Name: Fangyao Liu
Partner Name: No Partner
Section: Wednesday 3:30
Date: October 12, 2016

Appendix

Exercise 1

```
//include necessary head files
#include <avr/interrupt.h>
#include <avr/io.h>
#include <util/delay.h>
#include "oi.h"

//name certain ports

#define USB 1
#define CR8 2

//declare functions that are used for communication
void setSerial(uint8_t com);
uint8_t getSerialDestination(void);
void writeChar(char c, uint8_t com);
void delay(void);
//declare function that are used for controlling the robot
void initialize(void);
void powerOnRobot(void);
void baud28k(void);
void delay10ms(uint16_t delay_10ms);
uint8_t byteRx(void);
void flushRx(void);
void Move_Forward(void);
void Move_Stop(void);
void byteTx(uint8_t value);
void intToBi(int k, char* Str);
void SendStringtoPC(char *Message);


int main (void)
{
   //set variables that can store the sensor information
   uint8_t p;
   uint8_t sensor[10];
```

```
char Str[5];

// Initialize the microcontroller
initialize();


// Turn on the Create power if off
powerOnRobot();


// Start the open interface
byteTx(CmdStart);

// Change to 19200 baud
baud28k();


// Take full control of the Create
byteTx(CmdFull);




while(1)
{
        // flush unnecessary data in the register. Delay some time to make sure data have been flushed
        flushRx();
        delay10ms(1);
        // set communication with robot. Delay some time to make sure statement has been executed.
        setSerial(CR8);
        delay10ms(10);
        // Request Sensor Packet 1
        writeChar(142,CR8);
        writeChar(1, CR8);
        // Seperate and Get feedeack of each sensor
        for(p=0;p<10;p++)
            sensor[p]=byteRx();
        // Change int number to binary
        intToBi(sensor[0], Str);

        // flush unnecessary data in the register. Delay some time to make sure data have been flushed
```

```
        flushRx();
        delay10ms(1);
        // set communication with computer. Delay some time to make sure statement has been executed.
        setSerial(USB);
        delay10ms(10);

        // Start to write sensor information to computer
        SendStringtoPC("Wheel Drop caster Value is :");
        writeChar(Str[4],USB);
        writeChar('\n', USB);
        writeChar('\r', USB);

        SendStringtoPC("Wheel Drop left Value is :");
        writeChar(Str[3],USB);
        writeChar('\n', USB);
        writeChar('\r', USB);

        SendStringtoPC("Wheel Drop right Value is :");
        writeChar(Str[2],USB);
        writeChar('\n', USB);
        writeChar('\r', USB);

        SendStringtoPC("Bump left Value is :");
        writeChar(Str[1],USB);
        writeChar('\n', USB);
        writeChar('\r', USB);

        SendStringtoPC("Bump right Value is :");
        writeChar(Str[0],USB);
        writeChar('\n', USB);
        writeChar('\r', USB);

        // use if-else statement to determine whether it is 0 or 1 and then output it
        SendStringtoPC("Cliff left Value is :");
        if((sensor[2]&0x01)==0x01)
        {
            writeChar('1', USB);
            writeChar('\n', USB);
            writeChar('\r', USB);
        }
        else
        {
            writeChar('0', USB);
```

```
        writeChar('\n', USB);
        writeChar('\r', USB);
    }
    SendStringtoPC("Cliff Front left Value is :");
        if((sensor[3]&0x01)==0x01)
    {
        writeChar('1', USB);
        writeChar('\n', USB);
        writeChar('\r', USB);
    }
    else
    {
        writeChar('0', USB);
        writeChar('\n', USB);
        writeChar('\r', USB);
    }
    SendStringtoPC("Cliff Front right Value is :");
        if((sensor[4]&0x01)==0x01)
    {
        writeChar('1', USB);
        writeChar('\n', USB);
        writeChar('\r', USB);
    }
    else
    {
        writeChar('0', USB);
        writeChar('\n', USB);
