PDV-E 132
September 1979

PDV-Entwicklungsnotizen

Programmiersprachen für Mikrorechner zur Steuerung nachrichtentechnischer Anlagen

S. Nestel, K. Schaper
SEL-Forschungszentrum, Stuttgart

Kernforschungszentrum Karlsruhe

PDV-Berichte

Die Kernforschungszentrum Karlsruhe GmbH koordiniert und betreut im Auftrag des Bundesministers für Forschung und Technologie das im Rahmen der Datenverarbeitungsprogramme der Bundesregierung geförderte Projekt Prozeßlenkung mit Datenverarbeitungsanlagen (PDV). Hierbei arbeitet sie eng mit Unternehmen der gewerblichen Wirtschaft und Einrichtungen der öffentlichen Hand zusammen. Als Projektträger gibt sie die Schriftenreihe PDV-Berichte heraus. Darin werden Entwicklungsunterlagen zur Verfügung gestellt, die einer raschen und breiteren Anwendung der Datenverarbeitung in der Prozeßlenkung dienen sollen.

Der vorliegende Bericht dokumentiert Kenntnisse und Ergebnisse, die im Projekt PDV gewonnen wurden.

Druck und Verbreitung:

Kernforschungszentrum Karlsruhe GmbH
Postfach 3640 7500 Karlsruhe 1

Bundesrepublik Deutschland

PDV

PDV-E 132

PROJEKT PROZESSLENKUNG MIT DV-ANLAGEN
ENTWICKLUNGSNOTIZ PDV-E 132

PROGRAMMIERSPRACHEN FÜR MIKRORECHNER ZUR STEUERUNG NACHRICHTENTECHNISCHER ANLAGEN

VON

S. NESTEL, K. SCHAPER

SEL-FORSCHUNGSZENTRUM, STUTTGART

41 SEITEN
TABELLEN
25 LITERATURSTELLEN

DEZEMBER 1978

ZUSAMMENFASSUNG

Der Bericht zeigt aus der Sicht eines Herstellers von nachrichtentechnischen Anlagen, welche Programmiersprachen beim Einsatz von Mikrorechnern unter gegebenen Voraussetzungen vorteilhaft sind.

Für kurzfristig beginnende Projekte und in Zusammenarbeit mit anderen Firmen wird als Übergangslösung ASSEMBLER 8085 und PL/M-80 empfohlen. Für Firmeninterne Projekte der Vermittlungstechnik ist bei SEL ESPL-1 als Standard eingeführt.

Auf lange Sicht gesehen können jedoch die europäischen Postverwaltungen die Anwendung der Programmiersprache CHILL für vermittlungstechnische Aufgaben fordern. Für nationale Projekte außerhalb der Vermittlungstechnik wird PEARL zunehmend an Bedeutung gewinnen. CHILL und PEARL sind beide für Steuerungsaufgaben sehr gut geeignet, besitzen jedoch unterschiedliche Schwerpunkte, nämlich die Verarbeitung komplexer Datenmengen bei CHILL und umfangreiche Realzeitbetriebssystemfunktionen bei PEARL. Wenn die Betriebssystemfunktionen von PEARL genutzt werden sollen, sind Übersetzer und Zielrechner an die Betriebssystemfunktionen anzupassen.

Zur Vorbereitung des Einsatzes von CHILL und von PEARL für Mikrorechner sollten frühzeitig die erforderlichen Dienstleistungsprogramme wie Übersetzer, Binder und Lader bereitgestellt werden.

PASCAL ist eine klar gegliederte und leicht erlernbare Hochsprache und ist insbesondere für arithmetische Probleme und zur Verarbeitung komplexer Datenmengen geeignet. PASCAL besitzt jedoch keine speziellen Eigenschaften für die Programmierung von Steuerungsaufgaben wie etwa CHILL oder PEARL.

Inhaltsverzeichnis

1. Die Aufgabenstellung und die wichtigsten Ergebnisse im Überblick

Die Auswahl einer Programmiersprache erfordert bei der Planung neuer Software-Entwicklungen sorgfältige Überlegungen, da die Eignung der verwendeten Sprache und die Effizienz der zugehörigen Dientsleistungs-Software (z.B. Übersetzer) die Produktivität der Software-Entwicklung wesentlich beeinflussen.

Höhere Programmiersprachen erlauben eine vielfach höhere Programmierleistung als Assemblersprachen, da mit einem Hochsprachenbefehl eine größere Anzahl Assembler-Befehle abgedeckt wird. Bei der hier ins Auge gefaßten Anwendung von Mikrorechnern sieht die Situation jedoch viel ungünstiger aus als bei eingeführten größeren Rechnern.

Zur Anwendung von fortschrittlichen Hochsprachen für Mikrorechner sind entweder noch keine Übersetzer vorhanden, die vorhandenen Übersetzer sind ineffizient oder die laufende Betreuung durch den Hersteller läßt zu wünschen übrig. Nach den ersten Versuchen, laufzeitkritische Steuerungsaufgaben in höheren Programmiersprachen auf Mikrorechnern zu implementieren, mußten viele Programmteile nachträglich im Assembler neu programmiert werden, da die Übersetzer unzureichend waren.

Der vorliegende Bericht hat zum Ziel, den heutigen Stand der Möglichkeiten und Grenzen bei der Auswahl von Programmiersprachen für nachrichtentechnische Steuerungsaufgaben zu zeigen und eine Entscheidungshilfe für die Sprachauswahl zu bieten.

Bei der Auswahl einer Hochsprache und einer maschinennahen Sprache für die Programmierung von Mikrorechnern wird in dieser Arbeit folgendes vorausgesetzt:

- Neues Projekt
- Zielrechner: Intel 8085
- Dienstrechner: Mikrorechner-Entwicklungssysteme und Rechner wie IBM 370, Siemens 7700 und PDP 11
- Anwendungsgebiet: Steuerung nachrichtentechnischer Anlagen

Steuerungsaufgaben, wie sie z.B. in Vermittlungs- oder Informationsabrufsystemen auftreten, verlangen von einer Programmiersprache vor allem Möglichkeiten zur Echtzeitverarbeitung von Schaltfunktionen (Boolesche Funktionen), Operationen mit einzelnen Bits oder mit Bitgruppen eines Rechnerwortes, Zeichenverarbeitung, Listen und Tabellenverarbeitung, sowie umfangreiche Möglichkeiten zum Datentransfer zwischen Rechner und Peripherie. Daneben sind Realzeit-Betriebssystemfunktionen erforderlich (z.B. zur Unterbrechungssteuerung).

Die Verarbeitung kontinuierlicher Signale wird hier nicht berücksichtigt. (z.B. zur Realisierung von Analog-Digital-Umwandlern, digitalen Filtern,

Entzerrern, Signaltransformationen, Modulation, Analogreglern oder zur Redundanzreduktion). Solche Signalverarbeitungsaufgaben stellen häufig andere Anforderungen an die Programmiersprachen und an die Mikrorechnerklasse als digitale Steuerungsaufgaben.

Folgende neun Sprachen wurden der Untersuchung zugrunde gelegt: ASSEMBLER 8085, PL/M-80, MISTRAL, ESPL-1, CPL-1, FORTRAN-80, PASCAL, PEARL und CHILL.

Aus der Sicht eines Herstellers von nachrichtentechnischen Anlagen sind als wichtigtes Ergebnis in Tabelle I die in die engere Wahl fallenden Sprachen zusammengestellt. Die ausführliche Begründung und Hintergrundmaterial befindet sich in den folgenden Abschnitten.

Bei bestimmten Anwendungen, wie in der Bahnsteuerungstechnik, wird ein Sicherheitsnachweis der Steuerungseinrichtungen einschließlich der Software gefordert. Dieser Gesichtspunkt wurde bei den vorliegenden Untersuchungen nicht berücksichtigt und kann zu anderen Entscheidungen führen.

Grundlage der Sprachuntersuchungen waren die zur Zeit verfügbaren Sprachbeschreibungen.

Tabelle 1: Empfehlung geeigneter Programmiersprachen am Beispiel der bei SEL bestehenden Randbedingungen

	Voraussetzungen		Bevorzugte Sprachen	Wichtige Gründe (Auszug aus Abschnitt 3)
	Terminsituation	Projektbeteiligung		
Fall I a	Fertigstellungstermin: kurzfristig	SEL/ITT alleine	ASSEMBLER 8085 ESPL-1	Verfügbare Dienstleistungsprogramme, ESPL-1 ist ITT-Standardsprache in der Vermittlungstechnik
Fall I b	Entwicklungsbeginn: sofort	zusammen mit anderen Firmen	ASSEMBLER 8085, PL/M-80 (als Übergangslösung	Verfügbare Dienstleistungsprogramme, Verbreitung von PL/M-80 auch bei anderen Firmen PL/M-80 gut durch Entwicklungssystem gestützt.
Fall II a	Fertigstellungstermin: langfristig Bei Entwicklungsbeginn müssen Übersetzer, Binder, Lader und Objektcode-Simulator verfügbar sein.	SEL/ITT alleine	MISTRAL, CHILL	Unterstützung der Strukturierten Programmierung durch MISTRAL. CHILL speziell für Vermittlungs- und Informationsabrufsysteme
Fall II b	Fertigstellungstermin: langfristig Bei Entwicklungsbeginn müssen Übersetzer, Binder, Lader und Objektcode-Simulator verfügbar sein.	zusammen mit anderen Firmen	ASSEMBLER 8085, CHILL oder PEARL	Sprachen von Fremdfirmen wahrscheinlich akzeptierbar. CHILL speziell für Vermittlungs- und Informationsabrufsysteme. PEARL für Steuerungsaufgaben, bei denen die programmiersprachliche Realisierung von Realzeit- Betriebssystemfunktionen im Vordergrund stehen.

