийому | $0х008064 |
| 23 | UART | Завершення Тх UART 2/3 | $0х008060 |
| 24 | UART | Регістр даних UART 2/3 заповнений | $0х00805С |
| 25 | ADC | Закінчення роботи АЦП | $0x008058 |
| 26 | ТІМЗ | Оновлення/ Переповнення | $0х008054 |
| 27 | FLASH | Закінчення операції запису в EEPROM | $0х008050 |

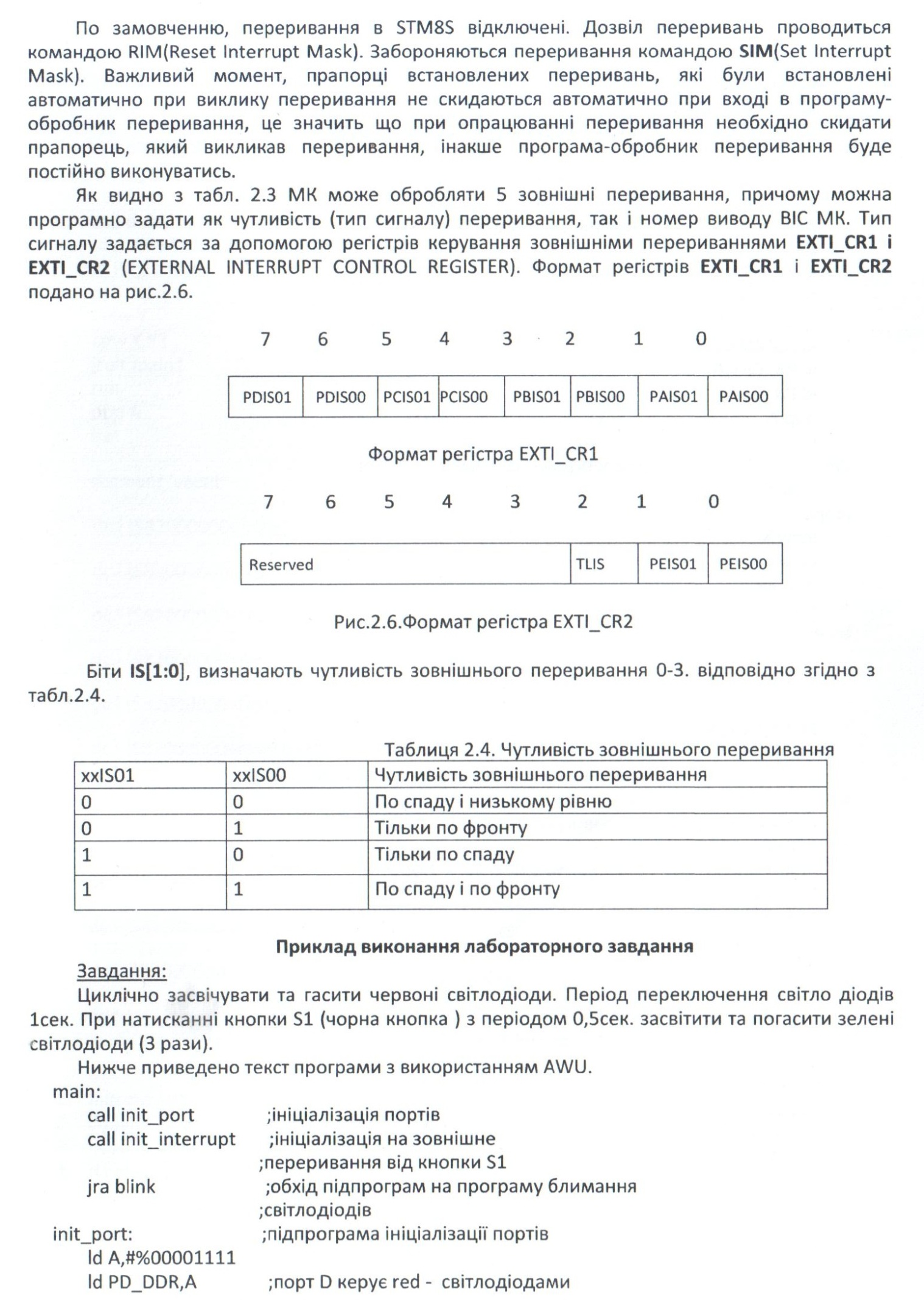
Програма обробки переривання повинна закінчуватися командою IRET, яка відновлює значення збережених регістрів із стека.

По замовченню, переривання в 5ТМ85 відключені, Дозвіл переривань проводиться

командою RIM(Reset Interrupt Mask). Забороняються переривання командою S|M(Set Interrupt Mask). Важливий момент, прапорці встановлених переривань, які були встановлені автоматично при виклику переривання не скидаються автоматично при вході в програму-обробник переривання, це значить що при опрацюванні переривання необхідно скидати прапорець, який викликав переривання, інакше програма-обробник переривання буде постійно виконуватись.

Як видно з табл. 2.3 МК може обробляти 5 зовнішні переривання, причому можна

програмно задати як чутливість (тип сигналу) переривання, так і номер виводу BIC MK. Тип сигналу задається за допомогою регістрів керування зовнішніми перериваннями EXT|\_CR1 i EXTI\_CR2 (EXTERNAL INTERRUPT CONTROL REGISTER). Формат регістрів EXTI\_CR1 i EXTI\_CR2 подано на рис4.2.6.



