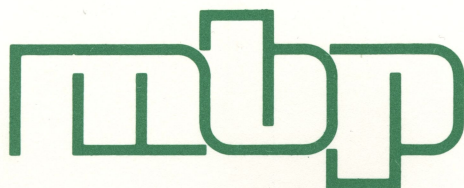

MBP-PEARL-COMPILER **COMPILER FÜR DIE** **PROZESSRECHNER-SPRACHE** **PEARL**

**(PROCESS AND EXPERIMENT
AUTOMATION REALTIME LANGUAGE)**



	Seite
PEARL: Die leistungsfähige Echtzeitprogrammiersprache	3
PEARL: Eine ausgereifte und verbreitete Programmiersprache	3
PEARL: Charakteristische Spracheigenschaften	4
mbp-PEARL: Sprachumfang	5
mbp-PEARL: Systemkomponenten	5
Struktur des mbp-PEARL-Compilers	6
mbp-PEARL-Compiler: Ein portables System	7
mbp-PEARL: Testunterstützung	7
mbp-PEARL: Prozeß-Simulation	9

PEARL: Die leistungsfähige Echtzeitprogrammiersprache

PEARL ist als höhere Programmiersprache dazu geeignet, anspruchsvolle und umfangreiche Prozeßsteuerungsaufgaben zu lösen.

Hohe Softwarekosten und große Risiken bei Prozeßrechnerprojekten erfordern strukturierte Programmierung in einer höheren Echtzeitprogrammiersprache als Alternative zur Assemblerprogrammierung. Zur Zeit ist PEARL die leistungsfähigste und modernste am Markt verfügbare und erprobte Echtzeitprogrammiersprache.

Von praxisorientierten Entwicklern ist PEARL in kurzer Zeit erlernbar und wirkungsvoll einzusetzen. PEARL ist, anders als etwa Prozeß-FORTRAN, eine Sprache aus einem Guß mit direkten Sprachelementen für beispielsweise Prozeß-Ein-/Ausgabe oder zeitliche Einplanungen.

Mit PEARL erstellte Programme sind mit diesen Spracheigenschaften verständlicher, leichter zu dokumentieren, zu ändern oder neuen Aufgabenstellungen anzupassen. Im Gegensatz zu Systemimplementierungssprachen werden keine Schnittstellen zum Prozeßrechner und seinem Betriebssystem offengelassen, sondern als Bestandteil von PEARL definiert. Mit einem solchen einheitlichen, genormten Programmiersystem kann die Softwareerstellung getrennt von der Hardware betrachtet werden. Die Definition von Schnittstellen wird erheblich erleichtert.

PEARL: Eine ausgereifte und verbreitete Programmiersprache

Die Einheitlichkeit der PEARL-Implementierungen ist durch Normung gesichert. Bereits seit 1978 liegt DIN 66 253 Teil 1 „Basic PEARL“ als Normentwurf vor und wurde im Juli 1980 zur Vornorm. Im August 1980 erschien der Normentwurf DIN 66 253 Teil 2 „Full PEARL“.

Es gibt eine PEARL-Benutzervereinigung. Zur Koordinierung der PEARL-Aktivitäten wurde der PEARL-Verein e. V. gegründet. Er ist zuständig für die Gewährleistung der Einheitlichkeit der Sprache und der Programmiersysteme, für das Sammeln und Austauschen von Anwendererfahrungen, für die Beratung von Herstellern und Anwendern und für die Erteilung von Auskünften an PEARL-Interessenten.

Die Geschäftsstelle des PEARL-Vereins befindet sich im VDI-Haus in Düsseldorf. Die Arbeit des PEARL-Vereins erfolgt in enger Zusammenarbeit mit der VDI/VDE-Gesellschaft für Meß- und Regelungstechnik und der Gesellschaft für Informatik in gemeinsamen Ausschüssen oder durch Veranstaltung von Tagungen. Die Mitglieder des PEARL-Vereins kommen aus vielen Bereichen der mit dem Einsatz von Computern befaßten Industrie und Forschung. Der PEARL-Verein gibt die Zeitschrift „PEARL-Rundschau“ heraus. Sie dient zur Information der Mitglieder und aller PEARL-Interessenten.

