Lab 4: Computation and Comparison

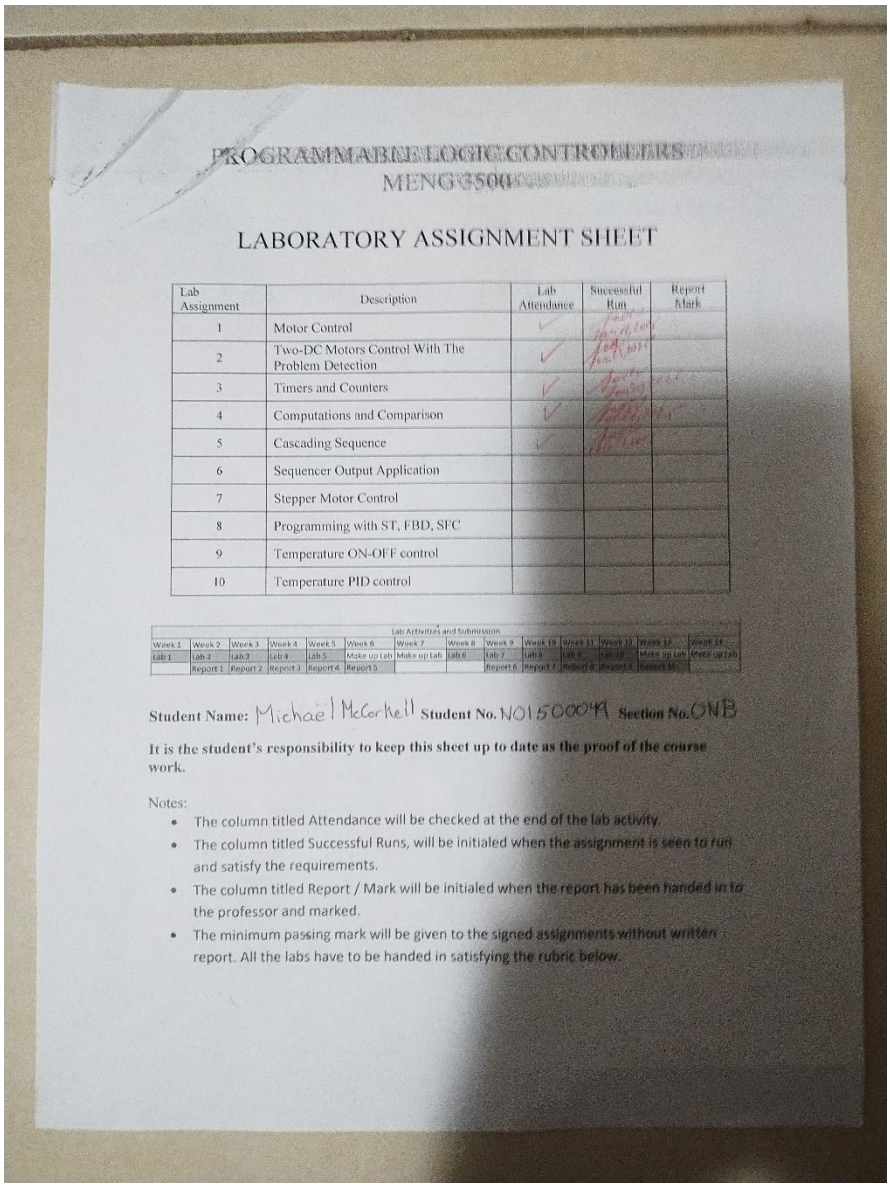
Michael McCorkell

Humber Polytechnics

Programmable Logic Controllers: MENG 3500 0NB

Savdulla Kazazi

February 3, 2025



PROGRAMMABLE LOGIC CONTROLLERS
MENG 3500

LABORATORY ASSIGNMENT SHEET

Lab Assignment	Description	Lab Attendance	Successful Run	Report Mark
1	Motor Control	✓		
2	Two-DC Motors Control With The Problem Detection	✓		
3	Timers and Counters	✓		
4	Computations and Comparison	✓		
5	Cascading Sequence	✓		
6	Sequencer Output Application			
7	Stepper Motor Control			
8	Programming with ST, FBD, SFC			
9	Temperature ON-OFF control			
10	Temperature PID control			

