Ein Zustandsautomat im µMSR-System

(Stand 09.2021, eine Teilbeschreibung des µMSR50 Systems: https://ing-brauns.de/umsr50)

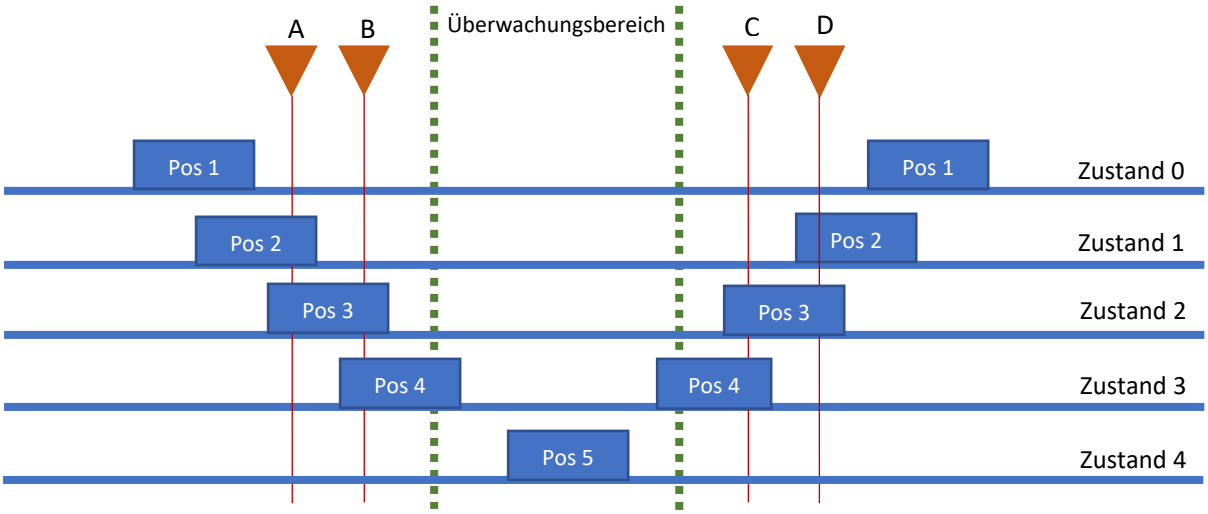
Beispiel: Überwachen, ob ein Bereich einer Förderstrecke frei ist

Es soll ermittelt werden, ob sich ein Fördergut in einem bestimmten Bereich einer Förderstrecke befindet. Besonderheit ist, dass sich die Förderrichtung nach Belieben ändern kann. Um dieses sicher zu erkennen sind jeweils zwei Initiatoren / Lichtschranken vor bzw. nach dem Überwachungsbereich notwendig. Ohne die Information der Förderrichtung mit einzubinden, wäre sonst nicht zu erkennen, ob sich ein Fördergut bei einer Richtungsänderung genau unter einem Initiator nach rechts oder links bewegt.

Vorab einige Festlegungen für dieses Beispiel:

- Ein Fördergut ist kleiner als der Überwachungsbereich
- Der Abstand der einzelnen Fördergüter ist größer als der Überwachungsbereich, es wird ausgeschlossen, dass sich mehrere Objekte im Bereich befinden
- Für die Überwachung spielt die Förderrichtung keine Rolle

Schritt 1: System untersuchen und Zustände erkennen:



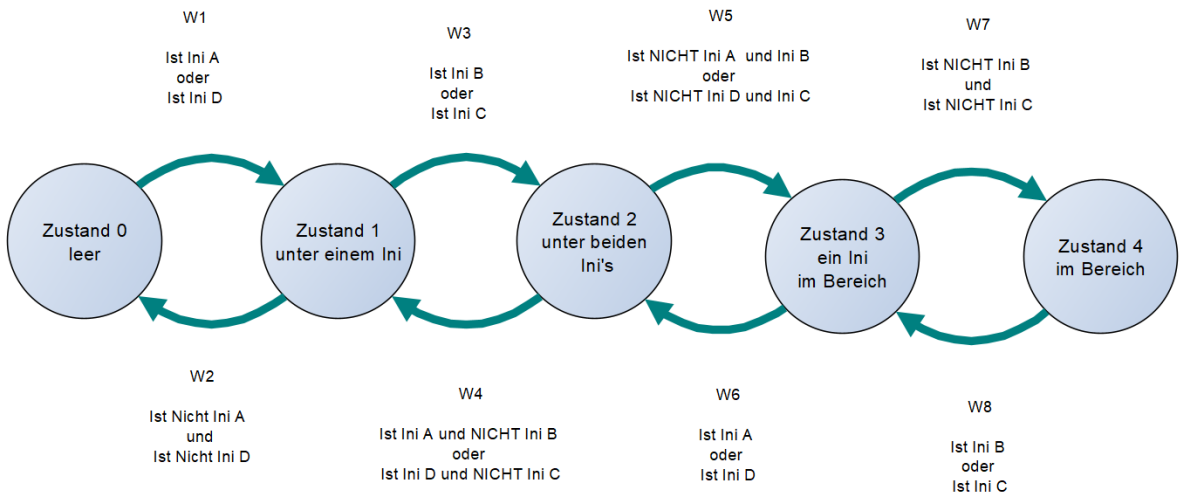
Zustände:

