Verlässliche Echtzeitsysteme

Übungen zur Vorlesung

Florian Franzmann, Martin Hoffmann

Friedrich-Alexander-Universität Erlangen-Nürnberg
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)
www4.informatik.uni-erlangen.de

24. Mai 2012

Überblick

Astrée [1]

- Ziel: Nachweis der Abwesenheit von Laufzeitfehlern
- findet alle potentiellen Laufzeitfehler
- leider auch *falsch-positive*
  ⇝ wegen Gödelschem Unvollständigkeitstheorem

  Jedes hinreichend mächtige formale System ist entweder widersprüchlich oder unvollständig.
- Programm ist korrekt, wenn
  - Astrée keine Alarme meldet
  - oder ∀ Alarme nachgewiesen, dass falsch-positiv

Astrée weist nach

- Überschreitung von Array-Grenzen
- Ganzzahldivision durch Null
- ungültige Dereferenzierung, arithmetische Überläufe
- ungültige Gleitkommaoperationen
- dass Code nicht erreichbar ist
- Lesezugriff auf nicht initialisierte Variabeln
- Verletzung benutzerdefinierter Zusicherungen

Was Astrée nicht kann

Astrée kann nicht umgehen mit

- Rekursionen
- dynamischem Speicher

Einschränkungen

Astrée nimmt an, dass als Einschränkungen gelten:

1. der C99-Standard
2. implementierungsabhängiges Verhalten
   - Größe von Datentypen
   - Gleitkommastandard
   - ...
3. benutzerdefinierte Einschränkungen
   - z. B. ob statische Variablen mit 0 initialisiert werden
4. außerdem benutzerspezifizierte Zusicherungen

Benutzerdefinierte Zusicherungen

`__ASTREE_known_fact((B))`

- wird nicht weiter überprüft
- Analyzer warnt, falls `B` nicht wahr werden kann

`__ASTREE_assert((B))`

- Analyzer erzeugt Alarm, falls `B` nicht immer wahr
- `B` kann nicht von der Form `e1 ? e2 : e3` sein

- Analyzer nimmt danach an, dass `B` wahr ist
- einziger Unterschied: wann ein Alarm ausgelöst wird

Beispiel

```
1  /*
2  #include <astree.h> // Astree-Makros abschalten
3  */
4
5  float filter(Alpha_State *s, float val) {
6    __ASTREE_known_fact((val == val));
7    __ASTREE_known_fact((val <= FLT_MAX
8      && val >= FLT_MIN));
9    __ASTREE_known_fact((s->val == s->val));
10   __ASTREE_known_fact((s->val <= FLT_MAX
11     && val >= FLT_MIN));
12   __ASTREE_assert((0 < s->alpha));
13   __ASTREE_assert((s->alpha < 1));
14
15   float residual = val - s->val;
16   s->val = s->val + s->alpha * residual;
17
18   __ASTREE_assert((s->val == s->val));
19   // ...
20   return s->val;
21 }
```

Variable auf „unbekannter Wert“ setzen

`__ASTREE_modifiy((V1, ..., Vn))`

- zeigt an, dass Variablen `V1` bis `Vn` unbekannten Wert haben

⇝ braucht man um Stubs zu bauen

volatile

`__ASTREE_volatile_input((V))`

- zeigt an, dass `V` sich jederzeit ändern kann

`__ASTREE_volatile_input((Vp, r))`

- `p` ist Pfad in der Variablen,
  z. B. `V.a[3-4].b` ⇝ Variable `V`, Arrayelemente `a[3]` und `a[4]`, Struct-Element `b`
  - `[i]` ⇝ Element `i`
  - `[i-j]` ⇝ Elemente `i` bis `j`
  - `[]` ⇝ alle Elemente
- `r` schränkt Wertebereich ein `[i, j]` ⇝ von `i` bis `j`

Analyse untersuchen

`__ASTREE_analysis_log(())`

- gibt Zustand der Analyse an dieser Stelle aus

`__ASTREE_log_vars((V1, ..., Vn))`

- zeigt Zustand der Analyse in Bezug auf einzelne Variabeln an

`__ASTREE_print(("text"))`

- gibt Text aus

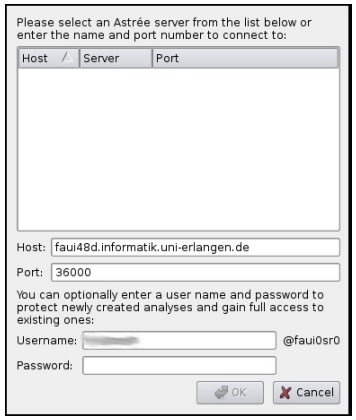
Astrée verwenden

- Astrée ist im CIP installiert und kann folgendermaßen gestartet werden:

