Дистанционное управление по инфракрасному каналу

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: для чего используется H-Мост?

Вспоминаем прошлое занятие

Вопрос: для чего используется H-Мост?

Ответ: позволяет изменять направление электрического тока в нагрузке

Вспоминаем прошлое занятие

Вопрос: в чем особенность микрошагового режима?

Вспоминаем прошлое занятие

Вопрос: в чем особенность микрошагового режима?

Ответ: обмотки шагового двигателя в каждый момент времени запитаны не полным током, а его уровнями, изменяющимися по закону sin в одной фазе и cos во второй

Вспоминаем прошлое занятие

Вопрос: какой вид сигнала используется для управления цифровым сервоприводом?

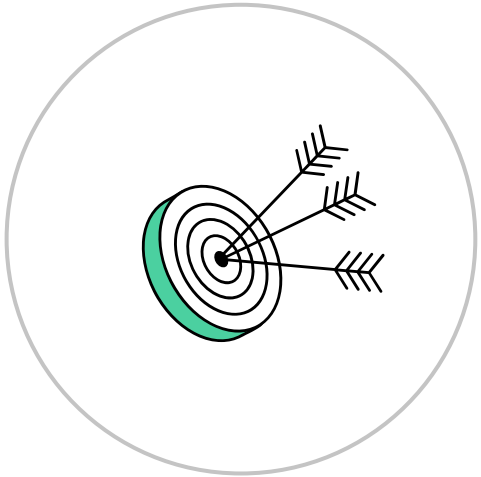
Вспоминаем прошлое занятие

Вопрос: какой вид сигнала используется для управления цифровым сервоприводом?

Ответ: широтно-импульсная модуляция

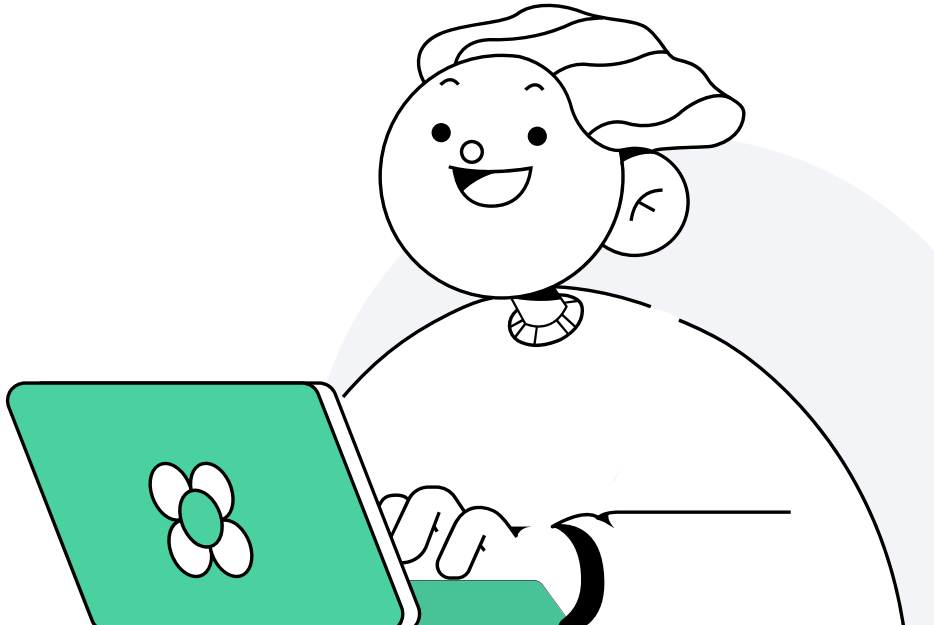
Цели занятия

- Узнаем, какие бывают протоколы для дистанционного управления по инфракрасному каналу
- Научимся принимать данные по инфракрасному каналу и декодировать их с помощью библиотеки
- Научимся использовать внешнее прерывание
- Научимся использовать прерывание от таймера



План занятия

1. Как использовать инфракрасный канал для передачи данных
2. Как использовать внешние прерывания
3. Как использовать прерывания от таймеров
4. Итоги



Как использовать инфракрасный канал для передачи данных

1

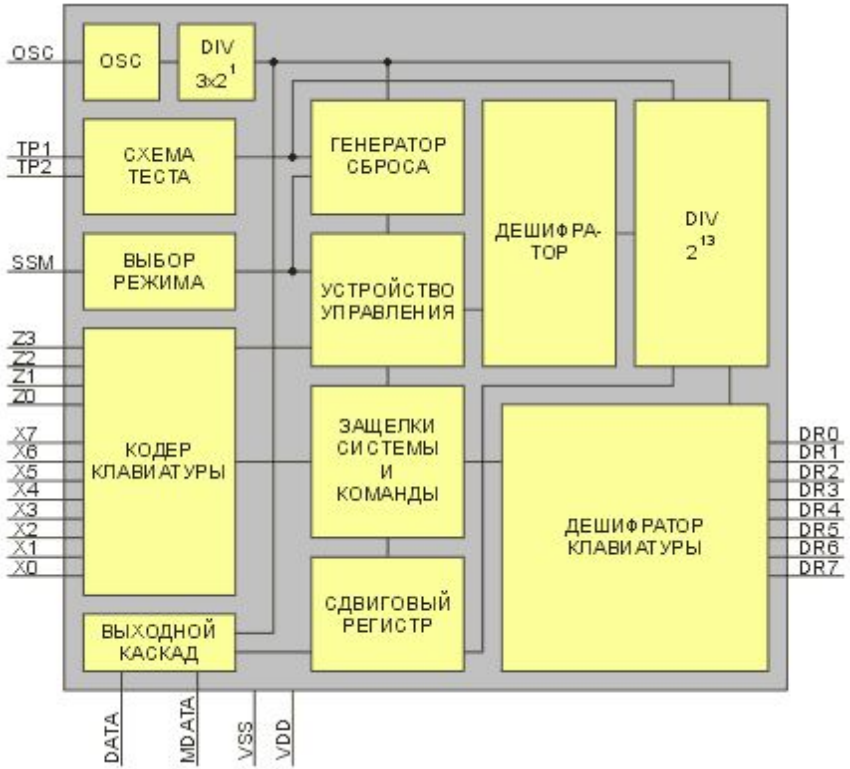
Состав оборудования для передачи

Пульт ИК управления при нажатии кнопки излучает кодированную посылку, а приемник, установленный в управляемом устройстве, принимает её и выполняет требуемые действия

