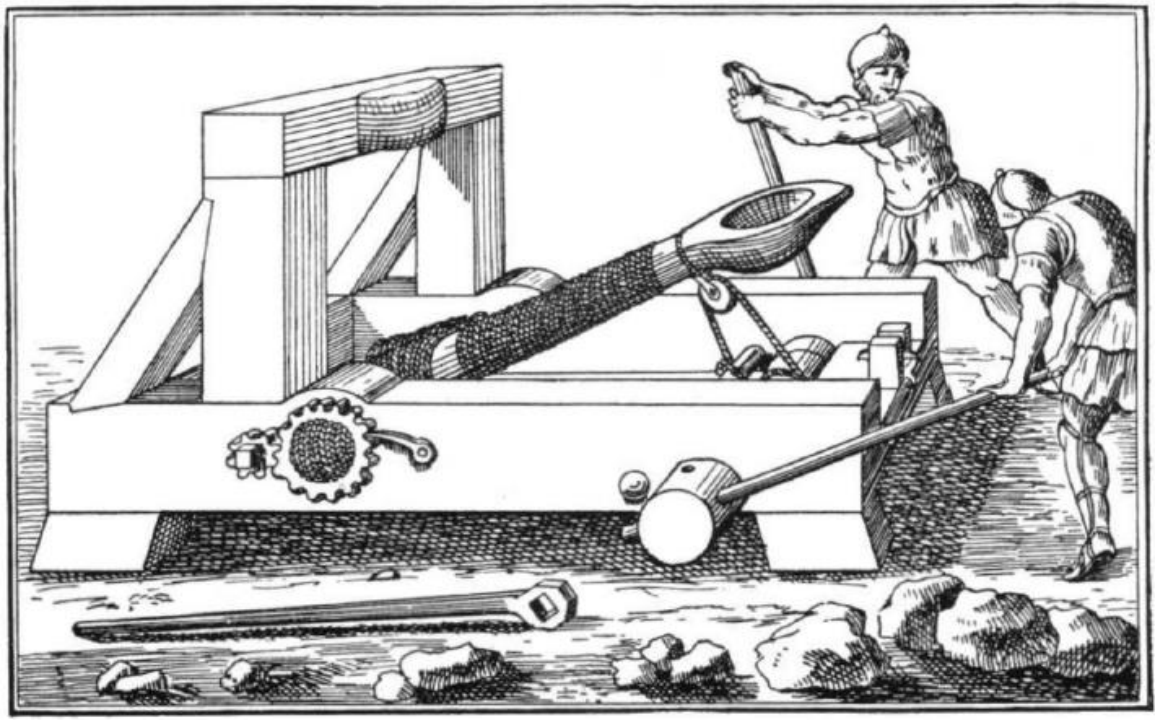
Engineering Design Project I
UTA013

Dr. Poonam Verma
Assistant Professor
ECED, TIET, Patiala

Engineering Design Project-I: Mangonel

- In Engineering Design Project-I, Mangonel (Roman catapult) is to be designed and implemented.



Mangonel: Electronics Part

The Electronic Part is divided into 4 sections:

- Programming of Arduino Digital I/O pins for various applications.
- Sensing any activity through Arduino and instructing accordingly. Also, data capturing through sensors.
- Interfacing of hardware and software to do a specific task (using 7-segment display)
- Develop a micro-electronic circuit to determine and display the angular velocity of the throwing arm.

Mangonel: Electronics Part

The Electronic Part is divided into 4 sections:

- Programming of Arduino Digital I/O pins for various applications.
- Sensing any activity through Arduino and instructing accordingly. Also, data capturing through sensors.
- Interfacing of hardware and software to do a specific task (using 7-segment display)
- Develop a micro-electronic circuit to determine and display the angular velocity of the throwing arm.

