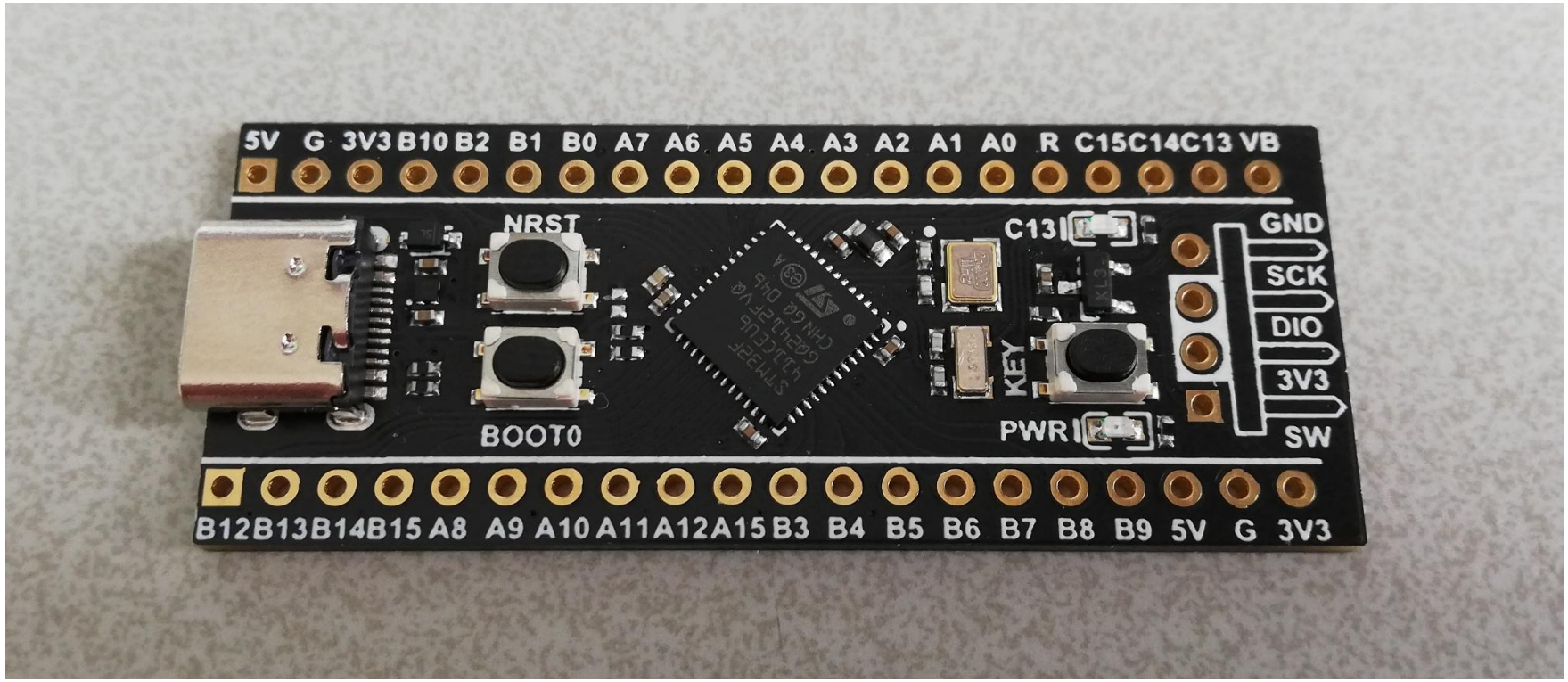
Программирование микроконтроллеров

Назаров Александр Александрович – Программист РЦР ДГТУ (2-101)

Телеграмм – @casonka

VK - https://vk.com/casonka

1. Что такое микроконтроллер ?



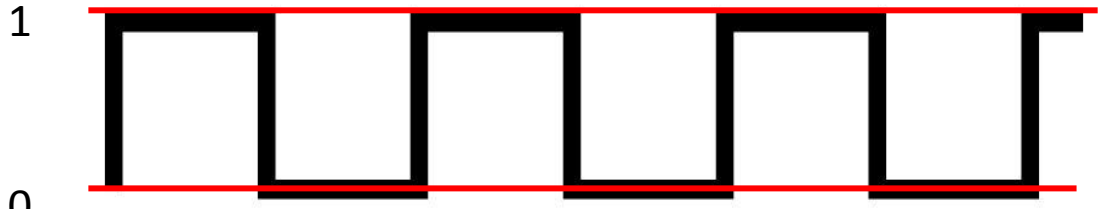
2. Обработка сигналов

Сигналы бывают двух видов: цифровые и аналоговые.

Цифровой (нижнее фото) имеет два состояния: 0 когда нет сигнала и 1 когда сигнал имеется.

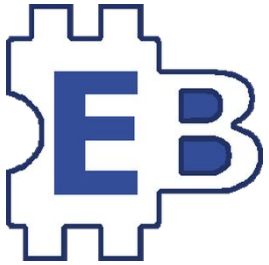
Аналоговый сигнал гораздо сложнее, его значение определяется амплитудным уровнем сигнала

Аналоговый сигнал



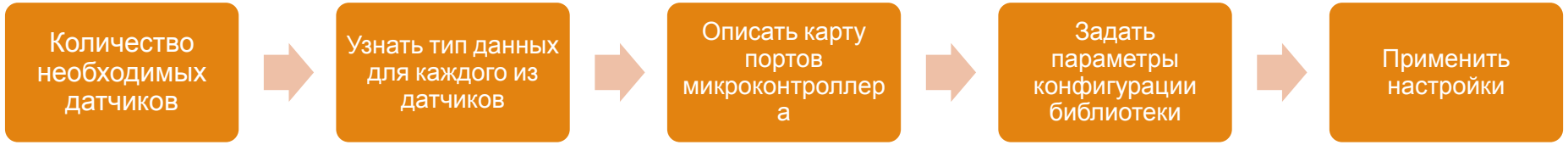
Цифровой сингал

3. На чем писать код ?

