mbp PEARL Programmiersystem für SIEMENS-Prozeßrechner der Serie 300

K.-W. Müller

1. Allgemeines

Ziel der mbp-PEARL-Implementierung, die anläßlich der Hannover Messe 1981 erstmals öffentlich vorgestellt wurde, war ein möglichst großer Sprachumfang, der entweder abgestuft mit verschiedenen Testinformationen und dynamischen Kontrollen oder im Hinblick auf Speicher- und Rechenzeiteffizienz übersetzt werden kann.

Ausgangspunkt war eine inzwischen mehrmals überarbeitete portable Compiler- und Laufzeitsystemimplementierung, wie sie auch für Siemens 404/3 verwendet wurde /1, 2/. Die auf Übertragbarkeit gerichtete Konzeption einerseits und die Ähnlichkeit der Maschinencodes andererseits erleichterten eine Anpassung an die verschiedenen 300-er Systeme.

2. Sprachumfang

Für eine anwendungsfreundliche und dennoch effiziente Programmierung reicht der durch die DIN-Norm 66253 vorgegebene Sprachumfang von Basic PEARL nicht aus.
Der Sprachumfang von mbp-PEARL ist daher um wesentliche Elemente von Full PEARL erweitert worden wie

* Referenzen (REFERENCE, CONT)
* Synchronisation über BOLT-Variable
* benutzerdefinierte Datentypen (TYPE)
* benutzerdefinierte OPERATORen mit Angabe von PRECEDENCE
* Strukturen (STRUCTURE)
  - wobei Subkomponenten Felder oder Strukturen sein dürfen
  - Subkomponenten verschiedener Strukturen gleiche Namen haben können
  - Strukturen auch Resultat von Funktionen sein dürfen
* Felder mit
  - mehr als drei Dimensionen
  - statische Initialisierung mit INIT
  - untere Dimensionsangabe auch negativ
* Felder auch von REF, SEMA, BOLT, TYPE und DATION-Objekten
* Mehrfachzuweisungen
* variable Sub-Strings von Typ BIT und CHARACTER
* Prozeduren auf TASK-Ebene
* Listen von SEMA und BOLT
* SEMA, BOLT, REF und neue Datentypen (TYPE) als Prozedurparameter
* REF, STRUCTURE und neue Datentypen (TYPE) als RETURN-Attribut
* UPB, LWB auch monadisch
* bedingte Zuweisung
* ACTIVATE, CONTINUE mit Prioritätsangabe
* SUSPEND auch auf fremde Task
* TRIGGER
* OPEN, CLOSE mit CAN und PRM

Nähere Einzelheiten sind dem mbp-PEARL-Sprachreport /4/ zu entnehmen, der neben einer informellen Beschreibung anhand von Syntaxdiagrammen (Wirth) und vielen Beispielen auch eine vollständige Syntaxbeschreibung in Backus-Naur-Form enthält.

3. Hardware Voraussetzungen

Der Compiler und die erzeugten PEARL-Programme sind ablauffähig auf den Rechnertypen

330 mit SIM30R
R10
R20
R30

unter den Betriebssystemen

ORG 330 K
ORG 300 P
ORG 300 PV

Für den Compiler ist ein Laufbereich von 32K Worten und ein Plattenspeicher erforderlich.
Die Länge des Laufbereichs für PEARL-Programme beträgt, abhängig von den benötigten Laufzeitfunktionen, im allgemeinen mehr als 32K.

4. Compiler

Der Compiler entspricht in seiner Bedienung dem anderer Compiler auf Siemens-300-Rechnern, ist also auch monitorfähig.

Während der Übersetzung entstehen wahlweise folgende Listen

- Quellprogramm-Liste mit zusätzlichen Angaben über Anweisungsnummer und Schachtelungstiefe
- Cross-Referenz-Liste
- Fehlerliste mit Fehlerklassifizierung
- Zwischensprachenausgabe für Demonstrations- und Testzwecke

Durch spezielle Compiler-Bedienung ist es möglich, entweder optimalen Laufzeitcode oder Code zur Anwendung der Testhilfen zu erzeugen. Der Compiler erzeugt Assemblercode (ASS 300), der entweder in einer Quellbibliothek abgelegt oder ausgedruckt werden kann.
Dadurch sind auch alle für den Assemblerprogrammierer vorhandenen Hilfsmittel verwendbar, beispielsweise Übersetzung für eine andere Anlage und anschließende Übertragung erzeugter Bibliothekselemente, Binde- und Lademechanismen. Das PEARL-Programmiersystem paßt sich so in bereits vorhandene Systemteile ein. Der auf Effizienz bedachte Programmierer kann im generierten Assemblercode die Realisierung verschiedener PEARL-Sprachelemente studieren und wird feststellen, daß fast optimaler und sehr effizienter Code erzeugt wird.

Es ist jedoch nicht erforderlich, daß man beim Testen und Verfolgen von Programmierfehlern diesen Assemblercode jemals zur Kenntnis nimmt.

