Echtzeit

Nr. 3, September 2015

Mitteilungen
des GI/GMA/ITG-Fachausschusses
Echtzeitsysteme

GESELLSCHAFT FÜR INFORMATIK E.V.

VDI/VDE-Gesellschaft
Mess- und Automatisierungstechnik

ITG INFORMATIONSTECHNISCHE GESELLSCHAFT IM VDE

Impressum

Herausgeber	GI/GMA/ITG-Fachausschuss Echtzeitsysteme http://www.real-time.de
Sprecher	Prof. Dr. Dr. Wolfgang A. Halang FernUniversität in Hagen Lehrstuhl für Informationstechnik 58084 Hagen wolfgang.halang@fernuni-hagen.de
Stellvertreter	Prof. Dr. Dieter Zöbel Universität Koblenz-Landau Institut für Softwaretechnik 56016 Koblenz zoebel@uni-koblenz.de
Redaktion	Prof. Dr.-Ing. habil. Herwig Unger Dipl.-Ing. Jutta Düring FernUniversität in Hagen 58084 Hagen pearl@fernuni-hagen.de
ISSN	2199-9244

Redaktionell abgeschlossen am 21. September 2015

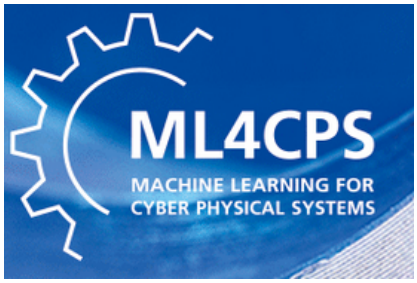
Einreichung von Beiträgen:
Alle Leserinnen und Leser sind aufgerufen, das Mitteilungsblatt auch zukünftig durch Beiträge mit zu gestalten, um den Informations- und Meinungsaustausch zwischen allen an den Fragen der Echtzeitprogrammierung Interessierten zu fördern.

In dieser Ausgabe:

1 Veranstaltungen mit Unterstützung des Fachausschusses

Jutta Düring, Fachausschuss Echtzeitsysteme

ML4CPS – Machine Learning for Cyber Physical Systems and Industry 4.0
1. und 2. Oktober 2015, CENTRUM INDUSTRIAL IT in Lemgo



Cyber-physische Systeme (CPS) zeichnen sich durch Anpassungs- und Lernfähigkeit aus: Sie analysieren ihre Umgebung und lernen auf Basis ihrer Beobachtungen Muster, Zusammenhänge und prognosefähige Modelle. Typische Anwendungen für solche Systeme sind das Condition Monitoring, Predictive Maintenance, Bildverarbeitung und Diagnose. Als Schlüsseltechnologie für die Entwicklung von CPS gilt das maschinelle Lernen.

Die erste Konferenz ML4CPS widmet sich nun genau diesem Themenfeld. Die zweitägige Veranstaltung bildet ein Forum, um neue Ansätze zum maschinellen Lernen für cyber-physische Systeme zu präsentieren, Erfahrungen auszutauschen, zu diskutieren und Visionen zu entwickeln. Dazu adressiert die Konferenz Forscher und Anwender aus unterschiedlichen Branchen wie der Produktionstechnik, Automatisierung, Automotive und Telekommunikation. Homepage: http://www.ml4cps.com/

KommA 2015 – 6. Jahreskolloquium in der Automation
17. und 18. November 2015 in Magdeburg

KommA ist ein Forum für Industrie und Wissenschaft zu allen Fragen rund um die industrielle Kommunikation. Veranstalter sind abwechselnd die beiden Institute inIT der Hochschule Ostwestfalen-Lippe in Lemgo und ifak e.V. in Magdeburg. Homepage: http://www.ifak.eu/Komma2015

2 Programm und Aufruf zur Teilnahme

2.1 Fachtagung Echtzeit 2015

Die Fachtagung „Echtzeit“ findet am 12. und 13. November 2015 wie gewohnt in Boppard am Rhein im Hotel Ebertor statt. Neu ist die Durchführung zusammen mit der GI-Fachgruppe Betriebssysteme (www.betriebssysteme.org). Das Leitthema lautet dementsprechend „**Betriebssysteme und Echtzeit**“.

Der Tagungsband erscheint in der Reihe „Informatik aktuell“ des renommierten Springer-Verlages. In Anbetracht des interessanten Tagungsprogramms und der anregend-freundlichen Atmosphäre sollten Sie sich unter http://www.real-time.de/workshop.html zur Teilnahme anmelden.

Erster Workshop-Tag: Donnerstag, der 12. November 2015

11:00 Uhr Treffen der Arbeitskreise

11:30 Uhr Mittagsimbiss (optional)

13:00 Uhr Begrüßung

13:15 Uhr **Sitzung 1: Testen von Echtzeitsystemen**

Testen von Echtzeiteigenschaften für verteilte Ablaufsteuerungen
Matthias Jurisch, Kai Beckmann (Hochschule RheinMain)

EMSBench: Benchmark und Testumgebung für reaktive Systeme
Florian Kluge, Theo Ungerer (Universität Augsburg)

14:15 Uhr Pause

15:00 Uhr **Sitzung 2: Systemarchitekturen**

Prozessorarchitektur zum Einsatz unter sicherheitsgerichteten Echtzeitbedingungen
Daniel Koß (FernUniversität in Hagen)

SAKKORO. Eine generische, echtzeitfähige Systemarchitektur für kognitive, kooperierende Roboter
Adrian Leu, Danijela Ristic-Durrant, Axel Gräser (Universität Bremen)

CPS-Remus: Eine Hochverfügbarkeitslösung für virtualisierte cyber-physische Anwendungen
Boguslaw Jablkowski, Olaf Spinczyk (Technische Universität Dortmund)

16:30 Uhr Pause

17:00 Uhr **Sitzung 3: Graduiertenwettbewerb**

Energieverbrauchsanalyse mittels implizierter Pfadaufzählung und genetischer Algorithmen
Peter Wägemann (Universität Erlangen-Nürnberg)

Testanwendungen zur Überprüfung des PEARL-Sprachsystems auf Sprachkonformität
Christian Ritzler (Hochschule Furtwangen)

Globale Kontrollflussanalyse von eingebetteten Echtzeitsystemen
Christian Dietrich, Martin Hoffmann, Daniel Lohmann (Universität Erlangen-Nürnberg)

18:30 Uhr Preisverleihung

18:45 Uhr Abendessen

20:15 Uhr Mitgliederversammlung des Fachausschusses Echtzeitsysteme

20:15 Uhr Fachgruppentreffen Betriebssysteme

Zweiter Workshop-Tag: Freitag, der 13. November 2015

9:00 Uhr **Sitzung 4: Entwurfsaspekte**

Ein hierarchisches Scheduling-Modell für unbekannte Anwendungen mit schwankenden Ressourcenanforderungen
Vladimir Nikolov, Franz J. Hauck, Lutz Schubert (Universität Ulm)

