(19) **Bundesrepublik Deutschland**
Deutsches Patent- und Markenamt

(10) **DE 100 17 708 B4** 2008.02.14

(12) **Patentschrift**

(21) Aktenzeichen: **100 17 708.5**
(22) Anmeldetag: **04.04.2000**
(43) Offenlegungstag: **18.10.2001**
(45) Veröffentlichungstag der Patenterteilung: **14.02.2008**

(51) Int Cl.[8]: ***G05B 19/042*** (2006.01)
G05B 15/02 (2006.01)

(73) Patentinhaber:
Technische Universität Dresden, 01069 Dresden, DE

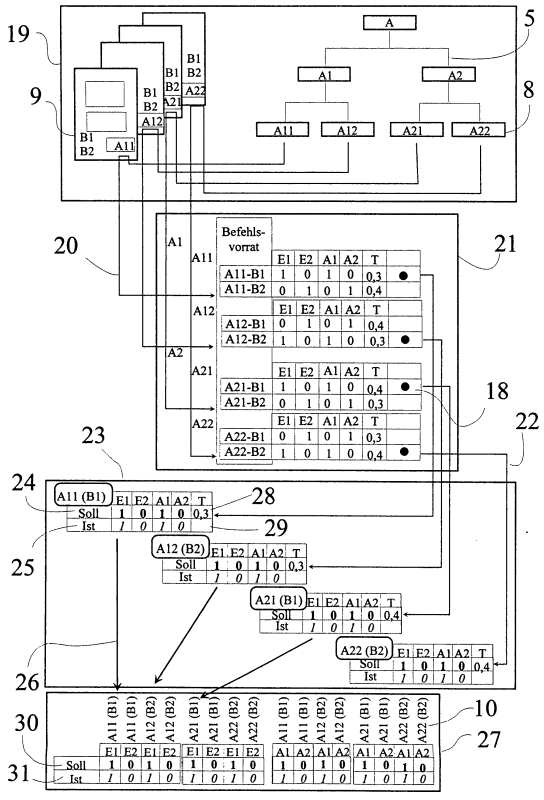
(74) Vertreter:
Sender, F., Dipl.-Ing., 01069 Dresden

(72) Erfinder:
Großmann, Knut, Prof. Dr.-Ing.habil., 01277 Dresden, DE; Möbius, Volker, Dipl.-Ing., 01069 Dresden, DE

(56) Für die Beurteilung der Patentfähigkeit in Betracht gezogene Druckschriften:
DE 197 41 959 A1
DE 195 13 801 A1
DE 44 07 334 A1
DE 37 43 438 A1
DE 691 21 712 T2
US 48 58 101

(54) Bezeichnung: **Verfahren zum Steuern von Mechanismen und technischen Systemen, Einrichtung und Steuerungssoftware**

(57) Hauptanspruch: Verfahren zum Steuern von Mechanismen oder technischen Systemen,
dadurch gekennzeichnet, dass
a) die zu steuernden Mechanismen oder technischen Systeme in ihren Elementarfunktionen (8) mit deren befehlsgemäß definierten Zuständen und den zugehörigen Signalvektoren (15) der Sensoren (13) und Aktoren (12) in der Steuerung gespeichert werden, wobei ausgehend von einem definierten Referenzzustand (18) zu Beginn der Steuerungsaktivierung ein ständiger Vergleich der von der technischen Anlage durch die Sensoren (13) gemeldeten Ist-Zustände mit den in der Steuerung gespeicherten Sollzuständen (24) für alle Elementarfunktionen erfolgt und damit jede Abweichung im zu steuernden System vom befehlsgemäßen Sollzustand (24) erkannt wird,
b) ein den Zustand der Mechanismen oder des technischen Systems verändernder neuer Elementarbefehl (16) mit seinem Start den Sollzustand (24) für den Vergleich aktualisiert und auf der Grundlage ebenfalls gespeicherter zulässiger Kontrollzeiten (17) die Zeit bis zur Rückmeldung des befehlsgemäßen neuen Zustandes überwacht,
c) wobei Sensorsignale und vergleichbare Informationen ausschließlich der...



Beschreibung

[0001] Die Erfindung bezieht sich auf ein Verfahren zum Steuern von Mechanismen und technischen Systemen sowie auf die dafür zu gestaltenden Einrichtungen einer elektronischen Steuerung und ein Verfahren zur Erstellung der Steuerungssoftware.

[0002] Aus der DE 44 07 334 A1 ist ein Verfahren zum Erstellen und Darstellen von Steuerungen bekannt, mit dem sich Steuerungen auf einfache Weise graphisch entwerfen lassen. Die gewünschte Funktion der Steuerung wird als ereignisgesteuertes Netzwerk von Symbolen mit frei wählbaren Verbindungen graphisch in einen Computer eingegeben oder von einem Computer dargestellt. Das Netzwerk in maschinenlesbarer Form umgewandelt, kann von dem Computer oder einem separaten Steuerungsrechner als Steuerungsprogramm verwendet werden. Das Verfahren eignet sich für speicherprogrammierbare Steuerungen sowie DDC-Anlagen.

[0003] Aus der DE 195 13 801 A1 ist ein Verfahren zur automatischen Erzeugung einer Steuerung für einen Prozess bekannt, bei dem ein nicht deterministischer Automat, der alle physikalisch möglichen Verhaltensweisen der Steuerung beschreibt, festgelegt wird, bei dem die erlaubten Zustandsübergänge des von der Steuerung zu beeinflussenden Prozesses beschrieben werden, bei dem der Automat so eingestellt wird, dass er vorgegebene Sicherheitsbedingungen erfüllt, bei dem der Automat so eingestellt wird, dass er die Funktion des aus der Steuerung und Prozess bestehenden Systems erfüllt. Das Verfahren macht von der Programmiersprache CSLxt Gebrauch, um die Komponenten der Spezifikation des Systems zu beschreiben. Für die Spezifikation des Prozessmodells werden nicht die Zustandsübergänge detailliert beschrieben, sondern sogenannte vordefinierte qualitative Constraints verwendet, die zur automatisches Generierung der Steuerung dienen.

[0004] Nachteilig ist, dass die Beschreibung von Zustandsübergängen auf einem höheren Sprachniveau fehlerhaft sein kann, und eine nachträgliche Korrektur der Steuerung nicht ohne weiteres möglich ist.

[0005] Aus der Druckschrift DE 37 43 438 A1 ist es bekannt, dass bei speicherprogrammierbaren Steuerungen die auf eine Änderung mindestens einer Prozesseingangsgröße erfolgende Erzeugung von Steuer- oder Stellsignalen wesentlich beschleunigt werden kann. Hierzu werden vom Momentanzustand und von der Eingangsvektorbelegung abhängige, vorab erstellte Steuerdaten und Folgezustands-Codes aus spezifischen Datenspeicherbereichen und Teilbereichen ausgewählt und direkt oder nach kurzer Weiterverarbeitung als Ausgangsvektor ausgegeben.

[0006] In der DE 197 41 959 A1 ist ein System zur Verarbeitung von Ereignissen in technischen Prozessen mit einem verteilten Datenverarbeitungssystem beschrieben, das universeller und flexibler handhabbar sein soll. Es wird ein System bereitgestellt, welches kurz „Ereignisse" genannte Datenobjekte, die insbesondere Änderungen von technischen Zuständen in technischen Prozessen datentechnisch beschreiben und vollkommen unterschiedlich sein können bezüglich Ursprung, Aufbau und Inhalt, in einer übersichtlichen Weise zentral verarbeiten werden können. Die Flexibilität dieses Systems ist dadurch gekennzeichnet, dass an sich eine nicht begrenzte Anzahl an Controllerbausteinen im gesamten, verteilten Datenverarbeitungssystem aktiv sein können. Es muss allerdings gewährleistet sein, dass ein Ereignis, von dem ein Controllerbaustein abhängig ist bzw. nach dessen Eintritt ein Zugriff auf einen Aktor ausgeführt wird, in der Ereignisbibliothek des zentralen Steuerungsbausteines definiert ist.

[0007] Aus der Druckschrift DE 691 21 712 T2 ist eine graphische programmierbare Schnittstelle für Maschine-/Prozesssteuerungsgeräte bekannt. Gegenstand dieser Erfindung ist es, ein intuitives, einfach zu verwendendes Benutzerinterface zur Erstellung und Einrichtung sowie Programmierung einer programmierbaren Steuerungsvorrichtung vorzusehen.

[0008] Durch Auswahl eines Kompilierungsbefehls aus einem Menu wird die Kompilierung des Programms in eine ausführbare Code-Datei veranlasst. Die Datei kann gesichert oder auf einen Zielprozessor übertragen werden, in dem sie installiert wird. Falls ein Kontaktplan involviert ist, kann es auf eine Kontaktplananweisungsdatei übersetzt werden.

[0009] Bei keiner der Druckschriften DE 37 43 438 A1, DE 197 41 959 A1 und DE 691 21 712 T2 gelingt die Lösung der Steuerungsaufgabe ohne den Einsatz Boolscher-Algebra-Bedingungen.

[0010] Darüber hinaus sind
Speicherprogrammierbare Steuerungen SPS
Hardware-SPS
Software-SPS
Programmiersysteme und Programmiersprachen
Simatic S7
Programmierung nach IEC 1131-3 Norm
Standardprogrammiersprachen: Kontaktplan, Funktionsplan, AWL, Structured Text
bekannt.

[0011] Nachteilig bei dem Stand der Technik ist, dass im Prinzip mit Boolscher Algebra Bedingungen aus Eingängen (Sensoren) für das Setzen von Ausgängen (Aktoren) formuliert werden, die ständig zyklisch neu durchgerechnet werden. Dieser Programmieransatz ist historisch entstanden. Beleg oder In-

diz für diesen Zustand ist die Tatsache, dass nach der allgemein akzeptierten Norm der „Kontaktplan" noch als Programmiersprache verwendet werden kann.

[0012] Trotz aller CAE-Unterstützung durch Grafikoberflächen und Hochsprachen bleiben prinzipbedingte Grundmängel, wie die Unübersichtlichkeit des Programms und dessen individuelle Prägung vom Programmierer, nie vollständige Prüfbarkeit des Programms in seiner Funktionalität, da das Ergebnis der zyklischen Berechnungen von kombinatorischen und zeitlichen Zufälligkeiten beeinflusst werden kann und die schwierige Gestaltung von differenzierten Fehlerreaktionen.

[0013] Die Aufgabe der Erfindung besteht darin, eine Steuerung für Mechanismen oder technische Systeme anzugeben, die die Steuerungsaufgabe ohne den Einsatz Boolscher-Algebra-Bedingungen löst, wobei ein übersichtliches Programm, frei von individueller Prägung und mit vollständiger Prüfbarkeit vorliegen soll.

[0014] Erfindungsgemäß wird die Aufgabe durch ein Verfahren mit den im Anspruchs 1 genannten Merkmalen gelöst. Die Aufgabe wird weiterhin durch ein Verfahren zur Erstellung einer Steuerungssoftware mit den im Anspruch 11 und durch eine Einrichtung mit den im Anspruch 16 aufgeführten Merkmalen gelöst.

[0015] Vorteilhafte Ausgestaltungen und Weiterbildungen sind Gegenstand zugehöriger Unteransprüche.

[0016] Das Wesen der Erfindung besteht darin, dass abgeleitet von der Funktionalität des zu steuernden Mechanismus oder technischen Systems, insbesondere mit dessen Entwicklung, mit technischen Mitteln die Funktionalität der zu steuernden Einrichtung in einem als Steuerung gestalteten Steuerungscomputer als ein vollständiges Abbild des befehlsgemäßen Sollzustand des Systems abgelegt, verwaltet und aktualisiert wird und ein Vergleich dieses Sollzustandes über die gemeldeten Sensorsignale mit dem Ist-Zustand der technischen Einrichtung erfolgt. Dieser Soll-Ist-Vergleich erfolgt ständig für alle Sensorsignale des zu steuernden Systems. Bei Abweichungen des Ist-Zustandes zum Sollzustand werden vorbereitete Algorithmen abgearbeitet und ebenso vorbereitete zweckmäßige Entscheidungen aktiviert. Jedes Sensorsignal wird damit nur mit genau einem Sollsignal verglichen und dieser Vergleich dient ausschließlich der Zustandsidentifikation des technischen Systems. Zustandsänderungen erfolgen ausschließlich über Befehle auf sprachlichfunktionellem Niveau. Diese Befehle werden in einem besonderen Bereich der Steuerung verwaltet, bei Start eines Befehls wird der Sollzustand im Abbild aktualisiert und die dem Befehl entsprechende Änderung des Ist-Zustandes des technischen Systems innerhalb einer vorgegebenen Zeit kontrolliert.

[0017] Die zu steuernden Einrichtungen sind in ihren Elementarfunktionen mit deren befehlsgemäß definierten Zuständen und den zugehörigen Signalbildern der Sensoren und Aktoren in der Steuerung gespeichert, wobei ausgehend von einem definierten Referenzzustand zu Beginn der Steuerungsaktivierung ein ständiger Vergleich der von der technischen Anlage durch die Sensoren gemeldeten Ist-Zustände mit dem in der Steuerung gespeicherten Sollzustand für alle Elementarfunktionen erfolgt und damit jede Abweichung im zu steuernden System vom befehlsgemäßen Sollzustand erkannt wird, wobei ein den Zustand des technischen Systems verändernder neuer Befehl mit seinem Start den Sollzustand für den Vergleich aktualisiert und auf der Grundlage ebenfalls gespeicherter zulässiger Übergangszeiten die Zeit bis zur Rückmeldung des befehlsgemäßen neuen Zustandes überwacht, wobei Sensorsignale und vergleichbare Informationen ausschließlich der Zustandsidentifikation dienen und Zustandsänderungen ausschließlich über den Start von dafür auf logisch-funktionellem Sprachniveau frei definierten Befehlen erfolgt, denen die mit Sensor- und Aktor-Signalen definierten Elementarbefehle zugeordnet sind.

[0018] Vorteilhaft werden in einem als EF-Controler bezeichneten Programmbaustein die Zustände aller Elementarfunktionen als aktueller Sollzustand mit den zugehörigen Aktoren und Sensoren geführt und damit jede über die Sensoren erkannte Zustandsänderung des technischen Systems auf Übereinstimmung mit dem in der Steuerung geführten Sollzustand bewertet.

[0019] Ein nicht dem Soll entsprechender Zustand einer Elementarfunktion des zustandsbeschreibenden Signalbildes wird vorteilhaft an einen als „Nichtsollbewerter" bezeichneten Programmbaustein übergeben, in dem für ausgewählte Zustände von Elementarfunktionen Reaktionsbefehle gespeichert sind, die bei Übereinstimmung mit dem zur Prüfung übergebenen Zustand gestartet werden, wobei in allen Fällen differenzierte Fehlermeldungen erzeugt werden.

[0020] Einem Befehl als Befehlssatz werden sowohl die neuen Sollzustände der Sensoren und Aktoren, die Übergangszeiten zum neuen Sollzustand als auch die bei Abweichungen zu startenden Reaktionsbefehle, jeweils unterschieden in vor dem Start und nach erfolgter Ausführung zu löschende und zu setzende Reaktionsbefehle auf ausgewählte Zustandsmeldungen zugeordnet, wobei vorteilhaft ein als „Befehlsaufbereiter" bezeichneter Programmbaustein die dafür erforderliche Organisation im System übernimmt und in diesem Programmbaustein auch die Freigabe eines nächsten Befehls bei Befehlsfolgen

nach Erfüllungsmeldung des vorhergehenden sowie die Organisation von Parallelbefehlen durch je nach Bedarf temporäres Eröffnen von parallelen Abarbeitsfolgen realisiert wird.

[0021] Vorteilhaft werden dem organisierten Steuerungssystem Sensorsignale und weitere zu kontrollierende Informationen in einem hier als „Zustandsüberwacher" bezeichneten Programmbaustein zu einem lückenlosen Datenwort zusammengezogen, wobei den Signalen die Adresse der zugehörigen Elementarfunktion im EF-Controler zugeordnet bleibt und für den Vergleich jedem Sollsignal das Ist-Signal in gleicher Struktur gegenübersteht, was einen programmtechnisch sehr effektiven Soll-Ist-Vergleich ermöglicht, wobei eine aufgetretene Abweichung eines Signals nach der Übergabe zur Auswertung als neuer Vergleichszustand eingetragen wird und damit ein Vergleich immer zum letzten ausgewerteten Zustand erfolgt und jede Zustandsänderung damit nur einmal ausgewertet wird, wobei der Vergleich der Soll-Ist-Signale gerichtet erfolgt und nach einer Unterbrechung für die Auswertung einer Abweichung der Vergleich bei dem der Unterbrechungsstelle folgenden Signal fortgesetzt wird, wodurch gesichert ist, dass jede zeitlich hinreichend lange Zustandsänderung erfasst und ausgewertet werden kann.

[0022] Bei einem so organisierten Steuerungssystem wird jede im Programmbaustein Zustandsüberwacher erfasste Zustandsänderung in einem Ereignis-Zeit-Protokoll gespeichert, wodurch auf einfachstem Weg damit beschriebene Prozessparameter zugänglich werden, damit auch beispielsweise Signalschwingungen erkannt und gegebenenfalls ausgefiltert werden können.

[0023] Die den Echtzeitforderungen unterliegenden Programmbausteine Befehlsaufbereiter, EF-Controler, Zustandsüberwacher und Nichtsollbewerter werden vorteilhaft zu einer Funktionseinheit zusammengefasst, die als „Ausführungsrechner" bezeichnet wird, wozu ein spezieller Prozessor eingesetzt wird, während die nur auf logisch-funktionellem Sprachniveau formulierten Befehle der eigentlichen Nutzungsprogramme in einer zweiten, nicht Echtzeitforderungen unterliegenden Funktionseinheit, die als „Befehlsrechner" bezeichnet wird, organisiert werden, wobei der Befehlsrechner zweckmäßigerweise bei größerem und variablen Befehlsumfang über einen eigenen Prozessor verfügt und hier auch die Kommunikation komfortabel gestaltet werden kann.