PEARL: Charakteristische Spracheigenschaften

PEARL bietet Spracheigen-
schaften, wie sie der Anwender
zur Lösung seiner Probleme
bei der Prozeßautomatisierung
erwartet.

Im Systemteil von PEARL läßt
sich eine Konfiguration, ins-
besondere die Prozeßperiphe-
rie, durch frei wählbare Be-
zeichner rechnerunabhängig
beschreiben. Damit erhöhen
sich der Dokumentationswert
und die Portabilität von PEARL-
Programmen entscheidend.
PEARL bietet dem Ingenieur
die für Prozeßanwendungen
nötigen Ein-/Ausgabeeinwei-
sungen neben den standardi-
sierten Zugriffen auf Peripherie-
geräte und Hintergrundspei-
cher.

Die Sprachelemente zur Be-
schreibung algorithmischer
Zusammenhänge und Abläufe
entsprechen dem Standard
moderner Programmier-
sprachen. Das Typkonzept von
PEARL bietet die Möglichkeit,
auch anwendungsbezogene,
zusammengesetzte Daten-
typen und neue Operatoren
zu definieren. Diese Form der
abstrakten Datentypen erlaubt
Prüfungen zur Übersetzungs-
zeit, effiziente Programmierung
und Modularisierung der Pro-
gramme.

Zur Formulierung der parallelen
Arbeit mehrerer Rechenpro-
zesse, ihrer Abhängigkeit von-
einander und ihrer zeitlichen
Abläufe besitzt PEARL leicht
anwendbare und einfach
erlernbare Sprachelemente.

Der Aufbau eines PEARL-Pro-
gramms aus getrennt übersetz-
baren Moduln mit genau
definierten Schnittstellen er-
leichtert die Kommunikation
zwischen Mitgliedern eines
Projektteams. Auch der schritt-
weise Aufbau komplexer Pro-
grammsysteme wird dadurch
wesentlich erleichtert.

1	1	/#SA01
2	1	MODULE)
3	1	SYSTEM) /* SYTEMENTEIL (DAS IST EIN KOMMENTAR) */
4	1	SA) DSSA(0)
5	1	EINGRIFF:INT*3)
6	1	PROBLEM)
7	1	RELAIS: TASK)
8	2	PUT 'RYRY' TO SA BY LIST)
9	2	END)
10	1	

Beispiel 1

11	1	/* FORTSETZUNG DES BEISPIELS AUS ZEILE 10. */
12	1	SPC EINGRIFF INTERRUPT)
13	1	STRRTASK:TASK)
14	2	WHEN EINGRIFF AFTER 5 SEC ALL 7 SEC
15	2	DURING 106 MIN ACTIVATE RELAIS
16	2	PRIORITY 5)
17	2	END)
18	1	MODEND)
19	1	/#

Beispiel 2

mbp-PEARL: Sprachumfang

Für eine anwendungsfreundliche und dennoch effiziente Programmierung reicht der durch die DIN-Norm 66 253 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),
- Datentypen (TYPE),
- user defined Operatoren mit Angabe von user defined PRECEDENCE (OPERATOR),
- Strukturen (STRUCTURE),
 - wobei Subkomponenten Felder oder Strukturen sein dürfen,
 - Subkomponenten verschiedener Strukturen gleiche Namen haben können und
 - Strukturen auch Resultat von Funktionen sein dürfen,
- statische Initialisierung von Feldern mit INITIAL,
 - variable Sub-Strings von Typ BIT und CHARACTER,
 - programmierte INTERRUPT-Simulation mit TRIGGER,
 - BOLT-Variable,
 - Standardfunktionen für Uhrzeit und Datum,
 - Prozeduren auch auf TASK-Ebene und
 - REENTRANT-GLOBAL-Prozeduren auf MODUL-Ebene.