Источник

Структура передатчика

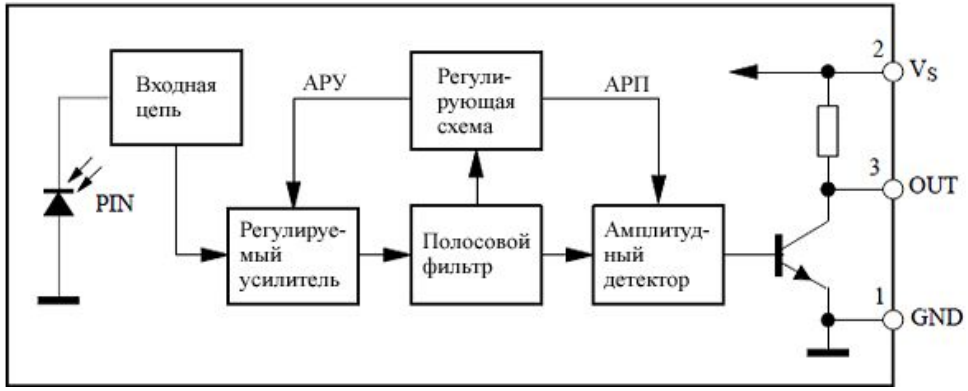
Микросхема передатчика опрашивают матрицу клавиатуры и формирует номер нажатой клавиши

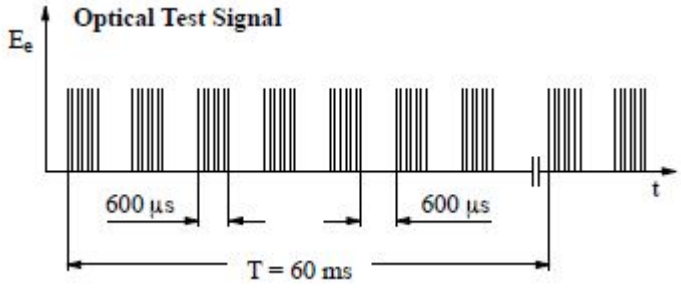


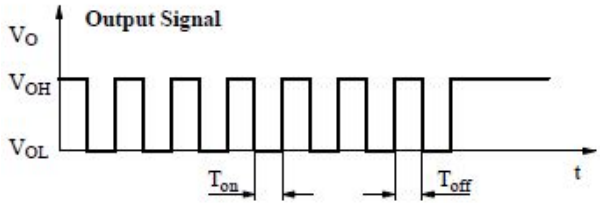
Источник

Структура приемника

Приемник преобразует модулированный сигнал ИК диапазона и производит его амплитудное детектирование



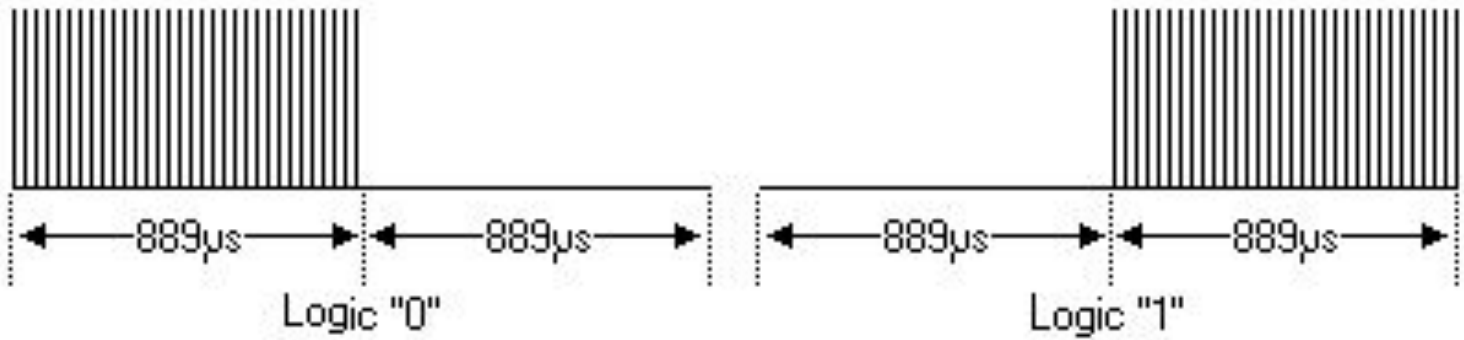




Источник

Протокол Philips RC5

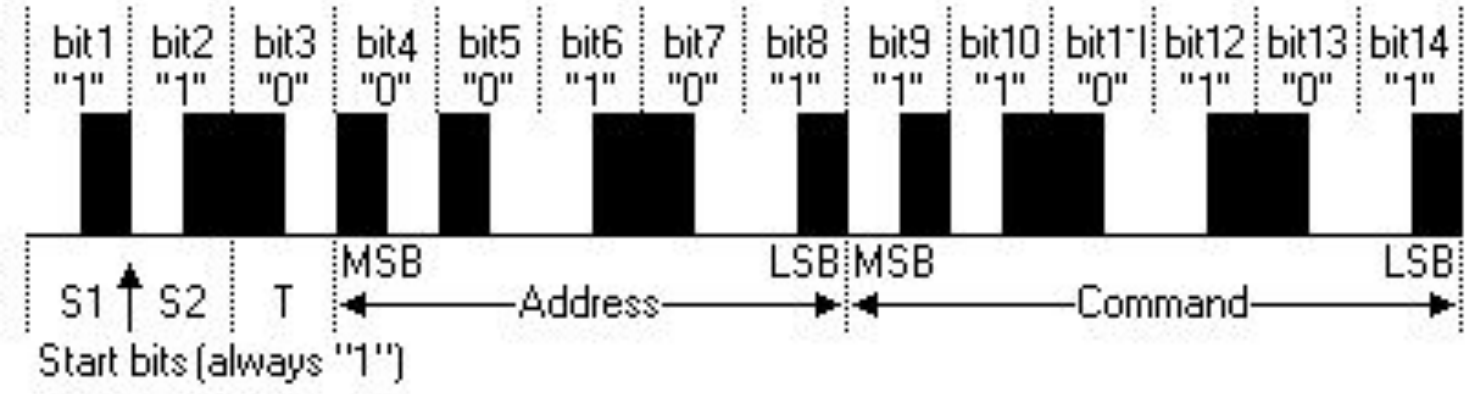
Использует двухфазную модуляцию (Манчестерское кодирование) несущей частоты на 36 кГц. Логический ноль представлен пакетом в первой половине времени передачи бита. Логическая единица представлена пакетом во второй половине времени передачи бит



