Ultrasound Pen and Tracking System

Prithvi Bhat - 1002033598

pnb3598@mavs.uta.edu.in | +1 (682) 340-1126

Introduction

The aim of the project is to capture the position of a stylus on any two-dimensional Cartesian plane and obtain its coordinates in software. This project primarily uses the principles of ultrasound waves and their propagation through air to determine the coordinates of the stylus. The project also uses peripherals such as Wide timers, PWMs, I2C and GPIOs in the TM4C123GH6PM microcontroller on the TI Tiva C launchpad board.

Hardware

The project implements included the development of the Stylus and the Receiver circuit

Stylus

The Stylus is a simple electronic circuit enclosed in a 3D-printed case. Electronics include

1. IC: 556 dual timer
2. Resistors: 1.8kΩ, 1.5kΩ, 470Ω, 33Ω
3. Diodes: IN914
4. Capacitors: 1μF, 0.1μF
5. IR emitter
6. Ultrasound emitter
7. Supply: 9V DC
8. Switch

Schematic

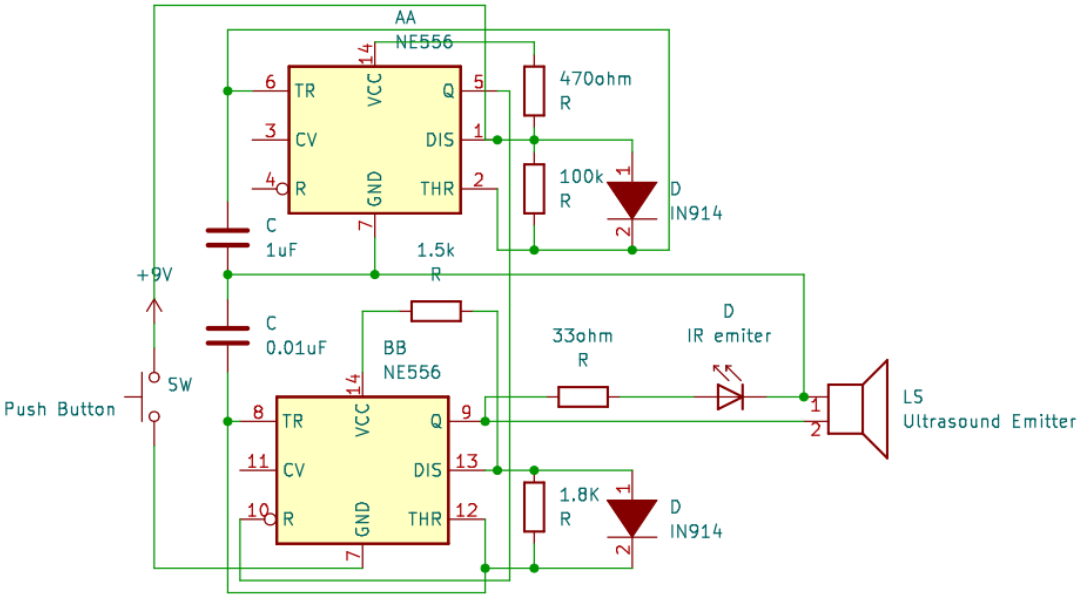


Figure 1: Schematic of Stylus

Figure 2a: Stylus

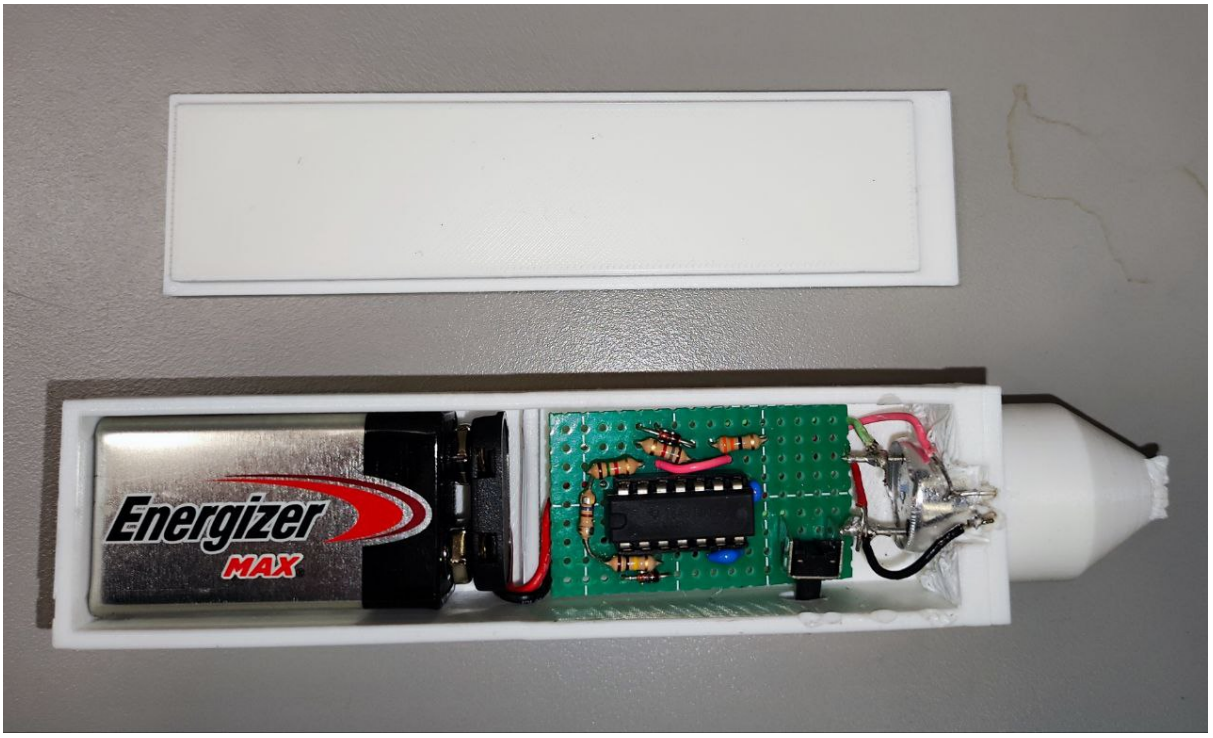


Figure 2b: Stylus Circuit

Receiver

The receiver circuit consists of the following components:

1. Sensors: Ultrasound Receivers, IR Receiver
2. IC: TLC072, MCP6544, RedBoard
3. Resistors: 470Ω, 4.7kΩ, 10kΩ, 47kΩ, 470kΩ
4. Capacitors: 1μF, 0.1μF
5. Diodes: IN914
6. Buzzer

