Anwendungssysteme auf Silizium

(Projekt .015)

- Vorlaeufige Ueberlegungen -

Erarbeitet von:
Dipl. ing. Wolfgang Matthes

Release: 1
vom: 4. 1. 88

Inhalt

1. Zielstellung

Moderne Schaltkreistechnologien bieten die Moeglichkeit, eine Vielzahl von Funktionen der Informationsverarbeitung auf einem Schaltkreis unterzubringen. Mit geeigneten Entwurfssystemen wird es - zusammen mit einer entsprechenden Fertigungsorganisation - praktikabel, die Schaltkreise anwendungsspezifisch zu gestalten (ASIC).

Andererseits haben Standardschaltkreise (vom Mikroprozessor bis zur Treiberbaustufe) ein sehr hohes Niveau in Bezug auf Leistungsfaehigkeit, Verfuegbarkeit, Zuverlaessigkeit usw. erreicht.

Solche Standardschaltkreise sind fuer den Entwickler industrieller Elektronik vergleichsweise einfach nutzbar. Es ist unproblematisch, Versuchsschaltungen aufzubauen und zu aendern. Zur Erprobung sind industrieuebliche Mess- bzw. Pruefmittel oft ausreichend.

ASICs bieten gegenueber Loesungen, die ausschliesslich auf Standardschaltkreise gestuetzt sind, offensichtliche Vorteile hinsichtlich der weiteren Miniaturisierung, der Geschwindigkeits-Erhoehung usw.

Diese Vorteile muessen durch weitgehende Umstellungen im Entwicklungsprozess erkauft werden.

Der Gebrauchswert/Kosten- Vergleich zwischen solchen Loesungen, die auf ASIC und jenen, die auf Standardschaltkreisen beruhen, braucht mithin nicht notwendigerweise von vornherein zugunsten von ASIC auszufallen.

Diese Schrift setzt sich nicht das Ziel, einen systematischen Vergleich auszufuehren bzw. zu diskutieren.

Vielmehr sollen - in einer gleichsam aggressiven Weise - die Schwierigkeiten des ASIC- Einsatzes in ihrem wechselseitigen Zusammenhang so beleuchtet werden, dass daraus forschungsmaessige bzw. erfinderische Zielstellungen hervorgehen, und es sollen erste Loesungsansaetze aufgezeigt werden.

Die endgueltigen Folgerungen sind dann aeusserst einfacher Natur: entweder es werden ueberzeugende und praktisch durchsetzbare Loesungen gefunden - dann ist die Ueberlegenheit des ASIC- Ansatzes bewiesen und es geht nur noch um das entschlossene Durchsetzen - oder solche Loesungen werden nicht gefunden - dann ist der wirtschaftliche Nutzen von Aufwendungen zur Entwicklung und Fertigung von ASIC neu zu ueberdenken.

2. Zweck der Schrift

Es sollen erste Ueberlegungen zu den Sachverhalten selbst sowie erste Loesungsansaetze dargestellt werden, wobei es weniger darum geht, technische Einzelheiten zu beschreiben, als verschiedene Prinzipien und Wahlmoeglichkeiten vorzustellen und zu bewerten.

Fuer die Beschreibung der bevorzugten Loesungen ist ein "Vorlaeufiges Handbuch der Wirkprinzipien" vorgesehen.

3. Technische und wirtschaftliche Tatsachen in ihrer Wechselwirkung

Es gilt, Sinnfaelligkeit und Nutzen von ASIC nachzuweisen.
Dies ist nur durch Loesungen moeglich, die dem Stand der Technik ueberlegen sind.
Dem Stand der Technik entsprechend koennen mit Mikroprozessor-Funktionsmoduln, die durch anwendungsspezifische Koppeleinheiten und andere Sonderschaltungen ergaenzt werden, praktisch alle anfallenden Steuerungsaufgaben der industriellen Elektronik geloest werden.
In welcher Hinsicht sind - ausgehend von diesem Stand - ueberhaupt noch Verbesserungen moeglich ?

a) schneller
b) flexibler
c) kleiner
d) billiger
e) zuverlaessiger.

Zu a): schneller

Das ist nur durch weitgehende funktionelle Spezialisierung (Spezialprozessoren, Ersatz von Software durch Hardware usw.) zu erreichen: grundsaetzlich bieten moderne Mikroprozessoren (z. B. 80 386) zusammen mit den einschlaegigen ergaenzenden Schaltkreisen (I/O- Steuerungen, Speicher, Coprozessoren) ein Leistungsvermoegen, das kaum wesentlich ueberboten werden kann.

Zu b): flexibler

Die programmseitige Flexibilitaet moderner Mikroprozessoren ist nicht zu ueberbieten.
Spezifische Anpass- Schaltungen koennen hingegegen durch Ausnutzung der ASIC- Entwurfsmoeglichkeiten (d. h. weitgehende Nutzung der verfuegbaren Gatter, Flipflop usw.) programmier- bzw. konfigurierbar gestaltet werden.

