|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство образования и науки Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТСПЕЦИАЛЬНОЕ МАШИНОСТРОЕНИЕ

КАФЕДРААВТОНОМНЫЕ ИНФОРМАЦИОННЫЕ И УПРАВЛЯЮЩИЕ СИСТЕМЫ

Домашнее задание №1 по курсу

**«Программно-алгоритмическое обеспечение вычислительных**

**средств АИУС»**

Вариант 12

**Группа**: СМ5-32Б

**Студент**: Курган Б.Е.

**Преподаватель***:* Чудников В.В.

Москва, 2019

**Задание:**  
Датчик на валу электродвигателя выдает значения угловой скорости и момента. Диапазон изменения угловой скорости 0...20000 об/мин. Критическими значениями считаются значения меньше 20 об/мин и более 20000 об/мин. Диапазон возможных значений момента 0...1000 Н\*м. Оптимальный момент - 500 Н\*м, допустимое отклонение от номинала 10%. Остальные значения соответствуют аварийному режиму. Произвести проверку значений и вывести пользователю сообщение при аварийном режиме.

**Исходный код:**

**main.c**

#include <stdio.h>

#include <stdlib.h>

#include <locale.h>

#include <string.h>

#include <time.h>

#include "transmitter.h"

#include "receiver.h"

#define PACKS 100

int main(int argc, char \*argv[])

{

setlocale(LC\_ALL,"Rus");

printf("Программа генерирует 100 пакетов со случайными данными\r\n"

"При создании любого пакета данных с вероятностью 25%% данные пакета при передаче изменятся (Симуляция потери данных)\r\n\r\n"

"При использовании ключа '-show' программа будет отображать содержимое каждого пакета после обработки\r\n"

);

srand(time(0));

dataPacket\* packets[PACKS];

int i;

int show = 0;

if((argc == 2) && (strcmp(argv[1],"-show") == 0))

{

show = 1;

}

else if(argc > 2)

{

puts("Unknown input");

return 1;

}

for (i = 0; i < PACKS; i++)

{

packets[i] = (dataPacket\*)malloc(sizeof(dataPacket));

if(packets[i])

{

packets[i] = createData(i);

}

else

{

printf("Memory allocation error");

return 1;

}

}

if(show == 0)

{

for (i = 0; i < PACKS; i++)

{

SendData(packets[i],sizeof(dataPacket));

}

}

else

{

for (i = 0; i < PACKS; i++)

{

SendData(packets[i],sizeof(dataPacket));

show\_packet(packets[i]);

}

}

for(i = 0; i < PACKS; i++)

{

free(packets[i]);

}

puts("Press Enter to exit the program");

getchar();

return 0;

}

**transmitter.h**

#ifndef TRANSMITTER\_H\_

#define TRANSMITTER\_H\_

#pragma pack(push,1)

typedef struct

{

short address;

short packetNumber;

unsigned short ang\_vel;

unsigned short torque;

unsigned char instruction;

unsigned short CRC16\_value;

}dataPacket;

#pragma pack(pop)

dataPacket\* createData(int packetNumber);//создает и заполняет пакет с номером packetNumber

int SendData(dataPacket\* data, unsigned int size); //SendData

void show\_packet(dataPacket\* packet); //отображение содержимого пакета данных

unsigned short CRC16(unsigned char \*pcBlock, unsigned short len); // расчет КС

int rand25(); //функция вероятности 75%

#endif //TRANSMITTER\_H\_

**transmitter.c**

#include <stdlib.h>

#include <stdio.h>

#include "receiver.h"

#include "transmitter.h"

unsigned short CRC16(unsigned char \*pcBlock, unsigned short len)

{

unsigned short crc = 0xFFFF;

unsigned char n;

while (len--)

{

crc ^= \*pcBlock++ << 8;

for (n = 0; n < 8; n++)

{

crc = crc & 0x8000 ? (crc << 1) ^ 0x1021 : crc << 1;

}

}

return crc;

}

dataPacket\* createData(int packetNumber)

{

dataPacket\* newPacket = (dataPacket\*)malloc(sizeof(dataPacket));

if(newPacket != NULL)

{

newPacket->address = 12;

newPacket->packetNumber = packetNumber+1;

newPacket->instruction = 0x01;

newPacket->ang\_vel = (rand()&1 == 0) ? (10000+(rand()%1000)+(rand25()\*10000)) : (10000-(rand()%1000)+(rand25()\*20000));

newPacket->torque = (rand()&1 == 0) ? (500+(rand()%100)+(rand25()\*1000)) : (500-(rand()%100)+(rand25()\*1000));

newPacket->CRC16\_value = CRC16((unsigned char\*)newPacket,(unsigned short)(sizeof(dataPacket)-sizeof(unsigned short)));

if(rand25())

{

newPacket->ang\_vel += 2;

}

return newPacket;

}

else

{

printf("Memory allocation error\r\n");

return NULL;

}

}

int SendData(dataPacket\* data, unsigned int size)

{

printf("(Packet %2d) ", (data->packetNumber));

if(!ReceiveData(data,size))

{

return 0;

}

else

{

return 1;

}

}

void show\_packet(dataPacket\* packet)

{

printf("Address: %d\r\n",packet->address);

printf("Packet Number: %d\r\n",packet->packetNumber);

printf("Instruction: %d\r\n",packet->instruction);

printf("Data: {Angular velocity = %5d | Torque = %5d}\r\n",packet->ang\_vel,packet->torque);

printf("CRC: %d\r\n\n",packet->CRC16\_value);

}

**receiver.h**

#include "transmitter.h"

#ifndef RECEIVER\_H\_

#define RECEIVER\_H\_

int ReceiveData(dataPacket\* data, unsigned int size); //ReceiveData

#endif //RECEIVER\_H\_

**receiver.c**

#include <stdio.h>

#include <stdlib.h>

#include "transmitter.h"

#include "receiver.h"

int ReceiveData(dataPacket\* packet, unsigned int size)

{

unsigned short CRC\_check = CRC16((unsigned char\*)packet,(unsigned short)(sizeof(dataPacket)-sizeof(unsigned short)));

if((CRC\_check) != (packet->CRC16\_value))

{

printf("CRC doesn't match\r\n");

return 1;

}

else

{

if((packet->ang\_vel) < 20 || (packet->ang\_vel>20000))

{

printf("Warning: Critical angular velocity\r\n");

}

else if((packet->torque < 400) ||(packet->torque > 600))

{

printf("Warning: Critical torque\r\n");

}

else

{

printf("All systems nominal\r\n");

}

return 0;

}

}

int rand25()

{

return (rand()&1)&(rand()&1);

}