The Portable PEARL Programming System of WERUM

Dr. Hans Windauer

1. Implemented Language Features

The implemented subset contains Basic PEARL (DIN 66 253, Part 1) and in addition the following language features of Full PEARL (DIN 66 253, Part 2).

Data Types

- **REF**
- User defined types (**TYPE**)
- **BOLT**
- Arrays with elements of type **SEMA, BOLT, REF,** user defined **DATION, STRUCT,** user defined type (**TYPE**)
- Arrays with more than 3 dimensions
- Arrays with lower bounds < 1
- Structures with components of type array, **STRUCT, REF,** user defined type (**TYPE**)
- B2 bit strings.

Declarations, Specifications, Definitions

- Modules may be identified (e.g. **MODULE** (TEST))
- Global attribute with module identifier (e.g. ... **GLOBAL** (TEST))
- Definition of new data types (**TYPE**)
- Declaration of new operators (**OPERATOR**) with precedences (**PRECEDENCE**)
- Declarations and specifications may be made in arbitrary sequence
- Local procedures, i.e. declaration of procedures also within tasks, procedures, blocks and loops
- Objects of type **SEMA, BOLT, IRPT, SIGNAL, REF** and user defined type (**TYPE**) may be parameters of procedures
- Objects of type **REF, STRUCT** and user defined type (**TYPE**) may be results of function procedures
- Long forms of **INIT** and **IDENT : INITIAL , IDENTICAL .**

Statements

- Values of reference variables and character string slices at the left side of assignments (e.g. STRING.**CHAR** (J) := 'N' ;)
- The schedule of an activate statement may be combined with a frequence and/or **AFTER** duration (e.g. **WHEN** interrupt **AFTER** duration **ALL** duration **DURING** duration **ACTIVATE** task ;)
- **SUSPEND** for other tasks
- **CONTINUE** with priority change
- Lists of **SEMA** variables after **REQUEST** and **RELEASE**
- **BOLT** statements **ENTER, LEAVE, RESERVE, FREE**
- Lists of **BOLT** variables in bolt statements
- **TRIGGER** statement.

Expressions

- Slices of character strings, variable slices of strings (e.g. X := INPUT.**BIT** (I : I + 3); OUTPUT.**CHAR** (J) := STRING.**CHAR** (K) ;)
- Dereferenciation (**CONT**)
- Conditional expression (e.g. A := **IF** B < 1 **THEN** B **ELSE** C **FIN** ;)
- Monadic operators LWB and UPB.

Input / Output

- **STRUCT,** user defined type (**TYPE**) and **ALL** may be transfer item type
- Arrays of user defined data stations
- Open parameter CAN, PRM
- Close parameter CAN, PRM.

System Division

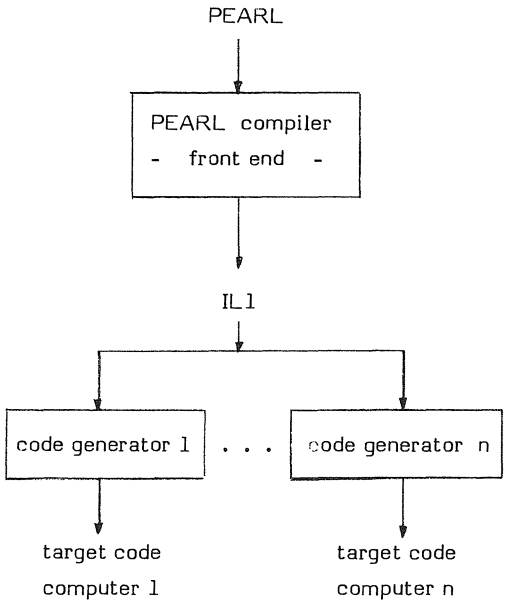
- Arbitrary sequence of connections
- Inverse notation of connections

- Identifier and / step possible after * in connection points.

The subset characterized here is implemented completely in the (portable) compiler of WERUM. There can be restrictions in the various implementations of the run time system on some target computers by reasons of size. E.g., signal handling and file handling are restricted on Siemens 404/3 (64 KB).

