МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Тульский государственный университет»

Институт высокоточных систем им В.П. Грязева
Кафедра «Приборы управления»

Утверждено на заседании кафедры
«Приборы управления»
«22» января 2024 г., протокол №1

Заведующий кафедрой

В.В. Матвеев

МЕТОДИЧЕСКИЕ УКАЗАНИЯ
по выполнению лабораторных работ
по дисциплине (модулю)
«Интеллектуальные фотонные системы»

основной профессиональной образовательной программы
высшего образования – программы бакалавриата

по направлению подготовки
12.03.03 Оптотехника

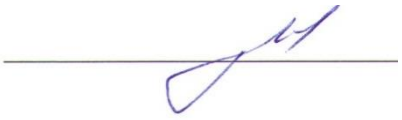
с направленностью (профилем)
Интеллектуальные фотонные системы

Форма обучения: очная

Идентификационный номер образовательной программы: 120303-01-24

Тула 2024 год

Разработчик методических указаний

Матвеев В.В., зав. каф., д.т.н., доц_____ 

СОДЕРЖАНИЕ

Введение

Современные космические технологии и приборостроение неразрывно связаны с использованием оптико-электронных систем. Данные системы находят широкое применение в различных областях, таких как передача информации, навигация, управление космическими аппаратами и многие другие.

В рамках данной лабораторного практикума студенты знакомятся с основами проектирования и исследования оптико-электронных систем, которые лежат в основе многих космических технологий. Представленные лабораторные работы охватывают широкий спектр тем, связанных с передачей данных по лазерным каналам связи, разработкой систем управления ориентацией солнечных батарей космических аппаратов, а также исследованием динамики их стабилизации.

Последовательность лабораторных работ позволяет последовательно сформировать у студентов целостное представление о принципах построения и функционирования оптико-электронных систем космического назначения. Начиная с простейших систем передачи данных, студенты постепенно переходят к более сложным задачам, связанным с разработкой алгоритмов управления ориентацией и стабилизацией малых космических аппаратов.

Выполнение данных лабораторных работ не только способствует закреплению теоретических знаний, но и развивает у студентов практические навыки проектирования, сборки и исследования оптико-электронных систем. Полученный опыт может быть впоследствии использован при решении инженерных задач в области космической техники и приборостроения.

Лабораторная работа №1. «Исследование простейшей системы передачи данных по лазерному каналу связи»

1. Цель и задачи работы

Изучение принципов передачи информации по лазерному излучению. Получение практических навыков в разработке и реализации систем передачи данных между микроконтроллерными устройствами на базе *Arduino*.

2. Основные теоретические положения

2.1 Основы передачи данных

В настоящее время передача данных может происходить при помощи различных технических средств связи. Так, основными устройствами, для быстрой передачи информации являются телеграф, радио, телевизионный передатчик и т.д. В процессе передачи данных, они могут искажаться или теряться. Так, например, реализация передачи данных по радиоканалам существенно подвержена возможности нарушения ее работы различными искусственными помехами: активными – специальными станциями помех, и пассивными – сбрасываемыми металлизированными лентами.

Принципиально новой формой доставки информации является передача информации по лазерному излучению. Данная технология передачи данных по лазеру обеспечивает высокий уровень защиты информации и отличается низкими задержками.

Схема простейшей системы передачи и приема информации по лазерному излучению представлена на рисунке 1. Такая система состоит из персональных компьютеров для формирования и приема информации, контроллеров передающей и приемной аппаратуры, модуля лазера, а также фотоприемного устройства (ФПУ).

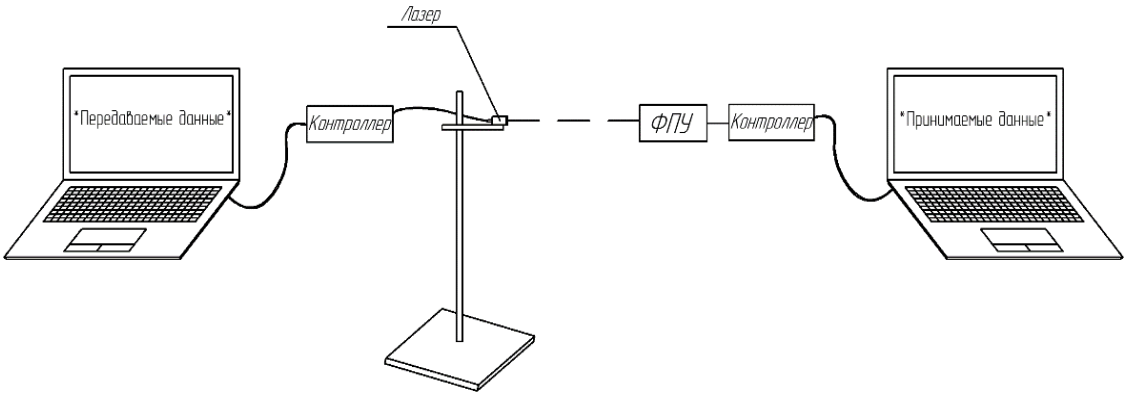


Рисунок 1 – Простейшая система передачи данных по лазерному излучению

2.2 Аппаратное и программное обеспечение для передачи данных

В качестве управляющего устройства в реализации макетного образца был выбран контроллер *Arduino UNO R3* (рис. 2)

Рисунок 2– Управляющий контроллер

Arduino UNO R3 – устройство на основе микроконтроллера *ATmega328*, в состав которого входят 14 цифровых входов/выходов, 6 аналоговых входов, разъемы *USB* и питания, разъем для внутрисхемного программирования (*ICSP*) и кнопка сброса.

Arduino UNO R3 предоставляет ряд возможностей для связи с компьютером или другими микроконтроллерами. Одним из способов передачи данных программно с помощью микроконтроллера *Arduino* является использование библиотеки *SoftwareSerial*, позволяющую реализовать последовательный интерфейс на любых других цифровых

выводах *Arduino* с помощью программных средств, дублирующих функциональность *UART*.

UART, в переводе, универсальный асинхронный приемопередатчик, который служит для преобразования передаваемых данных в последовательный вид так, чтобы была возможность передать их по цифровой линии другому аналогичному устройству.

Данные *UART* передаются последовательным кодом в следующем формате:

– в неактивном режиме выход *UART* находится в высоком состоянии;

– передача байта начинается со стартового бита, соответствующего низкому уровню;

– передача байта заканчивается стоповым битом, соответствующего высокому уровню;

– данные передаются младшим битом вперед;

– для передачи байта требуется 10 битов.

В качестве лазера целесообразно использовать лазерный модуль *KY*-008 (рис 3).

Рисунок 3 – Лазерный модуль *KY*-008

Модуль представляет собой лазерный излучатель, расположенный на плате адаптированной к совместной работе с блоками *Arduino*. Состав

модуля – это лазерный диод, запрессованный в цилиндрическом радиаторе охлаждения соединенный с токоограничивающим резистором, расположенным на плате.

Для получения сигнала с передающей аппаратуры можно выбрать модуль датчика лазерного детектора *GY*-530 (рис. 4).

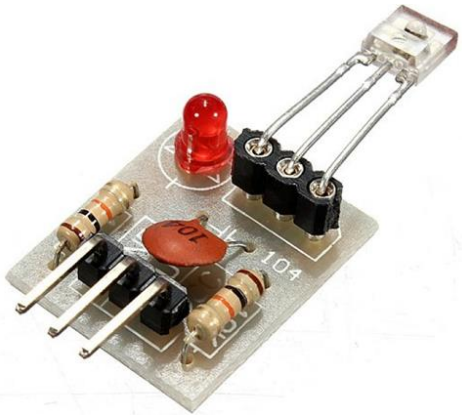


Рисунок 4 – Модуль датчика лазерного детектора *GY*-530

Этот модуль приема лазерного датчика представляет собой датчик для *Arduino* для приема лазерного сигнала и декодирования его цифровых данных. На входе модуль принимает лазерный сигнал при высоком уровне выходного сигнала.

3. Задание на работу

Ознакомиться с основными принципами и компонентами оптико-электронной системы передачи данных и на их основе разработать и собрать простейшую систему передачи данных.

4. Порядок выполнения работы

1. Ознакомление с техникой безопасности и основным теоретическим материалом.
2. Подготовка оборудования (подготовка двух *Arduino*-плат, соединительных проводов, необходимой элементной базы)
3. Реализация передатчика и приемника, включающих:

– Подключение библиотеки *SoftwareSerial*.

– Передачу данных через дополнительный последовательный порт.

– Прием данных через дополнительный последовательный порт.

– Вывод принятых данных в основной последовательный порт.

4. Проверка работы системы, тестирование и отладка.

5. Документирование и оформление отчета (описание проделанной работы, объяснение реализованных алгоритмов и принципов работы, анализ полученных результатов и выводы, оформление отчета в соответствии с требованиями).

5. Контрольные вопросы

1. Объясните принцип построения систем передачи данных по лазерному излучению.

2. Для чего используется библиотека *SoftwareSerial* в *Arduino*?

3. Объясните, как создается объект *SoftwareSerial* и какие параметры передаются при его инициализации.

4. Опишите, что происходит в секциях *setup()* и *loop()* кода передатчика и приемника.

5. Объясните, как происходит обмен данными между передатчиком и приемником в данной системе.

ПРИЛОЖЕНИЕ 1. ЛИСТИНГ ПРОГРАММ К ЛАБОРАТОРНОЙ РАБОТЕ №1

Код программы для передатчика:

```
#include <SoftwareSerial.h>
SoftwareSerial s1(9, 10, true);
void setup() {
  pinMode(10, OUTPUT);
  Serial.begin(9600);
```

```
  s1.begin(9600);
}
void loop() {
  if (Serial.available()) {
    char data = Serial.read();
    s1.write(data);
    Serial.print("Sent: ");
    Serial.println(data);
  }
}
```

Код программы для приемника:

```
#include <SoftwareSerial.h>
SoftwareSerial s1(10,11,true);
//SoftwareSerial s1(10,11);
void setup() {
pinMode(10,INPUT);
Serial.begin(9600);
s1.begin(9600);
}
void loop() {
  if(s1.available()){
    Serial.write
    (s1.read());}
   }
```

Лабораторная работа №2. «Разработка системы передачи команд управления по лазерному каналу связи»

1. Цель и задачи работы

Закрепление изученных принципов передачи информации по лазерному излучению. Получение практических навыков в разработке и реализации систем передачи данных между микроконтроллерными устройствами на базе *Arduino*.

2. Основные теоретические положения

2.1. Передача данных в системах наведения

Использование лазерной передачи данных в системах наведения имеет ряд преимуществ и особенностей:

1. Высокая скорость передачи данных;
2. Направленность передачи;
3. Низкое энергопотребление;
4. Отсутствие электромагнитных помех;
5. Безопасность передачи;
6. Небольшие размеры и вес.

