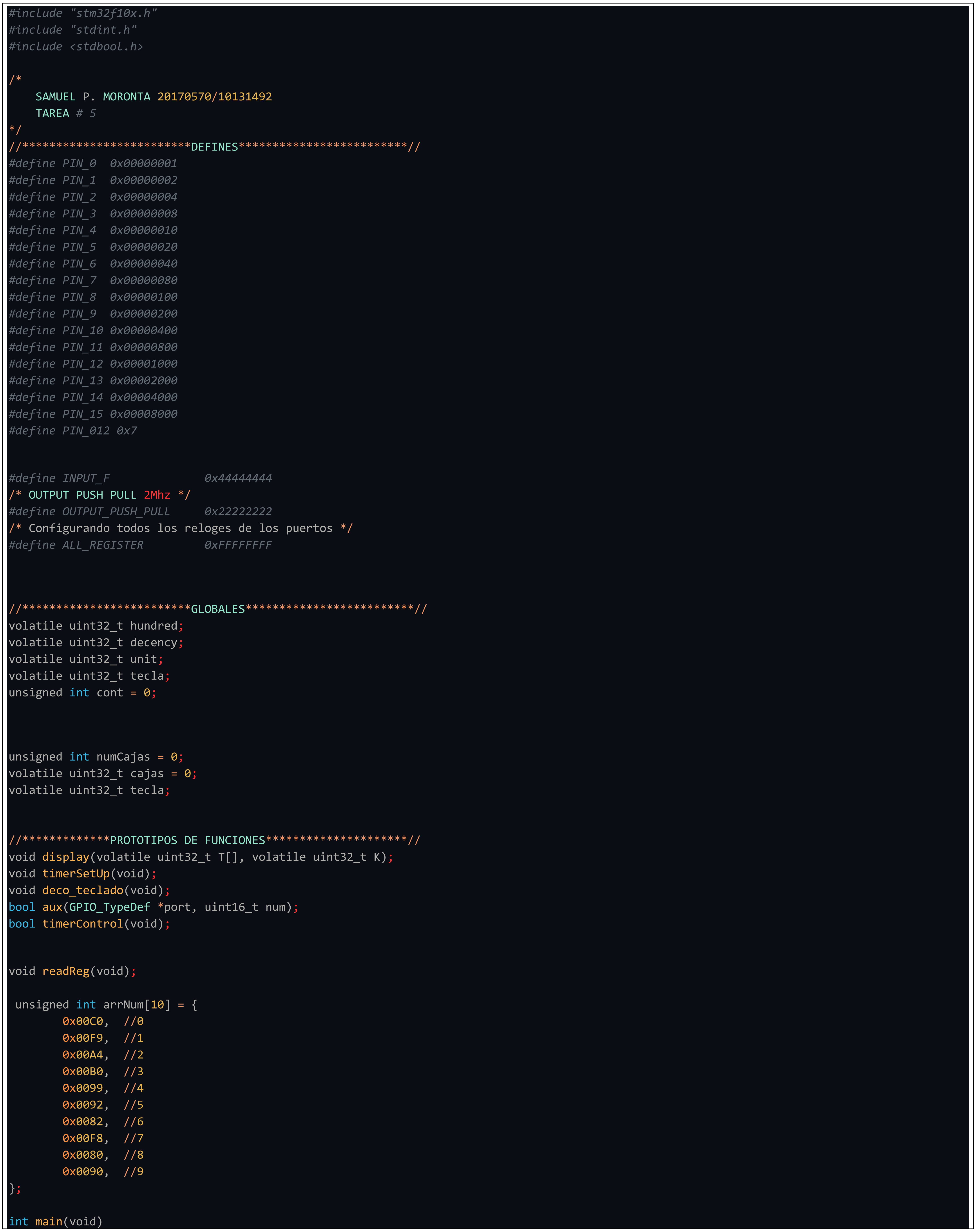
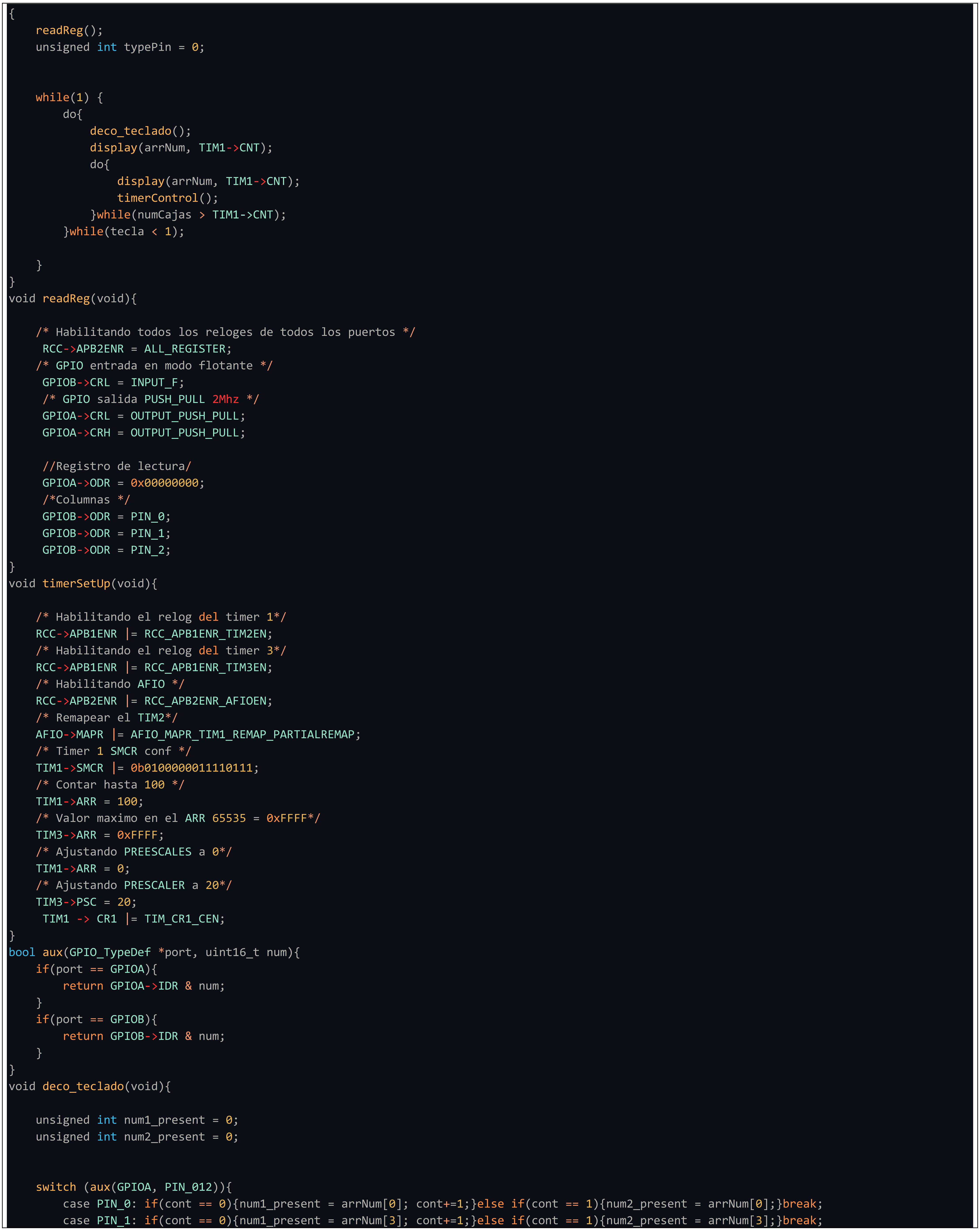
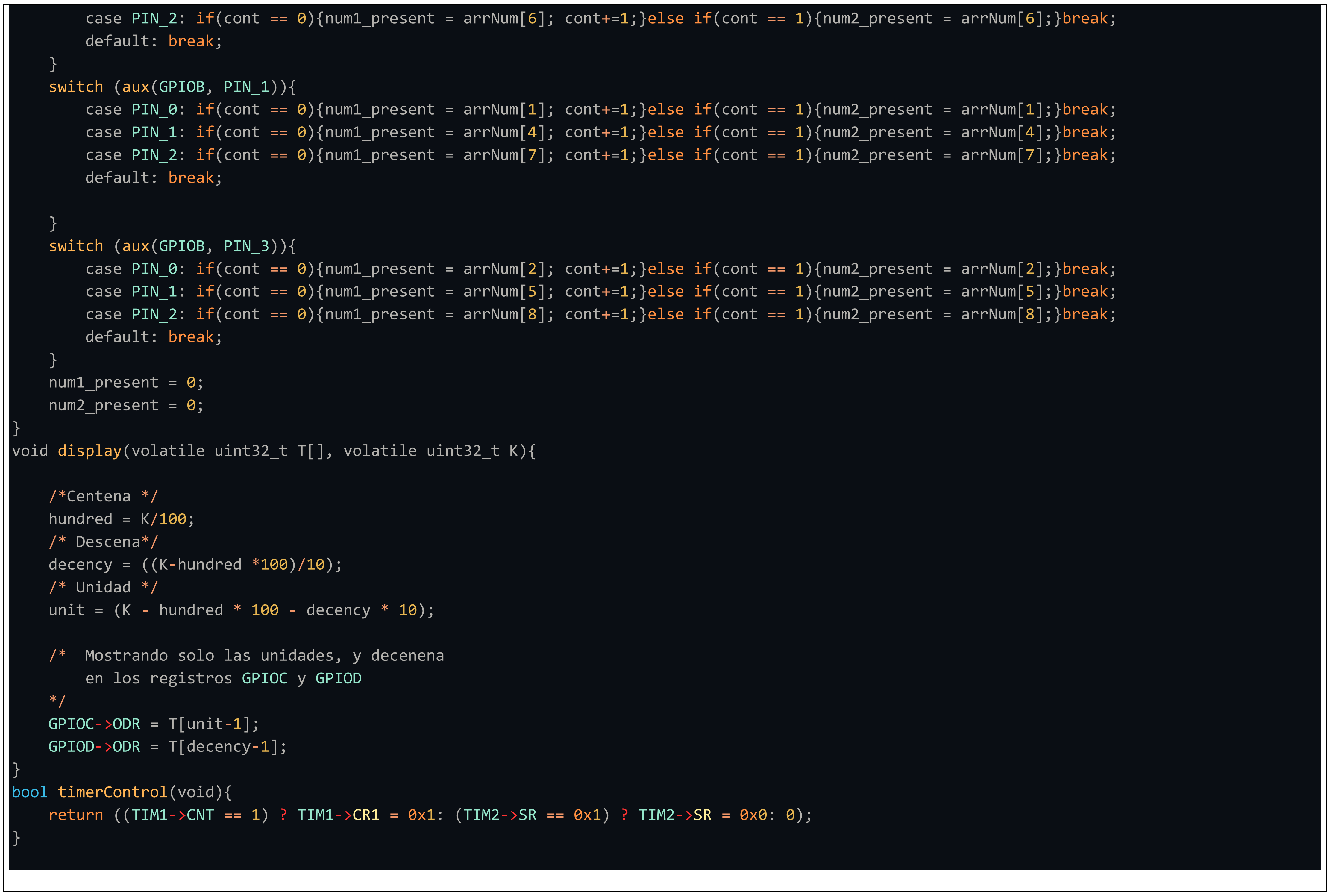
**SAMUEL PEÑA MORONTA 20170570 TAREA # 5 MICROPROCESADORES.**



```
#include "stm32f10x.h"
#include "stdint.h"
#include <stdbool.h>

/*
    SAMUEL P. MORONTA 20170570/10131492
    TAREA # 5
*/
//*************************DEFINES*************************//
#define PIN_0  0x00000001
#define PIN_1  0x00000002
#define PIN_2  0x00000004
#define PIN_3  0x00000008
#define PIN_4  0x00000010
#define PIN_5  0x00000020
#define PIN_6  0x00000040
#define PIN_7  0x00000080
#define PIN_8  0x00000100
#define PIN_9  0x00000200
#define PIN_10 0x00000400
#define PIN_11 0x00000800
#define PIN_12 0x00001000
#define PIN_13 0x00002000
#define PIN_14 0x00004000
#define PIN_15 0x00008000
#define PIN_012 0x7


#define INPUT_F              0x44444444
/* OUTPUT PUSH PULL 2Mhz */
#define OUTPUT_PUSH_PULL     0x22222222
/* Configurando todos los reloges de los puertos */
#define ALL_REGISTER         0xFFFFFFFF



//*************************GLOBALES*************************//
volatile uint32_t hundred;
volatile uint32_t decency;
volatile uint32_t unit;
volatile uint32_t tecla;
unsigned int cont = 0;



unsigned int numCajas = 0;
volatile uint32_t cajas = 0;
volatile uint32_t tecla;


//*************PROTOTIPOS DE FUNCIONES*********************//