[0024] Vom Befehlsrechner an den Ausführungsrechner übergebene Befehle werden dort vorteilhaft ohne Prüfung ausgeführt, wobei der Ausführungsrechner jeweils die auszuführende Aktion autark realisiert. Deshalb werden im Befehlsrechner auf logisch-funktionellen Befehlsniveau zu den sich ausschließenden Zuständen Sperrenverzeichnisse geführt und verwaltet, die den prozess- und maschinenseitig determinierten Anteil von Verriegelungen übernehmen, wobei hier im Befehlsrechner mit einem Nutzungs-Prozessbefehl außer den Informationen, welche Befehle dem Ausführungsrechner zu übergeben sind, auch festgelegt wird, für welche anderen Nutzungsbefehle Sperren während oder nach der Ausführung zu setzen oder aufzuheben sind.

[0025] Der Ausführungsrechner kann einen erhaltenen Befehl autark ausführen, wobei der Befehlsrechner dem Ausführungsrechner den nächstfolgenden geprüften Befehl in einem Befehlspuffer als Zwischenspeicher bereitstellt und nach dem Bereitstellen den Zustand im Befehlsrechner auf den Stand aktualisiert, der nach der Ausführung dieses bereitgestellten Befehls eintreten wird und damit eine Prüfung des nachfolgenden Befehls im Befehlsrechner bereits während Befehlsausführung des vorhergehenden Befehls im Ausführungsrechner erfolgt und damit in der Regel ein schnellerer Programmablauf realisiert werden kann. Nicht verträgliche Befehle werden bereits im Befehlsrechner als nicht zulässig erkannt und ausgewiesen und es erfolgt kein Start eines solchen Befehls. Ist der vorbereitete Befehl zulässig, tritt bei fehlerfreier Ausführung der für die Prüfung des Befehls im Befehlsrechner erwartete Zustand ein und der Ablauf wird fortgesetzt, während bei einem Fehler ein Rücksetzen auf den Zustand zum laufenden Befehl als Fehlerzustand vorgenommen wird.

[0026] Der Anwender dieser Steuerung wird bei der Erstellung eines Steuerungsprogramms vorteilhaft durch ein Entwicklungsprogramm dialoggeführt unterstützt, wobei die erste Beschreibung des zu steuernden Systems die Angabe der hierarchischen Funktionsstruktur des Systems verlangt, das jeweils untere Ende dieser Struktur als Elementarfunktion betrachtet wird und jede Elementarfunktion ebenfalls im Dialog in ihren Befehlszuständen zu definieren ist und diesen definierten Befehlen die Sensorsignale, die Aktoren, die Kontrollzeiten für den Übergang zwischen den befehlsgemäßen Zuständen und ein Referenzzustand für den Beginn zuzuordnen sind, gleichermaßen die Definition der Einbindung komplexerer Teilsysteme erfolgen kann, wobei der Anwender des Steuerungssystems nur die hier genannten primären Angaben macht und das Steuerungs-Entwicklungsprogramm daraus den System-Elementarfunktionsspeicher, den EF-Controler und den Signal-Vektor für den Zustandsüberwacher generiert und damit das technische System bereits inbetriebgenommen, auf fehlerfreie Signaldefinition im Referenzzustand überprüft, mit den definierten Elementarfunktionen gesteuert und soweit zulässig in Einzelbefehlen getestet und geprüft werden kann.

[0027] Für ein solches dialoggestütztes System werden die Nutzungsbefehle vorteilhaft derart erarbeitet, indem in einer Befehlsbibliothek prozessnah

zu definierenden Nutzungsbefehlen aus den vorher definierten Elementarbefehlen solche einzeln, parallel oder als Folge zugeordnet werden, dazu die Sperrbedingungen auf Befehlsniveau im Befehlsrechner und für den an den Ausführungsrechner zu übergebenden Befehlssatz auch die in den Nichtsollbewerter einzutragenden Reaktionsbefehle auf ausgewählte Abweichungen, verbunden mit geeigneten Fehlermeldungen, festgelegt werden.

[0028] Für ein so aufgebautes Steuerungssystem bleiben Änderungen an Elementarfunktionen lokal begrenzt. Es können jederzeit und ebenfalls mit überschaubarer lokaler Wirkung neue Nutzungsbefehle, Sperrbedingungen oder Fehlerreaktionen erweitert oder geändert werden oder ohne jede Rückwirkung auf schon definierte Programme, dazu unterschieden durch die Vergabe von Statusinformationen für das System, eine neue Zuordnung von Befehlen und Befehlsbedingungen erfolgen.

[0029] Jedes so erstellte Programm ist in seiner logisch-funktionellen Struktur vollständig prüfbar. Über das Ereignis-Zeit-Protokoll werden wichtige zusätzliche Prozessinformationen zugänglich, zu jeder Störung ist ohne zusätzliche Maßnahmen stets eine eindeutige Ursache diagnostiziert, der Systemzustand kann zu jeder Zeit vollständig und definiert ausgewiesen werden, eine gleichermaßen zum Zustand des zu steuernden Systems aussagefähige Kopie kann an einem zur Anlage vernetzten externen Steuerungscomputer geführt werden, die Elementarfunktionen und die definierten Befehle können direkt als funktionelle Basis für eine Visualisierung der zu steuernden Systeme und Prozesse dienen und in einfacher Weise kann die Kommunikation des Steuerprogramms mit anderen intelligenten Programmmodulen, wie beispielsweise Simulationen zur Prozessoptimierung, organisiert werden.

[0030] Für Kleinsteuerungen mit einem begrenzten Befehlsumfang können bei grundsätzlich gleichem Aufbau und gleicher Arbeitsweise der Steuerung in einem Steuerungs-Hardwarebaustein die Module des Ausführungsrechners und des Befehlsrechners mit festen Befehlssätzen eingebracht werden, die mit einfachen Bedienelementen aufgerufen werden, wobei über eine geeignete Schnittstelle ein externer Computer angekoppelt werden kann, womit das Einbringen der Steuerungssoftware und im Bedarfsfall auch eine komfortable Kommunikation und Diagnose realisierbar sind und damit vergleichbare Steuerungseigenschaften und vergleichbarer Komfort bei geringen Kosten erreicht werden.

[0031] Die erfindungsgemäße Lösung vermeidet die Mängel nach dem Stand der Technik durch einen bisher unüblichen Programmieransatz, der von der gestalteten und damit eingeprägten Funktionalität des Systems Gebrauch macht. Eine Signalverknüpfung mit Boolscher Algebra in Bedingungsgleichungen für das Setzen von Ausgängen wird vollständig durch neue Mittel ersetzt.

[0032] Nachfolgend wird die Erfindung an Hand von Ausführungsbeispielen näher erläutert. In den Zeichnungen zeigen:

[0033] Fig. 1 eine Darstellung einer grundlegenden Gliederung der Aufgabenbereiche in der Struktur der Steuerung

[0034] Fig. 2 eine hierarchisch gegliederte Funktionsstruktur eines technischen Systems

[0035] Fig. 3 für die Elementarfunktionen festzulegende Informationen an einem allgemeinen Beispiel

[0036] Fig. 4 eine einfache technische Anlage in schematischer Darstellung

[0037] Fig. 5 eine der Fig. 4 entsprechende Funktionsstruktur

[0038] Fig. 6 eine der Fig. 4 entsprechende Definition der Elementarbefehle

[0039] Fig. 7 eine Darstellung von Eingabe und Aufbau eines Datengerüsts zur Realisierung der Steuerung

[0040] Fig. 8 einen Aufbau eines Ausführungsrechners

[0041] Fig. 9 einen Inhalt eines Befehls als Befehlssatz für den Befehlspuffer

[0042] Fig. 10 eine Darstellung der Arbeitsweise eines Befehlsstarters

[0043] Fig. 11 eine Darstellung der Arbeitsweise eines EF-Controlers

[0044] Fig. 12 eine Darstellung der Arbeitsweise eines Nichtsoll-Bewerters

[0045] Fig. 13 eine Darstellung der Arbeitsweise eines Zustandsüberwachers

[0046] Fig. 14 einen Aufbau eines Ereignis-Zeit-Protokolls

[0047] Fig. 15 ein Beispiel einer Bildungsgrundlage formaler Befehlsnamen

[0048] Fig. 16 ein Beispiel für das Definieren von Nutzungsbefehlen

[0049] Fig. 17 ein Beispiel eines in einer Steuerung geführten Sperrenverzeichnisses

[0050] Fig. 18 ein Beispiel zur Festlegung von Fehlerbefehlen

[0051] Fig. 19 ein Beispiel einer Steuerung mit komplexeren Aufgaben

[0052] Fig. 20 eine Struktur und Namensdefinition nach Fig. 19

[0053] Fig. 21 alle Angaben für eine Befehlsbibliothek eines Befehlsrechner nach Fig. 19 und Fig. 20

[0054] Fig. 22 Merkmale einer Ausführung als Kleinsteuerung

[0055] Fig. 1 zeigt die grundlegende Gliederung der Aufgabenbereiche in der Struktur **1** der neuen Steuerung. Dem Ausführungsrechner **2** sind die zeitkritischen Aufgaben Soll-Ist-Vergleich, Reaktionen auf Abweichungen des Ist- zum Soll-Zustand und der befehlsgemäße Aufruf von zustandsändernden Aktoren zugeordnet. Vom Befehlsrechner **3** erhaltene Befehle werden vom Ausführungsrechner **2** ohne Prüfung umgesetzt, wobei die Ausführung eines Befehls und die Reaktion auf Abweichungen des Soll-Ist-Zustandes autark vom Ausführungsrechner **2** realisiert werden. Es ist zweckmäßig bzw. für kürzeste Reaktionszeiten der Steuerung bei komplexeren Systemen zwingend, dem Ausführungsrechner **2** eine eigene Hardware mit einem eigenen Prozessor zuzuordnen.

[0056] Im Befehlsrechner **3** werden alle Steuerungsoperationen auf logisch-funktionellem Niveau verwaltet. Hier werden aus geräteorientierten Elementarbefehlen prozessnahe Nutzungsbefehle als Einzelbefehle, als parallele oder serielle Befehlsfolgen definiert, abgelegt und aufgerufen. Hier erfolgt auch auf logischem Befehlsniveau die Verwaltung von Sperren für sich gegenseitig ausschließende Zustände als Alternative zu bisherigen Verriegelungen und Bedingungsformulierungen über Boolsche Signalverknüpfungen. Dem Anwendungsrechner **4** obliegen in diesem Steuerungskonzept alle Aufgaben, die nicht dem Ausführungs- oder dem Befehlsrechner zugeordnet sind. Das betrifft vor allem prozessnahe Probleme, wie sie beispielsweise im Bereich der Werkstückprogrammierung einer CNC-Steuerung vorliegen.

[0057] Die Steuerung kann dabei für unterschiedlichen Umfang und unterschiedliche Aufgabenkomplexität angemessen konfiguriert werden, wobei für alle Konfigurationen gleiche Prinzipien im Entwicklungssystem gelten. Bei sehr geringem Umfang an Befehlen kann der Anteil des Befehlsrechners **3** dem Ausführungsrechner **2** als Softwarebereich zugeordnet sein. Bei heute typischen SPS-Aufgaben kämen Ausführungs- und Befehlsrechner mit eigenen Prozessoren zur Anwendung, wobei die Systembedienung sowohl über Bedien- und Signalelemente bis zur Monitorausstattung erfolgen kann. In allen Ausführungen ist es darüber hinaus möglich, über eine einfach zu gestaltende Schnittstelle ein komfortables Kommunikationssystem – beispielsweise einen transportablen Computer – anzukoppeln und damit Programmierung und Inbetriebnahme oder Diagnose im Fehlerfall auszuführen.

[0058] Fig. 2 zeigt beispielhaft die hierarchisch gegliederte Funktionsstruktur **5** eines technischen Systems. Es ist entwicklungsmethodisch begründet, dass eine solche Systemstruktur von der Funktionseinheit für das Gesamtsystem **6** über unterschiedliche Funktionseinheiten der Teilsysteme **7** bis zu den Funktionseinheiten Elementarfunktionen **8** spezifisch für jedes technische System aufgestellt werden kann. Die Endzweige dieser Baumstruktur stellen im Sinne der neuen Steuerung Elementarfunktionen dar, die dadurch charakterisiert sind, dass diese Funktionseinheiten unterschiedliche Zustände einnehmen können und sich nicht zweckmäßig weiter untergliedern lassen, deren steuerungsseitig interessierende Funktionszustände also nicht mehr eine Repräsentation kombinierter Zustände anderer elementarer und zu steuernder Funktionsgruppen sind, wie es für übergeordnete nicht elementare Funktionseinheiten **6** und **7** in der Struktur charakteristisch ist. Entscheidend ist dabei die Stellung im zu steuernden System, so dass auch ein durch wenige elementare Befehle eingebundenes intelligentes Teilsystem als Elementarfunktion eingeordnet wird.

[0059] Fig. 3 beschreibt an einem allgemeinen Beispiel die für Elementarfunktionen festzulegenden Informationen in einem „Datenblatt zu Elementarfunktionen" **9**. In **10** wird der Name für die Elementarfunktion festgelegt, der diese Elementarfunktion identifiziert. Zweckmäßigerweise zeigt eine Funktionsskizze **11** die Merkmale der Zustände der Elementarfunktion mit der Zuordnung von Aktoren **12** und Sensoren **13**. In dem gekennzeichneten Bereich für die Zustandsdefinitionen **14** werden die für die Steuerung notwendigen Informationen datenverarbeitungsgerecht systematisiert und definiert. Die Zustandsdefinitionen **14** zeigen die Zustände, die die Elementarfunktion einnehmen kann und die Definition der den Zuständen zugeordneten Signalvektoren **15** für die Aktoren **12** und Sensoren **13**. Gleichermaßen werden hier die Befehle **16** definiert, die einen Übergang in einen bestimmten Zustand auslösen. Für jeden dieser Übergänge wird eine Kontrollzeit **17** vorgegeben, die in der Regel ein mehrfaches der wahrscheinlichen Funktionszeit sein kann und nur für das Erkennen von Ausführungsfehlern bei Nichterreichen des befohlenen Zustandes verwendet wird. Mit der Markierung wird von den möglichen Zuständen einer als Referenzzustand **18** festgelegt.

[0060] Fig. 4 zeigt eine einfache technische Anlage, für die in Fig. 5 die Funktionsstruktur und in Fig. 6 die

Definition der Elementarbefehle dargestellt sind.

[0061] Die hier beschriebene hierarchische Funktionsstruktur und die Definition der zugehörigen Elementarfunktionen sind vom Wesen her primäre Entwicklungsinhalte, die mit nur geringem zusätzlichem Aufwand und bereits in einer relativ frühen Phase bei der Produktentwicklung dokumentiert werden können.

[0062] Fig. 7 zeigt Eingabe und Aufbau des Datengerüsts bei der Anwendung der neuen Steuerung. Die Editierebene **19** beinhaltet die beiden Hauptbestandteile hierarchische Funktionsstruktur **5** und die Datenblätter der Elementarfunktionen **9**. Jede Funktionseinheit Elementarfunktion **8** in der Struktur muss mit einem entsprechenden Datenblatt zu Elementarfunktionen **9** beschrieben sein. Diese Vollständigkeit der Angaben und deren formale Korrektheit wird in der Editierebene automatisch geprüft. Bei positivem Prüfergebnis und Bestätigung des Nutzers für den Abschluss der Systembeschreibung wird die Eingabe geschlossen und aus den vorliegenden Angaben die Datenbasis der Steuerung für das beschriebene System generiert. Als erster Schritt erfolgt das Generieren des Elementarfunktions-Speichers **20**. Dieser Elementarfunktions-Speicher **21** enthält alle Elementarbefehle des Systems, alle Systemzustände und die dazu festgelegten Informationen, wie sie mit Fig. 3 beschrieben wurden. Die formalen Namen der Elementarfunktionen werden aus der Struktur abgeleitet, so dass Elementarfunktionen auch bei Verwendung eines gleichen Datenblattes einen unverwechselbaren Namen erhalten. Im zweiten Schritt erfolgt das Generieren des EF-Controlers **22**. Dazu wird im EF-Controler **23** der Referenzzustand des Systems aus den festgelegten Referenzzuständen **18** aller Elementarfunktionen aufgebaut. Für den hier ebenfalls geführten Istzustand der Elementarfunktionen wird durch Doppeln der Datenstruktur des Soll-Zustandes der Elementarfunktionen **24** die Datenstruktur zum Speichern des Istzustandes der Elementarfunktionen **25** angelegt. Es wäre damit hier bereits möglich, beim Betreiben der Steuerung einen Vergleich zwischen dem Soll- und dem Ist-Zustand der Sensoren der Elementarfunktionen vorzunehmen. Größere Effektivität ermöglicht aber der mit **26** gekennzeichnete dritte Schritt zum Generieren des (ebenfalls später noch genauer beschriebenen) Zustandsüberwachers **27**. Dabei werden aus den Soll-Signalvektoren der Elementarfunktionen **28** und gleichermaßen aus den Ist-Signalvektoren der Elementarfunktionen **29** der Sollsignalvektor des Systems **30** und der Istsignalvektor des Systems **31** gebildet. Jedem Sensor im System-Signalvektor bleibt seine Herkunftsadresse als Name der Elementarfunktion **10** im EF-Controler **23** zugeordnet.

