Al-Zahra University

Zahra Aqamohammadi
9915302001

Microprocessor
WD

1403/04/20

EEPROM Functions:

Store and Retrieve Data: Saves and reads data that stays even when the power is off.

Watchdog Timer:

Safety Timer: Resets the microcontroller if it stops working correctly.

I/O Initialization:

Setup for Devices: Gets ready to use an LED and an LCD display.

LCD Display Update:

Show Reset Count: Displays how many times the microcontroller has been reset due to the timer.

Main Program:

Manage Count: Keeps track of resets in memory.

Start and Monitor: Sets up everything, shows the count on the screen, and keeps the system running smoothly by resetting the timer when needed.

```
#include <mega32.h>
#include <alcd.h>
#include <delay.h>
#include <stdio.h>

#define F_CPU 8000000UL
#define LED_PIN PORTB1
#define EEPROM_ADDR 0x00

void eeprom_write(int addr, int data) {
    while (EECR & (1 << EEWE));
    EEAR = addr;
    EEDR = data;
    EECR |= (1 << EEMWE);
    EECR |= (1 << EEWE);
}

int eeprom_read(int addr) {
    while (EECR & (1 << EEWE));
    EEAR = addr;
    EECR |= (1 << EERE);
    return EEDR;
}

void init_watchdog() {
    WDTCR = (1 << WDE) | (1 << WDP2) | (1 << WDP1);
}

void reset_watchdog() {
    #asm
    wdr
    #endasm
}
```

```
void init_io() {
    DDRB |= (1 << LED_PIN);
    lcd_init(16);
    delay_ms(200);
    lcd_clear();
}

void display_reset_count(int count) {
    char buffer[16];
    sprintf(buffer, "wd timer %d", count);
    lcd_gotoxy(0, 0);
    lcd_puts(buffer);
}

void delay_long() {
    delay_ms(3000);
}

void main(void) {
    int count = eeprom_read(EEPROM_ADDR);
    count++;
    eeprom_write(EEPROM_ADDR, count);
    init_io();
    init_watchdog();
    display_reset_count(count);
    PORTB |= (1 << LED_PIN);

    while (1) {
        delay_long();
        reset_watchdog();
    }
}
```