

Die Gestaltung von Informationssystemen aus der Sicht eines Logikentwerfers beim Hardwareentwurf von DV-Anlagen des ESER

WOLFGANG LAMPENSCHERF

wolani@web.de

Im Beitrag soll an Beispielen deutlich gemacht werden, welche Überlegungen von Logikentwerfern beim VEB Kombinat Robotron in Karl Marx Stadt (Chemnitz) den Hardwareentwurf von ESER-DV-Anlagen bestimmt haben bzw. beeinflussen konnten. Nach der Darstellung der grundlegenden Entscheidungen für die Entwicklungslinie des ESER werden verschiedene Aufgabenkomplexe beim Hardwareentwurf sowie die einzuhaltenden Randbedingungen und einige Aspekte der verbliebenen Freiheitsgrade für die Logikentwerfer angeführt. Dabei ist das Spannungsfeld zwischen Kreativität und Innovationslust der Logikentwerfer auf der einen Seite und der mehr oder weniger restriktiven Leitung des Entwurfs- bzw. Entwicklungsprozesses sowie der begrenzten Zeit- und Materialressourcen andererseits Gegenstand der Betrachtung. An einigen Beispielen werden die aus diesem Spannungsfeld resultierende Arbeitsweise sowie erzielte Ergebnisse dargestellt. Abschließende Bemerkungen charakterisieren die Möglichkeiten und Probleme der Zusammenarbeit von Hardware- und Softwareentwicklern im Hause Robotron in Karl-Marx-Stadt.

1 Die strategischen Entscheidungen und Konzepte für die Entwicklung von ESER-DV-Anlagen in der DDR

Die strategisch wichtigen Entscheidungen, die den gesamten Entwicklungsprozess von ESER-DV-Anlagen im VEB Robotron bestimmt haben, sowie die dazugehörigen Hintergrundinformationen sind in [2] ausführlich dargestellt. Hier sei nur noch einmal wiederholt, dass die funktionelle Basis der ESER-DV-Anlagen der Reihe 2 (siehe [1]) sich an den Operationsprinzipien des Systems 370 von IBM orientierte. Die Daten- und Programmkompatibilität zu diesen IBM-Anlagen war zu gewährleisten. (Diese Entscheidung bekam ungewollt für einige Hardware- bzw. Programmentwickler eine besondere Bedeutung, da sie ihre Arbeitsmöglichkeiten nach der „Wende“ entscheidend verbessert hat.)

Vor der Realisierung einer DV-Anlage sind in aller Regel diverse Konzepte zu erarbeiten, die die Orientierungsrichtlinien für den Entwicklungsprozess enthalten. Von diesen Konzepten seien hier erwähnt:

- das logisch funktionelle Konzept (LFK)
- das Konzept der technischen Realisierung
- das Softwarekonzept
- das Fertigungs- und Vertriebskonzept
- das Wartungs- Reparatur- und Anwenderbetreuungskonzept
- das Ein-Ausgabegerätekonzept

Da Logikentwerfer für den funktionellen Teil einer DV-Anlage zuständig sind, ist für sie im Allgemeinen das LFK das wichtigste Dokument. In dieser recht umfangreichen Schrift werden die zu schaffenden Verarbeitungs- und Datenspeicherfunktionen im Rahmen der Operationsprinzipien detailliert beschrieben. Die Aufgabe für die Logikentwerfer besteht nun darin, auf der Basis der zur Verfügung stehenden Bauelemente, die beschriebenen Funktionen zu realisieren. Dabei sind die Richtlinien und Vorgaben aus dem Konzept der technischen Realisierung zu berücksichtigen. Die jeweiligen technologischen Möglichkeiten, die der Bauelementehersteller beherrscht, bestimmen dabei die erzielbare Packungsdichte, die gesicherten Schaltzeiten sowie die auftretende Verlustleistung in Verbindung mit dem Kühlungsbedarf der Bauelemente. Es ist leicht einzusehen, dass auf dieser Basis die Grenzen für die erzielbare Leistungsfähigkeit einer DV-Anlage weitgehend festgelegt sind. In aller Regel erfordert eine höhere Leistung auch einen erhöhten Schaltkreisaufwand, z. B. durch die Vergrößerung der Verarbeitungsbreite oder auch durch die Parallelisierung von Verarbeitungs- und Steuerungsabläufen.

