Pearl-Softwaresystem für gekoppelte Klein- und Mikrorechner

Von Dr. **H.J. Schneider**, Markdorf

1. Einleitung

Die Vorteile, die sich beim Einsatz höherer Programmiersprachen ergeben, sind hinreichend bekannt. Auch daß bei der Programmierung von Echtzeitanwendungen diese Sprache PEARL sein wird, liegt nahe.
Eine Implementierung von PEARL wurde auch bei Dornier System vorgenommen. Sie besteht nicht nur aus dem Compiler, sondern enthält folgende Komponenten:

- Generierungsprogramme
- Compiler-Oberteil
- Codegenerator
- Assembler
- Bibliotheksverwaltung
- modulares Betriebssystem
- Lader/Binder
- Test- und Integrationshilfen

Die Struktur des Systems zeigt Bild 1.

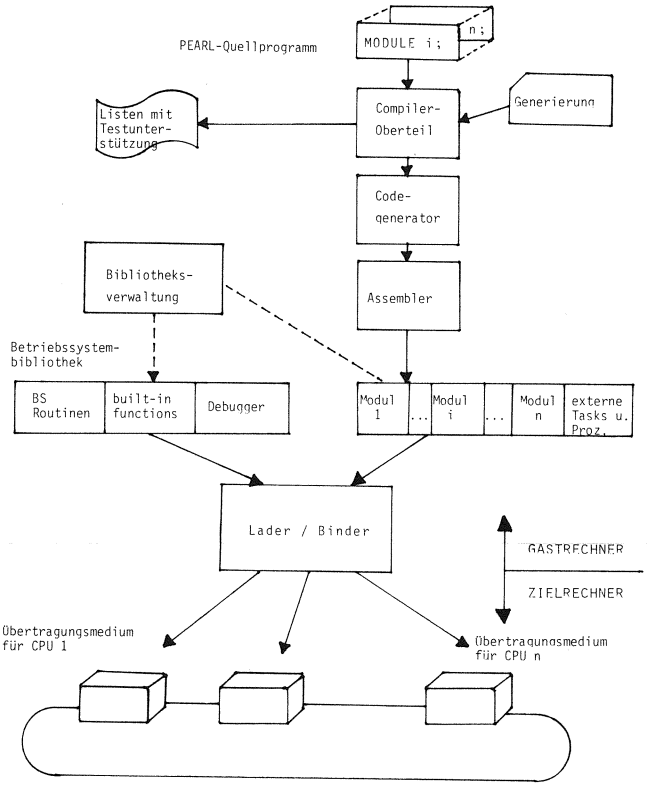

Bild 1: Struktur des PEARL-Softwaresystems

Das PEARL-System läuft auf folgenden Gastrechnern:

PDP 11/70
PDP 10
AEG 80-20
Siemens 7730
Interdata 8/32

und ist bzw. wird für folgende Zielrechner implementiert:

MUDAS 433
AEG 80-20
Intel 8086

Da der Einsatz von PEARL bei Dornier System hauptsächlich in Anwendungen aus der Luft- und Raumfahrt und der Wehrtechnik erfolgt - also primär für Bordrechner -, ergeben sich für eine PEARL-Implementierung gewisse Randbedingungen und Konsequenzen, nämlich:

- Zwang zu hoher Effizienz
- starke Unterstützung der Integrationsphase
- Notwendigkeit zur Erweiterung für verteilte Systeme
- Vermeidung jeglichen Overheads
- Notwendigkeit zur Aufteilung der Programme in PROM- und RAM-Anteile.

2. Sprachumfang

Diese Konsequenzen führten dazu, daß gewisse Basis-PEARL-Sprachelemente nicht verwendet werden, die von dieser Art der Anwendungen nicht benötigt werden und im Betriebssystem zu unnötigem Overhead führen würden. Im einzelnen sind es folgende Sprachelemente:

- Filehandling (kein Hintergrundspeicher bei Bordrechnern)
- Formatierung (sehr wenig Druckausgabe)
- Absolutzeit (Aufgaben des Bordrechners beginnen immer relativ zu Missionsanfang)
- Signals (im operationellen System dürfen keine Softwareinterrupts auftreten)
- Strukturen (meist gleichartige Messdaten)

Die Implementierung ist jedoch so angelegt, daß diese Sprachelemente leicht ergänzt werden können.

