平成 19 年度

組込みシステムにおける機能安全に関する調査研究

ネット社会における組込みシステム、2 つの課題

「情報セキュリティ」と「機能安全」

社団法人　組込みシステム技術協会

機能安全委員会

情報セキュリティワーキンググループ

はじめに

本報告書は、組込みシステムの開発にたずさわる人たちを対象に、組込みシステムでもその必要性が叫ばれ始めている「情報セキュリティ」と「機能安全」について、基本的な事柄を検討し報告することを目的としている。

「情報セキュリティ」とは何か、また「機能安全」とは何かは、本論で論ずるとしても、もとより両者は似て非なるものである。本来は別々に論じるものであるが、現場を担う組込みエンジニアにとって両者は等しく対応しなければならない課題である。両者を等しく結びつけるのは、われわれ委員会の討議では品質の方法論か、という議論に至っているが、ともかくも対応しなければならない現実の課題として両者を同じ俎上に乗せた。

またここで「開発にたずさわる人たち」とは、開発のキーマン、予算執行を行う部課長、あるいは、情報セキュリティや機能安全に関して認識を深めようとする経営者（以上、「マネージャ」と総称）を指している。したがって基本的な内容や制度などマネジメントに資するであろう事柄に重きをおいている。詳細な事柄は実際に必要になったときに、あらためて調査するであろうと想定している。その意味で本書はそのガイドブックともいえる。

このようなことから本書の構成は、組込みとは？という原則論から始めることはせず、組込みシステムのことはおおよそわかっているという前提のもとに、社会のネット化の流れを概観し、そのうえで情報セキュリティとはどういうものか、と始めている（第 1 章)。その後、組込みとは？を簡単に整理し（第 2 章)、ソフトウェア、ハードウェアのセキュリティについての技術を紹介している（第 3、4 章)。実際のところはどうなのかがいわゆる“組込み屋”の理解の仕方でもあるからである。その後、情報セキュリティの制度を述べ（第 5 章)、最後に機能安全について述べている（第 6 章)。機能安全に関しては、この 2、3 年先駆的に取り組んでいる企業や機関がみられるが、システムの認証の取得などということになるとまだまだのようである。「機能安全」という用語の認識もまちまちのようである。よって、認証の方法や安全の分析などを紹介するのではなく、機能安全という視点から開発への関わり方に議論の視点をおいている。

なお、この報告書は当委員会のこの 1 年の活動、またその前年の研究会としての活動をもとになされたものである。組込みを専門とする者たちが、情報セキュリティや機能安全という課題に対して、現場感覚をベースに 2 年間の討議や研究のなかから紡ぎ出した成果ととらえていただきたい。学術的な議論からは距離があるが、組込みという現場における実用を第一にしたところに、他と差別可能なもの、とご理解いただければ幸いである。

最後に、日本自転車振興会をはじめ本事業へご協力いただいた各位に感謝を申し上げたい。

ＪＡＳＡ機能安全委員会委員長兼キュリティワーキンググループ長
漆原 憲博

目　　次

第１章　組込みシステムにおける情報セキュリティ問題

近年のマイクロプロセッサの普及と低価格化に伴い、我々が身近に接する多くの商品にマイクロプロセッサが搭載されてソフトウェアで制御されるようになり、さらに利便性を追求した結果、レシピをダウンロードできる電子レンジなどのようにインターネットに接続される商品も増えてきた。

しかし、これらのいわゆる「組込みシステム」はマイクロプロセッサ上で動作する組込みソフトウェアで制御されるので、情報セキュリティ問題を避けては通れない。

一方、利用者側にとって「組込みシステム」は生活に役立つ便利な商品であって、情報機器という認識を持って利用することは少なく、ウイルス対策ソフトウェアやファイアウォールソフトウェアを適宜更新するなどの運用管理を期待することは困難である。

これこそ、「組込みシステムにおいてはパーソナルコンピュータなどの情報機器以上に情報セキュリティへの配慮が必要」な理由である。

情報セキュリティに関する技術の調査・研究や評価・認証制度の運営を行っている経済産業省傘下の独立行政法人情報処理推進機構（IPA）でも早くからこの問題に注目し、2007年５月には「組込みシステムの脅威と対策に関するセキュリティ技術マップの調査報告書」を、2008年１月には「複数の組込み機器の組み合わせに関するセキュリティ調査報告書」を発行して警鐘を鳴らしている。

そこで第１章では、組込みシステムで情報セキュリティ問題が浮上してきた経緯を振り返って問題点を整理し、情報セキュリティに関する基礎的事柄を確認しながら組込みシステムにおけるセキュリティ対策の進め方について検討する。

なお、上記報告書は

http://www.ipa.go.jp/security/products/products.html

からダウンロード可能であるので、一読願いたい。

1.1 国内の情報セキュリティの動向

ウイルスソフトに代表されるサイバーテロ、ウィニーに代表される情報漏洩などの情報セキュリティに関する事件は世界中で日常的に発生するようになり、攻撃と防御のいたちごっこが続いている。

国内に目を転じると、政府は 2005 年から「第 1 次情報セキュリティ基本計画」の検討を進め、「政府機関の情報セキュリティ対策のための統一基準」を定めて 2006 年から本格的に情報セキュリティ対策に取り組んでいる。

毎年 2 月 2 日が「情報セキュリティの日」として制定され、この日をはさんで情報セキュリティに関する国民の意識を向上させる目的で、全国でさまざまな行事が催されている。ちなみに 2008 年は 1 月 26 日（土）～3 月 2 日（日）の期間に全国の都道府県で 593 件の情報セキュリティ関連行事が開催される予定である。

また、「政府機関の情報セキュリティ対策のための統一基準」は「重要インフラ」としての情報通信、金融、航空、鉄道、電力、ガス、水道、医療、物流の基幹産業分野に焦点をあてており、これらの産業分野では産官をあげてその対策を進めている。

他方市民レベルにおいては、個人情報保護法の登場で個人においても情報に対するセキュリティの意識はかなり浸透したと思われるが、一部では過剰な反応も無いわけではない。

いずれにしろ、これらの産官民をあげての情報セキュリティに対する意識の向上が、情報の安全を求める社会の前提になる。引きも切らない迷惑メールへの法的対応など、現行の制度の不具合を改善しながら、情報セキュリティを健全に保持する社会へと進展することが期待されている。

1.2 「組込みシステム」における情報セキュリティ問題の登場

組込みシステムに情報セキュリティの問題が登場するようになった原因は、単にインターネットが普及したからではない。インターネットの利便性を組込みシステム側が積極的に取り込もうとしたからに他ならない。

表 1.1 にこのあたりのことを整理してみる。

インターネット	エンタプライズ系	組込み系		
		状況	環境別用途	個々の用途
登場以前 ↓	データ・メディアの外部からの持ち込みによるウィルス感染。問題はおおきな社会問にはいたらず。	情報セキュリティは問題とされず		
登場 ↓	→情報セキュリティの社会問題化：職場・家庭でのパソコンのウィルス感染等。			
普及拡大	セキュリティ対策センターなどを社会レベルで設立対応	→組込みシステムのネット利用の増加。 情報セキュリティ問題が顕在化。	個人の利用	携帯電話
				音楽ダウンロード機
				万歩計
			家庭：外出先からの自室のコントロール	部屋の冷暖房
				電気ポットの電源オン・オフ
				お風呂の沸かし
				冷蔵庫の在庫管理
			社会システムの利用	クレジットカード
				乗車パスカード
				ATM
				パスポート
			自動車	カーナビ
				ETCカード
			職場での利用	複合機
				ファックス
				プリンタ
			物流	FRID

表 1.1　インターネットの普及に伴うネットを利用する組込系製品の登場

1.2.1　インターネット登場以前

統計データを待つまでもなく、インターネットが登場する以前の情報セキュリティの問題は、外部から持ち込んだフロッピーディスクなどが何らかの事情でウイルスに感染しており、この感染したメディアを自らのパソコンにかけたところ自分のパソコンも感染してしまった、というものであった。情報セキュリティ問題としては、このような感染などの問題よりも顧客等の営業情報が盗まれるなどの事件の方が多かった。

しかしいずれの事件もその影響は限定的であり、今日のような大きな問題にはならなかった。

組込みシステム（当時は制御システムあるいはマイコンシステムと呼ばれていた）ではソフトウェアは ROM 化され、特別に“情報資産”を内部に持つなどということはなく、当然ながら情報セキュリティが問題になることはなかった。ROM 化された製品用ソフトウェアがディスアセンブルされて他に盗用されるというよう事件はあったが、「情報セキュリティ」事件という形では認識されず、産業スパイということで問題になっていた。

1.2.2　インターネット登場

「情報セキュリティ」というコトバが広く流布するようになったのはインターネットが登場してからである。職場や家庭でのパソコンがウイルスに冒され、あるいは政府機関のホームページなどが広く侵害され機能不全になったことも記憶に新しい。これらの事件の発生により、情報セキュリティは一躍社会全体で対応すべき問題という認識が持たれ始めた。

組込みシステム分野はこの時点においても、一部の専門家をのぞいては情報セキュリティの対象とは見なされていなかった。インターネットに接続された機器はほとんどなかったからである。すでに多くの組込みシステムが多く出回っていたが、たとえば今日では当たり前のゲーム機もいまだインターネットには接続されず、単独で稼働するものであった。

銀行の ATM などはインターネットの登場以前から専用線で結ばれた組込みシステムであり、これらの分野では様々の事件は散見されたが、専用線であることをもって事件への対応は関係機関以外には拡散せず、社会一般に共通する課題と目されることはなかった。

1.2.3　インターネットの普及拡大

インターネットの普及とその拡大は、組込みシステムにおいてもネットとの接続による利便性を追求する契機となった。かつてのプリンタやコピー機はファックスやスキャナーなどとも一体化されてネット接続され、ユーザーは高度なプリンティングサービスを得ることができるようになった。

表 1.1 に示すように、個人利用としての携帯電話、音楽ダウンロード機器は無線でネットに接続可能である。また健康管理のための万歩計も最近では歩数データをパソコンに自動的に取り込み、ネットを介してデータを全国のウォーキング愛好家と情報交換できるようになっているシステムもある。万歩計が直接ネットにつながる訳ではないが、メーカーは間接的にではあれ、ネットの利便を活用しようとしていることの証左である。この万歩計は情報セキュリティのリスクからは逃れることができそうであるが、ネットでデータを万歩計側に入れて、何らかのサービスをすることになったとしたならば（何かよい利便に気がついて)、情報セキュリティの問題がにわかに登場してくる。

家庭においては、外出先からネットを経由した自家の冷暖房や電気ポットの電源オン・オフや風呂を沸かすこと、冷蔵庫の在庫管理なども可能となってきており、独身者や共働きの夫婦などにとって大変便利な機能であると思われる。

クレジットカード、交通カード、キャッシュカードなども専用回線のみならず、インターネットでも利用されている。

自動車におけるカーナビ、ETC カード、物流での RFID、いずれもネットを介することでその利便性を向上させている。

組込み系システムのソフトウェア上の不具合やバージョンアップもネットを介して行うことが常態化している。

エンタープライズ系システムの情報セキュリティ問題に対処するため IPA にセキュリティセンターが設けられ、我が国の情報セキュリティ問題に関する情報の一元化と対策推進の方策が打たれている。インターネットの普及拡大は、単に一国家のみを単位とするものではなく、国家を超え、地球全体をネット網として、どこへでもアクセスできることを最大の特徴としている。

ここに至り、国防関係の情報システムを先頭に、金融や政府やガス・水道など市民生活のインフラにかかわるシステムなどが、明確なセキュリティ基準を設けて、システムの開発や、そのシステムに接続する複合機などの組込みシステムを調達するようになりつつある。基幹システムの情報セキュリティ対策が会社等の組織の信用に直結する問題ともなり、この数年、様々のセキュリティ制度の構築、教育、制度運用者の育成が行われてきている。

この地球レベルの網に、さらに、以上に見たような組込みシステムが接続されることになる。すでに想像できるように、ネット接続を行う最近の組込みシステムは、携帯電話に見られるような個人情報等の情報資産が豊富である。情報セキュリティの問題を組込みシステムも負わざるを得ないゆえんである。

ネットとの接続は、組込みシステムが蓄える顧客データなどの情報資産、また組込みシステムが持っている製品自身の制御等の技術情報（ソフトウェア）という情報資産を、ネットを介して盗まれる、改竄される、壊されるなどのリスクにさらすことになる。盗まれも、改竄もされず、壊されもしない家電製品が、単にネットにつながっていたというだけで、情報セキュリティに対する攻撃の「踏み台」にされ、他のネットに接続されていた PC のデータを盗んでいたなどの事例もある。

表 1.1 の組込み機器の事例はごく身近にみることのできるものばかりである。それ以外に直接われわれの目には触れないが、CPU で制御される組込みシステムは数しれない。

1.2.4　ネット非接続組込みシステムの情報セキュリティ・リスク

インターネットに接続しないからといって情報セキュリティ対応が不要なわけではない。かつて、フロッピーディスクなどを介したウイルスへの感染が問題となっていたことも思い出して欲しい。最近では USB メモリを介して感染するウイルスの存在も報告されている。

また、可搬メディアによるデータの外部への漏洩に関する問題も認識されつつあり、USB メモリなどでもデータの暗号化をするものが多くなっている。

また自身の情報を守るのみではなく、自らの USB などのメディアから（ネットを介して）他のシステムに害を及ぼす可能性も認識されつつあると思われる。

1.3 組込みソフト業界の社会的責任と守るべき情報資産

このような中で、情報システムや製品の開発にたずさわる企業にとっては、「安全な商品を供給する」という社会的責任も大きくなることが予想される。機器開発を主とする組込みシステムもネットに接続された途端に、情報セキュリティに関する攻撃にさらされることはすでにみてきた通りである。

社会的責任は一次的には製品を供給するセットメーカーが取ることになろうが、その開発を請け負うことになる多くのシステム開発会社はこの課題に関してどのような対応をすることになるのか。

ごく一般的な対応はいままでそうであったように開発の契約書にその責任の範囲が記載されることになろうが、その契約書を交わす、すなわちシステムの請負に至るための条件として、開発会社に対して情報資産を守るに足る環境が整っているのかが問われると想像される。少なくともこのような環境整備が開発会社の契約を有利に運ぶことは想像にかたくない。政府の施策は後述するようにこのような方向付けをすでにしている。

情報セキュリティに関する環境整備を第三者的に認証する制度が後述する ISMS である。ISMS とは Information Security Management System の略称であり、国際標準規格として ISO/IEC27001 で定義されている、組織に関する管理手法である。

管理の対象を組織の単位とし（会社全体でなく開発部門などと限定可能)、組織の受けるであろう脅威とそのリスクを分析し、それに対する管理策をルール化し、組織全体のセキュリティを守ることと、組織を構成する全員のセキュリティに対する意識向上をはかることを目的としたマネージメントシステムである。

実際の運用では、PDCA サイクルを回し、運用が単なる文書に終わらないことを義務づけている。

政府の施策にある「重要インフラ」においては、システムや機器の開発を外部に委託する場合には ISMS の認証を取得した、情報管理体制が取れている企業に発注することを推奨している。

たとえば、ネットバンキング等不特定多数のユーザーを抱えるシステムにサイバーテロに攻撃されるような脆弱性が存在していたとすると、これは重要な問題となるであろう。一次請け、二次請け、三次請けと段階が増える都度、その認識は薄れていくものと考えられる。

基幹システムだけでなく一般家庭や個人レベルにおいても、ネット接続機器を利用していることはすでに見たとおりである。携帯電話、テレビ、DVD はもとより、電気炊飯器、風呂の制御までもがネットワークを通じて行えるようになってきている。これら情報家電にも情報セキュリティ機能は当然組み込まれ、使用者はその利便に預かっているはずであるが、事故も起こっている。機器製造元であるメーカーおよびその開発を現場で担う組込

みソフト業界にとってはその取り組みは社会的要請である。

1.4 情報セキュリティの基礎

情報セキュリティの定義として『“情報”の機密性、完全性、可用性を保つこと』が良く使われている。

機密性とは情報の取り扱いを正当な権利を持った人のみに限定することであり、完全性とは情報が不当に改竄されないということを意味している。また、可用性とは、正当な権利者には何時でも情報の取り扱いを可能にするということである。

言葉の意味は何となく理解できるが、さて、組込みシステムに適用しようとした場合、具体的には何をどうすればよいのだろうか。これを考えるためには、先ず、一般的な情報セキュリティの基礎を理解しておく必要がある。

1.4.1 情報資産

情報セキュリティを実現するためにとる方策を“情報セキュリティ対策”と呼ぶ。情報セキュリティ対策を検討する場合に最初に行わなければならないのは、“情報”の洗い出しと定義である。

『“情報”の機密性、完全性、可用性を保つこと』という場合の“情報”とは全ての情報ではなく『守るべき価値を持った、定義された情報』を意味しており、“情報資産”という言葉で表現されることも多い。

“情報資産”としては、一般的には銀行口座の残高など経済的価値を示す情報や、これを利用するために必要となる口座番号と暗証番号、あるいは認証鍵、暗号鍵などが思い浮かぶ。

また誤動作すると大事故につながるような一部の制御システムでは、制御ソフトウェアのコード自身も“情報資産”として守る必要があると考えられる。

個々の情報システム、個々の製品ごとに“情報資産”は異なるのが当然であり、ユーザーの使い方によって異なる場合も多いので、情報セキュリティの問題を検討する際には最初に“情報資産”として扱うべき情報を洗い出し、きちんと定義することからはじめる必要がある。

先ず、検討の対象となっている情報システムや製品のなかに存在する情報をリストアップし、その情報の所有者、作成者、ユーザー等の関与者をきめる。ここで情報の所有者というのは情報の作成者と同一では無いことを理解しておいて欲しい。情報の管理は全て所有者の責任において行われ、盗聴や改竄などが発生した場合には所有者が責任を負うということである。情報の作成者には情報の書き込みが許可されているが、読み出し、更新

が許可されるかどうかはシステムごとに変わるであろうし、情報のユーザーには読み出しは許可されるが、書き込みや更新は許可されない場合もある。

1.4.2　情報資産の移動過程

情報資産が明確になったら、次は情報資産の移動過程を洗い出す必要がある。情報資産は情報システムや製品の中で作り出されることもあるが、多くは外部で生成された後に情報システムや製品に取り込まれて適切な情報セキュリティ機能の保護下に置かれるわけであり、その途中で漏洩や改竄が発生すれば情報セキュリティ機能を持つ意味が無くなってしまう。

図 1.1 に示す銀行の入出金の例を見ると、現金と入金伝票を窓口のテラーに渡して確認を受けてから責任者の承認を受けて端末オペレータに渡すまでは銀行の職員の行動であり、情報システムのセキュリティ機能で守ることは出来ない。従って、職員の服務規程で守るしかないのはおわかり戴けるであろう。

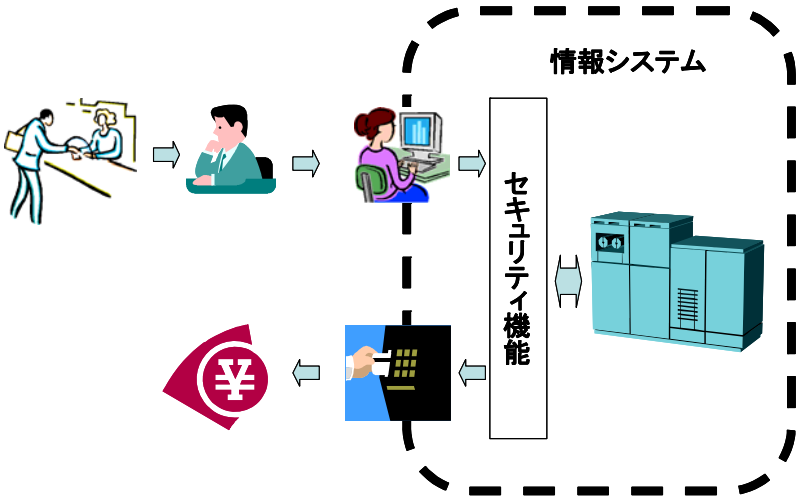


図 1.1　銀行での入出金事例

情報セキュリティ機能で守ることが出来るのは端末オペレータによって入金情報が入力された後である。

また、ATM での出金の場合は顧客が直接 ATM を操作するので、ATM と情報システムの中での出金情報は情報セキュリティ機能で守られる。

ただし、暗証番号を入力する際の指の動きなどは情報セキュリティ機能では守れないので、ATM の設置環境を工夫して盗み見られないようにしているわけである。

このように、情報資産が生成されてから情報システムや製品の中の保管場所に格納されるまでの移動過程と、保管場所から取り出して情報資産を利用する際の移動過程を明確に定義し、保管場所および移動過程における漏洩や改竄を防止するための適切な情報セキュリティ対策を行わなくてはならない。

このためには情報システムや製品の内部における情報セキュリティ機能のような技術的対策だけではなく、情報資産の生成および移動過程に関与する人間の行動やネットワークおよび接続先の外部機器を規制するといった運営管理が必要である。

1.4.3　情報セキュリティ対策

このように考えると、情報セキュリティ対策は大まかには図 1.2 のように分類できる。

(A)：環境における運用管理による対策とは、上記銀行の例では、適切な服務規程と職員の規程遵守を担保するための運用規則などを示す。

(B)：環境における技術的対策とは、例えば ATM の事例でいえば暗証番号入力時の指の動きを盗み見られないように隔壁を設け、後続の人を離れて待たせるためのロープを設置することなどである。また、コンピュータセンターなどでは

・コンクリートの外壁の厚さを一般の建物より増やす。

・窓を少なくする、あるいは設けない。

・建物への入退館時に ID カードによって開錠する。

・各室への入退室も ID カードによって限定し、開錠する。

・ID カードごとに入退館や入退室時間を自動的に記録しておく。

などの対策がとられている。

(C)：情報システムや製品自身の運用管理による対策とは、システムや製品の情報セキュリティ機能を適切に使いこなすために必要な管理機能の設定や使用上の留意点などのことである。情報システムや製品の管理者、ユーザーの理解と遵守が無ければ機能しないため、使用説明書などに明記しておくことが求められる。

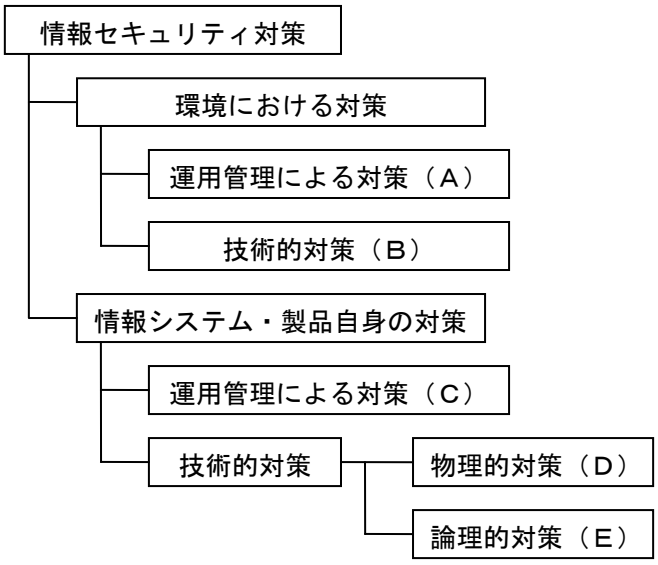


図 1.2　情報セキュリティ対策の区分

(D)：技術的で物理的対策は主として情報システムや製品のハードウェアにより実現されるもので、実際に起こった情報セキュリティ事故への対策をまとめた経験則的なものが多い。

・電子機器に規定外の電圧の電源を供給したり周波数やデューティーを規定外の値にしたクロックを供給したりして故意にハードウェアを誤動作させ、その結果として情報を漏洩させる攻撃手法に対抗するために、供給電圧やクロック周波数、クロックのデューティーが規定値を外れたことを検知して CPU を停止させる等の対策が実

施されている。

・電子機器の動作に伴い変化する、外部から観測可能な現象（消費電力の変化、電磁放射の変化、レスポンスタイムの変化など）を精密に観測し、内部動作状況を推定する攻撃に対抗し、動作状況と現象の変化が直接的に関連しないよう、ソフトウェアのダイナミックステップ数を調整するなどの対策がとられている。これは物理的攻撃方法に対するソフトウェア対策の事例である。

・一部の端末機器では端末機器自体を盗んで分解し、メモリを読み出す攻撃が行われたが、これに対しては端末を分解しようとする試みを検知してメモリ内の秘匿すべき情報を消去するといった対策がとられている。

(E)：技術的で論理的な対策は情報システムや製品のハードウェア機能とソフトウェア機能によって実現されるが、抽象的なものなので以下の簡単なモデルで説明する。

情報を守るためには、守るべき情報を安全な保管場所に格納することが基本である。つまり、情報の金庫を作る訳である。金庫を開けるにはダイアル番号と鍵が必要である。情報金庫の場合は本人確認とアクセス制御でこの機能を実現する。このモデルを図 1.3 に示す。

情報 m のアクセス権限を持っている A 氏は情報システムにログインする時点で ID とパスワードを入力し、情報セキュリティ機能の一つである本人確認機能のチェックを受ける。

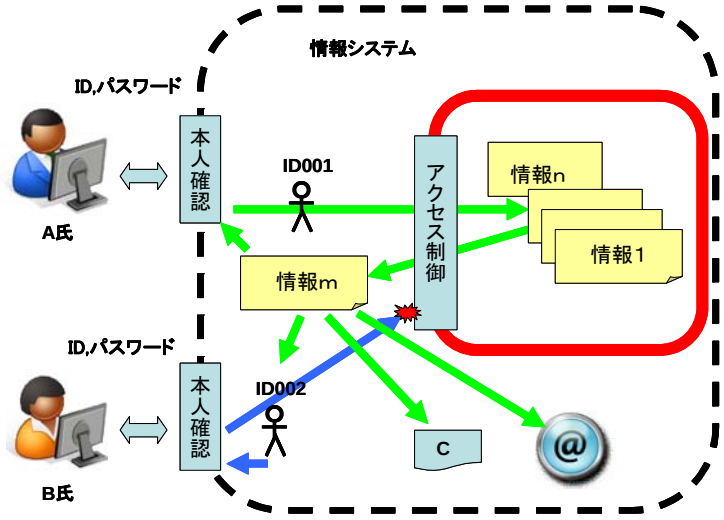


図 1.3　アクセス制御モデル

このチェックをパスすると情報システムに登録されているユーザーと認識されて、A 氏の情報システム内の代理人である ID001 が呼び出される。

ID001 は A 氏に許されている権限情報を持っており、権限の許す範囲で A 氏の指示する行動を実行する。今回は、情報 m を読み出すことである。ID001 は情報セキュリティ機能の一つであるアクセス制御機能のチェックを受けるが、アクセス権限を持っているのでチェックをパスし、情報 m を A 氏のディスプレイに表示する。

一方、情報 m に対するアクセス権限を待たない B 氏の代理人である ID002 が情報 m を読み出そうとしてもアクセス権限が無いのでアクセス制御機能のチェックをパスしない。ID002 は自分では情報 m を読み出すことは出来ないことになる。しかし、A 氏が B 氏に情報 m を転送するように指示すれば、ID001 は情報 m を ID002 に渡し、ID002 はこの情報を B 氏

のディスプレイに表示する。

このように、アクセス制御とは情報の格納場所からの出し入れの時点で制御するという考え方である。アクセス権限を、読み出し、書き込み、更新に分けて細かく制御する場合もある。通常、我々が接する情報機器や情報システムはアクセス制御を基本としているものが多く、情報セキュリティ機能の基本は本人確認とアクセス制御であるという言い方もできる。

しかしアクセス制御では、一旦、格納場所から取り出された情報は自由に加工し、何処にでも送ることができることになってしまう。A 氏の誤認によりアクセス権限の無い B 氏に情報 m を転送することを防ぐことは困難である。

このような問題に対処するためには“フロー制御”という考え方がある。これは情報に取り扱いを制御するためのタグを付けて情報の流れ自体をコントロールしようというものであるが、一般的には使われていないので説明は省略する。

このように、情報システムや製品に本人確認機能やアクセス制御機能などを正しく装備させることが論理的対策の基本である。

なお、論理的対策が効果を発揮するには、基盤となるハードウェアが正しく動作するという前提条件が必要であることも留意しておきたい。

1.5 攻撃と情報セキュリティ対策

図 1.2 に情報セキュリティ対策の大まかな区分を示した。ここで注意しなくてはならないのは、攻撃と情報セキュリティ対策は一対一に対応するものではないということである。

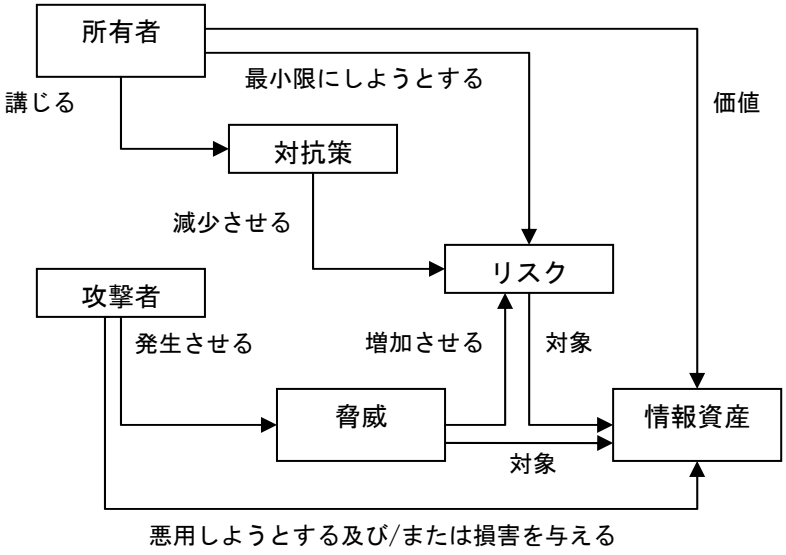


図 1.4　セキュリティの概念と関係　（出典：ＣＣver3.1）

攻撃が成功した場合に起こる可能性の有る情報資産の所有者にとって不都合な事柄を“脅威”と呼び、脅威が実現した場合の資産への影響を“リスク”と呼ぶが、情報セキュリティ対策の考え方の基本は直接的に“攻撃を食い止める”のではなく、脅威発生の“リスクを減少させる”あるいは“最小限にする”という考え方にたっている。この関係を図 1.4 に示す

なお、図 1.4 では脅威を発生させる主体を“攻撃者”と表記したが、出典元である Common Criteria（CC）では、より厳密に“脅威エージェント”と表記している。これは、脅威を発生させる主体はハッカーや悪意のある利用者だけではなく、不注意な管理者や利用者などが脅威を発生させる場合があることを示す表現である。

さて、CC では

・対象となる資産を保護することは、それらの資産の価値を認識している所有者の責任である。実在するまたは想定される攻撃者（脅威エージェント）もまたその資産の価値を認識しており、所有者の利益に反する形で資産を悪用しようとすることがある。攻撃者（脅威エージェント）の例には、ハッカー、悪意のある利用者、(誤りを犯すことがある)悪意のない利用者、コンピュータ処理、及び事故などがある。
・資産の所有者は、そうした脅威を、所有者にとっての資産の価値が減少することになるような資産の侵害の可能性と捉えるであろう。一般に、セキュリティ固有の侵害には、資産の機密性の損失、資産の完全性の損失、及び資産の可用性の損失などがあるが、これらだけではない。
・したがって、このような脅威が実現する可能性と、その場合の資産への影響に基づいて、

脅威から資産に対するリスクが生じる。その後、資産へのリスクを減らすために、対抗策が講じられる。対抗策は、IT 対抗策(ファイアウォール及びスマートカードなど)と非 IT の対抗策(警備及び手続きなど)から構成されることがある。

と説明されている。

概ね、CC で IT 対抗策と表現しているものが情報システム・製品自身の技術的対策に、非 IT の対抗策と表現しているものが環境やシステム・製品自身の運用管理による情報セキュリティ対策に相当する。

多少わかりにくい言い方になってしまったが、大まかに表現すれば物理的攻撃に対して論理的対策によって対抗する場合もあり、物理的対抗策と運用管理を併用する場合もあるということである。

情報セキュリティ対策は最小限のコストで必要な対抗策を講ずることを目的としており、情報システムや製品全体とそれを取り巻く環境（制御できる範囲）で適切な役割分担を行うことによって実現されるものである。