  `% /proj/i4ezs/tools/astree/bin/astreec`
- Anmeldung mit Benutzername und Passwort
  ⇝ Passwort wird bei der ersten Anmeldung festgelegt
- Dokumentation unter
  `/proj/i4ezs/tools/astree/share/astree_c/help`

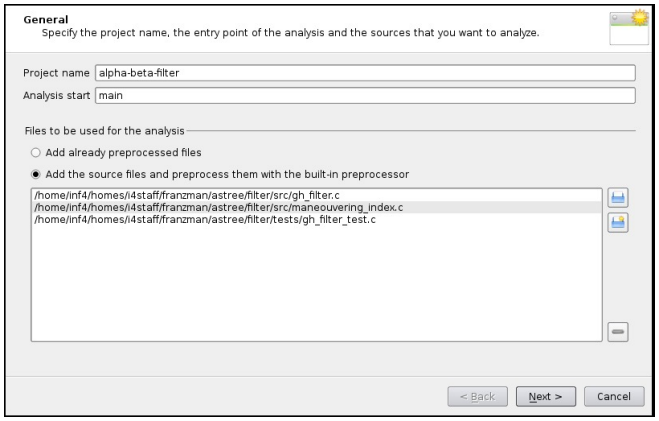
Anmelden

Host	faui48d.informatik.uni-erlangen.de
Port	36000
Username	nach Belieben
Passwort	bei der ersten Anmeldung festlegen und gut merken



Projekt anlegen

- Quelldateien zum Projekt hinzufügen



Präprozessoreinstellungen

- Include-Pfade festlegen

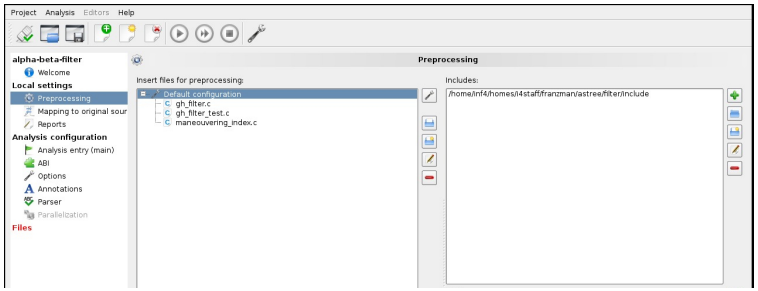


Table of Contents

Aufgabenstellung

- einfaches Filter implementieren
- Korrektheit der Implementierung nachweisen
  ⇝ Astrée
- zunächst mit Gleitkommaarithmetik
  ⇝ `float`
- dann für einen 8 Bit-Mikrocontroller mit Festkommaarithmetik

Festkommaarithmetik – Q-Notation

- kaum ein Mikrocontroller hat eine Gleitkommaeinheit
  ⇝ Festkommaarithmethik mit Ganzzahlen
- Zahlenformat häufig in Q-Notation [7] angegeben
- Q$m.n$ ⇝ Festkommazahl mit
  - m Bit vor dem Komma
  - n nach dem Komma
  - und einem Vorzeichenbit
  - Wertebereich: $[-2^m, 2^m - 2^{-n}]$
  - Auflösung: 2^{-n}

Implementierung als Integer

⇝ welches Q-Format Verwendung findet, ist dem Anwendungsprogrammierer überlassen

Q-Notation – Beziehung zu Gleitkommazahlen

von Gleitkomma nach Q

1. Multiplikation mit 2^n
2. Runden auf die nächste Ganzzahl

von Q nach Gleitkomma

1. Umwandlung in Gleitkommazahl ⇝ cast
2. Multiplikation mit 2^{-n}

Operationen – Addition/Subtraktion

- Addition und Subtraktion wie bei Ganzzahlen

Addition

```
1 int8_t       a = ...;
2 int8_t       b = ...;
3 int8_t result = a + b;
```

Subtraktion

```
1 int8_t       a = ...;
2 int8_t       b = ...;
3 int8_t result = a - b;
```