Block Diagram

Figure 3 shows the block diagram of the receiver circuit

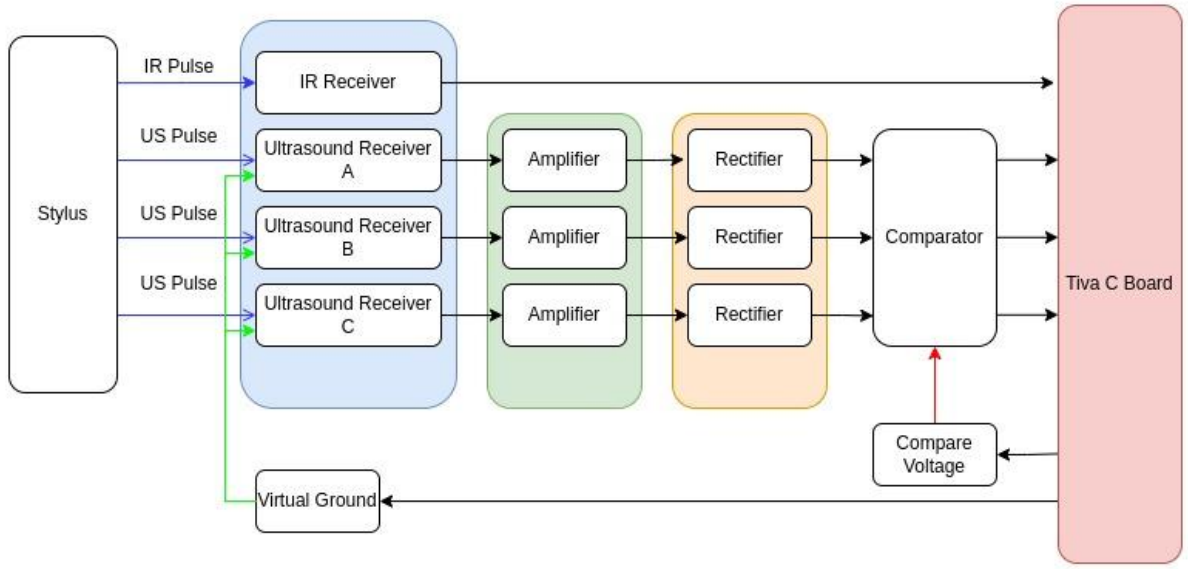


Figure 3: Receiver circuit block diagram

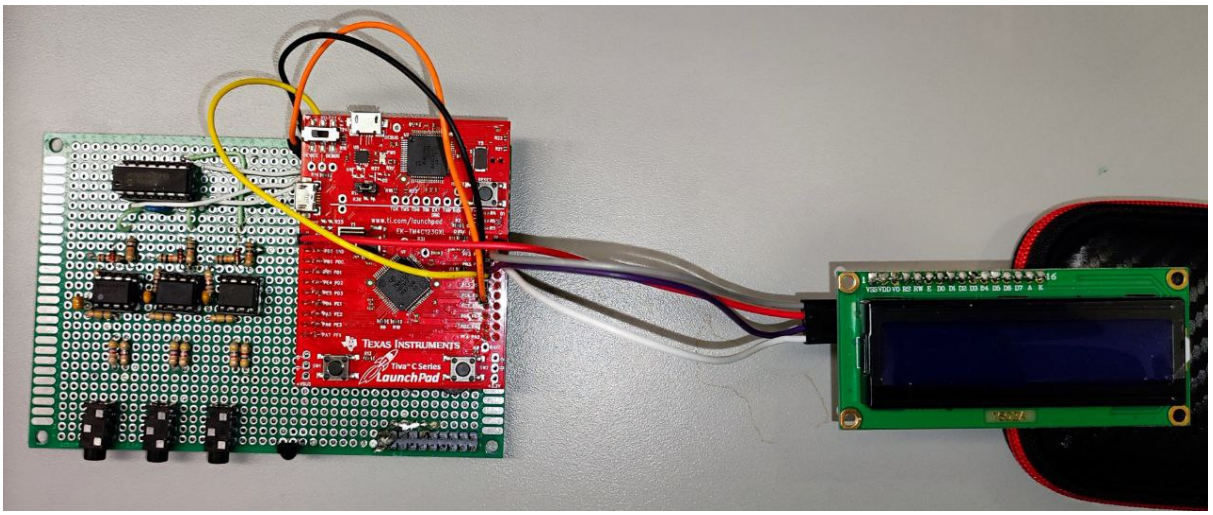


Figure 4: Receiver Circuit

Operation

When the user presses a button the Stylus, an IR pulse and a train of Ultrasound pulses are emitted. The IR pulse (an electromagnetic wave) reaches the receiver circuit first and thereby, triggers a falling edge interrupt on `IR_IN`.

The interrupt service routine of the IR pulse, `ir_interrupt_handler`, will start three wide timers - WTIMER0A, WTIMER0B and WTIMER1A, each configured as shown in Table 1.

Register	Value	Function
WTIMERn_CTL	`0x00000004`	Trigger interrupt on negative edge
WTIMERn_TxMR	`0x00000010`	Set count direction to UP
WTIMERn_TxMR	`0x00000004`	Configure as Edge timer
WTIMERn_TxMR	`0x00000003`	Configure for Capture Mode
WTIMERn_CFG	`0x00000004`	Configure as 32 bit wide timer
WTIMERn_IMR	`0x00000004`	Configure to capture on event

Table 1: Timer Configuration

When the Ultrasound receivers detect an input, timer interrupts are triggered and will be handled by their respective service routines - `sA_interrupt_handler`, `sB_interrupt_handler`, `sC_interrupt_handler`. Within each of the timer interrupt handlers, the timer register value at the instance of interrupt trigger is read and stored in an array - `g_timer_n_FIFO`, where n denotes the respective array for each timer-receiver pair. Additionally, the ISR also resets the timer value register, clears the interrupt flag, and sets one flag indicating the reception of an Ultrasound signal.

The above steps are elucidated in Figure 5.