また、悪意ある脅威エージェントである攻撃者は攻撃が失敗に終わればその結果を分析し、別の手段で攻撃を試みるものと予想される。

従って、情報セキュリティ対策は恒久的なものとは成り得ず、常時見直すことが必要である。

1.6 ライフサイクルと情報セキュリティ対策

ところで、前述の情報セキュリティ対策の詳細な内容が情報の盗聴や改竄を企てている“攻撃者”に知られると、攻撃者が情報セキュリティ対策をかいくぐった新たな手法を考案することが容易になり、攻撃を成功させる可能性が高くなる事が予想される。

攻撃が成功すれば原因を分析してセキュリティ対策の修正を行わねばならないが、特に技術的なセキュリティ対策を修正するためには多大な時間とコストが必要となる。

従って、1.4.1 に既述した情報システムや製品の中に存在する情報資産だけでなく、設計開発や製造にかかわる情報（文書、図面類など）も厳重に守る必要がある。つまり、開発サイトの情報セキュリティ対策が必要となる。

また、廃棄された製品などを分解して内部構造やメモリーデータを分析し、技術的なセキュリティ対策の詳細を探り出した事例もあるし、盗難製品を利用した不正改造品や新たに正規品に似せて製造したものを使用中の正規の製品とすり替えることによって情報セキュリティ対策を無効にし、情報の漏洩や改竄を行うといった攻撃事例も見受けられる。

製品の盗難自体を防ぐのは情報セキュリティ対策の範囲外だが、盗難による情報流出を最小限にする対策は必要であり、事前に対処しておく必要がある。

このように考えると、情報セキュリティ対策は情報システムや製品の使用期間のみを対

象とするのではなく、設計、製造、使用、廃棄というライフサイクルの全ての工程を対象として検討し、実施しなくてはならないことは当然のことである。

1.7 セットメーカーの責務

1.5 では、情報セキュリティ対策は情報システムや製品全体とそれを取り巻く環境（制御できる範囲）で適切な役割分担を行うことによって実現されるものであると述べた。

また 1.6 では、情報セキュリティ対策は情報システムや製品のライフサイクルの全ての工程を対象として検討し、実施しなくてはならないことを説明した。

このことから、組込みシステムの開発において情報セキュリティ対策は最終製品をまとめるセットメーカーが第一に検討すべき問題であり、対象製品で必要となる情報セキュリティ機能の検討から始め、製品のライフサイクルの検討、製品廃棄時の残存情報の保護、自らの開発サイトの情報セキュリティ対策等を充分検討する必要があると考えられる。

また、組込みソフトウェアの修正のためにネットワークを介してダウンロードを行うなら、その場合のダウンロード先の正当性検証についても検討しておかなければならない。

これらの検討の結果、ソフトウェアでの対策が必要だと判断された事項をソフトウェアに対するセキュリティ機能の要求仕様書として提示することが基本であろう。

また、組込みソフトウェア開発サイトの情報セキュリティ対策も必要な場合には、調達仕様として明確に要求するべきである。

セットメーカーからソフトウェア開発企業に対して“セキュリティ・ホールが無いこと”という要求仕様が出てくるといった冗談のような話も聞くが、情報セキュリティの本質がわかっていればこのような要求は出せるはずがない。

情報セキュリティの基本を理解していないのであれば、情報セキュリティ機能を有する機器を市場に出荷することはきわめて無責任な行為であることを改めて認識しておきたい。

・情報セキュリティに対する意識の向上が、情報の安全を求める社会の前提になる。
・インターネット登場以前における情報セキュリティ問題は、営業情報や技術情報が盗まれるという観点から「産業スパイ」と認識されることが多かった。
・インターネット登場後、ウイルス感染の拡大や政府機関のホームページ侵害などの事件によって情報セキュリティに対する認識が高まったが、組込みシステム分野での重要性はあまり認識されていなかった
・インターネットの普及拡大に伴って、利便性追求のため組込みシステムが積極的にインターネットに接続された結果、情報セキュリティ・リスクに曝されるようになった。
・USB メモリなどの可搬メディアを介したウイルス感染なども発生しており、ネット非接続組込みシステムであっても情報セキュリティ問題を避けることはできない。
・「安全な商品を供給する」という社会的責任を全うするため、組込みソフトウェア業界においても情報セキュリティに関する環境整備を推進する必要がある。
・情報セキュリティ対策の基本は情報資産の洗い出しである。
・情報資産は、生成から保管、使用の全ての移動過程において守らなければならない。
・情報セキュリティ対策には環境による対策と情報システム・製品自身の対策がある。
・情報システム・製品自身のセキュリティ対策の基本は、本人確認とアクセス制御である。
・情報セキュリティ対策は“攻撃を食い止める”のではなく、脅威発生の“リスクを減少させる”あるいは“最小限にする”という考え方にたっている。
・情報セキュリティ対策は情報システムや製品の使用期間のみを対象とするのではなく、設計、製造、使用、廃棄というライフサイクルの全ての工程を対象として検討し、実施しなくてはならない。
・組込みシステムの開発において情報セキュリティ対策は最終製品をまとめるセットメーカーが第一に検討すべき問題である。

第 2 章　組込みシステムの現状

産業機器や家電製品などに内蔵され、特定の機能を実現するコンピュータシステムを組込みシステムと呼び、このコンピュータシステムに組込まれて使用するソフトウェアのことを組込みソフトウェアと呼ぶ。自動車・工作機器・デジタル家電など日本の強みである製品のほとんどは組込みシステムである。

これらの製品はソフトウェア制御であるがゆえに高機能化が容易であり、インターネット接続機能などを備えた結果、開発対象が大規模化・高度化・複雑化している。

組込みソフトウェア技術者不足や短期開発、従来からの契約の慣行などの要因も加わり、業界として多くの課題を抱えている。

2.1 組込みシステムの定義と特徴

第 1 章では厳密な定義無しで漠然と、マイクロプロセッサが搭載されてソフトウェアで制御される機器を組込みシステムと呼んだが、あらためて組込みシステムとは何かを考えてみたい。

組込みシステムの定義には

- コンピュータ、ネットワーク、ソフトウェア等を用いて製品の機能を実現するシステムのこと。（東海大・IPA-SEC 大原氏、情報処理 05 年 6 月号）
- 各種の機械や機器に組み込まれてその制御を行うコンピュータシステムのこと。（名古屋大学 高田教授）
- コンピュータ、ネットワーク、ソフトウェア、計装制御技術、パワーエレクトロニクス技術等を用いて製品の機能を実現するシステムのこと。（東芝システムテクノロジー株式会社　金田氏）
- 組込みシステムは、1 つ以上の機能を持った単一の機器です。例えば、電話、FAX、コピー機、スキャナー、プリンタ機能を持つシステムが、機器が別々なら各機器を組込みシステムと呼んでいます。なお、それぞれの機能が機器として一体化されていれば、その一体化された複合機が組込システムの単位となります。組込みシステムは、1 つ以上の組込みソフトウェアと 1 つ以上のハードウェアで構成されます。（IPA-SEC　組込みソフトウェア向け　開発プロセスガイド）

など目的に応じて種々の表現があるが、おおむね色々な機器に組込まれて、それを制御するコンピュータシステム（機器と一体で製品を構成）を「組込みシステム」と呼ぶことができそうである。

これには専用のハードウェアと専用のソフトウェアを使用して特定の機能を実現するという意味合いがある。したがってソフトウェアの入れ替えにより多目的に使うことのできるパーソナルコンピュータなどは組込みシステムではないと考えられる。

しかし、大型の組込み機器、例えば半導体製造装置などにはパソコン本体やパソコンベースのボードが組込まれていてこれらの機器を制御している場合がある。この場合、半導体製造装置が組込みシステムであることは疑問の余地がない。パソコンであっても専用の目的に使用しているという点が特徴的であり、ここに組込みシステムである所以を求めることができそうである。

組込みシステムの特徴としては、特化したシステム、高い信頼性、リアルタイム性、リソースの制約が挙げられる。

・特化したシステム
パーソナルコンピュータ（汎用）と違い、目的の機能実現のために専用のハードウェア、ソフトウェアとなっている。
（機能実現のためにハードウェア、ソフトウェアが最適化）
・高信頼性
用途によっては誤動作により人命にかかわることもあり、高い信頼性が求められる。また機器がネットワーク接続することによる「脅威」に対抗することが不可欠となっている。
・リアルタイム性
ほとんどの組込みシステムでは、機器の仕様に定められたレスポンスタイム等の動作条件を保証するリアルタイム性が求められる。
・リソース
用途やマーケットにもよるが全体のコストダウンが求められ、CPU、メモリ等が制限される。また消費電力、小型化（及び軽量化）、使用条件などでも制約される。

しかし情報セキュリティの側面から見ると、さらに本質的な特徴は第一章で指摘したように、『利用者側からみると組込みシステムは目的とする機能を提供してくれる専用の機器であり、情報機器という認識を持っていないので、ウィルスウイルス対策ソフトウェアやファイアウォールソフトウェアを適宜更新するなどの運用管理を期待することは困難である。』という点にある。

2.2 組込みシステムの問題と対策の現状

この業界での関係者を役割によって区分すると、ハードウェアとソフトウェアを結合して最終製品として取りまとめるセットメーカーと、セットメーカーからの委託によりソフトウェアを開発する組込みソフトウェア開発者に大別できる。

一口にセットメーカーといっても、取り扱う製品分野によって状況は大きく異なるが、2.2.1 に共通的な概況を示した。

組込みソフトウェア開発者にも、セットメーカーの一部門、セットメーカーの関連企業、独立系など、組織の位置づけと成立の経緯によって状況が大きく異なるため、全てに共通するものではないが指摘された問題点を 2.2.2 に示した。この際、指摘された問題に対する異論や反論また同意も併記しながら、主に組込み系システムを受託する立場の課題をあげる。委員が本報告書の土台となる議論を交わした内容でもある。

セットメーカーの状況が厳しくなる中、組込みソフトウェア開発者に向けられる要求は厳しくなりつつあることがわかる。

2.2.1　セットメーカー側の概況

近年の組込み機器は多機能、高機能化しており、さらに日進月歩の進化を遂げている。この中で優位な製品を市場に供給すべくセットメーカーは熾烈な競争にさらされている。

マーケットの変化が速いため製品の開発期間も余裕を持って確保することは困難となり、ソフトウェア開発に費やすことができる時間を削減せざるを得ない状況にある。

一方で、製品に求められる機能や規模は大きくなり、ネットワーク接続機能などの情報セキュリティに与える影響が大きな機能の追加やソフトウェア開発規模の増大によって、情報セキュリティの確保はより難しくなってきている。

この様に、組込み製品のソフトウェア開発を情報セキュリティの確保という側面から見た場合、一方では機能の追加によってセキュリティの危険は増加しているにも関わらず時間は掛けられないという厳しい状況に置かれている。

セットメーカーにとってソフトウェア開発を全て自社内でまかなうことは困難であり、外部から調達したパッケージを使用する、IP（Intellectual property）とのトータルソリューションを購入する、外注するといったさまざまな対応を行っている。

一方、セットメーカー自身も市場に多くの製品を短期間で投入する必要があるため、セットメーカーの担当者も一つの製品の開発に割く事のできる時間が少なくなってきており、外部調達するパッケージの評価や外注先の評価に十分な時間を使えない状況がある。

2.2.2　組込みソフト開発者側の概況

以下における「概況」は問題を断定的に描いているのではない。様々の立場を有する業界の様相を描いているとご理解いただきたい。提起された問題は他社はすでにクリアしているかもしれないし、あるいは最初から問題になっていないかもしれない。問題を明確に掴むことは、全体を見通す専門家にゆだねるべきであると考える。しかし現場の、やや正確さに欠ける、また偏ったことも記しながら、次々と迫りくる時代の課題に IT 屋がそれぞれの持ち場で、ああではないか、こうではないかと試行錯誤している姿を現象として描いておくことは、情報セキュリティや機能安全以外にも問題を発見し、解決を試みる手がかりになるかもしれないと思う。そう考え以下を記す。

(1) ソフトウェア規模の増大

以前は 4 ビットマイコンや 8 ビットマイコン等、OS のない CPU 上にソフトウェアを載せていたが、最近は自動販売機でも Linux や Windows を載せている。ソフトウェアの量も桁違いに増えている。「量の変化は、質的変化を生む」事は自然科学の基本であり、本来なら開発量が 4～10 倍に増えたら設計手法も管理手法も抜本的（質的）に変えなければならない点があるはずだが、とりあえずの改善で済ませて根源的な課題を先送りにしてきた部署

が多いのではないだろうか。

もちろん、この見解に関しては異論、反論もあろう。組込み系に特化したライブラリ関数を整備し、ソフトウェアの増大化に備えているとか、また設計手法や管理手法も改善しているとかである。エンジン制御システムを開発する部門などでは、協力会社への委託案件を管理するために協力会社の開発管理体制を定期的にチェックするなども行っているようである。

しかしこれらの動きも「根源的な課題」に比べれば「とりあえずの改善」となりそうである。また「根源的」という言葉をどうとらえるかも重要である。6.3.2 の「検証と妥当性確認、そして日本の「合理性」」でも議論しているが、新たな方法論的展開（開発）ということもありうるのかもしれない。

（2）開発環境の問題

一般に使われているサーバーや PC には、ごく少数の勝ち残ったデファクトの CPU や OS が使われている。従って、プログラムの開発環境については OS の適合性などあまり議論する必要はなかった。せいぜい Version の違いやミドルウェアと業務パッケージの相性を気にしていれば良かった。CPU が誤動作することなど夢にも思わず、新しいハードやミドルウェアには、必ず（誤字のほとんどない）立派なマニュアルや「ヘルプ機能」がついていた。しかし組込み系の世界には多種多様な CPU や OS が存在する。乱戦模様でガリバーがいない。経験豊富な組込み系エンジニアならハードウェアは故障するものでソフトより性質が悪いと思っているから、ソフト設計を開始する前に候補となった CPU や OS の特性・品質を執拗に調べまわる。しかし、最近のソフト開発者は製品開発工程の下流に集中してしまい、事前調査もなく、試作が終わったばかりのハード上でいきなりデバッグをやらなくてはいけない事がしばしば発生する。業務系では開発途中でプラットフォーム仕様が変化していくことは想定していないと思うが、組込み系では開発途中においてプラットフォーム仕様が変形していくことは充分有り得る。

また、短期間での開発を行うためにエミュレート環境での開発を先行させる場合も増えてきており、実ハードウェアでの検証期間を十分に取れない状況での開発事例も生じている。

この指摘による最近の組込み系技術者のハード離れは、組込み系ソフトウェア量の増大に起因するともいえる。ソフトウェア量が増大化し、ハードを知らなくても仕事が成り立つ層が増えてきたといえるのではなかろうか。しかしこのような層が、組込み系エンジニアと称して、ハードとの調整をまずは行ってからソフトの仕事に取り掛かるというような旧来からあった、いわば“本来の組込み屋”の現場に回されてきたとしても大変困るという状況は目に浮かぶようである。この層の人たちは 16 進数のアルファベットの数え方も初

めてかもしれない。

しかし、最近の組込み系の好況の中では、ソフトのみを行うエンジニアだけではなく、このような“本来の”仕事を行うことも増えているであろう。確かにこの場に16進数も苦手なエンジニアが来たとしても、いくら構造化設計などが得意といっても目の前の仕事はこなせない。課題である。

(3) ソフトウェア部品の利用の問題

開発期間の短縮のために外部から調達したモジュールをセットメーカー等の要請で組み込むケースもある。この様な場合、各々のモジュールがセキュリティ的に十分に検証されたものであっても、使用する前提条件や組み合わせによって脆弱性を生むケースがあるが、この問題に関する検証は十分に行われてきていないのではないかと思われる。

また、契約上の問題、開発方法、環境の変化、製品サイクルの短縮によって、枯れたコードの蓄積を活用できない問題がある。開発プロダクトが変わるごとに同じような処理のコードを新たに記述することになり、毎回セキュリティ・ホールが作りこまれる。プロダクトが変わった際に流用できるのはノウハウだけで、ノウハウ以上のものは再利用されない。

組込み系がソフト資産の継承性に弱いことはいえよう。ハードなどの土台が変わるとその新たな土台にソフトウェアは対応しなければならないからである。ソフトウェアがあまりボリュームの大きいものでない場合は都度カスタマイズが必要であろう。しかしより一般的なミドルウェアのようなものが成り立つような場合は、ライブラリ化しより汎用的に様々のシステムで再利用できるようにすることができる。

このような蓄積はソフトウェアの品質も高めようが、しかし契約の問題など技術以外の要件が立ちはだかる場面も多々ある。ノウハウを形に残して第三者に再利用してもらえない部分である。ミドルウェア・クラスのソフトウェアの権利をどう扱うかの議論の余地はあるかもしれない。

(4) 入出力系の問題

業務系のシステムにおける入力はキーボードやOCR等であり、出力もディスプレイかプリンタという、いわゆるBI/O（Business Input/Output）である。アプリケーション・ソフトから見ればファイルへのリード・ライトでしかない。しかし、組込み系では多種多様なセンサーがインプット信号であり、多種多様なアクチュエータにアウトプットしないといけない。組込み系の入出力はPI/O（Process Input/Output）である。ファイルとはとても思えない。PI/Oのあるシステムを理解するのは純粋な情報工学出身者にはちょっとハードルが高いかもしれない。制御工学と電子電気工学のイロハぐらいは知らないとプログラムアルゴリズムは作れないし、当然デバッグやテストも苦戦する。サンプリング処理、ステ

ップ応答、ムダ時間など、デジタル制御の常識的用語が全くわかっていないどころか電子回路も読めない人達が必死でデバッグをやっても理解しがたい現象に悩むことになる。計測・制御工学と無縁な世界で育ったCOBOL系、UNIX系の業務系プログラマでは、設計精度にも試験の網羅性にも限界があると思われる。

業務系プログラマが自分の作ったプログラムによってモーターが動きメカが期待通りの働きをするというのを目の当たりにすることは、業務系プログラマに一種の感動をもたらすかもしれない。組込み系のI/Oは指摘のとおり業務系と異なり定型的ではない。組込み系とはI/Oの創意工夫ともいえる。I/Oを工夫することで新製品ができ、また既存製品は改良ができるからである。“I/O”という専門雑誌もあるが、組込み系の本領、あるいは固有の領域といえよう。組込み系教育に必要とされる内容と思われる。

（5）エンドユーザーの違い

業務系では最終ユーザーの姿がかなりはっきりしていると思う。どういう業務にどういう場面で使われるか、可能な限り明確に定義してもらえる。従ってプログラマにとっては、ＤＢを破壊するような誤入力を防止しておくぐらいで、積極的なヒューマンエラー対策や安全配慮はソフト開発者の担当外だったと思う。しかし、組込み系においてはユーザーは多種多様で、想定外の使われ方を覚悟しておかないといけない場合も多い。フェイルセーフ設計、フールプルーフ設計など安全工学的配慮が欠かせない。人間工学の知識がソフトウェア設計でも必要になっている。

業務系ではシステムが使われる場は、事務所などどの会社も同じような環境である。他方組込み系は様々な環境で使われ、人間を補助し仕事を行うための機械システムであったりする。生命への危険防止など組込み系エンジニアが考慮すべきことは業務系とは大きく異なる内容である。もっとも業務系に関しても、情報セキュリティに関して、ネットのみならず紙などのメディア管理システムをどうすべきか、など課題も多くある。

（6）発注者と受注者の関係

業務系では、モデル契約書が経済産業省から発行されているように、お互いの立場・責任を明確に（なるよう努力）している。しかし、組込み系では、発注仕様も契約内容も曖昧なままプロジェクトがスタートしているケースが多い。おまけに、発注者の意識と受注者の意識に大きなギャップがあるとIPA-SECの調査で指摘されている。

発注側の仕様が不明確なまま、受注側も見積仕様を明文化しないまま設計を進めていては、潜在バグだらけになりかねない。試験も不備が出てくる。その上、組込み系は業務系と比較にならないぐらい変更、後戻り作業が多い。それにも関わらず、発注側も受注側も工程管理や変更管理をしている例はきわめて少ないのではないだろうか。こうなると、ト

ラブルシューティングは、特に遅れて参加した（開発経緯を知らない）エンジニアには、手に負えなくなる。

IPA-SEC が指摘している受注側のスキル問題、プロセス管理問題よりも、ひょっとしたら、そもそもの契約関係の曖昧さと変更管理が最大の問題かもしれない。

また、契約時にセキュリティに関する要件が明確に定義されることはまれであり、不具合として対応しているケースがほとんどである。本来であれば、セキュリティに関する要件は仕様として盛り込まれるべきである。製品あるいはプロダクトが使用される前提条件の変化あるいは見落としに関しては製品仕様上の問題であり、発注者に帰属すべきである。

それにも関わらず、不具合の責任はソフトウェア開発者にあるとして処理する場合が多い。ソフトウェア開発者側も継続的な受発注関係を前提としてメンテナンスの形で対応せざるをえない事情があるが、情報セキュリティの本質から考えれば早急な改善が必要と思われる。

この問題は日本固有の商慣習、また他方最近の情報セキュリティや諸権利の順守という時代の方向性の中で、発注者と受注者がともに共有している問題であろう。しかし受注者が一方的に不利益を被っているのではなく、相互の損益バランスの中にある、という見解もある。なお、国際化にこのような関係が持続できるかという議論もある。技術論のみではない、文化論、社会論を含めた議論となろう。

（7）安全設計思想の欠如

組込み系は前述したように安全設計が欠かせないものが多いが、安全性の向上≒信頼性の向上と考えがちである。安全性と信頼性は重複する部分は多いが、本来、別な概念である（機械設計では、常識のようだ）。しかし、同様のものと考えているソフト開発者や発注者が多い。信頼性の向上には力を入れているものの、安全性確保が欠落している可能性がある。結果として、不具合の発生頻度は低下しても、影響度の大きい不具合を潜在化させている可能性は高い。

業務系で情報セキュリティ問題に対応する課題は、組込み系では安全であろう。しかし実際には組込み系もネットに接続することによって情報セキュリティに責任をもたなければならなくなったが、機器やシステムの稼働によって便益を供与する組込み系では便益の背後にある安全に関する理解は、単に学んだ方がよいを超えて、必須のことかもしれない。（6.1.2　組込み系企業が IEC 61508 へ取り組む利点、参照）

・産業機器や家電製品などに内蔵され、特定の機能を実現するコンピュータシステムを組込みシステムと呼ぶ。
・組込みシステムに内蔵されたコンピュータシステムに組込まれて使用するソフトウェアのことを組込みソフトウェアと呼ぶ。
・自動車・工作機器・デジタル家電など日本の強みである製品のほとんどは組込みシステムである。
・利用者側にとって、組込みシステムは目的とする機能を提供してくれる専用の機器であり、情報機器とは認識していない。
・利用者に、組込みシステムのウイルス対策ソフトウェアやファイアウォールソフトウェアを適宜更新するなどの運用管理を期待することは困難である。
・組込みシステム業界での関係者を役割によって区分すると、ハードウェアとソフトウェアを結合して最終製品として取りまとめるセットメーカーと、セットメーカーからの委託によりソフトウェアを開発する組込みソフトウェア開発者に大別できる。
・セットメーカーは熾烈な競争から短期間での製品開発を求められており、情報セキュリティ対策に十分な時間が掛けられない状況である。
・セットメーカーはソフトウェア開発を全て自社内でまかなうことは困難となっている。
・組込みソフトウェア開発者は、セットメーカーの一部門、セットメーカーの関連企業、独立系など、組織の位置づけと成立の経緯によって状況が大きく異なるが、開発環境、ソフトウェア部品の利用、セットメーカーとの契約条件など多くの問題を抱えている。

第 3 章　ソフトウェア・セキュリティ

第 1 章では情報セキュリティ対策の考え方や概要を説明したが、組込みソフトウェアの開発にあたって具体的には何をすれば良いのだろうか？

ソフトウェア・セキュリティを『ソフトウェアによる情報セキュリティ対策』と考えると 1.4.3 に示したような物理的攻撃方法に対するソフトウェア対策も存在するが、『ソフトウェア自身が意図しない動作を行わないようにすること』と捉えれば、基本的には

・ソフトウェア製品自体のセキュリティ対策

・ソフトウェア開発サイトのセキュリティ対策

の 2 つに大別される。

ソフトウェア製品自体のセキュリティについては、セキュリティを破る原理を再確認した後、いわゆるセキュアコーディングおよびツールによるチェックやプラットフォームのセキュリティ機能の活用について 3.1 にまとめた。

ソフトウェア開発サイトのセキュリティ対策は 3.2 に示すように多くの事柄からなっているが、基本的には ISMS に準じた対策を採ることが必要である。

3.1 ソフトウェア製品のセキュリティ

ソフトウェア製品自体のセキュリティに関しては

・ソフトウェアの耐タンパー性

・開発プロセスにおける誤りの排除

・コーディングにおける脆弱性の排除

の三つにまとめることができる。

（1）耐タンパー性とは、秘密情報や秘密情報の処理メカニズムを外部から不当に観測・改変することや秘密情報を処理するメカニズムを不当に改変することが極めて困難であるようにするための技術的対策のことである。り、ソフトウェアにおける耐タンパー技術としては、以下のものが知られている。

・実行コードの暗号化

コードを暗号化し、実行時に必要な部分のみがメモリ上に復号するようにすることで、ディスアセンブルなどの静的解析を困難にする。

・難読化

対象となるコードを等価でかつ分かりにくいコードに置き換えることでディスアセンブルなどの静的解析を困難にする。

・改竄検出コードの挿入

処理をバイパスするなどのモジュール改竄を防ぐために、モジュールの Hash 値を随時検証するメカニズムを組込む。

・デバッガ検知コードの挿入

デバッガが起動されているかをモニタするコードを埋め込み、実行時にデバッガが起動されている場合にはモジュールを強制終了する等の処理を行うことでデバッガによる動的解析を防ぐ。

（出典：耐タンパー性調査研究委員会報告書　財団法人日本規格協会情報技術標準化研究センター平成 15 年 3 月刊行）

(2)開発プロセスにおける誤りの排除は一般的ソフトウェア品質管理の問題であり、既にいくつかの標準が存在しているので、これを参考にして具体策を検討すれば良いと考えられる。

コーディングにおける脆弱性の排除はいわゆるセキュアコーディングの問題が中心であるが、ツールの利用やプラットフォームのセキュリティ機能を活用することも必要である。これについては 3.1.2 以降で紹介する。

3.1.1 セキュリティを破る原理

セキュリティが破られるとは、実行を奪われることと情報を盗まれることであろう。実行を奪うのも多くの場合、最終目的は情報を盗むことであるが、まれに実行時間、ネットワーク接続、ファイル格納場所をなどの計算機資源の剽窃を目的にしている場合がある。

(1) 実行を奪う

実行を奪うということは、すなわち任意のコードを実行させることである。実行が奪われればなんでもできるので重要な情報漏洩につながる。また、組み込み機器もネットワークに接続されて使用されるのが当たり前の時代なので、ある一台の機器の実行が奪われると同一ネットワークに接続された他の資源も危険にさらされることになる。つまり、他の資源への攻撃の踏み台に使われるわけである。例えば、プリンタに無駄な印刷をして仕事を邪魔するような物理 DOS アタックなどは比較的に容易に実行可能である。

実行を奪われた場合、伝統的な RTOS ではメモリ保護がないためやりたい放題である。UNIX、Linux ならばユーザーごとに権限が違う。ユーザーごとの権限を正しく決めて運用していれば、弱い権限のユーザーが破られてもシステム全体の危険は低めであると言える。弱い権限のユーザーには、通常、重要なファイルを書き直すことは許されていない。

また、機密度の高いファイルは内容を見ることもできない。ディレクトリも同様に読み出しを禁止できる。この場合、ディレクトリ内に存在するファイルのリストを見ることを禁止することも可能である。

ただし、UNIX 系 OS では root の権限は絶大である。したがって、root 権限を持てるプログラムはセキュリティを破られてはならない。例えば、昔、GNU-emacs は、root 権限を持っていたが、セキュリティ・ホールをつかれてネットワークから GNU-emacs を使用できる機械では様々な被害を受けた。

さて、実行を奪うためには、任意のコードを実行させねばならない。そのためには

・任意のプログラム・コードを機器内にロードし、
　ロードしたコードへ CPU の制御を移す
・プログラムの上書き（RTOS 系）

の二つの方法が考えられる。

プログラムの上書きとは正常によびだされているプログラムのコードを書き換えてしまい、次回の呼び出し時には、そのエントリ・ポイント以下がまったく別の乗っ取りのプログラムに置き換わってしまっているというものである。

伝統的な組込み機器ではプログラムを外部から送り込む方法など存在しないと考えられがちである。しかし、絶対に安全なのだろうか？例えば、ユーザーが入力したデータがほぼそのままメモリ上の文書バッファや画面のフレームバッファ（古い言葉でいう VRAM）に展開されていないだろうか？

1980 年代初頭には、表示画面のフレームバッファにキーボードから入力した文字をエコーバックで表示させ、その後、モニタ・プログラムからそのフレームバッファの文字にジャンプすることでユーザーが所望する動作を実現することがよく行われた。(例えば、MZ80K)

近代的な機器になるとネットワークに接続されるようになる。ネットワーク・パケットは任意のバイナリ・コードを組込み機器に展開する絶好の方法である。長さに制限はあるものの、バイナリの実行コードやデータをメモリ上に展開させるのに非常に適している。旧来の組込み機器でも外部から任意のデータやバイナリ・コードをメモリ上に展開することは、実際にはさほど難しくない。

では、送り込んだプログラムに制御を移す方法は存在するのだろうか？

その方法の一つがバッファ・オーバーフローとして知られている手法である。世界最初のネットワーク・ワームもこの方法を用いていた。

バッファ・オーバーフローの基本原理はスタック上にあるサブルーチンの戻りアドレスを上書きし、任意の番地に CPU の制御を移すことである。バッファ・オーバーフローでのセキュリティ突破は、以下のように行われる。

ネットワークのサーバ・ソフトウェアの入力受付状態のプロンプトに対してあるデータ列をどんどん送る。その入力受付ルーチンが脆弱な場合、入力バッファをローカル変数としてスタック上に持つ。そしてその脆弱な入力ルーチンはやって来たデータをどんどん受け付けてしまい、やがてバッファであるローカル変数を越えて入力データをスタック上に置いてしまう。

そのローカル変数を踏み潰してスタック上に置かれたデータ列には機械語プログラムが含まれている。また同時に、ローカル変数を通り過ぎた高位アドレスに位置しているサブルーチンからの戻りアドレスを、送り込んだバッファ上の機械語を指すように置き換える。これでサブルーチンからのリターンを実行すると、CPU の制御がバッファ上に送り込んだ任意のプログラムに移る。

これを実施するデータ列は、当然ながらサーバ・ソフトウェアを実行するハードウェアのアーキテクチャと OS に非常に強く依存している。

世界最初のインターネット・ワーム製作者は VAX 用 BSD と Sun の SUNOS のスタック構造を熟知してそれぞれの機械語コードを用意しており、sendmail（電子メールの基本的サーバ・プログラム）の脆弱性を利用してバッファ・オーバーフローで実行した。

当時のインターネット･サーバーの大多数は BSD で駆動される VAX と Sun であったので、このネットワーク・ワームは一夜にしてインターネットを崩壊させ、世界中の VAX と Sun を使用したインターネット・サーバーは、一度、ネットワークから切り離して再起動しなければならなかった。

これより後、インターネットを崩壊させるようなワームやウイルスは出現していない。

もうひとつはプロトコル・スタックの脆弱性を狙う方法である。プロトコル・スタックの脆弱性を狙うことは現実に可能である。

1989 年頃に出荷されていた Mips テクノロジー社の RISC OS では TCP/IP プロトコル・スタックに重大なバグがあり、TCP/IP パケット中に特定のビット・パターンが現れると OS そのものがフォールトを起こしてシステム全体が機能不全に陥ってしまった。

Mips 社の RISC OS は AT&T 社の UNIX SystemVR2 を基にした正統派の UNIX であったが、このような脆弱性があった。任意のプログラムが実行されたわけではないが、DOS アタックとして損害を与えるには十分であった。

Mips 社は世界に Mips CPU アーキテクチャを提供している会社で、多くの Mips CPU 採用のワークステーションやサーバーが RISC OS を採用していたことを考えるとこの脆弱性は非常に恐ろしいものだったが、幸いなことに現実には大きな社会問題にはならなかった。

このような先例を見ると、パケットにプログラムのバイナリ・コードなどを入れて対象機器に送りつけたのちプロトコル・スタックの脆弱性を突き、そのコードを実行できるようなプロトコル・スタック実装が存在し得るので、十分な注意が必要である。