Wartefreie Synchronisation von Echtzeitprozessen mittels abgeschirmter Abschnitte
Gabor Drescher, Wolfgang Schröder-Preikschat (Universität Erlangen-Nürnberg)

dOSEK: Maßgeschneiderte Zuverlässigkeit
Martin Hoffmann, Florian Lukas, Christian Dietrich, Daniel Lohmann (Universität Erlangen-Nürnberg)

10:30 Uhr Pause

11:00 Uhr **Sitzung 5: Verteilte Systeme**

Fork-Join-Parallelismus in einer gemischt-kritischen Mehrprozessorumgebung
Marc Bommert (Hochschule RheinMain)

React in Time: Ereignisbasierter Entwurf zeitgesteuerter verteilter Systeme
Florian Franzmann, Tobias Klaus, Fabian Scheler et al. (Universität Erlangen-Nürnberg, Siemens AG)

Collaborative Resource Management for Multi-Core AUTOSAR OS
Renata Martins Gomes, Fabian Mauroner, Marcel Baunach (Technische Universität Graz)

12:30 Uhr Verabschiedung

12:45 Uhr Mittagsimbiss (optional)

2.2 Konferenz Autonome Systeme 2015

Die vom GI-Arbeitskreis Echtzeitkommunikation veranstaltete Konferenz „Autonome Systeme“ findet vom 25. bis 30. Oktober 2015 in Cala Millor auf Mallorca statt. Der Tagungsband erscheint in den „Fortschritt-Berichten“ des VDI-Verlages (Reihe 10, Band 842, ISBN 978-3-18-384210-0). Homepage: http://www.fernuni-hagen.de/kn/autsys/

Montag, der 26. Oktober 2015

08:30 Uhr **Keynote: Jörg Keller**
Extending Static Scheduling Algorithms for Moldable Tasks towards Dynamic Scheduling of Multiple Applications with Soft Real-time Properties

09:45 Uhr **Session 1: Systems**

A Cloud-based Building Automation System
Daniel Versick, Tom Jaster, Johannes Karow, Djamshid Tavangarian

Adaptive Instructional Videos
Ingolf Waßmann, Martin Müller, Djamshid Tavangarian

Design of a Consensus-oriented and Crowd-verifiable Control Unit
Marcel Schaible

11:15 Uhr Pause

11:30 Uhr **Session 2: Hardware and Architectures**

An Approach to Model-based Code Generation for a Specialized Soft Microprocessor
Elena Rozova, Bernd Däne

Architecture of Universal Routing Element for Networks on Chip
Andriy Osadchuk, Wolfgang Fengler

In-circuit Verification of Logic Controllers
Michal Doligalski, Jacek Tkacz

12:30 Uhr Mittagspause

14:00 Uhr **Tutorial „Challenges on Interactive Network Search", Mario Kubek**

18:30 Uhr Abendessen

Dienstag, der 27. Oktober 2015

Social Event: Tagesausflug

Mittwoch, der 28. Oktober 2015

09:00 Uhr **Session 3: Brain like Computing**

Automatic Classification of Serrated Patterns in Direct Immunofluorescence Images
Chenyu Shi, Joost M. Meijer, Jiapan Guo, George Azzopardi, Marcel F. Jonkman, Nicolai Petkov

Automatic Optic Disk Localization and Diameter Estimation in Retinal Fundus Images
Jiapan Guo, Shenyu Shi, Nomdo M. Jansonius, Nicolai Petkov

Towards Unsupervised Familiar Scene Recognition in Egocentric Videos
Estefania Talavera, Nicolai Petkov, Petia Radeva

10:30 Uhr Pause

11:00 Uhr **Session 4: Applications and Platforms**

CLAUDE: Cloud Computing for Non-interactive Long-running Computationally Intensive Scientific Applications
Andrei Lapin, Eryk Schiller, Peter Kropf

Automatic Code Generation - Some Observations
Theodor Tempelmeier

Social Media in Disaster Response - A Literature Review
Jens Schwarter, Ulrike Lechner

Robustness and Resilience in Transport Logistics - A Literature Review
Florian Maurer, Ulrike Lechner

12:30 Uhr Mittagspause

14:00 Uhr **Session 5: Data and Text Mining**

Why Google Isn't the Future. Really Not.
Robert Eberhardt, Mario M. Kubek, Herwig Unger

Empiric Studies of Word Use in Heterogeneous Text Corpora
Patrick Krause

Ein IR-Verfahren im P2P-Netz auf Basis dezentraler Indizierung
Dimitri Samorukov

Text Mining with Graph Databases: Traversal of Persisted Token-level Representations for Flexible On-demand Processing
Thomas Efer

The Role of Context Information for Email Analysis
Christine Klotz, Anthony M. Ocana

Natural Disaster Classification in Thai Tweets
Maleerat Sodanil

16:00 Uhr Pause

16:30 Uhr **Session 6: User Behaviour**

Human-Computer Interaction
Thippaya Chintakovid

Applying User Preferences and Co-purchasing Analysis for Item Recommendations
Sunantha Sodsee, Maytiyanin Komkhao

Mutual Influences between Trust and Online Social Network Systems
Son Doan Trung, Mario M. Kubek, Herwig Unger

Refining User Lifetime Model by Big Five Characteristics
Suwimon Vongsingthong, Sirapat Boonkrong, Herwig Unger

Emotional Data Collection and Classification
Son Hoang Huu

19:00 Uhr Abendessen

Donnerstag, der 29. Oktober 2015

09:00 Uhr **Session 7: Nonlinear Systems**

Cellular Neural Networks-based Emulation of Analog Filters: Theory and Design Principle
Nkiediel A. Akwir, Jean Ch. Chedjou, Kyandoghere Kyamakya

Cellular Neural Networks-based Emulation of Analog Filters: Selected Applications
Nkiediel A. Akwir, Jean Ch. Chedjou, Kyandoghere Kyamakya

Neurocomputing-based Matrix Inversion Involving Cellular Neural Networks: Black-box Training Concept
Ahmad Haj Mosa, Kyandoghere Kyamakya, Mouhannad Ali, Fadi Al Machot, Jean Ch. Chedjou

A Recurrent Neural-network-based Ultra-Fast, Robust and Scalable Solver of Linear Equation Systems with Potential Integration in a Speeding-up of FEM Solver Architectures
Vahid Tavakkoli, Jean Ch. Chedjou, Kyandoghere Kyamakya

