CONESTOGA

Connect Life and Learning

HARDWARE SOFTWARE INTERFACING (CNTR8005)

Spring term 2020

Date: 20th August 2020

AUTOMATIC SLIDING DOOR

Submitted by:

Abhisha Bhesaniya

Simran Padaniya

Contents

Introduction

Automatic sliding doors are part of every building today. It reduces human efforts to open and close door manually.

We can control sliding door using following methods:

1. Remote
2. Push Button

For this project, we are using push button to operate door.

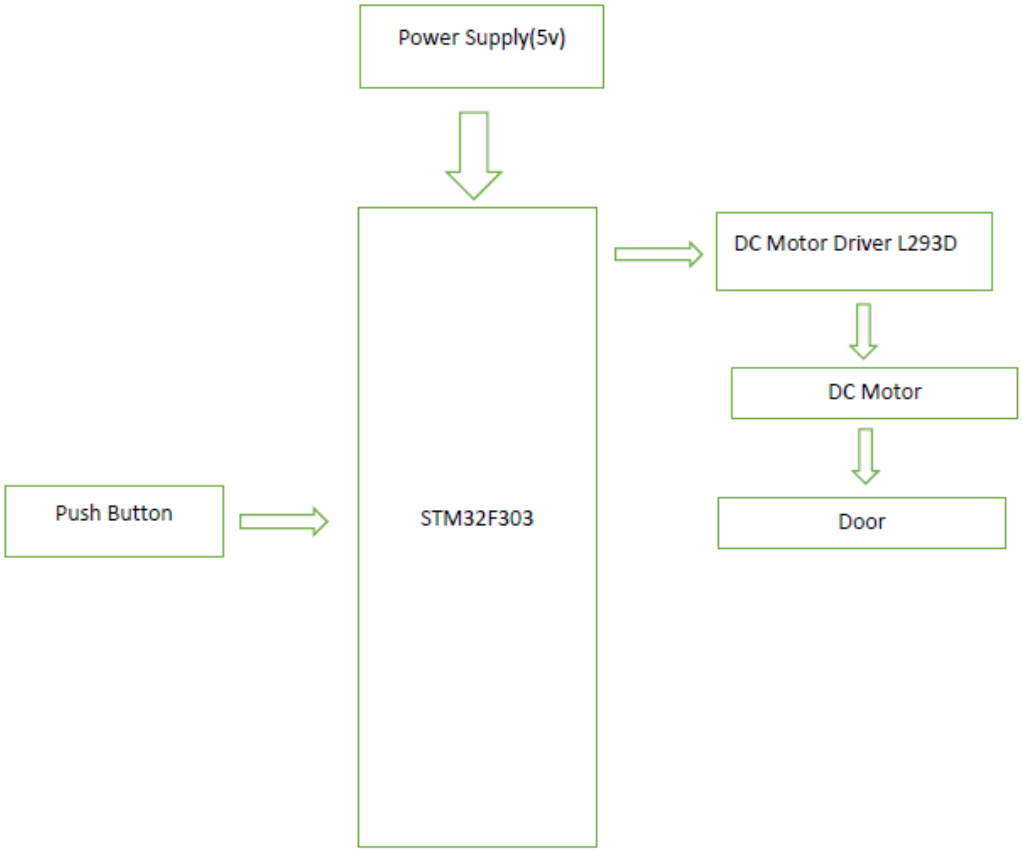

Figure 1. Basic Block Diagram

From the above block diagram, following are main components in projects:

1. Push Button
2. STM32f303 and
3. DC motor

Additional features:

Led is interfaced for the user to know the status of door.

1. Blue – door is opening.
2. Green – door is opened.
3. Red – door is closing and
4. White – door is closed.

Hardware implementation

For real life hardware implementation, user must use component with high values. But for prototyping we are using components with following specifications.

Table 1. Motor Specifications.

