Spezifikation der Systemarchitektur		
Version	2	15.05.18
Autor	Kessener	
Index	1 Überblick 2 Das Base System 3 Die Kommunikationslayer 4 Die Master Logik	

1 Überblick

Unsere Systemarchitektur ist grundlegend eine *Layered Architecture*. Sie besteht aus 4 Layers:

4	Master Logic
3	Inter-Module Communication
2	Base System
1	HAL

Das System beinhaltet zwei unabhaengige module (“Fließbänder”), die durch eine serielle Schnittstelle miteinander verbunden sind. Programmlogik kann in zwei Teilbereiche separiert werden: Grundlegende Logik, die in beidem Modulen identisch angewandt wird (zB tracking eines Werkstückes über das Fließband) und System-Logik, die sich auf das Zusammenspiel beider Module bezieht (zB Sicher zu stellen, dass sich nie mehr als ein Werkstück auf Modul 2 befindet). Die Systemlogik wird durch Anwenden des *Master-Slave* Patterns realisiert. Dies bedeutet, dass der Layer 4 (Master Logic) sich nur auf dem Master System befindet. Das Slave System hat keine eigene Layer 4 Implementierung.

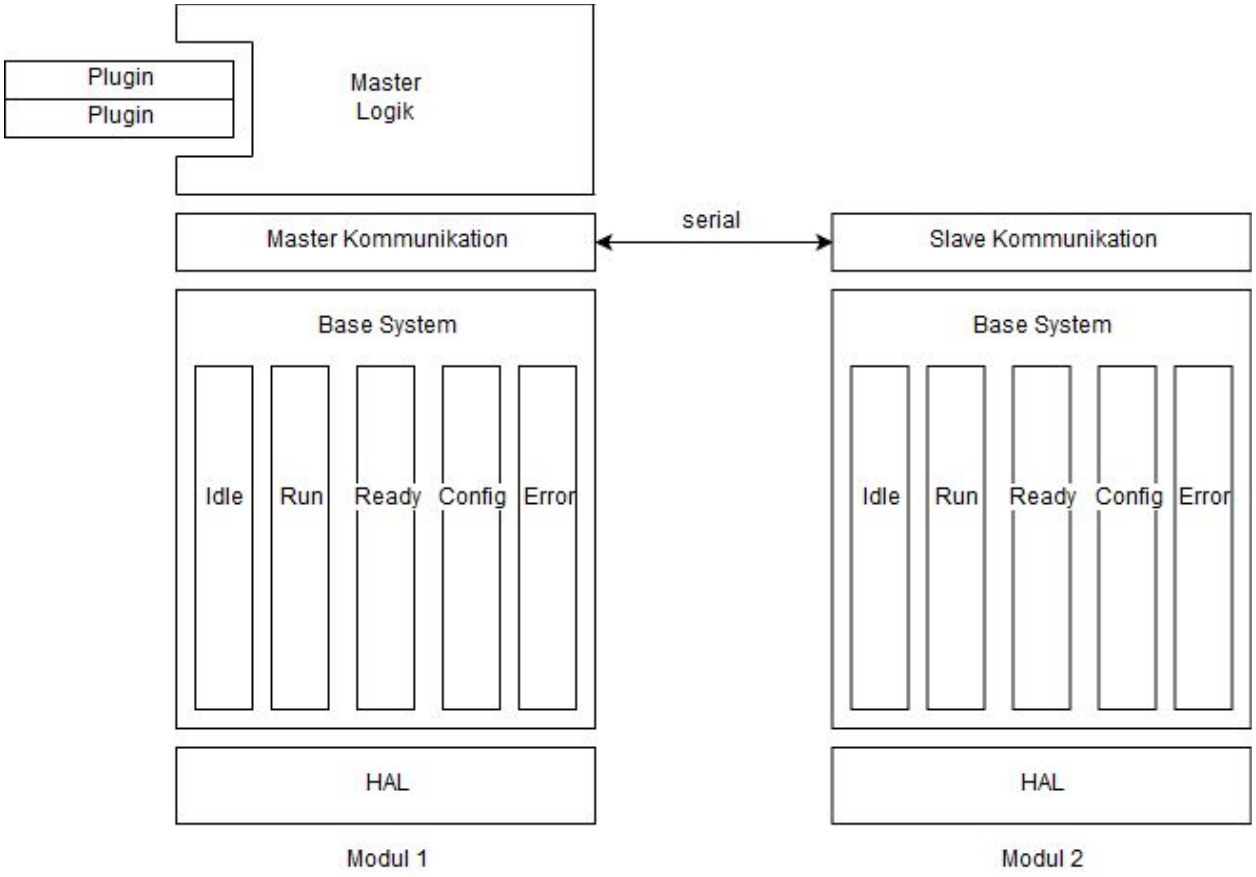

Diagramm 1 - Übersicht der Systemarchitektur

2 Das Base System

Das Base System kommuniziert direkt mit der HAL. Es steuert die Aktorik mittels Wrapper Klassen. Änderungen in der Sensorik werden durch Events von der HAL an das Base System weitergegeben. Zudem hat das Base System Zugriff auf programmierbare Timer Events, die es dem Base System ermöglichen auf das Ausbleiben erwarteter Sensor Events zu reagieren (also zB auf das außerplanmäßigem Entfernen eines Werkstücks bzw. das daraus resultierende Nicht-Unterbrechens einer Lichtschranke.)

Das Base System ist völlig agnostisch gegenüber der Existenz mehrerer Module; es weiss nur von sich selbst, der HAL unter ihm und dem Master über ihm. Kommunikation zwischen den Base Systemen und der Master Logik erfolgt durch *Message Passing*. Dabei diehnt der Kommunikationslayer als Vermittler zwischen den beiden Base Systems (die voneinander nichts wissen) und dem Master, und routet Messages über die serielle Verbindung.