10:30 Uhr Pause

11:00 Uhr **Tutorial „Simulations for Research Results Validation“, Part I**
Hauke Coltzau

12:30 Uhr Mittagspause

14:00 Uhr **Tutorial „Simulations for Research Results Validation“, Part II**
Hauke Coltzau

16:00 Uhr Pause

16:30 Uhr **Session 8: Analysis and Simulation**

A Model-based Approach on Superpopular Contents in Online Social Networks
Hauke Coltzau, Herwig Unger

A New Sneak Circuit Analysis Method for Zero-Voltage Switching Power Electronic Converters
Lili Qu

Simulation und Steuerung gekoppelter Systeme – Wasser und Energie in einer Megastadt
Gloria Robleto

Role-specific Node Selection via Activating Areas in MANETs
Adrian Dilo

18:30 Uhr Abendessen

3 Änderungen in der Sprachdefinition in OpenPEARL in Bezug auf PEARL90

Rainer Müller, Hochschule Furtwangen, mueller@hs-furtwangen.de
Marcel Schaible, FernUniversität in Hagen, marcel.schaible@fernuni-hagen.de

Projekt-Homepage: http://sourceforge.net/projects/openpearl/

3.1 Einleitung

Bei der Realisierung von OpenPEARL hat sich herausgestellt, dass die Sprachdefinition von PEARL90 [1] weder widerspruchsfrei noch eindeutig fomuliert ist. Dies liegt vermutlich daran, dass bezüglich dieser Sprachdefinition noch kein neues Übersetzungssystem realisiert wurde.

Widersprüche in einer Sprachnorm müssen auf jeden Fall beseitigt werden. Unklarheiten bezüglich des Verhaltens von Sprachelementen sind bei den meisten Sprachnormen normal. PEARL fordert allerdings, dass der komplette Problemteil unabhängig vom Zielsystem ist. Für Anwendungen im Bereich der Prozeß- und Automatisierungstechnik ist es essentiell notwendig, dass das Ablaufverhalten einer Anwendung eindeutig ist. Hieraus ergibt sich, dass gerade bei PEARL keine offenen Punkte in der Sprachdefinition enthalten sein dürfen.

In diesem Artikel werden die bislang für OpenPEARL gemachten Änderungen bezüglich der Sprachreferenz von PEARL90 [1] dargelegt. Die Änderungen in diesem Artikel sind sicherlich nicht vollständig. Bei der Diskussion der Änderungen wurde natürlich die Dokumentation von RTOS-UH [2] mit berücksichtigt.

3.2 Widersprüche

Der Umgang mit Prozeßgeräten ist widersprüchlich geregelt. Einerseits wird geregelt, dass nur über sogenannte Benutzerdatenstationen kommuniziert werden darf, andererseits sind die meisten Beispiele bei TAKE und SEND derart, dass direkt mit dem Systemgerätenamen gearbeitet wird.

Die Neuregelung sieht hier nun vor, dass

1. bei der Spezifikation eines Systemgerätes das neue Attribut `SYSTEM` anzugeben ist, damit Systemdatenstationen von globalen Benutzerdatenstationen eindeutig unterschieden werden können. Weiter ist der Transfertyp zu spezifizieren.

```
MODULE(new);
SYSTEM;
   conout : StdStream(1);   ! stdout Kanal
PROBLEM;
  SPC conout DATION  OUT SYSTEM ALL FORWARD DIM(*,80) GLOBAL;
  !                      ^^^^^^ ^^^   --  neu
```

2. die Spezifikation von Systemgeräten nun zusätzlich eine Angabe der zulässigen Transferdatentypen enthält, sodass der Compiler eine Typprüfung für die Werte in der TAKE und SEND- Anweisung durchführen kann.

```
MODULE(...);
SYSTEM;
   led1 : OctopusDigOut('A',5,1);  ! device for 1 digital output bit
PROBLEM;
  SPC led1 DATION  OUT SYSTEM BASIC BIT(1) GLOBAL;
  !                    ^^^^^^^      ^^^^^   -- neu
```

3. auch für TAKE und SEND eine Benutzerdatenstation anzulegen ist. Auch hier ist der Transfertyp wie gehabt anzugeben. Die Typverträglichkeit bezüglich des zugeordneten Systemgerätes wird spätestens bei Programmstart geprüft.

```
...;
  SPC led1 DATION  OUT SYSTEM BASIC BIT(1) GLOBAL;
  ...
  DCL running DATION OUT BASIC BIT(1) CREATED (led1);
  OPEN running;
  SEND '1'B1 TO running;
  CLOSE running;
```

3.3 Unklarheiten

3.3.1 Taskprioritäten

Die Sprachnorm legt fest, dass die Priorität 1 die höchste Taskpriorität festlegt. Für den Defaultwert wird 255 genannt. Es ist anzunehmen, dass dies die niedrigste Priorität ist, exakt spezifiziert ist dies leider nicht.

Die Neuregelung sieht vor, dass der Bereich der Prioritäten von 1 bis 255 reicht, wobei eine Implementierung nicht alle Werte bereitstellen muss. Im Falle der Benutzung einer nicht verfügbaren Taskpriorität wird ein definiertes SIGNAL erzeugt. Bei Benutzung einer Priorität außerhalb des Bereichs 1 bis 255 wird ebenfalls ein SIGNAL erzeugt.

3.3.2 Fehlerbehandlung

Die Sprachnorm von PEARL90 nennt die Möglichkeit, dass im Falle von Ablaufproblemen Fehler in Form von SIGNALs erzeugt werden können. Es ist weder klar formuliert, ob dies zwingend passiert, noch bei welchen Operationen welches SIGNAL erzeugt wird.

Die Neuregelung sieht vor, dass

1. bei jeder Bereichsüberschreitung in den Datentypen FIXED, FLOAT, DURATION und CLOCK ein konkretes SIGNAL erzeugt wird.
2. bei Division durch 0 ebenfalls ein passendes SIGNAL erzeugt wird.
3. bei Taskingaufrufen, die nicht im Taskzustandsdiagramm vorgesehen sind, konkrete SIGNALs erzeugt werden. Die notwendigen Ausnahmen bei eingeplanten Taskingaufrufen wurden in der Sprachdefinition von OpenPEARL aufgenommen.

3.3.3 GOTO-Anweisung

Die Sprachnorm von PEARL90 verbietet nicht, dass ein per GOTO-Anweisung ausgelöster Sprung inmitten eines Blocks geht. Dies ist sicherlich ein schlechter Programmierstil und würde zu nicht initialisierten Variablen führen.

Die Neuregelung sieht vor, dass das Sprungziel der GOTO-Anweisung nur in der gleichen Blockebene oder in einer übergeordneten Blockebene definiert sein darf.

