FAPS

Walter
Colombo

27.9.96

Konrad Grampp

Rechnerunterstützung bei Test und Schulung an Steuerungssoftware von SMD-Bestücklinien

Konrad Grampp

Rechnerunterstützung bei Test und Schulung an Steuerungssoftware von SMD-Bestücklinien

Herausgegeben von
Professor Dr.-Ing. Klaus Feldmann,
Lehrstuhl für
Fertigungsautomatisierung und Produktionssystematik
FAPS

HANSER

Carl Hanser Verlag München Wien

Als Dissertation genehmigt von der Technischen Fakultät
der Friedrich-Alexander-Universität Erlangen-Nürnberg

Tag der Einreichung:	1. Juli 1994
Tag der Promotion:	17. Oktober 1994
Dekan:	Prof. Dr. Dr. h. c. F. Durst
Berichterstatter:	Prof. Dr.-Ing. K. Feldmann
	Prof. Dr. F. Hofmann

Die Deutsche Bibliothek - CIP - Einheitsaufnahme

Grampp, Konrad:
Rechnerunterstützung bei Test und Schulung an
Steuerungssoftware von SMD-Bestücklinien / Konrad Grampp
- München ; Wien : Hanser, 1995
(Fertigungstechnik - Erlangen ; 40)
Zugl.: Erlangen, Nürnberg, Univ., Diss., 1994
ISBN 3-446-18173-3
NE: GT

Herstellung: Gruner Druck GmbH, Erlangen-Eltersdorf
Printed in Germany

Vorwort

Die vorliegende Dissertation entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Fertigungsautomatisierung und Produktionssystematik der Friedrich-Alexander-Universität Erlangen-Nürnberg.

Herrn Prof. Dr.-Ing. K. Feldmann, dem Leiter dieses Lehrstuhls am Institut für Fertigungstechnik, danke ich für die großzügige Förderung bei der Durchführung dieser Arbeit.

Herrn Prof. Dr. F. Hofmann, dem Leiter des Lehrstuhls für Betriebssysteme danke ich für die Übernahme des Korreferats.

Bedanken möchte ich mich auch bei allen Kollegen des Lehrstuhls, vor allem bei denen aus der Steuerungs- und Sensorgruppe, sowie allen Studenten und wissenschaftlichen Hilfskräften, die durch ihre Unterstützung zum Gelingen dieser Arbeit beigetragen haben. Besonders erwähnen möchte ich hierbei Herrn Dipl.-Ing. Christian Reh, Herrn cand.-ing. Klaus Wölflick, Herrn Dipl.-Ing. Thomas Herrmann, Herrn Dipl.-Inf. Peter Koch, Herrn Dipl.-Inf. Hartmut Bonny, Herrn cand.-inf. Grigorios Bisbinikakis, Herrn cand.-inf. Johannes Köttke und Herrn Dipl.-Inf Stefan Gryscyk.

Mein besonderer Dank gilt jedoch meiner Frau Heike für die moralische Unterstützung und das dauerhafte Verständnis.

Konrad Grampp

1. Einleitung

Im Bereich der Elektronikproduktion setzt sich die Bestückung von Leiterplatten mit oberflächenmontierbaren Bauelementen gegenüber der Technik mit bedrahteten Bauelementen immer mehr durch (Bild 1). Die Vorteile der SMD-Technik liegen unter anderem in einer einfacheren Prozeßkette bei der Bestückung der Leiterplatten und einer einfacheren Handhabbarkeit der Bauelemente. Die automatisierte Komplettbearbeitung der Leiterplatten in SMD-Bauweise, inklusive dem Auftrag von Kleber oder Lotpaste, der Bestückung und dem Löten, wird durch SMD-Bestücklinien ermöglicht.

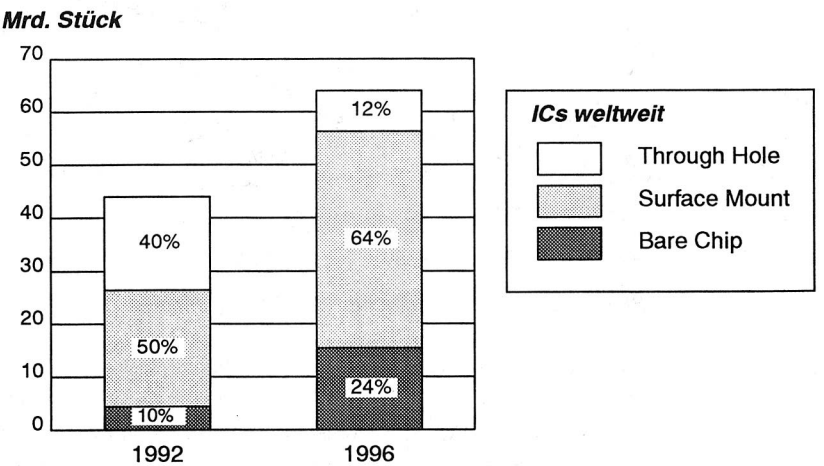


Bild 1: Stückzahl prognose der weltweit verarbeiteten ICs (Quelle: Siemens) /72/

Die technologische Entwicklung der SMD-Bestücksysteme wird durch zwei grundlegende Faktoren beeinflußt. Auf der einen Seite stehen die Anforderungen der Anwender nach hoher Bestückleistung und hoher Bestückqualität bei gleichzeitig hoher Flexibilität, um angesichts steigender Variantenvielfalt und sinkenden Produktlebensdauern wirtschaftlich fertigen zu können. Auf der anderen Seite stehen die Anforderungen durch den technologischen Fortschritt, der sich im Bauelementebereich durch die fortschreitende Miniaturisierung ausdrückt. Die Miniaturisierung führt einerseits zu immer kleineren Bauelementen (Bild 2), andererseits zu hochpoligen ICs

in Fine-Pitch-Gehäusen mit mehreren hundert Anschlußbeinen und sehr kleinem Rastermaß (Bild 3) /113,71/. Beide Extreme fordern von den Bestückautomaten eine hohe Bestückgenauigkeit.

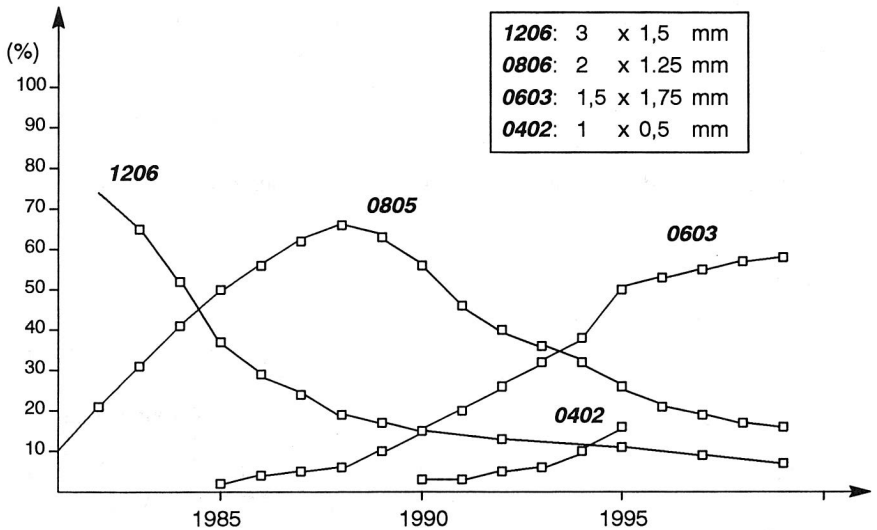


Bild 2: Passive SMD-Bauformen: Trendverlauf (Quelle: Panasonic)

Zur Erhaltung ihrer Marktfähigkeit sind die Hersteller von Bestücksystemen gezwungen, auf die oben genannten Anforderungen zu reagieren und immer wieder neue Maschinenkonzepte und Zusatzgeräte zu entwickeln. Durch diese Entwicklungen steigt auch die Komplexität der Steuerungssoftware stark an, was zu sehr langen Entwicklungszeiten mit enormen Kosten führt. Ein wichtiges Kriterium für eine rechtzeitige Marktreife der Systeme stellt die frühzeitige Durchführung von Tests der Steuerungssoftware mit geeigneten Testwerkzeugen dar.

Nach der Abnahme eines Bestücksystems ist der Anwender dafür verantwortlich, eine möglichst hohe Verfügbarkeit seiner Maschine zu erreichen /44/. Neben organisatorischen Ursachen sind viele Stillstände durch unzureichende Mitarbeiterqualifikation bedingt. Um Stillstände durch Fehlbedienung und Qualifikationsmängel zu vermeiden, benötigt das Bedienpersonal Einblicke in die Struktur und die inneren Abläufe des Systems, insbesondere der Linienrechnersoftware, um einen optimierten Betrieb zu ermöglichen und auftretende Störungen schnell und sicher zu beheben.

Zur Ausbildung des Personals sind also geeignete Maßnahmen zu treffen, die sowohl die Vermittlung des theoretischen Hintergrunds als auch die praktische Erfahrung im Umgang mit dem System betreffen.

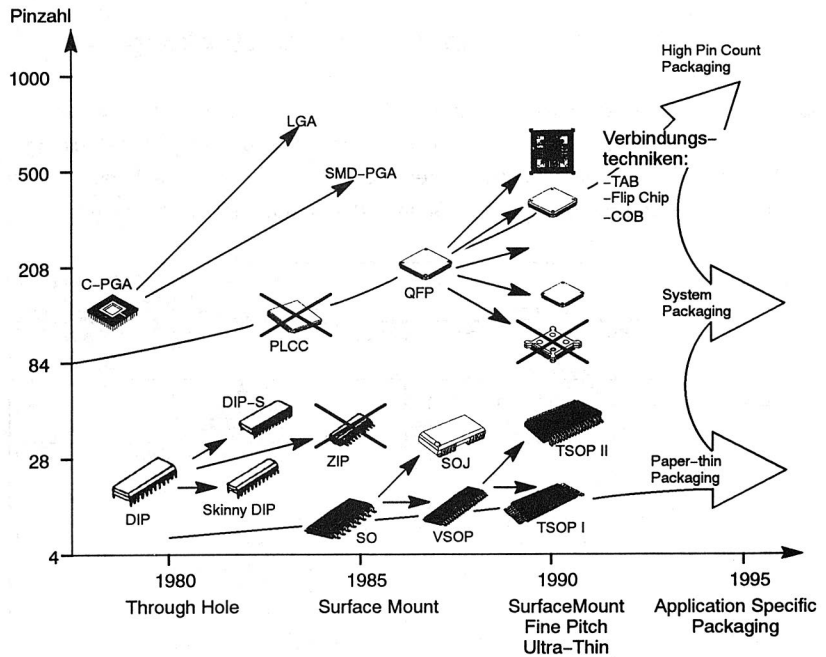


Bild 3: Technologischer Wandel bei IC-Gehäusen (Quelle: Siemens) /72/

Das Ziel dieser Arbeit liegt nun darin, die Simulation als geeignetes Werkzeug zum frühzeitigen Test der Linienrechnersoftware vorzustellen und an Hand des entwickelten Systems zu verifizieren. Der Aufbau und die Funktionsweise des Systems wird vorgestellt und es wird gezeigt, daß dieses System auch zur Personalausbildung eingesetzt werden, kann um die notwendigen Steuerungsvorgänge zu trainieren.

Zur Vermittlung des theoretischen Hintergrunds werden verschiedene Ausbildungsformen gegenübergestellt und die Vorteile des computerunterstützen Lernens herausgearbeitet. Durch die Kopplung des entwickelten Lernprogramms mit der Simulation entsteht ein Gesamtsystem, mit dem sowohl Theorie als auch Praxis vermittelt werden.

2. Aufbau und Steuerung von SMD-Bestücklinien

2.1 Verfahrensketten in der SMD-Bestückung

Bei der konventionellen Einsteckmontage (***THD***: Through Hole Devcie) werden die Anschlußdrähte der Bauelemente von der Leiterplattenoberseite (Bestückungsseite) in die entsprechenden Löcher gesteckt. Die elektrische und mechanische Verbindung der Anschlüsse erfolgt durch Schwall-Löten auf der Leiterbahnseite.

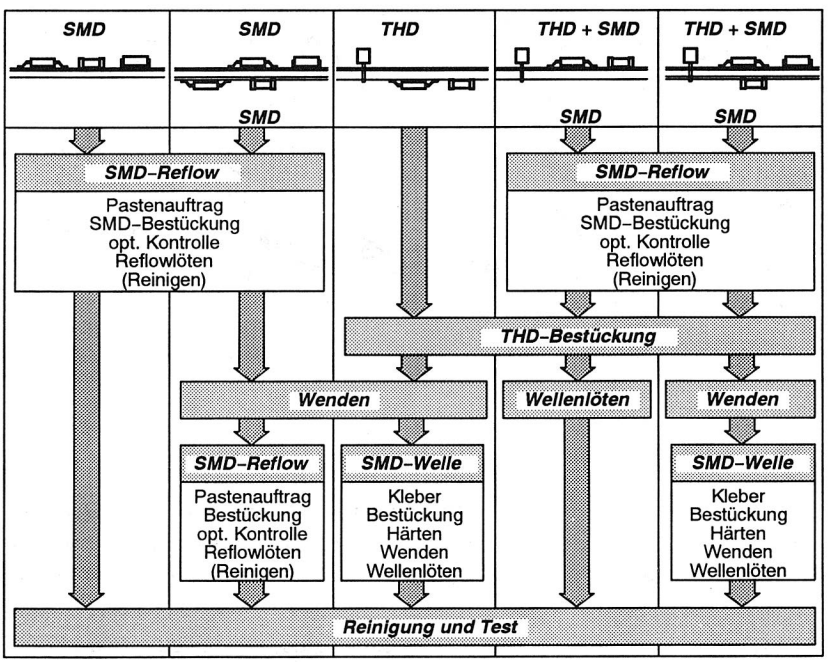


Bild 4: Verfahrensketten in der SMD-Bestückung /26/

Bei den SMD-Bauelementen dagegen wird die mechanische Verbindung und der elektrische Kontakt über metallische Kappen (z. B. Melf oder MiniMelf, siehe Tabelle 1) oder die Anschlußbeine (Gullwing oder 'J'-Ausführung) hergestellt. Die Fixierung der Bauelemente vor dem Löten ist abhängig von dem eingesetzten Lötverfahren.

Beim Reflow-Löten erfolgt die Fixierung durch Lotpaste, die mittels Siebdruck oder Dispensen vor dem Bestücken auf die Lotflächen aufgebracht wird. Beim Schwall-Löten, das bei beidseitiger Bestückung der Leiterplatte oder bei gemischter Bestükkung eingesetzt wird, werden die SMD-Bauelemente auf die Leiterplatte geklebt.

In Bild 4 werden die wesentlichen Verfahrensketten bei der SMD-Bestückung vorgestellt. Da noch nicht alle Bauelementetypen in SMD-Bauform erhältlich sind, wird in vielen Fällen eine gemischte Bestückung mit bedrahteten und SMD-Bauelementen durchgeführt */26/*.

2.1.1 Vorteile der SMD-Technik gegenüber der bedrahteten Technik

SMD-Bauelemente können durch den Wegfall der Anschlußdrähte erheblich kleiner ausgeführt werden als bedrahtete Bauelemente. Sie benötigen dadurch weniger Platz auf der Leiterplatte wodurch die Größen der Baugruppen sinken. Da auch die Bestückwerkzeuge bei der SMD-Technik erheblich kleiner sind wie bei der bedrahteten Technik, entfallen Kollisionsprobleme zwischen dem Bestückwerkzeug und bereits bestückten Bauelementen. Durch die Möglichkeit, Leiterplatten beidseitig mit SMD-Bauelementen zu bestücken, kann die Leiterplattenfläche bis zu 50% reduziert werden. Darüberhinaus entfällt bei der SMD-Technik die Bauteilevorbereitung (Schneiden und Biegen der Anschlußdrähte). Durch den Wegfall der Anschlußdrähte erhöht sich auch die mechanische Belastbarkeit der Baugruppen gegenüber Schock und Vibration. Insgesamt können durch die einfachere Handhabbarkeit der SMD-Bauelemente die Fertigungskosten gegenüber der herkömmlichen bedrahteten Leiterplattentechnik reduziert werden.

2.1.2 Gehäuseformen von SMD-Bauelementen

SMD-Bauelemente werden in unterschiedlichen standardisierten Gehäuseformen hergestellt, die sich durch ihre Form, ihre Abmaße und die Anzahl und Anordnung der Anschlüsse unterscheiden (Bild 1). Zur Bestückung werden die Bauelemente mittels Vakuumpipetten aufgenommen und zur Bestückposition transportiert. Um die unterschiedlich großen und geformten (zylindrisch, kubisch) Bauelemente handhaben zu können, werden Pipetten mit unterschiedlicher Ansaugfläche eingesetzt.

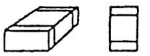
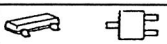
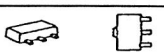
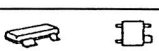
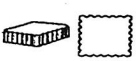
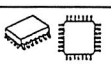
Bauform	*Typ*	*Gehäuseform*	*Zentrierart/Sonstiges*	*Adapter*
Chips 0504...2510	R, C		Zangenzentrierung	nein
Chips 3225, 4032	R, C		Zangenzentrierung	klein
Melf, SOD, Mini-melf, Mikromelf	R, D		Zangenzentrierung	nein
MKT Chip Kondensatoren A, B	C		Zangenzentrierung	nein
MKT Chip Kondensatoren C, CC, D, E, F	C		Zangenzentrierung	klein
Tantal-Chip-Kondensatoren	C		Zangenzentrierung	nein
SOT23	TR		Zangenzentrierung	nein
SOT89	TR		Zangenzentrierung	nein
SOT143	TR		Zangenzentrierung	nein
SO6 / SO8	IC		Zangenzentrierung	nein
SO10...SO16	IC		Zangenzentrierung	klein
SO16L ...SO24	IC		Zangenzentrierung	groß
SO24L ... SO40	IC		Optische Zentrierstation, Koplanaritätsmessung	groß
SOJ20 ... SOJ40	IC		Optische Zentrierstation	groß
VSO40 /VSO 56	IC		Optische Zentrierstation, Koplanaritätsmessung	groß
PLCC18 ... PLCC28	IC		Zangenzentrierung	groß
PLCC32 ... PLCC124	IC		Optische Zentrierstation	groß
LCCC16 ... LCCC28	IC		Zangenzentrierung	groß
LCCC32 ... LCCC68	IC		Optische Zentrierstation	groß
FP44 ... FP168	IC		Optische Zentrierstation, Koplanaritätsmessung	groß
TAB	IC		Optische Zentrierstation	groß

Tabelle 1: Gehäuseformen von SMD-Bauelementen

Durch Zentrierung der Bauelemente wird eine Erhöhung der Bestückgenauigkeit erreicht. Je nach Ausführung des Bestückkopfes können kleinere Bauelemente mittels der am Bestückkopf befindlichen Zentrierzangen zentriert werden. Größere Bauelemente werden mit zusätzlichen mechanischen oder optischen Zentrierstation zentriert. Bei Fine-Pitch-Bauelementen empfiehlt sich vor dem Bestücken eine Überprüfung auf verbogene Anschlüsse mittels eines Koplanaritätsmeßgerätes.

2.1.3 Anlieferungsformen von SMD-Bauelementen

Die am häufigsten verwendete Anlieferform ist die Gurtverpackung (Blistergurt, Pappgurt). Die Gurte haben den Vorteil, daß die Bauelemente geschützt und verwechslungssicher aufbewahrt und verarbeitet werden können. Darüberhinaus bieten sie eine hohe Kapazität bis zu mehreren tausend Bauteilen. Entsprechend den unterschiedlichen Gehäuseformen werden Gurte in unterschiedlichen Breiten angeboten. Weitere Anlieferformen für SMD-Bauelemente sind Stangenmagazine, Flächenmagazine und Schüttgut.

2.2 Mechanischer Aufbau von SMD-Bestückautomaten

Der mechanische Aufbau eines SMD-Bestückautomaten wird im wesentlichen bestimmt durch seine Kinematik sowie die Art und die Anzahl seiner Bestückköpfe. Durch die Kinematik des Bestückautomaten wird festgelegt, ob Leiterplatte, Bestückkopf oder Abholspuren während des Bestückvorgangs ortsfest oder bewegt sind. Die Art des Bestückkopfes legt das Bestückprinzip fest. Im folgenden werden das Pick-and-Place- und das Collect-and-Place-Prinzip betrachtet.

Beim Pick-and-Place-Prinzip wird jeweils ein Bauelement an der Abholposition aufgenommen und zur Bestückposition gebracht. Beim Collect-and-Place-Prinzip mittels eines Revolver-Bestückkopfes werden zunächst mehrere Bauelemente aufgenommen und anschließend bestückt. Dadurch reduzieren sich die zurückzulegenden Wege und die Bestückleistung wird erhöht.

Durch den Umfang des verarbeitbaren Gehäuseformspektrums lassen sich die Bestückautomaten in Chip-Shooter, Fine-Pitch-Bestücker und General Purpose Placer unterscheiden /41, 317/. Chip-Shooter bieten eine hohe Bestückleistung, sind

aber auf die Bestückung kleiner Gehäuseformen beschränkt. Fine-Pitch-Bestücker ermöglichen das hochgenaue Bestücken von hochpoligen Bauelementen mit kleinem Rastermaß. Mit General Purpose Placern kann das gesamte verfügbare Gehäuseformspektrum bestückt werden. Die Universalität geht aber zu Lasten der Bestückleistung bei der Bestückung kleiner Gehäuseformen.

Kinematik	*Bestückkopf*	*SMD-Bereitstellung*
❍ ortsfeste oder bewegte Leiterplatte ❍ ortsfester oder bewegter Kopf ❍ ortsfeste oder bewegte Abholspuren	❍ single head ❍ Multiple Head ❍ Revolver	❍ Gurt ❍ Schüttgut ❍ Flächenagazin ❍ Stangenmagazin

Bild 5: Kriterien für Aufbau und Funktionsweise von SMD-Bestückautomaten

2.2.1 Grundaufbau

Im folgenden soll exemplarisch der Grundaufbau von SMD-Bestückautomaten mit ortsfester Leiterplatte, ortsfesten Abholspuren und bewegtem Bestückkopf betrachtet werden. Die wesentlichen Komponenten eines solchen Bestückautomaten sind:

- ein massiver Maschinenständer, der alle Funktionsbausteine der Station beinhaltet
- die x/y-Positioniereinheit, die das exakte Verfahren des Bestückkopfes auf eine definierte Position innerhalb der Station ermöglicht.
- das Leiterplattentransportsystem, das zum Transport der Leiterplatten in die Station hinein und aus der Station heraus dient. Es sollte in der Breite verstellbar sein. Bei einer Dreiteilung des Transportsystems in Eingabeband, Mittentransport und Ausgabeband kann es darüberhinaus als Puffer zwischen zwei Stationen dienen.
- Digitale Ein/Ausgabebaugruppen, zur Koordination und Ansteuerung der verschiedenen Baugruppen.
- ein Stationsrechner zur Benutzerkommunikation und zur Ansteuerung der digitalen Ein/Ausgabebaugruppen.

Auf dem Maschinenständer befinden sich neben dem Leiterplattentransportsystem die Förderbereiche, die zur Aufnahme von Förderern und Zusatzeinrichtungen dienen. Durch eine Rasterung werden die Förderbereiche in Stellplätze unterteilt und damit die Abholpositionen der Bauelemente festgelegt. Bei Bestückautomaten mit Förderbereichen auf beiden Seiten des Leiterplattentransportsystems ist eine Seite beweglich ausgeführt, um eine Einstellung des Automaten auf unterschiedlich breite Leiterplatten zu ermöglichen. Während des Bestückvorgangs ist diese Seite aber ebenfalls ortsfest (siehe Bild 6).

Zur Angabe von Positionen innerhalb eines Bestückautomaten werden mehrere Koordindatensysteme definiert. Alle Koordinaten innerhalb des Bestückautomaten, die Positionen von Förderern und Zusatzeinrichtungen betreffen, werden bezüglich des Referenzpunktes ***R*** festgelegt. Dabei handelt es sich um den Punkt, in dem die x- und die y-Achse ihre Nullstellung haben.

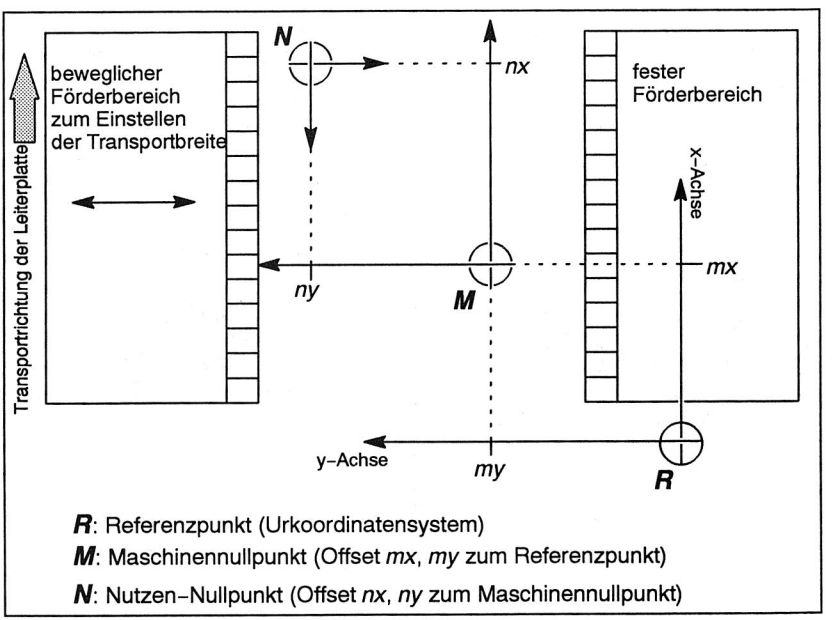


Bild 6: Koordinatensysteme in einem Bestückautomaten

Relativ zum Referenzpunkt wird auch der Maschinennullpunkt ***M*** definiert, der für den Benutzer als Nullpunkt zur Eingabe von Koordinaten dient. Bezüglich zum Maschinennullpunkt wird z. B. die Lage des Nutzennullpunktes ***N*** angegeben, auf den sich die Bestückpositionen innerhalb der Leiterplatte beziehen. Neben dem Versatz in x- und y-Richtung kann sich das Nutzenkoordinatensystem auch durch seine Ausrichtung vom Maschinenkoordinatensystem unterscheiden.

2.2.2 Förderer

Die Aufgabe der Förderer in einem Bestückautomaten ist das Bevorraten und das Bereitstellen der zu bestückenden Bauelemente. Je nach Anlieferform der Bauelemente (siehe Kapitel 2.1.3) werden unterschiedliche Förderer verwendet (Bild 7). Darüberhinaus unterscheiden sich die Förderer durch den Auslösemechanismus für den Weitertransport der Bauelemente (elektrisch, mechanisch).

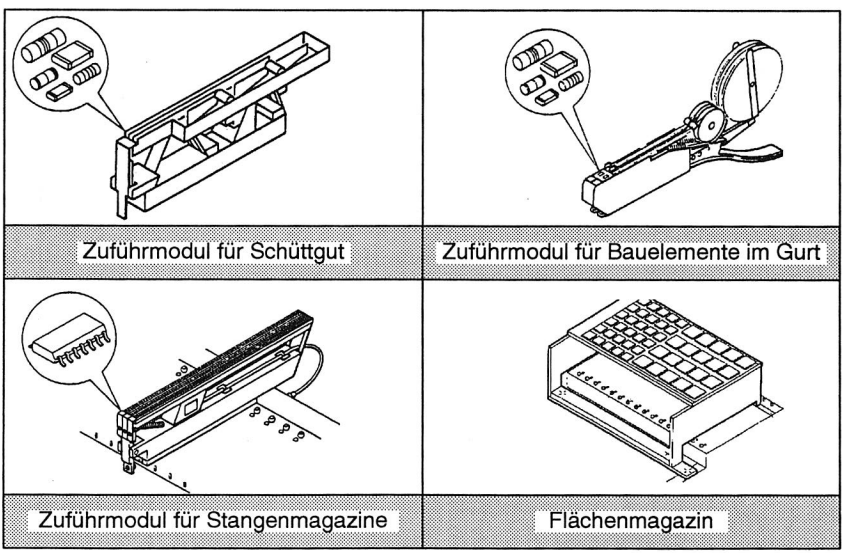
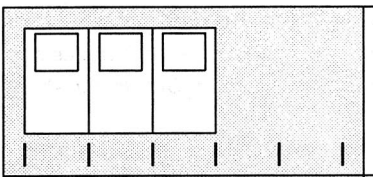


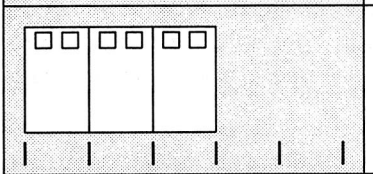
Bild 7: Fördererarten für SMD-Bauelemente (Quelle: Siemens)

Jeder Förderertyp hat eine Breite, durch die der Platzbedarf auf den Förderbereichen festgelegt ist. Bei Förderbereichen mit fester Rastereinteilung wird die Breite in Viel-



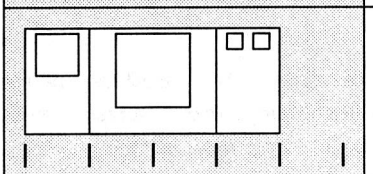
1. Modell:

- Eine Fördererbreite
- Eine Spur je Förderer (z. B. Gurtförderer für 12 und 16 mm-Gurte)



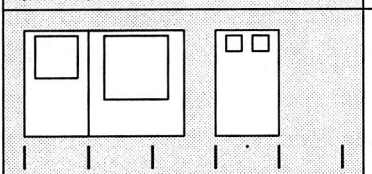
2. Modell

- Eine Fördererbreite
- Mehrere Spuren je Förderer (z. B. Gurtförderer für 8mm-Gurte)



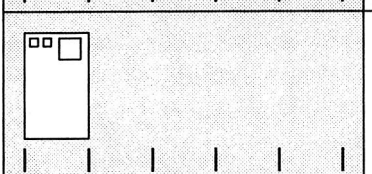
3. Modell

- Verschiedene Fördererbreiten, aber jeweils ganzzahlige Vielfache des Förderbereichsrasters.
- Eine oder mehrere Spuren je Förderer



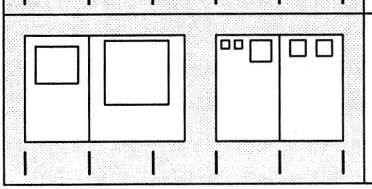
4. Modell

- Verschiedene Fördererbreiten, eventuell nicht ganzzahlige Vielfache des Förderbereichsrasters
- Eine oder mehrere Spuren je Förderer



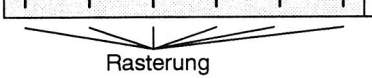
5. Modell

- Anzahl der Spuren auf dem Förderer ist abhängig von der Größe der gerüsteten Bauelemente



6. Modell

- Beliebige Mischformen der obigen Modelle



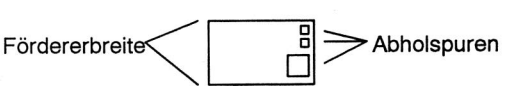


Bild 8: Klassifikation von Förderertypen nach Breite und Anzahl Abholspuren

fachen der Modulplätze angegeben. Die Förderer stellen eine oder mehrere Abholspuren zur Verfügung. Die Abholpositionen der Baulemente ergeben sich aus dem Rüstort eines Förderers (Förderbereich und Stellplatz) und den Offsets der Abholspuren innerhalb des Förderers.

Bild 8 zeigt eine Klassifikation der Förderertypen nach ihrer Breite und der Anzahl der Abholspuren. Je komplexer das Modell ist, desto größer ist der Aufwand für den Bediener, eine (optimale) Rüstung für den Bestückautomaten bzw. die Bestücklinie zu finden.

2.2.3 Zusatzeinrichtungen

Durch Zusatzeinrichtungen werden Maßnahmen realisiert, die die Bestückgenauigkeit und die Bestückqualität erhöhen sollen. Die Einführung dieser Zusatzeinrichtungen hat aber andererseits Auswirkungen auf die Komplexität der eingesetzten Steuerungssoftware und den Umfang der notwendigen Bedientätigkeiten und erhöht somit den Schulungsaufwand für das Personal. Einige dieser Zusatzeinrichtungen werden im folgenden vorgestellt.

Mechanische und optische Zentrierstationen
Zentrierstationen sind erforderlich, um Bauelemente, die auf Grund ihrer Grundfläche nicht mehr von den Zentrierzangen des Bestückkopfes erfaßt werden können, zu zentrieren. Bei einer mechanischen Zentrierstation werden die Bauelemente vor dem Bestücken durch Zangenbacken in x- und y- Richtung, sowie bezüglich ihres Drehwinkels ausgerichtet und in dieser normierten Lage vom Bestückkopf wieder aufgenommen. Optische Zentrierstationen müssen dann eingesetzt werden, wenn

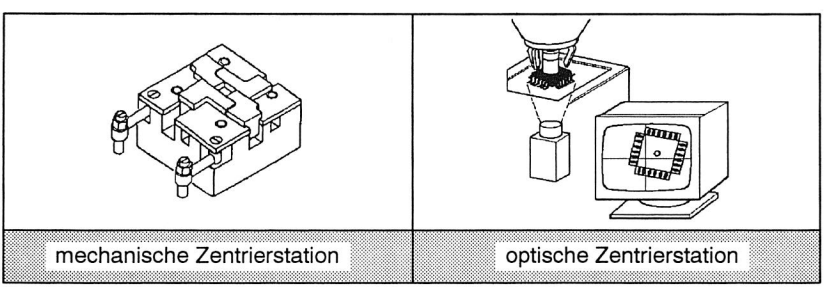


Bild 9: mechanische und optische Zentrierstation

die seitlichen Kräfte, die in einer mechanischen Zentrierstation auf die Bauelemente ausgeübt werden, zum Verbiegen der Beinchen führen. Bei einer optischen Zentrierstation wird das Bauelement am Bestückkopf mit einer Kamera aufgenommen und somit die Lage des Bauelements relativ zum Pipettenmittelpunkt ermittelt. Dazu wird das von der Kamera aufgenomme Bild mit einem hinterlegten Modell verglichen. Die ermittelte Lage und der Drehwinkel des Bauelements werden zur Korrektur der Bestückposition verwendet. Neben der Zentrierung der Bauelemente können auch seitliche Verbiegungen der Beinchen ermittelt werden.

Koplanaritätsmeßgerät

Ein Koplanaritätsmeßgerät dient zum Ermitteln von in z-Richtung verbogenen Anschlußbeinen. Dazu werden mittels Lasertriangulation die Bauelemente vermessen und die Aufsetzebene berechnet. Die Aufsetzebene ist die Ebene, die durch die drei Anschlußbeine gebildet wird, die als erste auf der Leiterplatte aufsetzen. Bezogen auf diese Referenzebene werden die vertikalen Verbiegungen der Beinchen bestimmt. Das Verfahren eliminiert auch den Einfluß einer eventuellen Schräglage des Bauelements an der Pipette. An Hand eines Grenzwertes, der auf der Lotpastendicke basiert, wird ermittelt, ob alle Anschlußbeinchen in die Lotpaste eingedrückt werden und schließlich eine gut/schlecht-Entscheidung getroffen.

Optische Leiterplattenlageerkennung

Die optische Leiterplattenlageerkennung ersetzt die mechanische Leiterplattenlagekorrektur durch Zentrierstifte. Vor dem Bestücken wird die Lage der Leiterplatte an Hand der Position von sogenannten Pass-Marken erkannt und korrigiert. Dazu werden die Pass-Marken mittels einer am Bestückkopf angebrachten Kamera aufgenommen und mit einem zuvor geteachten Muster verglichen. Auf eine Leiterplatte werden in der Regel drei, mindestens aber zwei Passmarken aufgetragen. Aus der Lage der Marken können Positionsabweichungen, die beim Transport der Leiterplatte entstehen, sowie Stauchungen und Verzüge des Layouts erkannt und korrigiert werden.

Meßgeräte für elektronische Bauelementekennwerte

Mit Hilfe eines zusätzlichen Zangenpaares am Bestückkopf können die elektrischen Werte von zweipoligen Bauelementen während des Transports von der Abholposition zur Bestückposition überprüft werden. Durch die Messung wird die Bestückleistung der Automaten nicht beeinflußt. Es können die Werte von Kapazitäten und ohmschen Widerständen, sowie die Spannung und die Polarität von Dioden und Tantalkondensatoren ermittelt und mit einem vorgegebenen Wert verglichen werden.

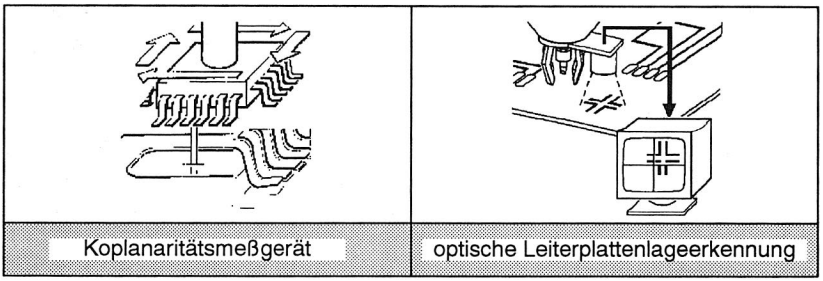


Bild 10: Koplanaritätsmeßgerät und optische Leiterplattenlageerkennung

Der Bestückkopf wirft als fehlerhaft erkannte Bauelemente ab und wiederholt den Bestückvorgang.

Adapterwechselstation

Mit Hilfe von Adaptern können auch solche Bauelemente bestückt werden, die wegen ihrer Abmaße oder wegen ihres Gewichts nicht mehr mit der Saugpipette des Bestückkopfs aufgenommen werden können. Die Adapter werden in einer Halterung, die mehrere Adapter aufnehmen kann, bereitgestellt. Die Abholen und das Ablegen der Adapter wird vom Bestückkopf automatisch ausgeführt.

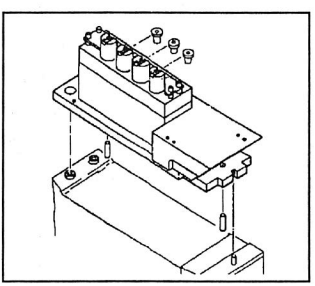
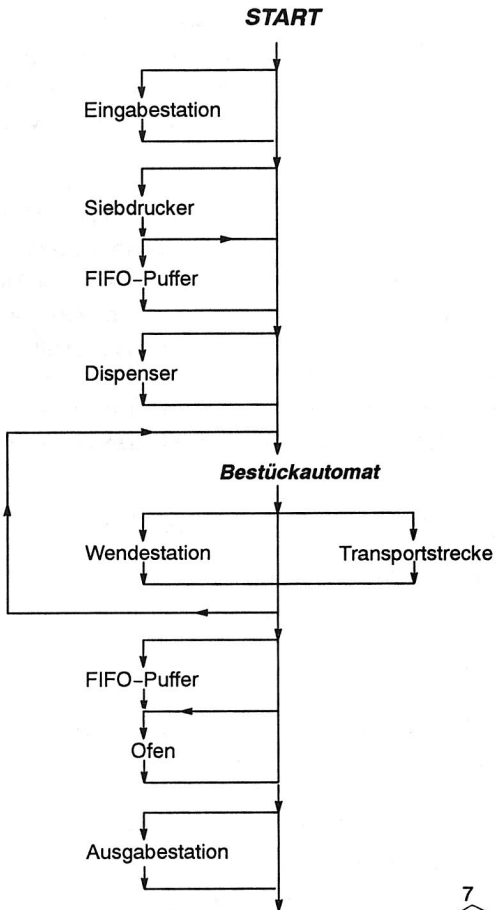


Bild 11: Adapterwechselstation

2.3 Aufbau von SMD-Bestücklinien

SMD-Bestücklinien ermöglichen die automatisierte Komplettbearbeitung von Flachbaugruppen inklusive Kleber- oder Lotpastenauftrag, Bestücken und Löten /21/. Die



1. Eingabeeinheit
2. Siebdrucker
3. Dispenser
4. Bestückautomaten
5. Transportstrecke
6. Ofen
7. Ausgabeeinheit

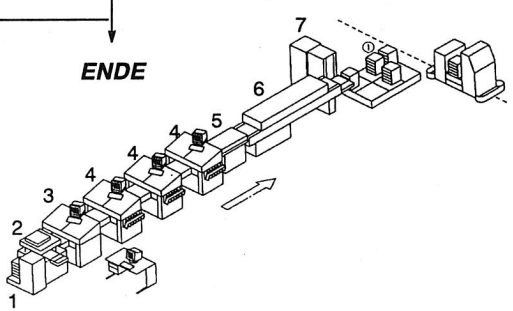


Bild 12: Aufbauvarianten von SMD-Bestücklinien

Linien bestehen aus einem oder mehreren Bestückautomaten und werden je nach Verfahrenskette um Siebdrucker, Dispenser für Kleber oder Lotpaste, Transportstrecken und Wendestationen ergänzt. Zur Erhöhung der Verfügbarkeit empfiehlt sich das automatische Be- und Entladen der Leiterplatten mit Hilfe von Eingabe- und Ausgabestationen.

Durch die Integration von Puffern mit FIFO-Speichercharakteristik werden Geräte mit unterschiedlichen Arbeitsinhalten und unterschiedlichen Rüstzeiten voneinander entkoppelt. Insbesondere Verkettungen von Siebdruckanlagen mit Bestückautomaten und von Bestückautomaten mit Reflowlötöfen sind als Problemstellen bekannt und werden zur Erhöhung der Linienverfügbarkeit entkoppelt /78/.

In Bild 12 werden Aufbauvarianten von SMD-Bestücklinien an Hand eines Graphen dargestellt. Jeder Pfad durch den Graphen beschreibt den möglichen Aufbau einer SMD-Bestücklinie.

2.4 Steuerung von Bestücklinien

2.4.1 Steuerungsstruktur

Die Produktivität von Bestückautomaten bzw. Bestücklinien hängt wesentlich von der Leistungsfähigkeit und der Flexibilität der eingesetzten Steuerungen und Softwaresysteme ab. Zur Steuerung von flexiblen Fertigungssystemen haben sich hierarchische Steuerungslösungen etabliert /12/. Hierbei werden vier Planungsebenen unterschieden (Maschinensteuerungen, Zellenrechner, Leitrechner, PPS), die sich vor allem durch den jeweils betrachteten Planungshorizont unterscheiden. Diese Hierarchie läßt sich auch bei der Steuerung von SMD-Bestücklinien wiederfinden (Bild 13).

Der Linienrechner übernimmt hierbei die Aufgaben eines Zellenrechners. Er verwaltet alle zur Steuerung der Bestücklinie benötigten technischen und dispositiven Daten. Dialogprogramme ermöglichen es dem Bediener, Bestückprogramme zu erstellen und die Bestücklinie zu steuern. Über ein Bussystem oder Punkt-zu-Punkt-Verbindungen kommuniziert der Linienrechner mit den Bestückautomaten und den Zusatzgeräten innerhalb der Bestücklinie und sorgt so für die Ausführung vorgegebener Aufträge.

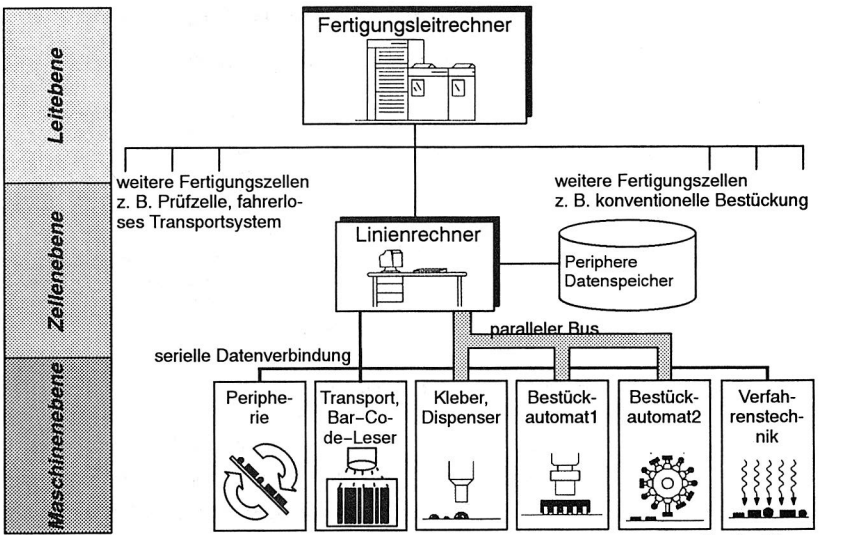


Bild 13: Hierarchische Steuerungslösung in der Elektronikproduktion

2.4.2 Technologische Daten

Zur Steuerung einer Bestücklinie werden drei Gruppen von technologischen Daten benötigt:

Maschinendaten beschreiben den Aufbau der Bestückautomaten. Dazu gehören Parameter zur Achsansteuerung, Offsets und Positionsangaben sowie Daten für Zusatzgeräte. Die Maschinendaten werden bei der Inbetriebnahme einer Bestücklinie erstellt. Änderungen ergeben sich nur durch Verschleiß oder durch Umbaumaßnahmen auf Grund von Reparaturen oder Erweiterungen.

Die **Rüstdaten** bestehen aus den Bauelementeinformationen (elektrische Werte, Gehäuseformdaten) und den Spurinformationen. Durch die Spurinformationen wird der Rüstort der Bauelemente beschrieben. Der Rüstort eines Bauelements ist festgelegt durch:

- Angabe der Bestückstation, auf dem das Bauelement gerüstet ist
- Angabe des Förderbereichs in der Bestückstation

- Angabe des Stellplatzes des Förderers innerhalb des Förderbereichs
- Angabe der Abholspur des Bauelements auf dem Förderer

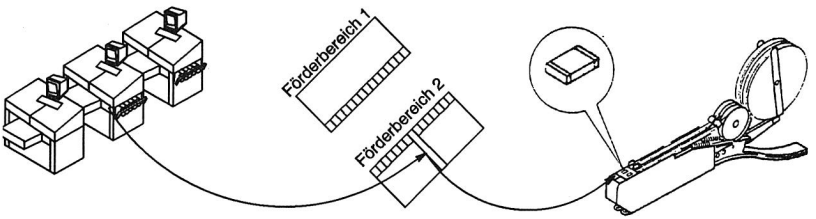


Bild 14: Angabe des Rüstortes eines Bauelements

Die **Nutzendaten** beschreiben die Struktur der zu bestückenden Leiterplatte sowie die Bestückpositionen der Bauelemente. Die Struktur einer Leiterplatte ist festgelegt durch ihre Abmaße (Länge, Breite, Dicke) und ihre Aufteilung in Einzelschaltungen.

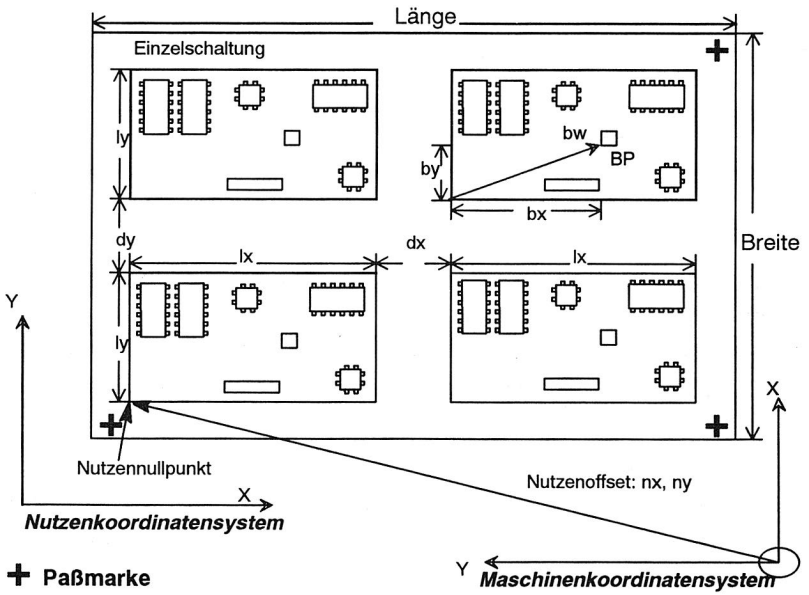


Bild 15: Nutzendaten und Bestückpositionsdaten

Es sind sowohl regelmäßige Anordnungen von Einzelschaltungen möglich, bei der alle Einzelschaltungen die gleiche Drehrichtung haben und äquidistant angeordnet sind, als auch Musteranordnungen, bei der die Lage jeder Einzelschaltung über x- und y-Offset und den Drehwinkel angegeben wird. Die Angabe der Einzelschaltungspositionen auf der Leiterplatte erfolgt relativ zum Leiterplattennullpunkt (Bild 6). Die Bestückpositionen der Bauelemente werden durch x- und y-Koordinaten sowie den Drehwinkel des Bauelementes beschrieben. Die Angabe erfolgt relativ zu den Einzelschaltungen. Aus dem jeweiligen Einzelschaltungsoffset und den Bestückpositionen lassen sich die Koordinaten des Bauelements relativ zum Leiterplattennullpunkt berechnen. Wird zur Erhöhung der Bestückgenauigkeit eine optische Leiterplattenlageerkennung verwendet, muß die Lage und die Art der Paßmarken beschrieben werden.