0. Der Überwachungsbereich ist leer, es befinden sich nur Objekte außerhalb
1. Ein Fördergut befindet sich unter der ersten linken oder rechten Lichtschranke
2. Ein Fördergut befindet sich unter den beiden Lichtschranken einer Seite
3. Ein Fördergut befindet sich unter der inneren linken oder rechten Lichtschranke
4. Ein Fördergut befindet sich im Überwachungsbereich

Schritt 2: Einen Zustandsautomaten entwerfen

Ein Zustandsautomat ist ein abstraktes Modell der Automatisierung. Ein System hat mehrere Zustände. Unter bestimmten Bedingungen werden die Zustände gewechselt. Befindet sich das System in einem bestimmten Zustand, werden entsprechende Aktionen ausgeführt.

Für unser Beispiel haben wir 5 Zustände ermittelt. Zustände werden z.B. als Kreis gezeichnet. Unter bestimmten Bedingungen werden die Zustände gewechselt. Zustandswechsel werden durch Pfeile gekennzeichnet und deren Bedingungen daran geschrieben. Für die Übersichtlichkeit bekommt hier jeder Zustandswechsel noch eine Bezeichnung. (Hier: W1 .. W8)



Hinweis:

Diese Aufgabe kann auf Papier oder in einem geeigneten Software-Tool durchgeführt werden. Ich verwende gern das Freeware-Tool „yEd“.

Hinweis:

Die Analysen und Beschreibungen von Aufgaben, wie hier gezeigt sind erstmal universell und haben nichts mit dem µMSR-System zu tun.

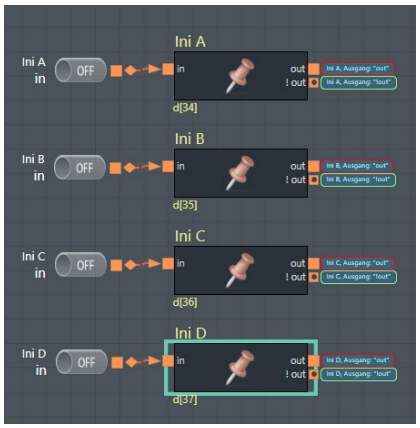
Jetzt geht es an die Umsetzung. Mann kann dieses natürlich einfach in einem C-Programm umsetzen und auf einem Controller aufspielen. Vorausgesetzt man verfügt über das nötige Knowhow.

Alternativ kann man auch versuchen diese Struktur mit SPS, unter zu Hilfenahme von UND/ODER/NICHT/FLIPFLOP’s/Zeitverzögerungen/ … aufzubauen? Natürlich würde der klassische Weg auch mit der µMSR-SoftSPS möglich sein.

Oder man nutzt die Zustandsautomatenfunktionen der µMSR-Soft-SPS wie im Folgenden gezeigt.

Schritt 3: Umsetzung der Automatisierung direkt als Zustandsautomat

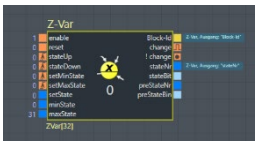
Ein kleine Schritt für Schritt-Anleitung mit einigen zusätzlichen Erläuterungen.



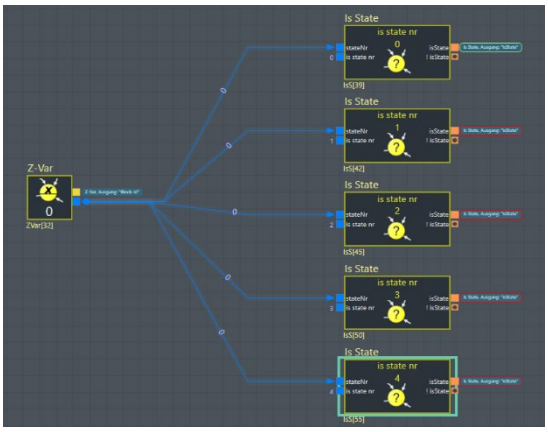
Je nachdem welche Hardware man nutzt, hat man diverse digitale Eingänge zur Verfügung um Initiatoren anzuschließen. Für dieses Beispiel brauchen wir erstmal eine konkrete Hardware, wir wollen die Aufgabe einfach nur mal simulieren können.