2. Charakterisierung und Beurteilung der untersuchten Sprachen

2.1 Übersicht

Der Untersuchung wurden folgende Sprachen zugrunde gelegt:
ASSEMBLER 8085, PL/M-80, MISTRAL, ESPL-1, CPL-1, FORTRAN-80, PASCAL, PEARL und CHILL.

Für den nur mit ASSEMBLER-Sprachen vertrauten Leser werden im folgenden mit einer kurzgefaßten Zusammenstellung die wichtigsten Merkmale der höheren Sprachen dargestellt, welche insbesondere für Steuerungsaufgaben in der Nachrichtentechnik wichtig sind. Daraus lassen sich die Schwerpunkte der einzelnen Sprachen erkennen.

In [22] wurden neben PASCAL einige andere neue Sprachen (CONCURRENT PASCAL, MODULA, DoD1 u.a.) untersucht.

2.2 ASSEMLBER 8085

ASSEMBLER 8085 ist die maschinennahe Sprache für den Zielrechner INTEL 8085 [1] . Die Eigenschaften dieser Sprache werden hier als bekannt vorausgesetzt.

2.3 PL/M-80

PL/M-80 ist eine von INTEL angebotene Hochsprache [4]. Sie ist eine Untermenge von PL/1.

Charakteristische Eigenschaften von PL/M-80

(1) Programmgliederung
- Getrennt übersetzbare Unterprogramme (PROCEDURES)
- Schachtelbare Unterprogramme
- Rekursive Unterprogramme
- Unterprogramme mit dem Attribut REENTRANT
- Programmblöcke (DO...END)
- Verzweigung (IF...THEN...ELSE, DO...CASE...OF)
- Schleifen (DO...WHILE, DO...TO...BY)

(2) Gliederung umfangreicher Datenmengen
- Verbund von Daten verschiedenen Typs (Mischstruktur STRUCTURE)
- Verbund von Daten gleichen Typs (ARRAY, eindimensional)

(3) Bedarfsweises Reservieren von Speicherplatz bei Programmausführung durch
- Bereitstellung und Belegung von Speicherplatz über Zeiger. Diese Zeiger werden implizit vereinbart.

(4) Adressierungsarten
- Zugriff über symbolische direkte und indirekte Adressierung
- Ermittlung des absoluten Wertes einer symbolischen Adresse während der Programmausführung durch speziellen Operator DOT

(5) Operationen:
- Boolesche, arithmetische und vergleichende Operationen
- Mischbarkeit arithmetischer und logischer Operationen
- Operationen für die Verschiebung oder Rotation der Bitkette eines Wortes

(6) Programmieren eines Realzeitsystems
- Taktzähler
- Kennzeichnung von Unterprogrammen, die nach einer prioritätsgesteuerten Unterbrechung abgearbeitet werden.
- Freigabe oder Sperren von Unterbrechungen

(7) Steuerung des Datentransfers
- Ein-Ausgabe über adressierbare Zwischenspeicher (Tore)

PL/M-80 unterstützt die Strukturierte Programmierung, die Gliederung größerer Datenmengen und die Programmierung eines Realzeitsystems. Die Mischbarkeit von Booleschen und arithmetischen Operationen erleichtert es, einzelne Bits oder Bitgruppen aus einem 8-bit bzw. 16-bit-Wort zu maskieren und weiter zu verarbeiten. Die getrennt übersetzbaren Unterprogramme erlauben das Aufteilen umfangreicher Software in überschaubare Programme, die zunächst einzeln und von einander getrennt entwickelt werden können.

2.4 MISTRAL

MISTRAL (Microcomputer Structured Assembler) berücksichtigt den Befehlsumfang des Zielrechners INTEL 8085 und wurde von ITT entwickelt [2,3].

Charakteristische Eigenschaften von MISTRAL

(1) Programmgliederung
- Aufteilung des Programms (PROGRAM) in einen Vereinbarungsteil für Unterprogramme (PROCEDURE) und in einen Hauptblock (MAIN). Im Hauptblock werden zuerst Variable, Konstanten, Marken und Datenstrukturen vereinbart, dann folgen die auszuführenden Befehle,
- Programmblöcke (BEGIN...END)
- Verzweigung (IF...THEN...ELSE...)
- Schleifen (WHILE...DO...,REPEAT...UNTIL..., LOOP BEGIN...EXIT IF... END)

(2) Vereinbarung von Datentypen
- Zwei Wortlängen: Ganzwort (BYTE) oder Doppelwort (DOUBLET) (8 bit-Wort oder 16 bit-Wort)
- Variable oder Konstante
- Datenstruktur: Liste (TABLE) mit einer festgelegten Anzahl von Daten

(3) Adressierungsarten:
- Direkte symbolische Adressierung
- Einfache und zweifache indirekte Adressierung
- Absolute Adressierung
- Direkte Adressierung der Ein- und Ausgabetore

(4) Operationen
- Arithmetische, Boolesche und vergleichende Operationen
- Zahlreiche Schiebeoperationen

(5) Möglichkeiten zur Erhöhung der Laufzeiteffizienz der Programme
- Einbau der maschinennahen Befehle des ASSEMBLERS 8085 in den Quellprogrammtext

Übersetzer für den Zielrechner INTEL 8085 sind von ITT für die IBM 370/158 als Dienstrechner und für den Rechner HP 1000 entwickelt worden.

MISTRAL ist eine maschinennahe Sprache mit interessanten hochsprachlichen Ergänzungen. MISTRAL kann an Stelle von ASSEMBLER 8085 bei der Programmierung von Steuerungsprozessen eingesetzt werden, ohne daß der Ineffizienzfaktor der Objektcodeerzeugung wesentlich zunimmt.

2.5 ESPL-1

ESPL-1 ist eine von PL/1 abgeleitete Hochsprache, die von ITT speziell für die Vermittlungstechnik geschaffen wurde [5,6] . ESPL-1 ist bei SEL/ITT Standardsprache für Vermittlungstechnische Projekte.

Charakteristische Eigenschaften von ESPL-1

(1) Programmgliederung
- Block gemeinsam benützter Daten (EXTERNAL, EXTERNAL SUBSYSTEM)
- Getrennt übersetzbare Module (SUBSYSTEM)
- Unterprogramme (PROCEDURE, FUNCTION)
- Programmblöcke (DO...END)
- Verzweigungen (IF...THEN...ELSE...)
- Schleifen (DO WHILE..., DO...TO...BY)

(2) Unterprogrammtechnik
- Übergabe der Adresse der Parameter bei Unterprogrammaufruf (call by reference)
- Übergabe der symbolischen Adresse eines parameterlosen Unterprogramms als Parameter

(3) Vereinbarung von Datentypen
- Ganze Zahlen, wahlweise mit ganzer (32 bit) oder halber Wortlänge (16 bit), Bit/Bitgruppe (2 bis 31 bit pro Gruppe), Zeiger, Marken, Text (Feld mit Elementen von 8 bit-Wörtern)

(4) Gliederung umfangreicher Datenmengen
- Felder (ARRAY), ein- und zweidimensional
- Anordnung mehrerer Bitgruppen in einem Wort (32 bit Wortlänge, die einzelnen Bitgruppen dürfen sich überschneiden)
- Mischstruktur (STRUCTURE)
- Anordnung gleicher Strukturen (BLOCK ARRAY oder TABLE ARRAY)
- Angabe der Wiederholung gleicher Strukturen durch das Attribut DEFINED

(5) Bedarfweises Reservieren von Speicherplatz bei Programmausführung
- Attribut BASED POINTER für Daten

(6) Adressierungsarten
- Direkte Adressierung, absolut (LOCATION) oder symbolisch
- Indirekte Adressierung über Zeiger

(7) Programmieren eines Realzeitsystems
- Befehle für die Programmierung eines Realzeitsystems sind nicht in der Sprache implementiert. Der Übersetzer stellt Routinen zur Verfügung, die mit Namen und Angabe der Parameter aufgerufen werden können und in der Sprache als INTRINSIC FUNCTIONS bezeichnet werden. Diese Routinen erlauben z.B. den Start eines Prozesses.

(8) Operationen
- Boolesche und vergleichende Operationen sind mit einzelnen Bits oder Bitgruppen möglich.
- Boolesche, arithmetische und vergleichende Operationen mit Zahlen

(9) Möglichkeiten zur Erhöhung der Laufzeiteffizienz der Programme
- ESPL-1 kann im Programmtext mit der maschinennahen Sprache des Zielrechners ITT 3200 gemischt werden. Entsprechendes ist für die ESPL-1 - Version für den Zielrechner INTEL 8085 vorgesehen.