- Embitz 

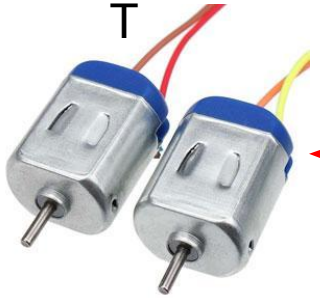
- Eclipse
- STM32CubeIDE

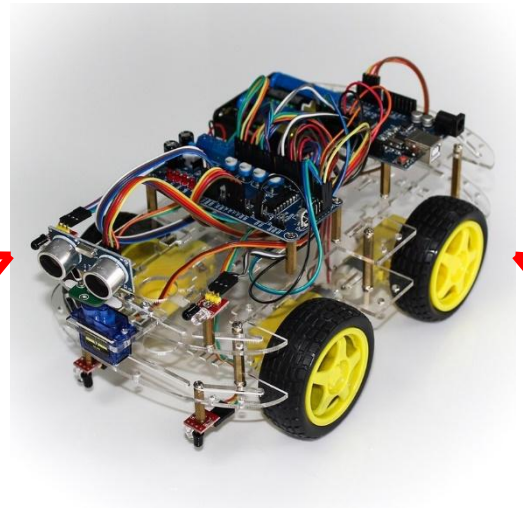
4. Последовательность программирования для настройки библиотеки



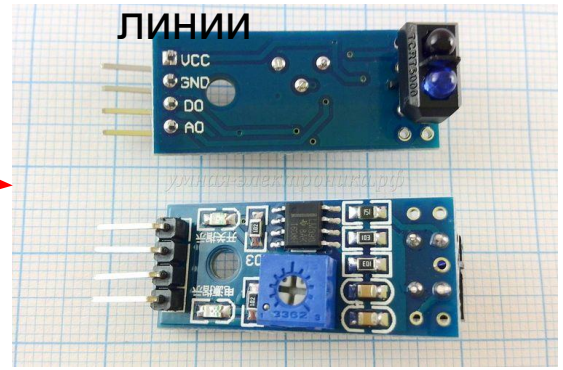
5. Что потребуется для программирования обыкновенного робота ?

ДП
Т

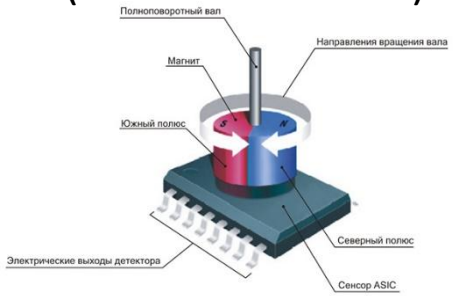




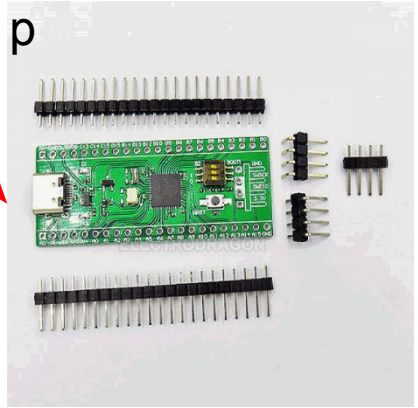
Датчик
линии



Энкодер
(необязательно)



Микроконтролле
р



6. Библиотека FIL

Почему используется FIL, а не стандартная библиотека ?

- Ничего лишнего
- Удобные инструменты отладки и отслеживания
- Модульное, параметрическое программирование на языке C
- Облегченные команды для программирования микроконтроллера

