Modeled Radar System using Sonar

https://youtu.be/Z2UKC_wSfbI

Henri Malahieude

11/30/22

High-Level Description

The “Radar System Model using Sonar” project is exactly as described. It models the Radar System often seen in use in the military and the meteorological science in miniature form. Using the Sonar as a replacement for the Radar, the system uses a Servo to actuate the Model with a 4-Digit 7-Segment Display and Buzzer as output.

Complexities

The first complexity is the 4-Digit 7-Segment Display. It boasts 12 pins and proper timing for effective use. However, 2 pins (Decimal Point & Digit 4) were tied to their respective “off” states to free those pins on the microcontroller for the Servo and the Passive Buzzer. Timing for this piece requires a Digit to be selected and the Segment Pins to be turned on. Since the Segment pins control all Digits, it is important to only have one Digit pin selected at a time. The minimum interval between “lighting” each Digit is ~4ms. Anything less leaves a ghost trail in the other digits than the intended one.

The second complexity is the HC-SR04 Ultrasonic Sensor. Pins are self-explanatory, but timing is most important. The datasheet reads that to begin the distance measurement, only a 10 microsecond pulse is needed on the “Trig” pin in order to obtain a reading from the “Echo” pin. Distance measurement isn’t obtained from measuring the time between sending the trigger and reading the echo, rather that logic is done on the Sensor itself. Distance is relayed by the length of time that the “Echo” pin is *HIGH* for. Then that time must be divided by 58 in order to obtain a reasonable approximation of the distance in centimeters.

The last complexity is the SG90 Analog Servo Motor. There is only 1 pin of importance to control the motor. However, issues stemmed from supplying the motor with proper voltage. Putting the supply wires too far from the “source” wires increased the resistance enough for the motor to not have enough power to run itself. Figuring this out was the most annoying. The pin of importance is simply handled by a PWM where the pulse is at most 2ms of 20ms, where the range between 1 - 2 ms controls the exact angle of the motor.

Another issue related to the SG90 Analog Servo Motor was the jitter experienced once the state machine implementing it was added with the other state machines. My hypothesis is that since the other state machines also take CPU clock cycles, the timing (which was already very sensitive) is thrown off by the other state machines ticking.

User Guide

Point Servo in direction to watch. Power system on. If the system buzzes/beeps, then an object has entered the set range of the “Radar.”

