Работа призера заключительного этапа

командной инженерной олимпиады школьников

Олимпиада Национальной технологической инициативы

Профиль «Системы связи и дистанционного зондирования Земли»

Ярошевич Александр Владимирович

Класс: 11

Город: г. Москва

Школа: ГБОУ школа №1375

Регион: Москва

Уникальный номер участника: 944

Команда на заключительном этапе: ВИАС

Параллель: 10-11

Результаты заключительного этапа:

ИНДИВИДУАЛЬНАЯ ЧАСТЬ																	КОМАНДНАЯ ЧАСТЬ									Итог
МАТЕМАТИКА				ФИЗИКА				ИНФОРМАТИКА							Сумма	Макс. балл	0	1	2	3	4	5	6	Сумма	Макс. балл	
1	2	3	4	1	2	3	4	1/1	1/2	1/3	2/1	2/2	3/1	3/2												
16	15	0	0	6	6	0	0	0	0	0	0	0	0	0	43	300	2	-	9	8	28	20	-	67	131	32,74

Индивидуальная часть

Персональный лист участника с номером 944:

Математика

Олимпиада НТИ

ФИО Ярахмат Александр Владимирович

Город Москва

Школа № 1375

Командная инженерная олимпиада «Олимпиада НТИ»

Направление Системы связи и ВЭЗ (космонавтика)

Предмет Математика.

Номер участника 944

1	2а	2б	3а	3б	3в	Σ
16	0	15	0	0	0	31

1.

Давайте рассмотрим концентрацию кофе в самом полном стакане после каждого переелива.

0) 1 (стакан Саши)

1) $\frac{1+0}{2} = \frac{1}{2}$ (стакан Руслана)

2) $\frac{\frac{1}{2}+1}{4} = \frac{3}{4}$ (стакан Саши)

3) $\frac{\frac{3}{4}+\frac{1}{2}}{2} = \frac{5}{8}$ (стакан Руслана)

4) $\frac{\frac{5}{8}+\frac{3}{4}}{2} = \frac{11}{16}$

5) $\frac{\frac{11}{16}+\frac{5}{8}}{2} = \frac{21}{32} \approx 0{,}6562$

6) $\frac{21+22}{64} = \frac{43}{64} \approx 0{,}671\ldots$ (стакан Саши)

7) $\frac{43+42}{128} = \frac{85}{128} \approx 0{,}66\ldots$ (стакан Руслана)

8) $\frac{\frac{85}{128}+\frac{43}{64}}{2} = \frac{141}{256} \approx 0{,}66\ldots$ (стакан Саши)

если 7 переливов.
то у Саши 100 мл
а у Руслана 200 мл
по этому ответ 8,
тогда у Руслана 100 мл
а у Саши 200 мл.

ответ: 8 +16

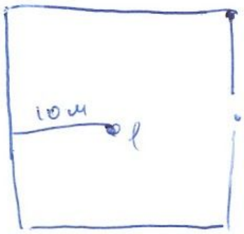
Командная инженерная олимпиада «Олимпиада НТИ»

Направление Системы связи и ДЗЗ (космонавтика)

Предмет Математика

Номер участника 944

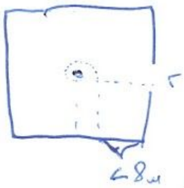
2.



а) • если робот пойдет по прямой до центра то он никак не сможет догнать.

т.к. $r = 10$ м а скорость робота $V_р = 8$ м/мин

• если робот пойдет сначала по прямой а потом по спирали до центра, то он опять не успеет.



т.к наибольший периметр спирали будет примерно равен $2\pi \cdot 2 \approx 12,6$ м, а его скорость всего лишь ~7,9 м/мин.

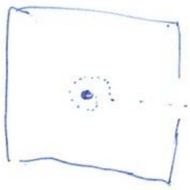
— (А кто сказал, что он будет двигаться именно так?)

б) Сможет.

к примеру если он пройдет 9,9м, со скоростью 9,99 м/мин, по прямой, а затем начнет двигаться по спирали.

Наибольший радиус спирали будет равен $R = 0,1$ м и его периметр. $P = 0,2 \cdot \pi$.

(Почему в этом радиусе его не догоняет лазер?)



Робот может пройти этот путь меньше чем за минуту т.к $9,99 > 0,2\pi$

???

(34)

Командная инженерная олимпиада «Олимпиада НТИ»

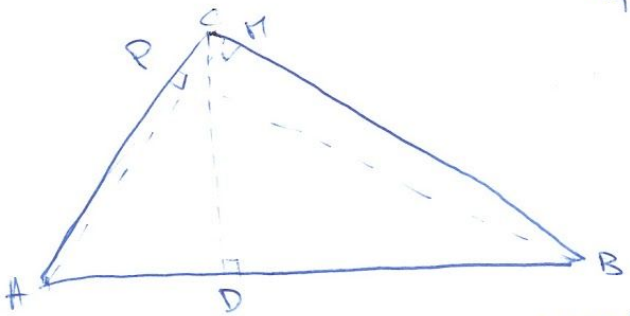
Направление Системы связи и РЭС (космонавтика)

Предмет Математика

Номер участника 944

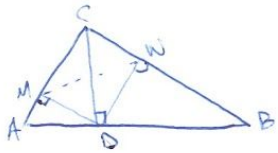
3. в)

PHD - высоты треугольника ABC



Рассмотрим 3 ~~отдельных треугольника~~ случая.

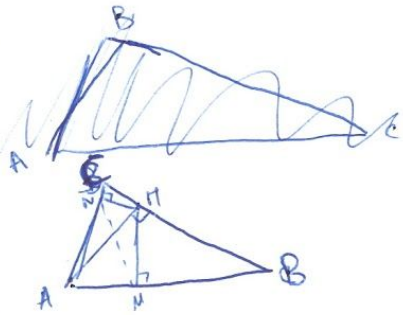
1)



$\triangle DNB \sim \triangle CDB \sim \triangle CND$ по 2м углам и стороне

$\triangle CDA \sim \triangle AMD \sim \triangle DMC$ по 2 углам и стороне.

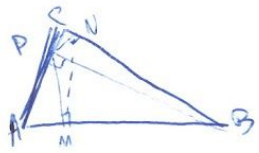
2)



$\triangle ABB \sim \triangle HBN \sim \triangle AHN$

$\triangle AHB \sim \triangle HMB \sim \triangle AMH$

3)



$\triangle APB \sim \triangle PMB \sim \triangle AMP$

$\triangle PCB \sim \triangle NPB \sim \triangle NCP$

~~0~~ (И хорошо треугольники подобны и что?)

Физика

Олимпиада НТИ

ФИО Ярошевич Александр Владимирович

Город Москва

Школа № 1375

Командная инженерная олимпиада «Олимпиада НТИ»

Направление Системы связи и ОЗЗ (космонавтика)

Предмет Физика

Номер участника 944

1	2	3	4	Σ
6	6	0	0	12

1. Дано:

$l = 20000$ м

$v_л = 1{,}77$ км/с

$H = 35786$ км

$R_з = 6371$ км

$M_з = 5{,}97 \cdot 10^{24}$

$G = 6{,}674 \cdot 10^{-11}$

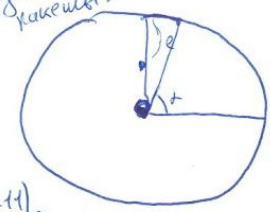
$\frac{GMm}{(R+H)^2} = ma$ $\quad \frac{GM}{(R+H)^2} = \frac{v_c^2}{(R+H)^2}$

$v_c = \sqrt{\frac{GM}{R+H}}$ +4

$v_c = 3074{,}29$ м/с $\quad t = \frac{l}{v_л}$

S - путь спутника

X - расстояние от спутника до ракеты.