Zu c): kleiner

Dies haengt vom Anteil "reiner Logik" bzw. "reiner Informationsverarbeitung" zwischen den unumgaenglichen prozessspezifischen Koppelschaltungen und dem Verarbeitungskomplex (z. B. Mikroprozessor mit Speicher) ab.
Je groesser dieser Anteil und je flexibler die Gestaltungsmoeglichkeiten des ASIC, desto mehr wird die Verkleinerung wirksam.
Die unmittelbare Prozess- Kopplung laesst sich groessenmaessig nicht vermindern.
Ebenso koennen beim derzeitigen und absehbaren Stand der Technik umfangreiche Logik- und Speicherkomplexe (ab z. B. 16 k Bytes) nicht zusammen auf einem Schaltkreis vorgesehen werden (somit erfordern groessere Speicher stets gesonderte Schaltkreise, d. h. aufwandsmaessig ist ASIC + Speicher = Mikroprozessor + Speicher).

Zu d): billiger

Dies bedarf der genauen Durchrechnung aller Kosten fuer die Systeme, die zu vergleichen sind.

Zu e): zuverlaessiger

Wenn die Schaltungen kleiner werden (insgesamt weniger Gehaeuse und Verbindungen), steigt die Zuverlaessigkeit an.
Zudem wird es durchaus praktikabel, Redundanz einzufuehren.
Weiterhin ermoeglicht das an sich gegebene "scan/set" Testprinzip der ASIC ein genaues Testen der "Restlogik" (die erfahrungsgemaess einer automatischen Pruefung nur schwer zugaenglich ist) und ein besseres Testen (mit deutlich hoeherer Genauigkeit der Fehlerlokalisierung) des gesamten Systems. Voraussetzung: "scan/set" muss in der "Systemphilosophie" von Anfang an und durchgaengig beruecksichtigt werden.

Das grundsaetzliche uebergeordnete Ziel kann nur sein: wirtschaftliche Erfolge durch Verkauf kompletter Maschinen (bzw. Maschinensysteme) anstelle des Angebotes "Maschine + fremde Elektronik".
Die komplette Maschine muss dazu billiger sein als die Kombination "Maschine + fremde Elektronik", und sie sollte anwendungsseitige Vorteile aufweisen (von der verminderten Stellflaeche durch direkten Einbau der Elektronik in die eigentliche Maschine ueber deutlich hoehere Zuverlaessigkeit - ggf. durch Redundanz - bis hin zu neuen Gebrauchswerten, die nur bei Ausfuehrung "aus einer Hand" geschaffen werden koennen).
Um auf Fragen des Service, der Erweiterungs- und Komplettierungsmoeglichkeiten, der Kompatibilitaet bei Neuentwicklungen usw.) eine ueberzeugende Antwort geben zu koennen, sind beispielsweise folgende Ansaetze denkbar:

- Es wird eine durchgehendes automatisches Testen der Elektronik in der Maschine gewaehrleistet, das die fehlerhafte Baugruppe in den weitaus meisten Faellen eindeutig lokalisieren kann.

- Es werden konstruktive Vorkehrungen getroffen, die einen extrem einfachen Baugruppentausch gewaehrleisten.

- Die Betriebszuverlaessigkeit an sich wird deutlich erhoeht.

- Damit wird es praktikabel, den (sehr seltenen) Tausch der (automatisch lokalisierten) Baugruppen vom Anwender selbst ausfuehren zu lassen.

- Die komplette Maschine muss fuer den Kunden guenstiger sein als die Kombination "Maschine + Fremdelektronik + Wartungsvertrag fuer die Fremdelektronik".

- Die ASICs selbst beruhen auf Standardzellen und Technologien, fuer die es international Zweitlieferanten gibt.

- Fuer die Kopplung mit Fremdsystemen werden international uebliche Schnittstellen bereitgestellt (z. B. Ethernet, MAP,

VME- Bus usw.).

Daraus leitet sich die grundlegende technische Zielstellung ab:

Vollstaendige Vermeidung der "Restlogik", erweiterte Flexibilitaet durch hoehere Verarbeitungsleistung, deutliche Verbesserung der Zuverlaessigkeit, Miniaturisierung auf Subsystemebene: Speicher + Mikroprozessor + Sonderschaltkreise + "Restlogik" (derzeitig auf mehrere Leiterplatten verteilt, die ueber ein Bussystem untereinander verbunden sind) wird ersetzt durch Speicher + ASIC, so dass jedes Subsystem nicht mehr als eine (moeglichst kleine) Leiterplatte belegt.

Verschiedene Betrachtungsweisen fuehren zu zwei unterschiedlichen "idealen" Ansaetzen:

Ideal A

Modulares System von Funktionseinheiten (Prozessorkernen, Speicheranschluss- Steuerungen, seriellen und parallelen E/A-Schnittstellen, Zaehlern usw.) sowie ein ausgebautes System zur Entwurfsunterstuetzung, das die komplette rechentechnische Bearbeitung des Anwendungsfalles gewaehrleistet und ein jeweils aufwandsoptimiertes Chip- Layout erzeugt ("silicon compiler").

Ideal B

Nur sehr wenige ASIC- Typen, die ausschliesslich "soft" an den jeweiligen Verwendungszweck angepasst werden.

Es kann als weitgehend sicher gelten, dass sich mit Ideal A die Ziele a), c), e) erreichen lassen (ein ASIC bietet so viele Moeglichkeiten, dass jeweils eine Anwendungsloesung komplett in Form einer optimierten Schaltung untergebracht werden kann).

