الجامعة المصرية اليابانية للعلوم و التكنولوجيا
エジプト日本科学技術大学
EGYPT-JAPAN UNIVERSITY OF SCIENCE AND TECHNOLOGY

Embedded Systems

MTE323 Individual Report

ATmega32 Microcontroller and Stepper Motor Interfacing

Submitted by:

Mahmoud Esam Abualfadl 120210049

Submitted to:

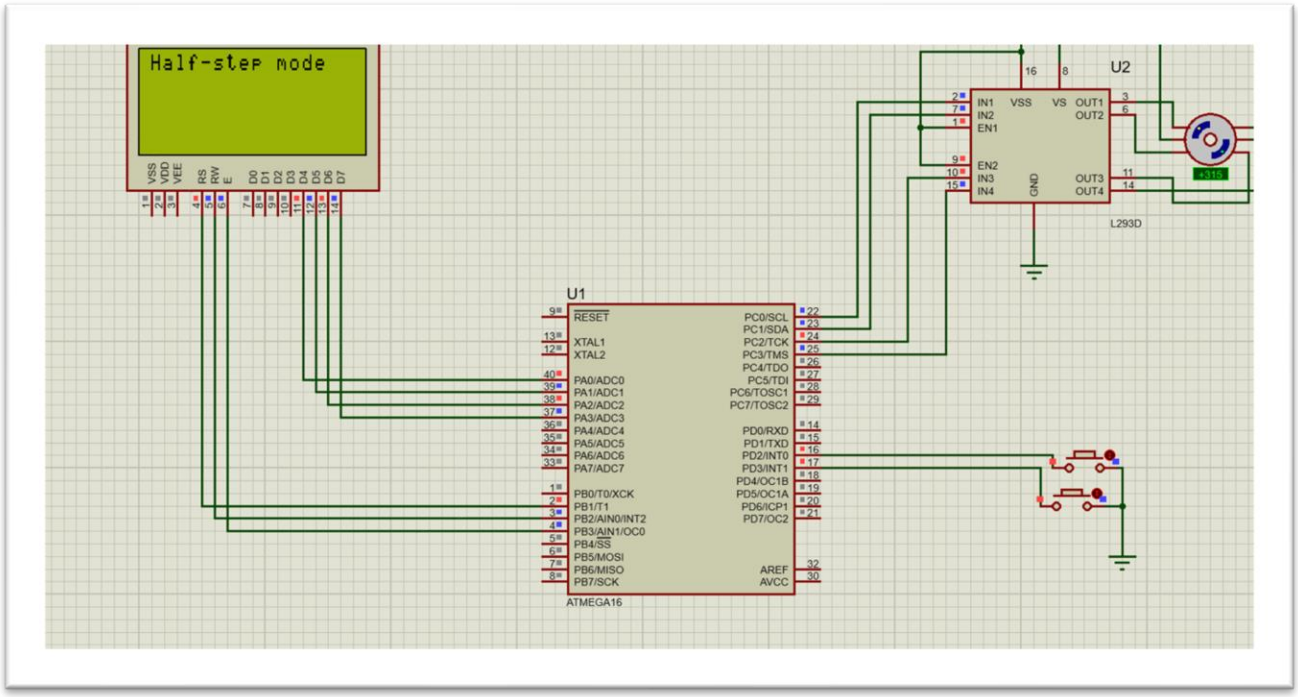
Pro.Haitham El-Hussieny

Project Description:

This lab experiment involves simulating the interface between an AVR ATmega32 microcontroller and a stepper motor using electronic circuit simulation software. The objective is to write a C program to control the stepper motor's direction (clockwise or counter-clockwise) and stepping mode (full or half stepping) using pushbuttons.

Design:

1. AVR ATmega32 Microcontroller
2. Stepper Motor
3. L293D Motor
4. Pushbuttons (2)



Functionality

The program sets up the microcontroller to control a stepper motor via PORTC0-PORTC3. It uses external interrupts (INT0 and INT1) to change the motor's direction and stepping mode in response to pushbutton inputs. Timer0 creates precise delays to control the timing between steps, ensuring that the motor steps every second. The Timer0 overflow interrupt service routine handles the actual stepping logic, updating the stepper motor's position based on the current direction and stepping mode. This structure enables the motor control to be efficient and responsive to user inputs without the use of blocking delay functions, thereby fully utilizing the microcontroller's interrupt capabilities.

1. **Global Variables and Definitions:**
    - step-index: Tracks the current step in the sequence
    - direction: Stores the motor's direction (0 for clockwise, 1 for counter-clockwise
    - overflow count: Counts Timer0 overflows to achieve the desired delay

2. **Function**
    - Purpose: Initializes the microcontroller’s ports, external interrupts, and Timer0
    - Sets PORTC0-PORTC3 as outputs for the stepper motor control
    - Configures PD2 (INT0) and PD3 (INT1) as inputs with internal pull-ups for pushbuttons
    - Enables external interrupts INT0 and INT1, triggered on the falling edge
    - Configures Timer0 with a prescaler of 1024 and enables its overflow interrupt
    - Enables global interrupts with sei().
    - LCD Driver

Summary of Program Challenges and Improvements

Challenges Faced:

1. Initial Approach with Increment Steps:

The first attempt involved adding steps without using Timer0 and avoiding for loops. This method was found to be inefficient and unoptimized.

2. Interrupt-Based Approach:

- The program was revised to use interrupts to toggle the direction and stepping mode flags.
- Timer0 was utilized to manage the autonomous increment of steps with a controlled delay.
- This method solved some problems but introduced new ones, such as difficulty seeing specific angles (90, 180, 270, 360 degrees) clearly.

3.

- When steps were incremented within the Timer0 overflow ISR, they were displayed correctly and the stepper motor rotated smoothly.

Improved Solution:

- The revised solution makes use of interrupts and Timer0 to efficiently and precisely control the stepper motor. Here's a simple and unique way to achieve the desired functionality.