

Работа призера заключительного этапа
командной инженерной олимпиады школьников
Олимпиада Национальной технологической инициативы

Профиль «Системы связи и дистанционного зондирования Земли»

Сечинский Егор Валерьевич

Класс: 10

Город: Санкт-Петербург

Школа: ГБОУ Лицей №393 Кировского
района Санкт-Петербурга

Регион: Санкт-Петербург

Уникальный номер участника:

**Команда на заключительном
этапе:** SMAAAAW

шифр: 630570

stepik ID: 30232196

Параллель: 10-11

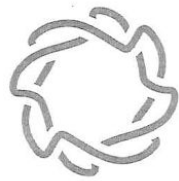
Результаты заключительного этапа:

ФИО	Командный балл max(215)	Физика max(60)	Информатика max(100)	Общий балл max(100)
Сечинский Егор Валерьевич	125,7	0	0	39,10666667

Индивидуальная часть

Физика

Персональный лист участника с номером 630570:



Олимпиада НТИ

ФИО Сосинский Егор Валерьевич

Город Санкт-Петербург

Школа № 393

Командная инженерная олимпиада «Олимпиада НТИ»

Направление Системы связи и Дистанционное зондирование Земли

Предмет Физика

Номер участника 630570

Время: 12:28 - 12:32



Индивидуальная часть

Информатика

Решения участника с stepik ID 30232196

Участником решений не предоставлено.

Командная часть

Результаты были получены в рамках выступления команды: SMAAAAW

Личный состав команды:

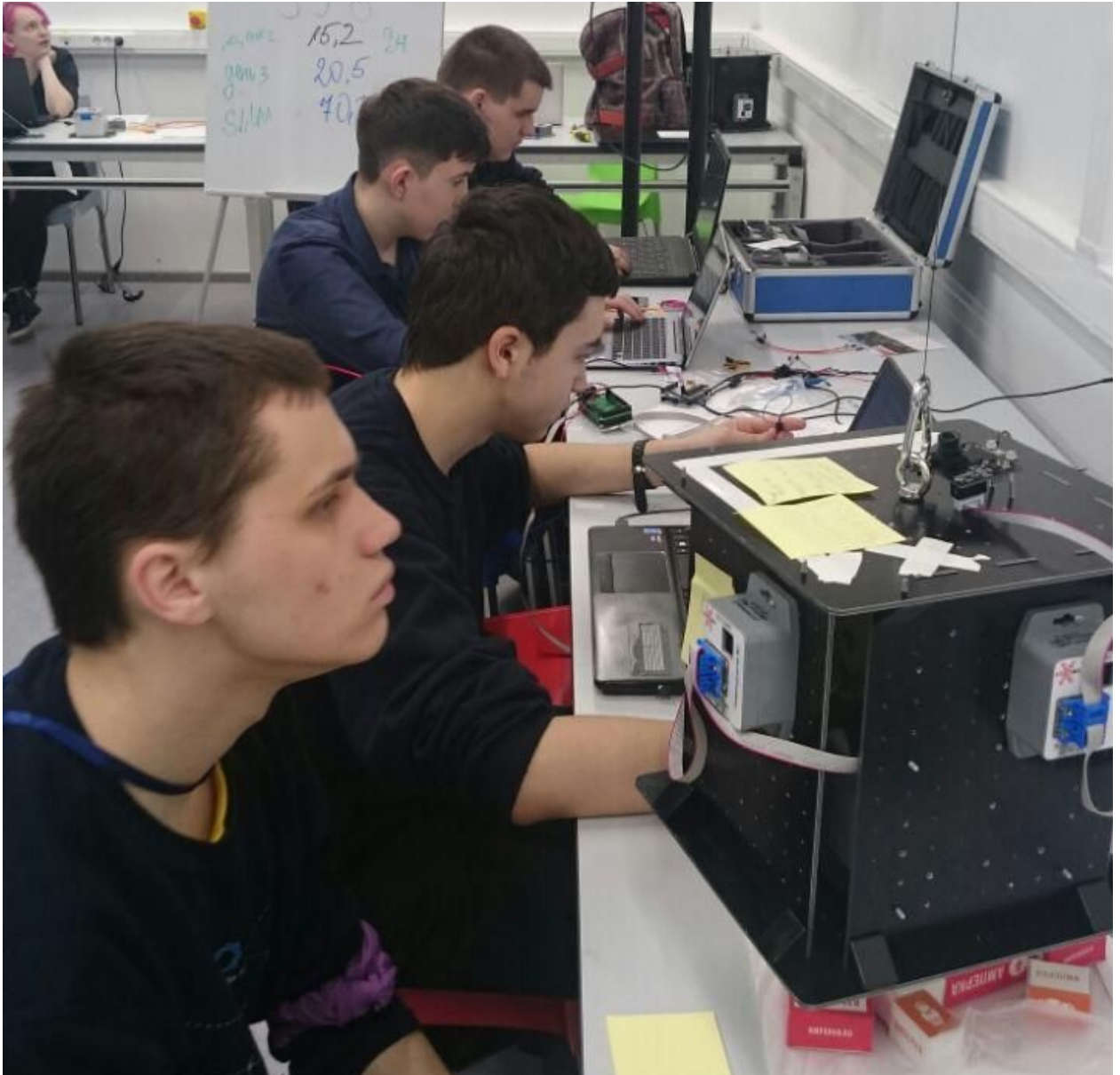
- Ярмолинский Арсений Маркович
- Сечинский Егор Валерьевич
- Рязанцев Федор Дмитриевич
- Пэвхенен Вячеслав Максимович