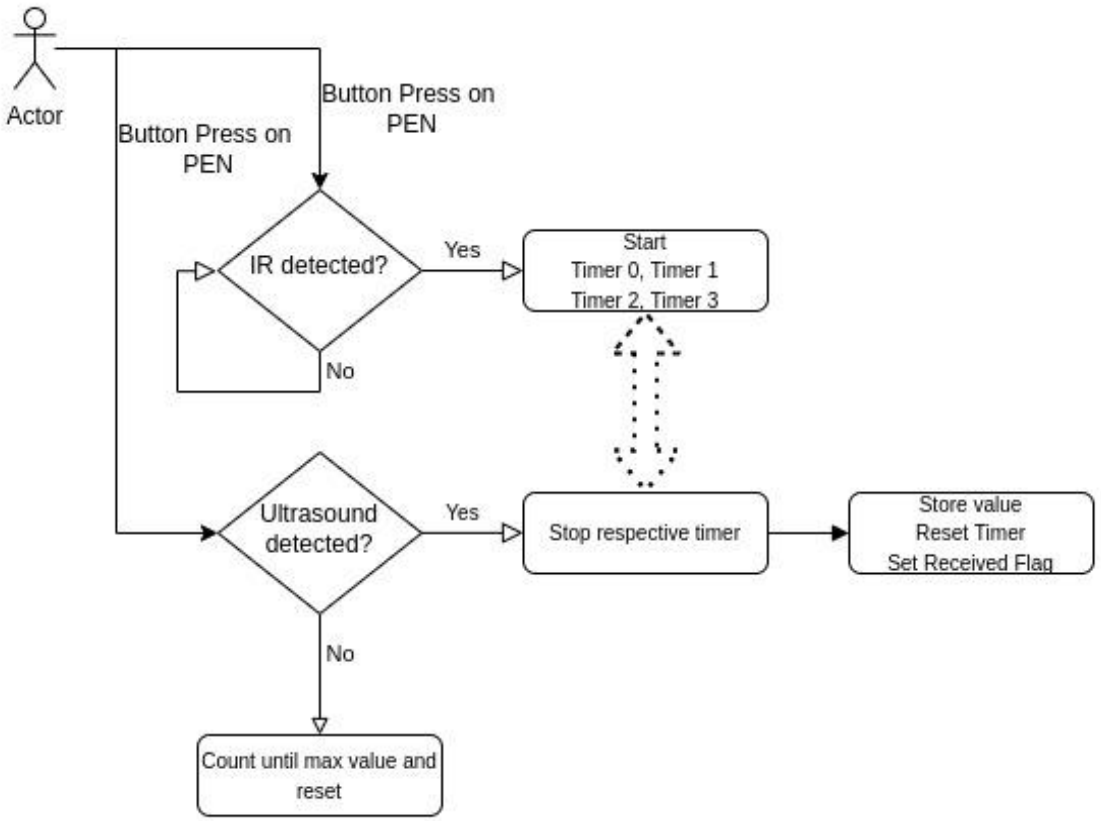


Figure 5: Signal detection and Interrupt handling

Triangulation

The raw timer values in the `g_timer_n_FIFO` arrays are converted to manipulable distance (metric, mm) values by the following formula:

```
((RAW_TIMER_VALUE  / (40e6)) * 1000 * 343)
```

Where,
40e6 = System clock speed
1000 = Convert to mm
343 = Speed of sound in air at standard temperature and pressure

The distance so calculated shall be the distance of the source of the ultrasound (position of the Stylus) from the Ultrasound sensors.

From Figure 6, applying principles of overlapping triangles and solving for linear equation of two variables, we obtain the coordinates x and y of the Stylus as:

```
y = (int32_t)(((g_distance_B * g_distance_B) - (g_distance_A * g_distance_A) + (D1 * D1)) / (2 * D2));
x = (int32_t)(((g_distance_B * g_distance_B) - (g_distance_C * g_distance_C) + (D2 * D2)) / (2 * D1));
```

The coordinates of the three sensors shall be configured by the user and stored in the EEPROM.