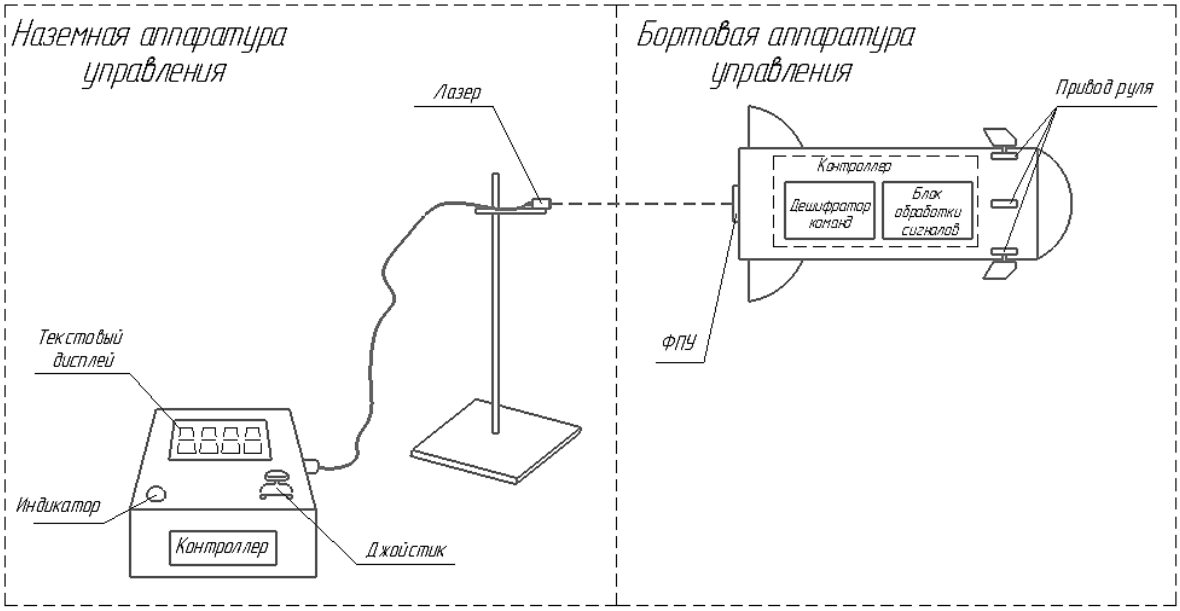


Рисунок 1 – Функциональная схема лазерно-лучевой системы наведения

Согласно приведенной схеме, конструкция разрабатываемого устройства включает в себя две основные части: наземная аппаратура управления и бортовая аппаратура управления.

В блоке наземной аппаратуры управления осуществляется связь лазерно-лучевой системы наведения с бортовой цифровой вычислительной машиной. Блок формирует информационное поле управления ракетой с помощью джойстика. Сигналы, отправляемые на бортовую аппаратуру, отражаются через индикаторы.

Определение переданного направления движения на ракете происходит путём анализа принимаемого лазерного излучения за счёт установки в хвостовой части ракеты фотоприёмного устройства. Бортовая аппаратура управления принимает, дешифрует и обрабатывает принятые данные, а также и вырабатывает сигналы формирования команд управления рулями.

2.2. Аппаратное и программное обеспечение

Для вывода передаваемых данных, а также заданного направления движения летательного аппарата, можно использовать жидкокристаллический дисплей *LCD* 1602 (рис. 2).

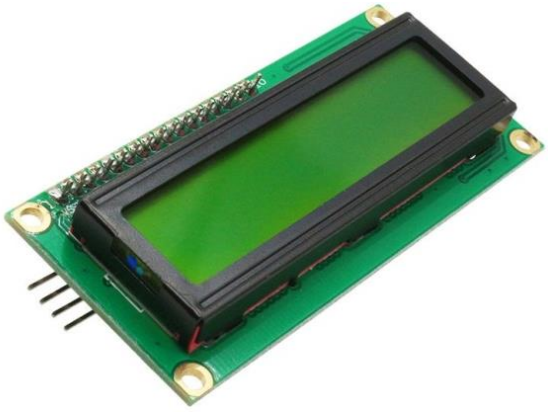


Рисунок 2 – Жидкокристаллический дисплей *LCD* 1602

Жидкокристаллический дисплей *LCD* 1602 – устройство, использующееся для вывода информации с датчиков, отображения меню или

подсказок. Представляет собой экран, выполненный на жидкокристаллической матрице, отображающей 2 строки по 16 символов, каждый из которых состоит из отдельного знакоместа 5×8 пикселей.

Реализация задания символов, указывающих на изменение положения приводов рулей, можно проводить с помощью двухосевого джойстика *KY*-023 (рис 3).

Рисунок 3 – Двухосевой джойстик *KY*-023

Двухосевой джойстик *KY*-023 – устройство, состоящее из одной тактовой кнопки и двух потенциометров на 10 кОм, определяющие положение осей *ox* и *oy* с помощью изменения сопротивления рычагом.

Для реализации приведения в движение рулей на бортовой аппаратуре управление можно установить четыре сервопривода *SG*90 (рис. 4).

Рисунок 4 – Сервопривод *SG*90

Сервопривод *SG*90 – устройство, используется в управлении небольшими легкими механизмами с углом поворота от 0° до 180°, что полностью удовлетворяет требованиям создания макетного образца.

3. Задание на работу

Ознакомиться с основными принципами и компонентами оптико-электронной системы передачи данных и разработать и собрать систему на основе рисунка 1 и лабораторной работы №1.

4. Порядок выполнения работы

1. Ознакомление с техникой безопасности и основным теоретическим материалом.
2. Подготовка оборудования (подготовка двух *Arduino*-плат, соединительных проводов, необходимой элементной базы)
3. Реализация передатчика и приемника со следующими замечаниями:

– На дисплее должны быть выведены направления движения ЛА.

– Угол отклонения сервоприводов – 50°.

– В реализованной программе должен передаваться пакет *ASCII*-символов от «1» до «9», обозначающих положение сервоприводов. Для проверки соответствия в таблице 1 приведены передаваемые *ASCII*-символы и их коды в десятичной и двоичной системах счисления.

Таблица 1 – Сопоставление кодов

ASCII-символ	Десятичный код	Двоичный код
1	49	00110001
2	50	00110010
3	51	00110011
4	52	00110100
5	53	00110101
6	54	00110110

7	55	00110111
8	56	00111000
9	57	00111001

4. Проверка работы системы, тестирование и отладка.

5. Документирование и оформление отчета (описание проделанной работы, объяснение реализованных алгоритмов и принципов работы, анализ полученных результатов и выводы, оформление отчета в соответствии с требованиями).

4. Контрольные вопросы

1. Объясните принцип действия системы, представленной на рисунке 1.

2. Что такое дополнительные последовательные порты на платах *Arduino* и для чего они используются?

3. Расскажите о библиотеке *SoftwareSerial*. Что она позволяет делать и какие особенности ее использования?

4. Объясните принцип работы передатчика в представленном коде. Как он определяет положение джойстика и кодирует его?

5. Как реализована передача данных от передатчика к приемнику? Какие последовательные порты задействованы?

6. Опишите, как работает приемник в представленном коде. Как он интерпретирует полученные данные и управляет сервоприводами?

ПРИЛОЖЕНИЕ 2. ЛИСТИНГ ПРОГРАММ К ЛАБОРАТОРНОЙ РАБОТЕ №2

Код программы для передатчика:

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

```
SoftwareSerial t1(8,9);
LiquidCrystal lcd(13,12,11,10,5,4);
int a;
int b;
int min1=0; //пороги
int max1=694;
boolean pusk = false;
void setup() {
  pinMode(6,HIGH);
  digitalWrite(6,HIGH);
  lcd.begin(16,2);
  pinMode(A1,INPUT);
    pinMode(A2,INPUT);
    pinMode(7, OUTPUT);
  Serial.begin(9600);
  t1.begin(9600);
}
int joystick(){
a=analogRead(A1);
b=analogRead(A2);
if (digitalRead(6)==0){
pusk = true;
}
  if(pusk == true){
digitalWrite(7,HIGH);
 lcd.clear();
if(b>=max1&&a>=max1) {
  lcd.setCursor(0,0);
  lcd.print("CODE=1");
  lcd.setCursor(0,1);
```

```
  lcd.print("DOWN_RIGHT");
  return 1;};
if(b<max1&&b>min1&&a>=max1) {
  lcd.setCursor(0,0);
  lcd.print("CODE=2");
    lcd.setCursor(0,1);
  lcd.print("DOWN");
  return 2;};
if(b==min1&&a>=max1) {
  lcd.setCursor(0,0);
  lcd.print("CODE=3");
    lcd.setCursor(0,1);
  lcd.print("DOWN_LEFT");
  return 3;};
if(a<max1&&a>min1&&b>=max1) {
  lcd.setCursor(0,0);
  lcd.print("CODE=4");
    lcd.setCursor(0,1);
  lcd.print("RIGHT");
  return 4;};
if(a<max1&&a>min1&&b<max1&&b>min1) {
  lcd.setCursor(0,0);
  lcd.print("CODE=5");
    lcd.setCursor(0,1);
  lcd.print("STRAIGHT");
  return 5;};
if(a<max1&&a>min1&&b==min1){
  lcd.setCursor(0,0);
  lcd.print("CODE=6");
    lcd.setCursor(0,1);
```

```
  lcd.print("LEFT");
  return 6;};
if(a==min1&&b>=max1){
  lcd.setCursor(0,0);
  lcd.print("CODE=7");
    lcd.setCursor(0,1);
  lcd.print("UP_RIGHT");
  return 7;};
if(a==min1&&b>min1&&b<max1) {
  lcd.setCursor(0,0);
  lcd.print("CODE=8");
    lcd.setCursor(0,1);
  lcd.print("UP");
  return 8;};
if(a==min1&&b==min1) {
  lcd.setCursor(0,0);
  lcd.print("CODE=9");
    lcd.setCursor(0,1);
  lcd.print("UP_LEFT");
  return 9;};
} else {return 0;}}
void loop() {
 delay(50);
 t1.print(joystick());
}
```

Код программы для приемника:

```
#include <SoftwareSerial.h>
#include <Servo.h>
SoftwareSerial s1(10,11,true);
Servo servo1;
```

```
Servo servo2;
Servo servo3;
Servo servo4;
char control;
int digit;
void setup() {
Serial.begin(115200);
s1.begin(9600);
pinMode(9, OUTPUT);
pinMode(5, OUTPUT);
pinMode(3, OUTPUT);
pinMode(6, OUTPUT);
servo1.attach(9); //верхний
servo2.attach(5); //нижний
servo3.attach(3); //слева
servo4.attach(6); //справа
}
void loop() {
if(s1.available() > 0){
    control=(s1.read());
    digit=int(control);
if(digit==49){
servo1.write(40);
servo2.write(140);
servo3.write(140);
servo4.write(40);
Serial.println(49);
}
else if(digit==50){
servo1.write(90);
```

```
servo2.write(90);
servo3.write(140);
servo4.write(40);
Serial.println(50);
}
else if(digit==51){
servo1.write(140);
servo2.write(40);
servo3.write(140);
servo4.write(40);
Serial.println(51);
}
else if(digit==52){
servo1.write(40);
servo2.write(140);
servo3.write(90);
servo4.write(90);
Serial.println(52);
}
else if(digit==53){
servo1.write(90);
servo2.write(90);
servo3.write(90);
servo4.write(90);
Serial.println(53);
}
else if(digit==54){
servo1.write(140);
servo2.write(40);
servo3.write(90);
```

```
servo4.write(90);
Serial.println(54);
}
else if(digit==55){
servo1.write(40);
servo2.write(140);
servo3.write(40);
servo4.write(140);
Serial.println(55);
}
else if(digit==56){
servo1.write(90);
servo2.write(90);
servo3.write(40);
servo4.write(140);
Serial.println(56);
}
else if(digit==57){
servo1.write(140);
servo2.write(40);
servo3.write(40);
servo4.write(140);
Serial.println(57);
}
}
}
```

Лабораторная работа №3. Исследование амплитуды первой гармоники потока излучения на выходе модулятора на базе секторного растра

1.Цели и задачи работы

Ознакомление с принципом работы оптико-механического модулятора на базе секторного растра.

2.Основные теоретические положения

2.1. Модуляция оптических сигналов с помощью растров

Механическая модуляция оптических сигналов с помощью растров широко применяется в ОЭП. При этом сравнительно просто осуществляется амплитудная, частотная и фазоимпульсная модуляция. Схема такой модуляции можно разделить на две группы:

- с перемещением растра относительно неподвижного изображения;
- с неподвижным растром-модулятором, относительно которого перемещается изображение или световой пучок.