6.1 Карта портов

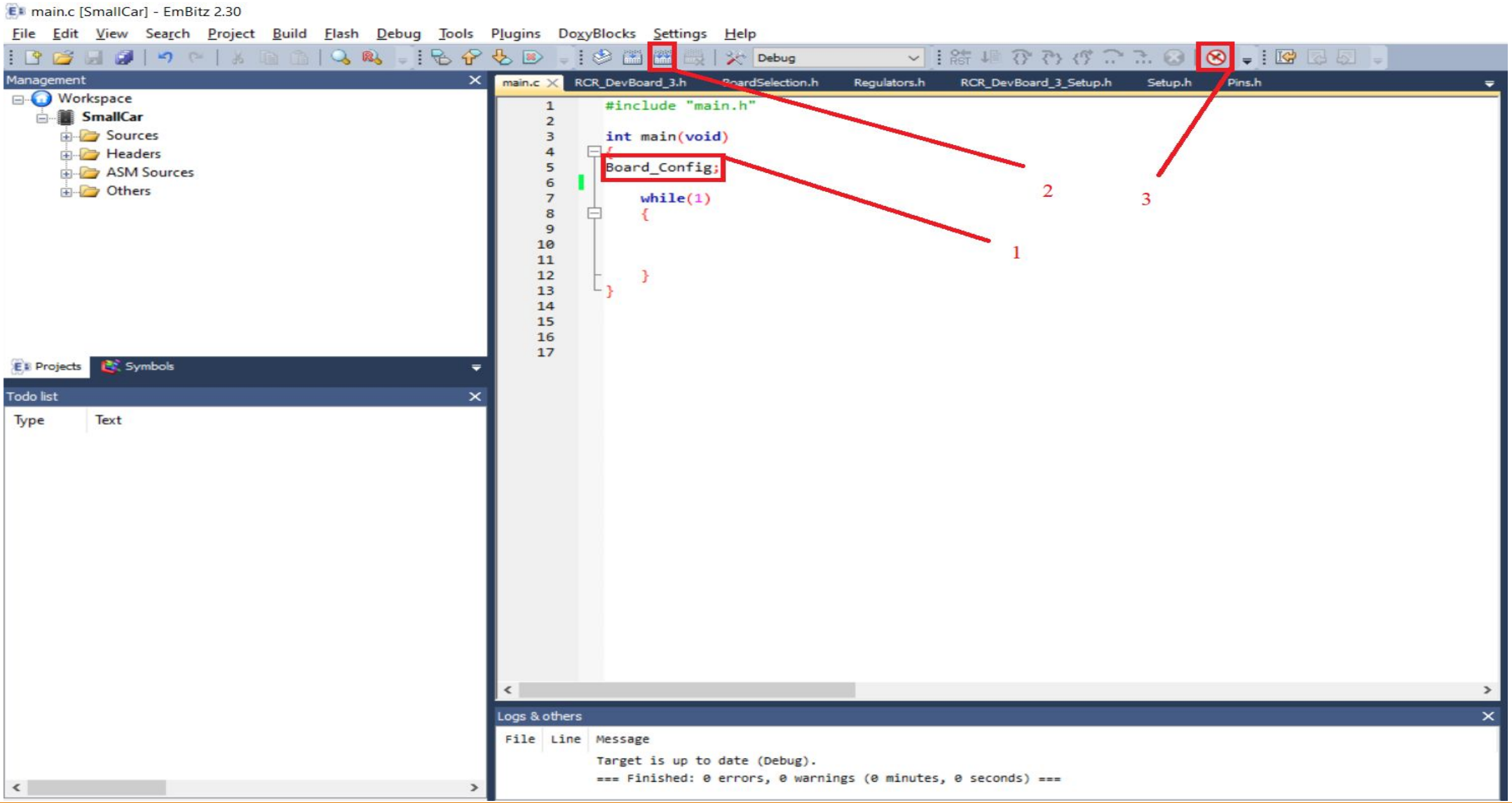
Карта портов формирует уникальные I для каждого пина микроконтроллера. Создается пользователем и может иметь названия связанные с :

- Название группы и номера порта (Например PA1)
- Название порта у датчика (например SDA_PIN)

```
////////////////////////////////////////////////////////////////////////////////
#define PA1  GPIOPinID(PORTA,1)


#define Sensor_pin      GPIOPinID(PORTA,1)


////////////////////////////////////////////////////////////////////////////////
#define BTN1_DIR_PIN            GPIOPinID(PORTB,14)       // Wheels direction
#define BTN2_DIR_PIN            GPIOPinID(PORTB,15)       // NONE
////////////////////////////////////////////////////////////////////////////////
//_____________________________________Encoders_____________________________________//
////////////////////////////////////////////////////////////////////////////////
#define ENCODER1A_PIN           GPIOPinID(PORTA,0)
#define ENCODER1B_PIN           GPIOPinID(PORTA,1)
#define ENCODER2A_PIN           GPIOPinID(PORTA,6)
#define ENCODER2B_PIN           GPIOPinID(PORTA,7)

#define ENCODER1_CNT           ((uint32_t *)&(TIM3->CNT))
#define ENCODER2_CNT           ((uint32_t *)&(TIM4->CNT))
```

```
#define __configUSE_RCC                 1
#define __configUSE_GPIO                1
#define __configUSE_TIM                 0
#define __configUSE_USART               0
#define __configUSE_DMA                 0
#define __configUSE_I2C                 0
#define __configUSE_ADC                 0
#define __configUSE_EXTI                0
#define __configUSE_RTC                 0
#define __configUSE_FREERTOS            0

#define __configCALC_RCC                0
#define __configCALC_TIM                0
#define __configCALC_USART              0
#define __configCALC_Regulators         0
#define __configCALC_Matrix             0

#define Board_Config                                                    {\
                    SetGPIOA;                                            \
                    GPIOConfPin(PA1, ALTERNATE, PUSH_PULL, FAST_S, NO_PULL_UP);\
                                                                         \}
```