Um hier zu tragbaren Kompromissen zu kommen, wird üblicherweise eine zusätzliche Steuerungsebene, die Mikroprogrammierung, eingesetzt. Es handelt sich dabei um eine spezielle auf den jeweiligen DV-Anlagentyp bezogene interne Steuerung, die den Ablauf der Verarbeitung in den eigentlichen Bauelementen kontrolliert. Der für die Realisierung der Mikroprogrammsteuerung zusätzlich erforderliche Bauelementeaufwand hält sich in Grenzen, bietet aber die Möglichkeit zur Schaffung einer universellen Funktionalität der DV-Anlage. Eingedenk dieser Vorgehensweise versteht der Logikentwerfer unter Hardware sowohl die eigentliche Bauelementebasis als auch die Mikroprogramme (mit Mikroprogrammsteuerwerk) mit deren Hilfe die Schaltkreise gesteuert werden. So gesehen besteht z. B. die Ausführung eines Additionsbefehles aus dem Ablauf eines Mikroprogrammes, welches die Durchführung der Additionsfunktionalität einschließlich der Adressenbildung und der Operandenzugriffe zum Hauptspeicher auf der „eigentlichen“ Hardware steuert. Es liegt in der Natur der Sache, dass Mikroprogramme nur in Verbindung mit dem für sie bestimmten Schaltkreisentwurf sinnvoll verwendet werden können.

Die je nach zu realisierender Funktion unterschiedlich langen und komplexen Mikroprogramme werden in einem Mikroprogrammspeicher, der früher häufig als ROM ausgeführt war, untergebracht. Die in der Folgezeit eingeführten ladbaren Mikroprogrammspeicher stellten aber ein weites Feld für eine erweiterte Funktionalität einer DV-Anlage dar, und wurden in diesem Sinn auch zu einer Herausforderung für die Kreativität von Logikentwerfern.

Für seine Arbeit steht dem Logikentwerfer üblicherweise ein leistungsfähiges Unterstützungssystem zur Verfügung. Es handelt sich dabei z. B. um ein Simulationsprogramm, mit dessen Hilfe die Funktionen der entworfenen Logikkomplexe einschließlich der dazugehörigen Mikroprogramme simuliert werden können. Ein sehr großer Teil von Entwurfsfehlern lässt sich damit bereits in der Frühphase des Entwicklungsprozesses finden und korrigieren. Der neue Rechner wird gewissermaßen auf einer bereits bestehenden DV-Anlage als Softwarelösung dargestellt und getestet. Bei seiner Entwurfsarbeit hat der Logikentwerfer neben dem Bestreben nach möglichst geringem Schaltungsaufwand auch auf die Platzierung seiner Logik auf Steckeinheiten zu achten. Signallaufzeiten, der Anschluss an die zu verwendende Taktsteuerung, aber auch die Erreichbarkeit für das Fehlererkennungs- und -maßnahmesystem spielen dabei eine wichtige Rolle. Die umrissenen Aufgaben gehören zur „Pflicht“ bei der Logikentwurfsarbeit.

2 Gestaltungsräume der Logikentwerfer

Neben der pflichtgemäßen Realisierung aller im LFK spezifizierten Verarbeitungs-, Speicher- und Steuerfunktionen der zu entwickelnden DV-Anlage ergeben sich für den Logikentwerfer gewissermaßen als „Kür“ weitere Gestaltungsspielräume. Vorzugsweise betreffen diese Spielräume solche Systemkomponenten, deren Existenz nicht vorgeschrieben oder deren Funktionen nicht exakt definiert sind. Folgende Beispiele aus dem persönlichen Erfahrungsbereich des Autors sollen betrachtet werden:

- Die Befehlsvorbereitungseinheit
- Schaltkomplexe zur Fehlererkennung und Systemdiagnose
- Funktionserweiterungen auf der Basis von Mikroprogrammen

Diese Liste umfasst nur einen Ausschnitt aus den vorhandenen Möglichkeiten. Größere Objekte, wie z. B. der Matrixmodul oder der Bedien- und Serviceprozessor, erfordern im Allgemeinen einen separaten Entwicklungsprozess und sollen hier nicht Gegenstand der Betrachtung sein (siehe [4], [5]).