Dieser Sprachumfang liegt dem 1978 erschienenen Sprachreport „PEARL Process and Experiment Real-Time Language“ aus dem Programm „Angewandte Informatik“ des Vieweg-Verlags zugrunde.

mbp-PEARL: Systemkomponenten

Der Übersetzer besteht aus einem weitgehend rechnerunabhängigen Oberteil, der einen PEARL-Modul in eine Standard-Zwischensprache übersetzt, und rechnerabhängigen Codegenerierungen, die die Zielcode-Form des PEARL-Moduls erzeugen.

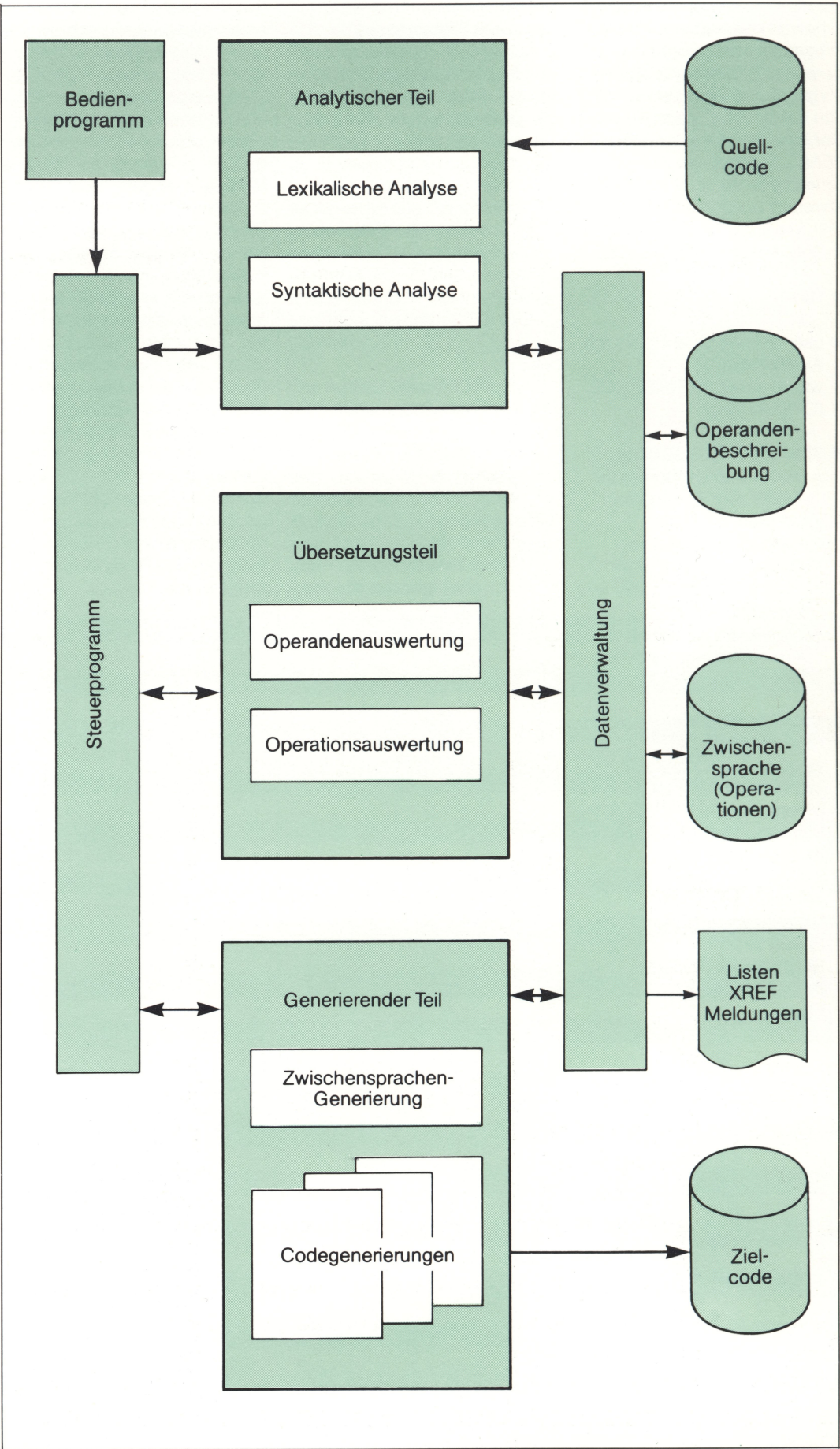
Die Übersetzung beginnt mit der lexikalischen Analyse, die die lexikalischen Einheiten aus den Symbolen eines PEARL-Programms erzeugt und eine Liste der verwendeten Bezeichner anlegt. Die Folge lexikalischer Einheiten wird in der syntaktischen Analyse überprüft, und es werden entsprechende Semantikfunktionen zugefügt. Die syntaktische Analyse wird durch eine Tabelle gesteuert, die automatisch aus einer metasprachlichen Beschreibung der PEARL-Syntax erzeugt wurde.