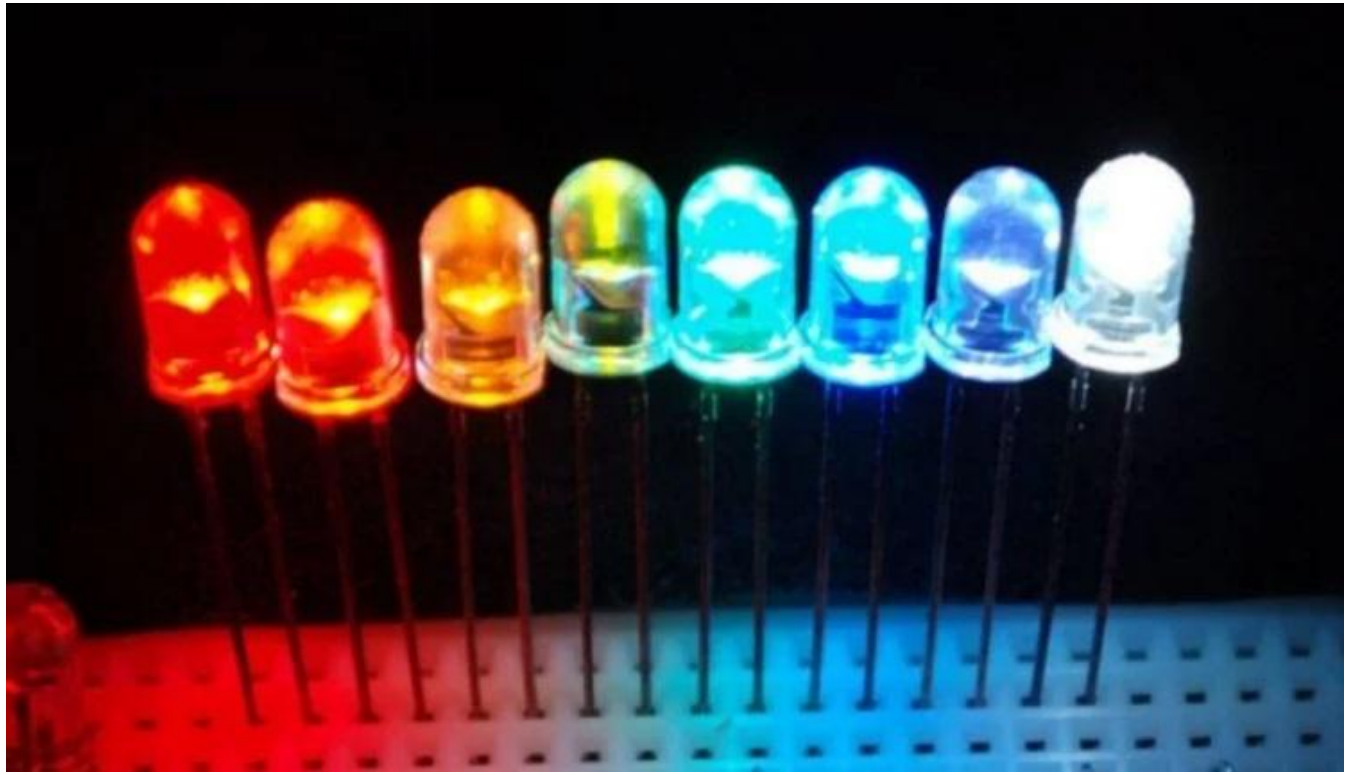
6.3 Применение настроек



7.1 Управление светодиодом

Светодиод используется в робототехнических и других системах для визуального оповещения человеку о успешного/неудачного запуска того или иного участка кода.

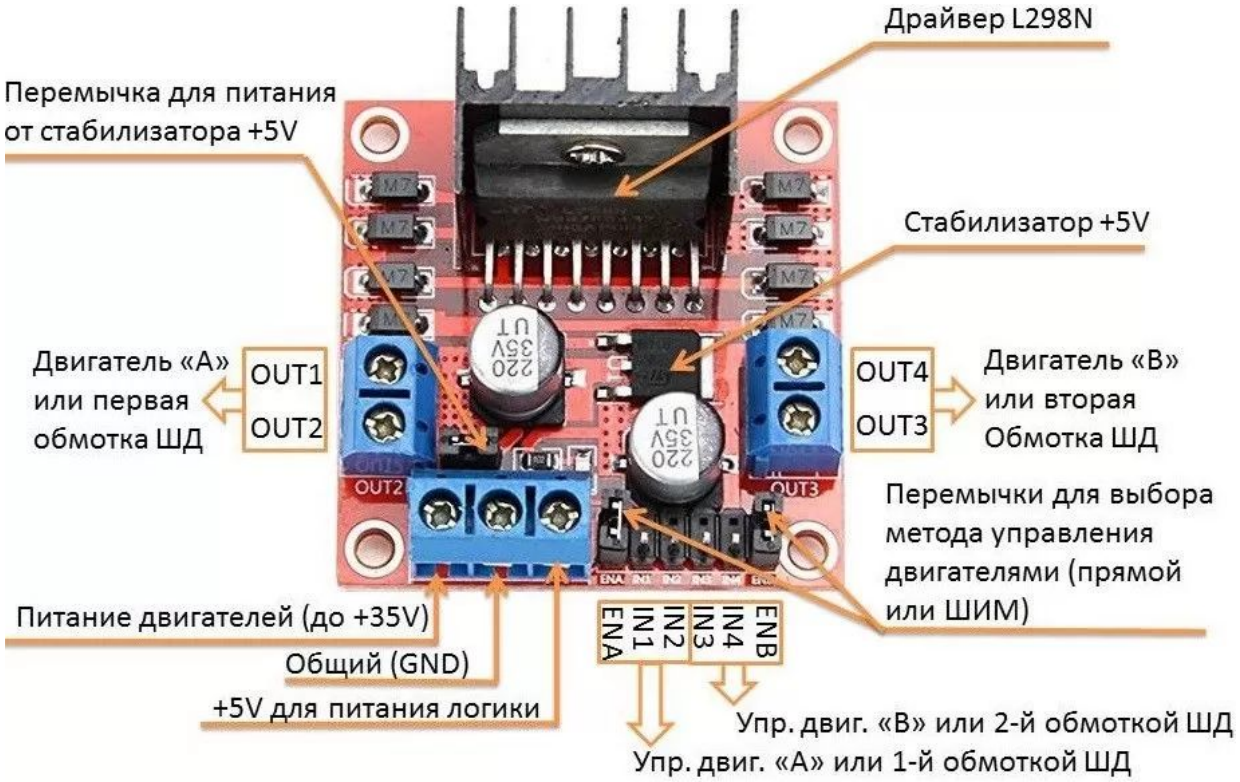
Не является обязательным в роботах, но желателен для быстрой оценки поломки чего-либо (или плохого контакта).