Dem stehen entgegen:

- die Kosten
- die deutlich schlechteren Wartungs- Eigenschaften (Vielzahl an ASIC- Typen erzwingt entsprechende Lagerhaltung)
- die praktische Beherrschbarkeit an sich (das Entwurfssystem muss eine fehlerfreie Entwicklung gewaehrleisten; man haette bestenfalls einen Iterationsschritt fuer Layout- Aenderungen, wenn man industrieuebliche Terminablaeufe zugrunde legt).

Ideal B ist fragwuerdig in der Hinsicht, ob damit die gewuenschten durchgreifenden Verbesserungen gegenueber Standard-Mikroprozessorsystemen erreichbar sind: ein solches ASIC unterscheidet sich von einem ueblichen Mikroprozessor naemlich nur dadurch, dass einige der ueblichen E/A- Schnittstellen (seriell, parallel, Zaehler/Zeitgeber usw.) im Gehaeuse des Prozessors mit enthalten sind. Zudem ist das schematische Nachempfinden eines eingefuehrten Mikroprozessors mit ASIC sicherlich sehr unwirtschaftlich gegenueber der Nachfertigung des Mikroprozessors mit Original- Layout.

Wenn es ueberhaupt einen Weg gibt, durch breite Nutzung von ASIC in der maschinenbauenden Industrie zu wirtschaftlichen Erfolgen zu kommen, so erfordert dies zwingend eine weitgehende Annaehrung an Ideal B.

Die forschungsmaessige und erfinderische Zielsetzung muss also sein: im Sinne eines technisch- oekonomischen Kompromisses die Vorteile von Ideal A weitgehend mit technischen Mitteln zu erreichen, die Ideal B angenaehert sind, so dass fuer die weitaus meisten Einsatzfaelle Systeme nach Ideal B voellig zufriedenstellende Loesungen sind. Lediglich fuer ausgewaehlte strukturbestimmende Finalerzeugnisse waeren u. U. dann einzelne Systeme gemaess Ideal A notwendig.

4. Grundsaetzliche Loesungswege

4.1. Uebersicht

4.1.1. Gedanklicher Ansatz

Im Sinne einer allgemeinen Methodenlehre fuer technische Entwicklungen ist davon auszugehen, dass ein iterativer Prozess in Gang gesetzt werden muss, der die Bewertung von Gegebenem, die Analyse von Anforderungen und konstruktive Vorschlaege gleichermassen umfasst. Um einen solchen Prozess anzuregen, seien im folgenden einige erste konstruktive Ansaetze skizziert.

Hinweise:

1. Den folgenden Vorschlaegen liegt eine Vielzahl von Erfahrungen und Erkenntnissen aus der Literatur zugrunde, aber (noch) keine Analyse kuenftiger Anwendungsfaelle.

2. Ueber die Moeglichkeiten der ASIC- Technologie sind lediglich ueberschlaegige Kenntnisse vorhanden. Es wird davon ausgegangen, dass es auch hier Iterationen geben muss:
   Die Basistechnologie wird stets als gegeben angesehen.
   Hinsichtlich bestimmter Parameter (z. B. der RAM- Groesse auf einem Chip) sollte man hingegen durchaus auf Aenderungen draengen, sofern der potentielle Erfolg dies rechtfertigt.

4.1.2. Bemerkungen zur "Systemphilosophie"

1.

Grundsaetzlich geht es darum, informationsverarbeitende Systeme der industriellen Elektronik, vorzugsweise fuer Maschinensteuerungen, zu entwickeln.
Die Systemstruktur basiert auf ASIC, die intern so auszugestalten sind, dass komplette Subsysteme auf einer einzigen (moeglichst kleinen) Leiterplatte Platz finden und dass neben dem eigentlichen ASIC nur noch die unumgaenglich notwendigen Schaltmittel fuer die direkte Prozesskopplung sowie lokale Speichermittel notwendig sind.

2.

Zur Kopplung zwischen den Subsystemen sind serielle Verbindungen vorzusehen, die auch eine galvanische Trennung (Optokoppler, Lichtwellenleiter) ermoeglichen.
Die Steuermittel dafuer sind in den ASIC anzuordnen.
Jeder ASIC sollte zwei unabhaengige serielle Schnittstellen treiben koennen: eine fuer die Verbindung zum uebergeordneten System und eine zu nachgeordneten Subsystemen.
Der vermutlich sinnvollste Ansatz ist ein LAN nach dem "token passing"- Prinzip in weitgehender Anlehnung an eingefuehrte Industriestandards (z. B. MAP).
Wichtig: es kommt darauf an, auf Basis aller zugaenglichen Erkenntnisse einen eigenen, in sich wohldefinierten Standard zu schaffen; man kann es sich nicht leisten, sich zu einem gegebenen Standard zu bekennen, auf den man selbst keinen Einfluss hat und der einem nicht beeinflussbaren Aenderungsgeschehen unterworfen ist.
Zur Verbindung mit Fremdsystemen sind entsprechende Koppeleinheiten vorzusehen, bei denen besonderer Wert auf freie Programmierbarkeit zu legen ist.

3.

