Eine sicherheitsgerichtete Echtzeitprogrammiersprache für die Sicherheitsstufe SIL 3 gemäß DIN EN 61508

Jürgen Hillebrand

Inhalt

1. Einleitung
   - Vorgaben zur Definition der sicherheitsgerichteten Echtzeitprogammiersprache (SGEZ)
   - Relevante Empfehlungen für Programmiersprachen für SIL 3.
2. Moduldefinition und Nutzung von Programmobjekten
   - Programm- und Moduldefinition.
   - Spezifikation von Variablen, Funktionen und Funktionsprozeduren.
   - Aufruf von Funktionen und Funktionsprozeduren.
3. Ursache-Wirkungstabellen in SIL 3 Programmen
4. Echtzeitbetrieb von Programmen
   - Sprachelemente zur zeitgerechten und deterministischen Tasksteuerung.
   - Der Schedulingteil zur Überwachung des Programmablaufs.
5. Zusammenfassung

SGEZ für die Sicherheitsstufe SIL 3 gemäß DIN EN 61508

1/5 EINLEITUNG

Vorgaben zur Definition der SGEZ

- Definition einer sicherheitsgerichteten Echtzeitprogrammiersprache, welche
  - den Sicherheitsanforderungen nach SIL 3 und SIL 4 (DIN EN 61508) genügt,
  - auf den Definitionen
    - der Echtzeitprogrammiersprache PEARL 90 (DIN 66253),
    - Verifiable PEARL und Table-PEARL (erste Ansätze) basiert,
  - eine Programmierung von besonders einfach und verlässlich verifizierbaren Programmen ermöglicht,
    - speziell mit der diversitären Rückwärtsanalyse,
  - die Sicherheit von Programmen inhärent fördert und
  - eine Programmierung mit Funktionsplänen und Ursache-Wirkungstabellen ermöglicht.

Empfehlungen aus Normen und Literatur für SIL 3 und SIL 4

Empfehlung aus relevanten Normen (DIN EN 61508, DIN EN 61511 und ISO 26262)	
Modulare Programmierung	✓
▸ Kapselung von Daten und Objekten (Schutz vor Datenmanipulation)	✓
▸ Klar definierbare Modulschnittstellen (keine globalen Vereinbarungen)	✓
Strenge Typisierung (von Datenelementen ➡ ermöglicht statische Tests)	✓
Ausschluss von Sprachelementen die die Verifikation erschweren:	✓
▸ dynamische Objekte und Variablen (z.B. Stacks)	✗
▸ Zeiger und prozedurale Parameter	✗
▸ unbedingte Sprünge (**GOTO**) und bedingte Sprünge (wie z.B. Schleifen)	✗
▷ **IF-THEN** & **CASE**-Anweisung (DIN EN 61511)	👆

Empfehlungen aus Normen und Literatur für SIL 3 und SIL 4

Empfehlungen aus der Literatur	
Zeitgerechte und deterministische Zuteilungsverfahren (Antwortzeitalgorithmus und zeitsynchrone Programmierung)	✓
Semaphore und Bolts (Systemverklemmungen)	✗
Erkennung & Behandlung von Ausnahmen (Try-Catch-Anweisung)	✓
Abgrenzung von Code mit unterschiedlichen Sicherheitsstufen	✓
Keine Deklaration von Variablen und Funktionen in Anwenderquelltexten	✗
▸ nur Verwendung von bereits geprüften und zertifizierten Komponenten	✓