Формат регістра EXTI\_CR1

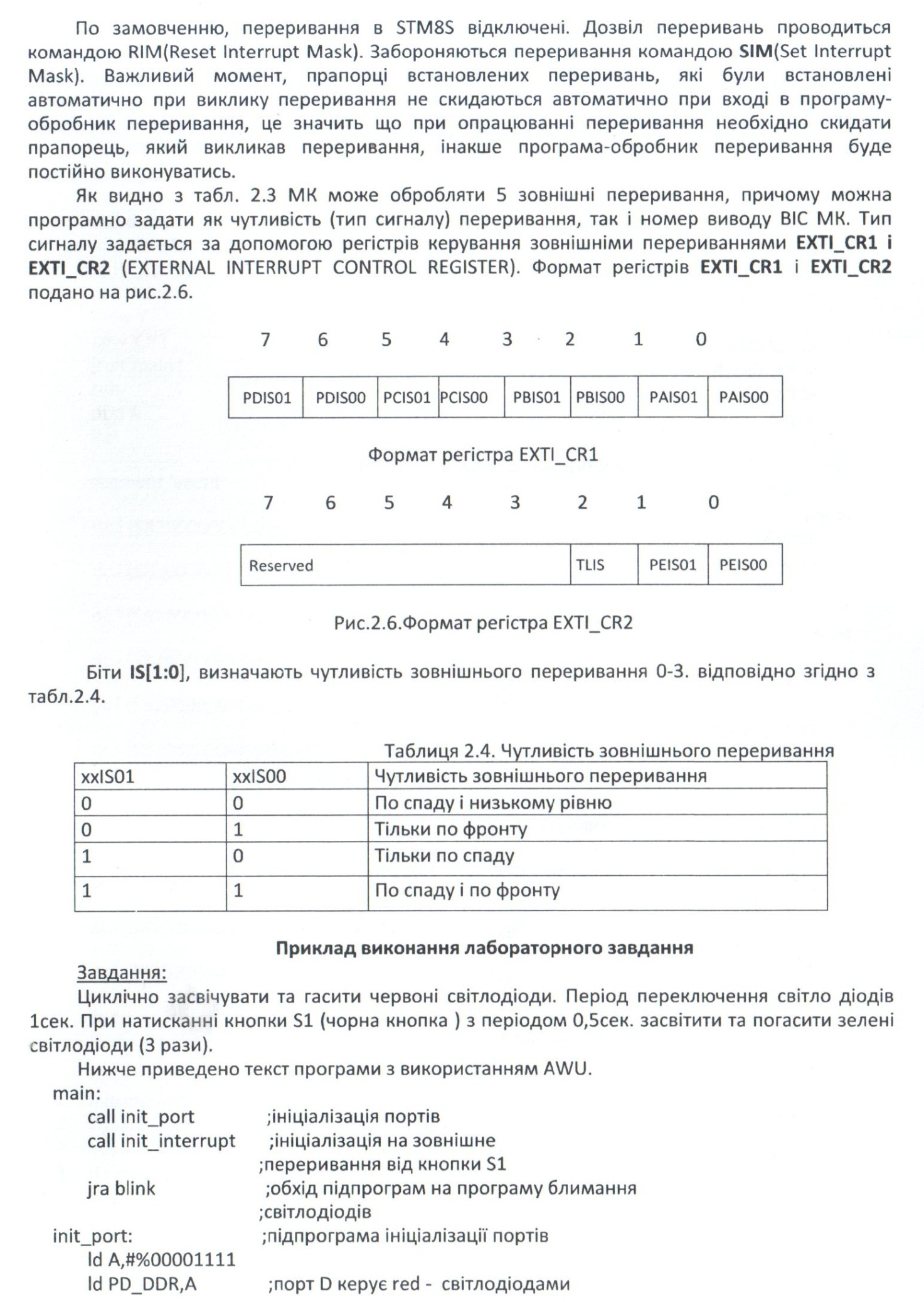


Рис.2.6.Формат регістра EXTI\_CR2

Біти ІS[1:0], визначають чутливість зовнішнього переривання 0-3. відповідно згідно з

табл.2.4.

Таблиця 2.4. Чутливість зовнішнього переривання

**Приклад виконання лабораторного завдання**

Завдання:

Циклічно засвічувати та гасити червоні світлодіоди. Період переключення світло діодів 1сек.При натисканні кнопки S1 (чорна кнопка ) з періодом 0,5сек. засвітити та погасити зелені світлодіоди (3 рази).

Нижче приведено текст програми з використанням AWU.

main:

call init\_port ;ініціалізація портів

call init\_interrupt ;ініціалізація на зовнішне

;переривання від кнопки 51

jra blink ;обхід підпрограм на програму блимання

;світлодіодів

init\_port: ;підпрограма ініціалізації портів

Id A,#%00001111

ld PD\_DDR,A ;порт D керує red - світлодіодами

Id PD\_CR1,A ;DO — D3 в режимі виводу

Id PD\_CR2,A

Id А,#%11110000

Id PC\_DDR,A ;п0рт С керує green - світдодіодами

Id PC\_CR1,A ;C4 — C7 B режимі виводу

ret

init\_interrupt: ;підпрограма ініціалізації

push A ;переривання від кнопки 51

Id A,#%OOOOOOOO ;пoрт В в режимі вводу

Id PB\_DDR,A

Id A,#%00001000

ld EXTI\_CR1,A

Id A,#%00000001

Id PB\_CR1,A

Id PB CR2,A

pop A V

ret

delay: ;підпрограма затримки на 0,5сек.

cIrw X

Іоор:

incw X

cpw X,#$ffff

jrult Ioop

ret

blink:

call delayl

Id A,PD\_ODR ;переключення світлодіодів

срІ А

Id PD\_ODR,A

jra blink

;підпрограма встановлення затримки

delay: ;від пробудження зі сну

Id A,#%00010001 ;дозвіл роботи AWU

Id AWU\_CSR,A

Id А„#%00001101 ;діапазон 1,024сек. - 2,048сек.

Id AWU\_TBR,A

Id A,,#%00100000

ld AWU\_APR,A

HALT

ret

;підпрограма-обробник переривання від

;пробудження зі сну

interrupt AWUInterrupt

AWUInterrupt:

push A

Id A,AWU\_CSR

pop A

iret

;підпрограма-обробник переривання від кнопки

interrupt EXTI1|nterrupt

EXTIlInterrupt:

push A

sim

cler

main1:

Y'PWNP

Id A,#%OOOOOOOO ;3-x кратне переключення

Id PC\_ODR,A ;зелених світлодіодів з періодом

;переключення 0,5сек.

call delay

Id А,#%11110000 ;викл. світлодіодів

Id PC\_ODR,A

call delay

incw Y

cpw Y,#3

jrult mainl

rim

pop A

iret

segment 'vectit'

dc.l {$82000000+main} ; reset

dc.l {S82000000+NonHandledlnterrupt) ; trap

dc.l {$82000000+NonHandledlnterrupt} ; irqO

ch {$82000000+AWUInterrupt} ;AWU

dc.l {$82000000+NonHandledlnterrupt) ;irq2

dc.l ($82000000+NonHandledlnterrupt} ; irq3

dc.l {$82000000+EXTI1lnterrupt} ; порт РВ

………………………

сіс.І {$82000000+NonHandledInterrupt) ; irq27

dc.l {$82000000+NonHandledInterrupt) ; irq28

dcel {$82000000+NonHandledlnterrupt) ; irq29

end

**Контрольні запитання**

1. Яка структура пам'яті STM8?
2. Назвіть джерела тактових сигналів STM8.
3. Назвіть випадки,коли STM8 входить в режим скидання STM8.
4. Ha які види сигналів зовнішнього перервання реагує STM8?
5. Як відбувається перехід на підпрограму переривання?

