コンパイラの品質と機能安全

～ ユーザーがコンパイラをテストすべき理由 ～

FUJI SETSUBI

www.fuji-setsu.co.jp

Solid Sands

© 2018 - 2021 Solid Sands

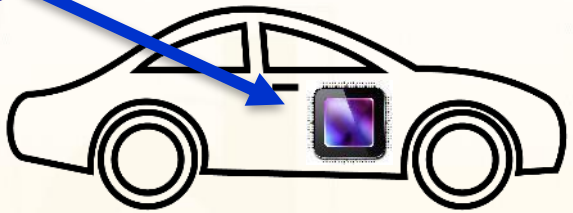
検証作業とコンパイラ品質

コンパイラ品質の重要性

- たった1つのエラーでもシステム全体に大きな影響
- 間違ったコードが生成されることで被る損害は、コンパイラの品質管理への投資と比較にならない

ブレーキに
対する
Cソースコード

C/C++コンパイラ



コンパイラが正しいことの証明には、C/C++言語規格への適合性・正確性・堅牢性を厳密にテストすることが必要

コンパイラ：テストすることの難しさ

- 巨大で非常に複雑
  - C/C++コンパイラのソースコードサイズ：500万行
  - 開発期間40年近く（GCC）、何千人ものエンジニア
  - カバレッジテストなし、MISRA準拠なし、・・・

品質の検証には膨大な条件の組合わせによるテストが必要
• コンパイラメーカーは全オプションの組合せをテストできない

- 開発はいつも進行中
  - エラー修正、改良、機能追加が常にある

1つのステージでの小さな変更が、以降の流れを全く違ったものにすることもあり得る

バージョン3.4が信頼できる
≠ 3.5も信頼できる

SuperTest：言語規格に対するテスト集

- C/C++言語規格への適合性・正確性・堅牢性をチェックする300万件以上のテストプログラム集でコンパイラが正しいことを証明
  - 手書きのテストと（算術演算に対して）生成されたテスト
  - 30年以上の経験の積み重ねで開発
- ポジティブテストとネガティブテスト
  - コード生成機能を評価
  - 診断機能を評価
- ISO C 標準ライブラリの要件ベーステスト

SuperTestのテスト機能

- 最適化の正しさ
- MISRA C:2004, 2012
- C/C++ 標準ライブラリ
- コードジェネレータ開発
- データモデル固有の算術演算
- 呼出し規約


要件トレーサビリティ


ベアメタルのターゲットをサポート


Windows, Linux ホストで動作


マルチプロセスで高速化


柔軟なテストジェネレータ


テスト集合を柔軟に制御


Pythonによる設定プラグイン

だれが使うのか？

コンパイラメーカーやデバイスメーカーが自社製コンパイラのテストやツール認定を目的に活用するなか、コンパイラユーザーによる採用も増えている

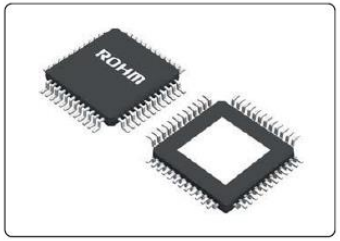
典型的な三つの事例

顧客事例
https://www.fuji-setsu.co.jp/products/SuperTest/#customers

CASECARD ROHM

SuperTest – ローム社製コンパイラ – SuperTestで信頼性を向上

ローム株式会社は、1958年に京都で設立して以来、60年以上の長い歴史をもつ半導体メーカーです。英語表記社名のROHMは、創業製品である抵抗器（Resistor）の頭文字「R」に抵抗値の単位Ω「ohm」を組み合わせたもので、「R」は企業目的である「品質（Reliability）第一」も意味しています。現在は、成長を続ける自動車、産業機器、民生機器市場に向けて、集積回路（IC）、個別半導体、抵抗器やその他の電子部品を設計、製造しており、各市場に適した幅広いソリューションを提供することができます。



その中で、現在主力事業の一つであるIC事業においては、注力する車載や産機機器市場で幅広く採用されるために、ASSPとしてICに柔軟性を持たせることができる組込みプロセッサの開発を必要としていました。これはプロセッサ用のCコンパイラを開発する必要があることも意味しています。これ対して、ローム社が新たに開発したコンパイラは、ClangフロントエンドとLLVMバックエンドに基づいており、ターゲットである組込みソフトウェアが要求する低消費電力かつ省ROMサイズという