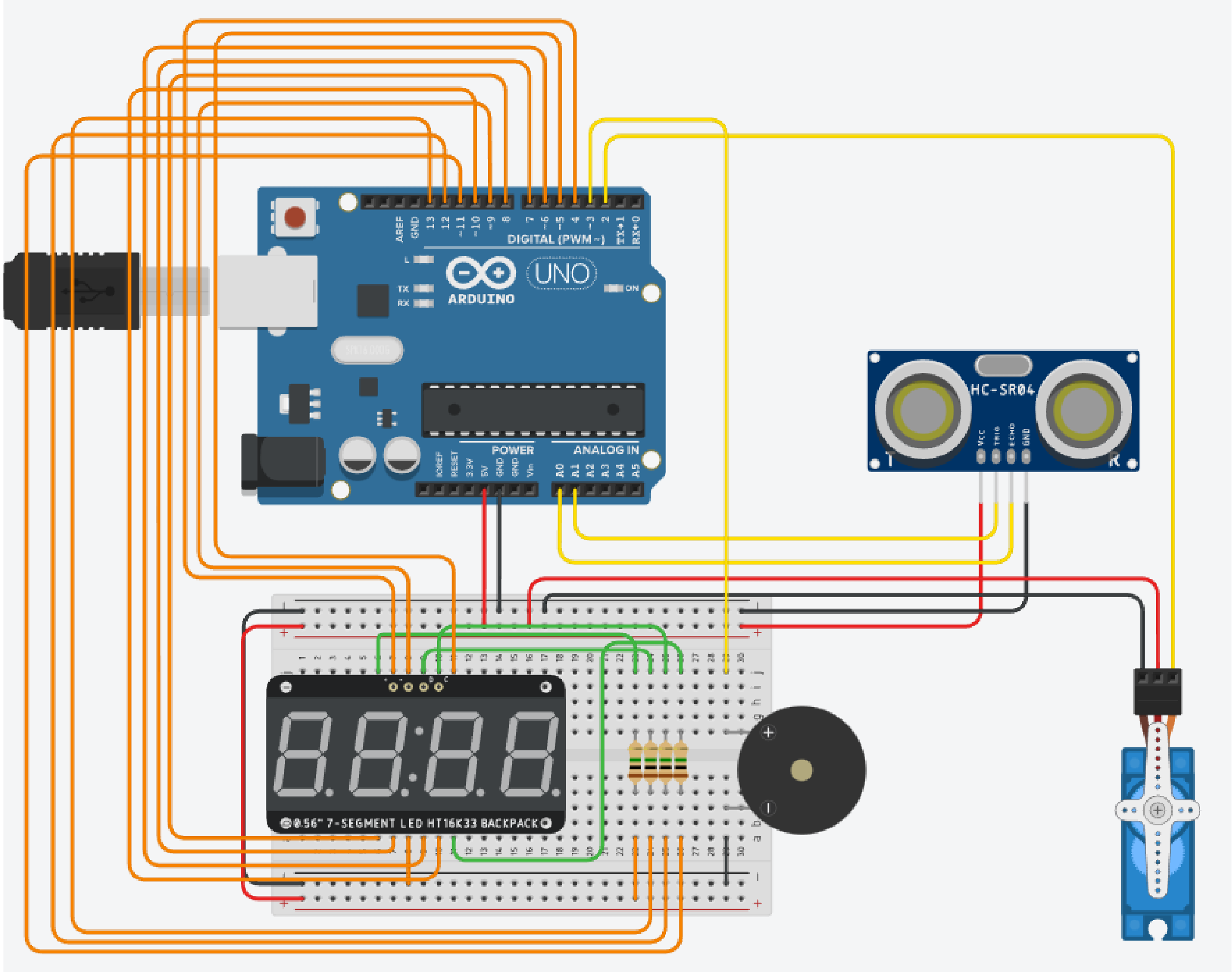
List of Hardware Components Used

1. Elegoo Uno R3
2. HC-SR04 Ultrasonic Sensor
3. SG90 Analog Servo
4. 5641AS 4-Digit 7-Segment Display
5. Passive Buzzer
6. Four 1000k Ohm Resistors