Für ESPL-1 gibt es jetzt einen neueren Übersetzer, der für den Dienstrechner IBM 370 und den Mikrorechner INTEL 8080 als Zielrechner entwickelt wurde. Es ist ein Dienstleistungsprogramm vorgesehen mit dem Ziel, den Objektcode in das Entwicklungssystem MDS 800 laden zu können.

2.6 CPL-1

CPL-1 ist eine Hochsprache auf der Grundlage von PL/1, die von der französischen Software-Firma CAP SOGETI LOGICIEL angeboten und betreut wird [7] .

Charakteristische Eigenschaften von CPL-1

(1) Programmgliederung
- Schachtelbare Unterprogramme (PROCEDURE, FUNKTION)
- Programmblöcke (DO...END)
- Schleifen (DO WHILE, DO...TO...BY...)

(2) Vereinbarung von Datentypen
- Variable, Konstante
- Arithmetische Operanden, Zeichen und Bitkette, Marken, Zeiger

(3) Gliederung umfangreicher Datenmengen
- Felder (Daten gleichen Typs), Mischstrukturen (STRUCTURE)

(4) Bedarfsweises Reservieren von Speicherplatz bei Programmausführung durch
- Attribut BASED POINTER für Daten

(5) Operationen
- Boolesche, arithmetische und vergleichende Operationen auch mit einzelnen Bits oder Bitgruppen

Für CPL-1 sind Übersetzer auf mehreren Dienstrechnern (IBM 370, Siemens 7740, PDP 11) für den Zielrechner INTEL 8085 verfügbar.

CPL-1 wird wegen verfügbarer Dienstleistungsprogramme für die Mikrorechnerprogrammierung relativ verbreitet eingesetzt.

2.7 FORTRAN-80

FORTRAN-80 ist eine von INTEL für den Mikrorechner 8085 ausgewählte Untermenge von ANSI FORTRAN 77 [8] . Die Sprache ist für Anwendungen mit vorwiegend arithmetischen Operationen vorgesehen. Übersetzer sind für das Entwicklungssystem MDS 800 verfügbar.

Charakteristische Eigenschaften von FORTRAN-80

(1) Programmgliederung
- Getrennt übersetzbare Unterprogramme
- Speicherbereich mit Daten, die gemeinsam von mehreren Unterprogrammen benützt werden können (COMMON)
- Verzweigung (IF...THEN...ELSE...ENDIF)
- Schleifen (DO...CONTINUE)

(2) Unterprogrammtechnik
- Übergabe der Adresse der Parameter bei Unterprogrammaufruf (call by reference)

(3) Vereinbarung von Datentypen
- Komma-Zahl, ganze Zahl, Zeichen und Zeichenkette mit festgelegter Anzahl von Zeichen

(4) Gliederung umfangreicher Datenmengen
- Ein- und mehrdimensionale Felder mit berechenbaren Feldgrenzen

(5) Adressierungsarten
- Direkte symbolische Adressierung

(6) Steuerung des Programmablaufs durch Sprünge
- Unbedingter Sprung (GO TO, COMPUTED GOTO, ASSIGNED GOTO)
- Bedingter Sprung

(7) Datenverarbeitung
- Eröffnen/Schließen einer Datei (OPEN/CLOSE)
- Vorrücken des Ein/Ausgabezeigers auf den nächsten Platz in der Datei (BACKSPACE)

(8) Datentransfer
- Lesen/Schreiben mit Anweisung für die Formatierung (READ, WRITE)

In [8] sind die Befehle aufgeführt, die nicht mit ANSI FORTRAN IV kompatibel sind (Insgesamt 11 Befehle). Zum Beispiel sind in FORTRAN-80 folgende Befehle und Leistungsmerkmale enthalten, die in ANSI FORTRAN IV (1977) nicht verfügbar sind:
- Binär (oktal oder hexadezimal) dargestellte Zahlen
- Boolesche Operationen für Bitketten.

Das Mischen von Booleschen und arithmetischen Operationen ist bei FORTRAN-80 nicht erlaubt (Bei der DEC-Version des erweiterten FORTRAN IV möglich).

2.8 PASCAL

PASCAL ist eine von N. Wirth geschaffene, universell verwendbare Hochsprache auf mathematischer Grundlage. PASCAL wurde mit dem Ziel entwickelt, eine Nachfolge für ALGOL-60 zu schaffen, jedoch mit beschränkter Komplexität und besseren Bedingungen zur Erstellung wirksamer Übersetzer [9, 10, 11]

Charakteristische Eigenschaften von PASCAL

(1) Programmgliederung
- Aufteilen des gesamten Programms und der Unterprogramme (PROCEDURE, FUNCTION) in je zwei Teile. Der erste Teil enthält nur Vereinbarungen, der zweite Teil enthält nur Befehle (Aktionen). Diese Befehle sind in einem BEGIN...END Block zusammengefaßt (compound statement).
- Schachtelbare, wahlweise rekursive Unterprogramme
- Verzweigung (IF...THEN...ELSE..., CASE...OF)
- Schleifen (WHILE...DO, REPEAT...UNTIL, FOR...TO/DOWN TO...DO)

(2) Übergang des strukturierten Programmablaufs von der Entwurfphase in die Codierphase ohne Umstellung möglich

(3) Vereinbarung von Datentypen
- Variable, Konstante, Marken
- Festlegen eines neuen Datentyps durch den Programmierer

(4) Gliederung umfangreicher Datenmengen durch
- Feld (ARRAY STRUCTURE)
- Mischstruktur (RECORD STRUCTURE)
- Zusammenfassung und Benennung von Daten (SET STRUCTURE)
- Liste (FILE STRUCTURE)

(5) Bedarfweises Reservieren von Speicherplatz bei Programmausführung
- Routine NEW, die der Übersetzer bereitstellt. Die dynamisch bereitgestellte Speicherzelle wird über einen Zeiger (indirekte Adresse) angesprochen.

(6) Adressierungsarten
- Direkte symbolische Adressierung
- Indirekte Adressierung (Zeiger)

(7) Operationen
- Boolesche, arithmetische und vergleichende Operationen

(8) Verarbeiten von Listen auf Massenspeichern
- Anfügen eines neuen Wertes (PUT)
- Vorrücken auf den folgenden Wert beim Speicherzugriff (GET)
- Zugriff auf den Anfang der Liste (RESET)
- Überschreiben der Liste (REWRITE)

(9) Datentransfer
- Lesen/Schreiben von Text (READ/WRITE)
- Zeilenweises Lesen/Schreiben von Text (READLN/WRITELN)

(10) Mögliche Erhöhung der Effizienz bezüglich Speicherplatz
- Dichte Packung von Daten (PACK)

PASCAL unterstützt die klare und systematische Darstellung der Struktur von Abläufen und Daten [9,10]. Durch die hierarchische Gliederung nach den

Regeln der Strukturierten Programmierung wird die stufenweise fortschreitende Überprüfung der Korrektheit der Programme wesentlich erleichtert und auf eine mathematische Grundlage gestellt [11] . Es existieren mehrere PASCAL-Dialekte [12] . Einige Dialekte (UCSD PASCAL u.a.) erlauben es, umfangreiche Software in getrennt übersetzbare Unterprogramme zu gliedern (UCSD = University of California, San Diego). Auch Anbieter von Mikrorechnern entwickeln eigene PASCAL-Versionen. Die von N. Wirth stammende ursprüngliche Fassung sieht die Vereinbarung getrennt übersetzbarer Unterprogramme nicht vor und ist deshalb für die Programmierung großer Softwarekomplexe wenig geeignet.

Für einen Dialekt von PASCAL ist von TEXAS INSTRUMENTS ein Übersetzer entwickelt worden. Er ist für den Zielrechner TMS 9900 und für den Dienstrechner TI 990/10 bestimmt. Für einen anderen Dialekt von PASCAL wurde von der Firma KONTRON ein Übersetzer für den Mikrorechner Z 80 der Firma ZILOG angekündigt. Die Firma NORTHWEST MICROCOMPUTER SYSTEMS (Eugene, Oregon, USA) bietet für den Mikrorechner INTEL 8085 ein Entwicklungssystem mit einem Übersetzer für UCSD-PASCAL [13] an. Siemens hat für das INTEL 8085 Entwicklungssystem SM 800 einen PASCAL-Übersetzer entwickelt. Dieser Übersetzer ist zur Zeit in Erprobung.

2.9 PEARL

PEARL ist eine von mehreren deutschen Firmen und Instituten entwickelte Sprache. PEARL ist insbesondere zur Steuerung von Fertigungsprozessen entwickelt worden. Die Gesamtsprache FULL PEARL ist in [14] beschrieben.