Redmineに登録されたコン
結果の新しいソースコードはGi
Jenkinsプロセスフローは
動的に抽出し、ビルドして、
SuperTestを一晩かけて実
結果がチェックされ、新たに
一部としてRedmineに登録さ
の使用を開始する前に、この
いたため、SuperTestをツー
もかかりませんでした。

ローム社のエンジニア大岡
た理由はいくつかあります。一
C言語仕様をカバーし、ISO
テストを実行し、自動検証レオ
です。また、日本国内にSolid
重要なことでした。」と話して

現在、ローム社のコンパイラ
がほぼ完了したため、機能安全
コンパイラツールの認定に必
ています。

大岡氏は続けて「SuperTes
コンパイラの品質を継続的に
が99.9%を超えるようになり

ローム社は、ツールの品質改

CASECARD eSOL

eSOL が提供する安全準拠のリアルタイム組込みソフトウェアプ
支援する SuperTest

現代のセーフティクリティカルな組込みソフトウェアアプリケーションはほとんど、開発者により実装されるアプリケーションコード、標準ライブラリのコンポーネント、そしてリアルタイムオペレーティングシステム（RTOS）の三つから構成されます。eSOL Europe のエンジニアリング担当副社長 Rolland Dudemaine 氏は、この RTOS こそが、自動車用 ISO 26262 のような安全規格への準拠が非常に重要である理由であると考えています。氏は自身のチームが開発・サポートする RTOS のリリースで、堅牢性とコンプライアンスが徹底的にテストされていることを保証する責任者です。だからこそ、eSOL はこの仕事の重要な部分を SuperTest に託したのです。現在、自動車などのセーフティクリティカルアプリケーション向けのマルチコアプロセッサは、単一のチップ上に数十～数百のコアを搭載しており、オペレーティングシステムのコンプライアンスを検証することは、同社の開発において欠かせないものとなっています。

Dudemaine 氏によりますと、「当社ではマルチコア処理がトレンドであることに以前から注目しており、eMCOS と呼ぶオペレーティングシステムを開発しました。これは最大 256、場合によってはそれ以上のコア数のシステム上で動作するように設計され、現在、主に自動車市場のお客様に提供しています。また、Autoware や AUTOSAR Classic Platform、AUTOSAR Adaptive Platform の製品をサポートするチームもあります」とのことです。

eMCOS や AUTOSAR に基づいて安全認定されたオペレーティングシステムとプラットフォームを提供するため、eSOL には完全にテストされた標準 API を提供することが求められており、そこに SuperTest がぴったりとあてはまります。

トフォームベンダーが残りの部分を
いるからです」と言います。

eSOL の eMCOS 開発チームで
ティングシステムを新しくリリース
行し、SuperTest がカバーする多く
ム API が期待どおりに機能してい

「当社では SuperTest を、機能テ
テストスイート、そしてコンプライ
で使用しています。お客様は、eSO
に準拠するオペレーティングシステ
ているからです」と Dudemaine 氏
パートナー企業の中ではすでに Supe
自身で SuperTest を使用してコンプ
ことも多いという利点もあります。
レーティングシステムとライブラリ
ことが、お客様の信頼につながって

「SuperTest には大きな価値が二つ
トの能力そのもの、第二にそれらの
開発されているということです。そ
満たす新しいテストを追加すること
んでした。SuperTest に含まれる C
その一部となり、弊社のオペレーテ
面をカバーする他のテストを追加し

SuperTest のインストール、起動、

Dudemaine 氏によると、「初めて

DENSO

CASECARD DENSO

車載システムの機能安全のリスクを軽減するSuperTest

実はコンパイラを検証する必要があるのは、コンパイラ開発者だけではない。コンパイラによってアプリケーションコードに不測のエラーが混入されることがないように、ソフトウエア開発者であってもコンパイラの品質を意識する必要がある。特にセーフティクリティカルな製品では、その傾向が顕著になる。