• Ракета пройдет расстояние l за $\frac{l}{v_л} = 11{,}3$ с

• За t спутник пройдет расстояние $v_c \cdot t = 34739{,}5$ м

$\arcsin(1 - 2{,}289 \cdot 10^{-11}) =$

$= 89{,}999596$

$l_{орб} = 2\pi R_{орб} = 5{,}58328 \cdot 10^{15}$

, спутник пройдет $6{,}222 \cdot 10^{-12}$ орбиты.

отв: 34739,5 м если считать по расстоянию по орбите.

-2 это не инженерное решение.

I_2 (187)

I_1 $2{,}18 I_1 = 1249 I$

6

Командная инженерная олимпиада «Олимпиада НТИ»

Направление КОСМОС

Предмет ФИЗИКА

Номер участника 944

2. $R = 5$ м

$L = 11{,}78$

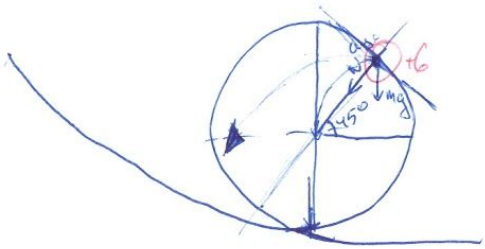
$P_{круга} = 2\pi R = 31{,}42$

$\frac{11{,}78}{31{,}42} = 0{,}375.$

Тело прошло 0,375 $P_{круга}$

⇓

135°.



$\alpha = 45°$ (б)

ЗС:

$mgh = \frac{mv^2}{2} + gh_2$

$gh = \frac{v^2}{2} + gh_2$

$mg\sin\alpha = m a_{цс}$

$10\frac{\sqrt{2}}{2} = \frac{v^2}{R}$

$25\sqrt{2} = v^2$

$l_{пад} = \frac{v_0^2 \sin 2\alpha}{\frac{\sqrt{2}}{2} g} = \frac{25\sqrt{2} \cdot 1.2}{\sqrt{2} \cdot 10} =$

$= 2{,}5$

⇓

тело приземлилось в нижней точке пещеры.

(−)

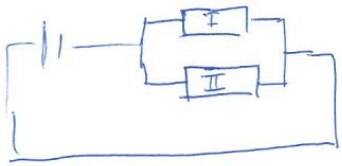
Командная инженерная олимпиада «Олимпиада НТИ»

Направление системы связи и ВЗЗ (космонавтика)

Предмет Физика

Номер участника 944

3.



$R_1 = R_2 = 10$ Ом

1) $I_1 = I_2$ расчет $I_1 + I_2 = I$.

2) $I_1' = 1{,}1 I_1$

$I_2' = I_2$

$I' = 1{,}145 I$

$I_1' + I_2' = I'$

$1{,}1 \cdot I_1 + I_1 = 1{,}145 I$

$I = 1{,}857 I_1$

Ответ: 1,1 Ом

Информатика

Задача 1.2

Предложенный код программы на языке С:

```
1    #include <stdio.h>
2
3    int main() {
4      int n,m,k,i,z,x,y,s,f;
5      scanf("%d",&n);
6      scanf("%d",&m);
7      scanf("%d\n",&k);
8      y=0;
9      f=26;
10        for(i=1;i<=n;i++){
11             scanf("%d",&s);
12             if(s==m)
13             z=i;
14        }
15             if(z>(n/2)){
16             if(z<f){
17             x=n-z;
18                  f=z;}
19             if(z=n)
20                  x=0;
21             if(z>n){
22                  z=n;
23                  x=n-z;
24             }
25        }
26             else
27             x=z-1;
28     y=x+k+1;
29     printf("%d",y);
30      return 0;
31    }
```

Данный код приводил к ошибке в тесте №5 (неправильный ответ) (0 баллов).

Командная часть

Результаты были получены в рамках выступления команды: ВИАС

Личный состав команды:

- Ярошевич Александр, 11 класс, Москва, школа 1375
- Гирин Иван, 11 класс, Москва, школа 1375
- Буссе Александр, 11 класс, Звенигово, ЗСОШ №3
- Сизов Иван, 10 класс, Вологда, Лицей №32

Команда «ВИАС» с финальной версией собранного устройства:

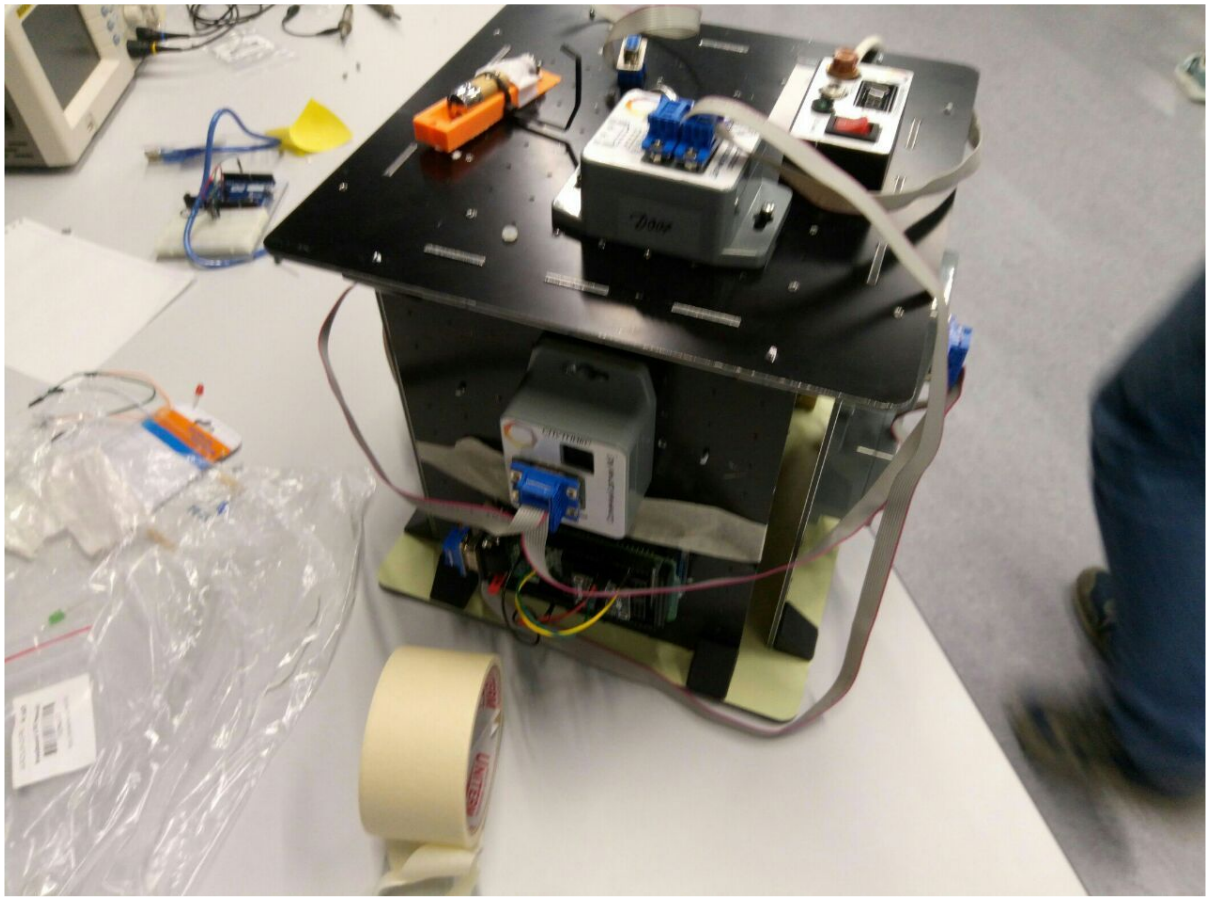
Ярошевич Александр за сборкой устройства.