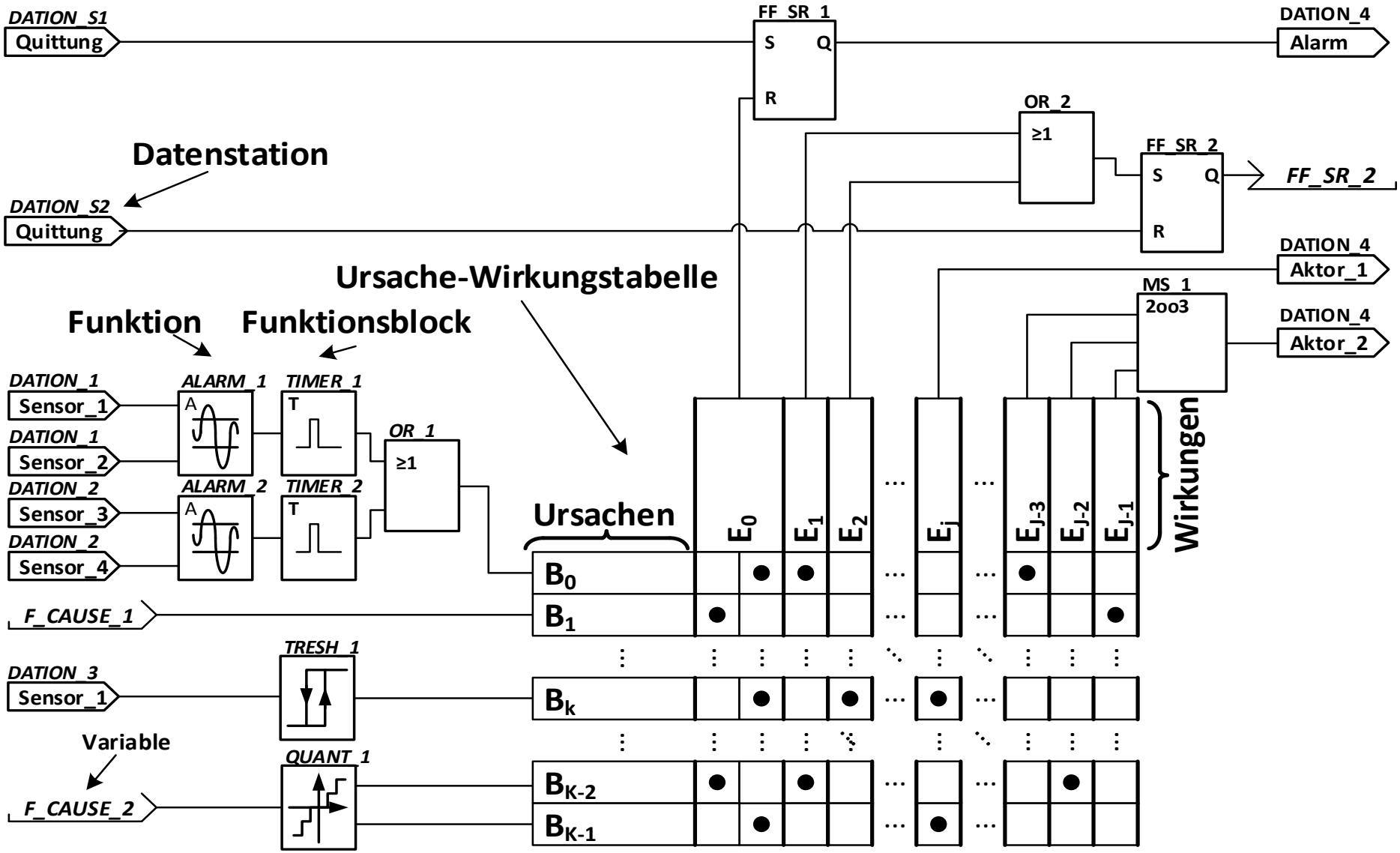
PEARL90 (DIN EN 66253) in Bezug auf relevante Empfehlungen

Empfehlung aus relevanten Normen (DIN EN 61508, DIN EN 61511 und ISO 26262)		
Modulare Programmierung	✓	
▸ Kapselung von Daten und Objekten	✗	▸ Kein geeignetes Zugriffsschutskonzept.
		▸ Manipulation an Daten durch globale Zugriffe möglich.
▸ Klar definierbare Modulschnittstellen	✗	▸ Modulschnittstellen werden über globale Vereinbarungen definiert.
		▸ Zugriff auf externe Programmobjekte durch Spezifikation (**SPECIFY**) möglich.
Zeitgerechte und deterministische Zuteilungsverfahren	✗	▸ Zuteilung nach Prioritäten
		▸ Zuteilung nach Startterminen (keine Antwortzeiten).
Erkennung & Behandlung von Ausnahmen (Try-Catch-Anweisung)	✗	Nicht unterstützt.
Abgrenzung von Code mit unterschiedlichen Sicherheitsstufen	✗	Nicht unterstützt.

