**LED control using push button**

* **CODE**

#include "stm32f4xx\_hal.h"

void Init\_Pushbutton(void);

void Init\_leds(void);

int main(void)

{

void Init\_Pushbutton();

void Init\_leds();

while(1)

{

if(HAL\_GPIO\_ReadPin(GPIOA,GPIO\_PIN\_0))

{

HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15,GPIO\_PIN\_SET);

}

else

HAL\_GPIO\_WritePin(GPIOD,GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15,GPIO\_PIN\_RESET);

}

}

void Init\_Pushbutton(void){

\_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();

GPIO\_InitTypeDef pushbutton;

pushbutton.Pin = GPIO\_PIN\_0;

pushbutton.Mode = GPIO\_MODE\_INPUT;

pushbutton.Pull = GPIO\_NOPULL;

HAL\_GPIO\_Init(GPIOA, &pushbutton);

}

void Init\_leds(void){

\_\_HAL\_RCC\_GPIOD\_CLK\_ENABLE();

GPIO\_InitTypeDef ledpins;

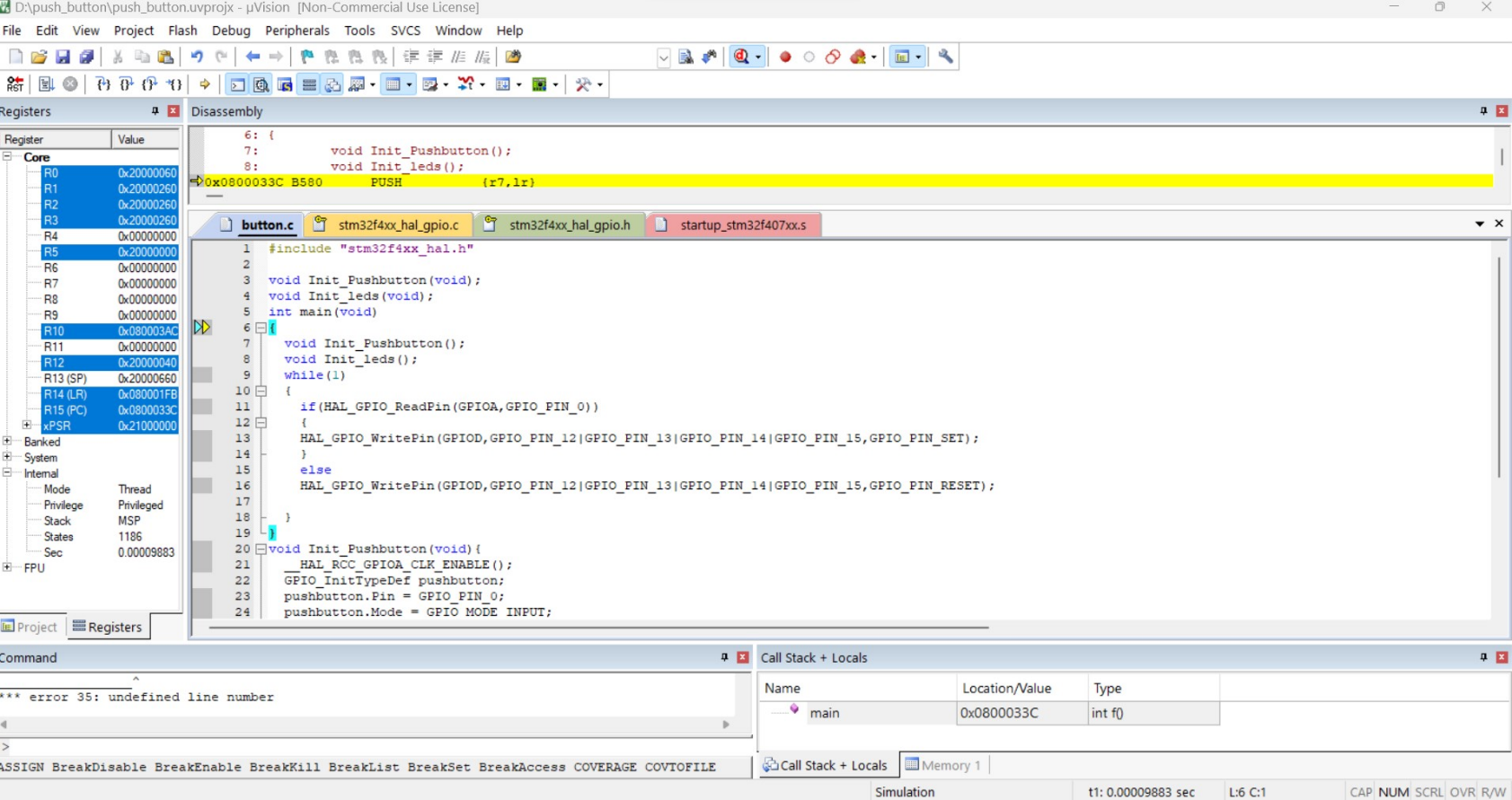
ledpins.Pin = GPIO\_PIN\_12|GPIO\_PIN\_13|GPIO\_PIN\_14|GPIO\_PIN\_15;