Dazu werden 4 dig. Merker in den Arbeitsbereich gezogen. Ein dig. Merker entspricht dann einem Initiator. Die Merker werden wie zuvor festgelegt mit Ini A … Ini D beschriftet. Durch einen Doppelklick auf den Eingang eines Merkers, werden im Designer Schaltflächen hinzugefügt, mit dem die Werte der Merker gesetzt werden können.

Als nächstes wird für den geplanten Zustandsautomaten ein Zustandsvariablen-Block hinzugefügt. Die vielen Anschlüsse mögen kompliziert erscheinen, jedoch benötigen wir nur zwei von diesen. Das ist die Ausgabe der aktuellen Zustandsnummer und den gelben Anschluss, mit dem später auf diese Variable zugegriffen werden kann. Mit der Schaltfläche am Block können die nicht verwendeten Anschlüsse ausgeblendet werden, was die Darstellung erheblich übersichtlicher erscheinen lässt.

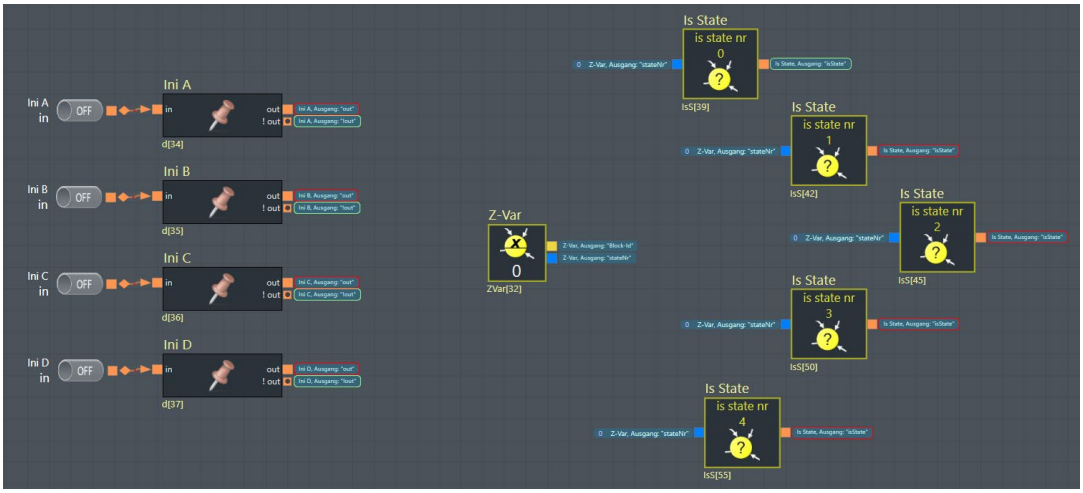


Für die Zustandswechsel benötigen wir noch die Information, in welchem Zustand sich das System gerade befindet. Eine von vielen Möglichkeiten ist die „Is State“-Funktion. Davon werden jetzt 5 Blöcke in den Arbeitsbereich gezogen und mit der Zustandsvariablen verbunden. Dann bekommt jede dieser Funktion noch die Info, bei welchem Zustand der Ausgang auf 1 gesetzt werden soll.



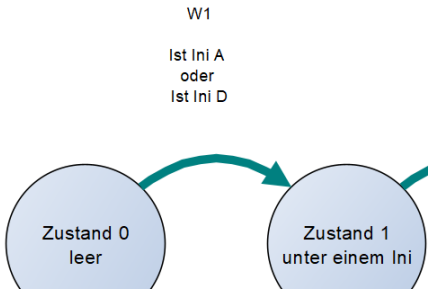
Ein Doppelklick auf eine Verbindung blendet diese aus und zeigt stattdessen das entsprechende Label an. Auch hier kann jetzt auf die vereinfachte Darstellung umgestellt werden.