List of Software Libraries Used

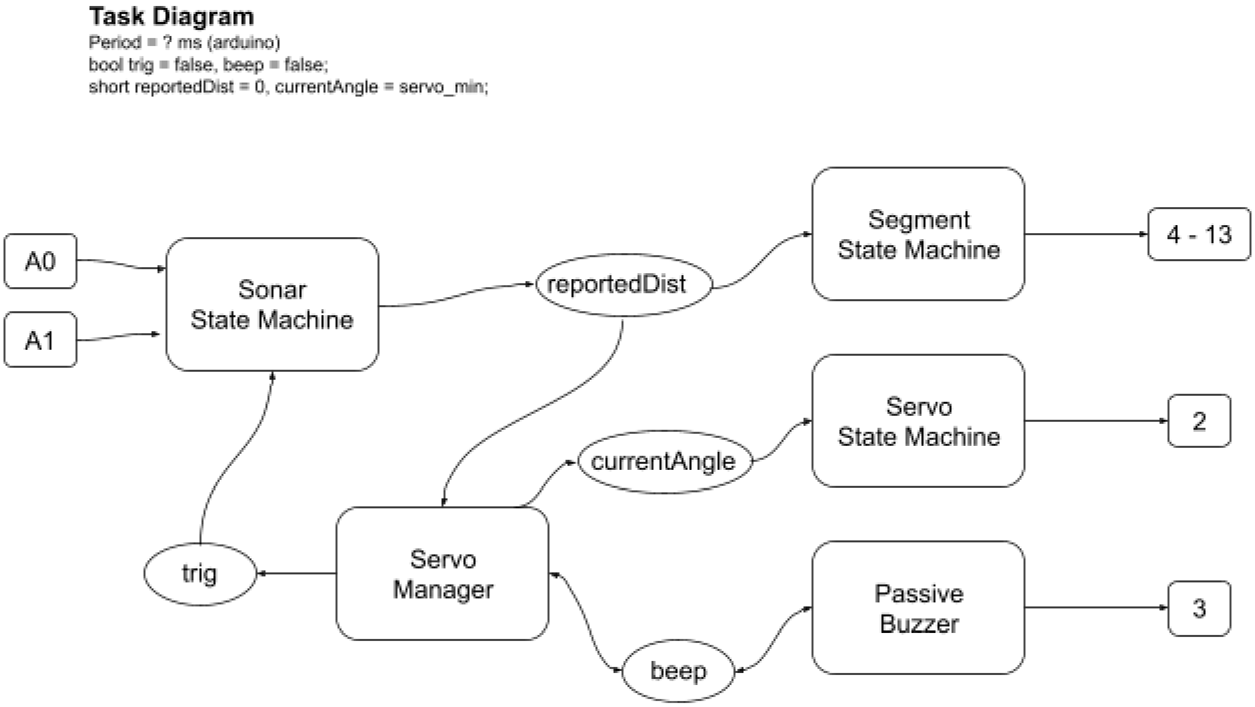
None ;(

Wiring Diagram



Note: TinkerCAD does not have the 5641AS 4-Digit 7-Segment Display I used, so I used a placeholder. The wiring diagram is accurate otherwise.

Task Diagram



Task Diagram

Period = ? ms (arduino)
bool trig = false, beep = false;
short reportedDist = 0, currentAngle = servo_min;

SynchSM Diagrams

Global Variables:

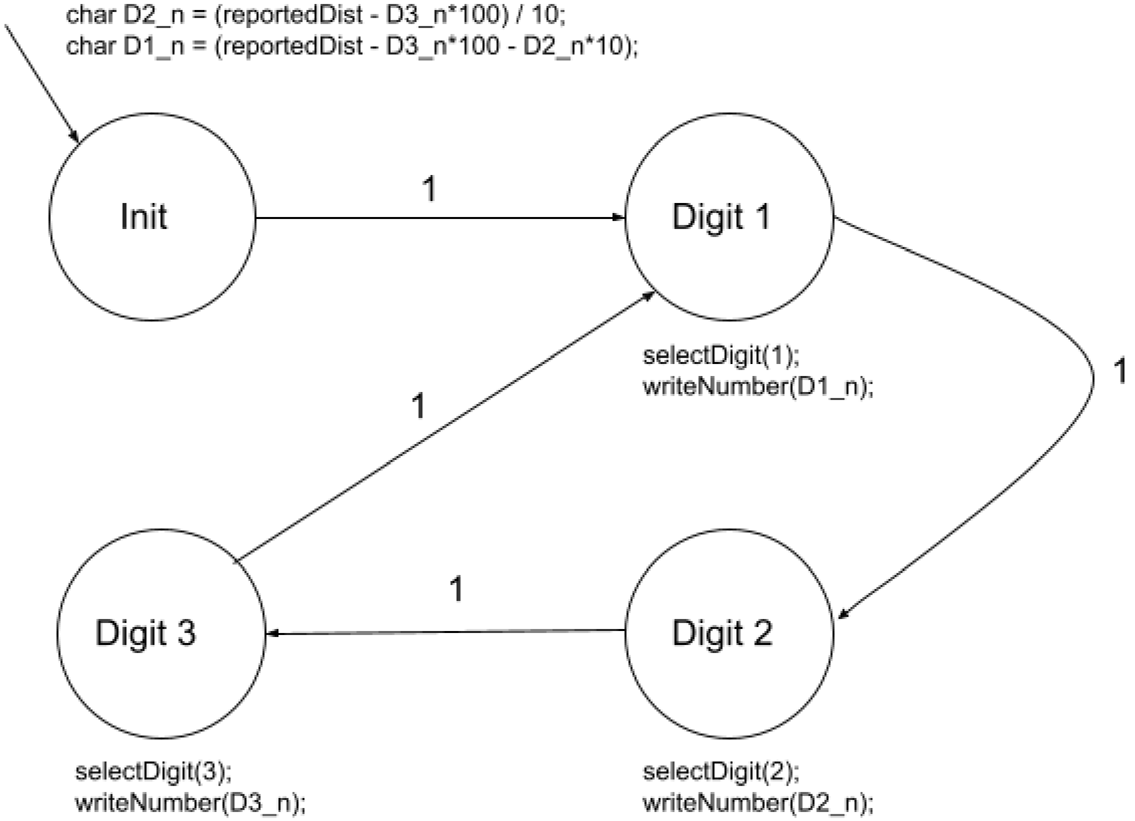
- bool trig = false;
- bool beep = false;
- short reportedDist = 0;
- short beepDist = 15;
- short servo_min = 800;
- short servo_max = 1800;
- short currentAngle = servo_min;