[0063] Fig. 8 zeigt den Aufbau des Ausführungsrechners **2** und das Zusammenwirken mit dem Befehlsrechner **3**. Der Ausführungsrechner **2** erhält vom Befehlsrechner **3** einen auszuführenden Nutzungsbefehl **32**. Dieser wird in einem Baustein Befehlsaufbereiter **33** decodiert. Dabei werden Nutzungsbefehle in ihre entsprechenden Elementarbefehle überführt und diesen wird aus dem Elementarfunktions-Speicher **21** der vollständige zugeordnete Informationsinhalt des Befehlssatzes übergeben. Dieser Befehlssatz wird in dem Befehlspuffer **34** des Ausführungsrechners eingetragen. Nach Quittung, dass der vorhergehende Befehl abgeschlossen wurde, startet der Baustein Befehlsstarter **35** den im Befehlspuffer **34** wartenden Befehl und führt alle damit verbundenen Aktivitäten aus. Das betrifft das Aktualisieren im Baustein EF-Controler **36**, im Baustein Zustandsüberwacher **37** und im Baustein Nichtsoll-Bewerter **38**. Vom Baustein Befehlsstarter **35** wird im Baustein EF-Controler **36** für die betreffende Elementarfunktion der neue Sollzustand für die Sensoren eingetragen und durch das befehlsgemäße Setzen der Ausgänge der entsprechende Aktorbefehl gestartet. Ebenso wird die der Befehlsausführung zugeordnete Kontrollzeit **17** gestartet. Im Nichtsoll-Aktionsspeicher **39** werden die Bestandteile des Befehlssatzes „Nichtsoll-Befehle und -Meldungen" eingetragen.

[0064] Nach Ausführung der Start-Aktivitäten vom Befehlsstarter **35** übernimmt der Baustein Zustandsüberwacher **37** wieder den Vergleich des Soll-Signalvektors des Systems **30** mit dem Ist-Signalvektor des Systems **31**. Bei einer in diesem Vergleich erkannten Abweichung zwischen Soll und Ist wird im EF-Controler **36** der Ist-Zustand des abweichenden Signals im Ist-Signalvektor der Elementarfunktion **29** aktualisiert. Im EF-Controler **36** erfolgt die Bewertung der Abweichung (in Fig. 11 näher beschrieben), entweder a) ohne weitere Reaktion durch den über das laufende Zeitglied erkannten Status „beim Ändern" und damit Rückgabe der Aktivitäten an den Baustein Zustandsüberwacher **37**, b) über das Erkennen eines ausgeführten Befehls bei Übereinstimmung von Sollsignalvektor der Elementarfunktion **28** und Ist-Signalvektor der Elementarfunktion **29** im EF-Controler **36** und damit Aufruf des Bausteines Befehlstarter **35**, oder – wenn beides nicht zutrifft – c) die Übergabe des Ist-Signalvektors der Elementarfunktion **29** an den Baustein Nichtsoll-Bewerter **38**. Dort wird dieser Ist-Signalvektor **29** mit den im Nichtsoll-Aktionsspeicher **39** stehenden Signalvektoren verglichen und bei Übereinstimmung der diesem Fall zugeordnete Nichtsoll-Befehl **40** über den Baustein Befehlsstarter **35** gestartet. Gibt es keine Übereinstimmung, erfolgt eine Rückkehr zum Zustandsüberwacher **37**. In allen Fällen wird eine entsprechende Meldung **41** erzeugt. Der mit **42** gekennzeichnete Bereich markiert die zeitkritischen Aktivitäten.

[0065] Fig. 9 zeigt den Inhalt eines Befehls **43**, wie er als Befehlssatz in den Befehlspuffer **34** des Ausführungsrechners **2** eingetragen wird. Zeile (1) ent-

hält die Bezeichnung der zur Änderung angewiesenen Elementarfunktion **10**, Zeile (2) und Zeile (3) beinhalten den neuen Sollzustand der Sensoren bzw. der Aktoren und damit den Soll-Signalvektor der Elementarfunktion **28**, Zeile (4) gibt die Kontrollzeit **17** vor, in der die Änderung des Zustandes auf die neue Vorgabe erfolgt sein muss, Zeile (5) beinhaltet die Angaben zur Aktualisierung der ab Start des Befehls geltenden Einträge im Nichtsoll-Aktionsspeicher **39** für Reaktionen mit Nichtsoll-Befehlen **40**, und Zeile (6) beinhaltet gleiches für die Aktualisierung nach erfolgreicher Befehlsausführung. Die Angaben der Zeilen (1) bis (4) entsprechen dabei direkt den Definitionen aus der Editierebene **19** zu den Elementarfunktionen **8**. Die Zeilen (5) und (6) können darüber hinaus Nichtsoll-Befehle **40** aus Festlegungen zu prozessbezogenen Vorgaben über die Nutzungsbefehle **32** beinhalten.

[0066] Fig. 10 beschreibt die Arbeitsweise des Bausteins Befehlsstarter **35** und die Behandlung von Folgebefehlen **44** sowie von Parallelbefehlen **45**. Der Befehlspuffer **34** wird vom Befehlsrechner **3** immer mit dem der laufenden Befehlsausführung folgenden Befehl geladen. Definierte Befehlsfolgen (= Folgebefehle **44**) unterscheiden sich dabei nicht von einzeln definierten, voneinander unabhängigen Befehlen.

[0067] Parallelbefehle **45** sind dadurch charakterisiert, dass sie funktionell wie zeitlich voneinander unabhängig ausgeführt werden können und für einen zeitlich optimalen Ablauf auch parallel ausgeführt werden müssen. Für jeden als parallel definierten Befehl wird deshalb ein eigener Befehlspuffer **46** an der Schnittstelle zwischen Befehlsrechner **3** und Ausführungsrechner **2** definiert, aus dem voneinander unabhängige parallele Befehlsfolgen **45** abgearbeitet werden können. Stehen nach Abarbeitung von Parallelbefehlen **45** keine solchen mehr an, werden die eröffneten Speicherbereiche wieder geschlossen, so dass nur aktuell benötigte Pufferspeicher **46** geführt werden. Fig. 10 zeigt ein Beispiel für drei eröffnete Parallelbefehle **45**, aus denen nacheinander die eingetragenen Befehle **43** gestartet werden. Sollte die Prüfung **47** zeigen, dass kein weiterer Befehl im Speicherpuffer ansteht, wird der entsprechende Parallel-Befehlspuffer **46** geschlossen und der Baustein Zustandsüberwacher **37** aufgerufen. Bei positivem Prüfergebnis **48** wird entsprechend Befehlsinhalt **43** aktualisiert und gestartet. Nach Abschluss dieser Operationen wird der Baustein EF-Controler **36** aktiviert **49**. Nach dem Erreichen des befehlsgemäßen Zustandes **50** werden vom Befehlsstarter **35** die dafür im Befehlssatz **43** festgelegten Aktualisierungen ausgeführt und danach der nächste Befehl ermittelt und gestartet.

[0068] Fig. 11 zeigt die Arbeitsweise des Bausteins EF-Controler **36**. Der Start **51** für eine Aktivität des EF-Controlers wird stets von einer aktuellen Änderung angestoßen. Das ist entweder ein neuer Sollsignalvektor einer Elementarfunktion **28**, den der Baustein Befehlsstarter **35** bei einem neuen Befehl einträgt **52**, oder eine vom Baustein Zustandsüberwacher vorgenommene Aktualisierung **53** des vorliegenden Ist-Zustandes **29**. Die erste Prüfung **54** vergleicht den Soll- mit dem Ist-Zustand. Bei Übereinstimmung wird geprüft, ob der Änderungsstatus **55** gesetzt war. Ist das der Fall **56**, wurde ein laufender Befehl abgeschlossen, andernfalls **57** wurde der befohlene Zustand nach einer fehlerhaften Abweichung wieder erreicht. In beiden Fällen wird eine entsprechende Meldung erzeugt und der Baustein Befehlsstarter **35** gestartet **58** – Stimmen Ist- und Sollzustand nicht überein, wird der Zweig **59** abgearbeitet. Wieder wird der Änderungsstatus geprüft **60**. Liegt der Änderungsstatus bei dieser Elementarfunktion vor **61**, wird die Meldung „EF beim Ändern" **62** erzeugt und der Baustein Zustandsüberwacher **37** gestartet. Liegt jedoch kein Änderungsstatus vor **63**, werden der Name und der vorliegende Nichtsoll-Istsignalvektor **64** der Elementarfunktion in den Auswertungsspeicher des Nichtsoll-Bewerters **65** eingetragen und der Nichtsoll-Bewerter **38** wird gestartet.

[0069] Fig. 12 zeigt die Arbeitsweise des Nichtsoll-Bewerters **38**. Der Start **66** des Nichtsoll-Bewerters wird durch den EF-Controler **36** nach Übergabe eines Nichtsoll-Signalvektors **64** ausgelöst. Im ersten Schritt wird geprüft, ob unter dem Namen der Elementarfunktion **10** Eintragungen im Nichtsoll-Aktionsspeicher **39** vorhanden sind. (Wie schon bei Fig. 10 beschrieben, werden diese Einträge vom Befehlsstarter **35** als Informationsbestandteile eines Befehls **43** aktualisiert.) Sind für die Elementarfunktion keine Festlegungen im Nichtsoll-Aktionsspeicher **39** enthalten **67**, wird nur eine Fehlermeldung **67a** mit der Bezeichnung der Elementarfunktion und dem Nichtsoll-Istsignalvektor **64** mit Kennzeichnung des fehlerhaften Signals an die übergeordnete Steuerungsebene – dem Befehlsrechner **3** – zur Auswertung übergeben. Dann wird der Baustein Zustandsüberwacher **37** wieder gestartet.

[0070] Liegt im Nichtsoll-Aktionsspeicher **39** ein Nichtsoll-Signalvektor zu der Elementarfunktion vor 68, wird als nächster Schritt der Nichtsoll-Istsignalvektor **64** auf Übereinstimmung mit dem gespeicherten Signalvektor verglichen **69**. Liegt keine Übereinstimmung vor **70**, wird wieder nur eine konkrete Fehlermeldung **67** erzeugt und ebenso der Zustandsüberwacher **37** wieder gestartet. – Liegt jedoch eine Übereinstimmung des Signalvektors mit Eintragungen im Aktionsspeicher vor **71**, werden für diesen Fall festgelegten Reaktionsbefehle **72** dem Befehlsstarter **35** zur sofortigen Ausführung übergeben. Parallel dazu wird für die zu erzeugenden Meldung geprüft **73**, ob eine Ereignissteuerung **74** vorliegt. Bei einer Ereignissteuerung **74** bewegt sich das System in dem normalen funktionellen Rahmen, ein erkanntes

Ereignis ruft eine entsprechende Aktion auf (beispielsweise das Abschalten einer Pumpe bei Erreichen des oberen Füllstandes). In diesem Fall weist die entsprechende Meldung **75** den Ereignisbefehl der Elementarfunktion **76** in deutlicher Unterscheidung zu Fehlerzuständen aus. Handelt es sich nicht um eine Ereignissteuerung **77**, wird eine entsprechende Fehlermeldung **78** erzeugt.

[0071] Fig. 13 zeigt Arbeitsweise und Merkmale des Bausteins Zustandsüberwacher **37**. Wenn keine anderen Baustein-Aktivitäten laufen, startet der Zustandsüberwacher ständig den Vergleich **80** des Sollsignalvektors **30** und des Istsignalvektors **31** des Systems. Dieser Vergleich erfasst immer den gesamten System-Signalvektor und wird bei Übereinstimmung der verglichenen Zustände ständig wiederholt **81**. Bei einer erkannten Abweichung wird zunächst geprüft, ob das System den Wartezustand verlassen und einen neuen Befehl ausführen soll. Ist das der Fall **82**, wird der Baustein Befehlsstarter **35** gestartet. Im anderen Fall wird das abweichende Ist-Signal in den Ist-Signalvektor der betreffenden Elementarfunktion im EF-Controler eingetragen **83** und dort – wie zu Fig. 11 erläutert – ausgewertet. Entstehen kann eine Abweichung entweder durch Vorgabe eines neuen Sollzustandes bei Start eines neuen Befehls und erfolgtem Eintrag in den Sollsignalvektor **30** des Systems durch den EF-Controler **36**, oder im anderen Fall durch ein geändertes Sensorsignal im Istsignalvektor **31** des Systems. Nach dem Eintrag des abweichenden Ist-Signals in der betroffenen Elementarfunktion im EF-Controler wird dieser gemeldete Ist-Zustand als neuer Vergleichszustand in dem Soll-Vergleichsvektor eingetragen **84**. Damit wird erreicht, dass jede Veränderung nur einmal ausgewertet wird. Der Soll-Vergleichszustand des System-Signalvektors ist damit als Vergleich mit „dem letzten ausgewerteten Zustand" des Systems **84** definiert. Damit ist es möglich und sinnvoll, das erkannte Ereignis in ein Ereignis-Zeit-Protokoll einzutragen **85**, auf das in Fig. 14 näher eingegangen wird. Nach den genannten Aktionen des Zustandsüberwachers **37** startet dieser den Baustein EF-Controler **36**. Nach erfolgter Auswertung wird wieder – wie bei der Arbeitsweise von EF-Controler bzw. Befehlsstarter beschrieben – der Baustein Zustandsüberwacher aufgerufen. Dieser setzt dann den Soll-Ist-Vergleich im Systemsignalvektor bei dem Signal fort, das dem zuletzt nicht übereinstimmenden Signal folgt. Damit wird gesichert, dass nacheinander alle Signale des Systemsignalvektors verglichen werden und nicht ein schwingendes Signal eine Endlosschleife verursachen kann. Ein solches Phänomen wäre bei Wiederaufnahme des Vergleiches bei dem gerade ausgewerteten Signal denkbar, wenn dieses Signal seinen Zustand im Takt der Signallaufzeit ändern würde.

[0072] Fig. 14 soll den Aufbau eines Ereignis-Zeit-Protokolls **85** in Form einer Liste verdeutlichen. Die erste Spalte nimmt die Bezeichnung der vom Ereignis betroffenen Elementarfunktion auf, die Spalte **2** das von der Änderung betroffenen Signal, die Spalte **3** den geänderten Signalzustand. Das sind Kopien der Informationen, die der Zustandsüberwacher dem EF-Controler übergibt. Wird in Spalte **4** die aktuelle Systemzeit beim Erkennen des Ereignisses eingetragen, entsteht ein in vielfacher Hinsicht nutzbares Ablaufprotokoll. Im Beispiel ist mit dem ersten und dem letzten Eintrag markiert, dass die Elementarfunktion All mit dem Signal E1 wieder den ersten Zustand erreicht hat. Die den Ereignissen zugeordneten Zeiten könnten gegebenenfalls als genaues Maß für eine solche Periode dienen. Mit diesem Protokoll ist es auch möglich, Signalschwingungen zu erkennen und im Bedarfsfall Filter zu aktivieren, die beispielsweise die Abtasthäufigkeit für das schwingende Signal verringern können. Je nach Prozess und Bedeutung der Informationen sowie zur Verfügung stehendem Speicher können längere Zeiten erfasst und gespeichert werden, was für Diagnosezwecke hinsichtlich Langzeitveränderungen genutzt werden kann, oder nur mit fest begrenztem Speicher der jeweils letzte Abschnitt beispielsweise für eine Havarieauswertung zur Verfügung stehen.

[0073] Fig. 15 zeigt für das in Fig. 4 bis Fig. 6 ausgeführte Beispiel die Bildungsgrundlage der formalen Befehlsnamen **86**, die aus der Funktionsstruktur **5** abgeleitet werden und für eine eindeutige Bezeichnung der Elementarfunktionen in Nutzungsbefehlen verwendet werden können.

[0074] In Fig. 16. ist an dem Anwendungsbeispiel von Fig. 4–Fig. 6 das Definieren von Nutzungsbefehlen und das Festlegen von Befehlssperren **88** zu den Nutzungsbefehlen dargestellt.

[0075] Fig. 17 zeigt als Beispiel das in der Steuerung geführte Sperrenverzeichnis **89** des Systems Schließanlage und soll das dynamische Wirken der in Fig. 16 festgelegten Sperrbedingungen deutlich machen. Bei der Fig. 17 muss betont werden, dass es eine Hilfs-Darstellung ist und eine solche Tabelle in der Steuerung nicht existiert. Vorhanden ist immer nur ein Speicherbereich, in den zu unterschiedlichen Zeiten (hier mit t1 bis t8 ausgewiesen) durch die bis dahin wirkenden Befehle unterschiedliche Bedingungen eingetragen sind.

[0076] In der Spalte **1** sind alle Befehle des Systems aufgelistet. Enthält ein aktivierter Nutzungsbefehl eine Sperrbedingung für einen anderen Befehl, wird der verursachende Befehl als Sperrbedingung bei dem anderen Befehl eingetragen. Hier im Beispiel sperrt mit seiner Aktivierung zur Zeit t7 der Befehl EF2-B2 (verriegeln) den Befehl EF1-B1 (Türbeweger öffnen). Der Riegel soll – wie festgelegt – nur bei geschlossener Tür eingelegt werden können, deshalb der Eintrag der Sperre bei EF2-B2 mit Erteilen des

Befehls „(Tür öffnen) EF1-B1" zur Zeit t3.

[0077] Für die Wirkungsweise der Steuerung ist wichtig, dass der Befehlsrechner **3** nach Übergabe eines Nutzungsbefehls **32** in den Befehlspuffer **34** des Ausführungsrechner **2** – den dieser dann wie beschrieben autark ausführen kann – seinen Zustand so aktualisiert, wie er nach ordnungsgemäßer Ausführung dieses Befehls eintreten wird. Für diesen Zustand wird die Zulässigkeit des nächsten Befehls noch während der Ausführung des vorhergehenden Befehls geprüft und ggf. freigegeben. Im Beispiel befindet sich während der Ausführung „Riegelschloss entriegeln" zu t1 im Befehlspuffer der Befehl „Tür öffnen", der zum Zeitpunkt t3 gestartet werden wird und dann gleichzeitig die Prüfung des Befehls „Tür schließen" für den Zeitpunkt t5 nach den Bedingungen vom Zeitpunkt t4 aktiviert.

