**SPRING 2024**

**CSE-3442 EMBEDDED**

**THE UNIVERSITY OF TEXAS AT ARLINGTON**

Shape

Description automatically generated with medium confidence

**SURVEILANCE ROBOT**

**05/01/24**

**Under the guidance of**

**Prof. Jason Losh**

**By,**

**Luis Contreras**

**(1001918267)**

**Section 1: Introduction**

The purpose of the project is to create a simple surveillance robot that can detect motion of an individual as it navigates though a terrain with limited autonomous travel or can be controlled with IR remote commands. This is achieved with the use of the following hardware which will be discussed further in the report. Motor Drive circuit, Odometer Sensors, Passive Ir sensor, Ultra Sonic sensors, IR sensor ,TM4C123GH6PMI microcontroller.

**Hardware of robot:**

MOTORDIRVER:

**A circuit board with wires and wires

Description automatically generated**As show in the image the pink wire are the ports for the H-bridge motor driver which are ports PB4 through PB7with Peripherals PWM on Motor zero in which GEN 0 and 1 are used with respective CMP A and B are used to drive the PWM functionality Outputs are the purple wires which are connected to the motors.

IR SENSOR:

Is connected to port pC5 that uses the peripheral WideTimer0ccp1 and a GPIO interrupt. It’s also connected to 3.3 v and GND from red board.

PIR:

Is connected to 3.3v and GND the output is connected to Port PA2

Sonar sensor:

Which is top right is connected to GND Trig (PB3), Echo(PD6), 5 V. PD6 use peripheral WT5cpp0 for the to measure the distance

IR LED/IR PHOTORRANSISTOR:

As shown in image the phototransistor is connected in PC4 on the left wheel and PC5 on the right wheel for GPIO interrupts

Photos of Robot:

A circuit board with wires

Description automatically generated A round black object with wires and batteries

Description automatically generated

A machine with wires and a round object

Description automatically generated

Section 2: theory of operation

The Pulse width Modulation works by varying the width of pulses in a periodic signal, which alternates between two states, ON and OFF, by adjusting the ratio of time spent in the ON state to the total period. The PWM can create varying levels of analog signals. For the robot ports, PB4 through PB7 have the peripherals PWM motor 0, which uses generators A and B. Setting each cmp value to different values changes the motor speed. The Universal Asynchronous Receiver-Transmitter is used for serial communication to send data into the microcontroller to do a specific action. Because the receiver transmitter is not synonymous with the system clock, we have to provide a rate at which the data is transmitted, otherwise known as the baud rate. This provides a good foundation for creating and testing new utilities through putty.

Next would be the odometry, which Is made by having an IR led and IR phototransistor in which the IRP is connected to a port with a GPIO interrupt. This will count the number of holes that the IRP has detected via the flashing of the IR LED. This can be done because the design of the robot has a wheel that is 70 mm in radius with 20 perforations in each wheel. Using a simple equation, we know that the linear distance from hole to hole is 14 mm. However, to ensure and protect the system from abnormal amounts of interrupts, we implemented a debounce function that would turn off the interrupts for a while and turn them back on. This was achieved by having a one-shot timer for each wheel that would be interrupted 10 Milliseconds after it turned on. Therefore, this allows the creation of a forward, reverse, clockwise, and counterclockwise function. Odometry was used to stop a specific input for distance and angles. All that was done was a global variable of left and right wheel count that would convert the required distance in mm to the number of holes counted on the left and right wheels. For the steer correction, if either wheel is greater than the other by two, the offset of cmp Val will be added to the lesser wheel cmp value and subtracted from the great wheel cmp value while making sure that the cmp values for the PWM are between 650 and 1023 and pulse or mis twenty-five of the original PWM values. The distance, wheel rotation, and wheel correction are in frequency timers set to the same speed as the odometer because they are background processes that happen every time; locks are made to prevent these inputs from happening when not wanted. The locks, or, In other words, an unsigned 8-bit number, will be set to 1 or zero during the function to execute the desired command. Global variables for cmp values on the left and right wheels were also modified to keep track of the values put in by the user.

The IR rescuer is configured in a port with a wide time that is configured for 32 bits, and interrupts happen on negative edges. In order to parse the data bits transmitted, which are 32 bits, the last one is the trailing bit, which must be taken into account too. A finite state machine parsed this data. That is, the first state is ideal, a timer is zeroed out, and the state is sent to the second. In the second state, the Wtimer is stored in a current time variable and checks if it is a preamble of 9 ms of data set past time to current. If it is not, it returns to state zero. Else, it goes to state three. The data is stored in an unsigned 32-bit number that is shifted left every time it finds a zero or one, but if it finds a one, it adds one to the inside 32-bit number. This happens till all data has been parsed in the final state. It checks if the data equals the predefined button that will execute the command. The sonar is set to a 32-bit wide timer.

The sonar sensors have a trigger and an echo. The trig port sends a ten micro second plus high then low to ping the distance. This is in a void function. The echo is a widetimer inturput configured to detect both edges and happens as fast as the odmotery. Inside, it checks if the first edge is detected. It sets the wtimer to zero, and a global variable phase is set to 1 on the second edge. It converts that time to distance in mm. This is done by multiplying the timer time by the speed of sound in millimeters and dividing that by two times the system clock, which is then stored in a global variable distance sonar. The PIR can check motion because a port is enabled for digital inputs, so the function in motion is called n, which waits 3 seconds, then checks if the port is set to high, then it turns on the red LED ON, the microcontroller.

The final code uses all the previous code in the main. It has a while true loop with a switch case for modes. In case zero, it checks whether auto navigation is turned on. It does putty commands using the UART in. It breaks out of the UART if auto navigation is on. In case one, it stops the robot and checks for motion using the PIR pings for different 90-degree angles, comparing the angles to find the farthest one, reorientates itself to the best distance found, and sets case to case three. If auto navigation is set to zero, it goes back to case zero. In case four, it turns on a frequency timer that happens every .01 sec, which calls the ping distant and stops the robot if a will is 100 mm from the robot. It also calls a function and returns a random distance between the distance found and100 mm. Then, forward is called to travel that distance, and the case is set to zero. However, the second case can only happen if the robot stop flag is officially stopped.

Section 3: observation and conclusion

Observation made throughout the code where that the IR remote the data transmitted had to have a lenient because the timer values would not be exactly the same as the one that was found on the oscilscope.not a difficult modification but something to keep in mind. Also, the debouncing method used for the odometry is necessary as it allows for more accurate wheel counts with the removal of any weird behaviors this could explain why some times the remote would loss connectivity as other implementation of my timers did not have a debounce method for a possible improvement in my project for latter. Also, the wheel perforation being setup as they are with the odomtry did not allow for a precision of less than 14 mm in liner distance but that not a big issue as the project was more base on a un expensive solution to the surveillance robot this can be fix later by adding gyro to the robot making the precision a lot more accurate something I would like to implement later on. Conclusion all in all the project provides a great indication to embed systems as we get to see different integration of make system embed with UART, PWM, Timers and interrupts, how to configure ports and just the design process of a project of this magnitude.