Lab 7: Stepper Motor Control

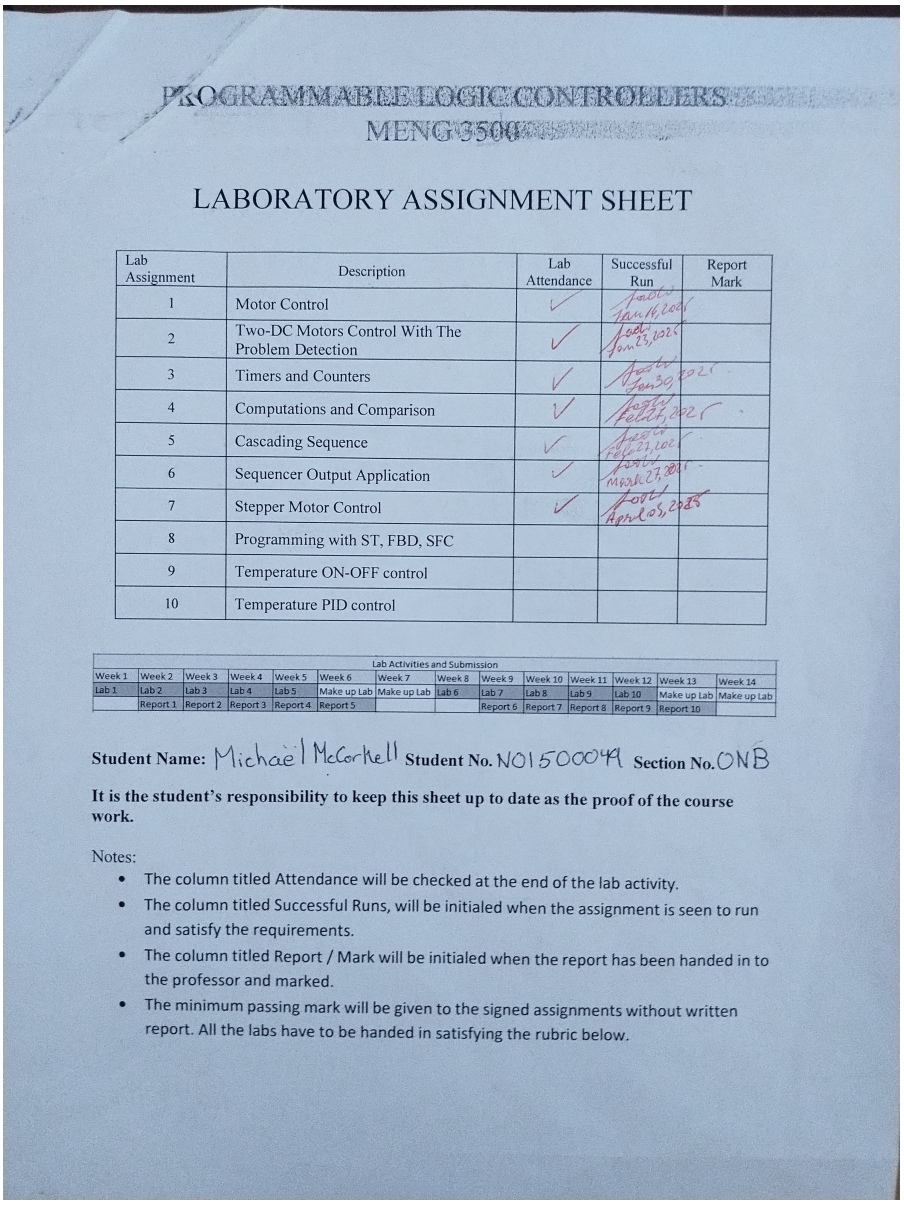
Michael McCorkell

Humber Polytechnics

Programmable Logic Controllers: MENG 3500 0NB

Savdulla Kazazi

April 6th, 2025



PROGRAMMABLE LOGIC CONTROLLERS
MENG 3500

LABORATORY ASSIGNMENT SHEET

Lab Assignment	Description	Lab Attendance	Successful Run	Report Mark
1	Motor Control	✓	Jan 16, 2025	
2	Two-DC Motors Control With The Problem Detection	✓	Jan 23, 2025	
3	Timers and Counters	✓	Jan 30, 2025	
4	Computations and Comparison	✓	Feb 27, 2025	
5	Cascading Sequence	✓	Feb 27, 2025	
6	Sequencer Output Application	✓	March 27, 2025	
7	Stepper Motor Control	✓	April 03, 2025	
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.** N01500049 **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.