さらに、DMA(Direct Memory Accoss) を壊してプログラムやデータを上書きすることも可能である。組込み機器では、I/O へのアクセスも野放しになっていることが多いので、前述の何れかの手法で小さなプログラム・コードを送り込んで DMA ハードウェアにアクセスし、

・メモリの内容を外部に出力する

・DMA で既存のタスクのコードや、システムを制御する重要なデータを上書きする

・外部入力から捏造した偽データを読み込み、正しいデータを上書きする

などを行うこと可能になる。

DMA は駆動するソフトウェアは小さくても効率的な攻撃が可能となるので大変危険であり、十分な配慮が必要である。

(2) 情報漏洩

組込み機器から情報漏洩があると考えている人は少ないが、いくつかの漏洩が考えうる。動的なメモリー・ヒープ管理はセキュリティ上の危険を持つことがある。

・動的なメモリ管理の危険性

伝統的な RTOS 系 OS では、malloc()に代表される動的なメモリ管理は重大なセキュリティの鍵を握っている。伝統的 RTOS では通常、malloc()で管理される領域は、OS 全体で共通である。RTOS 管理下でアプリケーションが malloc()で獲得した領域は、通常、一時的な作業記憶領域として使用する。そして、そこに機密情報を置く。作業後、特に何もせずに当該領域を free()で解放する。その直後に別なアプリケーションが malloc()で記憶領域獲得

を行うと、一般的なRTOSでは機密情報が入ったままのメモリ領域を獲得できることが普通である。これが悪意のあるアプリケーション、または、外部と不用意にやりとりするアプリケーションであると、そこから機密情報を抜き取ることができる。

・動的なメモリ管理とネットワークの組み合わせ

伝統的な組込み機器では悪意のあるプログラムを実行することができないので動的メモリ管理の危険性を杞憂と断じる人も多いが、それは誤りである。悪意のある危険なプログラムを一切動作させなくとも、前述のようにネットワーク・プロトコル・スタックに問題があれば情報は漏洩する。

現在の組込み機器には、それが単なるAV機器であっても、絶対に一般に公開されてはならない動画再生のための秘密鍵が組み込まれており、非常に重要な情報を内部に持っていることを忘れてはいけない。例えば DVD などに格納されている動画ファイルは暗号化されているが、それを解読するための鍵は世界中のファイルで共通のただ一つの秘密鍵で解読することができる。DVD再生機器の1台からでも、万一、秘密鍵が抜き出されて公開されれば、その被害は世界中のすべてのDVDに及ぶ。

また、多くの組込み機器がネットワークに接続されようとしている。しかし、前述のようにネットワークに接続すると常に危険にさらされており、組込み機器といえども適切な情報セキュリティ対策を施さねばならないことを再認識する必要がある。

3.1.2 セキュアコーディング

組込みソフトウェアのコーディング工程においては、顧客または会社が指定または推奨する何らかのコーディング規約に従ってコーディングを行うが、下記の一般的なコーディング規約では、セキュリティが考慮されていないという問題が存在する。

コーディング規約の例：

- GNUコーディング規約(Free Software Foundation)
- Indian Hill C Style and Coding Standards
- MISRA-C:2004(MISRA)
- 組込みソフトウェア開発向けコーディング作法ガイド[C言語版](IPA-SEC)
- 組込み現場の「C」プログラミング 標準コーディングガイドライン
  (福岡知的クラスタ(第1期)組込みソフト開発プロジェクト)

そのため、実際には、バッファ・オーバーフロー等の脆弱性はこのコーディング工程においてプログラマにより意図せずして実装されてしまうのである。よって、このコーディング工程においてこそセキュリティを考慮し、それにあったコーディングを行わなければ

ならない。

そこで提唱されているのがセキュアコーディングというコーディング手法である。特に組込みシステムに搭載される組込みソフトウェアは、C や C++といった言語で記述されることが多く、クラッキングなどの脅威に対処するためにはセキュアコーディングと呼ばれるコーディング手法を実践することが必須となる。米国では国防総省が CERT/CC（コンピュータ緊急対応センター）という団体に資金援助を行い、CERT セキュアコーディング規約を策定するなどの活動が行われているが、このことから国防上においてもセキュアコーディングがいかに重要であるかがわかる。

- CERT Secure Coding Standards
  (https://www.securecoding.cert.org/confluence/display/seccode/CERT+Secure+Coding+Standards)
- CERT C Secure Coding Standard
  (https://www.securecoding.cert.org/confluence/display/seccode/CERT+C+Secure+Coding+Standard)
- CERT C++ Secure Coding Standard
  (https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=637)

一方、組込みソフトウェアにおいてもこの規約に従えばセキュリティ性の向上につながるが、処理のオーバーヘッドが大きくなるため、すべての規約に従うとパフォーマンスが落ちてしまうという弊害が生じてしまう。しかしながら、人命を扱う医療機器や人々の生活や経済に重大な影響を及ぼすインフラを制御する機器に搭載される組込みソフトウェアはこの規約に従って実装されるべきである。

次に、文字列と数値の扱いに関するセキュアコーディングの具体例について紹介する。

①文字列の扱い

特に文字列の扱いには十分な注意が必要である。なぜならば、バッファ・オーバーフローなど重大な問題を発生させることになるからである。バッファ・オーバーフローを発生させる可能性のあるソースコードの例をリスト 3.1 に示す。

```
func_input(void)
{
  char a[8];
  gets(a);
  ・
  ・
}
```

リスト 3.1　無制限文字列コピーを行ってしまう可能性のあるコードの例

このコードでは、a という変数では 8 バイト分の領域しか確保されていないため、文字列終端文字（¥0）を除くと 7 文字しか格納することができない。よって、もし 8 文字以上の入力があるとバッファ・オーバーフローが発生することになる。その結果、スタックが破壊され、プログラマが意図しない動作をすることになる。
もし、悪意のある攻撃者が文字列を入力した場合、攻撃者が意図したプログラムの実行が可能にしてしまう。

よって、この問題を回避するには、リスト 3.2 のように gets()関数の代わりに fgets()関数を使用して、上の例で 8 文字以上の文字列が入力されても、実際には 7 文字までしか読み込まないように実装すればよい。

```
func_input(void)
{
  char a[8];
  fgets(a, 8, stdin);
  ・
  ・
}
```

リスト 3.2　無制限文字列コピーに対処したコードの例

②数値の扱い

数値の扱いも注意が必要である。整数演算などにより整数の表現範囲を超えてしまうオーバーフローはよくあるミスの一つである。オーバーフローを発生させる可能性のあるソースコードの例をリスト 3.3 に示す。

```
int func_add(int a, int b)
{
  int c=0;
  c=a+b;
  return c;
}
```

リスト 3.3　オーバーフローを発生させる可能性のあるコードの例

オーバーフローは、int 型の最大値が 2147483647 の場合（32 ビットの場合)、加算して 2147483647 を超えてしまうような整数を加算するとオーバーフローが発生する。その結果、演算結果が予期せぬ値となり、プログラムの動作に影響を及ぼすことが考えられる。

よって、この問題を回避するには、リスト 3.4 のように加算する前に加算してもよい値であるかをチェックすればよい。

```
#include<stdio.h>
#include<stdlib.h>
#define INT_MAX 2147483647
int func_add(int a, int b)
{
  int c=0;
  if(b>(INT_MAX-a))  ・・・a と b を加算するとオーバーフローしないかを事前にチェック
  {
    pritnf(“Error:Overflow¥n”);
    exit(1);  ・・・プログラムの実行を停止
  }
  else
    c=a+b;
  return c;
}
```

リスト 3.4　オーバーフローに対処したコードの例

これらの例のように、意図しない入力が発生してもプログラムの動作に影響が出ないようにコーディングを行うのがセキュアコーディングである。是非とも C/C++セキュアコーディング規約を参照し、安全なソースコードの作成に努めていただきたい。

また、メモリの領域の再利用の際にも注意が必要となる。UNIX/Linux の場合、malloc(), free()は同一プロセスに閉じているので、セキュリティ情報を扱うソフトウェアのみを注意深く書いておけば良い。ただし、多くの初期の UNIX では、仮想記憶による物理メモリの再利用の際に別なプロセスが情報を書き込んだ領域が、別のプロセスにそのまま割り当てられた。

当然、それはセキュリティ・ホールである。現在のほとんどの UNIX 系 OS ではプロセスに物理メモリを割り当てるとき、その内容を 00 とか 0xdeadbeef などで埋めている。

RTOS 系の場合、既述のとおり malloc()、free()がセキュリティ・ホールを招くので、free()する際に重要な情報は 0 でクリアなどをする必要がある。

ネットワークとメモリ領域の再利用を行う場合、UNIX/Linux、RTOS いずれの OS においても、ネットワーク・ドライバではパケットのパディング領域を 0 クリアすべきである。

3.1.3　処理系やツールによるチェック

（1）Java のクラスローダによる安全性

バッファ・オーバーフローなどのプログラマが意図しない不当な領域へのメモリー・アクセスは実行時チェックを行えば禁止できる。

多くの仮想機械を使用するシステムでは、その仮想機械の実行系(インタープリタ)がメモリー・アクセス範囲の実行時チェックを動的に行う。

最近の仮想機械を使用するシステムでもっともよく知られ、かつ、広く使用されているのは Java であろう。

Java はインターネット時代になって作られた言語であるから、セキュリティ面の安全について配慮されている。

Java の応用プログラムにはアプレットとアプリケーションがある。
Java アプレットはインターネットからダウンロードして使用されるのが前提であり、実行時の制限が非常に強いが、より安全である。

Java アプリケーションは、あらかじめシステムにインストールして実行されるものであり、システム内の資源へのアクセスは通常のプログラミング言語で作成したアプリケーションと同様である。Java アプレットの安全性で、有名なのは、

・外部からクラス（プログラム）をロードしたときにクラスローダが行うクラスの一貫性のチェックと、機械語の安全性をチェックするコード・ベリファイアの機能。
・いわゆるサンドボックス(砂箱)といわれる機能。あらかじめ許可したホスト(サーバー)かアプレットのダウンロードもとのホストとしか通信ができない。また、実行を行う機械(以下 実機)内のローカルなファイル操作も(あらかじめ設定を行っておかなければ)できない。

などであろう。

これらの機能は、インターネットからダウンロードされる Java プログラム(アプレット)が仮に悪意を持っていても、それを実行する実機内の情報を盗み出してネットワーク上に送出したり、ファイルを改変することを防いでくれる。したがって、ユーザーはインターネットからダウンロードした Java 機械語のプログラムを安心して実行することができる。

Java アプレットにおけるサンドボックスはファイル、メモリなどのローカルな資源へのアクセスを制限するので、情報の読み出しによる情報漏洩や、ファイルを書き直すことによる情報改竄、また実行ファイルを改ざんすることによるウイルスやスパイウェアの植え付けなどに耐性がある。また、ネットワーク・アクセスが制限されているので、こちらの実機の情報は必要最小限のホストにしか与えずに済む。

Java が他の仮想機械を使用するシステムに比べて優れているのは、コード・ベリファイアと、クラスのチェックである。

コード・ベリファイアは、機械語にコンパイルされたバイナリ・コードを実行前に走査して安全でない組み合わせのコード列をはじいて実行させない。これにより、悪意をもったプログラマがプログラムをクラッシュさせるコードを書いて忍ばせていても、コード・ベリファイアがそれをはじくので不当なコードは実行されることがない。仮に安全なコードであっても、ベリファイアが安全であると認識しない限りは、はじかれてしまうのだが、Java コンパイラを使用している限りは、普通のプログラマはコードの安全性には気を配る必要はない。

クラスのチェックは、ロード対象のクラスの一貫性や、既存クラスとの一貫性をチェックし、不適切な場合はロードを行わないので、Java では安全なクラス(プログラム)だけがロードされ実行される。

なお、Java には実行時チェックの機構もあり、Java アプリケーションもかなり安全に実行される。

(2) Java の実行時チェックによる安全性

Java はアプリケーションなどを記述する場合にも安全である。サーバーへの攻撃が成功した事例の多くは、バッファ・オーバーフローというプログラムの配列チェックの緩さを突いたものがほとんである。セキュリティを破る者は、プログラムの配列チェックの緩さをついてプログラマの意図しないメモリ領域にアクセスし、それによって不正なプログラム・コードを実行させるのである。

現在、組込みアプリケーションを書くのによく使用される C 言語、C++言語などでは、一般的に配列アクセスの暗黙の実行時チェックは行われない。つまり C 言語などでは、プログラマが注意深くすべての配列アクセスのチェック機能を作りこまねばならない。そうしなければ、意図しない不当なメモリ領域に書き込みを行われてしまうことがありえる。

ひるがえって、Java ではその機械語の実行は仮想機械の実行系(インタープリタ)が行う。その仮想機械は配列アクセスの範囲チェックを実行時に自動的に行うので、プログラマの意図しないメモリー・アクセスを自動的に排除してくれる。つまり、Java で記述したプログラムはいかなる方法でも不当なメモリ領域にアクセスすることができない。すなわち、プログラマが配列アクセスのチェックについて何もしなくても、不正を試みるものはその意図を達成することができない。

Java の不当な領域へのアクセスのチェックは、正規のプログラム開発の効率向上にも寄与する。現在、組込みソフトウェア開発が長期化する大きな原因の一つが、意図しないメモリ領域へのアクセスである。Java では不正なメモリー・アクセスが検出されるので、プログラムの問題の発見が容易である。

ただし、Java には JNI(Java Native Interface)というものがある。JNI は、C 言語などの Java 以外の言語で記述したサブルーチンと、Java で書いたサブルーチンとの間で、自由に呼び出しあう機能である。通常の Java アプリケーションは JNI を使用する必要性はまったくない。しかし、組込みソフトウェアでは特殊なハードウェアへアクセスするためや、性能の向上のために JNI を使用して、C 言語で記述したサブルーチンを呼び出さざるを得ない場合がまれにある。JNI は、データの型チェックなどを厳密に行うので、下らないミスの多くは防がれる。しかし、結局は C 言語で書いたプログラムが走行する。つまり Java の恩恵にはあずかれないので、C 言語で書かれる部分は、注意深く作成されなければならない。

3.1.4 プラットフォームの機能の活用

組込みソフトウェアでも OS の使用が一般的となりつつあるが、これらの OS には一定のセキュリティ機能を持つものが少なくない。また、4 章に述べるようにハードウェア自身でセキュリティ機能を高めたチップも登場してきている。

従って、これらプラットフォームの機能をうまく活用する技術が重要となってきている。

(1) メモリー・プロテクション

RTOS を使用してシステムを作っている人々は、メモリー・プロテクションの採用に後ろ向きだが、 メモリー・プロテクションはセキュリティを向上させるのに有用である。

・読み書き禁止

万が一どこかのタスクでセキュリティが破られても、他のタスクのメモリが読み出せなければ機密情報が読み出される危険性は劇的に低くなる。ただ RTOS の場合にはファイルへのアクセス制御が弱いので、ファイル上のデータの保護は別問題である。

・書き込み禁止

読み出しができて書き込みのみ禁止というメモリー・プロテクションは多くの既存の RTOS プログラムを比較的容易に移行させることができるだろう。これでもメモリー・プロテクションが無いことに比べれば非常に有効である。なぜなら、万が一どこかのタスクでセキュリティが破られても、他のタスクのプログラム・コードを上書きして侵食をどんどん進めることを困難にするからである。

(2) プログラムの難読性

最近では、プログラムの対タンパー性が重要だという認識が浸透してきている。

海外には市場に出荷されている機器から情報を剽窃したり、ソフトウェアを改造したりすることを仕事として行っている者たちがいる。機器を購入して解体し、ハードウェアに計測機器などを接続して情報の抜き出しやソフトウェア、データの書き換えなどを行う。

そういう行為に耐えるのが対タンパー性である。

対タンパー性は通常はハードウェアの補助が重要である。例えば、DVD の秘密鍵はハードウェアで保護されていて、特別な方法でしかアクセスできないことが多い。また、機器内部のデータバスなどを覗き見るような装置を接続しても、データバス上のデータがハードウェアによって暗号化されていることが期待されている。

組込みソフトウェアとして対タンパー性を高める方法もある。

基本的にはプログラムの「難読性」を高める。つまり、ディスアセンブルを難しくする。無関係なデータを一つのワードに入れるなどしてデータ構造を無意味に複雑にするとか、プログラムをむやみに断片化したりするなどの手法を使うのである。また、データを暗号化したりする。ただし、同一の機械で解読を行うような暗号化は非常に弱いので、本質的には攻撃者の手間を増やすということにしか役立たない。

難読性を高めた場合でも高級言語のソースコードはマクロ定義などを使用すればある程度以上には複雑にはならない。しかし、実機上での実コード、実データでのデバッグ作業は難読性にほぼ比例して面倒になることを覚悟しなければならない。

(3) Linux、UNIX 系 OS のセキュリティ機能の使用

UNIX 系 OS はそもそもセキュアである。UNIX 系 OS がインターネット・サーバーをはじめとする攻撃を受けやすい場面に多数使用されているのは衆知のとおりである。(以下、しばらく UNIX、Linux などの UNIX 系 OS を UNIX と表記する)

UNIX のアクセス権限（すなわち制限）の考えは単純だが強力である。もし、一般ユーザーのパスワードが漏れるなどでセキュリティが破られても、そのユーザーがアクセス可能なファイル以外が被害に合うことはない。UNIX 系 OS ではアクセス権限の設定を適切に行うことは重要である。UNIX はユーザーやグループごとに資源へのアクセスの権限を設定できる。グループとはユーザーが属することができる管理単位である。ユーザーはグループにはいくつでも入ることができる。

UNIX のアクセスの制限（権限）の実現はファイル・システム上の資源へのアクセスの制限によっている。UNIX の資源のほとんどはファイル・システム上のファイルなどに対応付けされている。そして、そのファイル・システム上のファイルなどに対する属性、 「読み出し可能」、「書き込み可能」、「実行可能」を 「ユーザー」、「グループ」、「その外」ごとに設定する。通常、普通に運用されている UNIX で、一般ユーザーが自分のファイルを作ると、

ユーザー：読み書き可能
グループ：読み出し可能
その外　：読み出し可能

としてファイルが作成される。ここでいうユーザーとはファイルの持ち主であり、UNIX ではしばしば「オーナー」とも表現されている。このファイル・アクセス権限の属性は、ファイルのオーナーであれば chmod コマンドで簡単に変更することができる。また、ファイル新規作成時の属性の初期値はシェル変数などを設定することで変更できる。

UNIX では「root(ユーザーID=0)」というユーザーの権限はすべての資源にアクセス可能で、またシステムの重要なパラメータを変更可能である。したがって、root 権限で動作するプログラムは必要最小限にしなければならない。root 権限で動作するプログラムにセキュリティ・ホールがあってそれが乗っ取られた場合には、システムのすべてが危険にさらされるからである。root 権限で動作するプログラムはセキュリティ上、十分に精査されたものでなければならない。UNIX では、アクセス権限の設定を正しく運用すれば非常に高いセキュリティが得られる。

しかし、これまでの組み込み Linux などを使用した開発では UNIX 系 OS のユーザー権限とそれに関するアクセス権限(制限)の設定を理解していない開発者がほとんどであり、アクセス権限の設定が非常にいい加減なため OS のセキュリティ機能を生かさず、ややもするとセキュリティ的に安心できないことがよく見受けられる。例えば、すべてのアプリケーション・プログラムを root 権限で実行させていたりする事例も見受けられる。

3.2 ソフトウェア開発サイトのセキュリティ

1.6 では設計開発や製造にかかわる情報（文書、図面類など）も厳重に守る必要があり、開発サイトの情報セキュリティ対策が必要となることを述べた。

また、ソフトウェアを出荷してからユーザーに届くまでの間でもセキュリティの確保が必要である。CD-R に書き込まれて出荷された PC 向けソフトウェア製品においても、ウイルスが混入した事例があり、USB メモリに組み込まれて出荷されたソフトウェアにおいてもウイルス混入の事例がある。

ここでは、ウイルス混入に限らず、ソフトウェア製品のデリバリーにおいて、留意すべきポイントを見ていく。

まず、開発時のソースコードの構成管理であるが、一つの部署あるいは会社で開発をしている際には、CVS, SVN 等の構成管理ソフトウェアが活用でき、適切な構成管理を行うことは比較的容易である。

開発体制が大きくなって分業が進んだ場合、全ての開発者が全体のソフトウェアのソースコードにアクセスできることはまれであり、各担当者は自らが担当するモジュール以外の部分については、オブジェクトのみの供給を受けて開発することがある。この様な場合、開発・検査を行った環境と、最終的に組み合わせられるオブジェクトが異なることがある。

この様なケースでは、各オブジェクトでの変更履歴と構成の管理を慎重に行わないと、最終製品となる構成でのテストが不十分だったといったケースが発生しやすい。

最悪のケースでは、最終製品となったバイナリを出荷後に再現するのが困難になったという事例もある。少なくとも、最終製品と同一のバイナリを間違いなく生成できることを担保することは重要である。

製品バイナリを生成する際には、バイナリの同一性の担保の上でも、ビルド環境を再現できることは大切である。コンパイラやライブラリのバージョンやパッチの有無などについても記録しておくと共に、ウイルス感染などの被害を避ける上からも、最終ビルドを行う環境は専用マシンとして、多くのアプリケーションを導入することは避けるべきであろう。

組込み製品は一般のソフトウェアに比べて長いライフサイクルを持つことがあるため、ビルド環境を製品ライフサイクルにわたって確保しておくことが重要になるケースもある。特定の PC 環境でしか動作しないコンパイラが必要になる場合などは、PC 環境そのものを保存しておく必要があろう。

最終バイナリを機器に組み込む際には、SCM 的な管理が必要になる。工場に展開する際に、書き込みを依頼したバージョンのバイナリが確実に渡っているのか、以前のバージョンのマスターロムが残っていて誤ったバージョンが製品に混入することが無いようなシステムを構築する必要がある。

生産後、出荷までの間に不具合が発覚したような場合には、在庫品を書き換えるようなイレギュラーな対応が発生することがあるが、この様な場合に事故が起きやすい。書き換え後の製品を識別する方法を考慮する必要がある。最終製品において、バージョンを確認する手段を用意しておくことが最後の砦になるが、梱包を解いてバージョン確認をすることは大変な作業となるので、混在することが無いような手順で作業を行うことが重要である。

以上、いくつかのポイントを説明したが、組込みソフトウェア開発は情報サービス業そのものであることを考えると、開発サイト全体として 5.5 に紹介する ISMS に準じたセキュリティ対策を実施すべきであり、理想的には ISMS 適合性評価制度に則った認証を取得すべきであろう。

・ソフトウェア・セキュリティにはソフトウェア製品自体のセキュリティ対策とソフトウェア開発サイトのセキュリティ対策がある。
・ソフトウェア製品自体のセキュリティ対策は、ソフトウェアの耐タンパー性、開発プロセスにおける誤りの排除、コーディングにおける脆弱性の排除の三つに区分できる。
・耐タンパー性とは、秘密情報や秘密情報の処理メカニズムを外部から不当に観測・改変することや秘密情報を処理するメカニズムを不当に改変することが極めて困難であるようにするための技術的対策のことである。
・開発プロセスにおける誤りの排除は、一般的ソフトウェア品質管理の問題である。
・コーディングにおける脆弱性の排除は、セキュアコーディング、適切なツールの利用、プラットフォームのセキュリティ機能を活用することなどのよって対処できる。

《参考文献》

1　C/C++セキュアコーディング, Robert C. Seacoed,歌代和正、2006,株アスキー

2　C/C++セキュアコーディング, John Viega,Matt Messier, 岩田哲、2004, 株オーム社

3　A Tour of the Worm, Donn Seeley
(Department of Computer Science University of Utah),???
http://securitydigest.org/phage/resource/seely.pdf

第 4 章　ハードウェア・セキュリティ

情報セキュリティ対策におけるハードウェアの重要性に関しては1.4.3で情報システムや製品の情報セキュリティ対策のうち

・物理的対策は主としてハードウェアにより実現されること
・論理的対策が効果を発揮するには、基盤となるハードウェアが正しく動作することが前提条件であること

を述べた。

ハードウェア・セキュリティについては筐体、電子基板、LSI チップなどのレベルで多くのセキュリティ対策が検討されているが、ここではハードウェアの耐タンパー性に関する話題と、ソフトウェアへの攻撃を防御するプロセッサのセキュリティ機能について紹介する。

耐タンパー性とは、秘密情報や秘密情報の処理メカニズムを外部から不当に観測・改変することや秘密情報を処理するメカニズムを不当に改変することが極めて困難であるようにするための技術的対策のことであり、ハードウェアに対する耐タンパー技術としては、以下のようなものが知られている。

・暗号処理回路とメモリなどの 1 チップ化
　暗号処理回路とメモリ回路を別の LSI にすると、インタフェース回路パッド部分や配線部分を流れる信号を解析されてしまうため、1 チップ化することでインタフェース回路パッド部分や配線部分の解析を困難にする。
・電子基板やモジュール表面のコーティング
　外部から回路パターンが判別できないように電子基板やモジュールの表面をコーティングする。また、攻撃の痕跡が残るようなコーティングを行うことにより、攻撃を検知することもできる。さらに、剥がすと回路パターンまで壊れてしまうようなコーティングも存在する。
・クロック周波数の異常を検出
　電子基板やモジュールが誤動作を起こすまでクロック周波数を変化させるという攻撃を防ぐため、クロック周波数の異常を検出し規定周波数以外では電子基板やモジュールが動作しないようにする。
・電圧の異常を検出
　電子基板やモジュールが誤動作を起こすまで電圧を変化させるという攻撃を防ぐため、

電圧の異常を検出し規定電圧以外では動作しないようにする。

・バスのスクランブル

バスを流れる信号の解析を困難にするため、バスを流れる信号のスクランブルを行う。スクランブル以外にもダミーの信号を流すことにより解析を困難にすることができる。

・LSI 検査パッドの除去

LSI の機能検査用のアドレスパッドなどがあると内部動作を解析しやすいので、検査終了後には検査用のアドレスパッドを除去する必要がある。

・光検知によるメモリの消去

光を検知する回路を組み込み、LSI の回路パターンなどを露出させると光を検知し、メモリに記憶されていた秘密情報やプログラム等を消去する。

・消費電力の変動を抑える実装

消費電力の変動を調べることで内部の秘密情報が解析されることを防ぐため、消費電力の変動を抑える実装が必要となる。

・処理時間の変動を抑える実装

処理時間の変動を調べることで内部の秘密情報が解析されることを防ぐため、処理時間の変動を抑える実装が必要となる。

（出典：耐タンパー性調査研究委員会報告書　財団法人日本規格協会情報技術標準化研究センター平成 15 年 3 月刊行）

LSI チップ自体の耐タンパーを高めたセキュリティチップに関する動向を 4.1 に、最新プロセッサのセキュリティ機能の動向については 4.2 に記す。

4.1 セキュリティチップ

セキュリティチップとはチップ自体の耐タンパー性を高めて、それをベースとしてプラットフォーム全体をセキュアにしようとするものであり、セキュアなドメインをユーザードメインから分離することによりセキュリティを実現する。セキュアなドメインでは暗号、認証、DRM（Digital Rights Management）などを処理する。

事例として TCG の TPM、Intel の TXT、 Intel, IBM, Docomo の TMP（Trusted Mobile Platform)、Trusted Logic の STIP（Small Terminal Interoperability Platform)、ARM の TrusteZone などの業界標準がある。

また、1 つの LSI の上に複数の CPU を搭載した CMP（Chip Multi-Processor, マルチコア）により、異なる CPU 上に異種（リアルタイム性やセキュア機能等）の実行環境を載せてセキュリティと省電力を同時に実現しようとするアプローチもある。仮想技術との連携で、仮想化の負荷の軽減やリアルタイム性の保証などにも役立つ。ここではセキュアチップの TPM、最新プロセッサのセキュア機能としてインテルの TXT、ARM の TrustZone, Microsoft の BitLocker 等について取り上げる。

(1) TCG 提案の TPM

TCG（Trusted Computing Group）はセキュアなコンピュータプラットフォームを構築するためのハードウェア、ソフトウェアの業界標準仕様の開発・普及を目的とした業界団体で、AMD、HP、IBM、Intel、Microsoft、Sun Microsystems 等の IT 企業により 2003 年 4 月に結成された。

正規ユーザーとしてコンピュータに認識されたユーザーや PC 上で実行中のソフトウェアはファイルやメモリにアクセスできるので PC の内部を実質的に支配し、不正行為を実行可能な状況にある。TCG はソフトウェアで実現される信頼性には自ずと限界があるとして、特に正規ユーザーによる不正行為から守るために TPM（Trusted Platform Module）と呼ばれる耐タンパー領域をセキュリティチップという形で設け、そこにコンピュータ固有の情報、暗号鍵、ユーザーの秘密データを格納・管理しようという提案をしている。TPM から OS まで「信頼の連鎖」（chain of trust）でつなぎ、プラットフォーム全体をセキュアにしようとする Transit Trust（遷移的信頼）という手法が用いられる。その構造を図 4.1 に示す。

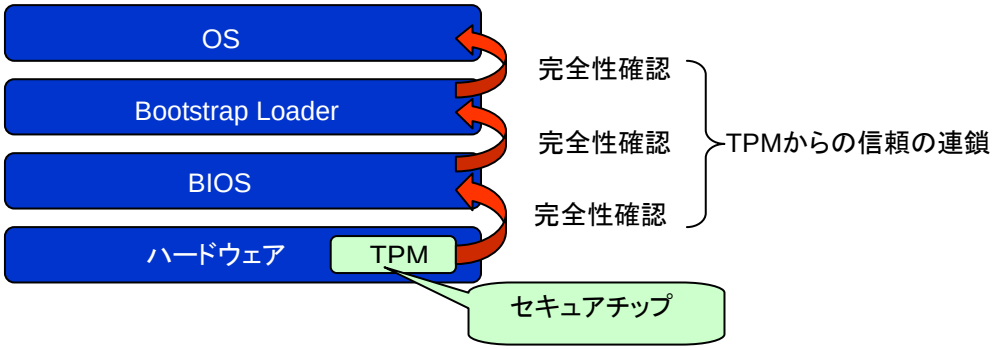


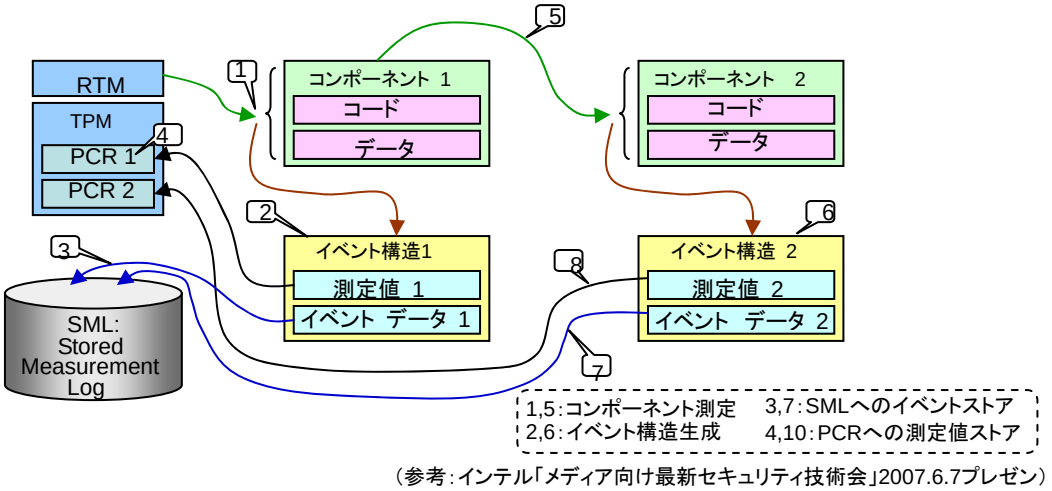
図 4.2 セキュアなコンピュータプラットフォーム

【参考】: Business Communication@net 最新技術トレンド

http://www.bcm.co.jp/site/2004/2004Feb/techo-trend/04techo-trend02.htm

これは最も信頼できるコンポーネント（RTM：Root of Trust for Measurement）から順次コンポーネントを計測（Measurement）して完全性を確認していく方法である。図 4.1 ではハードウェアの TPM が起点となって BIOS の計測を行い、BIOS が Bootstrap Loader の計測を行ない、Bootstrap Loader が OS の計測を行って、更に OS がアプリケーションの計測をおこなっていき、Transit Trust を広げていく。