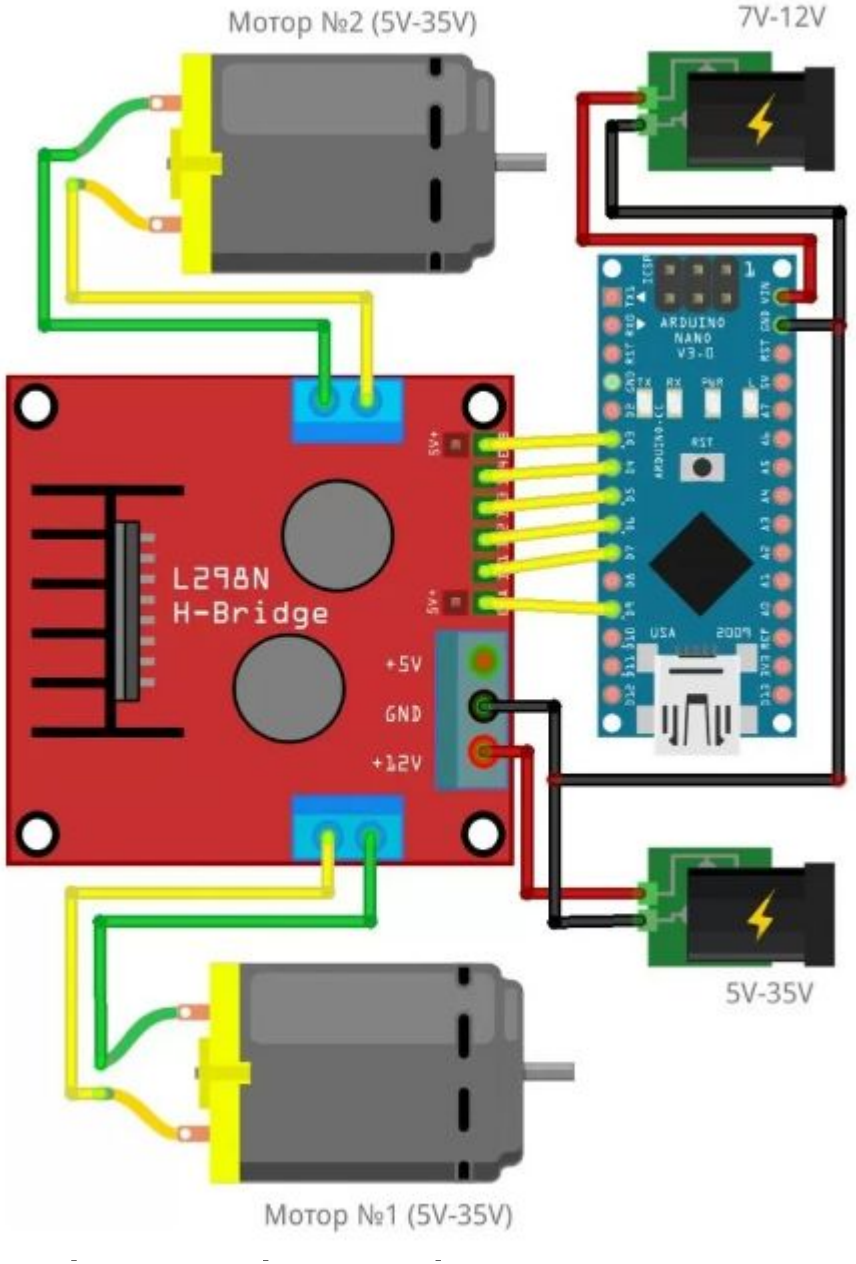


```
#pragma once

#define __configUSE_RCC                 1
#define __configUSE_GPIO                1
#define __configUSE_TIM                 0
#define __configUSE_USART               0
#define __configUSE_DMA                 0
#define __configUSE_I2C                 0
#define __configUSE_ADC                 0
#define __configUSE_EXTI                0
#define __configUSE_RTC                 0
#define __configUSE_FREERTOS            0

#define __configCALC_RCC                0
#define __configCALC_TIM                0
#define __configCALC_USART              0
#define __configCALC_Regulators         0
#define __configCALC_Matrix             0

#define LED_PIN  GPIOPinID(PORTA,1)

#define Board_Config                                                    {\
                    SetGPIOA;                                            \
                    GPIOConfPin(LED_PIN, GENERAL, PUSH_PULL, FAST_S, NO_PULL_UP);\
                                                                        }
```

```
1   #include "main.h"
2
3   int main(void)
4   {
5   Board_Config;
6
7   SetPin(LED_PIN);
8
9   ResetPin(LED_PIN);
10
11      while(1)
12      {
13
14
15
16      }
17  }
18
19
```