Фото команды:



1. Сборка типовой платформы

На фото команда с собранным устройством в процессе работы:



2. Код, написанный командой для решения задачи ориентации (включает в себя подзадачи по тестированию устройств, калибровке магнитометра и задаче стабилизации):

```
#include "libschat.h"

/*
** Lab 2: get a raw data from a magnetometer
*/

void magCalibration(int xin, int yin, int zin, float *x, float *y, float *z)
{
    float A[3][3] = {{-0.037, -0.004, -3.876},
```

```

        {1.169, 1.101, -0.981},
        {-0.336, 0.001, -0.104}}};
float b[3] = {2.201, 5.279, -2.039};
//float xin = *x, yin = *y, zin = *z;
*x = (A[0][0])*(xin-b[0])+(A[0][1])*(yin-b[1])+(A[0][2])*(zin-b[2]);
*y = (A[1][0])*(xin-b[0])+(A[1][1])*(yin-b[1])+(A[1][2])*(zin-b[2]);
*z = (A[2][0])*(xin-b[0])+(A[2][1])*(yin-b[1])+(A[2][2])*(zin-b[2]);
}

void control(void)
{
    int u = 0, err = 0, erold = 0, errsum = 0, speed = 0;
    double kpm = 100, kim = 0.005, kdm = -1, ke = 10;
    double kps = 100, kis = 0, kds = 0;
    int target;
    int i;
    int16_t temp;
    const int num = 1; /* magnetometer #1 */
    const int motorNum = 1, gyroNum = 1;
    double alpha = 0;
    printf("Enable magnetometer #%d\n", num);
    magnetometer_turn_on(num);
    if(LSS_OK != motor_turn_on(motorNum)) {printf("Motor connection fail!"); return;}
    if(LSS_OK != hyro_turn_on(gyroNum)) {printf("Gyroscope connection fail!");
return;}
    printf("Connection complete!");
    float xf, yf, zf;
    Sleep(2);
    int16_t x, y, z;
    if (LSS_OK == magnetometer_request_raw(num, &x, &y, &z)) {
    magCalibration(x, y, z, &xf, &yf, &zf);
    alpha = atan2(xf, yf)*180.0/3.1415926;
    printf("%f\n", alpha);
    //printf("%d: x=%d y=%d z=%d\n", i, x, y, z);
    } else {
    puts("Fail!");
    }
    target = 0/*alpha + 180 < 180 ? alpha + 180 : alpha - 180*/;
    for (i = 0; i < 6000; i++) {
    if (LSS_OK == magnetometer_request_raw(num, &x, &y, &z)) {
        magCalibration(x, y, z, &xf, &yf, &zf);
        alpha = atan2(xf, yf)*180.0/3.1415926;
        printf("%f\n", alpha);
        //printf("%d: x=%d y=%d z=%d\n", i, x, y, z);
    } else {
        puts("Fail! Mag");
    }
    }

    err = alpha - target;
    speed = err * kps;
    u = speed * kds;

    if(u > 3000) u = 3000;