Input Voltage	6-12 volts
Minimum Speed	90 RPM
Maximum Speed	175 RPM

Components used in prototype:

1. DC Motor
2. Motor driver
3. RGB led
4. Breadboard
5. Push button
6. STM32f303 Board and
7. Resistors.

Table 2. PIN connection Table

Component pin	Stm32 Pin number
RGB Led	
Red	PB10
Ground	GND
Green	PB9
Blue	PB8
Push Button	
PIN 1 and PIN 3	GND
PIN 2 and PIN 4	PA12
DC Motor	PC0
	PC1

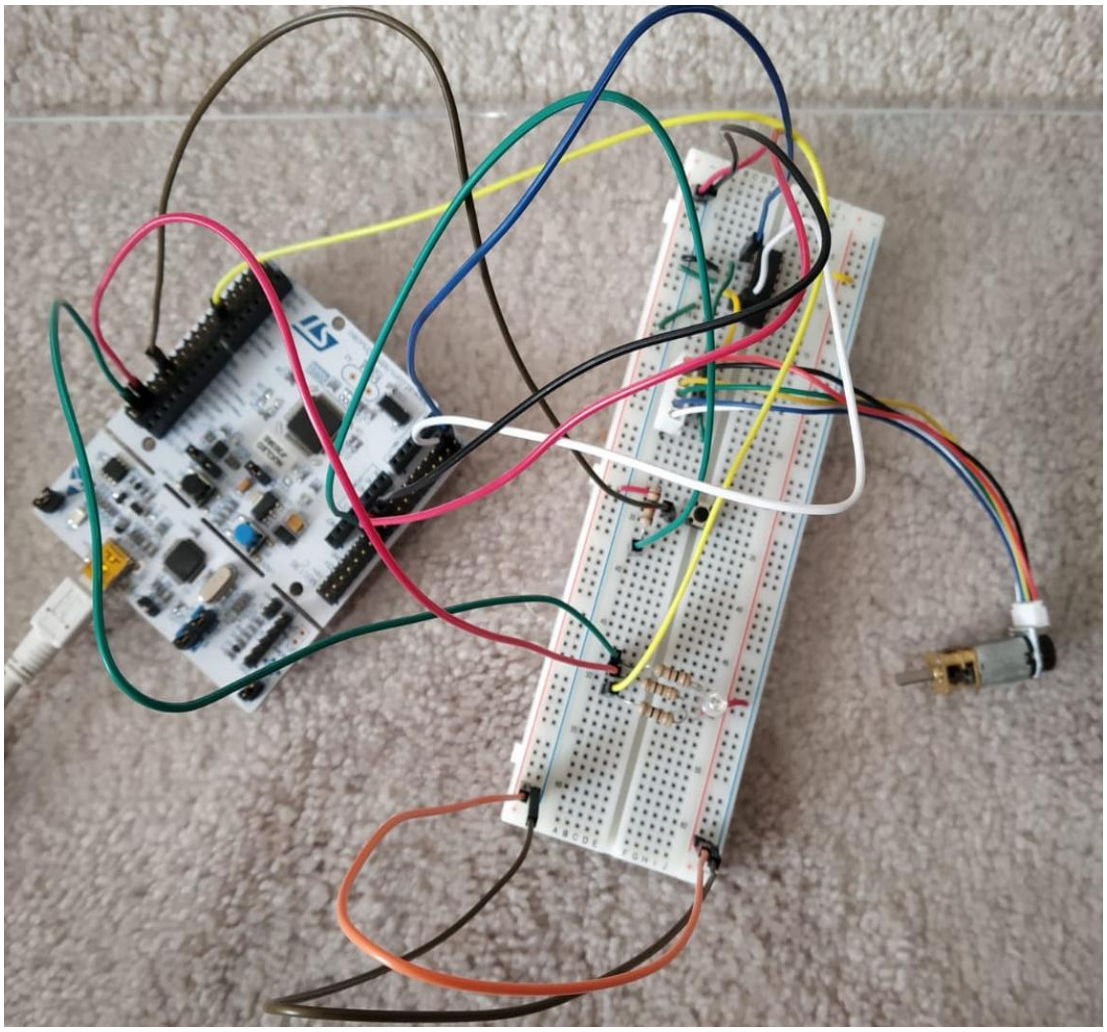

Figure 2 Hardware implementation.