Objective

1. To compute the time taken by Mangonel arm to cross two IR sensor pairs using Arduino.

2. To display the computed time on 7-segment display.

3. To compute the angular velocity of the throwing arm.

Objective-1: Computation of time

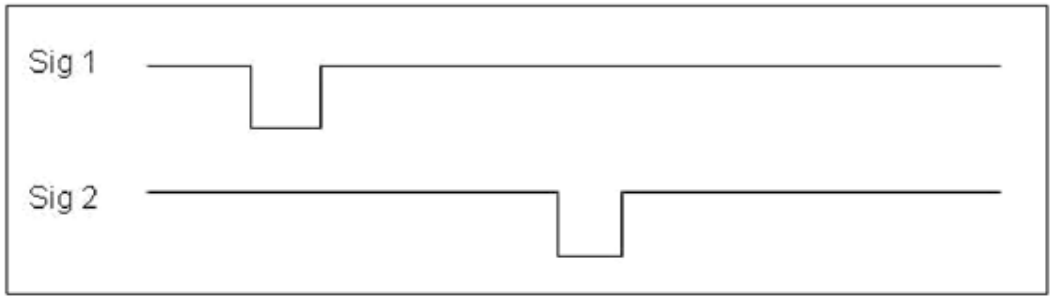
Following steps are executed to compute the time of throwing arming

1. Capture two IR sensor signals from the Mangonel.
2. Combine these two signals into a single signal using AND gate.
3. Convert the resulting signal into a long pulse using JK flip-flop.
4. Compute the length of the pulse with the help of pulseIn function in Arduino.

Objective-1: (Cont…)

Following steps are executed to compute the time of throwing arming

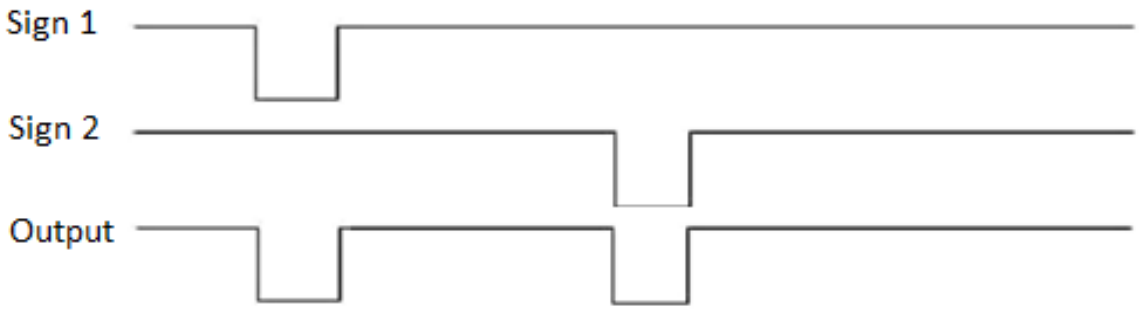
1. **Capture the IR sensor signals from the Mangonel.**
2. Combine these two signals into a single signal using AND gate.
3. Convert the resulting signal into a long pulse using JK flip-flop.
4. Compute the length of the pulse with the help of pulsein function in Arduino



Objective-1: (Cont…)

Following steps are executed to compute the time of throwing arming

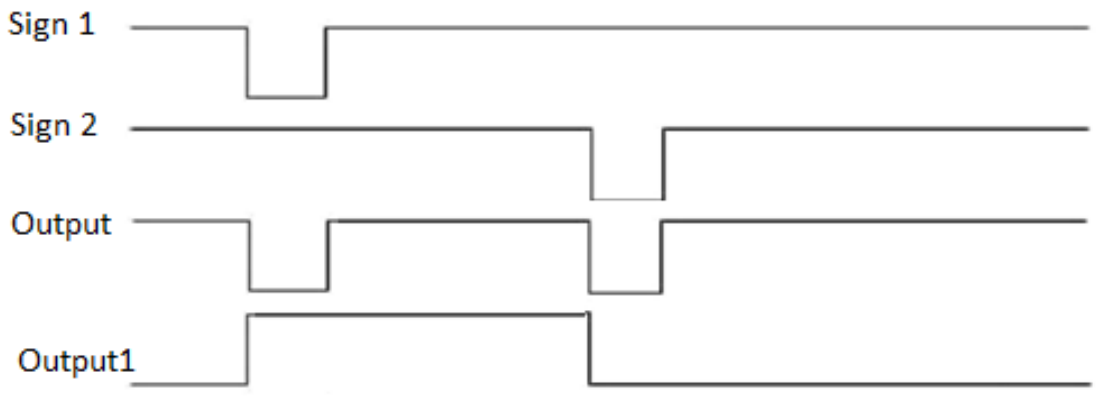
1. Capture two IR sensor signals from the Mangonel.
2. **Combine these two signals into a single signal using AND gate (IC CD4081).**
3. Convert the resulting signal into a long pulse using JK flip-flop.
4. Compute the length of the pulse with the help of pulsein function in Arduino.



