МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМ. ІГОРЯ СІКОРСЬКОГО»

ФАКУЛЬТЕТ ІНФОРМАТИКИ ТА ОБЧИСЛЮВАЛЬНОЇ ТЕХНІКИ

КАФЕДРА АВТОМАТИЗОВАНОГО УПРАВЛІННЯ ТЕХНІЧНИМИ СИСТЕМАМИ

Лабораторна робота № 4

Варіант 1

По дисципліні «Програмування мікроконтролерних систем»

Тема: «Вимірювання ширині імпульсів по збільшенню.»

Виконали:

студенти групи ІТ-51

Бессмертний Р.С.

Цитовцева А.С.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис, дата)

Перевірив:

ст. викладач кафедри АУТС Катін П. Ю.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ (підпис, дата)

Київ-2018

**Мета:** Користуючись результатами лабораторної роботи 2, написати програму, що здійснює вимірювання ширини імпульсів по попередньому фронту за допомогою TIM2.

**Хід роботи:**

*Варіант 1*

Користуючись результатами лабораторної роботи 2, побудувати програму для приймача і передавача, що використовує ПДП для *USART1*. Програма має передавати блоки даних розміром у 16 байт, після успішного прийому відображати інформацію про прийом на цифровому індикаторі. Початок передачі визначається натисканням кнопки на передавачі.

Написати і налагодити програми. Протестувати програми на реальному обладнанні, зробити тестування. При необхідності зробити корекцію у програмних і апаратних налаштуваннях.

#include "stm32f10x.h"

#include "stm32f10x\_gpio.h"

#include "stm32f10x\_rcc.h"

#include "stm32f10x\_tim.h"

#include "timer\_lib.h"

#include "tm1637.h"

#define STATE\_GET\_TIMER\_INPUT 1

#define STATE\_IDLE 0

#define TIMER\_NO\_INPUT 410

#define TIMER\_INPUT\_IS\_ZERO 404

#define PERIOD 2000 // probably miliseconds

volatile uint16\_t PPMBuffer = 0x0000;

volatile uint16\_t PPMValue\_Prev, PPMValue;

volatile uint16\_t finite\_state = STATE\_IDLE;

void TIM2\_IRQHandler(void){

volatile uint16\_t PPM;

if (TIM\_GetITStatus(TIM2, TIM\_IT\_CC2) != RESET)

{

/\* Reset flag \*/

TIM\_ClearITPendingBit(TIM2, TIM\_IT\_CC2);

PPMValue\_Prev = PPMValue;

PPMValue = TIM\_GetCapture2(TIM2);

PPM = (PPMValue >= PPMValue\_Prev) ? (PPMValue - PPMValue\_Prev) : (UINT16\_MAX - PPMValue\_Prev + PPMValue);

if (PPM < 10000) { // Pause

PPMBuffer = PPM;

}

else PPMBuffer = 9999;

/\* over-capture \*/

if (TIM\_GetFlagStatus(TIM2, TIM\_FLAG\_CC2OF) != RESET)

{

TIM\_ClearFlag(TIM2, TIM\_FLAG\_CC2OF);

// ...

}

}

}

int main(void)

{

SetSysClockTo72();

TIMER2\_input\_init\_A2(7200);

TIMER4\_PWM\_init\_B8(72, PERIOD);

double pulse\_width = 0.01;

TIM4->CCR3 = PERIOD \* pulse\_width;

TM1637\_init();

TM1637\_brightness(BRIGHTEST);

TM1637\_display\_all(TIMER\_NO\_INPUT);

while(1)

{

if (PPMBuffer != 0)

{

TM1637\_display\_all(PPMBuffer);

}

else

{

TM1637\_display\_all(TIMER\_INPUT\_IS\_ZERO);

}

}

}

Висновки: В даній лабораторній роботі був написаний і перевірений практично код, що здійснює вимірювання ширини імпульсів по попередньому фронту за допомогою TIM2.

https://github.com/roman-bessmertnyi/Seventh-semester/tree/master/Subjects/%D0%9F%D0%9C%D0%A1/Timer\_Capture