Charakteristische Eigenschaften von PEARL

(1) Programmgliederung
- Module mit Querbeziehung über globale Daten
- Schachtelbare Unterprogramme, wahlweise mit Attribut REENTRANT
- Programmblock (DO...END)
- Verzweigung (IF...THEN...ELSE..., CASE...ALT...OUT...FIN)
- Schleifen (FOR...FROM...BY...TO, WHILE...REPEAT...END)

(2) Vereinbarung von Datentypen
- Ganze Zahlen, Gleitpunkt-Zahlen, Zeichen- und Bitkette, Zeitdauer, Uhrzeiten, Sprungmarken

(3) Gliederung umfangreicher Datenmengen
- Mischstruktur (RECORD)
- Vereinbarung einer bestimmten Struktur als Typ

(4) Adressierungsarten
- Direkte symbolische Adressierung
- Indirekte Adressierung (Zeiger)

(5) Operationen
- Boolesche, arithmetische und vergleichende Operationen auch mit einzelnen Bits oder Bitgruppen (Einzelbitverarbeitung)
- Vereinbarung neuer Operationen. Eine häufig auftretende Kombination von Operationen kann mit einem Namen versehen werden und in der Art eines Makrobefehls bedarfsweise aufgerufen werden.

(6) Programmieren eines Realzeitsystems
- Definition eines Prozesses, Beschreibung der Zustände eines Prozesses
- Fortsetzen eines blockierten Prozesses (CONTINUE)
- Definition der Prozeßübergänge: blockiert - verzögert - bereit (AFTER...RESUME) und laufend - bekannt (TERMINATE)
- Ausplanung (Löschung eines eingeplanten Prozesses)
- Synchronisation von Prozessen
- Steuerung von Prozessen in Abhängigkeit von Zeitbedingungen und asynchronen Ereignissen (SCHEDULING)
- Semaphor - Operationen
- Steuerung der Betriebsmittelvergabe durch BOLT-Variable

(7) Datentransfer
- Kommunikation zwischen Prozessen (TAKE, SEND)
- Kommunikation zwischen Rechner und Peripherieeinheiten (GET, PUT) mit Anweisung für die Formatierung

Die Programmierung eines Realzeitsystems wird durch PEARL vorzüglich unterstützt. Die Leistungsmerkmale für die Verarbeitung umfangreicher, komplexer Programmabläufe und Datenmengen sind weniger stark ausgeprägt. Eine Untermenge von PEARL ist genormt (BASIS-PEARL, gelber Normdruck). Dies wird die nationale Verbreitung für Prozeßrechneranwendungen fördern. Für einen Mikrorechner ist die Anwendung der betriebssystemnahen Sprachelemente von PEARL nur möglich, wenn ein an PEARL angepaßtes Mikrorechner-Betriebssystem verfügbar ist. Eine Untermenge der Sprache PEARL könnte für Mikrorechner interessant werden. Mit einer kompatiblen Untermenge wäre dann eine einheitliche Programmiersprache für einen Verbund von Mikrorechnern und größeren Prozeßrechnern möglich. PEARL-Übersetzer wurden von den Firmen AEG für den Minirechner 80/20, von Siemens für die Prozeßrechnerserie 310 bis 340, von BBC für PDP 11 und von Krupp für EPR 1300 und EPR 1500 entwickelt. Übersetzer für den Mikrorechner INTEL 8085 sind noch nicht verfügbar (Zur Implementierung für Z 80 siehe [15]).

2.10 CHILL

Das CCITT-Gremium entwickelte auf internationaler Grundlage eine Hochsprache (CHILL) für die Vermittlungstechnik [17, 18, 20] . Bei sechs Postverwaltungen und Firmen, darunter auch ITT (LCT), wurden Untermengen dieser Sprache festgelegt [19] .

Charakteristische Eigenschaften von CHILL

(1) Programmgliederung
- Schachtelbare getrennt übersetzbare Module (MODULE...END) mit Querbeziehung über globale Daten
- Schachtelbare, getrennt übersetzbare Unterprogramme (PROCEDURE)
- Programmblöcke (BEGIN...END)
- Verzweigung (IF...THEN...ELSE...FI, CASE...OF...ELSE...ESAC)
- Schleifen (DO WHILE...OD, DO FOR EVER...OD, DO FOR...BY...TO/DOWN TO... OD)

(2) Unterprogrammtechnik
- Übertragen der Werte der Parameter (call by value)
- Übertragen der Adresse der Parameter (call by location)

(3) Vereinbarung von Datentypen
- Ganze Zahlen (wahlweise dezimale, binäre, oktale oder hexadezimale Darstellung), Boolesche Variable, Zeichen/Zeichenkette, Zeiger, Marken.

(4) Gliederung umfangreicher Datenmengen
- Zusammenfassende Benennung von Daten (SET)
- Festlegen von Untermengen (RANGE) aus zusammengefaßt benannten Daten mit Angabe der Grenzen
- Felder (ARRAY)
- Mischstrukturen (STRUCTURE)

(5) Bedarfweises Reservieren von Speicherplatz bei Prozessausführung
- Variable Feldgrenzen, variable Strukturen über spezielle Zeiger (ROW)

(6) Adressierungsarten
- Direkte symbolische oder absolute Adressierung (LOCATE)
- Indirekte Adressierung (REFERENCE MODE, ROW MODE)

(7) Operationen
- Boolesche, arithmetische und vergleichende Operationen auch mit einzelnen Bits oder Bitgruppen
- Verbinden von Zeichenketten (CONCATENATION)

(8) Programmieren eines Realzeitsystems
- Definition eines Prozesses (PROCESS)
- Definition eines Moduls, das bei Programmausführung nicht unterbrochen werden kann (CRITICAL REGION)

(9) Mögliche Erhöhung der Effizienz bezüglich Speicherplatz
- Dichte Packung von Daten (PACK)

CHILL bietet umfassende Befehle für folgende Aufgaben:
- Gliedern umfangreicher Programme und Datenmengen
- Verarbeiten komplexer Datenstrukturen, ganzer Zahlen (besonders als Binär-Zahl) und einzelne Zeichen oder Zeichenketten.

Insgesamt stellt CHILL von den hier untersuchten Sprachen die größte Anzahl von Befehlen zur Verfügung. Die gute Eignung von CHILL wird durch eine überaus große Anzahl von Befehlen erkauft. Aus folgenden Gründen wird die Auswahl einer Untermenge erforderlich:
- Kurze Laufzeit und geringer Speicherplatzbedarf der Übersetzer (besonders bei Mikrorechner-Entwicklungssystem zu fordern)
- Problemloserer Nachweis korrekt geschriebener Programme
- Leichte Erlernbarkeit der Sprache

Bei einer Auswahl einer Untermenge sollte deren Eignung zur Steuerung nachrichtentechnischer Anlagen berücksichtigt werden. Mit einer kompatiblen Untermenge könnten Programme für Mikrorechner in derselben Sprache geschrieben werden wie die Programme für Großrechner, die zusammen mit den Mikrorechnern in einem Verbund arbeiten. Mit der Programmierung in der

An Übersetzern für den Mikrorechner INTEL 8085 als Zielrechner unter Verwendung größerer Rechner als Dienstrechner wird zur Zeit gearbeitet [24,25].

2.11 Zusammenfassende Beurteilung der Sprachen

Um eine Auswahl treffen zu können, wurden die Sprachen in den Tabellen 2, 3 und 4 gegenübergestellt.

Anmerkungen zur Tabelle 2:

- Die Beurteilung der Sprachen nach ihrer Eignung für Steuerungsaufgaben ist das Ergebnis der Untersuchungen in Abschnitt 4 Anhang A
- Die Beurteilung der Sprachen nach ihrer Unterstützung der Strukturierten Programmierung ist das Ergebnis von Abschnitt 4 Anhang B

Für die Programmentwicklung ist neben den Eigenschaften einer Sprache die Unterstützung durch Dienstleistungsprogramme wichtig. In den Tabellen 3 und 4 sind die Sprachen auf diese Unterstützung hin untersucht.

Anmerkungen zur Tabelle 3:

- Die in Tabelle 3 enthaltenen Angaben gelten für die Dienstleistungsprogramme des Entwicklungssystems MDS 800 von INTEL
- Eine ausführliche Untersuchung der Leistungsmerkmale der Übersetzer wird in Abschnitt 4 Anhang C durchgeführt.

Anmerkungen zur Tabelle 4:

- Die in der Tabelle enthaltenen Angaben gelten für Dienstleistungsprogramme, die auf Universalrechnern betrieben werden.
- Eine ausführliche Untersuchung der Leistungsmerkmale der Übersetzer wurde in Abschnitt 4, Anhang D durchgeführt.

Zu den Aussagen über die derzeitige Unterstützung des Hardware-Software-Übergangs in Tabelle 4 ist folgendes zu bemerken:

Das Fehlersuchprogramm ICE und die Schnittstellen zwischen dem Entwicklungssytem von INTEL und des Hardwaremodells des Zielrechners unterstützen den Test des Objektes unter Realzeit-Bedingungen sehr gut. Es muß bei den von den Softwarehäusern gelieferten Crosscompilern und Crossassemblern untersucht werden, ob der Objektcode von den Fehlersuchprogrammen des Entwicklungssystems bearbeitet werden kann.