Протокол выполнения заданий

Задание	**Итог**
Задание 0. Подготовка к работе	Выполнено (2 балла)
Задание 1. Ориентация и стабилизация	Не выполнено (0 баллов) Максимальное достигнутое время удержания луча на мишени составило 8,5 секунд.
Задание 2. Оптический канал связи	Выполнена (9 баллов). Предложенный вариант решения не передавал символы 0x00. В итоге передано 6 из 9 сообщений.
Задание 3. Неизвестная планета	Выполнено (8 баллов)
Задание 4. Новый модуль спутника	Выполнено (28 баллов)
Задание 5. Канал телеметрии	Выполнено (20 баллов)
Задание 6. Миссия	Не выполнено (0 баллов)

Задание 0. Подготовка к работе

Собранный аппарат выглядел следующим образом:



З а д а н и е 1. Ориентация и стабилизация

Для выполнения задания был предложен следующий код:

```
#include "libschsat.h"
const int num = 1;
int16_t x,y,z,i,k,b,mSpeed,maxside;
uint16_t sun1[5],sun2[5];
//Стабилизация
//Можно поменять 7000 на 8000 чтобы быстрее стабилизироваться.
void stabil(void){
        int16_t SetSpeed;
   hyro_request_raw(num, &x, &y, &z);
        motor_request_speed(num, &mSpeed);
        while(abs(z)>250){
          if (abs(mSpeed)>8000){
                motor_set_speed(num, 0, &k);
          }
          SetSpeed=mSpeed-z;
          if (abs(z)>250){
                if (abs(SetSpeed)>8000){
                  motor_set_speed(num, 0, &k);
                }else{
                        motor_set_speed(num, SetSpeed, &k);
                }
          }
          printf("%d   %d\n",z,mSpeed);
          mSleep(200);
          motor_request_speed(num, &mSpeed);
          hyro_request_raw(num, &x, &y, &z);
        }
}
//Функция для поворота к солнцу
void sunturn(void){
        int16_t max,min,minside;
        max=0;
        min=30000;
        for (i=1;i<=4;i++){
                sun_sensor_request_raw(i, &sun1[i], &sun2[i]);
                printf("n%d  1: %d  2:%d\n",i,sun1[i],sun2[i]);
                /*if (i==2){
                        sun1[i]=sun1[i]*0.9;
                        sun2[i]=sun2[i]*0.9;
                }*/
                if (sun1[i]+sun2[i] > max){
                maxside=i;
                        max=sun1[i]+sun2[i];
                }
                if (sun1[i]+sun2[i] < min){
                  minside=i;
```

```
                min=sun1[i]+sun2[i];
            }
        }
        printf("maxside %d\n",maxside);
        motor_request_speed(num, &mSpeed);
        if (maxside=1){
            if ((sun1[4]+sun2[4]-sun1[2]-sun2[2])>15){
                motor_set_speed(num, mSpeed+100,&k);
           }else if((sun1[4]+sun2[4]-sun1[2]-sun2[2])<-15){
                motor_set_speed(num, mSpeed-100,&k);
            }
        }else{
            hyro_request_raw(num, &x, &y, &z);
            if (maxside=2){
                motor_set_speed(num, mSpeed-300,&k);
            }
            if (maxside=4){
                motor_set_speed(num, mSpeed+300,&k);
            }
            if (maxside=3){
                if ((sun1[4]+sun2[4])>(sun1[2]-sun1[2])){
                   motor_set_speed(num, mSpeed+600,&k);
                }else{
                   motor_set_speed(num, mSpeed-600,&k);
                }
            }
        }
}
void control(void){
        hyro_turn_on(num);
        mSleep(500);
    motor_turn_on(num);
        mSleep(500);
        for (i=1;i<=4;i++){
            sun_sensor_turn_on(i);
            mSleep(100);
        }
        if (LSS_OK == sun_sensor_request_raw(num, &sun1[1], &sun2[1])) {
            printf(" raw=%d\n", sun1[1]);
        } else {
            puts("Fail!");
        }
        if (LSS_OK == hyro_request_raw(num, &x, &y, &z)) {
            printf("%d: x=%d y=%d z=%d\n", num, x, y, z);
        } else {
            puts("Fail!");
        }
        if (LSS_OK == motor_request_speed(num, &mSpeed)) {
```

```
                printf("Got speed %d >>>\n", mSpeed);
        } else {
                puts("Fail! >>>");
        }
        motor_set_speed(num,-7000,&k);
        Sleep(15);
        motor_set_speed(num,0,&k);
        stabil();
        b=1;
        while(b>0){
          sunturn();
                if (maxside==1){
                if (abs((sun1[1]-sun2[1]))<2000){
                  mSleep(200);
                }else mSleep(500);
                }else
          mSleep(600);//Если mSleep поставить больше, то спутник будет поворачиваться к
солнцу быстрее, но не очень точно.
          stabil();
        }
}
```

Задание 2. Оптический канал связи

Код для передатчика:

```
#define tranPin 11

uint8_t buf[254], cnt = 0;

void UART_sleep(int mill)
{
 TCCR1A &= ~(1 << COM1A0);
 delay(mill);
 TCCR1A |= (1 << COM1A0);
}

int COBS_encode(uint8_t * buf, size_t sz = 254, bool ir=false)
{
 if (sz > 254)
  return 1;
 size_t last_null = 0;
 //Serial1.write(sz);
 //Serial.write(0xCC);
 //Serial.write(buf, sz);
 //Serial.write(0xDD);
 if (ir)
```

```
   UART_sleep(120);
  for (size_t i = 0; i <= sz; ++i)
  {
   if (i == sz || !buf[i])
   {
    size_t cnt = i - last_null + 1;
    if (ir)
    {Serial.write(cnt);
    Serial1.write(cnt);UART_sleep(120);}
    if (last_null != i)
    {
     //Serial.write((buf + last_null), cnt);
     for (size_t j = last_null; j < i; ++j)
      if(ir){Serial.write(buf[j]);
      Serial1.write(buf[j]);UART_sleep(120);}
    }
    last_null = i + 1;
   }
   //for (size_t j = i
  }
  if (!ir)
  {
  Serial.write(0);
  Serial1.write(0);
  }
  return 0;
};

uint8_t COBS_decode(uint8_t * buf)
{
  uint8_t i = 0xFF, next_null = 0;
  //while (Serial.available())
  while (true)
  {
   if (Serial.available())
   {
    uint8_t in_byte = Serial.read();
    //Serial.write(in_byte);
    if (!in_byte || i == 0xFE)
     break;
    if (i == 0xFF)
     next_null = in_byte - 1;
    else if (i == next_null)
    {
     next_null = i + in_byte;
     buf[i] = 0;
    }
    else
```

```
      buf[i] = in_byte;
     ++i;
    }
  }
  //Serial.write(0xAA);
  return (i < 0xFE) ? i : 0xFF;
}

void PWM_init()
{
  pinMode(tranPin, OUTPUT);

  // initialize timer1
  noInterrupts();           // disable all interrupts
  TCCR1A = 0;
  TCCR1B = 0;
  TCNT1  = 0;

  OCR1A = 141;            // compare match register 16MHz/1/140KHz
  TCCR1A |= (1 << COM1A0);
  TCCR1B |= (1 << WGM12);   // CTC mode
  TCCR1B |= (1 << CS10);    // 1 prescaler
  //TIMSK1 |= (1 << OCIE1A);  // enable timer compare interrupt
  interrupts();             // enable all interrupts
}

/*ISR(TIMER1_COMPA_vect)          // timer compare interrupt service routine
{
  digitalWrite(tranPin, digitalRead(tranPin) ^ 1);   // toggle LED pin
}*/

ISR(PCINT0_vect)
{
   if (TCCR1A & (1 << COM1A0))
    TCCR1A &= ~(1 << COM1A0);
   else
    TCCR1A |= 1 << COM1A0;
   /*if (!(TCCR1A & (1 << COM1A0)))
    TCCR1A |= 1 << COM1A0;
    delay(2);
    TCCR1A &= ~(1 << COM1A0);*/
}

/*void tran_init()
{
  // initialize timer3
  noInterrupts();           // disable all interrupts
  TCCRA = 0;
```

```
  TCCR3B = 0;
  TCNT3  = 0;

  OCR2A = 833;            // compare match register 16MHz/64/300Hz
  TCCR3B |= (1 << WGM12);   // CTC mode
  TCCR3B |= (1 << CS10) | (1 << CS11);    // 64 prescaler
  TIMSK1 |= (1 << OCIE1A);  // enable timer compare interrupt
  interrupts();             // enable all interrupts
}

ISR(TIMER1_COMPA_vect)          // timer compare interrupt service routine
{
  digitalWrite(tranPin, digitalRead(tranPin) ^ 1);   // toggle LED pin
}*/

int Hamming_encode(uint16_t * ibuf, uint32_t * obuf, size_t sz=16)
{
    if (sz > 254)
      return 1;
}

void setup() {
  // put your setup code here, to run once:
  PCICR |= (1 << PCIE0);
  PCMSK0 |= (1 << PCINT4);

  Serial.begin(115200);
  Serial1.begin(100);
  //uint8_t buf[254];
  for (size_t i = 0; i < 254; ++i)
    buf[i] = i + 0xAA; //i + 1;// i % 3;
  //Serial.write(buf, 254);
  //Serial.write(0xFF);
  //COBS_encode(buf);
  PWM_init();
  Serial.write(buf, 7);
}


void loop() {
  // put your main code here, to run repeatedly:
  /*if (Serial1.available())
  {
    in_byte = Serial1.read();
    Serial.println(in_byte, HEX);
  }*/

  //if (Serial.available())
```

```
//{
   //uint8_t in_byte = Serial.read();
   //if (in_byte == 0xEE)
   //{
     //Serial.println(cnt);
     //Serial.write(buf,cnt);
     //Serial.println();
     //COBS_encode(buf, 1);
     //Serial.write(0xAA);
     //COBS_encode();
     //cnt = 0;
   //}
   //else
    //buf[cnt++] = in_byte;
//}
/*if (Serial.available())
{
  uint8_t cnt = COBS_decode(buf);
  Serial.write(cnt);
  Serial.write(0xBB);
  Serial.write(buf, cnt);
}*/

//if (Serial.available())
//{
  //cnt = 0;
  //while (Serial.available())
  //{
    //uint8_t in_byte = Serial.read();
    //buf[cnt++] = in_byte;
  //}
  //Serial.write(buf[0]);
  //Serial1.write(buf[0]);
  /*UART_sleep(120);
  for (size_t i = 0; i < cnt; ++i)
  {
    Serial.write(buf[i]);
    Serial1.write(buf[i]);
    UART_sleep(120);
  }*/
  COBS_encode(buf,32,true);
//}
//Serial1.write(0x01);
//Serial1.write(buf[cnt++]);
//for (size_t i = 0; i < 5; ++i);
  //Serial1.write(0x00);
//COBS_encode(buf, 7);
//delay(2);
```

```
  //UART_sleep(120);
  //if (cnt == 10)
    //cnt = 0;
}
```