Источник

Протокол Philips RC5

В начале посылки передается два стартовых бита, имеющих значение "1". Третий бит (Toggle) меняет свое состояние при каждом следующем нажатии кнопки. Если кнопка удерживается, то посылка повторяется каждые 114 мс, при этом Toggle бит не меняется. Таким образом определяется различие между нажатием одной кнопки несколько раз и удерживанием.



Источник

Протокол Philips RC5

Протокол имеет набор стандартных команд, что позволяет обеспечить совместимость устройств, выпущенных в разные годы. Кроме того, при наличии двух однотипных устройств, протокол дает возможность управлять ими независимо, для чего в перечне присутствуют некоторые повторы

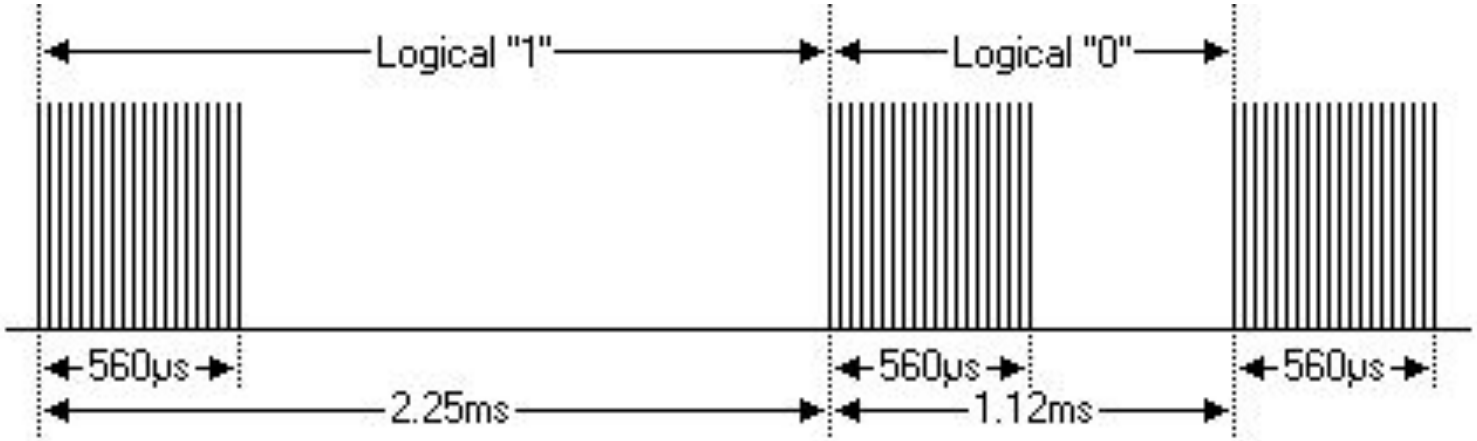
[Источник]

RC-5 Address	Device
$00 - 0	TV1
$01 - 1	TV2
$02 - 2	Teletext
$03 - 3	Video
$04 - 4	LV1
$05 - 5	VCR1
$06 - 6	VCR2
$07 - 7	Experimental
$08 - 8	Sat1
$09 - 9	Camera
$0A - 10	Sat2
$0B - 11	
$0C - 12	CDV
$0D - 13	Camcorder
$0E - 14	
$0F - 15	
$10 - 16	Pre-amp
$11 - 17	Tuner
$12 - 18	Recorder1
$13 - 19	Pre-amp
$14 - 20	CD Player
$15 - 21	Phono
$16 - 22	SatA
$17 - 23	Recorder2
$18 - 24	
$19 - 25	
$1A - 26	CDR
$1B - 27	
$1C - 28	
$1D - 29	Lighting
$1E - 30	Lighting
$1F - 31	Phone

RC-5 Command	TV Command	VCR Command
$00 - 0	0	0
$01 - 1	1	1
$02 - 2	2	2
$03 - 3	3	3
$04 - 4	4	4
$05 - 5	5	5
$06 - 6	6	6
$07 - 7	7	7
$08 - 8	8	8
$09 - 9	9	9
$0A - 10	-/--	-/--
$0C - 12	Standby	Standby
$0D - 13	Mute	
$10 - 16	Volume +	
$11 - 17	Volume -	
$12 - 18	Brightness +	
$13 - 19	Brightness -	
$20 - 32	Program +	Program +
$21 - 33	Program -	Program -
$32 - 50		Fast Rewind
$34 - 52		Fast Forward
$35 - 53		Play
$36 - 54		Stop
$37 - 55		Recording