Code

```c
/******************************************************************************
 * File Name          : Automatic_Sliding_Door.c
 * Description        : this program turn on PWM in Timer1 to controol the speed
 *                      of dc motor. this DC motor with some mechanical arrangemant
 *                      will help door to slide.

 * Author             : Abhisha Bhesaniya & Simran Padaniya
 * Date               : 15th August, 2020
 ******************************************************************************
 */

/* Includes----- */
#include <stdio.h>
#include <stdint.h>
#include <ctype.h>
#include <string.h>
#include "common.h"
#include "main.h"

TIM_HandleTypeDef tim1; // to run the DC motor
TIM_HandleTypeDef tim17; // to provide delay

int16_t speed;
int32_t delay;
int16_t PB = 1;

//Function declaration
void gpio_init_RGB(); // Gpio Init for RGB LED and Enable pin of ic
void gpio_init_Switch();// gpio for switch
void gpio_init_DCMotor(); //PWM init for DC motor
void timer_init_DCMotor(uint16_t); //Timer init for DC motor
void timer_init_Delay();// timer init for delay function
void start_DCMotor(uint16_t,uint16_t); // to start PWM
void stop_DCMotor(uint16_t); // to stop PWM
void RGB(int red, int green, int blue); //to contol RGB LED
void Delay(int32_t delay); // delay function

//timer 6 peripheral register declaration
uint32_t *ptrcr1 = (uint32_t *) 0x40001000 ;//pointer to control register 1
uint32_t *ptrdier = (uint32_t *) 0x4000100c ;//pointer to DIER register
uint32_t *ptrsr = (uint32_t *) 0x40001010 ; //pointer to STATUS REGISTER
uint32_t *ptrpre = (uint32_t *) 0x40001028 ;//pointer to PRESCALER
uint32_t *ptrarr = (uint32_t *) 0x4000102c ;//pointer to AUTO RELOAD REGISTER
```

Figure 3. functions declarations

```
45 /**************************************************************************
46  * Function Name      : timer_init
47  * Description        : This function will  initialize the timer 6 parameter.
48  *
49  * PARAMETERS         : N/A
50  *
51  * RETURNS            : N/A
52  * Author             : Abhisha Bhesaniya & Simran Padaniya
53  * Date               : 19th August, 2020
54  **************************************************************************
55  */
56 void timer_init()
57 {
58 __HAL_RCC_TIM6_CLK_ENABLE();//enable clock to timer 6
59 *ptrdier= 0x01;//enable interupt
60 *ptrarr=0xffff;//initialize ARR register
61 *ptrpre=0x1000;//initialize PRESCALER
62 HAL_NVIC_EnableIRQ(54);//enable interupt in NVIC Register for timer 6
63 }
64
65 /**************************************************************************
66  * Function Name      : CmddcInit
67  * Description        : This function will call functions to initialize gpio and timer.
68  *
69  * PARAMETERS         : int mode
70  *
71  * RETURNS            : CmdReturnOk
72  * Author             : Abhisha Bhesaniya & Simran Padaniya
73  * Date               : 1st August, 2020
74  **************************************************************************
75  */
76 ParserReturnVal_t CmdDisable(int mode)
77 {
78
79        if (mode != CMD_INTERACTIVE)
80             return CmdReturnOk;
81
82       WDTFeed();
83       RGB(0,0,0);//turn off the RGB LED
84       stop_DCMotor(0);//stop motion of DCMmotor in forward direction
85       stop_DCMotor(4);//stop motion of DCMmotor in reverse direction
86
87       WDTFeed();
88       *ptrsr=0x0;
89        return CmdReturnOk;
90 }
91 ADD_CMD("Disable" CmdDisable "   Disable the entire system")
```

Figure 4. Timer initialization

```
65 /**************************************************************************
66  * Function Name      : CmddcInit
67  * Description        : This function will call functions to initialize gpio and timer.
68  *
69  * PARAMETERS         : int mode
70  *
71  * RETURNS            : CmdReturnOk
72  * Author             : Abhisha Bhesaniya & Simran Padaniya
73  * Date               : 1st August, 2020
74  **************************************************************************
75  */
76 ParserReturnVal_t CmdDisable(int mode)
77 {
78
79        if (mode != CMD_INTERACTIVE)
80             return CmdReturnOk;
81
82       WDTFeed();
83       RGB(0,0,0);//turn off the RGB LED
84       stop_DCMotor(0);//stop motion of DCMmotor in forward direction
85       stop_DCMotor(4);//stop motion of DCMmotor in reverse direction
86
87       WDTFeed();
88       *ptrsr=0x0;
89        return CmdReturnOk;
90 }
91 ADD_CMD("Disable" CmdDisable "   Disable the entire system")
```

