# PART 1

/\*----------------------------------------------------------------------------

LED C file

\*----------------------------------------------------------------------------\*/

#include "leds.h"

void init\_RGB(void){

//Start clocks for Port A, Port B and Port C

RCC->AHB1ENR |= RCC\_PORTA\_MASK | RCC\_PORTB\_MASK | RCC\_PORTC\_MASK;

//Set the pins to output mode

GPIOB->MODER &= ~MODER(RED\_LED);

GPIOA->MODER &= ~MODER(GREEN\_LED) | ~MODER(BLUE\_LED);

GPIOB->MODER |= MODER\_0(RED\_LED);

GPIOA->MODER |= MODER\_0(GREEN\_LED) | MODER\_0(BLUE\_LED);

//Set pins to push-pull output state

GPIOB->OTYPER &= ~OTYPER(RED\_LED);

GPIOA->OTYPER &= ~OTYPER(GREEN\_LED) | ~OTYPER(BLUE\_LED);

//Set pins to pull-down mode

GPIOB->PUPDR &= ~PUPDR(RED\_LED);

GPIOA->PUPDR &= ~PUPDR(GREEN\_LED) | ~PUPDR(BLUE\_LED);

GPIOB->PUPDR |= PUPDR\_1(RED\_LED);

GPIOA->PUPDR |= PUPDR\_1(GREEN\_LED) | PUPDR\_1(BLUE\_LED);

//Set pins to 50MHz

GPIOB->OSPEEDR &= ~OSPEEDR(RED\_LED);

GPIOA->OSPEEDR &= ~OSPEEDR(GREEN\_LED) | ~OSPEEDR(BLUE\_LED);

GPIOB->OSPEEDR |= OSPEEDR\_1(RED\_LED);

GPIOA->OSPEEDR |= OSPEEDR\_1(GREEN\_LED) | OSPEEDR\_1(BLUE\_LED);

//Set outputs high

GPIOB->ODR |= ODR(RED\_LED);

GPIOA->ODR |= ODR(GREEN\_LED) | ODR(BLUE\_LED);

}

//Toggle state of red LED

void toggle\_r(void){

GPIOB->ODR ^= ODR(RED\_LED);

}

//Toggle state of green LED

void toggle\_g(void){

//Write your code here

GPIOA->ODR ^= ODR(GREEN\_LED);

}

//Toggle state of blue LED

void toggle\_b(void){

//Write your code here

GPIOA->ODR ^= ODR(BLUE\_LED);

}

//Toggle state of all LEDs

void toggle\_all(void){

//Write your code here

GPIOB->ODR ^= ODR(RED\_LED);

GPIOA->ODR ^= ODR(GREEN\_LED);

GPIOA->ODR ^= ODR(BLUE\_LED);

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARM University Program Copyright (c) ARM Ltd 2016\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*----------------------------------------------------------------------------

Switches C file

\*----------------------------------------------------------------------------\*/

#include "switches.h"

void init\_switches(void){

//Start clocks for ports A and D

RCC->AHB1ENR |= RCC\_PORTA\_MASK | RCC\_PORTB\_MASK;

//Set pins to pull-up mode

GPIOA->PUPDR |= PUPDR\_0(SWITCH\_1);

GPIOB->PUPDR |= PUPDR\_0(SWITCH\_2) | PUPDR\_0(SWITCH\_3) | PUPDR\_0(SWITCH\_4);

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARM University Program Copyright (c) ARM Ltd 2016\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

/\*----------------------------------------------------------------------------

LAB EXERCISE - LOW LEVEL DIGITAL IO

----------------------------------------

Digital input: switches on a breadboard

Digital output: RGB LED on the board

Function: Use the GPIO peripheral to control LEDs

GOOD LUCK!