```

```

if(u < -3000) u = -3000;

/*err = (target - alpha)*ke;
errsum += err;

speed = kps*err + kis * errsum + kds * (err - errold);

if(speed > 1000) speed = 1000;
if(speed < -1000) speed = -1000;

if (LSS_OK == gyro_request_raw(gyroNum, &x, &y, &z))
{
}
else
{
    puts("Fail! Gyro");
}

u = kpm*(z-speed);

if(u > 3000) u = 3000;
if(u < -3000) u = -3000;*/

motor_set_speed(motorNum, u, &temp);
//printf("    Motor speed: %d\n", temp);

errold = err;

Sleep(0.01);
}
printf("Disable magnetometer #%d\n", num);
magnetometer_turn_off(num);
motor_set_speed(motorNum, 0, &temp);
printf("Disable motor #%d\n", motorNum);
motor_turn_off(motorNum);
}

```

3. Работа с оптическим каналом (включает в себя все подзадачи, связанные с оптическим каналом, а также задачу по интеграции)

Код для БКУ, передающий данные:

```

#include "libschat.h"
/*
** Lab 9: UHF transceiver demo.
*/
void control(void)
{
    const uint16_t tx_num = 2;
    const uint16_t rx_num = 1;
    const uint8_t hello[] = {0x42, 0x4D, 0x7E, 0x06, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x3E, 0x00, 0x00, 0x00, 0x28, 0x00,

```


0x00, 0x00, 0x64, 0x00, 0x00, 0x00, 0x64,
0x00, 0x00, 0x00, 0x01, 0x00, 0x01, 0x00, 0x00, 0x00,
0x00, 0x00, 0x40, 0x06, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0xF8, 0x3F, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0xFF, 0x80, 0x0F, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0xFE, 0x00, 0x07, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0xF8, 0x00, 0x07, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x07, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0xC0, 0x1F, 0x83, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0x80, 0x7F, 0x8F, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFF, 0x01, 0xFF, 0xBF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFE, 0x07, 0xFF, 0xFF, 0xF0, 0x00, 0x3F,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFE, 0x0F, 0xFF, 0xFF, 0x80, 0x00, 0x07,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xFF, 0xFC, 0x0F, 0xFF, 0xFC, 0x00, 0x00, 0x01,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xC0, 0xFC, 0x1F, 0xFF, 0xF0, 0x00, 0x00, 0x00,
0x7F, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0x80, 0x7C, 0x3F, 0xFF, 0xC0, 0x03, 0xFE,
0x00, 0x1F, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0x00, 0x3E, 0x7F, 0xFF, 0x00, 0x1F, 0xFF, 0xE0,
0x0F, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFE,
0x00, 0x3E, 0xFF, 0xFC, 0x00, 0xFF, 0xFF, 0xF8,
0x07, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFC,
0x0C, 0x1F, 0xFF, 0xF0, 0x03, 0xFF, 0xFF, 0xFE,
0x03, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xF8,
0x1C, 0x1F, 0xFF, 0xE0, 0x0F, 0xFF, 0xFF, 0xFF,
0x81, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xF8,
0x3E, 0x0F, 0xFF, 0x80, 0x3F, 0xFF, 0xFF, 0xFF,
0xC0, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xF0,
0x7F, 0x07, 0xFE, 0x00, 0xFF, 0xCF, 0xFF, 0xFF,
0xE0, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xF0,
0x7F, 0x07, 0xF8, 0x03, 0xFE, 0x01, 0xFF, 0xFF,
0xF0, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xE0,
0xFF, 0x83, 0xF0, 0x0F, 0xFE, 0x00, 0x3F, 0xFF,
0xF8, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xC1,
0xFF, 0x83, 0xC0, 0x1F, 0xFF, 0x00, 0x07, 0xFF,
0xF8, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xC1,
0xFF, 0xC1, 0x00, 0x7F, 0xFF, 0xE0, 0x00, 0xFF,
0xF8, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xC3,
0xFF, 0xC0, 0x01, 0xFF, 0xFF, 0xF8, 0x00, 0x1F,