Протокол NEC

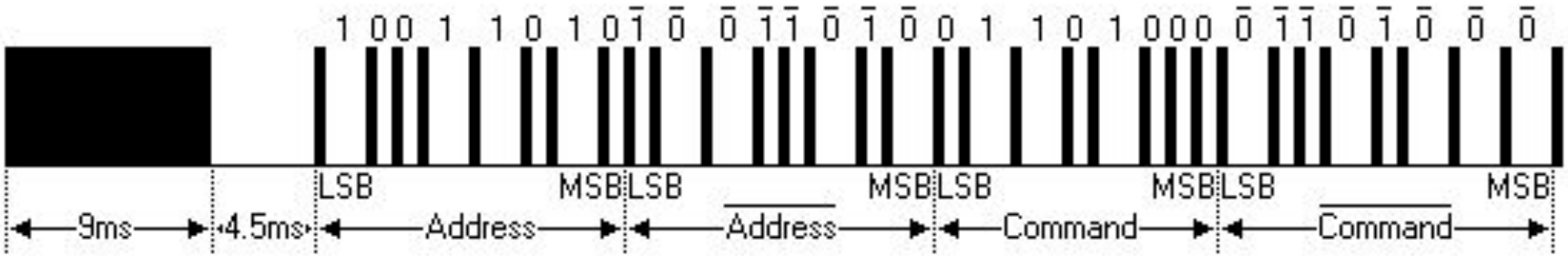
Использует для передачи информации опорную частоту 38 кГц. Все пакеты, кроме преамбулы, имеют длительность 560 мкс (21 импульс опорной частоты). "Единица" передается интервалом 2.25 мс, "ноль" - интервалом 1.125 мс



Источник

Протокол NEC

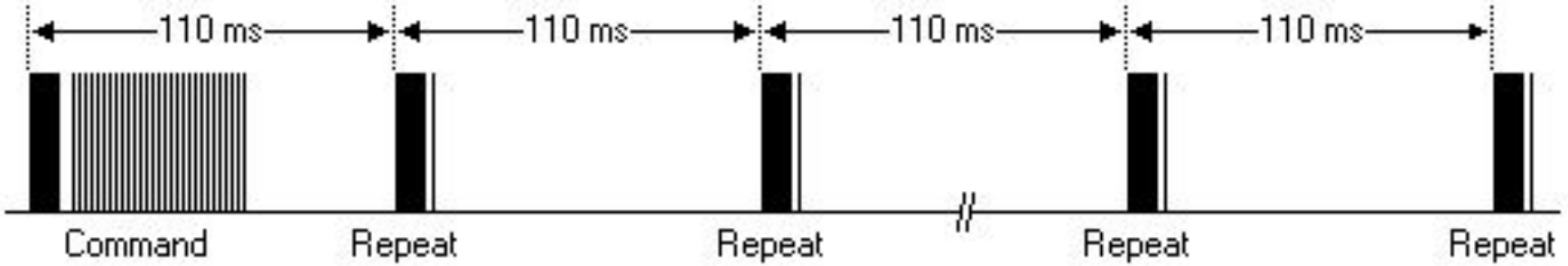
Сообщение начинается пакетом АРУ на 9 мс, который используется, чтобы установить усиление усилителя приемника. Пакет АРУ сопровождается задержкой на 4,5 мс, после которого следует адрес и команда. Адрес и команда передаются дважды. Во второй раз все биты инвертированы и могут использоваться для проверки полученного сообщения.



Источник

Протокол NEC

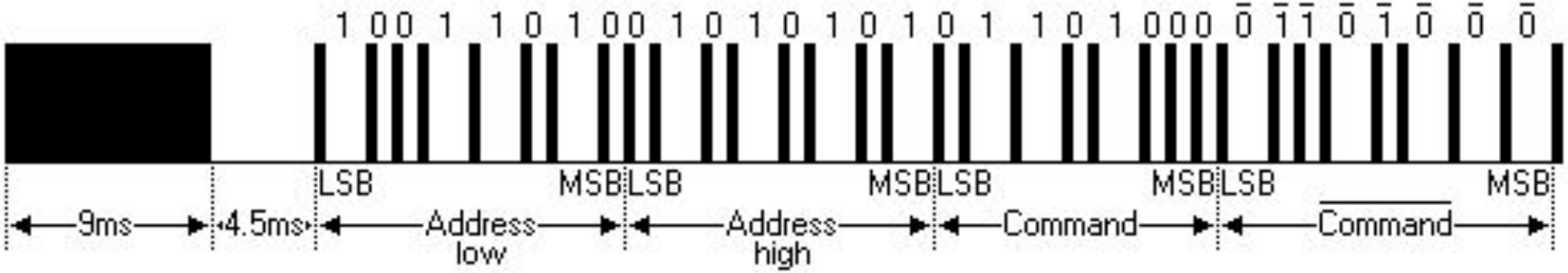
Команда передается только один раз, даже когда кнопка на пульте дистанционного управления остается нажатой. Каждый повторный код 110 мс передается до тех пор, пока кнопка пульта нажата. Этот повторный код - просто импульс АРУ на 9 мс, сопровождаемый задержкой на 2.25 мс и 560 мкс пакетами



Источник

Расширенный протокол NEC

Протокол NEC так широко используется, поэтому все возможные адреса были "израсходованы". Жертвуя избыточностью адреса диапазон адресов был расширен от 8 битов до 16 битов, не изменяя никакие другие свойства протокола.



Источник

Библиотека IRremote для работы с передачей данных по ИК

Библиотека имеет два класса: приемник и передатчик. Каждый из них имеет несколько десятков методов. Для работы используется таймер Timer 2 микроконтроллера. Далее приводятся некоторые методы для класса IRrecv (приемник).