Die Grundlage der ASIC- Strukturen ist ein frei programmierbarer Prozessor mit einer RISC- Befehlsliste, die fuer Steuerungsaufgaben und zur Emulation ueblicher Mikroprozessor- Befehlssaetze optimiert ist.

4.

Jeder ASIC enthaelt einen Prozessor- Kernel, Speicheradapter, serielle Anschluss- Steuerungen sowie anwendungsspezifisch Ein-Ausgabe- Schnittstellen.
Fuer all diese Funktionskomplexe gibt eine gewisse Auswahl. Im besonderen gibt es Prozessor- Kernel mit verschiedener Leistungsfaehigkeit, z. T. auch fuer "subsets" der Befehlsliste.

5.

Die Ergaenzung der Prozessor- Kernel durch Coprozessoren ist ausdruecklich vorgesehen (die jeweiligen Coprozessoren werden auf dem selben ASIC untergebracht).

6.

Jeder Prozessor- Kernel hat ein genau definiertes Leitungssystem (Interface) fuer die Verbindung zu den anderen Funktionseinheiten, im besonderen fuer den Anschluss von E/A- Schaltungen.

7.

Es wird besonderer Wert darauf gelegt, moeglichst viele Probleme der Prozesskopplung (bzw. allgemein der Ein- und Ausgabe) mit standardisierten Schaltmitteln zu erledigen.
Im besonderen soll die Mehrzahl dieser Probleme durch direkte Kopplung der E/A- Leitungen mit dem Prozessor- Kernel geloest werden. Prinzip:
Nichts ist flexibler als ein frei programmierbarer Prozessor.
Ob sich damit (d. h. durch befehlsseitiges Abfragen und Stellen von Leitungen) ein Steuerungsproblem loesen laesst, haengt lediglich von den zeitlichen Gegebenheiten ab.
Deshalb erscheint der Ansatz aussichtsreich, die E/A- Schaltungen selbst so einfach wie moeglich auszulegen und die gegebenen Ressourcen ("real estate") des ASIC weitgehend fuer einen extrem schnellen Prozessor- Kernel zu nutzen.

8.

Die ASIC sind fuer den Aufbau redundanter Anordnungen auszubilden (z. B. fuer das "Master/Checker"- Prinzip analog iAPX 432); die "scan/set"- Vorkehrungen sind fuer die Systemdiagnose nutzbar zu machen (Diagnose- Befehle, spezielle Protokolle fuer die seriellen Schnittstellen, evtl. ein "orthogonales" Diagnose- Interface). An den Subsystemen (Leiterplatten) ist eine eindeutige Fehleranzeige vorzusehen.

4.2. Prozessorstrukturen

4.2.1. Architektur und Befehlsliste

1.

Die Befehlsliste ist an den Prinzipien von RISC- Architekturen orientiert.
Es gibt nur drei grundlegende Befehlstypen:

1. Arithmetisch/logische Funktionen (ALF)
2. Verzweigungen (BRANCH)
3. Ausuebung von Steuerwirkungen (CTL.)

Jeder Befehl ist 32 bit lang.
Es wird grundsaetzlich auf moeglichst leistungsfaehige elementar wirkende Befehle orientiert, und es wird unterstellt, dass das System vorzugsweise in hoeheren Sprachen programmiert werden wird, so dass ein "missbraeuchliches" Nutzen von Befehlen durch den Compiler verhindert wird.

2.

Es handelt sich grundsaetzlich um eine 32- bit- Architektur mit interner 36- bit- Verarbeitung.
Die Befehle koennen jeweils 32 interne Register (Lokalspeicher-Zellen) zu je 36 bit ansprechen.
Diese 36 bit sind konzeptionell aufgeteilt in 4 TAG- Bits und 32 Informationsbits.
In welcher Breite die Daten jeweils verarbeitet werden, wird sowohl durch FORMAT- Felder in den Befehlen als auch durch die TAG- Bits gesteuert.
Im einzelnen sind folgende Formate moeglich:

- 16- bit- Halbworte
- 32- bit- Worte ohne TAG- Interpretation
- 32- bit- Worte mit TAG- Interpretation zwecks Adressierung
- 32- bit- Worte mit TAG- Interpretation fuer E/A
- 36- bit- Worte.

Der Adressierung liegen stets 32- bit- Worte zugrunde, denen zwei weitere Bits zwecks Segment- Auswahl vorangestellt sind (als Vorgriff auf kuenftige Erweiterungen der Architektur).
Manche Implementierungen bieten ueber ihre Anschluesse nur einen Teil der Adresse an (z. B. 20 bit).

3.
Es gibt keine besonderen E/A- Befehle. Vielmehr werden E/A-Schnittstellen wie Register angesprochen (es wird z. B. ein Register adressiert, dessen TAG- Bits eine sinngemaesse Ansteuerung der besagten Leitungen bewirken).

4.
Alle architekturspezifischen Register (z. B. der Befehlszaehler) sind Teile des Lokalspeichers und ueber feste Register- Adressen befehlsseitig zugaenglich.
Im besonderen ist ein Steuerregister (CONTROL- Register) vorgesehen, das alle wesentlichen Zustaende (Flags usw.) enthaelt.