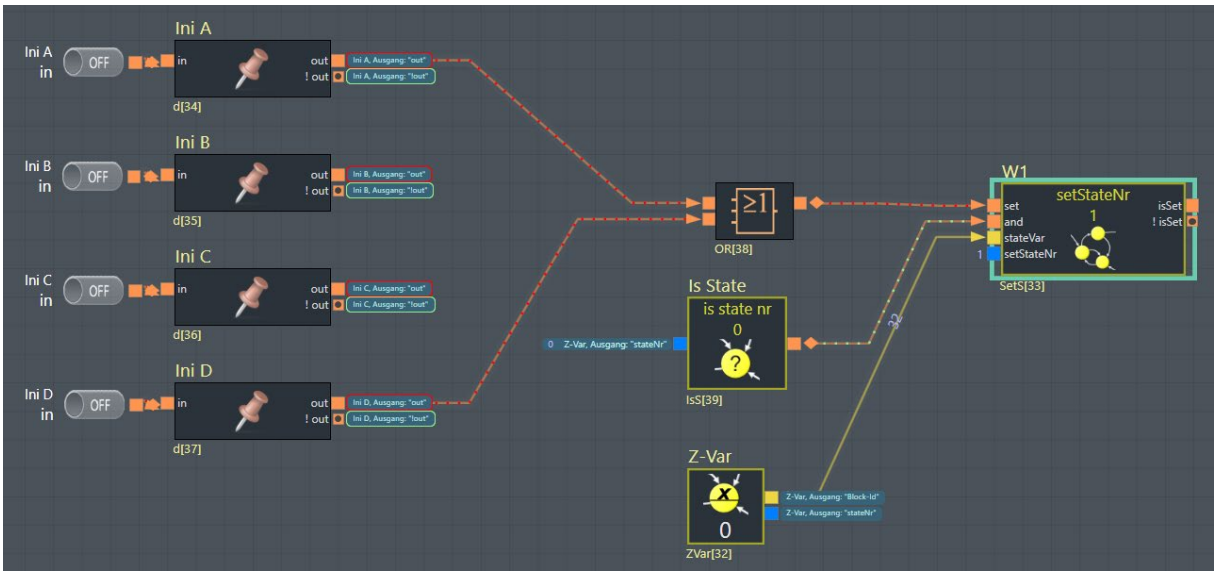
Zusammengefasst der Stand bis jetzt:



Um jetzt den Zustandsautomaten zum Leben zu erwecken, fehlen nur noch die Zustandswechsel.

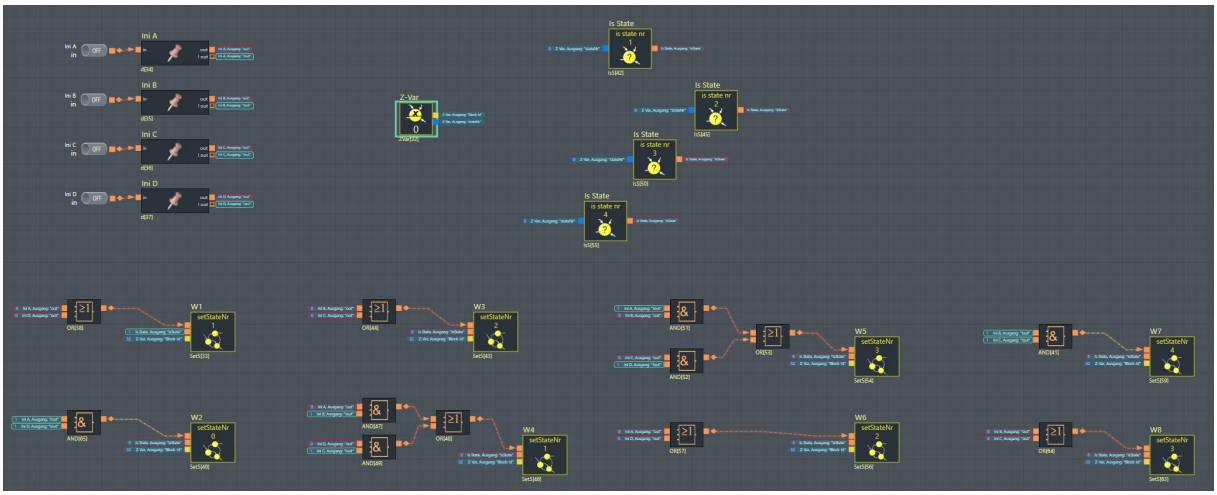
Hierzu fügen wir eine Zustandswechselfunktion „setStateNr“. Diese hat einen Eingang, auf dessen Flanke der Zustandswechsel durchgeführt werden. Dieses nur, wenn der „and“ Eingang ein EIN Signal hat. An diesem „and“ Eingang kommt jetzt die Überprüfung, in welchem Zustand sich das System befindet. Weiterhin benötigt diese Funktion noch eine Verbindung zu einer Zustandsvariablen (es könnte ja mehr als ein Zustandsautomat vorhanden sein) und noch eine Angabe, in welchen Zustand das System wechseln soll. Für den ersten Zustandswechsel W1 könnte es dann so aussehen:





Die anderen Zustandswechsel werden genau so hinzugefügt, ordentlich beschriftet und sinnvoll angeordnet. Der Übersichtlichkeitshalber werden die meisten Verbindungen nicht als Linie, sondern als Label dargestellt.