2.1 Die Befehlsvorbereitungseinheit (BVE)

Zur Abarbeitungsfolge von Programmbefehlen in einer DV-Anlage gehören unter anderem solche Aktivitäten wie das Lesen der Befehle und der zu verarbeitenden Operanden aus dem Hauptspeicher mit vorhergehenden eventuell notwendigen Adressenmodifikationen. Erst nach Ausführung dieser Vorbereitungen kann die eigentliche Befehlsausführung, die Operandenverknüpfung, begonnen werden. Die Idee einer BVE besteht nun darin, diese Vorbereitungsaktivitäten bereits während der Befehlsausführung von vorangehenden Befehlen ablaufen zu lassen. Durch diese Parallelarbeit bei der Befehlsausführung sollte sich ein bemerkbarer Gewinn für die Geschwindigkeit der Befehlsabarbeitung erzielen lassen.

Die Größe des erzielbaren Gewinns hängt jedoch nicht nur von der „Tiefe“ der realisierten Parallelisierung (wie viele Befehle können im Vorgriff aufbereitet werden) ab, sondern auch von der Anzahl der Fälle, in denen eine bereits erfolgte Befehlsvorbereitung wieder verworfen werden muss. Ein solcher Fall liegt z. B. dann vor, wenn durch einen Befehl Adressen oder Operanden modifiziert werden, die in einem bereits vorbereiteten Befehl in noch unmodifizierter Form benutzt wurden. Solche Koinzidenzen müssen erkannt werden und eine Serialisierung auslösen, die eine korrekte Befehlsausführung gewährleistet.

Da die Operationsprinzipien lediglich die korrekte Ausführung der Befehle in einer Befehlsfolge fordern, ergibt sich hier ein Gestaltungsraum für den kreativen Logikentwerfer. Die Variation der Parallelisierungstiefe aber auch die Verdopplung der Befehlsvorbereitungslinien beim Erkennen von bedingten Verzweigungsbefehlen bieten sich an. Das Ziel ist die Vermeidung möglichst vieler Wartezyklen, die bei der Befehlsabarbeitung entstehen können.

Die im VEB Robotron Karl-Marx-Stadt entwickelten ESER-DV-Anlagen verfügen über Befehlsvorbereitungseinheiten.

In diesem Zusammenhang sei auf einen Gesichtspunkt hingewiesen, der die nach meiner Meinung unzureichende Zusammenarbeit von Hard- und Softwareentwicklern betrifft. Softwareentwickler (Betriebssystem-, Compilerentwickler im Besonderen) sollten von der Möglichkeit der Existenz und der Funktionsweise einer BVE wissen, um bei ihrer Arbeit die Fälle der erwähnten Koinzidenzen möglichst klein zu halten. Durch häufig mögliche, einfache, formale Umstellungen von Befehlen in einer Folge ließen sich bemerkbare Beschleunigungen der Befehlsausführung erreichen (siehe Abbildung 1). Eine Aufgabe für das Management von DV-Anlagenentwicklungen besteht also darin, die Kommunikation zwischen Hard- und Softwareentwicklern zu organisieren.