Im laufenden Betrieb kümmert sich das Base System um das Routing von Werkstücken über das Fließband; dabei achtet es darauf, dass kein Werkstück verschwindet oder unerwartet Auftaucht. Es Benachrichtigt den Master beim Auftauchen eines neuen Werstücks am Anfang des Fließbandes, beim Erreichen des Ende des Fließbandes und beim Erreichen des Switches. Der Master kontrolliert wann das Fließband sich bewegt und wann es steht. Er entscheidet auch, ob ein Werstück durchgelassen wird oder aussortiert werden soll. Hierzu dienen Sensordaten, die vom Base System wärend des Transport des Werstücks gesammelt wurden und mit der Nachricht “Werkstück am Switch” an den Master geschickt werden.

Das Verhalten des Base Systems ist abhängig vom Gesamtzustand des System und kann in separate Abschnitte eingeteilt werden. Diese werden unter Andwendung des *Plugin* Patterns (auch *Micro Kernel* genannt) implementiert. Dabei wird - von der Master Logik gesteuert - immer je ein Plugin (was in diesem Kontext *Manager* genannt wird) als der aktive Manager eingesetzt. Aus den verschiedenen Zuständen in dem sich das System befinden kann ergeben sich folgende Manager:

State	**Manager**	**Beschreibung**
Idle	IdleManager	Startzustand des Systems. Kann mittels der START Taste in den *Run* oder *Config* Zustand überführt werden.
Run	RunManager	Betriebszustand. Erfordert eine valide Kalibrierung. Diese kann aus einer Konfigurationsdatei geladen sein oder im *Config* Zustand erstellt werden.
Calibration	ConfigManager	Selbstkalibrierung.
Error	ErrorManager	Fehlerzustand. Kann als einziger Manager auch von etwas anderem als der Master Logik instanziiert werden: Im Falle eines Abbruchs der seriellen Kommunikation veranlasst der Kommunikationslayer des Slave Systems einen entsprechenden Zustandswechsel.

3 Die Kommunikationslayer

Die beiden Kommunikationslayer der Module sind aus Sicht des Masters und des Base Systems völlig transparent. Der Master denkt, er spricht direkt mit den beiden Base Systemen wärend diese denken, dass sie direkt mit dem Master sprechen. Die Kommunikationslayer übernehmen die Aufgabe eines Vermittlers, der Nachrichten des Masters an den jeweiligen

Empfänger weiterleitet (wenn notwendig auch über die serielle Schnittstelle), und die Nachrichten der Base Systeme mit korrekten Absendern bestückt, sodass der Master erkennen kann, woher die Nachricht gekommen ist.

Außerdem sind sie dafür verantwortlich zu erkennen, wenn die serielle Verbindung unterbrochen wurde, und ihr jeweiliges Base System darüber zu informieren.

4 Die Master Logik

Der Zustand des gesamten Systems wird im Master modelliert.