Если пропускание растра-модулятора описывается действительной функцией $a\left(\bar{\rho},t\right)$, где $\bar{\rho}$ - радиус-вектор произвольной точки в плоскости растра; t – время; распределение освещенности в этой плоскости описывается функцией $E\left(\bar{\rho}\right)$, то сигнал (поток) на выходе растра в момент t равен

$$\Phi\left(t\right)=\int\limits_{\theta_{\bar{\rho}}} E\left(\bar{\rho},t\right)a\left(\bar{\rho},t\right)d\theta_{\bar{\rho}}, \qquad (1)$$

где $\theta_{\bar{\rho}}$ - область значений вектора $\bar{\rho}$. Таким образом, при взаимном перемещении растра и изображения происходит превращение «*пространственного*» оптического сигнала во «*временной*».

Если рисунок растра периодичен, т.е. рисунок повторяется с частотой ω_1, то

$$a\left(\bar{\rho},t\right)=a\left(\bar{\rho},t+2\pi/\omega_1\right).$$

Представляя эту функцию рядом Фурье

$$a\left(\bar{\rho},t\right)=\sum_{n=-\Gamma}^{\Gamma}a_n\left(\bar{\rho}\right)\exp\left(jn\omega_1 t\right), \qquad (2)$$

$$a_n\left(\bar{\rho}\right)=\frac{\omega_1}{2\pi}\int_{-\pi/\omega_1}^{\pi/\omega_1}a\left(\bar{\rho},t\right)\exp\left(-jn\omega_1 t\right)dt$$

и подставляя (2) в (1), получим

$$\Phi\left(t\right)=\sum_{n=-\Gamma}^{\Gamma}\Phi_n\left(t\right)\exp\left(jn\omega_1 t\right),$$

где

$$\Phi_n\left(t\right)=\int_{\theta_{\bar{\rho}}}a_n\left(\bar{\rho}\right)E\left(\bar{\rho},t\right)d\theta_{\bar{\rho}}. \qquad \textbf{(3)}$$

Каждая гармоника вида $\exp\left(jn\omega_1 t\right)$в формуле (**3**) является несущей частотой, которая модулируется функцией $\Phi_n\left(t\right)$, содержащей информацию о распределении освещенности в плоскости растра, например, информацию о координатах изображения, описываемого функцией $E\left(\bar{\rho},t\right)$.

2.2. Модуляция секторным растром излучения, равномерно распределённого в пределах круговой диафрагмы поля

Отверстие диафрагмы представляет собой круг радиусом r, имеющим на расстоянии ρ_0 угловой размер $2\alpha_0$ (рис. 1). Отверстие не закрыто фильтром, т.е. $\tau_\Phi(\varphi)=1$, и, следовательно, падающий поток излучения распределен равномерно по отверстию диафрагмы. Растр состоит из N непрозрачных секторов, пропускание которых равно нулю, и N прозрачных секторов, пропускание которых равно $\tau_р$.

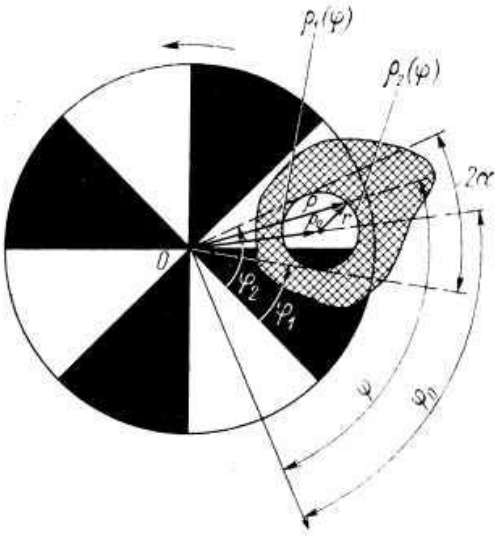


Рис. 1. Секторный растр с круглой диафрагмой поля

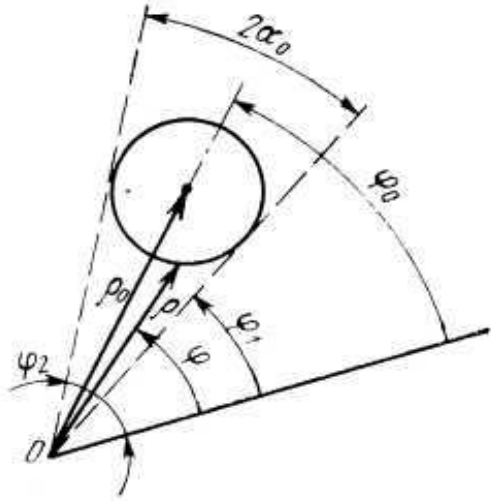


Рис. 2. Окружность в полярных координатах

Расчёт может производиться двумя путями.

Первый ***путь используется исключительно в случаях, когда*** $x_0 = \rho_0 / r > 1.$

Амплитуда первой гармоники спектра модулированного потока излучения

$$A_1 = \tau_p(\pi/2)[2J_1(y_0)/y_0] \qquad (4)$$

На рис. 3 приведена зависимость $|2J_1(y_0)/y_0|$ от $1/y_0$ (где $y_0 = Nk(r/\rho_0)$).

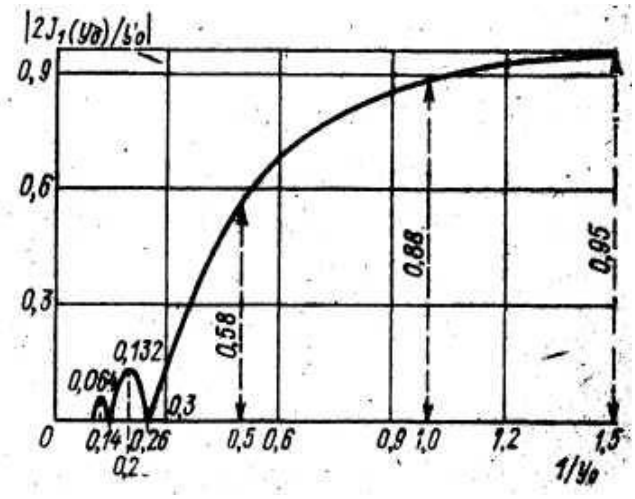


Рис.3 Зависимость $|2J_1(y_0)/y_0|$ от $1/y_0$

Второй путь расчёта используется, когда ρ_0 / r принимает значения как больше, так и меньше единицы.

Амплитуда первой гармоники модулированного сигнала равна

$$A_1 = \tau_p (2/\pi) a_{01}. \qquad (5)$$

В табл. 1 приведены значения коэффициента $a'_{01} = (\pi/2) a_{01}$ для различных значений N и x_0. Зная коэффициент, легко найти амплитуду первой гармоники, которая при $\tau_p = 1$ равна $A_1 = (4/\pi^2) a'_{01} \approx 0.4 a'_{01}$.

Сопоставление значений A_1, вычисленных из соотношения $A_1 = A'_1 = 0.4 a'_{01}$, со значениями A_1, найденными из формулы $A_1 = A''_1 = 0{,}64[2J_1(y_0)/y_0]$, даёт хорошее совпадение результатов. Действительно, для x_0, равного 1; 2; 5; 10; и N, равного 1; 2; 3; 10, имеем значения, приведённые в табл. 2.

Расхождения в значениях амплитуды первой гармоники, полученные при различных способах вычисления, объясняются тем, что случай, когда x0=1,является граничным, и ему свойственны наибольшие ошибки, связанные с принятыми в расчётах приближениями.

Таблица 1

Значение коэффициента a'_{01}

x_0	N													
	1	2	3	4	5	7	8	10	11	12	15			
1,0	1,34223	0,76468	0,24697	0,00631	—0,01994	0,00154	—0,00541	0,00494	—0,00000	—0,00375	0,00000			
1,5	1,49975	1,22809	0,85199	0,46041	0,13559	—0,14700	—0,12193	0,03108	0,07194	0,06713	—0,04561			
2,0	1,54482	1,39028	1,15623	0,87368	0,57838	0,08130	—0,07485	—0,19570	—0,14245	—0,08093	0,08072			
2,5	1,56471	1,46535	1,30926	1,10962	0,88284	0,41940	0,21591	—0,07530	—0,15240	—0,18612	—0,09715			
3,0	1,57529	1,50613	1,39541	1,24962	1,07713	0,69139	0,49860	0,18932	0,02678	—0,07520	—0,19424			
3,5	1,58160	1,53072	1,44836	1,33806	1,20450	0,89044	0,72251	0,39670	0,25005	0,12054	—0,13806			
4,0	1,58567	1,54668	1,48312	1,39709	1,29138	1,03495	0,89227	0,50916	0,45720	0,32336	0,00008			
4,5	1,58844	1,55762	1,50713	1,43830	1,35288	1,14134	1,02049	0,76329	0,63249	0,50515	0,16405			
5,0	1,59042	1,56544	1,52440	1,46814	1,39785	1,22122	1,11848	0,89434	0,77741	0,65985	0,32584			
5,5	1,59189	1,57124	1,53722	1,49042	1,43165	1,28239	1,19444	0,99914	0,80496	0,78866	0,47267			
6,0	1,59300	1,57564	1,54699	1,50747	1,45766	1,33012	1,25423	1,04350	0,99103	0,89538	0,40305			
6,5	1,59386	1,57907	1,55462	1,52082	1,47808	1,36799	1,30199	1,15202	1,06984	0,98412	0,71558			
7,0	1,59454	1,58179	1,56068	1,53144	1,49439	1,39850	1,34066	1,20822	1,13498	1,05804	0,81252			
7,5	1,59509	1,58398	1,56557	1,54005	1,50763	1,42340	1,37237	1,25476	1,18926	1,12005	0,89639			
8,0	1,59554	1,58578	1,56958	1,54710	1,51851	1,44399	1,39866	1,29367	1,23485	1,17242	0,96842			
8,5	1,59592	1,58726	1,57291	1,55796	1,52756	1,46118	1,42068	1,32647	1,27344	1,21694	1,03067			
9,0	1,59623	1,58851	1,57570	1,55788	1,53517	1,47569	1,43929	1,35436	1,30635	1,25505	1,08465			
9,5	1,59650	1,58957	1,57806	1,56205	1,54163	1,48803	1,45516	1,37424	1,33462	1,28788	1,13166			
10,0	1,59672	1,59047	1,58008	1,56562	1,54715	1,49862	1,46830	1,39884	1,35906	1,31633	1,17278			

Таблица 2

Значения амплитуд первой гармоники,

полученных различными способами

x_0	N=1		N=2		N=3		N=10	
	A'_1	A''_1	A'_1	A''_1	A'_1	A''_1	A'_1	A''_1
1.0	0.54	0.56	0.30	0.37	0.10	0.15	0.003	0.005
2.0	0.62	0.62	0.55	0.56	0.46	0.47	-0.08	-0.08
5.0	0.64	0.64	0.62	0.63	0.61	0.61	0.36	0.37
10.0	0.64	0.64	0.64	0.64	0.63	0.63	0.56	0.56

Описание лабораторного стенда

Общий вид лабораторного стенда приведен на рис. 4. Источник излучения 1 формирует параллельный световой поток. Далее (при выключенном тумблере 5) поток проходит через прозрачный сектор растра 4 и попадает на приёмник излучения 2, где происходит дальнейшая обработка сигнала. После включения двигателя 3 тумблером 5, приводится в движение растр 4, посредством которого происходит модуляция светового потока. В приёмнике излучения 2 происходит обработка уже модулированного потока.