5.
Die Architektur unterstuetzt ein hardwareseitiges Multitasking: der Lokalspeicher kann an sich beliebig gross sein; die Basisadresse fuer den aktuellen Registersatz ist im CONTROL-Register vorgesehen. Somit ist durch hardwareseitiges Umschalten dieses Registers ein Umschalten zwischen verschiedenen unabhaengigen Programmniveaus moeglich.

6.
Grundsaetzlich wird Wert darauf gelegt, dass jeder Speicherzyklus so effektiv wie moeglich genutzt werden kann. Da (jedenfalls in der ueberwiegenden Zahl der Faelle) fuer Befehle und Daten der selbe Speicher- Datenpfad benutzt wird, ist fuer einen moeglichst lueckenlosen Datenfluss Sorge zu tragen. Dies bedeutet eine Abkehr vom RISC- Prinzip in dem Sinne, dass das unnoetige Holen von Befehlen weitgehend zu vermeiden ist. An sich bekannte elementare Datenbewegungen sollten moeglichst von einem einzigen Befehl gesteuert werden. Dazu sind vorgesehen:
- TAG- Bits im Lokalspeicher zur Ausloesung von zusaetzlichen Wirkungen, die nicht im Befehlsformat codiert werden koennen.
- Blockbefehle, die ggf. Registerinhalte als "Steuerworte" nutzen.

So gibt es keine besonderen RR- Befehle, RS- Befehle usw. Ob Daten direkt aus Registern entnommen oder im Speicher adressiert werden, wird vielmehr durch TAG- Bits gesteuert. Damit kosten uebliche Adressenmodifikationen keine zusaetzlichen Speicherzugriffe fuer die sonst noetigen Modifikationsbefehle.

7.
Externe Speicher werden grundsaetzlich wortweise (32 bit) adressiert.
Register koennen wort- oder halbwortweise (16 bit) adressiert werden.
Es gibt keine Byte- Adressierung.
Fuer Multiplikation und Division sind (i. S. der RISC- Prinzipien) elementare Schritte als Befehle vorgesehen.
Zwischen den Registern sind Transporte und arithmetisch/logische Operationen beliebigen Formates von beliebigen Bitadressen zu beliebigen Bitadressen moeglich. Dies ist auch dynamisch steuerbar.
Das Verzweigen gemaess Einzelbit- Abfrage ist vorgesehen.
Alle komplexeren Entscheidungs- Operationen, die sich auf das Auswerten von BOOLEschen Gleichungen zurueckfuehren lassen, werden mittels Durchmusterung von Ternaervektorlisten (TVL) erledigt. Dafuer sind entsprechende Befehle vorgesehen.
Es ist moeglich, Befehle aus dem Lokalspeicher abzuarbeiten.

8.

Die Befehlsformate sind weitgehend so festgelegt, dass bei Implementierungen mit 16- bzw. 20- bit- Datenpfaden nach dem Holen des ersten Halbwortes bereits die ersten Teilablaeufe gestartet werden koennen.
Grundsaetzlich sind ausreichende Code- Reserven fuer kuenftige Erweiterungen vorgesehen.

9.

Es gibt einen speziellen Befehl, der es ermoeglicht, die aktuelle Implementierung programmseitig abzufragen.

4.2.2. Implementierungen

Es sind verschiedene Prozessor- Kernel vorzusehen, die je nach Vorzugs- Anwendung in Aufwand und Leistungsvermoegen unterschiedlich ausgelegt sind. Diese Kernel muessen jeweils durch angemessene Adapterschaltungen ergaenzt werden.
Im folgenden seien einige moegliche Auslegungen angefuehrt (Hinweis: Jede Ausfuehrung enthaelt stets 2 serielle Schnittstellen fuer die Systemkopplung):

1.

Verarbeitungsprozessor mit Befehls- Cache.
Dieser ist fuer maximale Verarbeitungsgeschwindigkeit (als RISC-Maschine) ausgelegt. Der Cache ist extern vorgesehen und hat separate 16- Bit- Adressen- und 32- Bit- Datenwege. Ob eine rein softwaremaessige Cache- Verwaltung vorgesehen wird oder eine ASIC-interne Hardware- Unterstuetzung (vgl. etwa Intel 386), muss weiteren Untersuchungen vorbehalten bleiben.
E/A- seitig ist ein elementares Bussystem (z. B. AT- Standard) fuer periphere Anschluesse vorgesehen.

2.

Prozessor zur Mikroprozessor- Emulation.
Dieser verfuegt ueber einen internen ROM (z. B. 1 k Worte zu 48 bit), der die Emulationsprogramme enthaelt.
Als sinnvolles Ziel der Emulation bietet sich eigentlich nur der Intel 8086 an.
E/A- seitig sollte ein elementares Bussystem vorgesehen werden. Freie Anschluesse sind fuer die universelle Nutzung zugaenglich zu machen (Abfragen bzw. Belegen seitens des Prozessors).
Ein solcher Prozessor koennte mehrere virtuelle 8086 unabhaengig voneinander simultan emulieren. Man kann damit z. B. unabhaengige Tasks unter MS/DOS fahren. Mittels der Coprozessor- Befehle des 8086 ist ein Umschalten in den direkten RISC- Modus moeglich (der Schaltkreis enthaelt also faktisch seinen eigenen Coprozessor fuer die Faelle, wo die Emulation zu langsam ist).
Hinweis: Die Befehle im ROM enthalten eine Erweiterung zur direkten Auswahl der Folge- Befehlsadresse (mit 4- fach-Verzweigung). Es ist aber auch grundsaetzlich moeglich, solche Emulator- Routinen aus dem normalen Speicher abzuarbeiten. Fuer die Verzweigungen sind dann sinngemaess EMULATOR BRANCH- Befehle einzufuegen.