[0078] Damit wird zeitoptimal erreicht, dass mit dem Abschluss eines Befehls sofort der schon geprüfte Folgebefehl starten kann oder gleichermaßen noch während der Laufzeit eines Befehls erkannt wird, dass der nächste vorbereitete Befehl für den kommenden Systemzustand nicht zulässig ist. Wenn ein Fehler bei dem im Ausführungsrechner laufenden Befehl auftritt, wird der Befehlsrechner auf den Fehlerzustand aktualisiert zurückgesetzt.

[0079] Fig. 18 (auf Blatt **12**) soll ein Beispiel zur Festlegung von Fehlerbefehlen zeigen. Es sei als kritisch angenommen, wenn die Tür beim Schließen – aus welchen Gründen auch immer – auf einen eingelegten Riegel trifft. Fig. 18 zeigt die Formulierung eines Fehlerbefehls als Bestandteil des Befehlssatzes „Türbeweger schließen". Aus dem Zustand der Elementarfunktion Riegelschloss E1 = 1 wird „Riegel nicht frei" geschlussfolgert und als Fehlerreaktion im Nichtsollbewerter der Vorgang „Tür schließen" „in Tür öffnen" umgesteuert.

[0080] Analysen zeigen, dass in der Regel zu einem bestimmten Zeitpunkt nur wenige Fehlerbefehle benötigt werden. Grundsätzlich ist es möglich, auf ein beliebiges Ereignis mit jedem Befehl zu reagieren.

[0081] Fig. 19 soll als weiteres Beispiel die Möglichkeiten der Steuerung bei komplexeren Aufgaben und unterschiedlichen Nutzungsanforderungen einer Anlage zeigen. Für solche unterschiedlichen Nutzungsanforderungen und daraus resultierende unterschiedliche Befehls- und Sperrbedingungen in einer Anlage soll der Begriff „Status" **90** verwendet werden.

[0082] Es sei angenommen, dass zwei Türanlagen gesteuert werden sollen, die einzeln dem bisherigen Beispiel gleich sind. Hinzu kommt ein Anlagenschalter für den jeweils gewünschten Betriebsstatus: im Status S1 können die Türen unabhängig voneinander bedient werden, in S2 werden beide synchron geöffnet oder geschlossen und in S3 als Schleuse bleibt immer eine von beiden Türen geschlossen.

[0083] Fig. 20 zeigt die Struktur und Namensdefinitionen, wie sie die Steuerung hier nach den in der Editierebene **19** gemachten Angaben aufbaut.

[0084] Fig. 21 zeigt alle Angaben für die Befehlsbibliothek **92** des Befehlsrechners, um die gestellte Aufgabe zu lösen. Die für die Schließanlage einer Tür erarbeiteten Festlegungen werden für die unmittelbare Türsteuerung eines Exemplars beim Generieren unter dem neuen Systemnamen gedoppelt. Alle Festlegungen zum Steuerungs-Status beider Türen werden über neu definierte Statusbefehlsblätter **91** realisiert, die über den Statusschalter ausgewählt werden. Bei dem Status S3-Befehlsblatt wurde für die Sperrenformulierung nicht eine Elementarfunktion sondern mit „Tür x" eine höhere Hierarchieebene in der Funktionsstruktur benutzt. Damit können sehr effektiv ganze Funktionsbereiche gegen Zustandsveränderungen gesperrt oder auch mit Formulierungen „alle außer XXX" sehr effektiv selektiert werden.

[0085] Fig. 22 zeigt Merkmale einer Kleinsteuerung **94** bei einem technischen Gerät **95**. Der relativ kleine und feste Befehlsumfang der Kleinsteuerung **95** ist in einem Steuerungs-Hardwarebaustein angeordnet, der die Funktionalität des Ausführungsrechners **2** und des Befehlsrechners **3** beinhaltet. Die Bedienung erfolgt mit üblichen Schalt- und Anzeigegeräten **96**. Über eine Schnittstelle **97** kann ein Computer **98** angekoppelt werden, womit alle Funktionalität der Steuerung zum Einbringen der Steuerungssoftware und komfortable Kommunikation und Diagnose möglich sind.

Bezugszeichenliste

1	Struktur der Steuerung ASS
2	Ausführungsrechner
3	Befehlsrechner
4	Anwendungsrechner
5	Funktionsstruktur
6	Funktionseinheit Gesamtsystem
7	Funktionseinheit Teilsystem
8	Funktionseinheit Elementarfunktion
9	Datenblatt zu Elementarfunktionen
10	Name der Elementarfunktion
11	Funktionsskizze
12	Aktoren
13	Sensoren
14	Zustandsdefinitionen
15	Signalvektoren
16	Elementarbefehle
17	Kontrollzeit
18	Referenzzustand
19	Editierebene
20	Generieren des Elementarfunktions-Speichers

21 Elementarfunktions-Speicher
22 Generieren des EF-Controlers
23 EF-Controler
24 Soll-Zustand der Elementarfunktionen
25 Ist-Zustand der Elementarfunktionen
26 Generieren des Zustandsüberwachers
27 Zustandsüberwacher
28 Soll-Signalvektor der Elementarfunktion
29 Ist-Signalvektor der Elementarfunktion
30 Sollsignalvektor des Systems
31 Istsignalvektor des Systems
32 Nutzungsbefehl
33 Befehlsaufbereiter
34 Befehlspuffer
35 Baustein Befehlsstarter
36 Baustein EF-Controler
37 Baustein Zustandsüberwacher
38 Baustein Nichtsollbewerter
39 Nichtsoll-Aktionsspeicher
40 Nichtsoll-Befehl
41 Zustandsmeldung
42 Zeitkritischer Funktionsbereich
43 Inhalt eines Befehls
44 Folgebefehle
45 Parallelbefehle
46 Befehlspuffer für Parallelbefehle
47 Prüfungsergebnis Befehlspuffer "nein"
48 Prüfungsergebnis Befehlspuffer "ja"
49 Aktivierung EF-Controler
50 Aktivitäten Befehlsstarter bei Erreichen "befehlsgemäßer Zustand"
51 Aktivitätsstart EF-Controler
52 Änderung Sollzustand im EF-Controler durch Befehl
53 Änderung Istzustand im EF-Controler durch Sensormeldung
54 Vergleich Soll- und Ist-Zustand im EF-Controler
55 Änderungsstatus im EF-Controler
56 Alternative "Änderungsstatus"
57 Alternative "Kein Änderungsstatus"
58 Aufruf Befehlsstarter vom EF-Controler
59 Alternativen bei Nichtübereinstimmung Soll- und Istzustand im EF-Controler
60 Prüfung Änderungszustand bei Nichtübereinstimmung Soll- und Istzustand im EF-Controler
61 Aktivität bei Änderungszustand und Nichtübereinstimmung Soll- und Istzustand im EF-Controler
62 Meldung vom EF-Controler "Elementarfunktion (Name der Elementarfunktion) beim Ändern"
63 Alternative im EF-Controler bei Ist-Zustand ungleich Sollzustand und keinem Änderungszustand
64 Nichtsoll-Istsignalvektor
65 Auswertungsspeicher Nichtsoll-Bewerter
66 Start Nichtsollbewerter
67 Aktion, wenn Nichtsoll-Elementarfunktion keinen Eintrag im Nichtsoll-Aktionsspeicher hat
67a Fehlermeldung zu Nichtsoll-Elementarfunktion
68 Aktion, wenn Nichtsoll-Elementarfunktion einen Eintrag im Nichtsoll-Aktionsspeicher hat
69 Vergleich Nichtsoll-Istsignalvektor mit gespeichertem Nichtsoll-Signalvektor im Nichtsollbewerter
70 Aktion bei fehlender Übereinstimmung Nichtsoll-Istsignalvektor mit Nichtsoll-Signalvektor im Nichtsoll-Bewerter
71 Aktion bei Übereinstimmung Nichtsoll-Istsignalvektor mit Nichtsoll-Signalvektor im Nichtsoll-Bewerter
72 Reaktionsbefehle im Nichtsoll-Aktionsspeicher
73 Prüfung, ob Nichtsoll-Istsignalvektor zu einer Ereignissteuerung gehört
74 Ereignissteuerung
75 Meldung Ereignissteuerung
76 Inhalt der Meldung Ereignissteuerung
77 Fehlermeldung, wenn keine Ereignissteuerung
78 Inhalt der Fehlermeldung
79 Start des Bausteins Zustandsüberwacher
80 Vergleich Sollsignalvektor des Systems mit dem Istsignalvektor des Systems
81 Programmschleife zum Vergleich Sollsignalvektor des Systems mit dem Istsignalvektor des Systems
82 Übergabe der Aktivität vom Zustandsüberwacher an Befehlsstarter
83 Eintrag geändertes Ist-Sensorsignal vom Zustandsüberwacher im EF-Controler
84 Vergleichs-Sollvektor "letzter ausgewerteter Zustand" im Zustandsüberwacher
85 Ereignis-Zeit-Protokoll
86 formale Befehlsnamen
87 Befehlssperren
88 Sperrenverzeichnis im Befehlsrechner
89 Sperrenverzeichnis des Beispiels Schließanlage
90 Status einer Anlage
91 Status-Befehlsblätter
92 Befehlsbibliothek
93 Nutzungsprogramme
94 Kleinsteuerung
95 Technisches Gerät mit Kleinsteuerung
96 Schalt- und Anzeigegeräte
97 Schnittstelle für Computeranschluss
98 Transportabler Computer

Patentansprüche

1. Verfahren zum Steuern von Mechanismen oder technischen Systemen,
dadurch gekennzeichnet, dass
a) die zu steuernden Mechanismen oder technischen

Systeme in ihren Elementarfunktionen (**8**) mit deren befehlsgemäß definierten Zuständen und den zugehörigen Signalvektoren (**15**) der Sensoren (**13**) und Aktoren (**12**) in der Steuerung gespeichert werden, wobei ausgehend von einem definierten Referenzzustand (**18**) zu Beginn der Steuerungsaktivierung ein ständiger Vergleich der von der technischen Anlage durch die Sensoren (**13**) gemeldeten Ist-Zustände mit den in der Steuerung gespeicherten Sollzuständen (**24**) für alle Elementarfunktionen erfolgt und damit jede Abweichung im zu steuernden System vom befehlsgemäßen Sollzustand (**24**) erkannt wird,
b) ein den Zustand der Mechanismen oder des technischen Systems verändernder neuer Elementarbefehl (**16**) mit seinem Start den Sollzustand (**24**) für den Vergleich aktualisiert und auf der Grundlage ebenfalls gespeicherter zulässiger Kontrollzeiten (**17**) die Zeit bis zur Rückmeldung des befehlsgemäßen neuen Zustandes überwacht,
c) wobei Sensorsignale und vergleichbare Informationen ausschließlich der Zustandsidentifikation von Elementarfunktionen (**8**) dienen, Zustandsänderungen ausschließlich über den Start von Elementarbefehlen (**16**) erfolgen, denen die Sensor- und Aktor-Signale als Sollzustand zugeordnet sind und die auf logisch-funktionellem Sprachniveau frei definierten Nutzungsbefehle (**32**) durch entsprechende Zuordnung von Elementarbefehlen (**16**) definiert sind.

2. Verfahren nach Anspruch 1,
dadurch gekennzeichnet,
a) dass in der Steuerung in einem speziellen Programmbaustein, hier als EF-Controler (**23**) bezeichnet, die Zustände aller Elementarfunktionen (**8**) als befehlsgemäß aktueller Sollzustand (**24**) und als aktueller Istzustand (**25**) mit den zugehörigen Aktoren (**12**) und Sensoren (**13**) geführt werden,
b) wobei damit jede über die Sensoren (**13**) erkannte Zustandsänderung des technischen Systems der betroffenen Elementarfunktion (**8**) als aktueller Istzustand zugeordnet und mit dem in der Steuerung geführten Sollzustand (**24**) verglichen und bewertet werden kann.

3. Verfahren nach Anspruch 1 oder 2,
dadurch gekennzeichnet, dass
a) bei einem erkannten, nicht dem Sollzustand (**24**) entsprechendem Istzustand (**25**) einer Elementarfunktion (**8**) der den Istzustand beschreibende Signalvektor (**15**) an einen anderen speziellen Programmbaustein der Steuerung, hier als Nichtsollbewerter (**38**) bezeichnet, übergeben wird,
b) wobei in diesem Nichtsollbewerter (**38**) für ausgewählte Zustände von Elementarfunktionen (**8**) Reaktionsbefehle (**72**) gespeichert sind, die bei Übereinstimmung mit dem zur Prüfung übergebenen Zustand gestartet werden,
c) und in allen Fällen differenzierte Fehlermeldungen mit Angabe der betroffenen Elementarfunktion und des abweichenden Signales erzeugt werden.

4. Verfahren nach einem der Ansprüche 1 bis 3,
dadurch gekennzeichnet, dass
a) einem Nutzungsbefehl (**32**) als Befehlssatz sowohl die neuen Sollzustände (**24**) der Sensoren (**13**) und Aktoren (**12**), die Kontrollzeiten (**17**) zum neuen Sollzustand (**24**) als auch die bei Abweichungen zu startenden Reaktionsbefehle (**72**), jeweils unterschieden in vor dem Start und nach erfolgter Ausführung zu löschende und zu setzende Reaktionsbefehle (**72**) auf ausgewählte Zustandsmeldungen, zugeordnet werden,
b) wobei ein weiterer spezieller Programmbaustein der Steuerung, hier als Befehlsstarter (**35**) bezeichnet, die dafür erforderliche Organisation im System übernimmt und damit auch die Freigabe eines nächsten Befehls bei Befehlsfolgen nach Erfüllungsmeldung des vorhergehenden sowie die Organisation von Parallelbefehlen (**45**) durch je nach Bedarf temporäres Eröffnen von parallelen Abarbeitsfolgen realisiert wird.

5. Verfahren nach einem der Ansprüche 1 bis 4,
dadurch gekennzeichnet, dass
a) Sensorsignale (**13**) und weitere zu kontrollierende Informationen in einem weiteren speziellen Programmbaustein, hier als Zustandsüberwacher (**37**) bezeichnet, zu einem lückenlosen Datenwort zusammengezogen werden, wobei den Signalen die Adresse der zugehörigen Elementarfunktion (**8**) in dem Programmbaustein EF-Controler (**36**) der Steuerung zugeordnet bleibt,
b) für den Vergleich jedem Sollsignal (**30**) das Ist-Signal (**31**) der Sensormeldung in gleicher Struktur gegenübersteht,
c) wobei der Programmbaustein Zustandsüberwacher (**37**) bei einer erkannten Abweichung eines Ist-Signals dieses im EF-Controler (**36**) als neuen Istzustand der Elementarfunktion (**29**) aktualisiert,
d) und nach der Aktualisierung und Übergabe zur Auswertung im EF-Controler (**36**) das geänderte Signal als neuer Vergleichszustand im Zustandsüberwacher (**37**) eingetragen wird, womit ein Vergleich im Zustandsüberwacher (**37**) immer zum letzten ausgewerteten Zustand erfolgt und jede Zustandsänderung damit nur einmal ausgewertet wird,
e) wobei der Vergleich der Soll-Ist-Signale (**30**, **31**) im Zustandsüberwacher (**37**) gerichtet erfolgt und nach einer Unterbrechung für die Auswertung einer Abweichung der Vergleich bei dem der Unterbrechungsstelle folgenden Signal fortgesetzt wird, wodurch gesichert ist, dass jede zeitlich hinreichend lange Zustandsänderung erfasst und ausgewertet werden kann.

6. Verfahren nach einem der Ansprüche 1 bis 5,
dadurch gekennzeichnet, dass
a) jede erfasste Zustandsänderung vom Programmbaustein Zustandsüberwacher (**37**) in einem Ereignis-Zeit-Protokoll (**85**) eingetragen und dort gespeichert werden kann,

b) wodurch auf einfachstem Weg zeitabhängige Prozessparameter zugängig werden, damit auch Signalschwingungen erkannt und gegebenenfalls ausgefiltert werden können.

7. Verfahren nach einem der Ansprüche 1 bis 6, dadurch gekennzeichnet, dass
a) der Teilbereich Ausführungsrechner (**2**) mit den Programmbausteinen Befehlsstarter (**35**), EF-Controler (**36**), Nichtsollbewerter (**38**) und Zustandsüberwacher (**37**) nach Übergabe eines Elementarbefehls (**16**) an den Programmbaustein Befehlsstarter (**35**) in der Steuerung keine Prüfung auf Zulässigkeit enthält,
b) die Ausführung eines erhaltenen Befehls von den dem Ausführungsrechner (**2**) zugeordneten Programmbausteinen jeweils autark realisiert wird,
c) im Teilbereich Befehlsrechner (**3**) der Steuerung auf logisch-funktionellen Befehlsniveau zu den sich ausschließenden Zuständen Sperrenverzeichnisse (**88**) geführt und verwaltet werden, die den prozess- und maschinenseitig determinierten Anteil von funktionellen Verriegelungen übernehmen,
d) wobei ein Nutzungsbefehl (**32**) neben dem Aufruf von zu ändernden Elementarfunktionen (**8**) auch die Informationen enthält, für welche anderen Befehle Sperren im Sperrenverzeichnis (**88**) während oder nach der Ausführung dieses Nutzungsbefehls (**32**) zu setzen oder aufzuheben sind.

8. Verfahren nach einem der Ansprüche 1 bis 7, dadurch gekennzeichnet, dass
der Ausführungsrechner (**2**) und der Befehlsrechner (**3**) zeitlich um einen Programmschritt entkoppelt arbeiten,
a) der ausführende Teil der Steuerung, der Ausführungsrechner (**2**), einen erhaltenen Befehl autark ausführt, wobei ein Befehle verwaltender Teil der Steuerung, der Befehlsrechner (**3**), dem ausführenden Teil Ausführungsrechner (**2**) den nächstfolgenden geprüften Befehl in einem Zwischenspeicher als Befehlspuffer (**34**) bereitstellt,
b) und nach dem Bereitstellen eines Befehls im Befehlspuffer (**34**) des Ausführungsrechners (**2**) der Zustand im Befehlsrechner (**3**) auf den Stand aktualisiert wird, der nach der Ausführung dieses Befehls eintreten wird, und zu diesem erwarteten Zustand die Prüfung des dann nachfolgenden Befehls auf Zulässigkeit im Befehlsrechner (**3**) bereits während der Abarbeitung des vorhergehenden erfolgt,
c) wenn wegen eines Fehlers der erwartete Zustand nicht eintritt, wird der geprüfte Befehl aus dem Befehlspuffer (**34**) zurückgesetzt und das System auf den Fehlerzustand aktualisiert.