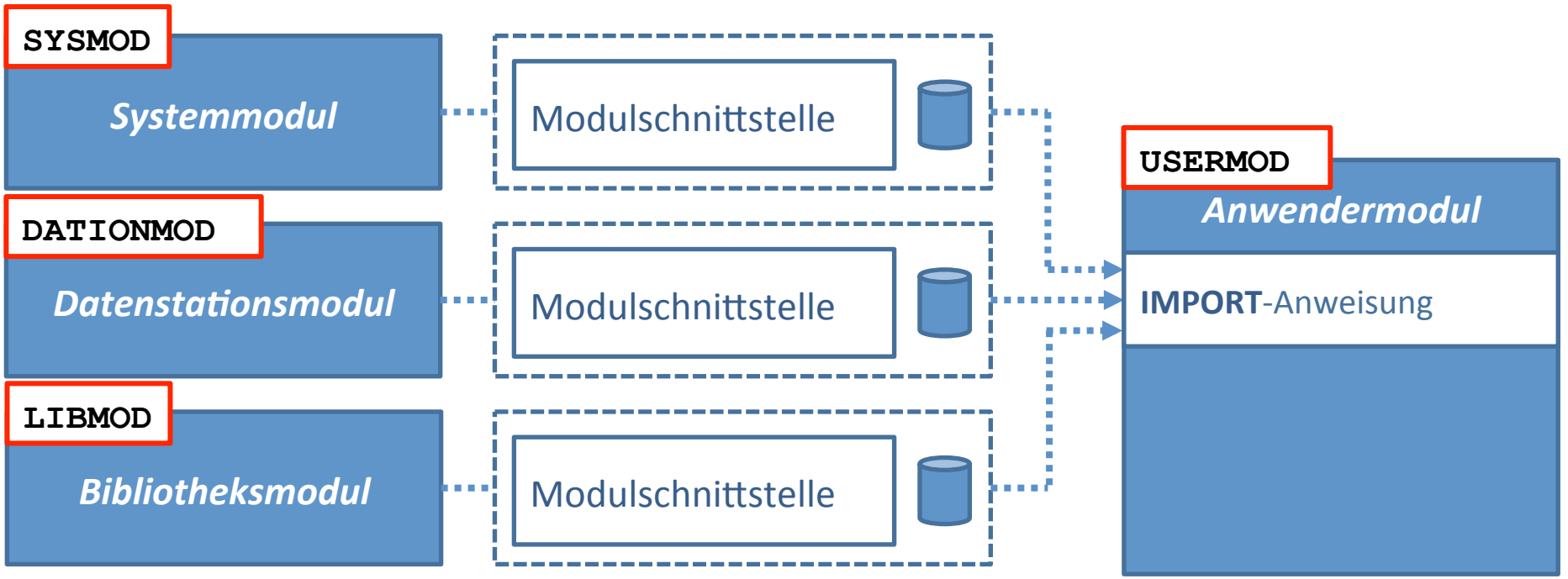
SGEZ für die Sicherheitsstufe SIL 3 gemäß DIN EN 61508

2/5 MODULDEFINITION UND NUTZUNG VON PROGRAMMOBJEKTEN

Programmierung mit Funktionsplänen und Ursache-Wirkungstabellen



Modulare Programmierung



- Programm besteht aus mehreren Modulen, die „typisiert sind“:
  - Anwendermodul ➡ Implementierung der Aufgabenstellung des Programms.
  - Systemmodul ➡ Betriebssystemfunktionen (z.B. Scheduling).
  - Datenstationsmodul ➡ Treiberfunktionen, Konfigurationsfunktionen.
  - Bibliotheksmodule ➡ Funktionsprozeduren (Funktionen und Funktionsblöcke) und Variablen.
- Module können auch in Form von Maschinencode vorliegen ➡ notwendige Informationen zum Linken werden als Dateien zusammen mit Schnittstellenbeschreibung übergeben.

Modul-Definition

```
<Modul> ::=
MODULE <Modulbezeichner> [<Modultyp>] [<Vereinbarung-SIL>] ";"
[<Modul-Import>]*
[<Schnittstellenteil>]
[<Scheduling-Teil>]
[<Systemteil>]
[<Problemteil>]
MODEND <Modulbezeichner> [<Vereinbarung-SIL>] ";"
```

- Schnittstellenteil ➡ Spezifikation von Objekten, die Modul anderen Modulen zur Verfügung stellen soll (Funktionsprozeduren, Variablen etc.) und Vereinbarungen von Datentypen und Konstanten.
- Schedulingteil ➡ Beschreibung des gewollten Echtzeitbetriebs des Programms durch Meilensteindiagramme und Kontrolltabellen (zur Überwachung des Echtzeitbetriebs).
- Systemteil ➡ Verbindungsbeschreibung zw. Modulintern genutzten Betriebsmitteln und angeschlossener Peripheriegeräte.
- Problemteil ➡ Formulierung der Aufgabenstellung des Programms / der Funktionen.