        writeChar('\r', USB);
    }
    SendStringtoPC("Cliff Right Value is :");
        if((sensor[5]&0x01)==0x01)
    {
        writeChar('1', USB);
        writeChar('\n', USB);
        writeChar('\r', USB);
    }
    else
    {
        writeChar('0', USB);
        writeChar('\n', USB);
        writeChar('\r', USB);
    }

        writeChar('\n', USB);
```

```
                writeChar('\r', USB);

    }

    //Move_Forward();
    //Move_Stop();

}

void Move_Forward(void)
{
        byteTx(137);    // drive opcode

        //Go forward 100 mm/s
        byteTx(0x00); // velocity high byte
        byteTx(0x64); // velocity low byte

        //Go in a straight line
        byteTx(0x80); // radius high byte
        byteTx(0x00); // radius low byte
        delay10ms(300);
}


void Move_Stop(void)
{
        byteTx(137);    // drive opcode
        //Stop the robot
        byteTx(0x00); // velocity high byte
        byteTx(0x00); // velocity low byte
        //Go in a straight line
        byteTx(0x80); // radius high byte
        byteTx(0x00); // radius low byte
}


void initialize(void)
{
    // Turn off interrupts
    cli();

    // configure the I/O pins
    DDRB = 0x10;
    PORTB = 0xCF;
```

```
    DDRC = 0x02;
    PORTC = 0xFF;
    DDRD = 0xE6;
    PORTD = 0x7D;
    UBRR0 = 19;           //set the baud rate to 57600

    // Set up the serial port for 57600 baud
    UBRR0 = Ubrr57600;
    UCSR0B = (_BV(TXEN0) | _BV(RXEN0));
    UCSR0C = (_BV(UCSZ00) | _BV(UCSZ01));
}

void powerOnRobot(void)
{
    // If Create's power is off, turn it on
    if(!RobotIsOn)
    {
        while(!RobotIsOn)
        {
            RobotPwrToggleLow;
            delay10ms(50);    // Delay in this state
            RobotPwrToggleHigh;    // Low to high transition to toggle power
            delay10ms(10);    // Delay in this state
            RobotPwrToggleLow;
        }
        delay10ms(350);    // Delay for startup
    }
}

void baud28k(void)
{
    // Send the baud change command for 28800 baud
    byteTx(CmdBaud);
    byteTx(Baud19200);

    // Wait while until the command is sent
    while(!(UCSR0A & _BV(TXC0))) ;

    // Change the atmel's baud rate
    UBRR0 = Ubrr19200;
    // Wait 100 ms
    delay10ms(10);
}
```

```
void delay10ms(uint16_t delay_10ms)
{
   // Delay for (delay_10ms * 10) ms
   while(delay_10ms-- > 0)
   {
      // Call a 10 ms delay loop
      _delay_loop_2(46080);
   }
}

void delay(void)
{
int i=0,j=0;

      for(i=1;i<=1000;i++)
      {
            for(j=1;j<=1000;j++)
            {
            }
      }

}

uint8_t getSerialDestination(void)
{
      if (PORTB & 0x10)
      return USB;
      else
      return CR8;
}

//set the communication interface

void setSerial(uint8_t com)
{
      if(com == USB)
      PORTB |= 0x10;
      else if(com == CR8)
      PORTB &= ~0x10;
}

void writeChar(char c, uint8_t com)
{
      uint8_t originalDestination = getSerialDestination();
```

```
      if (com != originalDestination)
      {
      setSerial(com);
      delay();
      }

      byteTx((uint8_t)(c));
      if (com != originalDestination)
      {
            setSerial(originalDestination);
            delay();
      }
}



uint8_t byteRx(void)
{
   // Receive a byte over the serial port (UART)
   while(!(UCSR0A & _BV(RXC0))) ;
   return UDR0;
}

void flushRx(void)
{
   uint8_t temp;

   // Clear the serial port
   while(UCSR0A & _BV(RXC0))
      temp = UDR0;
}

void byteTx(uint8_t value)
{
   // Send a byte over the serial port
   while(!(UCSR0A & _BV(UDRE0))) ;
   UDR0 = value;
}

void intToBi(int k, char* Str)          //function that transfer integer to binary string
{
      char bitable[]="01";
      uint8_t i = 0;
      while(i!=5)
```

```
    {
        Str[i] = bitable[k%2];
        k=k/2;
        i++;
    }

}

void SendStringtoPC(char *Message)