IRrecv(int recvpin) — конструктор, создающий объект IRrecv

Параметры:

- recvpin: номера выводов для подключения приемника

Возвращаемое значение: нет

Библиотека IRremote для работы с передачей данных по ИК

void enableIRIn() — разрешает работу приемника

Параметры: нет
Возвращаемое значение: нет

void disableIRIn() — запрещает работу приемника

Параметры: нет
Возвращаемое значение: нет

Библиотека IRremote для работы с передачей данных по ИК

bool decode() — декодирует входной пакет данных

Параметры: нет
Возвращаемое значение: false - данные не готовы, true - данные готовы, результат сохранен в полях объекта

void resume() — перезапуск конечного автомата приема данных
Параметры:нет
Возвращаемое значение: нет

Библиотека IRremote для работы с передачей данных по ИК

Структура данных:

```
struct IRData {
    decode_type_t protocol;     //тип протокола:  UNKNOWN, NEC, SONY, RC5, ...
    uint16_t address;           //декодированный адрес
    uint16_t command;           //декодированная команда
    uint8_t numberOfBits;    //количество принятых бит
    uint8_t flags;              //флаги
};
```

IRData decodedIRData - поле с результатами декодирования команды

Прием данных от ИК пульта

```
#include <IRremote.h>
#include <LiquidCrystal.h>

const int PIN_RECEIVER 2   // Вход для ИК приемника
IRrecv receiver(PIN_RECEIVER); // приемник иК
LiquidCrystal lcd(12, 11, 10, 9, 8, 7); // LCD дисплей

void setup()
{
  lcd.begin(16, 2);
  lcd.print("<press a button>");
  receiver.enableIRIn(); // Запуск ИК приемника
}

void loop()
{
  if (receiver.decode()) //проверка новых данных
  {
    translateIR();
    receiver.resume();  //запуск приема новой команды
  }
}
```

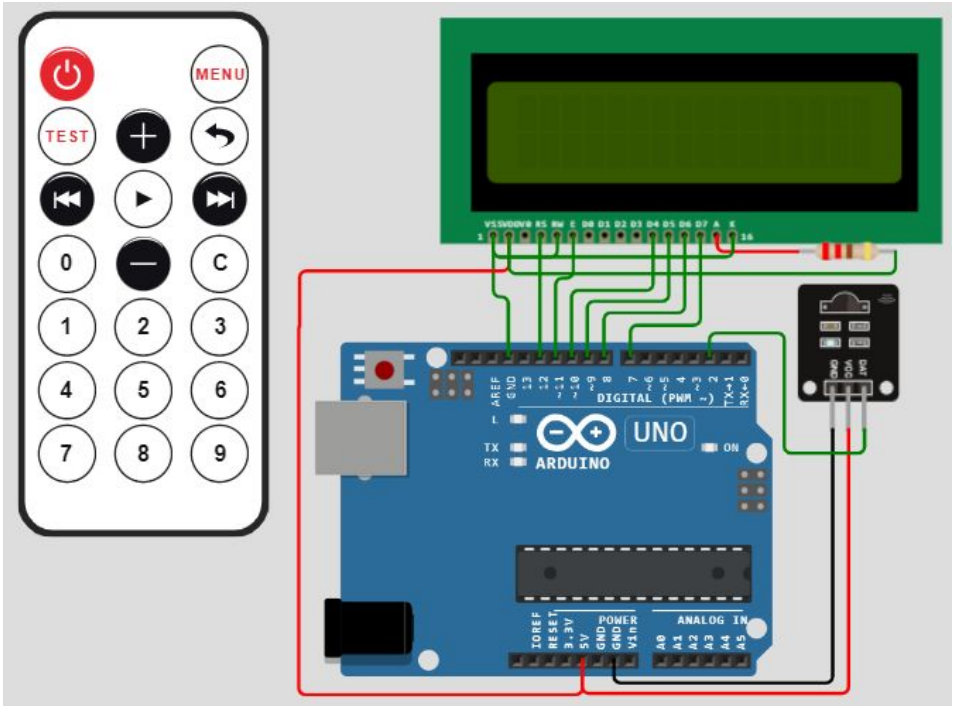
```
//Функция вывода нового значения на индикатор
void lcdPrint(char* text)
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("button pressed:");
  lcd.setCursor(0, 1);
  lcd.print(text);
  lcd.print(" code: ");
  lcd.print(receiver.decodedIRData.command);
}
```

Прием данных от ИК пульта

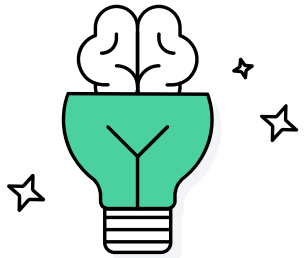
```
//Функция преобразования кода в название
void translateIR()
{
  switch (receiver.decodedIRData.command) {
    case 162:
      lcdPrint("POWER");
      break;
    case 226:
      lcdPrint("MENU");
      break;
    case 34:
      lcdPrint("TEST");
      break;
    case 2:
      lcdPrint("PLUS");
      break;
    case 194:
      lcdPrint("BACK");
      break;
    case 224:
      lcdPrint("PREV.");
      break;
    case 168:
      lcdPrint("PLAY");
      break;
    case 144:
      lcdPrint("NEXT");
      break;
    case 104:
      lcdPrint("num: 0");
      break;
    case 152:
      lcdPrint("MINUS");
      break;
    case 176:
      lcdPrint("key: C");
      break;
    case 48:
      lcdPrint("num: 1");
      break;
    case 24:
      lcdPrint("num: 2");
      break;
    case 122:
      lcdPrint("num: 3");
      break;
    case 16:
      lcdPrint("num: 4");
      break;
      case 56:
      lcdPrint("num: 5");
      break;
    case 90:
      lcdPrint("num: 6");
      break;
    case 66:
      lcdPrint("num: 7");
      break;
    case 74:
      lcdPrint("num: 8");
      break;
    case 82:
      lcdPrint("num: 9");
      break;
    default:
      lcd.clear();

lcd.print(receiver.decodedIRData.command);
      lcd.print(" other button");
  }
}
```

Прием данных от ИК пульта

Используется протокол NEC



Практическое задание №1



Практика: прием данных от ИК пульта

Задание:

1) соберите схему в симуляторе WOKWI, подключив жидкокристаллический индкатор к выводам с 7 по 1, а приемник ИК к выводу 2;
2) создайте скетч с текстом, приведенным выше;
3) проведите моделирование работы