Benutzen von Objekten

- Variablen, Funktionsprozeduren und Datenstationen können in Anwendermodulen nicht deklariert / vereinbart werden.
- Diese Objekte werden lediglich benutzt und von anderen Modulen bereitgestellt.
- Modulextern bereitgestellte Objekte müssen in den Schnittstellen der Module spezifiziert werden.

```
...
  INTERFACE Beispiel MODTYPE DATIONMOD; /* {LIBMOD | DATIONMOD | SYSMOD} */

    /* --- Spezifikation Variable ------------------------------------- */
    SPC var CAUSE READ SAFEGUARD SIL3;

    /* --- Spezifikation Funktionsprozedur ---------------------------- */
    SPC fkt PROC(x CAUSE) SAFEGUARD SIL3 MAXRUNTIME BYSYSTEM RETURNS(EFFECT);

    /* --- Spezifikation Datenstationen mit Datenrichtung IN und OUT ---------*/
    SPC dat_in DATION IN CAUSE SAFEGUARD SIL3;
    SPC dat_out DATION OUT EFFECT SAFEGUARD SIL3;

  END Beispiel;
...
```

Benutzen von Objekten

- Bevor Objekte in Anwendermodule benutzt werden dürfen sind diese über die Modulschnittstellen zu importieren.

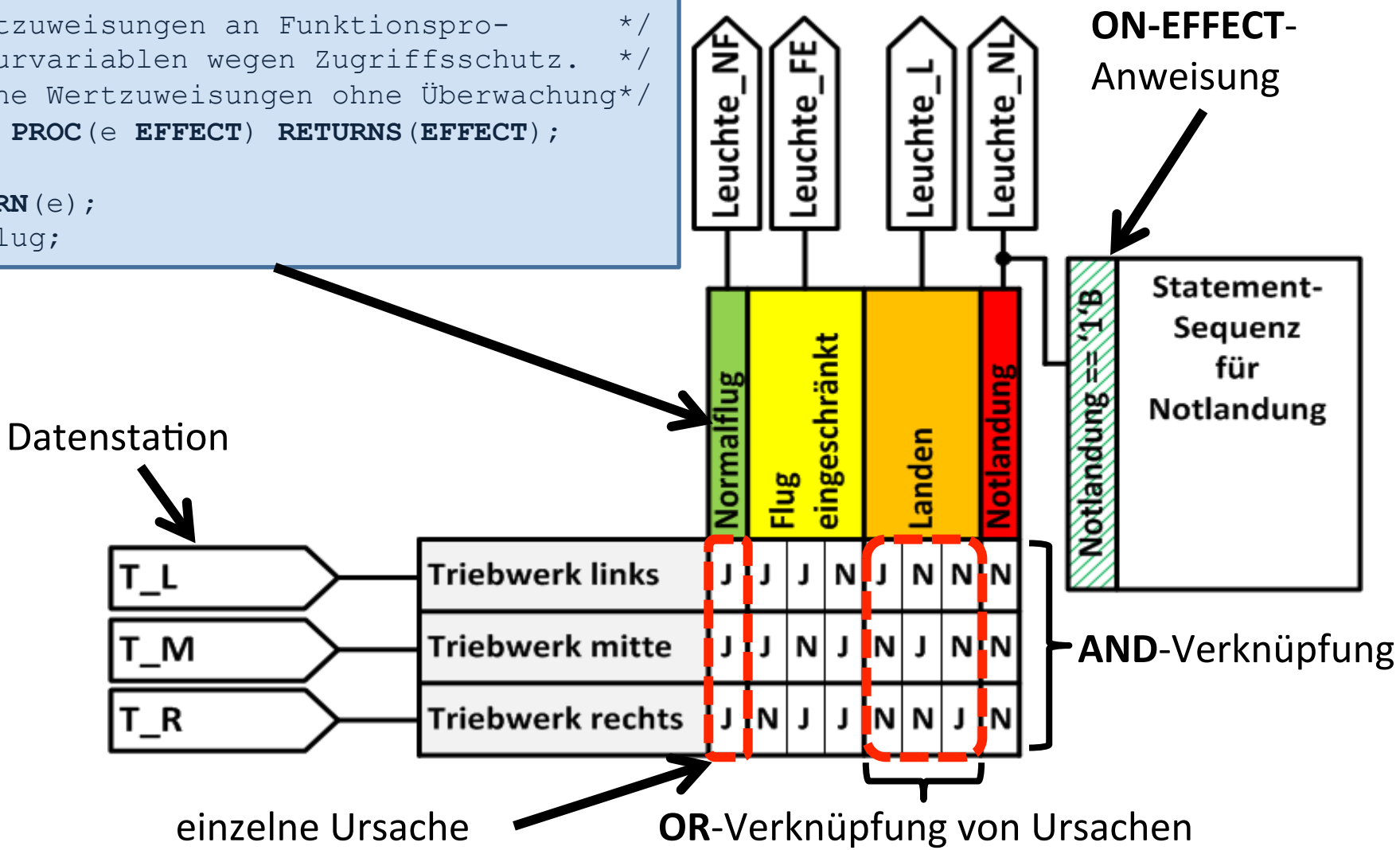
```
MODULE Anwender MODTYPE USERMOD SAFEGUARD SIL3;
  IMPORT Beispiel;
  ...
  PROBLEM Anwender MODTYPE USERMOD SAFEGUARD SIL3;
    ...
    /* --- Benutzen einer Variablen ------------------------- */
    CALL Beispiel.fkt(Beispiel.var);
    /* Rückgabewert wird Funktionsprozedurvariable zugewiesen. */

    /* --- Benutzen von Datenstationen ----------------------*/
    TAKEFORM Beispiel.dat_in;
    /* eingelesener Wert wird Datenstationsvariable zugewiesen */
    CALL Beispiel.fkt(Beispiel.dat_in);
    SEND Beispiel.fkt TO Beispiel.dat_out;
    ...
  PROBLEM Anwender SAFEGUARD SIL3;
MODEND Anwender SAFEGUARD SIL3;
```