Figure 5. Command for dc motor initialization

```
/*****************************************************************************
 * Function Name      : CmdEnable
 * Description        : This function will scane two argument delay and speed from user.
 *                      According to enterd value the motor will change it's speed and delay
 *                      and continuasly chen at the push button.
 *
 * PARAMETERS         : int mode
 *
 * RETURNS            : CmdReturnOk
 * Author             : Abhisha Bhesaniya & Simran Padaniya
 * Date               : 1st August, 2020
 *****************************************************************************
 */
ParserReturnVal_t CmdEnable(int mode)
{
               //variable initialization

               HAL_StatusTypeDef  rc;


               if (mode != CMD_INTERACTIVE)
                       return CmdReturnOk;

                rc = fetch_int16_arg(&speed);// scan the value of spped
                if(rc != HAL_OK)
                {
                   printf("please enter speed in percantage !!!\n");//print the message
                   return CmdReturnOk;
                }

                rc = fetch_int32_arg(&delay);// scan the value of delay
                if(rc != HAL_OK)
                {
                   printf("please enter delay !!!\n");// print the message
                   return CmdReturnOk;
                }

               speed = (speed * 32767)/100; //convert prtcentage to value
               speed = 32768 - speed;

                gpio_init_Switch();   //init gpio for switch
                gpio_init_RGB();      //init gpio for RGB LED
                gpio_init_DCMotor();  //init gpio for DC Motor
                timer_init_Delay();   //init timer17 for delay
                timer_init_DCMotor(0);//init timer1 chennal 1 for DC Motor
                timer_init_DCMotor(4);//init timer1 channel 2 for DC Motor
                timer_init();
                *ptrcr1=0x1;


        return CmdReturnOk;
}

ADD_CMD("Enable",CmdEnable,"<speed> <delay>         Enable the System")

```

Figure 6. Command for enabling dc motor

```
/*****************************************************************************
 * Function Name      : TIM6_DAC_IRQHandler
 * Description        : This  handler is for timer 6 this will execute at every 100ms.
 *                      this will turn on and off motor according to the status of GPIO pin
 *                      and speed and delay will be provided by user.
 *
 * PARAMETERS         : int mode
 *
 * RETURNS            : CmdReturnOk
 * Author             : Abhisha Bhesaniya & Simran Padaniya
 * Date               : 1st August, 2020
 *****************************************************************************
 */
void TIM6_DAC_IRQHandler(void)
{
                  *ptrsr=0x0;

                 PB = HAL_GPIO_ReadPin(GPIOA,GPIO_PIN_12); // read GPIO PIN 12 of Port B where the push button is connected in pull up mode

                 WDTFeed();// reset watch dog timer

                      if(PB == 0)
                      {

                             /* run Motor In Forward Direction for 10 sec */
                             RGB(0,1,1); //turn on the RGB LED with sky color
                             stop_DCMotor(0);//stop motion of DCMmotor in forward direction
                             stop_DCMotor(4);//stop motion of DCMmotor in reverse direction
                            // HAL_GPIO_WritePin(GPIOB,GPIO_PIN_14,1); // Send 1 signal on enable pin of IC
                             start_DCMotor(0,speed); //start motion of DCMmotor in forward direction

                             Delay(delay);

                             /* stop motor for 10 sec */
                             RGB(0,1,0); //turn on the RGB LED with green color
                             stop_DCMotor(0);//stop motion of DCMmotor in forward direction
                             stop_DCMotor(4);//stop motion of DCMmotor in reverse direction
                             HAL_GPIO_WritePin(GPIOB,GPIO_PIN_14,0);// Send 0 signal on enable pin of IC

                             Delay(delay);

                             /* run Motor In reverse Direction for 10 sec */
                             RGB(1,0,0); //turn on the RGB LED with red color
                             stop_DCMotor(0);//stop motion of DCMmotor in forward direction
                             stop_DCMotor(4);//stop motion of DCMmotor in reverse direction
                             //HAL_GPIO_WritePin(GPIOB,GPIO_PIN_14,1);// Send 1 signal on enable pin of IC
                             start_DCMotor(4,speed); //stop motion of DCMmotor in reverse direction

                             Delay(delay);

                             /* stop motor */
                             RGB(0,0,0); //turn off the RGB LED
                             stop_DCMotor(0);//stop motion of DCMmotor in forward direction
                             stop_DCMotor(4);//stop motion of DCMmotor in reverse direction
                            // HAL_GPIO_WritePin(GPIOB,GPIO_PIN_14,0);// Send 0 signal on enable pin of IC
                      }

                      else
                      {
                             RGB(0,0,0); //turn off the RGB LED
                             stop_DCMotor(0);//stop motion of DCMmotor in forward direction
                             stop_DCMotor(4);//stop motion of DCMmotor in reverse direction
                             //HAL_GPIO_WritePin(GPIOB,GPIO_PIN_14,0); // Send 0 signal on enable pin of IC
                      }


                      WDTFeed();//reset watch dog timer