Tabelle 2: Beurteilung der Sprachen nach dem Kriterium "Spracheigenschaften"

(Beurteilung: ++ sehr gut, + gut, 0 ausreichend, - unbefriedigend)

Kriterien/Sprachen	8085 ASSEMBLER	PLM-80	MISTRAL	ESPL-1	CPL-1	FORTRAN-80	PASCAL	PEARL	CHILL
Art der Sprache	maschinennah	Untermenge von PL/1	Untermenge von ALGOL maschinennah	Untermenge von PL/1	Untermenge von PL/1	Untermenge von Fortran IV	neu entw. Hochsprache	neu entw. Hochsprache für Echtzeitprogrammierung	neu entw. Hochsprache für Vermittlungstechnik
Verbreitung der Sprache	weltweit	weltweit	ITT	ITT Vermittlungstechnik	mehrere Anwender in Frankreich, Deutschl.	weltweit	Hochsch. in USA, Deutschl. Schweiz, Siemens	Deutschl. Prozeßrechner Anwendung	international, Vermittlungstechnik
Voraussichtliche Unterstützung der Sprache auf lange Sicht	durch INTEL	durch INTEL	durch ITT intern	durch ITT intern	durch CAP SOGETI LOGICIEL	durch INTEL und unabh. Softwarehersteller	Hochsch., Texas Instrument, Zilog	Deutsche Prozeßr. Hersteller AEG, Siemens	international durch Postverwaltungen und Postlieferanten
Abschluß des grundlegenden Entwurfs der Spr.	8080 ASSEMBLER 1974	1974	Anfang 1978	1970	1973	1978 FORTRAN IV 1957	1968	1977	1979
Eignung als Hochsprache für Steuerungsaufgaben	keine Hochsprache	-	keine Hochsprache	+	0	-	0	++	++
Unterstützung der Strukturierten Programmierung	keine	+	0	0	0	-	+	++	++

Tabelle 3: Beurteilung der Sprachen anhand der Unterstützung durch das Entwicklungssystem MDS 800 von INTEL

	8085 ASSEMLBER	PL/M-80	MISTRAL ESPL-1 CPL-1	FORTRAN-80	SUBSET PASCAL SUBSET PEARL SUBSET CHILL
Übersetzer für 8085 auf MDS 800 verfügbar	ja	ja	nicht relevant, da keine Übersetzer verfügbar	ja	nicht relevant, da keine Übersetzer verfügbar
Übersetzer für 8085 auf MDS 800 Ineffizienzfaktor	nicht relevant	2 (Schätzwert)		2 bis 3 (Schätzwert INTEL)	
Übersetzer für 8085 auf MDS 800 Mischbarkeit mit Assembler im Programmtext	nicht relevant	nein		nein	
Übersetzer für 8085 auf MDS 800 relokativer Code	ja	ja		ja	
Binder, Lader verfügbar	ja			ja	
Objektcode-Simulator verfügbar	ja			ja	
Bestehende Softwareunterstützung	gut, jedoch nur durch INTEL			gut, jedoch nur durch INTEL	
Derzeitige Unterstützung des Hard-Software-Übergangs	durch Fehlersuchprogramme des Entwicklungssystems gut unterstützt			wie für 8085 Assembler und PL/M-80 gut	
Bestehende Erfahrung	gut	ausr.		Compiler besteht seit 1978	
Gesamtbeurteilung der Leistungsmerkmale der Dienstleistungsprogramme	gut			gut	
Laufzeit der Dienstleistungsprogramme	unbefriedigend			unbefriedigend	

Tabelle 4: Beurteilung der Sprachen anhand der Unterstützung der Softwareentwicklung durch Cross-Software (Zielrechner INTEL 8085)

		ASSEMBLER 8085		PL/M-80	MISTRAL	ESPL-1	CPL-1	FORTRAN 80, Untermengen von PASCAL, PEARL und CHILL
Anbieter oder Hersteller der Cross-Software		INTEL	Hersteller ITT	INTEL	Hersteller ITT	Hersteller ITT	CAP SOGETI LOGICIEL	Nicht relevant, da keine Übersetzer verfügbar
Dienstrechner, auf denen die Cross-Software implementiert wurde		Rechner mit 32 bit	HP 1000	Rechner mit 32 bit	HP 1000	IBM 370	IBM 370, PDP11, Siemens 7740	
Crosscompiler für Zielrechner 8085	Sprache, in der Übersetzer geschrieben	ANSI FORTRAN IV	HP FORTRAN 95 %, HP ASSEMBLER 5 %	ANSI FORTRAN IV	HP FORTRAN 95 %, HP ASSEMBLER 5 %	POPS	PL/1	
	verfügbar	ja	ja	ja	ja	ja	ja	
	Ineffizienzfaktor	nicht relevant		2	1,1...2 Schätzw.	1,2 laut Anwender	1,3 laut Anbieter	
	Mischbarkeit mit Assembler im Progr.-text			nein	ja	ja	ja	
	verschiebbarer Objektcode	nein	ja	nein	ja	ja	ja	
Binder verfügbar		nicht relevant	ja	nicht relevant	ja	ja	ja	
Lader verfügbar		ja	ja	ja	ja	ja	ja	
Simulator verfügbar		ja	ja	ja	ja	nein	nein	
Beurteilung der bestehenden Softwareunterstützung		keine Unterstützung	gut	keine Unterstützung	gut	gut	gut	
Derzeitige Unterstützung des Hardware-Software-Übergangs		MDS 800				MDS 800 geplant, zur Zeit PROM-Emulator und Logiktester	PROM-Emulator und Logiktester	
Gesamtbeurteilung der Leistungsmerkmale der Dienstleistungsprogramme		befr.	gut	befr.	gut	z. Zt. nur befr., da Objektcode nicht auf Dienstr. getestet werden kann	nur befr. da Objektcode nicht auf Dienstr. getestet werden kann	
Beurteilung der Laufzeit der Dienstleistungsprogramme		gut						

3. Beispielhafte Auswahl bevorzugter Sprachen

Fall I a

Fertigstellungstermin: kurzfristig
Projektdurchführung: SEL/ITT alleine
Bevorzugte Sprachen: ASSEMBLER 8085, ESPL-1

Gründe für die Auswahl der Sprache ASSEMBLER 8085:

- Sofortige Verfügbarkeit erprobter Dienstleistungsprogramme des Mikrorechner-Entwicklungssystems
- Gute Unterstützung der Fehlersuche durch das Entwicklungssystem MDS von INTEL
- Fortschrittliche Crossassembler und Programme für den Test des Objektcodes auf Universalrechnern von Softwarehäusern angeboten [21]

Gründe für die Auswahl der Sprache ESPL-1:

- ITT-eigene Sprache, für ITT-Vermittlungssysteme standardisiert, für verwandte Steuerungsaufgaben ebenfalls gut geeignet
- Erfahrung in der Programmierung mit ESPL-1 speziell für Vermittlungstechnik
- Fortschrittliche Crosscompiler auf der IBM 370 verfügbar

Bemerkungen zu ESPL-1:

- Übersetzer auf Entwicklungssystem MDS 800 nicht verfügbar
- Test des Objektcodes zur Zeit nur durch PROM-Emulator und Logik-Tester möglich (Test des Objektcodes mit Hilfe des Entwicklungssystems MDS 800 geplant)
- ESPL-1 wurde ursprünglich für ITT-Rechner mit 32-bit-Wortformat entwickelt.

Fall I b

Fertigstellungstermin: kurzfristig
Projektdurchführung: mit anderen Firmen zusammen
Bevorzugte Sprachen: ASSEMBLER 8085, PL/M-80 (als Übergangslösung)

Gründe für die Auswahl der Sprachen ASSEMBLER 8085 und PL/M-80:

- Sofortige Verfügbarkeit erprobter Dienstleistungsprogramme des Mikrorechner Entwicklungssystems MDS von INTEL
- Gute Unterstützung der Fehlersuche durch das Entwicklungssystem
- Crossassembler und Crosscompiler sowie Programme für den Test des Objektcodes auf Universalrechnern von Softwarehäusern [21] angeboten
- Sprachen von Fremdfirmen wahrscheinlich akzeptierbar

Bemerkungen zu PL/M-80

PL/M-80 stellt eine Übergangslösung dar. CHILL oder PEARL sind besser für die Programmierung nachrichtentechnischer Steuerungsaufgaben geeignet. Diese Sprachen sollten für neue Projekte anstelle von PL/M-80 eingesetzt werden, sobald die notwendigen Dienstleistungsprogramme für den Zielrechner INTEL 8085 wie Übersetzer, Binder, Lader und Objektcode-Simulator verfügbar sind.

Programmteile innerhalb PL/M-80 mit hohen Amforderungen an die Effizienz bezüglich Laufzeit sollten in ASSEMBLER 8085 geschrieben werden.

Bemerkungen über die Portabilität (FORTRAN-80) und die Effizienz der Objektcodeerzeugung:

FORTRAN-80 ist wie PL/M-80 nur wenig für die Steuerung nachrichtentechnischer Anlagen geeignet. Mit PL/M-80 lassen sich jedoch Datenstrukturen und strukturierte Programmabläufe besser als mit FORTRAN-80 darstellen. Außerdem ist wegen der Mischbarkeit von Booleschen und arithmetischen Operationen die Verarbeitung einzelner Bits oder Bitgruppen mit PL/M-80 effizienter programmierbar als mit FORTRAN-80 (Ineffizienzfaktor der Objektcodeerzeugung bei FORTRAN-80 größer als bei PL/M-80). Demgegenüber steht bei FORTRAN-80-Programmen das Argument der Portabilität. Es wäre jedoch zu prüfen, in wieweit eine Untermenge von FORTRAN-80 mit einer standardisierten FORTRAN-Version kompatibel sein kann. Es ist möglich, daß die kompatible Befehlsmenge zu höheren Ineffizienzfaktoren führt oder daß die Portabilität durch den Einbau maschinennaher Befehle in den Programmtext wieder in Frage gestellt wird.