0xFE, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x83,
0xFF, 0xE0, 0x07, 0xFF, 0xFF, 0xFF, 0x00, 0x03,
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x87,
0xFF, 0xF0, 0x0F, 0xFF, 0xFF, 0xFF, 0xE0, 0x00,
0x7F, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x87,
0xFF, 0xFC, 0x3F, 0xFF, 0xFF, 0xFF, 0xFC, 0x00,
0x0F, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x07,
0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xC0,
0x03, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x07,
0xFC, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xF8,
0x01, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xFC, 0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE,
0x00, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xFC, 0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xC0, 0x1F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xF8, 0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xF0, 0x0F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xF8, 0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x9F,
0xFC, 0x07, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xF8, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x83,
0xFE, 0x03, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xF8, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xC1,
0xFF, 0x81, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xF0, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xC1,
0xFF, 0xC0, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x0F,
0xF0, 0x7F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xE0,
0xFF, 0xE0, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0x07,
0xF0, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xE0,
0x7F, 0xF0, 0x70, 0x00, 0x00, 0x00, 0xFF, 0x07,
0xE0, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xF0,
0x3F, 0xF8, 0x30, 0x00, 0x00, 0x00, 0xFF, 0x87,
0xE0, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xF8,
0x3F, 0xF8, 0x30, 0x00, 0x00, 0x00, 0xFF, 0x87,
0xC1, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC,
0x1F, 0xFC, 0x30, 0x00, 0x00, 0x00, 0xFF, 0x8F,
0xC1, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC,
0x0F, 0xFC, 0x10, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0xC3, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE,
0x0F, 0xFE, 0x10, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0x83, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0x07, 0xFE, 0x10, 0x00, 0x00, 0x00, 0xFF, 0xFF,
0x07, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0x03, 0xFE, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFC,
0x07, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0x83, 0xFE, 0x00, 0x00, 0x00, 0x00, 0xFF, 0x80,
0x0F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xCF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0xE0, 0x00,
0x1F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x00,
0x3F, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x80, 0x01,
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,
0xFF, 0xC0, 0x00, 0x00, 0x00, 0x00, 0x80, 0x1F,


```
0xFC, 0x1F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xE3,  
0xFF, 0xE0, 0x00, 0x7F, 0xFF, 0xF0, 0x00, 0x1F,  
0xFC, 0x3F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xE0,  
0xFF, 0xFC, 0x00, 0x0F, 0xFF, 0xC0, 0x1C, 0x1F,  
0xF8, 0x3F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xE1,  
0xFF, 0xFF, 0xC0, 0x07, 0xFF, 0x80, 0x7E, 0x0F,  
0xF8, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xE0,  
0xFF, 0xFF, 0xF8, 0x03, 0xFE, 0x01, 0xFE, 0x0F,  
0xF0, 0x7F, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xE0,  
0x7F, 0xFF, 0xFF, 0x0F, 0xF8, 0x03, 0xFF, 0x07,  
0xE0, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xF0,  
0x3F, 0xFF, 0xFF, 0xFF, 0xE0, 0x0F, 0xFF, 0x07,  
0xC0, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xF8,  
0x1F, 0xFF, 0xFF, 0xFF, 0xC0, 0x3F, 0xFF, 0x83,  
0x81, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFC,  
0x07, 0xFF, 0xFF, 0xFF, 0x00, 0x7F, 0xFF, 0x81,  
0x03, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFE,  
0x01, 0xFF, 0xFF, 0xF8, 0x01, 0xFF, 0xF7, 0xC0,  
0x03, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0x00, 0x3F, 0xFF, 0xE0, 0x07, 0xFF, 0xE3, 0xE0,  
0x07, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0x80, 0x03, 0xFE, 0x00, 0x1F, 0xFF, 0xC3, 0xE0,  
0x0F, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xE0, 0x00, 0x00, 0x00, 0x7F, 0xFF, 0xC1, 0xF0,  
0x3F, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xF8, 0x00, 0x00, 0x01, 0xFF, 0xFF, 0x83, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0x00, 0x00, 0x0F, 0xFF, 0xFF, 0x03, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xF0, 0x00, 0x7F, 0xFF, 0xFE, 0x07, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC, 0x0F, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFF, 0x9F, 0xF0, 0x1F, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFE, 0x1F, 0xC0, 0x3F, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFE, 0x0C, 0x00, 0x7F, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFE, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x03, 0xFF, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x0F, 0xFF, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00, 0xFF, 0xFF,  
0xFF, 0xFF, 0xFF, 0xFF, 0xC0, 0xFF, 0xFF, 0xFF,  
0xFF, 0xFF, 0xF0, 0x00, 0x00, 0x00};  
printf("Enable transceiver #%d\n", tx_num);  
transceiver_turn_on(tx_num);  
motor_turn_on(1);  
Sleep(1);  
bus_setup();  
int temp;
```

```

int i = 0;
printf("Send data from #%d to #%d\n", tx_num, rx_num);
//while(i < 1662)
//{
//static int send = int(hello[i]);
transceiver_send(tx_num, rx_num, hello, sizeof(hello));
printf("Send data from #%d to #%d\n", tx_num, rx_num);
//motor_set_speed(1, send, &temp);
Sleep(100);
//Sleep(1);
//    i++;
//}
/*for(int i = 0; i < 1662; i++)
{
char abs = hello[i];
transceiver_send(tx_num, rx_num, abs, sizeof(abs));
//Sleep(0.01);
}*/
printf("Disable transceiver #%d\n", tx_num);
transceiver_turn_off(tx_num);
return;
}

