PID регулятор

Павел Пронин
C++ разработчик

Проверка связи

Если у вас нет звука:

- убедитесь, что на вашем устройстве и на колонках включён звук
- обновите страницу вебинара (или закройте страницу и заново присоединитесь к вебинару)
- откройте вебинар в другом браузере
- перезагрузите компьютер (ноутбук) и заново попытайтесь зайти

Поставьте в чат:

\+ если меня видно и слышно

— если нет

Павел Пронин

О спикере:

- Разработчик на C++ более 8-ми лет
- Опыт в разработке беспилотных автомобилей
- С 2022 года разработчик в компании разработки мобильных игр Playrix (компания разрабатывает такие игры как homescapes и gardegscapes)

Вспоминаем прошлое занятие

Вопрос: Какая топология сети реализована в стандарте ZigBEE?

Вспоминаем прошлое занятие

Вопрос: Какая топология сети реализована в стандарте ZigBEE?

Ответ: Mesh

Вспоминаем прошлое занятие

Вопрос: Какой метод в протоколе HTTP используется для чтения данных с сайта?

Вспоминаем прошлое занятие

Вопрос: Какой метод в протоколе HTTP используется для чтения данных с сайта?

Ответ: Метод GET

Вспоминаем прошлое занятие

Вопрос: Что означает код 200 в ответе на запрос GET?

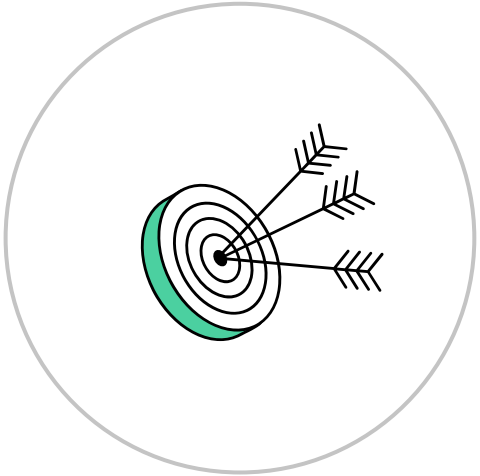
Вспоминаем прошлое занятие

Вопрос: Что означает код 200 в ответе на запрос GET?

Ответ: Данные успешно получены

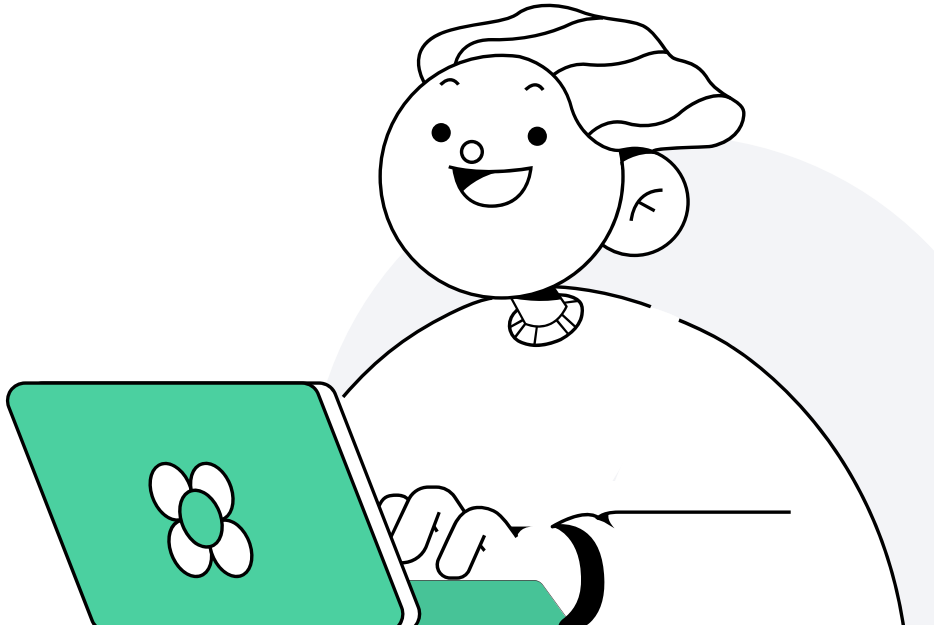
Цели занятия

- Узнаем, как использовать интерфейс 1-WIRE
- Научимся подключать цифровой датчик температур и влажности
- Узнаем, для чего используется ПИД регулятор
- Научимся использовать ПИД регулятор для управления нагревателем



План занятия

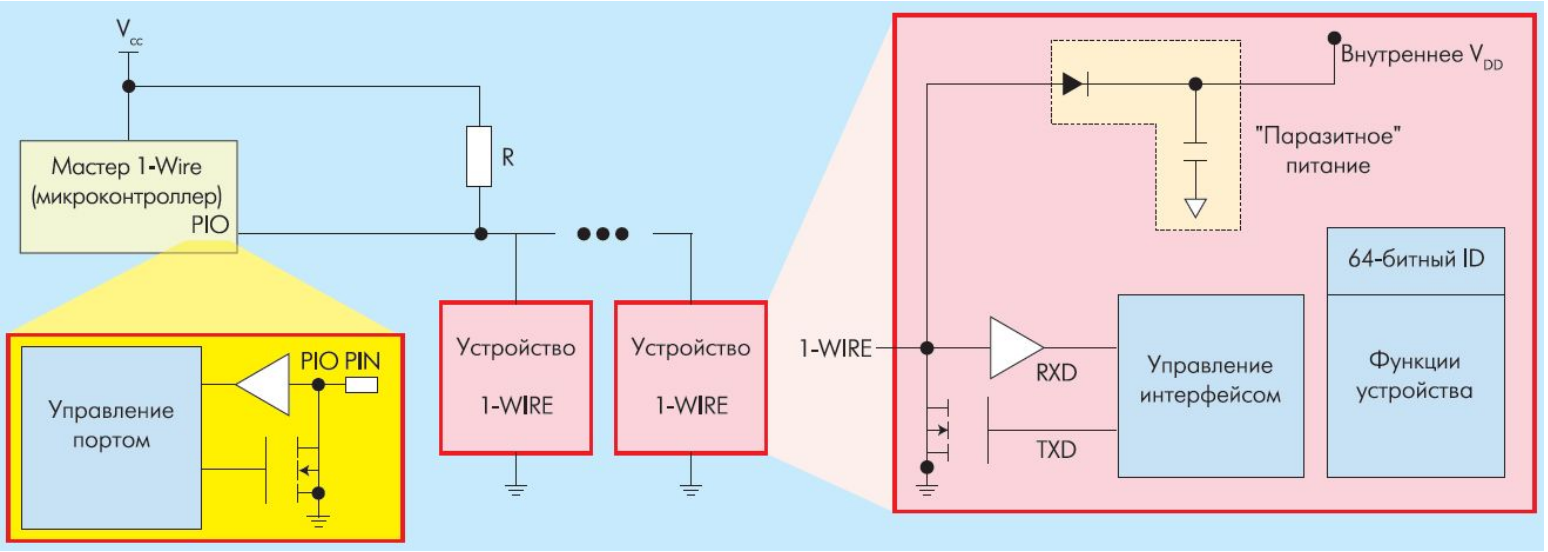
1. Как обрабатывать датчики с цифровым выходом 1-WIRE
2. Как обрабатывать цифровой датчик температуры и влажности
3. Как использовать ПИД регулятор
4. Итоги



