```
/*
* Title: gpio.c
* Author: Noah Rowbotham
* Date: Jan. 21st, 2020
* Lab: ENEL 387-091
*/

#include "stm32f10x.h"
#include "gpio.h"

uint16_t getBits_SW(void)
{
	uint16_t switches;

	switches =  (((GPIOA->IDR & (GPIO_IDR_IDR6 | GPIO_IDR_IDR7)) >> 6) \
			| ((GPIOC->IDR & (GPIO_IDR_IDR10 | GPIO_IDR_IDR11)) >> 8))
& 0x0F;

	return(switches);
}

uint8_t getUSER_PB(void)
{
	uint8_t userPB;

	userPB = (GPIOA->IDR & (GPIO_IDR_IDR0)) & 0x0F;

	return(userPB);
}

uint16_t getBits_PB(void)
{
	uint16_t buttons;

	buttons = ((( GPIOB->IDR & ( GPIO_IDR_IDR8 )) >> 5 ) \
		  | (( GPIOB->IDR & ( GPIO_IDR_IDR9 )) >> 7 ) \
		  | (( GPIOC->IDR & ( GPIO_IDR_IDR12 )) >> 11 ) \
		  | (( GPIOA->IDR & ( GPIO_IDR_IDR5 )) >> 5 )) & 0x0F;

	return(buttons);
}

void setBits_LED(uint32_t ledStates)
{
	uint32_t output;

	output = (GPIOA->ODR & 0xFFFFE1FF); //mask odr so we clear PA9-12
	output |= (ledStates << 9 & 0x00001E00); //shift our desired output to
aline with PA9 to PA12 and mask to clear the rest
	GPIOA->ODR = output;
```

```
}

void setBlueLED(uint32_t ledState)
{
	uint32_t output;

	output = (GPIOC->ODR & 0xFFFFFEFF); //mask odr so we clear PC8
	output |= (ledState << 8 & 0x00000100); //shift our desired output to aline
with PC8 and mask to clear the rest
	GPIOC->ODR = output;
}

void setGreenLED(uint32_t ledState)
{
	uint32_t output;

	output = (GPIOC->ODR & 0xFFFFFDFF); //mask odr so we clear PC9
	output |= (ledState << 9 & 0x00000200); //shift our desired output to aline
with PC9 and mask to clear the rest
	GPIOC->ODR = output;
}

bool configGPIO_Input(char port, int pin)
{
	uint32_t CNF_MODE = 0x00000004;

	//Ensure pin is not greater than 15
	if (pin <= 15)
	{
		//If pin greater than 7 then CRH else CRL
		if (pin > 7)
		{
			int shiftAmnt = pin - 8;
			CNF_MODE = CNF_MODE << (shiftAmnt * 4);

			switch (port)
			{
				case 'A':
					GPIOA->CRH |= CNF_MODE;
					break;
				case 'B':
					GPIOB->CRH |= CNF_MODE;
					break;
				case 'C':
					GPIOC->CRH |= CNF_MODE;
					break;
				default:
					return 0;
			}
```

```
                        return 1;
                }
                else
                {
                        int shiftAmnt = pin;
                        CNF_MODE = CNF_MODE << (shiftAmnt * 4);

                        switch (port)
                        {
                                case 'A':
                                        GPIOA->CRL |= CNF_MODE;
                                        break;
                                case 'B':
                                        GPIOB->CRL |= CNF_MODE;
                                        break;
                                case 'C':
                                        GPIOC->CRL |= CNF_MODE;
                                        break;
                                default:
                                        return 0;
                        }

                        return 1;
                }
        }
        else
        {
                return 0;
        }
}

bool configGPIO_Output(char port, int pin)
{
        uint32_t CNF_MODE = 0x00000003;
        uint32_t CNF_CLEAR_MASK = 0x33333333;

        //Ensure pin is not greater than 15
        if (pin <= 15)
        {
                //If pin greater than 7 then CRH else CRL
                if (pin > 7)
                {
                        int shiftAmnt = pin - 8;
                        CNF_MODE = CNF_MODE << (shiftAmnt * 4);

                        switch (port)
                        {
                                case 'A':
                                        GPIOA->CRH &= CNF_CLEAR_MASK;
                                        GPIOA->CRH |= CNF_MODE;
```

```
                                        break;
                                case 'B':
                                        GPIOB->CRH &= CNF_CLEAR_MASK;
                                        GPIOB->CRH |= CNF_MODE;
                                        break;
                                case 'C':
                                        GPIOC->CRH &= CNF_CLEAR_MASK;
                                        GPIOC->CRH |= CNF_MODE;
                                        break;
                                default:
                                        return 0;
                        }

                        return 1;
                }
                else
                {
                        int shiftAmnt = pin;
                        CNF_MODE = CNF_MODE << (shiftAmnt * 4);

                        switch (port)
                        {
                                case 'A':
                                        GPIOA->CRL &= CNF_CLEAR_MASK;
                                        GPIOA->CRL |= CNF_MODE;
                                        break;
                                case 'B':
                                        GPIOB->CRL &= CNF_CLEAR_MASK;
                                        GPIOB->CRL |= CNF_MODE;