Global Functions:

- void selectDigit(char d = 1);
- void writeNumber(char n = 0);
- void clamp(unsigned short *v, int bot, int top);

Segment State Machine

Period = 5 ms
char D3_n = reportedDist / 100;
char D2_n = (reportedDist - D3_n*100) / 10;
char D1_n = (reportedDist - D3_n*100 - D2_n*10);



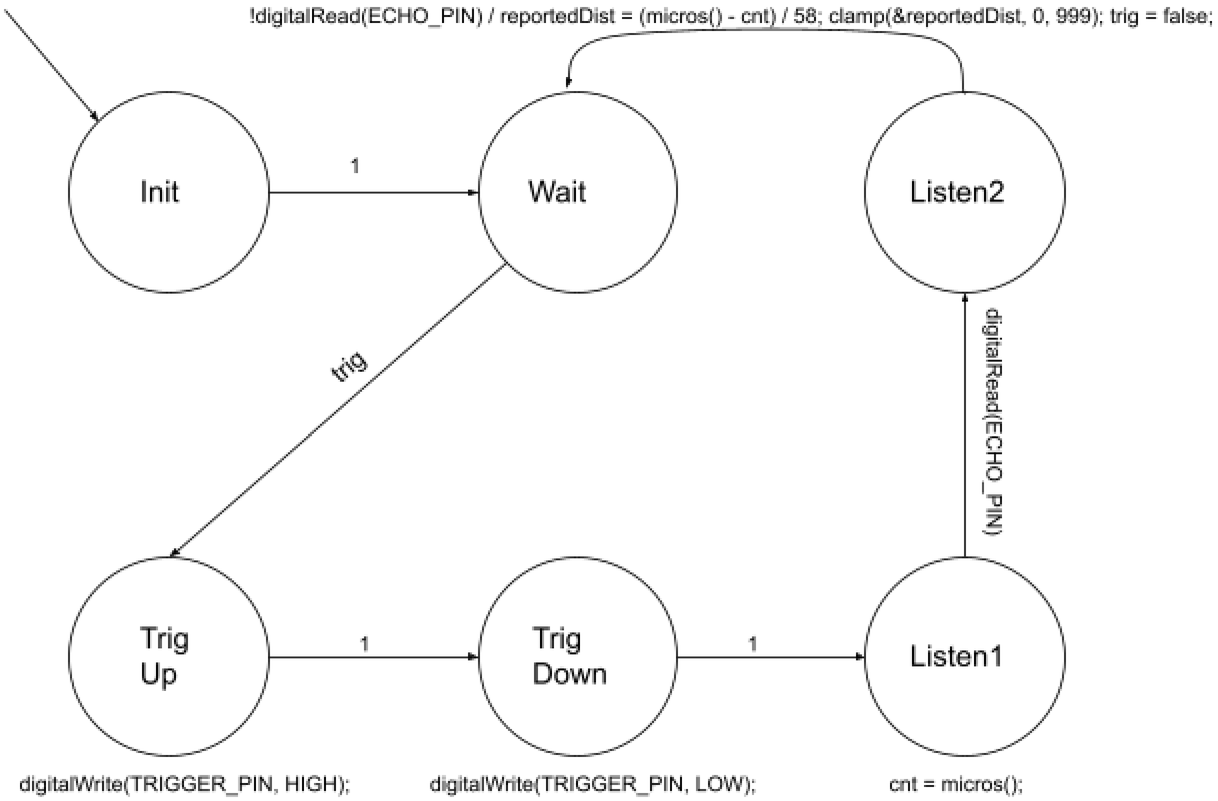
selectDigit(1);
writeNumber(D1_n);

selectDigit(3);
writeNumber(D3_n);

selectDigit(2);
writeNumber(D2_n);