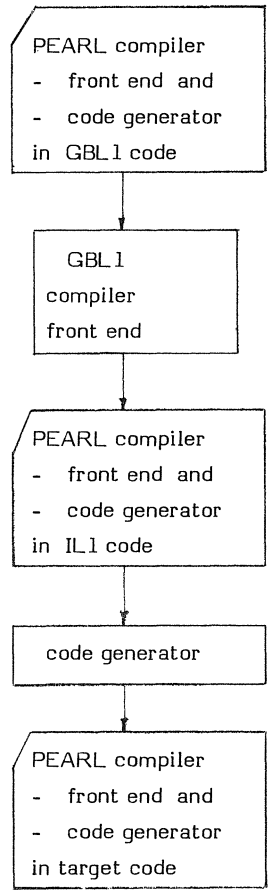
2. Characterization of the Compiler Technology

The **portable** PEARL compiler of WERUM consists of an analytic part ("front end") translating PEARL programs to the computer independent intermediate language IL1, and a code generating part ("code generator") transforming PEARL programs from their IL1 representation to target code (normally assembler or BRF, i.e. binary relocatable format). The front end is computer independent and therefore programmed only once; the computer dependent code generators have to be developed for any type of computer where PEARL programs are to be executed.



The front end and the code generators are programmed in GBL1, a proper subset of PL/I. GBL1 programs can also be translated to IL1 by a front end GBL1→ IL1. By means of this front end and the corresponding code generator the PEARL compiler front end and the code generator itself are translated to the target code (assembler or BRF) of the target computer.

This compilation is normally performed on one of the production computers of WERUM where the GBL1 compiler is implemented (Siemens 330 and NORD 10 S).



Therefore, the PEARL compiler can be implemented on target computers not having a PL/I compiler.

Of course PEARL specific run time routines and operating system functions have to be implemented too on the target computer in order to execute PEARL programs there.

The PEARL compiler can be also used for **cross compilation :** by reason of the characterized compiler technology it can be installed on every computer having a code generator or a PL/I compiler able to translate GBL1 programs, e.g. IBM or Siemens 7.760.

In addition the PEARL compiler can be transported to FORTRAN or PASCAL computers via a transformer from IL1 to ANSI-FORTRAN or PASCAL in order to work for this FORTRAN or PASCAL computer or to be used there as cross compiler for other target computers.

3. Existing Components of the PEARL Programming System

The portable PEARL programming system of WERUM consists of the following components:

- Compiler (front end and code generator)
- Kernel of the PEARL operating system
- Run time package for binary and process I/O
- Run time package for formatted I/O
- Symbolic debug system
- Real time data base system.

Up to now the run time routines for PEARL specific arithmetics, operators for bit and character strings, procedure organisation, array handling etc. were implemented computer dependent.

In case of HP 1000 a portable PEARL specific linker was implemented by the Technical University of Berlin in order to check the interfaces between the modules of the PEARL program.

Besides this, standard components of the target computer are used.

3.1 Compiler

The PEARL compiler translates PEARL programs to assembler or BRF. Because of its modular structure it only needs a segment of 50 KB to run; therefore it can operate on small computers. In spite of this it is able to translate "arbitrary" big programs.

The handling of the various compiler parameters corresponds to the handling of the other compilers of this computer.

Beside other functions, these parameter can be used to produce listings of the source program and of the translation result. In the assembler listing or BRF listing references to the corresponding source lines are included. In addition the compiler produces a cross reference list of all objects of the program showing their source lines of definition and use.
By compiler parameter index checking and reference checking may be switched on or off.

The compiler analyses programs thoroughly and exactly. The error messages consist of a text together with a reference to the source line causing the error.

A preprocessor allows to include program pieces from text files (%INCLUDE) and to compile conditionally (%IF).

The evaluation of the system parts of PEARL programs is driven by a so-called configuration list describing all configuration possibilities of the target computer. If these possibilities are to be extended, e.g. when adding a new peripheral device, this configuration can be adapted easily by the user himself. The compiler reads the configuration dynamically for any compilation; this is necessary in case of several cross compilations for different target configurations.

On request an optimizing version of the compiler is available performing the following optimizations when setting the corresponding parameter:

- Addresses of components of structures and referenced objects are kept as long as possible in order to avoid more than one address calculation for identical objects.
- Common sub-expressions are calculated only once.
- No index calculation at run time for fixed array indices.

The compiler is programmed in GBL 1, a proper PL/I subset.

3.2 Kernel of the Operating System

WERUM has developed a portable kernel of a PEARL operating system, called BAPAS-K, for the PEARL specific organisation and execution of tasks, their synchronisation and process I/O. BAPAS-K is programmed computer independently in GBL 1; therefore this kernel can be transported automatically to the target computer where its open interfaces are closed by hand.

BAPAS-K can be added to an existing host operating system; it can also operate without any host operating system.

