Smart home Developed Using the PIC18F4620 Microcontroller

My name is Yassin Khadrawy, a student at the Communication and Electronics Department, Faculty of Engineering, Alexandria University.

Today, I am excited to present my latest project—a Smart home system developed using the PIC18F4620 microcontroller.

This project leverages various peripherals of the PIC18F4620 microcontroller to build a functional and interactive Smart home system. Key features include controlling devices, monitoring environmental parameters, and enhancing home security. The project utilizes multiple microcontroller peripherals such as I2C, timers, PWM, ADC, and GPIO.

System Components

The components integrated into this project are:

1. **Microcontrollers:**
   - Two PIC18F4620 microcontrollers: one configured as the master and the other as the slave.
2. **Input Devices:**
   - **Keypad:** For user input and control.
   - **Buttons:** For interaction and control of various devices.
3. **Output Devices:**
   - **DC Motors:**
     - Used as a locker.
     - Used as a fan.
   - **Heater:** For temperature regulation.
   - **Speaker:** For audio notifications.
   - **7-Segment Display:** Common anode configuration, driven by a 7447 IC, to display numeric data.
   - **LCD:** To display information such as time, date, or sensor readings.
   - **LEDs:** For visual indications.
4. **Sensors:**
   - **DS1307 RTC (Real-Time Clock):** Used for time and date functionality via I2C communication.
   - **TC74:** To measure ambient temperature using I2C.
   - **Gas Sensor:** To detect harmful gases and ensure safety.
   - **PIR Sensor:** To detect motion for security purposes.
   - **Pressure Sensor:** To monitor atmospheric pressure.
   - **LDR (Light Dependent Resistor):** To measure ambient light intensity.
5. **Additional Components:**
   - **Motor Driver (L298):** To control the DC motors.
   - **Relay Module:** To control high-power devices.
   - **Resistors:** Part of the supporting circuitry.
   - **I2C Debugger:** For troubleshooting and debugging I2C communication.

System Overview

The system consists of two interconnected MCUs—one deployed outdoors and the other indoors—each managing specific components to ensure seamless operation and enhanced functionality.

Outdoor Unit

The **Master MCU** (PIC18F4620) manages components located outside the home. Its responsibilities include security, access control, and environmental sensing. The key components and their functions are:

- **LDR**: Controls an outdoor LED based on ambient light levels.
- **Motor and Motor Driver**: Operates as a locking mechanism for home security.
- **Speaker**: Alerts in case of intrusions, activated by the PIR sensor.
- **Push Buttons**: Used for entering and confirming the password for entry and exit.
- **PIR Sensor**: Detects motion for security purposes.
- **LCD**: Displays the entered password during access operations.
- **Keypad**: Allows users to input the password.
- **TC74 Temperature Sensor**: Measures ambient temperature.
- **DS1307 RTC**: Provides real-time clock and date functionality.

Indoor Unit

The **Slave MCU** (PIC18F5620) handles components inside the home, focusing on environmental monitoring, safety, and climate control. The key components and their functions are:

- **Push Button**: Controls an indoor LED for basic lighting.
- **Gas Sensor**: Detects harmful gases; triggers a speaker and LED alarm if gas is present.
- **LCD**: Displays time, date, temperature, and atmospheric pressure.
- **7-Segment Display (Common Anode with 7447 IC)**: Displays the time.
- **Fan (connected to relay)**: Operates as a cooling system.
- **Heater (connected to relay)**: Maintains a comfortable indoor temperature.

Communication

The two MCUs are connected via the **I2C** protocol, allowing the master to send sensor data and commands to the slave for coordinated operation of indoor and outdoor functionalities.

Timer0 Configuration on PIC18F4620

Timer0 is utilized to accurately calculate time intervals, specifically to count seconds. In each loop, the timer increments by one second. To achieve precise timekeeping, a Timer0 calculator was used to determine the optimal timer settings:

- **Prescaler**: 1:32
- **TMR0 Preload**: 3036
- **Actual Interrupt Time**: 1 second
- **Timer Mode**: 16-bit mode

The **Prescaler** of 1:32 slows down the timer, enabling it to increment in alignment with real-world time. The **TMR0 Preload** value of 3036 ensures the timer generates an interrupt at precisely one-second intervals.

This configuration ensures the smart home maintains accurate time by incrementing each second through the timer's interrupt mechanism.