Как выполнять: напишите в чат об удачной работе схемы

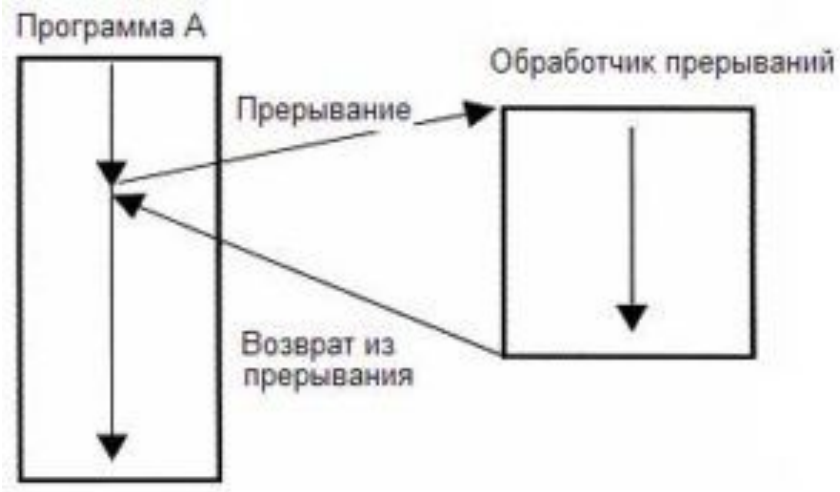
Время выполнения: 5 минут

Как использовать внешние прерывания

2

Что такое прерывания

Прерывание – это событие, которое требует незамедлительной обработки. Микроконтроллер должен отреагировать на это событие, прервав выполнение текущих инструкций и передав управление обработчику прерывания (ISR, Interrupt Service Routine). Обработчик – это обычная функция, которую мы пишем сами и помещаем туда тот код, который должен отреагировать на событие.



Источник

Виды прерываний

- **Аппаратные прерывания**. Прерывание на уровне микропроцессорной архитектуры. Самое событие может произойти в производительный момент либо от внешнего устройства (внешнее прерывание), либо от периферийного модуля микроконтроллера (внутреннее прерывание)
- **Программные прерывания**. Запускаются внутри программы с помощью специальной инструкции. Используются для того, чтобы вызвать обработчик прерываний.

Отличие реализации прерываний в разных платах Arduino

D зависимости от аппаратной реализации конкретной модели микроконтроллера есть несколько прерываний. Например, плата Arduino Uno имеет 2 прерывания на втором и третьем пине

Плата	int.0	int.1	int.2	int.3	int.4	int.5
UNO, Ethernet	2	3				
Mega2560	2	3	21	20	19	18
Leonardo	3	2	0	1	7	

Источник

Функции для работы с внешними прерываниями

void attachInterrupt(uint8_t interruptNum, void (*userFunc)(void), int mode) — разрешает внешнее прерывание и задает функцию для его обработки
Параметры:

- interruptNum: номер прерывания
- userFunc: функция обработки прерывания
- mode: режим обработки прерывания. Допустимо использование следующих констант: LOW - вызывает прерывание, когда на входе LOW; CHANGE - прерывание вызывается при смене значения на входе, с LOW на HIGH или наоборот; RISING - прерывание вызывается только при смене значения на выводе с LOW на HIGH; FALLING - прерывание вызывается только при смене значения на выводе с HIGH на LOW

Возвращаемое значение: нет

void detachInterrupt(uint8_t interruptNum) — запрещает внешнее прерывание
Параметры:

- interruptNum: номер прерывания

Возвращаемое значение: нет

Важные замечания

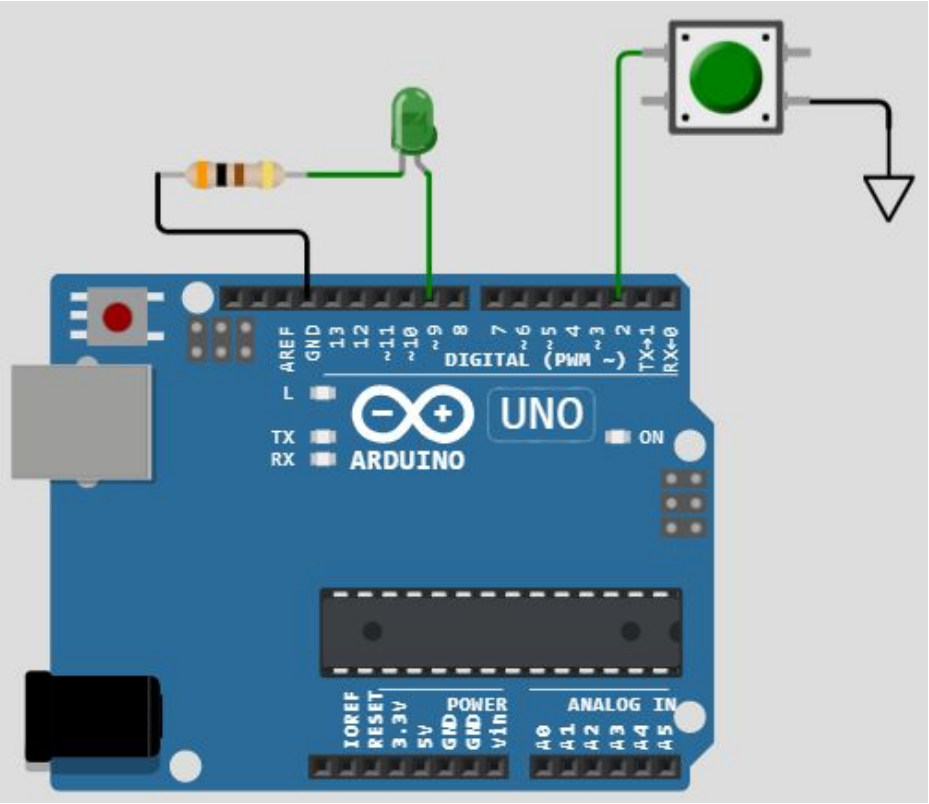
- Функция – обработчик не должна выполняться слишком долго. Пока выполняется ваша функция-обработчик, прерывания с меньшим приоритетом и основной цикл останутся без внимания и вы можете потерять данные.
- При использовании оптимизации для базовых типов переменных, изменяемых в обработчике прерываний, рекомендуется использовать спецификатор volatile.
- В обработчиках категорически нельзя использовать delay(). Механизм определения интервала задержки использует таймеры, а они тоже работают на прерываниях, которые заблокирует ваш обработчик. В итоге программа зависнет.