Operationen – Multiplikation/Division

- braucht Zwischenergebnis von doppelter Bitbreite

Multiplikation

```
1 #define K   (1 << (n - 1))
2 int8_t     a = ...;
3 int8_t     b = ...;
4 int16_t temp = (int16_t) a * (int16_t) b;
5        temp += K;
6 int8_t result= temp >> n;
```

Division

```
1 int8_t      a = ...;
2 int8_t      b = ...;
3 int16_t  temp = (int16_t) a << n;
4         temp += b / 2;
5 int8_t result = temp / b;
```

Größe von Datentypen bestimmen I

- `% cat test.c`

```
1  #include <stddef.h>
2
3  static size_t short_size      = sizeof(short);
4  static size_t ushort_size     = sizeof(unsigned short);
5  static size_t int_size        = sizeof(int);
6  static size_t uint_size       = sizeof(unsigned int);
7  static size_t long_size       = sizeof(long);
8  static size_t ulong_size      = sizeof(unsigned long);
9  static size_t longlong_size   = sizeof(long long);
10 static size_t float_size      = sizeof(float);
11 static size_t double_size     = sizeof(double);
12 static size_t longdouble_size = sizeof(long double);
13 static size_t ptr_size        = sizeof(void *);
14 static size_t fptr_size       = sizeof(void (*)(void));
15 void main(void) {}
```

- `% avr-gcc -O0 -S -c test.c`

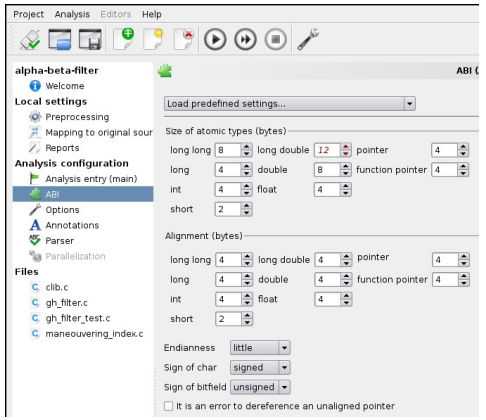
Größe von Datentypen bestimmen II

`% cat test.s`

```
...
1 short_size:
2   .word 2 ← short zwei Byte lang
3 .subsection   2
4   .align 2
5   .type   ushort_size,@object
6   .size   ushort_size,4
...
```

ABI

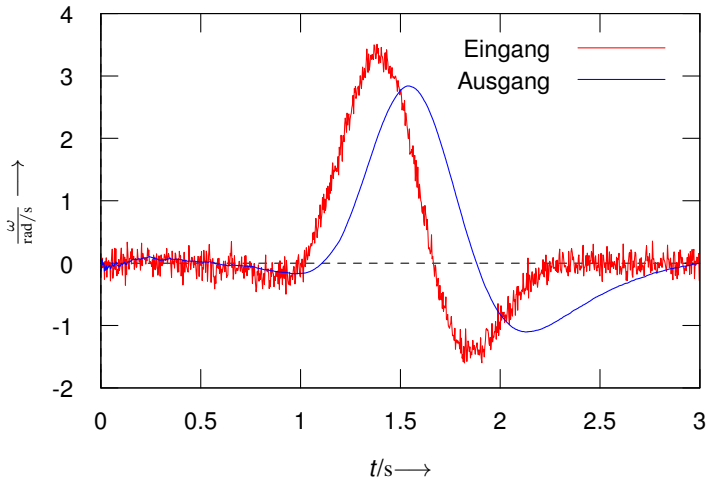
- ABI festlegen



α-β-Filter [3]

- in den 1940ern für die Radarzielverfolgung entwickelt
- in der Literatur manchmal auch *f*-*g*- oder *g*-*h*-Filter genannt
- liefert suboptimale Schätzung für Messgröße
  $\rightsquigarrow$ Rauschunterdrückung
- geeignet zur Schätzung von physikalischen Größen, deren Ableitung nicht 0 ist
  z. B. Position eines Flugzeugs, Lagewinkel ...
- im I4Copter
  - liefern Sensoren häufiger Werte als diese später verarbeitet werden
  $\rightsquigarrow$ α-β-Filter für Ratenwandlung verwendet
  $\rightsquigarrow$ nutzt gewonnene Information vollständig