```

Figure 7. Timer 6 initialization.

```
217 /*------------------------------------------------------------------------------
218  * Function Name      : timer_init_Delay
219  * Description        : This function will Initialise the timer 17.
220  *
221  * PARAMETERS         : N/A
222  *
223  * RETURNS            : N/A
224  * Author             : Abhisha Bhesaniya & Simran Padaniya
225  * Date               : 14th August, 2020
226  ******************************************************************************
227  */
228 void timer_init_Delay()
229 {
230     __HAL_RCC_TIM17_CLK_ENABLE(); //enable clock for Timer 17
231 
232         tim17.Instance = TIM17;
233         tim17.Init.Prescaler     =     (HAL_RCC_GetPCLK2Freq() / 1000000) - 1; //set prescaler
234         tim17.Init.CounterMode   =     TIM_COUNTERMODE_UP; //set counter mode to up
235         tim17.Init.Period        =     0xffff; // set period
236         tim17.Init.ClockDivision =     TIM_CLOCKDIVISION_DIV1;
237         tim17.Init.RepetitionCounter = 0; // repetation counter
238         HAL_TIM_Base_Init(&tim17);
239 
240         HAL_TIM_Base_Start(&tim17); // start timer
241 }
242 
```

Figure 8. Initialization for timer.

```
/*****************************************************************************
 * Function Name      : Delay
 * Description        : This function will provide delay in us.
 *
 *
 * PARAMETERS         : delay
 *
 * RETURNS            : N/A
 * Author             : Abhisha Bhesaniya & Simran Padaniya
 * Date               : 1st August, 2020
 *****************************************************************************
 */
void Delay(int32_t delay)
{
   uint16_t timerVal;

   if (delay > 65535)
        {

            while(delay > 65535)
            {
                TIM17->CNT = 0; //set timer register with 0

                /* Reset counter */
                while(TIM17->CNT < 65535)
                {
                      WDTFeed(); //watchdog timer feed function call
                      asm volatile ("nop\n");
                }

                delay = delay-65535;

            }

             timerVal = delay; //storing delayVal's data into timerVal variable
             TIM17->CNT = 0; //set timer register with 0

             /* Reset counter */
             while(TIM17->CNT < timerVal)
             {
                 WDTFeed(); //watchdog timer feed function call
                 asm volatile ("nop\n");
             }


       }

       else
       {
              timerVal = delay; //storing delayVal's data into timerVal variable
              TIM17->CNT = 0;  //set timer register with 0

             /* Reset counter */
             while(TIM17->CNT < timerVal)
             {
                    WDTFeed(); //watchdog timer feed function call
                    asm volatile ("nop\n");
             }
       }
}
```

