Modular Embedded Control System

Microcontroller-Based Display and Sensor Integration

1. Project Overview

This project involved the incremental development of a modular embedded system using the TCM4C123GH6PM microcontroller. Over multiple development phases, the system evolved from basic GPIO control to a fully integrated platform supporting multiple display devices, user-controlled operating modes, and real-time temperature sensing.

The final system supports several independent operating modes selected via the 8 DIP switches, including numeric counting on seven-segment displays, dynamic text scrolling on an LCD, and a sensor-driven override mode that displays real-time temperature reading from an external digital thermometer. The project emphasizes low-level hardware interaction, state-based logic control, and time-sensitive communication, reflecting real world embedded system design practices.

2. System Architecture

The embedded system platform consists of:

- TM4C123GH6PM microcontroller
- RGB LED (on board)
- Dual seven-segment displays
- Two-line character LCD
- DS1620 digital thermometer
- DIP switches for user input and mode selection
- PCF8574 GPIO expanders

User input and display peripherals including dual seven-segment displays, a two line LCD, and an 8 position DIP switch were interfaced using PCF8574 I/O expanders over the I2C bus. This approach significantly reduced direct GPIO usage on the microcontroller and enabled scalable integration of multiple subsystems while reserving GPIO resources for time-critical operations.

3. Incremental Development Timeline

The system was developed across six functional milestones, each adding new capabilities while preserving previous implementation behavior.

- **Phase 1 - GPIO Output Control:**
  - Initial control of the on-board RGB LED, including color cycling, reset behavior, and continuous switching modes.
- **Phase 2 - Numeric Display Integration:**
  - Integration of dual seven-segment displays capable of reading and displaying current switch position based on DIP switch input.
- **Phase 3 - Mode - Based Numeric Encoding:**
  - Added support of both decimal and hexadecimal counting using seven-segment displays.
  - User-selectable modes handle logic base and increment and decremental counting.
- **Phase 4 - LCD Display Integration:**
  - Implemented a two-line LCD interface displaying static text.
  - Default behavior displays the users first and last name on separate lines, with switch-based line swapping.
- **Phase 5 - Dynamic Text Scrolling:**

    - Expanded LCD functionality to support scrolling text across both lines, including bidirectional scrolling controlled by user input.
- **Phase 6 - Temperature Sensor Integration:**
    - Integrated a DS1620 digital thermometer using a software-based bit-banged serial protocol.
    - When enabled via switch input, the system software enters a sensor override mode that halts all other display functions and continuously displays temperature data.

4. **Final System Behavior:**

In its completed form, the system operates as a state-controlled embedded platform:

- Default behavior executes display and LED functions based on the current operation mode.
- DIP switches allow the user to change counting direction, numeric base, text behavior, and scrolling direction.
- Activating the temperature mode places the system into a mutually exclusive override state, suspending all other functions until the switch is cleared.

This structure ensured deterministic behavior and prevented resource conflicts between display subsystems.

5. **Sensor Integration Deep Dive (DS1620)**

The DS1620 digital thermometer communicates via a 3-wired serial protocol, which was implemented entirely in software using GPIO based bit-banging techniques. Custom routines were developed to manually generate clock pulses, transmitted command bytes, and read temperature data by bit by bit. Temperature data is received as a 9-bit two's complement value, parsed in software, and converted into both Celsius and Fahrenheit. Real-time values are formatted and displayed on the LCD, with observable response times under one second.

6. Challenges and Engineering Considerations

Key challenges addressed during development included:

- Managing precise timing requirements for software-driven serial communication
- Dynamic switching GPIO direction during bidirectional data transfer
- Preventing mode conflicts between independent subsystems
- Debugging state logic as system complexity increased

To address these challenges, the software was structured into modular user-defined functions, enabling targeted debugging and easier integration of new features.