**Лабораторна робота №3**

**Дослідження внутрішніх таймерів та широтно-імпульсної модуляції (PWM) на прикладі світлодіодної лінійки, пристрою "2 кнопки".**

**Завдання:**

Ініціалізувати таймер. Використовуючи таймер забезпечити блимання першого червоного світлодіода із заданою частотою F y двох режимах: дискретному та плавному (використовуючи PWM – широко-імпульсну модуляцію). Кнопкою А (51 - чорна кнопка) змінювати режими блимання між червоним та зеленим світлодіодом. (табл. 3.1). Кнопкою B (S2 А червона кнопка) вмикати та вимикати блимання.

Модифікувати програму, що була розроблена в лабораторній роботі №1, організувавши необхідні затримки за допомогою таймеру,

**Порядок роботи з макетом:**

* Під'єднати до базової плати пристрій «2 кнопки».
* Під'єднати до базової плати адаптер живлення.
* Під'еднати до базової плати «Програматор ST-LINK», останній USВ-кабелем А-В з персональним комп'ютером з інтегрованим середовищем розробки SТVD.
* Процес інсталяції пакету ST Visual Developer (Версія 4.3.1) наведено в Додатку Б.

Таблиця 3.1. Завдання для лабораторної роботи 3

|  |  |  |  |
| --- | --- | --- | --- |
| № варіанту | Кнопка А | Кнопка В | Частота F, Гц |
| 1. | S1 | S2 | 1 |
| 2. | S2 | S1 | 3 |
| 3. | S1 | S2 | 5 |
| 4. | S2 | S1 | 7 |
| 5. | S1 | S2 | 10 |

Теоретичні відомості

В мікроконтролерах STM8 є три типи таймерів:

- таймер з вдосконаЛеним керуванням (Advanced control) — **TIM1**;

— загального призначения (General purpose) — **TIM2**,**TIM3**,**TIM5**;

— базові (Basic) —- **TlM4**,**TIM6**.

Таймери ТІМ2, TIM3, ТІМ5 є 16-бітними, рахують уверх, дозволяють завантажування

певного значення при переповненні і дозволяють використовувати 4-бітні дільники, коефіцієнт ділення яких вираховується за формулою 2‘, тобто, максимально крок таймера можна сповільнювати у 32768 рази, Значення дільника є чинним лише після будь-якої події оновлення таймера. Їх можна використовувати для генерування періодичних сигналів, вимірювання довжини імпульсів, та генерування імпульсів за допомогою ШИМ. Можна обробляти події переповнення, порівняння з еталоном та захоплення значення за допомогою обробників переривань.

Таймери ТІМ2, TIM3, ТІМ5 мають незалежні канали для захоплення початкового значення, порівняння із заданим значенням і формування сигналу в залежності від результату порівняння (ШИМ). ТІМ2 має 3 канали, TlM3 має 2 канали. Таймер ТІМ5 ідентичний ТІМ2, але має 2 додаткових регістри для підтримки синхронізації з іншими таймерами або із зовнішніми

сигналами у вигляді скидання, встановлення прапорців, підрахунку імпульсів. Можуть включатися в режим одноразового підрахунку (як секундомір),

Опис таймерів мікроконтролерів STMS в документації (RM0016) займає 120 сторінок. Розглянемо таймер ТІМ1, який має найбільшу кількість різних функцій і формат основних його регістрів.

Таймер ТІМ1 являє собою 16-бітний таймер з автоперезавантаженням.

Призначений для:

* Реалізаціїчасових затримок;
* Генерації переривань при переповненні, захопленні, порівнянні;
* Генераціі4 незалежних ШІМ сигналів.

Таймер заснований на автономному (несинхронізованному) 16-6ітовому інкрементному лічильнику з вхідним автоперезавантажувальним регістром і 4-ма вихідними PWM каналами.

Наявність попереднього дільника до 65535 дозволяє гнучко конфігурувати таймер, відміряти тривалі часові відрізки. Використовуючи таймер ТІМ1 в STM8 i максимальний попередній дільник 65536 при тактовій частоті 16 МГц отримуємо:

(1/16 000 000)‘65536‘ 65 535: 268,43136 с., що дорівнює майже чотири з половиною хвилини. Ще більшу часову затримку можна отримати, використовуючи Repetition counter. Це такий модуль, який рахує події таймера. Якщо його налаштувати таким чином, щоб рахувати переповнення таймера, то ми отримуємо ще один попередній дільник з максимальним

значенням 255, Варто зауважити, що Repetition counter рахує вниз від 255 по замовченню (можно вибрати любе значення менше 255) до O.

Функцію зупинки (Break) можна використовувати в пристроях керування двигуном, коли таймер формує керуючі імпульси для обмоток, а на вхід останову підключений давач температури або давач струму. Тоді при перевантаженні двигуна автоматично відбувається його відключення. Таймер ТІМ1 має чотири канали захоплення-порівняння, Що це таке і як це можна використати? в режимі захоплення, ці канали дозволяють вимірювати тривалість сигналів на вході кожного з каналів. В режимі порівняння можна реалізувати різні часові затримки, а також формувати ШІМ-сигнал.

У таймера **ТІМ1** можливий вибір генератора лічильника. Крім очевидного основного генератора частоти мікроконтролера генератором може бути зовнішній генератор, а також інші таймери. В цьому випадку таймер-джерело тактових сигналів виступає як джерело синхронізації або у вигляді додаткового попереднього дільника таймера **ТІМ1**.

Регістр лічильника **TІM1\_CNTR** — Timerl Counter таймера **ТІМ1**. Це 16-бітний регістр, в якому зберігається значення лічильника таймера **ТІМ1**, Доступ до цього регістру можливий через регістри **TlM1\_CNTRH** i **TlM1\_CNTRL** - Counter high i Counter low відповідно, доступ до них (як в режимі читання, так і в режимі запису) проводиться спочатку до старшого регістру (**TlM1\_CNTRH**), а потім - до молодшого (**TIM1\_CNTRL**). Так само відбувається доступ і до інших

16>6ітних регістрів 5ТМ8$` Регістр попереднього дільника TIM1\_PSCR — Timerl Prescaler. Як і **TIM1\_CNTR** — це 16-бітний регістр, і доступ до нього проводиться аналогічним чином, через регістри **TlM1\_PSCRH** i **TlM1\_PSCRL** Значення, **TlM1\_PSCR** визначає, на скільки ділиться вхідна частота за формулою fCK\_CNT = fCK\_PSC/ (PSCR[15:O]+1), де fCK\_CNT — частота, яка поступає на таймер, fCK\_PSC— частота, яка поступає на дільник.

Регістр керування 1 **TlM1\_CR1** - Control register 14 Формат регістра TlM1\_CR1 подано на рис.3.1.