ledpins.Mode = GPIO\_MODE\_OUTPUT\_PP;

HAL\_GPIO\_Init(GPIOD, &ledpins);

}

* Output



**SEVEN SEGMENT DISPLAY control using push button**

* **CODE**

#include "stm32f4xx\_hal.h"

#include "stm32f4xx\_hal\_gpio.h"

void displayDigit(int digit, int value);

void displayNumber(int number);

int main(void)

{

HAL\_Init();

GPIO\_InitTypeDef GPIO\_InitStruct;

\_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();

\_\_HAL\_RCC\_GPIOB\_CLK\_ENABLE();

GPIO\_InitStruct.Pin = GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_HIGH;

HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);

GPIO\_InitStruct.Pin = GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6| GPIO\_PIN\_7 | GPIO\_PIN\_8 | GPIO\_PIN\_9 | GPIO\_PIN\_10;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_HIGH;

HAL\_GPIO\_Init(GPIOB, &GPIO\_InitStruct);

while (1)

{

displayNumber(1234);

HAL\_Delay(1000);

displayNumber(5678);

HAL\_Delay(1000);

}

}

void displayDigit(int digit, int value)

{

switch (digit)

{

case 1:

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0, GPIO\_PIN\_SET);

break;

case 2:

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_1, GPIO\_PIN\_SET);

break;

case 3:

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_2, GPIO\_PIN\_SET);

break;

case 4:

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_3, GPIO\_PIN\_SET);

break;

}

switch (value)

{

case 0:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6 | GPIO\_PIN\_7

| GPIO\_PIN\_8, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_9 | GPIO\_PIN\_10, GPIO\_PIN\_RESET);

break;

case 1:

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_6 | GPIO\_PIN\_8, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_7 | GPIO\_PIN\_9

| GPIO\_PIN\_10, GPIO\_PIN\_RESET);

break;

case 2:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_5 | GPIO\_PIN\_6 | GPIO\_PIN\_7 | GPIO\_PIN\_8 | GPIO\_PIN\_9,

GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_10,

GPIO\_PIN\_RESET);

break;

case 3:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6 | GPIO\_PIN\_7 | GPIO\_PIN\_9,

GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_3 | GPIO\_PIN\_8 | GPIO\_PIN\_10,

GPIO\_PIN\_RESET);

break;

case 4:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_3 | GPIO\_PIN\_6 | GPIO\_PIN\_7 | GPIO\_PIN\_9,

GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_8 | GPIO\_PIN\_10,

GPIO\_PIN\_RESET);

break;

case 5:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6 | GPIO\_PIN\_8 | GPIO\_PIN\_9,

GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_3 | GPIO\_PIN\_7 | GPIO\_PIN\_10,