Код для приемника:

```
#define rxPin 7
uint8_t buf[254], cnt = 0;
uint8_t COBS_decode(uint8_t * buf)
{
  uint8_t i = 0xFF, next_null = 0;
  //while (Serial.available())
  while (true)
  {
    if (Serial1.available())
    {
      uint8_t in_byte = Serial1.read();
      //Serial.write(in_byte);
      if (!in_byte || i == 0xFE)
        break;
      if (i == 0xFF)
        next_null = in_byte - 1;
      else if (i == next_null)
      {
        next_null = i + in_byte;
        buf[i] = 0;
      }
      else
        buf[i] = in_byte;
      ++i;
    }
  }
  //Serial.write(0xAA);
  return (i < 0xFE) ? i : 0xFF;
}

void setup() {
  // put your setup code here, to run once:
  //PCICR |= (1 << PCIE0);
  //PCMSK0 |= (1 << PCINT4);

  Serial.begin(115200);
  Serial1.begin(100);
  //uint8_t buf[254];
  //for (size_t i = 0; i < 254; ++i)
  //  buf[i] = 0xAA; //i + 1;// i % 3;
  //Serial.write(buf, 254);
```

```
  //Serial.write(0xFF);
  //COBS_encode(buf);
  //PWM_init();
  //Serial.write(buf, 7);
}

void loop() {
  // put your main code here, to run repeatedly:
  if (Serial1.available())
  {
    //uint8_t in_byte = Serial1.read();
    //Serial.write(in_byte);
    cnt = COBS_decode(buf);
    Serial.write(buf[0]);
    Serial.write(buf + 2,cnt);
  }
}
```

Задание 3. Неизвестная планета

Для выполнения задания был предложен следующий код:

```
#include "libschsat.h"
#include "time.h"

void control()
{
	transceiver_turn_on(2);
	sleep(1);
	char a[]="                                                                                ";
	int start,tm,data;
	time(&start);
	bus_setup();
	//transceiver_request_reset(2);
	//[10];for(int i=0;i<10;i++){a[i]='f';}
	//transceiver_send(2,1,&a,1);
	while(1){

	//const char hello[] = "hello, world!";
	/*if (LSS_OK != transceiver_send(2, 1, (uint8_t *) hello, sizeof(hello)))
		puts("Fail!");*/
	//transceiver_request_buff(2,a);
	//transceiver_request_buff(2,&data);
	/*for (int i=0;i<10;i++)
	{*/
		transceiver_request_buff(2,&a);
		printf("%s",a);
	//}


	/*for(int i=0;i<10;i++)
	{
		printf("%s",a[i]);
	}*/
	sleep(1);

	}

}
```