計測には例えば SHA-1（Security Hash Architecture Versin1）のようなハッシュ関数が用いられ、各コンポーネントで計測されたハッシュ値が TPM 内の PCR（Platform Configuration Resisters）に保存され、これを調べることによってプラットフォームの信頼性が担保される。Transit Trust の具体的な手順を図 4.2 に示す。



（参考:インテル「メディア向け最新セキュリティ技術会」2007.6.7プレゼン）

図 4.2 Transit Trust（信頼遷移）

（【参考】http://journal.mycom.co.jp/articles/2007/06/08/IntelTXT/index.html ）

TCGには前述のAMD、HP、IBM、Intel、Microsoft、Sun Microsystemsなどのほか、日本企業ではSONY、富士通、日立、東芝、NECなどの主要ベンダーを含む中心的IT企業 100 社以上が参加している。このことから今後のTCGの仕様に準拠したコンピュータプラットフォームの普及が加速していくと予想される。

TPM はプラットフォームの認証，保護メモリ領域，暗号処理用コプロセッサ等の機能により、通常ソフトウェアで実行されている処理よりも高いセキュリティレベルを確保できるようにしている。TPM 自体はオープン仕様であり、コストや柔軟性を考慮してハードウェア部分は最小限にして他の部分はソフトウェアで実現している。また、プラットフォームを利用するソフトウェアとして TSS（TCG Software Stack）がある。TSS は異なる TPM 間の互換性も確保しており、Intel の TXT，Microsoft Vista の BitLocker などほとんどの PC はこの技術を採用している。

今後の課題としては、1)運用上の問題としてTPMから暗号鍵を取り出すのは困難であるため、2)TPM内の暗号鍵が物理的に破損した場合の復旧が困難であること、3)管理をユーザー任せだと暗号ファイルの復旧が不可能、などがある。また互換性の問題として、4)各社の実装が異なることからくる実装レベルの検証の問題、5)プラットフォーム認証に係るプラットフォームの保護、その他 6)利用者への認知度向上のための普及活動等解決すべき問題がある。

情報家電への適用に関して言えば、従来家電における製造形態は系列会社による縦割りの体制においてブラックボックスのアプライアンスとして製造されるケースが多かった。そのため必ずしもTCG で規定されている複雑なアーキテクチャが適切であるとは限らない。しかし、家電のデジタル化、ネットワーク化の進展とともにオープンアーキテクチャ化、水平分業化もある程度進んでいくことが予想される。今後も、組込み機器や携帯電話への適用範囲の拡大等も含め、TCG の動向を把握していく必要があると考えられる。

(2) TNC (Trusted Network Connect)

TCGの活動の中でプラットフォーム認証（構成証明）を実現するネットワークプロトコルに関してもTNC（Trusted Network Connect）を提案している。これは，TCGのサブグループの名称で、約70のメンバー企業が策定した標準オープンベースのネットワーク・アクセス・コントロール（NAC）のアーキテクチャ仕様である。NACの基本要素として、

1)Authentication（誰が？）

2)Endpoint Security Assessment（どの端末から？）

3)Environment Information（どんなネットワークで？）

の3つがあるが、現在のところ認証以外は標準化されていない。

（他にCiscoのNACやMicrosoftのNAPなどがある）

また、TNCは製品間の相互接続性を保証するためAPIやProtocolを全て公開し、既存のIEEE802.1x、RADIUS、IPsec、EAP、SSL/TLSなどの標準の上にも構築可能としている。

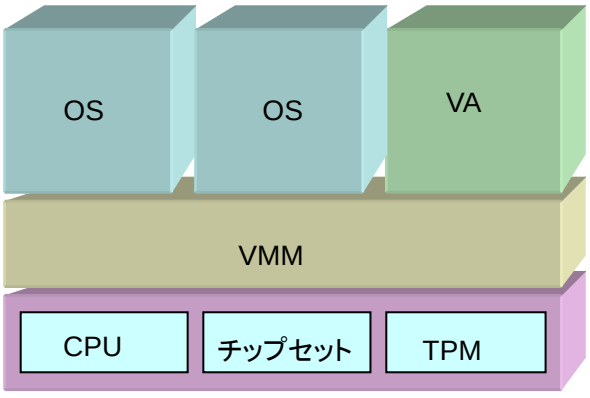
その他CiscoのNAC（Network Admission Control）やMicrosoftのNAP（Network Access Protection）等の規格化も進められており、これら標準化動向の把握も重要である。

4.2 最新プロセッサのセキュリティ機能

(1) Intel 社の TXT (Trusted Network Connect)

インテルはセキュリティ機能を強化した第 2 世代 vPro (企業向けデスクトップ技術) の発売に合わせて、従来 LaGrande Technology (LT) と呼ばれていたセキュリティ技術を TXT として公表した。

TXT は TPM1.0 を採用し、図 4.3 に示すように仮想化技術 VT 対応 CPU の上に OS と全てのネットワーク通信を経由する「Virtual Appliance」が載り、仮想化モニタ VMM がこれらを制御して個々のアプリケーションを切り離した領域で稼動させることによって外部からの攻撃を封じ込める。これは TPM と仮想技術(Intel VT)を組み合わせてシステムを保護する技術で、これによって TCG の chain of trust(信頼の連鎖)が実現される。



VA : Virtual Appliance　TPM : Trusted Platform Module

(参考:インテルプラットフォーム技術セミナー2006.11.14)

図 4.3　インテル TXT の概要

TPM では, 前述のように Transitive trust(遷移的信頼)を用いてプラットフォームの信頼性を実現する。結局インテル TXT とは Transitive Trust に VMM(仮想マシンモニタ)を加えたものである。プロセッサから順に信頼性を保障していくことでシステム全体の信頼性を確保するという考え方である。

また、Intel TXTに見られるアーキテクチャは従来ソフトウェアで実現しようとするのが一般的だったが、以下の理由からハードウェアで信頼性を確保する多くのメカニズムを提供している。

- 高い信頼性が確保できること(ソフトウェアは改竄が可能だが, ハードウェアの改竄は困難)
- 正確な計測が容易(何が起こっているか, コンポーネントは正しいかなどの計測が容易)

Integrity Management(保守管理)に関してはホワイトリストによる方法 (実行してもよいソフトウェアのみ実行を許す方法) に切り替えるとしており、具体的には、1)システム

の安全な起動(Secure Boot)、2)システム保全スキャン(System Integrity Scan)、3)信頼されたネットワーク接続(TNC：Trusted Network Connect)という3つの要素から成り立っている。

米国時間で2007年8月27日（日本では28日）インテルは第2世代のvProプロセッサ・テクノロジー（コード名：Weybridge）を発表した。これにより仮想化に関わるセキュリティ機能として「インテル トラステッド・エグゼキューション・テクノロジー」、仮想化のためのソフトウェアを rootkit などから保護する技術「ダイレクト I/O 対応インテル バーチャライゼーション・テクノロジー」、悪意あるパターンを検出する機能強化として「システム・ディフェンス・ネットワーク・フィルター」、OS 不在時・休止時にもクライアントを管理する「インテル Embedded Trust Agent」などの新機能が追加された。

新機能	基盤テクノロジー
仮想化のためのソフトウェアを検出の難しいrootkitやhyper-jack攻撃から保護	Intel Trusted Execution Technology Direct I/O Virtualization Technology
ネットワーク・トラフィック中の悪意あるパターンの検出機能の強化	時間ベースのシステム・ディフェンス・ネットワークフィルター
ネットワーク・セキュリティを維持しながらOS の不在・休止時にもクライアント管理を実現	Intel Embedded Trust Agentを通じてIEEE 802.1x及びCisco NACをサポート

表4.1　第2世代vProの新セキュリティ機能

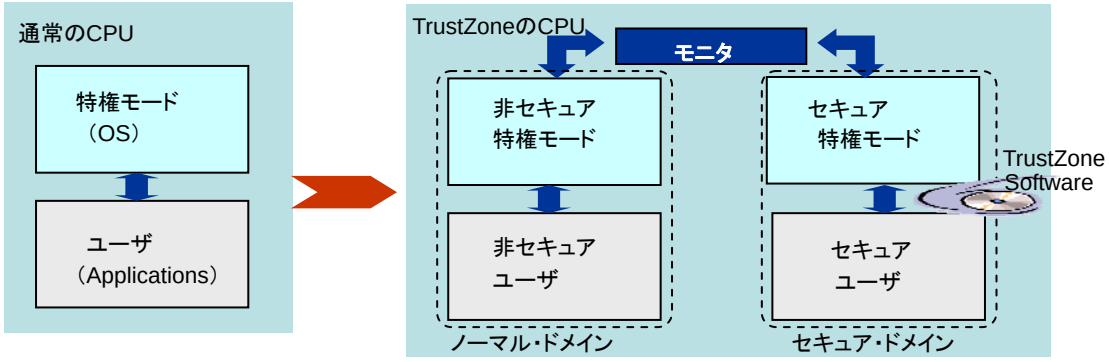
また 2007 年 9 月 18 日に米国で発表されたプレスリリースでは近い将来新しいチップセット「Eaglelake」（開発コード名）、TPM の統合、よりセキュアかつより管理が容易なデータ暗号化ソリューション「Danbury」（開発コード名）といった新しい構成要素が採用され、ソフトウェア・セキュリティー・ソリューションの向上を目指すとしている。

(2) ARM社のTrustZone

TrustedZoneは、ハードウェア・コンポーネントとソフトウェア・コンポーネントの組み合わせによりコア・プロセッサの消費電力、性能、実装面積にほとんど影響をせずにセキュアなハード・ソフト統合型プラットフォームを形成するもので、英 ARM 社（ソフトウェアの部分はフランスの Trusted Logic 社との共同開発）が開発した。これはマイクロプロセッサ自体の中にハードウェアで分離された実行環境を実装するもので、そのセキュリティ状態はCPUによってシステム全体が監視されるため、メモリや周辺も保護される。

TrustZone技術におけるドメイン分離の概要を図5に示す。1つの物理CPU上に仮想的にノーマルCPUとセキュアCPUを生成してノーマル・ドメインとセキュア・ドメインを分離し、ノーマル・ドメインからセキュア・ドメインへは直接アクセスはできず、必ずゲートキーパ役の「モニタ」の制御を受ける。CPU以外ではキャッシュ、メモリ、ペリフェラルもノーマル／セキュアの各ドメインで二重化され、特定のペリフェラルはノーマル CPU から隠蔽さ

れる。また、暗号、認証、DRM (Digital Right Management)などはセキュア・ドメイン上のソフトウェアにて処理または管理される。



参考：“TrustZone Technology Overview” Security Foundation by ARM
http://www.arm.com/products/esd/trustzone_home

図 4.4 英 ARM 社の TrustZone の概要

システムのセキュリティ要素がコアハードウェアに組み込まれているのでセキュリティが各機器の中心的部の本質的な機能として維持され、コアの面積や性能にはほとんど影響されないのが特徴である。 モニタがシステムをセキュア・ドメインに切り替えるとプロセッサが付加的に特権を与えて信頼性の高いコードを実行できるようになり、認証、署名、セキュアトランザクション処理などのタスクが実行可能となる。このあと TrustZone ソフトウェアがアプリケーションのセキュリティ部分のみを分離して、保護された専用メモリ (RAM)、専用固定ストレージ (フラッシュ)、SIM (Subscriber Identity Module) カードや TPM、暗号アクセレータ (暗号強度の強化)、信頼できるユーザーインターフェースへの排他アクセスを可能とする。また、各種 OS (Linux, Palm OS, Symbian, Windows CE 等) に互換性を有する柔軟性と移植性にも対応している点に特長がある。

2005 年 10 月に WindRiver 社が Linux ベースのプラットフォームでの TrustZone サポートを発表、2006 年 2 月には蘭の Philips Software 社が携帯電話の DRM に TrustZone を使って実装するなど実用化も軌道にのりつつある。

(3) Microsoft 社の BitLocker (TPM v-1.2 採用)

2003 年マイクロソフトは Windows Vista でセキュリティチップを使った NGSCB (Next-Generation Secure Computing base) という不正操作防止の機能を採用する計画を発表したが、2005 年 5 月 NGSCB の機能の一部である BitLocker Drive Encryption をサポートするにとどまると発表した。

Bit Locker は互換性のある TPM チップと BIOS が実装されている前提のもとに、1)Windows の OS ボリュームの古ボリューム暗号化機能と、2)TPM とブートストラップの整合性チェック後 Windows Vista の起動を許可する機能を持つ。互換性のある TPM が装備されたコンピュータでは、コンピュータを起動するたびに BIOS、マスタ ブート レコード (MBR)、ブートセクタ、ブートマネージャのコードなどの各初期起動コンポーネントが実行されるコー

ドを調べ、ハッシュ値を計算してその値をプラットフォーム構成レジスタ（PCR)に格納する。これはブートセクタに感染する rootkit 対策で、改竄されれば起動停止、ディスク、正確には Windows ボリュームが復号できなくなるという現実的な選択である。

TPM は Version1.2 で TCG が定義した Static Root of Trust Measurement と TPM をサポートする BIOS が必要とされるが、TPM1.2 対応の装備がなくても USB メモリ・キーによるドライブ暗号化機能をサポートした簡易版 BitLocker がある。

（4）その他（マルチコアチップや仮想化技術等）

1 つの LSI チップ上に複数の汎用プロセッサを置くものを CMP（chip multiprocessor）という。この場合個々のプロセッサをプロセッサ･コアあるいは単にコアともいうので、CMP はマルチコア（multi-core）チップとも呼ばれる。ちなみに同一チップ上に暗号処理やストリーム処理など専用プロセッサを混載したものは SoC（System on Chip）と呼ばれる。CMP は当初単一プロセッサの並列化による性能向上のために出現したが、LSI チップ自体が科学技術用超高速計算機（HPC: High Performance Computing）や高性能サーバー、PC から携帯電話、情報家電、PDA、ゲーム機、自動車といった社会の隅々まで浸透する文字通り Pervasive Computing の時代となり、信頼性，安全性に対する要求が極めて高くなってきた。特にコンシューマ向けの組込み機器においては性能は最重要要求ではなくなり、省電力とディペンダビリティ（信頼性と安全性の統合されたセキュア概念）が強く求められるようになっている。

NEC ではマルチコアチップを活用してコアを OS のレベルで独立させ、ウイルスに感染したコアだけ切り離して終了させる機能を開発している。

また Intel VT/AMD-V のようなハードウェアによる仮想化支援機能と相まって、組込み分野においても前述のように仮想化技術が使われるようになった。これにより 1 つの OS（ホスト OS）と仮想化ソフトウェアの上に複数の OS（ゲスト OS）を分割しておくことにより、1 つの OS の障害が他の OS に及ばないようにして被害を最小限に抑えることが実現できる。また、不安定なアプリケーションだけを別の OS に分離することにより他のアプリケーションの安定動作を確保することができる。このほか、例えば図 4.5 に示すように、リアルタイム OS（RTOS）と Linux を仮想化レイヤに載せることによって、実時間性と Linux の豊富な資産をそのまま利用することも可能となる。

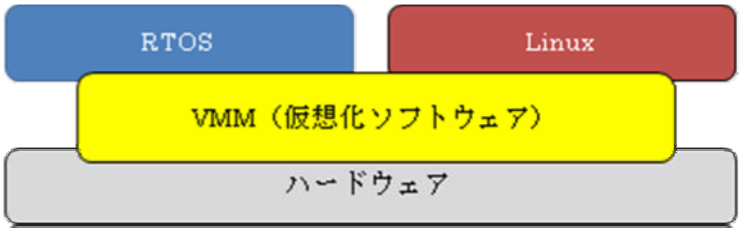


図 4.5　仮想化のアーキテクチャ

（参考：腰越　修「ハードウェア仮想化による複数 OS の並列実行」インタフェース　Nov. 2007）

・情報システムや製品の情報セキュリティ対策は、基盤となるハードウェアが正しく動作することが前提条件である。
・チップ自体の耐タンパー性を高めて、それをベースとしてプラットフォーム全体をセキュアにしようとするものがセキュリティチップであり、暗号、認証、DRM（Digital Rights Management）などを処理するセキュアなドメインをユーザードメインから分離することによりセキュリティを実現する。
・セキュリティチップにはいくつかの業界標準が存在する。
・1 つの LSI の上に複数の CPU を搭載し、異なる CPU 上に異種（リアルタイム性やセキュア機能等）の実行環境を載せてセキュリティと省電力を同時に実現しようとするアプローチもあり、仮想技術との連携で、仮想化の負荷の軽減やリアルタイム性の保証などにも役立つ。

《参考文献》

1) 2005 年度経産省委託事業「信頼できるコンピューティング環境の実現に向けて」2006 年 3 月日本画像情報マネジメント協会」
2) David Grawrock “The Intel Safer Computing Initiative - Building Blocks for Trusted Computing” Intel Press First Printing March, 2006
3) 【レポート】インテル，メディア向け最新セキュリティ技術説明会
http://journal.mycom.co.jp/articles/2007/06/08/IntelTXT/index.html
4) Intel News Release Aug. 27, 2007 “New Intel® vPro™ Processor Technology Fortifies Security for Business PCs”
http://www.intel.com/pressroom/archive/releases/20070827comp.htm
5) インテルとハイテク業界の急速な技術革新計画について講演（2007 年 9 月 18 日に米国発表プレスリリースの抄訳）
http://www.intel.co.jp/jp/intel/pr/press2007/index.htm#Sep
6) ARM, Richard York「CPU セキュリティの新しい基盤」（2003 年 5 月）
http://www.jp.arm.com/products/pdf/trustzone.pdf
7) Juniper Networks White Paper「ネットワーク・アクセス・コントロールの標準化の重要性」Part Number 200205-001 2006 年 11 月
http://juniper.co.jp/solutions/literature/white_papers/200205.pdf
8) 日経産業新聞「携帯・家電にマルチコアチップ，ウイルス感染特定部を停止」2007.10.3 記事（11 面）
9) 腰越 修「ハードウェア仮想化による複数 OS の並列実行」インタフェース Nov. 2007

第 5 章　情報セキュリティ対策に関する規格・評価制度

前章で、組込みシステムに関する情報セキュリティ対策の一部を紹介したが、必要な対策は製品分野や運用方法によって異なり、網羅することは不可能である。

しかし国内外で多くの分野において情報セキュリティ対策の検討が進んでおり、その一部は標準規格やガイドラインとして制定されている。また、評価制度として国内で運用されているものもある。

情報セキュリティ対策を検討する場合にこれらの標準規格やガイドライン、評価制度を参考にすれば多くの手間を省くことが出来るし、検討結果の妥当性を第三者に検証してもらうことも容易になるので、一読されることをお薦めする。

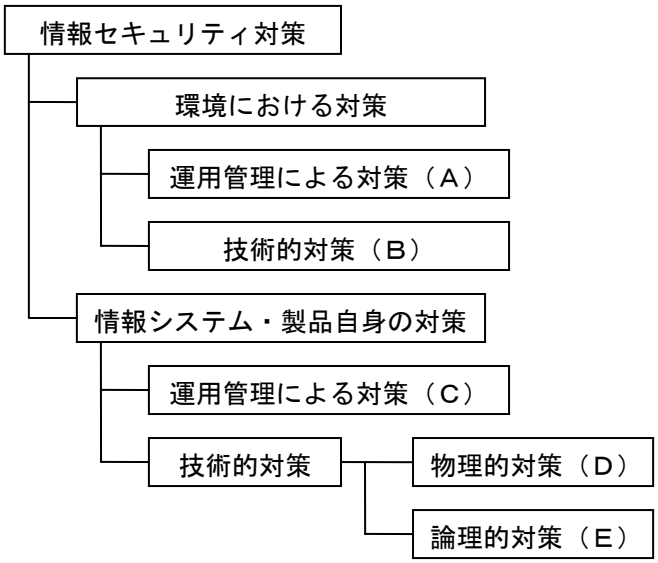


図 5.1　情報セキュリティ対策の区分

但し、これらの規格は歴史的に発展してきた経緯から様々な異なった切り口で対象物を捉えており、第 1 章で既述した情報セキュリティ対策とは若干異なった区分となっている。

そこで、以下に代表的な規格と評価制度を紹介し、組込みシステムに適用した場合にはどうなるかを検討したい。

5.1 ITセキュリティ評価及び認証制度

本制度は経済産業省傘下の独立行政法人情報処理推進機構（IPA）と独立行政法人製品評価技術基板機構（NITE）および評価機関としての民間企業で運営されているものである。制度の詳細に関する各種資料は IPA セキュリティセンターの以下の URL からダウンロード可能なので、一読されることをお勧めする。

http://www.ipa.go.jp/security/jisec/index.html

資料によれば、本制度は『IT 製品・システムについて、セキュリティ機能や目標とする保証レベルを、情報セキュリティの国際標準 ISO/IEC 15408 に基づいて、第三者が評価し、結果を公的に検証し、公開する制度』とされている。また、『評価対象は、デジタル複合機、IC カード、PKI（登録局、認証局）、デジタルカメラ、金融端末、ファイアウォール、データベース管理、オペレーティングシステム（OS）、生体認証システム、業務アプリケーションなど、ハードウェア、ソフトウェア、さらに情報システム全体など』とされており、組込みシステムも対象として取り上げられている。

この制度に則って評価・認証を受ければ評価結果は国内ばかりでなく国際的にも一定範囲で認められるので多大な効果があるが、評価・認証を受けるにはかなりの費用と期間が必要となるので二の足を踏まれる向きも多い。

しかし制度としては『第三者が評価し、結果を公的に検証し、公開する』ことを目的としているが、制度の基盤となる情報セキュリティの国際標準 ISO/IEC 15408 は情報セキュリティの概念や用語、情報セキュリティに関する仕様書の書き方、情報セキュリティ機能のコンポーネントなどについての標準が示されており、この標準の和訳や国内での解釈の統一も IPA セキュリティセンターが担当しているので、組込みシステム開発組織内での私的検証を行う際にもたいへん参考になるものである。

5.2 ISO/IEC 15408

本規格は 5.1 で述べた IT セキュリティ評価及び認証制度の基盤となる情報セキュリティの国際標準規格であり、Common Criteria、JIS X5070 と同一規格である。

規格の内容や解釈は必要に応じて改訂されるが、その順序は

①Common Criteria ⇒ ②ISO/IEC 15408 ⇒ ③JIS X5070

となるので、Common Criteria を参照すると情報セキュリティに関する議論の動向が良くわかる。Common Criteria も前述の URL からダウンロードできるので、用語はやや難解だが、

ぜひ一読願いたい。

情報セキュリティ分野ではこれらの規格の総称として“CC”という表記を使用することが多いので、本書でもCCと表記する。

この規格は図5.1に示した情報セキュリティ対策のうちで（E）技術的且つ論理的な対策に用いられる機能（本人確認機能、アクセス制御機能等）が中心となっており、標準となる機能コンポーネントが示されている。

物理的な対策や環境における対策については標準が示されていないが、評価機関の評価対象にはなるので、評価機関の評価者の判断に委ねられた形となっている。また、暗号機能に関しては、機能を適切に使うという観点で取り上げているだけで、暗号機能自体は対象としていない。暗号については5.3および5.4で紹介する。

本規格で最も重要な事は、情報セキュリティの概念や用語および情報セキュリティに関する仕様書の書き方の標準が示されていることである。

これによって立場や専門の異なる関係者（マーケティング担当者、ハードウェア設計者、ソフトウェア設計者、品質保証担当者、保守担当者等）が誤解なく厳密な議論が可能となるし、場合によっては情報システムや製品の調達担当者に情報セキュリティ機能と制約事項を正しく理解してもらうにも役立つものである。

5.2.1　CCの基本的考え方

CC 序説には『CC は、独立したセキュリティ評価結果間の比較を可能にするものである。このため、CC は IT 製品のセキュリティ機能性とセキュリティ評価の際に当該 IT 製品に適用される保証手段に関する共通要件のセットを規定している。当該 IT 製品は、ハードウェア、ファームウェア、またはソフトウェアに実装されることがある。』と記されている。

独立したセキュリティ評価結果間の比較を可能にすることは、異なった国や評価機関での評価結果を相互に認め合うことを可能とする前提条件であり、技術立国日本にとって避けては通れない重要課題である。

これを実現するためにCCは、情報セキュリティに係る“機能”それ自体と、機能が正しく実装されていることの“保証”を別けて取り扱うという考え方にたっている。

情報セキュリティ機能は、それが対抗しようとしている情報セキュリティの侵害に対して十分であることが必要なのであって、システムや製品によって必要な機能や性能は異なる。また、運用環境によっても変わるので単純比較は困難である。精々、同一環境下で使用される類似仕様の複数製品間でしか比較しても意味がないと考えられる。

一方、機能の実装の保証は一般的な品質保証の考え方そのものであり、設計～製造・テストの工程に係る文書、図面、報告書などの詳細な評価によって実現できるので、結果の比較は可能である。

そこで、CCでは評価行為の詳細さを示す評価保証レベルEvaluation Assurance Level（以

下 EAL と記す）という指標を用いている。EAL はレベル 1 からレベル 7 までが定義され、数字が多いほど厳密な保証が得られるもので、日本ではレベル 1 からレベル 4 までが使われている。

図 5.2 に示す組込みソフトウェア開発の V モデルの例では、EAL1 は機能仕様レベル、EAL2 は構造仕様レベルまで確認したことを示し、EAL4 ではコーディングまで確認したことを示している。

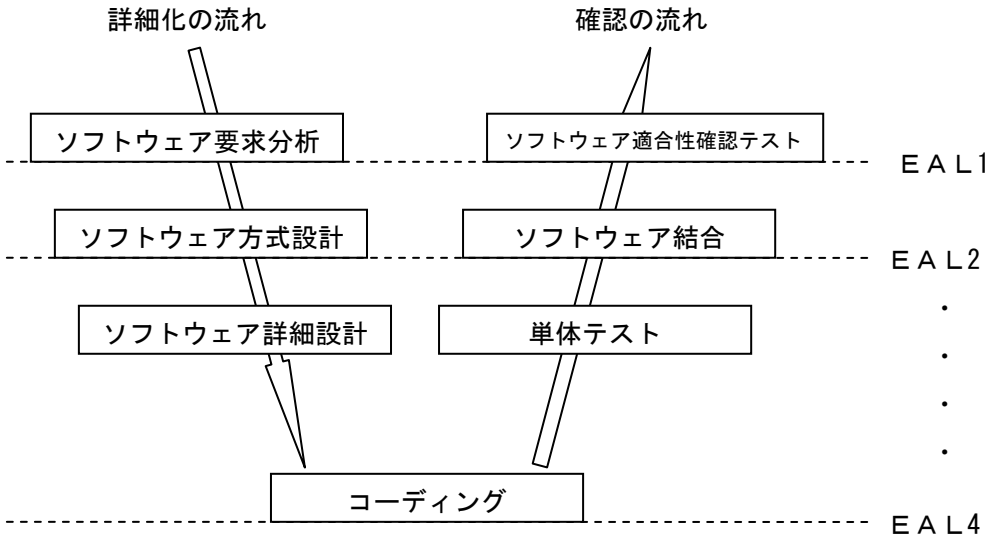


図 5.2　組込みソフトウェア開発の V モデル

（出典：IPA-SEC 組込みソフトウェア向け開発プロセスガイドライン）

但し、ここに示したものは大まかなイメージであり、厳密さを欠いている。詳しくは CC Part3 セキュリティ保証コンポーネントを参照願いたい。

EAL とは、『意図した情報セキュリティ機能が正しく実装されているということを、どの程度詳細に評価したか』を示す指標であり、情報システムや製品のセキュリティ機能の強さを示すものではないことを強調しておきたい。

さて、CC では情報セキュリティ問題だけを取り上げているので、情報システムや製品の開発にあたって作成される通常の製品仕様書や機能仕様書を基盤として議論や評価を行うのは極めて効率が悪い。

そこで CC では情報セキュリティ問題に特化した仕様書の作成を求めている。
これが、5.2.2 項で述べる“セキュリティターゲット”（Security Target）である。

CC の評価活動はセキュリティターゲット（以下 ST と記す）が出発点であり、これがないと評価対象としないことになっているほど重要な仕様書である。

ST の記載内容は 5.2.2 を参照戴きたいが、一言で表現すれば『設計者が対象製品の情報

セキュリティ機能を決定した際の思考過程』を記したものである。

ST 自体の記述内容に矛盾がなく、評価時点での情報技術レベルや情報セキュリティに対する攻撃の発生状況等からみても妥当な対策が盛り込まれていると判定されれば評価結果は合格となる。

その後、図 5.2 に示す詳細化の流れと確認の流れに沿って、設計ドキュメント類の評価および情報システムや製品自体の評価が行われる。

なお、評価だけでなく開発に際しても情報セキュリティ機能に関しては ST が出発点になり、以後、図 5.2 に例示したような詳細化の流れと確認の流れで開発を進めることが望ましい。

ST が出発点となるので、ST 評価はそれ自身の妥当性検証（Validation）のみとなるが、以後の評価段階では常に上流段階の設計ドキュメントが存在することになるので、対象段階の設計ドキュメントはそれ自身の妥当性検証（Validation）および上位ドキュメントとの照合による検証（Verification）によって評価される。

EAL によってどの段階まで評価するかは異なるが、ST から始まって上流から下流へと評価を進めるとともに、下流から上流へと遡行した評価も行われるので、上位設計で規定していない隠し機能（バックドアなど）を下流工程でまぎれ込ませても検出できることになる。

5.2.2 セキュリティターゲット

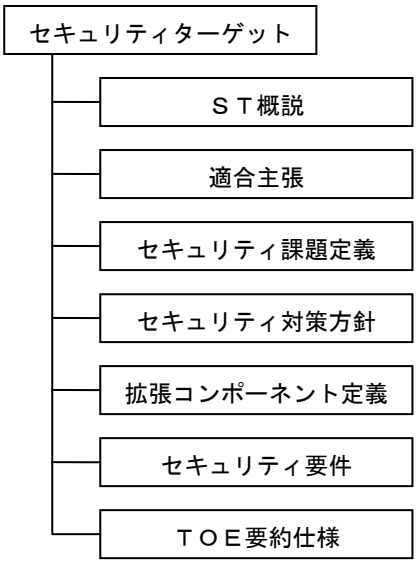


図 5.3　セキュリティターゲットの内容

セキュリティターゲットとは、対象とする情報システムや製品の情報セキュリティ機能に関する部分だけを取り上げて、ISO/IEC 15408 に規定された用語、内容、様式で記述した仕様書であり、通常は“ST”と表記されることが多い。

情報セキュリティ機能に関する部分だけを取り上げているので、情報セキュリティに関する議論や検証を行う場合には使い勝手の良い便利なものである。

ST の大まかな内容を図 5.3 に示す。

ST 概説：

この部分は仕様書と対象物を誤り無く対応付けることと、仕様書が記述している対象物

の概要を特にセキュリティ機能に関して読者に理解させることを目的としている。

したがって、この仕様書を特定する記述（ST のタイトル、バージョン、作成者、公表日など）と、対象物を特定する記述（開発者名、製品名、バージョン番号など）および対象物の使用法や主なセキュリティ機能の特徴などを記述する。

適合主張：

ここでは ST が準拠する CC のバージョン、この ST が独立して作成されたのか、他の仕様書を基盤とするのかなどを記述する。

セキュリティ課題定義：

ここには対象物が対処する必要のあるセキュリティに関する課題、即ち情報資産や脅威、脅威を発生させる攻撃能力や攻撃手法、セキュリティポリシーや前提条件などを記述する。

セキュリティ対策方針：

ここではセキュリティ課題定義によって示された課題に対し意図している解決策を簡潔に且つ抽象的に述べる。これは上位レベルの解決策であり、運用環境による解決策と対象物のセキュリティ機能による解決策の 2 つが存在する。

セキュリティ対策方針はセキュリティ課題定義と対応付けられ、全ての課題が十分に対策されていることが説明されていなければならない。

拡張コンポーネント定義：

CC に定義されたセキュリティ要件に規定されたコンポーネントを使用せず、独自のコンポーネントを使用する場合に記述するものであるが、一般的には不要であろう。

セキュリティ要件にはセキュリティ機能要件とセキュリティ保証要件があるが、ここでは詳細は省力する。詳しくは CC Part2 セキュリティ機能コンポーネントおよび CC Part3 セキュリティ保証コンポーネントを参照願いたい。ここでは、セキュリティ対策方針を受けて詳細化し、CC に定義されたコンポーネントと対応付ける記述と考えて戴きたい。これも全ての対策方針が十分にコンポーネントによって実現されていることを説明する必要がある。