```

Код для передатчика:

```

#include <stdint.h>
#include <SPI.h>
#include <Pixy.h>

Pixy pixy;

const uint8_t DIODE_PIN = 11;
const unsigned long UART_BAUD_RATE = 600;
const unsigned long UART_DELAY_US = 1000000 / UART_BAUD_RATE;
const unsigned long CARRIER_FREQUENCY = 38000;
const unsigned long CARRIER_DELAY_CYCLES = F_CPU / (CARRIER_FREQUENCY * 2);

byte incomingByte = 0;
byte byteMas[255];
int sizeotvet = 13;
byte otvetMas[255] = {
0x00,0x82,0x03,0x01,0x00,0x1A,0x00,0x2C,0x01,0x1F,0xFF,0x09,0xFF }; //0x00 будет
заменен на нужное смещение
byte imageMas[1800];
static void initPWM()
{
    TCCR1A = 0;
    TCCR1B = 0;
    TCCR1C = 0;
    TIMSK1 = 0;
}

```

```

static void startPWM()
{
    TCCR1B = 0;
    TCCR1A = 0;
    TCNT1 = 0;
    OCR1A = CARRIER_DELAY_CYCLES - 1;
    TCCR1A = (0 << COM1A1) | (1 << COM1A0) |
    (1 << WGM11) | (1 << WGM10);
    TCCR1B = (1 << WGM13) | (1 << WGM12) |
    (0 << CS12) | (0 << CS11) | (1 << CS10);
}

static void stopPWM()
{
    TCCR1B = 0;
    TCCR1A = 0;
}

static void sendZero()
{
    startPWM();
    delayMicroseconds(UART_DELAY_US);
}

static void sendOne()
{
    stopPWM();
    delayMicroseconds(UART_DELAY_US);
}

void setup() {
    Serial.begin(115200);
    Serial2.begin(115200); // в шилде системная шина подключена на Serial2
    pinMode(13, OUTPUT); // определяет режим прием/передача
    digitalWrite(13, LOW); // режим прием
    pinMode(DIODE_PIN, OUTPUT);
    digitalWrite(DIODE_PIN, LOW);
    initPWM();
    pinMode(6, OUTPUT);
    pixy.init();
}

void sendByte(uint8_t byte)
{
    sendZero();

    for (int i = 0; i < 8; i++) {
        // Serial.println(byte & 1);
        if ((byte & 1) != 0) {
            sendOne();
        }
        else {
            sendZero();
        }
    }
}