9. Verfahren nach einem der Ansprüche 1 bis 8, dadurch gekennzeichnet, dass Nutzungsbefehle (**32**) erarbeitet werden,
a) indem den sprachlich funktionell prozessnah zu definierenden Nutzungsbefehlen (**32**) aus den vorher definierten Elementarbefehlen (**16**) solche einzeln, parallel oder als Folge zugeordnet,
b) dazu die Sperrbedingungen auf Befehlsniveau für die mit Aufruf des Nutzungsbefehls (**32**) vorzunehmenden Aktualisierungen in dem Sperrenverzeichnis im Befehlsrechner (**88**) definiert,
c) die Reaktionsbefehle (**72**) auf ausgewählte Abweichungen sowie geeignete Fehlermeldungen festgelegt,
d) diese Informationen in einer Befehlsbibliothek (**92**) abgelegt werden, wo die Steuerung dann die Befehlsinhalte für Nutzungsbefehle (**32**) abruft.

10. Verfahren nach einem der Ansprüche 1 bis 9, dadurch gekennzeichnet, dass ein Nutzungsprogramm (**93**) für das Betreiben des technischen Systems die Reihenfolge von definierten Nutzungsbefehlen (**32**) festlegt, die jeweils nacheinander oder parallel abgearbeitet werden sollen.

11. Verfahren zur Erstellung von Steuerungssoftware,
dadurch gekennzeichnet, dass
die Erstellung der Steuerungssoftware durch ein Entwicklungssystem mit Dialogführung unterstützt wird,
a) wobei für die Beschreibung des zu steuernden Systems die Angabe der hierarchischen Funktionsstruktur (**5**) abgefragt wird,
b) das jeweils untere Ende dieser Struktur als Elementarfunktion (**8**) betrachtet wird und jede Elementarfunktion (**8**) ebenfalls im Dialog in ihren Befehlszuständen zu definieren ist,
c) wobei diesen definierten Elementarbefehlen (**16**) die Signale der Sensoren (**13**), der Aktoren (**12**), die Kontrollzeiten (**17**) für den Übergang zwischen den befehlsgemäßen Zuständen und ein Referenzzustand (**18**) für den Beginn zuzuordnen sind,
d) die Einbindung komplexerer Teilsysteme ebenso als Elementarfunktion (**8**) erfolgen kann, wenn die Stellung in der Funktionsstruktur (**5**) solches ausweist,
e) wobei das Dialogsystem als Grundlage für die Beschreibung der Funktionalität des technischen Systems nur die hier genannten primären Angaben zu Struktur (**5**) und zu den Elementarfunktionen (**9**) verlangt.

12. Verfahren nach Anspruch 11,
dadurch gekennzeichnet, dass
das dialoggeführte Entwicklungssystem zur Erstellung der Steuerungssoftware nach Eingabe der primären Angaben
a) den System-Elementarbefehlsspeicher (**21**),
b) den EF-Controler (**36**) und den
c) Sollsignalvektor (**30**) sowie den Istsignalvektor für den Zustandsüberwacher (**37**) anlegt und generiert und damit das technische System bereits inbetriebgenommen, auf fehlerfreie Signaldefinition im Referenzzustand (**18**) überprüft und mit den definierten Elementarbefehlen (**16**) in einem Inbetriebnahmestatus gesteuert und soweit zulässig mit diesen Einzel-

befehlen getestet und geprüft werden kann,

13. Verfahren nach Anspruch 11 oder 12, dadurch gekennzeichnet, dass Änderungen von Informationen zu Struktur (**5**) und Elementarfunktionen (**8**) nur über die Editierebene (**19**) möglich sind und die nachfolgende automatische Generierung die Konsistenz des geänderten Zustandes sichert.

14. Verfahren nach einem der Ansprüche 11 bis 13,
dadurch gekennzeichnet, dass das Entwicklungssystem für die Definition von Nutzungsbefehlen (**32**) in spezifischen Dialogen
a) die verfügbaren Elementarbefehle (**16**) des Systems zur Zuordnung anbietet,
b) Sperrbedingungen für das Sperrenverzeichnis (**88**) abfragt, wobei die Angaben für festzulegende Sperren grafisch über Auswahl in der Funktionsstruktur (**5**) und Sperrfestlegungen in Formulierungen „dieser Elementarbefehl", „diese Elementarfunktion", „dieser Zweig der Funktionsstruktur" oder „alle Funktionen dieses Funktionszweiges außer diesem Elementarbefehl" erfolgen können,
c) Festlegungen zu spezifischen Reaktionsbefehlen (**72**) auf besondere Fehler abgefragt werden,
d) alle Festlegungen gespeichert und in der Befehlsbibliothek (**92**) eingeordnet und verwaltet werden.

15. Verfahren nach einem der Ansprüche 11 bis 14,
dadurch gekennzeichnet,
a) dass für ein so aufgebautes Steuerungssystem Änderungen an Elementarfunktionen (**8**) lokal begrenzt bleiben,
b) dass jederzeit und ebenfalls mit überschaubarer lokaler Wirkung neue Nutzungsbefehle (**32**), Sperrbedingungen in dem Sperrenverzeichnis (**88**) oder Fehlerreaktionen durch Reaktionsbefehle (**72**) erweitert oder geändert werden können,
c) dass unterschieden durch die Vergabe von Statusinformationen (**90**) für das System ohne jede Rückwirkung auf schon definierte Programme neue Definitionen von Nutzungsbefehlen (**32**) und Befehlsbedingungen erfolgen können.

16. Einrichtung zum Steuern von Mechanismen oder technischen Systemen,
dadurch gekennzeichnet,
a) dass für die unterschiedlichen Aufgaben unterschiedliche Bereiche der Einrichtung vorgesehen sind und diese nach den wichtigsten Aufgabenmerkmalen konfiguriert werden, wobei insbesondere kurze Reaktionszeiten auf erkannte Ereignisse und sichere Programmabläufe erreicht werden,
b) dass die Programmbausteine für alle zeitkritischen Aufgaben der Steuerung Befehlsstarter (**35**), EF-Controler (**36**), Nichtsollbewerter (**38**) und Zustandsüberwacher (**37**) in einem hier als Ausführungsrechner (**2**) bezeichneten Teil der Einrichtung angeordnet sind,
c) der Ausführungsrechner (**2**) bei umfangreichen Programmen mit großer Programmvariabilität für diese zeitkritischen Aufgaben über einen eigenen Prozessor verfügt,
d) dass der Ausführungsrechner (**2**) für die Kommunikation mit der zu steuernden Einrichtung über die Sensoren (**13**), die Aktivierung von Aktoren (**12**), den Soll-Ist-Vergleich, Reaktionen auf Abweichungen des Ist-(**25**) zum Soll-Zustand (**24**) und die Abarbeitung eines erhaltenen Befehls autark ist,
e) zur Verwaltung von Nutzungsbefehlen (**32**) in Befehlsbibliotheken (**92**), das Führen von Sperrenverzeichnissen (**88**), die Abarbeitung von Nutzungsprogrammen (**93**) durch schrittweise Übergabe von Befehlen an den Ausführungsrechner (**2**) sowie zur äußeren Kommunikation von dem hier als Befehlsrechner (**3**) bezeichneten Bereich der Einrichtung ein weiterer Prozessor vorgesehen ist, wenn die Merkmale von c) zutreffen,
f) für die Aufgaben der Prozessgestaltung, soweit sie nicht Ausführungs- oder Befehlsrechner betreffen, ein Bereich Anwendungsrechner (**4**) vorgesehen ist.

17. Einrichtung nach Anspruch 16,
dadurch gekennzeichnet, dass
a) für Kleinsteuerungen (**94**) mit geringem Befehlsumfang und unkritischen Zeitforderungen in einem Steuerungs-Hardwarebaustein (**95**) die Module des Ausführungsrechners (**2**) und des Befehlsrechners (**3**) mit festen Befehlssätzen eingebracht sind,
b) zur Bedienung und Kommunikation übliche Schalt- und Anzeigegeräte (**96**) vorgesehen sind,
c) über eine Schnittstelle (**97**) ein externer Computer (**98**) für das Einbringen der Steuerungssoftware sowie im Bedarfsfall für eine komfortable Kommunikation und Diagnose ankoppelbar ist.

Es folgen 17 Blatt Zeichnungen

Anhängende Zeichnungen

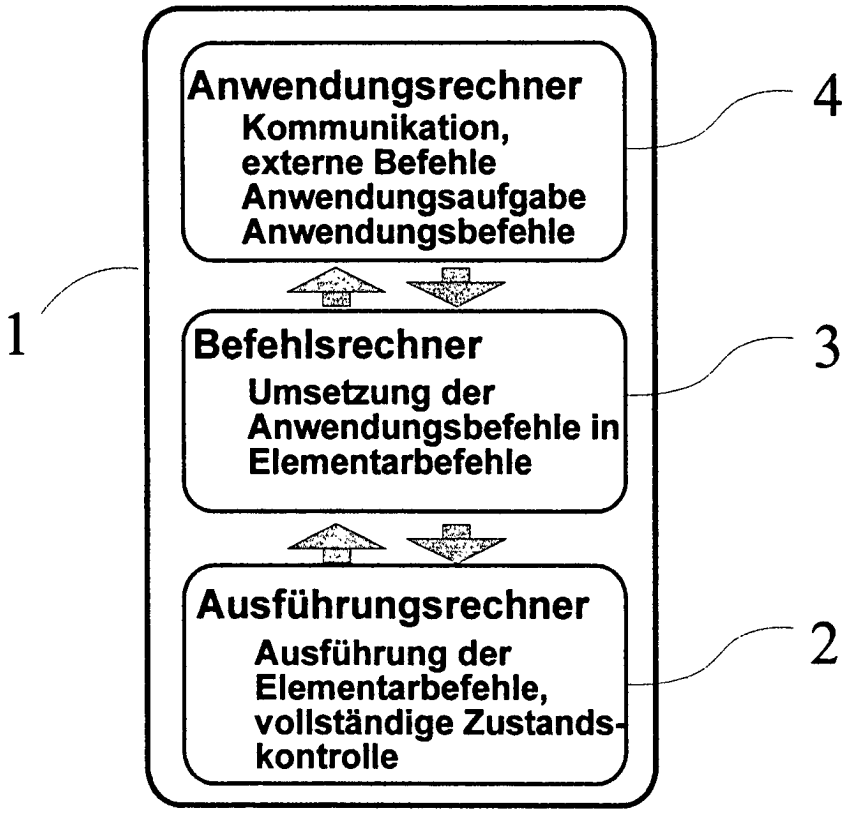


Fig. 1

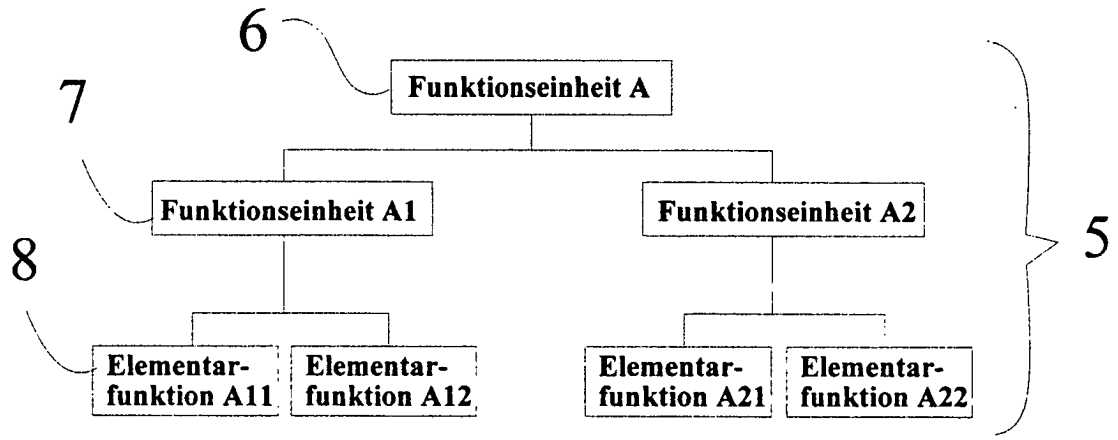


Fig. 2

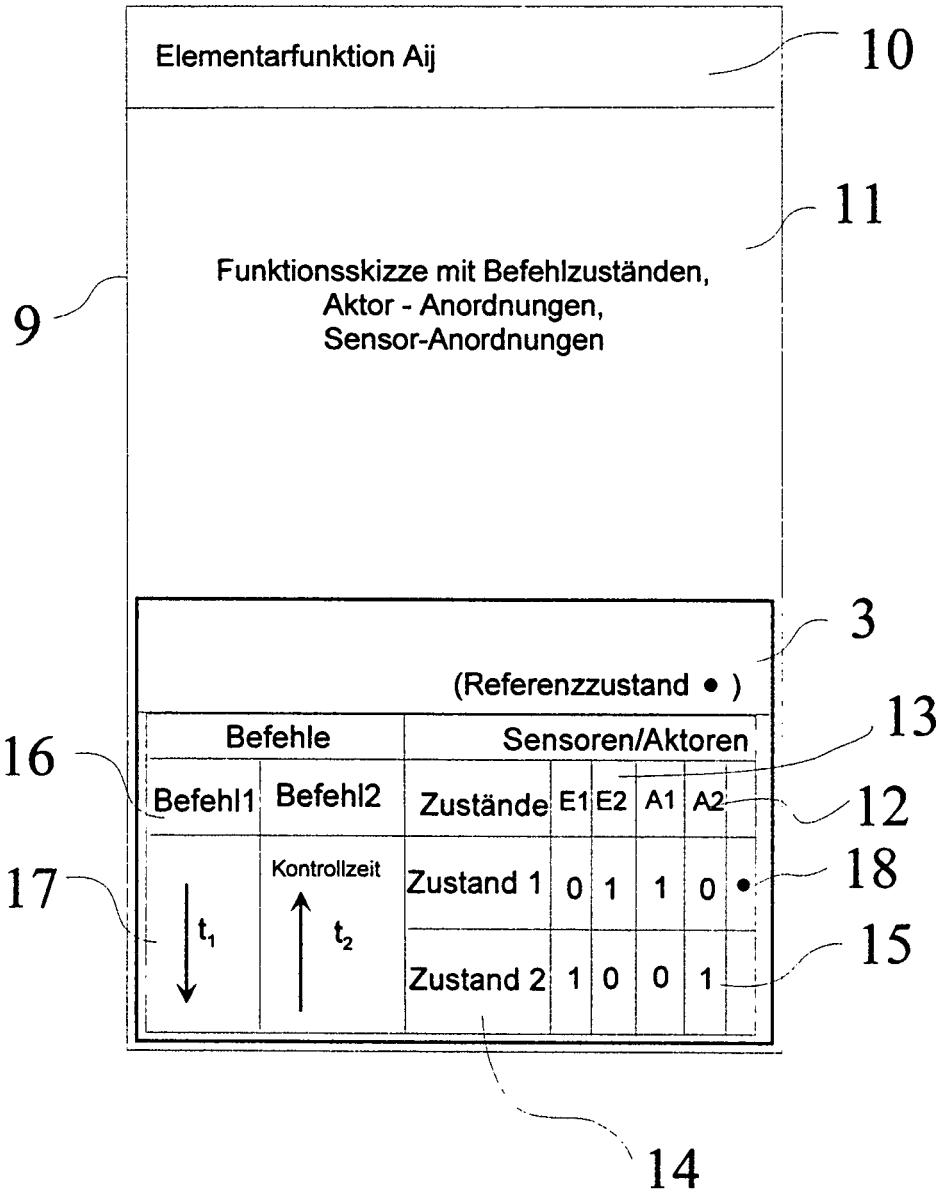

Elementarfunktion Aij
10
11
Funktionsskizze mit Befehlzuständen,
Aktor - Anordnungen,
Sensor-Anordnungen
9
3
(Referenzzustand •)
13
16
Befehle
Sensoren/Aktoren
Befehl1
Befehl2
Zustände
E1
E2
A1
A2
12
17
Kontrollzeit
t1
t2
Zustand 1
0
1
1
0
•
18
Zustand 2
1
0
0
1
15
14