Objective-1: (Cont…)

Following steps are executed to compute the time of throwing arming

1. Capture two IR sensor signals from the Mangonel.
2. Combine these two signal into a single signal using AND gate.
3. **Convert the resulting signal into a long pulse using JK flip-flop (IC CD4027).**
4. Compute the length of the pulse with the help of pulsein function in Arduino.



Objective-1: (Cont…)

Following steps are executed to compute the time of throwing arm

1. Capture two IR sensor signals from the Mangonel.
2. Combine these two signal into a single signal using AND gate.
3. Convert the resulting signal into a long pulse using JK flip-flop.
4. **Compute the length of the pulse with the help of *pulseIn* function in Arduino.**

PulseIn function

- Reads a pulse (either HIGH or LOW) on a pin.
- If value is HIGH, pulseIn() waits for the pin to go from LOW to HIGH, starts timing, then waits for the pin to go LOW and stops timing.
- It returns the length of the pulse in μs or gives up and returns 0 if no complete pulse was received within the timeout.
- Works on pulses from 10 μs to 3 minutes in length.

PulseIn(): Syntax

- pulseIn(pin, value)
- pulseIn(pin, value, timeout)

PulseIn(): Parameters

- pin: the number of the pin on which you want to read the pulse. (int)
- value: type of pulse to read: either HIGH or LOW. (int)
- timeout (optional): the number of μs to wait for the pulse to start; default is one second (unsigned long)
- It returns the length of the pulse (in microseconds) or 0 if no pulse started before the timeout (unsigned long)

PulseIn(): Example code

The example calculate the time duration of a pulse on pin 7.

```
int pin = 7;
unsigned long duration;
void setup()
{
  pinMode(pin, INPUT);
}
void loop()
{
  duration = pulseIn(pin, HIGH);
}
```

Objective-2: Display time on 7-segment

```
void setup()
{
  pinMode(2, OUTPUT);       // D0
  pinMode(3, OUTPUT);       // D1
  pinMode(4, OUTPUT);       // D2
  pinMode(5, OUTPUT);       // D3 (MSB)
  pinMode(6, OUTPUT);       // Latch Disable 0
  pinMode(9, OUTPUT);       // Latch Disable 1
  pinMode(8, OUTPUT);       // Latch Disable 2
  pinMode(10, OUTPUT);      // reset pin
  pinMode(13, INPUT);       // received from JK flip flop
}
```

Objective-2: Display time on 7-segment

- void loop()
- {
  digitalWrite(10, HIGH);
  digitalWrite(10, LOW);
- unsigned int a = pulseIn(13,HIGH,180000000);
- // returns a 5 digit number
  int b=a/100; // returns a 3 digit number
  int c=b%10; // extracts LSB
  int d=a/1000; // returns a 2 digit number
  int e=d/10; // returns MSB
  int f=d%10;

Objective-2: Display time on 7-segment

- int i= c%2;
- int j= (c/2)%2;
- int k= (c/4)%2;
- int l=(c/8)%2; // Converts the BCD equivalent of decimal number.

- digitalWrite(2, i);
- digitalWrite(3, j);
- digitalWrite(4, k);
- digitalWrite(5, l);
- digitalWrite(6, HIGH);
- digitalWrite(6, LOW);

Objective-2: Display time on 7-segment

- int m=f%2;
- int n=(f/2)%2;
- int o=(f/4)%2;
- int p=(f/8)%2;