Angaben zum SYSTEM-Teil eines PEARL-Moduls werden anschließend anhand einer Tabelle ausgewertet, die die Konfiguration der Zielanlage beschreibt. Die Operandenauswertung prüft die Deklarationen (Operanden) auf Konsistenz und Korrektheit. Das PEARL-Programm erscheint dann als eine Liste von Konstanten, eine Liste von INITIAL-Werten, den Operandenbeschreibungen und der Folge der Anweisungen. Die Anweisungsfolge wird unter Verwendung der Operandenbeschreibung und der INITIAL-Liste überprüft (Operationsauswertung) und dann in die Zwischensprache umgesetzt.

Die Folge der Anweisungen in der Zwischensprache wird von der Codegenerierung in Objektcode im entsprechenden Modulformat umgesetzt. Eine weitere Übersetzungsphase protokolliert das Quellprogramm und gibt optional Zwischensprachendarstellungen und eine Querverweisliste aus.

Struktur des mbp-PEARL-Compilers

Der Aufruf der einzelnen Phasen geschieht über das Steuerprogramm, wobei der Steuerungsablauf vom Benutzer her über das Bedienprogramm beeinflusst werden kann. Zum PEARL-Programmiersystem gehört zudem eine Laufzeitprogramm-bibliothek und eine Betriebssystem-schnittstelle. Mit der Bibliothek der rund 100 PEARL-Laufzeitprogramme können die übersetzten Programme vom standardmäßigen Binder der Zielanlage so gebunden werden, daß sie auf der PEARL-Betriebssystemschnittstelle ablauffähig sind, die das jeweilige Prozeß-Betriebssystem zu einem PEARL-Betriebssystem erweitert. Der Übersetzer ist so aufgeteilt, daß eine Übersetzung bereits auf Anlagen mit nur 32k 16-Bit-Wörtern möglich ist.



mbp-PEARL-Compiler: Ein portables System

mbp-PEARL-Implementierungen existieren bereits auf einer Reihe von Anlagen sowohl als Cross-Compiler- wie auch als Zielanlagen-Versionen.