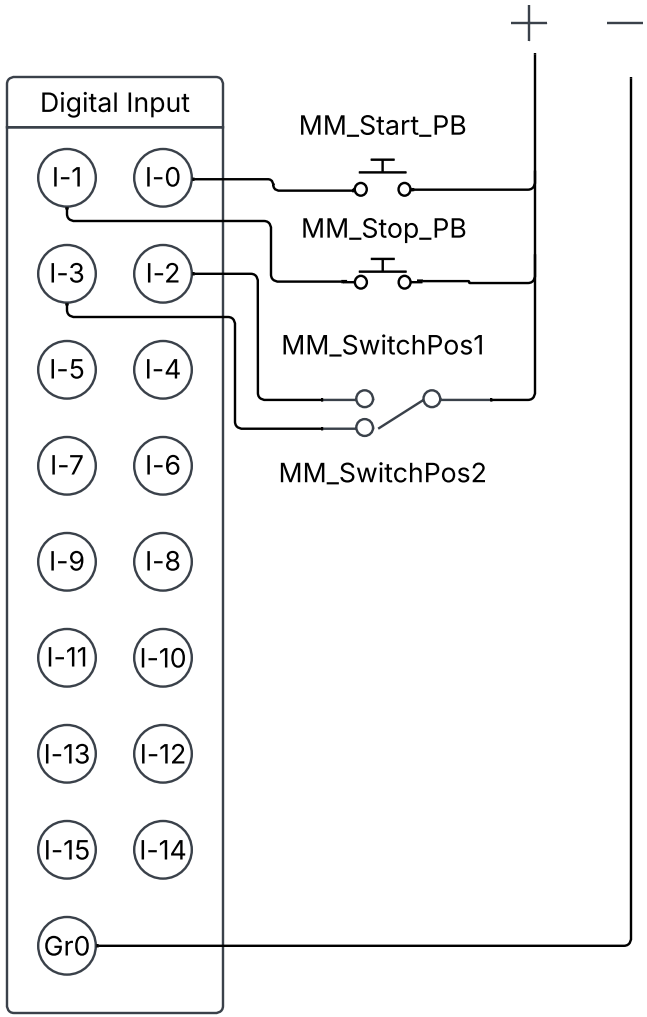
Objectives

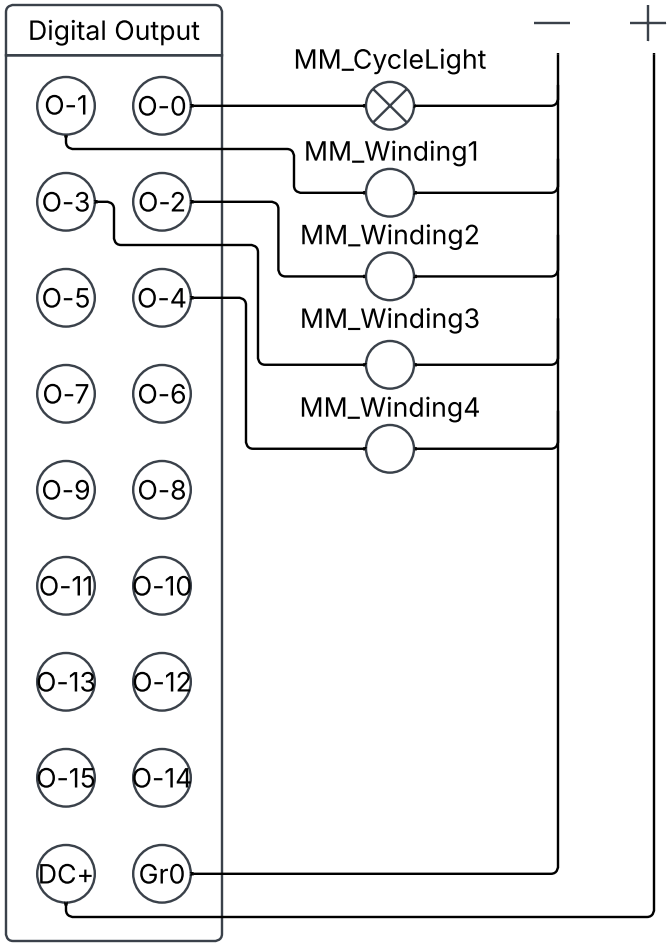
This lab focused on implementing **Bit Shift Left (BSL) and Bit Shift Right (BSR) instructions** to control a **four-coil stepper motor** using a PLC. The main objectives were:

1. Establishing stable communication between the **PLC and the computer**.
2. Implementing **BSL and BSR instructions** to control motor movement.
3. Developing a **Start-Stop-Run logic** for motor operation.
4. Controlling the **Clockwise (CW) and Counterclockwise (CCW) rotation** of the motor.
5. Wiring and testing the **stepper motor windings** to ensure correct excitation.
6. Running and troubleshooting the program to ensure accurate motor movement.

