**Домашняя работа к занятию №3 по курсу: “** **Микроконтроллеры. Вводный курс”**

*Студент: Андрей Федоров*

***Задание:***

*Слушателям необходимо создать проект в среде разработки STM32CubeIDE с имеющейся в наличии платой Nucleo. К плате подключить семи сегментный индикатор и написать программу «Электронный кубик». В качестве кнопки для остановки счета использовать кнопку на макетной плате или внешнюю (на усмотрением слушателя).*

По стечению обстоятельств, я недавно сделал устройство для тестирования адресов i2c датчиков. И так как в нем используется динамическая индикация, я решил использовать его в качестве отладочной платы для текущего ДЗ.

Устройство имеет два индикатора, тк в ДЗ не обходимо эмитировать кубик, я буду выводить две цифры от 1 до 7 и разделю их точкой (как будто мы бросили 2 кубика).

Индикация построена на сдвиговом регистре, динамическую индикацию я осуществляю двумя выводами контроллера через транзисторы.

Схема устройства прилагается.

Вывод цифр: создаю массив из шести элементов, каждый из которых содержит отображение одной цифры (1-7). Сегменты индикаторов подключены к выводам сдвигового регистра в том порядке, который позволял наиболее удобно развести плату в домашних условиях. Так как за индикацию будет отвечать прерывание то массив задаю сразу в файле stm32f0xx\_it.h:

/\* USER CODE BEGIN PV \*/  
extern SPI\_HandleTypeDef hspi1;  
extern uint8\_t count;  
uint8\_t phase = 0;  
uint8\_t hc595\_array[6] = {  
 0B10000010,  
 0B11101100,  
 0B11100110,  
 0B10110010,  
 0B01110110,  
 0B01111110,  
 };  
  
/\* USER CODE END PV \*/

Переменная count это то, что мы будем выводить, phase - флаг выбора текущего индикатора, ну и естественно ссылка на SPI интерфейс.

Пишем вывод в прерывании:

void TIM17\_IRQHandler(void)  
{  
 /\* USER CODE BEGIN TIM17\_IRQn 0 \*/  
  
 /\* USER CODE END TIM17\_IRQn 0 \*/  
 HAL\_TIM\_IRQHandler(&htim17);  
 /\* USER CODE BEGIN TIM17\_IRQn 1 \*/  
 uint8\_t buff[2]; // опять гуляю на широкую ногу и формирую сразу обе цифры   
 buff[0] = hc595\_array[count%6]; // шестеричная система исчисления от 1 до 7 :)   
 buff[1] = hc595\_array[count/6]+1; // добавляю точку (нулевой бит)  
 HAL\_GPIO\_WritePin(D2\_GPIO\_Port, D2\_Pin, 0); // гасим оба индикатора   
 HAL\_GPIO\_WritePin(D1\_GPIO\_Port, D1\_Pin, 0); // или на видео будет видна тень соседней цифры  
 HAL\_GPIO\_WritePin(CS\_GPIO\_Port, CS\_Pin, 0); // предупреждаем о том что сейчас будем пулять данные  
 HAL\_SPI\_Transmit(&hspi1, &buff[phase], 1, HAL\_MAX\_DELAY); // пуляем, при двух индикаторах значение флага совпадает с номером элемента  
 HAL\_GPIO\_WritePin(CS\_GPIO\_Port, CS\_Pin, 1); // заканчиваем нашу передачу  
 HAL\_GPIO\_WritePin(D2\_GPIO\_Port, D2\_Pin, phase); // зажигаем нужный индикатор  
 HAL\_GPIO\_WritePin(D1\_GPIO\_Port, D1\_Pin, !phase);  
  
 phase = !phase; // переключаем флаг  
  
 /\* USER CODE END TIM17\_IRQn 1 \*/  
}

Будем показывать цифры по очереди, сначала с ускорением, а потом с замедлением, время быстрой фазы будет выбираться случайным образом, и на выходе получится два случайных числа. Изначально этот визуальный эффект был подсмотрен в курсовой работе нашего одногрупника Петра Новикова.

main.c:

/\* USER CODE BEGIN PV \*/  
uint8\_t count=0;  
/\* USER CODE END PV \*/

/\* USER CODE BEGIN 2 \*/  
HAL\_TIM\_Base\_Start\_IT(&htim17); // запускаем прерывания  
uint32\_t next\_step\_1=0;  
uint16\_t current\_delay = 200, random\_delay;  
uint8\_t mode =0;  
/\* USER CODE END 2 \*/

/\* USER CODE BEGIN WHILE \*/  
while (1)  
{  
  
if (HAL\_GetTick()>next\_step\_1){  
 if (mode){  
 next\_step\_1=HAL\_GetTick()+current\_delay;  
 count++;  
 if (count>35) count = 1;  
 } else if (!HAL\_GPIO\_ReadPin(BUTTON\_GPIO\_Port, BUTTON\_Pin)) mode=1;  
  
 switch (mode) {  
 case 1:  
 if (current\_delay>10) current\_delay\*=.95;  
 else {  
 mode=2;  
 random\_delay=HAL\_GetTick();  
 while (random\_delay>3000)  
 random\_delay=random\_delay/3000+random\_delay%3000;  
 random\_delay+=HAL\_GetTick();  
  
 }  
 break;  
 case 2:  
 if (HAL\_GetTick()>random\_delay && HAL\_GPIO\_ReadPin(BUTTON\_GPIO\_Port, BUTTON\_Pin)) mode=3;  
 break;  
 case 3:  
 if (current\_delay<200) current\_delay+=8;  
 else mode=0;  
 break;  
 }  
  
 }  
   
 /\* USER CODE END WHILE \*/