3.3.4 Formatierte Ein- und Ausgabe

F-Format Bei der formatierten Ein- und Ausgabe ist beim F-Format detailliert beschrieben, welche Reaktion das PEARL-Programm bei zu großem Wert oder unzulässigen Formatierungsparametern hat. So wird z.B. bei zu großen Zahlen das Ausgabefeld komplett mit dem Zeichen '*' gefüllt werden und ein Fehler erzeugt werden. Bei manchen unsinnigen Formatierungsparametern erfolgt dies ebenfalls, bei anderen soll gleich ein SIGNAL erzeugt werden.

Die Beschreibung der Wirkung des Skalierungsparameters p im F-Format führt zu Problemen: Es ist festgelegt, das beim Format `F(w,d,p)` der um 10^p multiplizierte Wert transferiert wird. Damit ist es mit dem Format `F(6,1,-1)` möglich, eine FIXED Variable als Festpunktzahl mit einer Nachkommastelle auszugeben.

```
! nutze FIXED um auf Systemen ohne FPU effizient zu arbeiten
DCL temperatur FIXED(375); ! 37.5 Grad
...
PUT temperatur TO monitor BY F(5,1,-1), SKIP; ! gibt '_37.5'  aus
```

Die Wirkung für die Eingabe ist problematisch. Nach Beschreibung müsste hier das Format `F(5,1,1)` benutzt werden, damit die Zahl in der Interndarstellung entsprechend skaliert wird. Die Formatangabe für Ein- und Ausgabe einer Variable wäre dann unterschiedlich!

Die Neuregelung sieht vor, dass

1. bei Problemen mit zu großen Werten im Ausgabeformat das Feld nicht mehr mit '*' aufgefüllt wird, sondern die Ausgabeanweisung an dieser Stelle mit Fehlervariable, bzw. SIGNAL abgebrochen wird. Damit erhält der Programmierer mehr Möglichkeiten, die Ausgabe bei solchen Problemen anzupassen.
2. bei unzulässigen Werten für die Formatierungsparameter wird ein SIGNAL erzeugt und die Ein-/Ausgabeanweisung abgebrochen.
3. die Wirkung des Skalierungsparameters nun für Ein- und Ausgabe so definiert wird, dass die Wirkung jeweils symmetrisch ist, d.h. die gleiche Formatangabe für Ein- und Ausgabe genutzt werden kann. Positive Werte für den Parameter p sind zulässig, aber wenig sinnvoll.

E-Format Der Datentyp FLOAT(53) stellt einen Wertebereich bis 10^{308} bereit. Damit solche Zahlen auch ausgegeben werden können, wird das neue Format E3 mit einem dreistelligen Exponentenfeld bereitgestellt. Die Funktion ist sonst identisch zum bisherigen E-Format.

3.3.5 Alle Formate

Bei allen Formatierungsanweisungen führen unzulässige Formatierungsparameter zu einem SIGNAL und dem sofortigen Abbruch der Ein-/Ausgabeanweisung.

3.3.6 READ/WRITE mit POS

Die Sprachreferenz legt fest, dass `x,y FROM POS(3), POS(5);` zulässig ist. Allerdings sollen die Formatierungsanweisungen komplett abgearbeitet werden, bevor die Datenliste bearbeitet wird.

Die zwei Positionieranweisungen sind irritierend, da diese suggerieren, dass die beiden Werte von POS(3) bzw. POS(4) gelesen würden. Daher wird in OpenPEARL in diesem Fall eine Warnung vom Compiler erzeugt werden.

3.3.7 Datentypen

OpenPEARL stellt alle Größen für FIXED von 1 bis 63 zur Verfügung. Bei FLOAT werden nur die Größen 24 und 53 bereitgestellt.

Die Größe einer FLOAT-Konstanten wird wie folgt abgeleitet:

1. durch die explizite Angabe (z.B. 1.0(53))
2. durch die LENGTH-Anweisung
3. durch den Defaultwert von 24

Beim Typ BIT werden die Längen 1 bis 64 bereitgestellt. Im Falle eines fehlerhaften Indexwertes in einer der Slice-Operationen wird ein spezielles SIGNAL erzeugt.

Beim Typ CHAR werden die Längen von 1 bis 32767 bereitgestellt. Im Falle von größeren Ergebnissen wird ein spezielles SIGNAL erzeugt. Ebenso bei ungültigen Werten bei den Slice-Operationen.

Die Genauigkeit von CLOCK und DURATION Werten wurde auf $1\mu s$ fixiert. Der Wertebereich von DURATION liegt bei ca. $\pm$100 Jahren. Dies rührt aus dem Wertebereich des zugrunde gelegten FIXED(63) Typs her. Bereichsfehler bei DURATION führen auch hier zu einem SIGNAL: CLOCK-Werte werden entsprechend der Sprachdefinition Modulo 24 Stunden gerechnet.

3.3.8 Schleifen

Der Datentyp für die Schleifenvariable ergibt sich als Maximum für den Ergebnistyp der Ausdrücke für Anfangswert und Endwert. Falls beide fehlen, wird FIXED(31) gewählt.

Es wurde klargestellt, dass die Schleifenausdrücke bei Anfangs- und Endwert nur einmalig ausgewertet werden. Änderungen im Schleifenrumpf bleiben daher ohne Auswirkung auf den Schleifenablauf.

3.4 Signal Behandlung

Bislang war es syntaktisch korrekt, wenn ein eingeplanter Signalhandler nur aus einer einzelnen Anweisung bestand. Diese Anweisung wird im Falle eines Signals ausgeführt und die Prozedur beendet. Bei Funktionsprozeduren führt dies zu dem Problem, dass der Rückgabewert nicht definiert ist.

Die Neuregelung sieht vor, dass ein Signalhandler mit einer RETRN-Anweisung, TERMINATE, GOTO oder INDUCE beendet werden muß. Damit ist die Existenz von Rückgabewerten durch den Compiler prüfbar. Die INDUCE Anweisung erlaubt es, ein SIGNAL an die übergeordnete Prozedur oder Task zu schicken. Im Signalhandler darf bei INDUCE der Name des Signals entfallen, wenn das gleiche Signal weitergeleitet werden soll.

3.4.1 Taskingfunktionen

Es wurde klargestellt, dass

- in einer „WHEN ... AFTER ...“ -Konstruktion bei jedem Eintreffen des Interrupts die Zeitdauer neu gestartet wird (vgl. Abb. 1).
- eine gegebenenfalls spezifizierte Taskpriorität nur für den aktuellen Lauf dieser Task gilt.
- die Zeitberechnung bei OpenPEARL zum Zeitpunkt der Einplanung erfolgt, sodass eine spätere Umstellung der Uhrzeit keine Auswirkung zeigt.
- ein SUSPEND wird bei einer Task, die gerade mit Ein-/Ausgabe beschäftigt ist, solange verzögert, bis diese die Anweisung abgeschlossen hat, oder zumindest den Datensatz fertig bearbeitet hat — also ein SKIP-Format bearbeitet hat.