Description of Work Completed

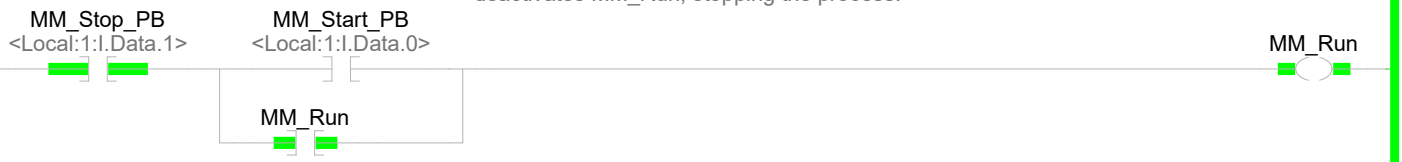
- **PLC Communication Setup:**
  - Communication between the PLC and the computer was established and verified.
- **Stepper Motor Control Logic Development:**
  - A **four-coil stepper motor** was programmed using **Single-Coil Excitation Mode**.
  - The **BSL (Bit Shift Left) instruction** was used to rotate the motor **Clockwise (CW)**.
  - The **BSR (Bit Shift Right) instruction** was used to rotate the motor **Counterclockwise (CCW)**.
  - A **0.03-second timer** controlled the bit shifting in both directions.
- **Start-Stop-Run Logic Implementation:**
  - **N.O. START_PB:** Initiates the motor operation.
  - **N.C. STOP_PB:** Stops the motor immediately.
  - **Maintained DIRECTION_PB:** Selects the rotation direction (CW or CCW).
- **Field Device Wiring & Testing:**
  - The input devices (**START_PB, STOP_PB, DIRECTION_PB**) and output devices (**stepper motor windings 1-4**) were wired properly.
  - The program was downloaded and executed to confirm proper motor movement.
- **Troubleshooting & Debugging:**
  - Live monitoring ensured **smooth bit shifting and accurate coil excitation**.
  - Incorrect rotation behaviors were identified and adjusted by fine-tuning the **timing and bit shift logic**.
  - Edge cases, such as abrupt stops or incorrect direction shifts, were tested and resolved.


Digital Input
I-1
I-0
I-3
I-2
I-5
I-4
I-7
I-6
I-9
I-8
I-11
I-10
I-13
I-12
I-15
I-14
Gr0
MM_Start_PB
MM_Stop_PB
MM_SwitchPos1
MM_SwitchPos2
+
—


Digital Output
O-1
O-0
O-3
O-2
O-5
O-4
O-7
O-6
O-9
O-8
O-11
O-10
O-13
O-12
O-15
O-14
DC+
Gr0
MM_CycleLight
MM_Winding1
MM_Winding2
MM_Winding3
MM_Winding4
—
+

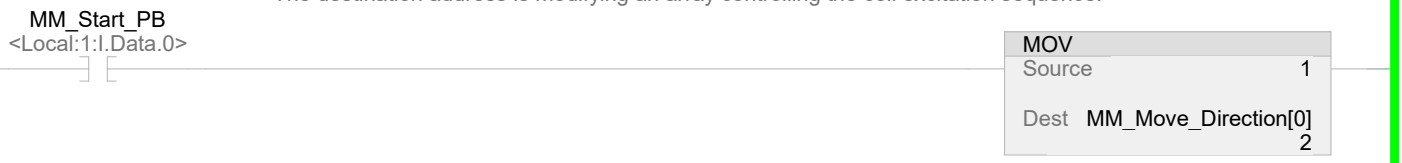
This rung defines the basic start-stop control for the stepper motor.
When the start button is pressed, the MM_Run, is energized, allowing the motor control sequence to proceed. Pressing the stop button deactivates MM_Run, stopping the process.

0



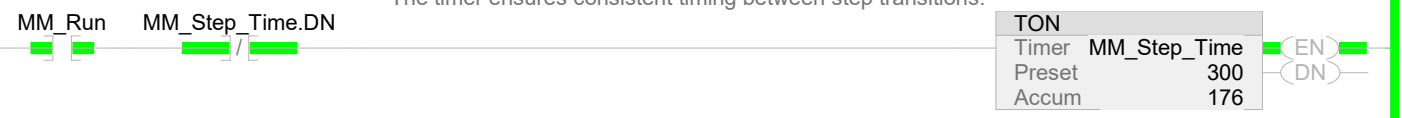
This rung sets the direction of movement for the stepper motor.
When MM_Start_PB is pressed, a MOV instruction moves a value into the MM_Move_Direction array, indicating the motor's direction.
The destination address is modifying an array controlling the coil excitation sequence.