                                        break;
                                case 'C':
                                        GPIOC->CRL &= CNF_CLEAR_MASK;
                                        GPIOC->CRL |= CNF_MODE;
                                        break;
                                default:
                                        return 0;
                        }

                        return 1;
                }
        }
        else
        {
                return 0;
        }
}

bool configGPIO_AnalogIn(char port, int pin)
{
        uint32_t CNF_MODE = 0xFFFFFFF0;
```

```
uint32_t MASK =         0x0000000F;

//Ensure pin is not greater than 15
if (pin <= 15)
{
        //If pin greater than 7 then CRH else CRL
        if (pin > 7)
        {
                int shiftAmnt = pin - 8;
                CNF_MODE = CNF_MODE << (shiftAmnt * 4);
                MASK = MASK << ((shiftAmnt - 1) * 4);
                CNF_MODE = CNF_MODE ^ MASK;

                switch (port)
                {
                        case 'A':
                                GPIOA->CRH &= CNF_MODE;
                                break;
                        case 'B':
                                GPIOB->CRH &= CNF_MODE;
                                break;
                        case 'C':
                                GPIOC->CRH &= CNF_MODE;
                                break;
                        default:
                                return 0;
                }

                return 1;
        }
        else
        {
                int shiftAmnt = pin;
                CNF_MODE = CNF_MODE << (shiftAmnt * 4);
                MASK = MASK << ((shiftAmnt - 1) * 4);
                CNF_MODE = CNF_MODE ^ MASK;

                switch (port)
                {
                        case 'A':
                                GPIOA->CRL &= CNF_MODE;
                                break;
                        case 'B':
                                GPIOB->CRL &= CNF_MODE;
                                break;
                        case 'C':
                                GPIOC->CRL &= CNF_MODE;
                                break;
                        default:
                                return 0;
```

```
                        }

                        return 1;
                }
        }
        else
        {
                return 0;
        }
}

bool configAFIO_Output(char port, int pin)
{
        uint32_t CNF_MODE = 0x0000000B;
        uint32_t MASK = 0x00000000;

        //Ensure pin is not greater than 15
        if (pin <= 15)
        {
                //If pin greater than 7 then CRH else CRL
                if (pin > 7)
                {
                        int shiftAmnt = pin - 8;
                        MASK = ~(0x0F << (shiftAmnt * 4));
                        CNF_MODE = CNF_MODE << (shiftAmnt * 4);

                        switch (port)
                        {
                                case 'A':
                                        GPIOA->CRH &= MASK;
                                        GPIOA->CRH |= CNF_MODE;
                                        break;
                                case 'B':
                                        GPIOB->CRH &= MASK;
                                        GPIOB->CRH |= CNF_MODE;
                                        break;
                                case 'C':
                                        GPIOC->CRH &= MASK;
                                        GPIOC->CRH |= CNF_MODE;
                                        break;
                                default:
                                        return 0;
                        }

                        return 1;
                }
                else
                {
                        int shiftAmnt = pin;
                        MASK = ~(0x0F << (shiftAmnt * 4));
```

```
                        CNF_MODE = CNF_MODE << (shiftAmnt * 4);

                        switch (port)
                        {
                                case 'A':
                                        GPIOA->CRL &= MASK;
                                        GPIOA->CRL |= CNF_MODE;
                                        break;
                                case 'B':
                                        GPIOB->CRL &= MASK;
                                        GPIOB->CRL |= CNF_MODE;
                                        break;
                                case 'C':
                                        GPIOC->CRL &= MASK;
                                        GPIOC->CRL |= CNF_MODE;
                                        break;
                                default:
                                        return 0;
                        }

                        return 1;
                }
        }
        else
        {
                return 0;
        }
}
```