Задание 4. Новый модуль спутника

Для выполнения задания был предложен следующий код:

```
#define tranPin 11

uint8_t buf[254], cnt = 0;

bool transmit = 0, strtflag = 0;

char to_addr_str[100], from_addr_str[100], msg_type_str[100];

/*int COBS_encode(uint8_t * buf, size_t sz = 254)
{
 if (sz > 254)
  return 1;
 size_t last_null = 0;
 //Serial1.write(sz);
 Serial.write(0xCC);
 Serial.write(buf, sz);
 Serial.write(0xDD);
 for (size_t i = 0; i <= sz; ++i)
 {
  if (i == sz || !buf[i])
  {
   size_t cnt = i - last_null + 1;
   Serial.write(cnt);
   Serial1.write(cnt);
   if (last_null != i)
   {
    //Serial.write((buf + last_null), cnt);
    for (size_t j = last_null; j < i; ++j)
     {Serial.write(buf[j]);
     Serial1.write(buf[j]);}
   }
   last_null = i + 1;
  }
  //for (size_t j = i
 }
 Serial.write(0);
 Serial1.write(0);

 return 0;
};*/

uint8_t COBS_decode(uint8_t * buf)
{
 uint8_t i = 0xFF, next_null = 0;
 //while (Serial.available())
```

```
  while (true)
  {
   if (Serial1.available())
   {
    uint8_t in_byte = Serial1.read();
    //Serial.write(in_byte);
    if (!in_byte || i == 0xFE)
     break;
    if (i == 0xFF)
     next_null = in_byte - 1;
    else if (i == next_null)
    {
     next_null = i + in_byte;
     buf[i] = 0;
    }
    else
     buf[i] = in_byte;
    ++i;
   }
  }
  //Serial.write(0xAA);
  return (i < 0xFE) ? i : 0xFF;
}

/*void PWM_init()
{
  pinMode(tranPin, OUTPUT);

  // initialize timer1
  noInterrupts();           // disable all interrupts
  TCCR1A = 0;
  TCCR1B = 0;
  TCNT1  = 0;

  OCR1A = 141;            // compare match register 16MHz/1/140KHz
  TCCR1A |= (1 << COM1A0);
  TCCR1B |= (1 << WGM12);   // CTC mode
  TCCR1B |= (1 << CS10);    // 1 prescaler
  //TIMSK1 |= (1 << OCIE1A);  // enable timer compare interrupt
  interrupts();             // enable all interrupts
}*/

/*ISR(TIMER1_COMPA_vect)          // timer compare interrupt service routine
{
  digitalWrite(tranPin, digitalRead(tranPin) ^ 1);   // toggle LED pin
}*/

ISR(PCINT0_vect)
```

```
{
  transmit = (transmit) ? 0 : 1;
}

int msg_type_decode(char * res_str, uint16_t msg_type)
{
 char *msg_type_str;
 //Serial.println("checking ");
 //Serial.println(msg_type, HEX);
 switch (msg_type)
 {
  case (0x0000): msg_type_str="RPI_OBC_BASE"; break;
  case (0x0200): msg_type_str="SENS_LIGHT_REQ_RAW"; break;
  case (0x0208): msg_type_str="SENS_LIGHT_CMD_RESET"; break;
  case (0x0210): msg_type_str="SENS_LIGHT_REQ_MAXRAW"; break;
  case (0x0220): msg_type_str="SENS_LIGHT_SET_MINVALUE"; break;
  case (0x0228): msg_type_str="SENS_LIGHT_SET_CALIBRATE"; break;
  case (0x0280): msg_type_str="SENS_LIGHT_RESP_RAW"; break;
  case (0x0288): msg_type_str="SENS_LIGHT_RESP_MAXRAW"; break;
  case (0x0300): msg_type_str="SENS_MAG_REQ_RAW"; break;
  case (0x0304): msg_type_str="SENS_MAG_CMD_RESET"; break;
  case (0x0380): msg_type_str="SENS_MAG_RESP_RAW"; break;
  case (0x0400): msg_type_str="SENS_HYRO_REQ_RAW"; break;
  case (0x0404): msg_type_str="SENS_HYRO_CMD_RESET"; break;
  case (0x0480): msg_type_str="SENS_HYRO_RESP_RAW"; break;
  case (0x0500): msg_type_str="SENS_SUN_REQ_RAW"; break;
  case (0x0508): msg_type_str="SENS_SUN_CMD_RESET"; break;
  case (0x0510): msg_type_str="SENS_SUN_REQ_MAXRAW"; break;
  case (0x0518): msg_type_str="SENS_SUN_SET_MINVALUE"; break;
  case (0x0580): msg_type_str="SENS_SUN_RESP_RAW"; break;
  case (0x0588): msg_type_str="SENS_SUN_RESP_MAXRAW"; break;
  case (0x0600): msg_type_str="SENS_ACCEL_REQ_RAW"; break;
  case (0x0604): msg_type_str="SENS_ACCEL_CMD_RESET"; break;
  case (0x0680): msg_type_str="SENS_ACCEL_RESP_RAW"; break;
  case (0x0D00): msg_type_str="CONTROL_WHEEL0_SET_SPEED"; break;
  case (0x0D04): msg_type_str="CONTROL_WHEEL0_CMD_RESET"; break;
  case (0x0D08): msg_type_str="CONTROL_WHEEL0_REQ_SPEED"; break;
  case (0x0D10): msg_type_str="CONTROL_WHEEL0_RESP_SPEED_CONFIRM"; break;
  case (0x0D0C): msg_type_str="CONTROL_WHEEL0_RESP_SPEED"; break;
  case (0x0E00): msg_type_str="CONTROL_FAN0_SET_SPEED"; break;
  case (0x0E04): msg_type_str="CONTROL_FAN0_CMD_RESET"; break;
  case (0x0E08): msg_type_str="CONTROL_FAN0_REQ_SPEED"; break;
  case (0x0E10): msg_type_str="CONTROL_FAN0_RESP_SPEED_CONFIRM"; break;
  case (0x0E0C): msg_type_str="CONTROL_FAN0_RESP_SPEED"; break;
  case (0x0F00): msg_type_str="CONTROL_COIL_SET_VAL"; break;
  case (0x0F10): msg_type_str="CONTROL_COIL_CMD_RESET"; break;
  case (0x0F20): msg_type_str="CONTROL_COIL_RESP_VAL_CONFIRM"; break;
  case (0x1000): msg_type_str="CONTROL_ENGINE_SET_VAL"; break;
```

```
    case (0x1010): msg_type_str="CONTROL_ENGINE_CMD_RESET"; break;
    case (0x1020): msg_type_str="CONTROL_ENGINE_RESP_VAL_CONFIRM"; break;
    case (0x2000): msg_type_str="MISC_LASER_CMD_ON"; break;
    case (0x2004): msg_type_str="MISC_LASER_CMD_OFF"; break;
    case (0x2008): msg_type_str="MISC_LASER_CMD_RESET"; break;
    case (0x200C): msg_type_str="MISC_LASER_RESP_ON_CONFIRM"; break;
    case (0x2010): msg_type_str="MISC_LASER_RESP_OFF_CONFIRM"; break;
    case (0x2100): msg_type_str="MISC_HF_REQ_STATE"; break;
    case (0x2110): msg_type_str="MISC_HF_CMD_RESET"; break;
    case (0x2120): msg_type_str="MISC_HF_RESP_STATE"; break;
    case (0x2200): msg_type_str="MISC_UHF_CMD_SEND"; break;
    case (0x2210): msg_type_str="MISC_UHF_CMD_RESET"; break;
    case (0x2220): msg_type_str="MISC_UHF_REQ_BUFFER"; break;
    case (0x2230): msg_type_str="MISC_UHF_RESP_BUFFER"; break;
    case (0x2300): msg_type_str="MISC_SUN_REQ_RAW"; break;
    case (0x2308): msg_type_str="MISC_SUN_CMD_RESET"; break;
    case (0x2310): msg_type_str="MISC_SUN_REQ_MAXRAW"; break;
    case (0x2380): msg_type_str="MISC_SUN_RESP_RAW"; break;
    case (0x2388): msg_type_str="MISC_SUN_RESP_MAXRAW"; break;
    case (0x2400): msg_type_str="MISC_EARTHFIELD_CMD_SET"; break;
    case (0x2408): msg_type_str="MISC_EARTHFIELD_CMD_AMPRESET"; break;
    case (0x2410): msg_type_str="MISC_EARTHFIELD_CMD_RESET"; break;