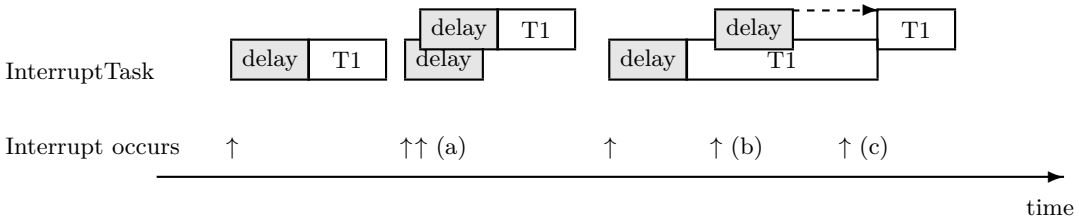


Abbildung 1: Die Task Interrupt Task wird durch einen Interrupt gestartet. Der Start selbst wird um 1 Sekunde verzögert. Falls der Interrupt innerhalb der Startverzögerung erneut auftritt, so wird die Verzögerung neu gestartet (a). Wenn der Interrupt während der Abarbeitung der Task erneut auftritt, so wird das Verzögerungsintervall sofort gestartet (b). Wenn die Task bei Ablauf des Verzögerungsintervalls noch nicht beendet ist, so wird der Start einmalig gepuffert und unmittelbar nach Terminierung der Task durchgeführt (c).

3.4.2 Semaphore

Wenn eine Semaphore zu oft hintereinander freigegeben wurde, sodass der interne Zähle überläuft, wird ein SIGNAL erzeugt.

3.5 Diskussionspunkte

Einige Regelungen in der Sprachdefinition erscheinen nicht mehr so ganz zeitgemäß. Hier würden sich die Autoren um Diskussionsbeiträge, gerne auch auf der Tagung Echtzeit in Boppard, freuen.

1. Ist es noch sinnvoll, auf Gleitpunktarithmetik zu verzichten und statt dessen intern mit FIXED-Variablen zu rechnen und diese mit dem F-Format bei der Ein-/Ausgabe entsprechend zu skalieren?
2. Ist die automatische Wiederholung der Formatliste bei der Ausgabe und längerer Variablenliste noch gewünscht? Bei der Eingabe sind längere Formatlisten eine Fehlerursache. Bei der Ausgabe scheint dies nicht zu stören. Ist diese Asymmetrie erwünscht? Wäre es nicht sinnvoller, wenn die Formatlisten und Datenlisten bei PUT und GET exakt aufeinander passen würden?

Literatur

[1] GI Fachgruppe 4.4.2: PEARL90 – Sprachreport, Version 2.0, 1995.

[2] W. Gerth: RTOS-UH, Version 21.6.2006

4 SPS-Projektierung mit SIMATIC STEP 7 am Beispiel einer Katzenmilch-Abfüllanlage

Christina K. Houben, FernUniversität in Hagen
christina.houben@fernuni-hagen.de

Die Programmierung einer SPS erfolgt zunächst am PC, wo die Hardware-Konfiguration festgelegt und die Ansteuerungsbefehle der verschiedenen Organisationsbausteine und ihrer Untereinheiten über eine der in IEC 61131-3 festgelegten Programmiersprachen definiert werden. Anschließend wird das Verhalten der so konfigurierten SPS am PC simuliert, um eine schnelle Rückmeldung über die Korrektheit des Programms zu erhalten, und darüberhinaus vorerst die Kosten für einen Test am realen technischen Prozess zu sparen. Konnte dabei die korrekte Funktionsweise des Automatisierungssystems validiert werden, wird das erstellte Programm über ein Programmiergerät auf die entsprechende SPS übertragen, die dann im Feld getestet und zuletzt produktiv eingesetzt werden kann.

Die Arbeitsschritte beginnend bei der Hardware-Auswahl über die Programmierung bis hin zur Simulation am PC eignen sich nicht nur zur Kostenreduzierung im Entwicklungsbetrieb, sondern auch für Zwecke der Lehre im Bereich Automatisierungstechnik. Daher möchten wir im Folgenden die Arbeitsabläufe bei der Projektierung einer Katzenmilch-Abfüllanlage mit dem Programm *STEP 7* und der zugehörigen Simulationssoftware *PLCSIM* darstellen.

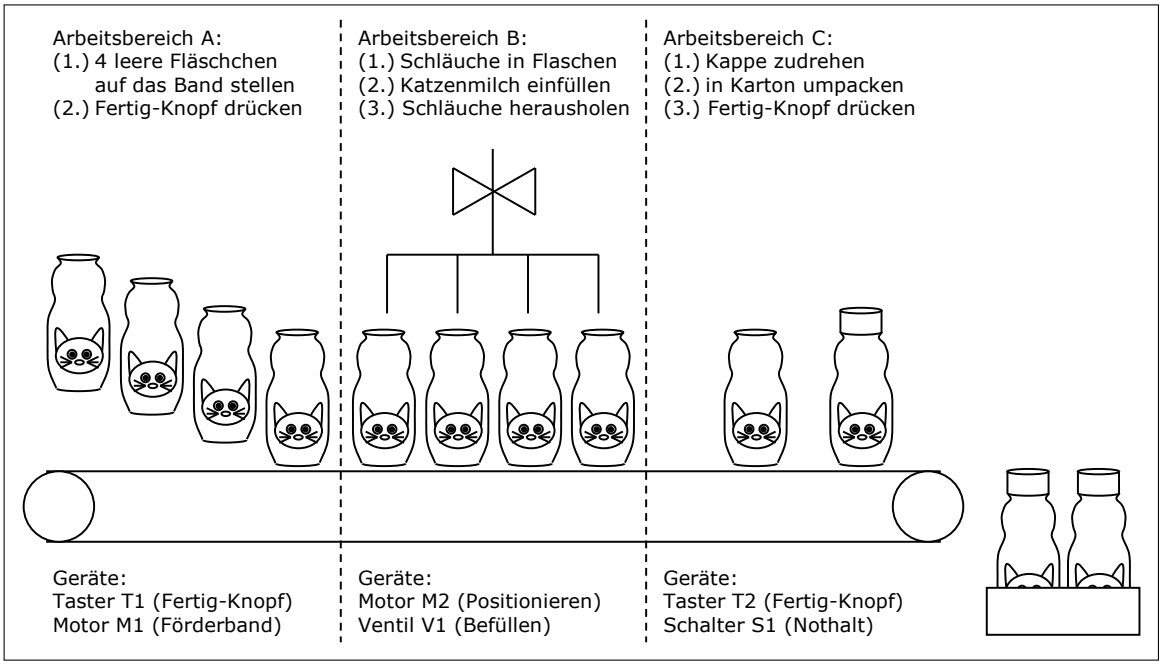


Abbildung 2: Aufbau der Abfüllanlage entlang eines Förderbandes

Beide Programme haben wir als Demo-Version von der CD zum Buch *SPS – Theorie und Praxis* [1] bezogen. Die Installation dauerte mehr als 90 Minuten, was an dem zusätzlich zu installierenden Lizenz-Manager, dem umfangreichen Hardware-Katalog und einiger anderer Extras wie der Möglichkeit zur Programmierung mit *S7 Graph* liegen könnte. Dem Buch lag auch eine CD