Lab Activities and Submission														
Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14
Lab 1	Lab 2	Lab 3	Lab 4	Lab 5	Make up Lab	Make up Lab	Lab 6	Lab 7	Lab 8	Lab 9	Lab 10	Make up Lab	Make up Lab
	Report 1	Report 2	Report 3	Report 4	Report 5			Report 6	Report 7	Report 8	Report 9	Report 10	

Student Name: Michael McCorkell **Student No.** N01500019 **Section No.** ONB

It is the student's responsibility to keep this sheet up to date as the proof of the course work.

Notes:

- The column titled Attendance will be checked at the end of the lab activity.
- The column titled Successful Runs, will be initialed when the assignment is seen to run and satisfy the requirements.
- The column titled Report / Mark will be initialed when the report has been handed in to the professor and marked.
- The minimum passing mark will be given to the signed assignments without written report. All the labs have to be handed in satisfying the rubric below.

Objectives

This lab aimed to develop a PLC program that integrates **comparison and computational instructions** to control a cyclic ramping process. The key objectives were:

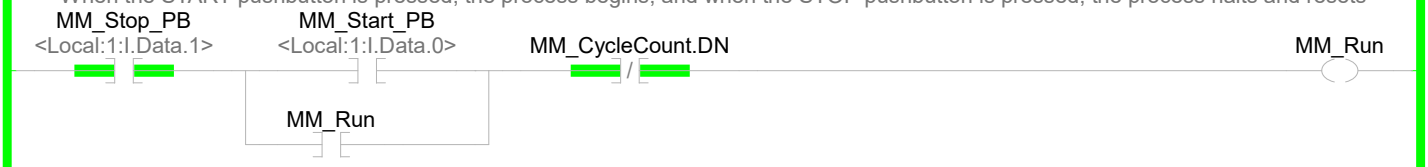
1. Establishing stable communication between the PLC and the computer.
2. Implementing **comparison and computational instructions** for value manipulation.
3. Developing a **Start-Stop control mechanism** for the process.
4. Creating a **cyclic ramp logic** where a value increases and decreases at set intervals.
5. Controlling **indicator lights** based on value thresholds.
6. Counting **completed cycles** and limiting the process to three cycles.
7. Implementing an **immediate stop function** using a pushbutton.

Description of Work Completed

- **PLC Communication Setup:** The PLC was connected to the computer, and communication stability was verified.
- **Program Logic Development:**
    - The **Start-Stop mechanism** was created using a **N.O. START pushbutton** and a **N.C. STOP pushbutton** to initiate and halt the process.
    - A **ramping logic** was implemented where a VALUE:
        - **Increases from 0 to 60** in increments of **10** every **1 second**.
        - **Decreases from 60 to 0** in decrements of **20** every **1 second**.
        - Repeats continuously for **three cycles** before stopping automatically.
    - **Light Control Logic:**
        - The **GREEN_LIGHT** turns ON only when the VALUE reaches **60**.
        - The **RED_LIGHT** turns ON only when the VALUE reaches **0**, except during the last cycle.
    - **Cycle Counting and Limitation:**
        - A **counter** tracked the number of complete ramp cycles.
        - After **three full cycles**, the process halted, and the counter reset.
    - **Immediate Stop Function:**
        - Pressing the **STOP_PB** immediately stopped the process and reset both the VALUE and COUNTER to **zero** using a **MOV instruction**.
- **Field Device Wiring & Testing:**
    - The input pushbuttons (START_PB, STOP_PB) and output lights (GREEN_LIGHT, RED_LIGHT) were wired according to specifications.
    - The program was downloaded, tested, and verified for proper functionality.
- **Troubleshooting & Debugging:**
    - Live data monitoring was performed to ensure correct value ramping, light activation, and cycle tracking.
    - Fault scenarios such as **early stopping, incorrect counter values, and improper light activation** were identified and resolved.