Обработка нажатия кнопки с помощью внешнего прерывания

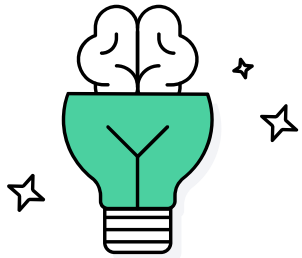
```
const int pinLED =  9;
volatile boolean actionState = LOW;

void setup()
{
  pinMode(pinLED, OUTPUT); // выход на светодиод
  pinMode(2, INPUT_PULLUP); // подтягивающий резистор на входе прерывания
  attachInterrupt(0, myEventListener, FALLING); //разрешение внешнего прерывания (вывод 2 Arduino UNO)
}

void loop()
{
// В функции loop мы ничего не делаем, т.к. весь код обработки событий будет в функции myEventListener
}

//Функция обработки внешнего прерывания
void myEventListener()
{
  actionState = !actionState;
  digitalWrite(pinLED, actionState); //переключение состояния светодиода
}
```

Обработка нажатия кнопки с помощью внешнего прерывания

Настройте модель кнопки, отключив имитацию дребезга контактов с помощью маркера Bounce



Практическое задание №2



Практика: обработка нажатия кнопки с помощью внешнего прерывания

Задание:

1) соберите схему в симуляторе WOKWI, подключив светодиод к выводу 9, а кнопку - 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

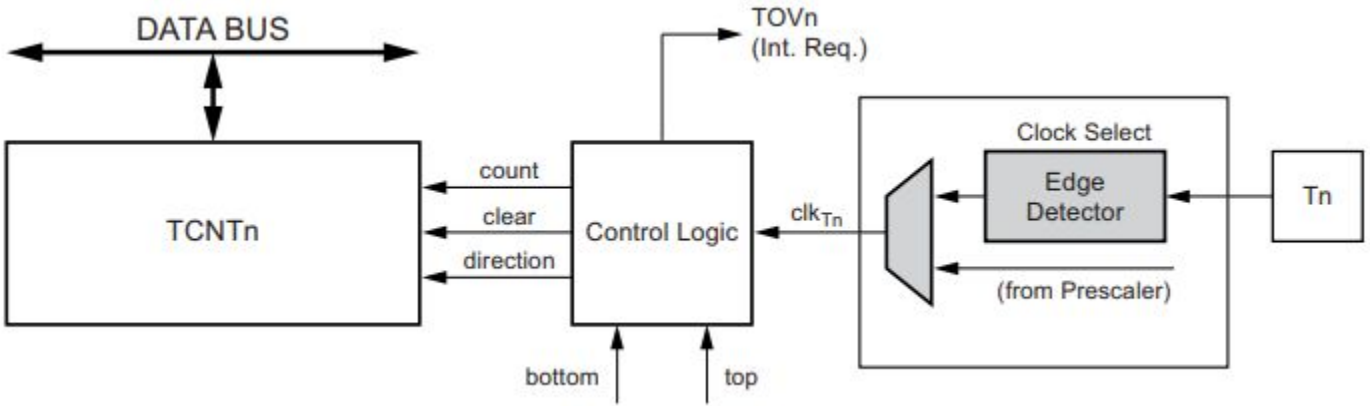
Как использовать прерывания от таймеров

3

Что такое таймер

Таймер - периферийный модуль, в основе которого лежит двоичный счетчик. Счетчик подсчитывает количество импульсов, поступающих на его вход. Импульсы обычно формируются из системного тактового сигнала путем его деления его частоты на целочисленный коэффициент (но может быть и внешнее тактирование).

Таймер формирует запросы на прерывания по особым событиям: переполнение, совпадение с константой и т.д.



Источник

Типы таймеров в Arduino UNO

- **Timer0** является 8 битным таймером/ Timer0 используется стандартными временными функциями Arduino такими как delay() и millis()
- **Timer1** это 16 битный таймер. Этот таймер использует библиотека Arduino Servo.
- **Timer2** — 8 битный и очень похож на Timer0. Он используется в Arduino функции tone() (формирование звукового сигнала)

Подробное описание управления таймерами в документации производителя: ссылка

Таймер T1

Точность таймера зависит от тактовой частоты процессора. Тактовая частота таймера Timer1 определяется установкой предварительного делителя частоты. Этот делитель может быть установлен в значения 1, 8, 64, 256 или 1024.

для тактовой частоты 16 МГц

Делитель	Длительность одного отсчета, мкс	Максимальный период, мс
1	0,0625	8,192
8	0,5	65,536
64	4	524,288
256	16	2097,152
1024	64	8388,608

Источник

Библиотека TimerOne для работы с периферийным модулем таймера T1

При подключении библиотеки объявляется объект Timer1.

void initialize(unsigned long microseconds=1000000) — инициализирует работу таймера с заданным периодом (с округлением до ближайшего допустимого значения)
Параметры:

- microseconds: (необязательный параметр) период работы таймера (по умолчанию 1000000)

Возвращаемое значение: нет

void setPeriod(unsigned long microseconds) — устанавливает период работы таймера (с округлением до ближайшего допустимого значения)
Параметры:

- microseconds: (необязательный параметр) период работы таймера (по умолчанию 1000000)

Возвращаемое значение: нет

Библиотека TimerOne для работы с периферийным модулем таймера T1

void start() — запускает работу таймера
Параметры: нет
Возвращаемое значение: нет

void stop() — останавливает работу таймера
Параметры: нет
Возвращаемое значение: нет

void pwm(char pin, unsigned int duty) — запускает ШИМ сигнал на заданном выводе

Параметры:

- pin: номер выхода (только 9 или 10, остальные значения игнорируются)
- duty: коэффициент заполнения - 10-битное значение в диапазоне от 0 до 1023 (0 соответствует постоянному логическому нулю на выходе, а 1023 – постоянной логической единице)