SGEZ für die Sicherheitsstufe SIL 3 gemäß DIN EN 61508

3/5 URSACHE-WIRKUNGSTABELLEN IN SIL 3 PROGRAMMEN

Ursache-Wirkungstabellen in SIL 3 Programmen

```
/* Wertzuweisungen an Funktionspro-      */
/* zedurvariablen wegen Zugriffsschutz.  */
/* Keine Wertzuweisungen ohne Überwachung*/
NFlug: PROC(e EFFECT) RETURNS(EFFECT);
   ...
  RETURN(e);
END Nflug;
```



Ursache-Wirkungstabellen in SIL 3 Programmen

```
CETABLE UW_Triebwerksausfall SAFEGUARD SIL3;

 TAKEFROM T_L; TAKEFROM T_M; TAKEFROM T_R;

 SETEFFECT(NFlug)   TO CAUSE(     T_L AND      T_M AND      T_R); FIN;
 SETEFFECT(EFlug)   TO CAUSE(     T_L AND      T_M AND NOT  T_R);
                    OR CAUSE(     T_L AND NOT  T_M AND      T_R);
                    OR CAUSE(NOT  T_L AND      T_M AND      T_R); FIN;
 SETEFFECT(FLand)   TO CAUSE(     T_L AND NOT  T_M AND NOT  T_R);
                    OR CAUSE(NOT  T_L AND      T_M AND NOT  T_R);
                    OR CAUSE(NOT  T_L AND NOT  T_M AND      T_R); FIN;
 SETEFFECT(FNLand)  TO CAUSE(NOT  T_L AND NOT  T_M AND NOT  T_R); FIN;

 SEND NFlug  TO Leuchte_NF;
 SEND EFlug  TO Leuchte_EF;
 SEND FLand  TO Leuchte_L;
 SEND FNLand TO Leuchte_NL;


 ONEFFECT(FNLand) PERFORM;
    /* Statement-Sequenzen bei Notlandung */
 FIN;

END UW_Triebwerksausfall SAFEGUARD SIL3;
```

Wenn Bedingung == ‘1‘B dann wird Wert ‘1‘B an Funktionsprozedur übergeben, sonst wird Wert ‘0‘B übergeben.

Block wird betreten wenn Bedingung == ‘1‘B.