Figure 9. Code for Delay

```
304 /*****************************************************************************
305  * Function Name      : RGB
306  * Description        : This function will turn on the specific color in  RGB LED
307  *                      as per the value in parameter.
308  *
309  *
310  * PARAMETERS         : red & green & blue
311  *
312  * RETURNS            : N/A
313  * Author             : Abhisha Bhesaniya & Simran Padaniya
314  * Date               : 14th August, 2020
315  *****************************************************************************
316  */
317 void RGB(int red, int green, int blue)
318 {
319
320   HAL_GPIO_WritePin(GPIOB,GPIO_PIN_10,red); // send signal to R pin of RGB LED
321   HAL_GPIO_WritePin(GPIOB,GPIO_PIN_9,green);// send signal to R pin of RGB LED
322   HAL_GPIO_WritePin(GPIOB,GPIO_PIN_8,blue); // send signal to R pin of RGB LED
323
324 }
325
```

Figure 10. RGB led

```
326 /*****************************************************************************
327  * Function Name      : gpio_init_RGB
328  * Description        : This function will initialize the GPIO pin 8, 9 and 10 of port B.
329  *
330  *
331  * PARAMETERS         : N/A
332  *
333  * RETURNS            : N/A
334  * Author             : Abhisha Bhesaniya & Simran Padaniya
335  * Date               : 1st August, 2020
336  *****************************************************************************
337  */
338 void gpio_init_RGB()//function definition to initialize gpio pins
339 {
340         __HAL_RCC_GPIOB_CLK_ENABLE();//enable clock to the gpio peripheral
341
342         GPIO_InitTypeDef GPIO_Init_Struct = {0};//typedef struct variable
343         GPIO_Init_Struct.Pin = (GPIO_PIN_8 | GPIO_PIN_9 | GPIO_PIN_10);
344         GPIO_Init_Struct.Pull = GPIO_NOPULL;
345         GPIO_Init_Struct.Mode = GPIO_MODE_OUTPUT_PP;
346         GPIO_Init_Struct.Speed = GPIO_SPEED_FREQ_HIGH;
347         GPIO_Init_Struct.Alternate =0;
348
349         HAL_GPIO_Init(GPIOB,&GPIO_Init_Struct);
350
```

Figure 11. GPIO initialization for RGB led.

```
355  * Function Name      : gpio_init_Switch
356  * Description        : This function will initialize the GPIO pin 12 of port A.
357  *
358  *
359  * PARAMETERS         : N/A
360  *
361  * RETURNS            : N/A
362  * Author             : Abhisha Bhesaniya & Simran Padaniya
363  * Date               : 1st August, 2020
364  ******************************************************************************
365  */
366 void gpio_init_Switch()//function definition to initialize gpio pins
367 {
368
369         __HAL_RCC_GPIOA_CLK_ENABLE();//enable clock to the gpio peripheral
370
371         GPIO_InitTypeDef GPIO_Init_Struct = {0};//typedef struct variable
372         GPIO_Init_Struct.Pin = GPIO_PIN_12;
373         GPIO_Init_Struct.Pull = GPIO_NOPULL;
374         GPIO_Init_Struct.Mode = GPIO_MODE_INPUT;
375         GPIO_Init_Struct.Speed = GPIO_SPEED_FREQ_HIGH;
376         GPIO_Init_Struct.Alternate =0;
377
378         HAL_GPIO_Init(GPIOA,&GPIO_Init_Struct);
379 }
380
```

Figure 12. GPIO initialization for Push Button

```
382 /******************************************************************************
383  * Function Name      : gpio_init_DCMotor
384  * Description        : This function will initialize the GPIO pin 0 and 1 of
385  *                      port c for PWM.
386  *
387  *
388  * PARAMETERS         : N/A
389  *
390  * RETURNS            : N/A
391  * Author             : Abhisha Bhesaniya & Simran Padaniya
392  * Date               : 1st August, 2020
393  ******************************************************************************
394  */
395 void gpio_init_DCMotor()//function definition to initialize gpio pins
396 {
397         __HAL_RCC_GPIOC_CLK_ENABLE();//enable clock to the gpio peripheral
398
399         GPIO_InitTypeDef GPIO_Init_Struct;//typedef struct variable
400         GPIO_Init_Struct.Pin = GPIO_PIN_0 | GPIO_PIN_1;
401         GPIO_Init_Struct.Mode = GPIO_MODE_AF_PP;
402         GPIO_Init_Struct.Speed = GPIO_SPEED_FREQ_HIGH;
403         GPIO_Init_Struct.Alternate =GPIO_AF2_TIM1;
404
405         HAL_GPIO_Init(GPIOC,&GPIO_Init_Struct);
406 }
407
```

