Лабораторная работа №2: Порты ввода/вывода(General-purposeinput/output). Библиотека SPL.

Цель работы: Изучить основные принципы работы GPIO. Ознакомиться с библиотекой StandartPeriphlibrary(SPL) и ее использованием.

Теоретический материал

Чтобы взаимодействовать с внешним миром – получать и передавать информацию – у микроконтроллера имеются порты ввода/вывода. У микроконтроллера STM32F051R8, используемого в лабораторных работах, 5 портов, содержащих в сумме 55 выводов.

Режимы работы выводов

Пользовательские выводы могут быть сконфигурированы в один из следующих режимов работы:

Input floating - это высокоимпедансный вход (Hi-Z), он же плавающий, так как не имеет подтяжки к питанию или земле, поэтому его логическое состояние (1 или 0) всегда определяется напряжением на нём.

Input pull-up - это вход с подтяжкой к питанию, т.е. между входом и питанием включен подтягивающий резистор (номинала порядка кОм). Подтягивающий резистор позволяет выходу находиться либо в высоком (“1”, подтянут к питанию), либо в низком (“0”, подтянут к земле) состоянии, когда к выходу не приложено внешнее напряжение. Это позволяет избежать спонтанных появлений 0 и 1 на входе.

Input-pull-down - по аналогии с предыдущим, этот вход — с подтяжкой к земле. При отсутствии напряжения имеет низкое логическое состояние (“0”).

Analog – это аналоговый вход или выход. Это касается отдельных периферийных блоков(АЦП, ЦАП).

Outputopen-drain - "выход с открытым стоком".

Outputpush-pull – “двухтактный выход”. Самый часто используемый тип выхода: подали 0 — выход подключился к земле, 1 — подключился к питанию.

Alternate function push-pull - уже описанный ранее двухтактный выход, только для альтернативной функции – периферии (SPI, USART…)

Alternate function open-drain -соответственно выход с открытым стоком, так же для альтернативной функции.

Альтернативные функции, соответствующие выводам, описаны в Datasheet на микроконтроллер (рисунок 4.1).

Pins						Pin name	Type[1]	I / O Level[2]	Main function[3] (after reset)	Alternate functions[4]	
LFBGA100	LQFP48/VFQFPN48	TFBGA64	LQFP64	LQFP100	VFQFPN36					Default	Remap
C5	41	C4	57	91	32	PB5	I/O		PB5	I2C1_SMBAI	TIM3_CH2 / SPI1_MOSI
B5	42	D3	58	92	33	PB6	I/O	FT	PB6	I2C1_SCL[8]/ TIM4_CH1[8]	USART1_TX
A5	43	C3	59	93	34	PB7	I/O	FT	PB7	I2C1_SDA[8]/ TIM4_CH2[8]	USART1_RX
D5	44	B4	60	94	35	BOOT0	I		BOOT0		
B4	45	B3	61	95	-	PB8	I/O	FT	PB8	TIM4_CH3[8]	I2C1_SCL / CANRX
A4	46	A3	62	96	-	PB9	I/O	FT	PB9	TIM4_CH4[8]	I2C1_SDA/ CANTX
D4	-	-	-	97	-	PE0	I/O	FT	PE0	TIM4_ETR	
C4	-	-	-	98	-	PE1	I/O	FT	PE1		
E5	47	D4	63	99	36	V_{SS_3}	S		V_{SS_3}		
F5	48	E4	64	100	1	V_{DD_3}	S		V_{DD_3}		

Рисунок 4.1 Альтернативные функции

Перед использованием вывода он должен быть сконфигурирован в один из вышеперечисленных режимов работы. Это можно сделать несколькими способами:

-Непосредственно через регистры микроконтроллера

-Используя библиотеку StandartPeripheralLibrary (SPL)

В данном лабораторном курсе для настройки и работы с периферией будет использоваться библиотека SPL.

StandartPeripheralLibrary – библиотека, созданная компанией STMicroelectronics. Она содержит функции и структуры для настройки и работы с периферией. Эти функции и структуры берут на себя работу с регистрами, значительно упрощая написание прошивки.

Структура программы

В качестве примера использования GPIO напишем программу, в которой с помощью микроконтроллера будет загораться светодиод после нажатия кнопки.

Все функции, классы и константы отвечающие за GPIO представлены в файлах *stm32f0xx_gpio.h* и *stm32f0xx_gpio.c*.

-Инициализация кнопки и светодиода

Нужная кнопка подключена к 4-ому выводу порта C, а светодиод – к 8-ому выводу порта A. (В файле «Распиновка» описано к каким выводам подключены кнопки, светодиоды и прочие детали макета)

За конфигурацию портов отвечает структура *GPIO_InitTypeDef*, поэтому объявим переменную такого типа:

GPIO_InitTypeDefport;

Как и для любой другой периферии сначала необходимо включить тактирование используемых портов (в данном случае Аи С):

RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOA, ENABLE);
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOC, ENABLE);

Функции для работы с тактированием представлены в файлах *stm32f0xx_rcc.h* и *stm32f0xx_rcc.c*.

Структура *GPIO_InitTypeDef*содержит переменные:

GPIO_Pin - отвечает за номер вывода порта, которую мы хотим настроить, *GPIO_Speed* – отвечает за скорость работы порта,

GPIO_Mode - отвечает за режим работы,

GPIO_PuPd – отвечает за подтяжку к земле или питанию.

Ход работы

Задание на лабораторную работу

1. Ознакомиться с теоретическими сведениями.
2.Написать, отладить и запустить программу.
3. Выполнить индивидуальное задание.

Текст программы

```
voidinitAll(void); voidinitAll()
{
GPIO_InitTypeDef port;
void GPIO_Init(GPIO_TypeDef* GPIOx, GPIO_InitTypeDef* GPIO_InitStruct); //
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOA, ENABLE);
RCC_AHBPeriphClockCmd(RCC_AHBPeriph_GPIOC, ENABLE); GPIO_StructInit(&port);
port.GPIO_PuPd = GPIO_PuPd_DOWN; port.GPIO_Mode = GPIO_Mode_IN; port.GPIO_Pin
= GPIO_Pin_4; port.GPIO_Speed = GPIO_Speed_2MHz; GPIO_Init(GPIOC, &port);
port.GPIO_Mode = GPIO_Mode_OUT; port.GPIO_OType = GPIO_OType_PP; port.GPIO_Pin
= GPIO_Pin_8; port.GPIO_Speed = GPIO_Speed_2MHz; GPIO_Init(GPIOA, &port);
}
int main()
{
uint8_t buttonState = 0;
initAll(); //инициализациявыводов while(1)
{
buttonState = GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_0); if (buttonState == 1)
//проверкасостоянияКнопки
{
GPIO_SetBits(GPIOC, GPIO_Pin_8);//включениесветодиода
}
else
{
GPIO_ResetBits(GPIOC, GPIO_Pin_8);//отключениесветодиода
}
}
}
```

Индивидуальные задания

1.Реализовать сумматор двоичных чисел

В ходе работы микроконтроллер должен считывать с переключателей два 4-х значных двоичных числа и выводить их сумму на светодиоды.

2.Реализовать декодер

Входе работы микроконтроллер должен считывать с переключателей двоичное число

–номер светодиода, который должен гореть.