Timer1 Configuration on PIC18F4620

Timer1 is used to calculate time intervals with precision, specifically to count seconds. At each loop, the timer increments by one second. A Timer1 calculator was utilized to determine the optimal settings for accurate timekeeping:

- **Prescaler**: 1:8
- **TMR1 Preload**: 55536
- **Actual Interrupt Time**: 100 ms
- **Timer Mode**: 16-bit mode

The **Prescaler** of 1:8 slows down the timer, aligning its operation with real-world time. The **TMR1 Preload** value of 55536 ensures that the timer generates interrupts at precise intervals, allowing accurate counting of seconds.

This configuration guarantees the smart home maintains precise time by incrementing seconds based on the timer's interrupt mechanism.

Timer2 Configuration on PIC18F4620

Timer2 is configured for precise time interval calculations, with a specific focus on supporting **Pulse Width Modulation (PWM)** functionality. The settings were determined using a Timer2 calculator for optimal performance:

- **Prescaler**: 1:1
- **Postscaler**: 1:1
- **TMR2 Preload**: 0

The **Prescaler** of 1:1 ensures the timer operates without additional scaling, allowing for fine control. The **Postscaler** of 1:1 ensures no additional division occurs after the timer completes its cycle. The **TMR2 Preload** value of 0 initializes the timer for use with the PWM module, enabling precise duty cycle and frequency adjustments.

This configuration is essential for achieving accurate **PWM** control, which is critical for various functionalities, such as driving **LEDs**, **motors**, or other components in the smart home.

GPIO (General Purpose Input/Output) on PIC18F4620

In this project, the **GPIO (General Purpose Input/Output)** peripheral is utilized to initialize and configure the microcontroller's pins as either input or output. GPIO serves as the foundation for interfacing with various sensors, modules, and other hardware components.

The GPIO functionality relies on three primary registers:

1. **TRIS Register**: Determines whether a pin is configured as an input or output.
2. **PORT Register**: Reads the voltage level on a pin (5V or 0V).
3. **LAT Register**: Writes the desired voltage level (5V or 0V) to a pin.

- **Direct Uses**:
  - Configuring **buttons**, **LEDs**, and **relay** as input or output pins.
- **Indirect Uses**:
  - Supporting peripheral drivers such as **ADC**, **PWM**, and **I2C** through proper GPIO initialization.

GPIO is also used in the ECU (Electronic Control Unit) layer to configure and operate components and peripherals. By accurately setting up the pins, GPIO ensures seamless communication and functionality across the system, enabling efficient interaction with sensors, actuators, and other modules.

ADC (Analog-to-Digital Converter) on PIC18F4620

The **Analog-to-Digital Converter (ADC)** is used in this project to convert analog signals from sensors into digital values, enabling the microcontroller to process and utilize the sensor data effectively. This ensures accurate readings from the sensors.

- **Number of Channels Used**: 2 analog channels
  - **AN0 Slave MCU**: Connected to the Pressure Sensor to measure atmospheric pressure.
  - **AN0 Master MCU**: Connected to the LDR (Light Dependent Resistor) to detect light intensity.

The ADC plays a vital role in interfacing these sensors with the microcontroller, ensuring that the analog signals are accurately converted into digital data for processing within the application.

I2C Configuration and Usage

The **I2C (Inter-Integrated Circuit)** module is configured to operate in **Master Mode** for reliable communication with peripheral devices. It enables seamless data transfer between the master MCU and connected components in the smartwatch system.

- Operates in **I2C Master Mode** with a defined clock speed of **100 kHz**, ensuring efficient communication.
- Supports communication with multiple devices, including:
  - **TC74 Temperature Sensor**: Receives ambient temperature data.
  - **DS1307 Real-Time Clock (RTC)**: Retrieves time and date information.
- Transmits collected data to the **Slave MCU** for additional processing.
- Configured with standard I2C settings, including disabled **SMBus control and slew rate.**

The I2C module acts as the backbone for communication between the microcontroller and its peripherals. This setup ensures accurate data exchange and synchronization between the temperature sensor, real-time clock, and slave MCU, contributing to the smartwatch's enhanced functionality.

This concludes the documentation of my latest project, **Smart Home**. I hope this documentation helps you understand the various components, peripherals, and functionality of the project. Thank you for your interest and time in exploring my work!

Eng: **Yassin Khadrawy**