void display(volatile uint32_t T[], volatile uint32_t K);
void timerSetUp(void);
void deco_teclado(void);
bool aux(GPIO_TypeDef *port, uint16_t num);
bool timerControl(void);


void readReg(void);

 unsigned int arrNum[10] = {
        0x00C0,  //0
        0x00F9,  //1
        0x00A4,  //2
        0x00B0,  //3
        0x0099,  //4
        0x0092,  //5
        0x0082,  //6
        0x00F8,  //7
        0x0080,  //8
        0x0090,  //9
};

int main(void)
```



```
{
    readReg();
    unsigned int typePin = 0;


    while(1) {
        do{
            deco_teclado();
            display(arrNum, TIM1->CNT);
            do{
                display(arrNum, TIM1->CNT);
                timerControl();
            }while(numCajas > TIM1->CNT);
        }while(tecla < 1);

    }
}
void readReg(void){

    /* Habilitando todos los reloges de todos los puertos */
     RCC->APB2ENR = ALL_REGISTER;
    /* GPIO entrada en modo flotante */
     GPIOB->CRL = INPUT_F;
     /* GPIO salida PUSH_PULL 2Mhz */
     GPIOA->CRL = OUTPUT_PUSH_PULL;
     GPIOA->CRH = OUTPUT_PUSH_PULL;

     //Registro de lectura/
     GPIOA->ODR = 0x00000000;
     /*Columnas */
     GPIOB->ODR = PIN_0;
     GPIOB->ODR = PIN_1;
     GPIOB->ODR = PIN_2;
}
void timerSetUp(void){

    /* Habilitando el relog del timer 1*/