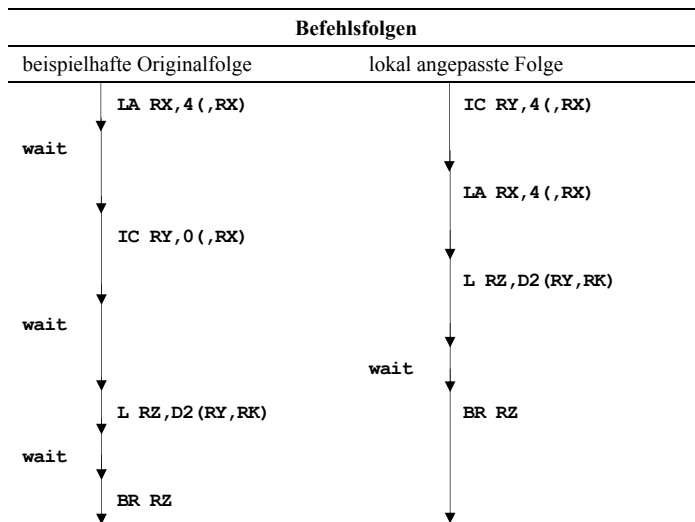


Abbildung 1: Lokale Anpassung von Befehlsfolgen

2.2 Verbesserungen des Fehlermaßnahmesystems

Die Gestaltung von Hardwarekomplexen des Fehlermaßnahmesystems unterliegt weitgehend den grundlegenden Entwurfsdirektiven. Prinzipiell eröffnet sich hier ein reiches Betätigungsfeld für Logikentwerfer. Die Spannweite reicht von fehlertoleranten Schaltungsgebilden in Mehrfachausführung mit Majoritätslogik bis zu eher „spartanischen“ Hardwarelösungen, die umfangreiche und zeitraubende Aktivitäten erfordern, um einen aufgetretenen Fehler zu diagnostizieren und zu beseitigen. Üblicherweise werden hier Kompromisslösungen gesucht, die durch Hard- und Softwareinsatz eine Fehlerlokalisierung bis auf Steckeneinheitenebene mit vertretbarem Zeitaufwand ermöglichen. Ein häufig verwendetes Verfahren besteht in der Bereitstellung geeigneter Testprogramme, die mit speziellen Testdatenbeständen die Funktionsfähigkeit der Hardware überprüfen. Der Nachteil dieses Verfahrens liegt darin begründet, dass zum Test nur die regulären Befehle mit den für sie definierten Operanden benutzt werden können. Bestimmte Hardwareteile sind jedoch auf diese Weise nur schwer erreichbar, weil sie eine Fehlfunktion erst nach dem Ablauf ganz spezieller Befehlsfolgen erkennen lassen. Um derartige Schaltungsteile auch unabhängig von der Ausführung bestimmter Befehle oder Befehlsfolgen überprüfen zu können, wurde ein Verfahren entwickelt, das sich vereinfacht wie folgt beschreiben lässt.

Im Regelfall besteht ein Schaltungskomplex aus einem Eingangsregister, welches mit den Eingangswerten für eine angeschlossene Verknüpfungslogik

geladen werden kann. Die Ergebnisse der Verknüpfung werden in einem Ausgangsregister aufgefangen. Die grundlegende Idee besteht nun darin, dass die Register neben dieser ihrer Normalfunktion auch über eine Schieberegisterfunktion verfügen (siehe Abbildung 2). Durch geeignete Zusammenschaltung der Register lässt sich somit für Testzwecke ein zum normalen Betrieb orthogonaler Datenweg nutzen. Über eine relativ einfache Steuerlogik lassen sich die Testdaten und die Ergebnisdaten seriell durch diesen Weg schieben und bieten die Möglichkeit, bis auf Gatterniveau zu testen. Die Taktversorgung T1 und T2 gewährleistet die Normalfunktion, während über die Schiebelinien E und A die Testeingangsdaten und die Testresultatdaten seriell verarbeitet werden können.

Obwohl diese als Diagnosebus bezeichnete Testeinrichtung bei ESER-DV-Anlagen nicht realisiert wurde, zeigt sich, dass für den Logikentwerfer viele Möglichkeiten existierten, mindestens Vorschläge für die Gestaltung einer Hardware auch über das Pflichtprogramm hinaus zu machen.