1



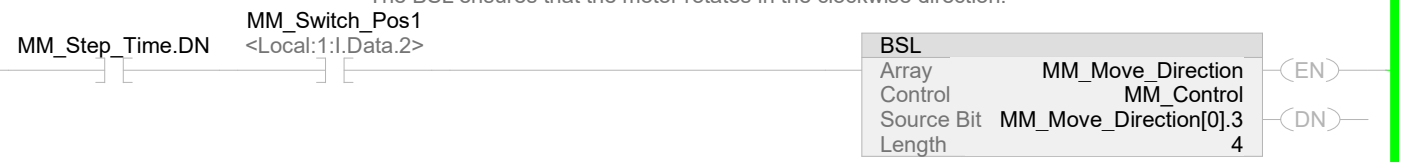
This rung manages the stepping interval of the motor using a timer (MM_Step_Time).
The timer has a preset value of 300 milliseconds, meaning it will execute a step shift every 0.3 seconds.
The timer ensures consistent timing between step transitions.

2



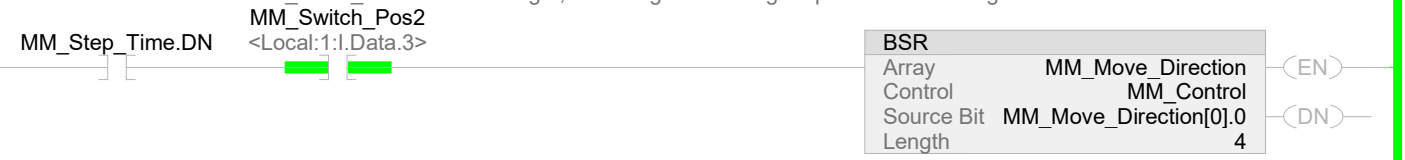
If MM_Step_Time.DN (timer done) is true and MM_Switch_Pos1 is active, the BSL (Bit Shift Left) instruction executes.
This shifts bits within the MM_Move_Direction array to the left, activating the next winding of the stepper motor.
The BSL ensures that the motor rotates in the clockwise direction.

3



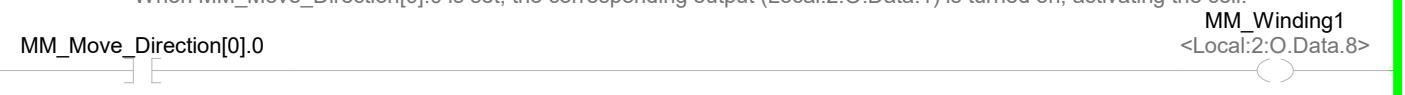
If MM_Step_Time.DN is true and MM_Switch_Pos2 is active, the BSR (Bit Shift Right) instruction executes.
This shifts bits within MM_Move_Direction to the right, reversing the winding sequence and making the motor rotate counterclockwise.

4



This rung energizes winding 1 of the stepper motor (MM_Winding1).
When MM_Move_Direction[0].0 is set, the corresponding output (Local:2:O.Data.1) is turned on, activating the coil.

5



Similar to the previous rung, this one activates winding 2 (MM_Winding2) when MM_Move_Direction[0].1 is true.
This is part of the stepper motor excitation sequence.

6

This rung controls the third winding (MM_Winding3).
When MM_Move_Direction[0].2 is set, the corresponding output coil is energized, continuing the motor's stepping sequence.

7

MM_Move_Direction[0].2 — MM_Winding3 <Local:2:O.Data.10>

This rung energizes winding 4 (MM_Winding4).
When MM_Move_Direction[0].3 is set, the final coil in the excitation sequence is activated before shifting again.

8

MM_Move_Direction[0].3 — MM_Winding4 <Local:2:O.Data.11>

This rung uses a counter (MM_NumbCycle) to track the number of completed cycles.
When MM_Control.UL (Upper Limit) is reached, the cycle light (MM_CycleLight) is turned on, indicating the motor has completed a predefined number of steps.

9

MM_Control.UL — MM_CycleLight <Local:2:O.Data.5>

CTU
Counter MM_NumbCycle
Preset 0
Accum 32
CU
DN

This rung resets the cycle counter (MM_NumbCycle) when MM_Stop_PB is pressed.
Additionally, a MOV instruction resets MM_Move_Direction[0] to zero, ensuring the motor is properly reset before restarting.

10

MM_Stop_PB <Local:1:I.Data.1> — MM_NumbCycle RES

MOV
Source 0
Dest MM_Move_Direction[0] 2

(End)

Conclusions

The lab hasn't been completed or shown, but the wiring diagram and the thought of the PLC programming has been completed, and the **stepper motor responded accurately** to the programmed instructions. The **BSL and BSR instructions** provided efficient and reliable motor control, allowing for **precise rotation in both directions**. The **Start-Stop logic and direction control** ensured ease of operation. This lab reinforced key concepts in **bit shifting, stepper motor excitation, and PLC-based motion control**.