Возвращаемое значение: нет

Библиотека TimerOne для работы с периферийным модулем таймера T1

void setPwmDuty(char pin, unsigned int duty) — изменяет параметры ШИМ на заданном выводе, когда формирование ШИМ уже запущено
Параметры:

- pin: номер выхода (только 9 или 10, остальные значения игнорируются)
- duty: коэффициент заполнения - 10-битное значение в диапазоне от 0 до 1023 (0 соответствует постоянному логическому нулю на выходе, а 1023 – постоянной логической единице)

Возвращаемое значение: нет

void disablePwm(char pin) — выключает ШИМ на заданном выводе
Параметры:

- pin: номер выхода (только 9 или 10, остальные значения игнорируются)

Возвращаемое значение: нет

Библиотека TimerOne для работы с периферийным модулем таймера T1

void attachInterrupt(void (*isr)()) — прикрепляет функцию обработки прерывания при переполнении таймера
Параметры:

- isr: функция обработки прерывания

Возвращаемое значение: нет

void detachInterrupt() — открепляет функцию обработки прерывания при переполнении таймера
Параметры: нет
Возвращаемое значение: нет

Использование прерывания от таймера

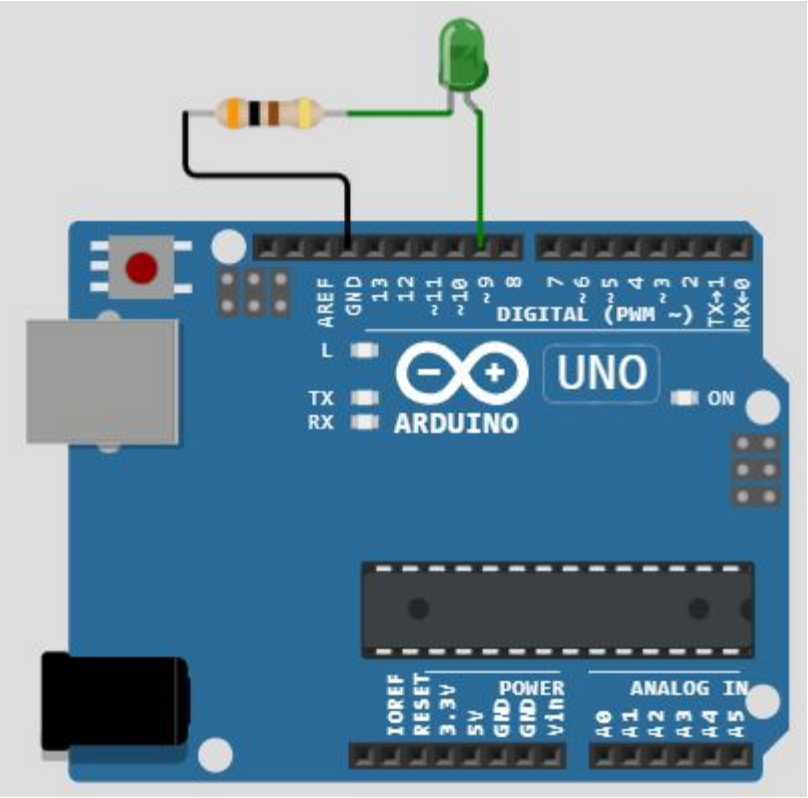
```
#include <TimerOne.h>

const int led = 9;  // вывод со светодиодом
int ledState = LOW; //состояние светодиода
volatile unsigned long blinkCount = 0; // используйте volatile для общих переменных

void setup(void)
{
  pinMode(led, OUTPUT);
  Timer1.initialize(150000);
  Timer1.attachInterrupt(blinkLED); // вызывать blinkLED каждые 0.15 сек.
  Serial.begin(9600);
}
```

Использование прерывания от таймера

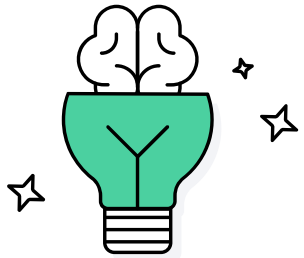
```
void blinkLED(void)
{
  if (ledState == LOW)
  {
    ledState = HIGH;
    blinkCount = blinkCount + 1;  //количество миганий
  } else
  {
    ledState = LOW;
  }
  digitalWrite(led, ledState);
}

void loop(void)
{
  unsigned long blinkCopy;  // хранит копию blinkCount

  noInterrupts(); //запрещение прерывания при доступе к общей переменной
  blinkCopy = blinkCount;
  interrupts();

  Serial.print("blinkCount = ");
  Serial.println(blinkCopy);
  delay(100);
}
```

Использование прерывания от таймера

Управление светодиодом происходит в прерывании от таймера



Практическое задание №3



Практика: использование прерывания от таймера

Задание:

1) соберите схему в симуляторе WOKWI, подключив светодиод к выводу 9;
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

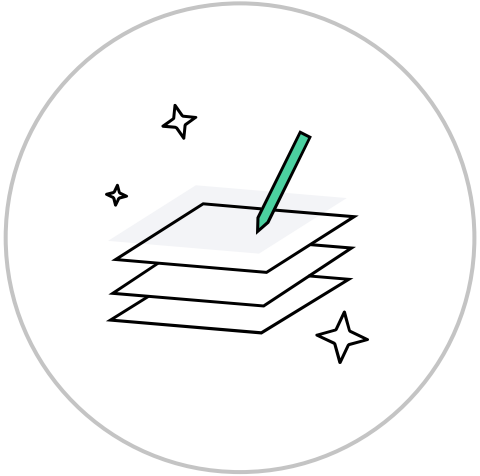
Сегодня мы

1. Узнали принцип построение и особенности протокол для дистанционного управления по инфракрасному каналу
2. Научились принимать данные по инфракрасному каналу с помощью специальной библиотеки
3. Научились использовать внешнее прерывание для обработки сигнала от кнопки
4. Научились использовать прерывание от таймера для периодического выполнения какой-либо задачи

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