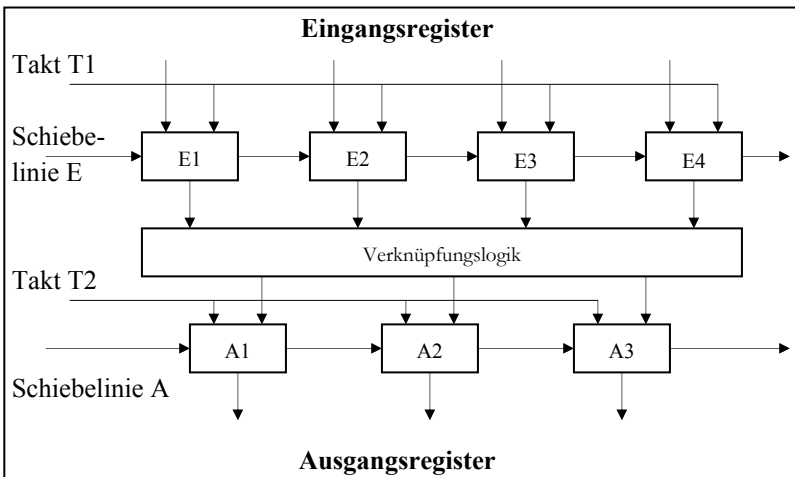


Abbildung 2: Diagnosebus (vereinfacht)

2.3 Funktionserweiterungen auf der Basis von Mikroprogrammen

Speziell durch den Übergang zu größeren Programm- bzw. Datenkomplexen wurden neue Konzepte für das Speichermanagement benötigt. Wie bekannt, wurde in diesem Zusammenhang das Konzept des virtuellen Speichers mit einer dynamischen Adressumsetzung entwickelt. Dabei werden virtuelle Programm- oder Operandenadressen über einen in der Hardware realisierten Umsetzungsmechanismus (DAT) in reale Hauptspeicheradressen umgewandelt.

Für die Anwendungsprogramme bleibt dieser Vorgang transparent. Welche Adressbereiche sehr großer Adressräume dabei im Hauptspeicher resident gehalten werden und welche auf Massenspeicher ausgelagert werden, obliegt der Steuerung des Betriebssystems. Eine Verallgemeinerung dieser „virtuellen“ Arbeitsweise wurde mit der Einführung des Systems virtueller Maschinen (SVM/ES) erreicht. Hier kann durch ein spezielles Betriebssystem die reale Hardware einer DV-Anlage mehreren Gastsystemen in jeweils einer virtuellen Maschine zeitgeteilt zur alleinigen Nutzung zur Verfügung gestellt werden. Diese Arbeitsweise ist insbesondere für Betriebssystementwickler sehr vorteilhaft. Es ist ohne weiteres einzusehen, dass sowohl die dynamische Adressumsetzung als auch der Betrieb virtueller Maschinen einen erheblichen Steuerungsaufwand erfordern. Der zusätzliche Zeitaufwand für die Steuerung war mit Hardwaremitteln, insbesondere durch mikroprogrammierte Lösungen zu verringern. Für die Mikroprogrammierer ergab sich hier ein weites Feld zur kreativen Betätigung.

2.3.1 Mikroprogrammierte Unterstützung des SVM/ES

Der zugehörige Firmwarekomplex, d. h. die mit Mikroprogrammen realisierten zusätzlichen Funktionen, wird allgemein als die SVM/ES-assists bezeichnet.

Da alle Gastbetriebssysteme unter SVM/ES im realen Problemzustand laufen müssen, ergibt sich ein großer Zusatzaufwand z. B. für die Behandlung der Privilegierungsausnahmebedingungen, die bei der legitimen Ausführung von privilegierten Befehlen in den Gastsystemen auftreten. Hier liegt ein Schwerpunkt der SVM/ES-assists.

Ohne auf Details einzugehen, werden nachfolgend deren wesentliche Funktionen skizziert. Dabei werden die Basis-assists (SVMA), die erweiterten assists (ESVMA), die Steuerprogramm-assists (CPA) sowie einige neue Funktionen unterschieden. In den DV-Anlagen des ESER Reihe 2 wurden bei Robotron große Teile dieser Unterstützungen realisiert. Die vorhandenen Entwicklerkapazitäten begrenzten deren Umfang.