When the START pushbutton is pressed, the process begins, and when the STOP pushbutton is pressed, the process halts and resets

0



Do not Move Rung because of Scanning Sequence for program.
Controls the green light. When the green light is activated, the MM_Half_Cycle variable is also triggered.

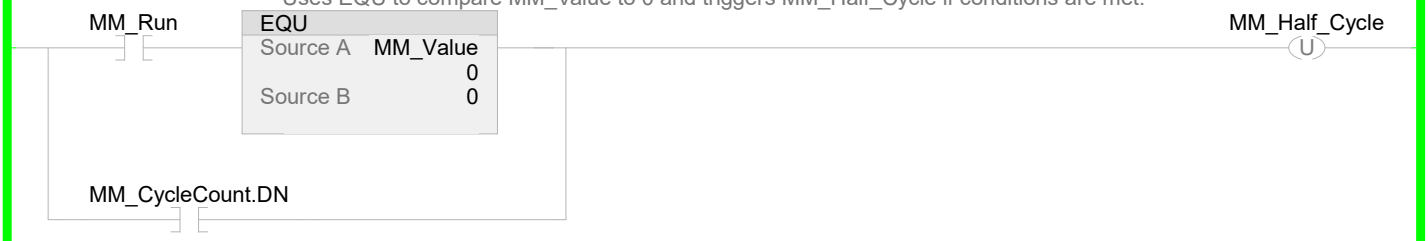
1

counter MM_Half_Count that increments when the green light is activated.

2

Uses EQU to compare MM_Value to 0 and triggers MM_Half_Cycle if conditions are met.

3

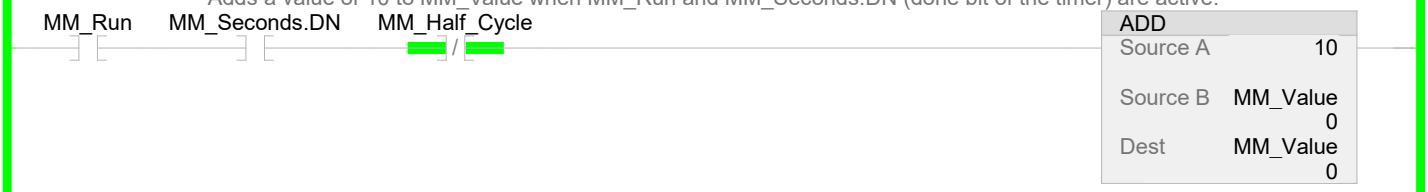


Starts a timer (MM_Seconds) with a preset of 1000 when MM_Run is active.

4

Adds a value of 10 to MM_Value when MM_Run and MM_Seconds.DN (done bit of the timer) are active.

5



Turns on the green light when MM_Value reaches 60 and remains on for that duration.

6

Uses a subtraction operation to decrease MM_Value by 20 when MM_Seconds.DN and MM_Half_Cycle are active.

7

MM_Seconds.DN MM_Half_Cycle

SUB
Source A MM_Value 0
Source B 20
Dest MM_Value 0

Similarly, the RED_LIGHT will turn ON when the VALUE reaches 0, but will only remain ON for the duration of the 0 value.
For the final cycle (when the VALUE is ramping down to 0), there is no need for the RED_LIGHT to turn ON

8

MM_Run

EQU
Source A MM_Value 0
Source B 0

GRT
Source A MM_Half_Count.ACC 0
Source B 0

MM_RedLight <Local:2:O.Data.1>

The process will stop automatically after 3 complete ramp cycles, at which point the counter resets to zero.