Fig. 3

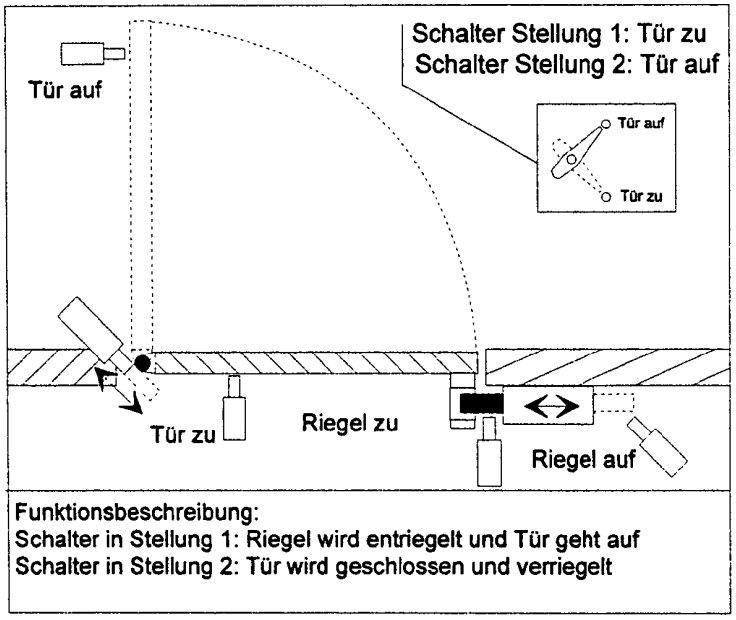


Fig. 4

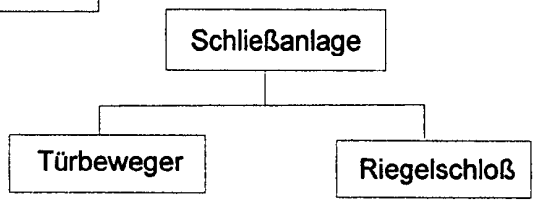
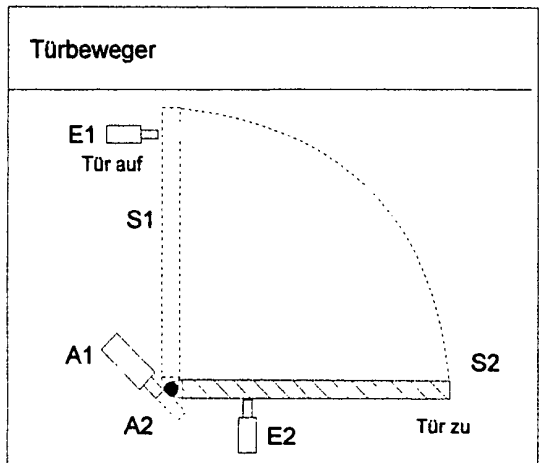


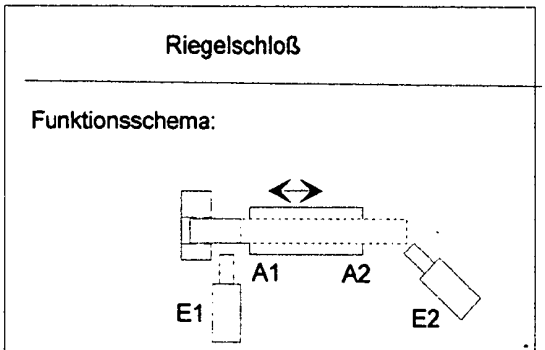
Fig. 5



Funktionsgruppe: **Türbeweger**

(Referenzzustand •)

Befehle		Sensoren/Aktoren					
öffnen!	schließen!	Zustände	E1	E2	A1	A2	
↓ 10s	Kontrollzeit ↑ 12s	ge-schlossen	0	1	1	0	•
		offen	1	0	0	1	



Funktionsgruppe: **Riegelschloß**

(Referenzzustand •)

Befehle		Sensoren/Aktoren					
ent-riegeln	ver-riegeln	Zustände	E1	E2	A1	A2	
↓ 1 s	Kontrollzeit ↑ 2 s	ent-riegelt	0	1	1	0	
		ver-riegelt	1	0	0	1	•

Fig. 6

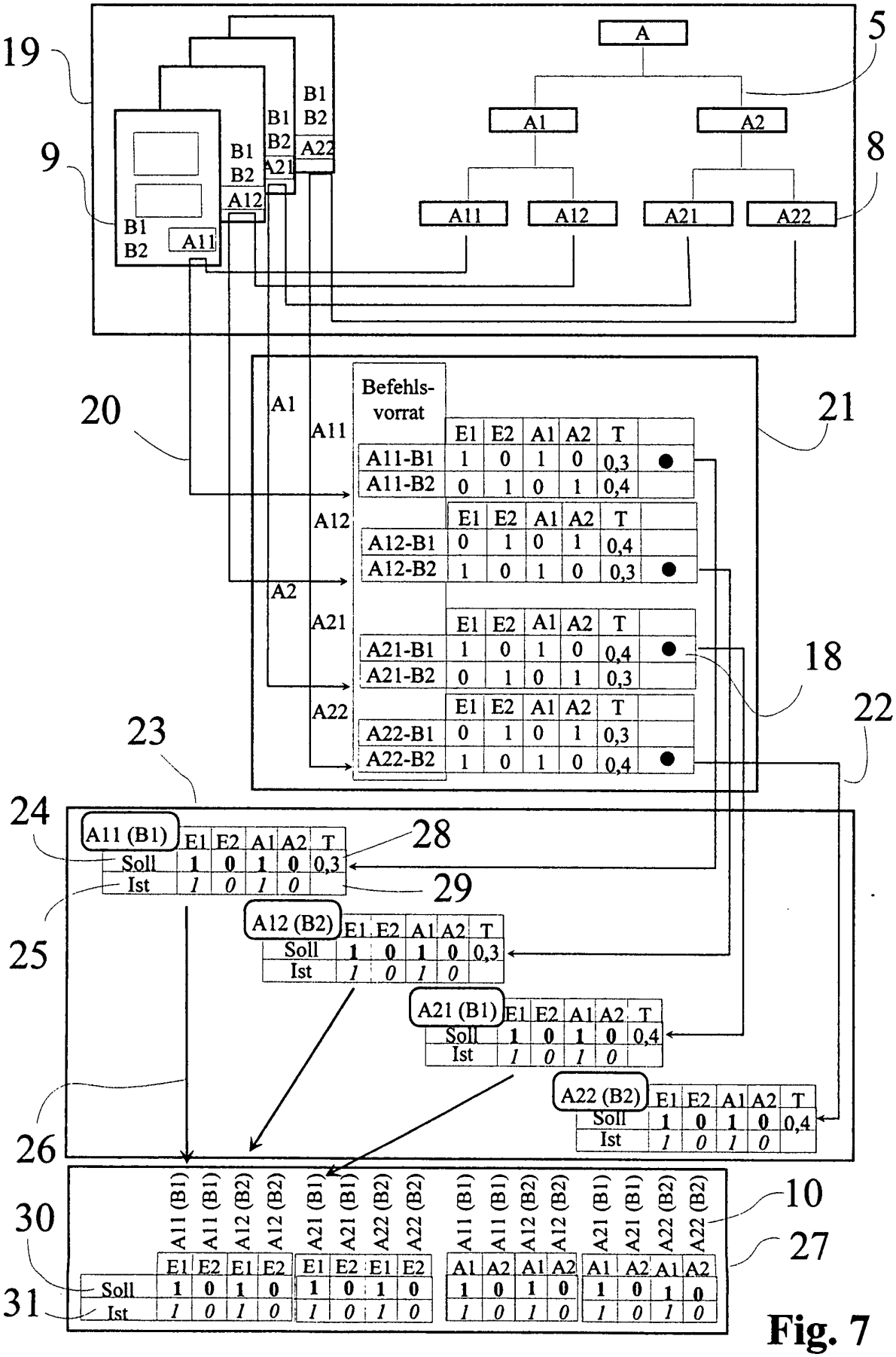

19
9
5
8
A
A1
A2
A11
A12
A21
A22
B1
B2
20
21
Befehls-
vorrat
E1 E2 A1 A2 T
A11-B1 1 0 1 0 0,3
A11-B2 0 1 0 1 0,4
A12-B1 0 1 0 1 0,4
A12-B2 1 0 1 0 0,3
A21-B1 1 0 1 0 0,4
A21-B2 0 1 0 1 0,3
A22-B1 0 1 0 1 0,3
A22-B2 1 0 1 0 0,4
18
22
23
24
25
26
28
29
A11 (B1) E1 E2 A1 A2 T
Soll 1 0 1 0 0,3
Ist 1 0 1 0
A12 (B2) E1 E2 A1 A2 T
Soll 1 0 1 0 0,3
Ist 1 0 1 0
A21 (B1) E1 E2 A1 A2 T
Soll 1 0 1 0 0,4
Ist 1 0 1 0
A22 (B2) E1 E2 A1 A2 T
Soll 1 0 1 0 0,4
Ist 1 0 1 0
30
31
10
27
A11 (B1) A11 (B1) A12 (B2) A12 (B2) A21 (B1) A21 (B1) A22 (B2) A22 (B2) A11 (B1) A11 (B1) A12 (B2) A12 (B2) A21 (B1) A21 (B1) A22 (B2) A22 (B2)
E1 E2 E1 E2 E1 E2 E1 E2 A1 A2 A1 A2 A1 A2 A1 A2
Soll 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0
Ist 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0

Fig. 7

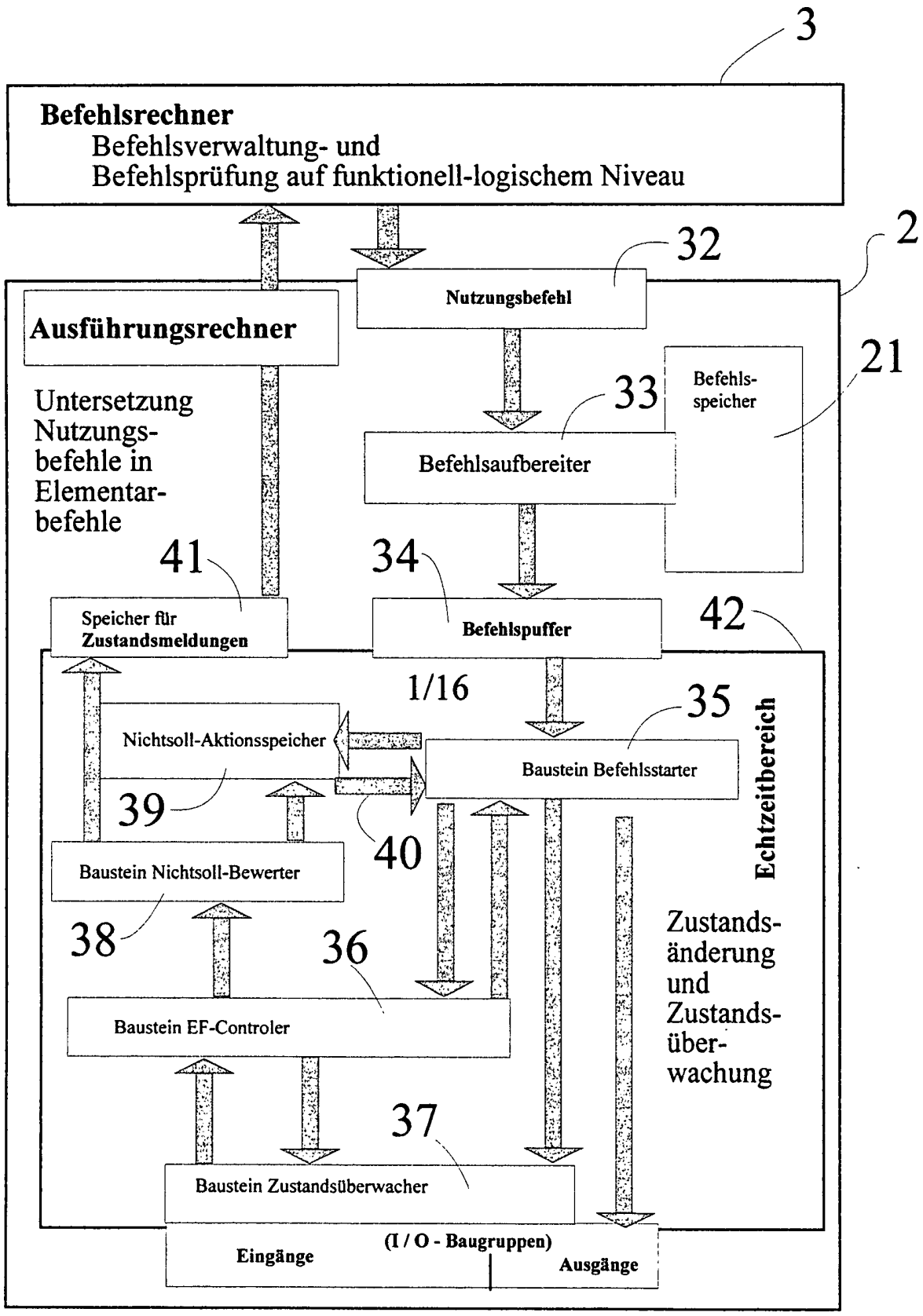

3
Befehlsrechner
Befehlsverwaltung- und
Befehlsprüfung auf funktionell-logischem Niveau
2
32
Nutzungsbefehl
Ausführungsrechner
21
Befehls-
speicher
33
Befehlsaufbereiter
Untersetzung
Nutzungs-
befehle in
Elementar-
befehle
41
34
Speicher für
Zustandsmeldungen
Befehlspuffer
42
1/16
35
Nichtsoll-Aktionsspeicher
Baustein Befehlsstarter
39
40
Echtzeitbereich
Baustein Nichtsoll-Bewerter
38
36
Zustands-
änderung
und
Zustands-
über-
wachung
Baustein EF-Controler
37
Baustein Zustandsüberwacher
(I / O - Baugruppen)
Eingänge
Ausgänge

Fig. 8

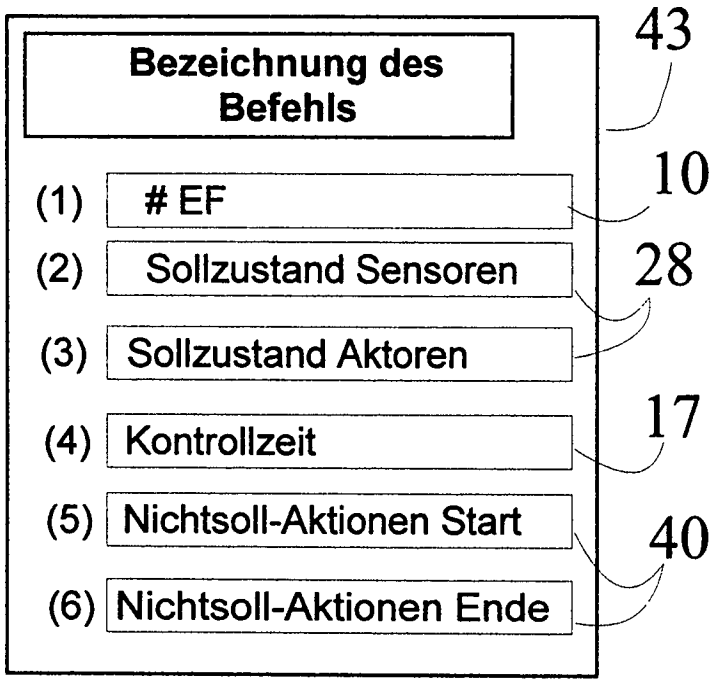

Bezeichnung des Befehls
43
(1) # EF
10
(2) Sollzustand Sensoren
28
(3) Sollzustand Aktoren
(4) Kontrollzeit
17
(5) Nichtsoll-Aktionen Start
40
(6) Nichtsoll-Aktionen Ende

Fig. 9

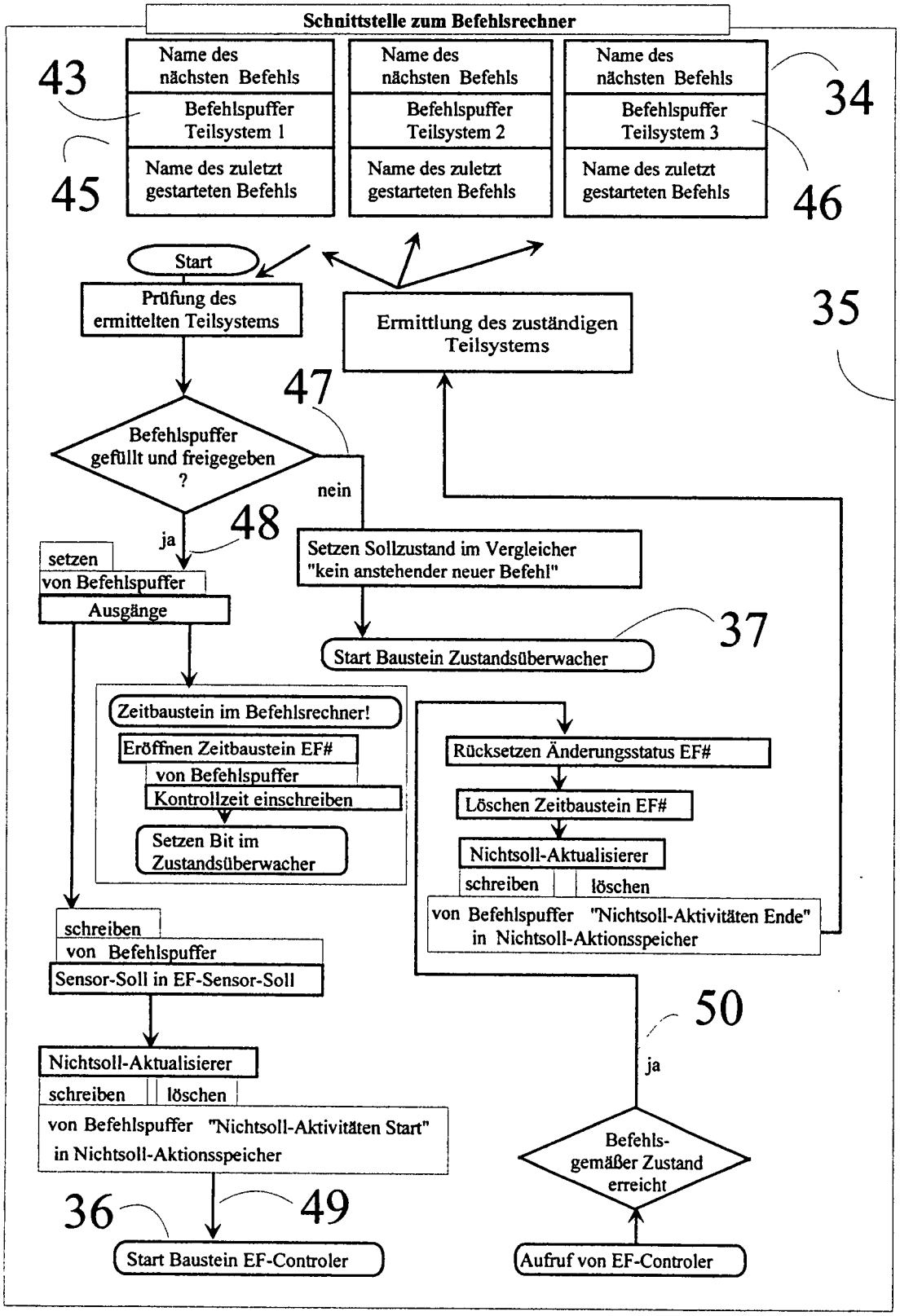

Schnittstelle zum Befehlsrechner
43
Name des nächsten Befehls
Befehlspuffer Teilsystem 1
Name des zuletzt gestarteten Befehls
45
Name des nächsten Befehls
Befehlspuffer Teilsystem 2
Name des zuletzt gestarteten Befehls
Name des nächsten Befehls
Befehlspuffer Teilsystem 3
Name des zuletzt gestarteten Befehls
34
46
Start
Prüfung des ermittelten Teilsystems
Ermittlung des zuständigen Teilsystems
35
47
Befehlspuffer gefüllt und freigegeben ?
nein
ja
48
setzen von Befehlspuffer Ausgänge
Setzen Sollzustand im Vergleicher "kein anstehender neuer Befehl"
Start Baustein Zustandsüberwacher
37
Zeitbaustein im Befehlsrechner!
Eröffnen Zeitbaustein EF#
von Befehlspuffer
Kontrollzeit einschreiben
Setzen Bit im Zustandsüberwacher
Rücksetzen Änderungsstatus EF#
Löschen Zeitbaustein EF#
Nichtsoll-Aktualisierer
schreiben
löschen
von Befehlspuffer "Nichtsoll-Aktivitäten Ende" in Nichtsoll-Aktionsspeicher
schreiben von Befehlspuffer Sensor-Soll in EF-Sensor-Soll
50
ja
Nichtsoll-Aktualisierer
schreiben
löschen
von Befehlspuffer "Nichtsoll-Aktivitäten Start" in Nichtsoll-Aktionsspeicher
Befehls-gemäßer Zustand erreicht
36
49
Start Baustein EF-Controler
Aufruf von EF-Controler