2.4.3 Aufgaben des Linienrechners

Der Linienrechner ist innerhalb einer Bestückline für alle stationsübergreifenden Steuerungsaufgaben zuständig:

- Verwaltung von Bestückprogrammen und Auftragsdaten
- Verwaltung von Maschinen- und Rüstdaten
- Generierung von automatenspezifischen Bestückprogrammen
- Generierung von automatenspezifischen Rüstungen und Rüstanweisungen
- Benutzerkommunikation
- Diagnose, Statistik
- Kommunikation mit Bestückautomaten und Zusatzgeräten
- Kommunikation mit Leitrechnern und CAD-Systemen
- Störfallbehandlung und Ausgabe von Fehlermeldungen

Die Erstellung von Bestückprogrammen für eine Bestücklinie erfolgt zentral am Linienrechner durch Eingabe der Nutzendaten. Über eine CAD-Kopplung können die Bestückpositionen direkt aus einem CAD-System übernommen werden. Die zu einem Bestückprogramm gehörigen Rüstdaten werden ebenfalls zentral am Linienrechner durch den Bediener vorgegeben oder aber automatisch durch die Linienrechnersoftware generiert. Bei einer automatischen Erstellung der Rüstdaten

können die Gesichtspunkte der Rüst- und Umrüstoptimierung berücksichtigt werden, die zu einer Reduzierung des Umrüstaufwands und einer gleichmäßigen Austaktung der Bestückautomaten führen /21, 22, 23, 90/.

2.4.4 Durchführbarkeitsanalyse von Aufträgen

An Hand der Nutzendaten, der Rüstdaten und der Maschinendaten kann die Durchführbarkeit von Aufträgen auf einer Bestücklinie überprüft und die Bestückautomaten-spezifischen Bestückprogramme generiert werden (Bild 16). Dazu sind der Reihe nach folgende Fragen zu beantworten:

1. Ist das Leiterplattenformat verarbeitbar?
   Dazu muß das Leiterplattentransportsystem auf die Breite der Leiterplatte einstellbar sein und der Bestückbereich der Bestückautomaten muß die gesamte Leiterplatte abdecken.
2. Sind Zusatzgeräte vorhanden, die für die Leiterplatte benötigt werden?
   Dabei kann es sich z. B. um Leiterplatten-Vision-Systeme oder Wende- und Drehstationen handeln. Sind die Zustzgeräte nicht vorhanden, muß auf eine alternative Linie ausgewichen werden oder die Linie umkonfiguriert werden.
3. Sind alle verwendeten Bauelementetypen in der Bauelemente-Bibliothek?
   Wenn nicht, ist die Bauelemente-Bibliothek zu ergänzen.
4. Sind alle verwendeten Bauelementetypen von der Größe her bestückbar?
5. Sind alle benötigten Bauelementetypen gerüstet?
6. Sind Zusatzgeräte vorhanden, die für die Bauelementetypen benötigt werden?
   Dabei kann es sich z. B. um Zentrierstationen, Koplanaritätsmeßgeräte, Adapter, Pipetten, Vision-Systeme und Meßgeräte für elektrische Werte handeln. Fehlen diese Geräte, ist ein Umkonfigurieren der Linie oder das Ausweichen auf eine andere Linie erforderlich.

Bei der automatischen Erstellung einer Rüstung durch den Linienrechner sind folgende zusätzlichen Fragen zu beantworten:

1. Ist für jeden Bauelementetyp die Anlieferform bekannt?
   Aus der Anlieferform ergeben sich die möglichen Förderer, die für den Bauelemente-Typ verwendet werden können, und damit der notwendige Platzbedarf. Ist die Anlieferform nicht für jeden Bauelementetyp bekannt, müssen die Daten vom Bediener ergänzt werden.

2. Gibt es für jede Anlieferform einen Förderertyp?

3. Können alle benötigten Bauelementetypen gerüstet werden?
   Auf Grund von Festrüstanteilen, Restriktionen oder zu geringem Rüstplatz kann es sein, daß nicht alle benötigten Bauelementetypen gerüstet werden können. Abhilfe ist hier das Ändern der Festrüstanteile, das Aufheben von Restriktionen oder der zweimalige Durchlauf der Leiterplatten durch die Bestücklinie.

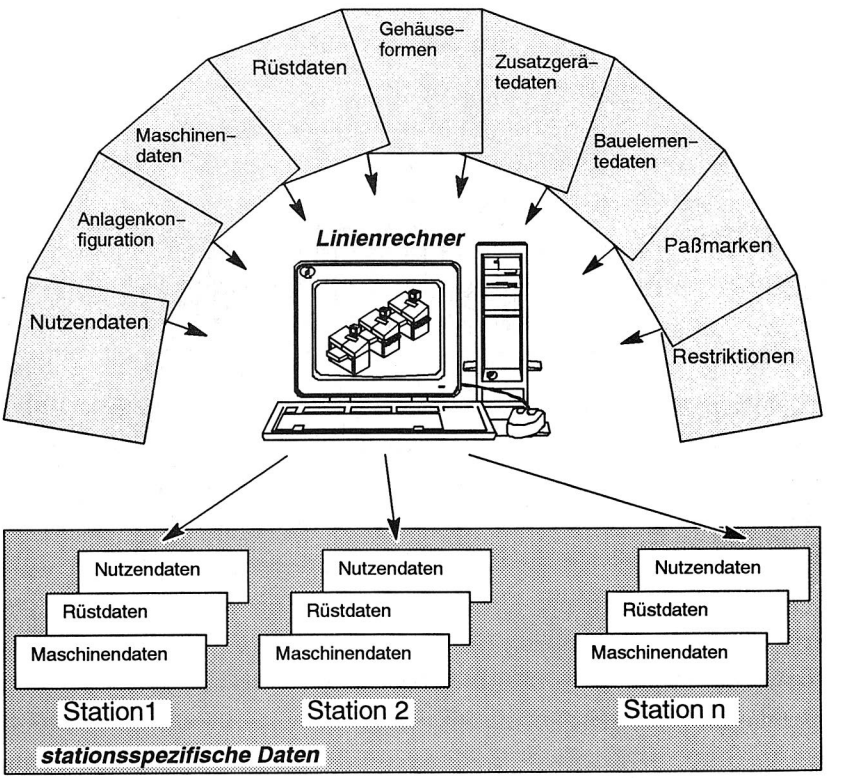


Bild 16: Datenfluß bei der Generierung der Bestückprogramme

Generierung der Bestückprogramme

Nach der erfolgreichen Durchführbarkeitsanalyse eines Auftrags können die Bestückautomaten-spezifischen Bestückprogramme generiert werden. Jedem Bestückautomaten wird dabei auf Grund seiner Rüstung eine Teilmenge der zu bestükkenden Bauelemente zugeordnet (Bild 17). Ist jeder Bauelementetyp nur einmal in der Bestücklinie gerüstet ist, ist diese Zuordnung eindeutig. Sind Bauelementetypen mehrfach gerüstet, erfolgt die Zuordnung nach Kriterien der gleichmäßigen Stationsauslastung /32/.

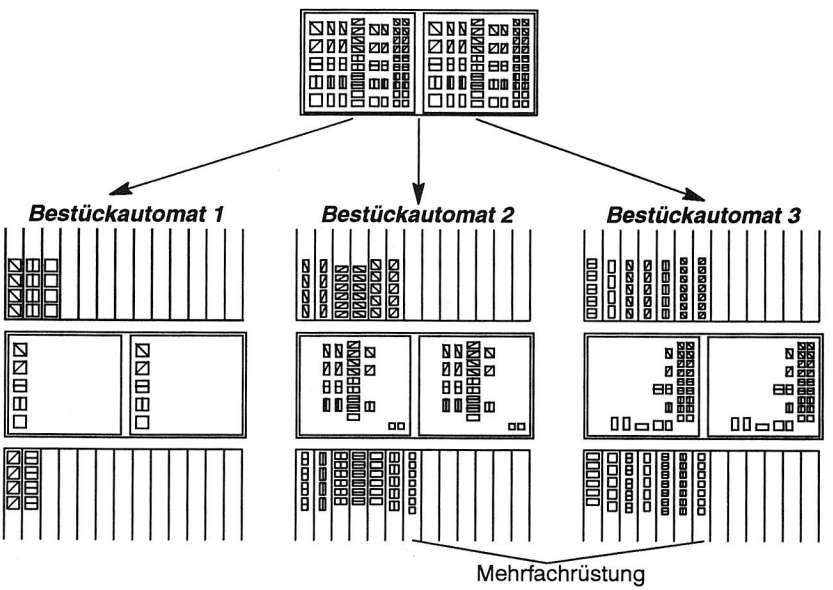


Bild 17: Zuordnung von Bestückpositionen zu Bestückautomaten

2.5 Steuerung der Bestückautomaten

2.5.1 Aufgaben der Steuerung

Die Steuerung eines Bestückautomaten hat für die zuverlässige und schnelle Bestückung der Leiterplatten zu sorgen. Dazu gehören folgende Teilaufgaben:

- Organisation des stationsinternen Materialflusses
  Der stationsinterne Materialfluß ist für den Transport der Leiterplatten verantwortlich. Die Leiterplatten werden über das Eingabetransportband zu der Position gebracht, an der sie bestückt werden und über das Ausgabetransportband zur Folgestation gebracht.
- Steuerung der Förderer
  Nach der Entnahme eines Bauteils müssen die Förderer zur Bereitstellung des nächsten Bauteils aktiviert werden. Dabei sind die unterschiedlichen Auslösemechanismen bei mechanischen und elektrischen Förderern zu beachten.
- Kontrollieren der zum Bestücken notwendigen Mechanik
- Steuerung der Zusatzeinrichtungen
- Wegeoptimierter Bestückablauf
- Benutzerkommunikation

Die vom Linienrechner in Form der Rüst- und Nutzendaten vorgegebenen Bestückprogramme enthalten noch keine Bewegungsanweisungen für die Mechanik des Bestückautomaten. Hauptaufgabe der Steuerung des Bestückautomaten ist es deshalb, an Hand der vorgegebenen Bestückpositionen und Bauelementeabholpositionen Bewegungsanweisungen für die auf Grund der kinematischen Eigenschaften bewegten Komponenten (Bestückkopf, Abholspuren, Leiterplatte) zu generieren und durchzuführen. Durch eine Bestückreihenfolgeoptimierung können die zurückzulegenden Wege minimiert und somit die Bestückzeiten reduziert werden. Die dabei anzuwendenden Optimierungsalgorithmen sind von der Kinematik und der Art und der Anzahl der Bestückköpfe abhängig und erreichen eine hohe Komplexität. Bereits bei einem einfachen Pick-and-Place-Automaten mit ortsfester Leiterplatte, ortsfesten Abholspuren und bewegtem Bestückkopf handelt es sich bei dem Optimierungsproblem um das Problem des Handlungsreisenden, dessen Komplexität fakultativ mit der Anzahl der Orte (hier der Bestückpositionen) anwächst. Zur Optimierung können deshalb nur suboptimale heuristische Verfahren mit kurzen Laufzeiten eingesetzt werden.

2.5.2 Interne Steuerungsstruktur

Auf Grund der komplexen Steuerungsaufgaben werden zur Steuerung von Bestückautomaten meist sogenannte Stationsrechner eingesetzt, deren Leistungsfähigkeit

im Bereich von Personal Computern liegt. Je nach Hersteller kommen dabei Rechner unterschiedlicher Ausprägung zum Einsatz. Der Vorteil dieses Konzepts liegt darin, daß die Bestückautomaten ohne steuernde Eingriffe von außen arbeiten können, solange keine Veränderungen im Fertigungsablauf notwendig sind (Loswechsel, Umrüstung).

Zur Umsetzung der zur Bestückung notwendigen Bewegungsanweisungen werden Achsansteuerungen verwendet, die die Positionierung der Achsen eines Bestückautomaten kontrollieren, korrigieren und verändern. Den Achsansteuerungen werden vom Stationsrechner Aufträge zugewiesen, die die Zielposition, die Beschleunigung und die Maximalgeschwindigkeit beinhalten. Die Achsansteuerung führt den Auftrag aus und sendet dem Stationsrechner nach erreichen der Zielposition eine Endemeldung. Dadurch wird der nächste Schritt in der Steuersequenz initialisiert.

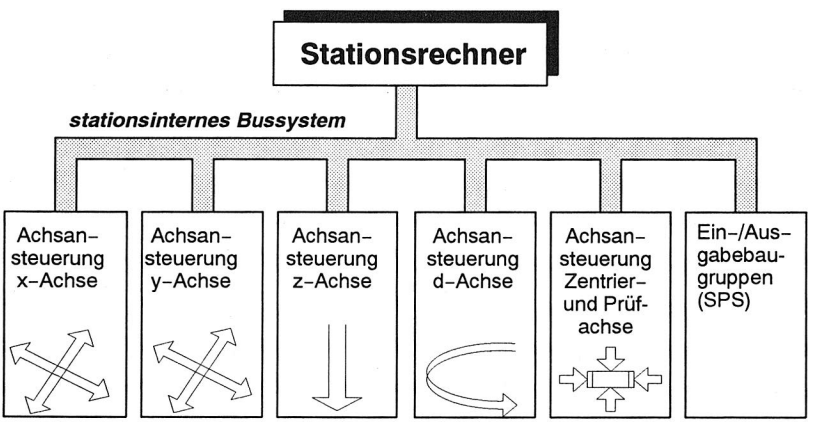


Bild 18: Steuerungsstruktur eines Bestückautomaten

2.6 Datenaustausch zwischen Linienrechner und Bestückautomaten

Der notwendige Datenaustausch zwischen Linienrechner und Bestückautomaten erfolgt über Bussysteme oder Punkt-zu-Punkt-Verbindungen (z. B. serielle Leitung). Die technologischen Daten, die die Bestückautomaten benötigen, sind zentral auf

dem Linienrechner gespeichert und müssen über den eingesetzten Kommunikationsmechanismus an die Bestückautomaten übertragen werden. Dazu werden die Daten zu sinnvollen Einheiten (Maschinendaten, Rüstdaten, Nutzendaten, usw.) zusammengefaßt. Derartige Einheiten werden im folgenden ***Telegramme*** genannt. Die Telegramme werden nach ihrem Inhalt klassifiziert (***Telegrammtyp***). Neben der Übertragung von Daten dienen die Telegramme auch zum Auslösen von Aktionen wie dem Starten oder dem Stoppen des Bestückens. Durch ein ***Protokoll*** wird eine sinnvolle Übertragungsreihenfolge für die Telegramme festgelegt. Die Telegramme bestehen aus einem Telegrammkopf, an Hand dessen der Telegrammtyp erkannt wird, und den Nutzdaten. Von den Bestückautomaten werden mit Hilfe von Telegrammen Statusmeldungen, BDE-Daten und Datenanforderungen an den Linienrechner übertragen.

Rückmeldungen der Stationen

Die Bestückstationen liefern Kennzahlen an den Linienrechner zurück, die den Fertigungsverlauf charakterisieren. Die Auswertung dieser Kennzahlen zeigt Schwachstellen auf, die zu einem suboptimalen Betrieb der Anlagen führen. Typische Kennzahlen sind die Anzahl produzierter Leiterplatten, die Anzahl verbrauchter Bauelemente, die Anzahl fehlerhafter Zugriffe auf Abholspuren, Störungsursachen und -häufigkeiten, sowie Zeitinformationen.

Zeiten

Die Betriebszeit eines Bestückautomaten setzt sich aus Hauptzeiten, Wartezeiten, Leiterplattenwechselzeiten, Nachrüstzeiten und Störungszeiten zusammen. Eine Auswertung der Zeiten liefert Hinweise, die zu einer Verkürzung der Bestückzeiten führen kann. So können zu lange Wartezeiten auf die Leiterplatten, die auf eine ungleichmäßige Verteilung des Bestückaufwands auf die einzelnen Bestückstationen zurückzuführen sind, durch Umrüstmaßnahmen behoben werden.

Anzahl verbrauchter Bauelemente

Die Gesamtzahl der verbrauchten Bauelemente setzt sich aus der Zahl der bestückten Bauelemente und der Zahl der als schlecht erkannten Bauelemente zusammen. Bauelemente werden als schlecht erkannt und abgeworfen, wenn ihr elektrischer Wert außerhalb des geforderten Toleranzbereichs liegt oder wenn auf Grund der optischen Inspektion Anschlußbeine als verbogen klassifiziert werden. Eine zu hohe Zahl abgeworfener Bauelemente kann z. B. auf eine schlechte Charge hinweisen oder auf einen zu eng eingestellten Toleranzbereich bei der Überprüfung der elektrischen Werte.

3. Entwicklungstendenzen bei SMD-Bestücklinien

3.1 Anforderungen an SMD-Bestücklinien

Die kontinuierlichen Entwicklungen im Bereich der SMD-Bestücktechnik resultieren, wie in der Automatisierungstechnik allgemein, aus den steigenden Anforderungen durch die Anwender der Systeme und aus der fortschreitenden technologischen Entwicklung in den Bereichen Mikroelektronik, Informationstechnik, Kommunikationstechnik und Software /115/. Die Anwender verlangen von den Bestücklinien hohe Bestückleistung und Bestückqualität bei gleichzeitig hoher Verfügbarkeit und Flexibilität der Anlagen. Nur so lassen sich bei steigender Variantenvielfalt und sinkenden Produktlebensdauern auch kleine Lose wirtschaftlich und damit konkurrenzfähig fertigen /24, 64/. Die technologische Entwicklung im SMD-Bereich wird beeinflußt durch neue Gehäuseformen, neue Verpackungsformen und neue Verbindungstechniken. Die fortschreitende Miniaturisierung führt zu immer kleineren Bauelementen, wie z. B. 0402-Bauelementen mit einer Größe von 1,00mm x 0,5mm, auf der einen Seite und zu hochpoligen IC's mit Abmessungen bis 55mm x 55mm und mehreren hundert Anschlußbeinen auf der anderen Seite /13, 64/. Stand der Technik bei diesen Bauelementen ist zur Zeit ein Rastermaß von 0,5mm mit der Tendenz zu 0,3mm. Die neuen Gehäuseformen bedingen die Entwicklung von neuen Anlieferformen und damit auch neuen Förderertypen. So werden hochpolige Bauelemente meist in Flächenmagazinen angeliefert, um ein Verbiegen der Beine zu verhindern. Bei den Verbindungstechniken gewinnen alternative Verfahren wie COB (Chip on Board) und TAB (Tape Automated Bonding) im Hinblick auf hohe Integrationsdichten zusehens an Bedeutung und verlangen erweiterte Maschinenfunktionen /26, 113/.

Um den Anforderungen der Anwender gerecht werden zu können, sind herstellerseitig Erweiterungen der Steuerungssoftware hinsichtlich organisatorischer und technologischer Funktionalitäten durchzuführen. Desgleichen erfordern die technologischen Anforderungen eine Hardware- und Software-technische Integration notwendiger Zusatzeinrichtungen in die Bestückautomaten.

3.1.1 Erweiterungen um Zusatzgeräte

Die zur Erhöhung der Bestückgenauigkeit und -zuverlässigkeit erforderlichen Zusatzgeräte, wie sie in Kapitel 2.2.3 aufgeführt sind (optische Zentrierstationen, Ko-

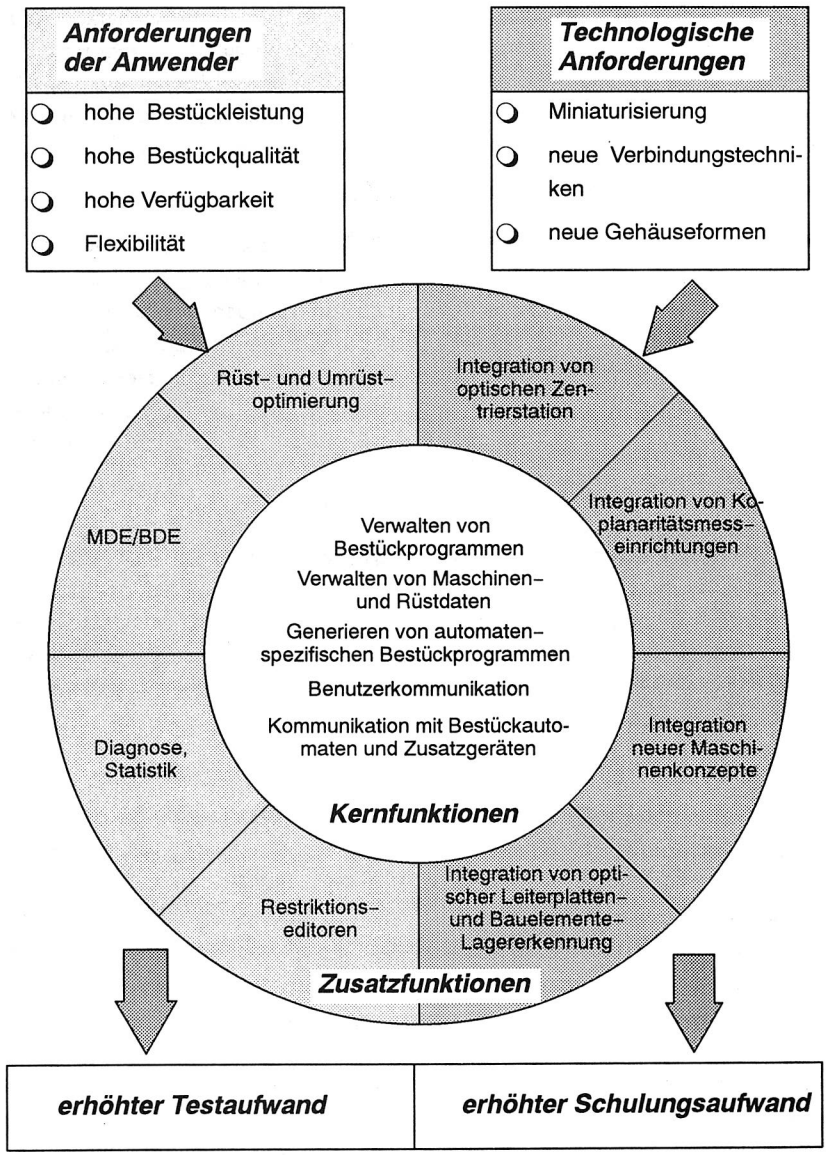


Bild 19: Erweiterung des Funktionsumfangs der Linienrechnersoftware durch Marktanforderungen und Technologische Anforderungen

planaritätsmeßgeräte, optische Bauelemente- und Leiterplattenlageerkennung, usw.) , erhöhen sowohl den Umfang als auch die Komplexität der Steuerungssoftware von Linien- und Stationsrechnern. Der Umfang wird erhöht durch die Bereitstellung und Verwaltung zusätzlicher Daten, wie z. B. Passmarkeninformationen für die optische Bauelementelageerkennung oder Gehäuseforminformationen für die optische Bauelementezentrierung.

Die Komplexität steigt dadurch, daß sich durch die Zusatzgeräte zusätzliche Abhängigkeiten zwischen den Bauelementetypen und den Rüstorten ergeben, die z. B. bei der Ermittlung einer Bauelemente-Rüstung überprüft werden müssen. So können Fine-Pitch-Bauelemente nur auf einer Bestückstation gerüstet und bestückt werden, auf der auch ein Koplanaritätsmeßgerät und eine optische Zentrierstation vorhanden sind. Bei der Integration neuer Zusatzgeräte muß also sowohl die bestehende Software um zusätzliche Module erweitert werden, als auch bestehende Module grundlegend verändert werden.

3.1.2 Erweiterungen um Zusatzfunktionen

Die Zusatzfunktionen der Linienrechnersoftware sorgen für eine höhere Verfügbarkeit der Anlagen und führen somit zu einer verbesserten Wirtschaftlichkeit. Beispiele für derartige Zusatzfunktionen sind die Rüst- und Umrüstoptimierung sowie MDE/BDE- und Diagnose-Funktionen.

Insbesondere die Rüst- und Umrüstoptimierung stellt in Anbetracht von sinkenden Losgrößen und steigender Variantenvielfalt eine wichtige Maßnahme dar, mit der sowohl die Auslastungen der Anlagen erhöht, als auch die Durchlaufzeiten der Aufträge reduziert werden können /113, 32, 21, 81/. Um trotz der kombinatorischen Vielzahl an Möglichkeiten eine Optimierung vornehmen zu können, werden in einer mehrschichtigen Vorgehensweise Bestücklinien-übergreifende, Bestücklinien-interne und Bestückautomaten-spezifische Methoden eingesetzt, durch die der Aufwand zur Umrüstung der Bestücklinien für einen gegebenen Auftragsmix bei gleichzeitig optimierter Anordnung der Bauelemente reduziert wird. Bei der linienübergreifenden Optimierung werden die Aufträge nach Ähnlichkeitskriterien (z. B. Anzahl übereinstimmender Bauelementetypen) zusammengefaßt und den Bestücklinien Gruppen mit ähnlichen Aufträgen zugewiesen. Innerhalb einer Bestücklinie kann die Zahl der notwendigen Umrüstungen durch eine Variation der Auftragsreihenfolge reduziert werden. Bei der Ermittlung der Rüstung ist auf eine gleichmäßige Aufteilung des Be-

stückvolumens auf die Bestückautomaten zu achten (Balancing), da so die Durchlaufzeiten der Leiterplatten reduziert werden. Eine Verkürzung der Bestückzeiten innerhalb eines Bestückautomaten wird schließlich durch eine optimierte Anordnung der Bauelemente und eine Optimierung der Bestückreihenfolge erreicht /21, 22, 23/.

MDE/BDE und Diagnose-Funktionen machen auf Probleme im organisatorischen Umfeld und auf technische Störungen aufmerksam und helfen, den Nutzungsgrad und damit die Produktivität einer SMD-Bestücklinie zu steigern. Dazu wird der Anlagenzustand mit Fehlerraten, Trendmeldungen und Nutzungsgrad erfasst und grafisch dargestellt /64, 92/.

3.1.3 Integration neuer Bestückautomatentypen

Neue Bestückautomatentypen können sich hinsichtlich der in Bild 5 genannten Kriterien (kinematische Eigenschaften, Bestückkopf, SMD-Bereitstellung, usw.) von ihren Vorgängern unterscheiden. Durch diese Unterschiede wird eine Erweiterung der Linienrechnersoftware notwendig, um die spezifischen Daten für die neuen Bestückautomatentypen zu verwalten (neue Eingabefenster, neue Daten- und Dateistrukturen). Darüberhinaus erfordert die Integration der neuen Automatentypen eine Anpassung der bereits existierenden Algorithmen zur Steuerung und Optimierung der Bestückvorgänge. So sind z. B. beim Übergang von einem einfachen Pick-and-Place-Bestückkopf zu einem Revolverkopf zusätzlich die Daten für die Pipetten und die Rotationsgeschwindigkeit des Bestückkopfes zu verwalten. Die Optimierung der Bestückreihenfolge unterscheidet sich bei diesen beiden Bestückprinzipien grundsätzlich voneinander. Während sich das Optimierungsproblem bei einem einfachen Pick-and-Place-System auf das Problem des Handlungsreisenden zurückführen läßt, der n Orte (die Bestückpositionen) mit dem insgesamt kürzesten Weg je einmal anfahren muß, ist bei einem System mit einem Revolverkopf hauptsächlich darauf zu achten, die Anzahl der Bestückzyklen zu minimieren, d. h. bei jedem Bestückzyklus sollte möglichst mit allen Pipetten ein Bauelement aufgenommen und bestückt werden. Das kann durch Optimierung der Bauelementerüstung und der Pipettenkonfiguration am Bestückkopf erreicht werden.

3.1.4 Portierungen auf neue Hardware und/oder Software-Plattformen

Die Erweiterungen der Linienrechnersoftware erfordern sowohl mehr Speicherkapazität als auch mehr Rechenleistung. Dazu kommt der Wunsch der Kunden nach

mehr Bedienkomfort und entsprechenden Bedienoberflächen. Diese Anforderungen lassen sich zum Teil nur durch den Umstieg auf neue, leistungsfähigere Rechnergenerationen und/oder Betriebssystemversionen realisieren und ziehen mehr oder weniger umfangreiche Software-Änderungen nach sich bzw. erfordern in weiten Teilen Neuentwicklungen.

3.2 Problemfelder bei der Entwicklung

Die Entwicklungskosten für SMD-Bestücklinien setzen sich, wie im Werkzeugmaschinenbau allgemein, aus den Kosten für die mechanische Konstruktion, die Elektrokonstruktion und die Software-Entwicklung zusammen mit wachsendem Wertanteil für die Software-Entwicklung /42, 51, 83, 102/. Die Entwicklung der informationstechnischen Funktionalitäten zum bestimmenden Leistungs- und Kostenfaktor führt nicht nur zu längeren Durchlauf- und Inbetriebnahmezeiten beim Hersteller, sondern auch zu einer Verringerung der technischen Verfügbarkeit beim Betreiber /108/. Insbesondere steigt mit der wachsenden Komplexität der Systeme die Gefahr des Auftretens von Entwicklungsfehlern /4/.

Es hat sich gezeigt, daß die Test- und Inbetriebnahmephasen bei der Entwicklung automatisierter Produktionsanlagen den größten Engpaß innerhalb der Auftragsabwicklung darstellen. Während der Test- und Inbetriebnahmephasen treffen die Komponenten Mechanik, Elektronik und Software aufeinander und viele konzeptionelle Fehler werden erst hier erkannt. Zur Vermeidung von konzeptionellen Fehlern wird eine eindeutige Spezifikation aller Funktionalitäten benötigt, da nur so eine wirkungvolle Programmerstellung möglich ist. Die Erstellung von eindeutigen Spezifikationen beruht auf einem intensiven Informationsaustausch zwischen allen an der Entwicklung beteiligten Abteilungen, insbesondere zwischen der mechanischen Konstruktion und dem Bereich der Steuerungstechnik (Elektronik, Software) /108/. Problematisch ist, daß häufig nach dem gegenseitigen Austausch von Anforderungen während der Analysephase und der Konstruktion die weiteren Entwicklungsarbeiten meist ohne ausreichende gegenseitige Absprachen ablaufen.

Da die Kosten zur Fehlerbehebung um so höher sind, je später sie erkannt werden, und vor allem während der Inbetriebnahme und während des Einsatzes drastisch ansteigen (Bild 21) /102/, ist es also zwingend erforderlich, möglichst viele Fehler bereits in der Testphase zu erkennen und zu beheben. Derzeit werden 54% der Fehler erst während der Nutzungsphase lokalisiert und beseitigt. Zu bemerken ist auch, daß

64% der Fehler ihren Ursprung in der Spezifikations- und Entwurfsphase haben und nur 36% der Fehler auf Unzulänglichkeiten in der Kodierung zurückzuführen sind /38/.

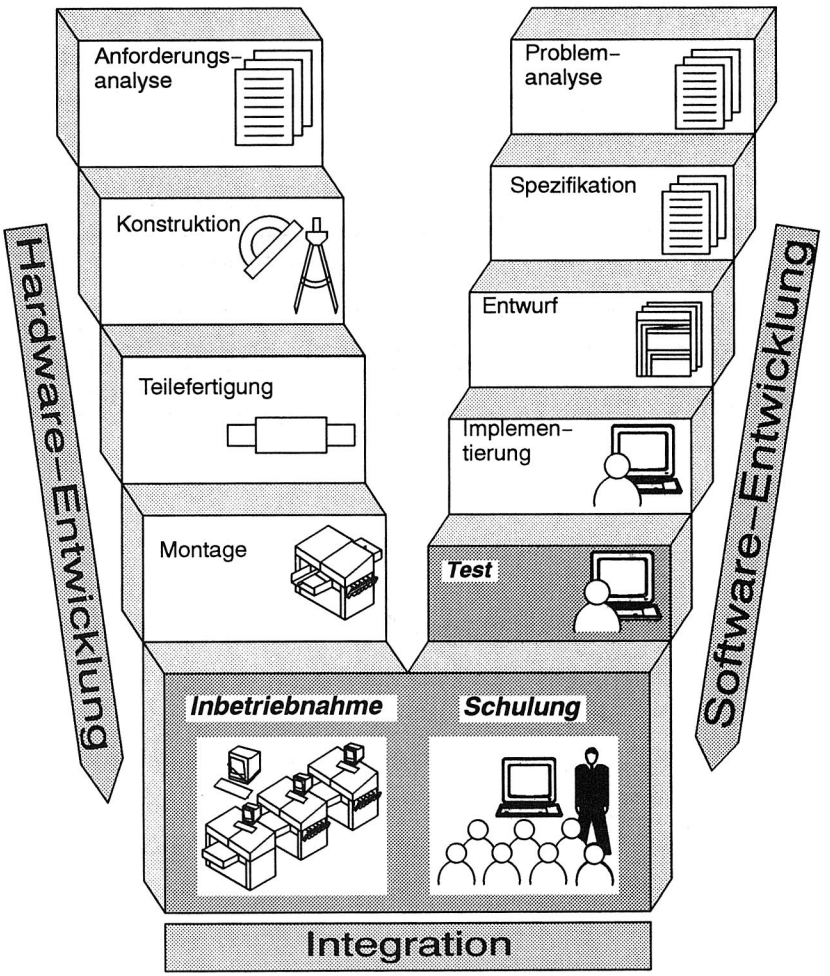


Bild 20: Test , Inbetriebnahme und Schulung als Komponenten der Entwicklung von Fertigungssystemen

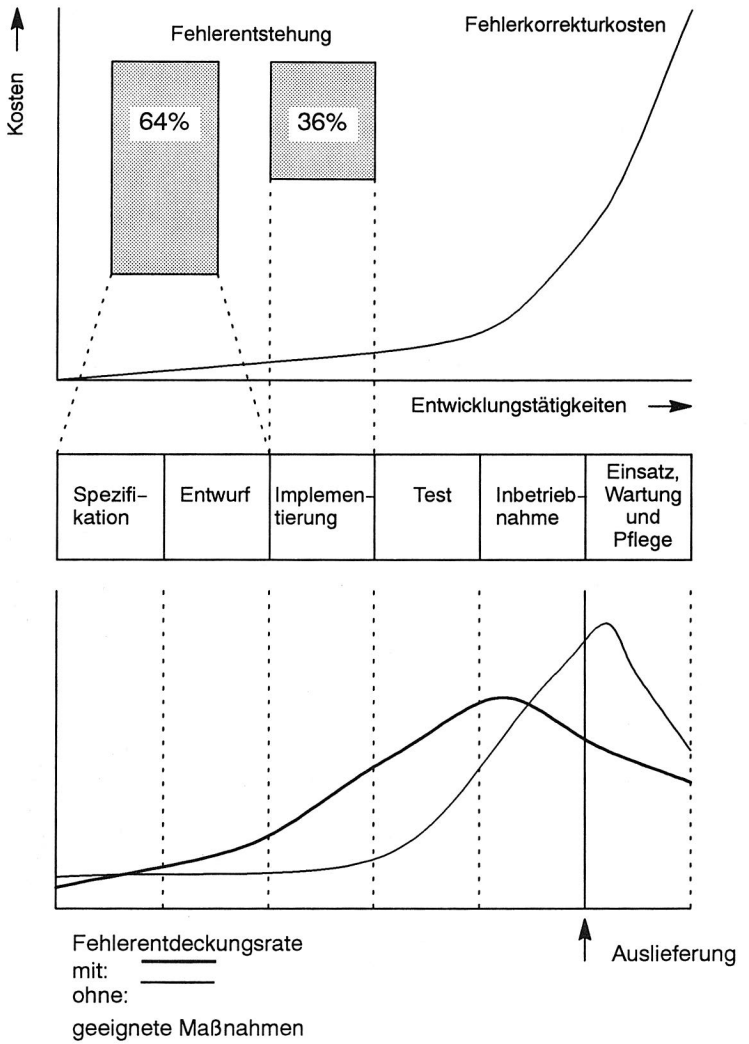


Bild 21: Fehlerentstehung, Fehlerkorrekturkosten und Fehlerentdeckungsrate bei der Entwicklung von Steuerungssoftware

3.3 Abgrenzung von Linienrechnersoftware und allgemeiner Zellenrechnersoftware

Flexible Fertigungszellen und Flexible Montagezellen stellen moderne Automatisierungskonzepte dar. Auf Grund der Heterogenität der eingesetzten Automatisierungsgeräte existiert jedoch für beinahe jede Automatisierungslösung eine individuelle Softwarelösung. Folglich tragen die Kosten für die Softwareerstellung wesentlich zu hohen Gesamtinvestitionskosten bei /88/. Die Entwicklung der Steuerungssoftware erfolgt hierbei häufig erst in der Inbetriebnahmephase neuer Produktionsanlagen, also dann, wenn das Zusammenspiel der Steuerungsmodule mit der Produktionsanlage getestet werden kann. Konzeptionelle Fehler, die erst während der Inbetriebnahme auftreten, führen zur unvorhersehbaren Verzögerungen /65/. Im Gegensatz zu kundenspezifischen Einzellösungen kann bei Serienmaschinen, wie SMD-Bestückautomaten, die einmal erstellte und verifizierte Software in aller Regel wiederverwendet werden. Kundenspezifische Anpassungen bilden hier die Ausnahme.

Gemäß der Unterscheidung zwischen individuellen Lösungen und Serienprodukten kann bei der Anpaßbarkeit von Steuerungssoftware an konkrete Automatisierungslösungen zwischen der "**a priori**" und der "**a posteriori**"-Anpassungsfähigkeit unterschieden werden /19/.

Bei der **a priori**-Anpaßbarkeit wird die Erfüllung aller zu erwartenden Anforderungen von vorneherein in das System mit eingebaut. Die Anpassung an eine spezifische Anwendung erfolgt über die Parametrierung des Systems. Dieser Ansatz ist nur für begrenzte Anwendungsgebiete einsatzfähig, da eine umfassende Kenntnis aller nur möglichen Anforderungen erforderlich ist. Derartige Systeme sind meist sehr komplex. Deshalb sind nachträgliche Änderungen bei zusätzlichen Anforderungen schwer durchzuführen.

Bei der **a posteriori**-Anpaßbarkeit erfolgt die Anpassung der Software durch das Zusammensetzen von bereits vorhandenen oder neu zu erstellenden Basiselementen. Dies setzt eine starke Modularisierung der Software voraus. Derartige Systeme sind weniger komplex als die Systeme mit **a priori**-Anpaßbarkeit. Dadurch reduzieren sich neben der Entwicklungszeit und den Entwicklungskosten bis zum möglichen Ersteinsatz der Software auch die Anforderungen an die benötigten Hardware-Ressourcen. Die Anpassung an spezifische Anwendungen ist durch den erforderlichen Programmieraufwand höher wie bei der a priori-Anpaßbarkeit. Ein weiterer Nachteil

ist, daß durch die entstehende Vielzahl von Software-Versionen die Verwaltung und Aktualisierung erschwert wird.

Bei der Entwicklung von Leitstands- und Zellenrechnersoftware wird häufig ein Kompromiß zwischen a priori- und a posteriori- Anpaßbarkeit geschlossen. Über parametrierbare Kernfunktionen können allgemeingültige Funktionen der Steuerung mit Hilfe unterschiedlicher Konfigurationsdaten für verschiedene Anlagen verwendet werden. Die Anpassung der Kernfunktionen an das Anforderungsprofil der spezifischen Anlage erfolgt mit Hilfe von Anpassungsfunktionen /19, 97/.

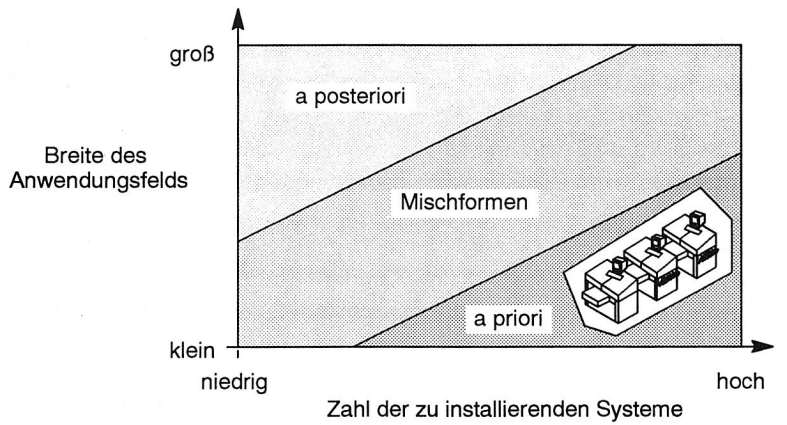


Bild 22: Eignung von a priori und a posteriori-Anpaßbarkeit

Eine Entscheidung zwischen **a priori**- und **a posteriori**-Anpaßbarkeit hängt davon ab, wie groß des betrachtete Anwendungsfeld und wie groß die Zahl der zu realisierenden Systeme ist (Bild 22). Bei kleiner Breite des Anwendungsfeldes und einer größeren Zahl von zu installierenden Systemen bietet es sich an, einmalig mehr Aufwand zu betreiben und die Steuerungssoftware über Parameter an die verschiedenen Anlagen anpaßbar zu gestalten. Im Gegensatz dazu ist es bei großer Breite des Anwendungsfelds mit jeweils nur wenigen zu installierenden Systemen sinnvoller, auf die Modularität der Software zu achten und aus den Modulen jeweils individuelle Lösungen zu generieren.

Für SMD-Bestücklinien gilt nun das folgende:

- Bezüglich der Steuerungssoftware sind herstellerspezifische Lösungen die Regel. Das heißt, daß die Linienrechnersoftware nur zur Steuerung des eigenen Produktspektrums eingesetzt wird. Die Breite des Anwendungsgebietes ist also als klein einzustufen.
- Die Anzahl der zu installierenden Systeme ist als hoch einzustufen.

Es ist daher sinnvoll, bei der Entwicklung der Linienrechnersoftware für SMD-Bestücklinien auf **a priori**-Anpaßbarkeit zu achten und die Anpassung an die individuellen Bestücklinien über die Parametrierung von Anzahl und Ausstattung der Bestückautomaten vorzunehmen.

3.4 Konsequenzen

Bedingt durch die Entwicklungen im SMD-Bereich werden die Phasen der Hardware- und Softwareentwicklung zyklisch durchlaufen. Bezogen auf die Software führt dies am Ende jedes Zyklus zu einer neuen Software-Version, die sowohl umfangreicher als auch komplexer wie die Vorgängerversion ist. Die Folge ist, daß sowohl der Aufwand für die Phase des Softwaretests als auch der Schulungsaufwand zur Bedienung der Bestücklinien enorm ansteigt. Gerade auf den Softwaretest sollte großer Wert gelegt werden, um die Anzahl der Fehler so weit wie möglich zu reduzieren. In den folgenden Kapiteln werden nun Möglichkeiten aufgezeigt, mit denen durch eine geeignete Rechnerunterstützung der Aufwand für Softwaretest und Schulung deutlich reduziert werden kann.

4. Anforderungen an ein Testsystem

4.1 Testen von Software

Das Überprüfen von Software auf Korrektheit ist ein in der Informatik behandeltes Problem. Dabei wird zwischen **Verifikation** und **Testen** unterschieden. Bei der Verifikation wird das korrekte Verhalten eines Programms mit formalen Mitteln nachgewiesen. Dieser Ansatz hat sich in der Praxis nicht durchgesetzt, da selbst zur Beschreibung trivialer Programme komplexe mathematische Notationen notwendig sind. Bei komplexen Systemen ist die Verifikation auf Grund des Aufwands völlig unmöglich /53/.

Testen ist neben anderen qualitätssichernden Maßnahmen die am weitesten verbreitete analytische Maßnahme zur Aufdeckung von Fehlern in Software-Produkten:

> "Ein Fehler in einem Software-Produkt ist jegliche Abweichung in Inhalt, Aufbau und Verhalten eines Testobjekts zwischen ermittelten, beobachteten und gemessenen Daten einerseits und den entsprechenden in den Zielvorgaben spezifizierten oder theoretisch gültigen Daten andererseits" /94/.

Testen ist eine informelle Vorgehensweise, durch die zwar hinsichtlich bestimmter Qualitätsmerkmale (Richtigkeit, Robustheit, usw.) das Vorhandensein von Fehlern, nicht aber deren Nicht-Vorhandensein nachgewiesen werden kann. Der Erfolg des Testens ist im wesentlichen von der systematischen Vorgehensweise abhängig. Die prinzipielle Vorgehensweise besteht aus den vier in Bild 23 dargestellten Schritten, die solange zyklisch durchlaufen werden, bis keine Fehler mehr gefunden werden.

Für die Durchführung von Tests gibt es verschiedene Testarten /35/. Diese können differenziert werden nach

1. dem Grad der Einbeziehung des Testobjekts bei der Testvorbereitung und Testauswertung in **Black-Box-** und **White-Box-Testen**

   Beim Black-Box-Testen werden die Testfälle auf der Grundlage des im Pflichtenheft spezifizierten Verhaltens des Programms ausgewählt. Der Interne Aufbau des Programms wird nicht berücksichtigt. Der Nachteil beim Black-Box-

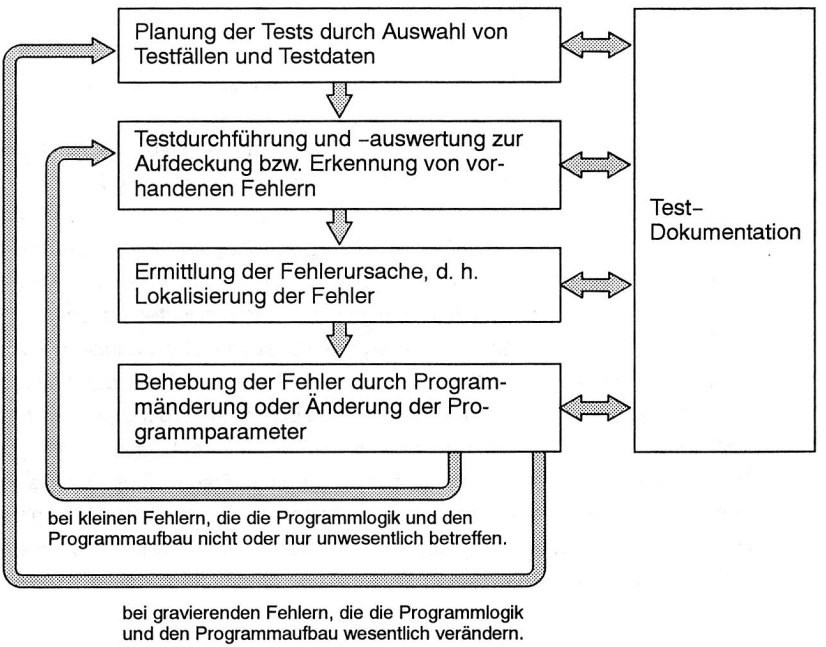


Bild 23: Prinzipieller Ablauf von Softwaretests

Testen ist daher, daß durch die fehlende Berücksichtigung der Programmstruktur nicht sichergestellt ist, daß alle Teile des Programms beim Testen durchlaufen werden. Darüberhinaus werden beim Black-Box-Test lediglich die Auswirkungen der Fehler erkannt, nicht aber deren Ursachen.

Der White-Box-Test ist ein Testverfahren, bei dem die Testfälle und Testdaten auf der Grundlage der Programmstruktur ausgewählt werden. Aus der kombinatorischen Vielzahl von Möglichkeiten, mit denen ein Programm auf Grund von Verzweigungen und Schleifen durchlaufen werden kann, werden repräsentative Pfade ausgewählt. Es empfielt sich, die Auswahl so zu treffen, daß primär diejenigen Programmteile durchlaufen werden, auf denen am ehesten mit Fehlern gerechnet werden muß (neue Module) oder auf denen auftretende Fehler die gravierendsten Auswirkungen haben. Der Nachteil des White-Box-Testen

besteht darin, daß nicht feststellbar ist, ob das Programm alle im Pflichtenheft spezifizierten Vorgaben erfüllt.

2. der Art der Testausführung in **Statisches** und **Dynamisches Testen**

   Beim statischen Testen wird das Software-Produkt an Hand von Listings oder Struktogrammen als Datenbestand analysiert und ausgewertet (Code-Inspektion, Walkthrough). Im Gegensatz dazu wird beim dynamischen Testen das Programm wirklich ausgeführt. Zur Verfolgung des Programmablaufs werden Debugger eingesetzt, die ein schrittweises Abarbeiten des Programms, das Setzen von Breakpunkten und das Anzeigen von Variablenwerten ermöglichen. Da bei Einsatz von Debuggern der Programmablauf meist verlangsamt wird, bzw. durch das schrittweise Abarbeiten und das Setzen von Breakpunkten sogar unterbrochen wird, können Programme mit Echtzeitanforderungen nur bedingt mit Debuggern getestet werden. Um ein Programm mit einem Debugger testen zu können, muß der Quellcode mit speziellen Debug-Optionen übersetzt werden. Dadurch werden zusätzliche Informationen in die vom Compiler erzeugten Objekt-Dateien eingefügt, die die ausführbaren Programme um ein vielfaches vergrößern.

3. dem Umfang der Testausführung und der Bereitstellung der Testdaten in **Repräsentatives Testen**, **Statistisches Testen** und **Schwachstellenorientiertem Testen**.

   Beim repräsentativen Testen werden genau festgelegte Teile des Testobjekts betrachtet (bei einer Steuerungssoftware z. B. nur die Kommunikationsfunktionen). Bei der statistischen Testart werden die zu testenden Programmteile und die Testdaten zufallsabhängig ausgewählt. Das schwachstellenorientierte Testen orientiert sich bei der Auswahl der Testfälle an Vermutungen über bestimmte Fehlerarten, die aus Erfahrung oder Intuition resultieren.

Ein wichtiger Bestandteil von Softwaretests ist die Testdokumentation, die parallel zu den anderen Testaktivitäten durchzuführen ist. Die Testdokumentation beinhaltet das Erfassen, Ordnen, Speichern und Bereitstellen von Daten für das Testen und/oder über den Ablauf und die Ergebnisse des Testens. Insbesondere unter dem Gesichtspunkt der Produkthaftung dient die Testdokumentation als Nachweis dafür, daß aufgetretene Schäden nicht auf die Software zurückzuführen sind /55/.