Как обрабатывать датчики с цифровым выходом 1-WIRE

Интерфейс 1-WIRE

Интерфейс 1-Wire был предложен фирмой Dallas Semiconductor в конце 90-х годов XX века. Системы 1-Wire привлекательны благодаря легкости монтажа, низкой стоимости устройств, большому числу устройств в сети и т.д



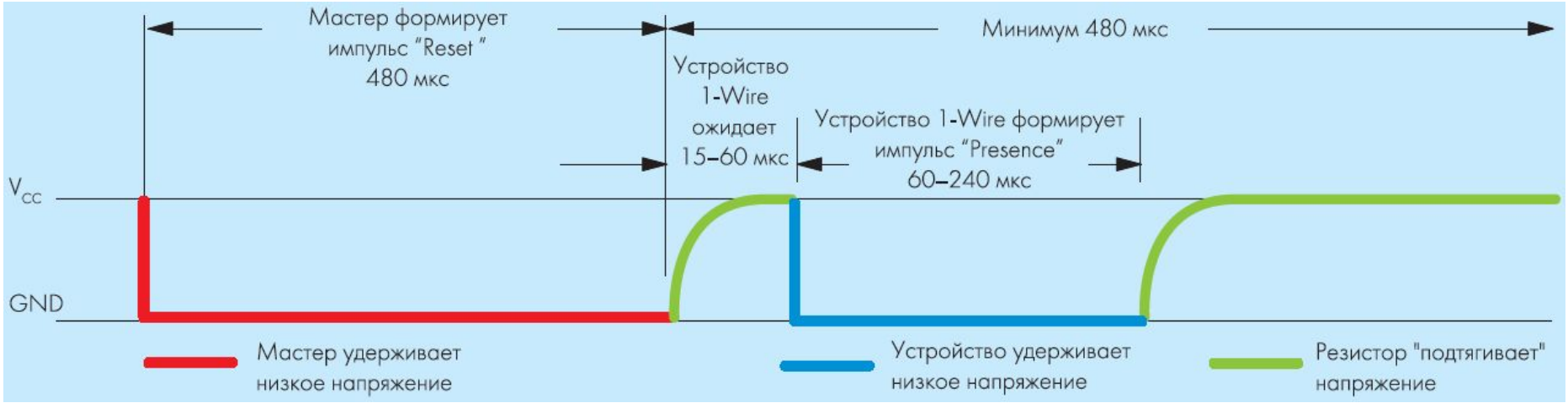
Источник

Особенности интерфейса 1-WIRE

- Система состоит из управляющего контроллера и одного или нескольких ведомых устройств
- Устройства подключаются по схеме с открытым коллектором (стоком) и подтягивающим резистором
- Для подключения ведомых устройств достаточно 2- проводов: сигнальный и общий (GND)
- Некоторые ведомые устройства поддерживают питание от сигнальной линии
- Стандартная скорость передачи данных: 15 кбит/с, повышенная - 111 кбит/с
- Режим передачи данных по шине 1-Wire – полудуплексный: мастер и ведомые устройства передают данные по очереди

Последовательность инициализации интерфейса 1-WIRE

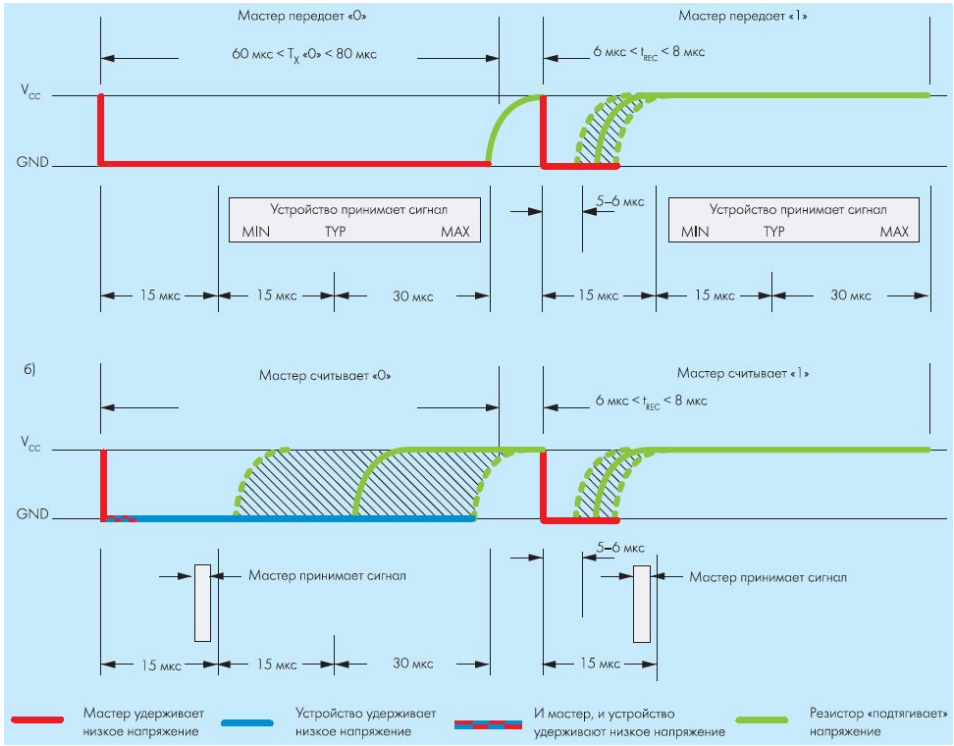
Каждая транзакция через интерфейс 1-Wire начинается с того, что мастер передает импульс Reset. Все ведомые устройства, обнаружив сигнал Reset и дождавшись его окончания, передают свой сигнал – Presence.