Beispiel α-β-Filterung



Filteralgorithmus

- wird für jeden Messwert ausgeführt
- $y[\kappa]$: Eingabewert für Abtastschritt κ
- $\hat{x}[\kappa]$: Schätzung der Messgröße zum Abtastschritt κ
- T Abtastintervall, α, β Filterparameter
- Initialisierung: z. B. $\hat{x}[0] = \hat{\dot{x}}[0] = 0$

$$r[\kappa] = y[\kappa] - \hat{x}[\kappa - 1] \qquad \rightsquigarrow \text{Schätzfehler} \qquad (1)$$

$$\hat{\dot{x}}[\kappa] = \hat{\dot{x}}[\kappa - 1] + \frac{\beta}{T} \cdot r[\kappa] \qquad \rightsquigarrow \text{1. Ableitung} \qquad (2)$$

$$\hat{x}[\kappa] = \hat{x}[\kappa - 1] + T \cdot \hat{\dot{x}}[\kappa] + \alpha \cdot r[\kappa] \qquad \rightsquigarrow \text{Schätzwert} \qquad (3)$$

- sinnvolle Werte für α und β?
  - Literatur beschreibt viele Verfahren $\rightsquigarrow$ hier beispielhaft nur eines [5, 4]

Filterparameter

T Abtastintervall
$\rightsquigarrow$ in welchem Zeitabstand gemessen wird

σ_w^2 Prozessvarianz
$\rightsquigarrow$ wie lebhaft der gemessene Prozess ist

σ_v^2 Rauschvarianz
$\rightsquigarrow$ wie verrauscht das Signal ist

$$\lambda = \frac{\sigma_w T^2}{\sigma_v} \qquad \rightsquigarrow \text{Tracking Index} \qquad (4)$$

$$\theta = \frac{4 + \lambda - \sqrt{8\lambda + \lambda^2}}{4} \qquad \rightsquigarrow \text{Dämpfungsparameter} \qquad (5)$$

$$\alpha = 1 - \theta^2 \qquad \rightsquigarrow \text{Gewicht für Wert} \qquad (6)$$

$$\beta = 2(2 - \alpha) - 4\sqrt{1 - \alpha} \qquad \rightsquigarrow \text{Gewicht für Ableitung} \qquad (7)$$

Sinnvolle Wertebereiche

- Erfahrungen mit dem I4Copter haben gezeigt, dass sich die Parameter in folgenden Bereichen bewegen:

  Abtastintervall $T \in (0 \ldots 1]$
  Prozessvarianz $\sigma_w^2 \in [0.5 \ldots 2.0]$
  Rauschvarianz $\sigma_v^2 \in \left[10^{-3} \ldots 10^{-1}\right]$

Korrektheitsbedingung

- Filter ist nur dann korrekt, wenn es auch stabil ist
  - ⇝ für wertbegrenzte Eingabe erfolgt wertbegrenzte Ausgabe [6]
  - ⇝ für Eingabe 0 geht der Filterausgang asymptotisch gegen 0
- α-β-Filter stabil, wenn gilt

$$0 < \alpha \leq 1 \tag{8}$$
$$0 < \beta \leq 2 \tag{9}$$
$$0 < 4 - 2\alpha - \beta \tag{10}$$

- Außerdem: laut [2] Rauschunterdrückung nur dann, wenn

$$0 < \beta < 1 \tag{11}$$

- andernfalls wird das Rauschen verstärkt!

Literatur I

[1] AbsInt Angewandte Informatik GmbH.
The Static Analyzer Astrée, April 2012.

[2] C. Frank Asquith.
Weight selection in first-order linear filters.
Technical report, Army Intertial Guidance and Control Laboratory Center, Redstone Arsenal, Alabama, 1969.

[3] Eli Brookner.
Tracking and Kalman Filtering Made Easy.
Wiley-Interscience, 1st edition, 4 1998.

[4] E. Gray, J. and W. Murray.
A derivation of an analytic expression for the tracking index for the alpha-beta-gamma filter.
IEEE Trans. on Aerospace and Electronic Systems, 29:1064–1065, 1993.

[5] Paul R. Kalata.
The tracking index: A generalized parameter for α-β and α-β-γ target trackers.
IEEE *Transactions on Aerospace and Electronic Systems*, AES-20(2):174–181, mar 1984.

Literatur II

[6] Richard G. Lyons.
Understanding Digital Signal Processing.
Prentice Hall, 3rd edition, 11 2010.

[7] Erick L. Oberstar.
Fixed-point representation & fractional math.
Technical report, Oberstar Consulting, August 2007.

Fragen?