    case (0x2500): msg_type_str="MISC_GROUNDLEDS_CMD_SET"; break;
    case (0x2510): msg_type_str="MISC_GROUNDLEDS_CMD_RESET"; break;
    case (0x2600): msg_type_str="MISC_MOTOGLOBE_CMD_GO_SLOW"; break;
    case (0x2610): msg_type_str="MISC_MOTOGLOBE_CMD_GO_FAST"; break;
    case (0x2620): msg_type_str="MISC_MOTOGLOBE_CMD_START"; break;
    case (0x2630): msg_type_str="MISC_MOTOGLOBE_CMD_STOP"; break;
    case (0x2640): msg_type_str="MISC_MOTOGLOBE_CMD_ENABLEINT"; break;
    case (0x2650): msg_type_str="MISC_MOTOGLOBE_CMD_DISABLEINT"; break;
    case (0x2660): msg_type_str="MISC_MOTOGLOBE_CMD_RESET"; break;
    case (0x2700): msg_type_str="MISC_ARM_CMD_SET_STATE"; break;
    case (0x2710): msg_type_str="MISC_ARM_CMD_RESET"; break;
    case (0x2720): msg_type_str="MISC_ARM_RESP_STATE"; break;
    default: return 1; break;
  }
  strcpy(res_str, msg_type_str);
  return 0;
}

int addr_decode(char * res_str, uint16_t addr)
{
  char *addr_str;
  //Serial.println("checking ");
  //Serial.println(addr, HEX);
  switch (addr)
  {
    case (0x0001): addr_str="OBC_ADDRESS"; break;
```

```
    case (0x0010): addr_str="SENS_LIGHT_ADDRESS"; break;
    case (0x0018): addr_str="SENS_MAG_ADDRESS"; break;
    case (0x001C): addr_str="SENS_HYRO_ADDRESS"; break;
    case (0x0020): addr_str="SENS_SUN_ADDRESS"; break;
    case (0x0028): addr_str="SENS_ACCEL_ADDRESS"; break;
    case (0x0100): addr_str="CONTROL_WHEEL0_ADDRESS"; break;
    case (0x0110): addr_str="CONTROL_FAN0_ADDRESS"; break;
    case (0x0120): addr_str="CONTROL_COIL_ADDRESS"; break;
    case (0x0130): addr_str="CONTROL_ENGINE_ADDRESS"; break;
    case (0x0200): addr_str="MISC_LASER_ADDRESS"; break;
    case (0x0210): addr_str="MISC_HF_ADDRESS"; break;
    case (0x0220): addr_str="MISC_UHF_ADDRESS"; break;
    case (0x0230): addr_str="MISC_SUN_ADDRESS"; break;
    case (0x0240): addr_str="MISC_EARTHFIELD_ADDRESS"; break;
    case (0x0250): addr_str="MISC_GROUNDLEDS_ADDRESS"; break;
    case (0x0260): addr_str="MISC_MOTOGLOBE_ADDRESS"; break;
    case (0x0270): addr_str="MISC_ARM_ADDRESS"; break;
    case (0xFFFF): addr_str="MISC_ADDRESS_BROADCAST"; break;
    default: return 1; break;
  }
  strcpy(res_str, addr_str);
  return 0;
}

void setup() {
  // put your setup code here, to run once:
  PCICR |= (1 << PCIE0);
  PCMSK0 |= (1 << PCINT4);

  Serial.begin(115200);
  Serial1.begin(115200);
}


void loop() {
  // put your main code here, to run repeatedly:
  /*if (Serial1.available())
  {
    in_byte = Serial1.read();
    Serial.println(in_byte, HEX);
  }*/


  /*if (transmit)
  {
    if (Serial.available())
    {
      uint8_t out_byte = Serial.read();
```

```
    Serial1.write(out_byte);
    //COBS_encode(buf
  }
}
else
{
  if (Serial1.available())
  {
    uint8_t in_byte = Serial.read();
    Serial.write(in_byte);
  }
}*/
//if (Serial1.available())
if (Serial1.available())
{
  //Serial.println("ololo");

  //uncomment this!!!!
  uint8_t cnt = COBS_decode(buf);
  /*Serial.write(0xAA);
  Serial.write(cnt);
  Serial.write(0xBB);
  Serial.write(buf, cnt);
  Serial.write(0xCC);*/

  /*for (size_t i = 0; i < 6; ++i)
  {
    while (!Serial.available()) {};
    buf[i] = Serial.read();
  }*/


  uint16_t msg_type = ((uint16_t*)buf)[0],
  to_addr = ((uint16_t*)buf)[1],
  from_addr = ((uint16_t*)buf)[2];

  if (msg_type == 0x2308)
    strtflag = 1;
  if (!strtflag)
    return;

  /*Serial.write(msg_type);
  Serial.write(msg_type >> 8);
  Serial.write(0xDD);
  Serial.write(to_addr);
  Serial.write(to_addr >> 8);
  Serial.write(0xEE);
  Serial.write(from_addr);
```

```
    Serial.write(from_addr >> 8);
    Serial.write(0xFF);*/


    /*Serial.println(msg_type, HEX);
    //Serial.write(msg_type >> 8);
    //Serial.write(0xDD);
    Serial.println(to_addr, HEX);
    //Serial.write(to_addr >> 8);
    //Serial.write(0xEE);
    Serial.println(from_addr, HEX);
    //Serial.write(from_addr >> 8);
    //Serial.write(0xFF);*/

    if (to_addr == 0x0222)
    {
    Serial.println();
    uint16_t c_msg_type = msg_type, c_to_addr = to_addr, c_from_addr = from_addr;
    bool aflag = true;

    while (msg_type - c_msg_type < 5 && msg_type_decode(msg_type_str, c_msg_type) || (aflag =
false) == true)
    {
      --c_msg_type;
      //aflag = false;
    }
    if (!aflag)
    {
      Serial.print("type ");
      Serial.print(msg_type_str);
      Serial.print(' ');
      Serial.print(msg_type - c_msg_type);
      Serial.println();
    }
    else
      Serial.println("msg_type_fail");


    aflag = true;
    while (to_addr - c_to_addr < 5 && addr_decode(to_addr_str, c_to_addr) || (aflag = false) == true)
    {
      --c_to_addr;
      //aflag = false;
    }
    if (!aflag)
    {
      Serial.print("to ");
      Serial.print(to_addr_str);
```

```
    Serial.print(' ');
    Serial.print(to_addr - c_to_addr);
    Serial.println();
  }
  else
    Serial.println("to_addr_fail");

  aflag = true;
  while (from_addr - c_from_addr < 15 && addr_decode(from_addr_str, c_from_addr) || (aflag =
false) == true)
  {
    --c_from_addr;
    //aflag = false;
  }
  if (!aflag)
  {
    Serial.print("from ");
    Serial.print(from_addr_str);
    Serial.print(' ');
    Serial.print(from_addr - c_from_addr);
    Serial.println();
  }
  else
    Serial.println("from_addr_fail");



  //if (to_addr == 0x0222)
  //{
    //Serial.write(buf, cnt);
    for (size_t i = 6; i < cnt; ++i)
    {
      char nice_out[10];
      //sprintf(nice_out, "%x ", buf[i]);
      Serial.print(buf[i], HEX);
      Serial.print(' ');
      //Serial.write(nice_out, 2);
    }

    Serial.println();
  //}

  /*uint8_t in_byte = Serial1.read();

  if (!in_byte and cnt++)
  {
    Serial.write(in_byte);
    Serial.println();
```

```
  }
  else if (in_byte)
  {
   Serial.write(in_byte);
   cnt = 0;
  }*/
  }
 }

 /*if (Serial.available())
 {
  uint8_t cnt = COBS_decode(buf);
  Serial.write(cnt);
  Serial.write(0xBB);
  Serial.write(buf, cnt);
 }*/
 //Serial.write(buf, 7);
 //COBS_encode(buf, 7);
 //delay(100);
}
```