TOE 要約仕様：

TOE という用語は Target of Evaluation の略であるが、ST の対象物を示すことばで、CC の記述では良く使われる表現である。ここは対象物がセキュリティに関する機能要件を実現する一般的な技術的メカニズムを説明する部分である。例えば、本人を確認するのに ID とパスワードを使用するのか、虹彩のパターンによる生体認証を使うのかということを記述する。

以上 ST の概略を説明したが､極力直感的に把握していただくために厳密さは犠牲にした。詳細は CC Part1 付属書Ａを参照願いたい。

また、ST を理解するには具体的な製品の ST を読むことが早道である。但し、ST は個別の製品の仕様書であるので、公表例は少ない。特定の分野では Protection Profile (PP) と呼ばれる複数製品に適用できる共通的なセキュリティに関する仕様書が公表されているので、これを読んでいただくと参考になると思う。

5.3 暗号モジュール試験及び認証制度

製品認証制度の１つとして独立行政法人 情報処理推進機構（IPA）により運用されるもので、Japan Cryptographic Module Validation Program の頭文字をとって JCMVP と呼ばれているものである。

本制度は暗号モジュールのみを対象としている。電子政府推奨暗号リスト等に記載されている暗号化機能、ハッシュ機能、署名機能等の承認されたセキュリティ機能を実装したハードウェア、ソフトウェア等から構成される暗号モジュールが、その内部に格納するセキュリティ機能並びに暗号鍵及びパスワード等の重要情報を適切に保護していることを証明するため、第三者による試験及び認証を組織的に実施するものである。

この制度の運用によって、利用者は暗号モジュールのセキュリティ機能等に関する正確で詳細な情報を把握することが可能となる。

暗号機能を搭載する組込みシステムを開発する必要がある場合、ぜひ、下記 URL を参照願いたい。

http://www.ipa.go.jp/security/jcmvp/index.html

5.4 暗号技術評価委員会（CRYPTREC）

CRYPTREC とは Cryptography Research and Evaluation Committees の略であり、電子政府推奨暗号の安全性を評価・監視し、暗号モジュール評価基準等の策定を検討するプロジェクトである。総務省及び経済産業省が共同で開催する暗号技術検討会（座長：今井秀樹中央大学教授）と、独立行政法人情報通信研究機構（NICT）及び独立行政法人情報処理推進機構（IPA）が共同で開催する暗号技術監視委員会（委員長：今井秀樹中央大学教授）及び暗号モジュール委員会（委員長：松本勉横浜国立大学教授）で構成されている。

CRYPTREC は、公募された暗号技術及び業界で広く利用されている暗号技術を評価・検討し、安全性及び実装性能ともに優れたものを選択した。

この評価結果を踏まえ 2003 年 2 月 20 日に総務省と経済産業省が公表したものが「電子政府」における調達のための推奨すべき暗号のリスト(電子政府推奨暗号リスト)である。

省庁の調達を目的としたものではあるが、暗号技術に関する多くの情報を参照することができるので、暗号機能を搭載する組込みシステムを開発する必要がある場合、ぜひ、下記 URL を参照願いたい。

http://www.ipa.go.jp/security/enc/CRYPTREC/index.html

5.5 ISMS適合性評価制度

ISMS とは情報セキュリティマネジメントシステム（Information Security Management System：以下、ISMS という）のことである。本制度は経済産業省所管の公益法人である財団法人日本情報処理開発協会（JIPDEC）情報マネジメント推進センターを認定機関とし、JIPDEC から認証機関として認定された民間企業の協力で運営されている。

制度の詳細に関する各種資料は JIPDEC 情報マネジメント推進センターの以下の URL からダウンロード可能なので、一読されることをお勧めする。

http://www.isms.jipdec.jp/

資料によれば、『ISMS とは、個別の問題ごとの技術対策の他に、組織のマネジメントとして、自らのリスクアセスメントにより必要なセキュリティレベルを決め、プランを持ち、資源配分して、システムを運用することである。』とされている。

また、『組織において ISMS を確立、導入、運用、監視、見直し、維持し、かつその ISMS の有効性を改善する際に、プロセスアプローチを採用すること』がポイントであると述べている。

つまり、ISMS 基本方針を基に、

- ●Plan　：　情報セキュリティ対策の具体的計画・目標を策定する。
- ●Do　　：　計画に基づいて対策の導入・運用を行う。
- ●Check　：　実施した結果の監視・見直しを行う。
- ●Act　　：　経営陣による改善・処置を行う。

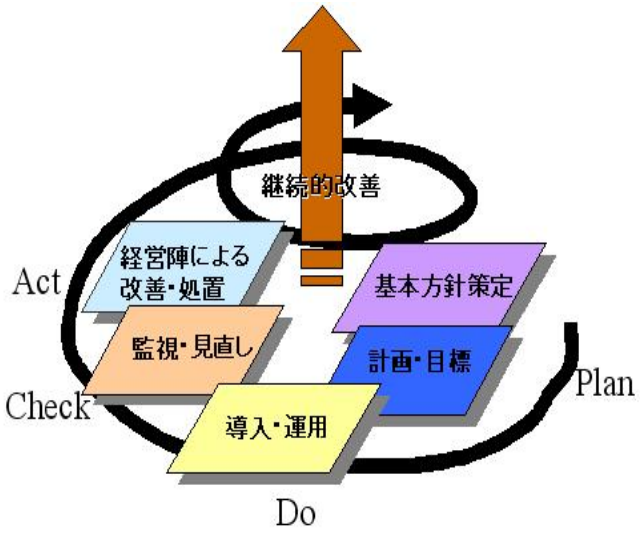


図 5.4　ISMS の PDCA サイクル

このPDCAサイクルを継続的に繰り返し、組織の情報セキュリティレベルの向上を図ることを目指しているわけである。

従来、我が国では情報処理サービス業のコンピュータシステムが十分な安全対策を実施しているかどうかを認定する制度として、昭和 56 年 7 月 20 日通商産業省告 示 342 号による「情報システム安全対策実施事業所認定制度」(以下、安対制度という)があった。安対制度では、集中管理されていた情報システムの施設・ 設備等の物理的な対策に比較的重点がおかれていたが、技術的対策だけでなく人的セキュリティ対策を含む組織全体のマネジメントを確立する必要性がでてきた。

こうした状況を受けて、経済産業省では「情報セキュリティ管理に関する国際的なスタンダードの導入および情報処理サービス業情報システム安全対策実施事業所認定制度の改革（平成 12 年 7 月 31 日)」を公表するとともに、従来の安対制度を平成 13 年 3 月 31 日をもって廃止した。

この安対制度の廃止に伴い、技術的なセキュリティのほかに、人間系の運用・管理面をバランス良く取り込み、時代のニーズに合わせた新しい制度として本制度が創設された。

制度創設の経緯からみても、組込みソフトウェア開発者にとって最も重視すべき制度のひとつである。

5.6　ISO/IEC 27001

ISO/IEC 27001:2005（Information technology －Security techniques－Information security management systems－Requirements：情報技術－セキュリティ技術－情報セキュリティマネジメントシステム－要求事項）は、組織が ISMS を構築するための要求事項をまとめた国際規格である。

また、前述の ISMS 適合性評価制度の認証基準 JIS Q 27001：2006 は、この評価制度において、第三者である認証機関が本制度の認証を希望する組織の適合性を評価するための基準であり、ISO/IEC 27001：2005 と同一の規格である。ISO/IEC 27001 で求めている情報セキュリティマネジメントシステムにおけるセキュリティ管理策を図 5.5 に示す。

附属書 A「管理目的と管理策」

1. セキュリティ基本方針
2. 情報セキュリティのための組織
3. 資産の管理
4. 人的資源のセキュリティ
5. 物理的及び環境的セキュリティ
6. 通信及び運用管理
7. アクセス制御
8. 情報システムの取得、開発及び保守
9. 情報セキュリティインシデントの管理
10. 事業継続管理
11. 順守

図 5.5　セキュリティ管理策

ISMS の要求事項は、図 5.4 に示したように組織の自らの事業の活動全般及び直面するリスクを考慮して、文書化された ISMS を確立、導入、運用、監視、見直し、維持し、かつこれを継続的に改善することである。

具体的な進め方を図 5.6 に示す。

ISMS の確立にあたり ISMS の適用範囲を定義(STEP1)し、ISMS 基本方針を策定(STEP2)する。策定した ISMS の適用範囲及び基本方針に基づき、リスクアセスメントの体系的な取組方法を策定(STEP3)する。保護すべき情報資産に対するリスクを識別(STEP4)し、リスクアセスメントを実施(STEP5)する。リスクアセスメントの結果、リスクの受容ができない場合にはリスク対応の選択肢を明確にして評価(STEP6)する。リスク対応に基づき、実施すべき管理目的と管理策を選択(STEP7)する。

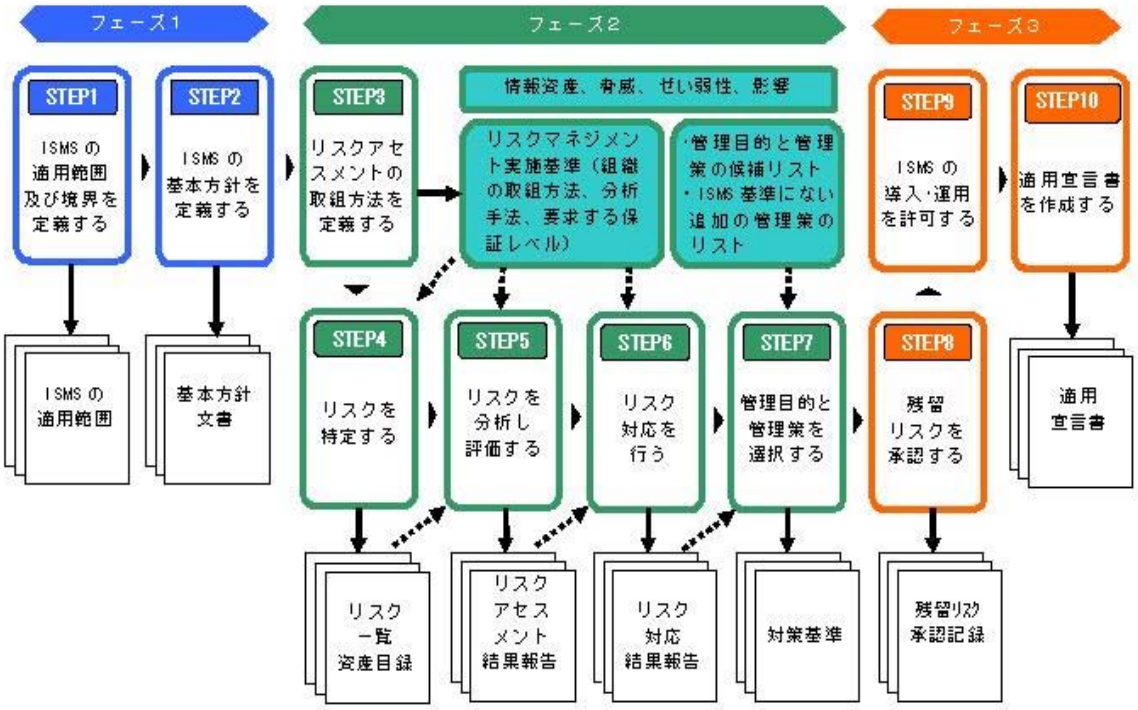


図 5.6　ISMS 確立のためのステップと作成文書

附属書 A「管理目的と管理策」にある全ての管理策が実施されなければならないわけではなく、リスクアセスメントに基づき、管理策を選択して実施できる。上記の管理策だけでなく、組織がリスクアセスメントやリスクマネジメントなどを通じて、必要と思われるより良い管理策を追加することができる。リスクアセスメントの結果、何が残留リスクなのか、残留リスクはどの程度あるのかを明確にした上で経営陣が承認し(STEP8)、ISMS を運用することを許可(STEP9)する。特に重要なことは、この選択については適用宣言書で明確に公表(STEP10)することにある。

（出典：JIPDEC　情報セキュリティマネジメントシステム（ISMS）とは　2006 年 2 月 7 日）

・情報システムや製品のセキュリティ機能や目標とする保証レベルを、情報セキュリティの国際標準 ISO/IEC 15408 に基づいて、第三者が評価し、結果を公的に検証し、公開する制度が IT セキュリティ評価及び認証制度である。
・IT セキュリティ評価及び認証制度の基盤となる情報セキュリティの国際標準規格が ISO/IEC 15408 であり、Common Criteria、JIS X5070 と同一規格である。
・ISO/IEC 15408 では、対象とする情報システムや製品の情報セキュリティ機能に関する部分だけを取り上げて、規定された用語、内容、様式で記述した仕様書を作成することになっている。
・暗号モジュールについては、暗号モジュール試験及び認証制度が存在する。

・暗号技術に関する情報を参照するには、暗号技術評価委員会（CRYPTREC）の報告書が適している。
・ISMS とは、個別の問題ごとの技術対策の他に、組織のマネジメントとして、自らのリスクアセスメントにより必要なセキュリティレベルを決め、プランを持ち、資源配分して、システムを運用することである。
・ISO/IEC 27001 は、組織が ISMS を構築するための要求事項をまとめた国際規格である。

第 6 章　機能安全の問題

機能安全は制御システムが安全に稼働するための国際標準規格である。コンピュータによるシステムの制御が大規模、複雑化する中で、安全に関する問題が大きな課題になっており、その対応を各国の個別の対応ではなく、国際的に共通の規則で行うことが、この 10 年の間に一つの流れとなっている。

機能安全は認証取得という側面もあるが、認証を取得するのは大手であり、大手メーカーから組込み系システムを受託開発する中・小規模のソフトウェアやハードウェアの開発企業（以下「組込み系企業」と呼ぶ）は認証取得に関係はしないであろう。本論は組込み系企業にとって有用であることを論じることを目的にしているが、機能安全の考えた方を習得することは極めて有益であると考える。機能安全を取り上げる理由である。

6.1 節では機能安全の定義と組込み企業が機能安全に取り組む利点を述べる。6.2 節では安全のライフサイクルを概観する。

機能安全は IEC 61508 で扱われる電気系の規格である。他に機械系の規格があり、機械と電気の両者の規格で機械電気からなる今日の制御システムの安全をカバーしている。その中で機能安全が大きくクローズアップされるのは、今日の制御がソフトウェアに大きく依存しているからである。IEC 61508 はシステムに安全機能を盛り込むためのトータルな工程を定め、さらにハードとソフトウェアの開発工程もトータルに定めている。

IEC 61508 はシステムの規格適合を第三者認証するものでもある。認証を取得するためにはそれらの全工程が規格通りになされたことを立証する文書が必要とされる。また安全度を示す基準に安全度水準があり、高い水準のものを取得するする場合には、ソフトウェアの品質を論理、数学的に証明するなどのことが必要とされる。

組込み系企業にとって、これらの技術的方法を習得することはシステムの品質を高めることに寄与し、また安全に関する要求仕様柄を習得することは、安全が時代の課題であるだけに、企業の業務領域を広めることになろう。

以上が 6.2 節までの内容であり、6.3 節は機能安全に関連する事柄を論じる。

機能安全が高い品質のシステムを要求することから、これを機に品質や安全性、また信頼性に関する知見を持つことは、組込み企業にとって重要な企業姿勢の表明になる。

機能安全のライフサイクルにおける検証と妥当性確認の重要性の認識は、安全に限らず開発一般においても重要な観点であり、さらに議論を深めれば、日本的な物を作っては新ためす試行錯誤の方法が、実は極めて合理的な方法であるといえ、そのような合理性に即した方法論の開発や教育が日本のソフトウェア産業の発展に通じるともいえる。

また、機能安全機能安全が情報セキュリティの問題も取り上げつつある最近の事情にもふれる。最後に、機能安全には直接関係するものではないが、複雑・微細となった最近のハードウェアの信頼性を向上させるテストツールにもふれる。

6.1 機能安全の基礎

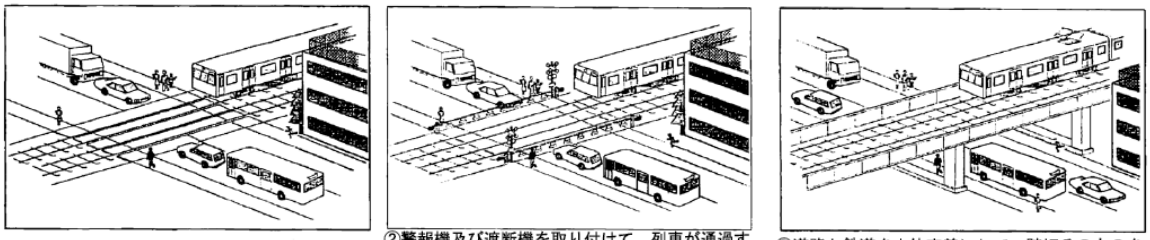
ここでいう「機能安全(Functional Safety)」とは国際電気標準規格 IEC 61508 の中に記述されている内容を指している。規格 IEC 61508 は「電気・電子・プログラマブル電子安全関連系の機能安全」に関して記述し、システムの安全性の分析、検証、認証に関する基準を定めている。その基準は国際電気標準会議(IEC)の規格(standards)であり、共通の規則や取り決めを標準化(standardization)したものである。後にもふれるが化学プラントなどに大きな事故が比較的多く発生したことが成立の背景になっている。

また、安全性の評価や認証は、評価や認証の対象が規格に準じているのか否かを第三者が判断して初めて客観性を持つが、わが国ではまだ評価認証の制度は創設されていない。

6.1.1 機能安全の定義

機能安全とは、危険をもたらす可能性がある機器やシステムに対して、制御機能により安全を保つことを指す。規格によれば、たとえば、過熱防止をする装置において、温度センサーによりモータを制御し温度調整をして過熱を防いでいる場合には機能安全というが、単に断熱材を設置して熱を遮断している場合には、特別な制御を働かせているわけではないので機能安全とはいわない[1)]。

また機能安全と異なる概念に「本質安全(inherent safety)」がある。危険そのものを取り除いた状態での安全をいう。道路と鉄道が同じ地面に交差している状態（①）から、鉄道の踏切に警報機や遮断機を設けて、鉄道と、路上を行き交う人間や自動車などとを制御している状態（②)、また踏切を無くして道路と鉄道を立体交差にして状態（③）の 3 例に関していえば、③は本質安全の状態、②は機能安全の状態、①は特別な装置による制御がなく、機能安全も本質安全もない、危険な状態といえる[2)]。

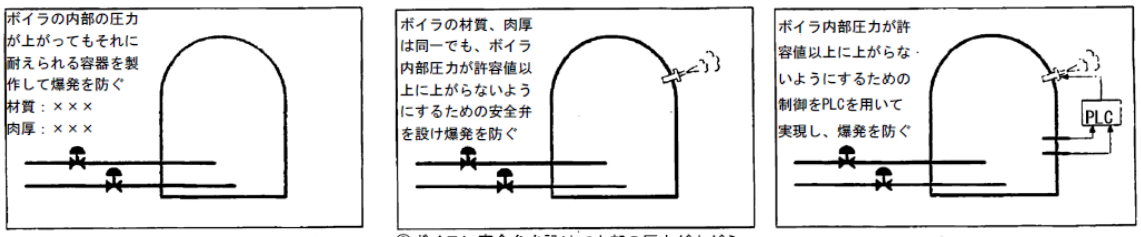


①道路に対して鉄道が交差して敷設された状態　②警報機及び遮断機を取り付けて、列車が通過する際に人や車が踏切内に侵入しないようにした状態　③道路と鉄道を立体交差にして、踏切そのものをなくした状態

図 6.1　踏切の事例

次のボイラの事例は制御をプログラムで行うというものである(PLC(Programmable Logic Controller）の利用)。機能安全 IEC 61508 は電気や電子、またプログラムなど、今日の IT

技術に依存するところが特徴である[2)]。



①ボイラの材料や肉厚を想定して爆発を防いだ段階　②ボイラに安全弁を設けて内部の圧力が上がらないようにした段階　③ボイラ内部の圧力を電子的に制御した段階。PLC：プログラマブルコントローラ

図 6.2　ボイラの事例

機能安全に関する定義は、上のような例示的な定義とは別に IEC 61508 固有の、より専門的な定義がある。しかし、その定義のためには、“安全機能（safety functions)”、“安全度（safety integrity)”、“安全関連系（safety-related systems)”の意味とその関係を理解しなければならない。これらは“機能安全”を定義するための定義項（定義するコトバ）だからである。このような用語に初めて接する読者は戸惑うであろうが、しかし、これらの用語はおおよそ理解できるとして、定義を続けてみる。やや長いが用語がどのように使用されているかを示すために、IEC 61508 の第 0 部、P.13 より以下に引用する[3)]。

一般に、制御つき装置システムにどのような重大な潜在的な危険（hazards）があるのかは、そのシステムが用いられる環境を想定し危険の解析を行い、仕様提示者や開発者によって事前に特定されていなければならない。その解析によって、個々の重大な潜在的な危険に対して適切な防御を保証するために、機能安全が必要か否かが決まる。もし必要であれば、機能安全は適切な方法で設計の中に考慮されなければならない。機能安全は潜在する危険を扱う単に一つの方法に過ぎない。設計による本質安全(inherent safety)のような、潜在する危険をなくしたり、少なくしたりする他の方法が、まず第一に重要である。

用語「安全関連的(*safety-related*)」は、次の機能の実行が要求されているシステムを記述するために使用される。それは、リスクが受容レベルに保たれていることを保証する機能である。そのような機能は、定義によれば、安全機能(*safety functions*)である。機能安全を達成するためには、2 つのタイプの要求事項が必要である：

安全機能要求事項(その機能が何をするか)　と

安全度(safety integrity)要求事項(問題なく実行されている安全機能の見込み)　である。

安全機能要求事項は、潜在的危険の解析から得られ、安全度要求事項はリスクアセスメントから得られる。安全度のレベルがより高くなればなるほど、危険な故障はより低くなる。何らかの技術で実行され、安全機能を実行するシステムは、どれも安全関連系（*safety-related system*）である。安全関連系は、ある装置制御システムから分けられるか

もしれないし、あるいはその制御装置システムはそれ自身安全機能を実行するかもしれない。後者のケースではその装置制御システムは安全関連系である。安全度のより高い水準は安全関連系のエンジニアリングにおいてはより厳密さが必要とされる。

以上が機能安全の定義である。定義項と被定義項を明確に分けた定義からは遠いが、機能安全が、制御システムに関係すること、ただし安全の要求項目を実現するための制御システムであること、安全機能は相互に関係していること、また安全は危険に対応するものであり、その対応度によって安全度のようなレベルが設けられていることが理解されたと思う。

このような安全の議論は直接組込み系技術と関係するものではない。組込み系企業にとって直接関係する事柄は技術的な課題、たとえばソフトウェアやシステムの品質をどう保つか、開発の効率をいかに上げるかであろう。とりわけ開発を専らとする組込み系企業にとっては、要求仕様はどれも one of them に過ぎない。

次に機能安全、すなわち IEC 61508 規格へ取り込むことの利点を述べる。

6.1.2 組込み系企業がIEC 61508 へ取り組む利点

IEC 61508 はシステムの認証の取得制度であるので、自社に所有等の権利のある、安全性に関係する制御システムを持たない、多くの中・小規模の組込み系企業にとっては、IEC 61508 の規格に関して細かく学ぶ必要はない。しかし次の点で、IEC 61508 の意図する方向を理解し、求められる技術や方法へ挑戦する価値はある。

(1) 技術的観点からの利点

IEC 61508 のいう制御システムは「電気・電子・プログラマブル電子系 (electrical/electronic/programmable electronic System、また E/E/PES、と表現)」であり、これは組込み系企業が対象とする組込系のシステムそのものである。IEC 61508 規格から仮に“安全”というコトバを除いたとしても、“E/E/PES”だけは残る。

IEC 61508 は、システムが安全の制御機能を実行するために必要とされる開発方法を提供している。組込み系の品質が問題となっている今日、その方法論は大いに学ぶべきである。

システムやソフトウェアの開発を、全ライフサイクルの中で展開すべきこと、高品質のソフトウェア開発のためには、数学・論理学的方法でソフトウェアを論証するべきなどの指摘が綴られている。

IEC 61508 は組込み系企業が学ぶべき技術や方法を提示している。

(2)「安全」という時代の要求を知る利点

「安全」という要求事項は時代の要求である。今日多くの組込みシステムにおいて「安

全」が語られないことはない。よって、「安全」は多かれ少なかれ組込み企業にとって、情報セキュリティと同様、対応すべき課題となろう。

もちろん「安全」への責任はメーカーにある。メーカーの仕事を請け負う組込み系企業にはその社会的責任を第一義的に負う必要はなく、組込み系企業は品質を第一に保つことが責任である。しかし、そうであるためにも、組込み系企業は、品質に密接に関係する概念である「信頼性」やまた「安全性」の違いをきちんとわきまえる必要がある。

それがまず第一である。しかし、方法を専らとするともいえる“組込み屋”が、安全という時代のコンテンツ（要求仕様）にアプローチし、メーカーに対して積極的に提案して行くこともできよう。IEC 61508 の求める方法は、形式手法などかなりの専門性を要する技術もあるので、そのような技術が安全、あるいは機能安全とどのように関係するのかを明確にする専門家が必要である。組込み系企業がその役割を担うことも有益と思われる。

6.2 IEC 61508 の概観

IEC 61508 の全体を概観し、すでに述べた IEC 61508 を学ぶこと等の利点を確認する。そのために IEC 61508 成立の背景を一瞥し、機能安全という規格が、安全というより大きな規格のどのような位置を占めるのかを確認し、そして IEC 61508 の規格がどのような構成になっているのかを論じる。そして、IEC 61508 の基本的概念である安全ライフサイクルの全体、またその工程の中に重要な位置を占めるハードウェア等からなるシステム、およびソフトウェアの安全ライフサイクルについても確認する。とりわけソフトウェアの安全ライフサイクルにおいては、検証と妥当性確認のライフサイクルが重要となる。

6.2.1 IEC 61508 の成立の背景

IEC 61508 制度に先立ち、1970-80 年代に、欧米の石油・化学プラントにおいていくつかの大事故が発生している。成立の経緯を文献[4]より概観する。

(1) フリックスボロ事故

1974 年、英国（イングランド中部） ナイプロ社のフリックスボロ化学工場でシクロヘキサンが漏洩し、蒸気雲爆発が発生。火災も数日続いた。最終的に 28 人が死亡、工場内外で 89 人が傷害を負った。

(2) セベソ惨事

1976 年、イタリア北部のセベソにあるイメクソ社の除草剤工場から、ダイオキシン蒸気が放出された。死者は発生しなかったものの、2000 人以上がダイオキシン中毒と診断され

た。当該地域に住んでいた人々は移住を強いられ、皮膚炎・内臓疾患などの障害、その後はガンの多発・奇形児の出産などの被害が発生し、最終的には 22 万人が被災したとされている。

(3) ボパール事故

1984 年、インド中部のボパールで、米国のユニオンカーバイド社殺虫剤工場からイソシアン酸メチル、塩化水素等を含む猛毒ガスが放出された。2000 人以上がほぼ即死したといわれているが 4000 人という説もある。50 万人が被害に合い、今なお 10 万人以上の人に医学的な手当てが必要といわれている。

(4) パイパーアルファ

1988 年、英国東の北海上の石油・ガスプラットフォームであるパイパーアルファが、ガス漏れへの引火から大規模火災、大爆発へと連鎖して焼け落ちた。167 名死亡。

(5) その他

この他にも 1989 年米国ヒューストンのフィリップス社での爆発事故や、原子力産業におけるスリーマイル島（米国、1979 年)、チェルノブイリ（旧ソ連、1986 年）などの事故が発生している。

セベソ惨事をきっかけとして EC（現 EU）では「セベソ指令」が 1982 年に発令された。「特定産業活動の大事故ハザードに対する欧州閣僚理事会指令」がそれである。プラントの巨大化と事故の大規模化、そのための対応技術ということが具体的な課題となった。その後、1996 年の「セベソ指令Ⅱ」では、セーフティ・マネージメントというヒューマン・ファクターに関する要項も組み込まれた。産業界にはこのような管理レポートの提出が義務づけられ、政府にはそれを審査する機関の設立が求められた。

ヨーロッパ各国、また米国の規則制定を経て、1999 年に国際規格として IEC 61508 が発行された。

日本においても、国際規格に合わせる WTO 加盟国の義務として、2000 年 IEC 61508 はそのまま JIS C 0508 として発行された。

6.2.2 IEC 61508 の位置づけ

安全の評価認証に関しては、ISO/IEC Guide 51 に、機械関係での ISO 12100（JIS B 0008/9)、電気関係における IEC 61508(JIS C 0508)など各種工業製品を対象にした「安全」の原則がまとめられ、標準化、規格化の動きが活発といわれている。機械系、電気系の階層図が IOS/IEC ガイド 51 に記されている。IEC 61508 がどの位置にあるのかをこの図から知るこ

とができる。その図を向殿政男氏の論文[5]より引用する。

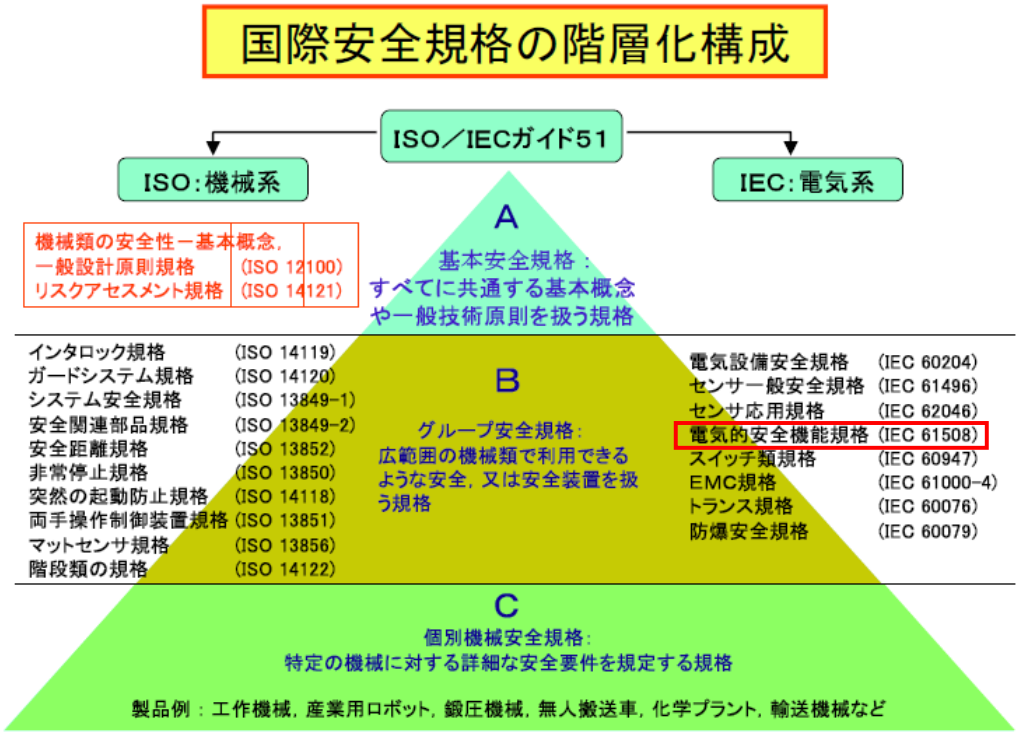


図 6.3　国際安全規格の階層化構成

IEC 61508 は電気系のグループ安全規格 B に属する（図中赤線枠）。また規格のグループによる違いは次の通りである[5]。

(1) A規格：すべての規格類で共通に利用できる基本概念や一般技術原則を扱う基本安全規格：ISO12100 は、基本安全規格である。
(2) B規格：広範囲の機械類で利用できるような安全規格や安全装置を扱うグループ安全規格
(3) C規格：特定の機械に対する詳細な安全規格を扱う個別機械安全規格

この引用は機械系（ISO）に関する言及であるが、これを電気系（IEC）に読み替えれば、IEC 61508 ということになる。IEC 61508 の特徴は、電気、電子に加え、プログラムの規格を加えたところにある。

また B グループである IEC 61508 は、C グループの個別適用分野規格の制定を促進する役割を担っている。「近年電気･電子･プログラマブル電子を応用した、組み込み機器に関する安全性への関心が高まっている。その結果安全性に関する B 規格である、IEC61508 及び、

その個別適用であるC 規格及び指針が次々と制定されている。」[6)]