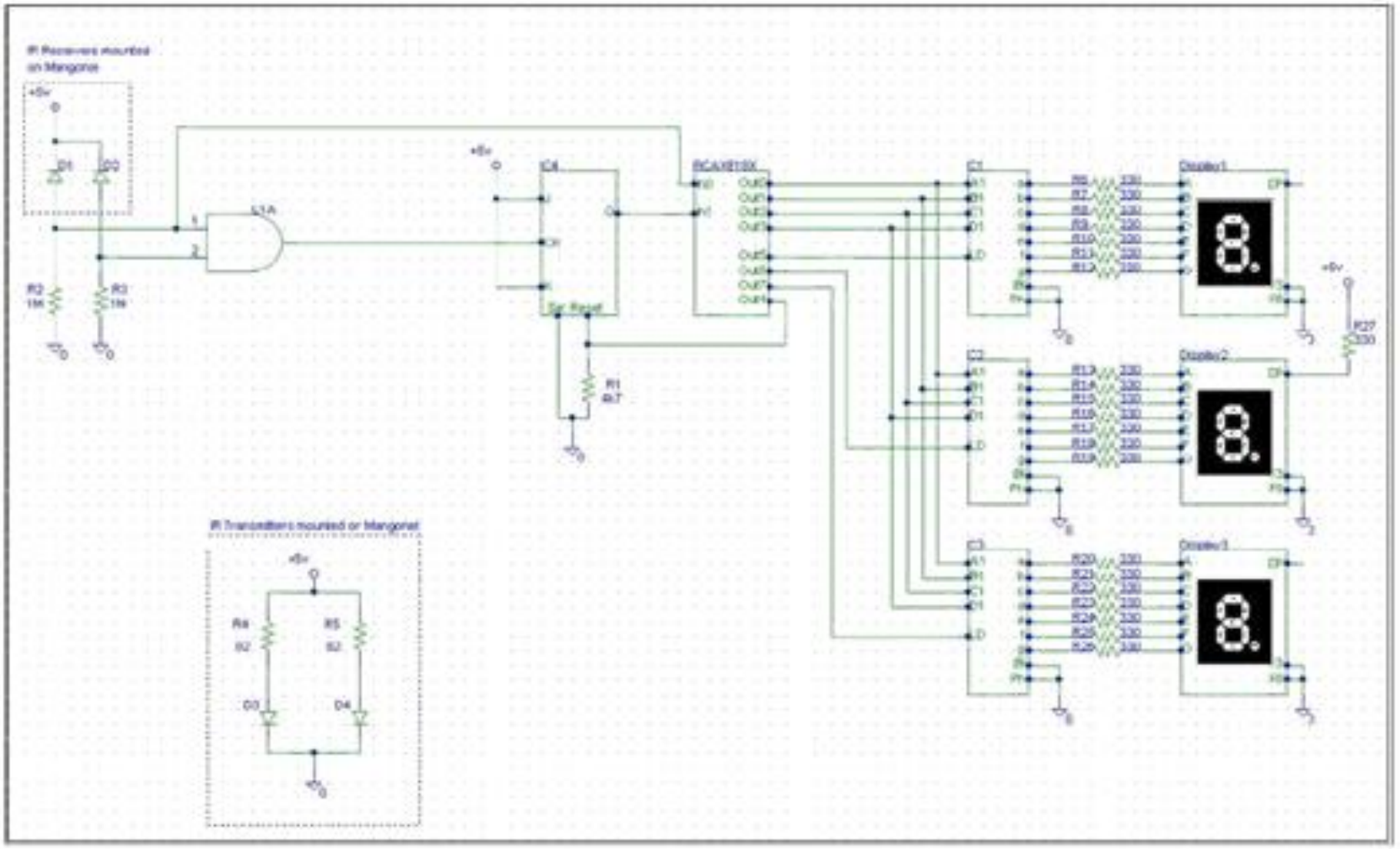
- digitalWrite(2, m);
- digitalWrite(3, n);
- digitalWrite(4, o);
- digitalWrite(5, p);
- digitalWrite(8, HIGH);
- digitalWrite(8, LOW);

Objective-2: Display time on 7-segment

- int q=e%2;
- int r=(e/2)%2;
- int s=(e/4)%2;
- int t=(e/8)%2;