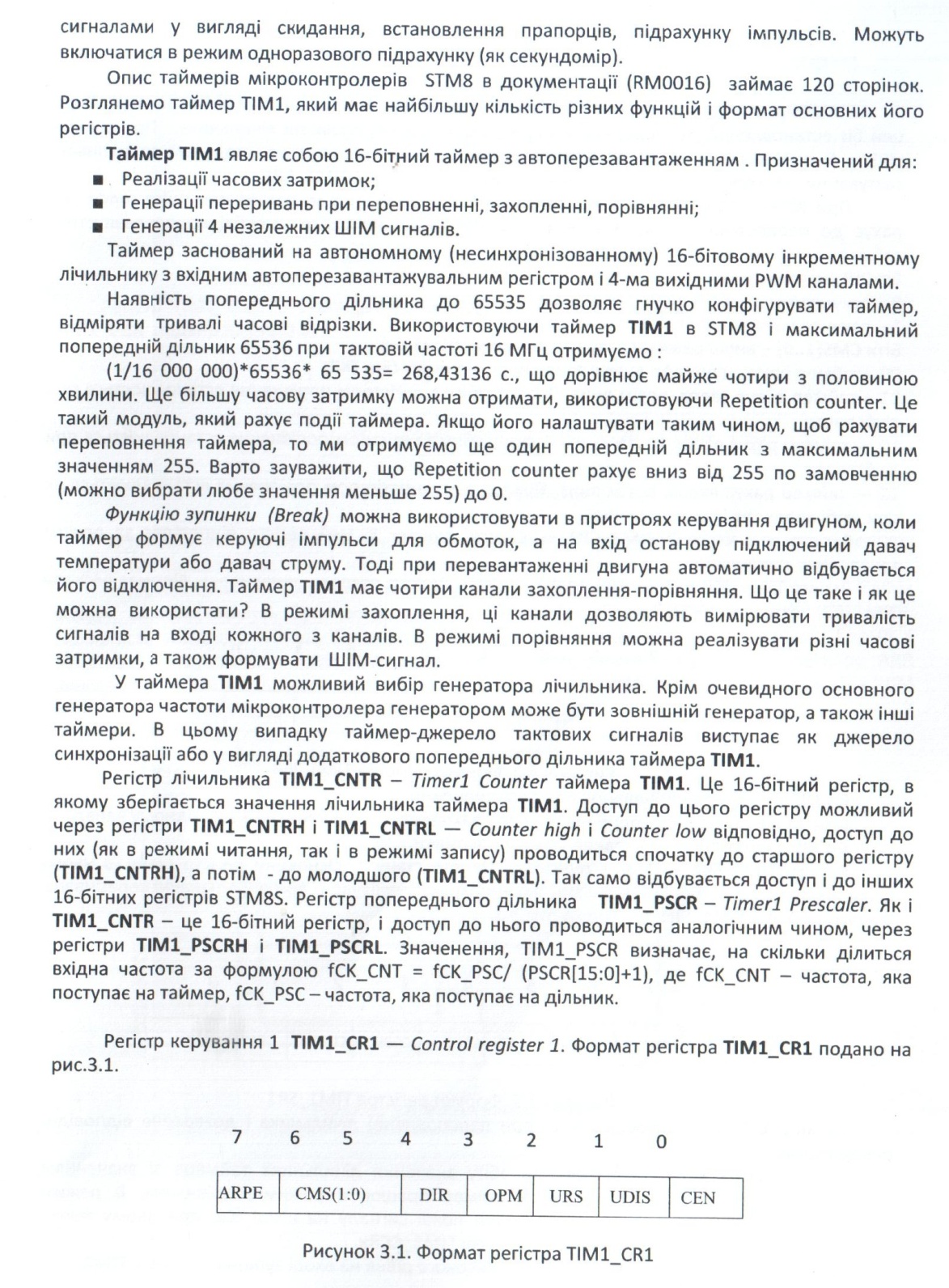


Рисунок 3.1. Формат регістра TІM1\_CR1

Біт CEN запускає таймер.

Встановлення біта UDIS забороняє переривання таймера Ті 1.

Біт URS відповідає за вибір подій, при яких виникає переривання від таймера ТІМ1. Якщо цей біт встановлений, то переривання виникають при переповнені лічильника. При значені URS = 0 переривання, крім звичайного способу, може викликатись модулем зовнішнього тактування, або програмно, встановленням біта UG в peric1pi TIM1\_EGR.

При встановленому біті 0PM таймера ТІМ1 переходить в режим one pulse mode: він рахує до переповнення і зупиняється. Для продовження роботи необхідно перезапустити таймер.

Біт DIR визначає напрямок рахунку:

0 - рахунок вверх, від 0 до 0xFFFF;

1 - рахунок вниз, від 0xFFFF до O.

Біти CMS[1..O] — вибір режиму рахунку; можливі значения:

00 - таймер рахує вверх або вниз в залежності від стану біта DIR;

01 — таймер рахує вверх, потім вниз. Прапор збігу з регістром порівняння встановлюється при

лічбі вниз;

10 - таймер рахує вверх, потім вниз. Прапор збігу з регістром порівняння встановлюється при

лічбі вверх;

11 - таймер рахує вверх, потім вниз. Прапор збігу з регістром порівняння встановлюється як при лічбі вверх такі при лічбі вниз.

Регістр керування 2 TIM1\_CR2 — Control register 2. Цей регістр відповідає за режим захоплення-порівняння.

Регістр TlM1\_lER - Interrupt enable register. Регістр дозволу переривань, Формат регістра TIM1\_IER подано на рис.3.2.