Fall II a — Fertigstellungstermin: langfristig
Projektausführung: SEL/ITT alleine
Bevorzugte Sprachen: MISTRAL, CHILL

Gründe für die Auswahl der Sprache MISTRAL:

- Unterstützung der Strukturierten Programmierung
- gute Laufzeit - Effizienz durch maschinennahe Befehle

Gründe für die Auswahl der Sprache CHILL:

- Fortschrittliche Sprache für Vermittlungssysteme und Informationsabrufsysteme
- International von Postverwaltungen gefördert
- gute Unterstützung der Strukturierten Programmierung

Bemerkungen

- Für CHILL sind zur Zeit keine Dienstleistungsprogramme zur Software-Entwicklung für den Mikrorechner INTEL 8085 verfügbar.
- Für MISTRAL sind keine Dientsleistungsprogramme auf dem Entwicklungssystem MDS 800 verfügbar.

Fall II b Fertigstellungstermin: langfristig
Projektdurchführung: mit anderen Firmen zusammen
Bevorzugte Sprachen: ASSEMBLER 8085, CHILL, PEARL

Gründe für die Auswahl der Sprachen CHILL und PEARL:

- Sprachen von anderen Firmen wahrscheinlich akzeptierbar

Gründe für die Auswahl von CHILL:

- Forschrittliche Sprache, für Vermittlungssysteme und Informationsabrufssystem gut geeignet
- International von Postverwaltungen gefördert
- Gute Unterstützung der Strukturierten Programmierung

Gründe für die Auswahl von PEARL:

- Für nationale Projekte mit vorwiegendem Einsatz von Prozeßrechnern für Steueraufgaben, bei denen Betriebs-

systemfunktionen im Vordergrund stehen.

Bemerkungen:

- Für CHILL und PEARL sind zur Zeit keine Dienstleistungsprogramme zur Software-Entwicklung für den Mikrorechner INTEL 8085 verfügbar.

4. Anhang A

Eignung der Sprachen

Zur Beurteilung der Sprachen wurden die in Tabelle A 1 zusammengestellten Befehle und Leistungsmerkmale ausgewählt, die für die Steuerung nachrichtentechnischer Anlagen erforderlich sind. Mit einer ähnlichen Tabelle analysierte das CCITT-Gremium unter anderem die Sprachen ESPL-1, PASCAL und PEARL [16] . Die Ergebnisse dieser Analyse werden im folgenden mitverwendet. Es sei in diesem Zusammenhang darauf hingewiesen, daß in [23] ein Pflichtenheft für eine neu zu entwickelnde universelle Sprache für das amerikanische Verteidigungsministerium veröffentlicht wurde (Vorläufige Bezeichnung für die Sprache 'DoD1'). Das Einsatzgebiet dieser Sprache ist jedoch wesentlich weiter als das von CHILL.

In Tabelle A 10 ist die Eignung der einzelnen Sprachen beurteilt. Grundlage für die Beurteilung ist die Summe der fehlenden Befehle. Dabei werden die einzelnen Befehle nach ihrem Gewicht berücksichtigt.

Tabelle A 1: Befehle und Leistungsmerkmale zur Beurteilung der Sprachen

(Bewertung: ++ notwendig, + wünschenswert, O brauchbar)

Befehle und Leistungsmerkmale	Einstufung
Unterprogramme, Module	
Vereinbarung getrennt übersetzbarer, schachtelbarer Module und Unterprogramme	++
Vereinbarung globaler Daten als Schnittstelle zwischen getrennt übersetzbaren Modulen	++
Vereinbarung von Unterprogrammen mit Übergabe der Parameter als Wert (call by value)	++
Vereinbarung von Unterprogrammen mit Übergabe der Adresse der Parameter (call by reference)	++
Vereinbarung von Unterprogrammen, deren Name die Ergebnisse der Unterprogramme symbolisch adressiert (FUNCTION)	++
Gliederung des Programmablaufs	
Verzweigung (IF...THEN...ELSE...)	++
Fallunterscheidung (CASE...OF...)	++
Schleife (DO...WHILE,REPEAT...UNTIL) oder (WHILE...DO)	++
Zählschleife, mit Festlegung der Schleifenparameter vor Eintritt in die Schleife	++
Vereinbarung von Datentypen	
Ganze Zahl (INTERGER)	++
Kommazahl (REAL, FIX/FLOAT POINT)	+
Boolesche Variable (BOOLEAN)	++
Zeichen (CHARACTER)	++
Zeiger (POINTER)	++
Auf Datentypen spezialisierte Zeiger (POINTER TYPE)	+
Marken (LABEL)	++
Bitkette (BINARY STRING)	++
Zeichenkette (CHARACTER STRING)	O
Durch den Wertbereich festgelegter Datentyp (POWER SET)	+
Ereignisse (EVENT)	+
Felder (ARRAY)	
- Eindimensional	++
- Mehrdimensional	+
Datenstruktur mit Daten verschiedenen Typs (RECORD, STRUCTURE)	++
Vereinbarung einer vom Programmierer festgelegten Struktur als Datentyp (NEW MODE STRUCTURE)	+
Liste	O
Stapelspeicher (Bezieht sich nicht auf den Stapelspeicher im Mikrorechner) (STACK)	O

Tabelle A 1 (Fortsetzung)

Befehle und Leistungsmerkmale	Einstufung
Lebensdauer der Daten	
Festlegung der Lebensdauer der Daten	
- Durch Blockgrenzen (LOCAL)	++
- Durch Anfang und Ende des Programms (OWN)	++
- Durch den Programmierer (ALLOCATE, FREE)	0
Zugriff auf Daten im Programmablauf	
Festlegung des Datenzugriffs bei geschachtelten Lebensdauerbereichen für Daten mit gleicher symbolischer Adresse (STATIC)	++
Gleichsetzen symbolischer Adressen	
Gleichsetzen mehrerer symbolischer Adressen, die sich auf denselben Datentyp beziehen (OVERLAY, EQUIVALENCE, UNITING)	+
Festlegung eines gemeinsamen Speicherbereichs	
Festlegung eines mehreren Unterprogrammen gemeinsamen Speicherbereichs mit Zugang über verschiedene symbolische Adressen (COMMON)	0
Operationen	
Arithmetische Operationen	++
Vergleichende Operationen	++
Boolesche Operationen	++
Verarbeitung einzelner Bits oder Bitgruppen	++
Arithmetische und vergleichende Operationen mit Zeigern	++
Wertzuweisung zu einem Feldelement	++
Operationen mit Feldelementen	++
Verarbeitung von Zeichenketten (z.B. CONCATENATION)	0
Sprungbefehle	
Unbedingter Sprung (GO TO)	++
Unbedingter Sprung an das Blockende (EXIT)	++
Unbedingter Sprung mit berechenbarer Sprungweite (COMPUTED-, ASSIGNED GO TO)	0

Tabelle A 1 (Fortsetzung)

Befehle und Leistungsmerkmale	Einstufung
Realzeit-System	
Vereinbarung von Prozessen	+
Vereinbarung von Ereignissen	+
Synchronisation paralleler Prozesse	+
Durch Unterbrechungssignale bedingte Verzweigung des Programmablaufs	+
Datentransfer	
Ein- und Ausgabe von Daten an die Peripherie	++
Effizienz der Programme bezüglich Speicherplatz	
Dichte Packung der Daten (PACK)	++
Effizienz der Programme bezüglich Laufzeit	
Mischbarkeit von Hoch- und Maschinensprache	
—Innerhalb des Programmtextes	0
—Über getrennt übersetzbare Module	++
Effizienz bezüglich Programmierleistung	
Makro - Befehle	++

Tabelle A 2: Ergebnis der Analyse der Sprache PL/M-80

Fehlende Befehle gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Boolesche Variable	Komma-Zahl	Zeichenkette mit variabler Anzahl von Zeichen
Bitkette mit variabler Anzahl von Bits	Auf Datentypen spezialisierte Zeiger	Liste
Verarbeitung einzelner Bits oder Bitgruppen	Durch den Wertebereich festgelegter Datentyp	Stapelspeicher
Bedingter Sprung an das Blockende	Vereinbarung von Ereignissen	Verarbeitung von Zeichenketten
Vereinbarung von Unterprogrammen mit Übergabe der Parameter als Wert (call by value)	Vereinbarung einer vom Programmierer festgelegten Struktur als Datentyp	Unbedingter Sprung mit berechenbarer Sprungweite
Zählschleife, mit Festlegung der Schleifenparameter vor Eintritt in die Schleife	Vereinbarung von Prozessen	Mischbarkeit von Hoch- und Maschinensprache innerhalb des Programmtextes
Dichte Packung der Daten	Synchronisation paralleler Prozesse	Festlegung der Lebensdauer der Daten durch den Programmierer
Makro-Befehle	Gleichsetzen mehrerer symbolischer Adressen, die sich auf denselben Datentyp beziehen	Festlegung eines mehreren Unterprogrammen gemeinsamen Speicherbereiches mit Zugang über verschiedene symbolische Adressen