GPIO\_PIN\_RESET);

break;

case 6:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6 | GPIO\_PIN\_7 | GPIO\_PIN\_8

| GPIO\_PIN\_9, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_3 | GPIO\_PIN\_10, GPIO\_PIN\_RESET);

break;

case 7:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_8,

GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_6 | GPIO\_PIN\_7 | GPIO\_PIN\_9 | GPIO\_PIN\_10,

GPIO\_PIN\_RESET);

break;

case 8:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6 | GPIO\_PIN\_7

| GPIO\_PIN\_8 | GPIO\_PIN\_9, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_10, GPIO\_PIN\_RESET);

break;

case 9:

HAL\_GPIO\_WritePin(GPIOB,

GPIO\_PIN\_3 | GPIO\_PIN\_4 | GPIO\_PIN\_5 | GPIO\_PIN\_6 | GPIO\_PIN\_8

| GPIO\_PIN\_9, GPIO\_PIN\_SET);

HAL\_GPIO\_WritePin(GPIOB, GPIO\_PIN\_7 | GPIO\_PIN\_10, GPIO\_PIN\_RESET);

break;

}

}

void displayNumber(int number)

{

int digit1 = number / 1000;

int digit2 = (number / 100) % 10;

int digit3 = (number / 10) % 10;

int digit4 = number % 10;

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3,

GPIO\_PIN\_RESET);

displayDigit(1, digit1);

HAL\_Delay(1);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3,

GPIO\_PIN\_RESET);

displayDigit(2, digit2);

HAL\_Delay(1);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3,

GPIO\_PIN\_RESET);

displayDigit(3, digit3);

HAL\_Delay(1);

HAL\_GPIO\_WritePin(GPIOA, GPIO\_PIN\_0 | GPIO\_PIN\_1 | GPIO\_PIN\_2 | GPIO\_PIN\_3,

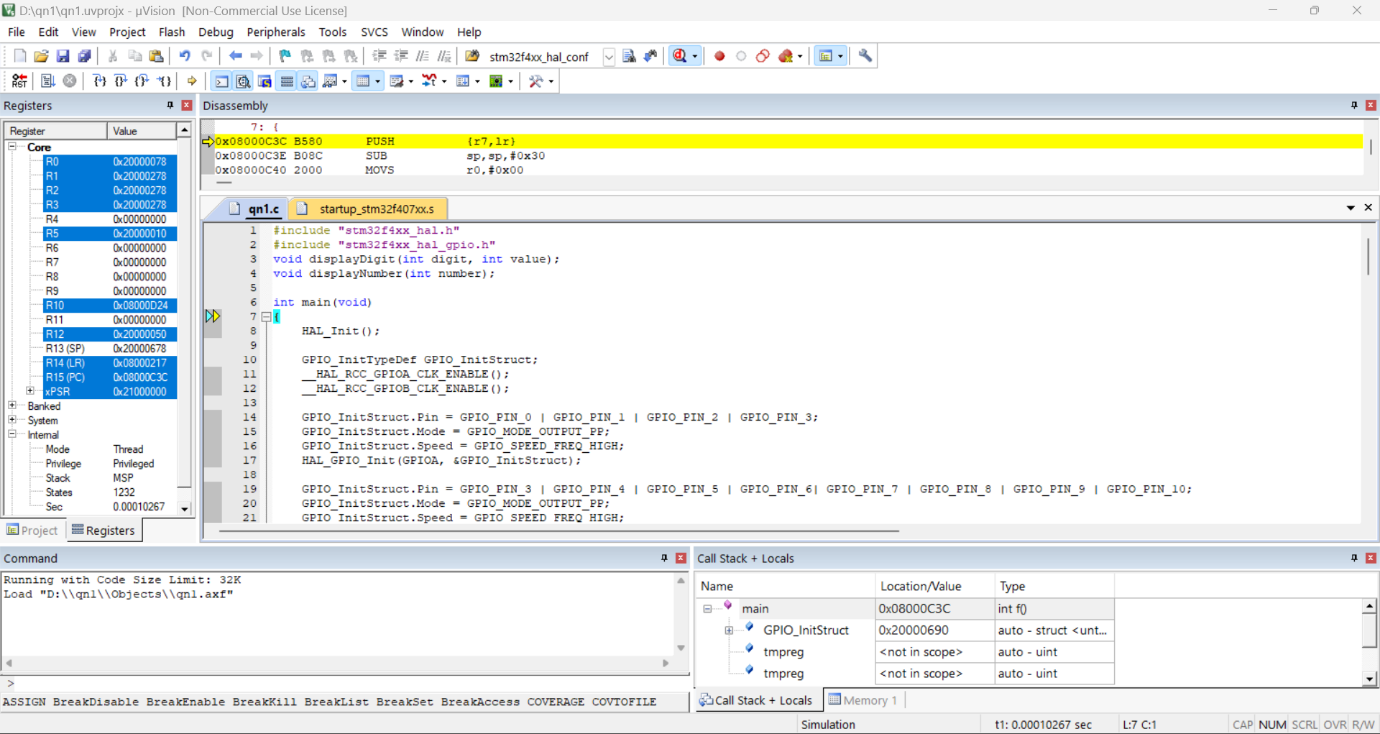
GPIO\_PIN\_RESET);