7.2 Управление ДПТ





```
#pragma once

#define PWM1_PIN          GPIOPinID(PORTA,8)
#define PWM2_PIN          GPIOPinID(PORTA,9)

#define PWM1_CCR         ((uint32_t *)&(TIM1->CCR1))
#define PWM2_CCR         ((uint32_t *)&(TIM1->CCR2))

#define DIR1_PIN          GPIOPinID(PORTA,10)
```

```
#include "main.h"

void SetVoltage(float Duty)
{
    if(Duty >= 0.0)
    {
        SetPin(DIR1_PIN);
        ResetPin(DIR2_PIN);
            *PWM1_CCR = ((int32_t) (Duty * 2000));
        return;
    }
    else
    {
        SetPin(DIR2_PIN);
        ResetPin(DIR1_PIN);
            *PWM1_CCR = ((int32_t)(2000 +  (Duty * 2000)));
        return;
    }
}

int main(void)
{
Board_Config;

SetVoltage(0.2);
    while(1)
    {



    }
}
```

```
#pragma once

#define __configUSE_RCC                 1
#define __configUSE_GPIO                1
#define __configUSE_TIM                 1
#define __configUSE_USART               0
#define __configUSE_DMA                 0
#define __configUSE_I2C                 0
#define __configUSE_ADC                 0
#define __configUSE_EXTI                0
#define __configUSE_RTC                 0
#define __configUSE_FREERTOS            0

#define __configCALC_RCC                0
#define __configCALC_TIM                0
#define __configCALC_USART              0
#define __configCALC_Regulators         0
#define __configCALC_Matrix             0

#define Board_Config                                                                {\
                    SetGPIOA;                                                        \
                    SetTIM1;                                                         \
                    GPIOConfPin(PWM1_PIN, ALTERNATE, PUSH_PULL, FAST_S, NO_PULL_UP);  \
                    GPIOConfPin(PWM2_PIN, ALTERNATE, PUSH_PULL, FAST_S, NO_PULL_UP);  \
                    GPIOConfAF(PWM1_PIN,AF1);                                        \
                    GPIOConfAF(PWM2_PIN,AF1);                                        \
                    GPIOConfPin(DIR1_PIN, GENERAL, PUSH_PULL, FAST_S, NO_PULL_UP);    \
                    GPIOConfPin(DIR2_PIN, GENERAL, PUSH_PULL, FAST_S, NO_PULL_UP);    \
                    TimPWMConfigure(TIM1,840,2000,1,1,0,0);                          \
                                                                                     }
```

7.3 Обработка данных с датчика линии

```
#define __configCALC_USART              0
#define __configCALC_Regulators         0
#define __configCALC_Matrix             0

#define __configADC_Mode                    (2)
#define __configCONVERT_Volts               (0)
#define __configUSE_Battery_Charging        (0)
#define __configUSE_Temperature_Sensor      (0)
//-------------------------------------------------------------------------//

#define __configUSE_SENSOR_1                (0)
#define __configUSE_SENSOR_2                (1)
#define __configUSE_SENSOR_3                (-1)
#define __configUSE_SENSOR_4                (-1)
#define __configUSE_SENSOR_5                (-1)
#define __configUSE_SENSOR_6                (-1)
#define __configUSE_SENSOR_7                (-1)
#define __configUSE_SENSOR_8                (-1)
#define __configUSE_SENSOR_9                (-1)
#define __configUSE_SENSOR_10               (-1)

#define __configADC_InterruptRequest        (0)
#define __configADC_Divider                 (3)
#define __configADC_RESOLUTION              (12)        // 12-bit resolution
#define __configADC_CYCLES                  (ADC_480_CYCLES)
#define __configADC_DIVIDER                 (3)

#define Board_Config                                                                {\
                    SetGPIOA;                                                        \
                    SetTIM1;                                                         \
                    SetADC1;                                                         \
                    SetDMA2;                                                         \
                    GPIOConfPin(PWM1_PIN, ALTERNATE, PUSH_PULL, FAST_S, NO_PULL_UP);    \
                    GPIOConfPin(PWM2_PIN, ALTERNATE, PUSH_PULL, FAST_S, NO_PULL_UP);    \
                    GPIOConfAF(PWM1_PIN,AF1);                                        \
                    GPIOConfAF(PWM2_PIN,AF1);                                        \
                    GPIOConfPin(DIR1_PIN, GENERAL, PUSH_PULL, FAST_S, NO_PULL_UP);      \
                    GPIOConfPin(DIR2_PIN, GENERAL, PUSH_PULL, FAST_S, NO_PULL_UP);      \
                    GPIOConfPin(SENSOR1_PIN, ANALOG, PUSH_PULL, FAST_S, NO_PULL_UP);    \
                    GPIOConfPin(SENSOR2_PIN, ANALOG, PUSH_PULL, FAST_S, NO_PULL_UP);    \
                    TimPWMConfigure(TIM1,840,2000,1,1,0,0);                          }
```