4.2 Testen der Anlagenkommunikation durch Simulation

Unabhängig davon, ob White-Box oder Black-Box-Testverfahren angewendet werden und wie groß der Umfang der Testausführung ist, müssen bei dynamischen Softwaretests im Gegensatz zu statischen Tests während der Programmausführung alle Ein/Ausgabe-Beziehungen des Systems erfüllt werden. Dazu zählen neben den Schnittstellen zur Datenverwaltung und zum Benutzer auch die Ein/Ausgabebeziehungen zu der zu steuernden Anlage. Vielfach werden die Tests deshalb an den realen Anlagen durchgeführt.

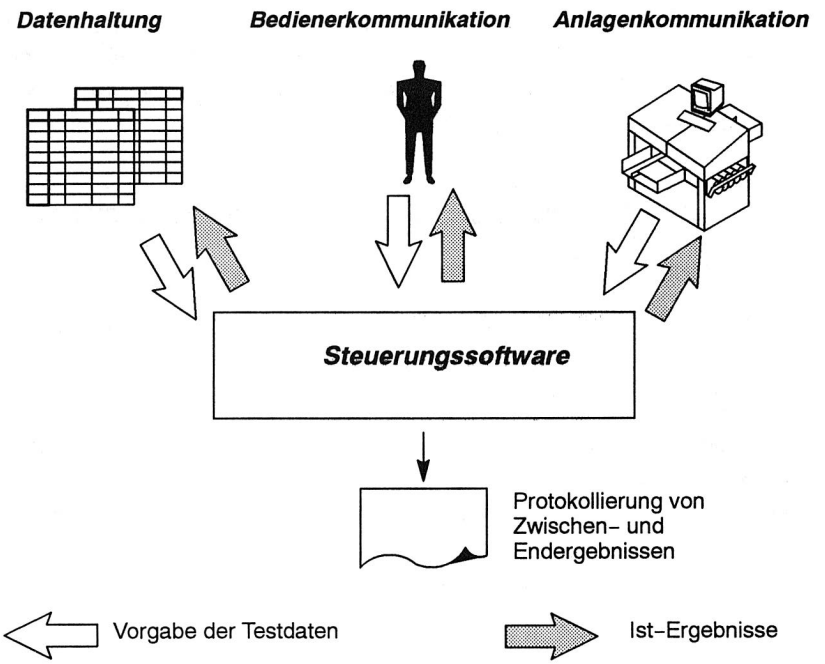


Bild 24: Datenkanäle von und zur Steuerungssoftware

Das hat folgende Nachteile:

- Die Tests können erst zu einem sehr späten Zeitpunkt durchgeführt werden, nämlich dann, wenn die Anlagen bereits aufgebaut sind.

- Fehler in der Software können durch Vorgabe falscher Steuerinformationen zu Beschädigungen der Anlagen führen.
- Es entstehen hohe Kapitalbindungskosten, da die Tests und das anschliessende Beseitigen von Fehlern relativ lange dauern und der größte Teil der Investitionen bereits durchgeführt wurde.
- Während der Tests entstehen Kosten für Verbrauchsmaterial und Energie.
- Software mit hoher a-priori-Anpaßbarkeit kann nur für einige wenige Anlagenkonfigurationen getestet werden.

Um nun den Entwicklern ganzheitliche Tests der Steuerungssoftware unabhängig von den realen Fertigungsanlagen zu ermöglichen, werden Testumgebungen benötigt, die während der Testphasen die realen Anlagen ersetzen und dasselbe Kommunikationsverhalten wie diese aufweisen. Dabei bietet sich die Simulation als geeignetes Werkzeug an /66, 84, 62, 52/ (Bild 25).

4.3 Klassifizierung und Bewertung von Simulationssystemen

Nach der VDI-Richtlinie 3633 /107/ ist die Simulation das Nachbilden eines dynamischen Prozesses in einem System mit Hilfe eines experimentierfähigen Modells, um zu Erkenntnissen zu gelangen, die auf die Wirklichkeit übertragbar sind.

Die Simulationstechnik ist ein allgemein anerkanntes Hilfsmittel bei Planung, Realisierung und Betrieb von technischen Systemen /107, 2, 18/. Der Schwerpunkt des Simulationseinsatzes lag ursprünglich auf der Planungsabsicherung. Zunehmend wird die Simulation durchgängig in allen Phasen des Planungs- und Realisierungsprozesses genutzt, und findet auch Anwendung in der Prozeßsteuerung während des Betriebes zur Dispositionsunterstützung und zur Anlagenüberwachung.

Die Simulation wird eingesetzt, wenn mathematisch-analytische Methoden bei der Untersuchung von Systemen an ihre Grenzen stoßen. Mit Hilfe der Simulation kann das zeitliche Ablaufverhalten komplexer technischer Systeme ohne Einschränkungen untersucht und beurteilt werden /107/.

Unter einem System wird dabei eine abgegrenzte Anordnung von Komponenten verstanden, die miteinander in Beziehung stehen. Das System ist gekennzeichnet durch

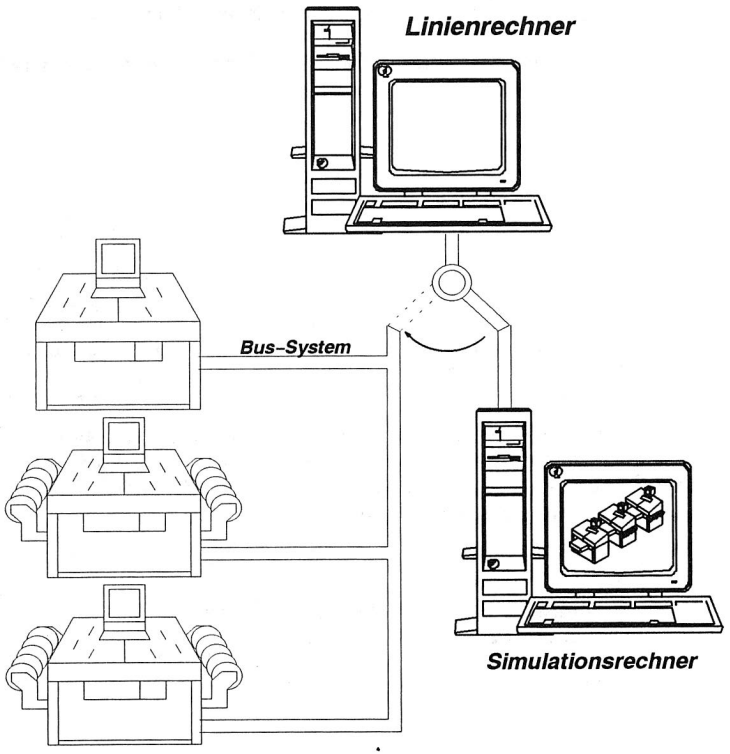


Bild 25: Simulation von Bestückautomaten zum Test der Steuerungssoftware und zur Schulung

die Zustände seiner Komponenten, die wiederum durch die Angabe der Werte aller konstanten und variablen Attribute (Zustandsgrößen) beschrieben werden. Die Abbildungsgenauigkeit sollte nicht so groß wie möglich sein, sondern so groß wie zur Zielerfüllung erforderlich. Zustandsübergänge der Komponenten ergeben sich durch die kontinuierliche oder diskrete Änderung mindestens einer Systemvariablen.

Signifikante Vorteile der Simulation liegen in der Möglichkeit der Untersuchung

- real (noch) nicht existierender Systeme,
- real existierender Systeme ohne direkten Betriebseingriff
- mehrerer Gestaltungsvarianten bei geringem Arbeitsaufwand

- des Systemverhaltens über lange Zeiträume hinweg (Zeitraffung),
- von Anlaufvorgängen, Einschwingphasen und Übergängen zwischen definierten Betriebszuständen.

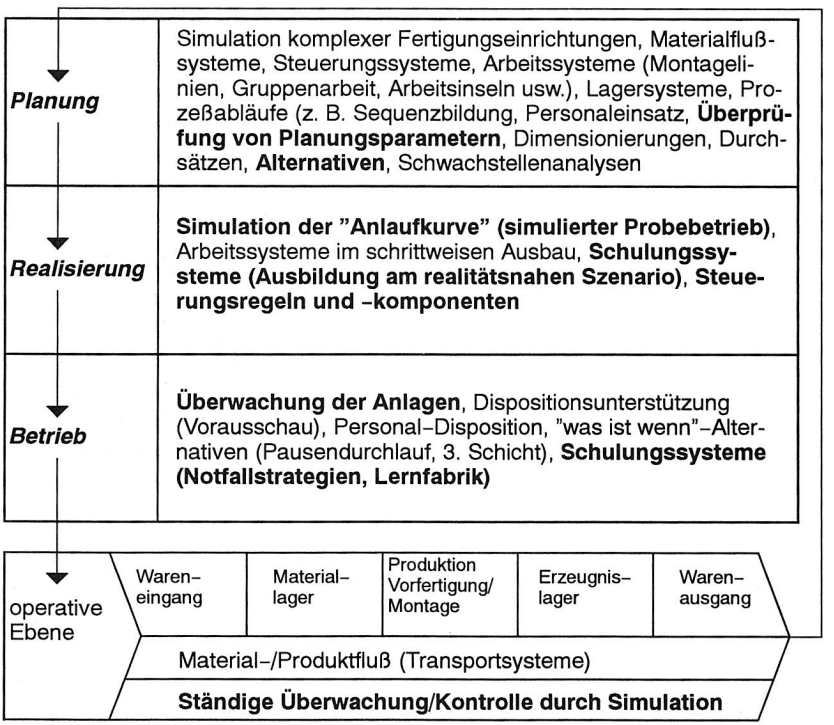


Bild 26: Einsatz der Simulation im Lebenszyklus von technischen Systemen nach /107/

Der Ablauf einer Simulation läßt sich untergliedern in die Modellierung, bei der ein konkretes Originalsystem in ein experimentierbares Softwaremodell umgesetzt wird, die eigentliche Durchführung der Simulationsexperimente und die anschließende Analyse der Ergebnisse.

4.3.1 Klassifikation nach Planungsebenen

Simulationssysteme können im Bereich der Produktion und Logistik auf unterschiedlichen Hierarchieebenen mit unterschiedlichen Zielrichtungen eingesetzt werden (Bild 27). In Abhängigkeit von der Zielrichtung wird prinzipiell zwischen Ablaufsimulation, 3D-Bewegungssimulation und der Finiten Elemente Methode (FEM) unterschieden.

Die Ablaufsimulation wird in unterschiedlichen Detaillierungsstufen hauptsächlich zur Untersuchung von logistischen Zusammenhängen (Materialfluß) und Steuerungsstrategien eingesetzt. Die eingesetzten Modelle bilden die kapazitiven und zeitlichen Zusammenhänge in den untersuchten Produktionssystemen nach. Mit Hilfe der 3D-Bewegungssimulation lassen sich Kollisions- und Arbeitsraumuntersuchungen von Fertigungs- und Montagezellen durchführen. Die Modellierung erfolgt mit 3D-Volumenmodellen und ist somit wesentlich detaillierter als die Modellierung bei der Ablaufsimulation. Die Struktur und die Belastung einzelner Komponenten wird mit der FEM-Simulation untersucht. Dazu werden die zu untersuchenden Komponenten in passende, endlich große Elemente zerlegt, die über Knoten miteinander verbunden sind /67, 109/.

Für den Test von Steuerungssoftware werden abhängig von der Hierarchiestufe, auf der die Software eingesetzt wird, Simulationsmodelle mit unterschiedlichem Detaillierungsgrad benötigt /65/. Auf Maschinenebene ist vor allem die Kollisionsfreiheit von Bedeutung. Zur Untersuchung der hier eingesetzten NC- und RC-Programme sind bereits geeignete Simulationswerkzeuge verfügbar /20, 52, 58, 96, 114/. Auf den höheren Ebenen ist die Ablaufsimulation ausreichend um die Funktion von Steuerprogrammen zu überprüfen. Es sind hier allerdings noch keine Werkzeuge mit einem ausreichendem Entwicklungsstand vorhanden, die den Aufbau von Modellen für den Test von Steuerungssoftware ermöglichen /2, 8/. Da die Ablaufsimulation weniger rechenzeitintensiv ist als die 3D-Bewegungssimulation, kann im gleichen Zeitraum eine wesentlich größere Anzahl von Testfällen erzeugt werden, wodurch die Anzahl der gefundenen Fehler steigt /3/.

4.3.2 Klassifikation von Simulationsinstrumenten nach Entwicklungsumgebung

Simulationsinstrumente können nach ihrer Entwicklungsumgebung in 3 Klassen eingeteilt werden /107/. In der ersten Klasse befinden sich anweisungsorientierte, ob-

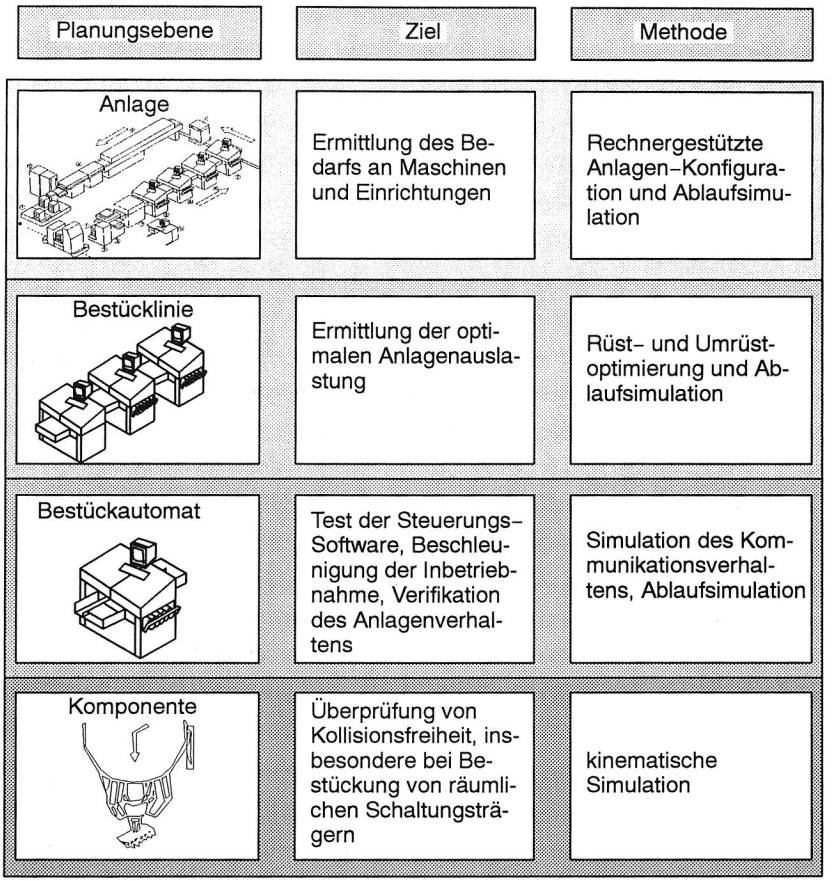


Bild 27: Einsatz der Simulation auf unterschiedlichen Planungsebenen

jektorientierte und blockorientierte Programmiersprachen, die hinsichtlich Anwendungsgebiet und Fragestellung die größtmögliche Flexibilität bieten, insbesondere bei der Anbindung an Kommunikationssysteme. Der Nachteil liegt im hohen Aufwand und der Fehleranfälligkeit bei der Erstellung und Änderung der Simulatoren.

Simulationsentwicklungsumgebungen stellen alle erforderlichen Simulatorkomponenten bereit und bieten die Möglichkeit zur graphisch-interaktiven Modellerstellung

Klasse 1	**a) anweisungsorientierte Programmiersprachen** keine Anpassung an simulationsspezifische Belange z. B.: Fortran, Pascal, C
	b) objektorientierte Programmiersprachen Die wesentlichen Eigenschaften des Simulationsinstruments werden mit Hilfe von Klassenkonzepten, Koroutinenkonzepten oder Listenverwaltungen beschrieben.
	c) blockorientierte Programmiersprachen Auf dieser Ebene werden Komponenten für Modellklassen bereitgestellt. Mit Hilfe von Petri-Netzen und Sprachkonzepten können allgemeine Instrumente für beliebige Anwendungen realisiert werden. Typische Simulatorfunktionen, wie z. B. die Ereignisverwaltung werden unterstützt.
Klasse 2	**Simulationsentwicklungsumgebungen** Es werden Komponenten für spezifische Anwendungsbereiche, wie z. B. Produktion oder Logistik, bereitgestellt. Häufig sind es baustein- oder listenorientierte Simulatoren mit relativ breitem Anwendungsspektrum. z. B.: GRAPHSIM, SIMPLE, DOSIMIS-3
Klasse 3	**Simulatormodellierungsumgebungen** Hier befinden sich Simulationsinstrumente, die spezifisch für Teilgebiete eines Anwendungsbereichs sind, wie z. B. Simulatoren für fahrerlose Transportsysteme, flexible Fertigungssysteme oder Montagesysteme.

Bild 28: Einteilung von Simulationsinstrumenten nach Entwicklungsumgebung /25/

und -änderung unter Verwendung parametrierbarer Modellbausteine. Das eingegebene Modell wird durch die Simulationsentwicklungsumgebung in ein ablauffähiges Softwaremodell umgesetzt. Im Vergleich zu den Programmiersprachen reduziert sich der Enwicklungsaufwand und die Fehleranfälligkeit. Die Flexibilität bleibt weitgehend erhalten, da nur wenige und relativ allgemeine Bausteine für die Modellierung zur Verfügung stehen.

In einer Simulatormodellierungsumgebung erfolgt die Modellerstellung ebenfalls graphisch-interaktiv auf der Basis von parametrierbaren Modellbausteinen, wobei allerdings spezialisiertere und komplexere Bausteine verwendet werden. Die Spezialisierung der Modellbausteine führt zu einer Einengung auf bestimmte Anwendungsbereiche bei erhöhter Anwenderfreundlichkeit und reduziertem Aufwand für die Modellerstellung. Simulatormodellierungsumgebungen werden entweder unter Verwendung von Programmiersprachen oder von Simulationsentwicklungsumgebungen entwickelt und für unterschiedliche Anwendungsgebiete angeboten.

Kriterien / Entwicklungs-umgebung	Flexibilität	Erweiterbarkeit	Kopplungsfähigkeit zu anderen Hilfsmitteln	Anschlußmöglichkeit an Kommunikationssysteme	Benutzerfreundlichkkeit	Modellierungsaufwnad	Selbsterkläruing
Klasse 1a anwendungsorientierte Programmiersprachen	●	●	●	●	○	○	○
Klasse 1b objektorientierte Programmiersprachen	●	●	●	●	○	○	○
Klasse 1c blockorientierte Programmiersprachen	◒	◒	◒	◒	○	○	○
Klasse 2 Simulationsentwicklungsumgebungen	○	○	○	○	◒	◒	◒
Klasse 3 Simulatormodellierungsumgebungen	○	○	○	○	●	●	●

● Kriterium ist gut erfüllt
◒ Kriterium ist einigermaßen gut erfüllt
○ Kriterium ist schlecht erfüllt

Bild 29: Bewertung von Simulationssoftware nach /2/

In Bild 28 erfolgt eine Bewertung der Simulationswerkzeuge auf den verschiedenen Implementierungsebenen.

4.3.3 Weitere Klassifikationsmöglichkeiten

Interne algorithmische Konzeption
Nach der Art der internen algorithmischen Konzeption für die Ablaufsteuerung werden taktgetriebene, ereignisgetriebene und aktivitätsgetriebene Simulationssysteme unterschieden. Bei den taktgetriebenen Simulationssystemen erfolgt die Ablaufsteuerung auf der Basis konstanter Modellzeitschritte. Solche Systeme sind zum Test von Steuerungssoftware nicht einsetzbar, da sich die Abläufe bei einer Steuerung nicht im Abstand von Zeitschritten abspielen.

Ereignisgetriebene Simulationssysteme, bei denen die Ablaufsteuerung auf der Basis von Ereigniszeitpunkten mit spezieller Ereignisverwaltung abläuft, sowie aktivitätsgetriebene Simulationssysteme, bei denen die Ereignisverwaltung aus dem Fluß dynamischer Objekte generiert wird, sind zum Test von Steuerungssoftware geeignet, da auch die fertigungstechnische Realität durch Ereignisse, wie z. B. Start der Bearbeitung, Ende der Bearbeitung, Ankunft von Material usw., geprägt ist.

Interaktionsmöglichkeit
Zum Testen von Steuerungssoftware werden interaktive Simulationssysteme benötigt, bei denen es möglich ist, während des Simulationslaufes Ein- und Ausgaben von Daten vornehmen zu können. Simulationssysteme, die im Stapelbetrieb arbeiten, sind zum Test von Steuerungssoftware ungeeignet, da die Simulationsergebnisse erst nach der vollständigen Ausführung ausgegeben werden. Das gleiche gilt für Systeme, die während der Laufzeit lediglich verschiedene Sichten auf die Modelldaten erlauben. Hier sind während des Simulationslaufs keine Eingaben möglich.

Über die Interaktionsmöglichkeit sollte jederzeit ein Wechsel zwischen dem automatischen Ablauf der Simulation und einem manuellen Modus möglich sein. Im manuellen Modus können die auszuführenden Prozeßschritte einzeln zur Durchführung freigegeben werden und damit gezielt die Reaktionen der getesteten Steuerungssoftware beobachtet werden. Darüberhinaus können Störungen provoziert werden, um die Robustheit der Software in Extremsituationen zu überprüfen. Durch eine gezielte oder permanente Ausgabe von Modellparametern in Form einer Prozeßvisualisierung erhält der Benutzer einen Einblick über den Zustand des Simulationsmodells und kann daraus die weiteren Aktionen ableiten.

4.4 Anforderungen an Simulationssysteme zum Test von Steuerungssoftware

Simulatoren, die zum Test von Steuerungssoftware und zur Schulung eingesetzt werden sollen, müssen eine Reihe von Anforderungen erfüllen, die über die Fähigkeiten von Simulatoren zur Anlagenplanung hinausgehen. Diese Anforderungen resultieren daraus, daß das System während der Simulation mit der Steuerungssoftware kommunizieren muß. Dabei sind die Vorgaben der Steuerungssoftware entgegenzunehmen und zu verarbeiten. Die Ergebnisse der Simulation werden in Form von Rückmeldungen an die getestete Steuerung zurückgemeldet.

Kopplungsmöglichkeit an das zu testende System
Als problematisch erweist sich bei den zur Anlagenplanung eingesetzten Simulationssystemen die Anbindung an die Kommunikationsschnittstelle des zu testenden Systems. Die Anbindung muß sowohl Hardware- als auch Software-technisch erfolgen. Das heißt, daß für den Rechner, auf dem das Simulationssystem läuft, die notwendigen Anschlußmöglichkeiten vorhanden sein müssen, z. B. in Form von Einsteckkarten, und daß für das Betriebssystem, unter dem das Simulationssystem läuft, die notwendigen Treiber vorhanden sein müssen. Der Zugriff auf die Kommunikationsschnittstelle erfordert dann trotz allem noch Programmieraufwand in einer problemorientierten Sprache wie z. B. C, Pascal oder Fortran. Eine enge Zusammenarbeit zwischen einem Simulationsexperten und einem Netzwerkspezialisten ist hier erforderlich.

Der Anpassungsaufwand an unterschiedliche Kommunikationsmechanismen entfällt, wenn ein einheitliches Protokoll verwendet wird, wie z. B. MAP und MMS /98, 46/.

Zwei mögliche Arten der Anbindung der Simulation an die Kommunikation unter dem Betriebssystem UNIX sind in Bild 30 aufgezeigt. Im ersten Fall werden die Ein/Ausgabe-Möglichkeiten des Simulationssystems auf Datei-Basis verwendet. Die Datenströme werden in (Named-)Pipes umgeleitet, an deren anderen Ende ein Kommunikationsprozeß angekoppelt ist. Dieser realisiert den Zugriff auf die Kommunikation über die entsprechenden Treiber.

Im zweiten Fall erfolgt der Zugriff auf die Kommunikation direkt vom Simulationssystem aus, indem mit Hilfe einer vorhandenen Programmierschnittstelle entsprechende Funktionen dazugebunden werden. Dies ist im übrigen bei Simulationssyste-

men, die unter Single-Tasking-Betriebssystemen, wie z. B. MS-DOS, laufen, die einzige Möglichkeit des Zugriffs.

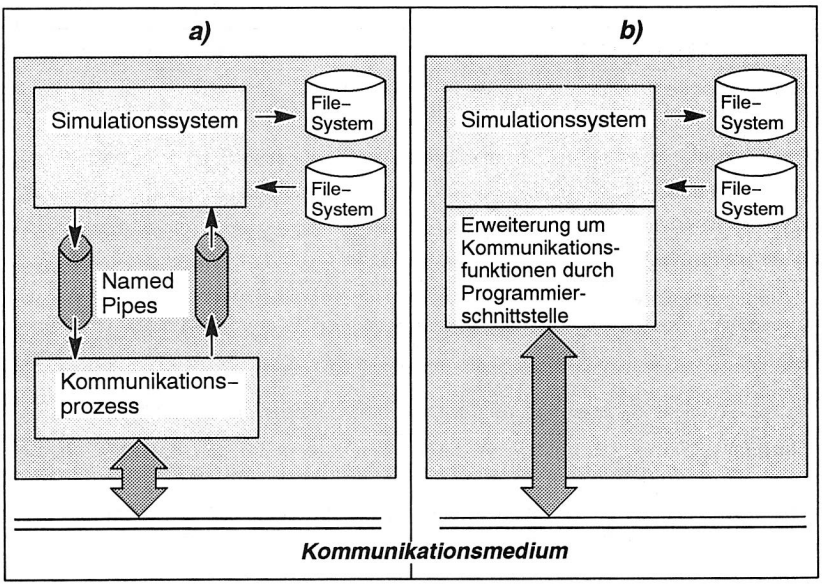


Bild 30: Anschlußmöglichkeiten eines Simulationssystems auf einem UNIX-Rechner an das Kommunikationsmedium

Analyse von Datentelegrammen

Simulationssysteme, die zum Test von Steuerungssoftware eingesetzt werden, müssen während der Tests die empfangenen Datentelegramme analysieren, verarbeiten und daraus entsprechende Antworten generieren. Datentelegramme dienen zum Übertragen von Befehlen und/oder Daten und sind durch folgende Kriterien gekennzeichnet:

1. Sender, Empfänger
   Sender und Empfänger werden aus kommunikationstechnischer Sicht durch eindeutige Adressen identifiziert oder ergeben sich implizit, wie z. B. bei 1:1-Kopplungen, über die serielle RS232-Schnittstelle. Durch Angabe von logischen Namen für Sender und Empfänger in den Telegrammen wird die Telegrammauswertung unabhängig von den eingestellten Adressen, bzw. unter einer Adresse können mehrere logische Objekte angesprochen werden.

Kriterien zur Charakterisierung von Telegrammen:	implizit	explizit	optional
physikalische Senderadresse	✓	✓	✓[1]
logische Senderadresse		✓	✓[2]
physikalische Empfängeradresse	✓	✓	✓[1]
logische Empfängeradresse		✓	✓[2]
Telegrammtyp	✓	✓	
Nutzdaten		✓	
Telegrammlänge	✓	✓	✓
Übertragungszeit	✓	✓	✓
Übertragungsparameter			✓

explizit: Das Kriterium ist als Komponente im Telegramm enthalten.

implizit: Das Kriterium ist nicht als Komponente im Telegramm enthalten, kann aber ermittelt werden.

optional: Das Kriterium ist zur Analyse des Telegramms nicht unbedingt erforderlich.

[1] Die physikalische Adresse kann fehlen, wenn eine logische Adresse vorhanden ist.

[2] Die logische Adresse kann fehlen, wenn die physikalische Adresse vorhanden und eindeutig ist.

Bild 31: Kriterien zur Charakterisierung von Telegrammen

2. Telegrammtyp
   Der Typ eines Datentelegramms (Start, Stop, Maschinendaten, usw.) ergibt sich explizit durch eine entsprechende Codierung im Telegramm oder implizit durch den Zustand der Steuerungsgeräte.

3. Übertragungszeitpunkt
   Aus den Übertragungszeitpunkten ergeben sich die Zeitpunkte, an denen auf die empfangenen Telegramme reagiert werden muß.

4. Übertragungsstatus
   Im Übertragungsstatus werden Fehler vermerkt, die während der Übertragung

aufgetreten sind und zu einem Abbruch oder einer Wiederholung der Übertragung geführt haben.

5. Übertragungsparameter
   Unter Übertragungsparametern werden die zur Parametrierung des verwendeten Kommunikationsmechanismus verwendeten Werte, wie z. B. Baud-Rate, Timeout-Intervall, Handshake-Verfahren und Übertragungsendekennzeichen verstanden.

6. Nutzdaten
   Der Aufbau der Nutzdaten ergibt sich aus dem Telegrammtyp. Die Nutzdaten können im Binär- oder im ASCII-Format übertragen werden. Sie enthalten numerische Werte im Integer- oder Real-Format, Bit-Informationen oder Strings, bzw. jeweils Felder davon. Felder können von fester oder variabler Länge sein. Mit Hilfe von Flags wird die Existenz von optionalen Telegrammkomponenten angezeigt. Bei der Übertragung von Binärdaten ist auf die verwendeten Zahlendarstellungen von Integer- und Real-Zahlen zu achten.

Generieren von Datentelegrammen

Die Anworttelegramme, die von einem Simulationssystem an die zu testende Steuerungssoftware zurückgemeldet werden, müssen den selben Aufbau und Inhalt haben wie die von einer realen Anlage erzeugten Telegramme. Auf Grund der funktionalen Abhängigkeit zu den empfangenen Datentelegrammen kann die Generierung der Antworttelegramme bezüglich ihrer Komplexität in die in Bild 32 gezeigten 4 Ebenen eingeteilt werden.

Auf der ersten Ebene besteht zwischen dem Inhalt der Datentelegramme und den Antworttelegrammen keinerlei funktionale Abhängigkeit, d. h. der Inhalt der Antworttelegramme kann fest vorgegeben werden. Nach diesem Schema arbeiten z. B. einfache Schnittstellentester, die nach dem Empfang eines bestimmten Byte-Musters eine vorgegebene Antwort liefern.

Auf der zweiten und dritten Ebene besteht eine einfache funktionale Abhängigkeit zwischen Datentelegrammen und Antworttelegrammen. Teile der empfangenen Datentelegramme können in die Antworttelegramme an vorgegebener Stelle eingefügt werden. Auf der dritten Ebene können die Daten zuvor mit einfachen Bit-Operationen, String-Operationen oder arithmetischen Operationen verändert werden. Um z.

B. die Fertigmeldung für einen Auftrag zu erzeugen, ist es notwendig, die Auftragsnummer, die in einem Datentelegramm übertragen wurde, in das Antworttelegramm zu übernehmen.

Die vierte Ebene ist geprägt durch eine komplexe funktionale Abhängigkeit zwischen Datentelegrammen und Antworttelegrammen. Diese Ebene ist vor allem dann notwendig, wenn die Antworttelegramme Daten enthalten, die aus dem Prozeßverlauf heraus entstehen, wie z. B. Maschinen- oder Betriebsdaten. Im Falle der SMD-Be-

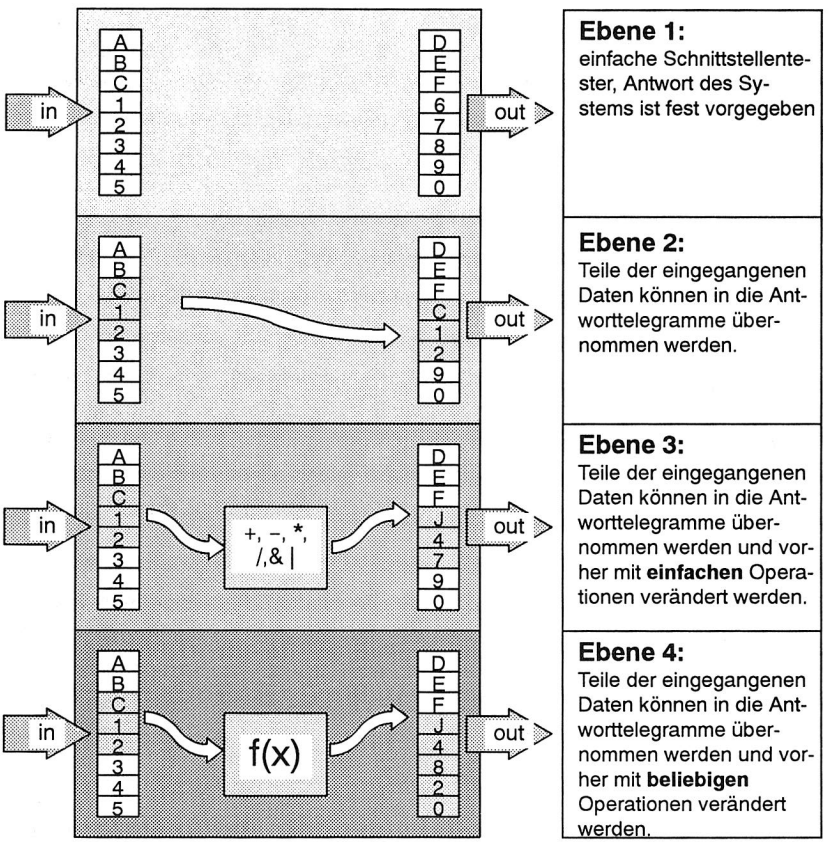


Bild 32: Einteilung der syntaktischen Verarbeitung von Nachrichten in vier Ebenen

stückung werden von den Bestückautomaten unter anderem die Bearbeitungszeiten und die Anzahl der verbrauchten Bauelemente an den Linienrechner gemeldet. Diese Daten sind funktional abhängig von den zuvor übertragenen Maschinen-, Rüst- und Nutzendaten.

Echtzeitfähigkeit

Die Testumgebung muß die gleiche Echtzeitfähigkeit besitzen wie die simulierte Anlage. Das bedeutet zum einen, daß die Reaktionszeit des Systems gewisse Maxima nicht überschreiten darf, was zu Time-Out-Problemen führt. Zum anderen müssen die Antworten der Testumgebung in der gleichen Reihenfolge wie die der simulierten Anlage generiert werden. Zum Austesten der Grenzbelastung kann der zeitliche Ablauf im Simulationssystem beschleunigt oder verzögert werden.

Die Echtzeitfähigkeit der Testumgebung ist sowohl vom verwendeten Simulationssystem als auch vom verwendeten Betriebssystem abhängig. Das Simulationssystem muß über eine Echtzeit-Uhr verfügen, um das Anstoßen von Ausgaben zur richtigen Zeit veranlassen zu können.

Anpaßbarkeit an unterschiedliche Anlagenlayouts

Eine Steuerungssoftware sollte im allgemeinen durch entsprechende Parameter an verschiedene Anlagenlayouts anpaßbar sein. Die Layouts können sich durch die Anzahl und die Art sowie die Ausstattung der zu steuernden Komponenten unterscheiden. Ein Test der Steuerungssoftware an einer realen Anlage mit verschiedenen Layouts bedingt unter Umständen einen Umbau der Anlage und ist damit sehr aufwendig. Die Testumgebung dagegen sollte ebenso einfach parametrierbar sein wie das zu testende System.

Umgekehrt sollte es nicht nötig sein, das zu testende System an das Simulationssystem anpassen zu müssen. Zum ersten entsteht dadurch zusätzlicher Aufwand und zum zweiten können durch die Anpassungen zusätzliche Fehler auftreten.

Automatische und Manuelle Simulation

Im Automatikbetrieb verhält sich das Simulationssystem gegenüber dem zu testenden System wie die reale Anlage, ohne daß Benutzereingaben erforderlich sind. Der Automatikbetrieb ist vor allem für Langzeittests von Vorteil. Darüberhinaus sollte das Simulationssystem dem Benutzer aber auch die Möglichkeit geben, jederzeit eingreifen zu können, um z. B. den Systemzustand zu ändern oder Störungen zu provozie-

ren. Anzahl und Zeitpunkte der erzeugten Störungen im Automatikbetrieb sollten über Wahrscheinlichkeitsfunktionen parametrierbar sein.

Protokollierung
Während eines Simulationslaufes sollten alle Informationen, die ein nachträgliches Rekonstruieren und ein Lokalisieren eventuell aufgetretener Fehler ermöglichen, aufgezeichnet werden. Das Simulationssystem hat dabei nur Zugriff auf die über das Kommunikationssystem übertragenen Informationen, nicht aber auf interne Zustände und Daten des zu testenden Systems. Im Protokoll sollte der Übertragungszeitpunkt, die übertragene Information, der Übertragungsstatus sowie Sender und Empfänger enthalten sein. Die Protokolldateien können als Eingabe für ein Playback dienen, bei dem zur Auswertung die aufgezeichneten Vorgänge je nach Anforderung verlangsamt oder mit höherer Geschwindigkeit abgespielt werden.

Visualisierung
Während der Simulation sollten die Zustände und eventuell interne Parameter der simulierten Komponenten visualisiert werden. Die Visualisierung erfolgt über Grafiken oder über textuelle Ausgaben. Besonders geeignet sind hierbei grafische Benutzeroberflächen, wie z. B. MS-Windows.

4.5 Kopplung von Simulationssoftware und Steuerungssoftware

Beim Test der Steuerungssoftware mit einem Simulationssystem muß eine Kopplung zwischen beiden Systemen realisiert werden. Für diese Kopplungen gibt es mehrere Möglichkeiten, die von der Entwicklungsumgebung der Steuerungssoftware und dem verwendeten Simulationssystem abhängen. Im folgenden werden zunächst die Kopplungsmöglichkeiten aus der Sicht der Steuerungssoftware beschrieben.

In /97/ werden in einem Leitsystem, das an verschiedene Anlagenkonfigurationen adaptierbar ist, drei Funktionsebenen unterschieden. Es handelt sich dabei um **Kernfunktionen**, die ohne Veränderungen an den Funktionen selbst, sondern nur mit Hilfe unterschiedlicher Konfigurationsdaten für verschieden Anlagen genutzt werden können, **Anpassungsfunktionen**, die die Kernfunktionen an das Anforderungsprofil der spezifischen Anwendung angleichen, sowie **Kommunikationsfunktionen**, die die Schnittstelle zwischen dem Leitsystem und der unterlagerten Steue-

rungsebene herstellen und für den Nachrichtenaustausch innerhalb des Leitsystems sorgen (Interprozeßkommunikation). An Hand dieser Funktionsebenen werden verschiedene Einsatzebenen für Testumgebungen mit jeweils bestimmten Anforderungen an den Simulator definiert.

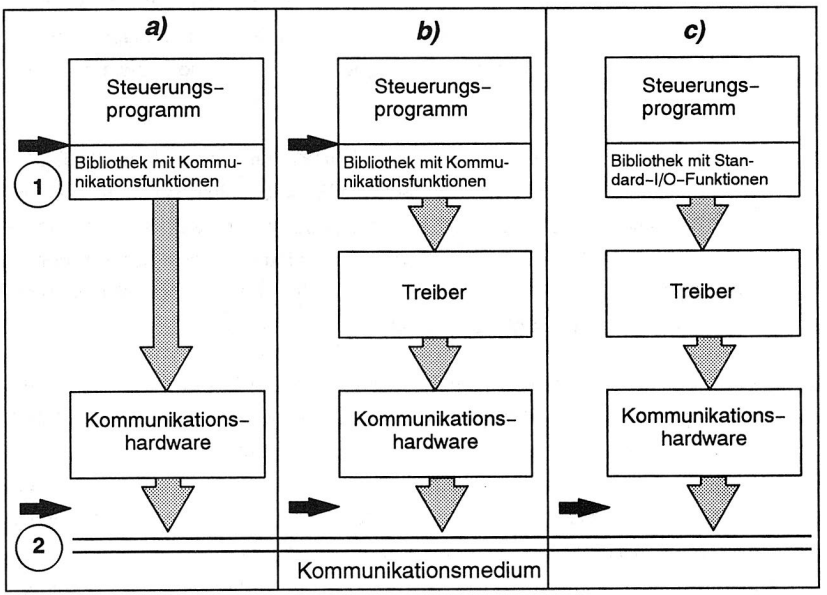


Bild 33: Zugriffsmöglichkeiten auf Kommunikationsschnittstellen.

Betrachtet man die Kopplung an die Steuerungssoftware aus der Sicht der Kommunikationschnittstelle, ergeben sich die drei in Bild 33 dargestellten Möglichkeiten des Zugriffs.

Bei der Variante a) werden im Steuerungsprogramm spezielle Funktionen für das Kommunikationssystem aufgerufen. Der Objekt-Code der Funktionen wird über eine Bibliothek zum ausführbaren Programm gebunden. Die Funktionen greifen direkt auf die Kommunikationshardware (spezielle Baugruppen bwz. Einsteckkarten) zu.

Die Variante b) entspricht der Variante a) mit dem Unterschied, daß die Funktionen der Bibliothek nicht direkt auf die Hardware zugreifen, sondern auf den Treiber.

Bei der Variante c) werden Standard-I/O-Funktionen des Betriebssystems verwendet, um über einen Treiber auf die Hardware zuzugreifen.

Fehler bei der Kommunikation können auftreten durch Inkompatibilitäten zwischen Funktionsbibliotheken und Treibern, Treibern und Hardware, Funktionsbibliotheken und Betriebssystem-Version, Treiber und Betriebssystem-Version sowie durch falsche Parametrierungen von Hardware und/oder Software.

In Bild 33 sind durch die schwarzen Pfeile mögliche Kopplungspunkte an die Steuerungssoftware angedeutet. Im ersten Fall wird die Funktionsbibliothek durch eine Bibliothek mit identisch aufrufbaren Funktionen ersetzt, im zweiten Fall wird das Simulationssystem direkt mit der anlagenseitigen Schnittstelle des Steuerrechners verbunden. Der Vorteil der zweiten Lösung liegt darin, daß der komplette Kommunikationsmechanismus getestet werden kann.

Wird auf dem Steuerrechner ein Multiprozessing-Betriebsystem verwendet, besteht die Steuerungssoftware meist aus mehreren miteinander kooperierenden Prozessen, die gegenseitig Daten austauschen. Zur Vermeidung von Zugriffskonflikten werden die Kommunikationsfunktionen meist in Server-Prozessen zusammengefaßt. Der Datenaustausch zwischen den Serverprozessen und den anderen an der Steuerung beteiligten Prozessen wird über Interprozeßkommunikation innerhalb eines Rechners oder über Rechnergrenzen hinweg realisiert. Unter dem Betriebssystem UNIX können hierzu Pipes, Shared Memory, Semaphoren und Sockets verwendet werden. Daneben ist auch ein Datenaustausch über Dateien oder Datenbanken möglich. In einer solchen Multiprozessing-Umgebung können neben den in Bild 33 aufgezeigten Möglichkeiten, an der Steuerung beteiligte Prozesse durch Simulationsprozesse ausgetauscht werden.

4.6 Einsatzmöglichkeiten bestehender Simulationssysteme

Im Gegensatz zu der Vielzahl an Simulationswerkzeugen für den Aufbau von Ablaufsimulationsmodellen, die sich für Materialflußuntersuchungen eignen, gibt es keine Werkzeuge mit einem vergleichbaren Entwicklungsstand für den Aufbau von Model-

len, die sich für den Test von Steuerungssoftware auf den der Maschinenebene übergeordneten Ebenen eignen /2/. Dies wird auch deutlich, wenn man die im "Marktspiegel Simulationstechnik in Produktion und Logistik" /76/ vorgestellten Simulationssysteme betrachtet. Die Anwendungsfelder der vorgestellten Simulatoren sind unterteilt in

Beliebige Anwendungen
Die Anwendung ist nicht nur auf den Bereich Produktion und Logistik beschränkt, sondern steht auch für allgemeine (beliebige) technische und wirtschaftliche Problemstellungen zur Verfügung.

Bereichsbezogene Anwendungen
Simulatoren für Produktion, Materialfluß und Distribution, unter Umständen mit einer Ausrichtung auf Werkstattsteuerung

Spezialanwendungen
Simulatoren für Fahrerlose Transportsysteme und Staplersysteme, Flexible Fertigungssysteme, Lagersysteme, Montagearbeitsplätze und Montagesysteme, Werkzeugmaschinen, Robotersysteme, Werkstattsteuerung sowie Speicherprogrammierbare Steuerungen.

Eine Analyse der vorgestellten Simulationssysteme ergab, daß keines der Simulationssysteme den speziellen Anforderungen genügte, die durch die Anwendungsumgebung gestellt wurde. Daher war es notwendig, die Entwicklung des im folgenden Kapitel vorgestellten Systems HSSIM auf Programmiersprachenebene durchzuführen.

5. Das Simulationsprogramm HSSIM

Die in Kapitel 3 erarbeiteten Anforderungen wurden exemplarisch in ein Simulationssystem umgesetzt, mit dem die Entwicklung der Linienrechnersoftware für Bestückautomaten des Typs HS180, SP120 und Siplace (Hersteller: Siemens) in der Testphase unterstützt wird. Das System wird auch in der Trainingsphase während der Schulung an diesen Bestücksystemen eingesetzt. Zunächst werden die speziellen Charakteristika der Anwendungsumgebung vorgestellt. Daraus werden die Merkmale des Simulationssystems abgeleitet und im Anschluß näher beleuchtet.

5.1 Anwendungsumgebung

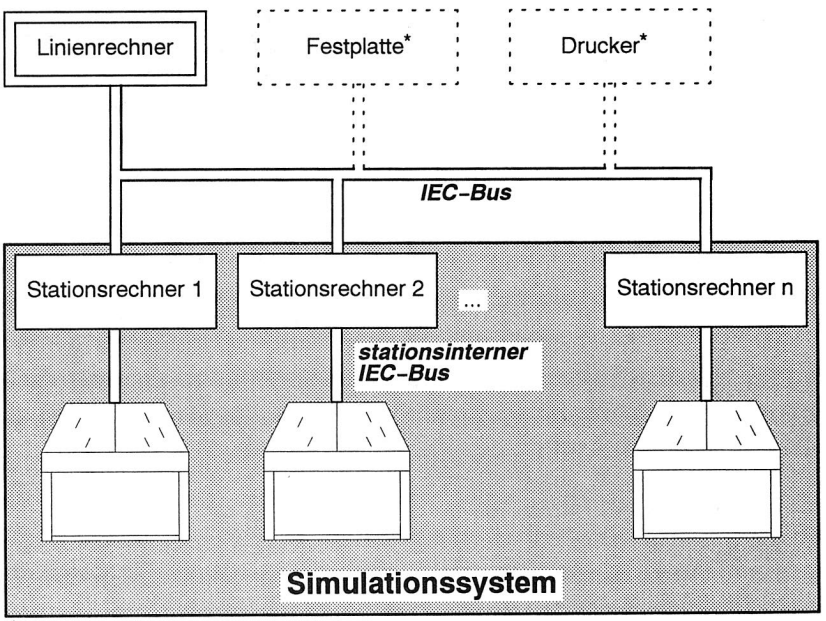
5.1.1 Einordnung der Bestückautomaten

Bei dem betrachteten Universalbestückautomaten HS180 handelt es sich um einen Pick-and-Place-Automaten mit bewegtem Bestückkopf, ortsfesten Abholspuren und ortsfester Leiterplatte, der durch Integration der in Kapitel 2 vorgestellten Zusatzgeräte das komplette Spektrum an verfügbaren Gehäuseformen bestücken kann. Bestückstationen des Typs HS180 lassen sich mit Dispensern und Stationen des Typs SP120 beliebig zu Bestücklinien kombinieren. Alle drei Typen von Automaten haben einen gemeinsamen Grundaufbau (Maschinenständer, Transportbänder und XY-Positionierachsen). Im Gegensatz zur HS180 verfügt der SP120 über einen Bestückkopf, der als 12-fach-Revolver ausgeführt ist. Als Chip-Shooter bietet er eine hohe Bestückleistung, die Größe der bestückbaren Bauelemente ist aber begrenzt. Der Dispenser verfügt statt des Bestückkopfes über zwei Kleber- bzw. Lotpastenauftragsköpfe, die Förderbereiche entfallen bei diesem Automaten.

Die HS180 kann auch mit einer neuen Generation von Bestückautomaten (SIPLACE) kombiniert werden. Diese Automaten mit ortsfester Leiterplatte und ortsfesten Abholspuren verfügen über je zwei Bestückköpfe, die entweder als Pick-and-Place-Kopf oder als 12-fach-Revolver ausgeführt sind. Zur Vermeidung von Kollisionen wechseln sich die beiden Bestückköpfe beim Abholen und Bestücken von Bauelementen ab. Ziel der Rüst- und Umrüstoptimierung muß es bei diesen Automaten sein, beide Bestückköpfe gleichmäßig auszulasten.

5.1.2 Charakteristika der Steuerungsstruktur

Die Steuerungshierarchie der betrachteten Bestücklinien entspricht der in Kapitel 2 vorgestellten Struktur. Ein der Bestücklinie übergeordneter Linienrechner ist zum Datenaustausch über einen ***IEC-Bus*** (genormt nach IEEE-488.1, bzw. DIN IEC66.22) mit den Stationsrechnern der Bestückautomaten verbunden. Als Linien- und Stationsrechner werden Rechner des Typs HP-300 eingesetzt, die mit einem Singletasking-BASIC-Betriebssystem ausgestattet sind. Parallel zu den SIPLACE-Bestückern wird eine neue Linienrechnergeneration entwickelt, die auf einem PC mit UNIX-Betriebssysstem basiert. Das zur Steuerung eingesetzte Bussystem erleichtert die Erweiterung einer Bestücklinie um zusätzliche Bestückautomaten. Innerhalb der Bestückautomaten sorgt ein stationsinterner IEC-Bus für die Kommunikation zwischen dem Stationsrechner und den Achssteuerungen.



