Praktikumsbericht – Drehteller

1. Problemstellung/Aufgabenstellung

Die Aufgabe des Praktikums bestand darin sich nach und nach vertraut mit CoDeSys und seinen fünf standardisierten Programmiersprachen nach der IEC 61131-3 zu machen. Das E-Book diente zur Orientierung.

Nachdem das Herantasten in CoDeSys erledigt war, sollten wir einen Drehteller realisieren, der anhand einer sogenannten Zweihandschaltung zum Laufen gebracht werden soll. Dafür haben wir uns entschieden die Zweihandschaltung separat zu dem eigentlichen Ablauf der Steuerung zu realisieren. Als dritten Baustein haben wir ein Not_Aus realisiert. Die Zweihandschaltung dient dafür, dass die Steuerung gesichert mit zwei Schaltern aktiviert wird, d.h. dass das Drücken von den beiden Schaltern innerhalb vorgegebener Zeit erfolgen muss, damit die Steuerung das Startkommando erhält, um den Zylinder zu aktivieren und die Schutztür herunterzufahren. Sobald die Schutztür unten ist, fängt der Teller an sich, um exakt 90 Grad zu drehen, bis die Ruheposition wieder eintrifft.

Die Ruheposition ist hierbei, dass die Schutztür oben ist und der Teller stillsteht, damit das fertige Produkt entnommen und ein neues hereingelegt werden kann. Für die nächste Drehung müssen die beiden Schalter wieder gedrückt werden.

Der Not_Aus Schalter schaltet alle Motoren und Zylinder umgehend aus und die Steuerung stoppt. Nachdem die Steuerung gestoppt ist, muss der Not_Aus Schalter wieder deaktiviert werden, um den Ablauf der Steuerung wieder lauffähig zu machen, d.h. dass sobald der Not_Aus Schalter gedrückt ist, funktioniert nichts mehr, auch wenn man die beiden Schalter drückt.

2. Lösungskonzept

Wir haben in dem PLC_PRG, das als Funktionsbaustein definiert ist, drei Netzwerke angelegt. Das erste Netzwerk ist für den *Ablauf der Steuerung*, das zweite Netzwerk ist die *Zweihandschaltung* und das letzte Netzwerk für den *Not_Aus.*

Als globale Variablen haben wir den *„Global_START“*, die beiden Schalter (*„Rechts“ und „Links“*) sowie den Zylinder (*„K2“*) und den Motor (*„K1“*) vom Typ „BOOL“ definiert und die boolesche Variable *„Not_Aus“.*
Der Grund hierfür ist, dass *„Global_START“, „K2“* und *„K1“* beim Betätigen des Not_Aus Schalters resettet werden müssen und da sie in einem anderen Netzwerk definiert sind, können wir nur darauf zugreifen wenn wir diese global definieren. Wir haben das Netzwerk "Ablauf" in AS(Ablaufsteuerung) erstellt, da wir hier Schritt für Schritt durch das Vorgehen durchlaufen und in den Schritten mit KOP oder ST die gewünschten Zustände und Variablen setzten können, um Visulisierung und Funktionalität nach Bedürfnis zu formen. Die restlichen Netzwerke (*"Zweihandschaltung" und "Not_Aus"*) sind nach EBook in FUP realisiert worden und durch eigene Zustände ergänzt worden, diese haben wir jedoch nicht als eigene Bausteine realisiert, sondern haben im parallelen Netzwerk (im PLC_PRG) die Realisierung der Schaltung gestaltet.
Um einen automatisierten Ablauf, dass nur das Betätigen der beiden Schalter benötigt, haben wir in den einzelnen Schritten den Motor (*„K1“*), den Zylinder (*„K2“*) und die restlichen Variablen in der Spule als Ausgang resettet oder gesetzt. Dadurch konnten die Transitionen/Bedingungen erfüllt werden ohne das manuelle Schalten der jeweiligen

Autoren: Nils Halwas, Matr-Nr.: 5204636 und Mehmet Algül, Matr-Nr.: 5273858
Modul: Leit-, Steuerungs- und Kommunikationstechnik (LSK)
Professor: Bernd Müller