Источник

Передача данных по интерфейсу 1-WIRE

Весь информационный обмен в шине происходит под управлением мастера. Для передачи каждого бита выделяется специальный временной промежуток (тайм-слот) длительностью порядка 80 мкс. В начале каждого тайм-слота мастер переводит линию на нулевой уровень.



Источник

Формат пакета данных при обмене по 1-WIRE

Обмен данными по 1-WIRE включает три фазы: фазу сброса, фазу выборки устройства и фазу записи/чтения данных

Источник

Библиотека OneWire для работы с интерфейсом 1-WIRE

OneWire(uint8_t pin) — конструктор, создающий объект OneWire

Параметры:

- pin: номер вывода, к которому подключаются внешние устройства

Возвращаемое значение: нет

uint8_t reset(void) — формирование сброса на линии 1-WIRE

Параметры: нет
Возвращаемое значение: 1, если устройство подтвердило наличие импульса присутствия, 0 в противном случае

Библиотека OneWire для работы с интерфейсом 1-WIRE

void select(const uint8_t rom[8]) — выбирает устройство, чей адрес указан в скобках. Используется после сброса, чтобы выбрать устройство, с которым необходимо вести обмен. Последующая коммуникация будет именно с этим устройством, но до нового сброса.

Параметры:

- rom: указатель на массив, содержащий адрес устройства

Возвращаемое значение: нет

void skip(void) — выполняет команду ROM SKIP на линии 1-WIRE, чтобы обратиться ко всем устройствам на шине

Параметры: нет
Возвращаемое значение: нет

Библиотека OneWire для работы с интерфейсом 1-WIRE

void write(uint8_t v, uint8_t power = 0) — записывает байт в ведомое устройство
Параметры:

- v: записываемое значение
- power: состояние линии после записи: 0 или 1

Возвращаемое значение: нет

void write_bytes(const uint8_t *buf, uint16_t count, bool power = 0) — записывает несколько байтов в ведомое устройство
Параметры:

- buf: указатель на массив записываемых данных
- count: количество записываемых байтов
- power: состояние линии после записи: 0 или 1

Возвращаемое значение: нет

Библиотека OneWire для работы с интерфейсом 1-WIRE

uint8_t read(void) — считывает байт из ведомого устройства

Параметры: нет
Возвращаемое значение: прочитанный байт

void read_bytes(uint8_t *buf, uint16_t count) — считывает несколько байтов из ведомого устройства

Параметры:

- buf: указатель на массив, куда необходимо поместить считанные данные
- count: количество считываемых байтов

Возвращаемое значение: нет

Библиотека OneWire для работы с интерфейсом 1-WIRE

void reset_search(void) — инициализирует новый поиск, следующее использование поиска начнется на первом устройстве

Параметры: нет
Возвращаемое значение: нет

void target_search(uint8_t family_code) — настраивает поиск на определенный тип устройства

Параметры:

- family_code: код устройства из стандартного перечня

Возвращаемое значение: нет

Библиотека OneWire для работы с интерфейсом 1-WIRE

uint8_t search(uint8_t *newAddr) — запуск нового поиска.

Параметры:

- newAddr: указатель на массив, куда записывается новый адрес устройства

Возвращаемое значение: 1 - новый адрес получен, 0 - нет нового адреса

static uint8_t crc8(const uint8_t *addr, uint8_t len) — рассчитывает 8-и разрядную контрольную сумму для массива данных

Параметры:

- addr: указатель на массив данных
- len: длина массива

Возвращаемое значение: рассчитанная контрольная сумма

Ваши вопросы?

Как обрабатывать цифровой датчик температуры и влажности

2

Датчик температуры и влажности DHT-22

Основные характеристики:

- Напряжение питания: 3 .. 6 В
- Диапазон измерения влажности: 0 .. 100%
- Диапазон измерения температуры: -40℃ ..+80℃
- Погрешность измерений влажности: 2%
- Погрешность измерений температуры: 0.5°C
- Шаг измерения влажности: 0.1%
- Шаг измерения температуры: 0.1 °C
- Период измерений: 1 сек.

Источник

Библиотека DHT для работы с датчиком DHT22

DHT(uint8_t pin, uint8_t type, uint8_t count = 6) — конструктор, создающий объект DHT
Параметры:

- pin: номер вывода, к которому подключается датчик
- type: тип датчика: DHT11, DHT12, DHT22, DHT21
- count: количество датчиков на линии

Возвращаемое значение: нет

void begin(uint8_t usec = 55) — инициализирует объект DHT
Параметры:

- usec: задержка перед чтение значения датчика (в мкс)

Возвращаемое значение: нет

Библиотека DHT для работы с датчиком DHT22

float readTemperature(bool S = false, bool force = false)— возвращает значение температуры, полученное от датчика
Параметры:

- S: false - по шкале Цельсия, true - по шкале Фаренгейта
- force: false - возвращает данные, прочитанные не позднее 2 секунд назад, true - всегда запрашивать новые данные

Возвращаемое значение: значение температуры

float readHumidity(bool force = false) — возвращает значение влажности, полученное от датчика
Параметры:

- force: false - возвращает данные, прочитанные не позднее 2 секунд назад, true - всегда запрашивать новые данные

Возвращаемое значение: значение влажности

Библиотека DHT для работы с датчиком DHT22

```
float computeHeatIndex(float temperature, float percentHumidity,
                       bool isFahrenheit = true);
```

— возвращает вычисленное значение теплового индекса, используя уравнения Ротфуза и Стедмана

Параметры:

- temperature: значение температуры
- percentHumidity: значение влажности
- isFahrenheit: false - по шкале Цельсия, true - по шкале Фаренгейта

Возвращаемое значение: значение теплового индекса

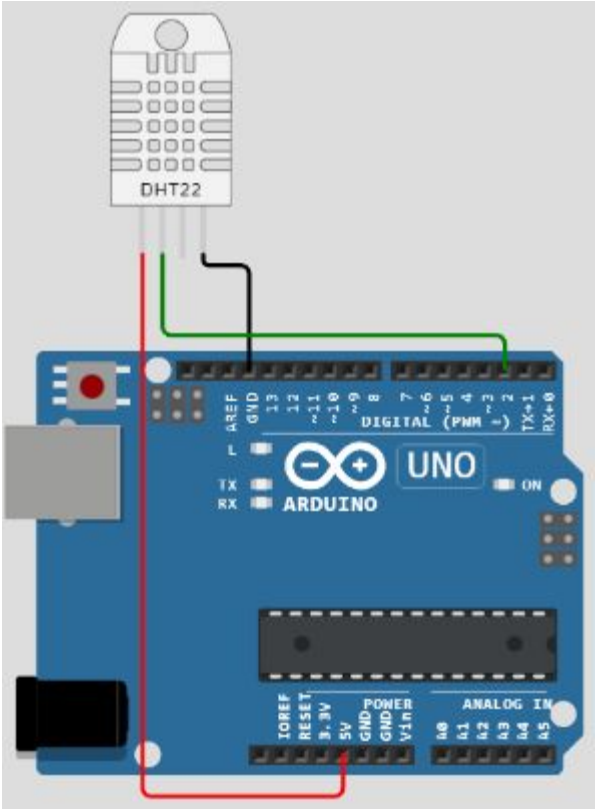
Считывание данных с цифрового датчика температуры и влажности

```
#include "DHT.h"
#define DHTTYPE DHT22   //тип датчика
const int DHTPin = 2; //номер вывода для подключения датчика
DHT dht(DHTPin, DHTTYPE); //объект DHT
void setup()
{
  Serial.begin(9600);
  Serial.println("DHT22 test!");
  dht.begin(); //Инициализация
}

void loop()
{
  // опрос датчика раз в 2 секунды
  delay(2000);

  // чтение показаний
  float h = dht.readHumidity();
  float t = dht.readTemperature();
  float f = dht.readTemperature(true);
```

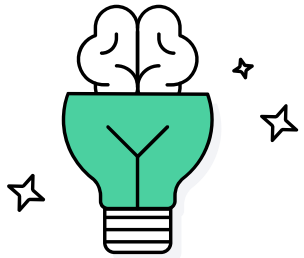
```
// Если считано не число, то ошибка!
  if (isnan(h) || isnan(t) || isnan(f))
  {
    Serial.println(F("Failed to read from DHT sensor!"));
    return;
  }
  // вычисление индекса температуры
  float hif = dht.computeHeatIndex(f, h);
  float hic = dht.computeHeatIndex(t, h, false);
  Serial.print(F("Humidity: "));
  Serial.print(h);
  Serial.print(F("%  Temperature: "));
  Serial.print(t);
  Serial.print(F("°C "));
  Serial.print(f);
  Serial.print(F("°F  Heat index: "));
  Serial.print(hic);
  Serial.print(F("°C "));
  Serial.print(hif);
  Serial.println(F("°F"));
}
```

Считывание данных с цифрового датчика температуры и влажности

В примере используется датчик DHT22



Практическое задание №1



Практика: считывание данных с цифрового датчика температуры и влажности

Задание:

1) соберите схему в симуляторе WOKWI, подключив датчик к выводу 2;
2) создайте скетч с текстом, приведенным выше;
3) проведите моделирование работы

Как выполнять: напишите в чат об удачной работе схемы

Время выполнения: 5 минут

Ваши вопросы?

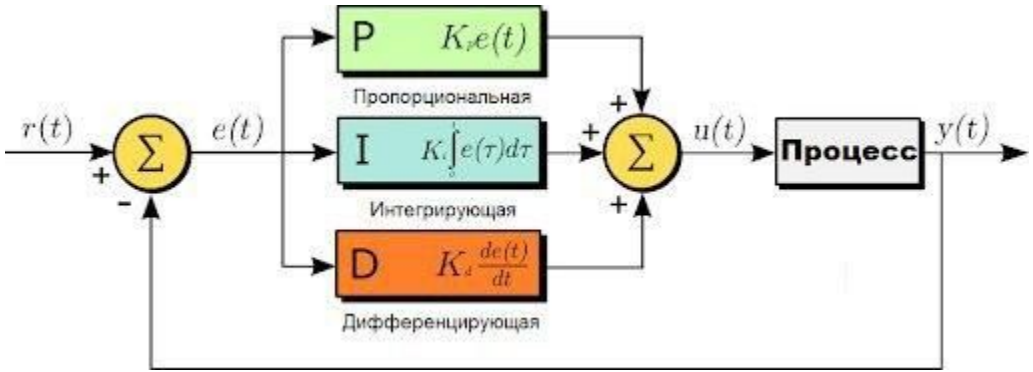
Как использовать ПИД регулятор

3

Что такое ПИД-регулятор

Пропорционально-интегрально-дифференциальный регулятор (ПИД) – устройство для автоматического поддержания в заданном интервале одного или нескольких параметрах. Такие устройства универсальны, при помощи ПИД-регуляторов можно реализовать любые законы регулирования.