Tabelle A 3: Ergebnis der Analyse der Sprache MISTRAL

Fehlende Befehle gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Eindimensionale Felder	Komma-Zahlen	Zeichenkette mit variabler Anzahl von Zeichen
Mehrdimensionale Felder	Auf Datentyp spezialisierte Zeiger	
Datenstruktur mit Daten verschiedenen Typs	Durch den Wertebereich festgelegter Datentyp	Stapelspeicher
Unbedingter Sprung	Vereinbarung von Ereignissen	Festlegung der Lebensdauer durch den Programmierer
Vereinbarung von Unterprogrammen mit Übergabe der Parameter als Wert (call by value)	Vereinbarung einer vom Programmierer festgelegten Struktur als Datentyp	Festlegung eines mehreren Unterprogrammen gemeinsamen Speicherbereichs mit Zugang über verschiedene symbolische Adressen
Vereinbarung von Unterprogrammen und Übergabe der Adresse der Parameter (call by reference)	Gleichsetzen mehrerer symbolischer Adressen, die sich auf denselben Datentyp beziehen	Verarbeiten von Zeichenketten
Zählschleife, mit Festlegung der Schleifenparameter vor Eintritt in die Schleife	Vereinbarung von Prozessen	Unbedingter Sprung mit berechenbarer Sprungweite
Dichte Packung der Daten	Synchronisation paralleler Prozesse	
Vereinbarung getrennt übersetzbarer, schachtelbarer Module und Unterprogramme	Durch Unterbrechungssignale bedingte Verzweigung des Programmablaufs	
Vereinbarung globaler Daten als Schnittstelle zwischen getrennt übersetzbaren Modulen		

Tabelle A 4: Ergebnis der Analyse der Sprache ESPL-1

Fehlende Befehle, gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Boolesche Variable	Komma-Zahlen	Liste
Operationen mit Feldelementen	Auf Datentyp spezialisierter Zeiger	Stapelspeicher
	Vereinbarung von Ereignissen	Festlegung der Lebensdauer der Daten automatisch durch Blockgrenzen
	Vereinbarung von Prozessen	Festlegung der Lebensdauer durch den Programmierer
	Synchronisation paralleler Prozesse	
	Durch Unterbrechungssignale bedingte Verzweigung des Programmablaufs	

Tabelle 5: Ergebnis der Analyse der Sprache CPL-1

Fehlende Befehle, gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Boolesche Variable	Auf Datentyp spezialisierte Zeiger	Stapelspeicher
Vereinbarung getrennt übersetzbarer, schachtelbarer Module und Unterprogramme	Durch den Wertebereich festgelegter Datentyp	Festlegung der Lebensdauer durch den Programmierer
Vereinbarung globaler Daten als Schnittstelle zwischen getrennt übersetzbaren Modulen	Vereinbarung von Ereignissen	Festlegung eines mehreren Unterprogrammen gemeinsamen Speicherbereichs mit Zugang über verschiedene symbolische Adressen
Fallunterscheidung	Mehrdimensionale Felder	
Ein- und Ausgabe von Daten an die Peripherie	Vereinbarung einer vom Programmierer festgelegten Struktur als Datentyp	Unbedingter Sprung mit berechenbarer Sprungweite
Dichte Packung der Daten	Gleichsetzen mehrerer symbolischer Adressen, die sich auf denselben Datentyp beziehen	
Mischbarkeit von Hoch- und Maschinensprache über getrennt übersetzbare Module		
Makro-Befehle		

Tabelle A 6: Ergebnis der Analyse der Sprache FORTRAN-80

Fehlende Befehle, gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Zeiger	Auf Datentyp spezialisierte Zeiger	Zeichenkette als Variable
Bitkette als Variable	Durch den Wertebereich festgelegter Datentyp	Liste
Datenstruktur mit Daten verschiedenen Typs	Vereinbarung von Ereignissen	Stapelspeicher
Festlegung des überlappenden Zugriffs bei Lebensdauerbereichen für Daten mit gleicher symbolischer Adresse	Vereinbarung einer vom Programmierer festgelegten Struktur als Datentyp	Festlegung der Lebensdauer durch den Programmierer
Festlegung der Lebensdauer der Daten durch Blockgrenzen	Vereinbarung von Prozessen	Mischbarkeit von Hoch- mit Maschinensprache innerhalb des Programmtextes
Fallunterscheidung	Synchronisation paraller Prozesse	Verarbeitung von Zeichenketten
Schleife	Durch Unterbrechungssignale bedingte Verzweigung des Programmablaufs	
Dichte Packung der Daten		
Verarbeitung einzelner Bits oder Bitgruppen		
Arithmetische und vergleichende Operationen mit Zeigern		
Unbedingter Sprung an das Blockende		

Tabelle A 7: Ergebnis der Analyse der Sprache PASCAL

Fehlende Befehle, gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Bitkette	Vereinbarung von Ereignissen	Vereinbarung von Zeichenketten
Zeichenkette	Vereinbarung von Prozessen	Unbedingter Sprung mit berechenbarer Sprungweite
Vereinbarung einzelner Bits oder Bitgruppen	Synchronisation paralleler Prozesse	Mischbarkeit von Hoch- mit Maschinensprache innerhalb des Programmtextes
Arithmetische Operationen mit Zeigern	Durch Unterbrechungssignale bedingte Verzweigung des Programmablaufs	Festlegung eines mehreren Unterprogrammen gemeinsamen Speicherbereichs mit Zugang über verschiedene symbolische Adressen
Unbedingter Sprung an das Blockende		
Vereinbarung getrennt übersetzbarer, schachtelbarer Module und Unterprogramme		
Vereinbarung globaler Daten als Schnittstelle zwischen getrennt übersetzbaren Modulen		
Mischbarkeit von Hoch- mit Maschinensprache über getrennt übersetzbare Module		
Makro-Befehle		

Tabelle 8: Ergebnis der Analyse der Sprache PEARL

Fehlende Befehle, gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Boolesche Variable Festlegung der Lebensdauer der Daten durch Anfang und Ende des Programms	Gleichsetzen mehrerer symbolischer Adressen, die sich auf denselben Datentyp beziehen	Liste Stapelspeicher Festlegung eines mehreren Unterprogrammen gemeinsamen Speicherbereichs mit Zugang über verschiedene symbolische Adressen

Tabelle A 9: Ergebnis der Analyse der Sprache CHILL

Fehlende Befehle, gegliedert nach ihrer Einstufung (++, +, 0)

++	+	0
Makro-Befehle	Komma-Zahlen Gleichsetzen mehrer symbolischer Adressen, die sich auf denselben Datentyp beziehen Durch Unterbrechungssignale bedingte Verzweigung des Programmablaufs	Festlegung eines gemeinsamen Speicherbereichs mit Zugang über verschiedene symbolische Adressen Mischbarkeit von Hoch- mit Maschinensprache innerhalb des Programmtextes

Tabelle A 10: Bewertung der Sprachen nach der Summe der gewichteten fehlenden Befehle

Gewichtung der Befehle: ++ = 3
+ = 2
0 = 1

Sprache	Anzahl der fehlenden Befehle gegliedert nach ihrer Einstufung			Summe der gewichteten fehlenden Befehle	Bewertung der Sprache
	notwendig	wünschenswert	brauchbar		
CHILL	1	3	2	11	sehr gut
PEARL	2	1	3	11	
ESPL-1	2	6	4	22	gut
PASCAL	8	4	4	36	ausreichend
CPL-1	8	6	4	40	
PL/M-80	9	8	8	51	unbefriedigend
FORTRAN-80	11	7	6	53	
MISTRAL	10	9	6	54	ohne Bewertung, da keine Hochsprache

Anhang B

Unterstützung der Strukturierten Programmierung

Als Grundlage für die Beurteilung wurde die Anzahl der fehlenden Befehlskonstruktionen der Strukturierten Programmierung gewählt. Tabelle B 1 zeigt das Ergebnis.

Die in einer Sprache nicht verfügbaren Befehlskonstruktionen können behelfsmäßig durch

- Makro-Befehle
- Unterprogrammtechnik
- Einsatz von Sprungbefehlen

nachgebildet werden.

In Tabelle B 2 ist die Unterstützung der Strukturierten Programmierung zusammenfassend beurteilt.