Auf der anderen Seite waren jedoch Erweiterungen von Basis-PEARL notwendig, und zwar

- für verteilte Systeme: GLOBAL NET
- für die Zulassung externer in Assembler geschriebener Tasks und Prozeduren: EXTERNAL
- für Testzwecke und Integrationsunterstützung: CHECK, NOCHECK

3. Compiler

Beim Compiler wurde die klassische Aufteilung in zielmaschinenunabhängiges Compiler-Oberteil und zielmaschinenabhängigen Codegenerator gewählt. Als Schnittstelle dazwischen wurde kein Pseudoassembler für eine virtuelle Maschine gewählt, sondern eine numerische Form, die Tripel:

Operator, Operand 1, Operand 2,

die noch nahe an der Sprache sind. Dadurch wurde eine Abbildung der virtuellen Maschine auf den tatsächlichen Zielprozessor vermieden, was der Effizienz des erzeugten Codes stark zugute kam. Weiterhin konnten Vorteile der vorhandenen Hardware bei der Codegenerierung ausgenutzt werden. Außerdem ist eine Optimierung bei dieser Zwischenform leicht möglich, bei Pseudoassembler dagegen nicht.

Der Compiler beinhaltet Testhilfsmittel und Benutzerunterstützung, die im allgemeinen nur in besonderen Verifikationstools enthalten sind. Auf Wunsch erhält der Anwender folgende unterstützende Ausgaben:

- eine Cross-Reference-Liste, die alle Variablen, Marken, Tasks, etc. mit Attributen, Anweisungsnummern ihrer Vereinbarung und Anweisungsnummern ihrer Verwendung enthält, wobei Veränderungen ihres Wertes besonders gekennzeichnet sind.
- eine Liste über das Taskingverhalten
- eine Liste über das Semaphoreverhalten
- eine Liste über die Aufrufhierarchie von Unterprogrammen
- eine Liste über die Speicherplatzvergabe taskinterner Variabler

Bei den Assemblern der Zielmaschinen wurde auf bereits existierende zurückgegriffen.

4. Modulares Betriebssystem

Aufgrund des Effizienzzwanges wurde für das Betriebssystem die Lösung gewählt, daß alle Funktionen des Systems sauber getrennt und dann einzeln realisiert werden. So kann dann das Anwenderprogramm automatisch vom Lader/Binder mit genau den Betriebssystemfunktionen versehen werden, die es benötigt. Die nicht benötigten Routinen werden auch nicht dazugebunden. Der Umfang des Betriebssystem richtet sich also nach dem Anwenderprogramm, er liegt bei

MUDAS 433	zwischen	0,5 und 6 K Worten
AEG 80-20	zwischen	0,3 und 4 K Worten
Intel 8086	zwischen	0,3 und 3 K Worten

Das Betriebssystem enthält im einzelnen die folgenden Funktionen:

- Initialisierung
- Dispatcher (Taskmanager)
- Exitroutine (Kurzfassung des Dispatchers, wird durchlaufen, wenn sicher ist, daß kein Taskwechsel erfolgen kann)

} Kern

- Uhrverwaltung
- Interruptverwaltung
- Taskbeendigung (regulär)
- Taskbeendigung (irregulär) (TERMINATE)
- Taskunterbrechung (SUSPEND)
- Taskfortsetzung (CONTINUE)
- Taskaktivierung (ACTIVATE)
- Schedule löschen (PREVENT)
- Semaverwaltung (REQUEST, RELEASE)
- Interrupt sperren (DISABLE)
- Interrupt freigeben (ENABLE)
- Charakter-E/A (PUT und GET)
- Prozedureintritt/-austritt
- Feldelementadressrechnung

- Arithmetikroutinen für FLOAT und DUR
- Vergleichsroutinen für FLOAT und DUR
- Datentypwandlungsroutinen
- Standardfunktionen (ABS, SIGN)
- Laufzeitfehlerroutine
- Bedienroutine
- Kommunikation

5. Lader/Binder

Die Hauptaufgabe des Lader/Binder ist das Zusammenstellen von anzugebenden Moduln zu einem ablauffähigen PEARL-Programm, wozu auch das automatische Dazubinden der benötigten Betriebssystemroutinen gehört, sowie das Sortieren der Code-, Daten- und Verwaltungslistenbereiche. Dabei kann wahlweise auch eine PROM/RAM-Aufteilung des Programms vorgenommen werden.
Im Falle von verteilten Systemen kann an dieser Stelle angegeben werden, welcher Modul in welchen Prozessor geladen werden soll. Soll die Prozessorverteilung später einmal geändert werden, ist das ohne neue Übersetzung möglich, da eine Korrespondenz durch den Zusatz

```
TASK(SEMA) IN MODULE NR;
```

für den Betrieb herstellbar ist.

6. Erfahrungen und Ausblick

Die Implementation des Softwaresystems hat sich inzwischen in konkreten Projekten bewährt. Die Vorteile liegen bei

- der Erstellung der Programme
- des Tests und der Inbetriebnahme
- der Dokumentation.

Der Zielforderung nach Effizienz konnte voll Rechnung getragen werden.
Für Projekte im Haus wird dieses PEARL-Softwaresystem weitgehend eingesetzt werden.

(Die Entwicklung des Systems erfolgte im Auftrag des BMVg.)