MXEN2003: Ultrasonic Range Sensor Assignment

Written by Istvan Savanyo (21492387)

Purpose/Functionality of the Program

The purpose of the range sensor and code is to light up an LED whenever an object is detected within 38.7cm of the sensor. This is achieved by pinging the sensor to release a short burst of sound waves, the sensor then 'listens' for any echo response that may have rebounded from the waves reflecting off a solid surface. The time-length of this response is used to determine the relative distance of the object, as distance and signal length share a closely proportional relationship. Control structures in the algorithm dictate whether the LED will light up.

The general functionality of the program is to perform a task depending on the proximity of the sensor to other objects. In the current program, the task is simply turning on an LED. However, this functionality can be applied to other tasks, for example, if a robot detects an object too close to its sensors it could shut off its motors to prevent a collision.

How the Program & Sensor Operate (Code Logic)

The PING))) sensor works by sending out a short ultrasonic burst and then listening for any echo that may occur from the sound waves reflecting off a surface. An Input Trigger Pulse (t_{out}) is sent to the sensor to produce the sound, typically 5 microseconds in length. There is also a 200-microsecond delay between measurements to prevent the overlapping of signals.

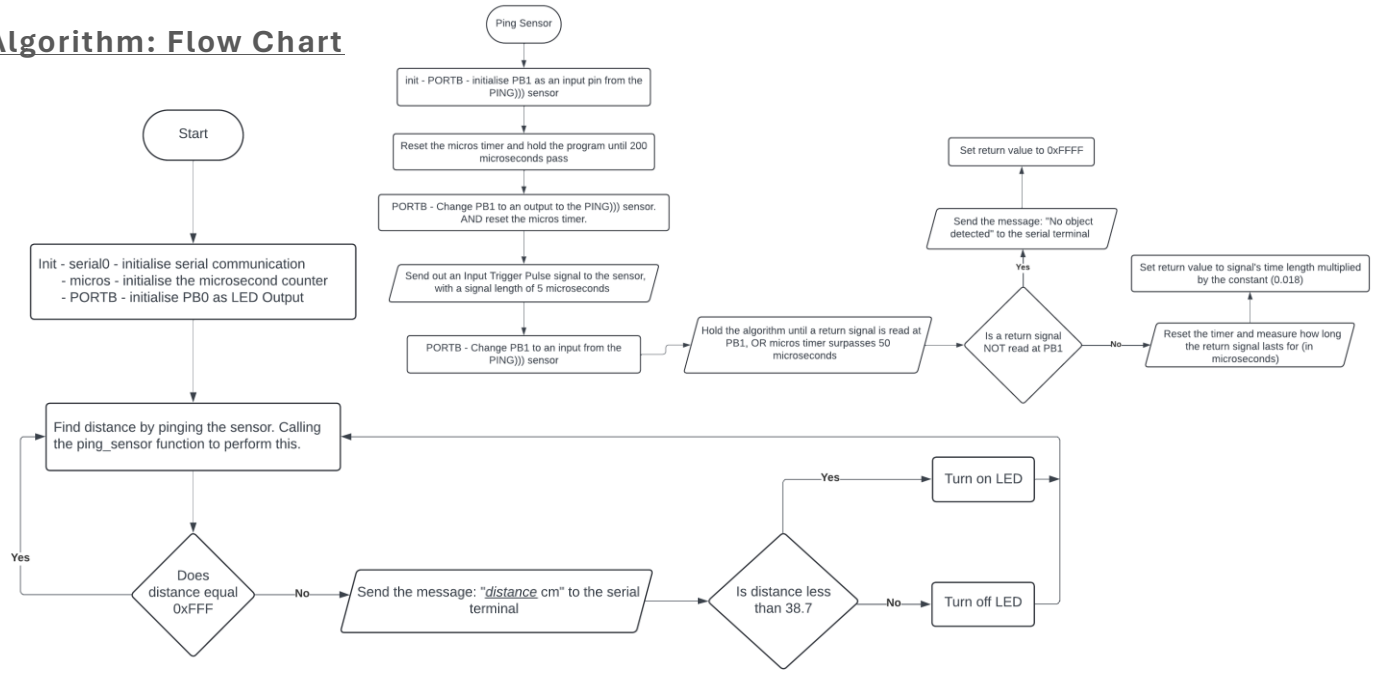
The ping_sensor() function in the code is responsible for pinging signals to the sensor and reading the response. The function first sets pin PB1, which is connected to the PING)))'s SIG pin, as an input. The program is held in this state using polling until 200 microseconds have passed, acting as the PING)))'s delay. Tracking the elapsed time was achieved by using one of the Arduino's internal timers, labelled *micros* in the code.