株式会社デンソー（最先端のオートモーティブテクノロジー、システムおよびコンポーネントのサプライヤー）も、この問題を解決するコンパイラテストと検証用のパッケージとして、Solid Sands 社の SuperTest を支持している。

コンパイラで実績済みのソースコードを再利用することは、ソフトウエアや製品の品質を維持する最善策のひとつである。しかしながら新たに製品系列を展開する場合に、新しく生成されるソースコードが、コンパイラの欠陥を表面化させる可能性がある。さらにソフトウエア開発担当者ごとで異なる様々なコードスタイルによって、コンパイラの欠陥が浮き彫りにされることもある。また一方、コンパイラに潜在する問題が露呈するのは、新規のソースコードをコンパイルする場合に限らない。

“新しいバージョンのコンパイラを入手するたびに、コンパイルされたコードが旧バージョンと一致することをチェックする必要があります。以前はアセンブラレベルでの手動比較や、機能

またSuperTest は定期的にアップデートされるので、ライブラリコードを採用する際に必要となる検証にも有効である。例えば SuperTest で検出される欠陥には、数学関数の C 言語規格との不一致動作や、異なるコンパイラバージョン、メーカ間でのライブラリのふるまいの違いなどがある。

“コンパイラに潜む問題点が、製品開発途中や製品出荷後に見つかり手戻りが発生する可能性があったことを考えると、コンパイラ入手時に検出できる SuperTest の採用は、大きな投資対効果があったと言えます。”
— 株式会社デンソー 基盤ソフト技術部 中里 弘樹 氏

SuperTest に提供される相当な規模のテストスイートを自前で開発することや、様々なコンパイラの問題を検出するコードサ

SuperTestの典型的な実行結果

C++14に対するG++7.3.0のテスト結果サマリー（テストの一部、約12000件）

	LOG
Compile errors	fail/chck/pass
Language Library	81/ 93/10250 22/ 0/2029
subtotal	103/93/12279
Run errors	fail/chck/pass
Language Library	4/ 0/8026 0/ 0/1934
subtotal	4/ 0/9960
Total	107/ 93/12275

コンパイラエラーの実例(実行時のエラー)

SuperTest テストプログラム(部分)

```
s[0] = 42;
*(sp[0]) = -1;    /* *(sp[0]) is an alias of s[0] */
if( s[0] == 42 )  /* Incorrectly yields true */
```

- このエラーの特徴
  - エイリアス解析に欠陥
  - 最適化を指定しておらず、オプションで制御できない
  - 特定の構文に依存しておらず、回避するのが困難

コンパイラエラーの実例（実行時のエラー）

SuperTest テストプログラム（部分）

```
#define BASE INT_MIN
void s482(double *a, double *b, double *c, int len)
{
    int i;
    for (i = BASE; i < BASE+len; i++) {
        a[i-BASE] = b[i-BASE] + c[i-BASE];
        if (c[i-BASE] > b[i-BASE])
            break;
    }
}
```

- あるコンパイラでは -O2 での実行時にセグメンテーションフォールトが発生

コンパイラユーザーが得た成果

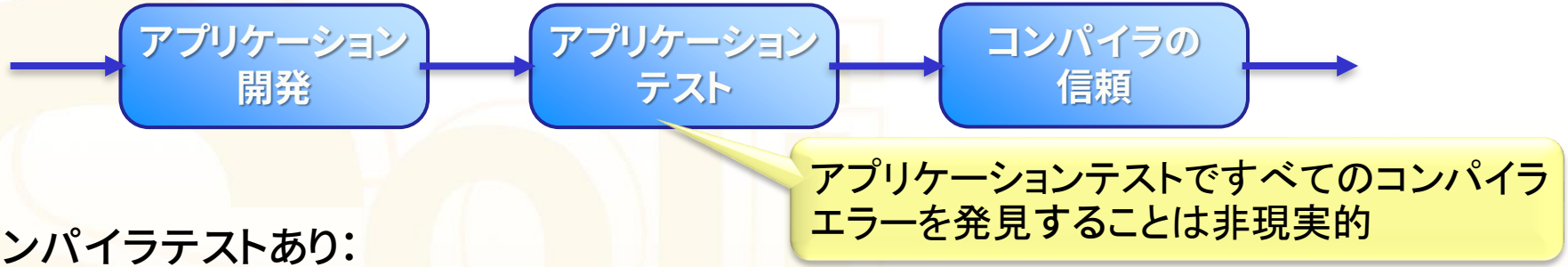
- 同じコンパイラを1000本以上使用
  - バージョンアップごとにチェックし、使ってはいけない最適化などの社内ルールを施行。過去バージョンに戻すような問題が起こっていない
- ライブラリのテスト
  - C99にC89のライブラリが混在し数学関数で問題になるところを事前に回避
- Depthスイートテスト機能
  - アーキテクチャ固有のデータ長（int が24ビット）に合わせて演算のテストができる
- 呼出し規約テスト機能でABI規則への準拠を確認
  - 他のサプライヤーの製品と繋がることによる問題の切り分け