Als Zielrechner sind erreichbar:

- SIEMENS 330
- SIEMENS 300/R-Serie
- SIEMENS 404/3
- Hewlett Packard HP 3000

Als Übersetzungsrechner wurden bisher verwendet:

- SIEMENS 330 (Entwicklungsrechner)

- SIEMENS 404/3-Serie
- SIEMENS 404/3 mit PBS
- Hewlett Packard HP 3000
- SIEMENS-BS 2000-Anlagen

Weitere Cross-PEARL-Programmiersysteme erfordern:

- Die Erstellung eines Codegenerators für den Zielrechner
- Die Übertragung und Anpassung der portablen Komponenten der PEARL-Betriebssystemschnittstelle und des Laufzeitsystems.

mbp-PEARL: Testunterstützung

mbp-PEARL-Übersetzer erzeugen optional im Objektcode Lokalisierungsinformationen, aus denen die PEARL-Betriebssystemschnittstelle für jede Anweisung folgende Informationen erhält:

- Name des PEARL-Moduls,
- Name der aktuellen Task,
- soweit definiert, Name der Prozedur, in der sich die Anweisung befindet,
- Zeilennummer im Quellprogramm des PEARL-Moduls.

Die PEARL-Betriebssystemschnittstelle kann mit dieser Information optional TRACE-Ausdrucke erzeugen, wobei der Bediener des PEARL-Systems diese Ausdrucke jederzeit durch Bedienmaßnahmen ein- und ausschalten und auf bestimmte Prozeduren oder Tasks beschränken kann. Im folgenden Beispiel wurden die TRACE-Ausdrucke auf die Prozedur Y9PTLI bezogen. Nachdem das Programm Zeile 49 der Prozedur Y9PTLI (Bestandteil des MODULS Y1PTLI) unter der Regie der TASK STRTTA erreicht hatte, machte es eine Ausgabe und setzte seinen Lauf mit Zeile 50 fort.

MBP-PEARL/3 1 WE. 4 02 81 29H03				PAGE: 2	
PEARL - OPERATING - SYSTEM					
TASK "STRTTA" LINE: 45 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 47 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 48 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 52 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 53 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 54 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 55 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 56 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 57 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 59 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 99 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 100 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 47 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 48 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 49 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 50 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 103 Y9PTLI (Y1PTLI)					
TASK "STRTTA" LINE: 11 OPERATINGSYSTEM-SERVICE PRB CALLED.					
TASK "CONTRD" PRI0= 1 , ACTIVE.					
TASK "STRTTA" LINE: 11 PRI0= 255 , ACTIVE.					

Beispiel 3

MBP-PEARL/3.1 WE.28.01.81 21H56			
PEARL - OPERATING - SYSTEM			
TASK "STRTTA" LINE: 45 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 47 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 48 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 52 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 53 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 54 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 55 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 56 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 57 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 59 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 99 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 100 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 47 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 48 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 49 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 50 Y9PTLI (Y1PTLI)			
TASK "STRTTA" LINE: 103 Y9PTLI (Y1PTLI)			

Beispiel 4

Auch wenn alle Ausdrücke abgeschaltet sind, läßt sich im Fehlerfall nachträglich ermitteln, welche Programmstellen zuletzt ausgeführt wurden.

Im obigen Beispiel werden protokolliert: TASK-Name, Zeilennummer im Quellprogramm, Prozedurname und Modulname in der Reihenfolge, wie die letzten Kontrollpunkte bei der PEARL-Betriebssystemschnittstelle registriert wurden.

