# Управление моторами

## Константы портов моторов

Порты обозначенные буквами, предназначаются для моторов. Управление моторами наиболее удобно при использовании следующих констант:

* OUT\_A – константа, обозначающая порт A.
* OUT\_B – константа, обозначающая порт B.
* OUT\_C – константа, обозначающая порт C.
* OUT\_AB – константа, обозначающая порт A и порт B.
* OUT\_AС – константа, обозначающая порт A и порт C.
* OUT\_BС – константа, обозначающая порт B и порт C.
* OUT\_ABС – константа, обозначающая порт A и порт B, и порт C.

## Функции управления моторами

Для управления моторами необходимо определять порты, к которым они подключены, и устанавливать уровень мощности, который будет использован. Уровень мощности определяется в пределе от -100 до 100, причем отрицательные значения имеют отрицательный смысл, т.е. для функции вращения вперед отрицательная мощность будет вращать мотор назад. Предусмотрены следующие функции:

* OnFwd(byte *outputs*, char *power*) – вращение моторов вперед,
* OnRev(byte *outputs*, char *power*) – вращение моторов назад,

где outputs – порты, power – мощность.

Синхронное:

* OnFwdSync(byte *outputs*, char *power*, char *turnratio*) – синхронизированное вращение моторов вперед,
* OnRevSync(byte *outputs*, char *power*, char *turnratio*) – синхронизированное вращение моторов назад,

где outputs – порты, power – мощность, turnratio – коэффициент поворота от -100 до 100.

Вращение моторов на заданный угол:

* RotateMotor(byte *outputs*, char *power*, long *angle*) – вращение мотора, на заданный угол,
* RotateMotorEx(byte *outputs*, char *power*, long *angle,* char *turnratio, bool sync, bool stop*) – вращение мотора, на заданный угол,

где outputs – порты, power – мощность, angle – угол в градусах, причем отрицательные значения задают противоположное мощности направление вращения, turnratio – коэффициент поворота от -100 до 100, sync – флаг синхронизации моторов, stop – флаг резкой остановки моторов.

Off(byte *outputs*) – остановка моторов.

# Датчики

## Константы портов

Все порты датчиков также имеют обозначения:

* IN\_1, IN\_2, IN\_3, IN\_4
* S1, S2, S3, S4
* И т.д.

Для настройки датчиков рекомендуется использовать IN обозначения, для получения данных с датчиков S.

## Датчик ультразвука

Для того, чтобы взаимодействовать с датчиком ультразвука, необходимо выполнить следующие функции:

1. SetSensorUltrasonic(const byte &*port*) – настройка порта датчика ультразвука,
2. byte SensorUS(const byte *port*) – получение данных с датчика ультразвука в сантиметрах от 0 до 255.

То есть взаимодействие с датчиком ультразвука можно представить, например, следующим образом:

task main(){

int x = 0;

SetSensorUltrasonic(IN\_1);

x = SensorUS(S1);

}

## Датчик касания

Для того, чтобы взаимодействовать с датчиком касания, необходимо выполнить следующие функции:

1. SetSensorTouch (const byte &*port*) – настройка порта датчика касания,
2. byte Sensor(const byte *port*) – получение данных с датчика.

## Датчик интенсивности света

Для того, чтобы взаимодействовать с датчиком интенсивности света, необходимо выполнить следующие функции:

1. SetSensorLight (const byte &*port*) – настройка порта датчика интенсивности света,
2. byte Sensor(const byte *port*) – получение данных с датчика.

## Датчик цвета

Для того, чтобы взаимодействовать с датчиком цвета, необходимо выполнить следующие функции:

1. SetSensorFullColor (const byte &*port*) – настройка порта датчика цвета,
2. byte Sensor(const byte *port*) – получение данных с датчика.