Name: Balkirat Singh Nagra

SIT315 – Programming Paradigms

Task – M1T1D

Reflection Report : Extensive Interrupt based Sense – Think – Act system.

The embedded system for this project uses an Arduino Uno as the control unit and implements a total of three types of sensors (PIR motion sensor, IR obstacle sensor, Ultrasonic distance sensor) based on the Sense-Think-Act framework (i.e., the microcontroller senses events occurring in the real world and then thinks and decides based on input from the sensors and executes commands to generate actions in the real world). In addition, this project is targeted at demonstrating the concurrent handling of both event-driven types of behaviour (using hardware interrupts) and time-driven types of behaviour.

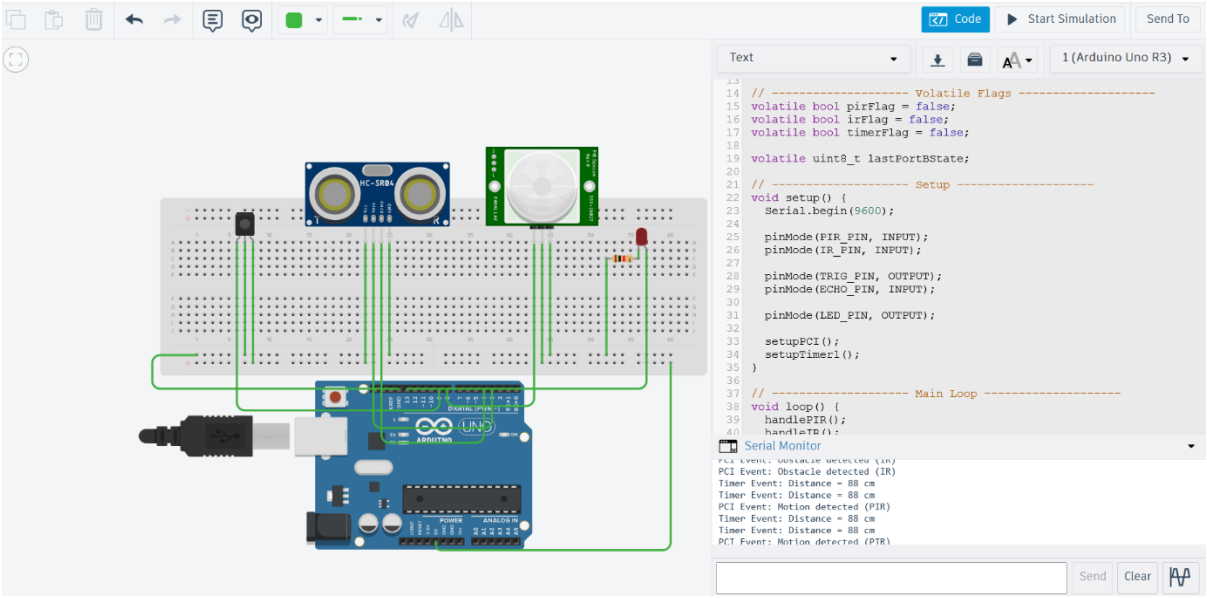
With regard to hardware interrupts, the Arduino Uno divides the system architecture into two primary interrupt types: pin change interrupt (PCI) hardware and timer1 interrupt (i.e., the two types of interrupts are combined). The PCI hardware has been configured to monitor two pins from the same port group (PORTB – D8 and D9) which are connected to the PIR and IR sensors. Thus, once either the PIR sensor detects motion and/or the IR sensor detects an obstacle, an associated PCI Interrupt Service Routine (ISR) will be called. The system does not use the ISR to perform long processing operations; rather it simply sets some volatile state flags in order to use interrupt safe coding practices and avoids blocking behaviour.

The Timer 1 interrupt is designed to provide periodic interrupts, separate from the sensor events. This is an example of time-based logic (e.g. periodic status logging or LED flashing). This is done without relying on polling which repeatedly checks the status of a sensor, so asynchronous events can be dealt with while still providing consistent time-dependent operations. By keeping the event-driven (PCI) logic independent from the time driven (Timer 1) logic, the two can run at the same time without interfering with one another.