ПИД регулятор был изобретён ещё в 1910 году, однако активно использоваться начала только в 1980-х годах.



Источник

Математическое описание работы ПИД-регулятора

Выходная величина регулятора описывается формулой:

$$u(t) = Ke(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt},$$

где t – время, а K, T_i, T_d – пропорциональный коэффициент, постоянная интегрирования и постоянная дифференцирования соответственно.

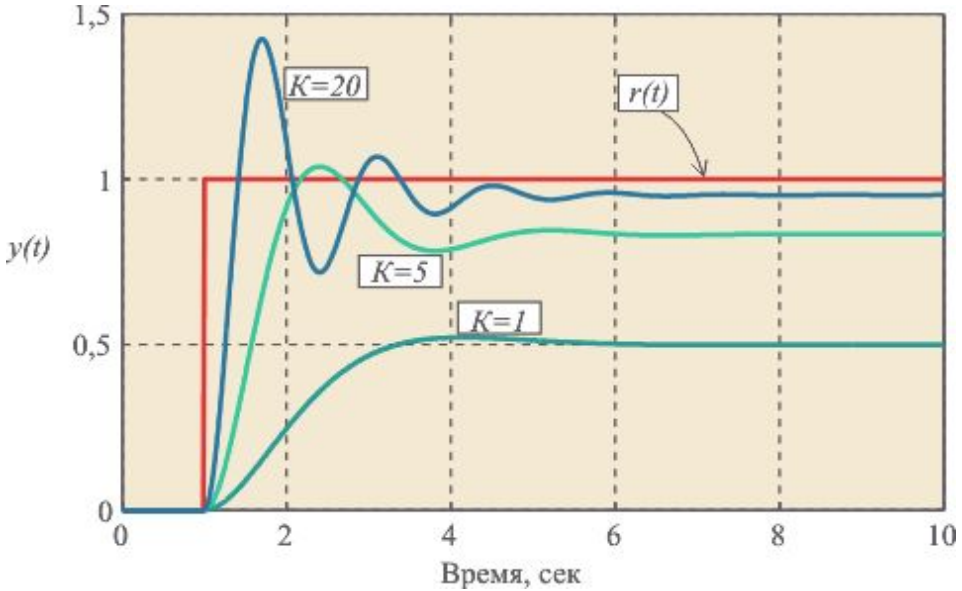
Распространены также следующие модификации выражения:

$$u(t) = K_o\left(e(t) + \frac{1}{T_i}\int_0^t e(t)dt + T_d\frac{de(t)}{dt}\right),$$

$$u(t) = ke(t) + k_i\int_0^t e(t)dt + k_d\frac{de(t)}{dt},$$

Пропорциональная составляющая

При малых *K* система имеет малое перерегулирование, но большую статическую погрешность (50%). С ростом погрешность уменьшается, но возрастает перерегулирование.

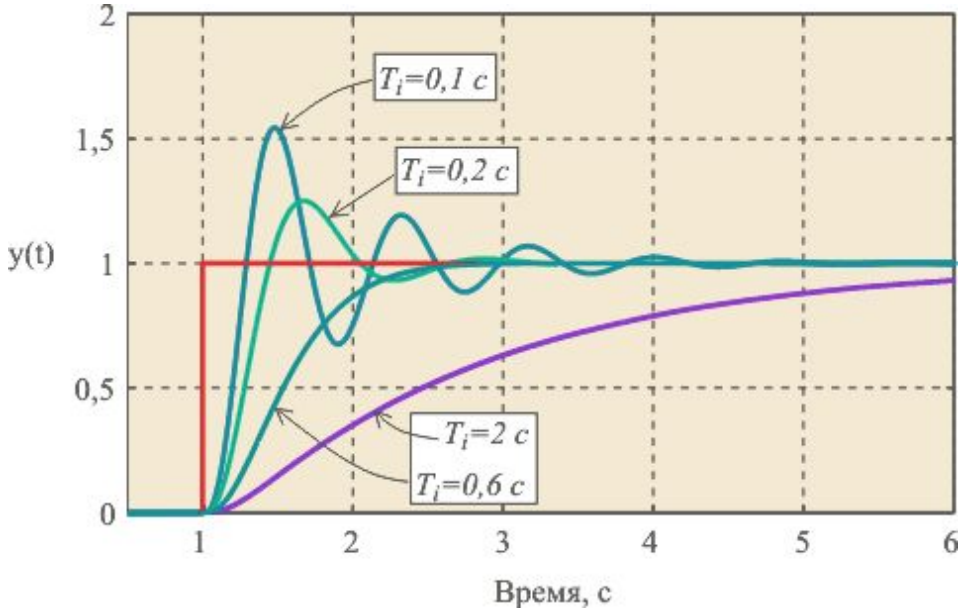


Источник

Интегральная составляющая

Система с И-регулятором не имеет ошибки в установившемся режиме.