Описание электронного тракта лабораторной установки

Схема электронного тракта лабораторной установки показана на рис. 5. Световой поток попадает на ФПУ, где происходит предварительное усиление сигнала по току (на осциллографе, подключённом к выходу ФПУ, наблюдаем импульсы прямоугольной формы). Далее обработка сигнала происходит в резонансном усилителе, где выделяется первая гармоника (на осциллографе, подключённом к выходу резонансного усилителя, наблюдаем сигнал синусоидальной формы). Обработанный резонансным усилителем сигнал поступает на вход выпрямителя, предназначенного для

детектирования сигнала (на осциллографе, подключённом к выходу выпрямителя, наблюдаем выпрямленный сигнал).

3. Порядок выполнения работы

1. Рассчитать амплитуду первой гармоники A'_1 спектра модулированного потока излучения, в случае, когда $x_0 = \rho_0 / r > 1$ по формуле 4.

2. Рассчитать амплитуду первой гармоники A''_1 спектра модулированного потока излучения, в случае, когда ρ_0 / r принимает значения как больше, так и меньше единицы по формуле 5.

3. Подключить лабораторную установку к источнику питания, предварительно отключив питание от двигателя (тумблер SA1).

4. Установить прозрачный сектор диска напротив луча.

5. Измерить максимальный поток, падающий на фотоприёмник (клеммы 0и1) осциллографом.

6. Включить тумблер SA1 (поток модулируется вращающимся растром).

7. Снять сигнал с клемм 0(земля) и 1(после усилителя). Зарисовать полученный график модулированного потока.

8. Снять сигнал с клемм 0(земля) и 2(после резонансного усилителя). Зарисовать полученный график амплитуды первой гармоники.

9. Снять сигнал с клемм 0(земля) и 3(после выпрямителя). Зарисовать полученный график выпрямленного напряжения.

10. Измерить амплитуду 1-ой гармоники по полученному графику (п. 8).

11. Найти отношение амплитуды 1- ой гармоники к максимальному потоку A'''_1.

12. Сравнить измеренное значение с рассчитанными и сделать вывод о соответствии результатов измерения и результатов расчета.

13. Повторить п.**1-12** при отклонении источника излучения на ±5°.

14. Результаты измерений и расчётов записать в таблицу

Угол отклонения, ^{0}	A'_1	A''_1	A'''_1
0			
+5			
-5			

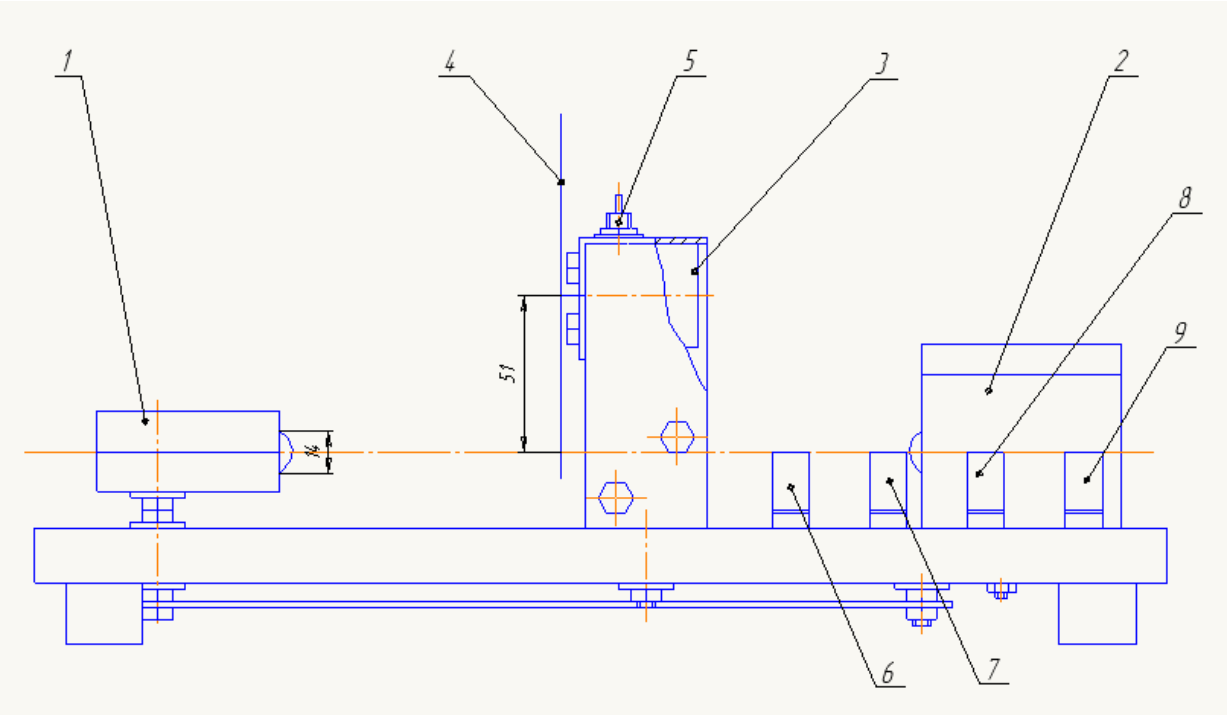


Рис. 4. Лабораторный стенд: 1-источник излучения; 2-приёмник излучения; 3-двигатель; 4-растр; 5-тумблер SA1; 6-клемма-0; 7-клемма-1; 8-клемма-2; 9-клемма-3.

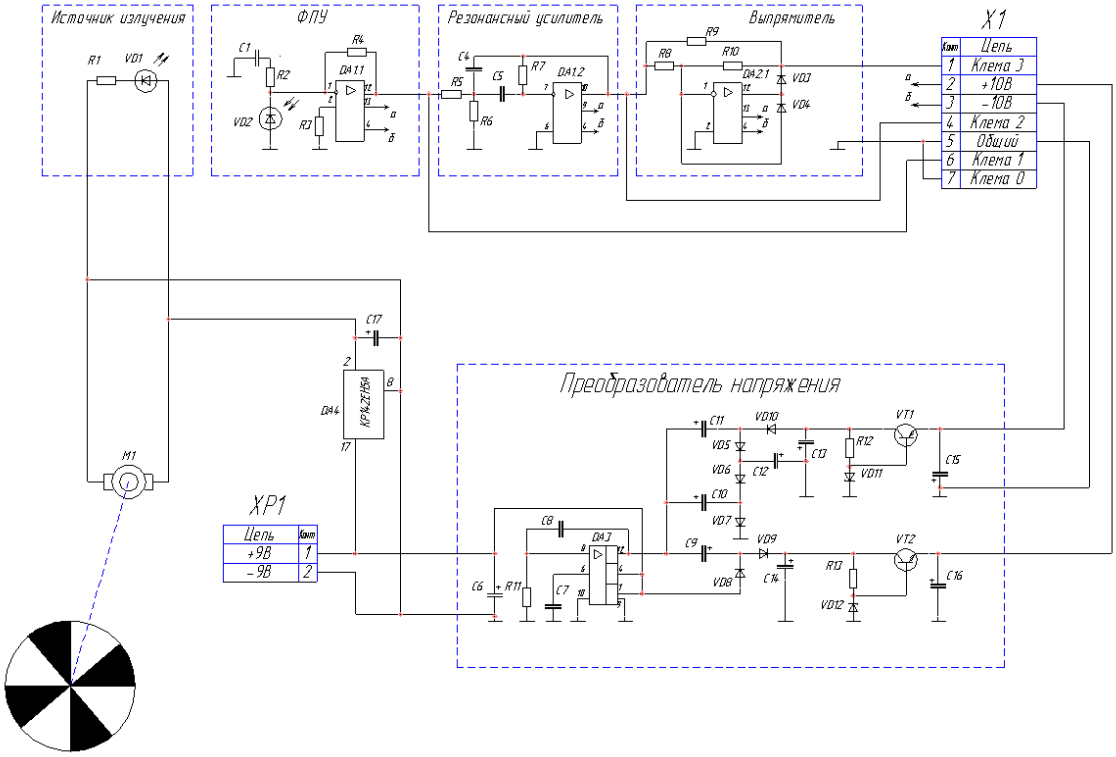


Рис. 2. Схема электрическая принципиальная

Оборудование

1. Лабораторный стенд.
2. Источник питания 9 В HY 3020.
3. Осциллограф REGOL DS1102

Список литературы

Основная литература

1. Погорельский С.Л. Прикладная оптика. Курс лекций: Учебное пособие для вузов /С.Л. Погорельский; ТулГУ – Тула: Изд-во ТулГУ, 2010. – 253 с. – 20 экз.

2. Погорельский С.Л. Прикладная оптика: учебное пособие для вузов. Ч I /С.Л. Погорельский; ТулГУ; Фак. Механики и систем управления; каф. «Приборы управления». – Тула: Гриф и К, 2005. – 186 с. –50 экз.

Дополнительная литература

1. Ю.Г. Якушенков. Теория и расчет оптико-электронных приборов. Учебник для вузов. – 5-е изд., перераб. и доп. – М.: Логос, 2004. – 472 с.

2. Парвулюсов Ю.Б. Проектирование оптико-электронных приборов: Учеб. пособие для втузов / Ю.Б. Парвулюсов, В.П. Солдатов, Ю.Г. Якушенков; Под ред. Ю.Г Якушенкова. – М.: Машиностроение, 1990. – 431 с.

3. Мосягин Г.М. Теория оптико-электронных систем: Учебник для втузов – М.: Машиностроение, 1990. – 431 с.

Лабораторная работа №4. Сборка макета малого космического аппарата для решения задач ориентации солнечных батарей

Цель: сборка и исследование работы макета малого космического аппарата с системой ориентации солнечных батарей.

Задачи:

1. Ознакомиться с конструкцией и принципом работы основных элементов малого космического аппарата (корпус, солнечные батареи, система ориентации).

2. Собрать макет малого космического аппарата с системой ориентации солнечных батарей.

3. Изучить принцип работы системы ориентации солнечных батарей, используя датчики освещенности.

4. Провести экспериментальные исследования по определению эффективности системы ориентации солнечных батарей в различных режимах работы.

5. Оценить влияние ориентации солнечных батарей на генерируемую ими мощность.

6. Сформулировать выводы по результатам лабораторной работы.

Введение

Космическая инженерия и бортовые системы ориентации малых космических аппаратов являются чрезвычайно актуальными направлениями современной космической отрасли. Миниатюризация технологий позволяет создавать все более компактные и доступные спутники, такие как *CubeSat*, открывая новые возможности для космических миссий благодаря их гибкости, быстрому внедрению и относительно низкой стоимости. Малые спутники используются для широкого спектра задач: наблюдение Земли, связь, навигация, научные исследования, технологические демонстрации и многое другое, позволяя большему количеству организаций и стран участвовать в космической деятельности.

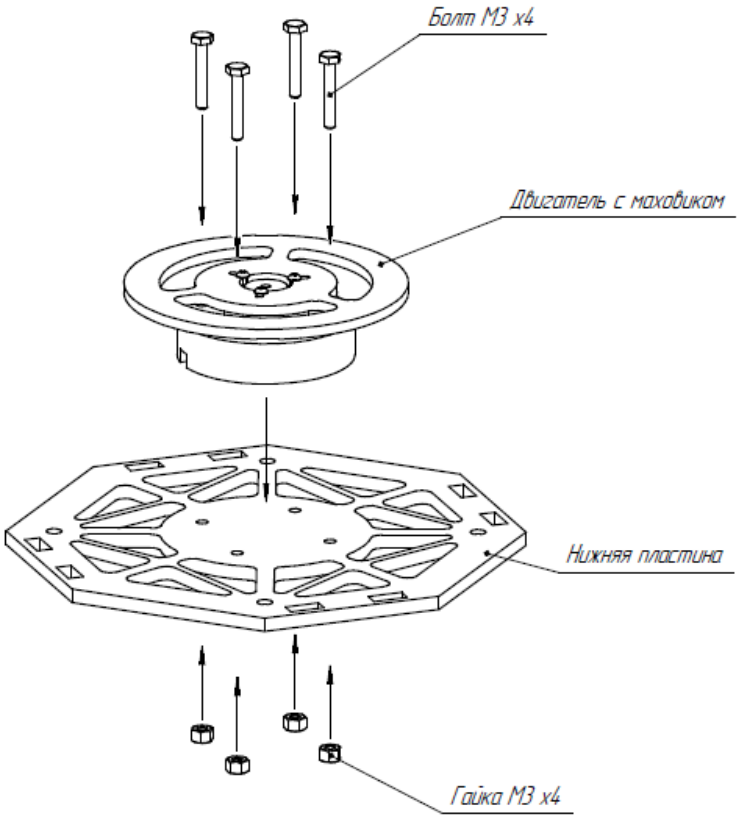
Точная ориентация и стабилизация космического аппарата (КА) имеют критическое значение для выполнения большинства космических миссий, и потребности в точности ориентации постоянно растут, особенно для задач наблюдения Земли, астрономических наблюдений и научных исследований. Для малых космических аппаратов (МКА) особенно важны компактность, энергоэффективность и надежность систем ориентации, что создает технологические вызовы по разработке высокоточных, миниатюрных и энергоэкономичных датчиков, исполнительных механизмов и алгоритмов управления ориентацией.