{
    while(*Message)
    {
         while(!(UCSR0A & _BV(UDRE0))) ;
        UDR0 = *Message;
        Message++;
    }
}
```

Exercise 2

```
#include <avr/interrupt.h>
#include <avr/io.h>
#include <util/delay.h>
#include "oi.h"

//name certain ports

#define USB 1
#define CR8 2


void setSerial(uint8_t com);
uint8_t getSerialDestination(void);
void writeChar(char c, uint8_t com);
void delay(void);

void initialize(void);
void powerOnRobot(void);
void baud28k(void);
void delay10ms(uint16_t delay_10ms);
uint8_t byteRx(void);
void flushRx(void);
void Move_Forward(void);
void Move_Stop(void);
void byteTx(uint8_t value);
```

```
void intToBi(int k, char* Str);
void SendStringtoPC(char *Message);

// declare functions that will be used to control robot
void Move_Forward(void);
void Move_Stop(void);
void Move_Backward(void);
void Turn_Right(void);
void Turn_Left(void);
void Turn_Right180(void);


int main (void)
{
   uint8_t p;
   uint8_t sensor[10];
   char Str[5];

   // Initialize the microcontroller
   initialize();


   // Turn on the Create power if off
   powerOnRobot();


   // Start the open interface
   byteTx(CmdStart);

   // Change to 28800 baud
   baud28k();


   // Take full control of the Create
   byteTx(CmdFull);




   while(1)
   {
         // keep moving straight
         Move_Forward();
```

```
flushRx();
delay10ms(1);
setSerial(CR8);
delay10ms(10);
writeChar(142,CR8);
writeChar(1, CR8);

for(p=0;p<10;p++)
    sensor[p]=byteRx();

intToBi(sensor[0], Str);
//determine state of sensors
//if there is a bump ahead of car, stop, move back, turn around and move away
//if there is a bump at the right side, stop, move back, turn left and move away
//if there is a bump at the left side, stop, move back, turn right and move away
if(Str[1]=='1')
{
    if(Str[0]=='1')
    {
        Move_Stop();
        Move_Backward();
        Move_Stop();
        Turn_Right180();
        Move_Stop();

    }
    else
    {
        Move_Stop();
        Move_Backward();
        Move_Stop();
        Turn_Right();
        Move_Stop();
    }
}
else
{
    if(Str[0]=='1')
    {
        Move_Stop();
        Move_Backward();
        Move_Stop();
        Turn_Left();
        Move_Stop();
```

```
        }
        else
            Move_Forward();
    }


    //right and left front sensor triggered
    //cliff ahead of the car, stop, move backward, turn around, move straight
    if((sensor[3]&0x01)==0x01)
    {
        if((sensor[4]&0x01)==0x01)
        {
            Move_Stop();
            Move_Backward();
            Move_Stop();
            Turn_Right180();
            Move_Stop();
        }
    }
    //left front sensor triggered
    //cliff at the left side, stop, move backward, turn right, move straight
    if((sensor[3]&0x01)==0x01)
    {
        Move_Stop();
        Move_Backward();
        Move_Stop();
        Turn_Right();
        Move_Stop();
    }


    //right front sensor triggered
    //cliff at the right side, stop, move backward, turn left, move straight
    if((sensor[4]&0x01)==0x01)
    {
        Move_Stop();
        Move_Backward();
        Move_Stop();
        Turn_Left();
        Move_Stop();
    }

}
```

```
   //Move_Forward();
   //Move_Stop();

}

void Move_Forward(void)
{
       byteTx(137);    // drive opcode

       //Go forward 100 mm/s
       byteTx(0x00); // velocity high byte
       byteTx(0x64); // velocity low byte

       //Go in a straight line
       byteTx(0x80); // radius high byte
       byteTx(0x00); // radius low byte
       delay10ms(1);
}



void Move_Backward(void)
{
      byteTx(137);    // drive opcode

      //Go backward 100 mm/s
      byteTx(0xFF); // velocity high byte
       byteTx(0x9C); // velocity low byte

      //Go in a straight line
      byteTx(0x80); //radius high byte
      byteTx(0x00); //radius low byte