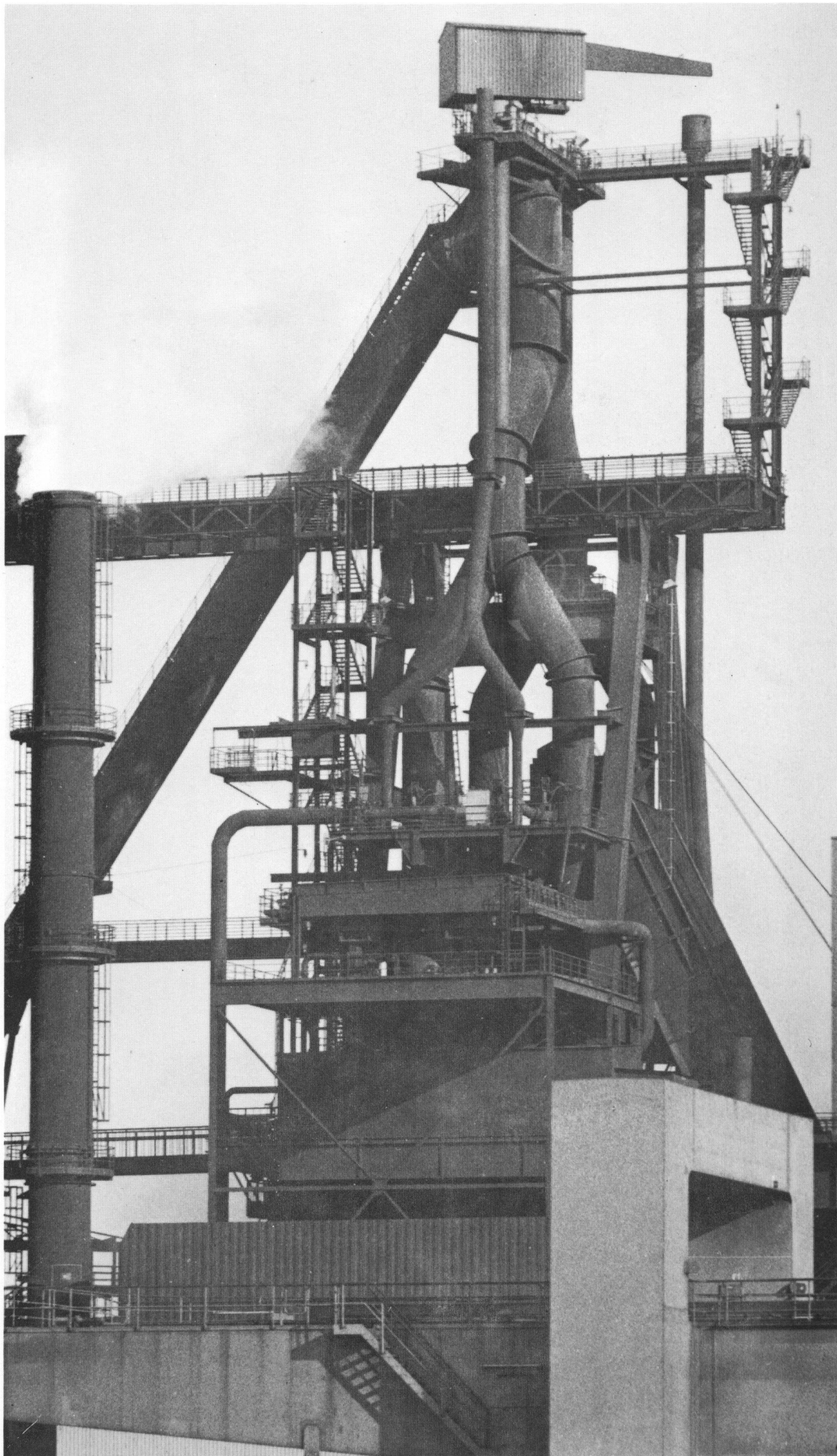
Änderungen in der Parallelarbeit mehrerer Rechenprozesse durch das PEARL-Betriebssystem lassen sich auf Wunsch ebenfalls mit Bezug auf das ursprüngliche Quellprogramm protokollieren. Das folgende Beispiel zeigt, wie sich mehrere Tasks gegenseitig aktivieren, sich beenden und wie die PEARL-Betriebssystemschnittstelle jeweils diejenige Task fortsetzt, die die günstigste Priorität hat.