Tabelle B 1: Unterstützung der Strukturierten Programmierung

(1 = vorhanden, 0 = nicht vorhanden)

Befehlskonstruktionen	PL/M-80	MISTRAL	ESPL-1	CPL-1	FORTRAN-80	PASCAL	PEARL	CHILL
Verzweigung	1	1	1	1	1	1	1	1
Fallunterscheidung	1	0	0	0	0	1	1	1
Schleife WHILE..DO oder DO WHILE (REPEAT.. UNTIL)	1	1	1	1	0	1	1	1
Allgemeine Schleife mit vorzeitigem Aussprung	0	0	0	0	0	0	1	1
Andere Schleifenkonstruktionen	Zählschleife	Schleife mit Abfrage im Schleifenkörper	Zählschleife mit vorzeitigem Ausprung	Zählschleife mit vorzeitigem Aussprung	Zählschleife	Zählschleife	Zählschleife mit vorzeitigem Aussprung	Zählschleife mit vorzeitigem Aussprung

Tabelle B 2: Beurteilung der Sprachen anhand der Unterstützung der strukturierten Programmierung

Beurteilung (++ sehr gut, + gut, 0 ausreichend, - unbefriedigend)

Sprache	Anzahl der vorhandenen Befehlskonstruktionen	Beurteilung
PEARL CHILL	4	++
PASCAL PL/M-80	3	+
MISTRAL ESPL-1 CPL-1	2	0
FORTRAN-80	1	-

Anhang C

Übersetzer, welche auf Mikrorechner-Entwicklungssystemen betrieben werden

In Tabelle C 1 sind die Eigenschaften der Übersetzer zusammengestellt, die für das Mikrorechner-Entwicklungssystem MDS von INTEL und für den Zielrechner INTEL 8085 geliefert und betreut werden. (In [21] sind einige Softwarehäuser als Anbieter weiterer Übersetzer aufgeführt. Diese Übersetzer sind hier nicht untersucht worden).

Tabelle C 1: Analyse der Übersetzer des Entwicklungssystems von INTEL (Zielrechner: INTEL 8085)

Kriterien / Sprache	8085 ASSEMBLER	PL/M-80 FORTRAN-80	MISTRAL ESPL-1 CPL-1 PASCAL PEARL CHILL
Stufenweise Übersetzung von Hochsprache in ASSEMBLER 8085, dann von ASSEMBLER 8085 in Maschinensprache	nicht relevant	ja	nicht relevant, da keine Übersetzer für INTEL Entwicklungssystem vorhanden
Möglichkeit zur Laufzeit- oder Speicherplatzoptimierung durch zusätzlichen Übersetzerlauf		ja, Optimierung bez. Datentransfer zwischen Registern und Arbeitsspeichern	
Umfassender, verständlicher Fehlerausdruck		ja	
Übersichtliche Gegenüberstellung von Programmtext (Quellcode) und Maschinencode (Objektcode)		ja	
Ausdruck von Tabellen mit symbolischen Adressen und Querverweisen		ja	
Speicherplatzbedarf (Übersetzer + Tabellen)	48 KByte	64 KByte	
Testbarkeit des Objektcodes an einem Hardware-Modell des Zielrechners	gut unterstützt durch MDS-ICE Fehlersuchroutine		
Beurteilung der Leistungsmerkmale und des Komforts der Übersetzer	ausreichend		
Beurteilung der Laufzeit der Übersetzer	unbefriedigend		

Anhang D

Analyse der Übersetzer, welche auf Universalrechnern betrieben werden

In Tabelle D 1 sind einige verfügbare Übersetzer für den Zielrechner INTEL 8085 aufgeführt. Darunter befinden sich auch Übersetzer, die von INTEL früher einmal geliefert worden sind, heute aber nicht mehr von INTEL gewartet werden.

Für die Entwicklung von Software auf dem Großrechner für einen Mikrorechner als Zielrechner sind unbedingt notwendig:

- Ein Übersetzer für eine Hochsprache und ein Übersetzer für die maschinennahe Sprache des Zielrechners. Die Übersetzer sollten verschiebbaren Objektcode erzeugen. Der vom Universalrechner und vom Entwicklungssystem erzeugte Objektcode sollte kompatibel sein.
- Binder und Lader für den vom Großrechner erzeugten Objektcode
- Simulationsprogramm für den Test des Objektcodes des Mikrorechners auf dem Universalrechner

Der Dialog mit dem Universalrechner sollte durch Bildschirm-Terminals (z.B. unter TSO-SPF bei IBM 370) möglich sein. Wünschenswert ist eine Verbindung zwischen Großrechner und Mikrorechner-Entwicklungssystem für den Datentransfer.

Tabelle D 1: Übersetzer, die auf Universalrechnern als Dienstrechner betrieben werden (Zielrechner: INTEL 8085)

Kriterien / Sprache	ASSEMBLER	8085	PL/M-80	MISTRAL	ESPL-1	CPL-1	FORTRAN-80 PASCAL PEARL CHILL
Herkunft des Übersetzers	INTEL	ITT	INTEL	ITT	ITT	CAP SOGETI LOGICIEL	nicht relevant, da zur Zeit keine Crosscompiler für diese Sprachen vorhanden
Sprache, in der Übersetzer geschrieben	ANSI FORTRAN IV	HP FORTRANund HP ASSEMBLER	ANSI FORTRAN IV	HP FORTRAN und HP ASSEMBLER	POPS (Makrosprache für ITT-Rechner)	PL/1	
Dienstrechner	Rechner mit 32-Bit Integer Format	HP 1000	Rechner mit 32-Bit Integer Format	HP 1000	IBM 370	IBM 370, Siemens 7740, PDP 11 u.a.	
Verfügbarkeit des Übersetzers	ja	ja, innerhalb ITT	ja	ja, innerhalb ITT	ja, innerhalb ITT	ja	
Möglichkeit, den unveränderten Objektcode in das Mikrorechnerentwicklungssystem von INTEL zu laden	ja	ja	ja	ja	z. Zt. in Entwickl.	nein	
Anwendbarkeit der MDS-ICE Routinen auf den vom Übersetzer erzeugten Objektcode	ja nach Umwandlung mit MDS 'HEXOBT' Programm	ja	ja nach Umwandlung mit MDS 'HEXOBT' Programm	ja	z. Zeit in Entw.	nein	
Bereitstellung weiterer wichtiger, kompatibler Dienstleistungsprogramme durch den Anbieter des Übersetzers	Objektcode-Simulator	Linkage Editor, Objektcode-Simulator	Objektcode-Simulator	Linkage Editor, Objektcode-Simulator	Linkage Editor auf IBM 370, Testprogramme auf ITT 3200	Linkage Editor	

5. Literaturverzeichnis

[1] 8080/8085 Assembly Language Programming Manual. INTEL, 98-301 A, 1977

[2] I.A. DeMan, R.T. Boute: MISTRAL-M, User Manual. Bell Telephone Manufacturing Company, May 1978

[3] I.A. DeMan, R.T. Boute, D.W. Wright: MISTRAL-M, a Medium Level Programming Language for Microcomputers. Fourth EUROMICRO Symposium on Microprocessing 17-19, 1978, Munich. NORTH-HOLLAND

[4] PL/M-80 Programming Manual. INTEL, 98-268, 1976

[5] ESPL-1 Language Reference Manual. ITT, Edition 3

[6] D. Cohen: ESPL-1, a Programming Language for Switching Systems. Conference on Software Engineering for Telecommunication Switching Systems, 2-5 April 1973

[7] CPL-1 Language Reference Manual. CAP SOGETI LOGICIEL, Edition 9

[8] FORTRAN-80 Programming Manual. INTEL 9800481 A, 1978

[9] N. Wirth: Systematisches Programmieren. Teubner Studienbücher, Informatik 1972

[10] K. Jensen, N. Wirth: PASCAL User Manual and Report. Springer Verlag 1978

[11] S. Alagić, M.A. Arbib: The Design of Well-Structured and Correct Programs.
Springer Verlag (Reihe 'Texts and Monographs in Computer Science'), 1978

[12] J.G. Posa: PASCAL becomes Software Superstar.
Electronics, October 12, 1978

[13] A Brief Description of the UCSD PASCAL Software System. University of California, San Diego, June 1, 1978

[14] KFK-PDV 130: Full PEARL Language Description. Gesellschaft für Kernforschung mbH, Karlsruhe 1977

[15] Jahresbericht der Fakultät für Informatik an der Universität Karlsruhe 1977

[16] Extracts from the Minutes of an Informal C.C.I.T.T. Meeting held in London from 25-27 March 1974.
CCITT-Period 1973-1976, Study Group XI - Contribution No. 73, May 1974

[17] Proposal for a Recommendation for a CCITT High Level Programming Language. (Second Edition) Blue Document, CCITT-Period 1977-1980, Study Group XI, March 1977

[18] Introduction to CHILL, Second Edition. Temporary Document No. 22-E CCITT Study Group XI Geneva, 12-16 June 1978

[19] Subset Languages and CCITT Conformity. CCITT Study Group XI, COM XI-No. 147-E, CCITT-Period 1977-1980, April 1978

[20] K. Schulz: CCITT-Programmiersprachen, Empfehlungen für Fernsprechvermittlungssysteme. NTG Fachausschuß 6 u. 9., "Steuerung in Datenverarbeitungs- und Vermittlungssystemen". 23.-25. Oktober, Baden-Baden

[21] M. Roony: Currently Available Microprocessor Software.
Small Systems Software, Vol. 2, No. 4, 1977, S. 9-13

[22] R. Weickert: Neue Konzepte und Entwürfe für Programmiersprachen.
Informatik Spektrum, Band 1, Heft 2 (Nov. 1978) S. 101-112

[23] Department of Defense Requirements for High Order Computer Programming Languages (Revised 'IRONMAN', July 1977).
ACM SIGPLAN 12, 1977, No. 12, S. 39-54

[24] Software Engineering for Telecommunication Switching Systems.
Third International Conference, 27 - 29 June 1978, IEE Conference Publication No. 164

[25] International Switching Symposium, Paris, 7 - 11 May 1979.