\*----------------------------------------------------------------------------\*/

#include "leds.h"

#include "switches.h"

//Simple software debouncing

void delay(float time){

volatile int i;

for(i=0; i<1000000\*time; i++);

}

/\*----------------------------------------------------------------------------

MAIN function

\*----------------------------------------------------------------------------\*/

int main(){

//Initialise switches and LEDs

init\_RGB();

init\_switches();

//Infinite loop

while(1){

// Button 1 => Red

if(!(GPIOA->IDR >> SWITCH\_1 & 0x1)){

toggle\_r();

delay(1);

}

// Button 2 => Green

if(!(GPIOB->IDR >> SWITCH\_2 & 0x1)){

toggle\_g();

delay(1);

}

// Button 3 => Blue

if(!(GPIOB->IDR >> SWITCH\_3 & 0x1)){

toggle\_b();

delay(1);

}

// Button 4 => All

if(!(GPIOB->IDR >> SWITCH\_4 & 0x1)){

toggle\_all();

delay(1);

}

}

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARM University Program Copyright (c) ARM Ltd 2016\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# PART 2-1

/\*----------------------------------------------------------------------------

LAB EXERCISE - DIGITAL INPUTS AND OUTPUTS

PROGRAMMING USING MBED API

----------------------------------------

In this exercise you need to use the mbed API functions to:

1) Define BusIn, BusOut interfaces for inputs and outputs

2) The RGB LED is controlled by the buttons:

+ Button 1 - light RED

+ Button 2 - light BLUE

+ Button 3 - light GREEN

+ Button 4 - light WHITE (RED, GREEN and BLUE at the same time)

GOOD LUCK!

\*----------------------------------------------------------------------------\*/

#include "mbed.h"

//Define input bus

DigitalIn button\_1(D2);

DigitalIn button\_2(D3);

DigitalIn button\_3(D4);

DigitalIn button\_4(D5);

//Define output bus for the RGB LED

DigitalOut led\_r(D6);

DigitalOut led\_g(D7);

DigitalOut led\_b(D8);

/\*----------------------------------------------------------------------------

MAIN function

\*----------------------------------------------------------------------------\*/

int main(){

while(1){

//Check which button was pressed and light up the corresponding LEDs

//Write your code here

if (!button\_1) {

led\_r = !led\_r;

}

if (!button\_2) {

led\_g = !led\_g;

}

if (!button\_3) {

led\_b = !led\_b;

}

if (!button\_4) {

led\_r = !led\_r;

led\_g = !led\_g;

led\_b = !led\_b;

}

wait(0.25);

}

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARM University Program Copyright (c) ARM Ltd 2016\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

# PART 2-2

/\*----------------------------------------------------------------------------

LAB EXERCISE - DIGITAL INPUTS AND OUTPUTS

PROGRAMMING USING MBED API

----------------------------------------

In this exercise you need to use the mbed API functions to:

1) Define BusIn, BusOut interfaces for inputs and outputs

2) The RGB LED is controlled by the buttons:

+ Button 1 - light RED

+ Button 2 - light BLUE

+ Button 3 - light GREEN

+ Button 4 - light WHITE (RED, GREEN and BLUE at the same time)

GOOD LUCK!

\*----------------------------------------------------------------------------\*/

#include "mbed.h"

//Define input bus

DigitalIn button\_1(D2);

DigitalIn button\_2(D3);

DigitalIn button\_3(D4);

DigitalIn button\_4(D5);

//Define output bus for the RGB LED

DigitalOut led\_r(D6);

DigitalOut led\_g(D7);

DigitalOut led\_b(D8);

/\*----------------------------------------------------------------------------

MAIN function

\*----------------------------------------------------------------------------\*/

int main(){

int i = 0;

while(1){

//Check which button was pressed and light up the corresponding LEDs

//Write your code here

if (!button\_1 || !button\_2 || !button\_3 || !button\_4) {

i++;

if (i == 30) // 30 => blue

led\_b = !led\_b;

else if (i == 20) // 20 => green

led\_g = !led\_g;

else if (i == 10) // 10 => red

led\_r = !led\_r;

wait(0.25);

}

}

}

// \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*ARM University Program Copyright (c) ARM Ltd 2016\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*