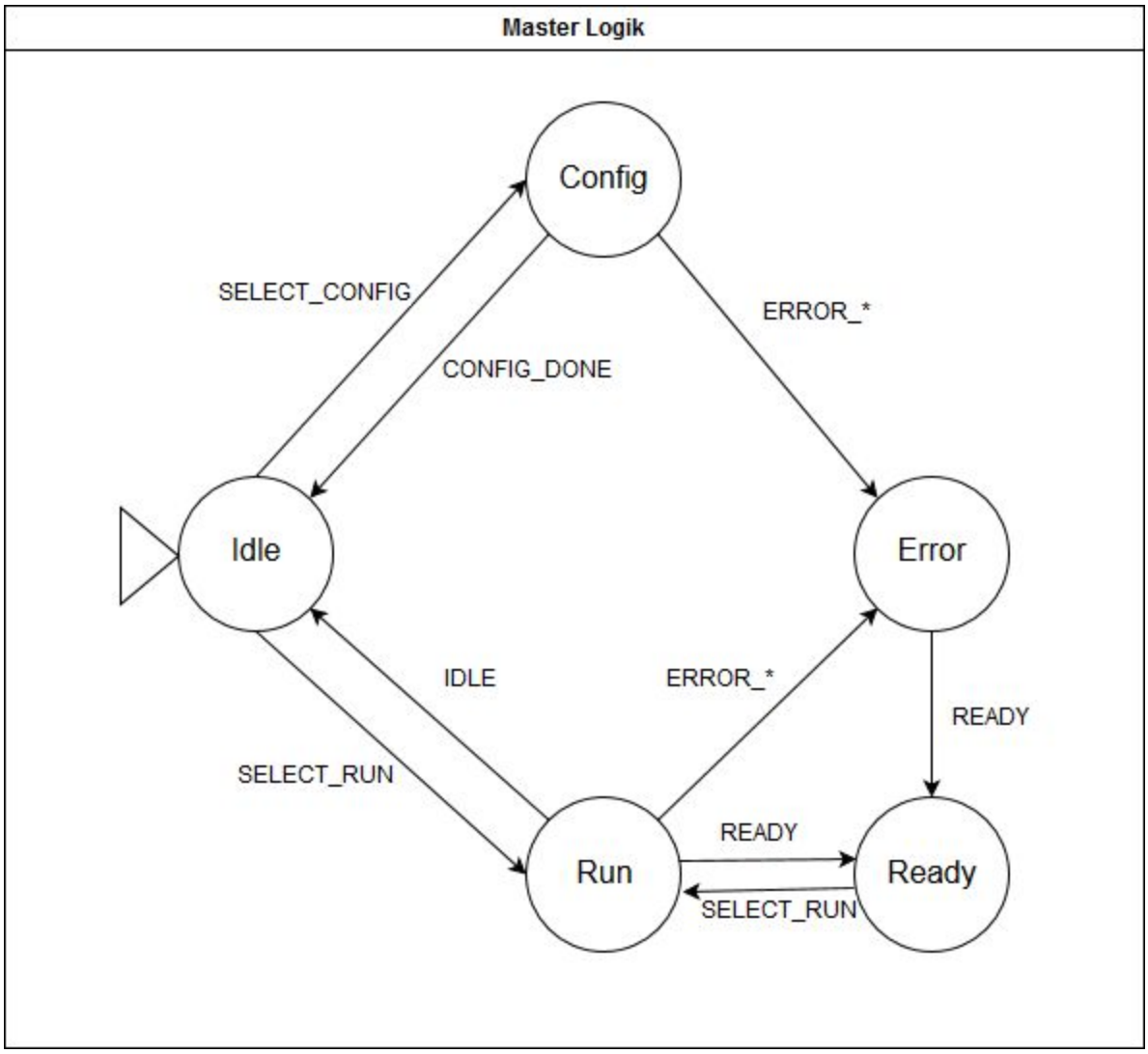

Diagramm 2 - Master Logik State Machine - Top-Level View

Bei der Identifizierung der Werstücke wird das *Plugin* Pattern verwendet. Die Sensordaten, die das Base System gesammelt hat, werden an eine Reihe von Plugins

übergeben. Das Plugin, das den größten Grad der Übereinstimmung des Plugin-Internen Modells mit den Sensorwerten meldet, bestimmt was mit dem Werstück passieren soll. Dies wird von der Master Logik nur als ein Vorschlag gewertet; es allein trifft die entgültige Routing Entscheidung (abhängig von Faktoren wie zB die Fülle der Rutschen).