ユーザーがコンパイラをテスト：メリット

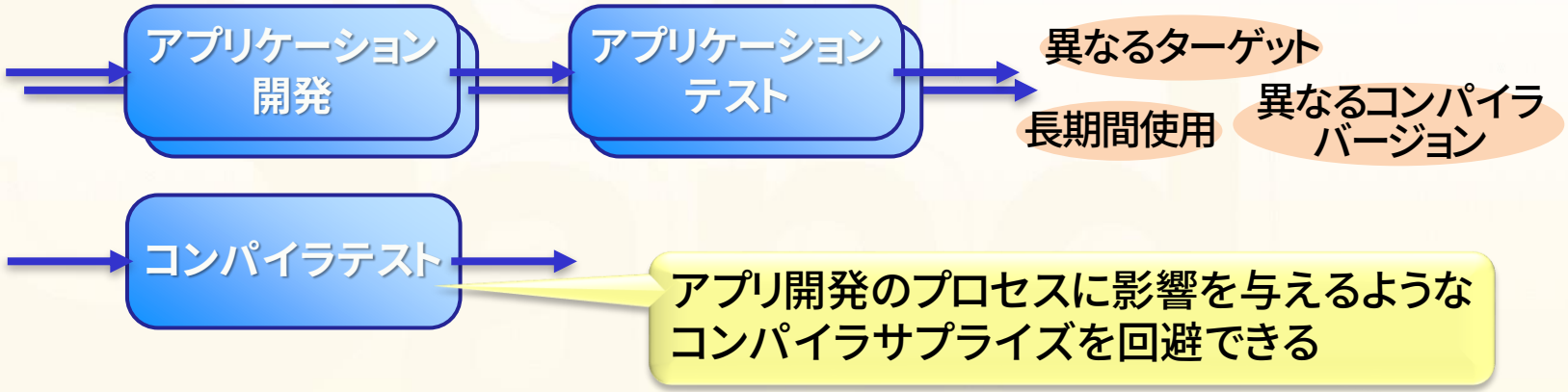
- コンパイラの弱点（欠陥）を事前に回避
  - 新規生成のソースコードや担当者ごとのコードスタイルにより発現する欠陥
- コンパイル済みコードに対するチェックの軽減
  - 新バージョンのコンパイラを入手するたびに、コンパイルされたコードが旧バージョンのものと一致することをチェックしなくてよい
    - 数学関数の動作の規格との不一致や、異なるバージョン・メーカ間でのライブラリの動作の違いなど
- コンパイラの継続的テストで得られること
  - コンパイラ改訂に対して最新を維持
  - コンパイラベンダーに縛られない
  - コンパイルオプションやコンパイル環境適用の自由度

市場投入までの時間を加速

コンパイラテストなし：コンパイラの欠陥が開発途中や製品出荷後に見つかると手戻り発生



コンパイラテストあり：



機能安全とコンパイラ

- 開発に使用するソフトウェア手法・プロセス・ツールチェーンが、関連する機能安全規格に準拠することを実証する責任は開発者
- 一方で、ツールチェーンの重要な部分は開発者のコントロール外であり、コンパイラの検証は重要な課題

バグのないコンパイラはなく、コンパイラの欠陥を事前に知ってそれを回避することも大切

コンパイラの認定

- コンパイラが生成するコードはシステムの一部となるため、「ツール認定」、あるいは「生成されたマシンコードでどんなエラーも発見しうる強力なアプリケーションテスト」による証明が求められる