Развитие коммерческой космической индустрии, космического туризма, спутниковой связи и навигации требует большого количества недорогих космических аппаратов, а малые спутники становятся важным инструментом для космической науки, образования и технологических инноваций, открывая новые возможности благодаря новым технологиям, таким как миниатюризация, аддитивное производство и нанотехнологии.

Таким образом, совершенствование космической инженерии, особенно в области бортовых систем ориентации малых космических аппаратов, является ключевым фактором для расширения возможностей космической деятельности и дальнейшего развития космической отрасли в целом.

Этапы сборки макетного образца малого космического аппарата

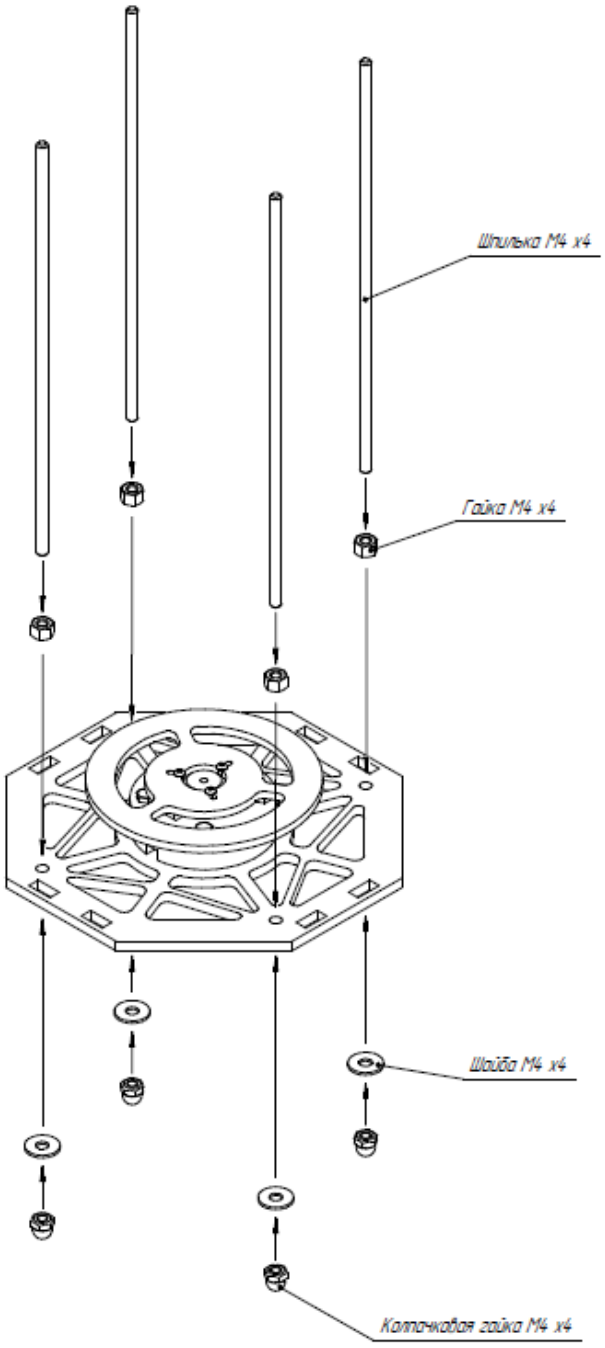
Этап 1.



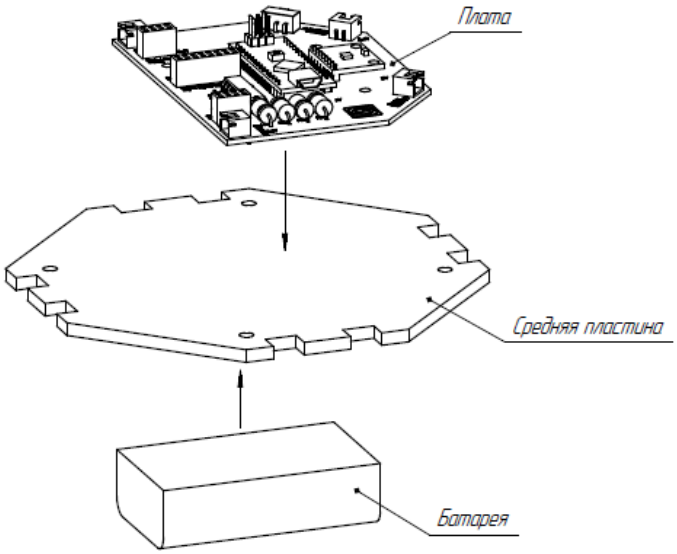
Крепление двигателя на нижней пластине

Вторым шагом следует закрепить шпильки М4х160 на нижней пластине с помощью гаек М4 и колпачковых гаек М4 с шайбами Ø4.

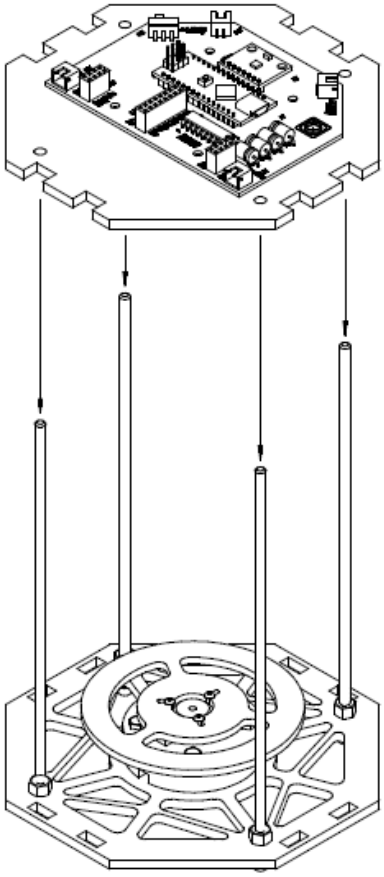
Этап 2. Крепление шпилек к нижней пластине

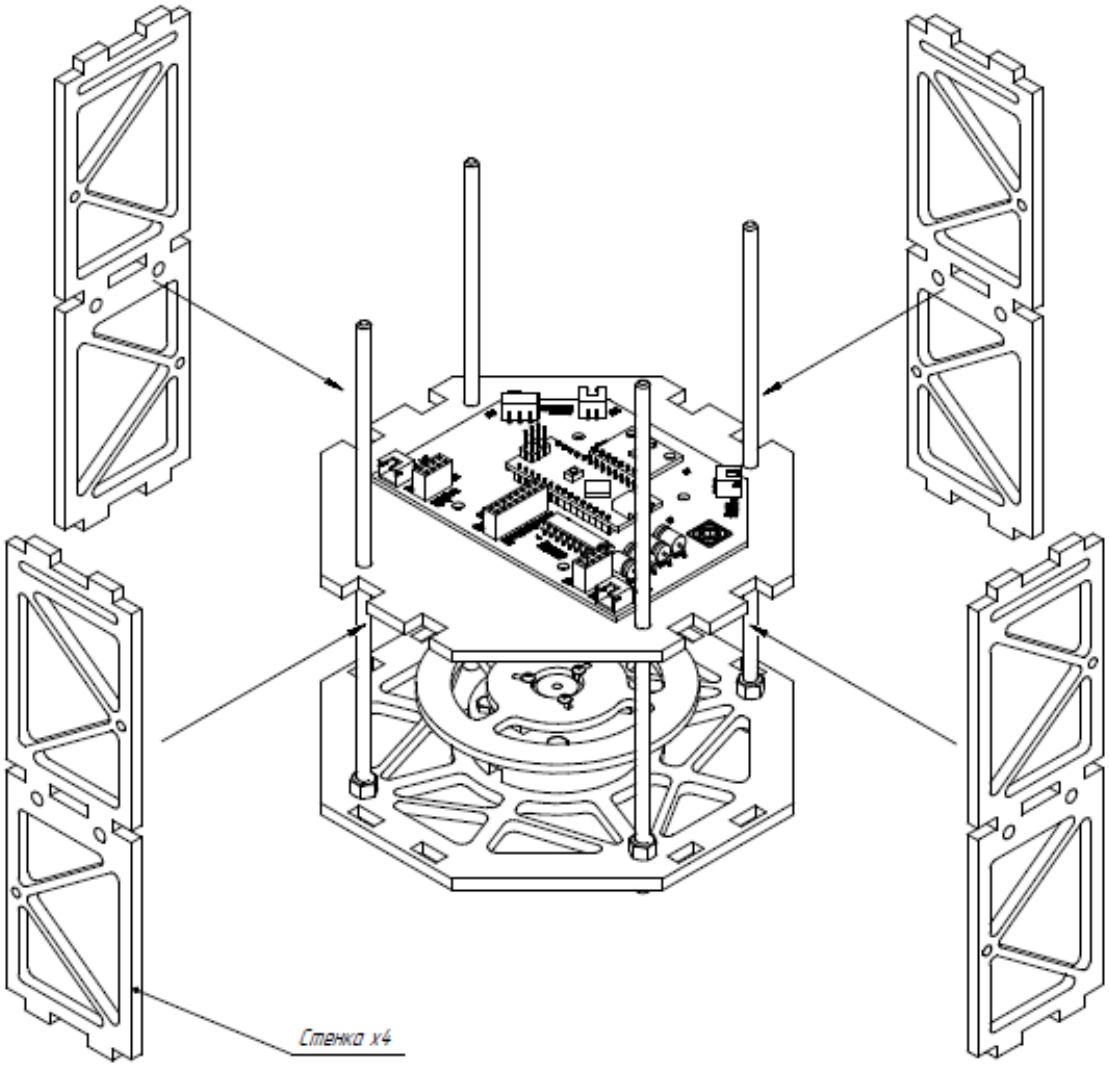


Этап 3. Далее на среднюю пластину крепятся электронная печатная плата и отсек батареи.