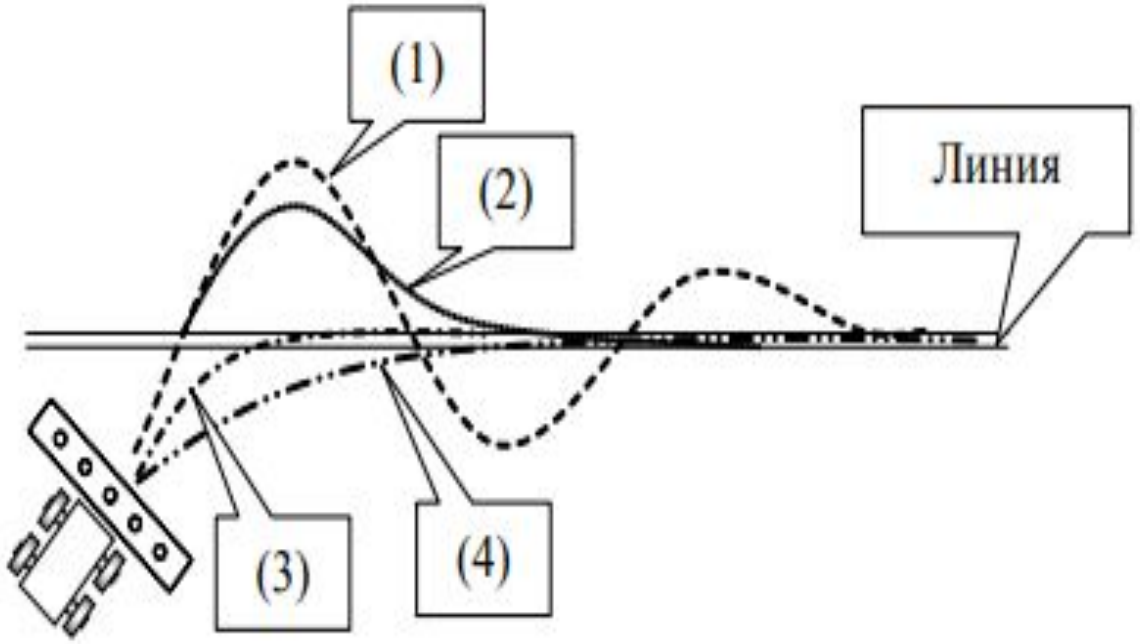
```
#include "main.h"

uint16_t ADC1_Data[ADC1_NUMB];
int main(void)
{
Board_Config;
    ADCAddRegularChannel(ADC1,0,ADC_480_CYCLES);
    ADCAddRegularChannel(ADC1,1,ADC_480_CYCLES);
    ConnectADCTODMA(HIGH_P,ADC1_Data,0);
    ADCSimpleConfigure(ADC1);
    while(1)
    {



    }
}
```

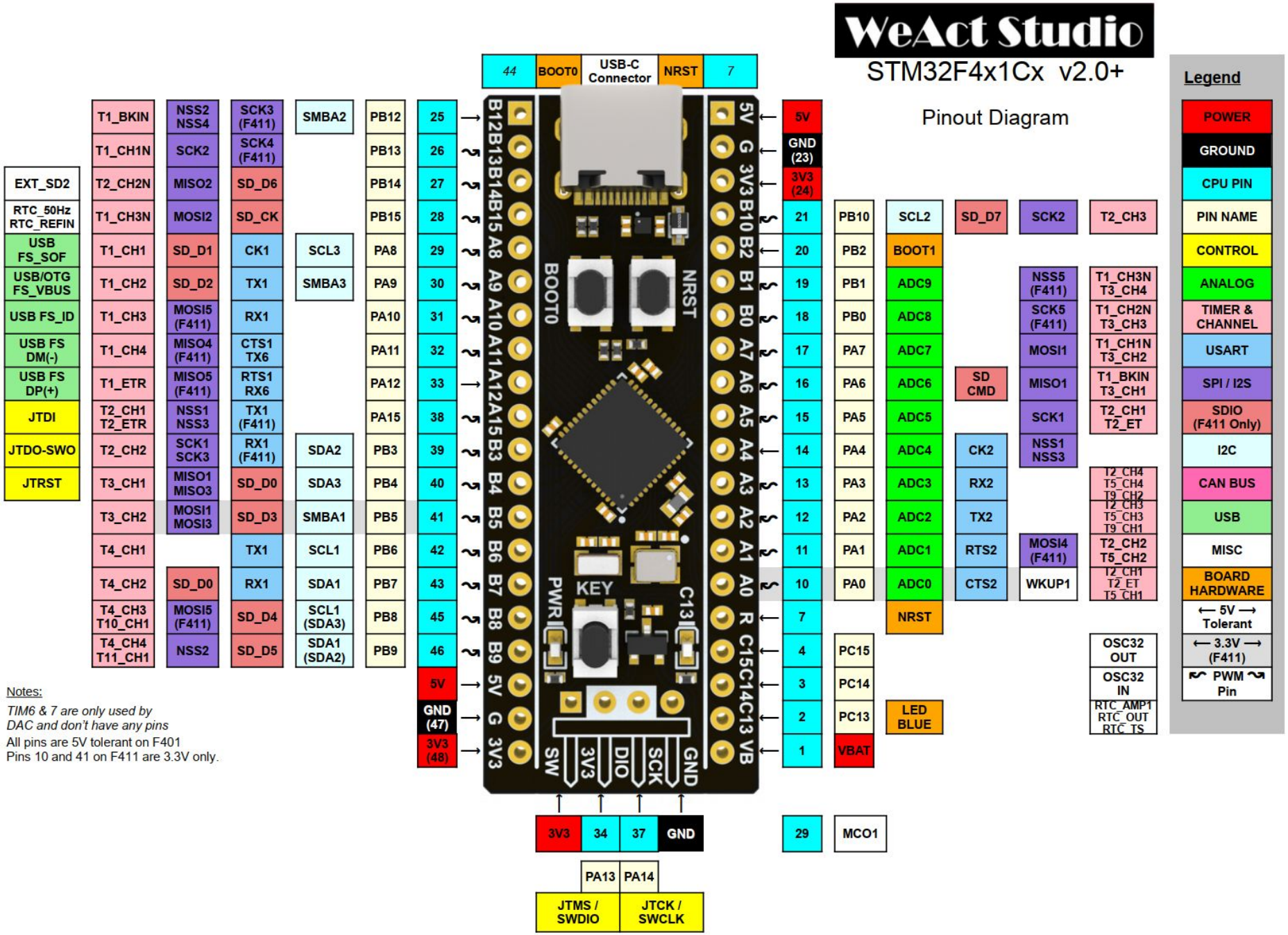
```
#pragma once

#define LED_PIN          GPIOPinID(PORTC,13)

#define PWM1_PIN         GPIOPinID(PORTA,8)
#define PWM2_PIN         GPIOPinID(PORTA,9)

#define PWM1_CCR        ((uint32_t *)&(TIM1->CCR1))
#define PWM2_CCR        ((uint32_t *)&(TIM1->CCR2))

#define DIR1_PIN         GPIOPinID(PORTA,10)
#define DIR2_PIN         GPIOPinID(PORTA,11)

#define SENSOR1_PIN     GPIOPinID(PORTA,0)
#define SENSOR2_PIN     GPIOPinID(PORTA,1)
```