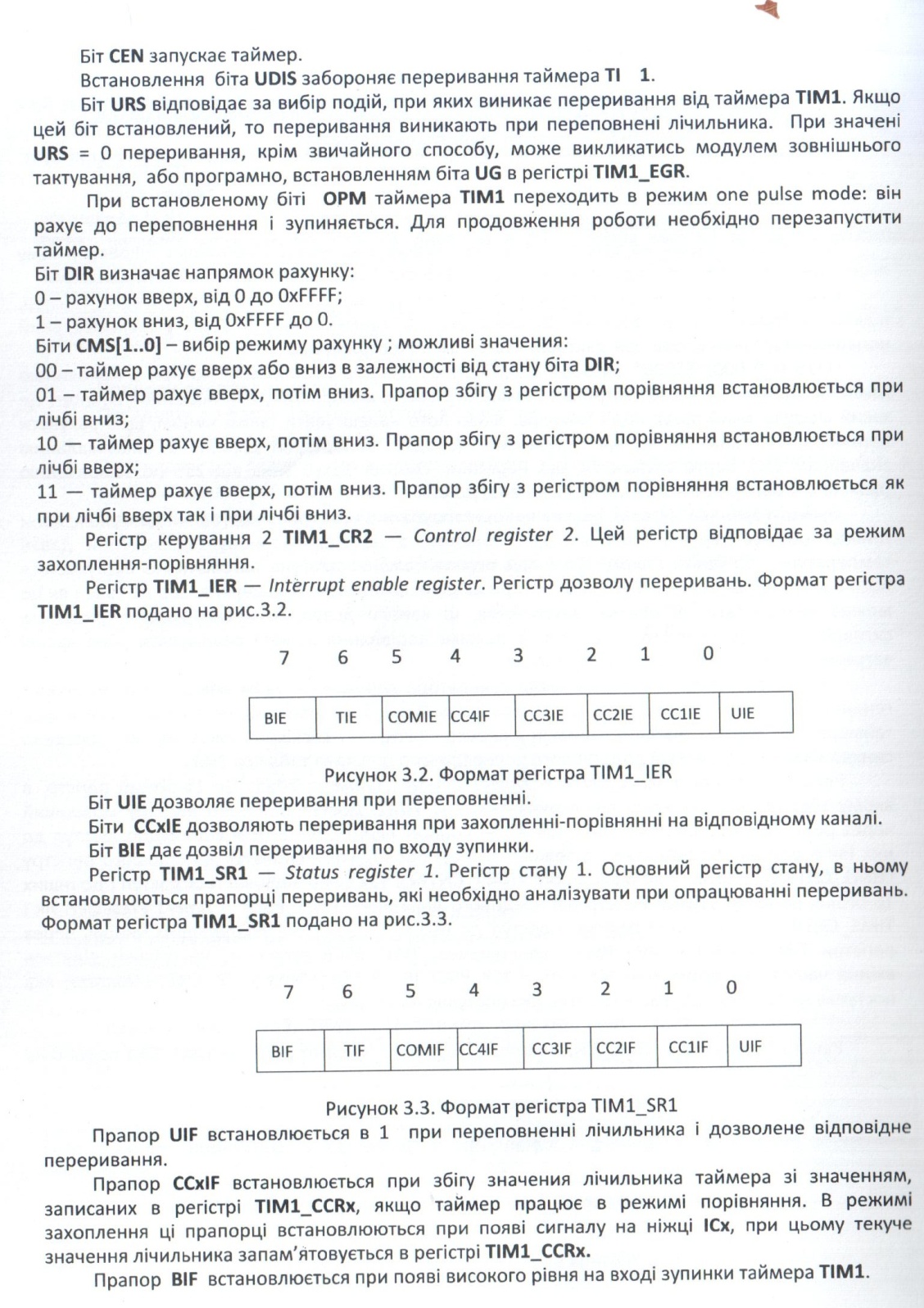


Рисунок 3.2. Формат регістра TlM1\_lER

Біт UIE дозволяє переривання при переповненні.

Біти CCxlE дозволяють переривання при захопленні-порівнянні на відповідному каналі.

Біт ВІЕ дає дозвіл переривання по входу зупинки.

Регістр TlM1\_SR1 - Status register 1. Pегістру стану 1. Основний регістр стану, в ньому встановлюються прапорці переривань, які необхідно аналізувати при опрацюванні переривань.

Формат регістра TIM1\_SR1 подано на рис.3.3.

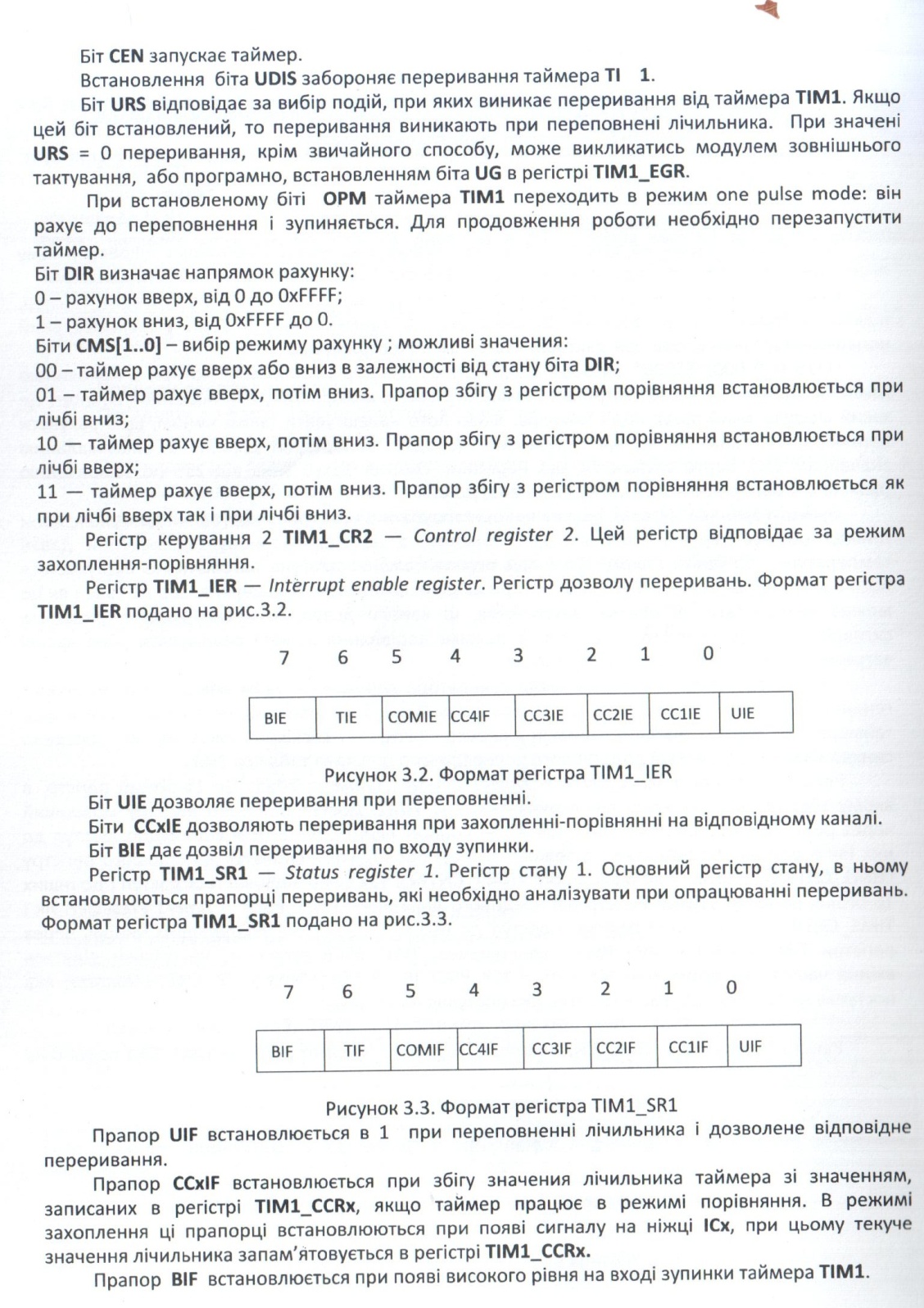


Рисунок 3.3. Формат регістра TIM1\_SR1

Прапор UIF встановлюється в 1 при переповненні лічильника і дозволена відповідне

переривання.