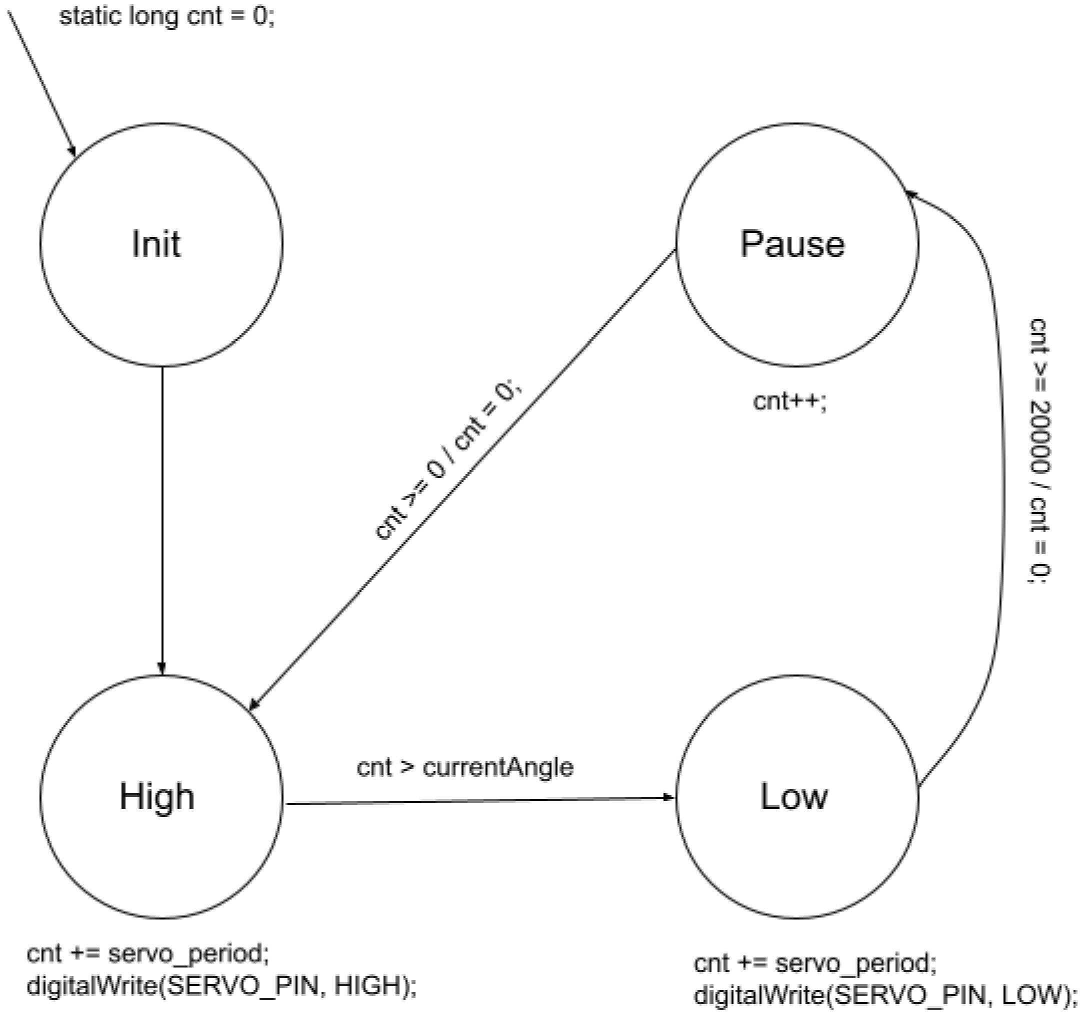
Sonar State Machine

Period = 10 us (microseconds)
static unsigned long cnt = 0;


!digitalRead(ECHO_PIN) / reportedDist = (micros() - cnt) / 58; clamp(&reportedDist, 0, 999); trig = false;
Init
1
Wait
Listen2
trig
digitalRead(ECHO_PIN)
Trig Up
1
Trig Down
1
Listen1
digitalWrite(TRIGGER_PIN, HIGH);
digitalWrite(TRIGGER_PIN, LOW);
cnt = micros();