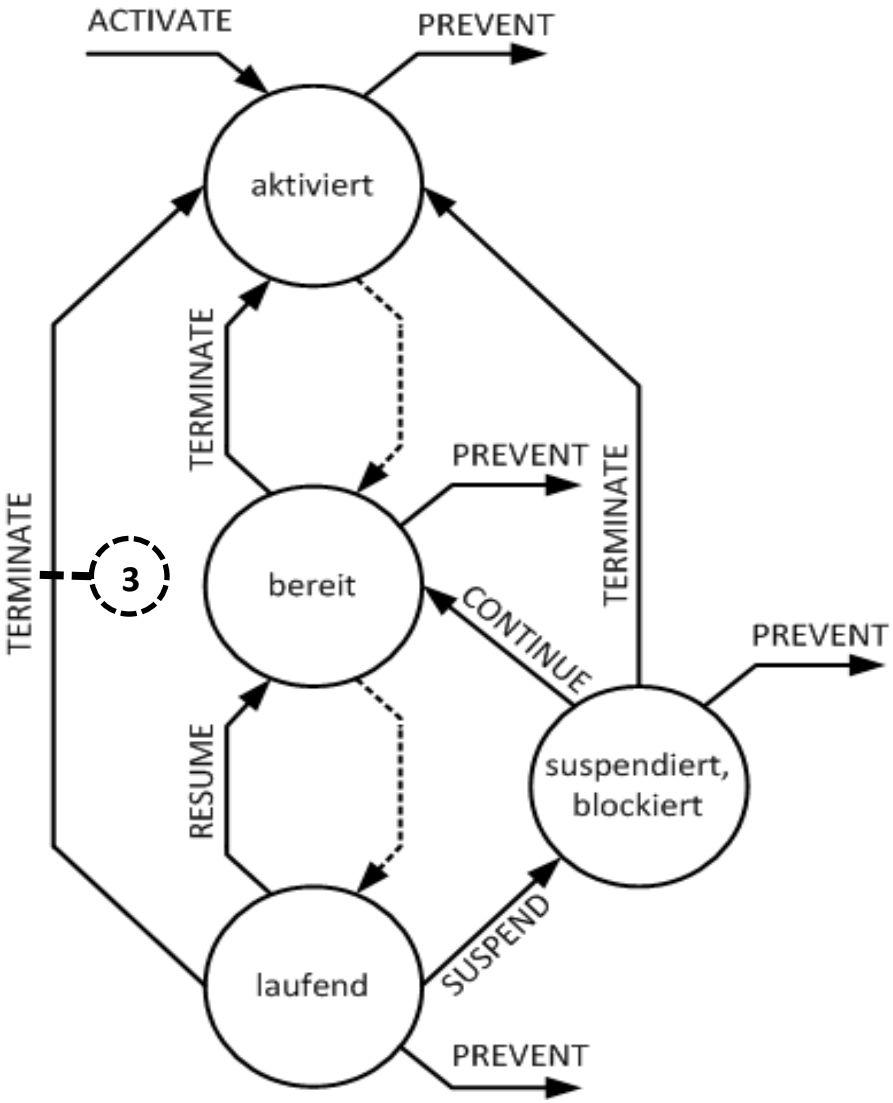
Auch (**NOT** FNLand) ist erlaubt, dann wird Block betreten, wenn FNLand == ‘0‘B ist.