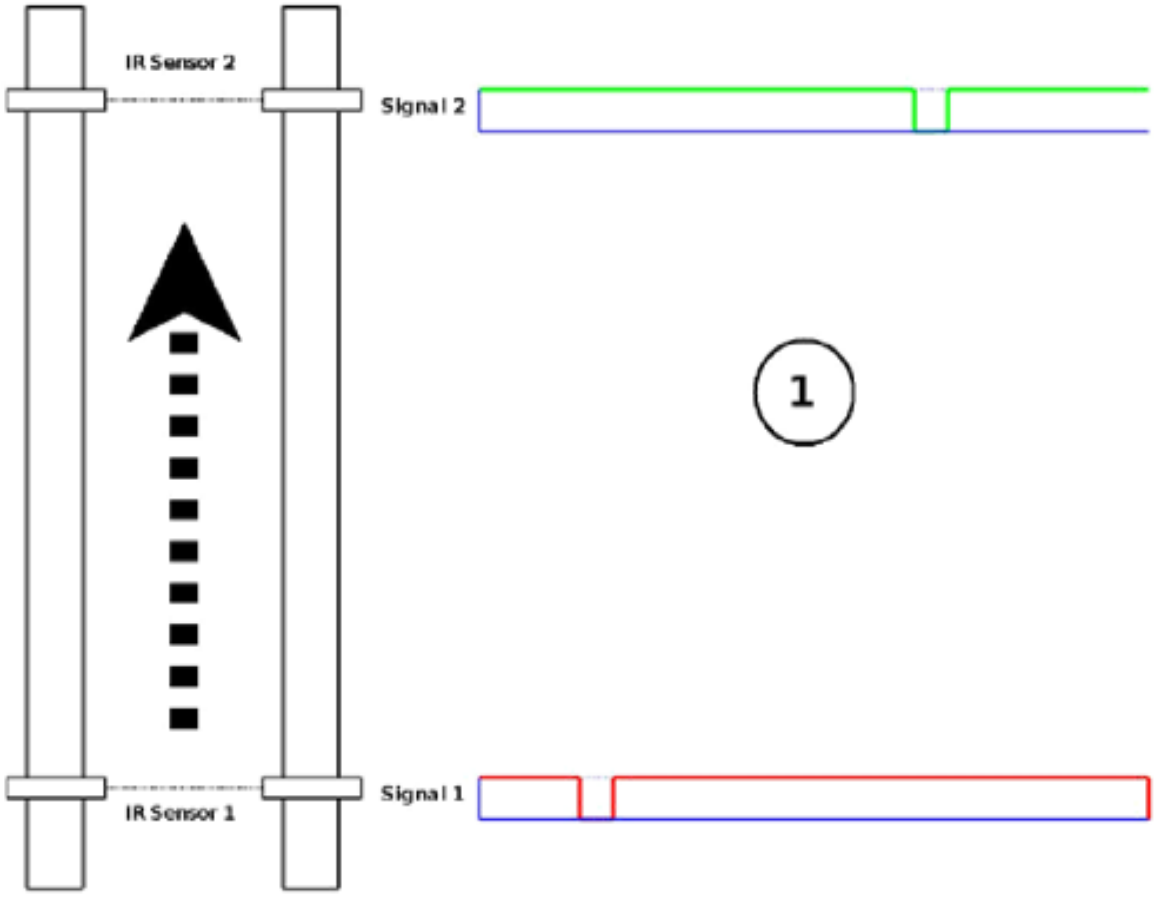
- digitalWrite(2, q);
- digitalWrite(3, r);
- digitalWrite(4, s);
- digitalWrite(5, t);
- digitalWrite(9, HIGH);
- digitalWrite(9, LOW);