Задание 5. Канал телеметрии

```
#include "libschsat.h"
#include "time.h"

void control()
{
        transceiver_turn_on(2);
        sleep(1);
        char a[]="                                                                                ";
        int start,tm,data;
        time(&start);
        bus_setup();
        //transceiver_request_reset(2);
        //[10];for(int i=0;i<10;i++){a[i]='f';}
        //transceiver_send(2,1,&a,1);
        while(1){
        time(&tm);
        //tm=tm-start;
        //const char hello[] = "hello, world!";
        /*if (LSS_OK != transceiver_send(2, 1, (uint8_t *) hello, sizeof(hello)))
                puts("Fail!");*/
        //transceiver_request_buff(2,a);
        //transceiver_request_buff(2,&data);
        /*for (int i=0;i<10;i++)
        {*/
                transceiver_request_buff(2,&a);
```

```
                printf("%s",a);
        //}
        tm=tm-start;
        printf("\nВИАС:time %d start %d;\n",tm);

        /*for(int i=0;i<10;i++)
        {
                printf("%s",a[i]);
        }*/
        sleep(1);

        }

}
```

Задание 6. Миссия

Для выполнения задания был предложен следующий код:

```
#define tranPin 11

uint8_t buf[254], cnt = 0;

bool transmit = 0, strtflag = 0;

char to_addr_str[100], from_addr_str[100], msg_type_str[100];

/*int COBS_encode(uint8_t * buf, size_t sz = 254)
{
 if (sz > 254)
   return 1;
 size_t last_null = 0;
 //Serial1.write(sz);
 Serial.write(0xCC);
 Serial.write(buf, sz);
 Serial.write(0xDD);
 for (size_t i = 0; i <= sz; ++i)
 {
  if (i == sz || !buf[i])
  {
   size_t cnt = i - last_null + 1;
   Serial.write(cnt);
   Serial1.write(cnt);
   if (last_null != i)
   {
    //Serial.write((buf + last_null), cnt);
    for (size_t j = last_null; j < i; ++j)
```

```
        {Serial.write(buf[j]);
        Serial1.write(buf[j]);}
      }
      last_null = i + 1;
    }
    //for (size_t j = i
  }
  Serial.write(0);
  Serial1.write(0);

  return 0;
};*/

uint8_t COBS_decode(uint8_t * buf)
{
  uint8_t i = 0xFF, next_null = 0;
  //while (Serial.available())
  while (true)
  {
    if (Serial1.available())
    {
      uint8_t in_byte = Serial1.read();
      //Serial.write(in_byte);
      if (!in_byte || i == 0xFE)
        break;
      if (i == 0xFF)
        next_null = in_byte - 1;
      else if (i == next_null)
      {
        next_null = i + in_byte;
        buf[i] = 0;
      }
      else
        buf[i] = in_byte;
      ++i;
    }
  }
  //Serial.write(0xAA);
  return (i < 0xFE) ? i : 0xFF;
}

/*void PWM_init()
{
  pinMode(tranPin, OUTPUT);

  // initialize timer1
  noInterrupts();           // disable all interrupts
  TCCR1A = 0;
```

```
  TCCR1B = 0;
  TCNT1  = 0;

  OCR1A = 141;           // compare match register 16MHz/1/140KHz
  TCCR1A |= (1 << COM1A0);
  TCCR1B |= (1 << WGM12);   // CTC mode
  TCCR1B |= (1 << CS10);    // 1 prescaler
  //TIMSK1 |= (1 << OCIE1A);  // enable timer compare interrupt
  interrupts();             // enable all interrupts
}*/

/*ISR(TIMER1_COMPA_vect)          // timer compare interrupt service routine
{
  digitalWrite(tranPin, digitalRead(tranPin) ^ 1);   // toggle LED pin
}*/

ISR(PCINT0_vect)
{
   transmit = (transmit) ? 0 : 1;
}

int msg_type_decode(char * res_str, uint16_t msg_type)
{
  char *msg_type_str;
  //Serial.println("checking ");
  //Serial.println(msg_type, HEX);
  switch (msg_type)
  {
    case (0x0000): msg_type_str="RPI_OBC_BASE"; break;
    case (0x0200): msg_type_str="SENS_LIGHT_REQ_RAW"; break;
    case (0x0208): msg_type_str="SENS_LIGHT_CMD_RESET"; break;
    case (0x0210): msg_type_str="SENS_LIGHT_REQ_MAXRAW"; break;
    case (0x0220): msg_type_str="SENS_LIGHT_SET_MINVALUE"; break;
    case (0x0228): msg_type_str="SENS_LIGHT_SET_CALIBRATE"; break;
    case (0x0280): msg_type_str="SENS_LIGHT_RESP_RAW"; break;
    case (0x0288): msg_type_str="SENS_LIGHT_RESP_MAXRAW"; break;
    case (0x0300): msg_type_str="SENS_MAG_REQ_RAW"; break;
    case (0x0304): msg_type_str="SENS_MAG_CMD_RESET"; break;
    case (0x0380): msg_type_str="SENS_MAG_RESP_RAW"; break;
    case (0x0400): msg_type_str="SENS_HYRO_REQ_RAW"; break;
    case (0x0404): msg_type_str="SENS_HYRO_CMD_RESET"; break;
    case (0x0480): msg_type_str="SENS_HYRO_RESP_RAW"; break;
    case (0x0500): msg_type_str="SENS_SUN_REQ_RAW"; break;
    case (0x0508): msg_type_str="SENS_SUN_CMD_RESET"; break;
    case (0x0510): msg_type_str="SENS_SUN_REQ_MAXRAW"; break;
    case (0x0518): msg_type_str="SENS_SUN_SET_MINVALUE"; break;
    case (0x0580): msg_type_str="SENS_SUN_RESP_RAW"; break;
    case (0x0588): msg_type_str="SENS_SUN_RESP_MAXRAW"; break;
```

```
case (0x0600): msg_type_str="SENS_ACCEL_REQ_RAW"; break;
case (0x0604): msg_type_str="SENS_ACCEL_CMD_RESET"; break;
case (0x0680): msg_type_str="SENS_ACCEL_RESP_RAW"; break;
case (0x0D00): msg_type_str="CONTROL_WHEEL0_SET_SPEED"; break;
case (0x0D04): msg_type_str="CONTROL_WHEEL0_CMD_RESET"; break;
case (0x0D08): msg_type_str="CONTROL_WHEEL0_REQ_SPEED"; break;
case (0x0D10): msg_type_str="CONTROL_WHEEL0_RESP_SPEED_CONFIRM"; break;
case (0x0D0C): msg_type_str="CONTROL_WHEEL0_RESP_SPEED"; break;
case (0x0E00): msg_type_str="CONTROL_FAN0_SET_SPEED"; break;
case (0x0E04): msg_type_str="CONTROL_FAN0_CMD_RESET"; break;
case (0x0E08): msg_type_str="CONTROL_FAN0_REQ_SPEED"; break;
case (0x0E10): msg_type_str="CONTROL_FAN0_RESP_SPEED_CONFIRM"; break;
case (0x0E0C): msg_type_str="CONTROL_FAN0_RESP_SPEED"; break;
case (0x0F00): msg_type_str="CONTROL_COIL_SET_VAL"; break;
case (0x0F10): msg_type_str="CONTROL_COIL_CMD_RESET"; break;
case (0x0F20): msg_type_str="CONTROL_COIL_RESP_VAL_CONFIRM"; break;
case (0x1000): msg_type_str="CONTROL_ENGINE_SET_VAL"; break;
case (0x1010): msg_type_str="CONTROL_ENGINE_CMD_RESET"; break;
case (0x1020): msg_type_str="CONTROL_ENGINE_RESP_VAL_CONFIRM"; break;
case (0x2000): msg_type_str="MISC_LASER_CMD_ON"; break;
case (0x2004): msg_type_str="MISC_LASER_CMD_OFF"; break;
case (0x2008): msg_type_str="MISC_LASER_CMD_RESET"; break;
case (0x200C): msg_type_str="MISC_LASER_RESP_ON_CONFIRM"; break;
case (0x2010): msg_type_str="MISC_LASER_RESP_OFF_CONFIRM"; break;
case (0x2100): msg_type_str="MISC_HF_REQ_STATE"; break;
case (0x2110): msg_type_str="MISC_HF_CMD_RESET"; break;
case (0x2120): msg_type_str="MISC_HF_RESP_STATE"; break;
case (0x2200): msg_type_str="MISC_UHF_CMD_SEND"; break;
case (0x2210): msg_type_str="MISC_UHF_CMD_RESET"; break;
case (0x2220): msg_type_str="MISC_UHF_REQ_BUFFER"; break;