*nur bei der alten Linienrechnergeneration

Bild 34: Steuerungstruktur der betrachteten Bestücklinien und Umfang des Simulationssystems

5.1.3 Beschreibung des Telegrammverkehrs

Zur Steuerung der Bestücklinien werden zwischen dem Linienrechner und den Bestückautomaten Datentelegramme in einer durch ein Protokoll vorgegebenen Reihenfolge ausgetauscht.

Die Telegramme bestehen aus Telegrammköpfen, durch die der Telegrammtyp festgelegt wird (Bild 35), sowie Nutzdaten. Aufbau und Länge der Nutzdaten sind vom Telegrammtyp und von den Nutzdaten selbst abhängig. Durch Flags innerhalb der Telegramme wird das Vorhandensein von optionalen Telegrammteilen und die Länge von Feldern angezeigt.

Telegrammtypen vom Linienrechner		***Telegrammtypen der Bestückautomaten***	
LART:	Übertragen der Stations-Art	SART:	Anfordern der Stations-Art
LMAD:	Übertragen von Maschinendaten	SMAD:	Anfordern der Maschinendaten
LBSD:	Übertragen von Rüstungsdaten	SBSD:	Anfordern der Rüstungsdaten
LNUD:	Übertragen von Nutzendaten	SNUD:	Anfordern der Nutzendaten
LSTA:	Bestücken Starten	SSTA:	Anfordern "Bestücken Starten"
LSTP:	Bestücken Stoppen	SBES:	Zustand Bestücken erreicht
LBCE:	Barcode-Leser einschalten	SEND:	Bestücken Ende
LBCA:	Barcode-Leser ausschalten	SABR:	Bestücken Abbruch
LBRV:	Transportbandbreite verstellen	SBDE:	Übertragen eines Barcode
LBDE:	BDE-Daten Anfordern	SBRV:	Quittung der Breitenverstellung
LSSO:	Eingabeschnittstelle öffnen	SBDE:	Übertragen der Betriebsdaten
LSSP:	Eingabeschnittstelle schliessen	SFHL:	Übertragen einer Fehlernummer
LPAS:	Einstellen der Benutzerklasse	SVSE:	Anzeigen des Übergangs in den Teach-Modus

Bild 35: Telegrammtypen von Linien- und Stationsrechner bei den betrachteten Bestückautomaten

Telegramme vom Linienrechner dienen

- zum Übertragen von Daten an die Bestückautomaten (Maschinendaten, Rüstdaten, Nutzendaten)
- zum Übertragen von Befehlen ("Bestücken starten", "Bestücken stoppen") oder
- zum Anfordern von Daten wie z. B. BDE-Daten oder Fehlermeldungen

Telegramme von den Stationen

- rufen Daten vom Linienrechner ab
- zeigen einen Zustandswechsel an, der durch interne Abläufe oder Benutzerinteraktionen ausgelöst wurde ("Bestücken Ende", "Quittung der Breitenverstellung") oder
- enthalten angeforderte Daten

Das Empfangen von Telegrammen bewirkt in den Bestückautomaten:

- das Ändern des aktuellen Zustands
- das Quittieren mit einem Antworttelegramm, ohne Zustandswechsel oder
- das Ausführen interner Abläufe, ohne Zustandswechsel, wie z. B. Barcode-Leser ein- bzw. ausschalten, Eingabeband freigeben oder sperren.

Das Senden von Telegrammen bewirkt in den Bestückautomaten

- das Ändern des aktuellen Zustands
- das Warten auf ein Antworttelegramm vom Linienrechner, ohne Zustandswechsel

5.2 Merkmale des Simulationssystems

Mit dem Simulationssystem werden ganzheitliche Test der Linienrechnersoftware unterstützt, also Tests, die auch die Anlagenkommunikation umfassen. Dazu ersetzt das System während der Testphasen eine reale, aus einem oder mehreren Bestückautomaten bestehende Bestücklinie. Das System hat dabei folgende Aufgabe:

- Es dient als Kommunikationspartner
- Es dient zur Überprüfung des Telegrammverkehrs auf Fehler
- Es dient zu Überprüfung der Robustheit der Software in Extremsituationen

Das System zeigt gegenüber dem Linienrechner das gleiche Verhalten wie die realen Bestückautomaten. Dieses Verhalten ist durch das Kommunikationsprotokoll festge-

legt, also durch die Reihenfolge, in der Telegramme zwischen Linienrechner und Bestückautomaten ausgetauscht werden.

In Bild 36 sind die wesentlichen Merkmale des Simulationssystems aufgeführt. Auf diese Merkmale wird im folgenden noch genauer eingegangen. Zunächst wird in Kapitel 5.3 die Struktur des IEC-Busses und die Anschlußmöglichkeiten an diesen Bus vorgestellt. Anschließend wird auf die Objektorientierte Entwicklung und die zur Dokumentation des Systems verwendete Notationsweise eingegangen. Die erforderliche Echtzeitfähigkeit des Systems wurde durch den Einsatz eines Multitasking-Kerns erreicht. Zur Abbildung des Verhaltens der Bestückautomaten wurde das Konzept der abstrakten Automaten verwandt, das in Verbindung mit der entwickelten Modellierungssprache eine einfache Modellerstellung erlaubt.

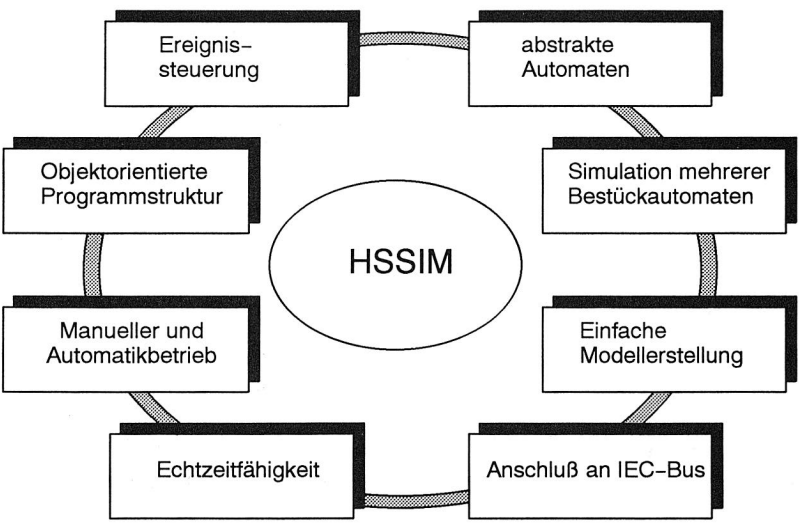


Bild 36: Merkmale des Simulationssystems

5.3 Anbindung an den IEC-Bus

5.3.1 Der IEC-Bus

Der IEC-Bus, auch HPIB, GPIB oder IEEE-488-Bus genannt, wurde als Interface-Bussystem zur Verbindung von Meß- und Prüfsystemen mit Steuerrechnern entwikkelt /11, 48, 17, 80/. 1975 wurde der Bus unter der Bezeichnung IEEE-488.1 als amerikanische Industrienorm dokumentiert und 1977 von der Internationalen Elektrotechnischen Kommission (IEC) unter der Bezeichnung IEC 625-1 als internationaler Standard übernommen. 1987 erschien die ergänzende IEEE-488.2-Norm. Während IEEE-488.1 sich hauptsächlich auf die Normung der Hardware beschränkte, werden in IEEE-488.2 auch Datenformate, Statusmeldungen und besondere Funktionen zur Fehlerbehandlung und zur Initialisierung des Systems festgelegt. Der Bus gilt auch heute noch als Standard im Bereich der professionellen Meß- und Prüftechnik.

Der IEC-Bus ist ein asynchroner, bidirektionaler, bitparalleler und byteserieller Bus. Maximal 15 Geräte können zusammengeschaltet werden. Die maximalen Leitungslängen sind auf ca. 20 Meter begrenzt. Die angeschlossenen Geräte können folgende Funktionen haben:

- **Talker**, zum Senden von Daten
- **Listener**, zum Empfangen von Daten
- **Controller**, zum Steuern des Busses.

Einzelne Geräte können mehrere Funktionen in sich vereinigen. So kann z. B. der Controller auch als Talker oder Listener fungieren. Jedes Gerät, das an den Bus angeschlossen ist, verfügt über eine eindeutige Adresse, über die es angesprochen wird.

Über den Bus können zwei Arten von Nachrichten übertragen werden:

- Schnittstellennachrichten
  Dabei handelt es sich um Kommandos, die zur Initialisierung des Busses, oder zur Adressierung von Geräten dienen.
- Nutzdaten
  Das können Befehle oder Meßergebnisse sein, oder wie im betrachteten Beispiel die Daten zur Steuerung der Bestückautomaten.

Der Controller legt durch Übertragung von Talker- und Listener-Adressen die Kommunikationspartner fest. Es kann zu einer Zeit nur einen Talker, durchaus aber mehrere Listener geben. Geräte, die Daten übertragen wollen, zeigen dies durch Hochsetzen der SRQ (Service-Request)-Leitung an. Der Controller führt zur Ermittlung des Gerätes mit Übertragungswunsch ein paralleles oder ein serielles Polling durch.

Die Übertragung von Schnittstellennachrichten und Nutzdaten wird durch die ATN-Leitung unterschieden, die nur der Controller bedienen darf. Zur Anzeige des Übertragungsendes von Nutzdaten gibt es zwei verschiedene Möglichkeiten. Zum einen kann mit dem letzten übertragenen Byte die EOI-Leitung angehoben werden, zum anderen kann das Datenende durch ein spezielles Ende-Byte angezeigt werden. Diese Möglichkeit kann allerdings nur beim Übertragen von ASCII-Daten verwendet werden.

Die Datenübertragung auf dem IEC-Bus wird durch drei Handshake-Leitungen geregelt, die vom Sprecher und den Hörern bedient werden. Die Übertragungsgeschwindigkeit wird durch das langsamste der an der Kommunikation beteiligten Geräte bestimmt.

5.3.2 Anschluß von PC's an den IEC-Bus

Zum Anschluß von PC's an den IEC-Bus werden von verschiedenen Firmen Einsteckkarten mit zugehöriger Software vertrieben /16, 74/. Die Software besteht aus Treibern, Testprogrammen und Programmierschnittstellen für alle auf PC's gängigen Programmiersprachen (C, Pascal, Fortran). Darüberhinaus gibt es spezielle Analyzerkarten, mit denen der Datenverkehr auf dem Bus überwacht werden kann.

Mit Hilfe der Einsteckkarten kann ein PC als IEC-Bus-Controller, Talker oder Listener betrieben werden. Über Konfigurationsprogramme oder über die Programmierschnittstelle werden verschiedene IEC-Bus-spezifische Parameter eingestellt. (Adresse des Gerätes, Timeout, Ende-Kennung bei Datenübertragung, System-Controller ja/nein, usw).

Für das entwickelte System ergaben sich folgende Anforderungen:

- Der PC wird nicht als Controller eingesetzt. Diese Aufgabe hat der Linienrechner.
- Im Simulationsbetrieb soll der PC alle Bestückautomaten einer Bestücklinie simulieren. Jeder Automat hat eine eigene IEC-Bus-Adresse.

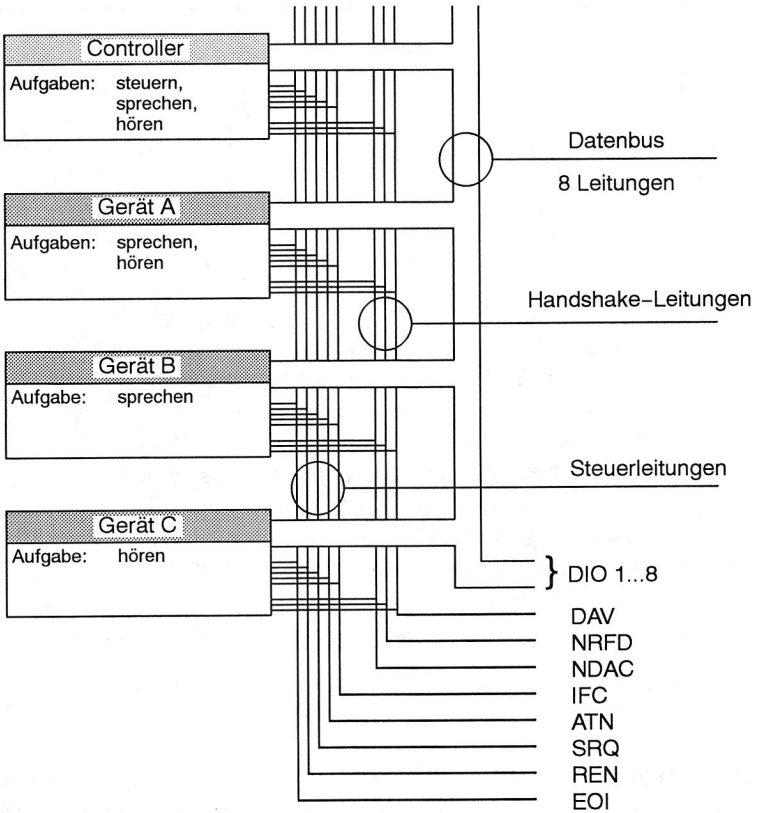
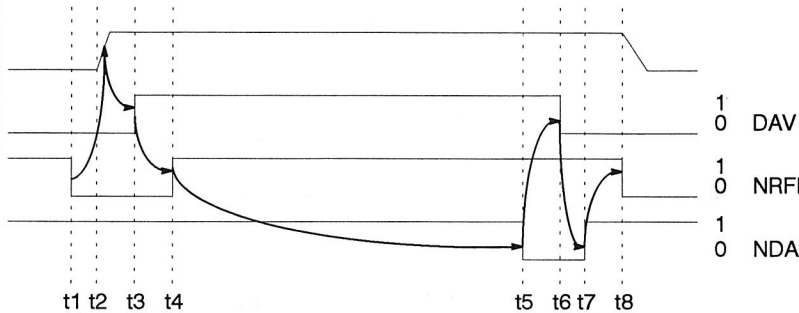


Bild 37: Struktur des IEC-Busses

Diese Forderungen lassen sich im Normalbetrieb mit fest eingestellter IEC-Bus-Adresse nicht erfüllen. Bei der verwendeten IEC-Bus-Karte gibt es aber neben dem Normal-Modus einen Mehradress-Modus. In diesem Modus kann die Karte flexibel auf eine beliebige Adresse eingestellt werden, mit der ein Datentransfer durchgeführt werden soll.

Das Umschalten in den Mehradress-Modus geschieht über die Programmierschnittstelle. Dabei wird dem Treiber die Adresse einer Funktion mitgeteilt, die über Interrupt immer dann aufgerufen wird, wenn über den Bus eine Hörer- oder Sprecher-Adresse übertragen wird. Innerhalb der Funktion wird an Hand der übergebenen



t1: NRFD auf 0: Empfänger können Daten übernehmen
t2: NDAC auf 1 Geräte übernehmen keine Daten
t3: DAV auf HIGH: alle Daten sind ungültig
t4: NDAC auf 0: alle Geräte haben Daten vom Bus gelesen
t5: NRFD auf 1: Geräte nehmen Daten auf
t6: DAV auf 1: Daten sind gültig
t7: Daten werden auf den Bus gelegt
t8: NRFD auf 0: Empfänger bereit

Bild 38: Datenübertragung auf dem IEC-Bus durch Handshake

Parameter (Adressierungsart, Adresse) überprüft, ob auf die übertragene Adresse reagiert werden soll. Wenn ja, liefert die Interrupt-Funktion einen entsprechenden Rückgabewert an den Treiber. Die Karte verhält sich nun so, als ob diese Adresse fest eingestellt ist. Die Reaktion auf die Adressierung ist abhängig von dem Adressier-Modus. Bei einer Adressierung als Hörer sind Daten vom Bus einzulesen, bei einem Polling wird als Antwort ein *Serial-Poll-Byte* übertragen und bei einer Adressierung als Sprecher müssen Daten gesendet werden.

Die Funktion, die durch den Interrupt aufgerufen wird, muß folgende Bedingungen erfüllen /74/:

- Die Funktion muß so schnell wie möglich beendet sein, um nicht andere Interrupt-Aktivitäten zu behindern. Größere Berechnungen sollten also vermieden werden.

- Es dürfen keine Funktionen aufgerufen werden, die nicht wiedereintrittssfähig ("non-reentrant") sind, wie z. B. DOS-Funktionen, BIOS-Funktionen oder IEC-Bus-Funktionen.
- Der benötigte Platz für den Funktions-Stack sollte so klein wie möglich sein.

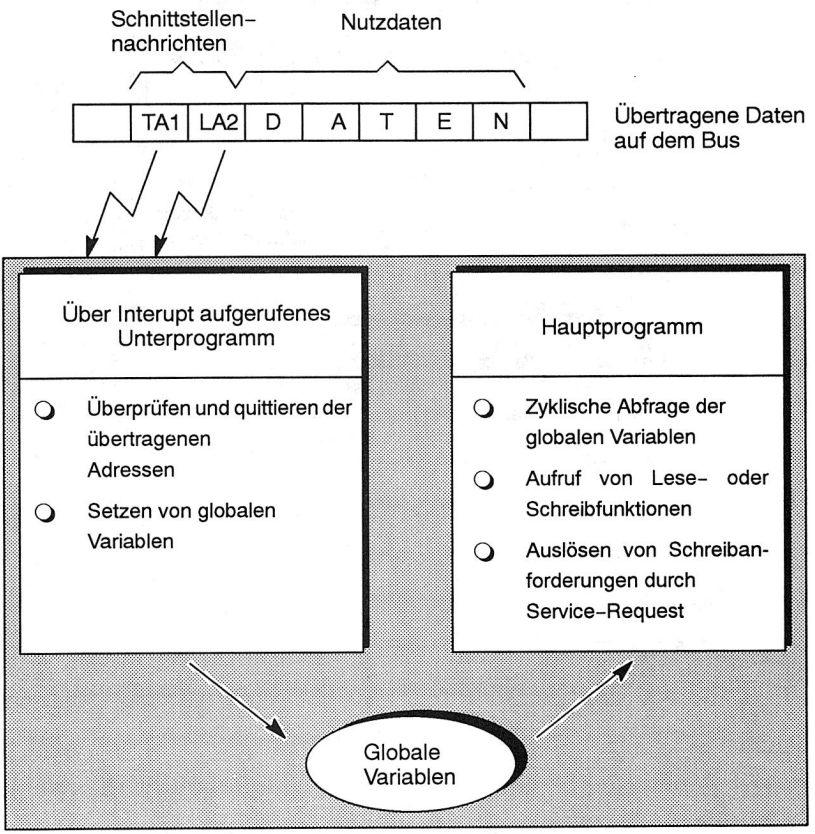


Bild 39: Adressierung im Mehradress-Modus

Da innerhalb der Interrupt-Funktion keine IEC-Bus-Funktionen aufgerufen werden dürfen, muß die Reaktion auf die Adressierung, d. h. das Aufrufen von Lese- oder Schreibfunktionen, außerhalb der Interrupt-Funktion erfolgen. Dazu werden bei er-

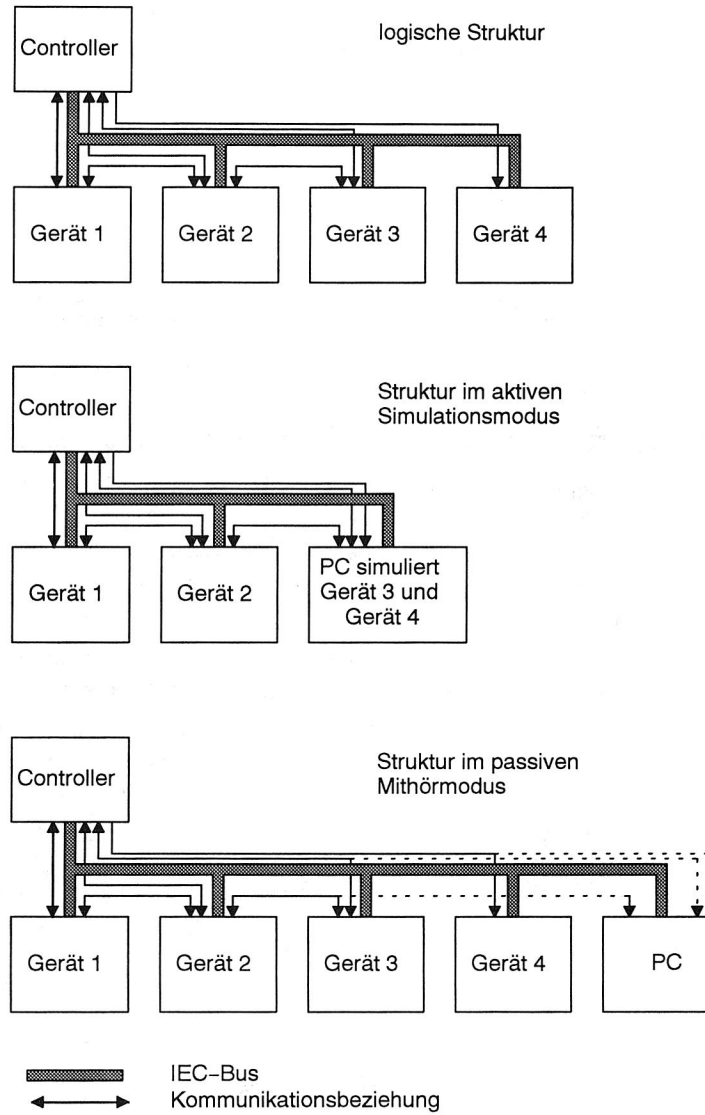


Bild 40: Busstruktur im Simulationsmodus und im Mithörmodus

folgter Adressierung globale Variable gesetzt. Diese Variablen müssen im Hauptprogramm zyklisch überprüft und die erforderlichen Lese- und Schreibfunktionen ausgelöst werden.

Durch die Mehradress-Funktionalität lassen sich zwei verschiedene Modi realisieren (Bild 40):

- **aktiver Simulationsmodus**
  Im aktiven Simulationsmodus werden mehrere nicht vorhandene Geräte durch den PC simuliert. Das bedeutet, daß alle Daten, die an eines der simulierten Geräte übertragen werden, vom PC eingelesen werden und daß alle Daten, die von einem der simulierten Geräte übertragen werden, vom PC versendet werden. Dabei sind keine Datenübertragungen von einem simulierten Gerät zu einem simulierten Gerät möglich.

- **passiver Mithörmodus**
  Im passiven Mithörmodus findet die Datenübertragung auf dem Bus zwischen den realen Geräten statt. Der PC ist zusätzlich an den Bus angeschlossen und kann den Datenverkehr zwischen ausgewählten Kommunikationspartnern aufzeichnen. Während einer Datenübertragung befinden sich zwei Geräte mit der selben Adresse auf dem Bus, nämlich der reale Empfänger und der PC. Durch den Handshake-Mechanismus des IEC-Busses (Bild 38) ist dennoch eine sichere Datenübertragung garantiert.

5.4 Einsatz eines Multitaskingkerns

Nach der Adressierung der Kommunikationspartner auf dem IEC-Bus muß innerhalb eines vorgegebenen Time-Out-Intervalls die Datenübertragung erfolgen. Im Fall des oben beschriebenen Mehrfachadress-Modus bedeutet dies, daß außerhalb der Interruptfunktion, in der die Adressen überprüft werden, die globalen Variablen, mit denen eine durchzuführende Datenübertragung angezeigt wird, innerhalb des vorgegebenen Time-Out-Intervalls abgefragt und die entsprechenden Schreib-/Lesefunktionen ausgelöst werden müssen. Da das verwendete Betriebssystem MS-DOS ein Singletasking-Betriebssystem ist und Programme, die den Mehradress-Modus verwenden, auch Benutzerinteraktionen ermöglichen sollen, kann eine entsprechende Reaktionszeit nur durch den Einsatz eines Multitaskingkerns garantiert werden /104,60/.

Beschleunigungs-faktor	Timer-Interrupts pro Sekunde	Länge der Zeitscheiben in ms
1	18,2	54,9
2	36,4	27,5
4	72,8	13,7
8	145,6	6,9
16	291,2	3,4
32	582,4	1,7
64	1164,8	0,9
128	2329,6	0,4
256	4659,2	0,2

Bild 41: Zeitscheibenlängen bei Einsatz eines Multitaskingkerns unter DOS

Ein Multitaskingkern besteht im wesentlichen aus einem Scheduler, der über den Timer-Interrupt des PC's aktiviert wird und zwischen den einzelnen Tasks, die durch Funktionen innerhalb des Programms realisiert sind, umschalten kann /105/. Der Timer-Interrupt wird alle 18,2 mal pro Sekunde aktiviert /106/, was einem Abstand zwi-

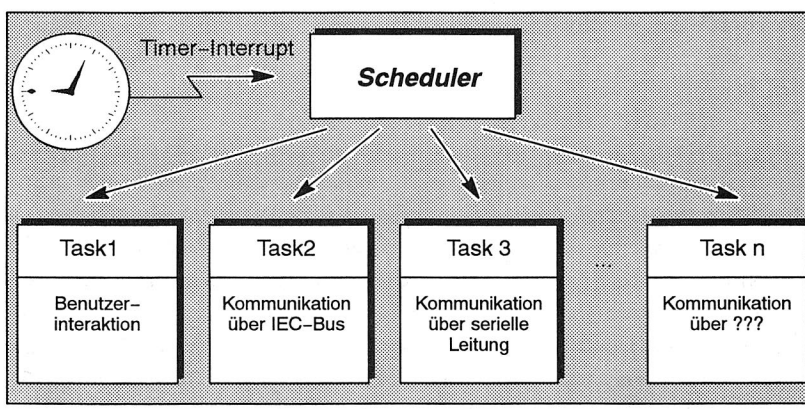


Bild 42: Struktur eines Multitaskingprogramms mit echtzeitfähiger Kommunikationsanbindung unter DOS

schen zwei Taskwechseln von 54,9 ms entspricht. Dieser Abstand kann in Zweierpotenzen verkürzt werden bis in den Bereich von mehreren hundert µs (siehe Bild 41) . Die Grenze ist abhängig von der Geschwindigkeit des eingesetzten Rechners. Werden die Zeitscheiben zu klein, wird die Belastung des Rechners durch den Scheduler zu groß und die Zeit für die einzelnen Tasks zu klein. Bei einem PC mit einem 80486-Mikroprozessor sind Zeitscheiben von 0,2ms noch durchaus möglich.

Durch Zuteilen von Prioritäten können besonders zeitkritische Tasks bei der Zuteilung von Zeitscheiben bevorzugt werden. Zur Synchronisation und zur Kommunikation zwischen den Tasks werden Semaphoren, Flags, Pipes, Buffer und Mailboxen verwendet.

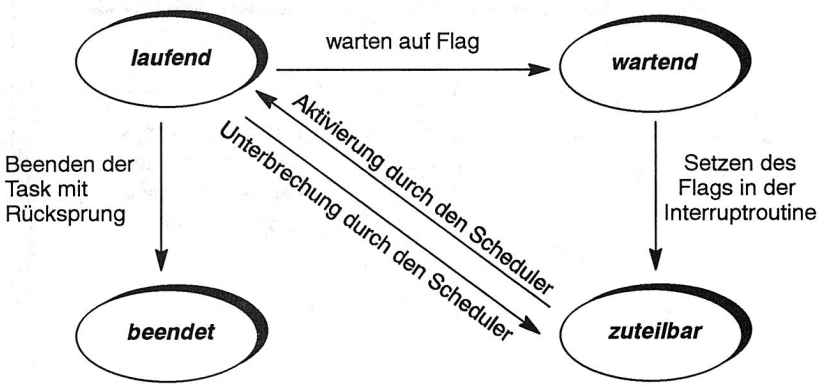


Bild 43: Zustände und Zustandswechsel der Kommunikations-Task

Ein Programm, das neben Benutzerinteraktionen über Menüs und Editoren auch Kommunikationsvorgänge ermöglichen soll, kann durch Aufteilung in eine Vordergrundtask, die das Menüsystem betreibt, und eine odere mehrere Kommunikationstasks die notwendigen Echtzeitbedingungen erreichen. Eine derartige Aufteilung hat darüberhinaus den Vorteil, daß nur eine Task Bildschirmausgaben durchführt und somit keine Konflikte beim Zugriff auf den Bildschirm entstehen können.

In Bild 43 und Bild 44 ist die Umsetzung dieses Schemas im Simulationsprogramm HSSIM dargestellt. Task2 ist für die Kommunikationsvorgänge mit dem IEC-Bus zuständig. Diese Task versetzt sich selbst in den Zustand "wartend", bis in der Interruptroutine nach erfolgreicher Adressierung das entsprechende Flag gesetzt wird. Da-

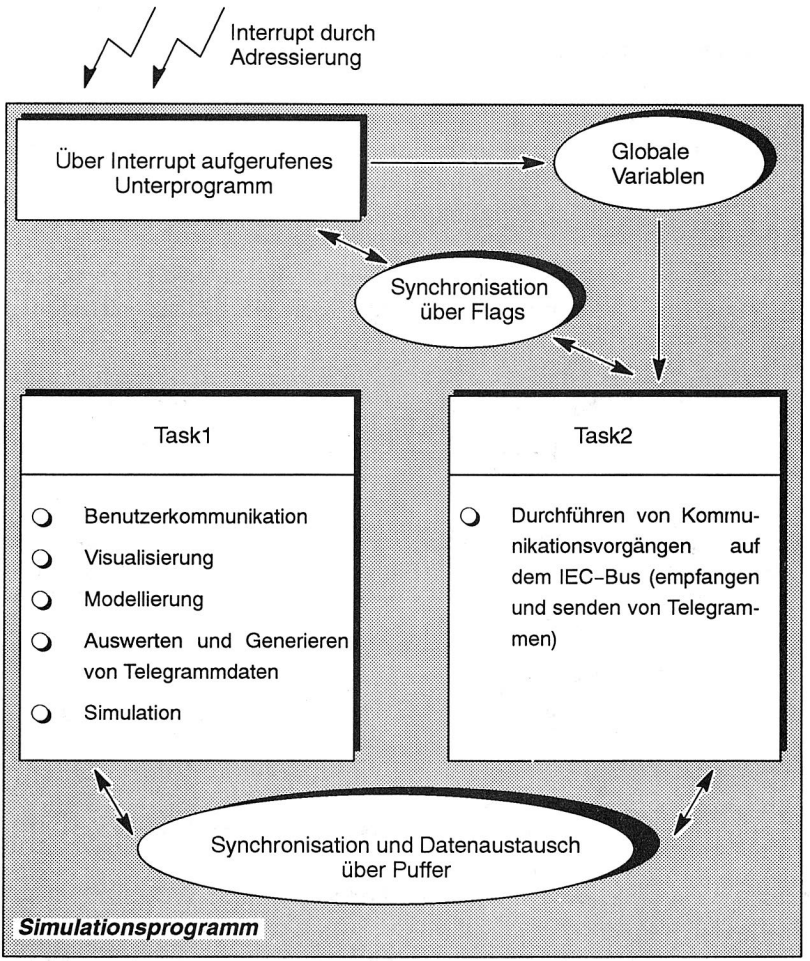


Bild 44: Echtzeitfähigkeit durch Einsatz ein Multitasking-Kerns

durch geht sie in den Zustand "zuteilbar" über und kann bei der nächsten Aktivierung durch den Scheduler das Empfangen oder Versenden von Telegrammen aktivieren. Nach der abgeschlossenen Datenübertragung werden die relevanten Kommunikationsparameter über einen Puffer an die Task1 übergeben, ausgewertet und entsprechende Reaktionen, wie z. B. die Benachrichtigung des Benutzers, angestoßen.

5.5 Objektorientierte Programmstruktur

Bei der Entwicklung eines Simulationssystems auf programmiersprachlicher Ebene hat man die Wahl zwischen anweisungsorientierten, objektorientierten und blockorientierten Programmiersprachen (Bild 28) /25, 76,107 /. Die Anbindung an Kommunikationssysteme, die bei Simulationssystemen für den Test von Steuerungssoftware und für Schulungen notwendig ist, läßt sich vornehmlich mit anweisungsorientierten und objektorientierten Programmiersprachen durchführen (Pascal, Fortran, C, C++), da für diese Sprachen meist entsprechende Programmierschnittstellen zur Verfügung stehen. Bei der anweisungsorientierten Programmierung erfolgt die Modellierung durch Abbildung von Vorgängen (bestücken, Leiterplatte transportieren, Bauelement zentrieren) auf Funktionen. Beim objektorientierten Ansatz hingegen werden die in der Realität vorhandenen Objekte (Bestückautomat, Leiterplatte, Bauelemente) mit ihren spezifischen Eigenschaften und Fähigkeiten auf entsprechende programmiersprachliche Objekte abgebildet. Wie auch in der Realität wird das Verhalten des Gesamtsystems durch die Zusammenarbeit der Objekte bestimmt /2/.

5.5.1 Merkmale objektorientierter Programmierung

Die Objekt-orientierte Programmierung bietet gegenüber der herkömmlichen funktionalen Programmierung mehrere Vorteile, die bereits in vielen Arbeiten herausgestellt wurden /63, 77, 89, 103/. Im Fertigungsbereich wurden objektorientierte Ansätze vorwiegend im Steuerungsbereich /19, 57, 59, 75, 86, 93/ und zur Simulation eingesetzt /1, 36, 111/.

Durch die Paradigmen der Objekt-orientierten Programmierung wie Vererbung und Polymorphismus ergeben sich neue Möglichkeiten der modularen Softwareentwicklung. Als Hauptvorteil wird die Wiederverwendbarkeit einmal geschriebener Software genannt.

Objekt-Orientierte Programmierung zeichnet sich durch folgende wesentliche Eigenschaften aus /69/:

abstrakte Datentypen
Mit Hilfe von abstrakten Datentypen wird die Zuverlässigkeit und Änderbarkeit von Software-Systemen erhöht, indem die Abhängigkeiten zwischen den einzelnen

Komponenten reduziert werden. Ein abstrakter Datentyp besteht aus einer internen Repräsentation und einer Menge von Prozeduren, mit denen auf die Daten zugegriffen werden kann. Abstrakte Datentypen, wie sie z. B. in Modula-2 vorkommen, haben den Nachteil, daß sie nur auf einem ganz bestimmten Datentyp operieren können (z. B. Stack von Integers).

dynamisches Binden
Dynamisches Binden erleichtert die Verarbeitung von heterogenen Objekten, die eine gemeinsame Menge von Nachrichten verstehen. Jedes Objekt kann die Nachricht entsprechend seiner Klassenzugehörigkeit interpretieren und darauf reagieren. Dieser Mechanismus wird als Polymorphismus bezeichnet. Voraussetzung für Polymorphismus ist, daß die Adressen der aufgerufenen Prozeduren erst während der Laufzeit und nicht bereits beim Compilieren ermittelt werden.

Vererbung
Die Vererbung ermöglicht die Wiederverwendung von bereits geschriebener Software,durch Erzeugen von Klassen, die Spezialisierungen von bereits existierenden Klassen sind. Die so erzeugten Unterklassen erben alle Eigenschaften der Oberklassen. Die geerben Eigenschaften können durch Überschreiben der Klassen-Methoden verändert oder erweitert werden.

Insbesondere generische Datentypen und Objekt-Klassen-Bibliotheken zur Verwaltung abstrakter Datentypen (Listen, Sets, Stacks, Queues, Binary Trees) reduzieren den Programmieraufwand zur Datenverwaltung erheblich. Durch den Polymorphismus lassen sich Objekte verschiedener Klassen (die von einer gemeinsamen Basisklasse abgeleitet sein müssen) in eine gemeinsame übergeordnete Datenstruktur, wie z. B. eine Liste, einfügen. Durch die Ableitung von einer gemeinsamen Basisklasse verstehen alle diese Objekte eine gemeinsame Menge von Nachrichten, auf die sie aber je nach Klassenzugehörigkeit unterschiedlich reagieren.

5.5.2 Dokumentation objektorientierter Programme

Bei der herkömmlichen funktionalen Programmierung werden verschiedene Beschreibungsmethoden eingesetzt, um den Aufbau und den Ablauf eines Programms zu beschreiben (SADT, Flußdiagramme, Nassi-Schneidermann). Für die Objektorientierte Programmierung können diese Beschreibungsmethoden nur bedingt eingesetzt werden, z. B. um den Ablauf einer Methode einer Klasse zu beschreiben. Zur Beschreibung der gesamten Programmstruktur, die sich durch Beziehungen zwi-

schen Klassen, Objekten und Modulen ergibt, eignet sich die von Grady Booch /6/ entwickelte Notationsweise.

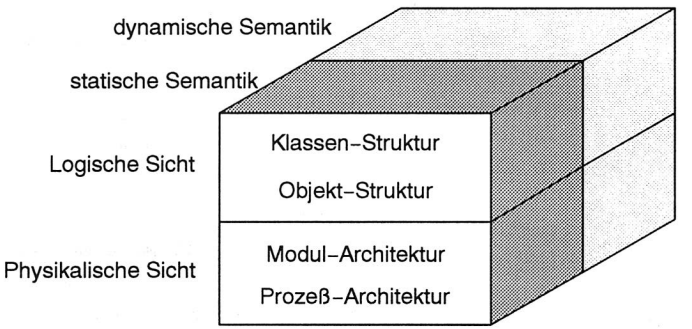


Bild 45: Die Modelle der objekt-orientierten Entwicklung

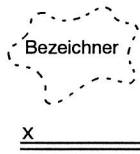
Hierbei werden gemäß Bild 45 Klassen-Diagramme, Objekt-Diagramme, Modul-Diagramme und Prozeß-Diagramme verwendet, um die Struktur der Objekte und Klassen abzubilden. Das individuelle Verhalten der Objekte und das dynamische Verhalten des gesamten Systems wird durch zwei zusätzliche Diagramm-Typen (Zustands-Übergangs-Diagramm und Zeit-Diagramm) dargestellt.

Klassen-Diagramme

Ein Klassen-Diagramm zeigt die Existenz und die Verwandschaftsbeziehungen von Klassen innerhalb des logischen Entwurfs eines Systems auf. Mit einem Klassendiagramm kann die Gesamtheit aller Klassen eines Systems oder nur einer Teilmenge dargestellt werden.

Klassen werden in Form von Wolken dargestellt. Die gestrichelte Umrandung soll andeuten, daß Operationen nur auf Instanzen der Klasse, nicht aber auf der Klasse selbst ausgeführt werden können. Der Name der Klasse steht innerhalb der Wolke.

Die Verwandschaftsverhältnisse zwischen Klassen, wie z. B. Vererbung, Benutzung, und Instanziierung können mit den Symbolen aus Bild 46 abgebildet werden.



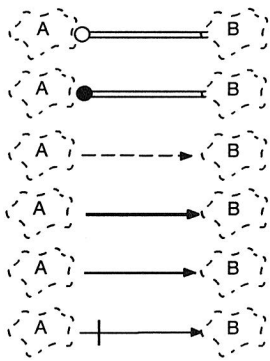
Klassensymbol

Kardinalitäten

Wie oft wird die Klasse von einer anderen benutzt.

x =	0	keinmal
=	1	einmal
=	?	einmal oder keinmal
=	*	keinmal oder mehrfach
=	+	einmal oder mehrfach
=	n	n-mal

Klassenbeziehungen



Klasse A benutzt Klasse B in der Definition des Headers (von Klasse A)

Klasse A benutzt Klasse B in der Implementierung der Methoden (von Klasse A)

Klasse B ist Metaklasse (virtuelle Oberklasse) von A

Klasse A instanziiert Objekte der Klasse B

Klasse A erbt von Klasse B. Klasse A ist Erweiterung

Klasse A erbt von Klasse B. Klasse A ist Einschränkung von B

Algorithmensammlung

Bild 46: Symbole für Klassen und Klassenverwandschaften

Objekt-Diagramme

Objekt-Diagramme stellen in Form von Momentaufnahmen die Existenz von Objekten und deren Kommunikationsbeziehungen dar.

- *Objekte* werden durch Wolken mit durchgehender Umrandung dargestellt.
- *Kommunikationsbeziehungen* werden durch eine Linie zwischen den Objekten dargestellt. Diese Linien können mit den Namen der Methoden versehen werden, die die Objekte gegenseitig aufrufen.

- Die gegenseitige *Sichtbarkeit* der Objekte wird durch zusätzliche Symbole an den Kommunikationsbeziehungen dargestellt.

Objekte kommunizieren miteinander durch den Austausch von Nachrichten, d. h. durch den gegenseitigen Aufruf von Methoden.

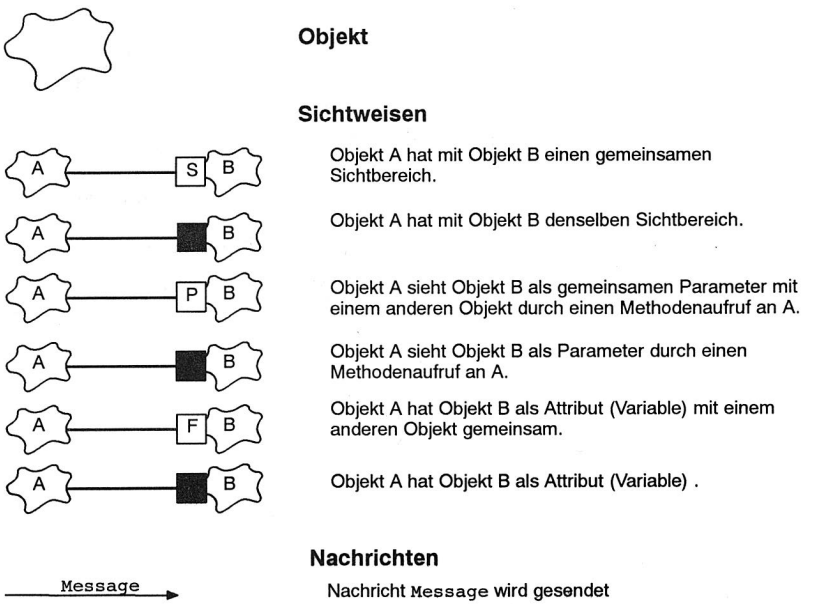


Bild 47: Komponenten des Objekt-Diagramms

Durch die Verwendung der vorgestellten Notationsweise zur Dokumentation wird die Übersichtlichkeit über die Abhängigkeit von Klassen und Objekten in einem objektorientierten Programm deutlich erhöht. Die Diagramme dienen insbesondere in der Entwurfs- und Implementierungsphase als Diskussionsgrundlage und erleichtern die Zusammenarbeit mehrerer Entwickler. In Bild 48 ist beispielhaft die Struktur des abstrakten Automaten, mit der das Verhalten der realen Bestückautomaten modelliert wurde, als Klassendiagramm dargestellt (siehe Kapitel 5.7.2). Zum Erstellen der Diagramme können Software-Werkzeuge verwendet werden, wie z. B. "Rational Rose For Windows".

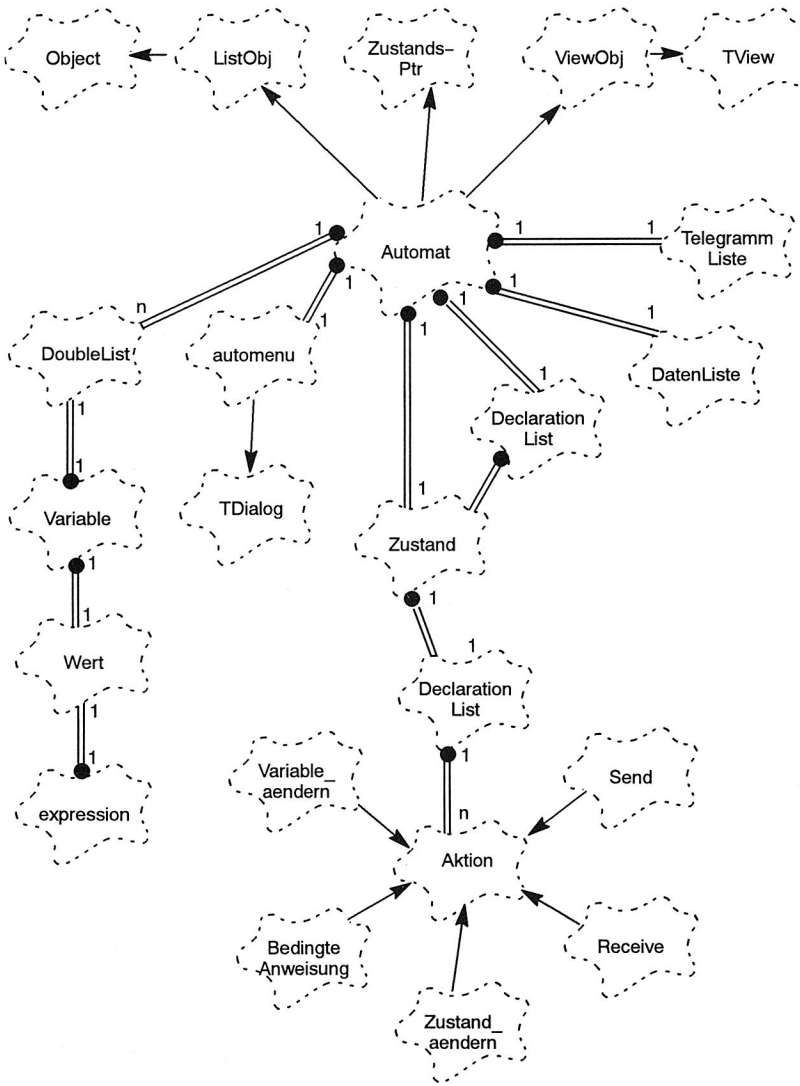


Bild 48: Beschreibung eines abstrakten Automaten in der Notationsweise von Grady Booch

Aus dem Klassen-Diagramm geht hervor, daß die Klasse "Automat" die Eigenschaften der Klassen "ListObj", "ZustandsPtr" und "ViewObj" erbt. Zur Implementierung der Methoden der Klasse "Automat" wird unter anderem die Klasse "Zustand" verwendet. In den zu den Klassendiagrammen gehörigen Templates, die als Grundlage für die Implementierung dienen, werden die Felder und die Methoden der Klassen spezifiziert.

5.6 Ereignissteuerung

Grundlage moderner fensterorientierter Benutzeroberflächen wie z. B. Motif, MS-Windows oder Turbo-Vision ist die Ereignissteuerung. Ereignisse können auf Eingaben des Benutzers zurückgehen, sie können aber auch durch andere Komponenten eines Programmes ausgelöst werden. Ereignisse sind gekennzeichnet durch

- einen Absender und einen Empfänger,
- die Art des Ereignisses (Tastatur, Maus, allgemeine Botschaften),
- einen Kommando-Code, der die auszuführende Aktivität kennzeichnet und
- optionalen Nutzdaten

Die Ereignisse werden von einer zentralen Ereignisverwaltung gesammelt, in eine Warteschlange eingereiht und zur Verarbeitung an die betreffenden Programmteile - bei einer objektorientierten Umgebung an die entsprechenden Objekte - weitergegeben. Alle Objekte, die Ereignisse verarbeiten sollen, müssen über eine entsprechende Methode verfügen. Das wird dadurch erreicht, daß die Klassen, zu denen die Objekte gehören, von einer gemeinsamen Basisklasse abgeleitet werden. Darüberhinaus müssen die Objekte von der zentralen Ereignisverwaltung erreichbar sein.

Ereignisse können gezielt an bestimmte Objekte verschickt werden. Daneben gibt es aber auch Ereignisse ohne spezifischen Adressaten. Diese Ereignisse werden von der Ereignisverwaltung solange weitergereicht, bis ein Objekt gefunden wird, das das Ereignis verarbeitet (Fall 1 und Fall 2 in Bild 49).

Bei der Entwicklung eines ereignisgesteuerten Simulationsprogramms mit einer ereignisgesteuerten Benutzeroberfläche bietet es sich an, die bereits vorhandene Ereignisverwaltung der Benutzeroberfläche um die für die Simulation benötigten Ereignisse zu erweitern. Die Erweiterungen betreffen

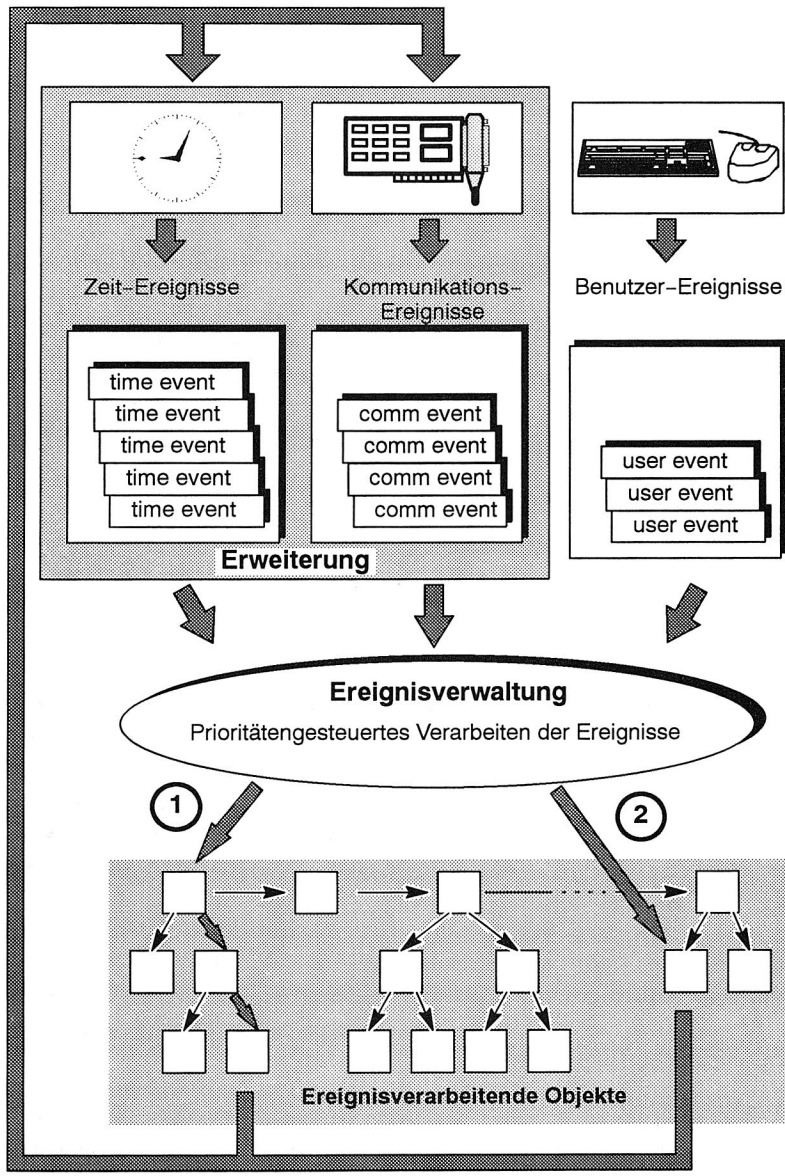


Bild 49: Ereignisverarbeitung

- das Einführen neuer Ereignis-Arten und Kommandos
- das Verwalten zusätzlicher Ereignis-Warteschlangen (mit unterschiedlichen Prioritäten)
- die Erweiterung der zentralen Ereignissammelstelle um zeitgesteuerte Ereignisse und Ereignisse die von der Anlagenkommunikation ausgelöst werden (Empfangen und Verschicken von Datentelegrammen, Bild 49).