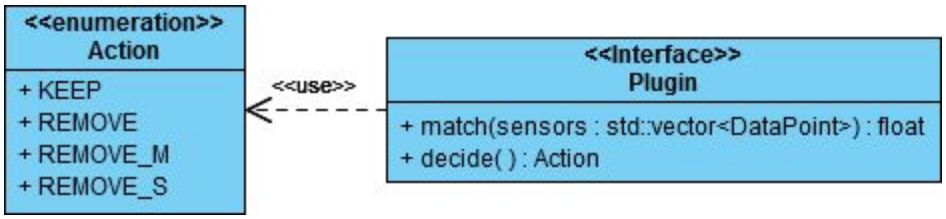

Diagramm 3 - Klassendiagramm der Plugins

Sollte ein Werkstück als verschwunden gemeldet werden, dann re-evaluiert der Master die Sortierentscheidung aller nachfolgenden Werkstücke (unter Rücksprache mit deren Plugins, wenn bereits ein passendes gefunden wurde).

5 Appendix

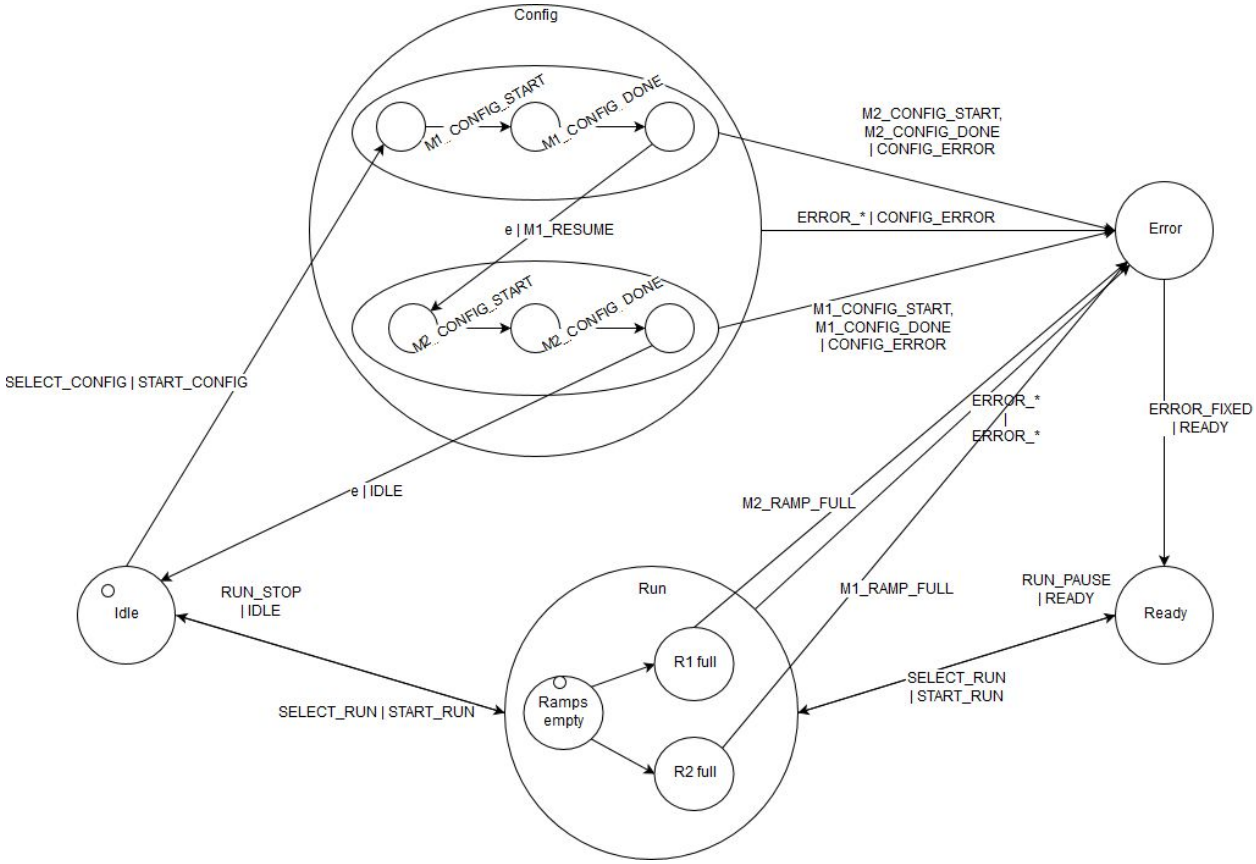

Diagram - Detailierte Darstellung der Master-Logik als hierarchische State Machine