case (0x2230): msg_type_str="MISC_UHF_RESP_BUFFER"; break;
case (0x2300): msg_type_str="MISC_SUN_REQ_RAW"; break;
case (0x2308): msg_type_str="MISC_SUN_CMD_RESET"; break;
case (0x2310): msg_type_str="MISC_SUN_REQ_MAXRAW"; break;
case (0x2380): msg_type_str="MISC_SUN_RESP_RAW"; break;
case (0x2388): msg_type_str="MISC_SUN_RESP_MAXRAW"; break;
case (0x2400): msg_type_str="MISC_EARTHFIELD_CMD_SET"; break;
case (0x2408): msg_type_str="MISC_EARTHFIELD_CMD_AMPRESET"; break;
case (0x2410): msg_type_str="MISC_EARTHFIELD_CMD_RESET"; break;
case (0x2500): msg_type_str="MISC_GROUNDLEDS_CMD_SET"; break;
case (0x2510): msg_type_str="MISC_GROUNDLEDS_CMD_RESET"; break;
case (0x2600): msg_type_str="MISC_MOTOGLOBE_CMD_GO_SLOW"; break;
case (0x2610): msg_type_str="MISC_MOTOGLOBE_CMD_GO_FAST"; break;
case (0x2620): msg_type_str="MISC_MOTOGLOBE_CMD_START"; break;
case (0x2630): msg_type_str="MISC_MOTOGLOBE_CMD_STOP"; break;
case (0x2640): msg_type_str="MISC_MOTOGLOBE_CMD_ENABLEINT"; break;
case (0x2650): msg_type_str="MISC_MOTOGLOBE_CMD_DISABLEINT"; break;
case (0x2660): msg_type_str="MISC_MOTOGLOBE_CMD_RESET"; break;
```

```
    case (0x2700): msg_type_str="MISC_ARM_CMD_SET_STATE"; break;
    case (0x2710): msg_type_str="MISC_ARM_CMD_RESET"; break;
    case (0x2720): msg_type_str="MISC_ARM_RESP_STATE"; break;
    default: return 1; break;
  }
  strcpy(res_str, msg_type_str);
  return 0;
}

int addr_decode(char * res_str, uint16_t addr)
{
  char *addr_str;
  //Serial.println("checking ");
  //Serial.println(addr, HEX);
  switch (addr)
  {
    case (0x0001): addr_str="OBC_ADDRESS"; break;
    case (0x0010): addr_str="SENS_LIGHT_ADDRESS"; break;
    case (0x0018): addr_str="SENS_MAG_ADDRESS"; break;
    case (0x001C): addr_str="SENS_HYRO_ADDRESS"; break;
    case (0x0020): addr_str="SENS_SUN_ADDRESS"; break;
    case (0x0028): addr_str="SENS_ACCEL_ADDRESS"; break;
    case (0x0100): addr_str="CONTROL_WHEEL0_ADDRESS"; break;
    case (0x0110): addr_str="CONTROL_FAN0_ADDRESS"; break;
    case (0x0120): addr_str="CONTROL_COIL_ADDRESS"; break;
    case (0x0130): addr_str="CONTROL_ENGINE_ADDRESS"; break;
    case (0x0200): addr_str="MISC_LASER_ADDRESS"; break;
    case (0x0210): addr_str="MISC_HF_ADDRESS"; break;
    case (0x0220): addr_str="MISC_UHF_ADDRESS"; break;
    case (0x0230): addr_str="MISC_SUN_ADDRESS"; break;
    case (0x0240): addr_str="MISC_EARTHFIELD_ADDRESS"; break;
    case (0x0250): addr_str="MISC_GROUNDLEDS_ADDRESS"; break;
    case (0x0260): addr_str="MISC_MOTOGLOBE_ADDRESS"; break;
    case (0x0270): addr_str="MISC_ARM_ADDRESS"; break;
    case (0xFFFF): addr_str="MISC_ADDRESS_BROADCAST"; break;
    default: return 1; break;
  }
  strcpy(res_str, addr_str);
  return 0;
}

void setup() {
  // put your setup code here, to run once:
  PCICR |= (1 << PCIE0);
  PCMSK0 |= (1 << PCINT4);

  Serial.begin(115200);
  Serial1.begin(115200);
```

```
}


void loop() {
  // put your main code here, to run repeatedly:
  /*if (Serial1.available())
  {
    in_byte = Serial1.read();
    Serial.println(in_byte, HEX);
  }*/


  /*if (transmit)
  {
   if (Serial.available())
   {
     uint8_t out_byte = Serial.read();
     Serial1.write(out_byte);
     //COBS_encode(buf
   }
  }
  else
  {
   if (Serial1.available())
   {
     uint8_t in_byte = Serial.read();
     Serial.write(in_byte);
   }
  }*/
  //if (Serial1.available())
  if (Serial1.available())
  {
   //Serial.println("ololo");

   //uncomment this!!!!
   uint8_t cnt = COBS_decode(buf);
   /*Serial.write(0xAA);
   Serial.write(cnt);
   Serial.write(0xBB);
   Serial.write(buf, cnt);
   Serial.write(0xCC);*/

   /*for (size_t i = 0; i < 6; ++i)
   {
    while (!Serial.available()) {};
    buf[i] = Serial.read();
   }*/
```

```
  uint16_t msg_type = ((uint16_t*)buf)[0],
  to_addr = ((uint16_t*)buf)[1],
  from_addr = ((uint16_t*)buf)[2];

  if (msg_type == 0x2308)
    strtflag = 1;
  if (!strtflag)
    return;

  /*Serial.write(msg_type);
  Serial.write(msg_type >> 8);
  Serial.write(0xDD);
  Serial.write(to_addr);
  Serial.write(to_addr >> 8);
  Serial.write(0xEE);
  Serial.write(from_addr);
  Serial.write(from_addr >> 8);
  Serial.write(0xFF);*/


  /*Serial.println(msg_type, HEX);
  //Serial.write(msg_type >> 8);
  //Serial.write(0xDD);
  Serial.println(to_addr, HEX);
  //Serial.write(to_addr >> 8);
  //Serial.write(0xEE);
  Serial.println(from_addr, HEX);
  //Serial.write(from_addr >> 8);
  //Serial.write(0xFF);*/

  if (to_addr == 0x0222)
  {
  Serial.println();
  uint16_t c_msg_type = msg_type, c_to_addr = to_addr, c_from_addr = from_addr;
  bool aflag = true;

  while (msg_type - c_msg_type < 5 && msg_type_decode(msg_type_str, c_msg_type) || (aflag =
false) == true)
  {
    --c_msg_type;
    //aflag = false;
  }
  if (!aflag)
  {
    Serial.print("type ");
    Serial.print(msg_type_str);
    Serial.print(' ');
```

```
  Serial.print(msg_type - c_msg_type);
  Serial.println();
 }
 else
  Serial.println("msg_type_fail");


 aflag = true;
 while (to_addr - c_to_addr < 5 && addr_decode(to_addr_str, c_to_addr) || (aflag = false) == true)
 {
  --c_to_addr;
  //aflag = false;
 }
 if (!aflag)
 {
  Serial.print("to ");
  Serial.print(to_addr_str);
  Serial.print(' ');
  Serial.print(to_addr - c_to_addr);
  Serial.println();
 }
 else
  Serial.println("to_addr_fail");

 aflag = true;
 while (from_addr - c_from_addr < 15 && addr_decode(from_addr_str, c_from_addr) || (aflag =
false) == true)
 {
  --c_from_addr;
  //aflag = false;
 }
 if (!aflag)
 {
  Serial.print("from ");
  Serial.print(from_addr_str);
  Serial.print(' ');
  Serial.print(from_addr - c_from_addr);
  Serial.println();
 }
 else
  Serial.println("from_addr_fail");



 //if (to_addr == 0x0222)
 //{
  //Serial.write(buf, cnt);
  for (size_t i = 6; i < cnt; ++i)
```

```
      {
        char nice_out[10];
        //sprintf(nice_out, "%x ", buf[i]);
        Serial.print(buf[i], HEX);
        Serial.print(' ');
        //Serial.write(nice_out, 2);
      }

      Serial.println();
    //}

    /*uint8_t in_byte = Serial1.read();

    if (!in_byte and cnt++)
    {
      Serial.write(in_byte);
      Serial.println();
    }
    else if (in_byte)
    {
      Serial.write(in_byte);
      cnt = 0;
    }*/
    }
  }

  /*if (Serial.available())
  {
    uint8_t cnt = COBS_decode(buf);
    Serial.write(cnt);
    Serial.write(0xBB);
    Serial.write(buf, cnt);
  }*/
  //Serial.write(buf, 7);
  //COBS_encode(buf, 7);
  //delay(100);
}
```