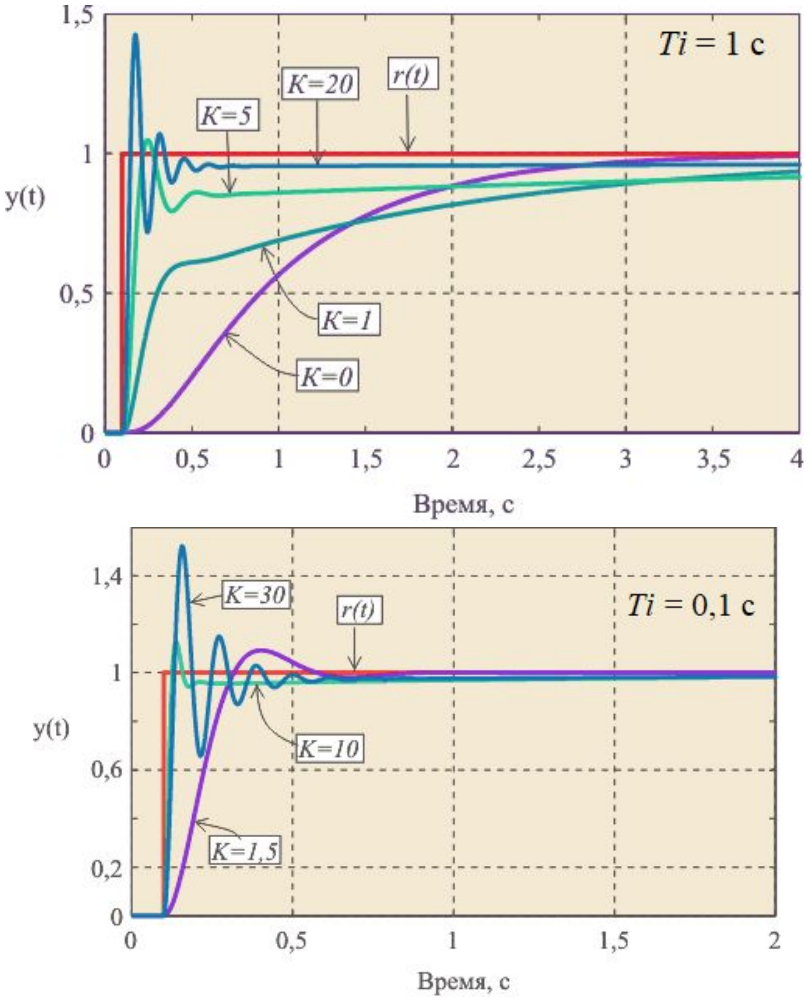
При больших постоянных интегрирования переходная характеристика не имеет выбросов. С уменьшением растет усиление регулятора и в системе появляются колебания



Источник

Пропорционально-интегральный регулятор

Когда *K* не равно нулю появляется дополнительная ошибка во время переходного процесса, которая уменьшается с ростом *K*, однако при этом снижается запас устойчивости системы. Это приводит к появлению затухающих колебаний в начале переходного процесса. Когда величина *K* становится достаточно большой для компенсации ослабления сигнала в объекте на частоте , в системе появляются незатухающие колебания

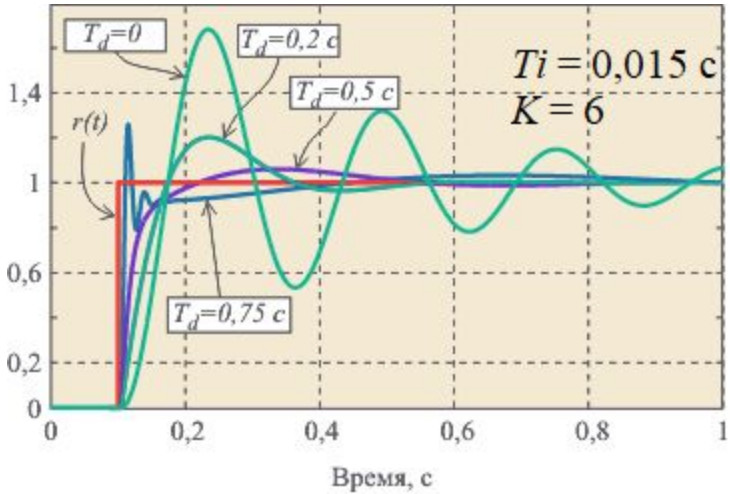


Источник

ПИД регулятор

Дифференциальный член позволяет обеспечить устойчивость или улучшить качество регулирования системы в случаях, когда это невозможно сделать с помощью ПИ-регулятора.

Однако при большом значении T_d схема может перейти в колебательный режим



Источник

Настройка ПИД-регулятора

Настройка ПИД-регулятора осуществляется двумя способами:

1. На основе математической модели системы. Этот способ точен, но требует глубокого понимания теории автоматического управления, а также наличия адекватной модели системы.
2. Ручная настройка. Для этого за основу берутся данные готовой системы, вносятся изменения в один или несколько коэффициентов регулятора. После включения и наблюдения за конечным результатом параметры меняются в желаемом направлении. И так до тех пор, пока не будет достигнут желаемый уровень производительности.

Теоретический метод анализа и настройки на практике используется очень редко, что связано с незнанием характеристик объекта управления и массой возможных мешающих воздействий. Чаще встречаются экспериментальные методы, основанные на наблюдении за системой.

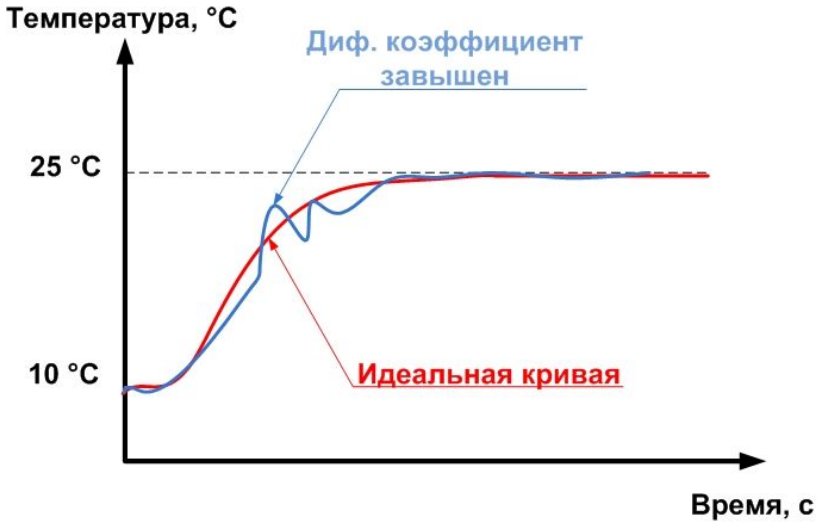
Настройка пропорционального коэффициента

При большом перерегулировании, необходимо уменьшать пропорциональный коэффициент, а если регулятор долго достигает уставки — увеличивать. Лучше оставить небольшое перерегулирование (его можно будет скорректировать другими коэффициентами), чем длительное нарастание графика.