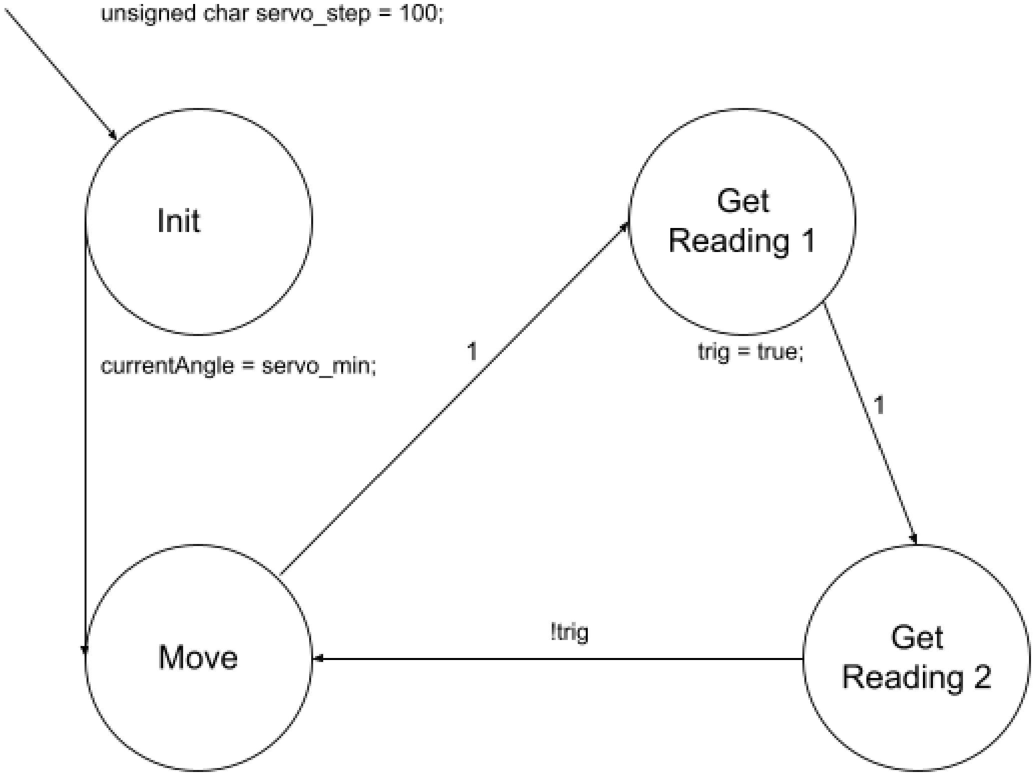
Servo State Machine

Period = 100 us (microseconds)
static long cnt = 0;



Servo Manager State Machine

Period = 500 ms
bool forward = true;
unsigned char servo_step = 100;



```
currentAngle = servo_min;
```

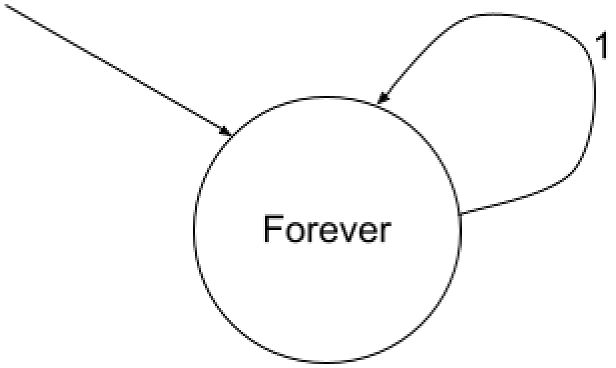
```
trig = true;
```

```
if (forward){
   currentAngle += servo_step;
   if (currentAngle >= servo_max){
     forward = false;
     currentAngle = servo_max;
   }
 }else{ //moving backward
   currentAngle -= servo_step;
   if (currentAngle <= servo_min){
     forward = true;
     currentAngle = servo_min;
   }
 }
```

```
if (!trig){
  if (reportedDist <= beepDist){
    beep = true;
  }
}
```

Passive Buzzer State Machine

Period = 250 ms
bool beep = false;



```
if (beep){
  digitalWrite(BEEP_PIN, HIGH);
  beep = false;
}else{
  digitalWrite(BEEP_PIN, LOW);
}
```