Zusätzlich enthält der Compiler noch einen Preprocessor, der folgende Aufgaben erfüllen kann

- Ausgabesteuerung (Seitenvorschub, teilweise Protokoll-Unterdrückung)
- bedingte Übersetzung
- Einkopieren von Quellteilen (auch aus einer anderen Bibliothek)
- Testunterstützung

5. PEARL-Betriebssystem

Die Implementierung der Realzeitfunktionen ist weitgehend unabhängig von dem zugrundeliegenden (Gast-) Betriebssystem. Das wird erreicht durch Verwendung einer portablen Betriebssystem-Schnittstelle /3/ .

Die Anpassung an das jeweilige Betriebssystem muß dann nur noch an fünf genau definierten Punkten hergestellt werden:

- Reaktion auf Programmunterbrechungen
- Wechsel des Prozessors von einem Rechenprozeß zum anderen
- Ablesen der Uhr und Behandlung des Taktgebers
- Ein/Ausgabe von und zu den verschiedenen Peripheriegeräten
- Datenhaltung

Eine wesentliche Eigenschaft dieser Betriebssystem-Schnittstelle ist die Tatsache, daß alle PEARL-TASKs in einem einzigen Programm vom PEARL-System so im Multiplex betrieben werden, daß alle Spracheigenschaften des PEARL-TASKing genau erfüllt sind und die Zahl möglicher TASKs nicht durch noch freie Programmnummern im (Gast-) Betriebssystem beschränkt ist.

Das PEARL-System hat damit die parallel arbeitenden TASKs besser unter Kontrolle, als wenn sie unmittelbar unter Regie des Betriebssystems laufen würden. Besonders wichtig ist dies in irregulären Fällen wie TASK-Abbruch mit Betriebsmittelfreigabe oder bei Fehlerrückverfolgung.

6. Testunterstützung

Durch Optionen bei der Übersetzung kann die Codegenerierung so beeinflußt werden, daß ein Test auf Quellsprach-Ebene möglich wird. Die entsprechenden Testanweisungen können über Preprocessor-Befehle in der Quelle aktiviert werden.
Folgende Testmöglichkeiten sind gegeben

- Backtrace im Fall von Laufzeitfehlern
- Protokollierung der Aufrufe an das PEARL-Betriebssystem
- Ausgabe der Taskzustände
- Auftragsgesteuerter Zeilentrace (s.u.)
- vollständiger Zeilentrace

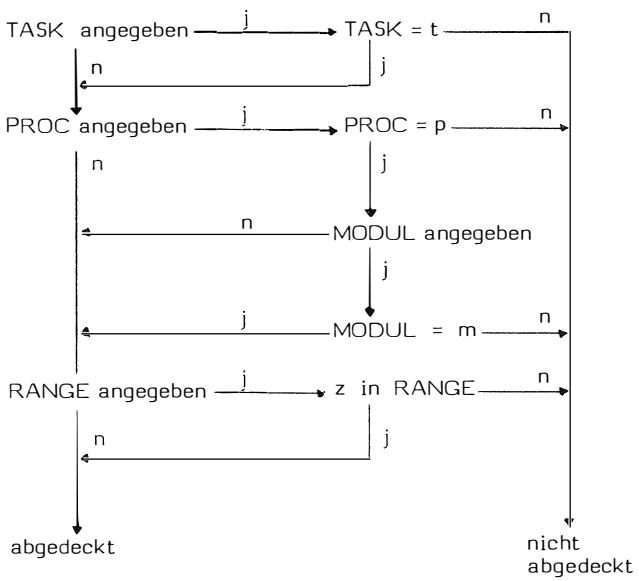
Die Ausgabe kann wahlweise auf einem Drucker oder Sichtgerät erscheinen.

Im Fall eines Auftragstrace werden nur diejenigen PEARL-Zeilen protokolliert, die durch wenigstens einen Auftrag abgedeckt sind.

Wann die aktuelle Zeile durch einen Auftrag abgedeckt wird, zeigt das nachfolgende Diagramm. Es bedeutet

t	aktueller Taskname, leer im Initvorlauf
p	aktuelle Procedure, leer wenn keine
m	Modul der Procedure p enthält
z	aktuelle Zeilennummer

TASK	Taskname	aus Auftrag
PROC	Procedurename	
MODUL	Modulname	
RANGE	Zeilenbereich	



Eine Tracezeile enthält folgende Information

- Name der Task, unter deren Kontrolle die Zeile durchlaufen wurde
- Zeilennummer
- Name der Prozedur
- Name des Moduls

Die Testschnittstelle ist dem Benutzer zugänglich. Er kann selbst weitere Testfunktionen implementieren.

Literaturhinweise

/1/ Allgemeine Informationen zur mbp-PEARL-Implementierung
PEARL-Rundschau, Band 2, Nr. 1, S. 22-23 (1981)

/2/ Anlage zum Projektbericht
Forschungsvorhaben P4.2/25: M-DVF/1
EBOSIPES

/3/ Echtzeit-Mehrprogramm-Betriebssystem
mbp-interne Veröffentlichung, Oktober 1980

/4/ mbp-PEARL
1 st edition, March 1981

Anschrift:

mbp Mathematischer Beratungs- und
Programmierungsdienst GmbH
Geschäftsstelle Lüneburg
Erbstorfer Landstraße 21
2120 Lüneburg