Schematic Diagram



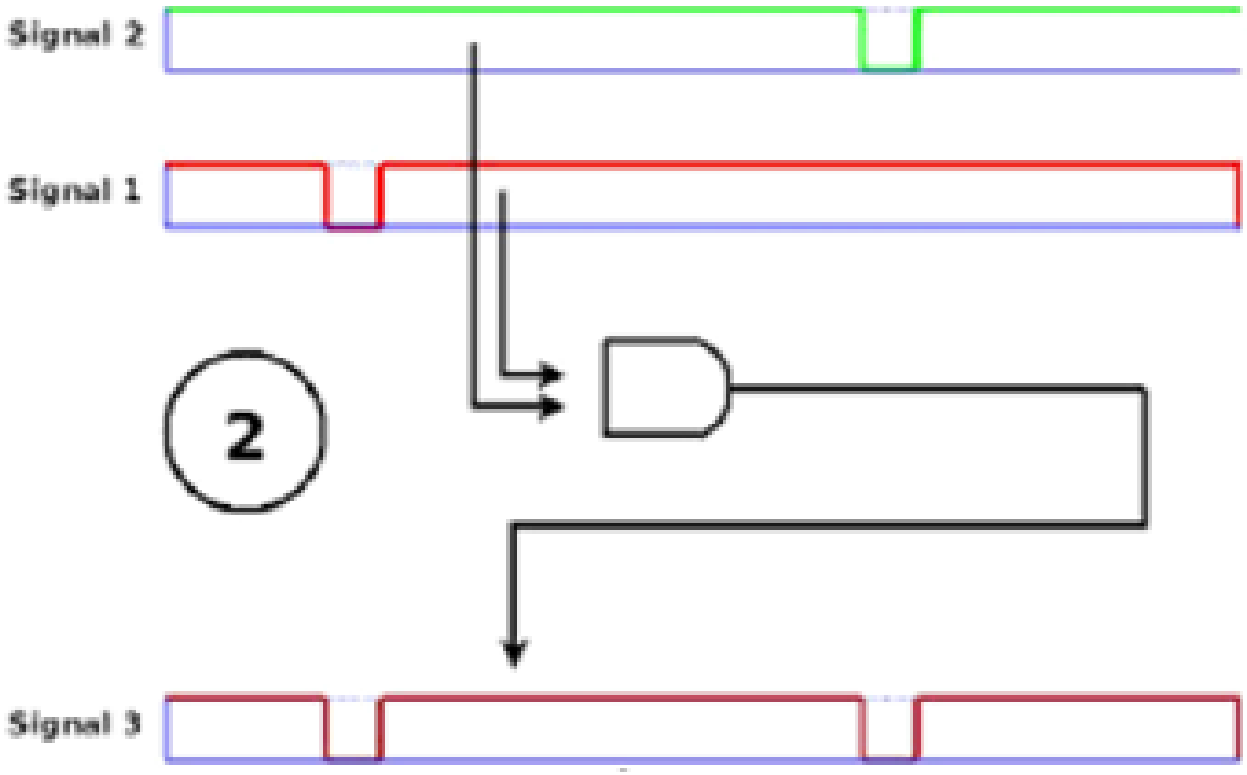
Flow diagram for the Task

- Step 1: Extraction of signals from sensors.



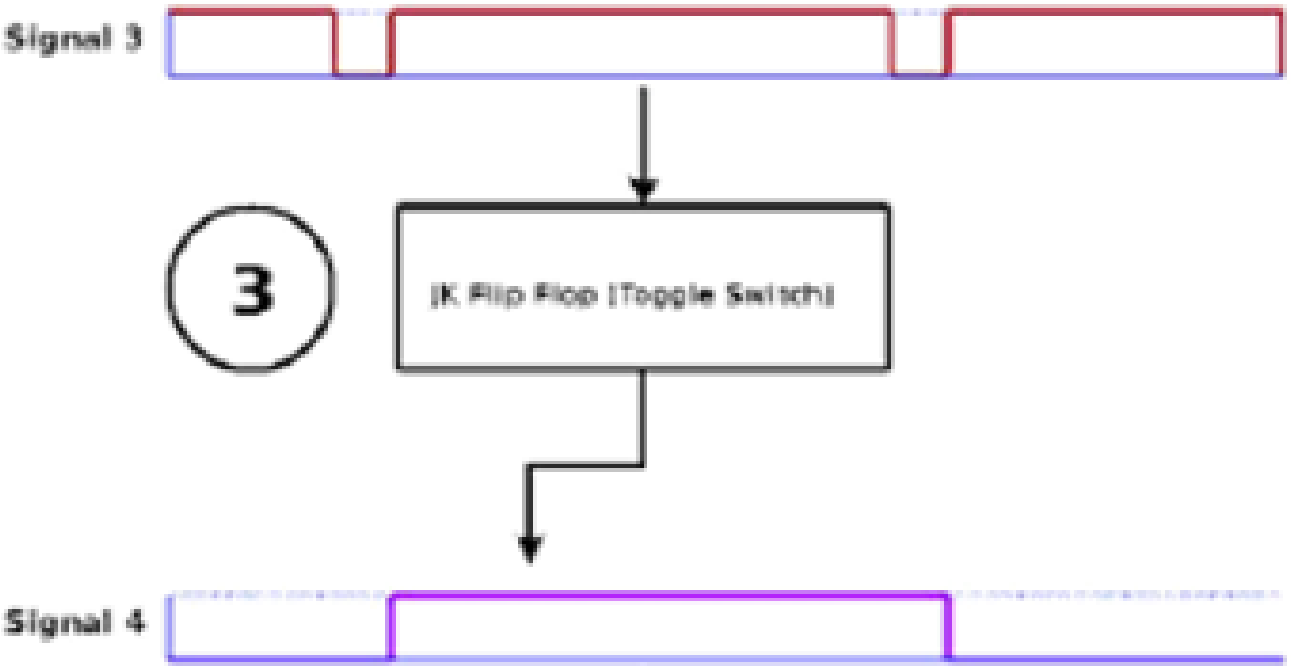
Flow diagram for the Task

- Step 2: Combining two sensed signals into one.



