Project - Smart Light Control System using CH32V003 RISC-V Processor and LDR Sensor

Overview - The Smart Light Control System project leverages the CH32V003 RISC-V processor to create an automated lighting system that adjusts based on ambient light levels. This system uses a Light Dependent Resistor (LDR) to detect light intensity and a relay module to control the lighting. When the ambient light level falls below a certain threshold, the system turns on the lights, and when the light level rises above the threshold, the system turns off the lights. This project aims to provide an energy-efficient and automated lighting solution, suitable for homes, offices, and outdoor areas.

Components Required

- **CH32V003 RISC-V Processor**
- **LDR Sensor**
- **Relay Module**
- **LED Bulb (or any other light source)**
- **Power Supply**
- **Breadboard**
- **Jumper Wires**

Electrical Properties

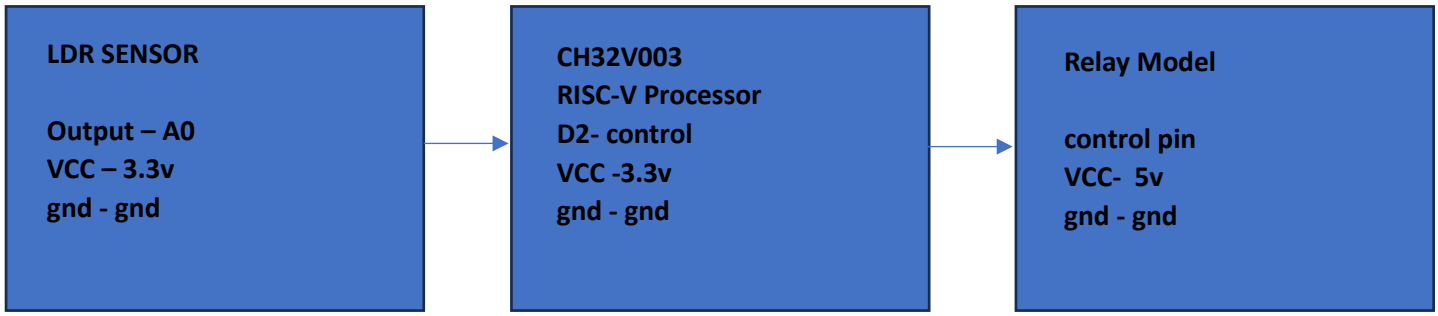
- **CH32V003 RISC-V Processor**: Operates at voltages between 1.8V to 3.6V, featuring GPIO pins for interfacing with external devices and supporting communication protocols like SPI, I2C, and UART.
- **LDR Sensor**: Typically operates at 3.3V or 5V, with resistance varying based on light intensity.
- **Relay Module**: Operates at 5V, capable of switching high-voltage devices (like lights) using a low-voltage signal from the microcontroller.

Circuit Connection

- **LDR Sensor Connection**:
  - **Output Pin**: Connected to the A0 pin of the CH32V003 processor (analog input).
  - **VCC Pin**: Connected to the 3.3V pin of the CH32V003 processor.
  - **GND Pin**: Connected to the GND pin of the CH32V003 processor.
- **Relay Module Connection**:
  - **Control Pin**: Connected to the D2 pin of the CH32V003 processor (digital output).
  - **VCC Pin**: Connected to the 5V pin of the power supply.
  - **GND Pin**: Connected to the GND pin of the power supply.
  - **NO (Normally Open) Pin**: Connected to the live wire of the light bulb.

- **COM (Common) Pin**: Connected to the live wire of the power supply.
- **NC (Normally Closed) Pin**: Left unconnected

Block Diagram :



Working

The Smart Light Control System continuously monitors the ambient light level using the LDR sensor. The CH32V003 processor reads the analog signal from the LDR sensor and converts it to a digital value. If the light level falls below the predefined threshold, the processor activates the relay module, turning on the light. Conversely, if the light level rises above the threshold, the processor deactivates the relay module, turning off the light. This automated process ensures that the lights are only on when needed, providing an energy-efficient lighting solution.

Code

```
#include <ch32v00x.h>
#include <debug.h>

#define LIGHT_THRESHOLD 500  // Adjust this value based on your LDR sensor

void GPIO_Config(void) {
    GPIO_InitTypeDef GPIO_InitStructure = {0};
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOD, ENABLE);
    GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2;
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    GPIO_Init(GPIOD, &GPIO_InitStructure);
}
```

```
void ADC_Config(void) {
    ADC_InitTypeDef ADC_InitStructure = {0};
    RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1, ENABLE);
    ADC_InitStructure.ADC_Mode = ADC_Mode_Independent;
    ADC_InitStructure.ADC_ScanConvMode = DISABLE;
    ADC_InitStructure.ADC_ContinuousConvMode = ENABLE;
    ADC_InitStructure.ADC_ExternalTrigConv = ADC_ExternalTrigConv_None;
    ADC_InitStructure.ADC_DataAlign = ADC_DataAlign_Right;
    ADC_InitStructure.ADC_NbrOfChannel = 1;
    ADC_Init(ADC1, &ADC_InitStructure);
    ADC_RegularChannelConfig(ADC1, ADC_Channel_0, 1, ADC_SampleTime_55Cycles5);
    ADC_Cmd(ADC1, ENABLE);
    ADC_ResetCalibration(ADC1);
    while(ADC_GetResetCalibrationStatus(ADC1));
    ADC_StartCalibration(ADC1);
    while(ADC_GetCalibrationStatus(ADC1));
    ADC_SoftwareStartConvCmd(ADC1, ENABLE);
}

int main(void) {
    uint16_t lightLevel = 0;
    NVIC_PriorityGroupConfig(NVIC_PriorityGroup_2);
    SystemCoreClockUpdate();
    Delay_Init();
    GPIO_Config();
    ADC_Config();

    while(1) {
        lightLevel = ADC_GetConversionValue(ADC1);
        if (lightLevel < LIGHT_THRESHOLD) {
            GPIO_WriteBit(GPIOD, GPIO_Pin_2, SET);  // Turn on the relay
```

```
        } else {
            GPIO_WriteBit(GPIOD, GPIO_Pin_2, RESET);  // Turn off the relay
        }
        Delay_Ms(100);
    }
}

void NMI_Handler(void) {}
void HardFault_Handler(void) {
    while (1) {}
}
```