    RCC->APB1ENR |= RCC_APB1ENR_TIM2EN;
    /* Habilitando el relog del timer 3*/
    RCC->APB1ENR |= RCC_APB1ENR_TIM3EN;
    /* Habilitando AFIO */
    RCC->APB2ENR |= RCC_APB2ENR_AFIOEN;
    /* Remapear el TIM2*/
    AFIO->MAPR |= AFIO_MAPR_TIM1_REMAP_PARTIALREMAP;
    /* Timer 1 SMCR conf */
    TIM1->SMCR |= 0b0100000011110111;
    /* Contar hasta 100 */
    TIM1->ARR = 100;
    /* Valor maximo en el ARR 65535 = 0xFFFF*/
    TIM3->ARR = 0xFFFF;
    /* Ajustando PREESCALES a 0*/
    TIM1->ARR = 0;
    /* Ajustando PRESCALER a 20*/
    TIM3->PSC = 20;
     TIM1 -> CR1 |= TIM_CR1_CEN;
}
bool aux(GPIO_TypeDef *port, uint16_t num){
    if(port == GPIOA){
        return GPIOA->IDR & num;
    }
    if(port == GPIOB){
        return GPIOB->IDR & num;
    }
}
void deco_teclado(void){

    unsigned int num1_present = 0;
    unsigned int num2_present = 0;


    switch (aux(GPIOA, PIN_012)){
        case PIN_0: if(cont == 0){num1_present = arrNum[0]; cont+=1;}else if(cont == 1){num2_present = arrNum[0];}break;
        case PIN_1: if(cont == 0){num1_present = arrNum[3]; cont+=1;}else if(cont == 1){num2_present = arrNum[3];}break;
```



```
        case PIN_2: if(cont == 0){num1_present = arrNum[6]; cont+=1;}else if(cont == 1){num2_present = arrNum[6];}break;
        default: break;
    }
    switch (aux(GPIOB, PIN_1)){