これまでに制定されたIEC 61508関係の個別適用分野の規格一覧を同論文から引用する[7)]。

EN規格	EN61508	一般(計装)
	EM50126	鉄道
IEC規格	IEC6151	プロセス産業
	IEC61513	原子力
	IEC62061	産業機械
	IEC62304	医療
	IEC61800	モータ
ISO規格	ISO/WD26262	自動車
指針類	EEMUA	アラーム指針
	DFT STAN	防衛(英国)
	海軍規格(英国)	
	MISRA-SA	自動車(英国)
	計量ソフトウェア指針	モータ
	EMCガイドライン	IEE(英国)

表 6.1　IEC 61508 関係適用分野

この表から、安全、その中でもIEC 61508における機能安全の要求事項が、広い分野に適用されようとしていることがわかる。安全を電気系に強く依存しなければならない今日では、どの業界でもIEC 61508は無視できない存在になりつつある。

このことは、組込み系企業がIEC 61508に関与してゆくことがビジネス上の利点が大きいことを示している。

6.2.3　IEC 61508の規格書の一覧

IEC 61508は安全規格の中の電気系の規格と位置づけられるが、その規格書の一覧は次の表の通りである。制定年も示し、この規格がごく新しい、この10年の間のものであることを示す。

同規格には強い期待と関心が寄せられているが、技術革新の激しい分野の規格でもあり、近く改訂も行われるようである。

IEC 61508は第2部のタイトルが、「電気・電子・プログラマブル電子安全関連系に対する要求事項 (Requirements for electrical/electronic/programmable electronic safety-related systems)」とあるように、また第3章で「ソフトウェア」とあるように電子機器を活用したプログラム制御による安全の規格であることがわかる。かつ第3章のページ数が多いことにも示されているように、ソフトウェアに対する規格にも大きな重点を置いている。

規格の文は、技術革新の激しさを吸収するためでもなかろうが、抽象度の高い文章からなり、意味を同定することが容易ではない。

なお、第 0 部は全体のテクニカル・レポートであり、機能安全の紹介、IEC 61508 全体を概観するものであり、規格書ではない。

部	表題	頁数	制定年
第1部	General requirements 一般的要求事項	115	1998
第2部	Requirements for electrical/electronic/programmable electronic safety-related sysytems 電気･電子･プログララマブル電子安全関連系に対する要求事項	143	2000
第3部	Software requirements ソフトウェア要求事項	93	1998
第4部	Definitions and abbreviations 用語の定義と略語	53	1998
第5部	Examples of methods for the determination of safety integrity levels 安全度水準を決定するための方法の事例	57	1998
第6部	Guidelines on the application of IEC 61508-2 and IEC 61508-3 第2部と第3部を適用する上でのガイドライン	145	2000
第7部	Overview of techiniques and measures 技術と方法の概観	229	2000
第0部	Functional safety and IEC 61508 機能安全とIEC 61508	33	2005

表 6.2　IEC 61508 の規格書の一覧

6.2.4　IEC 61508 の基本概念

(1)「安全」の考え

制御機器システムは故障しうる。このシステムが安全に対応していたとして、故障したとしたならば、安全の機能は停止し、システムの規模にもよるが大きな事故になりかねない。IEC 61508 では、「絶対安全は存在し得ない、リスクは残る」と考える。すでに述べた機能安全の定義の中にも「リスクが受容レベルに保たれていること」という表現があるが、これは、機能安全の対処をしたとしても、システムには残存リスクがあり、ただしそのリスクが受容できるなら、安全と考えてよい、という考えである。

ちなみに、日本では、事故は絶対にあってはならぬもの、という一部感情的な考え方が先行し、リスクは残存するものだからそれを前提に対応するというような、論理的な、あるいはコストとの関係から確率論的に対応する考え方は取られて来なかったといわれる。

IEC 61508 はリスクを許容可能にする技術ともいえる。IEC 61508 では、そのリスクの大小とリスクへの対応可能性との関係を ALARP モデルで表している[8)]。

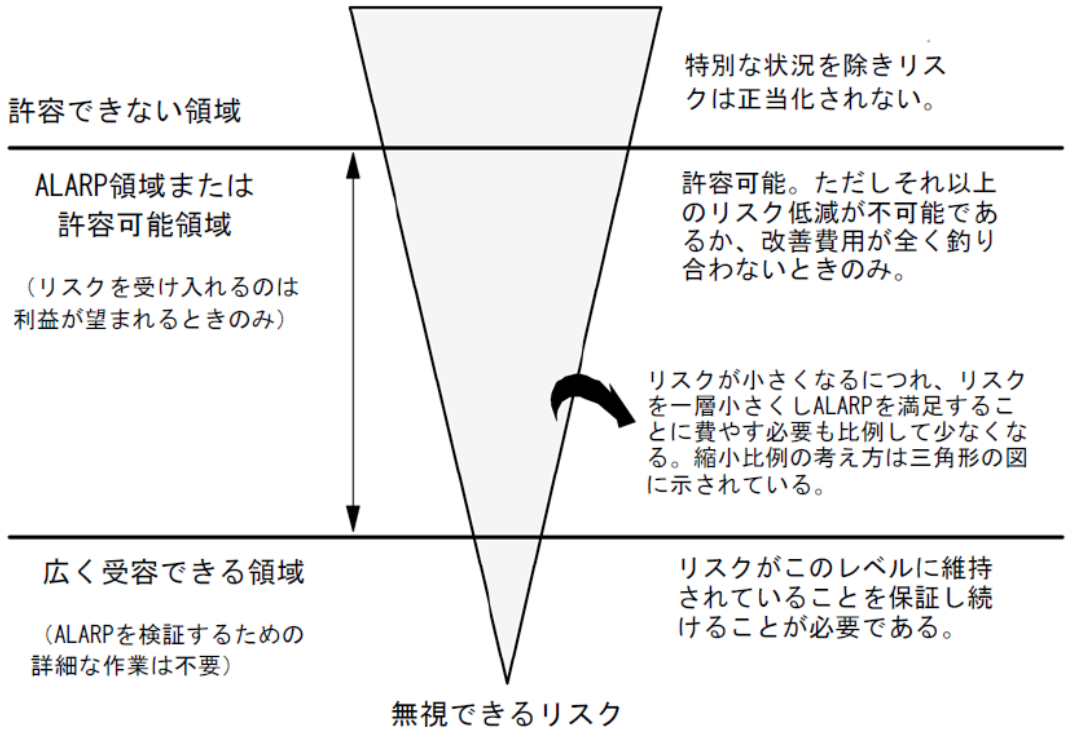


図 6.4　ＡＬＡＲＰモデル

ALARP とは、“As Low As Reasonably Practicable”の略である。リスクを“合理的に実行可能な範囲でできるだけ低くする”ことを意味している。

リスクの領域を、許容できない領域、許容できる領域（または ALARP 領域)、広く受容できる領域の 3 つに分けている。またリスクの大小を逆三角形で表し、リスク低減の方策や費用との関係で、許容できる領域、すなわち現実に対応しなければならない領域を特定している。すでに掲げた踏切やボイラの例も、可能な技術と同時に、その費用の合理性が現実的対応の可否を決める。

《故障の原因、安全度水準》

このように IEC 61508 ではリスクの大小を判断して、機能安全としての対応を行うが、その際リスクの内容（原因）に応じて、なすべき対応を確率論的設計、または定性的設計として行うべきことの基本的な考えを示している。

リスクの発生は、たとえばシステムを構成する機器部品の故障が考えられる。この場合、故障は何万回に 1 回故障することが確率的に予測可能なものと､そうでないものとがある。たとえば自動車のブレーキなどはメーカーが確率的な保証をしている。発生がランダムであるが確率的に故障を抑えているものである。これはランダムハードウェア故障(random

hardware failure)と呼ばれる。

他方、仕様段階、設計段階、プログラム実装段階などで把握できなかった不具合（故障）で、後日の問題が起こってからのみしか取り除くことができなかった故障は、原因と結果の関係が明確であるものとに分けられる。この因果的故障（systematic hardware failure）はその性質から確率でとらえられない。従って因果的故障については、その対応は、後に述べる全安全ライフサイクルによる安全設計、管理、保守などの工程別の規定に沿った方法論を通して行う必要がある。

ランダムハードウェア故障はデータを集積すれば定量化が可能である。そしてIEC 61508ではこの定量化可能なものに対して、安全度水準(SIL: Safety Integrity Level)を設け、安全度を数値化している。

安全度水準は、低頻度で作動する系に属するものか、高頻度(または連続運転)で作動する系に属するものかで分けられる。たとえば自動車でいえば、エアバックなどは低頻度モード系に属し、ブレーキは高頻度モード系に属する。ブレーキは頻繁にあるいは連続的に作動する系で作動要求のあるものだからである。

安全度水準は二つの作動要求モードに対して設けられ、それぞれを4段階に設定されている。一覧を示す。

低頻度作動要求モード

安全度水準 （SIL）	低頻度作動要求モード （作動要求当たりの設計上の機能失敗平均確率）
4	10^{-5}以上10^{-4}未満
3	10^{-4}以上10^{-3}未満
2	10^{-3}以上10^{-2}未満
1	10^{-2}以上10^{-1}未満

高頻度作動要求モード

安全度水準 （SIL）	高頻度作動要求または連続モード運用 （単位時間当たりの危険側故障確率[1/時間]）
4	10^{-9}以上10^{-8}未満
3	10^{-8}以上10^{-7}未満
2	10^{-7}以上10^{-6}未満
1	10^{-6}以上10^{-5}未満

表 6.3　安全度水準

SILｎのｎの数字が大きいほど安全性が高い。SIL4では形式手法は強く勧められるなど、安全度水準に応じて求められる手法が異なる。どのようなシステムにどのような安全度水準が求められるかなどの詳細な議論は割愛する。組込み系企業にとっては、安全の機能に対してこのような水準が割り当てられる仕様が存在すること、そのための方法が存在すること。このような技術の方向感を感じることがここでは重要と思われる。

(2) 安全対応のためのライフサイクル

IEC 61508 では、次の図[8)]のような工程を経て、安全要求事項に対応する。製品やシステムのみにフォーカスし機能安全の対応を行うのではなく、安全の全ライフサイクルに対応する。その結果、製品やシステムの使用を終えてのちの対応までも範囲に入れることになる。使用を終えた場合や廃棄をした場合の安全対応とは、その段階でシステムが暴走したりなどしないかなどの機能安全を実施することである。

ライフサイクルの概要を述べると以下である。

① まず対象となる制御装置（EUC：equipment under control）と使用環境をよく理解する。ここで EUC とは機械やプラントや制御システムなどである。
② EUC の中で機能安全の対象となる領域を定義する。安全関連系となる。
③ EUC の潜在する危険（ハザード）とそのリスク解析を行う。リスク解析には定性的解析、定量的解析の方法がある。
④ 対象となる機能安全（の機能）に関して、その要求仕様を決める。
⑤ それらの機能安全（の機能）に対して安全度水準を割り当てる。
⑥ 安全関連系は機能安全を達成できるように、運用と保守の計画を作成する。
⑦ 安全関連系の妥当性確認の計画を作成する。
⑧ 安全関連系の設置・引渡しの計画を作成する。
⑨ E/E/PE 安全関連系の実現をはかる。
⑩ IEC 61508 の対象外ではあるが、安全関連系の実現をはかる。
⑪ IEC 61508 の対象外ではあるが、外的リスクを軽減する設備を実現する。
⑫ E/E/PE 安全関連系の設置・引渡しを行う。
⑬ E/E/PE 安全関連系の妥当性の確認を行う。
⑭ 要求されている機能安全が維持できるように E/E/PE 安全関連系の運用、保守、修復を行う。
⑮ 修理・改造中に、あるいは修理・改造後にも、E/E/PE 安全関連系の機能安全が適切であることを保証する。この修理・改造は適切な工程にもどり、行う。
⑯ EUC を使用終了後または廃棄をしている最中の状況、あるいはその後の状況においても E/E/PE 安全関連系の機能安全が適切であることを保証する。

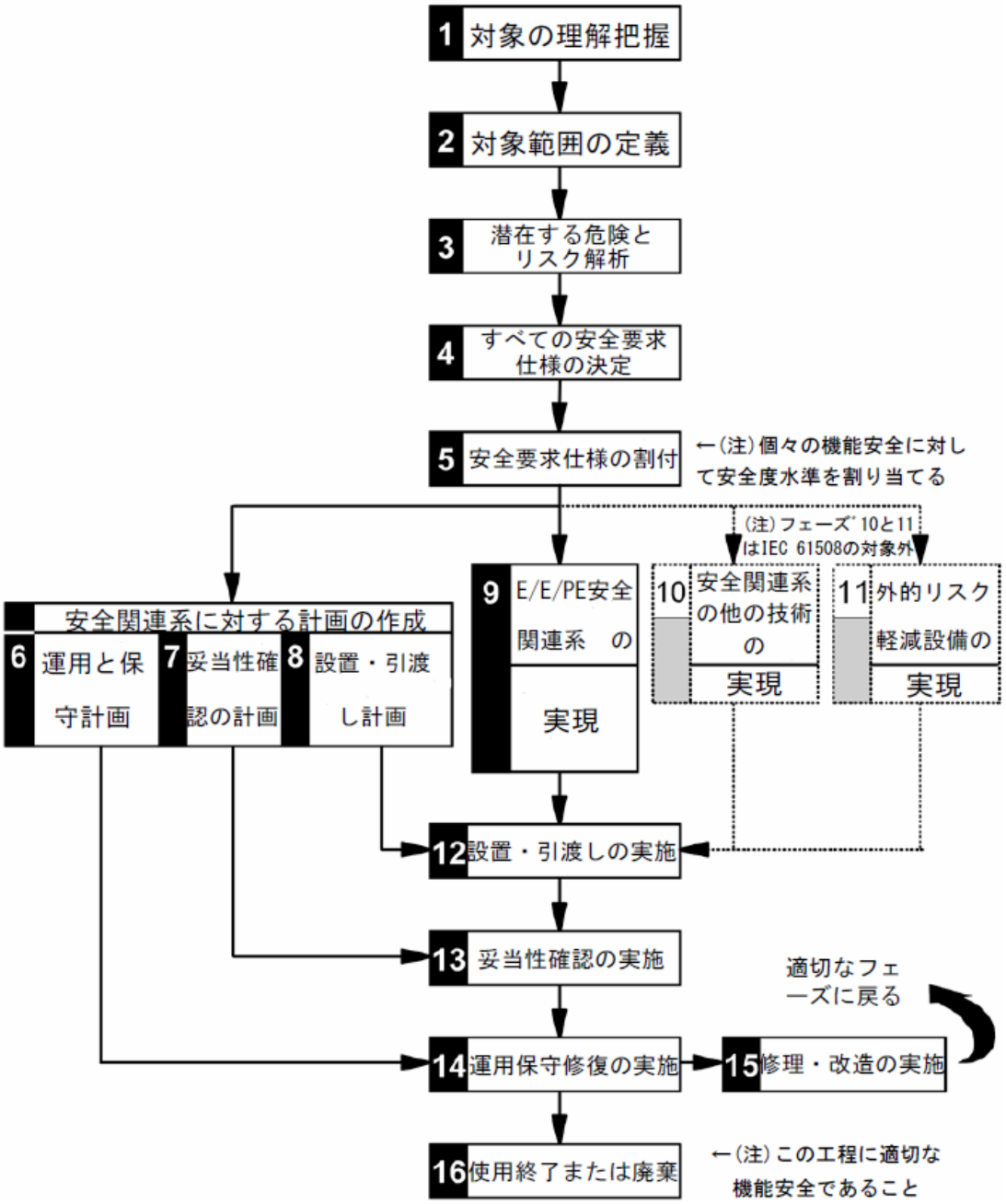


図 6.5　全安全ライフサイクル

《E/E/PES とソフトウェアの安全ライフサイクル》

工程 9 は E/E/PE 安全関連系の実現をはかるものである。図中、“E/E/PE 安全関連系”とは“E/E/PE safety-related systems”の略である。複数のシステムが想定される。これらの複数のシステムに対して、安全要求仕様が実現されなければならない。

工程 9 は、E/E/PES とソフトウェアの安全ライフサイクルに分かれる。図 6.6 がそれである[9)]。

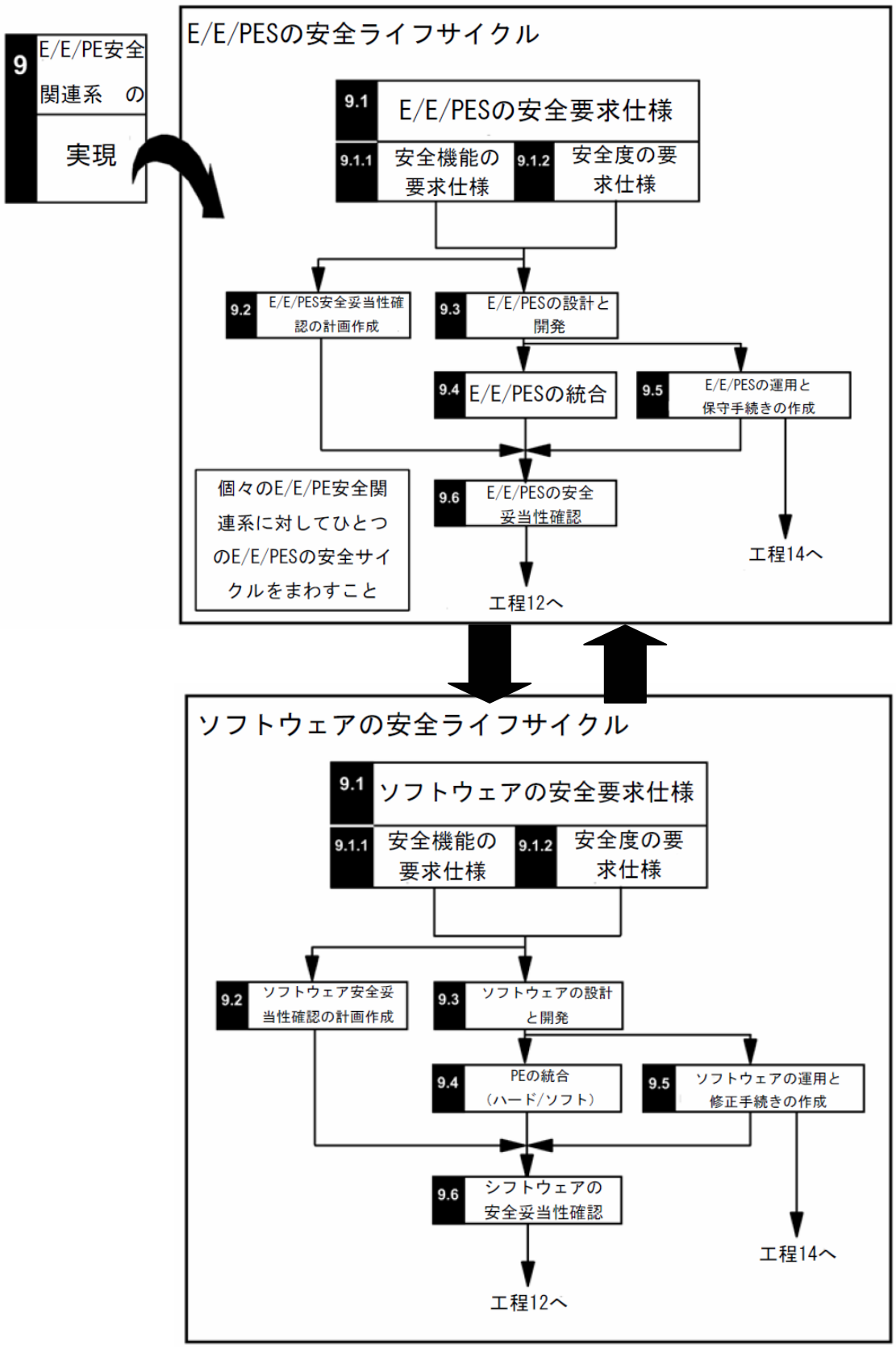


図 6.6　ハードとソフトの相関

ここで“E/E/PES”とは、“Electrical/Electronic/Programmable Electronic System ”の略である。このE/E/PESには、安全という概念は含まれていない。一般に“ハード&ソフトで何らかの機能を制御しているシステム”という意味である。よって、工程9.1の「E/E/PESの安全要求仕様」とは、「任意のハード&ソフトのシステムに安全要求仕様を持たせること」というような意味である。ソフトウェアの安全ライフサイクルにおける9.1に関しても、「ソフトウェアの安全要求仕様」とは、「任意のソフトウェアに安全要求仕様を持たせること」という意味である。そして上の図はシステムとソフトウェア相互の関係を示している。システムの安全ライフサイクルをきちんと実行するためには、ソフトウェアの安全ライフサイクルと同期を取りながら行うべきことを表現している。

《Vモデル》

ソフトウェアの開発においては、IEC 61508では「ソフトウェア安全度と開発ライフサイクル」としてVモデルが推奨されている[10)]。検証と妥当性確認を工程ごとにきちんと行うプロセスが重視される。

IEC 61508では認証の取得のためには文書が重視されといわれる。しかし文書の前に書くべき対象が存在し、きちんと機能していなくてはならない。Vモデルはソフトウェアの開発ライフサイクルとして、IEC 61508に推奨されていることは組込み系企業として承知しておくべきことであろう。

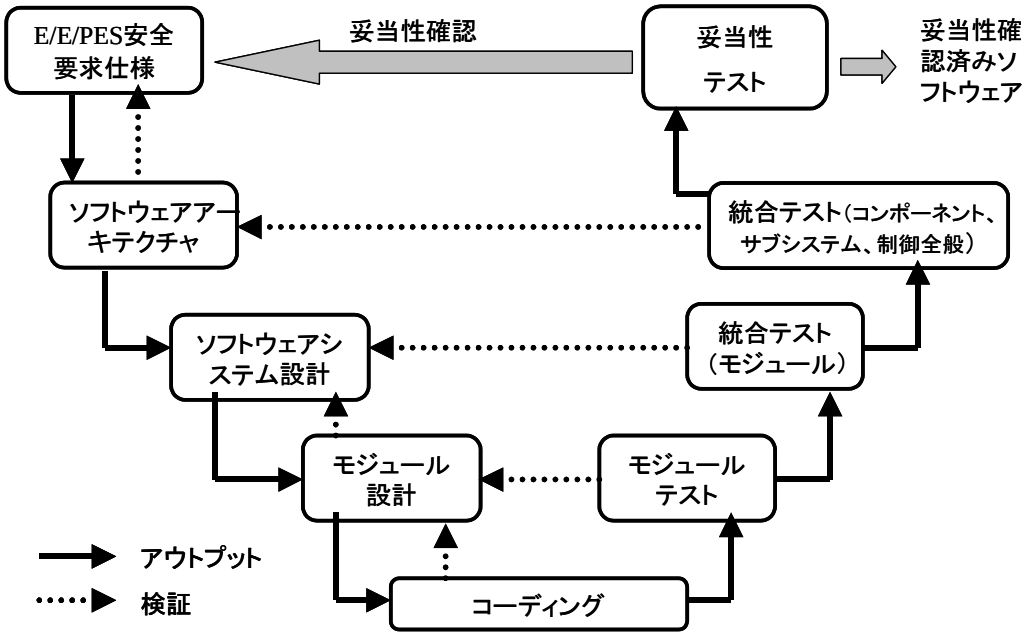


図6.7　Vモデル

(3) 認証制度

IEC 61508 はシステムの安全性を第三者が行う認証制度である。現在英独を中心にいくつかの認証機関がある。日本にはドイツの機関がブランチを作っている。日本法人はない。

①認証機関の業務

英国の認証機関ＣＡＳＳを例に仕事の内容を概観する。ＣＡＳＳの認証業務は表のようにタイプ 1 から 5 まで分かれている。

機能安全能力評価　タイプ5				
部品評価	サブシステム評価	総合システム評価	安全要求評価	運用･保守評価
タイプ1	タイプ2b	タイプ2a	タイプ3	タイプ4
部品供給会社	サブシステム供給会社	エンジニアリング･システム総合会社	運用会社	

表 6.4　認証業務

タイプ 1（部品評価）

部品供給会社に対して評価・認証を行う。

アプリケーションに依存しない一般製品として販売される部品やソフトウェアを対象とする。部品審査（型式承認)、一般用途部品、ＰＬＣ用部品、スマート計器、一般用途ソフトウェアパッケージ、等。

タイプ 2（システム評価）

特定目的のアプリケーション用の電気・電子・プログラマブル電子安全関連系（E/E/PE SRS）の審査を実施する。総合システム評価とサブシステム評価の 2 種類がある。

タイプ 2ａ（総合システム評価）

エンジニアリング・システム総合会社に対して評価・認証を行う。

統合システム審査は発注された装置全体に適用される。

総合システム審査、高度統合システム(監視・制御システム)、緊急時停止系、火災防御系、アンチロックブレーキシステム、分散型制御システム

タイプ 2ｂ（サブシステム評価）

サブシステム供給会社に対して評価・認証を行う。

サブシステム審査は入力インタフェース，プログラマブル電子装置や出力インタフェースのような供給部品からなる統合システムの一部に適用される。

タイプ３（安全要求評価）
　運用会社に対して評価・認証を行う。
　設置後運転される E/E/PE 完全関連系の機能安全維持のために必要な全ての方針、手順、文書、記録を含む。

タイプ４（運転・保守評価）
　運用会社に対して評価・認証を行う。
　特定の使用目的に関して安全関連系への要求事項がそれらの条件を反映しているか、また危害に対するリスク解析に対する評価を行う。

タイプ５（機能安全能力の評価 FSCA：Functional Safety Capability Assessment）
　組織の機能安全能力の評価を行う。
　個々の製品やシステムが対象となる評価ではなく、エンジニアリング組織、エンドユーザー、システム統合者など多くの分野に共通し、当該組織が機能安全を達成できる能力を有するかどうかについて評価する。各タイプの評価の共通的な要素の意味合いを持つ。

②認証製品の例
　オムロン株式会社は、コントローラ（シーケンサ）について IEC 61508 に適合認証取得。
　豊田工機がプログラマブルコントローラーで適合認証を取得。
　Wind River 社のリアルタイムＯＳである VxWorks が認証済み。”Safety Critical”パッケージとして販売されている。認証のカテゴリは不明。

　ところで上記の製品群は部品であり、それ単独では安全度水準が定義できないと理解される。SIL というのは系全体の指標であるからである。
　仮に部品に SIL 水準ｎを取得したとする。しかしシステム全体の SIL は下の SIL に合わせるということになるので、システムはｎ以上の SIL 水準を取れなくなる。
　また部品は SIL 水準でないとすれば、どのような指標で取得するのか。IEC 61508 の「故障率」（安全度水準の表）などが、IEC 61508 に適合しているという意味合いかと推測される。
　このような紛らわしい点に関して、田辺安雄氏（株式会社日本機能安全）は、「最近は個別のサブシステムに対しては、SIL Capability という概念を使いはじめており、全体システムの SIL と区別する動きがでてきている」と述べている。

　ソフトウェアの品質の向上が望まれる中、SIL の取得は自社の製品の品質を客観的に保証するためには効果的であり、企業にとっては魅力である。今後の改善が待たれると同時に、組込み系企業としても、より品質の高いミドルウェアの提供なども企図してみたいところ

である。

6.3 機能安全に関連する議論

機能安全を論じるにあたり関連するいくつかの議論を整理してみる。組込み系企業が機能安全に取り組むにあたり、いくつかの関連する問題を整理しておくことは、IEC 61508 をより客観的に理解する手助けになるからである。

6.3.1 品質と安全、そして信頼性

品質をどう考えどう実現するかは、システムの開発を請け負う組込み系企業にとって不可欠な課題である。業そのものである。一方、その組込み系企業は顧客から、“システムが安全である”ようにと要求されたとき、“当社は品質の高いシステムづくりには自信がありますから大丈夫です”と応えるだけで十分といえるのか、そのような問題意識から、品質と安全について考える。そして、その後に信頼性について言及する。

システムの品質が高いから、そのシステムは安全である、と思いがちである。果たしてそうであろうか。普段は申し分なく稼働していたシステムであっても、思わぬ事故を起こすことがある。よって、品質と安全性とは異なる、とまずはいえる。

しかしではどのように異なるのか。逆に、そのシステムの安全性が高いから、品質が高いといえるのか。安全性は高いが、性能がもうひとつとか、安全性とは関係がないが、何らかの機能がときどき故障する、というのはありそうである。

両者の関係を今この段階で明確に答えることはできないが、少なくてもこの段階では、両者は異なるとだけはいえそうである。以下では、両者の関係を立ち入ってみることにし、組込み系企業として自社のスタンスを顧客に説明できるような概念の分析をしてみたい。

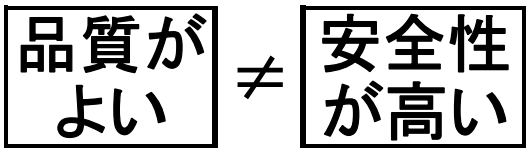


図 6.8　品質と安全性

この２つの概念の違いを明確にするために、実際にあった安全の事故に言及してみる。

六本木ヒルズでの自動回転ドアの事故は、自動回転ドアに男児が挟まれたというものであった。当該の自動回転ドアは、普段は立派なビルにふさわしく、性能もよく人々の入退をきちんとさばいていたであろう。しかし男児の場合には、通常通りの使われ方をされな

かった。男児は、自動回転ドアが入場者に向かってオープンしたタイミングではなく、閉まりかかったときにドアの中に入ろうとした。途端に頭が自動回転ドアと外壁の間に挟まれ、頭蓋骨圧迫による脳内損傷のため死亡してしまった。

われわれはこの事件に関して、自動回転ドアの専門の技術者でも研究者でもないのだが、事故を未然に防ぐために、次のようなことを想像はできる。(ちなみに当時自動回転ドアの安全規格がなかったそうである。)

自動回転ドアはこのような場合に自動停止をさせるべきであるとか、自動回転ドアがオープンして半分まで来たときは、ドアは危険領域に来たので客がドアに入ろうとするのを阻止する機能を付けられなかったのか。しかしドアの停止は、ドアの回転には慣性が働くため、とっさに行うことは物理的法則からして不可能であるかもしれない。とするならば、ドアに何かが挟まった場合、挟まった物体を締め付けない、何かやわらかい素材をドアの部品として使うべきだったかもしれない、などなど。

このようにわれわれは安全に関して、その機能を考えることができる。その機能が妥当かどうかは別にしてである。とりあえず、安全を得るための機能を考えることはできるのである。

しかしこのように安全に関する機能は考えられるが、“品質に関する機能”は考えられるであろうか。“品質に関する機能”とは何を指すのであろうか。“品質に関する機能”ではなく、たとえば“品質に関する方法”であるならば考えやすい。このコトバは、品質をよくする方法はなにか、という答えやすい問いに換言できるからである。

やや初歩的な(言語)論理学的な操作となるが、このような同じコトバの構造の中に他のコトバを代入してみることによる、答えやすさ答え難さは何を指すのか。二つのコトバは、回答として求められる概念領域が異なるところに位置する、といえる。すなわち、品質は“いかに”とか“how”とか方法を求めるコトバであり、これに対し、安全は“なにを”とか“what”とか方法ではなく実体を求めるコトバである。

品質と安全に関して、ここで一つの回答が得られたと思われる。組込み系企業は顧客に対して次のようなことをいうことができよう。

- 当社は方法論を専門とすること。よって品質に関しては最大限を尽くすことが仕事であること
- しかし安全の機能に関して、つまりどうすれば事故が未然に防げるかの機能を考えることは専門外であること
- しかし顧客の提示する機能の実現は方法であるので責任は持つこと
- ただし、安全の機能はどのシステムにも今後必要となるので今後の経験によっては、安全の機能に関する提案もできるようになるかもれないこと

などである。

以上より、次のコトバの用法は適切であるといえる。“品質の方法”と“安全の機能”、

また“品質を実現するための方法”であり、“安全を実現するための機能”という用法である。

ところで読者は奇妙な感覚にとらわれているのではなかろうか。これはコトバの議論であり、工学的な観点から品質や安全の議論をしているのに、やや妙な議論ではなかろうか、と。

ご指摘の通りこれはコトバの議論である。工学的な方法が細かいコトバの差異を凌駕し、新しい実体、すなわち“概念の具体”をつくって行くダイナミズムに比べ、コトバの議論、すなわち概念の定義や整理はスタティックであり、時に退屈である。しかし品質とか、安全、さらには安心、信頼性というような、今日求められているような時代のキーワードは、実にいろいろな局面で、いろいろな意味で使われている。そして混乱もきたしている。したがって、安全、安心などよくある議論に関しても、いまは安全、安心のこれこれに関して語っているなどの、視点を明確にした議論が必要である。