mit dem Programm *Mhd SPS Visu* bei, womit man nicht nur den Zustand der Ein- und Ausgabebaugruppen der SPS simulieren kann, sondern auch eine einfache bildliche Darstellung der Förderbänder, Flaschen, Taster etc. ermöglicht. Letzteres ist jedoch ohne Mehrwert für die Lehre bzw. Ausbildung zum Automatisierungstechniker, weshalb wir darauf nicht weiter eingehen.

Die Abfüllanlage ist in drei Arbeitsschritten entlang eines Förderbandes wie in Abbildung 1 aufgebaut. Im ersten Schritt werden jeweils vier Flaschen von Hand auf das Förderband gestellt. Im letzten Schritt schraubt das Bedienpersonal je einen Deckel auf jede Flasche und sortiert die Flaschen in kleine Kisten ein.

Für jeden der beiden Arbeitsschritte drückt das Bedienpersonal auf einen Taster um anzuzeigen, dass sie ihren Arbeitsschritt erledigt haben. Sind beide Taster betätigt worden, nimmt die Automatisierungsanlage ihre Arbeit auf. Das Förderband wird durch einen Motor so weit bewegt, dass die ersten vier Flaschen im Bereich des Arbeitsschrittes B stehen. Dort werden vier Schläuche durch einen Motor in den Flaschen positioniert, ein Ventil für eine vorgegebene Zeitspanne geöffnet und anschließend geschlossen. Zuletzt befördert das Band die Flaschen zum nächsten Schritt und das Bedienpersonal kann weiterarbeiten.

4.1 Hardware festlegen

(1.) Zuerst wird der *SIMATIC Manager* gestartet und ein neues Projekt erstellt: *SIMATIC Manager* → *Datei* → *Neu...* → *Typ: Projekt* → *Name: Katzenmilch* → *OK*.

(2.) Dann kann eine SPS eingefügt werden: *Einfügen* → *Station* → *2 SIMATIC 300-Station*, siehe Abb. 2.

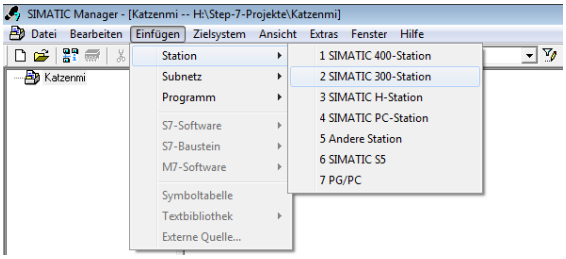


Abbildung 2: SPS einfügen

(3.) Nun kann der Hardware-Katalog geöffnet werden, indem doppelt auf das Icon *Hardware* geklickt wird, siehe Abb. 3.

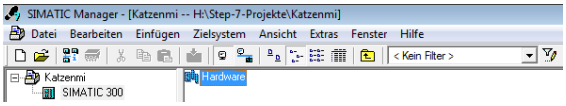


Abbildung 3: Hardware-Katalog öffnen

(4.) In dem sich öffnenden Fenster wird anschließend eine Montageschiene ausgewählt, an der weitere Baugruppen befestigt werden können, siehe Abbildung 4: *SIMATIC 300* → *RACK-300* → *Profilschiene*.

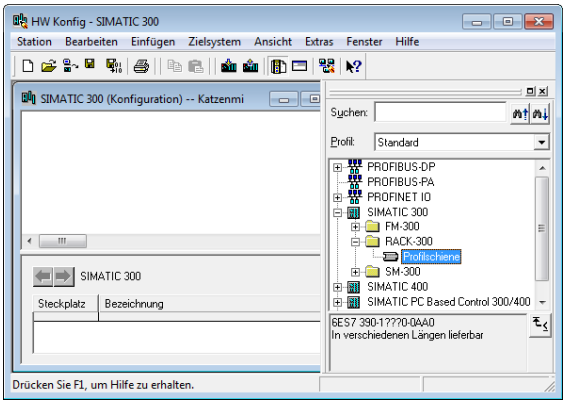


Abbildung 4: Montageschiene auswählen

(5.) Nun können wir nacheinander für jeden Steckplatz der Profilschiene weitere Hardware-Komponenten einfügen. Auf den ersten Steckplatz, den wir grün markieren, kommt aus dem Hardware-Katalog eine Stromversorgungsbaugruppe, siehe Abb. 5: *SIMATIC 300 → PS-300 → PS 307 10A*.

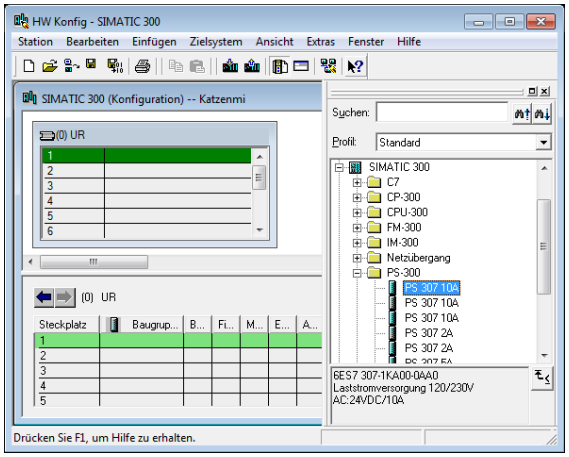


Abbildung 5: Stromversorgung auf Steckplatz 1

(6.) Weitere Baugruppen sind:
Steckplatz 2 – Prozessor:
SIMATIC → 300 CPU-300 → CPU 313C
Steckplatz 4 – Digitale Eingabebaugruppe:
SM-300 → DI-300 → DI16xAC120V
Steckplatz 5 – Digitale Ausgabebaugruppe:
SM-300 → DO-300 → DO16xUS24/48V

(7.) Zuletzt wird die Konfiguration mit dem in Abbildung 6 gezeigten Knopf gespeichert und übersetzt.