7.4 Реализация алгоритма езды по линии с регулятором

Для построения правильной с точки зрения алгоритма траектории движения робота используются **регуляторы.**

Регулятор позволяет корректировать сигнал управления на электродвигатели, основываясь на входной информации (в нашем примере значение с датчиков линии).




WeAct Studio
STM32F4x1Cx v2.0+
Pinout Diagram
Legend
POWER
GROUND
CPU PIN
PIN NAME
CONTROL
ANALOG
TIMER & CHANNEL
USART
SPI / I2S
SDIO (F411 Only)
I2C
CAN BUS
USB
MISC
BOARD HARDWARE
← 5V → Tolerant
← 3.3V → (F411)
PWM Pin
44 BOOT0 USB-C Connector NRST 7
EXT_SD2
RTC_50Hz RTC_REFIN
USB FS_SOF
USB/OTG FS_VBUS
USB FS_ID
USB FS DM(-)
USB FS DP(+)
JTDI
JTDO-SWO
JTRST
T1_BKIN T1_CH1N T2_CH2N T1_CH3N T1_CH1 T1_CH2 T1_CH3 T1_CH4 T1_ETR T2_CH1 T2_ETR T2_CH2 T3_CH1 T3_CH2 T4_CH1 T4_CH2 T4_CH3 T10_CH1 T4_CH4 T11_CH1
NSS2 NSS4 SCK2 MISO2 MOSI2 SD_D1 SD_D2 MOSI5 (F411) MISO4 (F411) MISO5 (F411) NSS1 NSS3 SCK1 SCK3 MISO1 MISO3 MOSI1 MOSI3 SD_D0 MOSI5 (F411) NSS2
SCK3 (F411) SCK4 (F411) SD_D6 SD_CK CK1 TX1 RX1 CTS1 TX6 RTS1 RX6 TX1 (F411) RX1 (F411) SD_D3 TX1 RX1 SD_D4 SD_D5
SMBA2 SCL3 SMBA3 SDA2 SDA3 SMBA1 SCL1 SDA1 SCL1 (SDA3) SDA1 (SDA2)
PB12 PB13 PB14 PB15 PA8 PA9 PA10 PA11 PA12 PA15 PB3 PB4 PB5 PB6 PB7 PB8 PB9
25 26 27 28 29 30 31 32 33 38 39 40 41 42 43 45 46
5V GND (47) 3V3 (48)
B12 B13 B14 B15 A8 A9 A10 A11 A12 A15 B3 B4 B5 B6 B7 B8 B9 5V G 3V3
BOOT0 NRST KEY PWR C13
5V G 3V3 B10 B2 B1 B0 A7 A6 A5 A4 A3 A2 A1 A0 R C15 C14 C13 VB
5V GND (23) 3V3 (24) 21 20 19 18 17 16 15 14 13 12 11 10 7 4 3 2 1
PB10 PB2 PB1 PB0 PA7 PA6 PA5 PA4 PA3 PA2 PA1 PA0 PC15 PC14 PC13 VBAT
SCL2 BOOT1 ADC9 ADC8 ADC7 ADC6 ADC5 ADC4 ADC3 ADC2 ADC1 ADC0 NRST LED BLUE
SD_D7 SD CMD CK2 RX2 TX2 RTS2 CTS2
SCK2 NSS5 (F411) SCK5 (F411) MOSI1 MISO1 SCK1 NSS1 NSS3 MOSI4 (F411) WKUP1
T2_CH3 T1_CH3N T3_CH4 T1_CH2N T3_CH3 T1_CH1N T3_CH2 T1_BKIN T3_CH1 T2_CH1 T2_ET T2_CH4 T5_CH4 T9_CH2 T2_CH3 T5_CH3 T9_CH1 T2_CH2 T5_CH2 T2_CH1 T2_ET T5_CH1 OSC32 OUT OSC32 IN RTC_AMP1 RTC_OUT RTC_TS
3V3 34 37 GND
PA13 PA14
JTMS / SWDIO JTCK / SWCLK
29 MCO1
SW 3V3 DIO SCK GND
Notes:
TIM6 & 7 are only used by DAC and don't have any pins
All pins are 5V tolerant on F401
Pins 10 and 41 on F411 are 3.3V only.