

Модуль Finger

Модуль библиотеки, для управления приводами пальцев протеза:

- [Вращение мотора](#)
- [Остановка мотора](#)
- [Примеры работы](#)

Синтаксис

```
Finger1.Drive(50)
Finger1.DriveRaw(127)
Finger1.Stop()
Finger1.Break()
Finger1.GetSpeed()
```

Привод пальца протеза представляет собой коллекторный двигатель постоянного тока. Это значит, что мотор работает крайне просто: есть напряжение на контактах - вращается, нет напряжения - не вращается. Чем выше уровень напряжения - тем быстрее вращается, чем ниже - тем медленнее.

Фактически уровень напряжения определяет не скорость вращения мотора, а мощность, которую может развить мотор.

Напряжение и токи, необходимы для работы мотора не могут обеспечиваться выводами микроконтроллера и потому управление моторами происходит посредством специальных микросхем - драйверов.

Для изменения уровня напряжения, подаваемого на контакты мотора, используется широтно-импульсная модуляция (ШИМ) - она обеспечивает высокочастотное включение/выключение подачи питания на контакты мотора, что позволяет получать различные уровни напряжения при усреднении по времени.

Фактически посредством ШИМ мы управляет количеством энергии, передаваемой мотору в единицу времени.

Как правило, для управления мотором в характеристиках ШИМ меняют только скважность - параметр, определяющий длительность импульса.

Для управления ШИМ выходами контроллера Smartli Education мы можем использовать значения от 0 до 255 (один байт или 8 бит). Это значит, что у нас в распоряжении 256 значений

уровней мощности, которые мы можем передать мотору. Как правило такого диапазона достаточно.

В модуле управления Smartli Education предусмотрено подключение пяти приводов:

- Finger1
- Finger2
- Finger3
- Finger4
- Finger5

Все примеры кода будут приводиться для датчика Finger1.

Примеры для остальных датчиков углового положения аналогичны.

Вращение мотора

Для управления скоростью (мощностью) и направлением вращения мотора доступно два метода:

```
void DriveRaw(int spd); // управление сырыми значениями: -255 .. 255
void Drive(int spd); // управление пропорциональными значениями: -100 .. 100
```

Как вы видите, не смотря на то, что ШИМ поддерживает только положительные значения в диапазоне: 0 .. 255, вам доступно указывать в качестве аргумента значения отрицательные. Знак в значении аргумента метода управления вращением мотора определяет направление вращения:

"+" - вращение в одну сторону

"-" - вращение в обратную сторону

"Следите за типами данных"

В вашей программе может возникнуть ситуация, когда значение аргумента функции управления приводом будет вычисляться. При этом, у вас может возникнуть ситуация, когда вычисляемое значение является дробным.

Обратите ваше внимание на то, как работает приведение типов в C/C++.

По умолчанию, при приведении дробного числа к целочисленному типу, дробная часть

просто отбрасывается.

Это значит, что команды:

```
Finger1.Drive(98); // установит фактическое значение скорости: 249  
Finger1.Drive(98.9); // установит фактическое значение скорости: 249
```

Пример

```
Finger1.Drive(90); // Пропорциональное управление приводом - установили мощность вращения привода  
Finger1.DriveRaw(213); // Прямое управление мощностью привода. Установили фактическое значение м
```

Остановка мотора

Модуль управления протезом Smartli Education оснащен специализированным драйвером, позволяющим управлять мощностью вращения электромотора (привода). Используемый драйвер позволяет реализовать два типа остановки мотора:

- Остановка: простое прекращение подачи мощности к приводу, привод продолжает вращение по инерции.
- Торможение: привод останавливается, вращения по инерции не происходит.

```
Finger1.Stop(); // Мягкая остановка  
Finger1.Break(); // Торможение
```

Примеры

Пример тестирования/настройки приводов

```
#include <smartlieduction.h>
```

```
void setup() {  
    Finger1.Stop();  
    Finger2.Stop();  
    Finger3.Stop();  
    Finger4.Stop();  
    Finger5.Stop();  
}
```

```
void loop() {  
    Finger1.Drive(80);  
    delay(1000);  
    Finger1.Drive(-80);  
    delay(1000);  
    Finger1.Stop();  
  
    Finger2.Drive(80);  
    delay(1000);  
    Finger2.Drive(-80);  
    delay(1000);  
    Finger2.Stop();  
  
    Finger3.Drive(80);  
    delay(1000);  
    Finger3.Drive(-80);  
    delay(1000);  
    Finger3.Stop();  
  
    Finger4.Drive(80);  
    delay(1000);  
    Finger4.Drive(-80);  
    delay(1000);  
    Finger4.Stop();  
  
    Finger5.Drive(80);  
    delay(1000);  
    Finger5.Drive(-80);  
    delay(1000);  
    Finger5.Stop();  
}
```