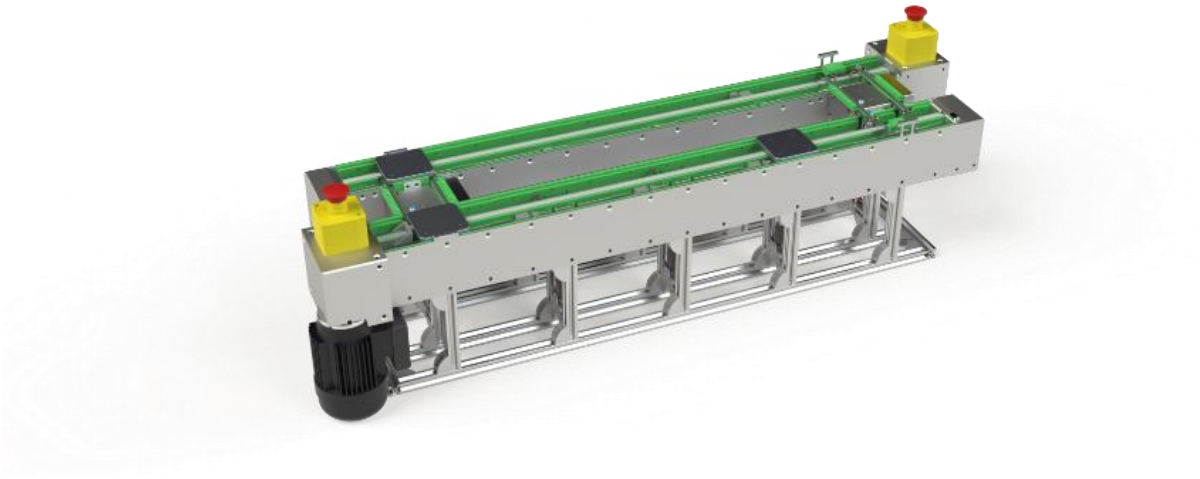
Implementation Document PLC

By Mini-FLUFFY team



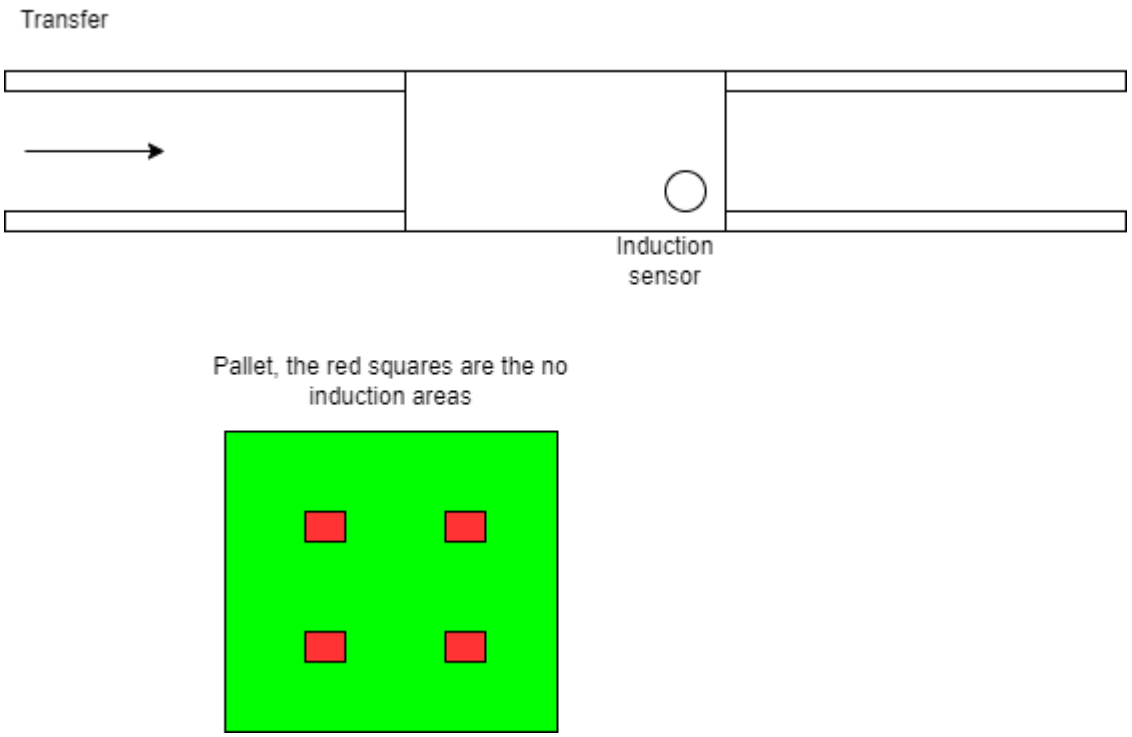
Contents

Introduction

In projects, there are always some hurdles to overcome. These problems take time and can sometimes be quite complex. In this document certain noteworthy details about implementing the program on Mini-Fluffy PLC. The problem and our solution for the problem will be described, along with the reason for picking this as the primary solution. This document aims to explain these intricate problems and not explain every single line or section of code.

Problem 1: Transfer induction stop

The transfer unit is the part of the machine where the pallet moves to a different belt. The transfer unit lifts the pallet, turns on the belt and once it gets to the end, it lowers to put it back on the belt. However, there is one step I didn't mention, as it isn't always necessary: stopping in the middle. In the middle there is a platform that can also raise the pallet up or down. To get it to stop, we must turn the belt off and we can check when we need to do that with the induction sensor on the transfer station. This is a digital sensor that sends a high signal when it detects induction with the pallet. The problem is that when it first senses the pallet, it isn't properly aligned yet with the middle. If we stop the belt now, the pallet will not be properly lifted. Below is a visualization of the relevant part. The green area of the square is what triggers the induction sensor, but this is too early, as it's not fully on the plate. What we do is we wait for it to hit the area of the pallet on which the induction sensor does not trigger. These are the red squares on the pallet in this visualization. We do this by waiting for the sensor to be triggered, and then waiting for the sensor to no longer be high. When this happens, it is perfectly aligned with the lift area. This implementation was chosen as it has worked consistently for us throughout the project.



Model of transfer and pallet.

Problem 2: Next station check

One of the main considerations while running the system was the way to run as many pallets as possible at one time, so a system is required to prevent collisions between these pallets. We had to make sure that even if a pallet was not on a station but on the belt in between stations, which is problematic as we have no sensors capable of checking this. For this reason, we implemented the next station check system as we have it now. Instead of the station being treated as idle while a pallet is being transported to the next station, the station has a separate release state in which we ensure that the pallet doesn't leave until the next station is free and we make sure the pallet arrives at that station. One of the major advantages of this system is that regardless of belt length and process times between stations, it should be impossible for the system to cause a collision with the system working as intended. It leaves the implementation robust enough to handle various variables.

Problem 3: Ghost pallets

With our next station check system implemented we had a new problem which appeared, the interference of outside users like ourselves. As we were running the system we noticed that we would often lock a station into the release state by picking the pallet of the belt mid transfer, as the system waits to change to the idle state by checking if the next station has received the pallet, removing it before it reaches it causes it to get stuck in an infinite loop until the pallet is placed back. Our solution is to add a time out feature to the release state, so if the station waits too long for the pallet to arrive at the next station, the station will reset and increase a counter we keep of the "ghost pallet". With this we can keep track of the number of pallets on the system and it also prevents the aforementioned infinite loop because of our intervention. Sadly, this does make the next station check require a timeout timer making it more dependent on the belt length between stations.