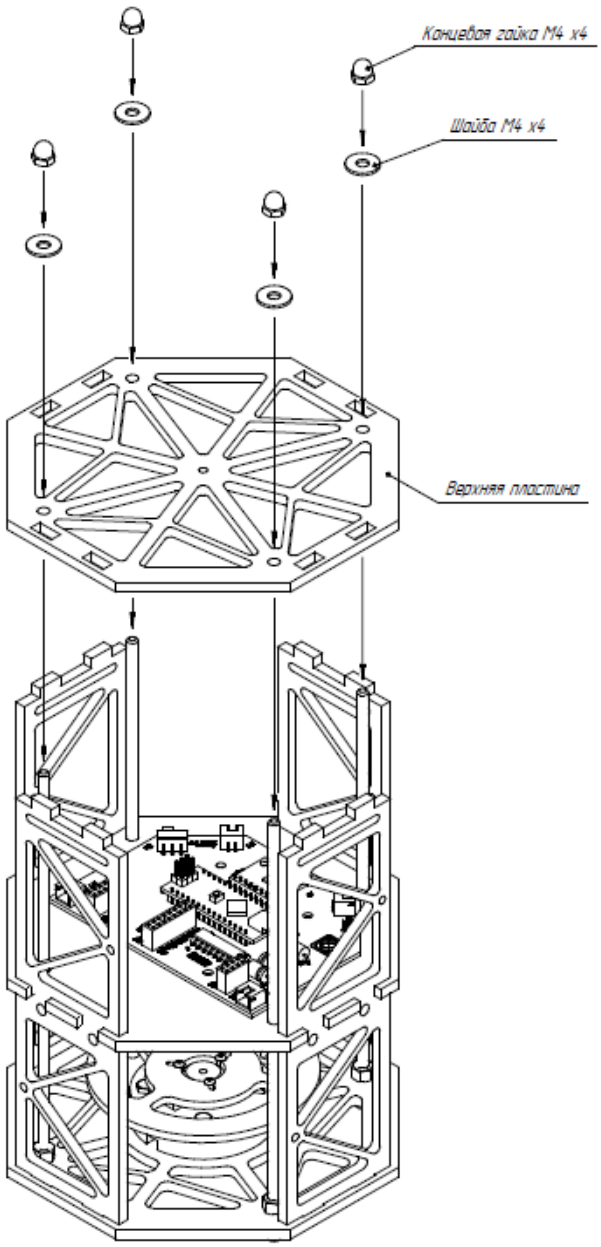
Этап 4. **С**ледующим шагом средняя пластина надевается на шпильки М4х160, которые закреплены на нижней пластине, и фиксируется с помощью стенок.





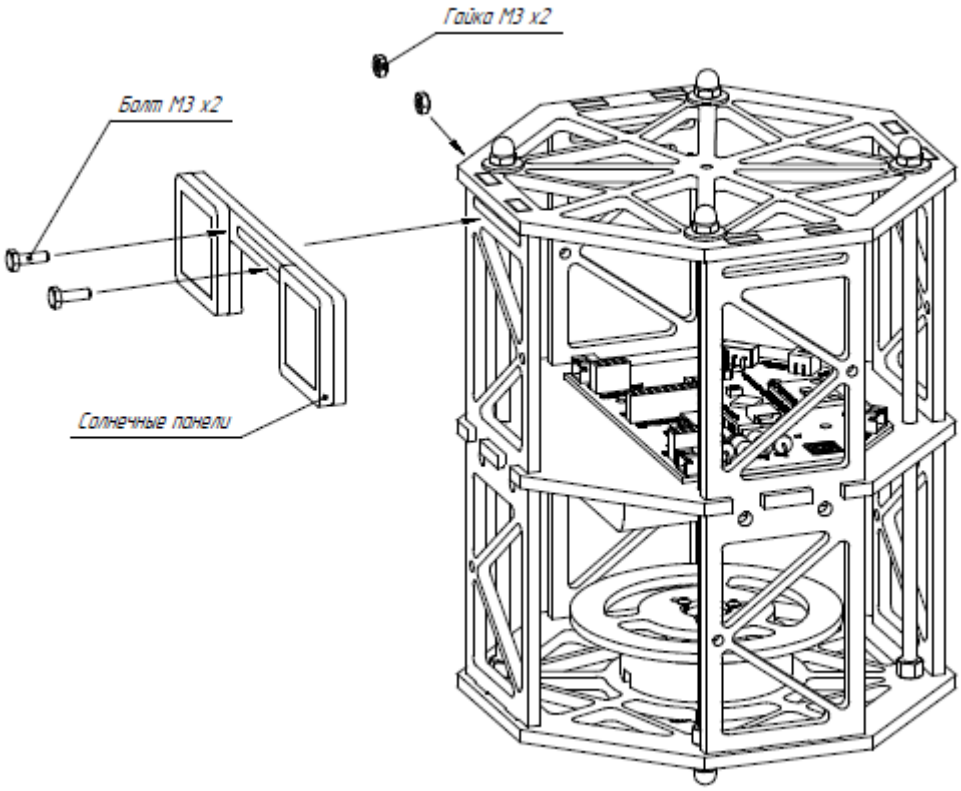
Фиксация средней пластины стенками

Этап 5. Далее конструкция закрывается верхней пластиной и закрепляется колпачковыми гайками М4 с шайбами Ø4 на шпильках М4х160.



Крепление верхней пластины

Этап 6. Последним шагом сборки является крепление солнечных панелей к одной из стенок с помощью болтов М3х10 и гаек М3 через специальный паз.



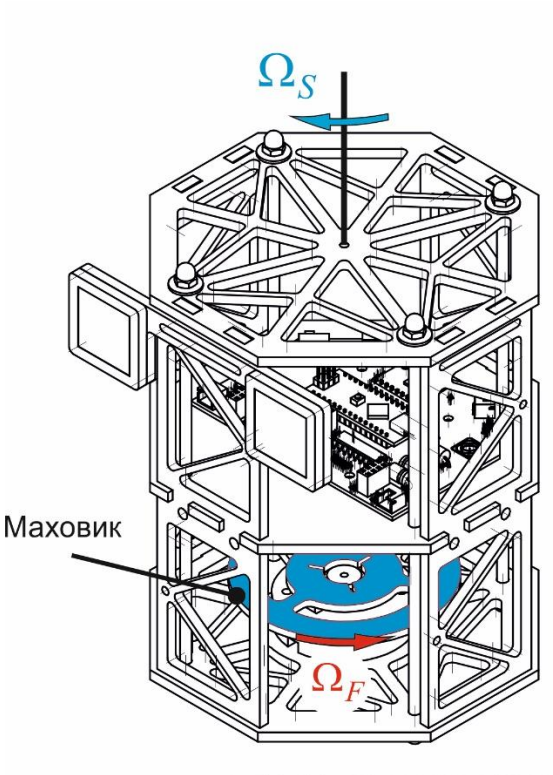
Крепление солнечных панелей к конструкции

Ориентация с помощью маховика

Ориентация и стабилизация космического аппарата с помощью маховиков является одним из ключевых элементов бортовых систем управления движением. Маховики представляют собой вращающиеся массивные инерциальные колеса, которые могут создавать управляющие моменты для изменения ориентации и стабилизации космического аппарата. Принцип работы маховиков основан на законе сохранения момента импульса. Когда маховик начинает вращаться, он накапливает момент импульса. Изменение скорости вращения маховика приводит к изменению его момента импульса, что в соответствии с законом сохранения момента импульса (1) вызывает возникновение противоположного по направлению момента силы, воздействующего на космический аппарат (рис.2).

$$J_S \Omega_S = -J_F \Omega_F, \qquad (1)$$

где J_S – момент инерции МКА, Ω_S – угловая скорость МКА, J_F – момент инерции маховика, Ω_F – угловая скорость маховика.



Ориентация МКА в пространстве

Управляя скоростью вращения маховиков, можно создавать моменты сил, необходимые для наведения космического аппарата в требуемую ориентацию или его стабилизации.

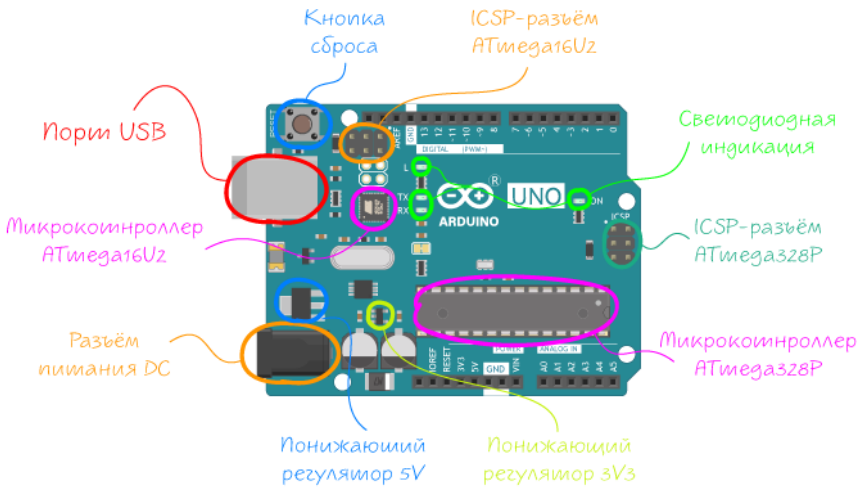
Изучение электронных компонентов

Бортовая вычислительная машина на базе Arduino Uno

Arduino Uno представляет собой популярную открытую аппаратную платформу на основе микроконтроллера *ATmega*328*P* (рис.4), широко используемую для создания самых разнообразных проектов в области электроники, робототехники, домашней автоматизации и многих других сферах. Это компактное, простое и в то же время мощное устройство, которое позволяет энтузиастам, инженерам, студентам и любителям электроники быстро прототипировать и воплощать в жизнь свои идеи.

Arduino Uno отличается доступностью, простотой в использовании, богатым набором периферийных устройств и обширным сообществом разработчиков, которые создают и делятся множеством библиотек и примеров кода.

Платформа поддерживает работу с цифровыми и аналоговыми входами/выходами, широко применяемыми для управления различными компонентами: светодиодами, моторами, датчиками, экранами и многим другим.



Элементы платы

1. Цифровые выводы (*Digital Pins*, 0-13):

- 14 цифровых выводов, пронумерованных от 0 до 13.

- Могут использоваться как входы или выходы.

- Каждый вывод может выдавать или принимать напряжение 5В.

- Вывод 0 (*RX*) и 1 (*TX*) используются для последовательной передачи данных.

- Выводы 2-13 могут использоваться для ШИМ (*Pulse Width Modulation*) на 6 выводах: 3, 5, 6, 9, 10, 11.

- Выводы 10 (*SS*), 11 (*MOSI*), 12 (*MISO*), 13 (*SCK*) используются для *SPI*-интерфейса.

2. Аналоговые входы (*Analog Inputs*, A0-A5):

- 6 аналоговых входов, пронумерованных от A0 до A5.

- Используются для считывания напряжения в диапазоне 0-5В.

- Разрешение преобразования - 10 бит (0-1023).

- Вывод A4 (*SDA*) и A5 (*SCL*) используются для I2C-интерфейса.

3. Питание:

- *Vin* - вход для подключения внешнего источника питания (7-12В).

- 5*V* - стабилизированное напряжение 5В для питания платы и подключаемых устройств.

- 3.3*V* - стабилизированное напряжение 3.3В.

- *GND* - заземление.

4. Другие порты:

- *AREF* - аналоговый эталонный вход для настройки аналоговых входов.

- *Reset* - кнопка сброса, позволяющая перезагрузить микроконтроллер.

- *ICSP* - 6-контактный разъем для программирования микроконтроллера.

- *USB* - разъем *micro-USB* для подключения к компьютеру и питания платы.

Правильное использование этих портов и интерфейсов позволяет подключать к *Arduino Uno* широкий спектр датчиков, исполнительных устройств и периферии, расширяя возможности платформы для реализации самых разнообразных проектов.

Драйвер двигателя L293D

Драйвер двигателя *L*293*D* - это интегральная микросхема, широко используемая для управления небольшими электрическими двигателями постоянного тока в электронных схемах и робототехнических проектах. Данный микросхемный модуль играет ключевую роль в реализации систем управления движением, позволяя подключать и контролировать двигатели с минимальными затратами.

*L*293*D* представляет собой четырехканальный Н-мост, который обеспечивает возможность независимого управления направлением вращения и скоростью вращения до двух двигателей постоянного тока (рис.6). Благодаря встроенным выходным каскадам на основе биполярных транзисторов, микросхема способна коммутировать токи до 600 мА на канал, что достаточно для питания небольших двигателей постоянного тока, используемых в различных робототехнических и автоматизированных устройствах.