3.

Universeller Prozessor fuer Steuerzwecke.
Dieser hat einen 16- bit- Datenpfad, um moeglichst viele Anschluesse fuer E/A- Zwecke verfuegbar zu haben. Er ist fuer die effektive E/A- Steuerung optimiert (u. U. auch in Form eines "subsets" der Befehlsliste).

4.

Spezialisierter E/A- Prozessor.
Siehe dazu Abschnitt 4.3.2.

4.3. Ein- und Ausgabe

4.3.1. Grundsaetzliches

1.

Allgemeines Ziel: es darf keine "Restlogik" mehr geben; die unumgaenglichen Schaltmittel fuer die Prozesskopplung muessen sich direkt an das ASIC anschliessen lassen.

2.

Die ASIC- Pins sind weitgehend auszunutzen, und zwar so, dass sich Koppelbaustufen auf einfachste Weise anschliessen lassen.
Fuer die Mehrfachnutzung von Anschluessen gilt:

1. Eingaenge sind nicht mehrfach nutzbar, es sei denn, es wird eine externe Auswahl vorgesehen. Dies erfordert aber zusaetzliche Schaltmittel und kann somit keine Vorzugsloesung sein.

2. Ausgaenge sind mehrfach nutzbar, sofern sie Datensignale fuehren, die zur Uebernahme in externe Register bestimmt sind.

So koennen die Daten- und Adressenleitungen zum Speicher als Daten- Ausgangsleitungen benutzt werden.

3.

Wesentliche Anforderungen im Sinne eines hohen Leistungsvermoegens sind:

1. Gleichzeitiges Erregen moeglichst vieler Ausgangssignale.

2. Flexibles und schnelles Abfragen vieler Eingangssignale, um kurze Reaktionszeiten fuer die programmseitige Verarbeitung zu gewaehrleisten.

Um der Forderung (1) gerecht zu werden, ist vorgesehen, dass mit einem Befehl 36- Bit- Worte an eine von 3 Bestimmungen ausgegeben werden koennen (der Prozessor belegt seinen Ausgabe- Bus und erregt eine von 3 Strobe- Leitungen).
An den physischen Ausgaengen des ASIC gibt es folgende Konfigurations- Moeglichkeiten:

a) Daten zwecks Uebernahme in externe Register (nur gueltig mit Strobe)
b) statische Daten

c) selektive Strobe- Impulse (gebildet durch ASIC- interne Verknuepfungen von Datenbit- Positionen mit Strobe- Impulsen); damit koennen z. B. "fertige" Strobes im 1-aus-n-Code als Register- Uebernahme- Impulse abgegeben werden, so dass eine externe Strobe- Decodierung unnoetig ist.
d) interne Strobes des Prozessors, die fuer externe Nutzung zugaenglich sind.

Charakteristisch ist weiterhin, dass durch die Organisation der Ausgabe ("memory mapped" bezueglich des Lokalspeichers) selektive Operationen mit beliebigen Ausgabeleitungen direkt programmierbar sind; im besonderen koennen Ausgangs- Belegungen programmseitig gelesen werden (tatsaechlich werden deren Register- Kopien gelesen).

Um Forderung (2) zu erfuellen, muessen Abfrage- oder Interrupt-Mechanismen vorgesehen werden. Vorschlaege dazu:

a) Fuer jede Task (i. S. des hardwareseitigen Multitasking) ist eine Eingangsleitung konfigurierbar, deren Erregung die Umschaltung zur betreffenden Task bewirkt (Interrupt- Ausloesung).

b) Dies wird dadurch erweitert, dass logische Verknuepfungen mehrerer Eingangsbelegungen angegeben werden koennen.

c) Eine sinnvolle Verknuepfung ist z. B. das Testen auf Orthogonalitaet mit einem ternaeren Vektor. Damit laesst sich auch eine "Ausloesung bei Aenderung" programmieren.

4.

Universalitaet und Leistungsvermoegen (i. S. von "Datendurchsatz") sind entgegengesetzte Ziele.
Spezialisierte Strukturen haben ein betraechtlich hoeheres Leistungsvermoegen (fuer den betreffenden Zweck) als universelle. Ob dieses Leistungsvermoegen, ueberhaupt benoetigt wird, kann nur durch Analyse konkreter Anwendungsfaelle festgestellt werden.

4.3.2. Universelle E/A- Strukturen

Es gibt folgende Prinzipien, nach denen universelle E/A-Strukturen aufgebaut werden koennen:

1. Programmierbare (bzw. konfigurierbare) Anordnungen fuer bestimmte Klassen von E/A- Operationen; in Analogie zu ueblichen Mikroprozessorsystemen, vgl. etwa SIO, PIO, CTC usw.