- *Die Modifikation bestehender Befehle.* Hierzu gehört der Großteil der privilegierten Befehle, die durch SVMA bzw. durch ESVMA behandelt werden können. In vielen Fällen erlauben die assist-Funktionen von SVMA diese Befehle für das Gastsystem direkt, ohne Aktivität des SVM/ES, auszuführen. In Sonderfällen, wo das nicht möglich ist, wird die dann unvermeidbare Unterbrechungsbehandlung durch das SVM/ES mit den Funktionen von ESVMA vorbereitet. Ein wesentlicher Teil von ESVMA umfasst dabei die mikroprogrammierte Rettung der Universal- und Gleitkommaregister, der Timer sowie weitere Vorbereitungen. ESVMA übergibt anschließend die Steuerung an eine Adresse, die von SVM/ES für diesen Fall bereitgestellt wurde.

- *Die Modifikation der Unterbrechungsbehandlung.* Hierher gehört z. B. der von SVMA unterstützte SVC-Befehl eines Gastsystems, der ohne Eingriff des SVM/ES zu einer SVC-Unterbrechung im Gastsystem führt. Ein weiterer Ablauf dieser Kategorie wird mit page-invalid-interrupt-routine (PIIR) bezeichnet. Bei der Arbeit mit DAT im Gastsystem wird eine doppelte Adressumsetzung erforderlich, deren Ergebnis in sogenannten Schattentabellen niedergelegt wird. Diese Aufgabe kann ebenfalls durch die mikroprogrammierte PIIR erledigt werden. Nach erfolgreicher PIIR kann der auslösende Befehl wiederholt werden, ohne dass SVM/ES eingreifen muss.
- *Die Funktionen der Steuerprogramm-assists (CPA).* Für alle die Fälle, in denen eine Aktivität des SVM/ES unerlässlich ist, wurde versucht, durch neue Funktionen die Arbeit dieses Steuerprogramms zu beschleunigen. Dazu wurden neue Befehle kreiert, die in sich die Funktionen kleiner häufig durchlaufener Befehlsfolgen vereinen. Repräsentiert werden diese neuen SS-Format-Befehle durch den Befehls-code X'E6' (das 2. Befehlsbyte enthält einen Subcode X'00' ... X'13'). In das Steuerprogramm sind diese Befehle gemäß Abbildung 3 eingebettet. Die beiden Operandenadressen kennzeichnen jeweils den Beginn einer Datenliste bzw. einer Exitadressenliste. In diesen Listen sind Parameter enthalten, die bei der Ausführung des Befehls benutzt werden. Auf den E6-Befehl folgt im Programmtext die Steuerprogrammroutine, deren Funktion der E6-Befehl realisiert. Während der Initialisierungsphase des Betriebssystems wird geprüft, ob CPA in dem jeweiligen Prozessor vorhanden ist. Fehlt CPA, werden vom Betriebssystem alle E6-Befehle mit NOP-Befehlen überschrieben. Damit ist gesichert, dass das System auch auf Prozessoren ausgeführt werden kann, die nicht über CPA verfügen.
- *Weitere Funktionserweiterungen auf der Basis von Mikroprogrammen.* Als Beispiel weiterer systemeffektivierender Funktionen soll hier noch die Unterstützung virtueller Intervallzeitgeber (VITA) erwähnt werden. Mit dieser Funktion kann Programmen in einer virtuellen Maschine ein genauerer Intervallzeitgeberwert zur Verfügung gestellt werden. Die Aktualisierung des virtuellen Intervallzeitgeberwertes erfolgt durch VITA immer dann, wenn während der Arbeit der zugehörigen virtuellen Maschine der reale Intervallzeitgeberwert aktualisiert wird.

