Reflection Report

The system I created focuses on reliable concurrent processing of time- and event-based logic by implementing a Sense–Think–Act architecture on an Arduino Uno platform. The setup includes three pushbuttons for input sensors, a green LED serving as a visual indicator, and a buzzer acting as an auditory output device.

The three pushbuttons I have included work for different input situations,

Button 1 and Button 2 belong to PCI group PCINT0_vect and are connected to digital pins D8 and D9.

In order to replicate a normal input check, button 3 is polled in the main loop and attached to D4.

In my system all button presses cause the buzzer to sound through a shared modular function, and the system uses a Timer1 interrupt to turn the LED every second. To prevent blocking or potentially harmful operations inside interrupts, logical separation is maintained through the use of volatile flags (button1Pressed, button2Pressed, and timerFlag) that are set inside the corresponding ISRs and evaluated in the loop() function. This guarantees a distinct separation between time-based periodic actions and event-based input detection. The Serial Monitor provides traceable diagnostics for input state changes and time events in real time by printing all user interactions and system responses.

By using Pin Change Interrupts for digital pins D8 and D9, which are members of the same interrupt vector group (PCINT0), the interrupt configuration conforms with best practices in embedded architecture. This satisfies the need for grouped pin interrupts by enabling the monitoring of numerous digital inputs with a single ISR (ISR(PCINT0_vect)). I setup the PCI by using PCICR to enable global PCI, then I set particular mask bits for PCMSK0 for PCINT0 and PCINT1, and these two responds to the digital pin 8 and digital pin 9. In order to identify rising edges and simulate a push, the ISR reads the digital pin states and compares them to their prior values. Timer1 was setup to operate in CTC (Clear Timer on Compare) mode for time management, using a 1024 prescaler and an OCR1A value of 15624, resulting in a 1 Hz interrupt. I set up a timerFlag, inside ISR(TIMER1_COMPA_vect). This flag generates a status message to the Serial Monitor and switches the LED in the main loop. The ISRs does not make use of any delay() or serial functions and are purposefully kept to a minimum. As an alternative, they set state flags that are handled in the loop() to control logging and actuator behaviour (buzz()).

One of the main issues I faced was improper wiring causing inconsistent button behaviour. Pushbuttons in TinkerCad internally short vertical pairs (such 1A–1B), but I had linked pins incorrectly on the same side and I had not included resistors. Because of this, digital pins began to float, producing HIGH readings at random even when not a single button was pressed. To fix this issue I rewired the pushbuttons and connected 10kΩ pull-down resistors to the input digital pins and GND to make the default low state stable. Another issue I encountered was me using D2 and D3, firstly along with attachInterrupt(). However, PCI utilisation on D8–D13 was specifically required by the criterion. To fix this issue I had to manually configure the interrupt registers and swap those out for PCI-capable pins.