ソフトウェアの工学的実現というべき開発現場においても、仕様書に見られるような概念の曖昧さが存在する。ソフトウェア規模が今日のように大きくなりすぎては、さすがに、つくってみてから、のような従来の方法では間に合わない。品質の向上のためには、コトバをキチンと定義し整理する重要性をあらためて認識すべきではなかろうか。

自動回転ドアの例にもどり、品質と安全の関係をもう少し関係づけてみる。先に「当該の自動回転ドアは、普段は立派なビルにふさわしく、性能もよく人々の入退をきちんとさばいていたであろう」と述べた。この例は自動回転ドアが人の入退させるという「本来の機能」をきちんと果たしていたことを示す。しかし、あるとき突然、事故を起こしてしまう。われわれはこの状態をシステムが“安全性を失った”といい、その原因を問い、安全の対策を議論する。

もちろん「本来の機能」に安全を保つ機能が含まれているのではないか、含まれているべきではないかという議論がある。ここで「本来の機能」とは自動回転ドアが固有に持つ機能であり、他の機器とは区別される機能を指すこととする。従って自動回転ドアがエレベータとかエスカレータとは異なることとなる。

しかし次のような見方ができる。回転をし、人を適切に運ぶはずの機能は、あるタイミングで人を挟んでしまった。このことは、ある 1 つのニュートラルな機能、すなわち自動回転ドアが一定の回転速度で円柱形の空間を回転するという機能は、そこを通る人間を一定の時間間隔で導き入れるという働きと、同じ機能で人を自動回転ドアと円柱の入り口の間に挟み込むという働きを行いうる、といえる。1 つの機能が便宜と危険を同居させている。

自動車は一定の速度以上を出さないように警告を鳴らす。船の航行は何分か前に前方の物体を発見し対応をしないと衝突を起こす。飛行機は飛行を阻害する要因に出会うと墜落する。原子力発電所は放射能が洩れるとしたならば発電という本来の機能すら停止となる。

X線撮影機は過剰な照射は医療機の機能を失う。

およそ安全が大きな問題となるシステムにおいては、一つの機能、それもシステムの本来の機能が便益のみならず、ある条件を満たさないと危険をもたらすという両方の結果をもたらすといえる。本来の機能に密接に関連する機能にも同じようなことがいえる。エレベータは上下の移動が主な機能であるが、同時に各階にキチンと停止しドアの開閉がスムーズになされなければならない。ドアが閉じたままとか階の途中で停止するなどは、直接人命に関わるものではないが、恐怖を与えることになる。

このことから安全の機能とは、本来の機能と裏腹に存在する危険に対して、その危険を引き起こすことになる条件や原因に対応することであるといえる。

最後にあらためて品質の問題にもどる。本来の機能の品質を高めることは、安全に対応したといえるであろうか。燃費がよくスピードがあがり、ハンドル操作も快適、また故障しないことなどは、安全性を高めることになるのであろうか。機能性が高いクルマといえようが、もしブレーキが甘かったりしたならば、安全とはいえないであろう。安全に対応するには、安全の機能が存在しなければならず、安全の機能の品質を高めることは、より安全性を高めることになるといえる。

以上の議論をまとめると次のようになる。

品質と安全は異なる概念であり、前者は方法の概念（方法から獲得できる概念)、後者は機能の概念（機能という実体を実行して獲得できる概念）である。安全性を求められるシステムにおいては、システム本来の機能に多く危険が潜んでおり、危険を防ぐために安全の機能が必要となる。なぜなら、単にシステムの品質を高めただけでは安全をもたらすことにはならないからである。品質は本来の機能に対しても、安全の機能に対しても、向上させることによって、システム全体の機能性と安全性が高まることになる。

《信頼性》

ソフトウェアの信頼性やシステムの信頼性に関しても、組込み系企業は回答することが求められる。

信頼性に関しては以下の定義がある[11)]。

信頼性の定義は「対象要素が与えられた条件のもと、一定の期間中に要求された機能を果たすことができる性質」であり、この機能に「他に損傷を与えぬ機能」を追加すればほぼ安全性にもなる。信頼性の強化により安全性が担保されることもあり、危険回避をシステムまたはその要素の機能維持によって達成する場合には信頼性により安全確保が行われることになる。

ここで対象要素とは、機器やシステムのことである。われわれの議論では品質と安全性の定義をあえてしてこなかった。これらのコトバが使用される実際にあわせて両者の違い、

関係を明らかにしようとしたからである。

この結果、品質に関しては先に「品質は本来の機能に対しても、安全の機能に対しても、向上させることによって、システム全体の機能性と安全性が高まることになる」という知見をえることができた。これは上の信頼性の定義が意味するところとほぼ同義である。機能の品質が向上するということは、システムの機能が本来の機能であれ、安全の機能であれ、上の定義のように「一定の期間中に要求された機能を果たすことができる」ことになるからである。

以上より、品質が高いということは、信頼性が高いといえるが、その逆はどうであろうか。信頼性はあるが、品質はよくない、ということはないであろうか？A社の製品は名ばかり、あるいはブランド負けしている、そのような場合である。

しかしこのような事例は、品質は高いが信頼性がないというケースも想い起こさせる。B社の製品は品質はよいのだが、いかんせん知名度がない、そのような言い方をする。

この議論は、“信頼性”というコトバを日常言語の場で使用しているからである。われわれが“信頼性（がよい)”というコトバに関して、上の定義のように日常言語的な語用を除外して用いるならば、それは“品質（が高いこと)”と同義と考えてよい。

上の定義は、安全性に関しても言及している。「この機能に「他に損傷を与えぬ機能」を追加すればほぼ安全性にもなる」がそれである。その通りであると思われる。

また、安全“性“とか信頼“性“を「性質」と表現しているのも正しい。しかし、われわれの考えにおいて明確な点は、安全性は「機能によって獲得されるもの」、品質、すなわち信頼性は、「方法によって獲得されるもの」という点である。引用の定義はその点を指摘していない。

組込み系企業というモノづくり屋にとって、「方法」という視点からこれらの概念を見据えているということは強調していい点ではなかろうか。今までのコトバを整理すると以下のような表になる。

	品質		安全性	機能性	信頼性
本来の機能	○	→	ー	○	○
	×	→	ー	×	×
安全の機能	○	→	○	ー	○
	×	→	×	ー	×

表 6.5　品質と安全、信頼性

表は本来の機能に品質を高めるようにすれば、安全性が高まるという訳ではないが、信頼性が高まると読む。

ところで、「機能安全は品質である」という、機能安全に関して品質の議論を強調する論調もあるようである。われわれはこの論調に基本的には賛成であるが、しかしこの論調に

賛成できるのは、あくまでも安全は機能が存在してのことであり、安全の機能を受け持つ（責任を持つ：製造者責任）のは、製造責任メーカーであり、このメーカーと組込み系モノづくり屋とが協調して安全と品質に対応するということがあってのことである。

セキュリティなど新たな時代のテーマが出てくるたびに、ソフト屋へのソリューション提供の期待が高まるようであるが、方法を専らとする者に万能の力があるわけでもない。中身（内容、機能）をよく知る専門家と方法の専門家が相互の境界に乗り入れ、時代の課題に取り組んでゆく態度が必要である。

6.3.2　検証と妥当性確認、そして日本の「合理性」

IEC 61508 では、安全性とまた高い品質を獲得するために、先に掲げたようなＶモデルでの開発プロセスを推奨している。このＶモデルでは、妥当性確認と検証が行われることが期待されている（図 6.7 のＶモデル図）。

一般にＶモデルでは、機能の実現のために、企画から設計、開発、コーディング、テストに至るまで、その開発のライフサイクルに応じて、工程単位に決められた作業内容を実行し、作業の結果がキチンとその通りになっているかを検証すること、また仕様内容それ自身が妥当であるかを検討・修正することが基本であると考えられている。

IEC 61508 では､システムの安全分析を行い､潜在的に起こりうる事故や危険を想定して、対象となるシステムで求められる安全度水準に応じて、機能安全を実現する機能仕様を決める。その過程は省くが、要求仕様が決められたあとは、Ｖモデルにそって開発が行われる。各工程は決められた文書や設計書、また与えられたテストデータにかなっているのかを比較検討される。これを「検証（Verification)」と呼んでいる。ここでは与えられた文書等の資料を正しいと前提し、あくまでもその資料に対応する対象が資料に合致しているか否かを判断する。資料の変更はしない。

しかし、そのような検証を経て仕様通りに実現されたはずのシステムが、最後の工程では本来の要求仕様に合致するか、あるいは単に記述された仕様内容に合致するだけではなく、仕様目的に沿っているかが検討される。また逆に仕様自身を変更する必要があるのではなかという検討も行われる。この両者を対象とする適切さの検討が「妥当性確認（Validation)」である。そして両者が合致したときのみ妥当性が確認されたとして次の工程に移ることができる。

このような「検証」と「妥当性確認」の認識とその作業は、機能安全のためには不可欠である。IEC 61508 の認証制度においては、各工程単位に文書を記録し、それが審査の対象になる。

しかし、このように「検証」と「妥当性確認」を行いライフサイクルをまわすことは、IEC 61508 に限らず従来のシステム開発においても明示的ではないかもしれないが、行なわ

れてきた。よく言われるように、わずか 10 ページ程度の仕様からモノができるとする日本のモノづくりの方法は、モノを見て次を決めるという伝統的な経験主義に基づいている。ある程度形ができてこないと、次の問題も具体的に見えてこない。そういういわば“認識の自然”に基づいている。問題をモデル化し、理論化して行くことよりも、いわば実物主義である。いずれ工程の作業を表すコトバとして明示されてはいないが（各企業の固有の用語はあろう）、このような日本的物づくりの工程にも上でいう“Verification”と“Validation”は行われている。

しかし日本のソフトウェアの開発現場に、このような機械等の製造に適合したモノづくりの方法が持ち込まれており、それはソフトウェアの開発に相応しくないのではないかという疑問を投げかけた林晋・黒川利明両氏の議論（論文）が存在する [12]。また実際組込み系開発の現場で開発エンジニアたちが苦闘する姿を描いたフジワラヒロタツ氏のマンガ&エッセーが存在する。

フジワラ氏の作品は次々に起こる開発会社の事件を実際の現場開発者の目を通して描いたもので、そこには林・黒川両氏によって当初は否定されるような「合理性」の乏しい日本のソフトウェア開発の実態が描かれている。仕様の頻繁な変更や仕様が決まらないのに納期だけが決まっているとか、徹夜徹夜連続の日々とか、またコミュニケーション能力に乏しい？ソフトエンジニアの話とか、そのような開発現場の様相が伝わってくる。フジワラ氏はそのような現場をマンガで表現しながら、時折開発方法論についての所見や願望もそのエッセーで述べている。しかし、“好きでなくてはこの仕事はやっていられない”。正にマンガで笑いでもしなければやっていられない、時に切ない現代版“ソフト工哀史”を思わせる作品である [13]。

図 6.9　仕事玉

林・黒川両氏は、ソフトウェアは合理性・論理性にかなったものでなければならないのに、日本人の非論理的・非合理的思考は、はじめからソフトウェアに向いていないのではないか、よっていくら国家予算をつぎ込んでも、ソフトウェアの輸出入は 100 対 1 という輸入超過ではないか、と疑問を投げかける。両氏のソフトウェアの合理性とは目的にかなったものを構築する態度、論理性とは言語等人工的ルールを遵守して作ることへの共感、と理解される。

しかし両氏は、日本人は規則を遵守し論理的に構築して行くことは不得手かもしれないが、目的にかなったものを作ってゆくことはむしろ得手としてきたのではなかろうか、と指摘する。ソフトウェアの本性が論理性ばかりではなく、合理性にもある以上、日本人のモノづくりの方法に日本のソフトウェア産業の可能性があるのではないかと主張する。

この「合理性」に関する議論は次のようなものである。ソフトウェアの最終的なものは「論理的」なものであったとしても、その実際の開発過程はとても「論理的」(ルール的)あるいは「計画的」になされるものではなく、作ってみては仕様を変えるなど、試行錯誤が当然である、というのが最近の考え方の一つになっている。アジャイル（柔軟な）法などと呼ばれ、作りながら次を考える柔軟な開発方法という意味である。このアジャイル法に分類される Lean Software Development という方法論には、「Ohno（大野耐一)、software kanban などのトヨタ生産法に関連する人名や用語が使われており、導入部からしてトヨタの話から始まる」という [14]。しかしアジャイル方法論は一部の日本びいきから始まったのではなく、たまたま結果的にトヨタの生産方式と酷似していたというものである。

いままでソフトウェアの開発は、仕様書をよく固めてから設計、製作に移るというウォータ・フォール型が提唱され、工程を前にもどるなどは各々の工程の作業をなおざりにしていたからと批判されてきた。日程や予算を狂わす原因として非難されてきたものである。

しかし林・黒川両氏によればアジャイル法は、今までの方法論を「現場での生産性の高さと品質の高さという現実により見事にヒックリ返してみせた」という。

実際最近のソフトウェアの開発においては、ソフトウェアの量の増加からいってもすべてを事前に仕様として記述するのは難しい。また UML による設計に関しても、オブジェクトの属性を最初からすべてあげることは難しい。仕様の分析をあれこれ試行錯誤する過程でオブジェクトや属性が変化することはよくあることである。このような思考の自然をサポートするよりよいツールが待たれるところであるが、アジャイル法は「作り込み」と試行錯誤という人間の認識の自然に近いものと考えられる。

V モデルの検証と妥当性確認は日本の得意とする「合理性」の追求と考えられる。そして日本人が弱いとされる「論理性」の強化との両者を合わせた開発の方法論を探求することが、フジワラヒロタツ氏が著名な組込み系雑誌で 6 年の長きに渡って連載し、全国の“マイコン制御系”エンジニアに共感を得た“現場”をよりカイゼンして行く、いわば日本式組込系モノづくりの端緒となろう。

6.3.3　機能安全と情報セキュリティ

機能安全と情報セキュリティを共通の課題として取り上げた本報告書である。両者の関係について把握しておくべきことはないであろうか。たとえば悪意あるシステムへの攻撃は安全性を奪うことになる。この場合 IEC 61508 ではどのように考えたらよいであろうか。

「安全」とは受容できないリスクから解放されていることをいう。また安全度とは、任意のシステムにおいて、あらかじめ定められた時間と条件のもとで、要求された安全機能が満足の行くように実行されているかどうかの確率をいう。この度合いが 4 つに分かれ、各々厳密度が異なるのは前に述べた。システムにより、要求される度合いは異なる。またこの安全度に関しては、ソフトウェアの安全度、ハードウェアの安全度、システムの安全度の違いもある。

このような安全の視点をもちながら、機能安全においては、対象となるシステムに対して、事故や災害の予見を行う。事故の中にはシステムの想定通りの使われ方ばかりではなく、それ以外の使われ方をされた場合のことなどを考慮し、リスクの見積をしなければならない。そしてリスクの評価を行い許容範囲のリスクか否かを決める。

先の自動回転ドアの例などは、想定外だったのか否か。想定していたが許容範囲としていたのか否か。もし想定外のことであったとしたならば、機能安全としての対策はできない。思いつかないからである。

開発側（申請者）と認証側の両者によって予見され適切と認められた完全機能のみが、IEC 61508 の適合性審査の対象となる。

さて、このように考えた上で、機能安全と情報セキュリティを考えるために、自動回転ドアやエレベータがネットに接続されたと仮定しよう。ネットに接続することによる危険はすでに述べた通りである。そしてリスク分析の結果、ネットから機器に送られる、或るデータに関して、その機器は常に誤動作をすることが判明したとする。このデータを第三者が偽造し、その機器に送り込むことができるとする。

あるいは、テロリストが自動回転ドアやエレベータにデータを送り、制御プログラムを書き換え、一層危険な動作をさせることができたとする。いささか虚構めくが、テロリストは標的の人物がこれらの機器に乗降しようとしたときに、攻撃をしかけようというものである。

このような脅威に関して、予見が成り立つとしたならば、開発者側は機能安全の分析において、情報セキュリティによるリスクも検討しなければならないということになる。

はたしてどうか。

IEC 61508 のパート 1 の 1.2 の j）には以下のようにある。

「この規格は、権限を持たない者が、E/E/PE(電気・電気・プログラム可能な電子的)安

全関係を持つシステムの機能安全に対して、危害を与えるとか、また影響を与えるとかを防ぐ必要があるかも知れないというようなことに関しては、対象としていない」とある。（This standard does not cover the precautions that may be necessary to prevent unauthorized persons damaging, and/or otherwise adversely affecting, the functional safety of E/E/PE safety-related systems.）

この指摘は、IEC 61508 に詳しい、水口大知氏（産業技術総合研究所 システム検証研究センター）からいただいたものである。

氏によれば、悪意ある攻撃によって機能安全が損なわれることへの対策は「規格の範囲外」ということである。範囲外とは、このような事態への対策をしていても IEC 61508 に適合しているとはいわないし、またこの対策に関して IEC 61508 の認証を取得しているということもない、という意味である。

しかし、氏は、適合とか認証とは別に悪意ある攻撃に対しても、IEC 61508 の考え方に沿ってある程度の対策を講じることはできるだろう、と述べる。このことは、すなわち、認証の取得云々とは別に、考え方として機能安全のすすめるところを、直接認証を取得することとは無関係な、組込み系の開発会社も取り入れることは有効であると考えることと等しい。情報セキュリティに関して、ISO/IEC15408 の基本的な事柄をよく学ぶことが、システム開発に利があることと類似である。

さて、そう考えたところで、機能安全と情報セキュリティの関係するところをさらに検討してみたい。

上の例と逆なこと、すなわち機能安全のリスク管理を怠ることによって、そのシステム、あるいは関連するシステムに情報セキュリティの問題の起因となることはないであろうか。機能安全の認証を受けたシステムが、情報セキュリティ問題、すなわち情報資産を盗まれたり、情報資産を使用する権利をなりすまされたり、自由に使用できなくなったり、暗号などをかけられてしまったりである。

あらためていうと、情報セキュリティは悪意の者が存在し、その者が意図的にネットなどを介しシステムの情報を脅威にさらすことを防御するというものである。他方、機能安全は、ネット接続の問題は主要な課題ではなく、よって悪意の介在とは無縁な「安全」が議論されている。

しかし、そのシステムがネットと接続されており、機能安全の制御が改竄されたらどうであろうか。ネットを介して組込み系のシステムの修正を行うことも、珍しくはなくなっている。

この議論は、組込みシステムがネットに接続された途端に安全ではなくなるという、すでに行った情報セキュリティの場で行うべきであろうか。しかし仮にそうだとしても、情報セキュリティの議論に、機能安全の議論も持ち込んで行わなければ、有効な問題の解決はむずかしい。

両者を同じ土俵でよくひも解いてみる必要がありそうである。

（注）

この稿の後、IEC 61508 制度の専門家である田辺安雄氏（株式会社日本機能安全）より、次のような指摘をいただいた。IEC61508 第 1 部ではセキュリティの改訂が予定されており、これは「ニューヨークのツインタワーのテロのあと、機能安全でセキュリティが話題になり、このような悪意ある攻撃が、機能安全上問題があると考えられる場合は、セキュリティを考慮するという方向へ変わった」ことによる。そのため「現在審議されている改訂版では、セキュリティに言及することになった」とのことである。氏によれば、「アメリカのオハイオの原子力発電所では、安全関連系に外部からワームが侵入し 5 時間ほど、作動不可状態が続いた事件もあった。時代の要請かと思う」とのことである。

6.3.4　JTAGテストによる組込みシステムの信頼性・安全性の向上

コンピュータ制御への依存が高まる中で、組込みシステムへの期待が大きくなっているのはすでに見たとおりである。しかし、組込みハードウエアに関する面では、ますます複雑化・高度化する製品のテストが従来の方式では困難になっているという大きな問題が生じている。これに対する有効なソリューションとして組込みシステムを陰で支えているのが IEEE std 1149.1 に基づく JTAG 方式ボード・テストである。

プログラム可能な電子系システムが常に安全に機能する機能安全に関する国際規格にＩＥＣ61508 があるが、動作するハードウエア上に問題がないことが前提となる。

本稿では、既に組込みシステム・機器のハードウエア・テストにおいて、広く適用が進んでいるが、産業界全体としては未だにその認識が十分とはいえない JTAG テストの解説と、このテストが組込みシステムの信頼性・安全性に寄与している現状およびこれからの可能性について述べる。

(1) 従来のボード・テスト・システムの限界と JTAG テストの誕生

TTL に代表される DIP(Dual In-line Package)全盛時代には、ボードに挿入実装されている端子やラウンドに金属製のプローブを接触させ、テスト信号を与える従来の ICT(In Circuit Test)方式でテストが可能だった。しかし、80 年代に入ってピン間の狭い QFP(Quad Flat Package)や SOP(Small Outline Package)などの周辺端子型の表面実装パッケージが実用化されたころから、プローブをピンに接触することが難しくなった。そして、BGA（ball grid array）や CSP(chip size package)といった格子端子型が主流となりつつある現在では、プローブを接触させることそのものが不可能になっている（図 6.10)。

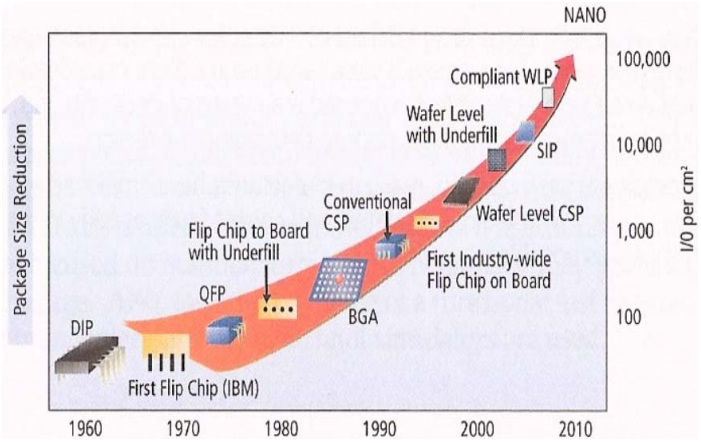


図 6.10 LSI パッケージ技術の変遷

従来のボード・テスト・システムでは、最新の複雑化・高度化する組込みシステム製品の実装テストに対応できなくなってきている状況を以下に記す。

①BGA/CSP パッケージ搭載ボードのテストは、従来の ICT ではプロービングが困難である。プロービングが可能な場合でも 2000 ピンを超えると接触信頼性の点で実用にならない。

② ファンクション・テストで検査精度を上げるためには、プログラミング作成工数が増大し、多品種少量生産のボードでは採算が合わない。　また不動作は発見できても故障箇所の解析は困難である。さらに、ボード上のＣＰＵが活きていない場合にはファンクション・テスト自体が機能しない。

③ 正常ボードと被テスト・ボードの比較検査は有効な方法であるが、ファンクション・テストと同様不動作は発見できても故障箇所の特定は困難である。

④ 高価なＸ線テスト・システムを導入しても、部分的な解析用であり、全数テストには不向きである。また電流を流したテストではないため、接続の信頼性の検査としては充分でない。

この状況をいち早く察知し、新しいボード・テスト方式の実現を検討するグループが 1985 年にヨーロッパで発足した。その名は JETAG(Joint European Test Action Group)で、その当時のヨーロッパの先端企業である Philips、British Telecom、Ericsson、Siemens および Thomson-CSF などが参画していた。

1986 年には、北米企業が加わり、発展的に JTAG(Joint Test Action Group)に改称した。1987 年には DOD（米国防総省）の VHSIC プロジェクトの賛同を得た。1988 年に JTAG Version 2.0 として IEEE(Institute of Electrical and Electronic Engineers)の Testability Bus Standard Committee に提案され、1990 年に IEEE std 1149.1「Test Access Port and Boundary-Scan Architecture」として承認されている。

(2) JTAG テスト・システムの仕組みと動作

JTAG テストは、図 6.11 に示すようにデバイスの内部の本来のコア･ロジックと各ピンの間にテスト・プローブと等価な働きをするセルと呼ばれるレジスタを配置し、これを結合してシフト・レジスタを構成、このシフト・レジスタを制御することによりテスト・コードの入力とこれに対するレスポンスによりテストを実行する。デバイス内部の境界をテスト・コードがスキャニングすることから、バウンダリ・スキャン･アーキテクチャと呼ばれる。

JTAG 対応デバイスの内部構造を図 6.12 に示す。JTAG に対応したデバイスには、本来のコア･ロジックのほかに、テスト機能を実現するための専用の簡易なマイクロプロセッサが組み込まれていると考えるとわかりやすい。　なお、本稿ではメカニズムを表す場合には「バウンダリ・スキャン」と呼ぶことにし、一般的な用語としては「JTAG」を用いることにする。

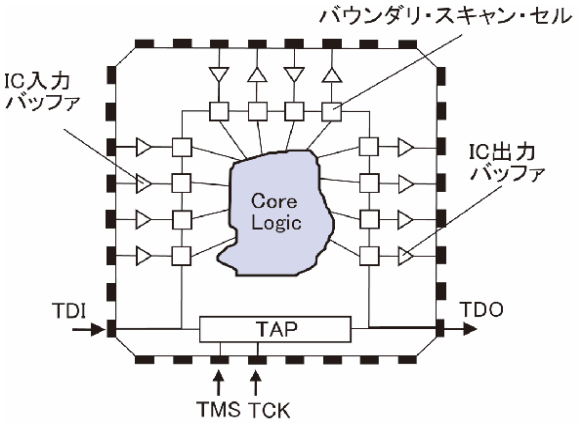


図 6.11 バウンダリ・スキャン・アーキテクチャ

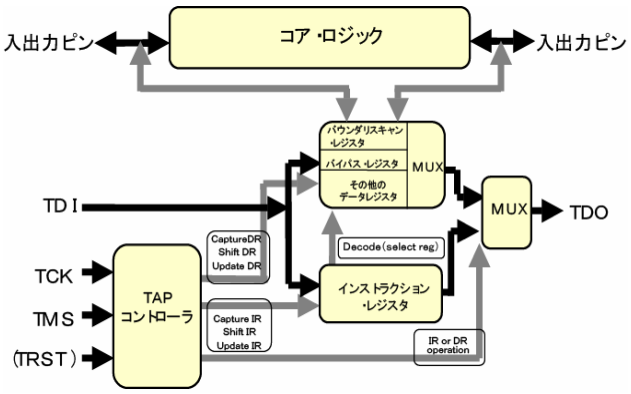


図 6.12　JTAG 対応 LSI の内部構造

① JTAG 命令セット

表 6.6 に JTAG デバイスの命令セットを示す。

パブリック命令は IEEE1149.1 に規定されているもので、このうち必須命令はバウンダリ・スキャン対応デバイスが基本的に備えるべき命令セットで、ボード・テストを行う際に必ず用いる。一方、オプション命令は LSI 設計者が必要に応じて選択する命令セットである。プライベート命令は、IEEE1149.1 規格では規定されていなく、LSI 設計者が独自に設定する命令セットである。

	命令の種類		命令	機能
インストラクション・レジスタに与えられる命令	パブリック命令	必須命令	EXTEST	デバイスの物理的接続をテストする。
			SAMPLE /PRELOAD	指定した時点でのデータの取り込み /指定データパターンの設定
			BYPASS	スキャン・パスをバイパスする。
		オプション命令	INTEST	デバイスの内部ロジックをテストする。
			RUNBIST	デバイスの内蔵するBIST (Built-In Self Test)を実行する。
			IDCODE	デバイス識別コードを確認する。
			USERCODE	ユーザプログラマブルな識別コードを確認する。
			CLAMP	バウンダリスキャン・セルをドライブしたままスキャン・パスをバイパスする
			HIGHZ	全ての出力セルをハイ・インピーダンスにする。
	プライベート命令		ユーザ定義	ユーザ定義

表 6.6　バウンダリ・スキャン命令セット

②JTAG 方式ボード・テストのシステム構成・

バウンダリ・スキャンによるボード・テストのシステム構成を図 6.13 に示す。

ボード上でバウンダリ・スキャン対応 LSI の TDO ピンと別の LSI の TDI ピンを接続し、すべての LSI を数珠つなぎにする。こうしてでき上がったテスト・データのパス(経路)をバウンダリ・スキャン・チェーンと呼ぶ。

バウンダリ・スキャン対応 LSI を搭載したテスト・ターゲット(UUT: Unit Under Test)に TAP コネクタまたはラウンドを設け、バウンダリ・スキャン・コントローラ(JTAG コントローラ)と呼ばれる装置からシリアルのテスト・データを与える。バウンダリ・スキャン・コントローラとホスト PC とのインタフェースはとしては、プリンタポートのほか、PCI、USB、IEEE1394、イーサネット、VXI,PXI が用意されている。

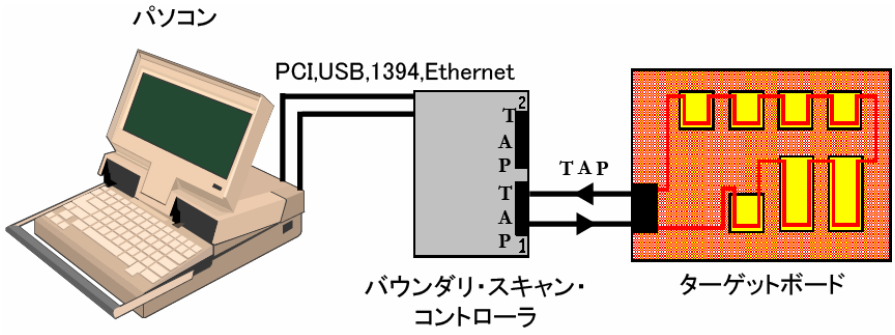


図 6.13　バウンダリ・スキャンによるボード・テスト

③バウンダリ・スキャン・テストの工程

バウンダリ・スキャン・テストの工程の手順は、以下のとおりである（図 6.14)。

・テスト・パターンの生成

・テスト実行

・テスト結果の解析

最初に、パソコン上で稼動するバウンダリ・スキャン用テスト･パターン生成ツールを使って、テスト・パターンを生成する。同ツールには、回路図エディタが出力するネット・リスト・ファイルと、半導体メーカが無償で提供しているバウンダリ・スキャン対応 LSI の BSDL(Boundary-Scan Description Language)ファイルを入力する。このため、従来のインサーキット・テスタの場合に必要となるテスト・プログラミング生成作業は原則として不要である。

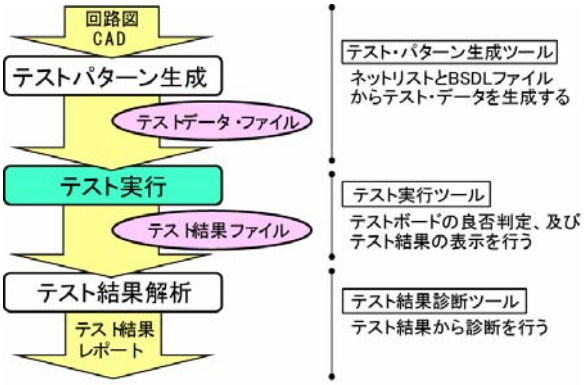


図 6.14 開放故障診断

テスト結果は、バウンダリ・スキャン用の故障診断ツールを使って解析する。開放(オープン)故障、短絡（ブリッジ）故障、0/1 縮退（stuck-at）故障などの異常を、ネット上に接続された各 LSI の端子番号とともにレポートしてくれる。

図 6.15 にインターコネクト・テスト(配線テスト)における接続故障の例を示す。
また、図 6.16 にテスト実行ツールと故障診断ツールの画面表示例を示す。テスト実行ツールは被テスト・ボードの良否とテスト結果をベクタ表示でレポートする。故障診断ツールが詳細診断結果を表示する。ボード・レイアウト及び回路図上での故障表示ソフトウエアも用意されている。