5.7 Parametrierung des Simulationsprogramms

In Kapitel 4.2 wurden als Anforderungen an ein Simulationssystem zum Test von Steuerungssoftware eine einfache Anpaßbarkeit an unterschiedliche Anlagenlayouts und an neue Software-Versionen genannt. Da es sich bei den Anwendern derartiger Simulationssysteme im allgemeinen nicht um Simulationsexperten handelt, ist es erforderlich, einfache und anwendungsorientierte Beschreibungsmechanismen zur Verfügung zu stellen, um die Anpassungen vornehmen zu können.

Zur Parametrierung des Simulationsprogramms HSSIM ist das Anlagenlayout, der Aufbau der Telegramme sowie das Verhalten der Bestückautomaten vom Benutzer vorzugeben (siehe Bild 52). Die Beschreibung des Telegrammaufbaus und des Automatenverhaltens erfolgt mit problemangepaßten Beschreibungssprachen. Für diese Sprachen wurde je eine in Backus-Naur-Form darstellbare kontextfreie Grammatik entwickelt.

Das Einlesen der Beschreibungsdateien erfolgt mit Einleseroutinen, die mit Hilfe der vom Betriebssystem UNIX bekannten Werkzeuge **lex** und **yacc** generiert wurden /61/. Das von lex generierte C-Unterprogramm ***yylex*** ermöglicht die lexikalische Analyse von Texten, d.h. die Umwandlung einer Text-Datei in einen Strom von lexikalischen Einheiten, den sogenannten Token. Das von yacc generierte C-Unterprogramm ***yyparse*** ordnet den ermittelten Token eine grammatikalische Struktur zu. Damit können Dateien in der definierten Grammatik auf korrekte Syntax überprüft werden und der Inhalt der Dateien auf entsprechende Hauptspeicherstrukturen (Listen von Objekten) abgebildet werden.

In Bild 51 ist der Datenfluß bei der Generierung und Parametrierung des Simulationsprogramms dargestellt. Aus den lex- und yacc-Dateien zur Beschreibung des Aufbaus der Datentelegramme und zur Beschreibung des Bestückautomatenverhaltens

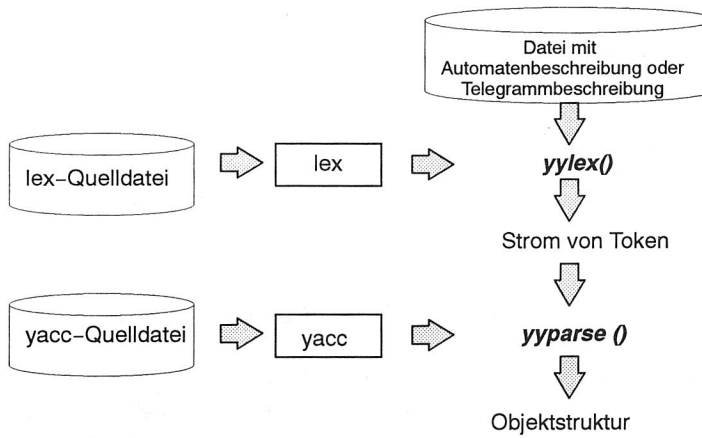


Bild 50: Parsen der Automatenbeschreibungen und der Telegrammbeschreibungen mit lex und yacc

werden Einlesefunktionen generiert. Diese werden zusammen mit den C++-Quelldateien compiliert und mit den notwendigen Bibliotheken zum ausführbaren Simulationsprogramm gebunden.

5.7.1 Parametrierung des Anlagenlayouts und der Bestückautomaten

Die simulierten Bestücklinien unterscheiden sich durch Anzahl, Typ und Konfiguration der Bestückautomaten sowie durch die Versionen der Steuerungssoftware. Der Typ des Bestückautomaten und die Version der Steuerungssoftware legt das automatenspezifische Verhalten und den Aufbau der Datentelegramme fest. Jeder Kombination aus Bestückautomatentyp und Software-Version ist somit je eine Automatenbeschreibung und eine Telegrammbeschreibung zuzuordnen. Individuelle Automateninstanzen entstehen durch Verknüpfung von Beschreibungsdateien mit einer IEC-Bus-Adresse (Bild 52).

Die Konfiguration eines Bestückautomaten beschreibt dessen Aufbau und kinematischen Eigenschaften (Pick-and-Place-Bestückkopf, Revolverkopf) sowie die Ausstattung mit Zusatzeinrichtungen (siehe Kapitel 2). Die Konfigurationsdaten werden zentral auf dem Linienrechner verwaltet und über die Datentelegramme an die Be-

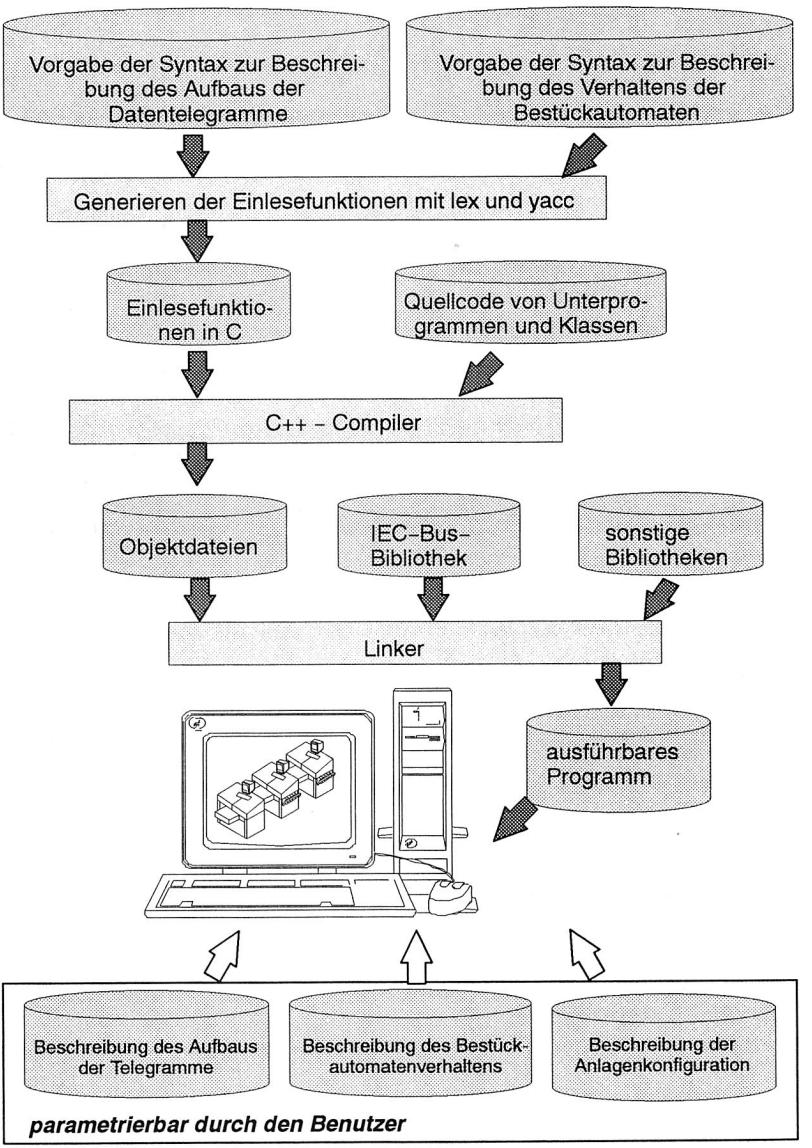


Bild 51: Datenfluß bei der Generierung und Parametrierung des Simulationsprogramms

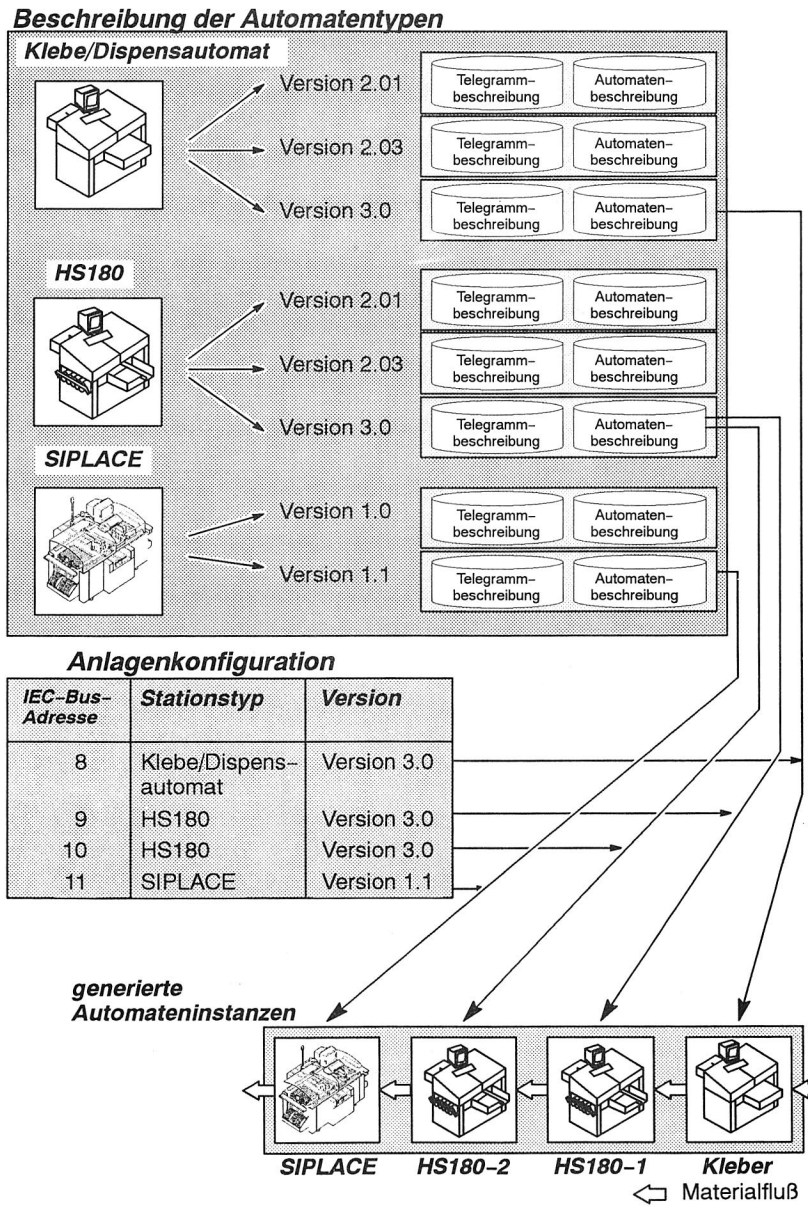


Bild 52: Konfiguration des Simulationsprogramms

stückautomaten, bzw. an das Simulationssystem übergeben. Da die Konfigurationsdaten für die Bestückautomaten über die Datentelegramme übertragen werden, müssen diese Daten nicht im Simulationsprogramm parametriert werden, sondern können durch Analyse der Datentelegramme ermittelt werden.

Ebenso werden durch die Analyse der Rüstdaten- und Nutzendatentelegramme die Rüstung und die Leiterplatten-spezifischen Daten ermittelt.

5.7.2 Modellierung der Bestückautomaten

Für die Simulation muß das Verhalten der Bestückautomaten in geeigneter Form modelliert werden. Dabei ist aus der Sicht des Linienrechners das Verhalten der Bestückautomaten durch den Austausch der Datentelegramme, also das Kommunikationsprotokoll, definiert. Die Modellierung von Bestückautomaten-internen Vorgängen muß nur soweit beachtet werden, wie es auf das Kommunikationsverhalten Einfluß nimmt. Durch die internen Vorgänge wird einerseits der Nutzdaten-Inhalt der Telegramme bestimmt, andererseits aber auch das Versenden von Telegrammen veranlaßt.

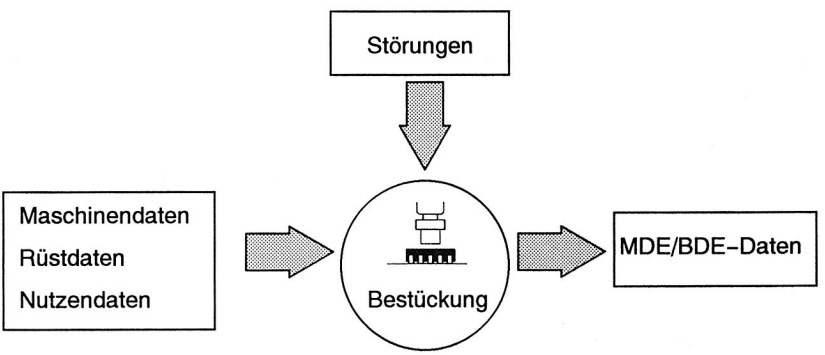


*Bild 53: Funktionale Abh*ängigkeit der MDE/BDE-Daten

Der Inhalt der Nutzdaten ist gemäß der Einteilung in Bild 32 funktional von den Ausgangsdaten abhängig. Von besonderer Wichtigkeit sind hierbei die MDE/BDE-Daten (Bild 53), die nach der Bestückung einer Leiterplatte an den Linienrechner übertragen werden.

Aus der Sicht des Linienrechners befinden sich die Bestückautomaten in festgelegten Zuständen ("Warten auf Daten", "Warten auf Leiterplatten", "Bestücken", "Fehler-

zustand", usw.). Ein Zustandswechsel tritt durch das Senden oder Empfangen von Telegrammen ein. Durch diese Zustandsübergänge läßt sich das Verhalten der Bestückautomaten gegenüber dem Linienrechner mit Hilfe der Automatentheorie durch einen abstrakten Automaten modellieren.

Ein ***abstrakter Automat*** ist definiert als ein Quintupel:

$$\alpha = \{ Q, \Sigma, \delta, Q_-, Q_+ \}$$

mit

- Q als Menge von Zuständen
- Σ als Alphabet von Eingabewerten
- $\delta \subseteq Q \times \Sigma \times Q$ als Übergangsrelation, die eindeutig festlegt, durch welche Elemente des Alphabets zwei Zustände ineinander überführbar sind
- Q_- als Menge von Startzuständen
- Q_+ als Menge von Endzuständen

Angewendet auf die Bestückautomaten ergibt sich:

- Die Menge Q ist die Menge aller Zustände, die ein Bestückautomat einnehmen kann.
- Die Menge Σ ist die Menge aller Telegramme (Bild 35)
- δ, die Übergangsrelation besteht aus allen erlaubten Zustandsübergängen der Form:

  <aktueller Zustand> <Telegramm> <Folgezustand>

  z. B.:
  <undefiniert> <SART> <Warten_Stationsart>

  Weitere erlaubte Übergänge können Bild 54 entnommen werden.

- $Q_- = \{$ undefiniert $\}$
- $Q_+ = \varnothing$

Die Modellierung von internen Abläufen wie z. B. "Eingabeband sperren" oder "Eingabeband freigeben" läßt sich als Erweiterung des Automatenkonzepts durch das

Setzen von Variablen modellieren. Dadurch wird eine zu große Zahl von möglichen Zuständen vermieden.

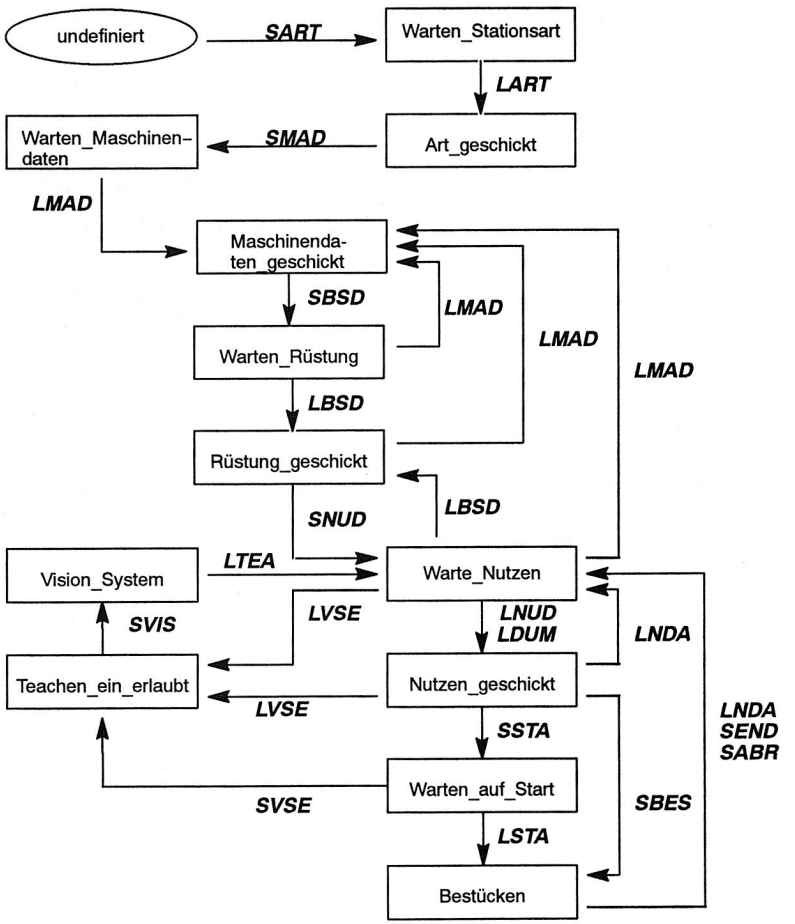


Bild 54: Ausschnitt aus den Zustandsübergängen eines Bestückautomaten

- Die Menge V wird definiert als die Menge der Automatenvariablen:
  { BarcodeLeser, Eingabeschnittstelle, Bestücken }

```
automat HS180
{
   EingabeSchnittstelle :
         offen, gesperrt;
   LeiterPlatte :
         in_Bearbeitung , durchtransportiert;
   BestueckVorgang :
         laufend, unterbrochen;
   BarcodeLeser :
         ausgeschaltet, eingeschaltet;
}

...

zustand 26 Nutzendaten_geschickt
{
   wait 100 seconds

   receive telegramm LSSP
   {
      wait 10 seconds
      set EingabeSchnittstelle = gesperrt;
   }

   receive telegramm LSSO
   {
      wait 60 seconds
      set EingabeSchnittstelle = offen;
   }

   receive telegramm LBDE
   {
      wait 61 seconds
      send telegramm SBDE;
   }

   receive telegramm LNDA
   {
      wait 64 seconds
      change to Warten_Nutzendaten;
   }

   if(BarcodeLeser = eingeschaltet)
   {
      send telegramm SBAR;
   }

   if (LMAD.Ma_opt[10] = 1)
   {
      wait 10 seconds
      send telegramm SNVD
   }

   send telegramm SART
   {
      change to Warten_Modus
   }
}

...
```

Definition von Simulationsvariablen

Definition eines Zustands

Warteanweisung

Setzen einer Simulationsvariablen nach dem Empfang eines Linienrechnertelegramms

Senden eines Telegramms nach dem Empfang eines Linienrechnertelegramms

Zustandswechsel nach dem Empfang eines Linienrechnertelegramms

Aktion in Abhängigkeit von einer Simulationsvariablen

Aktion in Abhängigkeit von einer Telegrammkomponente

Zustandswechsel nach dem Senden eines Telegramms

Bild 55: Ausschnitt aus einer Automatenbeschreibungsdatei

- Jeder Variablen wird eine Menge W_V von möglichen Variablenwerten zugeordnet:

$W_{BarcodeLeser}$	{ ausgeschaltet, eingeschaltet }
$W_{Bestücken}$	{ ausgeschaltet, eingeschaltet }
$W_{Eingabeschnittstelle}$	{ offen, gesperrt }, usw.

- $\sigma \subseteq Q \times V \times \Sigma \times W_V$ ist eine Relation, die einer Variablen einen neuen Wert zuweist, wenn in einem Automatnzustand ein Telegramm empfangen oder gesendet wird, z. B.:

  <undefiniert> <Eingabeband> <LSSP> <gesperrt>

Zur Behandlung der Quittierung eines Linienrechnertelegramms mit einem Antworttelegramm ohne Zustandswechsel wird eine weitere Relation Φ definiert:

$$\Phi \subseteq Q \times \Sigma \times \Sigma,$$

Diese Relation ordnet jedem Linienrechnertelegramm, das eine Quittierung verlangt, das entsprechende Stationstelegramm zu.

Bild 55 zeigt die Umsetzung der obigen Definitionen in die entwickelte Beschreibungssprache. Die Beschreibung eines Bestückautomaten besteht aus einem Definitionsteil für die Simulationsvariablen, der Definition der Automatenzustände und optionalen Kommentaren. Eine Zustandsbeschreibung besteht aus einem eindeutigen Bezeichner und den Aktionen, die der Bestückautomat in diesem Zustand ausführen kann. Dazu gehören Warteanweisungen, Anweisungen zum Senden und Empfangen von Telegrammen, bedingte Anweisungen und Zustandswechsel. Die bedingten Anweisungen können von Simulationsvariablen und/oder von Telegrammkomponenten abhängig gemacht werden. Durch die Warteanweisungen wird die Zeitdauer vorgegeben, die der reale Bestückautomat zum Durchführen von internen Aktionen benötigt. Die Send-Anweisung am Ende der Zustandsbeschreibung in Bild 55 wird nach der angegeben Wartezeit von 100 Sekunden ausgeführt, wenn nicht vorher durch das Empfangen eines Telegramms vom Linienrechner bereits eine andere Aktion mit Zustandswechsel durchgeführt wurde.

5.7.3 Analyse und Synthese der Daten-Telegramme

Durch die Analyse der Linienrechner-Telegramme werden in den realen Bestückautomaten die Daten ermittelt, die für die Bestückung benötigt werden. Im Simulationssystem dient die Telegrammanalyse

- zur Ermittlung von simulationsrelevanten Daten
- zur Überprüfung der Telegramme auf syntaktische und semantische Korrektheit
- zur Ermittlung von Daten, die zur Analyse anderer Telegramme benötigt werden.

Mit Hilfe der entwickelten Beschreibungssprache wird der Aufbau von Telegrammen oder allgemeiner Datensätzen beschrieben, die im Binärformat vorliegen. Die Telegramme können aus folgenden **Telegrammkomponenten** bestehen:

- Strings, Integern (mit 2 oder 4 Byte Länge) und Real-Werten (mit 4 oder 8 Byte Länge)
- Feldern von Strings, Integern und Real-Werten. Die Felder haben eine feste oder eine variable Länge. Die Länge kann vom Wert anderer Telegrammkomponenten abhängig gemacht werden.
- optionalen Telegrammteilen, deren Existenz von vorherigen Komponenten des Telegramms oder von Komponenten anderer Telegramme abhängen.

Durch die Beschreibung wird jeder Telegrammkomponente ein logischer Name zugeordnet. Über diesen Namen kann nach dem Parsen eines Telegramms auf die Komponente zugegriffen und ihr Wert abgefragt, bzw. verändert werden. In der Beschreibungsdatei kann den Telegrammkomponenten optional ein Wert zugewiesen werden. Zur Überprüfung der Korrektheit eines empfangenen Telegramms werden die Daten im Telegramm mit den Daten in der Telegrammbeschreibung verglichen.

In Bild 56 ist als Beispiel die Beschreibung des Aufbaus des Nutzendatentelegramms abgebildet. Hierin sind alle möglichen Komponentenarten (Einzelwerte, Felder variabler Länge, optionale Komponenten) enthalten. Die optionale Nadelkissen-Information ist abhängig von einer Komponenten eines anderen Telegramms. Der Zugriff auf derartige Komponenten erfolgt über:

```
telegramm LNUD
{
   string NUdatei;
   string BPdatei;
   string ZPdatei;
   string NKdatei;
   string ZPsatzname;

   /* Nutzen-Informationen */

   real8 Nu [1..20];
   real8 Nu_pass [0..9][1..2];
   real8 Nu_stopper [1..3][1..2];
   int2 Nu_muster [0..LNUD.Nu_muster [0][5]][1..5];

   int2 Nu_sub_ausl [1..LNUD.Nu[7]][1..LNUD.Nu[9]]
         [1..LNUD.Nu_muster[0][5]];
   int2 Nd [1..10][1..2];
   int2 Nu_mark_grp [1..15][1..3];
   real8 Nu_mark_off [1..15][1..2][1..2];

   /* Bestueck-Positionen */

   int2 Anfangbest [1..3];
   real8 Anz_Bp;
   if (LNUD.Anz_Bp > 0)
   {
      int2 Bp [1..LNUD.Anz_Bp] [1..8];
      string Bpstring [1..LNUD.Anz_Bp][1..2];
   }

   /* Zentrierplattensatz */

   real8 Anz_ZP;
   if (LNUD.Anz_ZP > 0)
   {
      int2 ZP_werte [1..LNUD.Anz_ZP][0..3];
      string ZP_name [1..LNUD.Anz_ZP];
   }

   /* Nadelkissen-Information */

   if (LMAD.Ma_opt[50] = 1)
   {
      int2 Nadel_kissen [0..16];
   }
}
```

Verweis auf vorherige Telegramm-komponenten

Verweis auf ein anderes Telegramm

Bild 56: Beschreibung des Aufbaus eines Nutzendatentelegramms

<Telegrammname>.<Komponentenname> [Index]

Durch das Parsen der Telegramme entsteht somit ein Datenpool aus den Komponenten der Telegramme (Bild 57). Jedes Element des Datenpools besteht aus dem Namen und dem zugewiesenen Wert. Da der Aufbau der Telegramme von den Bestück-

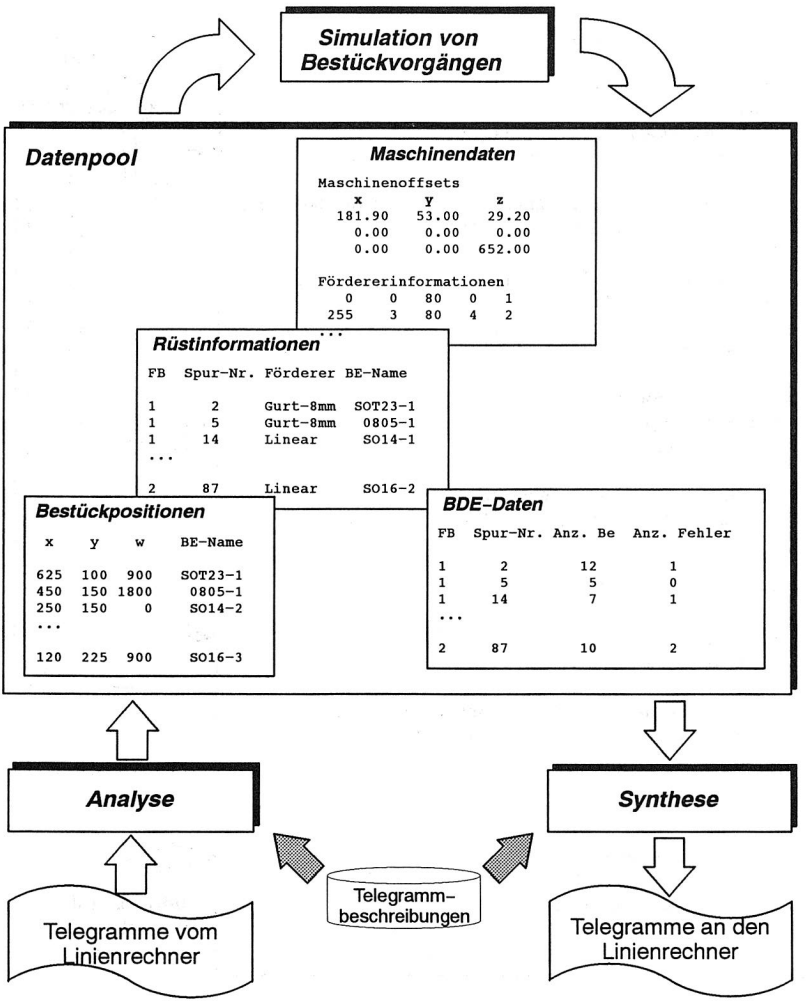


Bild 57: Analyse und Synthese der Datentelegramme

automaten auf die selbe Weise beschrieben wird wie der Aufbau der Telegramme vom Linienrechner, können an Hand der Daten in dem Datenpool die Telegramme an den Linienrechner generiert werden ("Synthese") . Der Wert der Telegrammkomponenten wird mit Hilfe des Inhalts der Linienrechnertelegramme ermittelt.

5.7.4 Benutzeroberfläche

Die Benutzeroberfläche von HSSIM entspricht durch die Verwendung der Turbo-Vision-Bibliothek dem SAA-Standard mit Menü- und Statuszeile und einem Desktop zur Aufnahme von Dialog- und Editorfenstern (Bild 58). Für jeden simulierten Bestückautomaten wird ein Statusfenster auf dem Bildschirm dargestellt, aus dem der aktuelle Zustand des Automaten und die zu versendenden Telegramme hervorgehen. Hier ist auch ein Umschalten zwischen automatischer und manueller Simulation möglich. Eine weitergehende Visualisierung der Simulationsvorgänge und der Daten erfolgt mit Hilfe des im folgenden Kapitel vorgestellten Protokollierungs- und Visualisierungssystem.

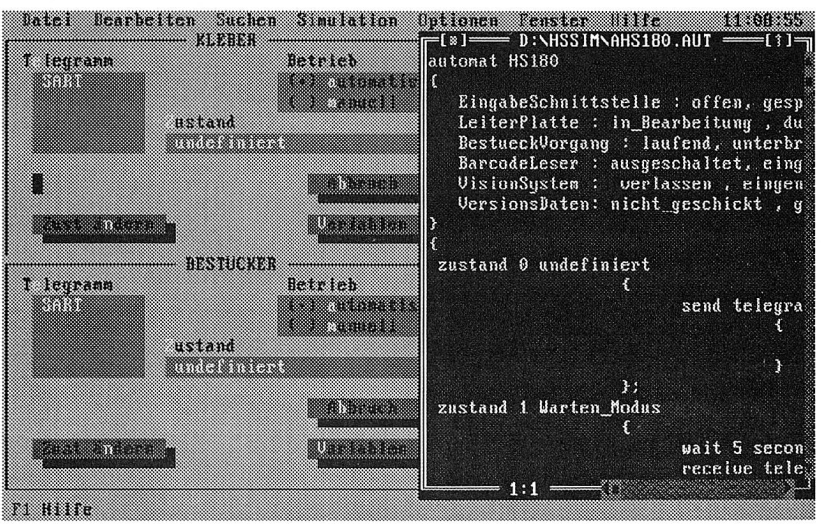


Bild 58: Oberfläche von HSSIM

6. Protokollierung und Visualisierung der Steuerungsvorgänge

Durch Protokollierung und Visualisierung von Kommunikationsbeziehungen zwischen Automatisierungsgeräten wird eine höhere Transparenz der Steuerungsvorgänge erreicht. Diese Transparenz ist während der Test- und Inbetriebnahmephasen von Automatisierungseinrichtungen ein wichtiger Beitrag zur Lokalisierung und Behebung von Kommunikationsfehlern /28, 62/. Angewendet auf SMD-Bestückanlagen bedeutet dies, daß durch geeignete Visualisierungsmaßnahmen überprüft werden kann, ob ein reibungsloser Datenaustausch zwischen Linienrechner und Stationsrechnern bzw. Linienrechner und Simulationssystem stattfindet. Durch die Analyse und graphische Darstellung der übertragenen Nutzdaten werden Fehler in den Telegrammen erkannt, die zu einem Fehlverhalten der Bestückautomaten führen.

6.1 Einsatzbereiche

Tests der Linienrechnersoftware

Während der Testphase der Linienrechner-Software mit dem Simulationssystem oder aber an einer realen Anlage werden die auf dem IEC-Bus übertragenen Daten protokolliert und visualisiert, um Protokollfehler erkennen zu können und die Datentelegramme manuell oder automatisch auf Korrektheit überprüfen zu können. Dadurch lassen sich Testläufe bezüglich des Datenverkehrs miteinander vergleichen. Durch Auswertung der Übertragungsdauern und der Übertragungsabstände wird die Auslastung des Busses ermittelt und kann durch Protokolländerungen optimiert werden.

Inbetriebnahme von Bestücklinien

Ein großer Zeitanteil bei der Inbetriebnahme von Fertigungsanlagen wird verwendet, um die Kommunikationsbeziehungen zwischen den an der Steuerung beteiligten Automatisierungsgeräten zum laufen zu bringen /98/. Beim IEC-Bus treten z. B. besonders viele Fehler durch falsche Parametrierung der Karten auf (falsche Adressen, zu kurze Timeout-Intervalle). Durch die Überwachung und Visualisierung der Kommunikation zwischen Linienrechner und Stationsrechnern können derartige Fehler erkannt und behoben werden. Durch die Visualisierung der übertragenen Daten läßt

sich die korrekt durchgeführte Parametrierung der Bestückautomaten in der Linienrechner-Software überprüfen. Vergißt man z. B., das Vorhandensein einer optischen Zentrierstation auf einem Bestückautomaten zu parametrieren, werden keine Gehäuseformdaten und keine Daten zur Initialisierung der Zentrierstation an den Automaten übertragen. Bei der automatisierten Erstellung einer Rüstung werden dem Automaten nur Bauelemente zugewiesen, die nicht optisch zentriert werden müssen, d. h. die Fähigkeiten des Automaten werden nicht voll ausgenutzt.

Schulung des Bedienpersonals
Setzt man das Simulationssystem HSSIM während der Schulung ein, können mit einem geeigneten Visualisierungssystem die von den Auszubildenden zur Übung vorgenommenen Dateneingaben überprüft werden. Wird z. B. nach der Eingabe von Nutzendaten ein Abbild der zu bestückenden Leiterplatte dargestellt, kann die Anordnung und Orientierung der Einzelschaltungen sowie die Position der Bauelemente überprüft und gegebenenfalls korrigiert werden. Darüberhinaus führt eine geeignete Visualisierung zu einem größeren Verständnis für das Gesamtsystem, da hierbei auch ansonsten verborgene Vorgänge, wie z. B. der Telegrammverkehr, transparent gemacht werden

6.2 Anforderungen an das Visualisierungssystem

6.2.1 Datenerfassung

Zunächst müssen die zu visualisierenden Daten erfaßt werden. Im betrachteten Anwendungsfall handelt es sich um alle steuerungsrelevanten Daten, die über den IEC-Bus übertragen werden. Neben den übertragenen Nutzdaten sind zusätzlich die in Kapitel 14.2.4 aufgeführten Parameter Sender- und Empfängeradresse, sowie Übertragungszeitpunkt, Übertragungsdauer und Übertragungsstatus zu ermitteln. Die Erfassung der auf dem IEC-Bus ablaufenden Vorgänge kann über einen entsprechenden Bus-Analyzer oder über den in Kapitel 5.3.2 vorgestellten passiven Mithörmodus unter Ausnutzung der Mehradressfunktionalität erfolgen.

Bei der Erfassung mit einem Analyzer wird jede Änderung der Zustände der 16 Leitungen des IEC-Buses erfaßt und in eine Datei mitprotokolliert. Jeweils zwei Byte in dieser Datei kennzeichnen einen Zustand der Leitungen. Eine Auswertung der Daten ist nur Off-Line nach der Erfassung möglich (Bild 59). Da nur die Leitungszustände

erfaßt werden, fehlen die geforderten Zeitinformationen . Die Protokolldateien können sehr groß werden, insbesondere dann, wenn wie bei dem Linienrechner mit BASIC-Betriebssystem auch die Festplatte an den Bus angeschlossen ist und damit auch alle Datenübertragungen, die zwischen Linienrechner und Festplatte ausgetauscht werden, aufgezeichnet werden.

Bei der Erfassung der Daten im passiven Mithörmodus werden diese Nachteile umgangen. Zum einen kann hier die Erfassung auf festgelegte Kommunikationspartner beschränkt werden, was die erfaßte Datenmenge deutlich reduziert, zum anderen können die Daten On-Line verarbeitet, analysiert und mit den zusätzlichen Parametern Zeit und Übertragungsstatus verknüpft werden.

6.2.2 Protokollierung

Die Aufgabe der Protokollierung ist es, die erfaßten Daten zu verwalten und für weitere Verarbeitung und Analyse in geeigneter Weise bereit zu stellen. Zum Abspei-

POSITION	CHR..HEX....DATA	CONTROL	REMARKS
0	'!' 21 00100001	01000 010	ATN
1	'f' 66 01100110	00000 000	ATNv
2	'g' 67 01100111	01001 000	ATN^ IFC
3	'!' 21 00100001	01001 010	IFC
4	'!' 21 00100001	01010 010	REN^
5	'!' 21 00100001	01011 010	IFC
6	'@' 40 01000000	01010 011	TA0
7	'?' 3F 00111111	01010 011	UNL
8	'!' 21 00100001	01010 011	LA1
9	'!' 21 00100001	01110 010	SRQ^
10	'?' 3F 00111111	01110 011	UNL
11	'!' 21 00100001	01110 011	LA1
12	CAN 18 00011000	01110 011	SPE
13	'@' 40 01000000	01110 011	TA0
14	'@' 40 01000000	00110 010	ATNv
15	'A' 41 01000001	00010 010	SRQv
16	'A' 41 01000001	00010 111	DAB
17	'A' 41 01000001	01010 110	ATN^
18	EM 19 00011001	01010 011	SPD
19	' ' 5F 01011111	01010 011	UNT
20	'?' 3F 00111111	01010 011	UNL
21	'!' 21 00100001	01010 011	LA1
22	'@' 40 01000000	01010 011	TA0
23	'@' 40 01000000	00010 010	ATNv
24	'b' 62 01100010	00010 111	DAB
25	'd' 64 01100100	00010 111	DAB
26	'2' 32 00110010	10010 111	DAB EOI
27	'2' 32 00110010	01010 110	ATN^
28	'A' 41 01000001	01010 011	TA1
29	'?' 3F 00111111	01010 011	UNL
30	' ' 20 00100000	01010 011	LA0
31	' ' 20 00100000	00010 010	ATNv
32	'L' 68 01101000	00010 011	DAB
33	'A' 61 01100001	00010 011	DAB
34	'R' 6C 01101100	00010 011	DAB
35	'T' 6C 01101100	00010 011	DAB

Bild 59: Abbildung der mit einem IEC-Bus-Analyzer erfaßten Daten

chern der erfaßten Daten in einer Protokolldatei werden Informationen über den Übertragungszeitpunkt, den Übertragungsstatus und die Kommunikationspartner ergänzt. Die Abspeicherung kann ohne weitere Aufbereitung oder mit gefilterten, ergänzten oder reduzierten Daten durchgeführt werden.

6.2.3 Visualisierung

Die Aufgabe der Visualisierung ist es, die empfangenen Daten nach einer Aufbereitung für den Benutzer in einer möglichst aussagekräftigen Form darzustellen. Hierzu müssen die Daten syntaktisch und semantisch analysiert werden. Die Visualisierung kann online erfolgen oder als Playback an Hand der protokollierten Daten. Dabei können die Daten komprimiert und als Verlauf über einen größeren Zeitraum dargestellt werden.

6.2.4 Datenverwaltung

Die Datenverwaltung speichert und verwaltet relevante Daten über einen längeren Zeitraum. Während bei der Protokollierung noch die Roh-Daten gespeichert werden, werden die in der Datenverwaltung komprimierten und analysierten Daten gespeichert. Der Zugriff auf die Daten erfordert eine effiziente Speicherung und verschiedene Retrieval-Mechanismen. Diese Forderungen werden am besten durch eine Datenbank erfüllt.

6.3 Das Visualisierungssystem HSMON

Das Visualisierungs- und Protokollierungssystem HSMON visualisiert den Telegrammverkehr zwischen Linienrechner und Stationsrechnern bzw. Linienrechner und Simulationssystem. Bei der Entwicklung des Systems wurde Wert gelegt auf:

- Größtmögliche Flexibilität für Anpassungen an neue Versionen der zu überwachenden Daten
- Optimale Nutzung bereits vorhandener Funktionalitäten in verschiedenen Programmen
- Funktionale Eigenständigkeit verschiedener Projektmodule

6.3.1 Struktur des Systems

Der Forderung nach größtmöglicher Flexibilität des Systems bei Anpassung an neue Daten-Versionen wurde durch einen hierarchischen Systemansatz entsprochen (Bild 60), bei dem versionsabhängige und versionsunabhängige Module unterschieden werden. Die versionsabhängigen Module sind soweit wie möglich parametrierbar, so daß versionsabhängige Eigenschaften durch Änderungen in Konfigurationsdateien ausgedrückt werden können. Das System wurde unter MS-Windows entwickelt, um die Vorteile einer grafischen Benutzeroberfläche ausnutzen zu können. Durch die Vielzahl der zur Verfügung stehenden Windows-Anwendungen, die zur Verarbeitung und Visualisierung der Daten verwendet werden können, wird der eigentliche Programmieraufwand deutlich reduziert /112/.

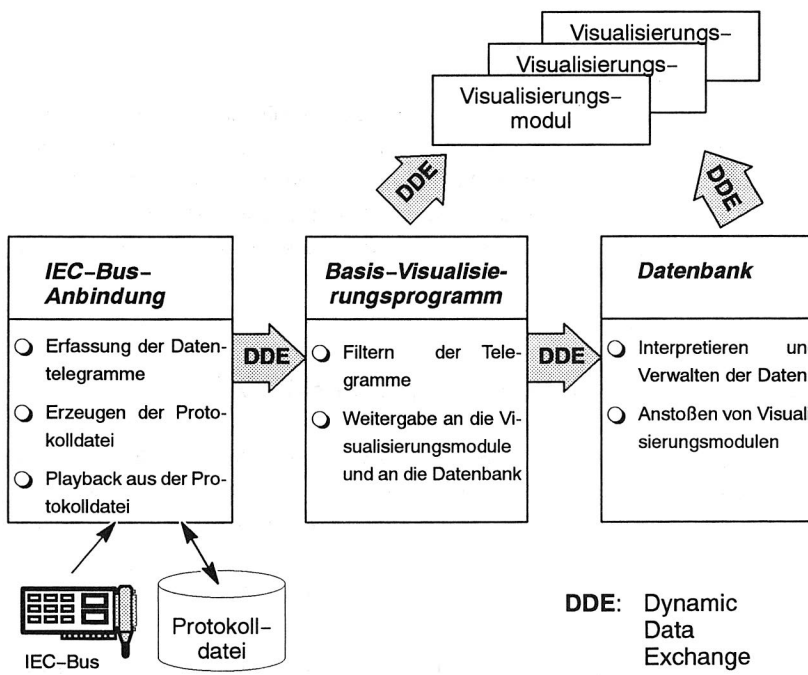


Bild 60: Struktur des Visualisierungssystems

Die Koordination der an dem Visualisierungssystem beteiligten Programme und die Kommunikation zwischen diesen Programmen wurde über den MS-Windows-Interprozeß-Kommunikationsmechanismus DDE (Dynamic Data Exchange) realisiert, der von vielen Windows-Programmen unterstützt wird /56, 79/. Über diesen Mechanismus ist sowohl ein Datenaustausch zwischen Programmen möglich, als auch die Steuerung von Programmen. Eines der Programme fungiert dabei als Server, das andere als Client. Während einer DDE-Kommunikation werden folgende Schritte durchlaufen:

- Initialisierung von Server und Client
- Verbindungsaufbau vom Client zum Server über eine zweistufige Adressierung (SYSTEM, TOPIC)
- Anforderung von Diensten durch den Client
- Verbindungsabbau

Die Anforderung eines Dienstes vom Client ist durch eine dreistufige Kennung charakterisiert:

SYSTEM: Name des Servers

TOPIC: Themenbereich, zu dem eine Verbindung aufgebaut wurde, und zu der ein Dienst angefordert wurde

ITEM: Bezeichnung des angeforderten Datums bei Datenaustausch oder Bezeichnung des angeforderten Dienstes

Als Dienste wurden verwendet:

ADVISE: Anweisung an den Server, neue Daten automatisch zu versenden

REQUEST: Der Client fordert gezielt Daten vom Server an.

EXECUTE: Der Client fordert den Server zu einer Aktion auf (Steuerung des Servers durch den Client).

6.3.2 Datenerfassung und Protokollierung

Die Datenerfassung erfolgt mit dem IEC-Bus-Server, dessen Aufgabe aus der Erfassung des Datentransfers, dem Erstellen einer Protokolldatei und der Weitergabe

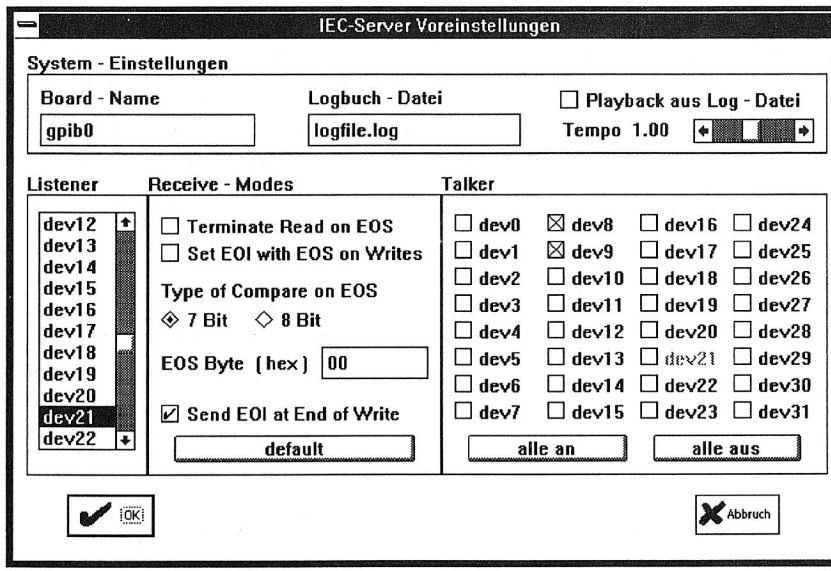


Bild 61: Konfigurationsmaske des IEC-Servers

der erfaßten Daten über die DDE-Schnittstelle besteht. Durch die Möglichkeit, den Telegrammverkehr mit Hilfe der erstellten Protokolldateien nachzubilden, wird neben einer On-Line auch eine nachträgliche Off-Line-Auswertung der Daten ermöglicht.

Der IEC-Bus-Server ist in der Lage, den Datenverkehr auf dem IEC-Bus zwischen beliebig konfigurierbaren Kommunikationspartnern aufzeichnen zu können. Dazu wurde die bereits erwähnte Mehradressfunktionalität verwendet. Die Konfigurierung des Servers erfolgt über die in Bild 61 dargestellte Eingabemaske, über die neben der Auswahl von Sender und Empfängeradresse auch die Übertragungsparameter eingestellt werden können, die ein Übertragungsende anzeigen. Beim Nachbilden des Telegrammverkehrs an Hand einer Protokolldatei kann die Geschwindigkeit, d. h. der zeitliche Abstand der Datenübertragungen verkürzt werden. In der erstellten Protokolldatei werden neben den übertragenen Daten auch die in Kapitel 14.2.4 aufgeführten Parameter Sender- und Empfängeradresse, sowie Übertragungszeitpunkt, Übertragungsdauer und Übertragungsstatus erfaßt.

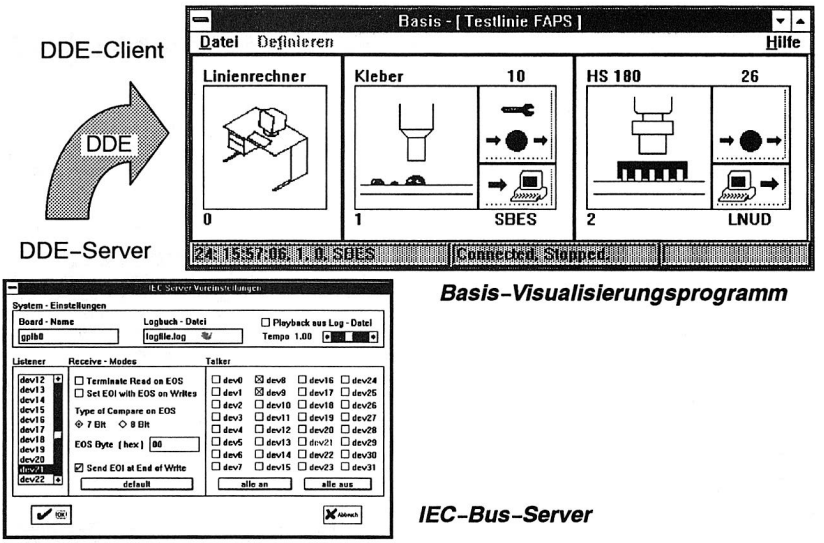


Bild 62: Kopplung von IEC-Bus-Server und Basis-Visualisierungsprogramm

Zur Weitergabe der erfaßten Daten ist der IEC-Bus-Server als DDE-Server konzipiert, der gleichzeitig mehrere Clients bedienen kann. Über die DDE-Schnittstelle können die Telegramme gezielt mit einem REQUEST angefordert werden oder aber die Clients fordern den Server mit einem ADVISE auf, neue Daten sofort an sie weiterzumelden. Darüberhinaus verfügt der Server über eine Steuerschnittstelle, über die die Clients spezielle Aktionen auslösen können (Start und Stop von Datenerfassung oder Playback, Parametrierung des Servers).