→ ツール認定の方が現実的

- 言語規格への準拠・正確性・堅牢性を厳密にテストする必要があるが、そのための*市販テストスイートがある*
- コンパイラに欠陥が無いということではなく、開発者が欠陥を避け得るということ

最適化やデバイスのバリアント指定など各種オプションによるコンパイラ
ユーザー固有のユースケースに対して、事前認定は十分ではない

https://www.fuji-setsu.co.jp/files/QualificationBenefitsSuperTest_JP.pdf

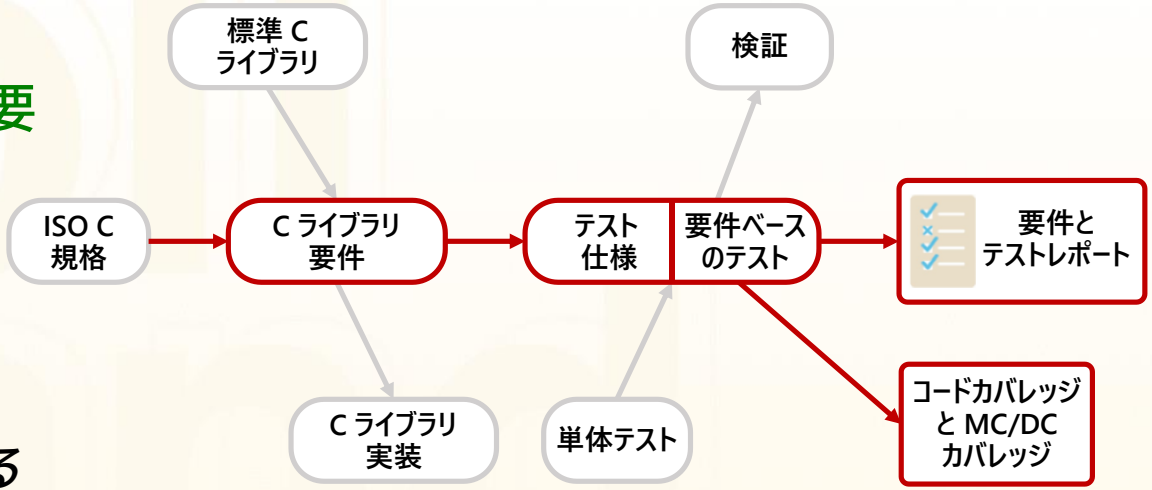
標準ライブラリのコンポーネント認定

- ライブラリのコードはアプリケーションと結合して実行されるため、ここに欠陥があると機能安全が満たせない
- ライブラリの実装が仕様に準拠することの証明が、各機能安全規格に共通して求められる

→ 要件ベーステストが必要

- 要件の分析
- 同値クラスの使用
- 境界値の定義
- 「エラー推測」＝経験

市販テストスイートがある



https://www.fuji-setsu.co.jp/files/SuperGuard_Whitepaper_JP.pdf

まとめ

- 完璧なコンパイラはない
  - ミッションクリティカルなアプリでコンパイラの弱点を知っておくことは重要
  - コンパイラテストはコンパイラのユーザーにとっても非常に有用
- SuperTest は日本を含む世界中で使用
- SuperTest はコンパイラユーザーも容易に使用できる
  - ユーザーでコンパイラをテストすることに多くのメリットあり
- Solid Sands社は、ISO 26262 のような機能安全の領域で専門技術とサービスを提供

各種資料を公開しています

- コンパイラユーザからよくある質問とその答え
  https://www.fuji-setsu.co.jp/files/SuperTestFAQs.pdf
- ポジティブテストによるコンパイラバックエンド（ジェネレータ）のテストについて
  https://www.fuji-setsu.co.jp/files/JP_Code_generator_validation.pdf
- コンパイラのテスト結果からアプリケーションコードの品質を改善する
  https://www.fuji-setsu.co.jp/files/CompilerTestResults.pdf
- コンパイラ最適化の正しさを検証する
  https://www.fuji-setsu.co.jp/files/jp_VerificationofOptimizationCorrectness.pdf

Thank You!

FUJI SETSUBI

www.fuji-setsu.co.jp

Solid Sands

© 2018 - 2021 Solid Sands