

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | | | | |
|  | | | Project Documentation | | | | | | |  | | |
|  | |  | | | | | | | | |  | |
|  | | | | |  | | |  | | | | |
|  | | | | | ULTRASOUND WATER LEVEL | | |  | | | | |
|  | | | | | August 1st, 2023—Ahmed Hadžiomerović | | |  | | | | |
|  | | | | Ultrasound Water Level – Project Documentation | | | | |  | | | |
|  | | | |  | |  |  | |  | | | |
|  | INTRODUCTION Welcome to the comprehensive documentation for the Arduino code that drives a sophisticated water level monitoring and display system. This document aims to provide a detailed understanding of the code's architecture, functions, and variables, facilitating effective utilization, maintenance, and further development of the system.  Water level monitoring is a crucial aspect of various applications, ranging from industrial processes to domestic systems. The presented code integrates an ultrasonic sensor with LED indicators to create a dynamic system that accurately measures water levels and provides visual feedback. The system's design caters to both accurate measurement and user-friendly visualization.  This documentation is structured to guide you through every aspect of the code. It will familiarize you with the global variables, setup procedure, utility functions, and the main loop's functionality. By the end of this document, you will be well-equipped to comprehend, modify, and extend the code to suit specific requirements.  The code leverages the capabilities of the Arduino platform, including digital and analog pins, EEPROM memory, and serial communication for debugging. The careful arrangement of functions ensures efficient execution and clear separation of concerns, promoting maintainability and scalability.  Thank you for choosing this documentation as your guide. Let's embark on a journey of discovery, learning, and exploration as we uncover the depths of the Arduino code powering the water level monitoring and display system. | | | | | | | | | | |  |
|  | GLOBAL VARIABLES  * **‘int baseLevel’**: Stores the sensor distance when there is no water. * **‘int levelBreak’**: Represents the distance threshold for changing water levels. * **‘enum Level { off, red1, red2, yellow1, yellow2, green1, green2 }’**: Defines water levels and their corresponding LED pins. * **‘int waterLevel’**: Stores the current water level. * **‘Switch Pins’**: Define pins for reset and water loss switches. * **‘Sensor Pins’**: Define pins for the ultrasonic sensor. * **‘int distance’**: Stores the measured distance from the ultrasonic sensor. * **‘int sensorMax’**: Defines the maximum measurable distance of the sensor. * **‘bool waterLoss’**: Indicates the presence of water loss. | | | | | | | | | | |  |
|  | SETUP FUNCTION The **‘setup()’** function plays a pivotal role in initializing the water level monitoring and display system. During this phase, crucial configurations are set, pins are initialized, and essential settings are retrieved from non-volatile EEPROM memory. Additionally, the serial communication interface is established to enable debugging and informative outputs.  The setup procedure ensures that the system is in a stable and coherent state before transitioning to the main operational loop. This function is executed only once when the Arduino board is powered on or reset.  The setup function prepares the system for operation by configuring pins, retrieving essential settings, and initializing the serial communication interface. Once this function completes its tasks, the system transitions to the main loop, where the core monitoring and display logic takes place. | | | | | | | | | | |  |
|  | UTILITY FUNCTIONS **‘int getLevelBreak()’**: Reads the levelBreak value from EEPROM memory.  **‘int reset()’**: Resets the system by recalibrating the base level, updating levelBreak, and resetting LED indicators.  **‘void resetLedControl()’**: Blinks LED indicators during system reset.  **‘int provjeraVode()’**: Reads the ultrasonic sensor and calculates the water distance.  **‘Level nivoVode(int distance)’**: Determines the water level based on the calculated distance.  **‘void ledcontrol(Level nivoVode)’**: Controls LED indicators according to the water level.  **‘void getBaseLossLevel()’**: Stores the base water level for water loss detection.  **‘void resetBaseLossLevel()’**: Resets the stored base water level.  **‘int readBaseLossLevel()’**: Reads the stored base water level from EEPROM memory.  **‘bool waterLossCheck(int baseLossLevel)’**: Checks for water loss by comparing the current water level with the stored base level.  **‘void waterLossLED(bool waterLoss)’**: Controls LED indicators to display water loss status. | | | | | | | | | | |  |