6.3.3 Visualisierung

Gemäß der hierarchischen Strukturierung des Gesamtsystems bestehen die Visualisierungsmodule aus einem Modul zur Darstellung von versionsunabhängigen Telegramm-Daten (Basis-Visualisierungsmodul) und aus spezialisierten Visualisierungsmodulen für Maschinendaten, Rüstdaten, Nutzendaten und MDE/BDE-Daten.

Das Basis-Visualisierungsmodul ist über die DDE-Schnittstelle als Client mit dem IEC-Bus-Server verbunden und fordert den Server über den ADVISE-Befehl auf, eintreffende Telegramme an weiterzugeben. Im Basis-Visualisierungsmodul werden

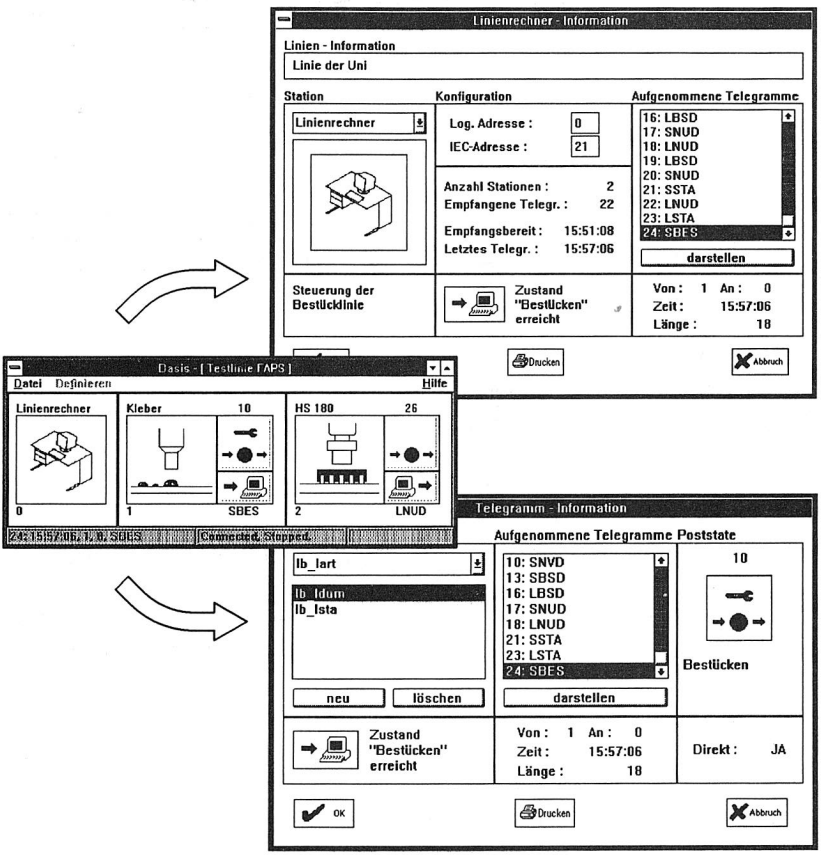


Bild 63: Visualisierung der Kommunikations-Ereignisse im Basis-Visualisierungsprogramm

die Zustände der am IEC-Bus angeschlossenen Geräte (Linienrechner und Bestückautomaten) abgebildet. Der jeweilige Zustand ergibt sich aus dem Ablauf des Telegrammverkehrs gemäß dem Zustandsübergangsgraph in Kapitel 5.7.2. Für Linienrechner und Bestückautomaten werden in Dialogfenstern zusätzliche Informationen dargestellt (Bild 63). Dazu gehört auch eine Liste aller Telegramme die von dem dargestellten Gerät empfangen oder versendet wurden, zusammen mit dem

Übertragungszeitpunkt, der Telegrammlänge und der Adresse des Kommunikationspartners.

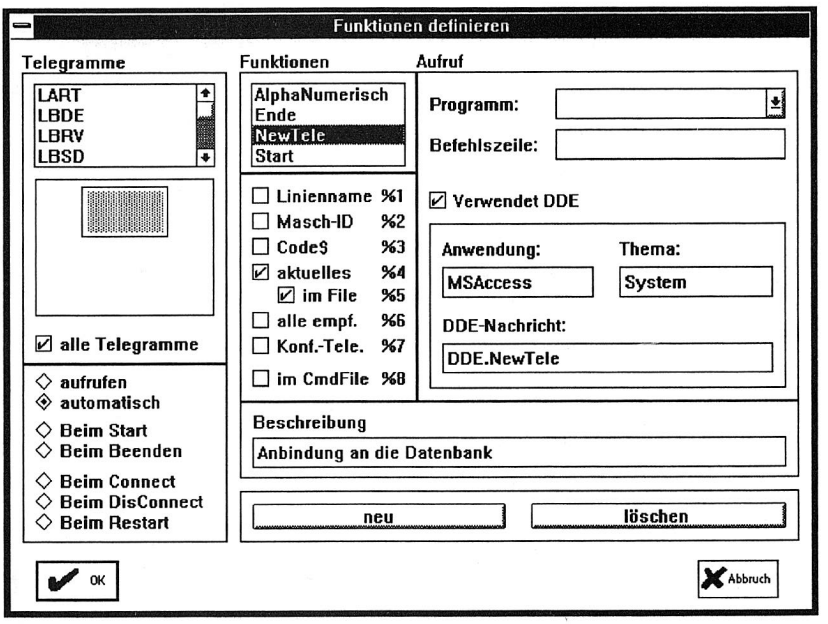
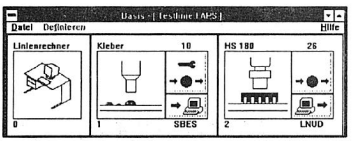


Bild 64: Definition der Programmschnittstelle

Über die Programmschnittstelle des Basis-Visualisierungsprogramms kann der Benutzer den Zeitpunkt und die Art der Visualisierung für jeden Telegrammtyp festlegen (Bild 64) . Neben dem Telegramminhalt können über diese Schnittstelle zusätzliche Daten, wie der Name der aktuell visualisierten Linie, die logische Adresse der Station, für die das Telegramm ausgewertet werden soll und der Telegrammtyp übergeben werden. Das Aktivieren der speziellen Visualisierungsmodule erfolgt wahlweise über einen parametrisierten Programmaufruf, über eine parametrisierbare DDE-Programmschnittstelle oder über eine Kommando-Datei. Durch diese unterschiedlichen Aktivierungsmöglichkeiten wird eine größtmögliche Flexibilität beim Einsatz von existierenden Anwendungsprogrammen gewährleistet. So lassen sich z. B. zur Darstellung von Zeitanteilen in den BDE-Daten die Balkendiagramm-Funktionalitäten von Tabellenkalkulationsprogrammen verwenden.



Basis – Visualisierungsprogramm

DDE

Allgemeine Visualisierungsmodule
(verwendbar für alle Telegrammtypen)

```
Telegramm: LNUD
0     0     1     14     LNUD
NU-Datei          NUvdi
BP-Datei          BPvdi
ZP-Datei          ZP_DATEI
NK-Datei
ZP-Satzname
NU[1] = 42.500000
NU[2] = 137.500000
NU[3] = 6.000000
NU[4] = 0.000000
NU[5] = 0.000000
NU[6] = 0.000000
NU[7] = 1.000000
NU[8] = 160.000000
NU[9] = 2.000000
NU[10] = 100.000000
NU[11] = 1.500000
NU[12] = 233.400000
NU[13] = 0.000000
NU[14] = 0.000000
NU[15] = 1.000000
NU[16] = 0.000000
NU[17] = 1.000000
NU[18] = 33.000000
NU[19] = 160.000000
NU[20] = 1.000000
```

DDE

Spezialisierte Visualisierungsmodule

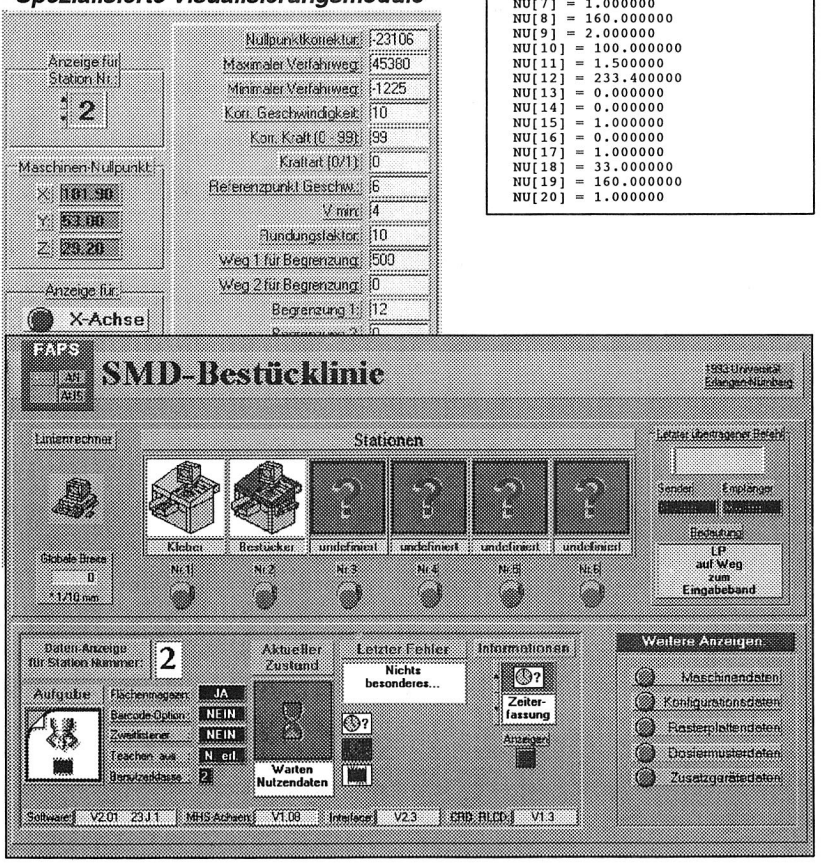


Bild 65: Spezielle Visualisierungskomponenten

Die Visualisierung eines Telegramms kann sofort nach Erhalt des Telegramms erfolgen oder sie wird durch den Benutzer über den Befehl "darstellen" (Bild 63) angestoßen. Bei der Darstellung der Telegramminhalte werden allgemeine und spezialisierte Visualisierungsmodule unterschieden. Allgemeine Module sind in der Lage, alle Telegrammtypen darzustellen. Dies entspricht einer Analyse der Telegrammdaten auf syntaktischer Ebene ohne Interpretation der Daten. Spezielle Module visualisieren nur einen oder einige wenige Telegrammtypen.

6.4 Datenverwaltung

Das Testen von Software setzt eine systematische Vorgehensweise voraus, um zum Erfolg zu führen. Neben der Testplanung, Testdurchführung und Testkontrolle stellt die Testdokumentationserstellung eine wichtige Phase während der Softwaretests dar. Die Testdokumentationserstellung umfaßt das Erfassen, Ordnen, Speichern und Bereitstellen von Daten für das Testen und/oder über den Ablauf und die Ergebnisse des Testens /34, 70, 94/. Die Testdokumentation bereits abgeschlossener Entwicklungs- bzw. Testphasen stellt die wesentliche Kommunikationsbasis der an den Testphasen beteiligten Personen dar und erleichtert die Wiederholung von Tests in der Wartung, um im Anschluß an Systempflege- und Erweiterungsarbeiten den Testaufwand zu reduzieren. Dies ist insbesondere dann sinnvoll, wenn wie im Fall der Software für einen Linienrechner eine lange Nutzungsdauer angestrebt wird und innerhalb dieser Nutzungsdauer zahlreiche Änderungen zu erwarten sind. Darüberhinaus dient die Dokumentation zum Nachweis der Durchführung der Tests.

Eine Testdokumentation beinhaltet alle Daten, die einen Testlauf eindeutig charaktierisieren. Mit Hilfe der Dokumentation kann ein Test unter den selben Bedingungen wiederholt werden. Die Dokumentation für den Test der Linienrechnersoftware mit dem Simulationssystem oder an einer realen Anlage umfaßt daher:

- Verwaltungsdaten
  Testperson, Datum, Version der getesteten Software
- Testparameter
  Konfiguration der Bestücklinien, mit denen die Tests durchgeführt wurden, Version der Bestückautomatensoftware, Rüstung und Nutzendaten
- Ergebnisse
  Telegrammprotokoll, Telegramminhalte, Fehler

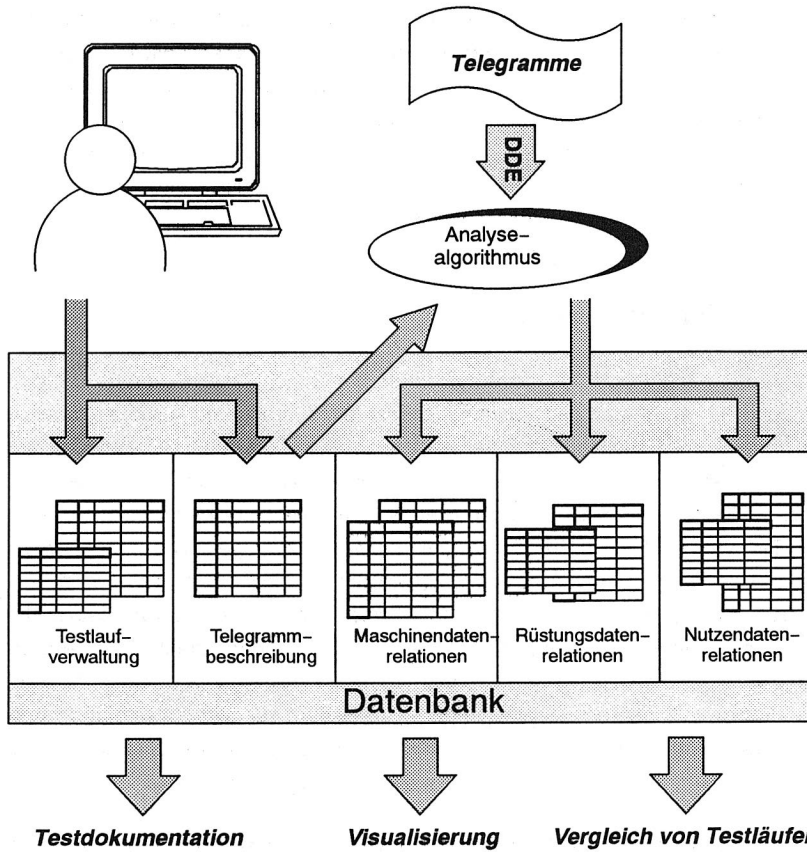


Bild 66: Testdatenverwaltung und -dokumentation mit einer relationalen Datenbank

Zur Verwaltung der Daten bietet sich auf Grund der Datenmenge der Einsatz einer relationalen Datenbank an. Ein großer Vorteil beim Einsatz einer Datenbank liegt vor allem darin, daß mit Hilfe von Report-Funktionen automatisiert und damit in kurzer Zeit schriftliche Testdokumentationen generiert werden können. Für die Linienrechnertests wurde daher basierend auf dem Windows-Datenbankprogramm ACCESS ein Verwaltungstool entwickelt, mit dem die Tests verwaltet und dokumentiert werden können. Die Datenbank besteht aus Relationen für die Verwaltungsdaten und aus

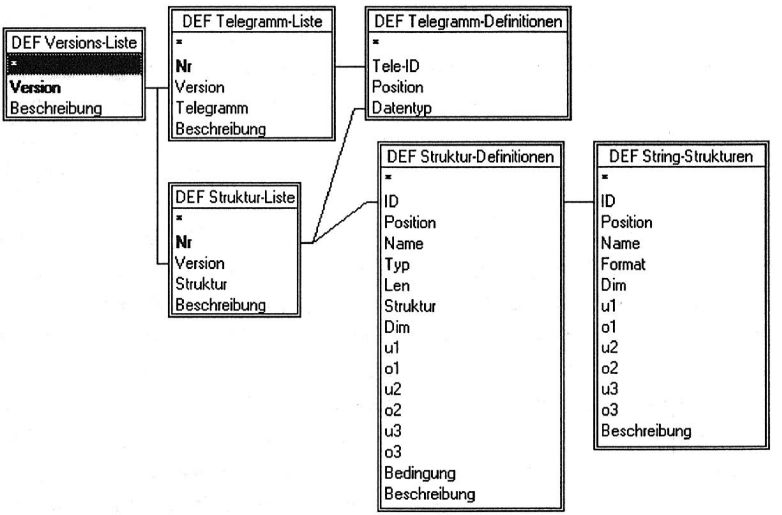


Bild 67: Struktur der Tabellen zur Spezifikation der Telegrammkomponenten

Relationen für Testparameter und Ergebnisse. Vor einem Testlauf beschreibt die Testperson durch Eingabe der Verwaltungsdaten die Art des durchzuführenden Tests und parametriert dadurch die Datenbank für die Aufnahme der Ergebnisdaten. Das Füllen der Datenbank mit den Ergebnisdaten erfolgt automatisch während der Tests durch die Anbindung über den DDE-Mechanismus an den IEC-Bus-Server (Bild 66). Dadurch wird der Dokumentationsaufwand auf ein Minimum reduziert. Die eintreffenden Telegramme werden mit Hilfe eines Analysealgorithmus', der in der Datenbank-eigenen Programmiersprache entwickelt wurde, analysiert und die Telegrammkomponenten in entsprechenden Tabellen abgelegt. Grundlage des Analysealgorithmus ist eine versionsabhängige Beschreibung des Aufbaus der Telegramme, die ebenfalls in relationalen Strukturen in der Datenbank abgelegt wurde. In Bild 67 ist die Struktur der Relationen zur Spezifikation der Telegrammkomponenten abgebildet. Zu jeder Version der Bestückautomatensoftware wird eine Liste der Telegrammtypen verwaltet. Für jeden Telegrammtyp wird eine Liste seiner Komponenten verwaltet. Dabei wurden auch optionale Telegrammteile und Datenfelder unterschiedlicher Längen berücksichtigt.

7. Ausbildung des Bedienpersonals

Kürzer werdende Produktlebenszeiten und schneller aufeinanderfolgende Innovationszyklen führen zu einer immer schnelleren Entwertung der Basisausbildung und einem stetig wachsenden Bedarf an beruflicher Weiterbildung und Nachqualifizierung von Erwachsenen /101, 7/. Daneben zeigt sich, daß neben den Produktionsfaktoren Betriebsmittel, Material und Information der Faktor Personal bei der Nutzung neuer Technologien von entscheidender Bedeutung ist, /39, 68, 100/, denn Leistung und Ertrag flexibler Fertigungssysteme hängen in starkem Maße vom Einsatz eines gut geschulten Personals ab /39, 40/. Für die möglichst störungsfreie und rationelle Fertigung wird ein Systemführer verlangt, der über die Kenntnis von Bedienoberflächen hinaus Struktur und Funktionen des Systems begreift und erst dadurch die erforderlichen Bedienoperationen wirklich beherrscht /108/. Dadurch ergeben sich erhöhte Anforderungen an die Herstellerschulung , da die Beistellehre, also das Vor- und Nachmachen, zur Führung eines flexiblen Fertigungssystems keineswegs ausreichen/15/. Eine Schulung, die sich auf die Vermittlung fachlichen Wissens und Könnens beschränkt, bleibt hinter den mit den flexiblen Fertigungssystemen gegebenen fachübergreifenden Anforderungen zurück, wie Team- und Kooperationsfähigkeit sowie situationsadäquate fachliche Verbalisationsfähigkeit. Zugleich muß der Werker in der Lage sein, sich auftragsbezogen die notwendigen Informationen selbständig zu beschaffen und zu verarbeiten.

Für das Bildungswesen bedeutet dies einen grundlegenden Wandel der Aufgaben, der Qualifikation der Lehrer und Dozenten sowie der organisatorischen Strukturen. Das Bildungssystem der Zukunft braucht eine dezentralisierte, zeitunabhängige und joborientierte Ausbildung möglichst direkt am Arbeitsplatz . Dabei wird das Selbstlernen mit dialogfähigen Kommunikationssystemen eine bedeutende Rolle spielen /47, 100/.

Ziel der Ausbildung des Bedienpersonals muß es sein, daß auf der Grundlage strukturierten Verständnisses die sichere und effiziente Führung des Fertigungssystems erreicht wird. Dazu gehört /108/:

- hohe Auslastung
- störungsfreie Bearbeitung
- optimale Produktqualität und, im Störfall, eine

- rasche Überwindung der Stillstandszeit

Gerade im Bereich der Elektronikproduktion, die geprägt ist von kapitalintensiven und hochkomplexen Fertigungsanlagen, ist eine gründliche Ausbildung des Personals unerläßlich, um die genannten Ziele zu erreichen /10/. Im folgenden werden deshalb an Hand der geforderten Lerninhalte zur Bedienung von SMD-Bestücklinien mögliche Unterrichtsformen gegenübergestellt und bewertet.

7.1 Lerninhalte zur Bedienung von Bestücklinien

Das für den Betrieb von SMD-Bestücklinien benötigte Wissen besteht aus den drei Bereichen SMD-Grundlagen (Kapitel 2), Bedienung und Steuerung sowie mechanische Tätigkeiten (Bild 68). Das Wissen läßt sich in Faktenwissen (Theorie) und Praxiswissen (Bedienung von Linienrechner und Bestückautomaten, mechanische Tätigkeiten) unterteilen.

7.1.1 Mechanische Tätigkeiten

Rüsten

Zum Rüsten eines SMD-Bestückautomaten oder einer Bestücklinie gehört die Versorgung mit Bauelementetypen, Pipetten und Adaptern, die für die Bestückung mindestens eines Flachbaugruppentyps erforderlich sind. Das Rüsten läßt sich einteilen in:

- das Abrüsten nicht mehr benötigter Bauelementetypen
- das Umrüsten von weiterhin benötigten Bauelementetypen auf einen anderen Modulplatz, Förderbereich oder eine andere Bestückstation
- das Neurüsten von Bauelementetypen.
- das Rüsten von Zusatzeinrichtungen

Zum Rüsten wird den Bauelementetypen ein zur Anlieferform (Gurt, Schüttgut, Stangenmagazin, Flächenmagazin) passender Förderer zugeordnet und dieser auf den durch die Rüstanweisung vorgegebenen Modulplatz gestellt.

Instandhaltung

Nach DIN 31051 beinhaltet die Instandhaltung "die Summe aller Maßnahmen zur Be-

Grundlagen der SMD-Bestückung	***Bedienung des Linienrechners***	***Bedienung der Bestückautomaten***
		Steuerungstätigkeiten
❑ mechanischer Aufbau der Anlagen ❑ Gehäuseformen der Bauelemente ❑ Anlieferungsformen ❑ Förderertypen ❑ Zusatzgeräte an den Bestückautomaten ❑ Steuerungsstruktur ❑ Verfahrensketten ...	❑ allgemeine Bedienung (Fenstertechnik, Menüs, Mausbedienung) ❑ Bedienung der Editoren ❑ Vorgabe und Durchführung von Aufträgen: ❑ Nutzendaten editieren ❑ Bauelementedaten editieren ❑ Rüstungen editieren ❑ Aufträge einplanen und durchführen ❑ Optimierung der Abläufe: ❑ BDE-Daten auswerten ❑ Rüst- und Umrüstoptimierung	❑ Bedienung des Stationsrechners ❑ Tastenfelder, Schalter ❑ Hochfahren der Bestückautomaten ❑ Verhalten im Fehlerfall ...
		mechanische Tätigkeiten
		❑ Rüsten von Bauelementen und Förderern ❑ Breitenverstellung des Transports ❑ einfache Wartungsarbeiten

Faktenwissen (Theorie)

Praxiswissen (Training am Linienrechner)

Praxiswissen (Training an den Bestückautomaten)

Bild 68: Lerninhalte an SMD-Bestücklinien für Bediener

wahrung und Wiederherstellung des Sollzustandes sowie zur Feststellung und Beurteilung des Istzustandes von Anlagen". Sie läßt sich in die Teilaufgaben Inspektion, Wartung und Instandsetzung einteilen.

Die Wartung bei SMD-Bestückautomaten umfaßt z. B. das Reinigen, Absaugen, Fetten, Ölen und Austauschen von Unterbaugruppen wie Zuführmodulen, Bestückkopf, Transportbänder und Antriebszahnriemen.

7.1.2 Steuerungstätigkeiten

Unter Steuerungstätigkeiten werden im folgenden alle Tätigkeiten verstanden, die an Linienrechnern, Stationsrechnern oder Bedieneinheiten durchgeführt werden. Zu diesen Tätigkeiten gehören Dateneingabe, Datenverwaltung, Automaten-Steuerung und Datenerfassung (MDE/BDE). Meist sind für Bedienung, Programmierung und Inbetriebnahme/Einrichtung verschiedene Personen zuständig. Dem entsprechend gibt es mehrere Benutzerklassen mit unterschiedlichen Privilegien und Aufgabengebieten. Dadurch kann verhindert werden, daß ungeschultes Personal Aktionen durchführt, die zu Sachschaden oder Verletzungen führen. Der Zugriff auf priviligierte Funktionen erfolgt über entsprechende Passwörter. Bei der Bedienung von SMD-Bestücklinien lassen sich gemäß der obigen Einteilung folgende Benutzerklassen festlegen:

Bediener
Dies ist die niedrigste Benutzerklasse. Hier sind nur diejenigen Funktionen zugänglich, die zum Betrieb unbedingt notwendig sind (Steuerung und Editieren von Nutzendaten).

Programmierer
Dem Programmierer stehen zusätzlich zum Bediener alle Funktionen zur Verwaltung, Erfassung und Eingabe von Daten zur Verfügung, die Rüstungen, Aufträge, Flachbaugruppen und Bauelemente umfassen.

Inbetriebnehmer/Einrichter
Dieser Gruppe hat die höchste Priviligierung. Ihr sind alle Funktionen der Maschine zugänglich, insbesondere die Veränderung von Maschinen- und Konfigurationsdaten. Diese Tätigkeiten dürfen deshalb nur von speziell geschultem Personal durchgeführt werden.

In der folgenden Tabelle sind Steuerungstätigkeiten an SMD-Bestücklinien aufgeführt, zusammen mit der Priviligierung für die einzelnen Benutzerklassen.

Tätigkeit	Bediener	Programmierer	Einrichter
Datenverwaltung			
• Datei/Diskette kopieren	●	●	●
• Diskette initialisieren	○	●	●
• Datei löschen	○	●	●
• Verzeichnis auflisten	●	●	●
Datenerfassung, MDE/BDE			
• Anzeige von Zeit- und Spurinformationen	●	●	●
• Laden und Speichern von Zeiten und Spurinformationen	○	●	●
• Rücksetzen von Zeiten und Spurinformationen	○	●	●
• Anzeige von Spurfehlern	●	●	●
• Logbuch drucken	●	●	●
• Anzeige der Passmarkenqualität	●	●	●
Automaten-Steuerung			
Steuerungs-Funktionen Linienrechner			
• Programme laden	●	●	●
• Schnittstellen öffnen/schließen	●	●	●
• Barcode-Betrieb ein/ausschalten	○	●	●
• Breite des Transportbands verstellen	●	●	●
Steuerungsfunktionen Stationsrechner			
• Bestücken ein/aus	○	○	●
• Schnittstellen freigeben/sperren	●	●	●
• Leiterplattenzähler setzen	●	●	●
• Bestückinformation ein/ausschalten	●	●	●
• Einzelfunktionsbetrieb ein/ausschalten	○	○	●
• ID-Prüfung ein/ausschalten	○	○	●
• Vision-System teachen	○	○	●
• Lageerkennung ein/auschalten	○	○	●
• Inkpunkterkennung ein/auschalten	○	○	●
• Neues Flächenmagazin	●	●	●
• Bestücken ohne Bauteil ein/ausschalten	○	○	●
Dateneingabe			
• Bauelemente-Daten ändern	○	●	●
• Rüst-Daten ändern	○	●	●
Nutzendaten			

Tätigkeit	Bediener	Programmierer	Einrichter
• Offset der Leiterplatte eingeben	●	●	●
• Auswahl des Koordinatensystems	●	●	●
• Leiterplattendaten eingeben (Länge, Breite, Höhe)	●	●	●
• Einzelschaltungsanordnung eingeben	●	●	●
• Lagererkennung aktivieren	●	●	●
• Passpunkte aktivieren	●	●	●
• Einzelschaltungen vom Bestücken ausnehmen	●	●	●
• Bestückpositionen eingeben	●	●	●
Maschinendaten			
• Daten ausdrucken	●	●	●
• Achswerte ändern	○	○	●
• Bestückkopfwerte ändern	○	○	●
• Positionen von Nullpunkten ändern	○	○	●
• Offsets von Abholspuren ändern	○	○	●
• Adapterdaten ändern	○	○	●
• Zentriererdaten ändern	○	○	●
• Daten kopieren	○	○	●
• Eingabe von Optionen	○	○	●
Konfigurationsdaten			
• Daten ausdrucken	●	●	●
• Anzahl Bestückstationen ändern	○	○	●
• Dosiermuster und Dosierzeiten für Klebestation editieren	○	○	●
• Barcode-Daten editieren	○	○	●
• Fördererdaten editieren	○	○	●
• Passwörter ändern	○	○	●
• Optionen ein/ausschalten (Barcode, CAD-Anbindung, usw.	○	○	●
• Gehäuseformdaten editieren	○	○	●

○ Tätigkeit ist für die Benutzerklasse nicht erlaubt

● Tätigkeit ist für die Benutzerklasse erlaubt

Tabelle 2: Tabelle der an einer SMD-Bestücklinie durchzuführenden Tätigkeiten

7.1.3 Ablauf von Programmierung und Steuerung

In Bild 69 sind die Tätigkeiten aufgezeigt, die von Programmierer und Bediener der Reihenfolge nach auszuführen sind, um eine Flachbaugruppe zu bestücken. Diese Tätigkeiten werden im folgenden näher betrachtet.

Ermitteln der Vorgaben
Aus der Beschreibung der Leiterplatte sind die Prozeßschritte abzuleiten, die zu durchlaufen sind. Aus den zu bestückenden Bauelementen und deren Anlieferform ergeben sich Anforderungen an die Bestückautomaten bezüglich Rüstvolumen und Zusatzeinrichtungen. Daraus kann ermittelt werden, auf welcher Bestücklinie der Auftrag gefertigt werden kann.

Datenaufbereitung
Die Beschreibung der Leiterplatte muß in das Format des Linienrechners gebracht werden. Dazu ist die Aufteilung in Einzelschaltungen als Muster oder regelmäßige Anordnung zu beschreiben und die Anpassung an das Maschinenkoordinatensystem vorzunehmen. Für die ausgewählte Bestücklinie ist eine Rüstung zu finden, in der alle benötigten Bauelementetypen enthalten sind. Dabei müssen neben der Anlieferform der Bauelemente die technologischen und benutzerbedingten Restriktionen bezüglich des Rüstortes beachtet werden. Die technologischen Restriktionen ergeben sich aus der Größe der Bauelemente und erforderlichen Zusatzeinrichtungen. Die benutzerbedingten Restriktionen umfassen Festrüstanteile und Ausschlüsse bezüglich des Rüstorts.

Dateneingabe
Die aufbereiteten Nutzendaten und die Daten zur Beschreibung der ermittelten Rüstung müssen mit Hilfe der dafür vorgesehenen Editoren am Linienrechner oder an einem dafür vorgesehenen Programmierplatz eingegeben werden. Dazu muß die Bedienung der Editoren und die Reihenfolge der Dateneingabe bekannt sein. Nach Abschluß der Dateneingabe führt der Linienrechner die in Kapitel 2.4.4 vogestellte Durchführbarkeitsanalyse durch, bei der überprüft wird, ob alle Bauelementetypen, die zu bestücken sind, in der Bauelementebibliothek und in der Rüstung enthalten sind, ob der Rüstort der Bauelemente den Restriktionen entspricht und ob die Zuordnung der Förderertypen zu den Bauelementetypen korrekt ist.

Fertigung und Prozeßoptimierung
Vor der Fertigung der Leiterplatte muß die Bestücklinie gemäß den Rüstanweisun-

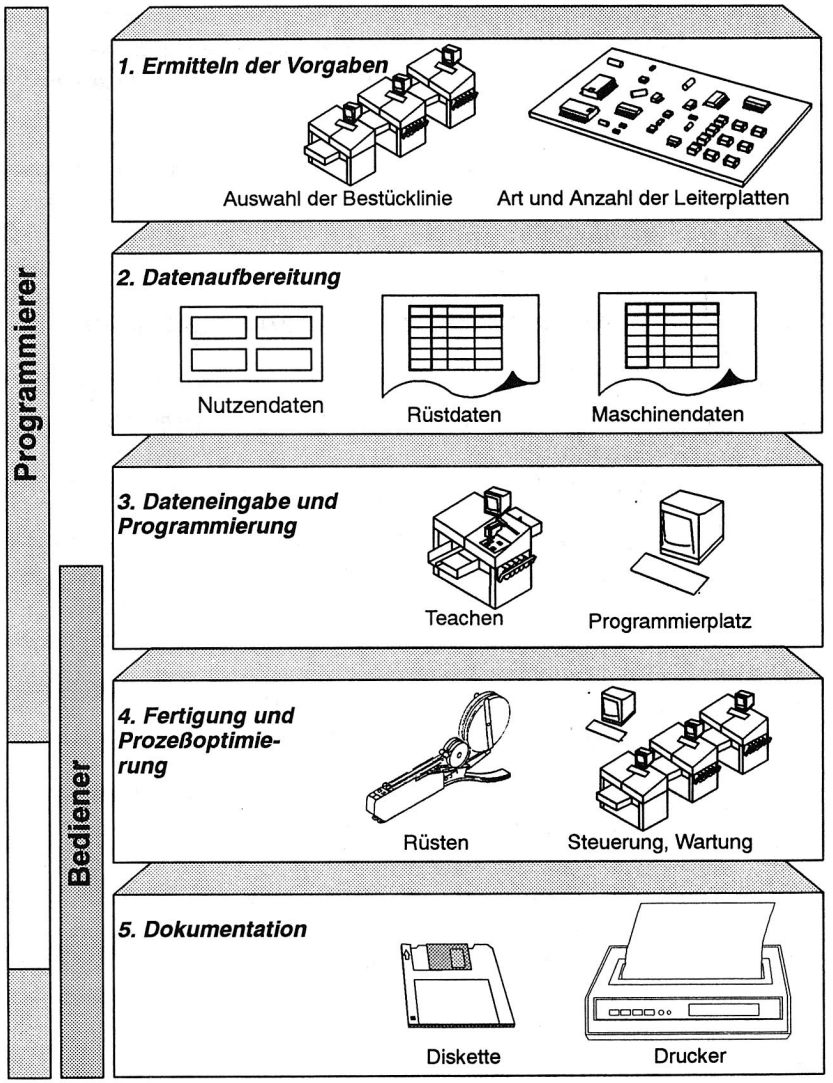


Bild 69: Tätigkeiten zur Bestückung von Flachbaugruppen

gen umgerüstet werden. Während des Bestückens können aus der Beobachtung des Prozesses Maßnahmen zur Optimierung abgeleitet werden, z. B. die Beseitigung von Schieflasten in der Bestücklinie durch Umrüstung einzelner Bauelemente.

Dokumentation
Zur Dokumentation des Fertigungsverlaufs werden Maschinen- und Betriebsdaten erfaßt, ausgewertet und abgespeichert. Die erstellten Nutzen- und Rüstdaten sollten ebenfalls gesichert bzw. archiviert werden.

7.2 Bewertung verschiedener Unterrichtsformen

Vor der Durchführung einer Bildungsmaßnahme muß die geeignetste Ausbildungsform ermittelt werden. Als Alternative zum herkömmlichen konventionellen Unterricht im Seminarbetrieb kann bei der Ausbildung an Automatisierungseinrichtungen der Einsatz von computerunterstützten Lernsystemen betrachtet werden /30/.

Kriterien zur Beurteilung der einzelnen Ausbildungsformen sind Anschaffungs- bzw. Entwicklungskosten, Unterrichtskosten, Ausfallzeiten für Personal und Maschinen, Akzeptanz bei den Lernenden, die Größe des Betriebs und damit die Zahl der Lernenden, sowie die Unterscheidung, ob ein Wissensgebiet neu gelernt oder wiederholt werden soll. Die Unterrichtskosten setzen sich aus den Lernplatzkosten (Hard- und Software-Ausstattung, Mobiliar und Betriebskosten) und den Kosten für den Unterrichtenden zusammen.

Als Unterrichtsform wird der hausinterne und der hausexterne konventionelle Unterricht sowie der computerunterstützte Unterricht (extern oder hausintern) gegenübergestellt. Zur Entscheidung zwischen computerunterstütztem Unterricht und konventionellem Unterricht wurde von H. Steppi eine Prüftabelle entwickelt (Bild 71). Die in der Tabelle gestellten Fragen sind durch Ankreuzen in der betreffenden Spalte zu beantworten. Dadurch wird jeder Antwort ein Wert zwischen 0 und 2 zugeordnet. Durch Multiplikation der Einzelwerte erhält man das Gesamtergebnis. Je deutlicher das Ergebnis über 1 liegt, desto besser ist die geplante Ausbildungsmaßnahme für computerunterstützten Unterricht geeignet. Wendet man die Prüftabelle für die Ausbildung an SMD-Bestücklinien an, ergibt sich ein Wert von 4.1, d.h. eine gute Eignung für computerunterstützten Unterricht.

Die Qualität einer Bildungsmaßnahme läßt sich an den Kriterien Akzeptanz bei den Benutzern, Umfang des erworbenen Wissens und benötigte Lernzeit beurteilen. Der

Kriterien / Unterrichtsformen	Anschaffungs-, Entwicklungskosten	Unterrichtskosten	Ausfallzeiten, Personal	Ausfallzeiten, Maschinen	Akzeptanz bei den Lernenden	Mittelständische Firma	Großbetrieb	Neu-Lernen	Wieder-Lernen
Herkömmlicher Unterricht, hausintern	◒	◒	○	◒	◒	◒	◒	◒	◒
Herkömmlicher Unterricht, extern	●	○	○	●	◒	●	◒	◒	◒
Nutzung externer Kurse auf Basis computerunterstützter Lernsysteme	●	◒	◒	●	●	●	◒	◒	◒
Hausinterne Schulung auf Basis computerunterstützter Lernsysteme	◒	●	◒	●	●	◒	◒	●	●

● Kriterium ist gut erfüllt ◒ Kriterium ist einigermaßen gut erfüllt

○ Kriterium ist schlecht erfüllt

Bild 70: Gegenüberstellung verschiedener Ausbildungsformen

Umfang des erworbenen Wissens kann durch Tests am Anfang und am Ende der Bildungsmaßnahme ermittelt werden. Tests am Anfang ermöglichen es darüberhinaus, den angebotenen Lernstoff an den Benutzer anzupassen. Schwierig zu ermitteln ist die Akzeptanz bei den Benutzern, da Menschen relativ schlecht in der Lage sind, stabile Absoluturteile abzugeben. Die Beurteilung erfolgt meist relativ zu einem beliebigen Vergleichsmaßstab. Bezüglich der benötigten Lernzeit wurde in mehreren Untersuchungen festgestellt, daß sich durch computerunterstützten Unterricht die Ausbildungsdauer auf ca. 50–60% der Zeit reduzieren läßt, die für konventionellen Unterricht benötigt wird /27, 73, 85/.

	2.0	1.8	1.6	1.4	1.2	1.0	0.8	0.6	0.4	0.2	0.0	
1. Wieviele Auszubildende gibt es insgesamt?			>1000		>500		>200		> 50			0.8
2. Wie hoch ist der Theorieanteil?	100	80	60	40	20	0%						1.6
3. Wieviele gedruckte Unterlagen müssen verwendet werden?			keine		wenige		einige		viele			0.8
4. Wie lange bleibt der Lehrstoff unverändert?			>2 Jahre		>1 Jahr		<1 Jahr					0.8
5. Ist eine dezentrale Ausbildung am Arbeitsplatz oder beim Kunden erwünscht?	ja					nein						2.0
6. Sind Spitzen und Schwankungen im Ausbildungsbedarf mit Kapazitätsproblemen zu erwarten?		ja				nein						1.8
7. Muß die Ausbildung häufig auch in einer Fremdsprache erfolgen?				ja		nein						1.4
8. Wie hoch ist das Interesse der Öffentlichkeit an dieser Ausbildung und damit die allgemeine Vermarktungschance?		hoch		mittel		null						1.0

Produkt der ermittelten Kennzahlen: **4,1**

Bild 71: Ermittlung der Eignung von CBT nach Steppi /101/

7.3 Wirtschaftlichkeit von Computer-unterstützten Lernsystemen

Die Entwicklung von Computer-unterstützten Lernprogrammen ist deutlich teurer als die Entwicklung herkömmlicher Unterrichtseinheiten für Kurse oder Seminare /31/. Während für eine Unterrichtsstunde im Seminarbetrieb 6 bis 12 Entwicklungsstunden benötigt werden, liegt der Aufwand für eine Stunde computerunterstützten Unterricht bei 30 bis 200 Stunden, je nach Ausstattung des Lernprogramms mit Graphik, Bewegtbildern, Ton und Video. In /31/ werden 40.000 DM Entwicklungskosten für eine Stunde CBT angegeben.

Als allgemeine Bedingungen für den wirtschaftlichen Einsatz von Lernprogrammen wird in /50/ und genannt:

- große Zielgruppen
- hohe Trainerkosten, begrenzte Kapazität an Trainern
- dezentrale Verteilung der Lerner
- konstante Trainingsinhalte

Zur Reduzierung der Entwicklungskosten sollten integrierte Werkzeuge zum Einsatz kommen, die sich durch Leistungsmerkmale wie die Verwaltung eines Data Dictionnary oder eines Repository und die Verfügbarkeit von Programmgeneratoren sowie Modulbliotheken auszeichnen.

Um die Frage nach dem sinnvollen Einsatz von CBT rein betriebswirtschaftlich beurteilen zu können, werden im folgenden die Kosten gegenübergestellt, die für eine Woche Unterricht im Seminarbetrieb bzw. für eine 2 Tage lange Ausbildung mit einem computerunterstützen Lernsystem bei n Auszubildenden pro Kopf anfallen. Die Betrachtung stützt sich auf die mehrfach gewonnene Erkenntnis, daß sich die Ausbildungszeiten durch den Einsatz von multimedialen Lernsystemen halbieren lassen. Berücksichtigt werden jeweils die fixen Kosten, die durch die Vorbereitung anfallen, und die variablen, Kosten die während des Kursbetriebs entstehen.

Beim Seminarbetrieb wurde von folgenden Kosten ausgegangen:

- Die Vorbereitungskosten für den Kurs sind fixe Kosten, die auf die Teilnehmer aufgeteilt werden. Je Stunde Seminar wurden 12 Stunden Vorbereitungszeit angenommen. Die Kosten für die Dozentenstunde wurden mit 200.- DM veranschlagt. Geht man weiterhin bei einer Seminarwoche von je einem halben Tag An- und Abreise aus, bleiben 4 Unterrichtsstage à 8 Stunden, zusammen also 32 Stunden.
- Die variablen Kosten je Kursteilnehmer setzen sich aus den Kosten für die Ausfallzeit (eine Woche), Reisekosten und den anteiligen Kosten für den Dozenten zusammen.

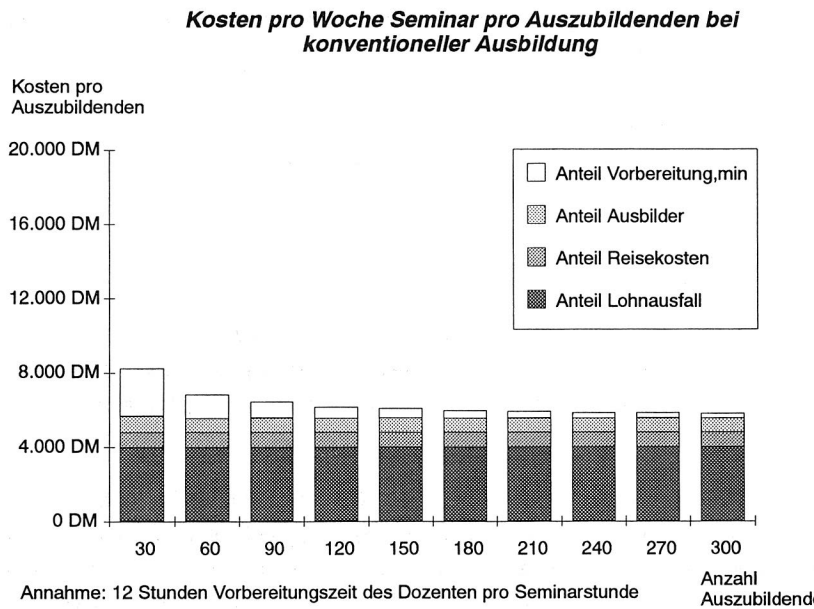


Bild 72: Kostenanteile bei Seminarausbildung

Beim computerunterstützten Unterricht wurde von folgenden Kosten ausgegangen:

- Für jede Stunde Unterricht fallen 200 Stunden Vorbereitungszeit an. Für jede Stunde werden 200.- DM veranschlagt. Die Ausbildung erfolgt vor Ort am Arbeitsplatz und dauert 16 Stunden, also zwei Arbeitstage.
- Für die benötigten Rechner wurden Abschreibungskosten berücksichtigt.

Als Ergebnis läßt sich festhalten:

- Beim Unterricht im Seminarbetrieb sind die fixen Vorbereitungskosten deutlich geringer als beim computerunterstützten Unterricht, dahingegen sind die variablen Kosten deutlich höher.
- Es gibt einen Break-Even-Punkt in Abhängigkeit von der Anzahl der Auszubildenden, ab dem computerunterstützter Unterricht günstiger ist als herkömmlicher Seminarunterricht. Dieser Punkt ist abhängig von der für den computerun-

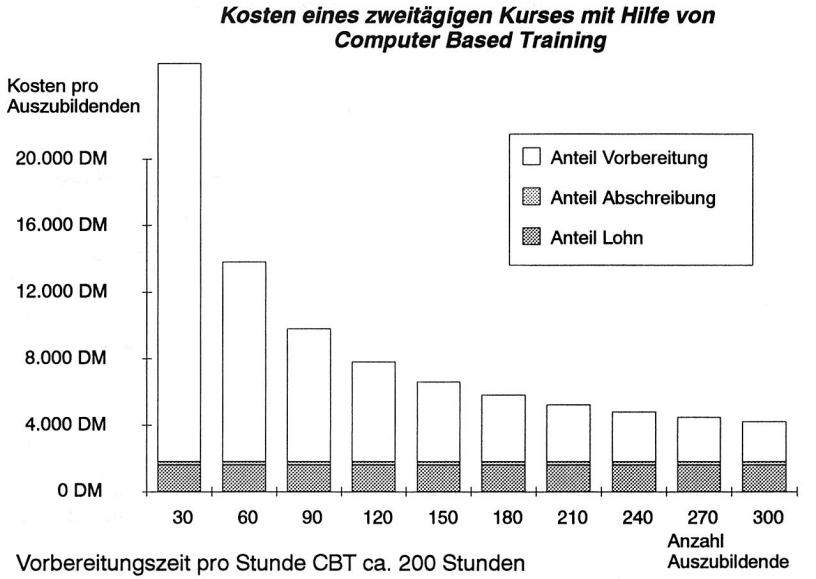


Bild 73: Kostenanteile bei computerunterstütztem Lernen

terstützten Unterricht benötigten Vorbereitungszeit. Selbst wenn man jedoch von der in /31/ angegebenen Obergrenze von 200 Stunden ausgeht, liegt der Break-Even-Punkt bei ca. 170 Teilnehmern. Durch den Einsatz von Entwicklungssystemen für computerunterstützten Unterricht läßt sich der Vorbereitungsaufwand noch deutlich reduzieren. Damit sinkt auch die Mindestanzahl der benötigten Teilnehmer. In /31/ wird die Grenze mit 100 Auszubildenden angegeben, in /87/ mit 150 Auszubildenden.