MBP-PEARL/3.1 WE, 4 02.81 20H03									
PEARL - O P E R A T I N G - S Y S T E M									
TASK "TASK36" LINE: 11 OPERATINGSYSTEM-SERVICE PRB CALLED.									
TASK "TASK36" LINE: 11 PRI0= 41 , ACTIVE.									
TASK "TASK37" PRI0= 42 , ACTIVE.									
TASK "TERM " LINE: 8 PRI0= 109 , ACTIVE.									
TASK "STRTTA" LINE: 5 PRI0= 201 , ACTIVE.									
TASK "TASK36" LINE: 12 OPERATINGSYSTEM-SERVICE PRE CALLED.									
***** SIGNAL: 13 *****									
TASK "TASK36" LINE: 12 PRI0= 41 , INACTIVE.									
TASK "TASK37" PRI0= 42 , ACTIVE.									
TASK "TERM " LINE: 8 PRI0= 109 , ACTIVE.									
TASK "STRTTA" LINE: 5 PRI0= 201 , ACTIVE.									
TASK "TASK37" LINE: 14 OPERATINGSYSTEM-SERVICE WTDUR CALLED.									
TASK "TASK36" LINE: 12 PRI0= 41 , INACTIVE.									
TASK "TASK37" LINE: 14 PRI0= 42 , ACTIVE, WAITING DURATION.									
TASK "TERM " LINE: 8 PRI0= 109 , ACTIVE.									
TASK "STRTTA" LINE: 5 PRI0= 201 , ACTIVE.									
TASK "TERM " LINE: 9 OPERATINGSYSTEM-SERVICE PRE CALLED.									
***** SIGNAL: 13 *****									
TASK "TASK36" LINE: 12 PRI0= 41 , INACTIVE.									
TASK "TASK37" LINE: 14 PRI0= 42 , ACTIVE, WAITING DURATION.									
TASK "TERM " LINE: 9 PRI0= 109 , INACTIVE.									
TASK "STRTTA" LINE: 5 PRI0= 201 , ACTIVE.									
TASK "STRTTA" LINE: 6 OPERATINGSYSTEM-SERVICE PRE CALLED.									
***** SIGNAL: 13 *****									
TASK "TASK36" LINE: 12 PRI0= 41 , INACTIVE.									
TASK "TASK37" LINE: 14 PRI0= 42 , ACTIVE, WAITING DURATION.									
TASK "TERM " LINE: 9 PRI0= 109 , INACTIVE.									
TASK "STRTTA" LINE: 6 PRI0= 201 , INACTIVE.									
*** SYSTEM IS IDLING ***									
TASK "TASK36" LINE: 12 PRI0= 41 , INACTIVE.									