Tasten und Sensoren.
Der Schritt *„Links_Rechts_Reset“* dient dazu, die global definierten booleschen Variablen zu resetten, damit wenn die erste Rotation des Tellers um 90 Grad erfolgt ist, diese für die zweite Rotation gedrückt werden können.
Der Schritt *„Schutztur_nach_unten“* soll den Sensoren *„oben“* und *„unten“* signalisieren, dass das Startsignal durch die beiden Tasten erfolgt ist und die Tür herunterfahren kann. Dafür wurde in der Programmiersprache KOP das *„K2“* gesetzt, die boolesche Variable *„unten“* gesetzt, *„S1“* gesetzt, der aussagt, dass sich der Teller noch nicht dreht, und die boolesche Variable *„oben“* resettet.
Demnach ist die Bedingung für den nächsten Schritt erfüllt und der TON-Timer *„T1“* startet. Parallel dazu im Schritt *„Tor_zu“* fährt die Schutztür herunter. Zu dem Timer in dem Schritt *„Station1“*, haben wir uns überlegt, dass bevor der Teller sich bereit macht, um zu rotieren, geben wir dem Mitarbeiter noch wenige Sekunden sich in Sicherheit zu wiegen.
Nun da der Timer *„T1“* abgelaufen ist, wird der nächste Schritt *„Start_Teller_drehen“* aktiviert und der Teller bereitet sich vor, um zu drehen. Dieser Schritt beinhaltet eine Anfangsaktion, die *„K1“* setzt und *„S1“* resettet.
Im Anschluss gelangen wir in den Schritt *„Teller_dreht“*, wo der Teller endlich seine erste 90 Grad Rotation macht.
Im nächsten Schritt *„Teller_stop_und_schutztur_hoch“* werden alle notwendigen Variablen gesetzt oder resettet, um den Startzustand für den Sprung zu *„Init“* zu gewährleisten.
Im *„Init“* wird die Schutztür hochgefahren und die boolesche Variable *„S1“* auf TRUE gesetzt.

Um den Not_Aus Schalter zu integrieren, prüfen wir in jeder Transition, ob er betätigt wurde oder nicht. Falls er betätigt wird, existieren Alternativzweige, die einen Sprung zu *Init* machen und die Ausgangssituation/die Startsituation wieder herzustellen.

3. Designentscheidungen

Natürlich muss der Ablauf dieses Systems auch den Ansprüchen der Realität gerecht werden, deswegen erklären wir hier kurz markante Entscheidungen, die wir getroffen haben und wieso.
Bei dem Notaus, ging es darum wie an einer richtigen Maschine den Pilzschalter zu simulieren. Dieser wird einmalig gedrückt und harkt sich fest, bis ein Techniker mit einem Schlüssel den Schalter wieder löst. Dies ist darin dargestellt, dass wir es dem Nutzer nach betätigen des Notaus, unmöglich machen die Maschine wieder anzumachen. Wir sperren die Motoren *K1* und *K2* und den *Global_START*, so ist das Gerät gesichert, bis ein Techniker den Notaus löst.
Das nächste was zu beachten war, war das die Maschine nach der ersten 90° Wende stoppt und erst beim nächsten Durchlauf die nächsten 90° überwindet. Wir haben hierzu in Ablauf die Variable *„angle“* eingeführt, im deutschen Winkel. Diese gibt uns den Winkel, den der Teller gerade annimmt. Hierfür haben wir die 4 Teller um den Mittelpunkt rotieren lassen um eine gleichmäßige Rotation von allen Tellern/Stationen zu bekommen. Der Winkel wurde den Objekten in ihrer *"absolute movement"* Einstellung übergeben und er wird im Schritt *Teller_dreht* gedreht. Nach einer 90° Drehung wird ein boolescher Wert namens *"Rotate"* auf TRUE gesetzt, um eine weitere Drehung zu verhindern. Nach 360° wir der Winkel wieder auf 0 gesetzt. Wenn eine 90° Drehung vollzogen wurde, wird *S1* wieder auf TRUE gesetzt.

Für die Schutztür haben wir innerhalb der Visualisierung ein kleines Tor gebaut, das an die Sensoren andockt. Dazu haben wir innerhalb des Netzwerks *"Ablauf"* einen parallelen Schritt zu *Station1* erstellt, der *Tor_zu* heißt. Hier wird die Schutztür geschlossen. Danach haben wir im Schritt *Init* das Tor nach der Transition oben wieder geöffnet. Für die Bewegungen haben wir innerhalb des Tors in der Visualisierung in den Einstellungen *"absolute movement"* den Y-Offset verändert, hierfür gab es auch wieder eine Variable namens *"tor"* als INTEGER.

Die Nummerierungen an den Stationen am Drehteller zeigen mithilfe des Pfeils welche Station gerade zum Entnehmen oder Hineinlegen eines Werkstückes dran ist.

Autoren: Nils Halwas, Matr-Nr.: 5204636 und Mehmet Algül, Matr-Nr.: 5273858
Modul: Leit-, Steuerungs- und Kommunikationstechnik (LSK)
Professor: Bernd Müller