7.4 Entwicklungstendenzen und Vorteile des Computer-unterstützten Lernens

7.4.1 Entwicklungstendenzen

Computer-unterstützte Lernsysteme gibt es bereits seit den 60er Jahren. Die damaligen Möglichkeiten waren allerdings sehr begrenzt und boten lediglich "schriftliche

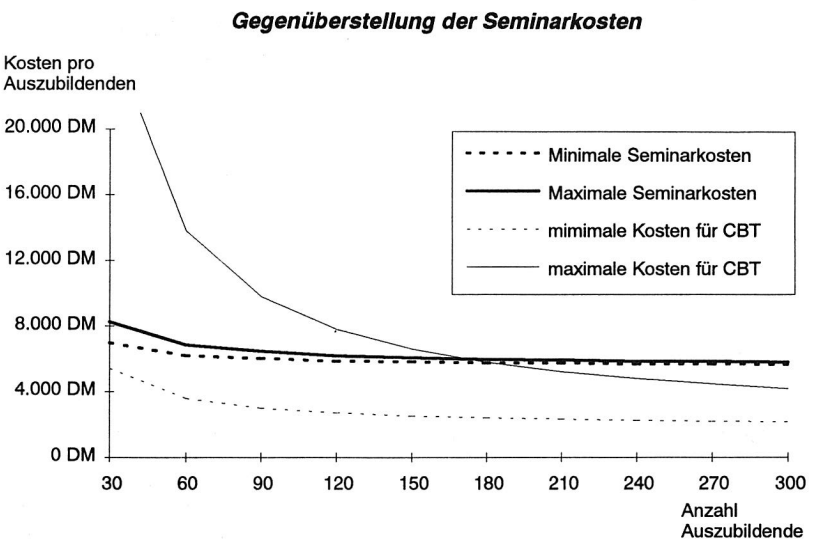


Bild 74: Kostengegenüberstellung von Seminarausbildung und Computer-unterstütztem Unterricht

Lerneinheiten" /85/. Die Arbeit an den stationären Großrechnern war geprägt durch lange Wartezeiten, als Gestaltungsmittel standen weder Grafik noch Farbe zur Verfügung /101/. In letzter Zeit hat die Bedeutung von Computer-unterstützten Lernsystemen in vielen Bereichen (Hochschulen, Industrie) zugenommen und wird auch in Zukunft noch steigen /31/. Diese Tatsache ist auf mehrere Gründe zurückzuführen:

Verkürzte Innovationszyklen

Mit der Verkürzung der Lebenszyklen von Produkten und Fertigungsverfahren kommt der fachlichen und beruflichen Aus- und Weiterbildung eine immer größere Bedeutung zu /5/. Das zur Berufsausübung benötigte Fachwissen wird immer anspruchsvoller und komplexer, ist aber trotzdem zum Teil nach wenigen Jahren wieder veraltet. In vielen Bereichen der beruflichen Bildung geht man von einer Erneuerung des Fachwissens in weniger als 10 Jahren aus /31/. Die Innovationszyklen sind zum Teil so kurz, daß eine Vermittlung des notwendigen Wissens mit einer herkömmlichen Bildungsmaßnahme wie z. B. dem Seminarbetrieb, die Dauer des Innovationszyklus übersteigen würde. So wurden z. B. bei der deutschen Bundespost mit dem

Lernprogramm "Allgemeine Geschäftsbedingungen" 40000 Schalterkräfte in einem Zeitraum von vier Monaten geschult /45/.

Mangel an Lehrkräften
Der steigende Bedarf an Aus- und Weiterbildung kann durch Wissensvermittlung über Lehrkräfte in Form von Vorlesungen und Seminaren nicht oder nur unzureichend abgedeckt werden /31/. Folglich wird es in Zukunft immer häufiger notwendig sein, sich neues Wissen selbst beizubringen. Da die Lernenden aber häufig nicht wissen, wie sie sich das Wissen logisch und chronologisch richtig erarbeiten sollen, ist eine Anleitung, z. B. in Form von Lernprogrammen, erforderlich. Unterstützt wird dieser Trend durch die zunehmende Verbreitung von Personal Computern im betrieblichen und privaten Bereich.

Steigende Kosten im Bildungsbereich
Bedingt durch die Tatsache, daß das Lernen zu einem lebenslangen Prozeß geworden ist, steigen auch die Kosten im Bildungsbereich immer mehr an. Bereits 1987 gab die Bundesrepublik Deutschland 50 Mrd. DM für das Schulwesen aus. Es müssen deshalb neue Wege gefunden werden, um bei gleichbleibender Qualität der Bildungsmaßnahmen die Kosten deutlich zu reduzieren, z. B. durch den Einsatz moderner Informationstechnologien. In /45/ und /27/ werden Beispiele aus der Ausbildung von Schalterbeamten bei der Deutschen Bundespost und bei der Ausbildung von Piloten genannt, bei denen zweistellige Millionenbeträge eingespart werden konnten.

Verbreitung von Personal Computern
Die steigende Verbreitung der Personal Computer führt dazu, daß der PC als Arbeitsgerät dort zur Verfügung steht, wo auch die Ausbildung erfolgen sollte, nämlich am Arbeitsplatz. Auf Grund der ausreichend hohen Rechenleistungen und der guten Grafikeigenschaften lassen sich Lernprogramme entwickeln, die nicht nur kognitive, sondern auch affektive und in Grenzen sogar psychomotorische Lernziele erreichbar machen. Im Gegensatz zu allen Medien, die nicht dialogfähig sind, ist der Personal Computer nicht nur als Lehrmedium, sondern auch als Führungsmedium, das den menschlichen Lehrer nachbilden muß, geeignet /101/.

7.4.2 Vorteile des Computer-unterstützten Lernens

Kennzeichnend für moderne Computer-unterstützte Lernsysteme ist der Einsatz von Multimedia (Grafik, Video, Audio) und die Möglichkeit der Interaktivität durch den Benutzer. Dies ist ein entscheidender Vorteil gegenüber konventionellen Lehrbü-

chern, da die Lernmotivation und der Grad an Aufmerksamkeit durch (bewegte) Bilder und Ton deutlich angehoben wird /85, 31/. Beim Lernen behalten die Menschen

- 25 Prozent von dem, was sie hören,
- 45 Prozent von dem, was sie sehen und hören und
- 70 Prozent von dem, was sie sehen, hören und tun

Der Einsatz von Multimedia wird begünstigt durch die angestiegene Rechnerleistung in den letzten Jahren, vor allem im PC-Bereich, und damit einhergehend durch grafische Benutzeroberflächen. Während z. B. bis vor wenigen Jahren der Einsatz von Video über Bildplattenspieler realisiert werden mußte, ist es mittlerweile möglich, Filmausschnitte zu digitalisieren, auf Festplatten zu speichern und mit einer Bildwiederholfrequenz von bis zu 20 Bildern pro Sekunde auf PC's mit normaler Grafikkarte und normalem Grafikbildschirm abzuspielen.

Die Interaktivität der Systeme kommt dadurch zum Ausdruck, daß der Lernende Zwischenfragen beantwortet, Übungen durchführt oder Tests bearbeitet. Darüberhinaus kann er Hilfen abrufen, Fragen stellen oder mit Simulationsmodellen experimentieren. Auf diese Weise hat der Benutzer auch jederzeit Kontrolle über seinen Lernfortschritt.

Weitere Vorteile des computerunterstützten Unterrichts sind /14, 29, 110/:

- die Multifunktionalität des Computers. Er präsentiert den Lernstoff, steuert durch das Lernprogramm, wertet Ergebnisse aus.
- die Möglichkeit der Simulation und damit des explorierenden Lernens
- die Schnelligkeit in der Distribution des Lehrmaterials über Datenträger oder in Zukunft verstärkt über Netzwerke
- der Direktzugriff des Lerners auf benötigte Informationen
- die unbegrenzte Wiederholbarkeit von Lektionen
- die Anpassung an die individuelle Lerngeschwindigkeit
- die Möglichkeit, jeder Zeit lernen zu können
- die Anpassung an die unterschiedlichen Wissensvoraussetzungen der Lerner
- kürzere Lernzeiten
- bessere Lernergebnisse

7.5 Varianten des Computer-unterstützten Lernens

Computer-unterstüzte Lernsysteme lassen sich hinsichtlich der Interaktion zwischen Benutzer und System in 3 Grundtypen einteilen:

CBT
Das klassische *computer based training* als älteste und verbreitetste Form von Lernsystemen läßt sich als aktives Trainingssystem klassifizieren. Die Dialogsteuerung erfolgt durch den **Lehrer**, der im Rahmen eines festen Lehr-Algorithmus schon vorab, während der Systemerstellung, alle Entscheidungen bezüglich Lehrstoff und -methode trifft. Der Benutzer besitzt keine Einflußmöglichkeiten auf den Ablauf einer Lernsitzung.

ITS (intelligent tutoring system)
Der Einsatz "künstlicher Intelligenz"-Mechanismen ist kennzeichnend für diese Art von aktiven Lernsystemen, die in allen Anwendungsbereichen zum Einsatz kommen können. Herausragendes Merkmal dieser Software ist ihre Fähigkeit zur Adaption von Lehrziel, -methode oder -zeit an den aktuellen Leistungsstand des Lerners. Es liegt also eine Dialogsteuerung durch das **System** vor.

Hypertext
Die dritte Möglichkeit der Interaktion zwischen Benutzer und System liegt darin, dem Lerner selbst die Auswahl von Stoff, Reihenfolge, Zeit und Erfolgskontrolle zu überlassen. Hypertext- bzw. Hypermedia-Systeme bieten durch ihre Struktur und Oberfläche die ideale Voraussetzung für ein individuelles Vorgehen, also eine Dialogsteuerung durch den **Lerner** /9/.

Infolge der mittlerweile realisierbaren Dialogsteuerungen und Möglichkeiten flexibler Ablaufprogrammierung von Anwendungsprogrammen erhalten Computer-unterstützte Lernsysteme mehr und mehr Charakteristika des zweiten und dritten Typs /95/.

Insgesamt können sieben verschiedene Lernsysteme in der Computer-unterstützten Ausbildung unterschieden werden /5/:

1. Hilfesysteme
Hilfesysteme sind ein wesentliches Kriterium für die Benutzerfreundlichkeit von Dialogsystemen. Sie stellen Informationen über die Handhabung und Nutzung der Soft-

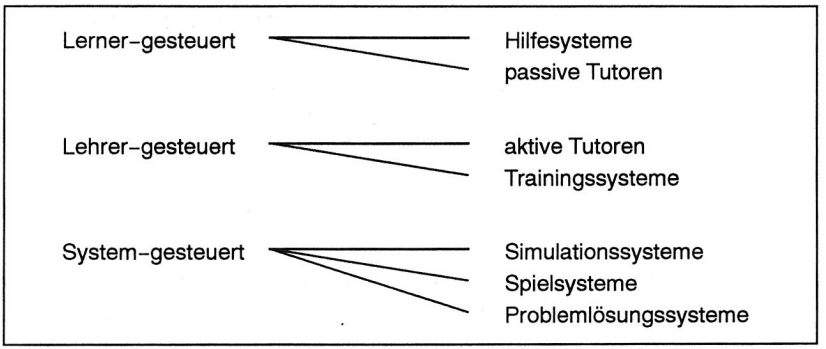


Bild 75: Klassifikation von Computerunterstützten Lernsystemen

ware bereit, um Handlungsfehler seitens des Benutzers beheben bzw. vermeiden zu helfen.

Passive Hilfesysteme werden durch eine Anfrage des Benutzers aktiviert. Der Zugang zu den im Online-Manual gespeicherten Informationen erfolgt problembezogen durch das System oder über ein Inhaltsverzeichnis durch den Benutzer.

Aktive Hilfesysteme erkennen und diagnostizieren fehlerhaftes Verhalten des Benutzers und reagieren daraufhin mit entsprechenden Hinweisen. Fortgeschrittene Formen werden als intelligente Hilfesysteme bezeichnet. Diese leiten die zur Verfügung gestellten Hilfetexte unter anderem aus der Beobachtung des Benutzers und einem daraus ermittelten Benutzermodell ab.

2. Lerner-gesteuerte Systeme

Im Unterschied zu Hilfesystemen sind Lerner-gesteuerte Systeme nicht an ein Anwendungssystem (z. B. einen Leitstand) gekoppelt, mit dem sie in dauernder Wechselbeziehung stehen. Hauptaufgabe der Lerner-gesteuerten Systeme ist die Präsentation von Lehreinheiten, aus denen der Benutzer auswählen kann. Durch den Übergang von linearer zu nichtlinearer Organisation der Informationseinheiten gelangt man zu Hypertext-Systemen, bei zusätzlicher Integration von Grafik, Audio- und Videosequenzen zu Hypermedia-Systemen. Die Benutzeroberfläche dieser Systeme sollte dem Benutzer eine effiziente Navigation im System ermöglichen und Orientierungshilfen in Form von Inhaltsverzeichnissen und grafischen Übersichten bieten.

Eine Variante von Hypermediasystemen bietet das elektronische Buch. Hier werden die Inhalte eines Lehrbuchs zusammen mit dem strukturellen Aufbau in Form einer vernetzten Datenbasis auf dem Computer bereitgestellt. Damit ist es möglich, auf bestimmte Themenbereiche direkt zuzugreifen, zusammenhängenden Stoff sequentiell zu bearbeiten, Querverweise zu verfolgen und Abbildungen einzublenden. Zusätzlich können die Interaktions- und Verarbeitungsfähigkeiten des elektronischen Mediums in Form von Rechen- und Simulationsmodellen oder bei Übungen und Tests genutzt werden.

3. Trainingssysteme

Bei Trainingssystemen werden Fähigkeiten durch "Drill and Practice" eingeübt. Dazu muß der Anwender bereits Vorwissen mitbringen da die Wissensvermittlung im Hintergrund steht. Das System stellt dem Benutzer Fragen und erwartet eine adäquate Antwort.

4. Tutorielle Systeme

Tutorielle Systeme übernehmen im Gegensatz zu den lernergesteuerten Systemen nicht nur die Präsentation der Lerneinheiten sondern führen den Lernenden auf einem bestimmten Weg durch den Stoff. Ähnlich wie bei den lernergesteuerten Systemen findet man verschiedene Arten von Lerneinheiten, beispielsweise Erklärungen, Übungen oder Tests, sowie unterschiedliche Darstellungsformen (Text, Grafik, Video, Audio)

5. Simulationssysteme

Das Hauptziel beim Einsatz von Simulationssystemen ist es, dem Lernenden ein mentales Modell eines realen Objekts oder Prozesses zu vermitteln sowie ein aktives Verwenden, Manipulieren und Testen dieses Modells vorzusehen /5/. Die Möglichkeit, entdeckend zu lernen und mit dem erworbenen Wissen zu hantieren, hebt Simulationssysteme deutlich von rein tutoriellen Systemen ab. Nach der Präsentation eines Szenarios erfolgt vom Simulationssystem ein Aktionsanstoß, der den Benutzer zu einer Aktion veranlaßt. Das System reagiert darauf und präsentiert das Ergebnis in Form eines veränderten Szenarios. Die Simulation läuft in der Regel so lange, bis ein vorgegebenes Ergebnis erreicht ist, oder der Lernende das Verfahren abbricht.

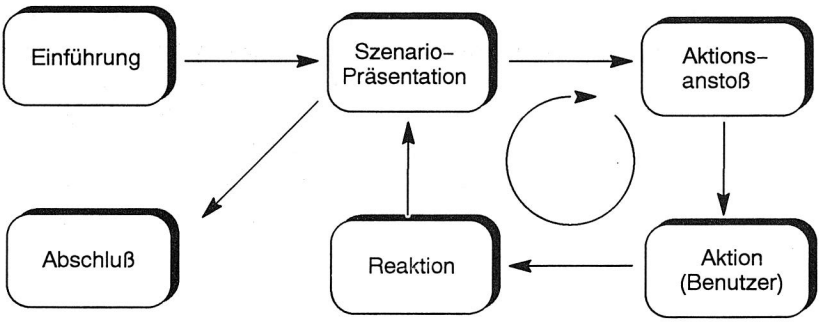


Bild 76: Grundschema des Lernens mit Hilfe der Simulation nach /5/

6. Spielsysteme

Spielsysteme haben große Ähnlichkeit mit Simulationssystemen. Der Unterschied liegt darin, daß Spielsysteme zusätzliche Anreize schaffen, um sich mit dem Lernstoff auseinanderzusetzen, z. B. durch Konkurrenz- oder Wettkampfsituationen.

7. Problemlösungssysteme

Bei Problemlösungssystemen steht die Anwendung von erworbenem Wissen zur Lösung einer bestimmten Aufgabe im Vordergrund. Das System stellt eine bestimmte Aufgabe, die der Lernende schrittweise bearbeitet. Das System beobachtet ihn dabei, beurteilt sein Vorgehen und gibt entsprechende Hinweise.

7.6 Bewertung der Varianten für den Einsatz zur Schulung an Bestücklinien

Wie in Kapitel 7.1 dargestellt wurde, müssen zur Bedienung von SMD-Bestücklinien sowohl die theoretischen Grundlagen als auch die notwendigen praktischen Fähigkeiten vermittelt werden. Gemäß der oben vorgestellten Einteilung der computerunterstützten Lernprogramme sind die tutoriellen Systeme zur Vermittlung des theoretischen Wissens am besten geeignet. Bei der Entscheidung zwischen aktiven und passiven Tutoren spielen mehrere Gründe eine Rolle. Die passiven Tutoren zeichnen sich durch einen deutlich niedrigeren Entwicklungsaufwand aus. Durch eine effiziente Navigation kann das System sowohl zum Lernen als auch zum Nachschlagen von Information (Hilfesystem) verwendet werden. Aktive Tutoren haben den Vorteil,

daß sie den Lernenden zielgerichtet durch den Stoff führen, da sie die Lerninhalte gemäß der Vorbildung des Lernenden auswählen.

Zum Erlernen der praktischen Fähigkeiten, wie der Bedienung des Linienrechners, sollten die Lernenden Übungen bearbeiten, wobei es sich anbietet, die Übungen nicht an einer realen Anlage, sondern mit Hilfe eines Simulationssystems durchzuführen. Im Bereich der CNC-Ausbildung und der PPS-Ausbildung wird dies bereits praktiziert /99, 100/. Als Szenario wird dem Lernenden eine Fertigungsumgebung präsentiert, mit der er einen vorgegebenen Auftrag durchzuführen hat (Aktionsanstoß). Nach Durchführung der notwendigen Bedientätigkeiten, wie Dateneingaben und Freigabe von Aufträgen (Aktion des Benutzers), übernimmt das Simulationssystem die Aufgabe einer realen Anlage (Reaktion) und liefert die entsprechenden Meldungen an den Linienrechner.

8. Das Lernprogramm HSLERN

8.1 Entwicklung von Computer-unterstützten Lernsystemen

Für die Entwicklung von Computer-unterstützten Lernsystemen ("Teachware Engeneering") kann eine erweiterte Systematik der Entwicklung von Software-Systemen verwendet werden /5/. Softwareprojekte lassen sich in die Phasen Problemanalyse, Anforderungsdefinition, Entwurf, Implementierung, Test, Betrieb und Wartung unterteilen /82, 87/. Bei Computer-unterstützten Lernsystemen wird zwischen Anforderungsdefinition und Entwurf eine weitere Phase eingeschoben, die sich mit dem pädagogischen Design des Systems befaßt. Während der Test- und Betriebsphase sollte mit Hilfe einer Evaluation der Zielerreichungsgrad des Systems ermittelt werden (Bild 77).

Bei der Entwicklung ist es vorteilhaft, das Prinzip des Prototyping anzuwenden. Dadurch ist es möglich, dem späteren Anwender frühzeitig Benutzeroberflächen und Interaktionsmöglichkeiten vorzuführen, um auf dessen Vorstellungen und Wünsche eingehen zu können /95/. Die Folge ist, das einzelne Projektphasen mehrfach durchlaufen werden müssen.

Als Entwicklungsumgebung für Computer-unterstützte Lernsysteme können herkömmliche Programmiersprachen, Autorensprachen oder Autorensysteme /37/ verwendet werden. Autorensprachen sind höhere Programmiersprachen, die spezielle Anweisungen zum Erstellen von Lernsystemen zur Verfügung stellen. Die Formulierung des Lernprogramms erfolgt damit in viel knapperer Form als in einer normalen Programmiersprache. Der Schwerpunkt des Programmieraufwands liegt in der Spezifikation, bei welchen Benutzerantworten das System wie reagieren soll. Autorensysteme unterscheiden sich von Autorensprachen durch eine sehr intensive Unterstützung und Führung des Benutzers bei der Lehrprogrammerstellung.

Als Beispiel für ein solches Autorensystem sei das Programm Toolbook genannt, das auf PC's mit der Benutzeroberfläche MS-Windows läuft. Mit diesem Programm können elektronische Bücher erzeugt werden. In diesen Büchern ist es wie in einem normalen Buch möglich, auf bestimmte Themenbereiche direkt zuzugreifen, zusam-

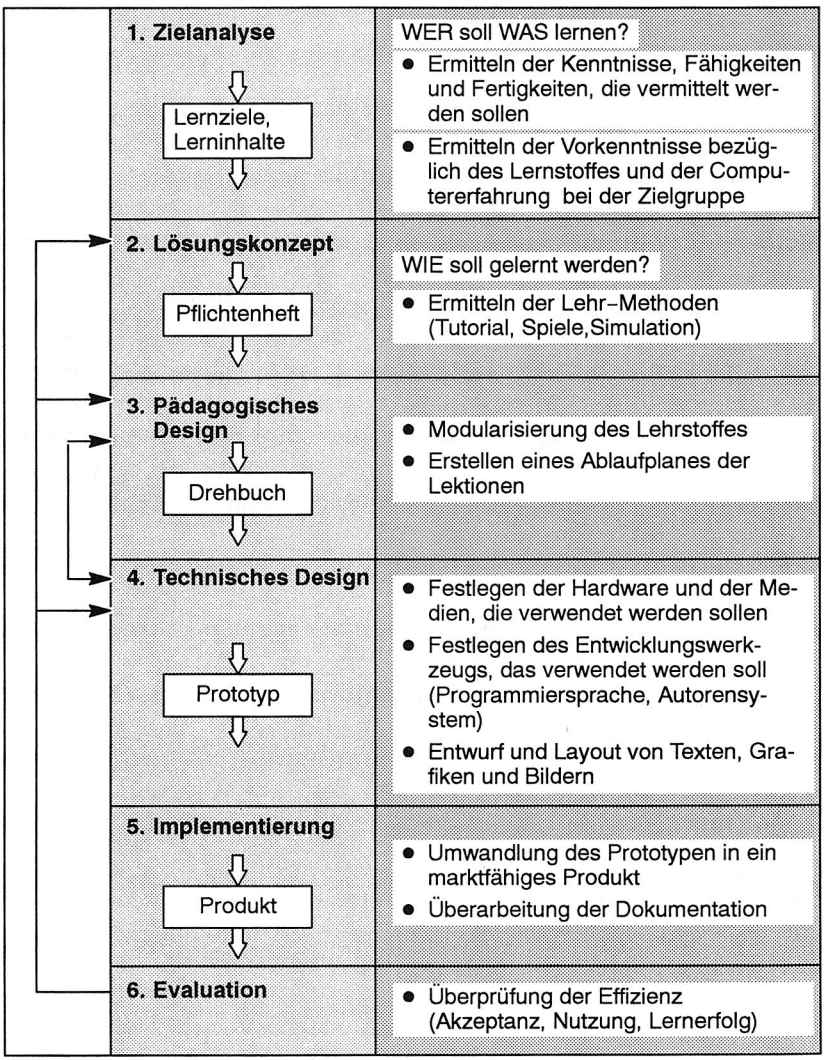


Bild 77: Phasen der Entwicklung von Lernsystemen

menhängenden Stoff sequentiell zu bearbeiten, Querverweise zu verfolgen und Abbildungen einzublenden.

Jedes Buch besteht aus einzelnen Seiten, auf denen sich grafische oder textuelle Objekte befinden. Jedem dieser Objekte können Aktionen zugeordnet werden, die vom Benutzer durch Tasten- oder Mausaktionen ausgelöst werden. Aktionen können das Umblättern auf eine neue Seite sein, das Aktivieren von Animationen, das Auswerten von Benutzereingaben oder die Kommunikation mit anderen Programmen.

Die Entwicklung von Lernsystemen ist ein interdisziplinäres Vorhaben, bei dem Repräsentanten aus verschiedenen Bereichen eng zusammenarbeiten müssen. Benötigt werden Experten aus dem eigentlichen Fachgebiet (Einbringen der Lerninhalte), aus dem Bereich Pädagogik, Mediendidaktik, Psychologie (Kursgestaltung, Medienmix und Drehbücher), aus dem Bereich Informatik, Grafik, Design, Psychologie (Programmumsetzung, Oberfläche, Interaktion) und aus dem Bereich Psychologie und Wirtschaftswissenschaften (Evaluation).

8.2 Programmstruktur

Das entwickelte Lernprogramm für die Bedienung des Linienrechners ist in mehrere "Bücher" aufgeteilt (Bild 78), die jeweils ein Lernkapitel enthalten. Der Einstieg erfolgt über ein Inhaltsverzeichnis, in dem der Lernstoff hierarchisch in Lektionen, Abschnitte und Kapitel unterteilt ist. Jede Lektion enthält ca. 8–12 Abschnitte, jeder Abschnitt 2–5 Kapitel. Die mittlere Bearbeitungszeit eines Kapitels beträgt ca. 20 Minuten. Dieses Zeitmaß entspricht der durchschnittlichen Dauer an Konzentrationsfähigkeit, die ein Lernender ohne Unterbrechung aufbringen kann /49/. Im Inhaltsverzeichnis kann der Lernende die bearbeiteten Kapitel markieren und erkennt so den noch verbleibenden Lernstoff.

Jedes Buch und somit jedes Lernkapitel besteht aus einer Verzweigungsseite, sowie Lern- Übungs- und Testteil (Bild 79). Die Seiten von Lern-, Übungs- und Testteil sind immer gleich aufgebaut und bestehen aus fünf Fenstern (Bild 80):

a) Orientierungsfenster
Der Lerner soll durchgehend wissen, an welcher Stelle im Lernstoff er sich befindet. Das Orientierungsfenster zeigt dazu die aktuelle Lektion, den Abschnitt und das Ka-

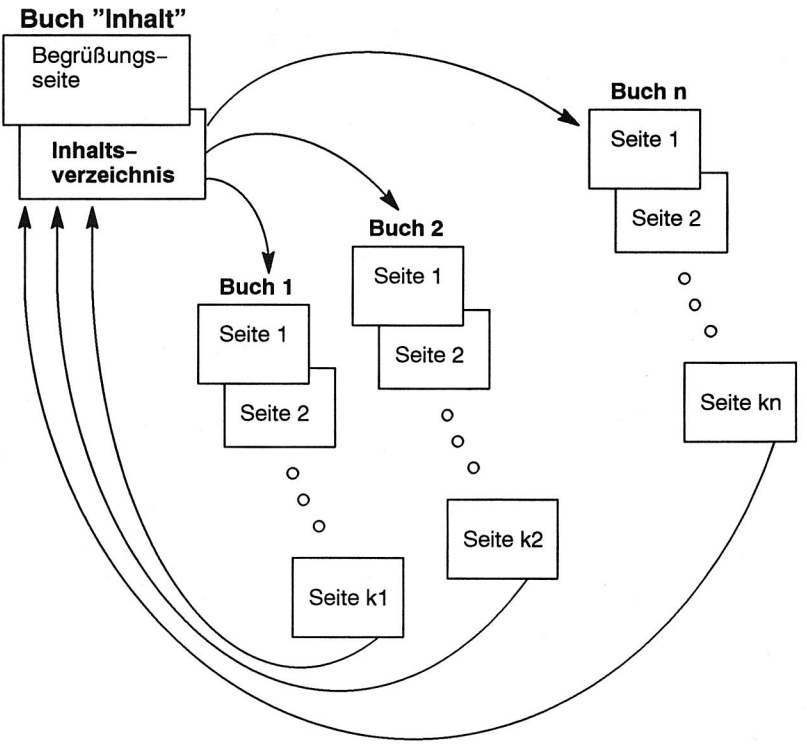


Bild 78: Aufbau des gesamten Lernprogramms

pitel an. Darüberhinaus wird auch angezeigt, ob sich der Lernende im Test- im Übungs- oder im Testteil befindet.

b) Steuerfenster

Mit Hilfe des Steuerfensters kann sich der Lernende frei und mit selbst festgelegtem Tempo durch das Lernprogramm bewegen. Der Zugriff auf den nächsten oder den vorherigen Lernschritt erfolgt durch Vorwärts- oder Rückwärtsblättern mit Hilfe der Pfeiltasten. Ein Abbruch der Lerneinheit verbunden mit einem Rücksprung in das Inhaltsverzeichnis ist ebenfalls möglich.

c) Lerntextfenster

Das Lerntextfenster enthält den Lernstoff in textueller Form. Um eine Informationsü-

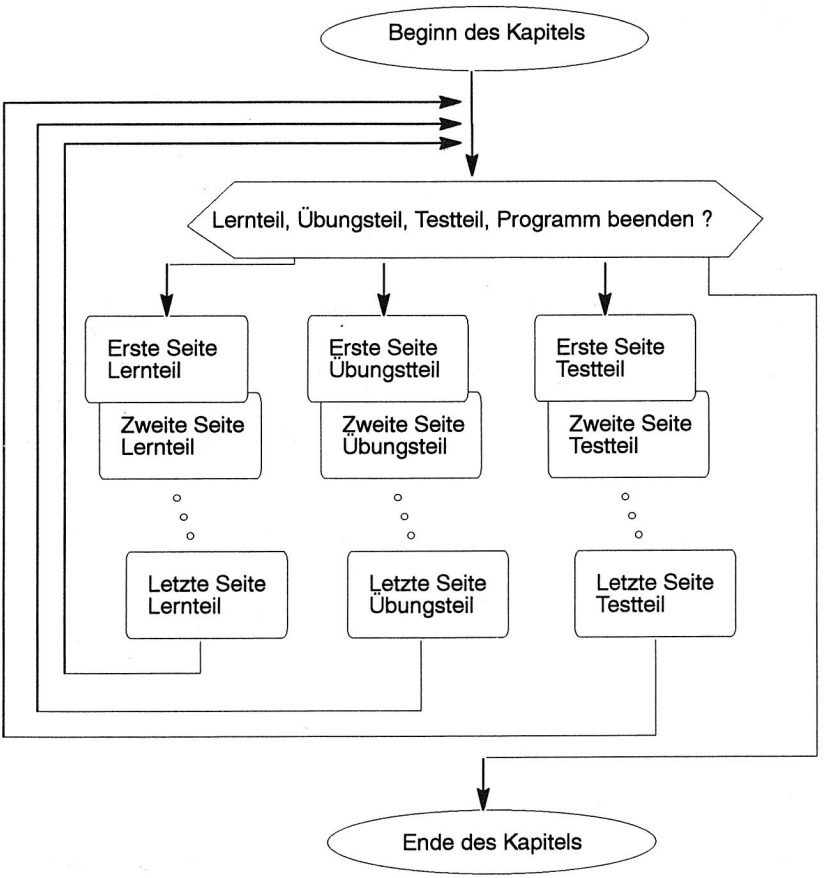


Bild 79: Struktur der einzelnen "Bücher"

berhäufung des Lernenden zu vermeiden, werden pro Seite nicht mehr als 10 Textzeilen dargestellt.

d) Informationsfenster

Das Informationsfenster stellt dem Lernenden zusätzliche Informationen zu einer Situation zur Verfügung und wird hauptsächlich im Übungs- und Testteil verwendet. Hier werden z. B. Hinweise zum bearbeiten der gestellten Aufgaben gegeben.

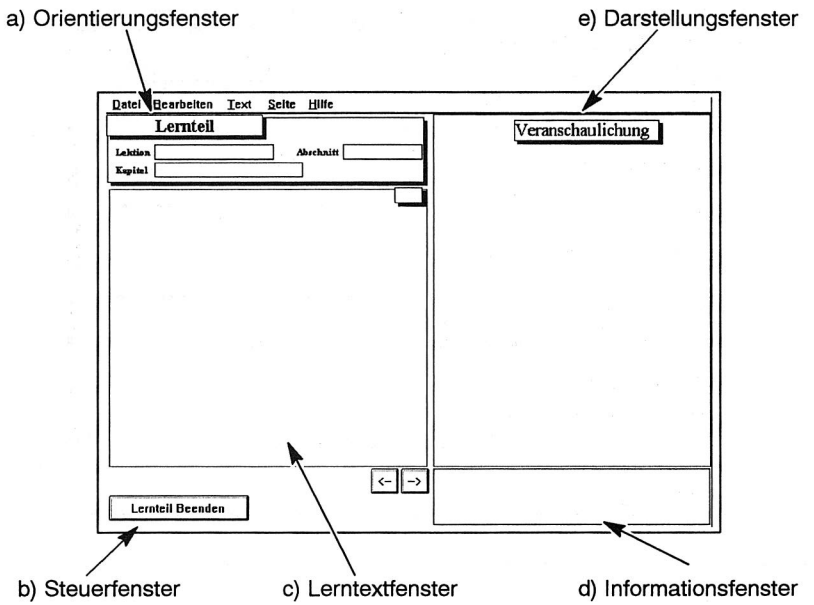


Bild 80: Aufbau der Seiten im Lernprogramm

e) Darstellungsfenster
Im Darstellungsfenster werden statische und bewegte Grafiken zur Veranschaulichung der textuellen Information angezeigt.

Der **Lernteil** vermittelt dem Lerner Wissen mit Hilfe von textuellen Darstellungen im Lerntextfenster. Dabei wurden allgemein gültige Gestaltungsregeln beachtet, nach denen eine Aufteilung in möglichst kleine Abschnitte erfolgen sollte und bei Aufzählungen nicht mehr wie sieben Items aufgeführt werden sollen. Zur Veranschaulichung des beschriebenen Sachverhalts werden im Darstellungsfenster Grafiken eingeblendet.

Der **Übungsteil** bietet die Möglichkeit, das Gelernte einzuüben. Dazu werden dem Lernenden Aufgaben gestellt, die in direktem Bezug zum vorangegegangenen Lernteil stehen. Die Darstellung der Aufgabe erfolgt in textueller Form im Lerntextfenster. Zur Lösung der Aufgabe müssen Benutzer-Eingaben in den im Darstellungsfenster abgebildeten Linienrechnermenüs vorgenommen werden. Diese Eingaben beste-

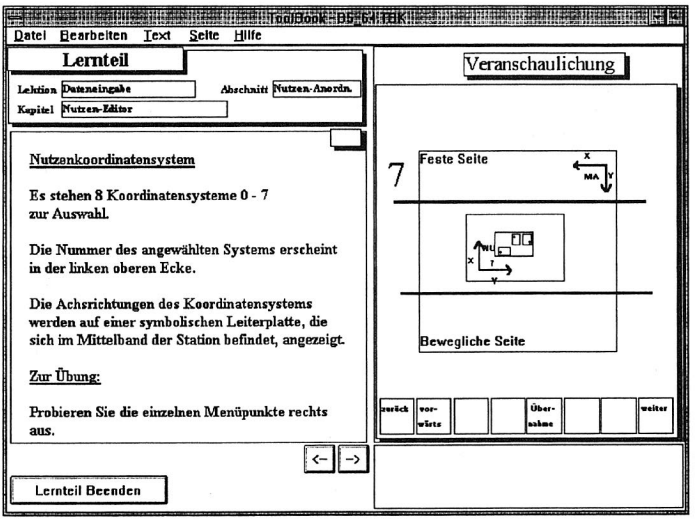


Bild 81: Beispiel einer Lernseite

hen aus dem Eintragen und Löschen von Werten und Ausdrücken und dem Navigieren durch die Menüs. Durch betätigen des Korrekturknopfes werden die Eingaben auf Korrektheit überprüft und das Ergebnis im Informationsfenster dargestellt. Kann der Lerner das Ergebnis nicht selbständig ermitteln, kann er sich die richtige Lösung am Bildschirm anzeigen lassen.

Der **Testteil** besteht aus einer Reihe gezielter Fragen zum Lernteil in Form von Multiple-Choice-Fragen oder Fragen mit freier Texteingabe. Die Darstellung der Frage erfolgt im Lerntextfenster. Hier erfolgt auch die Beantwortung durch Anklicken der Auswahlfelder mit der Maus oder durch Eingabe der Antwort in die Eingabefelder. Die Überprüfung der Antwort und das Darstellen der richtigen Antwort erfolgt wie im Übungsteil durch betätigen der entsprechenden Knöpfe.

Im Normalfall sollte der Lernende Lern-, Übungs- und Testteil Seite für Seite bearbeiten. Darüberhinaus ist aber auch von jeder Seite aus der Rücksprung zur Verzweigungsseite und damit der Abbruch des Lernens möglich.

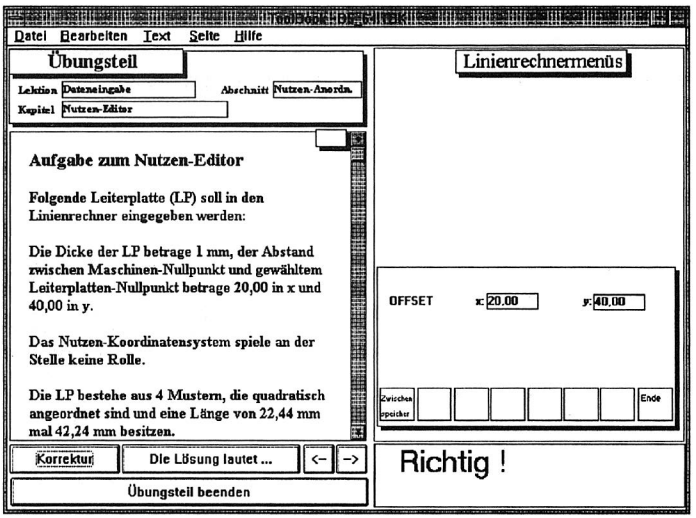


Bild 82: Beispiel einer Übungsseite

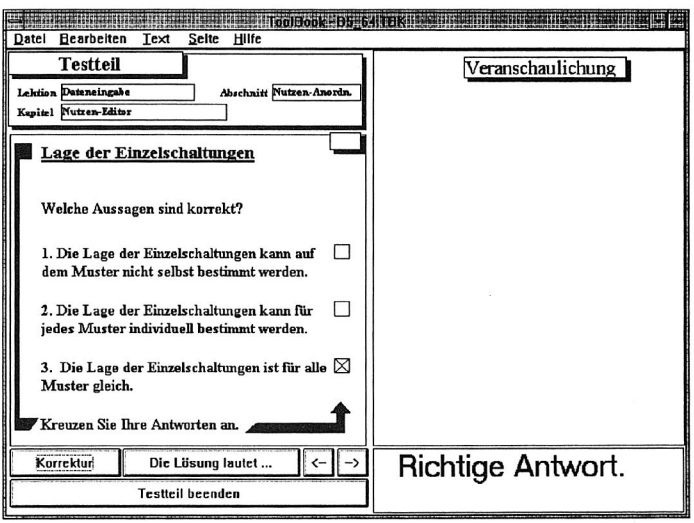


Bild 83: Beispiel einer Testseite

Durch den gewählten Aufbau des Lernprogramms wird eine einfache Erweiterbarkeit gewährleistet. Das Programm kann inhaltlich durch neue Bücher oder durch Erhöhung der Seitenzahl in den einzelnen Büchern erweitert werden.

9. Bewertung der vorgestellten Systeme

9.1 Vorteile des Simulationssystems in den Testphasen

Ein wichtiges Ziel bei der Entwicklung von Steuerungssoftware ist die Entwicklung eines qualitativ hochwertigen Produkts unter Einhaltung gegebener Termine und geplanter Kosten. Der Anwender legt dabei vor allem Wert auf die Qualitätsmerkmale Zuverlässigkeit, Korrektheit und Benutzerfreundlichkeit, während für den Hersteller die Frage der Wartbarkeit und dem optimalen Freigabezeitpunkt entscheidend ist. Gerade die Entwicklungsdauer eines Softwareprodukts ist eine kritische Einflußgröße im Hinblick auf den wirtschaftlichen Erfolg. Verspätet auf den Markt kommende Produkte führen ebenso zu einem Imageverlust wie zu frühe Freigaben, da in diesem Fall verstärkt Fehler beim Benutzer auftreten. Neben rein kalkulatorischen Argumenten ist der nicht quantifizierbare Imagegewinn durch qualitativ hochwertige Software und die damit verbundene Stabilisierung der Stellung gegenüber den Marktkonkurrenten ein nicht zu unterschätzender Faktor /54/.

Auf Grund steigender Anforderungen an die Bestücksysteme und der damit verbundenen Weiterentwicklungen spielt die Wartbarkeit der Steuerungssoftware eine große Rolle. Wartung beruht dabei nicht wie im herkömmlichen Sinn auf Verschleiß, sondern umfaßt alle Eingriffe in die Software während der Nutzungsphase, die der Erweiterung, Änderung, Stabilisierung oder Optimierung dienen. Durch die Wartungstätigkeiten entstehen neue Software-Versionen, wobei üblicherweise zwischen *Minor Releases* und *Major Releases* unterschieden wird. Minor Releases unterscheiden sich meist nicht oder nur geringfügig in der Bedienung und dem Funktionsumfang, sondern nur durch Stabilisierungsmaßnahmen, d. h. dem Entfernen von Fehlern. Major Releases hingegen beruhen auf Erweiterungen oder Änderungen, wodurch sich der Funktionsumfang und/oder die Bedienung der Software ändert. Die Zahl der Minor Releases sollte möglichst klein gehalten werden, da ein häufiger Versionenwechsel, der nur auf beseitigten Fehlern beruht, dem Ansehen des Herstellers schadet. Jede derartige Softwareversion ist darüberhinaus auch mit Kosten verbunden, da sie ebenfalls wieder getestet werden und bei den bereits ausgelieferten Systemen installiert werden muß.

Durch den Einsatz des vorgestellten Simulationssystems HSSIM werden nun folgende Vorteile erzielt:

Früherer Beginn der Tests

Durch den Einsatz der Simulation läßt sich der Beginn der Testphasen zeitlich nach vorn verlegen, da die Tests unabhängig vom Entwicklungsstand der Kommunikationspartner des Linienrechners, also der Bestückautomaten, durchgeführt werden können. Durch die einfache Parametrierbarkeit des Systems erfolgt eine schnelle Anpassung an ein geändertes Protokoll. Darüberhinaus kann der Entwickler die Tests direkt an seinem Arbeitsplatz in seiner Entwicklungsumgebung durchführen, wo er durch Debugger und Testumgebungen bei der Lokalisierung und Behebung von entdeckten Fehlern unterstützt wird.

Höhere Anzahl gefundener Fehler

In Bild 84 wird die Anzahl der Fehler gegenübergestellt, die in der Testphase und der Betriebsphase mit bzw. ohne Unterstützung durch Simulation gefunden werden. Zu Beginn einer jeweiligen solchen Phase ist die Anzahl der gefundenen Fehler relativ groß und steigt nach einiger Zeit nur noch geringfügig an. Durch den Einsatz des Simulationssystems werden mehr Fehler in der Testphase gefunden und folglich sinkt die Anzahl der Fehler, die erst nach der Auslieferung bei den Kunden auftreten. In Verbindung mit dem früheren Beginn der Testphase wird die Marktreife zu einem früheren Zeitpunkt erreicht. Da das Produkt darüberhinaus weniger Fehler enthält, entfallen einige der ansonsten notwendigen Minor Releases mit den damit verbundenen Kosten.

Langzeittests

Viele Fehler zeigen sich erst dann, wenn die Software lang genug ununterbrochen läuft. Die Ursachen liegen z. B. in zu klein dimensionierten Zählern oder Feldern, oder der Speicher reicht nicht mehr aus, da dynamische Strukturen nicht ordnungsgemäß wieder freigegeben werden. Derartige Fehler werden häufig nicht in der Testphase entdeckt, da die Testzeiten auf Grund der Kosten und des personellen Aufwands kurz gehalten werden. Auch hier schafft das Simulationssystem Abhilfe, da kostengünstig und durch Zeitraffung längere Tests in kürzerer Zeit durchgeführt werden können. Der Vorteil liegt hier insbesondere darin, daß die Tests auch ohne manuellen Eingriff, und damit über Nacht oder an den Wochenenden laufen können, im Gegensatz zu Tests an einer realen Anlage, bei der es auf Grund von Störungen oder fehlendem Material zu Stillständen kommt, die nur manuell zu beheben sind.

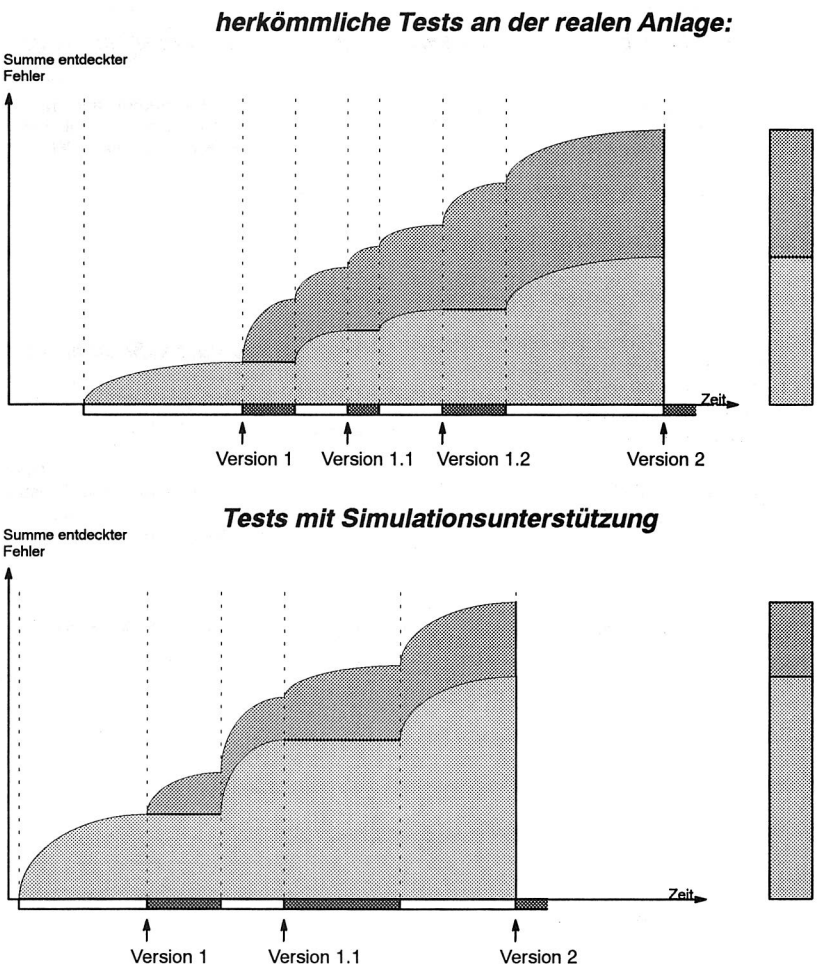


Bild 84: Fehlererkennung mit und ohne Simulationsunterstützung

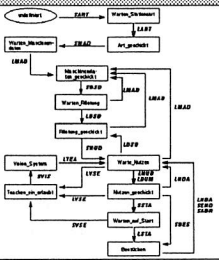
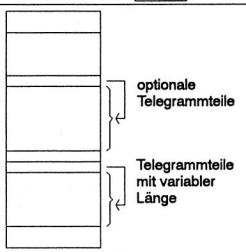
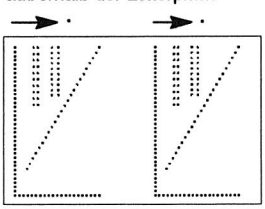
Fehlerart	Auswertung	Fehlererkennung
Linienrechner sendet falschen Telegrammtyp		Das Telegramm kann im aktuellen Zustand des abstrakten Automaten nicht verarbeitet werden.
Linienrechner sendet Telegramme mit falscher Struktur: a) Telegrammteile fehlen b) Telegrammteile sind vertauscht c) Telegramm enthält überflüssige Telegrammteile		***Syntaktische Analyse*** ☞ Vergleich mit vorgegebenen Telegrammen ☞ Parsen der Telegramme, Vergleich von Schlüsselwerten, die den weiteren Telegrammaufbau festlegen. (Flags für optionale Teile und Angaben von Feldlängen)
Linienrechner sendet Telegramme mit falschen Daten	Fehler: Bestückposition außerhalb der Leiterplatte 	***Semantische Analyse*** ☞ Interpretation und Visualisierung der Daten. ☞ Vergleich mit vorgegebenen Daten

Bild 85: Protokollfehler und Fehlererkennung

Höhere Transparenz

In Verbindung mit den in Kapitel 5 vorgestellten Visualisierungsmöglichkeiten wird eine höhere Transparenz der zwischen Linienrechner und Stationsrechnern ausgetauschen Daten erreicht, und somit das Erkennen von Protokollfehlern ermöglicht. Die in Bild 85 aufgelisteten Fehler können an Hand der Auswertungsmöglichkeiten des Systems erkannt werden.

Kostenersparnis

Die Kostenersparnis ist ein weiterer wichtiger Grund für den Einsatz des vorgestellten Simulationssystems. Während bei den herkömmlichen Tests eine komplette Bestücklinie eingesetzt werden mußte, genügt nun ein handelsüblicher PC. Da außer dem Einbau einer IEC-Bus-Karte keine besonderen Anforderungen an die Ausstattung des PC's zu stellen sind, kann ein eventuell bereits vorhandenes Gerät verwendet werden. Zur Quantifizierung der Kostenersparnis wurden die Kosten, die durch den Einsatz einer Bestücklinie entstehen, den Kosten bei Einsatz des Simulationssystems gegenübergestellt (Bild 86). Bei einer Bestücklinie (Wert ca. 2.000.000 DM) fallen neben den Kapitalbindungskosten Mietkosten für den Stellplatz sowie Kosten für Verbrauchsmaterialien (Bauelemente, Lotpaste, Leiterplatten) und Personalkosten zur Betreuung der Anlage während der Tests an. Demgegenüber stehen die Anschaffungskosten für einen PC (ca. 6000.- DM) und die Kosten für die Anpassung der Simulationssoftware an neue Versionen der Steuerungssoftware.

Höhere Anzahl von Testfällen

Die Tests an der realen Anlage lassen sich aus Kostengründen nur an einer oder einigen wenigen unterschiedlichen Linienkonfigurationen durchführen. Dabei wird insbesondere das Austesten von Grenzfällen, bei denen maximal mögliche Ausbaustufen von Bestückautomaten und/oder Bestücklinien realisiert sind, aus Kostengründen vernachlässigt. Der Vorteil bei Einsatz des Simulationssystems liegt nun in der einfachen Parametrierbarkeit und damit in der Anpaßbarkeit an unterschiedliche Bestückautomaten- und Bestücklinienkonfigurationen. Dadurch läßt sich die Zahl der Testfälle, also die Zahl der unterschiedlichen Linien, an denen die Software getestet wird, deutlich erhöhen. Darüberhinaus wird auch das Austesten der oben beschriebenen Grenzfälle problemlos ermöglicht.