Прапор CCxIF встановлюється при збігу значення лічильника таймера зі значенням, записаних в регістрі TIM1\_CCRx, якщо таймер працює в режимі порівняння. В режимі захоплення ці прапорці встановлюються при появі сигналу на ніжці ІСх, при цьому текуче значення лічильника запам'ятовується в регістрі TIM1\_CCRX.

Прапор BIF встановлюється при появі високого рівня на вході зупинки таймера ТІМ1.

Регістр стану2 TIM1\_SR2 - Status register 2. Цей регістр відповідає за режим захоплення-порівняння. Формат регістра TIM1\_SR1 подано на рис.3.4.

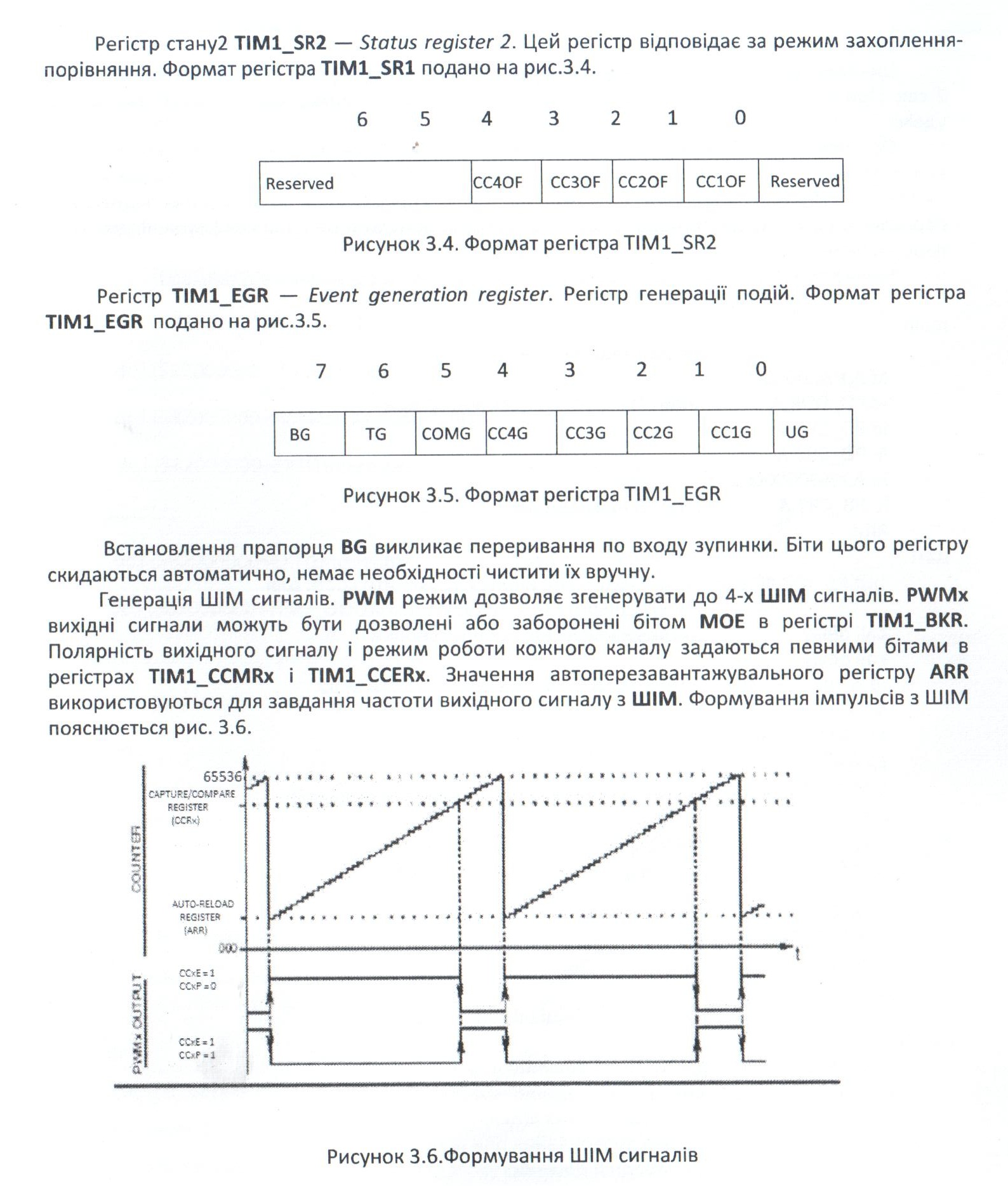


Рисунок 3.4. Формат регістра TIM1\_SR2

Регістр TlM1\_EGR - Event generation register. Регістр генерації подій. Формат регістра

TlM1\_EGR подано на рис. 3.5.