|  | MAIN LOOP The heart of the water level monitoring and display system lies within the **‘loop()’** function. This function is repeatedly executed after the **‘setup()’** function completes its initialization tasks. It orchestrates the system's continuous operation, constantly checking inputs, updating outputs, and ensuring responsive interaction with the environment.  **Switch Inputs and System Reset**  The main loop initiates by checking the state of the reset switch (**‘resetSwitch’**). If the reset switch is activated (high state), the system enters a reset sequence. The **‘reset()’** function is called to recalibrate the base level, update the **‘levelBreak’**, reset LED indicators, and store the updated settings in EEPROM memory. This process ensures that the system can adapt to changing conditions, maintaining accuracy in water level measurement.  **Water Level Monitoring and LED Control**  The primary functionality of the main loop revolves around water level monitoring and LED control. Once the **‘levelBreak’** has been initialized (not equal to zero), the **‘provjeraVode()’** function is invoked to read the ultrasonic sensor's data and calculate the current water distance. This distance is then used to determine the appropriate water level using the **‘nivoVode()’** function.  **Water Loss Detection**  The system's capability to detect water loss is another crucial aspect managed by the main loop. If the water loss switch (**‘waterLossSwitch’**) is activated (high state), the loop branches into two scenarios. If the base loss level has not been stored in EEPROM memory, the **‘getBaseLossLevel()’** function is executed. This function captures the initial water level as a reference for detecting potential water loss.  In cases where the base loss level is already stored, the system proceeds to check for water loss by invoking the **‘waterLossCheck()’** function. If water loss is detected, LED indicators are used to signify this occurrence. Alternatively, if no water loss is detected, a different LED pattern indicates normal conditions. | | | | | | | | | | |  |
|  | HARDWARE  |  |  |  | | --- | --- | --- | | Name | Quantity | Component | | U1 | 1 | Arduino Uno R3 | | D1, D2 | 2 | Red LED | | D3, D4 | 2 | Yellow LED | | D5, D6 | 2 | Green LED | | S1, S2 | 2 | Pushbutton | | DIST1 | 1 | Ultrasonic Distance Sensor | | R1, R2 | 2 | 10 kΩ Resistor | | | | | | | | | | | |  |
|  |  | | | | | | | | | | |  |
|  | APPLICATIONS AND BENEFITS The water level monitoring system has various applications and benefits, including:   * Water Tanks: The system can be used to monitor water levels in tanks, ensuring efficient utilization and preventing overflows or shortages. * Irrigation Systems: It enables efficient irrigation management by providing real-time information about water levels, allowing for timely refilling or conservation measures. * Industrial Applications: The system finds applications in industrial settings where water level monitoring is crucial, such as chemical processing, manufacturing, and storage facilities. * Home Automation: It can be integrated into home automation systems to automate tasks based on water levels, such as triggering water pumps or sending notifications. * Water Conservation: By providing accurate and timely water level information, the system promotes water conservation efforts and helps avoid unnecessary wastage. | | | | | | | | | | |  |
|  | CONCLUSION The water level monitoring system developed using Arduino and ultrasonic sensors offers a reliable and versatile solution for monitoring water levels. By providing real-time information and visual indications, it enhances water management, prevents overflows or shortages, and supports water conservation efforts in various applications. The system's flexibility allows for customization and integration into larger automation systems, making it a valuable tool in water resource management.  This detailed general explanation provides an overview of the project, including its purpose, hardware components, operation flow, additional features, and applications. It highlights the benefits of the system and its potential to contribute to efficient water management and conservation.  Thank you for choosing this documentation as your guide. | | | | | | | | | | |  |
|  | **Ultrasonic Water Level**  **-**  **Ahmed Hadžiomerović** | | | | | | | | | | |  |