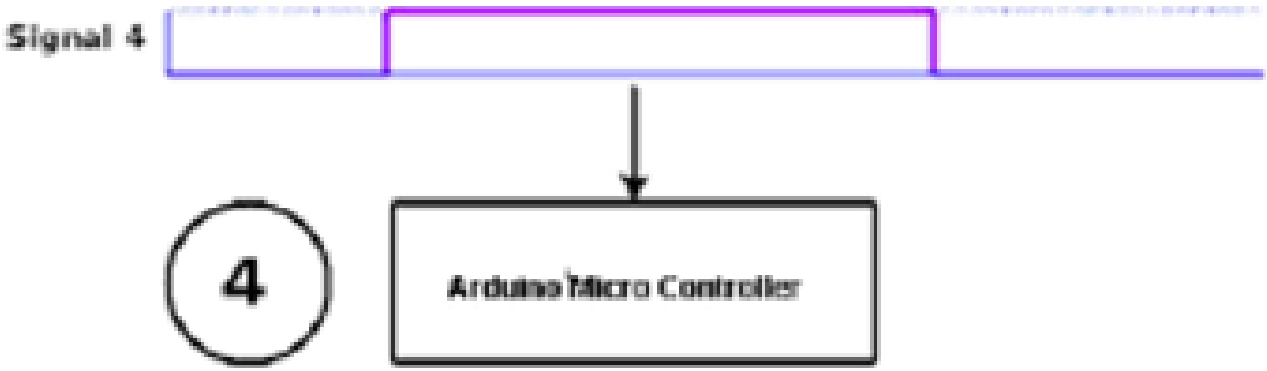
Flow diagram for the Task

- Step 3: Generate long pulse using JK flip-flop.



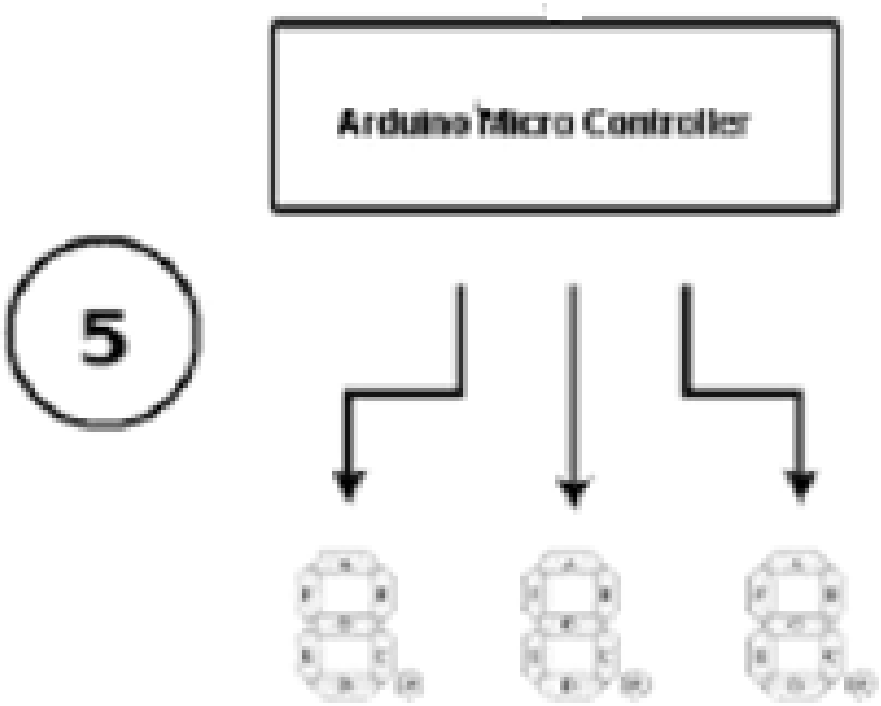
Flow diagram for the Task

- Step 4: Measure the pulse width using Arduino.



Flow diagram for the Task

- Step 5: Display the measured pulse width on 7-segment display.



Objective3: Computation of Angular Velocity

- Angular velocity (w) = θ/t
  where θ is angle in radian

- The value of $\theta = (\pi/180)x45 = 0.785$

- The angle 45^0 is the angle between two sensors with respect to the centroid.

- The display procedure will be same as that of time.

Cont…

- Two subroutine will be created.

- One subroutine for displaying the time and other for the angular velocity.

Conclusion

- View the Mangonel project

Thanks