Fig. 10

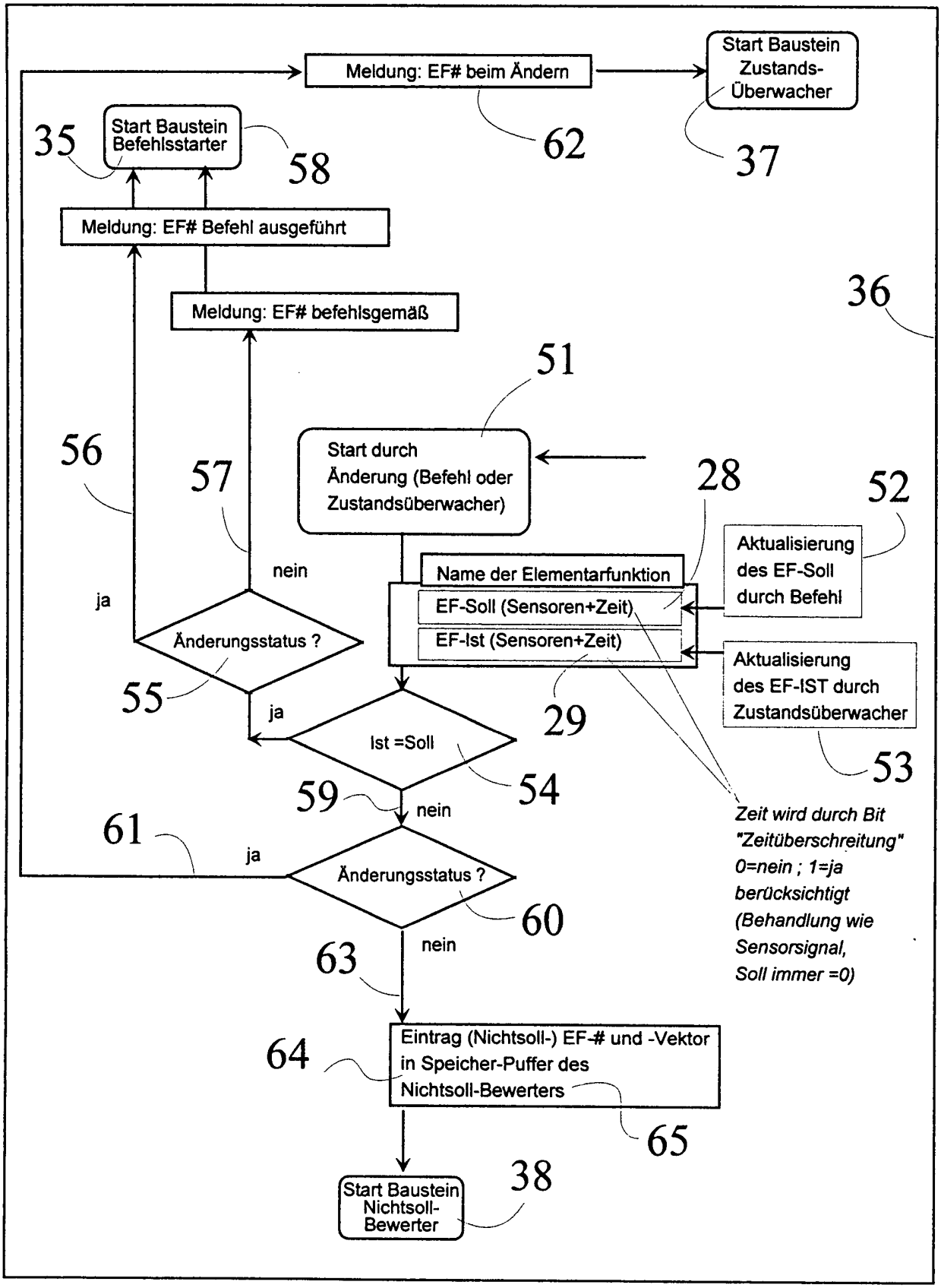

Meldung: EF# beim Ändern
Start Baustein Zustands-Überwacher
62
37
35
Start Baustein Befehlsstarter
58
Meldung: EF# Befehl ausgeführt
Meldung: EF# befehlsgemäß
36
51
Start durch Änderung (Befehl oder Zustandsüberwacher)
56
57
28
52
Aktualisierung des EF-Soll durch Befehl
Name der Elementarfunktion
EF-Soll (Sensoren+Zeit)
EF-Ist (Sensoren+Zeit)
Aktualisierung des EF-IST durch Zustandsüberwacher
nein
ja
Änderungsstatus ?
55
ja
Ist =Soll
29
54
53
59
nein
61
ja
Änderungsstatus ?
60
nein
63
Zeit wird durch Bit "Zeitüberschreitung" 0=nein ; 1=ja berücksichtigt (Behandlung wie Sensorsignal, Soll immer =0)
64
Eintrag (Nichtsoll-) EF-# und -Vektor in Speicher-Puffer des Nichtsoll-Bewerters
65
Start Baustein Nichtsoll-Bewerter
38

Fig. 11

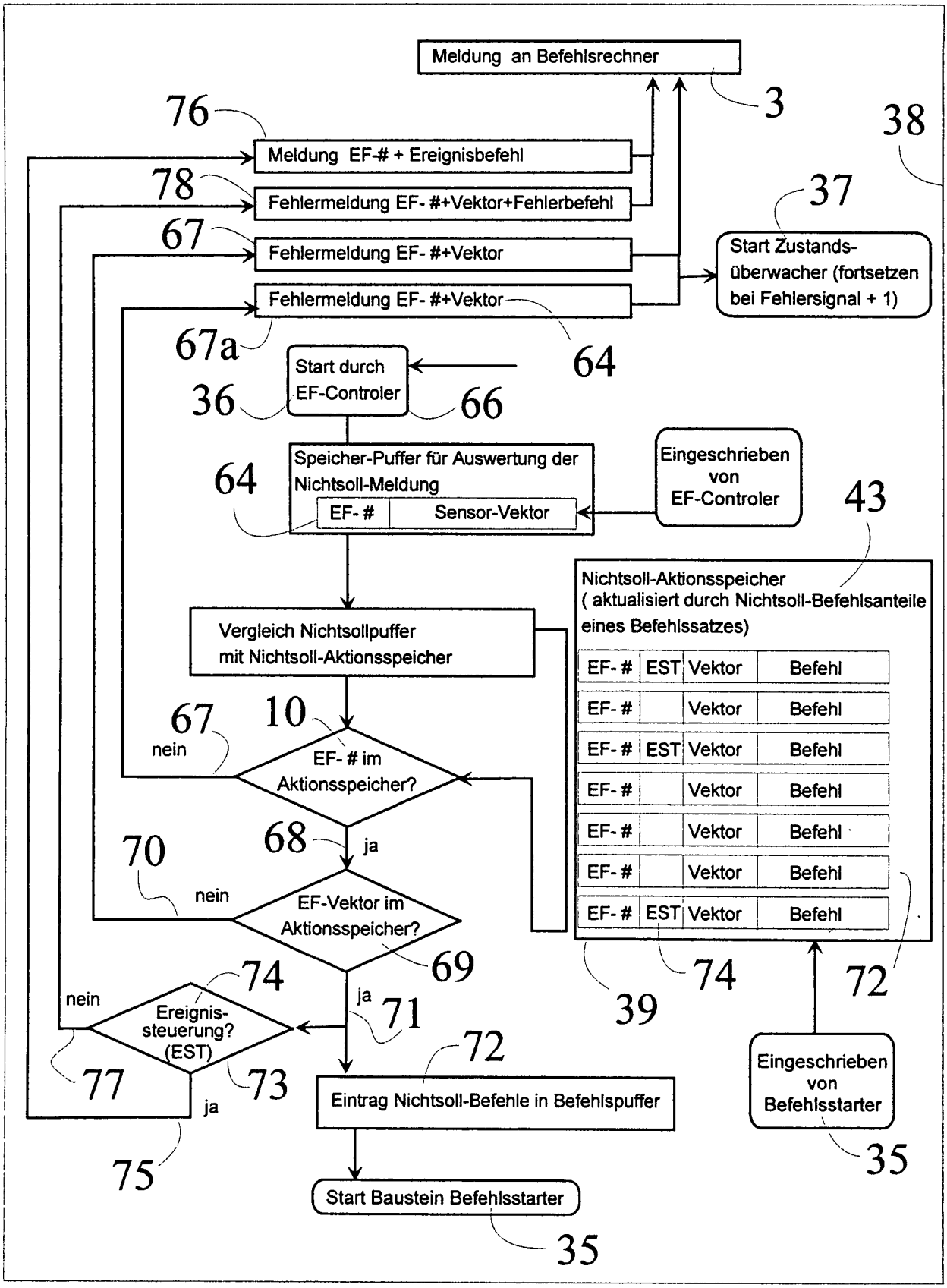

Meldung an Befehlsrechner
3
38
76
Meldung EF-# + Ereignisbefehl
78
Fehlermeldung EF- #+Vektor+Fehlerbefehl
37
67
Fehlermeldung EF- #+Vektor
Start Zustands-überwacher (fortsetzen bei Fehlersignal + 1)
Fehlermeldung EF- #+Vektor
67a
64
Start durch EF-Controler
36
66
Speicher-Puffer für Auswertung der Nichtsoll-Meldung
64
EF- #
Sensor-Vektor
Eingeschrieben von EF-Controler
43
Nichtsoll-Aktionsspeicher (aktualisiert durch Nichtsoll-Befehlsanteile eines Befehlssatzes)
Vergleich Nichtsollpuffer mit Nichtsoll-Aktionsspeicher
EF- # | EST | Vektor | Befehl
EF- # | | Vektor | Befehl
EF- # | EST | Vektor | Befehl
EF- # | | Vektor | Befehl
EF- # | | Vektor | Befehl
EF- # | | Vektor | Befehl
EF- # | EST | Vektor | Befehl
67
10
nein
EF- # im Aktionsspeicher?
68
ja
70
nein
EF-Vektor im Aktionsspeicher?
69
74
nein
ja
71
Ereignis-steuerung? (EST)
77
73
ja
72
Eintrag Nichtsoll-Befehle in Befehlspuffer
75
Start Baustein Befehlsstarter
35
39
74
72
Eingeschrieben von Befehlsstarter
35

Fig. 12

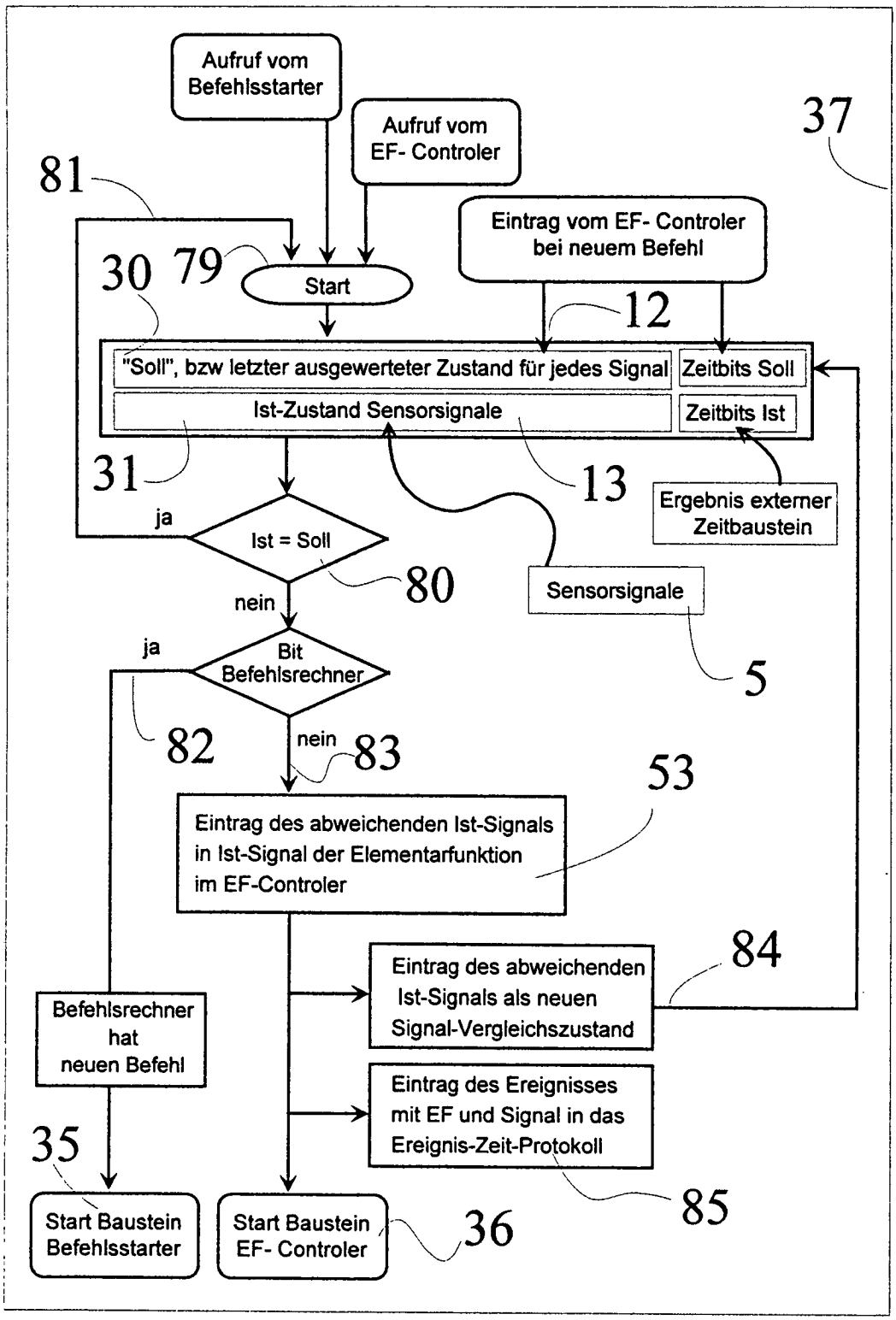

Aufruf vom Befehlsstarter
Aufruf vom EF- Controler
37
81
Eintrag vom EF- Controler bei neuem Befehl
30
79
Start
12
"Soll", bzw letzter ausgewerteter Zustand für jedes Signal
Zeitbits Soll
Ist-Zustand Sensorsignale
Zeitbits Ist
31
13
Ergebnis externer Zeitbaustein
ja
Ist = Soll
80
Sensorsignale
nein
ja
Bit Befehlsrechner
5
82
nein
83
53
Eintrag des abweichenden Ist-Signals in Ist-Signal der Elementarfunktion im EF-Controler
84
Eintrag des abweichenden Ist-Signals als neuen Signal-Vergleichszustand
Befehlsrechner hat neuen Befehl
Eintrag des Ereignisses mit EF und Signal in das Ereignis-Zeit-Protokoll
35
85
Start Baustein Befehlsstarter
Start Baustein EF- Controler
36

Fig. 13

85

(1)	(2)	(3)	(4)
Elementar-funktion	Signal		Systemzeit
Name	Bez.		Startzeit + ms
A11	E1	1	123,456 10E3
A12	E1	0	123,789 10E3
A11	E2	0	223,789 10E3
A21	E1	1	323,789 10E3
A22	E2	0	423,789 10E3
A22	E1	1	523,789 10E3
A21	E2	0	823,789 10E3
A23	E1	0	923,789 10E3
A31	E2	1	123,789 10E4
A11	E1	1	323,789 10E4
○	○	○	○
○	○	○	○
○	○	○	○
○	○	○	○
○	○	○	○
○	○	○	○