3.3 Run Time Package for Binary I/O

The run time package BAPAS-FILE contains all run time routines necessary for the PEARL specific organisation of files and execution of READ and WRITE statements. The interface of these computer independent, portable routines to the target computer consists of suitable control blocks and driver calls.

BAPAS-FILE is programmed in PEARL; therefore it can be transported automatically by the PEARL compiler to target computers.

3.4 Run Time Package for Formatted I/O

Analogously to BAPAS-FILE the run time package BAPAS-FORMEA contains all computer independent run time routines necessary to execute PUT and GET statements according to the PEARL semantics.

BAPAS-FORMEA is programmed in PEARL; therefore it can be transported automatically by the PEARL compiler to target computers.

3.5 Symbolic Debug System

The symbolic debug system allows to test interactively PEARL programs by use of PEARL like commands on host and target computers. It is programmed portable in GBL 1 and PEARL.

Version 1 for Small Target Computers

The first version offers the following possibilities on PEARL level:

- Line trace
- Breakpoints at lines
- Display of values of variables.

This version is already implemented on HP 3000, NORD 10/100 and Siemens 330.

Version 2 for Medium Target and Host Computers

The second version offers the following possibilities on PEARL level:

- Line, label and call trace
- Breakpoints at
  - Lines and labels
  - Entries and exits of tasks and procedures
- Display and change of values of variables
- Display and change of states of tasks, semaphores and bolts.

This version can be installed on host target computers with 128 KB and more. When being installed on a host computer the PEARL operating system of the target computer and the time scale are simulated (by means of BAPAS-K) in order to handle tasks in the right sequence.

Version 2 is already implemented on HP 3000, NORD 10/100 and Siemens 330.

Version 3 for Host Computers

The third version has been developed for host computers. In addition to version 2 it contains the following aids:

- Simulation of the run time behaviour of the target computer on statement level
- Simulation of the I/O of the target computer by means of
  - Dialogue with the user
  - Anti tasks
  - Files with test data
- Breakpoints at
  - Time events (analog to PEARL schedules)
  - Interrupts
  - I/O statements
- Deadlock analysis
- Interrupt statements.

Version 3 is already implemented on HP 3000, NORD 10 and Siemens 330.

3.6 Open Real Time Data Base System

In order to support the use of PEARL in automation systems with data base oriented problems WERUM has developed the **open real time** data base system BAPAS-DB allowing PEARL tasks and users (via terminal) to access common data concurrently. Important features of BAPAS-DB are:

- Interactive Data Description Language DDL for the data base administrator.
- Interactive Query Language QL for users.
- Data Manipulation Language DML to access the data base in PEARL tasks independently of the chosen access strategies.
- Concurrent access by users and PEARL tasks with implicit synchronisation on record level.
- Different data sets may be accessed by different access strategies.
- Access strategies can be exchanged or added without changing the interfaces to DDL, QL and DML. (The system is open.)

By reasons of these properties BAPAS-DB can be used very flexible in automizing technical processes or it can be adapted to meet special requirements in parallel to the production of the application software.

DDL, QL and DML offer the following functions:

Data Description Language DDL

- Data Base Level
  - Creation and deletion of data bases
  - Definition, modification and deletion of access rights
  - Definition, modification and deletion of administration data
- Data Set Level
  - Creation and deletion of data sets
  - Definition of the structure of the records of a data set
  - Definition, modification and deletion of access rights
- Access Strategies
  - Introduction of new access strategies
  - Attaching access strategies to data sets.

Query Language QL

- Searching records satisfying given conditions which can be complex logical combinations of all components of the records.

In this sense BAPAS-DB is a **relational** data base system.

- Output of found records to terminal, printer or data sets.
- Update of records.
- Deletion of records.
- Insertion of new records.

Data Manipulation Language DML

- Specification of data sets of the data base.
- Searching records satisfying given conditions which can be complex logical combinations of all components of the records.
- Use and update of found records.
- Deletion of found records.
- Insertion of new records.

By standard, BAPAS-DB contains access strategies for sequential (LIFO, FIFO) and direct access (Hash, B*-Tree) together with functions for recovery and chaining data sets. It is programmed portable in GBL1 and PEARL.

Installations have been made on NORD 10, Siemens 330 and Siemens R 30.

The development of BAPAS-DB has been sponsored by the German Ministry for Research and Development within the projects PDV/PFT of Kernforschungszentrum Karlsruhe GmbH.

4. Computers where PEARL Programs Can Be Translated

The PEARL compiler is implemented on the following computers:

- Amdahl 470/6
- Hewlett-Packard HP 1000
- Hewlett-Packard HP 3000
- Norsk Data NORD 10 S and NORD 100
- Siemens 310 and 330
- Siemens R 30
- Siemens 7.760
- Siemens 404/3.