2. Automatentheoretischer Ansatz: die auszuwertenden bzw. zu steuernden Leitungen werden als Ein- und Ausgaenge eines abstrakten Automaten angesehen, und das gewuenschte Verhalten wird in irgend einer geeigneten Form durch die Ueberfuehrungs- und Ausgangsfunktion beschrieben. Die Hardware interpretiert diese Beschreibung.

3. E/A- Prozessor. Die Ein- und Ausgangsleitungen sind direkt an einen frei programmierbaren Prozessor angeschlossen, und das gewuenschte Verhalten wird ausschliesslich programmtechnisch realisiert.

zu 1.: programmierbare E/A- Schaltungen

Die technische Ausgestaltung wird durch Vorbilder aus verschiedenen Mikroprozessor- Familien nahegelegt, so dass es an sich keine offenen Grundsatzfragen geben duerfte.
Was die Sinnfaelligkeit an sich anlangt, so gibt es zu bedenken:

a) die einzelne Anordnung ist in sich recht spezialisiert (z. B. auf serielle Ein/Ausgabe, auf Zeitzaehlung usw.).
Das gewaehrleistet eine vergleichsweise hohe Leistung fuer diese Klasse von Einsatzfaellen.
Andererseits laesst sich auf einem ASIC- Typ nur ein bestimmter "Mix" solcher Schaltungen vorsehen, und es ist fraglich, ob damit die gewuenschte Universalitaet des ASIC zu gewaehrleisten ist.

b) die Programmier- bzw. Konfigurierbarkeit erfordert einen beachtlichen Aufwand an Schaltmitteln (so ist bei Z 80 die SIO aufwendiger als die CPU). Der ueberwiegende Teil dieser Schaltmittel (ASIC "real estate") traegt in der konkreten Anwendung nichts zur Systemleistung bei.

zu 2.: Automatentheoretischer Ansatz

Es ist eine Vielzahl von "Bitprozessoren", "Sequencern", "Algorithmic state machines" usw. bekannt. Deren Prinzipien koennen studiert und sinngemaess schaltungstechnisch umgesetzt werden.
All diesen Loesungen ist gemeinsam, dass die Ausgangs- und Ueberfuehrungsfunktionen des Automaten durch Programme emuliert werden, wobei Prozessor- Architektur, Befehlsliste und Ablauforganisation auf diesen Zweck hin abgestimmt sind.
Zusaetzlich sind 2 "radikale" Ansaetze zu betrachten:

a) "naive" Implementierung durch Zuordner (ROM bzw. RAM), die die Ueberfuehrungs- und Ausgangsfunktionen des Automaten direkt enthalten.

b) Speicherung der kompletten "Phasenliste" (die das Verhalten des Automaten vollstaendig beschreibt) als Ternaervektorliste (TVL) und deren Durchmusterung, um Folgezustand und Ausgangsbelegung zu ermitteln.

Das Verfahren a) ermoeglicht fuer an sich beliebig komplizierte Ausgangs- und Ueberfuehrungsfunktionen kuerzeste Reaktionszeiten (jeweils 1 Speicherzyklus), ist aber nur dann praktikabel, wenn die Anzahl der Eingangs- und Zustandssignale einen bestimmten Wert nicht uebersteigt (insgesamt etwa 10...hoechstens 16). Der Aufwand an weiteren Schaltmitteln (ausser Speicher) ist minimal.

Das Verfahren b) ist bei geeigneter technischer Ausgestaltung allen anderen Loesungen dann ueberlegen, wenn das Automatenverhalten hinreichend "kompliziert" ist. (Man stelle sich dazu vor, dass dieses Verhalten durch BOOLEsche Gleichungen beschrieben ist. Die Anzahl der Variablen und Verknuepfungen laesst bereits intuitiv Schluesse auf die Kompliziertheit zu.)

zu 3.: E/A- Prozessor

Das Prinzip laesst ein Hoechstmass an Flexibilitaet erwarten, ist aber hinsichtlich des Zeitverhaltens nur schwer zu bewerten.
Weiterhin ist es nicht trivial, mehrere E/A- Operationen simultan zu steuern; somit ist der Rueckgriff auf die elementare E/A-Programmierung nicht ohne weiteres ein Ersatz fuer die Anordnung mehrerer spezialisierter Einrichtungen, die autonom arbeiten (wie dies bei ueblichen Mikroprozessoren gegeben ist).

Ein Ausweg ist durch das hardware- unterstuetzte Multitasking gegeben: fuer jede separate E/A- Operation wird eine unabhaengige Task vorgesehen. Um definierte Zeitverhaeltnisse zu gewaehrleisten, bietet sich das Prinzip der zeitstarren Umschaltung an. Dabei wird im festen Rhythmus zwischen den einzelnen Tasks umgeschaltet, und zwar zweckmaessigerweise nach jedem Maschinenbefehl bzw. nach jedem Speicherzugriff fuer Daten (infolge der RISC- Betriebsweise sind diese Zugriffe jeweils gleich lang). Man hat dann Umschaltzeiten zwischen 100 und etwa 800 ns (je nach Implementierung). Das Prinzip, durch extrem schnelles Umschalten einer Hardware zwischen verschiedenen Befehlsstroemen mehrere "virtuelle" Prozessoren fuer E/A- Zwecke bereitzustellen, wurde erstmalig beim Grossrechner CDC 6600 (1964) realisiert.
Wenn ein Umschalten im Rhythmus von 100 ns gelingt, haette man bei 8 Tasks 8 virtuelle Prozessoren mit jeweils 800 ns Befehlszyklus; damit waere jeder virtuelle Prozessor einem 4 MHz- Z 80 noch deutlich ueberlegen.