The state is handled using global volatile flags that are shared between ISRs and the main loop. The ISRs have intentionally been kept minimal so they do not cause delays or unsafe operations, like Serial printing. The main loop is considered to be the 'think layer' of the architecture. It reads the state flags, processes the appropriate logic, and causes the correct

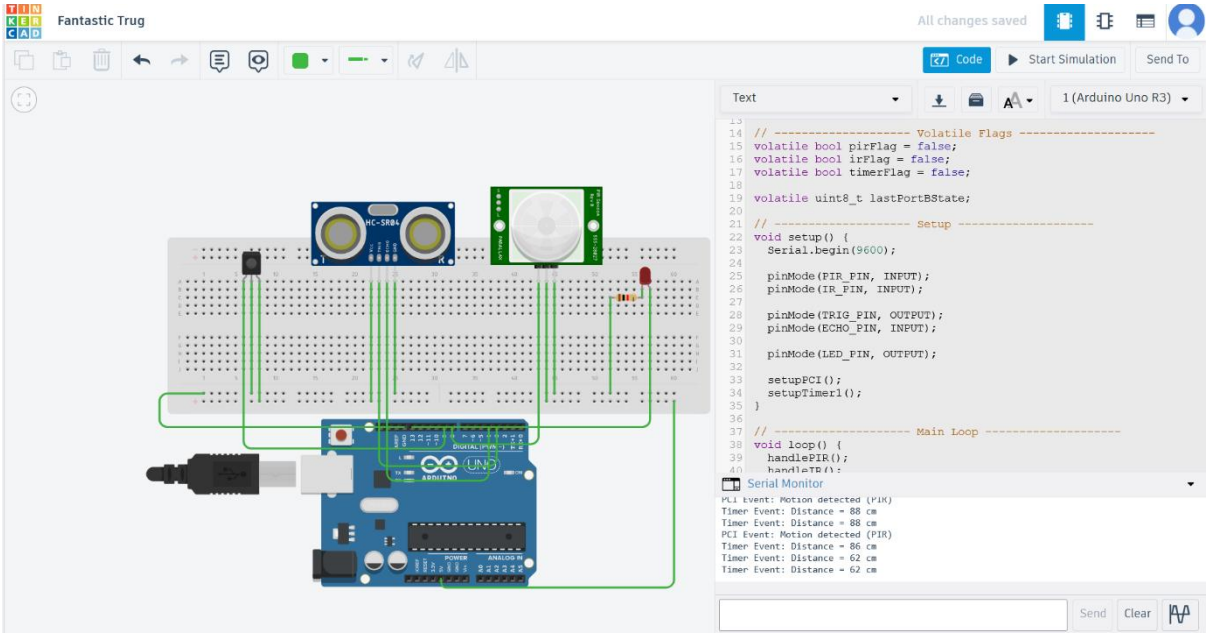
actions to happen. This architecture separates the sensing (interrupts), thinking (decision making within the loop), and acting (the output event).

On working both events showed confirmation by serial monitor outputs for both pin change interrupt (PCI) and timer 1 interrupt. When motion was detected by PIR sensor, PCI events were triggered instantaneously with obstacle being detected by IR. The Timer1 interrupt continued executing independently of these PCI events by remaining within its task period. The ultrasonic distance measurements were also verified by two Serial Monitor screen shots showing a difference in the measured distance of an object being brought closer to and farther away from the ultrasonic sensor. These screen shots provide evidence that the system correctly identifies changes in its environment while continuing to execute timer-based logic reliably.



This system clearly states that an event-based logic system (the PCI handling asynchronous events) is separated from a time-based logic system (Timer1 interrupt handles periodic tasks without blocking the main program's flow). The PCI interrupts being processed through Interrupt Service Routines (ISR) only updated volatile state flags, whereas all processing & decision-making occurred within the main loop to ensure state management safety and avoid race conditions. The separation of event-based and time-based logic enhances the responsiveness & reliability versus traditional designs that relied on a polling-type

architecture because sensor events were captured immediately without delaying any periodic tasks.



Overall, the project shows a clean interrupt – driven embedded system that efficiently manages both concurrent events and regular tasks, meeting the required status of a thorough hardware based sense – think – act system.