PB1 is then set to both output and HIGH, the program is then polled for 5 microseconds before the pin reverts to input and LOW. This is the Input Trigger Pulse sent out to the sensor. The program waits until either 50 microseconds pass have passed, or a signal is read at PB1. If the time is passed, it indicates that there is no object present. . An error message is then sent to terminal warning the user there is no object present, and 0xFFFF is returned to main. Otherwise, if a signal is read, then the micros timer is reset, and the time-length of the signal is recorded. This time is then multiplied by a predetermined constant to get the associated distance value, which is returned to main.

The main() program starts by initialising PB0 as the LED output, then calls ping_sensor() at the start of every loop. Control structures are used to determine how the program responds to the outputted distance value. If the distance is lower than 38.7 then PB0's output is set to HIGH, turning on the LED. Otherwise, the program will set PB0's output to LOW, turning the LED off.

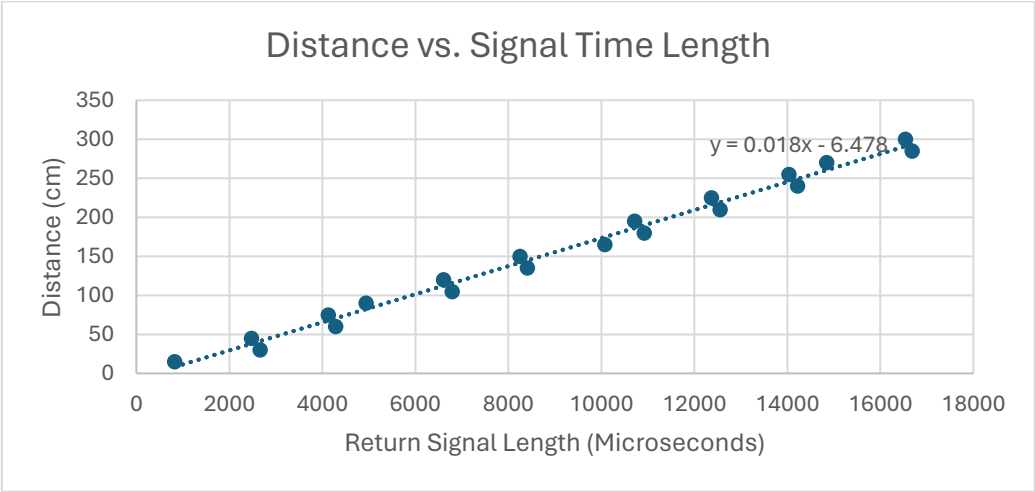
Following this algorithm, the code and sensor together are able to consistently track the distance of any object that is within the range of the sensor, and light up the LED when it becomes too close.

Algorithm: Flow Chart



Relevant Range Sensor Calculations

The original *constant* value (when multiplied by the signal length) produced distance values that were inaccurate within the TinkerCad space. To fix this problem, the constant was temporarily removed, and the raw microseconds signal length was recorded in 15cm intervals between the ranges of 15cm to 300cm. The data was then plotted on a graph with a line of best fit.



This line of best fit accurately represents the pattern between signal length and distance. The gradient of this line (0.018) ended up replacing the original constant so that the program produced a more accurate reading to the true distance.

The resistance of the resistor in the LED portion of the circuit also needed to be calculated. Using the "Standard LED Red Emitting Colour" datasheet, the continuous forward current through the LED should be 20mA. We assume a worst-case scenario where the LED has malfunctioned and acts as a short circuit when doing this calculation. With an input voltage of 5V, using ohms law the resistance value can be calculated as:

$$R = \frac{V}{I} = \frac{5}{20 * 10^{-3}} = 250\ \Omega$$