Fig. 14

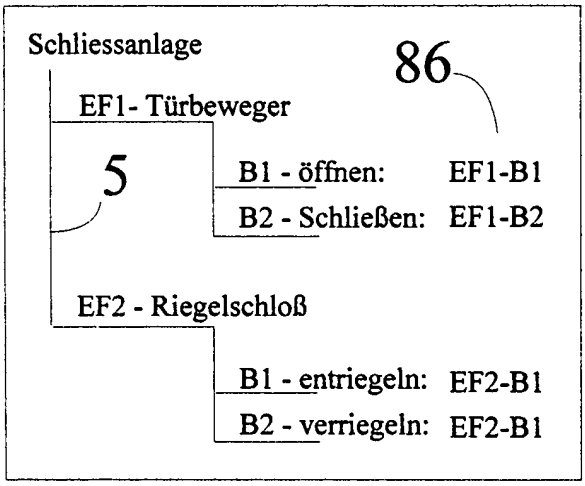


Fig. 15

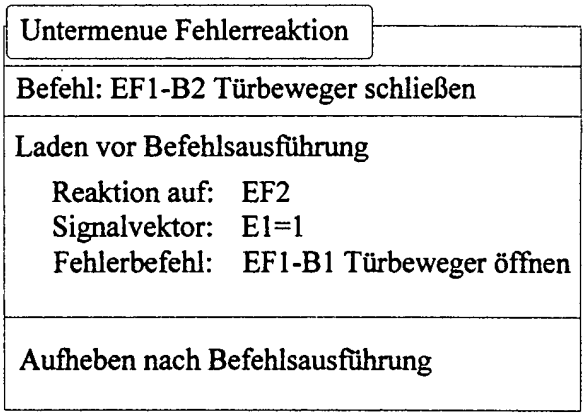


Fig. 18

87 88

Befehlsname:	Tür auf	Befehlsinhalt:	Tür öffnen		
Befehlsfolge:	Zu setzende Sperren		Aufzuhebende Sperren		
Zu aktivierende Befehle	während der Ausführung	nach der Ausführung	während der Ausführung	nach der Ausführung	
EF2-B1 1.) Riegel entriegeln	EF1-B1 Türbeweger öffnen			EF1-B1 Türbeweger öffnen	
EF1-B1 2.) Türbeweger öffnen	EF2-B2 Riegel verriegeln	EF2-B2 Riegel verriegeln			

Befehlsname:	Tür zu	Befehlsinhalt:	Tür schließen		
Befehlsfolge:	Zu setzende Sperren		Aufzuhebende Sperren		
Zu aktivierende Befehle	während der Ausführung	nach der Ausführung	während der Ausführung	nach der Ausführung	
EF1-B2 1.) Türbeweger schließen	EF2-B2 Riegel verriegeln			EF2-B2 Riegel verriegeln	
EF2-B2 2.) Riegel verriegeln	EF1-B1 Türbeweger öffnen	EF1-B1 Türbeweger öffnen			

Fig. 16

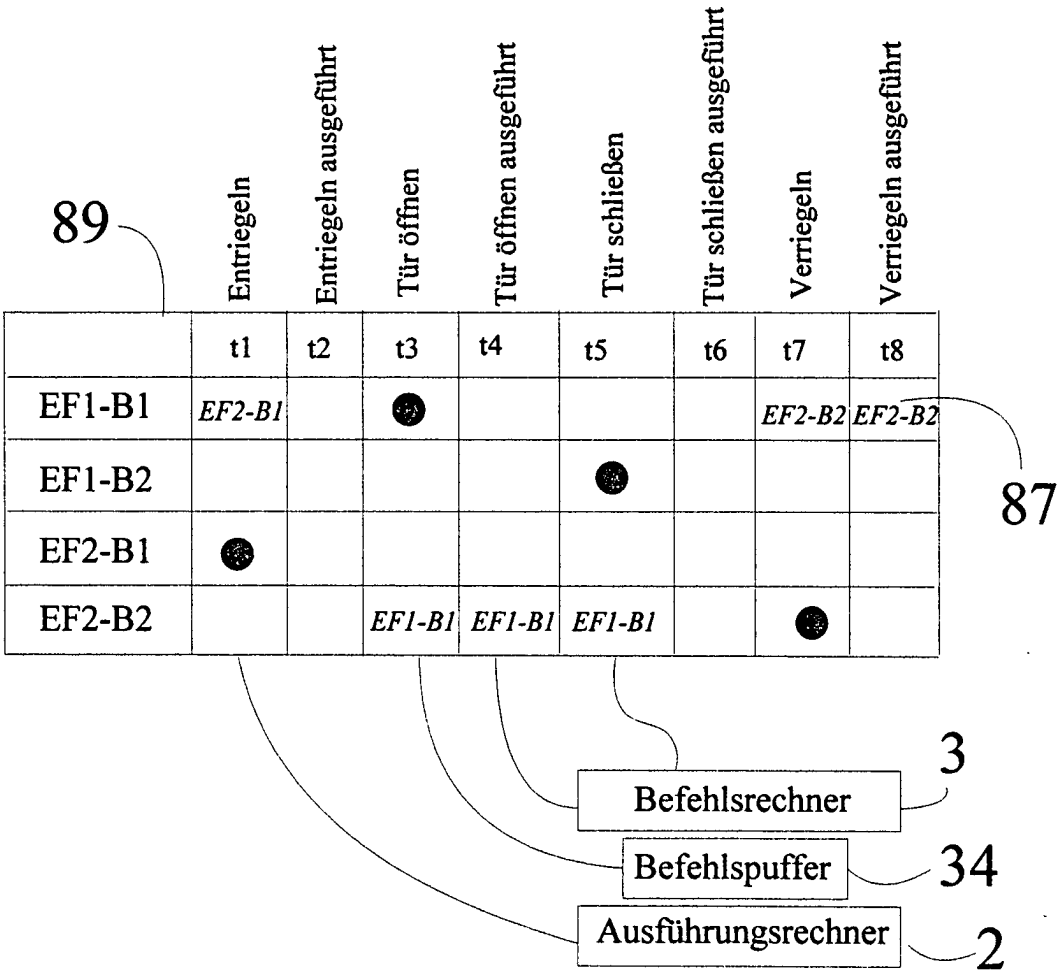
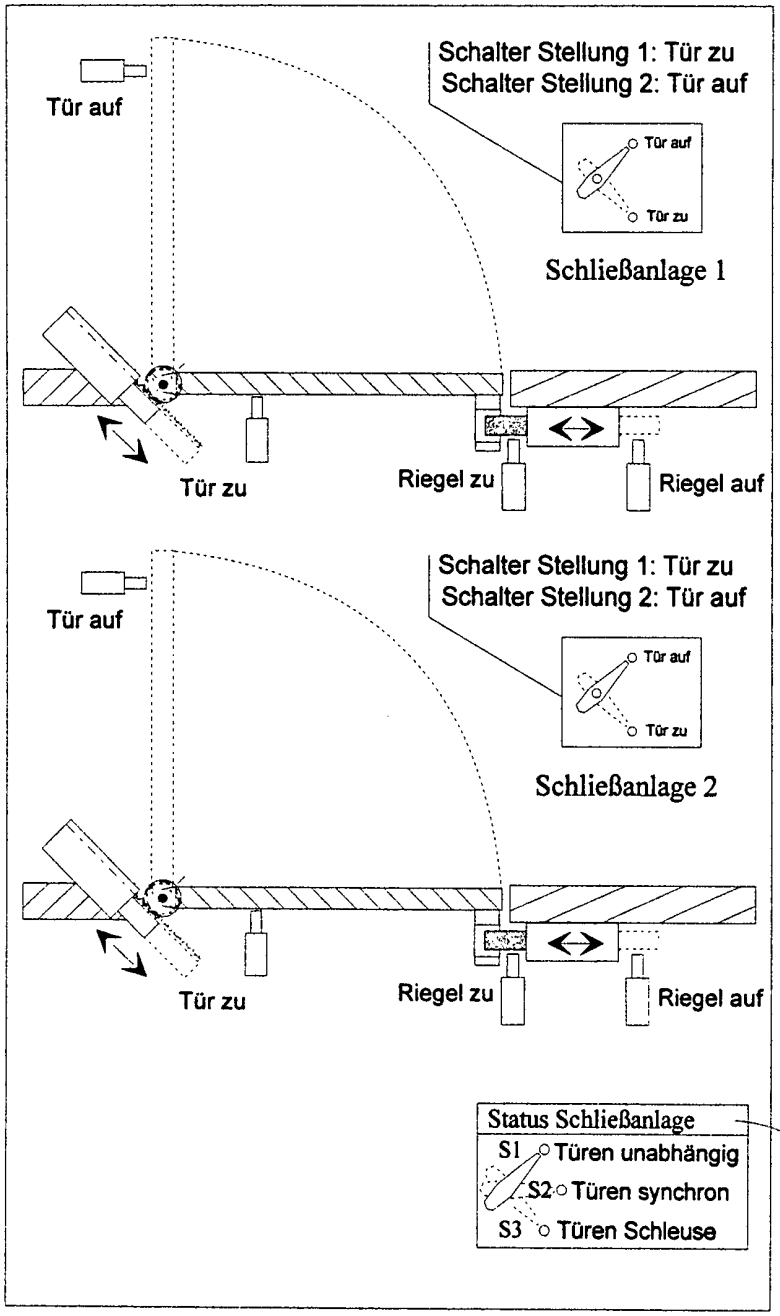

89
Entriegeln
Entriegeln ausgeführt
Tür öffnen
Tür öffnen ausgeführt
Tür schließen
Tür schließen ausgeführt
Verriegeln
Verriegeln ausgeführt
t1
t2
t3
t4
t5
t6
t7
t8
EF1-B1
EF1-B2
EF2-B1
EF2-B2
EF2-B1
EF2-B2
EF1-B1
EF1-B1
EF1-B1
87
Befehlsrechner
3
Befehlspuffer
34
Ausführungsrechner
2

Fig. 17


Schalter Stellung 1: Tür zu
Schalter Stellung 2: Tür auf
Tür auf
Tür zu
Schließanlage 1
Tür auf
Tür zu
Riegel zu
Riegel auf
Schalter Stellung 1: Tür zu
Schalter Stellung 2: Tür auf
Tür auf
Tür zu
Schließanlage 2
Tür auf
Tür zu
Riegel zu
Riegel auf
Status Schließanlage
S1 Türen unabhängig
S2 Türen synchron
S3 Türen Schleuse
90

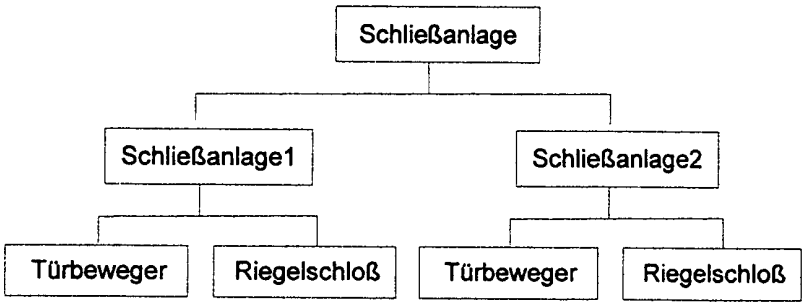

Schließanlage
Schließanlage1
Schließanlage2
Türbeweger
Riegelschloß
Türbeweger
Riegelschloß

Fig. 19

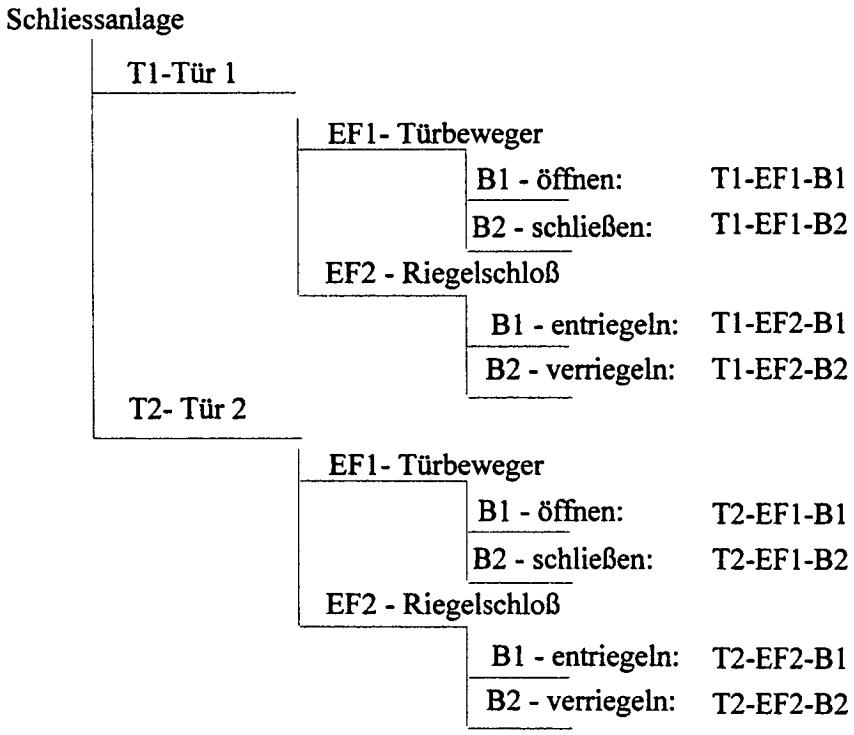

Schliessanlage
T1-Tür 1
EF1- Türbeweger
B1 - öffnen: T1-EF1-B1
B2 - schließen: T1-EF1-B2
EF2 - Riegelschloß
B1 - entriegeln: T1-EF2-B1
B2 - verriegeln: T1-EF2-B2
T2- Tür 2
EF1- Türbeweger
B1 - öffnen: T2-EF1-B1
B2 - schließen: T2-EF1-B2
EF2 - Riegelschloß
B1 - entriegeln: T2-EF2-B1
B2 - verriegeln: T2-EF2-B2

Fig. 20

91 92

Status Schließanlage: S1 (Türen unabhängig zu bedienen)

Befehlsvorrat	Befehlsgerät	Befehlsbibliothek
Tür 1 öffnen	Tür 1 Schalter1	T1-Tür auf
Tür 1 schließen	Tür 1 Schalter2	T1-Tür zu
Tür 1 öffnen	Tür 2 Schalter1	T2-Tür auf
Tür 1 schließen	Tür 2 Schalter2	T2-Tür zu

Status Schließanlage: S2 (Türen synchron) 93

Befehlsvorrat	Befehlsgerät	Befehlsbibliothek	
Türen öffnen	Änderung von Tür-Schalter 1 oder 2	Parallelbefehl:	T1-Tür auf T2-Tür auf
Türen schließen	Änderung von Tür-Schalter 1 oder 2	Parallelbefehl:	T1-Tür zu T2-Tür zu

Status Schließanlage: S3 (Türen Schleuse)

Befehlsvorrat	Befehlsgerät	Befehls-bibliothek	während Ausführung	nach Ausführung
Türe1 öffnen	Tür 1 Schalter1	T1-Tür auf	Sperre Tür 2	Sperre Tür 2
Türe1 schließen	Tür 1 Schalter2	T1-Tür zu	Sperre Tür 2	
Türe2 öffnen	Tür 2 Schalter1	T2-Tür auf	Sperre Tür 1	Sperre Tür 1
Türe2 schließen	Tür 2 Schalter2	T2-Tür zu	Sperre Tür 1	

Fig. 21

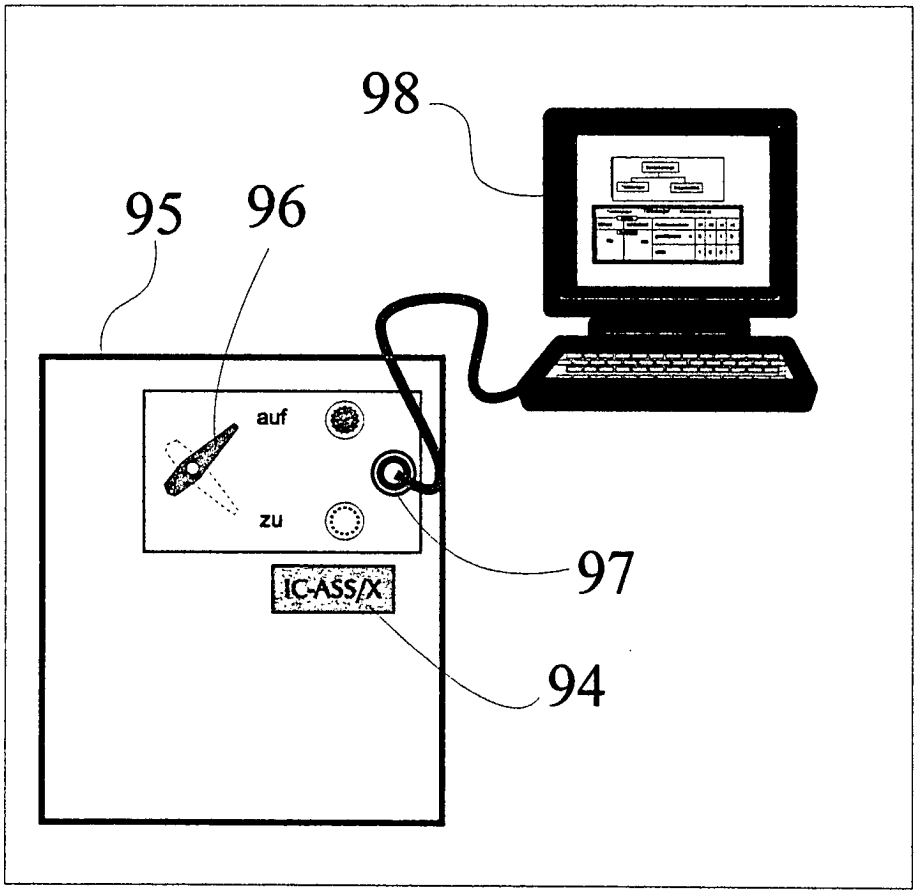


Fig. 22