SGEZ für die Sicherheitsstufe SIL 3 gemäß DIN EN 61508

4/5 ECHTZEITBETRIEB VON PROGRAMMEN

Echtzeitbetrieb

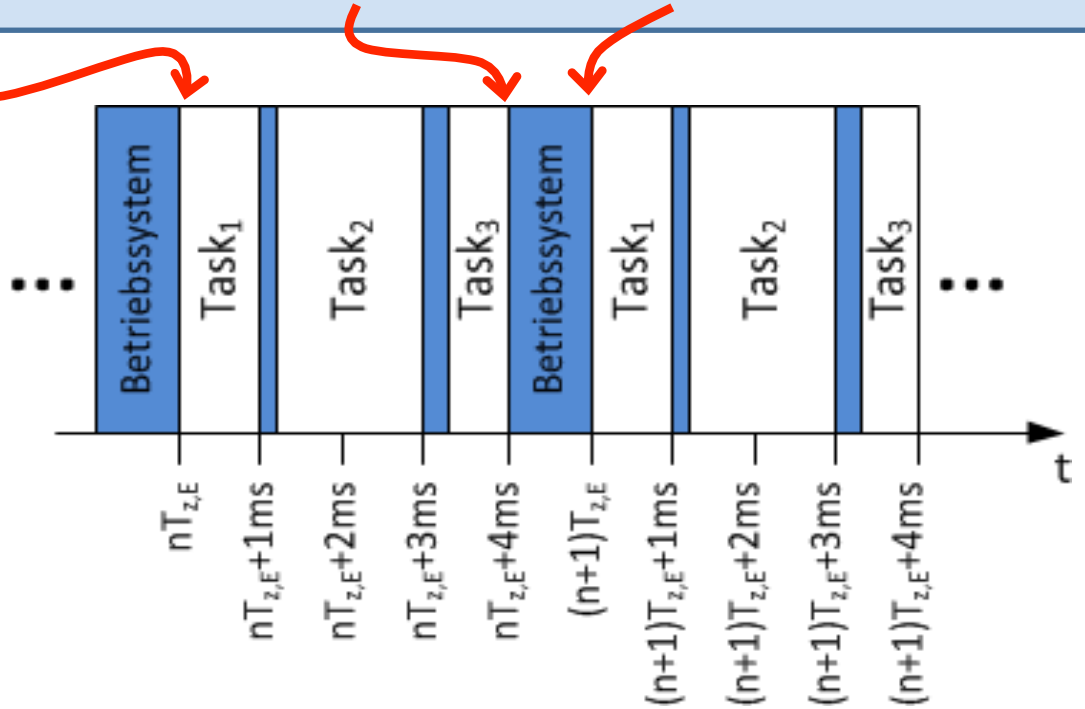
- Alle Anweisungen eines Programms müssen in Tasks programmiert werden.
- Tasks dienen zur Berücksichtigung des Echtzeitbetriebs von Programmen.
- Tasks werden unter Berücksichtigung der Zeit ausgeführt und können unterschiedliche Zustände annehmen.
- **Definierte Aktivierungsanweisung:**
  - Antwortzeit **[EXACTLY]**,
  - Synchronisation zum Takt von zyklisch arbeitenden Zuteilungsverfahren **[SYNCTONEXTSCHEDTICK]** und
  - alternative Tasks (falls was schief geht) **[ALTERNATIVETASK]**



Echtzeitbetrieb

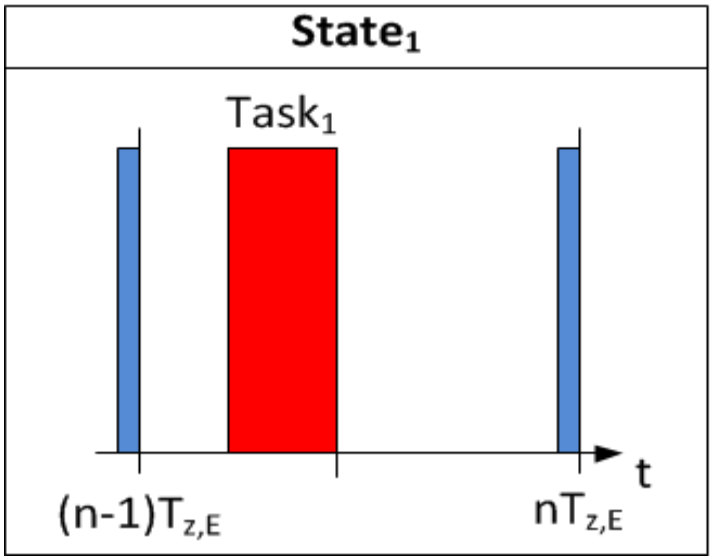
- Zeitsynchrone Einplanung in Phasen mit fester Zyklusdauer wie bei einer SPS.
  - Phase besteht aus Betriebssystemphase und Phase des Anwenderprogramms.
  - Leicht verifizierbar mit Meilensteindiagrammen und Kontrolltabellen!
- **ACTIVATE**-Anweisung wurde dazu speziell angepasst.