      delay10ms(120);
}

void Turn_Right180(void)
{
      byteTx(137);    //drive opcode

      //0 velocity
      byteTx(0x00); // velocity high byte
       byteTx(0x64); // velocity low byte
```

```
	//turn right 90 degrees
	byteTx(0xFF);
	byteTx(0xFF);
	delay10ms(410);
}

void Turn_Right(void)
{
	byteTx(137);    //drive opcode

	//0 velocity
	byteTx(0x00); // velocity high byte
	 byteTx(0x64); // velocity low byte

	//turn right 90 degrees
	byteTx(0xFF);
	byteTx(0xFF);
	delay10ms(205);
}

void Turn_Left(void)
{
	byteTx(137);    //drive opcode

	//0 velocity
	byteTx(0x00); // velocity high byte
	 byteTx(0x64); // velocity low byte

	//turn right 90 degrees
	byteTx(0x00);
	byteTx(0x01);
	delay10ms(205);
}

void Move_Stop(void)
{
	 byteTx(137);    // drive opcode
	 //Stop the robot
	 byteTx(0x00); // velocity high byte
	 byteTx(0x00); // velocity low byte
	 //Go in a straight line
	 byteTx(0x80); // radius high byte
	 byteTx(0x00); // radius low byte
}
```

```
void initialize(void)
{
   // Turn off interrupts
   cli();

   // configure the I/O pins
   DDRB = 0x10;
   PORTB = 0xCF;
   DDRC = 0x02;
   PORTC = 0xFF;
   DDRD = 0xE6;
   PORTD = 0x7D;
   UBRR0 = 19;           //set the baud rate to 57600

   // Set up the serial port for 57600 baud
   UBRR0 = Ubrr57600;
   UCSR0B = (_BV(TXEN0) | _BV(RXEN0));
   UCSR0C = (_BV(UCSZ00) | _BV(UCSZ01));
}

void powerOnRobot(void)
{
   // If Create's power is off, turn it on
   if(!RobotIsOn)
   {
         while(!RobotIsOn)
         {
               RobotPwrToggleLow;
               delay10ms(50);    // Delay in this state
               RobotPwrToggleHigh;    // Low to high transition to toggle power
               delay10ms(10);    // Delay in this state
               RobotPwrToggleLow;
         }
         delay10ms(350);    // Delay for startup
   }
}

void baud28k(void)
{
   // Send the baud change command for 28800 baud
   byteTx(CmdBaud);
   byteTx(Baud19200);
```

```
   // Wait while until the command is sent
   while(!(UCSR0A & _BV(TXC0))) ;

   // Change the atmel's baud rate
   UBRR0 = Ubrr19200;
   // Wait 100 ms
   delay10ms(10);
}

void delay10ms(uint16_t delay_10ms)
{
   // Delay for (delay_10ms * 10) ms
   while(delay_10ms-- > 0)
   {
      // Call a 10 ms delay loop
      _delay_loop_2(46080);
   }
}

void delay(void)
{
int i=0,j=0;

      for(i=1;i<=1000;i++)
      {
            for(j=1;j<=1000;j++)
            {
            }
      }

}

uint8_t getSerialDestination(void)
{
      if (PORTB & 0x10)
      return USB;
      else
      return CR8;
}

//set the communication interface

void setSerial(uint8_t com)
{
```

```
    if(com == USB)
    PORTB |= 0x10;
    else if(com == CR8)
    PORTB &= ~0x10;
}

void writeChar(char c, uint8_t com)
{
    uint8_t originalDestination = getSerialDestination();
    if (com != originalDestination)
    {
    setSerial(com);
    delay();
    }

    byteTx((uint8_t)(c));
    if (com != originalDestination)
    {
        setSerial(originalDestination);
        delay();
    }
}



uint8_t byteRx(void)
{
   // Receive a byte over the serial port (UART)
   while(!(UCSR0A & _BV(RXC0))) ;
   return UDR0;
}

void flushRx(void)
{
   uint8_t temp;

   // Clear the serial port
   while(UCSR0A & _BV(RXC0))
      temp = UDR0;
}

void byteTx(uint8_t value)
{
   // Send a byte over the serial port
```

```
   while(!(UCSR0A & _BV(UDRE0))) ;
   UDR0 = value;
}

void intToBi(int k, char* Str)          //function that transfer integer to binary string
{
      char bitable[]="01";
      uint8_t i = 0;
      while(i!=5)
      {
            Str[i] = bitable[k%2];
            k=k/2;
            i++;
      }

}

void SendStringtoPC(char *Message)
{
      while(*Message)
      {
            while(!(UCSR0A & _BV(UDRE0))) ;
            UDR0 = *Message;
            Message++;
      }
}
```