Источник

Настройка дифференциального коэффициента

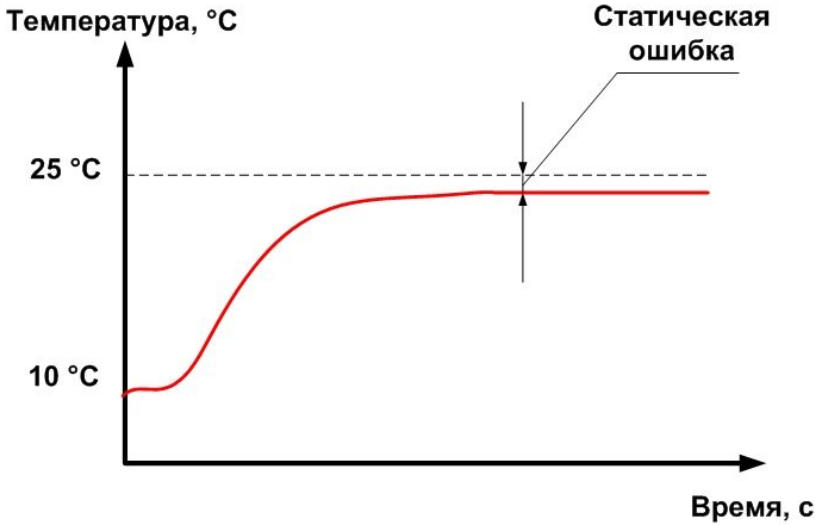
Постепенно увеличивая дифференциальную составляющую, необходимо добиться уменьшения или полного исчезновения «скачков» графика (перерегулирования) перед выходом на уставку. При этом кривая должна стать еще больше похожа на идеальную. Если слишком сильно завысить дифференциальный коэффициент, температура при выходе на уставку будет расти не плавно, а скачками (как показано на рисунке).



Источник

Настройка интегрального коэффициента

Для исключения статической ошибки используют интегральную составляющую. Её необходимо постепенно увеличивать до исчезновение статической ошибки. Однако, чрезмерное её увеличение может привести к возникновению скачков температуры.



Источник

Библиотека PID_v1 для работы с ПИД регулятором

PID(double* Input, double* Output, double* Setpoint,
double Kp, double Ki, double Kd, int POn, int ControllerDirection)
— конструктор, создающий объект PID

Параметры:

- Input, Output, Setpoint: указатели на вход, выход и уставку регулятора соответственно
- Kp, Ki, Kd: коэффициенты для пропорциональной, интегральной и дифференциальной составляющей
- POn: (необязательный параметр) P_ON_E (по умолчанию) - разрешена работа регулятора, P_ON_M - “ручное” управление по входному сигналу
- ControllerDirection: знак изменения выхода относительно ошибки - DIRECT или REVERSE

Возвращаемое значение: нет

Библиотека PID_v1 для работы с ПИД регулятором

bool Compute() — реализует вычисления ПИД регулятора с периодом, заданным SetSampleTime (), следует вызывать в voop()

Параметры: нет
Возвращаемое значение: true - были проведены вычисления, false - вычислений не было

void SetSampleTime(int NewSampleTime) — задает период вычисления алгоритма ПИД регулятора

Параметры:

- NewSampleTime: период вычисления в мс (по умолчанию 200 мс)

Возвращаемое значение: нет

Библиотека PID_v1 для работы с ПИД регулятором

void SetOutputLimits(double Min, double Max) — задает диапазон изменения выхода ПИД регулятора

Параметры:

- Min, Max: минимальное и максимальное значение выхода (по умолчанию от 0 до 255)

Возвращаемое значение: нет

void SetMode(int Mode) — задает режим работы ПИД регулятора

Параметры:

- Mode: автоматический или ручной режим: AUTOMATIC или MANUAL

Возвращаемое значение: нет

Управление нагревом металлической пластины

```
//подключение библиотеки ПИД регулятора
#include "PID_v1.h"
// номера выводов
const int PWMPin = 3;
const int inputPin = -1;
const int setPointPin = A1;
const int setPointIndicator = 6;
const int inputIndicator = 5;

//Переменные для ПИД регулятора
double Setpoint, Input, Output;

//Параметры ПИД регулятора
double Kp = 17, Ki = 0.3, Kd = 2; // тестовые параметры
//double Kp = 255, Ki = .0, Kd = 0; // пропорциональный регулятор
//double Kp = 2, Ki = 5, Kd = 1; // обычно используемые по умолчанию
//Объект ПИД регулятора
PID myPID(&Input, &Output, &Setpoint, Kp, Ki, Kd, P_ON_E, DIRECT);
```

```
void setup()
{
  Serial.begin(115200);
  myPID.SetOutputLimits(-4, 255); //диапазон выхода
  //индикаторы для отладки
  if (setPointIndicator >= 0) pinMode(setPointIndicator,
OUTPUT);
  if (inputIndicator >= 0) pinMode(inputIndicator, OUTPUT);
  Setpoint = 0; //начальное значение уставки - ноль
  myPID.SetMode(AUTOMATIC); //включение регулятора
  if(inputPin>0) //выбор источника входного сигнала
  {
    Input = analogRead(inputPin);
  }else{
    Input = simPlant(0.0,1.0); //внутренняя модель входного
сигнала
  }
  Serial.println("Setpoint Input Output Watts");
}
```

Управление нагревом металлической пластины

```
void loop()
{
  float heaterWatts = (int)Output * 20.0 / 255; //мощность нагревателя для модели
  if (inputPin > 0 )
  {
    Input = analogRead(inputPin);
  } else
  {
    Input = simPlant(heaterWatts,Output>0?1.0:1-Output); //моделирование нагрева
  }

  if (myPID.Compute()) //ПИД регулятор
  {
    analogWrite(PWMPin, (int)Output); //выход на нагреватель
    Setpoint = analogRead(setPointPin) / 4; // чтение уставки температуры (от 0 до 255)
    if (inputIndicator >= 0) analogWrite(inputIndicator, Input); // для отладки
    if (setPointIndicator >= 0) analogWrite(setPointIndicator, Setpoint);
  }
  report(); //вывод отладочных данных в com порт
}
```