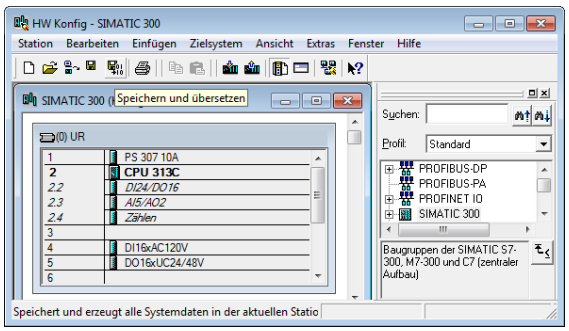


Abbildung 6: Fertige Hardware-Konfiguration

4.2 Symbole und Adressen

Als nächstes werden Symbolnamen definiert und mit Adressen der Ein- und Ausgabebaugruppen assoziiert. Die Symbolnamen können hinterher bei der Erstellung der Programmanweisungen z.B. mit der Funktionsplansprache FUP als Variablennamen benutzt werden. Außerdem werden auch die Funktionsbausteine, welche Teilaufgaben des Programms erfüllen, deklariert. Unsere Katzenmilch-Abfüllanlage enthält drei verschiedene Unterprogramme, nämlich die Anlage in einen sicheren Zustand zu überführen, das Förderband zu bewegen und die Flaschen zu befüllen. Beispielhaft werden wir als nächstes das Unterprogramm implementieren, welches das Förderband bewegt.

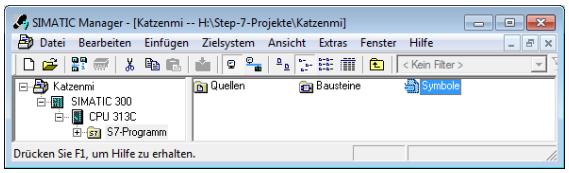


Abbildung 7: Symboltabelle aufrufen

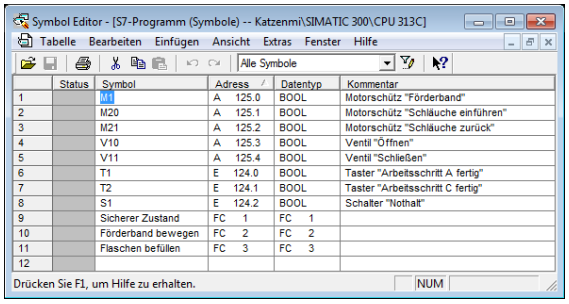


Abbildung 8: Symboltabelle befüllen

Zunächst markieren wir im Hauptfenster das Item *S7-Programm* und wählen aus dem Menü den Punkt *Einfügen* → *S7-Baustein* → *3 Funktion*, siehe Abb. 9. Die Eigenschaften der Funktion geben wir wie in Abb. 10 an.

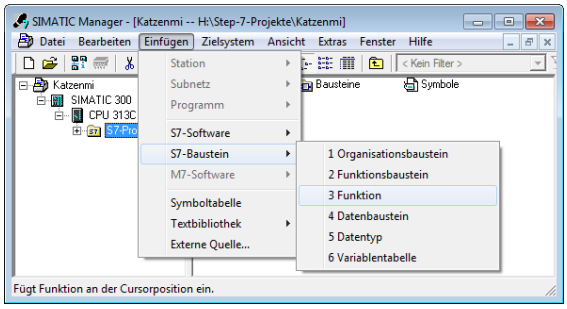


Abbildung 9: Funktion erstellen

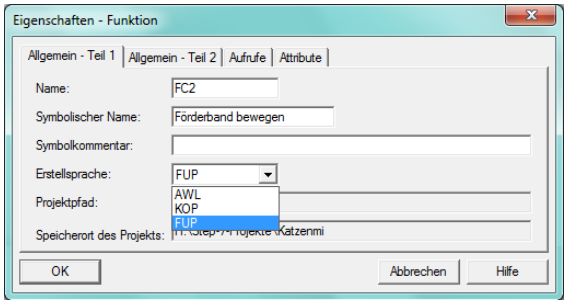


Abbildung 10: Eigenschaften einer Funktion

Als Programmiersprachen stehen Anweisungsliste (AWL), Kontaktplan (KOP) und Funktionsplan (FUP) zur Verfügung. Würde man ein zusätzliches Paket installieren, so könnte auch die Sprache *S7 Graph* gewählt werden, welche der Ablaufplansprache ähnelt. Zwischen den einzelnen Erstellsprachen kann später noch jederzeit gewechselt werden, und zwar über den Menü-Punkt *Ansicht*. Dort können auch über *Ansicht* → *Anzeigen mit ...* verschiedene Einstellungen wie die Anzeige von Kommentaren, Symbolinformationen etc. getätigt werden.

Im nächsten Schritt, siehe Abb. 11, programmieren wir die Ansteuerung des Förderbandes mittels FUP. Die einzelnen Funktionsbausteine können aus dem linken Baum mittels Drag & Drop im Netzwerk-Plan positioniert werden. Die beiden UND-Glieder, den Konnektor *M1.0*, Zuweisungsbaustein *M1*, die Negationskringel und zusätzliche Eingänge für den ersten UND-Baustein findet man unter dem Item *Bitverknüpfung*. Das Verzögerungsglied, um das Förderband zehn Sekunden nach dem Einschalten anzuhalten, findet man unter *Zeiten*.

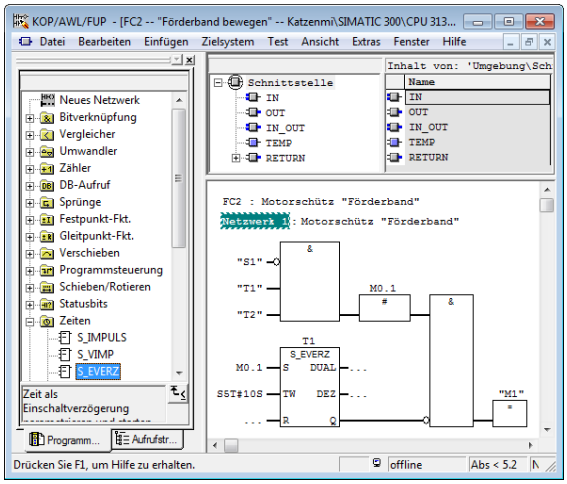


Abbildung 11: Funktionsplan erstellen

Wir mussten den Konnektor unter Zuhilfenahme des Merker-Bausteins *M0.1* verwenden, weil es im Baum links keine Verzweigung gab.