displayDigit(4, digit4);

HAL\_Delay(1);

}

* OUTPUT



STEPPER MOTOR

* CODE

#include "stm32f4xx.h"

#include "stm32f4xx\_hal\_gpio.h"

// Define motor control pins

#define STEP\_PIN GPIO\_PIN\_0

#define DIR\_PIN GPIO\_PIN\_1

#define ENABLE\_PIN GPIO\_PIN\_2

// Function to initialize GPIO pins

void GPIO\_Init(void) {

GPIO\_InitTypeDef GPIO\_InitStruct;

\_\_HAL\_RCC\_GPIOA\_CLK\_ENABLE();

GPIO\_InitStruct.Pin = STEP\_PIN | DIR\_PIN | ENABLE\_PIN;

GPIO\_InitStruct.Mode = GPIO\_MODE\_OUTPUT\_PP;

GPIO\_InitStruct.Speed = GPIO\_SPEED\_FREQ\_HIGH;

HAL\_GPIO\_Init(GPIOA, &GPIO\_InitStruct);

}

// Function to control the stepper motor

void StepperMotor\_Control(uint8\_t direction, uint16\_t steps, uint16\_t delay\_ms) {

// Set direction pin

if (direction == 0) {

HAL\_GPIO\_WritePin(GPIOA, DIR\_PIN, GPIO\_PIN\_RESET);

} else {

HAL\_GPIO\_WritePin(GPIOA, DIR\_PIN, GPIO\_PIN\_SET);

}

// Enable the motor driver

HAL\_GPIO\_WritePin(GPIOA, ENABLE\_PIN, GPIO\_PIN\_RESET);

// Perform steps

for (uint16\_t i = 0; i < steps; i++) {

// Set step pin high

HAL\_GPIO\_WritePin(GPIOA, STEP\_PIN, GPIO\_PIN\_SET);

// Delay

HAL\_Delay(delay\_ms);

// Set step pin low

HAL\_GPIO\_WritePin(GPIOA, STEP\_PIN, GPIO\_PIN\_RESET);

// Delay

HAL\_Delay(delay\_ms);

}

// Disable the motor driver

HAL\_GPIO\_WritePin(GPIOA, ENABLE\_PIN, GPIO\_PIN\_SET);

}

int main(void) {

// Initialize GPIO pins

GPIO\_Init();

while (1) {

// Rotate clockwise for 200 steps with a delay of 2ms

StepperMotor\_Control(0, 200, 2);

// Delay

HAL\_Delay(1000);

// Rotate counter-clockwise for 200 steps with a delay of 2ms

StepperMotor\_Control(1, 200, 2);

// Delay

HAL\_Delay(1000);

}

}

* OUTPUT