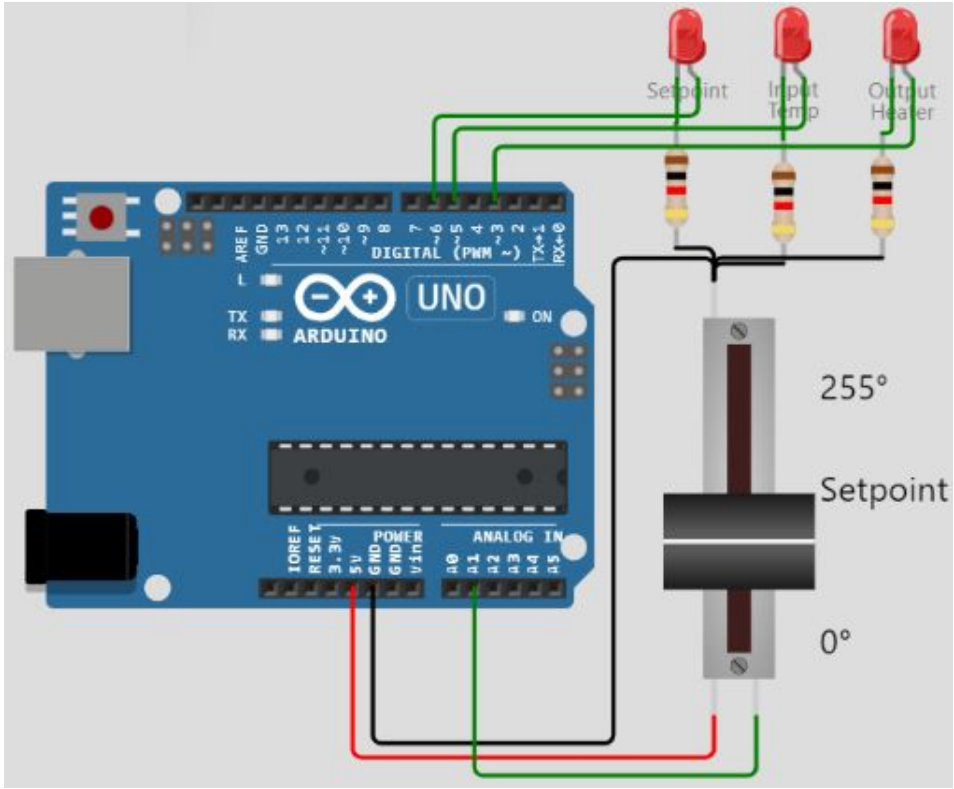
```
/*Функция вывода результата работы ПИД-регулятора*/
void report(void)
{
  static uint32_t last = 0;
  const int interval = 1000; //вывод раз в секунду
  if (millis() - last > interval)
  {
    last += interval;
    Serial.print("SP:");
    Serial.print(Setpoint);
    Serial.print(" PV:");
    Serial.print(Input);
    Serial.print(" CV:");
    Serial.print(Output);
    Serial.print(' ');
    Serial.println();
  }
}
```

Управление нагревом металлической пластины

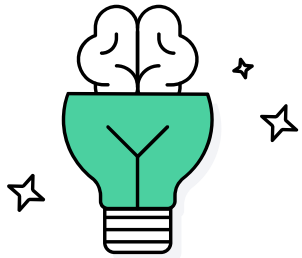
```
/*Функция модели нагрева
Q - мощность [Вт],
hfactor - относительный коэффициент теплопроводности*/
float simPlant(float Q,float hfactor)
{
  //Имитация алюминиевого блока размером 1x1x2 см с нагревателем и пассивным охлаждением
  float h = 5 *hfactor ; // Вт/м2К коэффициент конвекции
  float Cps = 0.89; // теплоемкость Дж/грамм°С
  float area = 1e-4; // площадь конвекции, м2
  float mass = 1 ; // масса, граммы
  float Tamb = 25; // температура окружающей среды °С
  static float T = Tamb; // текущая температура °С
  static uint32_t last = 0; // время для моделирования
  uint32_t interval = 100; // период обновления данных, мс

  if (millis() - last >= interval)
  {
    last += interval;
    T = T + Q * interval / 1000 / mass / Cps - (T - Tamb) * area * h;
  }
  return T;
}
```

Управление нагревом металлической пластины

Уставка задается внешним потенциометром



Практическое задание №2



Практика: управление нагревом металлической пластины

Задание:

1) соберите схему в симуляторе WOKWI, подключив переменный резистор ко входу A1, а выход управления и сигналов входа датчика и уставки - к светодиодам ;
2) создайте скетч с текстом, приведенным выше;
3) проведите моделирование работы

Как выполнять: напишите в чат об удачной работе схемы

Время выполнения: 5 минут

Итоги

4

Итоги занятия

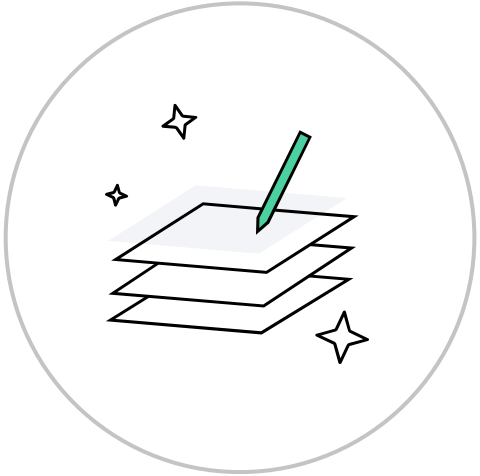
Сегодня мы

1. Узнали особенности применения интерфейса 1-WIRE
2. Научились подключать цифровой датчик температуры и влажности
3. Узнали, для чего используется ПИД регулятор
4. Научились использовать ПИД регулятор для управления нагревом

Домашнее задание

Давайте посмотрим ваше домашнее задание.

1. Вопросы по домашней работе задавайте в чате группы
2. Задачи можно сдавать по частям
3. Зачёт по домашней работе ставят после того, как приняты все задачи



Задавайте вопросы и пишите отзыв о лекции

Павел Пронин
C++ разработчик