Anschließend kann das soeben erstellte Netzwerk gespeichert werden. Ebenso verfahren wir mit den anderen Funktionen. Was neben den drei Funktionen noch fehlt, ist ein Organisationsbaustein *OB1*, welcher die Schnittstelle zwischen Anwenderprogramm und dem Betriebssystem der SPS darstellt und die Struktur des Programms aufzeigt. Diesen Organisationsbaustein bauen wir aus drei Netzwerken auf, die nacheinander einen der folgenden AWL-Befehle enthalten:

```
QB1: ''Katzenmilch Programmzyklus''
Netzwerk 1: Aufruf FC1
    CALL ''Sicherer Zustand''
Netzwerk 2: Aufruf FC2
    CALL ''FÃűrderband bewegen''
Netzwerk 3: Aufruf FC3
    CALL ''Flaschen befÃllen''
```

Abbildung 12: Organisationsbaustein

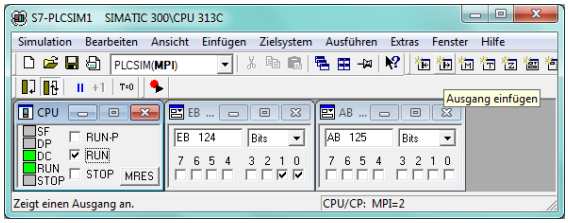


Abbildung 13: Simulation mit PLCSIM

4.4 Simulation mit PLCSIM

Nach der vollständigen Programmierung des Programms kann dieses mit PLCSIM getestet werden. Befindet man sich im SIMATIC Manager, klickt man auf das kleine Symbol *Simulation ein/aus*, welches das fünftletzte in der Schalterleiste ist. Es öffnet sich ein neues Fenster mit dem Titel *S7–PLCSIM1*. Darin muss man über *Einfügen* → *Eingang* und *Einfügen* → *Ausgang* eine Ein- und Ausgabebaugruppe einfügen. In die Adressleisten der beiden Baugruppen tragen wir die Bytes aus unserer Symboltabelle ein, nämlich *EB 124* und *AB 125*.

Im linken Teilfenster aus Abbildung 13 setzen wir bei *Run* ein Häkchen, was bedeutet, dass wir den Drehknopf an der SPS von *Stop* auf *Run* gestellt haben. Nun läuft unsere Simulation an. In die Bits *EB 124.0* und *EB 124.1* setzen wir ein Häkchen, um zu signalisieren, dass beide Arbeitsgruppen aus den Arbeitsbereichen A und C ihren jeweiligen Fertig-Knopf gedrückt haben. Nach Bearbeitung des ersten SPS-Zyklus erscheint ein Häkchen im Bit AB 125.0, was anzeigt, dass der Motorschütz, der das Förderband zehn Sekunden lang bewegt, aktiviert wurde.

4.5 Zusammenfassung

Wir haben in einer Schritt-für-Schritt-Anleitung gezeigt, wie man eine SPS programmieren und testen kann, ohne vor Ort die Hardware, geschweige denn die ganze Automatisierungsanlage, angeschlossen zu haben. Dies ermöglicht nicht nur eine investitionssparsame und schnelle Entwicklung, sondern auch den Einsatz in Lehre und Ausbildung.

Literatur

[1] H. Tapken: SPS – Theorie und Praxis, Vol. 3, Europa-Lehrmittel, 2013

5 Zusammenfassungen von Abschlussarbeiten

Jutta Düring, Fachausschuss Echtzeitsysteme

5.1 Aufruf zur Einreichung

Bereits im Jahr 2005 wurde die Idee geboren, Abstracts von Abschlussarbeiten und Dissertationen aus den Einrichtungen unserer Mitglieder mit starkem Bezug zu PEARL oder Echtzeitproblemen als festen Bestandteil in den Rundbrief aufzunehmen. Ziel war die Darstellung der Breite der aktuellen Forschungsarbeiten in diesem spannenden Gebiet.

Diese gute Idee ist im Laufe der Jahre leider in Vergessenheit geraten und ich möchte dieses Thema wieder aufgreifen. Die Leser der EchtZeit werden hiermit gebeten, auf geeignete Arbeiten zu achten und die betreffenden Zusammenfassungen mit den ergänzenden Daten an die Redaktion (pearl@fernuni-hagen.de) zu übermitteln.

Der Umfang der Zusammenfassung sollte ca. 15–20 Textzeilen umfassen. Die Einreichung sollte Angaben über Verfasser, Hochschule, Kontaktadresse (E-Mail) und evtl. eine URL für weitere Informationen enthalten. Als Textformat wird LaTeX bevorzugt.

5.2 Semantische Analyse einer Echtzeitprogrammiersprache mittels Verhaltensmustern

Jens Dörwald, B.Sc. Informatik, FernUniversität in Hagen, Lehrgebiet Kommunikationsnetze

Schwerpunkt dieser Arbeit ist die Identifizierung und Kennzeichnung von semantischen Fehlern in beliebigen PEARL 90 Quelltexten. Semantische Fehler sind z.B. solche wie die Verwendung von zuvor noch nicht deklarierten Bezeichnern im Quelltext oder Verstöße gegen die Typregeln der verwendeten Programmiersprache. Aus diesem Grund wurde eine semantische Analyse entwickelt, die insbesondere die Prüfung der Typkonformität in den PEARL 90 Quelltexten beinhaltet.

Bei der Typkonformität müssen im Gegensatz zur Typäquivalenz die gegenüber zu stellenden Typen nicht identisch sein, um zuweisungskompatibel zu sein. Entsprechend der im Sprachreport zu PEARL 90 angegebenen Regelmenge sind Verhaltensmuster zur Prüfung der Typkonformität entwickelt worden.

Dabei muss die Zuweisungskompatibilität der Operanden bei einfachen Zuweisungen ebenso berücksichtigt werden wie die Verwendung von gültigen Datentypen auf Operationen in beliebigen Ausdrücken. Zusätzlich spielt bei der Auswertung von Ausdrücken die Reihenfolge der angegebenen Operatoren eine Rolle, da diese entsprechend der allgemeinen arithmetischen Regeln auszuwerten sind.

6 Einladung zur Mitgliederversammlung mit Neuwahl

Jutta Düring, Fachausschuss Echtzeitsysteme

Wir laden ein zur Mitgliederversammlung des Fachausschusses Echtzeitsysteme am

Donnerstag, den 12. November 2015 um 20:00 Uhr

im Panoramasaal des Hotel Ebertor, Heerstraße 172, 56154 Boppard am Rhein.
Die Anfahrtsbeschreibung finden Sie unter http://www.ebertor.de/kontakt/anfahrt.html

Wir bitten um zahlreiches Erscheinen, da in diesem Jahr die Neuwahl der Fachausschussleitung ansteht.