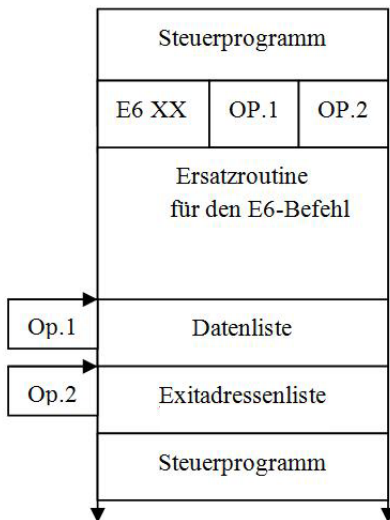


Abbildung 3: Struktur der E6-Befehlseinbindung in das Steuerprogramm SVM/ES

2.3.2 Mikroprogrammierte Unterstützung einer höheren Programmiersprache

Besonders die Forderung nach Übertragbarkeit von Software auf verschiedene Rechnerarchitekturen ließ in der Vergangenheit eine Tendenz bei den Compilern entstehen, die darin bestand, Quellprogramme nicht in eine der üblichen Maschinensprachen sondern in eine von der realen Hardware unabhängige Zwischensprache zu übersetzen. Solche Zwischensprachen besitzen in der Regel ein höheres Niveau als reale Maschinensprachen und sind den funktionellen Erfordernissen der Quellsprachen angepasst. Man kann die Zwischensprachen auch als Maschinensprachen hypothetischer Rechner betrachten. Häufig wird dabei auf die Architektur eines hypothetischen Stackcomputers orientiert. Programme, die in einer Zwischensprache vorliegen, sind auf den üblichen Rechnerarchitekturen nicht direkt ausführbar, sondern müssen vorher entweder in die Maschinensprache eines realen Rechners übersetzt werden oder mit Hilfe eines Interpretationsprogramms abgearbeitet werden. Die Idee für eine mikroprogrammierte Unterstützung betraf die Effektivierung dieser interpretierenden Arbeitsweise.

Als Grundlage für die experimentellen Untersuchungen wurde ein vorhandenes, arbeitsfähiges PASCAL-P-System verwendet. Das PASCAL-P-Laufzeitsystem beinhaltet ein Interpretationsprogramm, für welches neue Befehle zu seiner beschleunigten Abarbeitung geschrieben wurden. Freie Kapazitäten

des ladbaren Mikroprogrammspeichers der Zentraleinheit EC2655.M konnten für diese neuen Befehle genutzt werden. Für die Befehle wurde der Befehlscode X'E4' mit einem Subcode im zweiten Befehlsbyte vereinbart. In der ersten Phase wurden zwei SS-Format-Befehle entwickelt:

INTERPR - (X'E400') – Funktion der zentralen Steuerschleife
LDC - (X'E401') – P-Code-Befehl: „Lade Konstante in Stack“

Diese Befehle entsprechen den dargestellten kleinen Befehlsfolgen:

LDC LA R4,4(,R4) R4 – stack-pointer
CLR R4,R3 R3 – heap-pointer
BNL ERMES1 R8 – instruction-pointer
LH R0,2(,R8)
ST R0,0(,R4)
R INTERPR

ist äquivalent zu

LDC DC X'E401', S(2,(R8),4(R4))
DC X'E400', S(4(R8),TABLE)
B ERMES1

wobei gilt

INTERPR IC RA,4(,R8)
LA R8,4(,R8)
L R1,TABLE(RA)
BR R1

welches äquivalent ist zu

INTERPR DC X'E400', S(4(R8),TABLE)

Nach Einführung der Befehle E400 und E401 sowie von zwei weiteren P-Code-Befehlen in das Interpretationsprogramm ergab sich ein Laufzeitgewinn von ca. 23 %. Obwohl die beschriebenen Mikroprogramme nur für experimentelle Zwecke geschaffen wurden, zeigte sich doch, dass der eingeschlagene Weg durchaus Erfolg versprechend war.