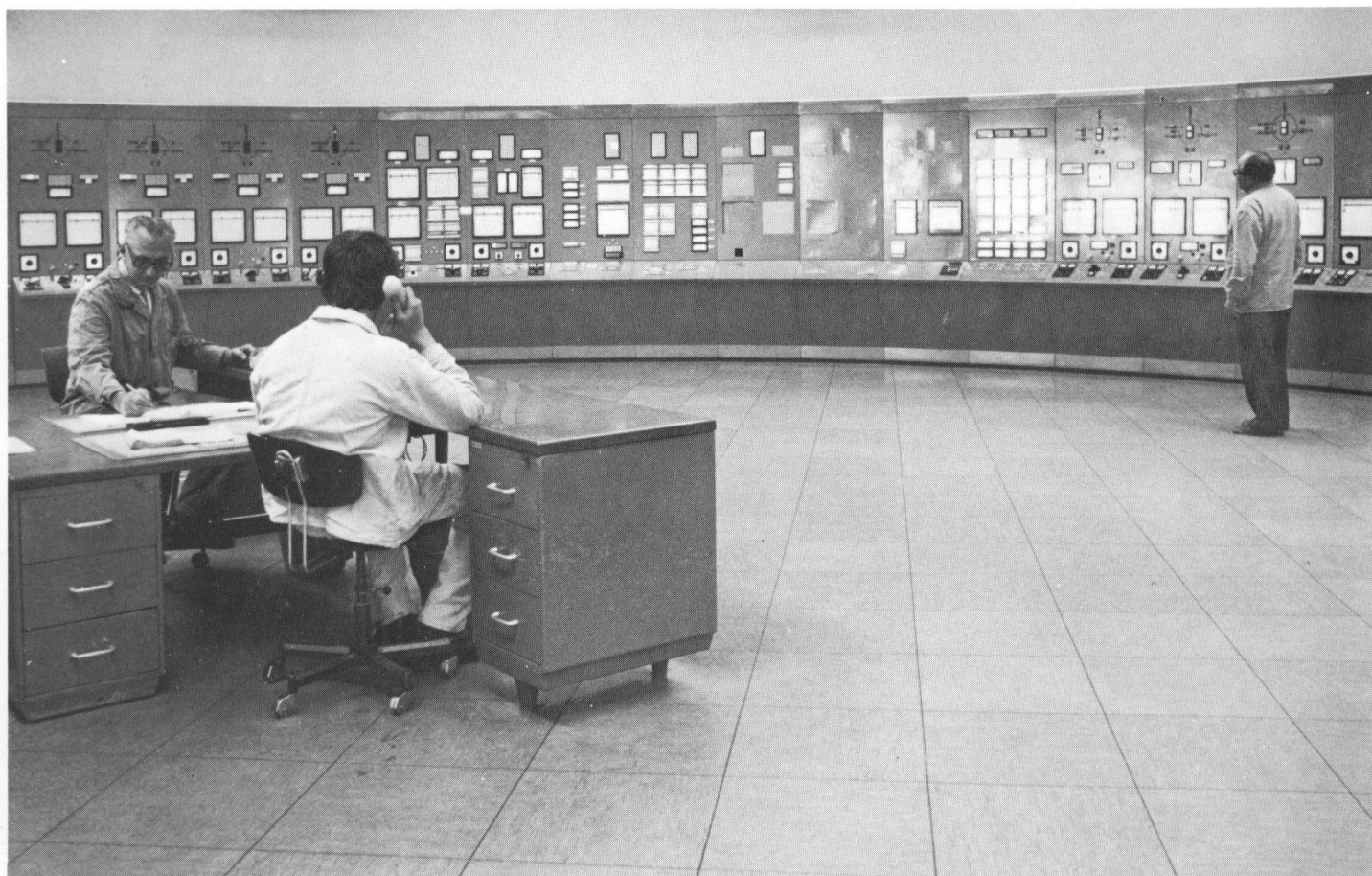
Beispiel 5

mbp-PEARL: Prozeß-Simulation

Die erwähnten Testhilfen können zeitkritische Anwendungen verfälschen oder aber bei sehr lang dauernden Prozessen extreme Wartezeiten erfordern. Um das zu vermeiden, gibt es eine geänderte Betriebssystemschnittstelle, die sich nur an den Testausdrucken orientiert und ansonsten mit simulierter, bei Bedarf geraffter oder gedehnter Zeit arbeitet.









Mathematischer Beratungs-
und Programmierungsdienst
GmbH
Semerteichstraße 47
4600 Dortmund 1
Telefon (02 31) 43 48-0
Telex 8 227 198 mbp d

Geschäftsstellen:

Ostpreußendamm 90-92
1000 Berlin 45
Telefon (0 30) 7 12 20 40

Am Wall 140
2800 Bremen
Telefon (04 21) 31 08 24
Telex 2 46 183 mbp bd

Lichtstraße 8
4300 Essen
Telefon (02 01) 23 37 49/23 37 40

Steinlestraße 6
6000 Frankfurt/Main 70
Telefon (06 11) 63 30 44
Telex 4 189 357 mbp fd

Aachener Straße 455
5000 Köln 41
Telefon (02 21) 40 90 32

Erbstorfer Landstraße 21
2120 Lüneburg
Telefon (041 31) 5 87 71/5 87 72

St.-Paul-Straße 9
8000 München 2
Telefon (0 89) 53 93 05
Telex 5 213 234 mbp md

Ostendstraße 159
8500 Nürnberg
Telefon (09 11) 57 29 57