```

```

    }

    byte >>= 1;
}

sendOne();
}
int recivePacket() //функция получения пакета из сериал порта
{
    int pos = 0;
    while (true)
    {
        if (Serial2.available()) {
            incomingByte = Serial2.read();
            byteMas[pos] = incomingByte;
            pos++;
            if (incomingByte == 0x00) return pos;
        }
    }
}
void decodePacket(byte *myMas, int mas_size) //декодируем размер указывается
последний символ
{
    int num = 0;
    num = myMas[0]; //присваиваем num первое смещение
    for (int i = 0; i < mas_size; i++)
    {
        if (i == num) //если дошли до смещения
        {
            num += myMas[i]; //прибавить смещение к нашему
            myMas[i] = 0; //что-то тут непонятное с последним байтом
        }
    }
}
int encodePacket(byte *myMas, int mas_size) //шифровка пакетов
{
    int i = 0;
    int old_i = 0;
    for (i = mas_size - 1, old_i = mas_size; i >= 0; i--)
    {
        if (myMas[i] == 0x00)
        {
            myMas[i] = old_i - i;
            old_i = i;
        }
    }
    myMas[mas_size] = 0x00;
    return i;
}
bool packetForMe(byte *myMas, int mas_size)
{
    //if (myMas[1] == 0x02 && myMas[2] == 0x03 && myMas[3] == 0x1A &&
    myMas[4] == 0x00) return 1;
}

```

```

    if (myMas[3] == 0x22 && myMas[4] == 0x02) return 1;
        return 0;
    }
void PrintData(byte *myMas, int mas_size)
{
    for (int j = 1; j < mas_size; j++)
    {
        Serial.write(byteMas[j]);
    }
}
void SendData(byte *myMas, int mas_size)
{
    digitalWrite(13, HIGH);
    int n = 0;
    n = encodePacket(myMas, mas_size);
    Serial2.write(myMas, mas_size+1);
    Serial.write(myMas, mas_size+1);
    Serial.write(0xFF);
    digitalWrite(13, LOW);
}
int dataReader(byte *myMas, int mas_size, const int start_data){
    int j=0;
    for (int i=start_data; i<mas_size; i++)
    {
        imageMas[j++]=myMas[i];
    }
    return j;
}
void sendCup(byte *myMas,const int mas_size)
{
    for(int i=0;i<mas_size;i++)
    {
        sendByte(myMas[i]);
        Serial.print(myMas[i]);
    }
}
void sendAngle()
{
    unsigned int x1, x2, x3, y1, y2, y3, ab, bc,ca, mini, maxi, midl;
    int xx, xx1, yy, yy1;
    static int i = 0;
    int j;
    uint16_t blocks;
    char buf[32];

    // grab blocks!
    blocks = pixy.getBlocks();

    // If there are detect blocks, print them!
    if (blocks)
    {
        i++;
    }
}

```



```

if (1)
{

    x1 = pixy.blocks [0] .x;
    y1 = pixy.blocks [0] .y;
    x2 = pixy.blocks [1] .x;
    y2 = pixy.blocks [1] .y;
    x3 = pixy.blocks [2] .x;
    y3 = pixy.blocks [2] .y;

    ab=sqrt(((x1-x2)*(x1-x2)) + ((y1-y2)*(y1-y2)));
    bc=sqrt(((x2-x3)*(x2-x3)) + ((y2-y3)*(y2-y3)));
    ca=sqrt(((x1-x3)*(x1-x3)) + ((y1-y3)*(x1-y3)));

    if (ab > ca && ab > bc){maxi = ab;xx= x1; xx1=x2; yy=y1; yy1=y2;}
    if (bc > ca && bc > ab){maxi = bc;xx= x2; xx1=x3; yy=y2; yy1=y3;}
    if (ca > ab && ca > bc){maxi = ca;xx= x1; xx1=x3; yy=y1; yy1=y3;}

    if (ab < ca && ab < bc){maxi = ab;}
    if (bc < ca && bc < ab){maxi = bc;}
    if (ca < ab && ca < bc){maxi = ca;}

    if ((mini != ab) && (maxi != ab)){midl = ab; }
    if ((mini != bc) && (maxi != bc)){midl = bc; }
    if ((mini != ca) && (maxi != ca)){midl = ca; }

    int x =(xx-xx1);
    int y =(yy-yy1);
    double angl = (atan2(x,y))*180/3.14159625358;
/* Serial.println(x1);
Serial.println(x2);
Serial.println(x3);
Serial.println(y1);
Serial.println(y2);
Serial.println(y3);

Serial.println(ab);
Serial.println(bc);
Serial.println(ca);
Serial.println("ee");
//Serial.println(x);
//Serial.println(y);*/
Serial.println(abs(180 - angl));
int angle = (abs(180 - angl));
analogWrite(6, map(angle, 0, 360, 0, 255));