The implementation for

- Intel 8086

is in preparation. The compiler can be transported to PL/I, FORTRAN and PASCAL computers without producing a new code generator.

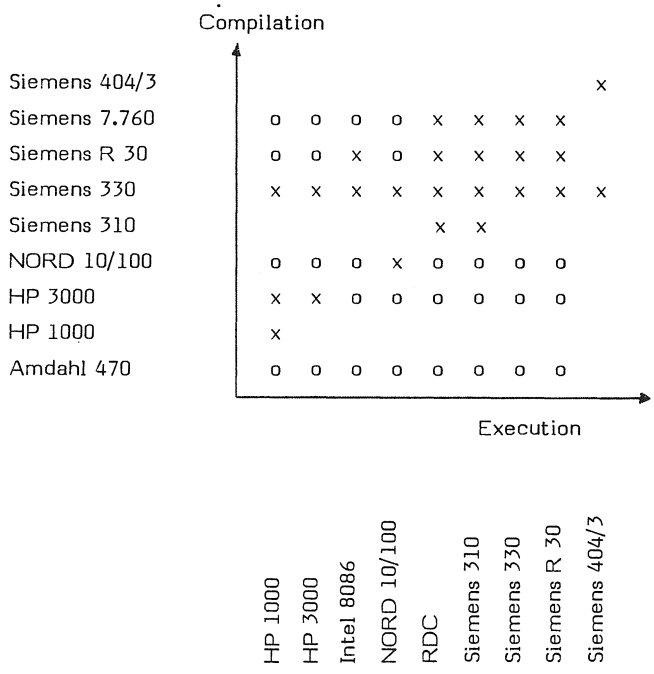
Number of installations: more than 25.

5. Computers where PEARL Programs Can Be Executed

The compiled PEARL programs can be executed on the following computers:

- Hewlett-Packard HP 1000
- Hewlett-Packard HP 3000
- Intel 8086
- Norsk Data NORD 10 S and NORD 100
- RDC (Really Distributed Computer Control System of the Fraunhofer Institute IITB, Karlsruhe)
- Siemens 310
- Siemens 330
- Siemens R 30
- Siemens 404/3

The following table shows the variant possibilities of installations. In this table, x means that this version is installed, and o means that this version can be installed in short time.



6. Conditions

6.1 Form of Delivery

The programming system can be delivered partially or at whole on magnetic disk, tape or floppy disk in the form generated by the code generator, FORTRAN, PASCAL or PL/I compiler. Sources and technical documentation can be delivered on request.

6.2 Training, Documentation

The implemented PEARL subset is described in

PEARL Language Reference Manual
Reg. FB 141/8008. WERUM, Lueneburg.

This manual is also published as book:

Wulf Werum, Hans Windauer

Introduction to PEARL
Process and Experiment Automation
Realtime Language
Description with Examples

Braunschweig: Vieweg 1982.XI, 183 S.
ISBN 3-528-03590-0.

General or computer dependent user manuals and training courses are available on request.

6.3 Guarantee, Maintenance

The guarantee time is one year after acceptance.

Afterwards a maintenance contract is offered containing fast removing of errors and delivery of releases.

7. References

Amdahl 470/6 : GRS, Garching: Safety related analysis.

HP 1000 : Technical University of Berlin: Process and experiment automation.

HP 3000 : IRT, Munich: Development of software for broadcasting / television purposes.

Intel 8086 : IITB, Karlsruhe: Process control.

NORD 100 : Halden Reactor Project, Halden, Norway: Disturbance analysis system, reliability investigations.

Technical University of Braunschweig: Control of a data base machine.

NORD 10 S : WERUM, Lueneburg: Development of systems software.

RDC : IITB, Karlsruhe: Process control, programming of industrial robot systems.

Thyssen AG, Duisburg: Process control in steel factories.

Siemens 310 : IITB, Karlsruhe: Development of process control software.

Siemens 330 : IITB, Karlsruhe: Cross compilation for RDC.

WERUM, Lueneburg: Development of systems software and process control software.

Siemens R 30 : IITB, Karlsruhe: Development of process control software, implementation of a software engineering environment for process control engineers.

Siemens 404/3 : Erprobungsstelle 71 der Bundeswehr, Eckernförde.

DFVLR, Oberpfaffenhofen: Experiment automation.

Siemens 7.760 : IITB, Karlsruhe: Cross compilation for RDC.