Das ist der gesamte Automat. Die Verbindung dieses Diagramms zu den Grafiken aus der Entwurfsphase ist zu erkennen, so dass diese Automation gut dokumentiert und nachvollziehbar ist.



Testen

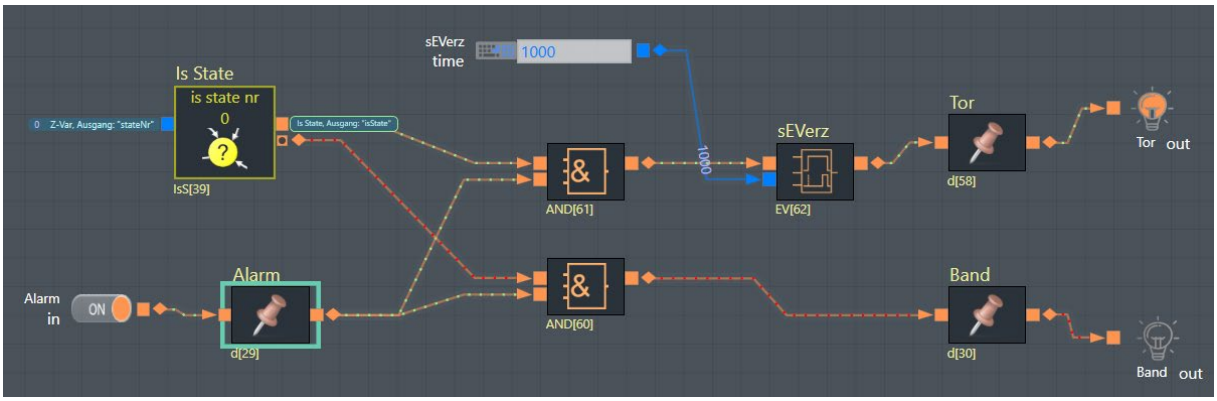
Über die Schaltflächen an den Merkern können jetzt die Funktionen getestet werden. Dazu werden jetzt die Schaltflächen nach und nach, wie Sie im Betrieb von den Initiatoren gesetzt werden, manuell geschaltet. Dabei kann man die Zustandsvariable überwachen und sehen, ob das gewünschte Verhalten umgesetzt wurde. Will man diese Automation auf eine reale Hardware übertragen, müssen nur noch die Merker gegen digitale Eingänge ausgetauscht werden.

Hier ein Beispiel, wie ein Fördergut durch den Überwachungsbereich bewegt wird:

Eine kleine Brand-Schließ-Automatik

Möchte nun eine folgende Automatisierung wissen ob sich kein Fördergut im Bereich befindet, muss diese nur prüfen ob die Zustandsvariable gleich 0 ist.

Zum Testen werden wieder 3 Merker hinzugefügt. Einer wird anstelle eines dig. Eingangs für einen Brandalarm verwendet. Die anderen beiden Merker repräsentieren zwei dig. Ausgänge. Der eine zum Ansteuern und Freifahren der Förderanlage und der andere schließt im Brandfall das Tor. Im folgenden Beispiel wird das Torsignal um 1000 ms verzögert.



Das µMSR-SoftSPS – System bietet jetzt natürlich noch unzählige weitere Möglichkeiten.

Wie zuvor könnten jetzt z.B. noch:

- Signalausgaben konfiguriert werden,
- Zähler an die verschiedenen Signale angebunden werden
- Laufzeiten von Förderanlagen und Tor schließen erfasst werden
- Alles geloggt werden (in einem internen Speicher, auf SD Karte, in die Cloud, usw.)
- Weitere Taster für z.B. manuellen Betrieb hinzugefügt werden
- Fehlerprüfungen mit Warnausgabe eingerichtet werden
- Usw.

Die Automatik ist fertig, und kann so auf eine Hardware mit min. 5 dig. Eingängen und zwei dig. Ausgängen laufen.

Ich bedanke mich für Ihr Interesse und freue mich über jede Kritik und Anregung.

Sie erreichen mich unter info@ing-brauns.de