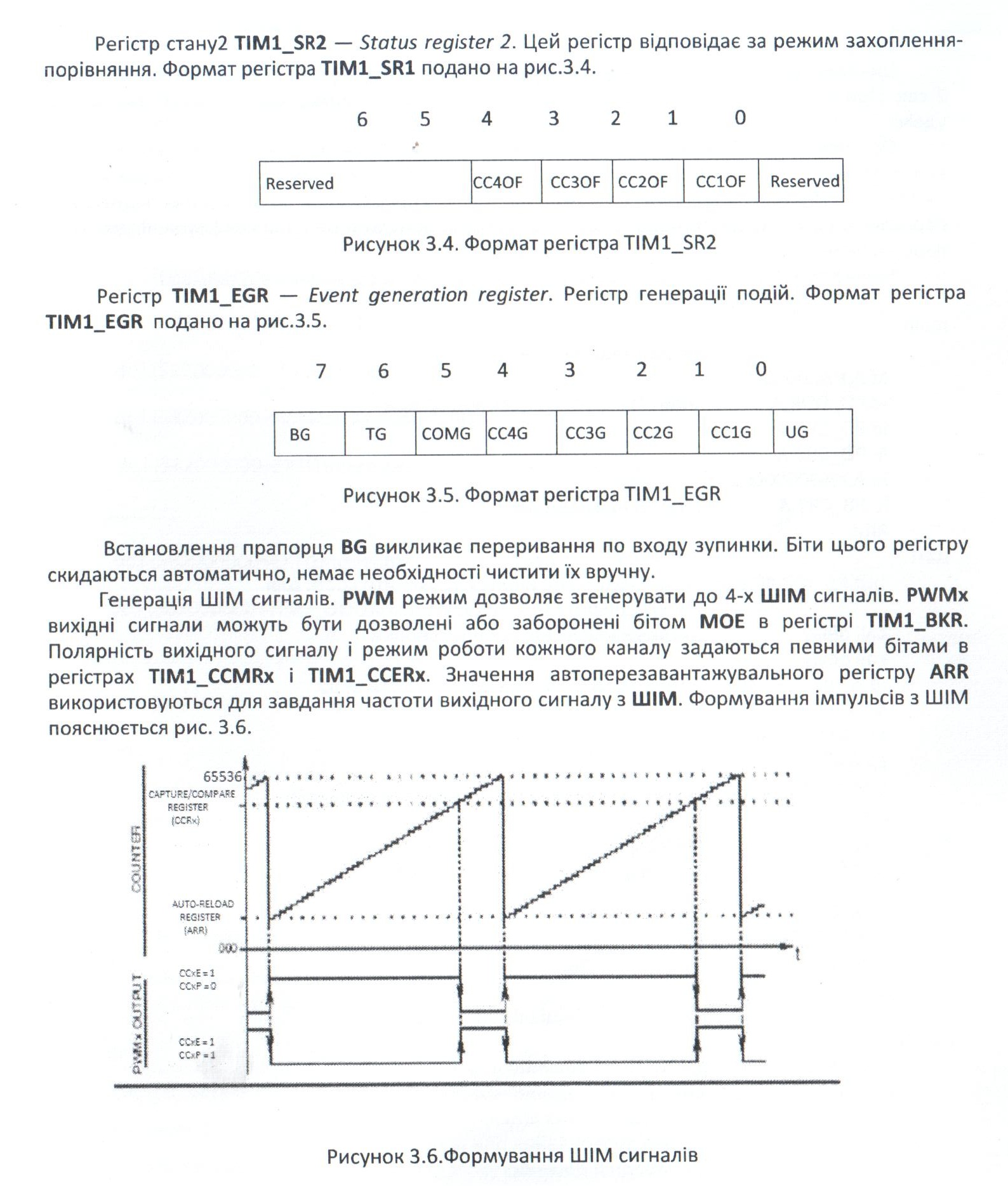


Рисунок 3.5. Формат регістра TlM1\_EGR

Встановлення прапорця BG викликає переривання по входу зупинки. Біти цього регістру скидаються автоматично, немає необхідності чистити їх вручну.

Генерація ШІМ сигналів. PWM режим дозволяє згенерувати до 4-х ШІМ сигналів. PWMx вихідні сигнали можуть бути дозволені або заборонені бітом МОЕ в регістрі TIM1\_BKR. Полярність вихідного сигналу і режим роботи кожного каналу задаються певними бітами в регістрах TIM1\_CCMRx i TIM1\_CCEth Значення автоперезавантажувального регістру ARR використовуються для завдання частоти вихідного сигналу з ШІМ. Формування імпульсів з ШІМ пояснюється рис. 3.6.

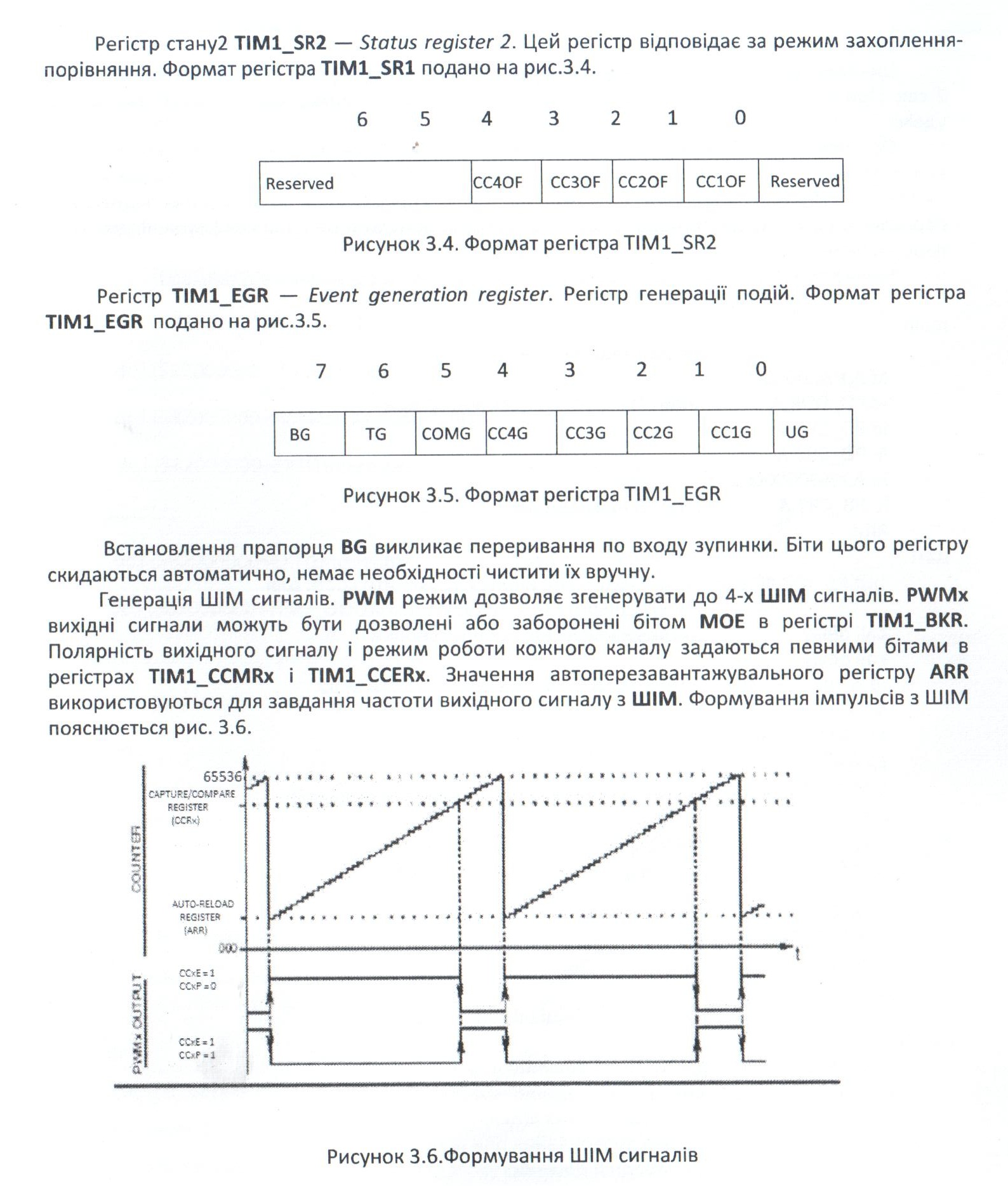


Рисунок 3.6.Формування ШІМ сигналів

**Приклад №1 виконання лабораторного завдання**

Завдання:

Циклічно засвічувати та гасити 1>й червоний світлодіод. Період переключення світло діода 2 сек, При кожному натисканні кнопки 51 (чорна кнопка ) період переключення зменшувати удвічі.

