**Remote Controlled Rollbot Robot**

## **Acceptance Criteria**

### **For the Remote:**

1. **Power control:**

Light on when remote control is powered, light off when it is not and a switch to turn the remote on and off

1. **Wireless remote:**

No wired connection to a non-portable power source and wireless communication between the car and the remote

1. **Remote control:**

The car responds to every direction of the joystick and the tilting of the gyroscope on the remote control

1. **LCD:**

Display distance to obstacle, a warning of possible collision or if collision happens alert the user

**For the Car:**

**Detect collisions:**

Display a warning signal on screen after detecting a collision

**Avoid collisions:**

The robot stops before colliding into object

**Control Robot:**

Movement of the car controlled from the joystick or by tilting the remote

**Detect obstacles:**

LCD shows distance from object in front of the car

**Wireless robot car:**

The car has a power source implemented and receives wireless signals from the remote

## **2. SOFTWARE**

**Robot software Documentation:**

*Overview:*

The code for the robot has multiple functions and libraries used to run the robot. How the robot moves is determined by the movements of a joystick on a remote. The robot knows how the stick moves according to the signals received through its wireless communication capabilities. The robot also sends values from its ultrasonic sensor back to the remove wirelessly.

The ultrasonic sensor is also used for obstacle detection. When the robot reads an object to be too close, it is programmed to come to an automatic stop which prevents further movement towards detected item.

The robot also utilises an accelerometer in order to detect collisions. When a change in acceleration outside of ‘normal’ values is read, the robot recognizes it as a collision and comes to a stop indefinitely.

**main()**

initializes functions, triggers signal to the ultrasonic sensor and makes the motors move according to received value. It also reads accelerator values and determines through the change in values whether or not a collision occurred (in which case the robot stops indefinitely).

**InitExtInt1()**

initializes Int1 to be triggered when echo signal has a logical change at pin D3

**ISR(INT1\_vect)**

Stores the length of the echo signal and sends it to the remote. This code is run when there is a change in logic in the interrupt pin for the ultrasonic sensor.

**InitTimer()**

initializes the timer that counts the length of the echo signal.

**InitTriggerPin()**

initializes PORTC pin2 as output to be used as a trigger pin

**trigger()**

triggers a signal for the ultrasonic sensor by setting the pin C2 as high for 10 microseconds

**initPWM()**

initializes the PWM. The PWM is used to control the voltage output to the motors.

**InitMotorDir()**

Initializes the pins for the H-bridge for the motor direction. Initial direction = forward.

**MotorDirection**

Sets the direction of the wheels depending on the input       . when the input is 0 the motors trigger the wheels to run backwards. 1 makes then turn forward.

**alcChangeAcc()**

This function calculates the change in acceleration. It takes the current acceleration in the x and y directions and compares them with the initialized threshold. If its bigger or smaller than the negative of the threshold the vehicle comes to a stop.

**collisionDetection()**

reads values from the mpu6050 accelerometer and stores them in an array to be read from later.

## **2. HARDWARE**

**Components list:**

* Xbee unit
* LCD I2C display
* Crystal clock 16MHz
* Capstans 22pF
* 3,7V Lithium battery
* Regulator MCP1824ST 300mA, 3,3V
* Switch
* LED
* Accelerometer I2C (GY-521)
* Resistors
* AVRISP programmer
* Joystick
* Atmega328p microprocessor

**Connection explanation:**

**Xbee unit:** Power is connected to the output pin of the regulator, RX is connected to PD0 of the microcontroller, TX is connected to PD1 of the microcontroller and ground is connected to the ground pin of the regulator.

**LCD I2C display:** SDA pin is connected to PC4 of the microcontroller (which is also shared with the SDA of the accelerometer), SCL pin is connected to PC5 of the microcontroller (which is also shared with the SCL of the accelerometer), power is connected to the output pin of the regulator and ground is connected to the ground pin of the of the regulator.

**Crystal clock 16MHz:** Each side of the clock is connected to one resistor. One end of each resistor is connected to the output pin of the regulator. For the other two ends of the resistors, one is connected to PB6 of the microcontroller and the other one is PB7 of the microcontroller.

**3,7V Lithium battery:** The power pin of the battery is connected to the switch, which in turn is connected to the input pin on the regulator. The ground pin of the battery is connected to both capstans and the pins that need to be grounded of joystick (V-, H-, SEL-).

**Regulator MCP1824ST 300mA, 3,3V:** The ground pin is connected to the ground pin of the battery and is also connected to many other components (all mentioned in their respective explanations).

**Switch:** One side of the switch is connected to the positive pin of the battery and the other side is connected to the input pin of the regulator.

**LED:** One side of the LED is connected to the ground pin of the regulator and the other side is connected to a resistor which in turn is connected to the output pin of the regulator.

**Accelerometer I2C (GY-521):**The component LIST8,1 is the power pin of the accelerometer and is connected to the output pin of the regulator. The component LIST8,2 is the ground pin of the accelerometer and is connected to the ground pin of the regulator. The component LIST8,3 is the SCL pin of the accelerometer and is connected to PC5 of the microcontroller (which is also shared with the SCL of the LCD). The component LIST8,4 is the SDA pin of the accelerometer and is connected to the PC4 pin of the microcontroller (which is also shared with the SDA of the LCD).

**AVRISP programmer:** The MISO pin is connected to PB4 of the microcontroller. The SCK pin is connected to PB5 of the microcontroller. The Reset pin is connected to PC6 of the microcontroller and to one side of a resistor. The Vcc pin is connected to the output pin of the regulator and to the other side of the resistor mentioned for the Reset pin. The MOSI pin is connected to PB3 of the microcontroller. The GND pin is connected to the ground pin of the regulator.

**Joystick:** Pins Y+ and H+ are connected to the output pin of the regulator. Pins Y-, H-, SEL-  are all grounded with the battery ground pin. SEL+ is connected to one side of a resistor and the other side is connected to H+. Y and H are connected to pins PC1 and PC0 of the microcontroller respectively.

**Atmega328p microprocessor:** All connections on the microprocessor are mentioned above in their respected explanation texts.