Figure 13. GPIO initialization for DC motor

```
/*****************************************************************************
 * Function Name      : timer_init_DCMotor
 * Description        : This function will initialize the timer 1 and configure
 *                      the channel as per paramerter enterd.
 *
 *
 * PARAMETERS         : timchannel
 *
 * RETURNS            : N/A
 * Author             : Abhisha Bhesaniya & Simran Padaniya
 * Date               : 1st August, 2020
 *****************************************************************************
 */
void timer_init_DCMotor(uint16_t timchannel)//function to initialize timer1 in pwm mode
{
        __HAL_RCC_TIM1_CLK_ENABLE();//enalbe clock to timer1

        tim1.Instance=TIM1;//initialize to timer1 pointer
        tim1.Init.Prescaler=24;//set the prescaler
        tim1.Init.CounterMode=TIM_COUNTERMODE_UP;
        tim1.Init.Period=0xffff;//period is set to ffff
        tim1.Init.ClockDivision=TIM_CLOCKDIVISION_DIV1;
        tim1.Init.RepetitionCounter=0;//repition counter is set to 0

        HAL_TIM_PWM_Init(&tim1);//initilizer timer in pwm mode

        TIM_OC_InitTypeDef sconfig;//output compare mode

        sconfig.OCMode=TIM_OCMODE_PWM1;
        sconfig.Pulse=0;
        sconfig.OCPolarity=TIM_OCPOLARITY_HIGH;
        sconfig.OCNPolarity=TIM_OCNPOLARITY_LOW;
        sconfig.OCFastMode=TIM_OCFAST_DISABLE;
        sconfig.OCIdleState=TIM_OCIDLESTATE_RESET;
        sconfig.OCNIdleState=TIM_OCNIDLESTATE_RESET;

        HAL_TIM_PWM_ConfigChannel(&tim1,&sconfig,timchannel);//config outuput channel in pwm mode
}
```

Figure 14. timer initialization for DC Motor

```
447 /*****************************************************************************
448  * Function Name      : start_DCMotor
449  * Description        : This function will start pwm and load CCR resister of timer 1
450  *                      as per entered parameter ans the this value will control the speed.
451  *
452  *
453  * PARAMETERS         : timchannel & speed
454  *
455  * RETURNS            : N/A
456  * Author             : Abhisha Bhesaniya & Simran Padaniya
457  * Date               : 1st August, 2020
458  *****************************************************************************
459  */
460 void start_DCMotor(uint16_t timchannel,uint16_t speed)//function to start timer pwm
461 {
462         if(timchannel==0)
463         {
464                 TIM1->CCR1=speed; //load register for channel 1
465         }
466         else
467         {
468                 TIM1->CCR2=speed; //load register for channel 2
469         }
470
471         HAL_TIM_PWM_Start(&tim1,timchannel);//start pwm
472 }
473
474 /*****************************************************************************
475  * Function Name      : stop_DCMotor
476  * Description        : This function will stop timer.
477  *
478  * PARAMETERS         : timchannel
479  *
480  * RETURNS            : N/A
481  * Author             : Abhisha Bhesaniya & Simran Padaniya
482  * Date               : 1st August, 2020
483  *****************************************************************************
484  */
485 void stop_DCMotor(uint16_t timchannel)//functio to stop pwm timer
486 {
487         HAL_TIM_PWM_Stop(&tim1,timchannel);//stop PWM
488 }
489
```

Figure 15. Code to start and stop DC Motor.

References

Dc motor datasheet

https://bestbrothersgroup.com/automatic-doors/

https://www.youtube.com/watch?v=unRAsTh5eNs&feature=youtu.be

https://www.youtube.com/watch?v=Yu1BsB-ToIY&feature=youtu.be

https://www.youtube.com/watch?v=JhcFTpQrbzA&feature=youtu.be

lectures notes and slide