```

```

    }
}
}
void loop() {
  //sendAngle();
  while (Serial2.available() > 0) {
    int n = recivePacket();//получаем колличесетво байт в массиве, сам массив
изменяется глобально
    decodePacket(byteMas, n);//декодируем
    PrintData(byteMas,n);
    int n2=dataReader(byteMas,n,9);
    if (packetForMe(byteMas, n)) sendCup(imageMas,n2); //SendData(otvetMas, 13);
    //delay(600);
//    delay(1);
  }
}
}

```

Код для приемника:

```

#include <stdint.h>

void setup()
{
  Serial.begin(115200);
  Serial1.begin(600);
}

void loop()
{
  if (Serial1.available() > 0) {
    // uint8_t received_byte = Serial1.read();
    Serial.write(Serial1.read());
  }
}

```

4. Код, написанный командой, для работы со звездным датчиком:

```

#include <SPI.h>
#include <Pixy.h>

// This is the main Pixy object
Pixy pixy;

void setup()
{
  Serial.begin(9600);
  Serial.print("Starting...\n");

  pixy.init();
}

```

```

}

void loop()
{
  unsigned int x1, x2, x3, y1, y2, y3, ab, bc,ca, mini, maxi, midl;
  int xx, xx1, yy, yy1;
  static int i = 0;
  int j;
  uint16_t blocks;
  char buf[32];

  // grab blocks!
  blocks = pixy.getBlocks();

  // If there are detect blocks, print them!
  if (blocks)
  {
    i++;

    if (i%50==0)
    {

      x1 = pixy.blocks [0] .x;
      y1 = pixy.blocks [0] .y;
      x2 = pixy.blocks [1] .x;
      y2 = pixy.blocks [1] .y;
      x3 = pixy.blocks [2] .x;
      y3 = pixy.blocks [2] .y;

      ab=sqrt(((x1-x2)*(x1-x2)) + ((y1-y2)*(y1-y2)));
      bc=sqrt(((x2-x3)*(x2-x3)) + ((y2-y3)*(y2-y3)));
      ca=sqrt(((x1-x3)*(x1-x3)) + ((y1-y3)*(x1-y3)));

      if (ab > ca && ab > bc){maxi = ab;xx= x1; xx1=x2; yy=y1; yy1=y2;}
      if (bc > ca && bc > ab){maxi = bc;xx= x2; xx1=x3; yy=y2; yy1=y3;}
      if (ca > ab && ca > bc){maxi = ca;xx= x1; xx1=x3; yy=y1; yy1=y3;}

      if (ab < ca && ab < bc){maxi = ab;}
      if (bc < ca && bc < ab){maxi = bc;}
      if (ca < ab && ca < bc){maxi = ca;}

      if ((mini != ab) && (maxi != ab)){midl = ab; }
      if ((mini != bc) && (maxi != bc)){midl = bc; }
      if ((mini != ca) && (maxi != ca)){midl = ca; }

      int x =(xx-xx1);

```

```
    int y=(yy-yy1);
    double angl = (atan2(x,y))*180/3.14159625358;
/* Serial.println(x1);
Serial.println(x2);
Serial.println(x3);
Serial.println(y1);
Serial.println(y2);
Serial.println(y3);

Serial.println(ab);
Serial.println(bc);
Serial.println(ca);
Serial.println("ee");
//Serial.println(x);
//Serial.println(y);*/
Serial.println(abs(180 - angl));

}

}

}
```