Організуємо затримку для програми керування включенням світлового діода,

використавши таймер ТІМ1 в самому простому і очевидному режимі - лічба вверх від O доOxFFFF (65535) i відповідному перериванні. Після спрацювання переривання відбувається переключення стану світлодіода, Для цього потрібно обнулити регістри конфігурації, дозволити переривання по переповненню і включити таймер.

Нижче приведено текст програми з використанням переривання від таймера.

main:

;ініціалізація портів

Id A,#%00000001

ld PD\_DDR,A ;порт D керує red - світлодіодами

Id PD\_CR1,A

Id PD\_CR2,A

Id A,#%00000001

ld PB\_CR1,A ;nopT В на введення

RIM

start:

btjt PB\_|DR,#0,start ;якщо натиснута кнопка Sl змінити коеф. для

;попереднього дільника TIM1\_PSCRL

call delay ;затримка після натискання кнопки

call delay

Id A,T|M1\_PSCRL ;зміна med). поперед. дільника таймера

srl A

Id T|M1\_PSCRL,A

cp A,#0

jreq main

jra start

delay: ;підпрограма затримки на 0,5сек.

clrw X

loop:

incw X

cpw Х,#65000

jruIt loop

ret

;підпрограма ініціалізаціїтаймера

init\_TIM1:

mov TIM1\_CR2,#0 ;синхрон. як ведучий відкл.

mov TlM1\_SMCR,#0 ;синхрон. як ведений відкл.

mov T|M1\_ETR,#0 ;зовн. тактування відкл.

mov T|M1\_|ER.~1 ;дозвіл переривання при переповненню

mov T|M1\_PSCRH,#0 ;nonepeAHiﬁ дільник

mov TlMl\_PSCRL.#$7f

Id ANS

Id T|M1\_CR1,A ;дозвіл перерив. при переповн.,запуск таймера

ret

;підпрограма-обробник переривання від

;кнопки

interrupt TІM1\_UOUTBІnterrupt

TІM1\_UOUTBІnterrupt:

push A

bcpl PD\_DDR,#0 ;переключення світло діода

mov TIM1\_CR1,#0 ;скид прапор. переривання

Id м5

Id T|M1\_CR1,A ;запуск таймера

рор A

iret

segment 'vectit‘

dc,|{$82000000+main} ; reset

ch ($82000000+NonHandledІnterrupt) ; trap

dc.|(582000000+NonHandledlnterrupt) ;irq0

dc,|($82000000+NonHandledІnterrupt} ; irq3

dc.|(SSZOOOOOOHZXTI1Interrupt) ;irg4

dc.|($82000000+EXT|1Іnterrupt) ; irg10

dc,|(S82000000+TIM1\_UOUTBІnterrupt) ;ТІМ1

dc.|($82000000+NonHandledІnterrupt) ; irq29

end

**Приклад №2 виконання лабораторного завдання**

Завдання:

За допомогою ШІМ керувати яскравістю світлодіода green №l. Яскравість збільшувати рівними кроками після кожного натискання кнопки 51 (чорна кнопка).

;головна програма

main:

var.b

sim ;маскуемо переривання

call init\_port ;ініціалізація портів

call pwm\_init ;ініціалізація таймеру, настройка PWM (UJIM)

ldA,#%00000000 ;задаємо сгаргову шпаруватість

гіт

start:

Id var,A ;налаштовуємо PWM на нову шпаруватість

call pwm

jp start ;зациклюємо програму

jp main

init\_interrupt: ;підпрограма ініціалізації

;переривання від кнопки 51

push A ;переривання тільки по спаду

Id A,#%00001000

ld EXTl\_CR1,A

ld A,#%00000001

ld PB\_CR1,A

Id PB\_CR2,A

pop A

ret

init\_port: ;підпрограма їніціалізації портів

Id А,#%00001111

Id PD\_DDR,A ;nop1 D керує red - світдодіодами

Id PD\_CR1,A ;D0— D3 в режимі виводу

Id PD\_CR2,A

Id A,#%11110000

Id PC\_DDR.A ;порт С керує green - світдодіодами

Id PC\_CR1,A ;C4 — C7 в режимі виводу

Id A,#%OOOOOOOO

ld PB\_DDR,A ;порт B нa введення

ret

init\_pwm: ;підпрограма ініціалізації PWM

mov TlM1\_ARRH,#$ff

mov TlM1\_ARRL,#$ff

Id A,#%1000OOOI

Id TlM1\_CR1,A

ld A,#%00110000

Id T|M1\_CCMR4,A

Id A,#%IOOOOOOO

ld TIM1\_BKR,A

ld A,#%00110000

ld T|M1\_CCER2,A

ret

pwm:

Id A,#%OOOOOOOU

Id TIM1\_CCR4L,A

Id A,var

Id TIM1\_CCR4H,A

ret

;підпрограма-обробник переривання від

;кнопки

interrupt EXT|1lnterrupt

EXTlllnterrupt

push A

Id A,var ;налаштовуємо PWM (LUIM) на нову

;шпаруватість

add A,#%00001111 ;збільшуємо шпаруватість на Of

Id var,A ;36epiraemo нову шпаруватість

рор А

iret

segment 'vectit'

dc.l {$82000000+main} ; reset

dc.l {$82000000+NonHandledInterrupt} ;trap

dc.l {$82000000+NonHandledlnterrupt} ;irq0

dc.l {$82000000+NonHandledlnterrupt) ;irq3

dc.l {$82000000+EXTlllnterrupt} ; порт В

…………………………

dc.l {$82000000+NonHandledlnterrupt} ; irgl0

……………………………

dc.l {$82000000+NonHandledlnterrupt) ; irq29

end

**Контрольні запитання**

1. Як залучити до роботи сторожовий таймер МК SТМ8?
2. Поясніть функцію останову ШІМ сигналів.
3. Поясніть принцип формування ШІМ сигналів
4. Назвіть призначення та склад ТІМ1 таймеру.