9

MM_Run MM_RedLight <Local:2:O.Data.1>

CTU
Counter MM_CycleCount
Preset 3
Accum 0
CU
DN

If the STOP_PB is pressed at any time during the operation, the process will immediately halt. The MOV instruction should be used to reset both the VALUE and the COUNTER to zero

10

MM_Stop_PB <Local:1:I.Data.1> /

MM_CycleCount.DN

MOV
Source 0
Dest MM_CycleCount.ACC 0

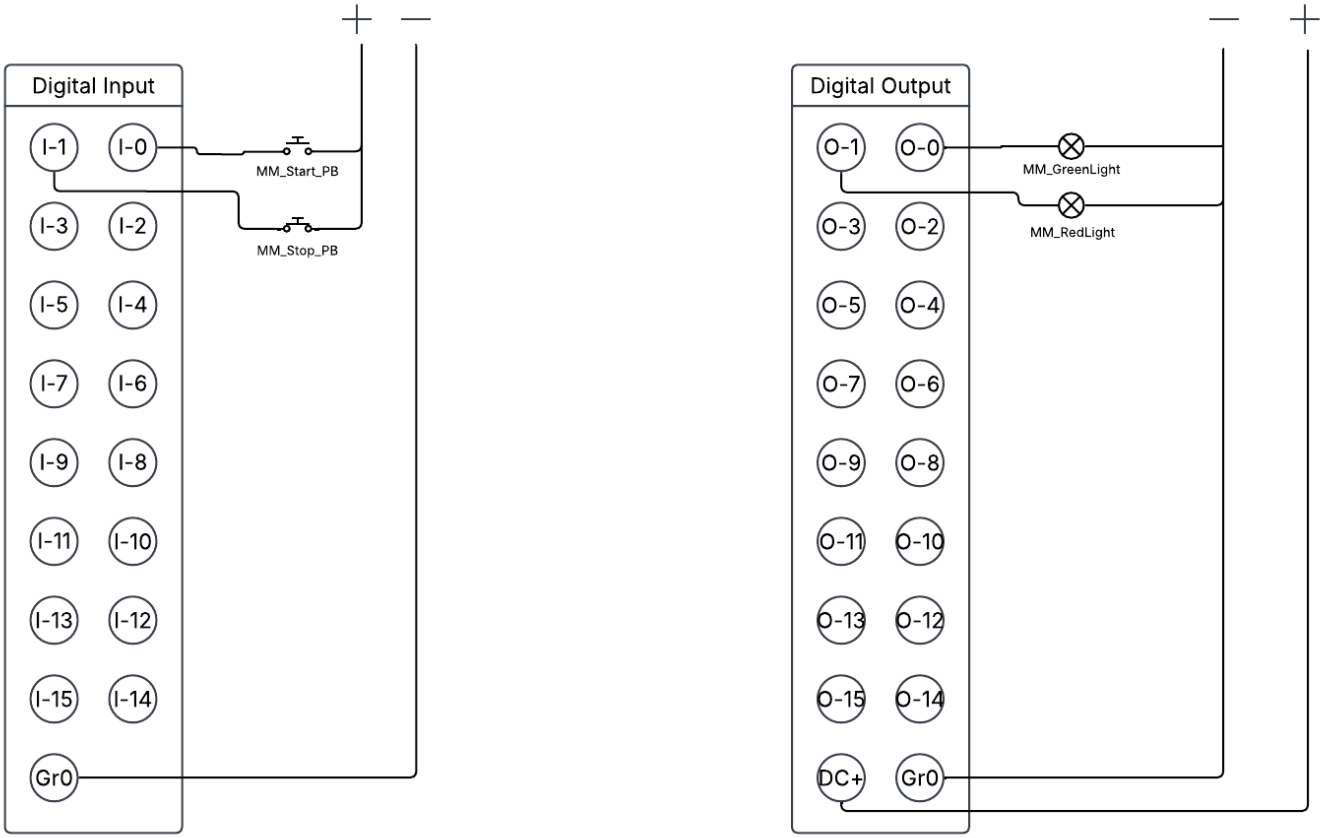
MOV
Source 0
Dest MM_Seconds.ACC 0

MOV
Source 0
Dest MM_Value 0

MOV
Source 0
Dest MM_Half_Count.ACC 0

(End)

Wiring Diagram



Conclusions

The lab was successfully completed, with all requirements met. The **value ramping, light control, and cycle tracking** functioned as expected. The integration of **comparison and computational instructions** provided valuable experience in real-time process control. Additionally, the **immediate stop functionality** demonstrated how PLCs handle emergency conditions efficiently. This lab reinforced **event-driven programming, data manipulation, and automated process control** using PLC logic.