Simulation von Störungen

Ein umfassender Test der Steuerungssoftware muß vor allem auch die Überprüfung des Verhaltens der Software bei Auftreten von Störungen an der Anlage beinhalten. Hier bietet das Simulationssystem die Möglichkeit jederzeit, und beliebig wiederholbar derartige Störungen zu generieren.

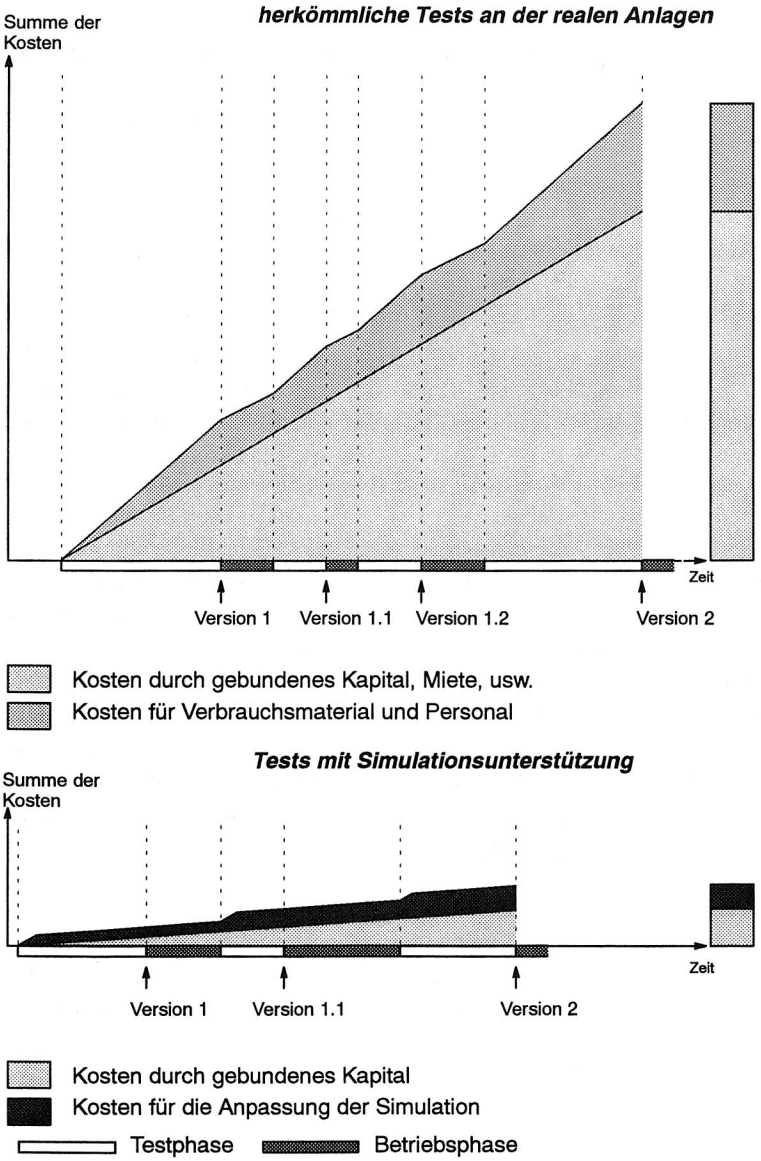


Bild 86: Kostenersparnis durch Anwendung des Simulationsystems

9.2 Vorteile des Simulationssystems während der Schulung

Wie in Kapitel 6 beschrieben, hängen Leistung und Ertrag von flexiblen Fertigungssystemen in starkem Maße vom Einsatz eines gut geschulten Personals ab. Es ist deshalb die Aufgabe des Herstellers, geeignete Ausbildungsmaßnahmen anzubieten, die sowohl das zur Bedienung notwendige theoretische Wissen, als auch die Praxis im Umgang mit den Systemen vermitteln.

Gemäß der Einteilung in Benutzerklassen unterschiedlicher Priviligierung (Bediener, Programmierer, Einrichter) werden von den Herstellern Kurse angeboten, die jeweils genau das Aufgabenspektrum der jeweiligen Benutzerklasse abdecken. Jeder Anwender benötigt zum Betrieb seiner Bestücklinien Personal, das zumindest über das Wissen von Programmierern und Bedienern verfügt (Kapitel 7.1.2). Bei Anwendern, die mehrere Bestückautomaten im Einsatz haben, werden Programmierung und Bedienung meist von verschiedenen Personen durchgeführt. Der Programmierer, der die Aufträge für mehrere Bestücklinien erstellt, hat dabei die Aufgabe, durch geeignete Zuordnung der Aufträge zu den Bestücklinien und durch eine geeignete Auftragsreihenfolge, den Durchsatz der Linien zu optimieren und den Rüst- und Umrüstaufwand zu minimieren. Durch entsprechende Rüst- und Umrüstkomponenten wird er dabei vom Linienrechner unterstützt. Bei Anwendern mit einer oder nur wenigen Bestücklinien wird Programmierung und Bedienung von den selben Personen durchgeführt. Die Anzahl der Bestücklinien, ab der der Einsatz eines Programmierers wirtschaftlich sinnvoll ist, hängt von mehreren Faktoren ab. Entscheidende Kriterien sind die Größe der zu bearbeitenden Lose, der Umfang des Produktspektrums und die Produktlebensdauern, da davon die Anzahl der Programmier- und Planungstätigkeiten abhängen.

Neben Programmierung und Bedienung sind regelmäßige und auf Störungen beruhende Wartungsarbeiten durchzuführen. Die Schulung von Wartungsarbeiten erfolgt ebenfalls in Kursen, wobei Kurse für einfachere Wartungsarbeiten, die relativ häufig durchzuführen sind, und Kurse für schwierigere Wartungsarbeiten angeboten werden. Ähnlich wie bei den Programmiertätigkeiten genügt es, wenn bei einem Anwender von mehreren Bestücklinien nur eine oder wenige Personen über das notwendige Wissen über schwierigere Wartungsarbeiten verfügen. Bei der Schulung der Wartungsarbeiten ist es wichtig, durch praktisches Training an einer realen Maschine die notwendigen Handgriffe zu erlernen.

Das Einsatzgebiet des Simulationssystems ist also im Bereich der Schulung für Bediener und Programmierer zu sehen. Durch die zunehmende Komplexität der Steuerungssoftware wächst die Bedeutung der Tätigkeiten, die von Bediener und Programmierer für einen optimierten und störungsfreien Betrieb der Anlagen auszuführen sind. Dementsprechend wächst auch die Bedeutung der Schulung für Programmier- und Bedientätigkeiten. Neben der theoretischen Einweisung spielt dabei das "Learning-by-doing", also die praktische Erfahrung mit dem System, eine große Rolle /33/. Während des Trainings sollten alle der in Bild 69 gezeigten Tätigkeiten, die zur Programmierung und Bestückung einer Leiterplatte durchzuführen sind, geübt werden. Diese Trainingsphasen werden bisher an einer realen Bestücklinie durchgeführt. Der Erfolg der von den Lernenden durchgeführten Bedienertätigkeiten zeigt sich in der korrekten Bestückung der vorgegebenen Leiterplatten. Durch den Einsatz des Simulationssystems während der Schulung ergeben sich folgende Vorteile:

Kostenersparnis
Wie auch in der Testphase spielt die Kostenersparnis durch den Einsatz des Simulationssystems während der Schulung eine entscheidende Rolle. Zur Schulung von Tätigkeiten, die direkt an einem Bestückautomaten durchzuführen sind, wie Rüsten und Wartung, werden zwar weiterhin reale Bestückautomaten benötigt, die Anzahl kann jedoch deutlich reduziert werden (Bild 87). Da das Simulationssystem auch die Simulation von Teil-Bestücklinien ermöglicht, kann die Schulung an einer Bestücklinie erfolgen, die zum Teil aus realen Bestückautomaten und zum Teil aus dem Simulationssystem besteht.

Verkürzung der Ausbildungszeiten
Der Einsatz des Simulationssystems erleichtert es, mehrere Trainingsanlagen bereit zu stellen, die aus jeweils einem Linienrechner und dem Simulationsrechner bestehen. Dadurch kann das Training in kleinen Gruppen durchgeführt werden (Bild 87), was einerseits den Lernerfolg verbessert und andererseits die Ausbildungszeiten reduziert.

Anpassung der Schulung an die Spezifika der Kunden
Bisher erfolgt die Schulung an einer der in den Trainingszentren des Herstellers aufgestellten Schulungsanlagen, die sich in den meisten Fällen bezüglich Ausstattung und Konfiguration von den beim Kunden eingesetzten Anlagen unterscheiden. Die einfache Parametrierbarkeit ermöglicht es nun, das Simulationssystem an die An-

lage des Kunden anzupassen und so ein Kunden-spezifisches Training durchzuführen.

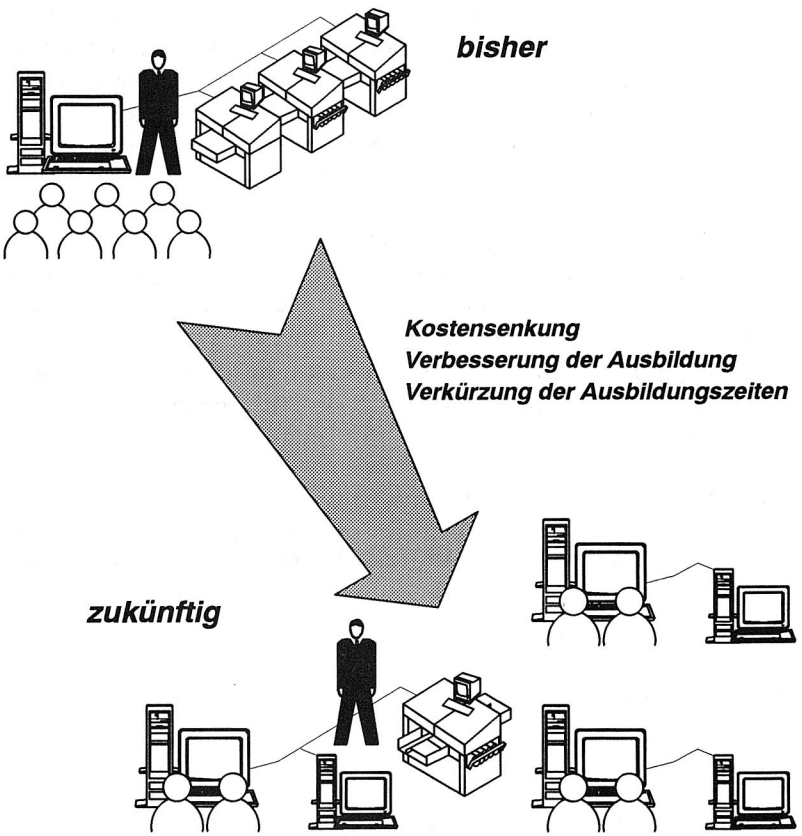


Bild 87: Einsatz des Simulationssystems in den Trainingsphasen der Schulung

Dezentrale Schulung beim Kunden

Der Einsatz des Simulationssystems ermöglicht es, die Schulung vor Ort beim Kunden durchzuführen. Dies führt einerseits zu Kosteneinsparungen bei den Kunden, da die Reisekosten für das zu schulende Personal entfallen, andererseits aber auch zu

Kosteneinsparungen beim Hersteller, da die Anzahl der Schulungsräume reduziert werden kann.

Training des Verhaltens im Fehlerfall
Mit dem Simulationssystem können vom Ausbilder jederzeit Fehlersituationen erzeugt werden, die über Fehlermeldungen am Linienrechner angezeigt werden und entsprechende Reaktionen des Bedieners verlangen. Derartige Fehlersituationen können an den realen Anlagen nur bedingt erzeugt werden. Das Simulationssystem ermöglicht somit eine umfassendere Trainingsphase, als dies an der realen Anlage möglich ist.

Ausprobieren von Optimierungsmaßnahmen
Durch das Training soll der Bediener einerseits die Fähigkeit erlangen, die Anlagen korrekt bedienen zu können, andererseits soll er aber auch ein Gefühl dafür bekommen, wie sich der Bestückablauf optimieren läßt. So lassen sich z. B. mit Hilfe des Simulationssystems ohne großen Aufwand die Auswirkungen einzelner Maßnahmen der Rüst- und Umrüstoptimierung (Ändern von Auftragsreihenfolgen, Umrüsten von Bauelementen) ermitteln.

Höhere Transparenz durch die Visualisierung
Während des Trainings werden den Auszubildenden Aufgaben gestellt, die durch Bedienung des Linienrechners gelöst werden (Bild 88). Bei der Durchführung der Steuerungstätigkeiten werden die vorher vom Auszubildenden eingegebenen Rüst- und Nutzdaten über die Datentelegramme an das Simulations- und Visualisierungssystem weitergegeben. Durch Analyse und Visualisierung der Daten werden fehlerhafte Eingaben erkannt und angezeigt und können vom Auszubildenden korrigiert werden.

9.3 Einsatz der Schulungssoftware

Ein großes Problem in der Ausbildung von Erwachsenen stellen die großen Unterschiede in der Vorbildung und in der Lerngeschwindigkeit dar. Bei der Schulung im Seminarbetrieb können deshalb die individuellen Lerngewohnheiten zum Teil nur ungenügend befriedigt werden. Hinzu kommen die hohen Kosten für die Seminare, die sich aus Teilnahmegebühren, Reisekosten und Lohnkosten zusammensetzen. Auf Grund unzureichender Kursangebote sind die Wartezeiten bis zur Teilnahme oft sehr lang.

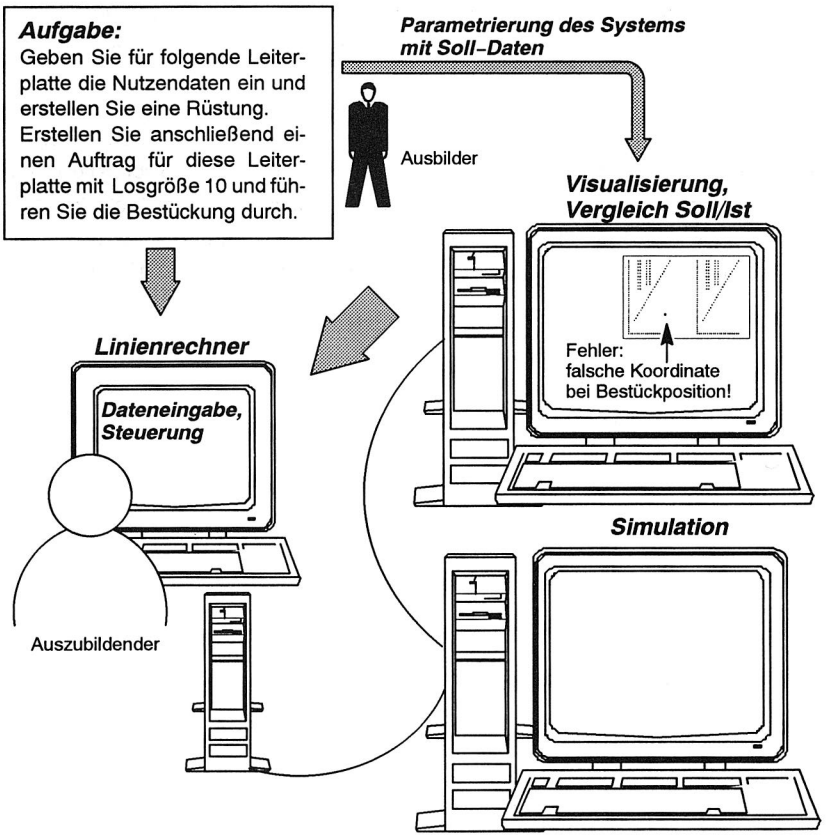


Bild 88: Einsatz des Simulationssystems in den Trainingsphasen der Schulung

Da sich das Wissen zur Bedienung der Bestückanlagen sowohl aus Theorie (also affektivem Wissen) als auch aus psychomotorischen Fähigkeiten zusammensetzt (Rüsten, Wartung), ist eine komplette Schulung mit CBT nicht möglich. Durch den Einsatz der Schulungssoftware wird aber im Vorfeld eines Kurses eine Nivellierung der theoretischen Vorbildung erreicht, so daß während des Kurses bei allen Teilnehmern von den gleichen Voraussetzungen ausgegangen werden kann. Das Hauptziel der Kurse stellt dann die Vermittlung der psychomotorischen Fähigkeiten dar. Da die

Theorie dann aber bereits zum großen Teil bekannt ist, können die Kurszeiten reduziert werden. Dadurch sinken zum einen die Kosten, zum anderen kann eine größere Zahl von Auszubildenden geschult werden.

Ein großer Vorteil der Vermittlung von Wissen über computerunterstützte Lernprogramme ist die einfache und im Vergleich zu gedruckten Medien preisgünstige Möglichkeit der Vervielfältigung und Verbreitung des Wissens. Stand der Technik ist hier zwar noch die Verbreitung über Datenträger wie Disketten und Bänder, in Zukunft wird sich aber die direkte Verbreitung über Rechnerkommunikation durchsetzen.

10. Zusammenfassung und Ausblick

Die harten Marktanforderungen an die Elektronikproduktion bezüglich niedriger Herstellungskosten und kurzer Produktlebenszeiten lassen sich nur durch den effizienten Einsatz moderner automatisierter Produktionsanlagen, wie z. B. SMD-Bestücklinien, erfüllen. Diese Anlagen sind auf Grund der geforderten Flexibilität, der hohen Produktionsleistung und der geforderten hohen Produktqualität geprägt durch komplexe Steuerungssoftware.

Häufig stellt die Entwicklung der Steuerungssoftware den entscheidenden Faktor dar, der die Marktreife der Produktionsanlagen verzögert. Gerade bei Steuerungssoftware mit hoher ***a-priori-Anpaßbarkeit***, die durch Parametrierung an unterschiedliche Anlagenlayouts angepaßt werden kann, kommt es entscheidend darauf an, bereits während der Entwicklungs- und Testphasen die Anzahl der Fehler so weit wie möglich zu reduzieren, da die Fehler, die erst nach der Auslieferung entdeckt werden, viele Kunden betreffen, damit sehr hohe Behebungskosten nach sich ziehen und das Image des Herstellers entscheidend beeinträchtigen.

Es wurde gezeigt, daß zum gesamtheitlichen Testen der auf einem Linienrechner eingesetzten Steuerungssoftware, bei dem neben der Benutzerkommunikation auch die Anlagenkommunikation getestet wird, die Simulation ein geeignetes Hilfsmittel darstellt. Das entwickelte Simulationssystem ***HSSIM*** bildet das Kommunikationsverhalten aller Bestückautomaten einer SMD-Bestücklinie gegenüber dem übergeordneten Linienrechner nach und überprüft die empfangenen Datentelegramme auf Fehlerfreiheit. Durch ***HSSIM*** wird während der Testphase eine beträchtliche Kosteneinsparung erzielt, da die Kapitalbindungskosten für eine statt des Simulationssystems benötigte Bestücklinie sowie die Verbrauchskosten während der Tests entfallen. Durch die einfache Parametrierbarkeit des Systems kann die Zahl der Testfälle erhöht werden, was dazu führt, daß während der Testphase in kürzerer Zeit eine größere Anzahl Fehler entdeckt und behoben wird.

Zum Speichern und Verwalten der zwischen Linienrechner und Bestückautomaten übertragenen Daten wurde das Programmpaket ***HSMON*** entwickelt, dessen Kern eine relationale Datenbank bildet. Mit den in der Datenbank gespeicherten Daten können während der Testphasen unterschiedliche Testläufe miteinander verglichen und Änderungen im Kommunikationsverhalten des Linienrechners gegenüber den Bestückautomaten ermittelt werden. Durch den Einsatz von Standardsoftware wur-

den verschiedene Möglichkeiten realisiert, die gespeicherten Daten zu visualisieren und somit einer optischen Plausibilitätskontrolle zu unterziehen. Diese Funktionalität ermöglicht vor allem während der Inbetriebnahme der Bestücklinien eine Überprüfung der übertragenen Daten.

Neben fehlerfreier Steuerungssoftware erfordert die steigende Komplexität der Produktionsanlagen qualifiziertes und gut geschultes Personal um einen effizienten Betrieb mit hoher Auslastung zu ermöglichen. Die kurzen Innovationszyklen und der Zwang zu Kostenersparnis führen dazu, daß die Einarbeitung in die Bedienung der Anlagen in Zukunft möglichst ohne konstenintensive Kurse in Eigenverantwortung erfolgen muß. Zur Ausbildung bietet sich hier der Einsatz vonLernsoftware an, die sich auf einfache Art vervielfältigen und an die Kunden verteilen läßt. Die Motivierung der Lernenden durch die multimedialen Möglichkeiten der Lernsoftware führen in kurzer Zeit zu einem hohen Lernerfolg. Das Lernprogramm ***HSLERN*** bietet dem Anlagenbediener die Möglichkeit, sich das zum Betrieb einer SMD-Bestückanlage notwendige Wissen anzueignen und durch die enthaltenen Tests und Übungen zu festigen.

Da der Lernerfolg entscheidend verbessert wird, wenn das Erlernte in geeigneter Form trainiert werden kann, wurde ein integriertes Gesamtsystem entwickelt, das neben der Lernkomponente ***HSLERN*** auch das Simulationssystem ***HSSIM*** und das Visualisierungssystem ***HSMON*** umfaßt. Nach dem Wissenserwerb kann der Bediener am Linienrechner die zur Bestückung von Leiterplatten notwendigen Dateneingaben sowie die Steuerung der Bestückautomaten üben. Das Simulationssystem ***HSSIM*** ersetzt dabei die reale Bestückanlage. Durch die Überprüfung und die Visualisierung der während der Bedienung eingegebenen Daten mit ***HSMON***, die in Form der Datentelegramme vom Linienrechner an das Simulationssystem übergeben werden, wird der Lernende auf etwaige Eingabefehler hingewiesen.

11. Literatur

1 Abels, S.
Modellierung und Optimierung von Montageanlagen in einem integrierten Simulationssystem
Hanser-Verlag, München Wien, 1993

2 Amann, W.
Eine Simulationsumgebung für Planung und Betrieb von Produktionssystemen
Dissertation, Technische Universität München, 1993

3 Amann, W.; Hartberger, H.
Produktionssysteme modellieren und simulieren
in ZwF 85 (1990) 7, 348-351

4 Baur P.
Entwicklungsbegleitendes Prototyping - Integration von CASE und Simulation
in Elektronik 12/1992

5 Bodendorf, E.
Computer in der fachlichen und universitären Ausbildung
R. Oldenbourg Verlag, München Wien, 1990

6 Booch, Grady
Object Oriented Design with Applications
The Benjamin/Cummings Publishing Company, Inc. 1990

7 Bullinger, H.-J.
Personalentwicklung und -qualifikation
Springer-Verlag, Berlin, 1992

8 Chmielnicki, S.
Flexible Fertigungssysteme - Simulation der Prozesse als Hilfsmittel zur Planung und zum Test von Steuerprogrammen
Springer-Verlag, Berlin, Heidelberg, 1985

9 Conklin, J.
Hypertext: An Introduction and Survey
in Computer, IEEE, 1988

10 Deutschländer A.
Integrierender Ansatz zur Reduzierung der Durchlaufzeit in der Leiterplattenmontage
in ZwF 86 (1991) 4, Hanser-Verlag München

11 DIN Deutsches Institiut für Normung E. V.
Entwurf DIN IEC66.22 - Elektrische Meßtechnik
Beuth Verlag GmbH, Berlin, 1976

12 DIN Deutsches Institiut für Normung E. V
Normung für Schnittstellen für die rechnerintegrierte Produktion (CIM)
DIN Fachbericht 15, Beuth Verlag GmbH Berlin/Köln, 1987

13 Draeger, G.
Gehäusestrategie - Schlüssel zum Markt
in Elektronik 12/1991

14 Dreyer, H.
Höhere Effizienz in Aus- und Weiterbildung durch Lehr- und Lernsoftware
in QZ 36 (1991) Heft 3

15 Ebert, J.; Herter, J.;Thomas, H.
Hersteller-Schulung für flexible Fertigungssysteme
in Mensch und Technik, Carl-Hanser-Verlag, München 1989, 714-718

16 Ehlers K.; Wolfram T.
Gute Karten fürs Labor - Neun IEEE-488-Karten im Test
in ELRAD 1994, Heft 1

17 Elsing, J.; Wiencek, A.
Schnittstellen Handbuch
IWT-Verlag, München, 1986

18 Eversheim, W.; Thome, H. G.
Einsatzgebiete der Simulation im Rahmen des Computer Integrated Manufacturing
in Simulation in der Fertigungstechnik, Springer Verlag 1988

19 Fähnrich, K. P.; Huthmann, A.; Kroneberg, M.;
Anpaßbarer Leitstand auf objektorientierter Basis
in Information als Produktionsfaktor, Tagungsband der 22. GI-Jahrestagung, 1992

20 Fedrowitz, C. H.
Fertigungszellen simulieren
in ZwF 88 (1993) 7-8, Carl Hanser Verlag, München, 1989

21 Feldmann, K.; Grampp, K.; Koch, M.; Rothhaupt, A.
Optimale Rüst- und Umrüststrategien steigern die Produktivität
in Leiterplattentechnik (Supplement zu den Hanser-Fachzeitschriften F&M, MO, QZ) Mai 1992, 46-50

22 Feldmann, K.; Grampp, K.
Improvement of Productivity by Computer Aided Process Planning, Control and Diagnosis
IEMT '92, Würzburg, Mai 1992

23 Feldmann, K; Grampp, K., Koch, M.
Rechnergestützte Verfahren in der Elektronikfertigung
pa Produktionsautomatisierung 1993, H. 5, S. 8–12

24 Feldmann, K.; Grampp, K.
Stand und Entwicklungslinien flexibler Bestücksysteme
Seminarunterlagen zum Seminar des VDI–Bildungswerk "SMT Rationelles Bestücken in der Oberflächentechnik", Düsseldorf: VDI, 1993

25 Feldmann K.; Schmidt B.
Simulation in der Fertigungstechnik
Springer–Verlag, Belin Heidelberg NewYork London Paris Tokyo,1988

26 Flohr, R.
Beitrag zur optimalen Verbindungstechnik in der Oberflächenmontage (SMT)
Hanser–Verlag, Wien, 1991

27 Flum, T.
Computerunterstütztes Lernen in der Pilotenausbildung
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus– und Weiterbildung
Springer–Verlag, Berlin Heidelberg, 1992

28 Frees, W.
Vernetzte Visualisierungs–PCs setzen sich durch
in Elektronik 7/1992

29 Freibichler H.
Multimediales Lernen – Konsequenzen aus einem Modellversuch
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus– und Weiterbildung
Springer–Verlag, Berlin Heidelberg, 1992

30 Glowalla, U.; Häfele, G.; Hasebrook, J.
Wiederlernen von Wissen
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus– und Weiterbildung
Springer–Verlag, Berlin Heidelberg, 1992

31 Glowalla, U.; Schoop, E
Entwicklung und Evaluation computerunterstützter Lernsysteme
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus– und Weiterbildung
Springer–Verlag, Berlin Heidelberg, 1992

32 Grampp, K.
Steuerung und Simulation von Bestückmaschinen und -linien
Seminarunterlagen zum Seminar des VDI-Bildungswerk "SMT Rationelles Bestücken in der Oberflächentechnik", Düsseldorf: VDI, 1993

33 Grampp, K.; Krug, S.; Walter, S.
Einsatz von Simulationssystemen zur Schulung an DV-Systemen im Fertigungsbereich
ASIM 1993 - 8. Symposium der Simulationstechnik, Berlin, 27.-30. September 1993

34 Grimm K.
Methoden und Verfahren zum systematischen Testen von Software
in atp, 30. Jahrgang, Heft 6/1988

35 Grimm K.
Klassifizierung und Bewertung von Software-Verifikationsverfahren
Forschungsbericht, AEG-Telefunken, Berlin

36 Grobel, T.; Kilger, C.; Rude, S.
Objektorientierte Modellierung der Produktionsorganisation
in Information als Produktionsfaktor, Tagungsband der 22. GI-Jahrestagung 1992

37 Hannemann, J.; Thüring, M.
Das Hypermedia-Autorensystem SEPIA
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus- und Weiterbildung
Springer-Verlag, Berlin Heidelberg, 1992

38 Haugg, F.
Software-Engineering und ihre Qualitätssicherung,
Franzis-Verlag 1983, München

39 Heeg, F.-J.
Management -Aus- und Weiterbildung für Ingenieure.
in VDI-Z 131 (1989) Nr. 1

40 Heeg, F.-J.; Deserno G.
Neue Techniken erfordern neue Ausbildungsformen
in VDI-Z 129 (1987) Nr. 4

41 Herrmann, G.; Egerer K.
Handbuch der Leiterplattentechnik, Band 2: Neue Verfahren, Neue Technologien
Eugen G. Leuze Verlag, Saulgau, 1991

42 Herrscher, A.; Grimm, W.; Storr, A.; Reichenbächer, J.
Systematische Softwareerstellung für Steuerungen
im Vortragsband zum Fertigungstechnischen Kolloquium (FTK) an der Universität Stuttgart 1991, S. 52 -58

43 Herzog, U., Gora, W. et al.:
Kommunikation in der rechnerintegrierten Fabrik
Proc. Fachtagung "Rechnerintegrierte Produktionssysteme", FAU, 1987

44 Höhner, U.
Informationstechnik und Qualifikation
in AV 27 (1990) 1, Carl Hanser Verlag, München, 1990

45 Hundt, R.
CBT am Lernort Betrieb am Beispiel der Deutschen Bundespost POSTDIENST
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus- und Weiterbildung
Springer-Verlag, Berlin Heidelberg, 1992

46 ISO IS 9506
Manufacturin Message Specification
Part I: Service Definition; Part II: Protocol Specification 1988

47 Jöns, I.
Möglichkeiten und Grenzen formativer Evaluation computerunterstützter Lernsysteme im Rahmen anwendungsorientierter Entwicklungsprojekte
in Neue Wege in der comnputerunterstützen Aus- und Weiterbildung, Berlin 1992

48 Kappus, J.
Die IEEE-488.2-Norm
in c't 1992, Heft 11

49 Keller, G.
Studiertechniken
Bock- Verlag, Bad Honnef 1986

50 Knabe, Gerald
Qualitätssicherung sowie Wirtschaftlichkeit und Nutzen von interaktiven Lernsystemen in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus- und Weiterbildung
Springer-Verlag Berlin Heidelberg 1992

51 Kohen, E.
Informationsverarbeitung in FFS
VDI-Verlag, Düsseldorf, 1990

52 Kohring, A.
Systematisches Projektieren und Testen von Steuerungssoftware für Werkzeugmaschinen
Verlag Shaker, Aachen 1993

53 Kowalke, P.
Korrekte Software: Semantik, Spezifikation, Verifikation und Testen von Programmen
Mannheim: BI-Wiss.-Verl. 1993

54 Krüger, M.
Software-Qualitätssicherung - Eine nutzenorientierte Betrachtung
in Elektronik 25/1992

55 Krüger, M.
Prüfen von Software - aber richtig
in Elektronik 24/1993

56 Lauer, T.
Client-Server-Programmierung mit DDE/DDEML
in c't 1993 Heft 1

57 Lindau, U.
Objektorientierte Zellensteuerung
in ZwF 88 (1993) 7-8, Carl Hanser Verlag, München 1993

58 Linner S.
Entwicklungszeiten verkürzen durch Simulation
in pa Produktionsautomatisierung 2/92

59 Lochner, C.; Widmaier, D.
Objektorientiertes Fertigungsleitstandskonzept
in ZwF 87 (1992), Carl Hanser Verlag, München 1992

60 Luetkens, L.
DOS für Echtzeitaufgaben
in Elektronik 23/1993

61 Mason, T.; Brown, D.
lex & yacc
O' Reilly & Associates, 1990

62 Meinberg, U.; Schürholz, A.
SImulation als Testumgebung für Steuerungssoftware - erste Anwendungserfahrungen
in ASIM (Hrsg.) Simulation und Integration, Tagungsband 1989

63 Meyer, B.
Object-oriented Software Construction
Prentice Hall International (UK) Ltd 1988

64 Meyer, H.; Geib, S.
SMT-Fertigungslinien rechnergestützt optimieren
in F & M 101 (1993) 5, Hanser-Verlag, München

65 Milberg, J.; Amann W.; Raith, P.
Beschleunigte Inbetriebnahme von Produktionsanlagen durch getestete Ablaufvorschriften
in VDI-Z 134 (1992) Nr. 2

66 Milberg, J.; Amann, W.; Schuster, G.
Simulation in der Produktionstechnik - Entwicklungen und Möglichkeiten
pa Produktionsautomatisierung 3/92

67 Milberg, J.; Hartberger, H.
PLATO-SIM - Wissensbasierte Simulation in der Anlagenplanung
in VDI-Z 132 (1990), Nr. 5 - Mai

68 Milberg, J.; Köpfer, Th.
Wettbewerbsvorteile durch rechnerintegrierte Konstruktion und Produktion
In: VDI-ADB (Hrsg.): VDI-Bericht 830 "Rechnerintegrierte Konstruktion und Produktion", VDI-Verlag Düsseldorf 1990

69 Mitterndorfer J.
Objektorientierte Programmierung mit C++ und Smalltalk
Addison-Wesley Publishing Company 1990

70 Myers, G.J.
Methodisches Testen von Programmen
Oldenbourg-Verlag, München 1989

71 N. N.
Der Weg zu Fine-Pitch
in SMT 6/92

72 N. N.,
Das Technologie-Handbuch aus der Praxis für die Praxis
Technologiebroschüre der Fa. Siemens (1993)

73 N. N.: Bericht der Merzedes Benz AG
Computer Based Training: Erfolgreicher und mit mehr Spaß
in Elektronik 6/1993

74 National Instruments
NI-488.2™ MS-DOS
Austin, 1991

75 Nietsch, M.; Rinschede, M.; Rautenstrauch, C.
Konzeption und Entwicklung eines Objektmodells für einen individualisierbaren Leitstand
in Information als Produktionsfaktor, Tagungsband der 22. GI-Jahrestagung 1992

76 Noche, B.; Wenzel, S.
Marktspiegel Simulationstechnik in Produktion und Logistik
Verlag TÜV Rheinland

77 Pacoe, G.-A.
Elements of Object-Oriented Programming
in Byte August 1986

78 Pawlischek, H.
Flexible Peripherie für kleine SMD-Bestücklose
in Feinwerktechnik & Messtechnik 98 (1990) 3, Carl-Hanser-Verlag München

79 Pietrek, M.
Inside Windows-Messaging - Ein Blick in das Herzstück von Windows
in c't 1993, Heft 6

80 Piotrowski, A.
IEC-Bus Software
Franzis-Verlag, München 1989

81 Pöhls, A.
Rüstoptimierung in der SMT-Baugruppenbestückung
in Feinwerktechnik & Messtechnik 98 (1990) 7-8, Carl-Hanser-Verlag München

82 Pomberger, G.
Software-Technik und Modula2
Carl Hanser Verlag München Wien 1984

83 Raith P.; Amann W.
Erstellen und Testen von Ablaufvorschriften für Produktionssysteme
in ZwF 87 (1992) 7, Carl Hanser Verlag, München 1992

84 Reinhardt, A.; Rudnig, M.; Wiegand, M.
Simulation und Realzeitsteuerung
in ASIM (Hrsg.) Simulation und Integration, Tagungsband 1990

85 Rosette, C.
Multimedia im Bereich Aus- und Weiterbildung
in Elektronik 10/1992

86 Schrödel, O.
Objektorientierte Software in der Fertigungsautomatisierung
in pa 3/92

87 Scheider, M
Computer Based Training als Mittel zur Unterstützung von Lernprozessen in der Produktion
in: Qualifizierung und Personalentwicklung, Hrsg: Dieter Seitz
Rationalisierungs-Kuratorium der Deutschen Wirtschaft e. V.

88 Scheller, J.
Modellierung und Einsatz von Softwaresystemen für rechnergeführte Montagezellen
Hanser-Verlag, München, Wien, 1991

89 Schimpf, A.
Objektorientierte Programmierung
in PCTechnik 1991, Heft 5, Heft 6, Heft 7, Heft 8 und Heft 9

90 Schlüter K.
Reserven bei den Rüstzeiten
in PRONIC H. 1/2, Februar 1993

91 Schlüter, K.
Nutzungsgradsteigerung von Montagesystemen durch den Einsatz der Simulationstechnik
Hanser-Verlag, Wien1989

92 Schlüter, K.; Nolting, F.-W.;
Produktionsüberwachung auf Workstation-Rechnern
in ZwF 84 (1989)

93 Schmidt, J.; Schelberg H.-J.
Objektorientierte Projektierung von Steuerungssoftware
in VDI-Z Bd.133 (1991), Nr.12-Dezember

94 Schmitz, P.
Software-Qualitätssicherung - Testen im Software-Lebenszyklus

95 Schoop, E.; Glowalla, U.
Computer in der Aus- und Weiterbildung: Potentiale, Probleme und Perspektiven
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus- und Weiterbildung
Springer-Verlag Berlin Heidelberg 1992

96 Schrüfer, N.
Rüstzeitreduzierung durch 3D-NC-Simulation
Dissertation, Technische Universität München 1990 186

97 Schürholz A.
Simulation als Testumgebung für Steuerungssoftware
in ASIM-Tagungsband Simulationshandbuch

98 Schwarz, K.
Manufacturing Message Specification (MMS)
in atp 33 (1991)

99 Seidel, Ch.; Lipsmeier, A.
Computerunterstütztes Lernen: Entwicklungen - Möglichkeiten - Perspektiven
Verlag für Angewandte Psychologie, Stuttgart 1989

100 Springer, G.
Simulationsgestützte Mitarbeiterausbildung am Beispiel der Fertigungssteuerung
VDI-Verlag Düsseldorf 1992

101 Steppi, H.
Computer Based Training: Planung, Design und Entwicklung interaktiver Lernprogramme
Klett-Verlage Stuttgart 1989

102 Storr, A.; Frank, H.; Walker, H.
Software als Produktkomponente für flexible Fertigungssysteme
in wt - Zeitschrift für industrielle Fertigung 76 (1986) S.. 313-318

103 Strobel R.
Objektorientierte Software-Entwicklung: Bestandsaufnahme und Ausblick
in ist - Intelligente Software-Technologien 1/92

104 Swik R.; Neuwirth S.
Echtzeitanwendungen auf 80386/486-Prozessoren im "Protected Mode"
in Elektronik 20/1992, Elektronik 21/1992 und Elektronik 22/1992

105 Swik R.
Multitasking zum Kennenlernen
in Elektronik 12/1992

106 Tischer
PC-Intern Systemprogrammierung 2.0
Data Becker Düsseldorf 1989

107 VDI Verein Deutscher Ingenieure
Simulation von Logistik-, Materialfluß- und Produktionssystemen, Grundlagen
VDI 3633, Oktober 1991

108 Weck, M.; Eversheimer, W.; König W.; Pfeifer T.
Wettbewerbsfaktor Produktionstechnik
VDI Verlag GmbH, Düsseldorf 1990

109 Weck, M.
Simulation in CIM
Springer-Verlag, 1991

110 Weidenmann, B.
Psychologische Aspekte des Lernens mit dem Computer
in Hypertext und Multimedia: Neue Wege in der computerunterstützten Aus- und Weiterbildung
Springer-Verlag Berlin Heidelberg 1992

111 Witte, H. H.
Konventionelle objektorientierte Simulation mit regelbasiertem Optimierungsansatz am Beispiel der Betriebsmittelorganisation

112 Wollert, J.; Fiedler, J.; Stockmann, Th.
Automatisieren mit dem PC?
in Elektronik 6/1994

113 Wolski, G. B.
Optimierung von SMT-Fertigung und Fine-Pitch-Prozeß
in SMD-Magazin 1/93

114 Wrba, P.
Simulation als Werkzeug in der Handhabungstechnik
Dissertation Technische Universität München, 1990

115 Zankl, A.; Engel, G.
Trends in der Automatisierungstechnik
in Elektronik 13/1992

Lebenslauf

Konrad Grampp
geb. am 07.02.1962 in Coburg

1968 - 1972	Grundschule in Coburg
1972 - 1981	math.-naturwiss. Gymnasium Ernestinum in Coburg Abschluß: Abitur
1981 - 1982	Grundwehrdienst bei der Bundeswehr
1982 - 1988	Studium der Informatik an der Friedrich-Alexander-Universität in Erlangen-Nürnberg Abschluß: Dipl.-Inf. Univ.
1988 - 1989	Angestellter bei der KWU (Siemens AG) als Systembetreuer für Apollo-WS30-Rechner
1989 - 1995	wissenschaftlicher Mitarbeiter in der Steuerungs- und Sensorgruppe am Lehrstuhl für Fertigungsautomatisierung und Produktionssystematik (Prof. Dr.-Ing. K. Feldmann) der Friedrich-Alexander-Universität Erlangen-Nürnberg

Reihe Fertigungstechnik Erlangen

Band 1
Andreas Hemberger
Innovationspotentiale in der rechnerintegrierten Produktion durch wissensbasierte Systeme
208 Seiten, 107 Bilder. 1988. Kartoniert.

Band 2
Detlef Classe
Beitrag zur Steigerung der Flexibilität automatisierter Montage-systeme durch Sensorintegration und erweiterte Steuerungskonzepte
194 Seiten, 70 Bilder. 1988. Kartoniert.

Band 3
Friedrich-Wilhelm Nolting
Projektierung von Montagesystemen
201 Seiten, 107 Bilder, 1 Tabelle. 1989.
Kartoniert.

Band 4
Karsten Schlüter
Nutzungsgradsteigerung von Montagesystemen durch den Einsatz der Simulationstechnik
177 Seiten, 97 Bilder. 1989. Kartoniert.

Band 5
Shir-Kuan Lin
Aufbau von Modellen zur Lageregelung von Industrierobotern
168 Seiten, 46 Bilder. 1989. Kartoniert.

Band 6
Rudolf Nuss
Untersuchungen zur Bearbeitungsqualität im Fertigungssystem Laserstrahlschneiden
206 Seiten, 115 Bilder, 6 Tabellen. 1989. Kartoniert.

Band 7
Wolfgang Scholz
Modell zur datenbankgestützten Planung automatisierter Montageanlagen
194 Seiten, 89 Bilder. 1989. Kartoniert.

Band 8
Hans-Jürgen Wißmeier
Beitrag zur Beurteilung des Bruchverhaltens von Hartmetall-Fließpreßmatrizen
179 Seiten, 99 Bilder, 9 Tabellen. 1989. Kartoniert.

Band 9
Rainer Eisele
Konzeption und Wirtschaftlichkeit von Planungssystemen in der Produktion
183 Seiten, 86 Bilder. 1990. Kartoniert.

Band 10
Rolf Pfeiffer
Technologisch orientierte Montageplanung am Beispiel der Schraubtechnik
216 Seiten, 102 Bilder, 16 Tabellen. 1990. Kartoniert.

Band 11
Herbert Fischer
Verteilte Planungssysteme zur Flexibilitätssteigerung der rechnerintegrierten Teilefertigung
201 Seiten, 82 Bilder. 1990. Kartoniert.

Band 12
Gerhard Kleineidam
CAD/CAP : Rechnergestützte Montagefeinplanung
203 Seiten, 107 Bilder. 1990. Kartoniert.

Band 13
Frank Vollertsen
Pulvermetallurgische Verarbeitung eines übereutektoiden verschleißfesten Stahls
XIII + 217 Seiten, 67 Bilder, 34 Tabellen. 1990. Kartoniert.

Band 14
Stephan Biermann
Untersuchungen zur Anlagen– und Prozeßdiagnostik für das Schneiden mit CO_2– Hochleistungslasern
VIII + 170 Seiten, 93 Bilder, 4 Tabellen. 1991. Kartoniert.

Band 15
Uwe Geißler
Material– und Datenfluß in einer flexiblen Blechbearbeitungszelle
124 Seiten, 41 Bilder, 7 Tabellen. 1991. Kartoniert.

Band 16
Frank Oswald Hake
Entwicklung eines rechnergestützten Diagnosesystems für automatisierte Montagezellen
XIV + 166 Seiten, 77 Bilder. 1991. Kartoniert.

Band 17
Herbert Reichel
Optimierung der Werkzeugbereitstellung durch rechnergestützte Arbeitsfolgenbestimmung
198 Seiten, 73 Bilder, 2 Tabellen. 1991. Kartoniert.

Band 18
Josef Scheller
Modellierung und Einsatz von Softwaresystemen für rechnergeführte Montagezellen
198 Seiten, 65 Bilder. 1991. Kartoniert.

Band 19
Arnold vom Ende
Untersuchungen zum Biegeumformen mit elastischer Matrize
166 Seiten, 55 Bilder, 13 Tabellen. 1991. Kartoniert.

Band 20
Joachim Schmid
Beitrag zum automatisierten Bearbeiten von Keramikguß mit Industrierobotern
XIV + 176 Seiten, 111 Bilder, 6 Tabellen. 1991. Kartoniert.

Band 21
Egon Sommer
Multiprozessorsteuerung für kooperierende Industrieroboter in Montagezellen
188 Seiten, 102 Bilder. 1991. Kartoniert.

Band 22
Georg Geyer
Entwicklung problemspezifischer Verfahrensketten in der Montage
192 Seiten, 112 Bilder. 1991. Kartoniert.

Band 23
Rainer Flohr
Beitrag zur optimalen Verbindungstechnik in der Oberflächenmontage (SMT)
186 Seiten, 79 Bilder. 1991. Kartoniert.

Band 24
Alfons Rief
Untersuchungen zur Verfahrensfolge Laserstrahlschneiden und –schweißen in der Rohkarosseriefertigung
VI + 145 Seiten, 58 Bilder, 5 Tabellen. 1991. Kartoniert.

Band 25
Christoph Thim
Rechnerunterstützte Optimierung von Materialflußstrukturen in der Elektronikmontage durch Simulation
188 Seiten, 74 Bilder. 1992. Kartoniert.

Band 26
Roland Müller
CO_2 – Laserstrahlschneiden von kurzglasverstärkten Verbundwerkstoffen
141 Seiten, 107 Bilder, 4 Tabellen. 1992. Kartoniert.

Band 27
Günther Schäfer
Integrierte Informationsverarbeitung bei der Montageplanung
195 Seiten, 76 Bilder. 1992. Kartoniert.

Band 28
Martin Hoffmann
Entwicklung einer CAD/CAM–Prozeßkette für die Herstellung von Blechbiegeteilen
149 Seiten, 89 Bilder. 1992. Kartoniert.

Band 29
Peter Hoffmann
Verfahrensfolge Laserstrahlschneiden und –schweißen : Prozeßführung und Systemtechnik in der 3D–Laserstrahlbearbeitung von Blechformteilen
186 Seiten, 92 Bilder, 10 Tabellen. 1992. Kartoniert.

Band 30
Olaf Schrödel
Flexible Werkstattsteuerung mit objektorientierten Softwarestrukturen
180 Seiten, 84 Bilder. 1992. Kartoniert.

Band 31
Hubert Reinisch
Planungs– und Steuerungswerkzeuge zur impliziten Geräteprogrammierung in Roboterzellen
XI + 212 Seiten, 112 Bilder. 1992. Kartoniert.

Band 32
Brigitte Bärnreuther
Ein Beitrag zur Bewertung des Kommunikationsverhaltens von Automatisierungsgeräten in flexiblen Produktionszellen
XI + 179 Seiten, 71 Bilder. 1992. Kartoniert.

Band 33
Joachim Hutfless
Laserstrahlregelung und Optikdiagnostik in der Strahlführung einer CO_2–Hochleistungslaseranlage
175 Seiten, 70 Bilder, 17 Tabellen. 1993. Kartoniert.

Band 34
Uwe Günzel
Entwicklung und Einsatz eines Simulationsverfahrens für operative und strategische Probleme der Produktionsplanung und –steuerung
XIV + 170 Seiten, 66 Bilder, 5 Tabellen. 1993. Kartoniert.

Band 35
Bertram Ehmann
Operatives Fertigungscontrolling durch Optimierung auftragsbezogener Bearbeitungsabläufe in der Elektronikfertigung
XV + 167 Seiten, 114 Bilder. 1993. Kartoniert.

Band 36
Harald Kolléra
Entwicklung eines benutzerorientierten Werkstattprogrammiersystems für das Laserstrahlschneiden
129 Seiten, 66 Bilder, 1 Tabelle. 1993. Kartoniert.

Band 37
Stephanie Abels
Modellierung und Optimierung von Montageanlagen in einem integrierten Simulationssystem
188 Seiten, 88 Bilder. 1993. Kartoniert.

Band 38
Robert Schmidt–Hebbel
Laserstrahlbohren durchflußbestimmender Durchgangslöcher
145 Seiten, 63 Bilder, 11 Tabellen. 1993. Kartoniert.

Band 39
Norbert Lutz
Oberflächenfeinbearbeitung keramischer Werkstoffe mit XeCl–Excimerlaserstrahlung
187 Seiten, 98 Bilder, 29 Tabellen. 1994. Kartoniert.

Band 40
Konrad Grampp
Rechnerunterstützung bei Test und Schulung an Steuerungssoftware von SMD–Bestücklinien
178 Seiten, 88 Bilder. 1995. Kartoniert.

Band 41
Martin Koch
Wissensbasierte Unterstützung der Angebotsbearbeitung in der Investitionsgüterindustrie
169 Seiten, 68 Bilder. 1995. Kartoniert.