3 Einige Bemerkungen zur Zusammenarbeit von Hard- und Softwareentwicklern im Hause Robotron KMST

Es ist ein großer Vorteil, bei der Entwicklung einer DV-Anlage auf bereits fertig formulierte Funktionsprinzipien zurückgreifen zu können. Sowohl die Funktions- als auch die Konsistenz- und Kompatibilitätsprüfungen dieser Prinzipien, ganz zu schweigen von ihrer Herstellung, können eingespart werden. Man kann auf dieser Basis Hardware- und Softwareentwickler weitgehend

unabhängig voneinander entwickeln lassen. Wenn fehlerfrei gearbeitet wurde, sollten die jeweiligen Entwicklungsergebnisse zueinander passen. Die entwickelte Software sollte auf der entwickelten Hardware operationsprinzipiengemäß ausgeführt werden können. Der reale Entwicklungsprozess solch hochgradig komplexer Gebilde wie der Hard- bzw. Software einer DV-Anlage bedarf jedoch zweckmäßigerweise einer Reihe zusätzlicher Verfahren und Hilfsmethoden. Die Simulation der entwickelten Hardware mittels eines Simulationsprogramms wurde bereits erwähnt. Weitere Hilfsprozesse resultieren aus den Aufgabenbereichen der Chipplatzierung auf den Steckeinheiten und der Trassierung dieser Mehrlagenleiterplatten sowie der Rückverdrahtung. Damit ergibt sich zwangsläufig eine gewisse Zusammenarbeit der Hardwareentwickler mit den Entwicklern dieser entwurfsunterstützenden, spezifischen Software. Für die Beziehungen der Hardwareleute zu den Betriebssystem- bzw. Compilerentwicklern ergeben sich solche Zwänge bei festliegenden Operationsprinzipien nicht unbedingt. Wie bereits erwähnt, könnte sich aber z. B. das Wissen über die Existenz und Funktionsweise einer BVE bei den Softwareleuten durchaus positiv auswirken (siehe 2.1). Lokal angepasste Programmteile des Betriebssystems aber auch von Compilern samt deren Übersetzungsergebnisse könnten Laufzeitverbesserungen für die Programme ergeben.

Ein Zwang zur Zusammenarbeit der entsprechenden Entwicklergruppen besteht in jedem Fall bei der Realisierung von Mikroprogrammen zur Unterstützung des Betriebssystems oder anderer Anwendungssoftware (siehe 2.3). Die Spezifikation solcher Funktionen, die ja eine Erweiterung der Operationsprinzipien darstellen, sollte unbedingt als Gemeinschaftsprojekt verstanden werden. Sowohl die Wirkung als auch die Verwendungsprinzipien der Zusatzfunktionen bzw. neuen Befehle erfordern eine enge Kommunikation zwischen Hard- und Softwareentwicklern.

Im Hause Robotron war der Informationsaustausch bzw. die Zusammenarbeit zwischen den betreffenden Arbeitsgruppen im dargestellten Sinne weitgehend der Initiative einzelner Mitarbeiter überlassen. Dadurch wurde sicher so mancher Synergieeffekt verschenkt.

4 Literatur und Internetquellen

- [1] AUTORENKOLLEKTIV (1979): Funktionsprinzipien des ESER, Reihe 2. *Schriftenreihe Informationsverarbeitung* .
- [2] MERKEL, G. (2006): Computerentwicklungen in der DDR – Rahmenbedingungen und Ergebnisse. In: Naumann, F., Schade, G. (Hrsg.), *Informatik in der DDR – eine Bilanz, Schriften zu den Symposien 7. bis 9. Oktober 2004 in Chemnitz und 11. bis 12. Mai 2006 in Erfurt*, S. 40-54.

-
- [3] LAMPENSCHERF, W.; LINZMANN, D. & OTTO, H. (1981): Die Zentraleinheit EC2655M. *rechentechnik/datenverarbeitung 18 (2)*.
 - [4] GEIBLER, R.; RIESEN, M.; SEIFERT, F.; DR. GRUNER, R. & WENDLIK, K. (1981): Matrixmodul. *rechentechnik/datenverarbeitung 18 (2)*.
 - [5] MARSCHNER, R. (1984): Bedien- und Serviceprozessor EC7069 M. *rechentechnik/datenverarbeitung 21 (8)*.
 - [6] JUNGNICHEL, G. viele weitere Informationen und links findet man unter: <http://www.eser-ddr.de/>.

Die zitierten Internetquellen wurden zuletzt am 25.07.2010 aufgerufen.