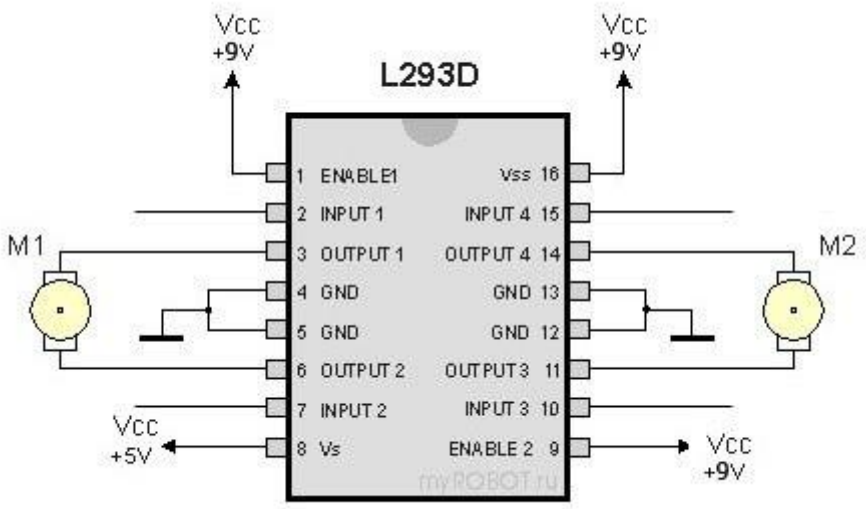


Схема микросхемы *L*293*D*

Рассмотрим выводы микросхемы и их назначение:

1. Входы *ENABLE1* и *ENABLE2* отвечают за включение каждого из драйверов, входящих в состав микросхемы.

2. Входы *INPUT1* и *INPUT2* управляют двигателем, подключенным к выходам *OUTPUT1* и *OUTPUT2*.

3. Входы *INPUT3* и *INPUT4* управляют двигателем, подключенным к выходам *OUTPUT3* и *OUTPUT4*.

4. Контакт *Vs* подключается к положительному полюсу источника питания двигателей или просто к положительному полюсу источника питания, если источник питания цепи и двигателей одинаков. Проще говоря, этот контакт отвечает за питание электродвигателей.

5. Контакт *Vss* подключен к положительному полюсу источника питания. Этот вывод обеспечивает питание самого чипа.

6. Четыре штыря *GND* подключены к "земле" (общий кабель или отрицательный полюс источника питания). Кроме того, эти контакты обычно обеспечивают теплоотвод чипа, поэтому лучше всего, чтобы они были припаяны на достаточно широкой контактной площадке.

Благодаря своим характеристикам, простоте подключения и управления, микросхема L293D широко применяется в различных робототехнических и электронных проектах для управления двигателями постоянного тока.

Микромеханический гироскоп MPU6050

Микромеханический гироскоп *MPU*6050 (рис.7) является ключевым элементом многих современных систем ориентации и навигации, используемых в робототехнике, беспилотной авиации, бытовой электронике и других прикладных областях. Данный шестиосевой инерциальный измерительный модуль (*IMU*) объединяет в одном чипе трехосевой гироскоп и трехосевой акселерометр, обеспечивая комплексную информацию об ориентации и движении устройства.

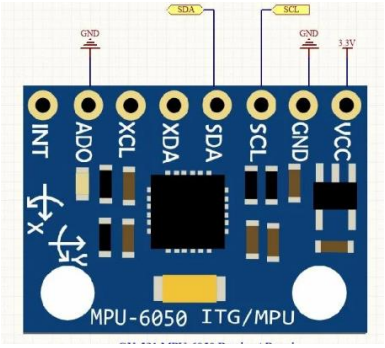


Рисунок 7 – Схема модуля *MPU*6050

MPU6050 основан на технологии микроэлектромеханических систем (МЭМС), которая позволяет интегрировать высокоточные датчики движения в компактные, энергоэффективные и недорогие интегральные схемы. Микрогироскоп в составе *MPU*6050 способен измерять угловые скорости с диапазоном до ±250, ±500 или ±1000 градусов в секунду, в зависимости от настроек, что делает его пригодным для широкого круга применений, от стабилизации изображения до систем управления движением.

Благодаря высокой точности, малым размерам и низкому энергопотреблению, *MPU*6050 широко используется в различных приложениях, где требуется определение ориентации и движения устройств.

Порядок выполнения работы:

1. Соберите схему подключения светодиода к *Arduino*:

- Соедините один вывод светодиода с цифровым контактом (например, контакт 13) на плате *Arduino*.

- Соедините другой вывод светодиода с резистором 10 КОм.

- Другой конец резистора подключите к земле (*GND*) на плате *Arduino*.

2. Подключите плату *Arduino* к компьютеру с помощью *USB*-кабеля.

3. Напишите программу для управления светодиодом:

- Откройте среду разработки *Arduino IDE*.

- Создайте новый скетч (файл -> Новый).

Объяснение:

- В начале программы мы определяем номер цифрового контакта, к которому подключен светодиод (в данном случае 13).

- В функции `setup()` мы настраиваем этот контакт как выход с помощью `pinMode(ledPin, OUTPUT)`.

- В функции `loop()` мы поочередно включаем и выключаем светодиод с помощью `digitalWrite(ledPin, HIGH)` и `digitalWrite(ledPin, LOW)`.

- Для задержки между включением и выключением используется функция `delay(1000)`, которая ожидает 1000 миллисекунд (1 секунду).

4. Загрузите программу в плату *Arduino*:

- Нажмите кнопку "Загрузить" (стрелка вверх) в среде *Arduino IDE*.

- Дождитесь загрузки программы и ее запуска на плате.

Результат:

После загрузки программы в плату *Arduino*, светодиод должен начать мигать с периодом в 1 секунду - включение на 1 секунду, выключение на 1 секунду и так далее.

Таким образом, мы реализовали простейшую программу для управления светодиодом, подключенным к плате *Arduino.*

Листинг программы

```
// Определяем номер цифрового контакта, к которому
подключен светодиод
int ledPin = 13;
void setup() {
  // Настраиваем выбранный контакт как выход
  pinMode(ledPin, OUTPUT);
}
void loop() {
  // Включаем светодиод
  digitalWrite(ledPin, HIGH);
  delay(1000); // Ждем 1 секунду
  // Выключаем светодиод
  digitalWrite(ledPin, LOW);
  delay(1000); // Ждем 1 секунду
}

  // Вычисление ошибки управления
  float error = target - angle;
  integralError += error * (currentTime - startTime) /
1000.0;

  // Вычисление управляющего воздействия
  float control = kp * error + ki * integralError;

  // Управление двигателем-маховиком
  if (control > 0) {
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
  } else {
    digitalWrite(in1, LOW);
```

```
    digitalWrite(in2, HIGH);
  }

  // Вывод данных в Serial Monitor
  Serial.print("Angle: ");
  Serial.print(angle);
  Serial.print(" rad, Control: ");
  Serial.println(control);

  // Задержка для стабилизации
  if (currentTime - startTime < 2000) {
    delay(10);
  } else {
    startTime = millis();
  }
}
```

Лабораторная работа № 5. Исследование режима автосопровождения солнечного светила посредством солнечных батарей и двигателя-маховика по одной оси

Цель работы:

Изучить принцип построения системы автоматического слежения за положением солнца с использованием фоторезисторов и двигателя-маховика для компенсации углового рассогласования по одной оси.

Оборудование:

- Микроконтроллер *Arduino Uno*
- Фоторезисторы для определения положения солнца (2 шт.)
- Двигатель постоянного тока с маховиком
- Драйвер двигателя *L293D*
- Источник питания
- Соединительные провода

Схема подключения:

1. Подключите питание:
- Подключите +5*V* Arduino к *Vcc* драйвера *L293D*.
- Подключите *GND Arduino* к *GND* драйвера *L293D*.
2. Подключите двигатель-маховик:
- Подключите вывод A двигателя к 1A или 2A драйвера *L293D*.
- Подключите вывод B двигателя к 3A или 4A драйвера *L293D*.
- Подключите общий вывод двигателя к *GND* драйвера *L293D*.
3. Подключите солнечный батареи:
- Подключите 2 фоторезистора к аналоговым пинам *Arduino* (например, A0, A1).
- Подключите другие выводы фоторезисторов к *GND*.
4. Подключите управляющие линии:

- Подключите вывод 1,2 (для управления направлением) к цифровым пинам *Arduino* (например, 9, 10).

- Подключите вывод 3,6 (для управления скоростью) к ШИМ-пинам Arduino (например, 5).

Объяснение:

1. *in*1 и *in*2 - выводы драйвера *L293D*, используемые для управления направлением вращения двигателя.

2. *pwm* - вывод *Arduino*, используемый для управления скоростью вращения двигателя.

3. `photocellPins[]` - массив аналоговых пинов, к которым подключены фоторезисторы.

4. `targetAngle` - целевой угол положения, на данный момент не используется.

5. `currentAngle` - текущий угол положения, который отслеживается и обновляется в процессе работы.

Ход работы:

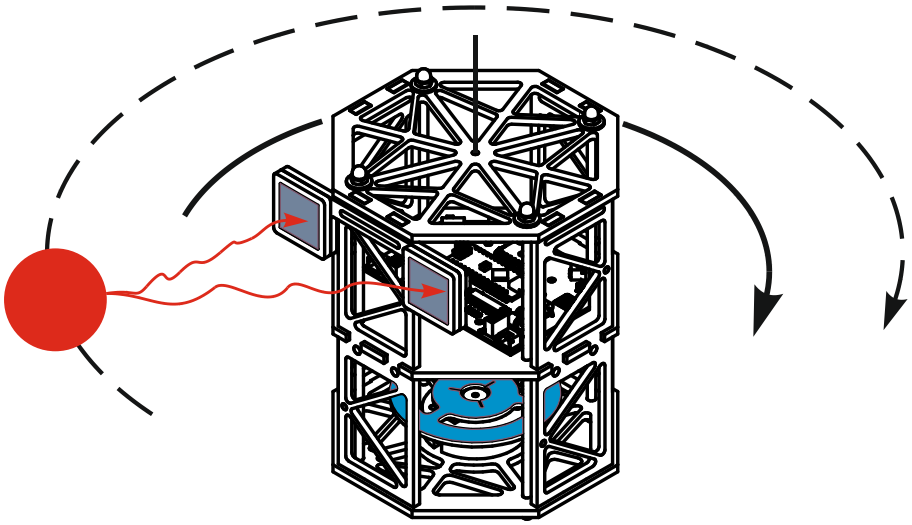
1. Соберите схему согласно приведенной выше.

2. Загрузите код в *Arduino*.

3. Откройте *Serial Monitor* в *Arduino IDE*.

4. Наблюдайте за показаниями фоторезисторов и положением двигателя-маховика в *Serial Monitor*.

5. Перемещайте источник света (например, лампу) вокруг установки и наблюдайте, как двигатель-маховик компенсирует изменение положения.



Указание.
Использовать сигналы, считываемые с аналоговых портов контроллера:

```
Sun1 = analogRead(A0);
Sun2 = analogRead(A1);
```

Сформировать разностный сигнал и задать соответствующее направление вращения маховика.

Вывод:

В ходе выполнения данной лабораторной работы вы изучили принципы построения системы автоматического слежения за положением солнца с использованием фоторезисторов и двигателя-маховика по одной оси. Вы научились использовать фоторезисторы для определения положения источника света, вычислять угол рассогласования и управлять двигателем-маховиком для компенсации этого рассогласования.

Данная система может быть использована в солнечных трекерах, системах слежения за источниками света, роботизированных платформах для наблюдения и другие приложениях, где требуется отслеживание положения светового источника по одной оси.