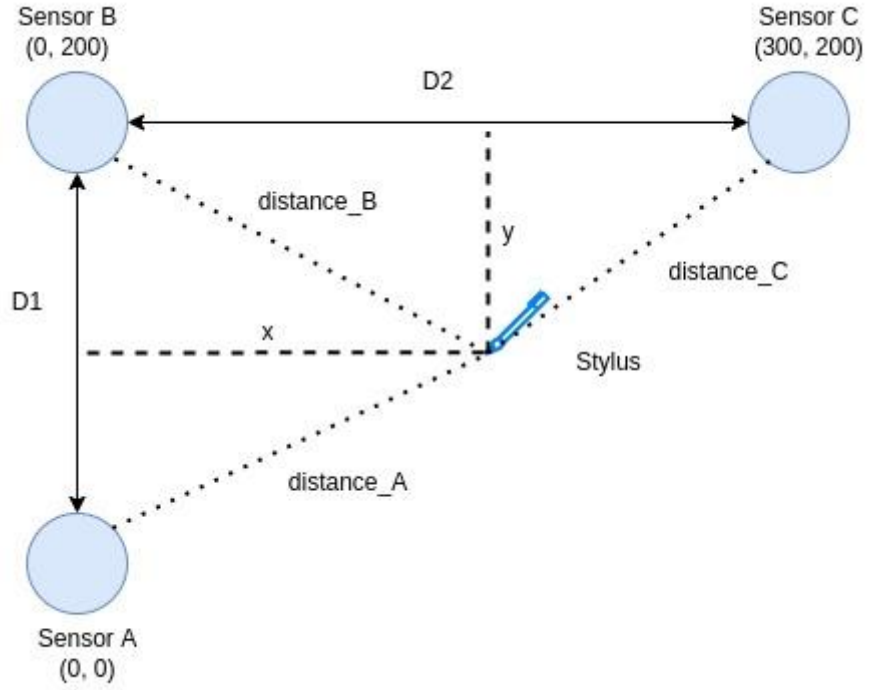


Figure 6: Triangulation

General Configurations

Name	Port	Pin	Direction	Function
IR_IN	PORTA	06	IN	Input from IR pulse
US_A_IN	PORTC	04	IN	Input from Ultrasound signal sensor A
US_B_IN	PORTC	05	IN	Input from Ultrasound signal sensor B
US_C_IN	PORTC	06	IN	Input from Ultrasound signal sensor C
BUZZ_OUT	PORTD	01	OUT	Output to Buzzer
I2C0SCL	PORTB	02	OUT	I2C Clock to Character Display
I2C0SDA	PORTB	03	OUT	I2C Data to Character Display

Table 2: GPIO Pins Configuration

Register	Value	Function
PWM1_0_GENB Register	0x00000800	Drive pwmB Low
PWM1_0_GENB Register	0x0000000C	Drive pwmB High
PWM1_0_LOAD Register	value	Load value for PWM period
PWM1_0_CMPB Register	Value / 2	Load Value for Duty Cycle
PWM1_0_CTL Register	0x00000001	Enable PWM Block
PWM1_ENABLE Register	0x00000002	Enable PWM Output

Table 3: PWM Configurations

Outputs

Table 4 lists the supported commands in software

Command	Operation	Output*

sensor	Update sensor coordinates	`sensor, A, 0, 0`
		`Sensor coordinates updated in EEPROM`
		`Assuming input coordinates are in mm`

beep	Update beep periods and tones	`beep, 1, 10000, 10000, 1000, 3` `Beep tones updated`
reset	Software reset the MCU system	
average	Input the number of samples to be averaged	`average, 1` `Averager updated`

distance	Calculate and print distance of stylus from each sensor	`distance`
		`Distance from Sensor A: 0mm`
		`Distance from Sensor B: 0mm`
		`Distance from Sensor C: 0mm`

variance	Calculate and print variance of readings	`variance`
		`Variance of Sensor A readings = 0.000000`
		`Variance of Sensor B readings = 0.000000`
		`Variance of Sensor C readings = 0.000000`

fix	Update the correction factor for calculated coordinates	`fix, 0, 20` `Fix values updated`

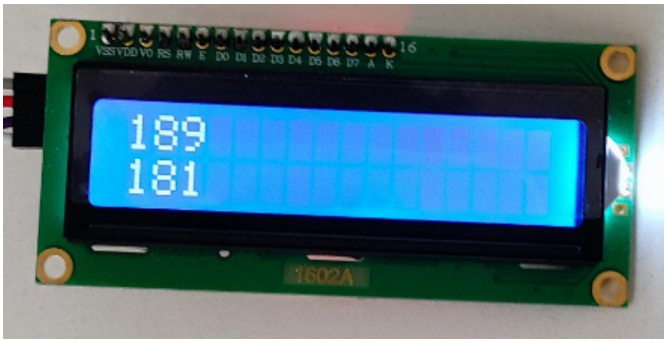
coord	Calculate and display the coordinates of the sensor	`coord`
		`(x,y):(-20mm, 0mm)`
		 

Table 4: Commands and their sample outputs

*** Screengrabs of outputs without stylus input**