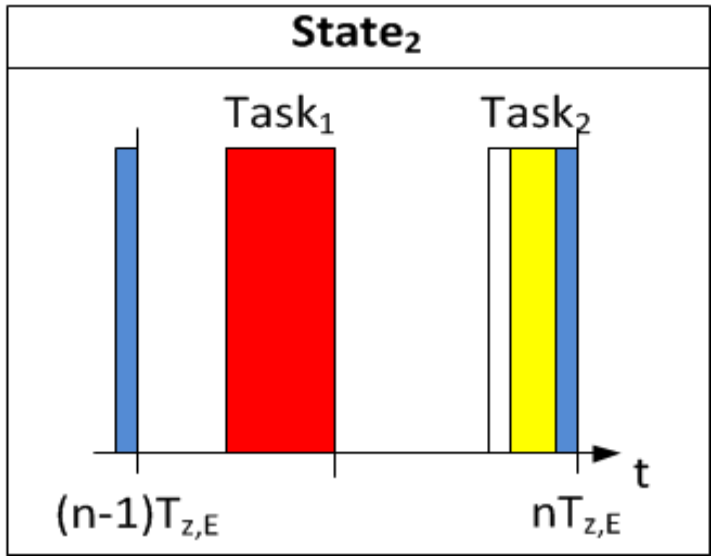
```
SYNCTONEXTSCHEDTICK DUETO 1 MSEC ALL 5 MSEC ACTIVATE Task_1;
SYNCTONEXTSCHEDTICK DUETO 3 MSEC ALL 5 MSEC ACTIVATE Task_2;
SYNCTONEXTSCHEDTICK DUETO 4 MSEC ALL 5 MSEC ACTIVATE Task_3;
```



Schedulingteil

- Einplanung der Tasks muss im Software-Entwurf festgelegt werden.
- Schedulingteil bietet
  - Hilfestellung zur Taskeinplanung (während Software-Entwicklung),
  - Informationen zur Überwachung der Ausführung von Tasks (zur Laufzeit).
- Im Schedulingteil werden Informationen zur Task-Einplanung eingegeben,
  - anhand von Kontrolltabellen und Meilensteindiagrammen (in Listenform).





Schedulingteil-Taskzustandsdiagramm

```
TASKSTATECHECKLIST;
 State_M :
     STATE MAIN( TM.RTIM, NOT T1.RTI1, NOT T2.RTI2, NOT TN.RTI3) VALID;
 State_1 :
     STATE   (NOT TM.RTIM,     T1.RTI1, NOT T2.RTI2, NOT TN.RTI3) VALID;
 State_2 :
     STATE   (NOT TM.RTIM,     T1.RTI1,     T2.RTI2, NOT TN.RTI3) VALID;
 State_3 :
     STATE   (NOT TM.RTIM, NOT T1.RTI1, NOT T2.RTI2,     TN.RTI3) VALID;
 State_4 :
     STATE   (NOT TM.RTIM,     T1.RTI1,     T2.RTI2,     TN.RTI3)
                                                  RAISE Exception_S1;
 State_Other : OTHER RAISE Exception_S2;
END;
```

```
RTINFOLIST T1;
  RTI1 : RTINFO;
    ACTCOND SYNCTONEXTSCHEDTICK DUETO 10 MSEC ALL 20 MSEC;
    MAXRUNTIME BYSYSTEM MSEC;
    NONPREEMPTIVESECTION FROM 0 MSEC TO 10 MSEC;
  END RTI1;
END T1;
```

SGEZ für die Sicherheitsstufe SIL 3 gemäß DIN EN 61508

5/5 ZUSAMMENFASSUNG

Zusammenfassung

- Sicherheitsgerichtete Echtzeitprogrammiersprache definiert, die
  - den Sicherheitsanforderungen nach SIL 3 genügt,
  - ermöglicht die Erstellung von leicht verifizierbaren Programmen und
  - den Einsatz der diversitären Rückwärtsanalyse anhand von Funktionsplänen und Ursache-Wirkungstabellen zur Software-Verifikation.
- Die Programmiersprache ermöglicht
  - Programmierweise mit Funktionsplänen und Ursache-Wirkungstabellen,
  - die Berücksichtigung des Echtzeitbetriebs von Programmen und
  - dessen Überwachung durch Kontrolltabellen und Meilensteindiagrammen.
- Gegenüber PEARL90 ist die Moduldefinition weitgehend geändert worden, diese ermöglicht nun
  - die Kapselung von Daten und Objekten sowie
  - den Schutz vor einer Manipulation von Daten.

Danke für ihre Aufmerksamkeit!

Fragen?