        case PIN_0: if(cont == 0){num1_present = arrNum[1]; cont+=1;}else if(cont == 1){num2_present = arrNum[1];}break;
        case PIN_1: if(cont == 0){num1_present = arrNum[4]; cont+=1;}else if(cont == 1){num2_present = arrNum[4];}break;
        case PIN_2: if(cont == 0){num1_present = arrNum[7]; cont+=1;}else if(cont == 1){num2_present = arrNum[7];}break;
        default: break;

    }
    switch (aux(GPIOB, PIN_3)){
        case PIN_0: if(cont == 0){num1_present = arrNum[2]; cont+=1;}else if(cont == 1){num2_present = arrNum[2];}break;
        case PIN_1: if(cont == 0){num1_present = arrNum[5]; cont+=1;}else if(cont == 1){num2_present = arrNum[5];}break;
        case PIN_2: if(cont == 0){num1_present = arrNum[8]; cont+=1;}else if(cont == 1){num2_present = arrNum[8];}break;
        default: break;
    }
    num1_present = 0;
    num2_present = 0;
}
void display(volatile uint32_t T[], volatile uint32_t K){

    /*Centena */
    hundred = K/100;
    /* Descena*/
    decency = ((K-hundred *100)/10);
    /* Unidad */
    unit = (K - hundred * 100 - decency * 10);

    /*  Mostrando solo las unidades, y decenena
        en los registros GPIOC y GPIOD
    */
    GPIOC->ODR = T[unit-1];
    GPIOD->ODR = T[decency-1];
}
bool timerControl(void){
    return ((TIM1->CNT == 1) ? TIM1->CR1 = 0x1: (TIM2->SR == 0x1) ? TIM2->SR = 0x0: 0);
}
```