Листинг программы:

```
int in1 = 9;
```

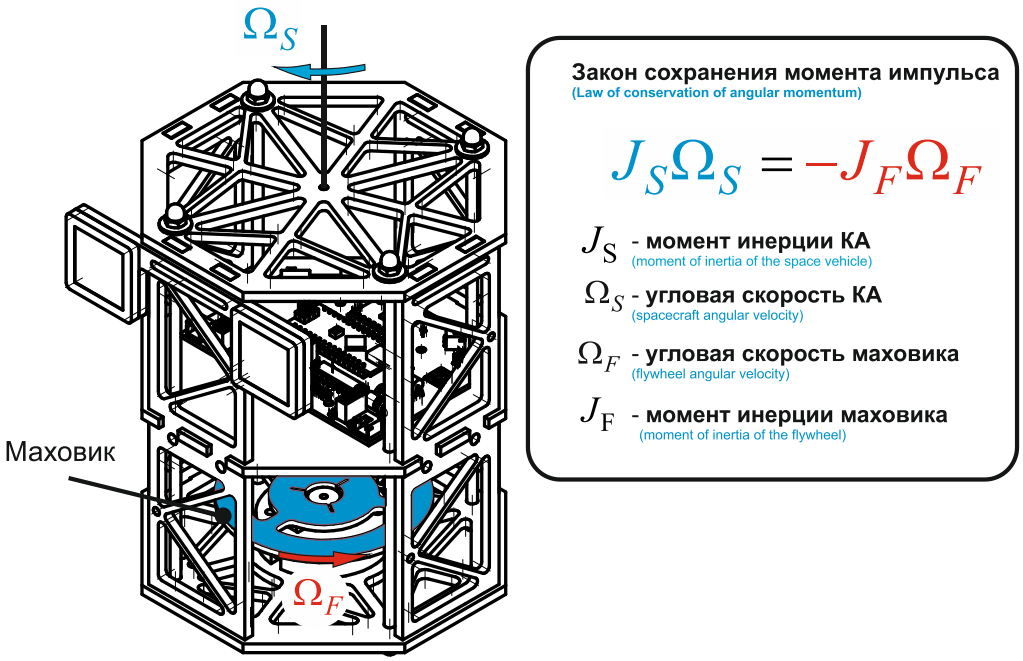
```
int in2 = 10;
int pwm = 5;
int photocellPins[] = {A0, A1};

int targetAngle = 0;
int currentAngle = 0;

void setup() {
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(pwm, OUTPUT);
  for (int i = 0; i < 2; i++) {
    pinMode(photocellPins[i], INPUT);
  }
  Serial.begin(9600);
}
void loop() {
  // Чтение показаний фоторезисторов и определение положения солнца
  int sunPos = readSunPosition(photocellPins[0], photocellPins[1]);

  // Вычисление угла рассогласования
  int error = map(sunPos, -512, 512, -90, 90) - currentAngle;

  // Управление двигателем-маховиком
  if (error > 0) {
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
```

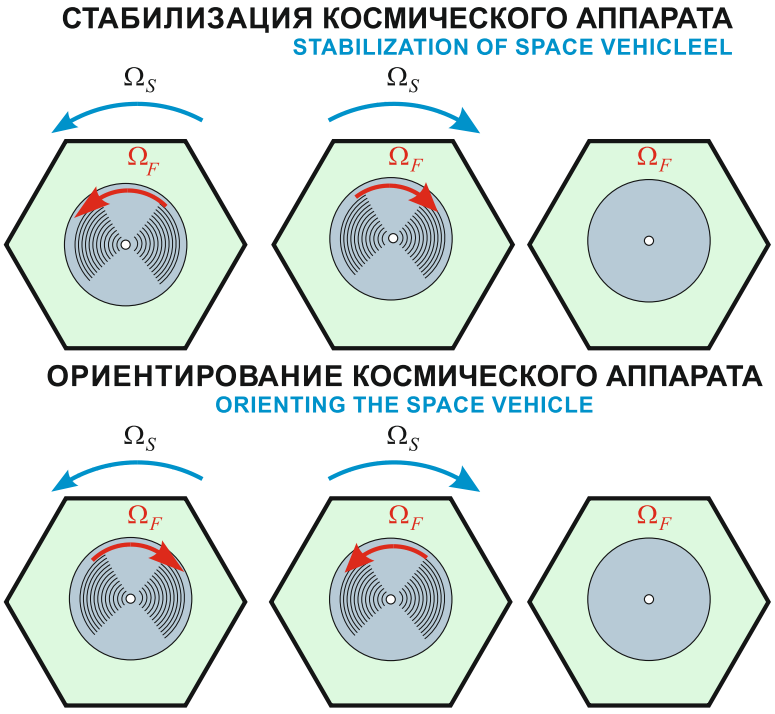
```
  } else {
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
  }
  int speed = abs(error * 2);
  analogWrite(pwm, speed);

  // Обновление текущего угла
  currentAngle += error;

  // Вывод информации в Serial Monitor
  Serial.print("Sun Position: ");
  Serial.print(sunPos);
  Serial.print(", Current Angle: ");
  Serial.println(currentAngle);

  delay(100);
}
int readSunPosition(int pin1, int pin2) {
  int val1 = analogRead(pin1);
  int val2 = analogRead(pin2);
  return val1 - val2;
}
```

Лабораторная работа № 6 Разработка алгоритма стабилизации малого космического аппарата по угловой скорости

Цель работы:

Разработать алгоритм стабилизации малого космического аппарата по угловой скорости при начальной закрутке, используя сигнал с микромеханического гироскопа.





Оборудование:

- Плата *Arduino* (например, *Arduino Uno*)

- Микромеханический гироскоп (например, *MPU*-6050)

- Двигатель-маховик (например, *DC*-мотор)

- Драйвер для управления двигателем (например, *L293D*)

- Источник питания для двигателя

Порядок выполнения:

1. Подключите микромеханический гироскоп к плате *Arduino*:

- Подключите *VCC* гироскопа к 3.3В платы *Arduino*.

- Подключите *GND* гироскопа к *GND* платы *Arduino*.

- Подключите *SCL* гироскопа к *SCL* (A5) платы *Arduino*.

- Подключите *SDA* гироскопа к *SDA* (A4) платы *Arduino*.

2. Подключите двигатель-маховик к драйверу *L293D*:

- Подключите положительный вывод двигателя к выходу *IN*1 драйвера.

- Подключите отрицательный вывод двигателя к выходу *IN*2 драйвера.

- Подключите *GND* драйвера к земле (*GND*) платы *Arduino*.

- Подключите питание драйвера (*VCC*) к источнику питания двигателя.

3. Подключите драйвер *L293D* к плате *Arduino*:

- Подключите вход *IN*1 драйвера к цифровому пину 9 платы *Arduino*.

- Подключите вход *IN*2 драйвера к цифровому пину 10 платы *Arduino*.

- Подключите землю (*GND*) драйвера к земле (*GND*) платы *Arduino*.

4. Подключите плату *Arduino* к компьютеру с помощью *USB*-кабеля.

5. Напишите программу для стабилизации малого космического аппарата:

Объяснение:

- Откройте среду разработки *Arduino IDE*.

- Создайте новый скетч (файл -> Новый).

Подключение библиотек и объявление переменных:

- Мы включаем библиотеку `MPU6050.h`, которая позволяет работать с микромеханическим гироскопом *MPU*-6050.

- Создаем объект *mpu* для взаимодействия с гироскопом.

- Объявляем переменные *in*1 и *in*2 для управления двигателем-маховиком.

- Объявляем переменные `gyroX, gyroY, gyroZ` для хранения данных с гироскопа.

- Объявляем переменную `angularVelocity` для вычисления угловой скорости.

- Объявляем коэффициент пропорционального регулирования `kp` и целевую угловую скорость `target`.

- Объявляем переменные `startTime` и `currentTime` для отслеживания времени.

Функция `setup()`:

- В функции `setup()` мы:

- Инициализируем шину *I2C*, используя `Wire.begin()`.

- Инициализируем гироскоп *MPU*-6050, используя `mpu.initialize()`.

- Настраиваем пины *in*1 и *in*2 как выходные, чтобы управлять двигателем-маховиком.

- Сохраняем текущее время в переменной `startTime`, чтобы отслеживать время работы программы.

Функция `loop()`:

- В функции `loop()` мы:

- Обновляем значение `currentTime` с помощью `millis()`.

- Считываем данные с гироскопа, используя `mpu.getRotation()`, и вычисляем угловую скорость в радианах в секунду.

- Вычисляем ошибку между целевой угловой скоростью (`target`) и текущей угловой скоростью (`angularVelocity`).

- Вычисляем управляющее воздействие, умножая ошибку на коэффициент пропорционального регулирования (`kp`).

- Управляем двигателем-маховиком, устанавливая состояния пинов *in*1 и *in*2 в зависимости от знака управляющего воздействия.

- Выводим текущую угловую скорость и управляющее воздействие в `Serial Monitor` для отладки.

- Добавляем задержку на 10 мс для стабилизации, кроме первых 2 секунд, чтобы дать системе время на начальную стабилизацию.

- Сбрасываем значение `startTime` для повторения цикла.

- В функции loop() мы:

- Обновляем значение `currentTime` с помощью `millis()`.

- Считываем данные с гироскопа, используя `mpu.getRotation()`, и вычисляем угловую скорость в радианах в секунду.

- Вычисляем ошибку между целевой угловой скоростью (`target`) и текущей угловой скоростью (`angularVelocity`).

- Вычисляем управляющее воздействие, умножая ошибку на коэффициент пропорционального регулирования (`kp`).

- Управляем двигателем-маховиком, устанавливая состояния пинов *in*1 и *in*2 в зависимости от знака управляющего воздействия.

- Выводим текущую угловую скорость и управляющее воздействие в `Serial Monitor` для отладки.

- Добавляем задержку на 10 мс для стабилизации, кроме первых 2 секунд, чтобы дать системе время на начальную стабилизацию.

- Сбрасываем значение `startTime` для повторения цикла.

6. Загрузите программу в плату *Arduino*:

- Нажмите кнопку "Загрузить" (стрелка вверх) в среде *Arduino IDE*.

- Дождитесь загрузки программы и ее запуска на плате.

Таким образом, программа будет поддерживать нулевую угловую скорость малого космического аппарата, компенсируя его начальную закрутку.

Листинг программы:

```
#include <Wire.h>
#include <MPU6050.h>

MPU6050 mpu;

int in1 = 9;
int in2 = 10;

int16_t gyroX, gyroY, gyroZ;
float angularVelocity;

float kp = 0.1; // Коэффициент пропорционального
регулирования
float target = 0.0; // Целевая угловая скорость (0
рад/с)

unsigned long startTime;
unsigned long currentTime;

void setup() {
  Wire.begin();
  mpu.initialize();

  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);

  startTime = millis();
}

void loop() {
```

```
  currentTime = millis();

  // Считывание данных с гироскопа
  mpu.getRotation(&gyroX, &gyroY, &gyroZ);
  angularVelocity = radians(gyroZ / 131.0); // Перевод
в рад/с

  // Вычисление ошибки управления
  float error = target - angularVelocity;

  // Вычисление управляющего воздействия
  float control = kp * error;

  // Управление двигателем-маховиком
  if (control > 0) {
    digitalWrite(in1, HIGH);
    digitalWrite(in2, LOW);
  } else {
    digitalWrite(in1, LOW);
    digitalWrite(in2, HIGH);
  }

  // Вывод данных в Serial Monitor
  Serial.print("Angular Velocity: ");
  Serial.print(angularVelocity);
  Serial.print(" rad/s, Control: ");
  Serial.println(control);

  // Задержка для стабилизации
  if (currentTime - startTime < 2000) {
```

```
    delay(10);
  } else {
    startTime = millis();
  }
}
```