Problem:
Fuer den Prozessor- Kernel werden keine besonderen Schwierigkeiten gesehen.
Hingegen erfordert ein Umschalten im Intervall von 100 ns entsprechend aufwendige Speichermittel (deren Anschluss weiterhin eine betraechtliche Anzahl der ASIC- Pins belegt).
Dies koennen - aus der Sicht des Systems - schwerwiegende Nachteile sein.

Vorschlaege:

1.
Es wird Wert darauf gelegt, BOOLEsche Gleichungen effizient nutzen zu koennen, und zwar durch folgende Massnahmen:

a) Nutzung der direkten Abbildung durch Adressierung gespeicherter Wahrheitstabellen (auch im Lokalspeicher); dies erfordert entsprechende Abfrage- Befehle

b) Verknuepfungsoperationen mit Ternaervektoren: "Testen auf Orthogonalitaet" als Verallgemeinerung des ueblichen "Vergleichen unter Maskierung"

c) Durchmusterungsablaeufe mit TVL ("Durchsuchen nach Orthogonalitaet bzw. Nichtorthogonalitaet; auch im Lokalspeicher); damit werden alle "zu komplizierten" Verknuepfungen erledigt

d) Emulation eines "Bitprozessors".

Im Sinne der RISC- Prinzipien werden die leistungsentscheidenden Ablaeufe als besondere Befehle vorgesehen; die Verbindungen zwischen diesen Ablaeufen sind mit elementaren Befehlen zu realisieren.

2.

Ein ASIC mit 16- Bit- Datenweg zum Speicher haette bei Einsatz ueblicher PROMs eine Befehlsausfuehrungszeit von etwa 800 ns, das sind bei 8 Tasks 6,4 us fuer jeden virtuellen Prozessor (je weniger Tasks, desto hoeher die Geschwindigkeit im einzelnen).

Um dies zu umgehen, wird vorgeschlagen:

Einflussnahme auf die grundsaetzliche ASIC- Auslegung dahin, dass fuer einen Typ der interne RAM vergroessert werden kann (z. B. auf 16 kbit bzw. 512 36- bit- Worte). Damit laesst sich ein E/A-Prozessor gestalten, bei dem etwa 100 Pins ausschliesslich fuer E/A- Anschluesse verfuegbar sind und der eine Befehlsausfuehrungszeit von 100...200 ns gewaehrleistet.
Zusaetzlich zum RAM sollte ein interner ROM vorgesehen sein (fuer: Steuerung der seriellen Anschluesse, universell nutzbare Unterprogramme zu Emulationszwecken - einschliesslich TVL-Durchmusterung und Bitprozessor - , Zeitgeberfunktionen usw.).
Ein solcher Prozessor ist auf niederen Ebenen des Systems als E/A-Subsystem einzusetzen. Er bezieht seine konkreten Steuerprogramme ueber die serielle Schnittstelle. Diese Steuerprogramme werden aus dem Lokalspeicher abgearbeitet. Wenn man annimmt, dass vielleicht 300...400 Befehle unterzubringen sind, so ist abzuschaetzen, dass bei der Leistungsfaehigkeit der Befehlsliste damit in vielen Faellen Aufgaben der Steuerung von Prozess- Interfaces zu loesen sind, und zwar so, dass die physischen Leitungen in Realzeit bedient werden koennen und dass die Kommunikation zu uebergeordneten Systemen auf Basis der Meldung von Ereignissen und der Uebernahme von Auftraegen organisiert ist.
Da der Prozessor keine externen Speichermittel braucht, ist es praktikabel, eine groessere Anzahl davon einzusetzen.

4.4. Aufgabenstellungen (vorlaeufige Uebersicht)

1. Prozessor- Architektur
1.1. Grundprinzipien
1.2. Befehlsliste
1.3. Erweiterungen
1.4. Coprozessoren
1.5. E/A- Kopplung an Kernel
1.6. Vorkehrungen fuer Emulation

2. System- Architektur
2.1. Anwendungs- Erfordernisse
2.2. Entwicklungs- bzw. Programmier- Umgebungen
2.3. Ziele fuer Emulation
2.4. Peripherie
2.5. serielle Anschluesse (TOKEN RING LAN)
2.6. Speichermittel
2.7. Prozesskopplungen
2.8. Diagnose und Wartung

3. Schaltkreis- Entwurf
3.1. Definition des Typenspektrums
3.2. Einflussnahme auf Standardzellen- Gestaltung
3.3. elementarer Prozessor- Kernel
3.4. Kernel- Ergaenzungen
3.5. Speicheradapter
3.6. Befehlslese- Anordnungen
3.7. E/A- Kopplung an Kernel
3.8. spezifische E/A- Schaltungen
3.9. Schaltmittel fuer seriellen Anschluss
3.10. Diagnostik und Wartung
3.11. Spezieller E/A- Prozessor