**STM32- BASED CHESS CLOCK WITH CUSTOM FUNCTIONALITIES**

**TEAM MEMBERS:**

**SUJAN KARTHIK-CB.EN.U4CSE22547**

**SANJAY PK-CB.EN.U4CSE22542**

**N SIDDHARTH- CB.EN.U4CSE22529**

**PRAYSUN- CB.EN.U4CSE22537**

**1. Introduction**

The Chess Clock Timer project aims to build an embedded system using the STM32 microcontroller that functions as a chess timer. The timer counts down from a specified duration, and the clock is managed using external inputs. The timer supports two main durations: 10 minutes (when the Bluetooth button "2" is pressed) and 5 minutes (when the Bluetooth button "3" is pressed). The project also includes a Bluetooth communication interface for controlling the timer remotely, along with a visual display using 7-segment displays and LEDs to indicate the timer status. The system is based on the STM32F4 microcontroller and uses various GPIO pins for input and output management.

**2. System Design**

2.1 Hardware Design

The hardware design for this chess clock timer consists of the following key components:

* STM32F4 Microcontroller: This is the central processing unit of the system, managing input, output, and timer functions.
* 7-Segment Displays: Two 7-segment displays are used to show the minutes and seconds of the countdown.
* LEDs (PC14, PC15): These LEDs indicate different states of the chess clock (e.g., active timer, paused state).
* Buttons (PA6, PB10, PC13): Three push buttons are used to control the system:
  + PA6: Toggles the timer.
  + PB10: Resets the timer to 5 minutes.
  + PC13: Pauses or resumes the timer.
* Bluetooth Module: A Bluetooth communication module connects to the STM32 via UART to remotely control the timer through button presses. The commands sent from a smartphone or other Bluetooth-enabled device toggle the timer durations (10 minutes or 5 minutes).

The clock duration is set using Bluetooth commands '2' for 10 minutes and '3' for 5 minutes. When the button on PA6 is pressed, the timer will start or stop, while PB10 resets the timer to its initial state.

**3. Code Structure**

The code is divided into several key functions and modules:

* USART1\_Init: Initializes the UART interface for Bluetooth communication.
* USART1\_Write and USART1\_Read: These functions are used for sending and receiving data via Bluetooth.
* display and display1/display2: These functions are responsible for displaying the current time on the 7-segment displays.
* SysTick\_Handler: This interrupt handler updates the timer and refreshes the display every 500 ms.
* EXTI9\_5\_IRQHandler and EXTI15\_10\_IRQHandler: These are interrupt service routines for handling button presses, toggling the timer state, and pausing the timer.
* delay: A simple delay function for debouncing the buttons.

The main function configures the GPIO pins, sets up the external interrupts, and manages the Bluetooth communication. It also continuously listens for commands to control the timer via Bluetooth.

**4. Functionality**

The system operates as a chess clock with the following functionalities:

* Start/Stop Timer: The timer starts when the user presses the PA6 button. The timer will stop when the button is pressed again.
* Reset Timer: Pressing the PB10 button resets the timer to 5 minutes.
* Pause/Resume Timer: The PC13 button toggles the pause state, which pauses and resumes the countdown.
* Bluetooth Control: The timer can also be controlled remotely via Bluetooth. The user can send a command via a Bluetooth-enabled device to:
  + Set the timer to 10 minutes by sending the character '2'.
  + Set the timer to 5 minutes by sending the character '3'.

When the timer is running, the 7-segment displays continuously update to show the remaining time in minutes and seconds.

**5. Challenges Faced**

5.1 Hardware Challenges

* Pin Configuration: One challenge was properly configuring the GPIO pins for the 7-segment displays. Since the STM32 microcontroller has a large number of GPIO pins, managing and configuring them correctly for display output was complex.
* Button Debouncing: Mechanical buttons can produce noise, resulting in multiple interrupts from a single press. We implemented a simple software debounce solution, but fine-tuning the debounce delay to avoid false triggering while ensuring responsiveness was a challenge.

5.2 Software Challenges

* Bluetooth Communication: Ensuring reliable communication between the STM32 microcontroller and the Bluetooth module was challenging. We had to account for different baud rates, handle buffer overflows, and implement proper UART communication protocols.
* Interrupt Handling: The interrupt system was crucial for managing button presses without blocking the main execution flow. Handling the multiple interrupt sources (buttons and timers) required careful management of priorities and flag handling.

**6. Solutions Implemented**

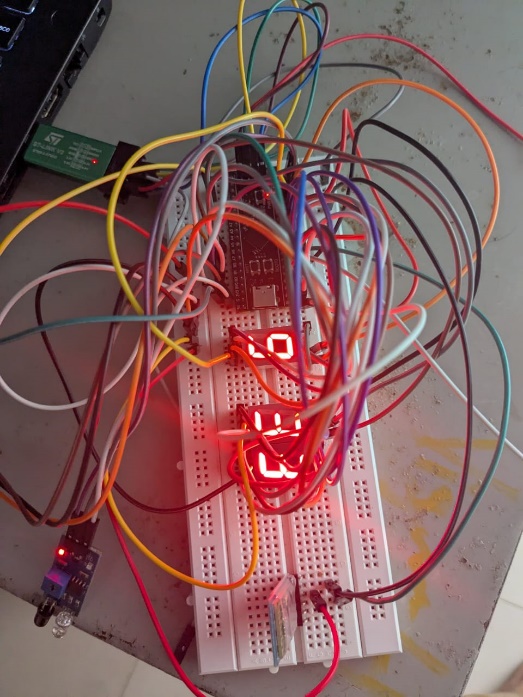
6.1 Hardware Solutions

* GPIO Configuration: The pins for the 7-segment displays were carefully configured as output, ensuring correct voltage levels and control for each segment of the displays.
* Button Debouncing: A simple delay-based debouncing method was used to ensure that button presses were registered correctly without false triggers.

6.2 Software Solutions

* UART Buffer Management: We ensured that the UART read and write functions were non-blocking, with proper checks to verify that the Bluetooth commands were received correctly.
* Interrupt Management: We implemented a non-blocking method for managing button presses, ensuring that each interrupt was handled quickly without blocking other tasks. The system could handle multiple buttons being pressed in succession, while the timer continued to update every 500 ms.
* Timer Control: The timer was updated based on the SysTick interrupt, and the logic ensured that the countdown occurred smoothly, including handling pauses and resets

**7. Image Demo**

****

**Conclusion**

The Chess Clock Timer project successfully demonstrated the capabilities of the STM32 microcontroller for handling time-sensitive tasks, external interrupts, and Bluetooth communication. Despite the challenges faced during the development, such as managing the interrupt system, handling the GPIO configuration for 7-segment displays, and ensuring reliable Bluetooth communication, the project achieved its goal of creating a fully functional chess clock timer with Bluetooth control. The system is capable of starting, stopping, resetting, and pausing the timer both via physical buttons and remotely via Bluetooth commands.

The project highlights the importance of effective hardware and software integration, careful handling of interrupts, and the need for debugging and fine-tuning to ensure reliable performance in embedded systems.