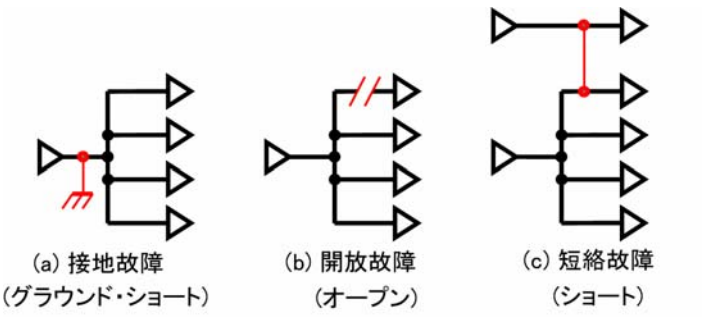


図 6.15　インターコネクト故障の例

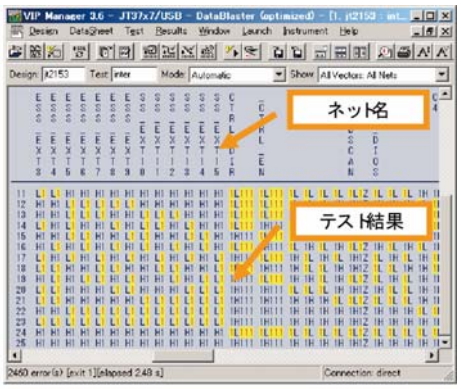


図 6.16（1）テスト結果の真理値表示

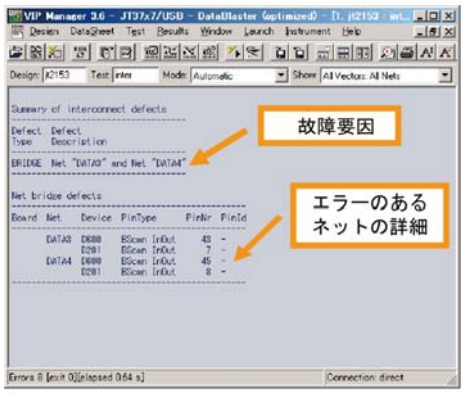


図 6.16（2） 詳細故障診断表示

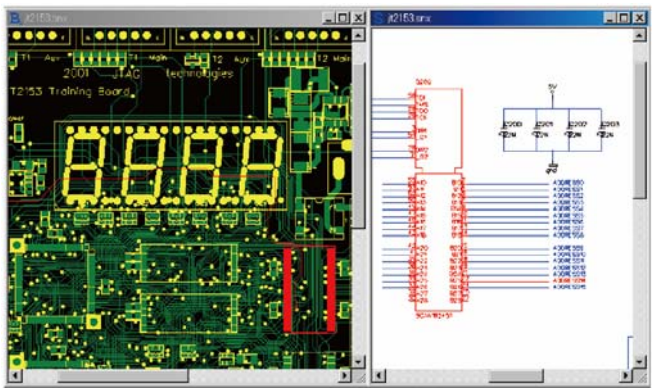


図 6.16（3） 故障ボード・レイアウト表示 / 故障回路図表示

④バウンダリ・スキャン・テストの内容について

図 6.17 を用いてバウンダリ・スキャン・テストによってテストする内容を説明する。

・インフラストラクチャ･テスト

実装ボードのバウンダリ・スキャン・チェーンが正しく機能しているかどうかをチェックするテスト。

・インターコネクト･テスト

インターコネクト･テストはバウンダ・スキャン･テストの基本である。ボード上のバウンダリ・スキャン対応 LSI どうしを結ぶ配線に開放故障や短絡故障などがないかをチェックする。

・ロジック・クラスタ・テスト

バウンダリ・スキャン･テストに対応していない IC/LSI の集合を「クラスタ」と呼び、このクラスタが正しく機能しているかどうかをテストする。

・メモリ・クラスタ・テスト

メモリは通常、バウンダリ･スキャン機能を持っていない。この意味で SRAM, DRAM, SDRAM, FIFO メモリなどの集合を「メモリ・クラスタ」と呼ぶ。バウンダリ・スキャン・チェーンからメモリのアドレス信号データ、コントロール信号を制御して、メモリに “1” や “0”

を書き込んだり、読み出したりする。正常に読み書きできるかどうかによって、メモリとバウンダリ・スキャン対応 LSI を結ぶ配線をテストする。

・コネクタ・テスト

ボード上の LSI と外部信号の入出力を行うコネクタを結ぶ配線は、そのままではテストできない。ボードの外部にバウンダリ・スキャン対応 LSI を内蔵したオプション・ユニットを接続することにより、インターコネクト・テストやロジック・クラスタ・テストも行えるようになる。

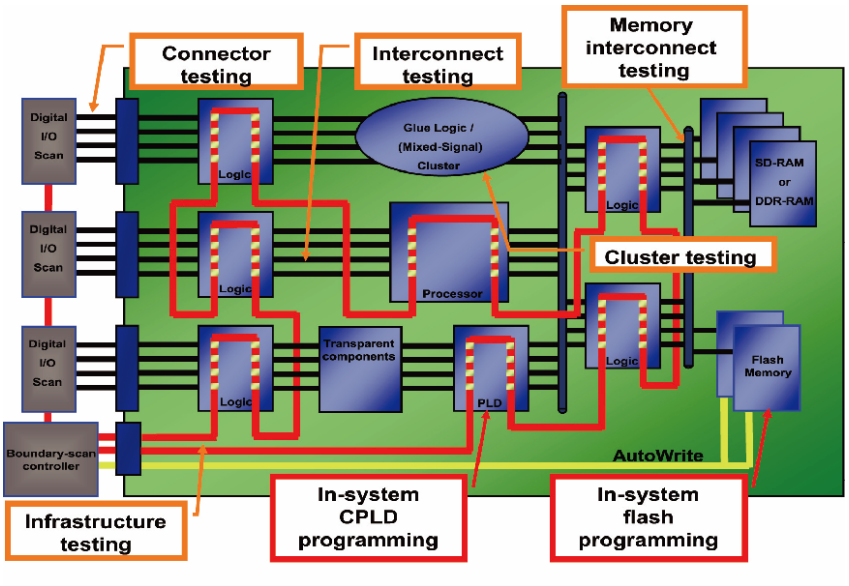


図 6.17　バウンダリ・スキャン・テストによってテストする内容

(3) 組込み製品の各段階でのテスト

① 研究開発段階でのテスト

組込みシステムの研究開発段階では、十分な実装テスト・システムが用意されてないのが通常である。このため研究開発段階では、プリント基板実装不良の原因の特定に相当の時間が必要とされていた。

JTAG テストを導入後は、研究・開発者の手元の PC をホストとするシステムで、試作品であっても不具合の内容とその箇所を容易・迅速に特定できる。このため、研究開発者はプリント基板自体の不良や、はんだ付け等に起因する不動作に煩わされる時間を削減し、本来の設計自体のデバックに専念できる。

また、設計自体に問題があることが判明して回路を実機上で修正した場合においても、これを回路図に反映して、手元の PC 上で改定したテスト・コードを再生成し、これに基づいて JTAG テストを実施することで、修正後の試作機が回路図どおりに実装できているかどうかを確認できる。これは特に製品の回路規模が大きい場合、研究開発者の負荷の低減に大きく寄与する。

②生産試作段階でのテスト

組込みシステムの生産試作段階では、従来のＩＣＴやファンクション・テスト・システムでは検査冶工具の製作が間に合わないのが通常である。これは、プリント基板自体のレイアウトが生産試作の後に最終確定するため、テスト・ラウンドを含めた冶具製作にかかれないためである。

JTAG テストを導入後は、原理的に 5 本のテスト制御線のみを基板に接続すればよく、ネットが同じであれば、レイアウトの異なる基板でもテストが可能である。これにより、これまで難しかった生産試作段階における複数の基板のテストおよびデバッグが可能になり、生産試作の工数の削減に寄与する。

また、生産試作段階のボードが不動作の場合、往々にして開発技術者がこの不動作の原因追求と修理を担当せざるを得ない状況が生じる。JTAG テストの導入により、この状況においても、ボードの実装不良とその故障箇所の特定が容易に可能であり、開発技術者の工数が節減できる。

③ 製品の量産段階でのテスト

組込みシステムの量産段階への JTAG テスト・システムの導入により、技術面では、以下のメリットが得られる。

・高速テストが可能。
・故障診断が詳細にできる。
・不良箇所の特定により修理時間を短縮できる。

さらに、経営・経済面では以下のメリットが得られる。

・テスト・システムのコスト・パフォーマンスが高い。
・汎用に使える。
・フィクスチャなどの費用が大幅に低減できる。
・高価な製品ボードの歩留まりが向上する。
・製品の信頼性が向上する。
・ＥＭＳに製造・検査～修理をアウトソーシングできる。
・発・生産試作・量産の各段階を通じて、製品開発の Time to Market が短縮され、競争有利となる。

(4) JTAG テストによる信頼性・安全性の向上

最近の組込みシステム・機器は、自動車、列車、航空機等の搭載ユニットに代表されるような振動する本体上に装着されるものも多くなっている。特に自動車の高級車では、70 個以上のマイコン組込みユニットが搭載されている。これらの車載ユニット等の実装テストでは、通常の製品・システムにおけるテストでは不充分で、安全性確保のため、被測定ボードに振動を与えての実装テストが必要になる。

従来のテスト手法においては、被テスト対象ボードとのテスト信号のプロービングは、基

板上にテスト・ラウンドまたは部品の端子そのものにテスト・プローブをあてる。このため、振動の激しいテスト環境においては、テスト・プローブとテスト・ラウンドとの接触信頼性が大幅に低下し、被測定ボードの実装不良とコンタクトの接触不良が判別不可能で、この種のテストに使用することは適さない。

これに対し、JTAG ボード・テストでは、基本的には 5 本の JTAG 制御線を被測定ボードにコネクタ等で固定し、振動環境の下で被測定ボードに通電しての実装テストが可能である。JTAG テスト・システムは、繰り返しテスト機能があり、たとえば 1000 回のテストが連続して行え、エラーログが取れる。このエラーログにより、実装ボード上の故障の有無がわかり、故障箇所が特定できる。

さらに、チェンバーの中に、被測定ボードを入れ、JTAG テスト・システムをチェンバーの外に置くことができるので、被測定ボードに温度、湿度を加えた環境テストにおいても適用可能である。

JTAG テストを信頼性向上の目的をもって適用している事例としては、以下があげられる。

① 温度加速環境化における非鉛はんだ使用の接続信頼性劣化のテスト
② 宇宙空間に打ち上げる人工衛星に搭載する電子装置のはんだ付け信頼性のテスト
③ 航空機搭載の電子機器のテスト
④ 防衛車両に搭載する電子機器のテスト
⑤ ポータブル電子機器のテスト

これらのテストを組込み製品の全数検査に適用すれば、基板上のはんだ付けの不良等を工場出荷前に発見できる可能性がある。これにより、はんだ接続の故障カーブ（バスタブ・カーブ）における初期不良が出荷後に発生することを防止でき、信頼性・安全性の向上に寄与できる。

(5) システム・レベル実装テストによる信頼性・安全性の向上

組込みシステム機器、とりわけ比較的に規模の大きい計測機器、制御機器、放送機器などでは、標準コンピュータ・バスを用いて、複数の実装ボードを結合してシステムとしての機能を構成する。また、ビデオ機器などの情報家電機器、複合機などの事務機器では、親基板と子基板（メザニン）からなるシステムを構成する場合がある。

複数（n 枚）の実装ボードでシステムを構成する機器では、システム全体の故障率は概略 n 倍になると考えてよいので、複数のボードで構成されるシステムでは、単一ボードで構成される機器と比べて、そのテストはより重要になる。

ここで、システムや機器に故障が生じてから、それが修復されるまでの平均時間である

MTTR(mean time to repair) を短くする行為は、信頼性・安全性の向上に通じる。 修復するまでの時間は、①不動作の発見と　②その原因の特定　および　③その修復にかかる時間の総和になるので、複数ボードで構成される機器では、システムの不動作を短時間で発見して、その原因をすばやく特定でき、ボード交換等でのシステムの修復に有効な手段が求められる。

JTAG によるシステム・レベル・テストは、これに対する解法を提供する。

事例として、現在広く使われている標準コンピュータ・バスである PCI バスには、JTAG の五本の制御信号端子が割り当てられている。 JTAG テストでは、マザー・ボードに割り当てられ JTAG 制御信号端子から、各実装基板の TAP にテスト制御信号を供給することにより、複数ボードの実装テストが可能になる。

ここで、JTAG によるシステム・レベル・テストで考慮すべき点は、JTAG の 5 本の制御信号を複数の実装基板に接続する方式にある。TCK、TMS、RESET 信号は各ボードにパラレルに接続するので問題は無いが、TDI および TDO 信号はシリアルに接続することになることから、マザー・ボードから入力された JTAG シリアル信号は、最初のボードの TDO から 2 番目のボードの TDI へ、さらに 2 番目のボードの TDO から 3 番目のボードの TDI へとスキャン・チェーンを構成して、最後のボードの TDO からマザー・ボードを経由して、JTAG テスト・コントローラに返される。この過程において、あるボード内のスキャン・チェーンに故障があるとシステム全体のスキャン・チェーンが成立しなくなり、テストが不可能になる（図 6.18)。

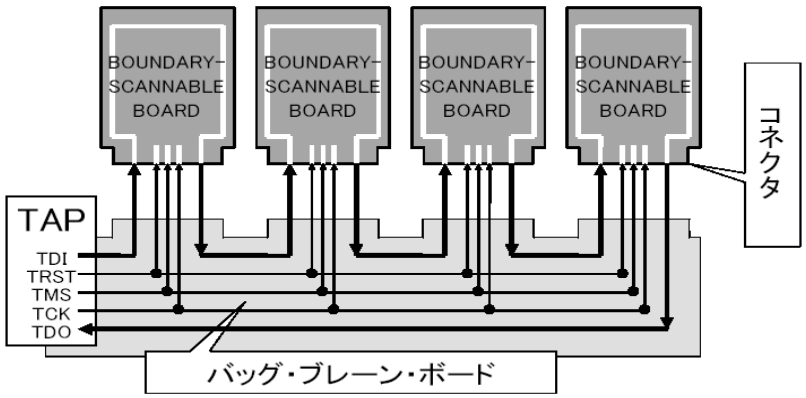


図 6.18　JTAG によるシステム・レベル・テスト

これに対して、図 **6.19** に示す構成では、NS 社の ScanBridge という専用デバイスを、各基板に装着する。　このデバイスは基板上の DIP・SW などで設定したアドレスが JTAG テスト・コントローラからシリアル信号として与えられるアドレス信号と一到した時に、搭載されたボードに対するテスト信号であると判断して、そのボードに対する JTAG 実装テストを可能にする。この方式によれば、マザー・ボードに搭載された複数のボードを個別にテスト可能である。また、特定のボードの不良によりシステム全体のスキャン・チェーンが途絶えることなく、かつ、マザー・ボードボード上の位置を選ばずにボードを装填できる。

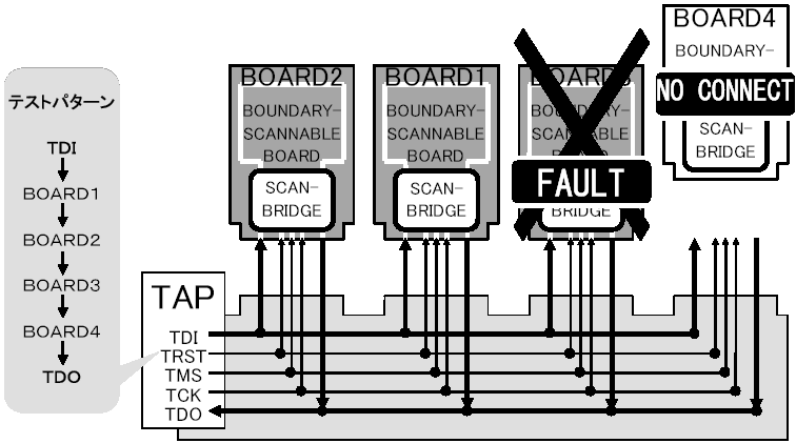


図 6.19　Scan Brodge を用いた JTAG システム・レベル・テスト

さらに、高信頼度システムを冗長システムとして構成して、あるボードが故障した場合に、これを切り離して、システムとして継続した稼動をさせることでトータルシステムの信頼性・安全性の向上に寄与する。

(6) オンボード LSI 機能テストによる信頼性・安全性の向上

組込みシステムをソフトウエェアだけで有効なテストを行おうとすれば、処理時間が長く、そのためのプログラム開発コストも高くなるばかりでなく、充分な故障診断ができないことが多い。こうした状況に対して、テストを容易にする設計技術のひとつで、テストの際に内蔵したハードウェアを利用する BIST(built-in self-test)に基づいた手法が普及し始めている。

BIST は、従来からチップ・レベルのテスト手法として、LSI ベンダーが LSI テスタを用いて活用している。これに対し、オンボード LSI 機能テストでは、JTAG テスト・システムより被テスト対象の製品ボード上に搭載された LSI に TAP よりアクセスして、オプション命令の一つである RUNBIST 命令を与えて起動し、LSI が自己診断を実行した結果をシステムへ返す。このオンボード LSI 機能テストの方法は、LSI エンベデッド・テストとも呼ばれている（図 6.20)。

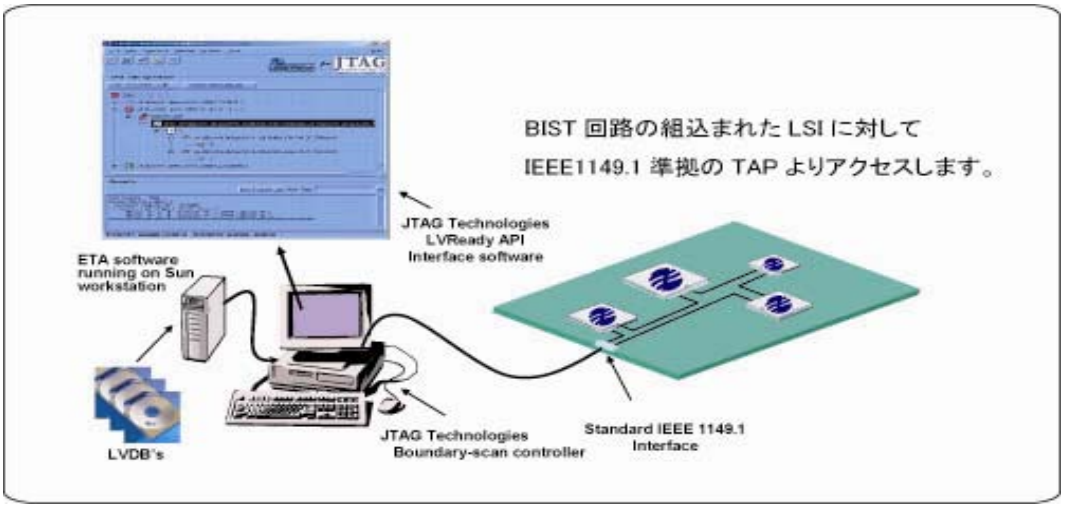


図 6.20　LSI エンベデッド・テスト

これにより、JTAG テスト・システムは、ボード・テスト診断ソフトウェアと LSI エンベデッド･テスト診断ソフトウエぇアを併用することで､本来のボード実装配線テスト機能に加えて、ボード上に実装された LSI に対しも LSI テスタと同様の機能テストが実施できる。

JTAG 実装テストは、原理的にスタティックなテストですあるが、LSI エンベデッド・テストでは、ロジックおよびメモリに対する実動作テスト（At-Speed）テストが可能になる。
図 6.21 に BIST 組込み LSI のイメージを示す。図中、LB はロジック BIST、MB はメモリ BIST である。図 6.22 に工程用 JTAG テスト・システムのロジック BIST 実行結果の表示例を示す。

最近の機能が複雑な組込みシステムでは、従来に比べて、集積度の高い LSI が複数個使用されており、実装された LSI 自体の不良率が、ボード配線テストの不良率を上回る場合が生じている。このため、従来のボード配線テストをパスした製品ボードがファンクション・テスト工程で不良となる場合が往々にしてある。ファンクション・テスト工程でボードが不良となった場合には、製品への良品ボードの再実装などの工数が増大することになり、生産性を低下させる。さらに、製品システムとしてのファンクション・テスト工程を経た後においても、LSI の欠陥を内在したまま市場に出る可能性もある。

LSI のサブミクロン化が進展する中で、プロセス上の欠陥が増大する傾向にあり、製品テストの重要性が増している。メーカサイドで LSI テストを全数実施したとしても、運搬時や製造工程での静電気による破壊、はんだ付け時の熱ストレス等がボードに実装された LSI の障害の要因になる。JTAG 方式のオンボードＬＳＩ機能テストを製品テスト工程において実施することで、市場に信頼性の高い製品を供給することができ、組込みシステムの信頼性・安全性の向上に寄与する。

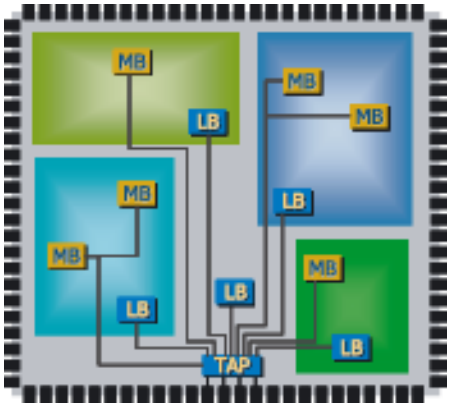


図 6.21　BIST 組込み LSI

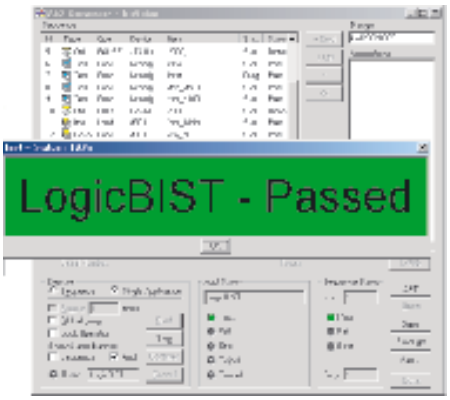


図 6.22　ロジック BIST 実行結果の表示例

(7) JTAG 遠隔診断によるオンボード LSI 機能テストが組込みシステムの信頼性・安全性の向上にもたらすインパクト

これまで記述したバンダリ・スキャンによるボード・テストでは、テスト・コントローラとターゲット・ボード（UUT: unit under test）を隣接して配置する必要があった。

これに対し、テスト・コントローラが、遠隔地にあるターゲット・ボードをテスト可能な仕組みを、図 6.23 に示す。

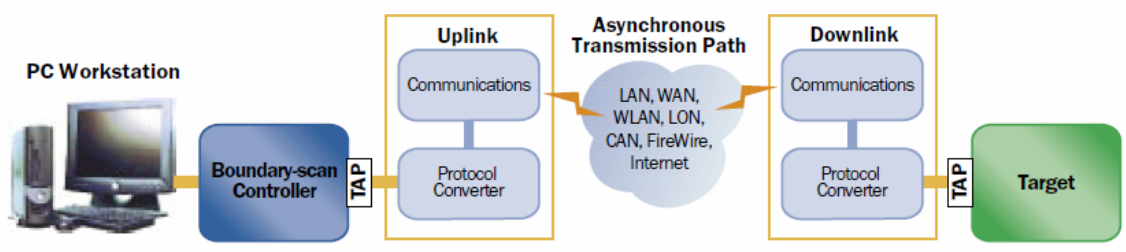


図 6.23　遠隔診断による JTAG テスト

Uplink は、テストサーバを構成する BS コントローラ側の TAP に接続する。コントローラから出力されるバンダリ・スキャン・テストデータは、ここでイーサネット通信フォーマットにプロトコル変換され、ギガビット・イーサを構成するネットワークへ送信される。

Downlink は、テストクライアントを構成するテスト・ターゲット側の TAP に接続する。ネットワークからの受信データをプロトコル変換し、バウンダリスキャン・テストデータに復元する。このデータにより UUT 上の TAP を制御して、バウンダリ・スキャン・テストを実行する。

Uplink と Downlink 間の通信リンクは、ギガビット・イーサだけでなく、無線 LAN、CAN 等が適用できる。さらに、JTAG コントローラ機能をモジュール化して製品ボードに組込み、フィールドの製品に対して、遠隔診断によるボード実装テストおよびオンボード LSI テストを実現することが可能である。

①複雑な組込みシステムがフィールドに出た後に、定期的に実装ボード・テストおよびオンボードLSIテストを実施することで、その製品が出荷検査時と同等の機能・性能を継続して保持しているか否かを判別する。
②特に、自動車等で、これらに搭載される組込みシステムに、熱や振動によるストレスが長時間加わり、実装およびLSIが劣化する可能性が大きいシステムで有効。
③複合機等で従来のサービスマンによるメンテナンスから、遠隔診断によるエンベデッド・テストを利用したメンテナンスおよび遠隔モニタリングに置き換え、トータルコストを削減する。
④ 地上のテスト装置から、地球を周回する人工衛星上のUUTをテストする。

システム・レベル・テストの結果、異常なボードが検出された場合、そのボードを切り離して運行させる。

以上のように、大規模組込みシステムの故障の診断、故障箇所の特定、故障からの回復、予防保全の面で、信頼性・安全性の向上にJTAG技術の適用が大きく寄与する可能性を持つ。

(8) おわりに

IEEE1149.1に関連する規格として、1995年にIEEE Std 1149.5「Module Test and Maintenance Bus (MTM-Bus) Protocol」、1999年にIEEE Std 1149.4「Mixed Signal Test Bus」がそれぞれ承認された。前者は、バス結合されたデジタルLSI搭載の複数ボード配線テストをシステムレベルで行うもので、後者は、バウンダリ・スキャン手法を拡張し、アナログ回路にも対応できるようにした規格であり、アナログ・デジタル混載ボードのテストの実現を目ざす。

さらに、2003年にはIEEE Std 1149.6 「Advanced Digital Networks」が承認され、IEEE1149.1の能力がAC結合や差動などの高速インターコネクトをテスト・診断できるように拡張された。

SOC (System on a Chip) 時代を本格的にむかえた今日、JTAG の三つの機能、実装テスト、ISP, デバックおよびモニタリングがますます重要になってきた。
この技術はデバイスのユーザーに最大の恩恵を与える。デバイス・メーカ各社が、よりユーザーのCS (顧客満足) を念頭においてJTAG技術の活用を図ることを期待する。

本稿ではJTAGテストが、組みシステムの信頼性・安全性の向上に寄与することを解説した。今後、組込みシステムの応用領域がより広がり、その機能・性能が急速に拡大していく中で、広義のJATGテスト手法は組込みシステムの信頼性・安全性の向上を培う手段として、より重要な役割を担っていくことになるであろう。

機能安全は単なる安全とは異なる。安全機能を制御することによって獲得される安全をいう。
・組込系企業が機能安全に取り組む利点は、技術と安全という時代の要請に取り組むという利点がある。
・機能安全の規格であるIEC 61508は大きな化学プラントなどの事故などを契機に成立した。
・IEC 61508は機械や電気などの安全規格のなかで、電気系をになう安全規格である。
・ただしこの規格はソフトウェア制御が社会的に増大してきていることから、ソフトウェアに関する規格が重要視されている。

・品質や安全性、また信頼性に関して知見を持つことが重要である。
・検証性、妥当性、また日本的なソフトウェア開発の仕方にも「合理性」があるという認識も重要である。
・機能安全は情報セキュリティとも関係するように規格の改訂が検討されているようである。
・組込みシステムの基盤を支えるハードウェアの新たな JTAG テスト法が存在する。

《節 6.1～6.3.3 までの引用文献》

1) IEC, TR-61508-0, Part0: Functional safety and IEC 61508, 2005-01, p.13
2) 社団法人日本電気計測器工業会、機能安全と JIS C 0508, ｐp.3-4, 2003
3) IEC, 61508-3, Part3: Software requirements, 1998-12, p.13
4) 社団法人日本電気計測器工業会、機能安全と JIS C 0508, ｐp.11-14,2003
5) 向殿政男、機械安全規格と欧州規格の動向、標準化と品質管理, Vol.157, No.11, pp.30-34, 日本規格協会, 2004-11
6) 飯泉紀子、田所孝文、吉澤智美、大野康昭、安全関連システムの分割要件の考察、日本科学技術連盟、2006 年度分科会成果報告、第三分科会 機能安全グループ p.1
7) 飯泉紀子、田所孝文、吉澤智美、大野康昭、安全関連システムの分割要件の考察、日本科学技術連盟、2006 年度分科会成果報告、第三分科会 機能安全グループ p.13
8) IEC 61508-5, Part5: Examples of methods for the determination of safety integrity levels, 1998-12, p.33
9) IEC 61508-5, Part5: Examples of methods for the determination of safety integrity levels, 1998-12, p.35
10) IEC 61508-3, Part3: Software requirements, 1998-12, p.27
11) 財団法人日本船舶標準協会、船舶の安全システムの評価に関する基礎調査報告書、付 4 安全に関する諸考察、4-1 安全性と信頼性、平成 12 年度
12) 林晋、黒川利明、二つの合理性と日本のソフトウェア工学、科学技術動向、文部科学省、2004
13) フジワラヒロタツ、フジワラヒロタツの現場検証、インターフェース、CQ出版、1997-7～2003-10 またhttp://www.cqpub.co.jp/interface/column/genba/default.htm
14) 林晋、黒川利明、二つの合理性と日本のソフトウェア工学、科学技術動向、文部科学省、2004,p.10

《節 6.1～6.3.3 までの特定の引用はないが参考になった主な文献》

1) 水口大知、長谷部浩二、ソフトウェアの安全性をめぐる課題について、日本信頼性学会、Vol.157, No.5,2007
2) 水口大知、機能安全対応のためのソフトウェア安全分析手法、エンベデッドフォーラム in グレーター・ナゴヤ（講演資料）、2008-1/29

3) 田辺安雄（株式会社日本機能安全)、機能安全の考え方と組み込みシステム、財団法人組込みシステム技術協会「機能安全・セキュリティーセミナー」(講演資料)、2007-7/10
4) 財団法人日本船舶標準協会、船舶の安全システムの評価に関する基礎調査報告書、平成12年度
5) 独立行政法人産業技術総合研究所システム検証研究センター、機能安全規格と適合認証（61508のさらなる理解に向けて)、2006

《節6.3.4の参考文献》

1) 宇賀神　孝、Peter van den Eijnden他、「ボード・テスト・クライシスの傾向と対策」バウンダリ・スキャン・テストのトラブルシューティング、 Design Wave Magazine 2004、February
2) 宇賀神　孝、「組込みシステムを陰で支えるJTAGボード・テスト入門」、Bulletin JASA 2006 Nov. vol.29 p18-21
3) Vishwani D.Agrrawal他、「LSIにBIST手法を組み込み、システムのテスト・コスト低減」、バウンダリ・スキャンの併用で効果が倍増、from AT&T Technical Journal vol 73,March/April 1994、日経エレクトロニクス 1994.10.20 (no.619)
4) 石川　禎、長部　均　他、「バウンダリ・スキャン・テストをシステム・レベルで実現」、 東芝が産業用コンピュータで適用、日経エレクトロニクス 1995.4.10 (no.633)

おわりに

ここで扱った情報セキュリティと機能安全は、技術的な方法のみではなく、セキュリティや安全の内容に多くかかわる問題である。それはまた技術や組織管理に関する第三者認証という社会制度も含んでいる。

情報セキュリティや機能安全を技術の問題のみとしてとらえようと期待した読者は、内容を整理するためのコトバの説明などにはうんざりしたかもしれない。しかし何を安全と呼ぶかは内容を明確にせざるを得ない。曖昧な仕様書でシステム開発を委ねられ、ヘキヘキとしないようにすることと似ている。

“組込み屋”も方法だけが仕事といっていられないかもしれない。たしかに開発するソフトウェア量が膨大になりつつある中では、品質の保証は最優先されねばならないであろう。状態遷移図の標準化、形式手法の取り込みなど、どれも容易ではない。

しかし時代が要求する情報セキュリティや機能安全は、多くのシステムに当てはまる汎用性の高い課題である。よって、これらの課題に取り組むことは企業価値を高めることにつながるであろう。

ところで、この委員会を通じて組込み企業に共通な課題を具体的につむぎだすことができた。比較的小規模の組込み系企業が共同して、情報セキュリティ管理体制の認証ＩＳＭＳを取得したらどうかという提案である。来期はこの実現も課題のひとつにしてみようと考えている。当然今期の課題を深めてゆく活動も行う。

本報告者を作成にするにあたり委員の方々には大変お世話になった。次頁に委員各位の名を記し（敬称略、50 音順)、あらためて謝意を表したい。

平成 20 年 3 月

機能安全委員会　情報セキュリティワーキンググループ

委員長　　漆原　憲博

機能安全委員会　情報セキュリティワーキンググループ委員

漆原　憲博　（委員長）株式会社ジェーエフピー
宇賀神　孝　　アンドールシステムサポート株式会社
小澤　健一　　株式会社ジェーエフピー
金田　光範　　東芝システムテクノロジー株式会社
岸本　輝昭　　岩手県立大学
済賀　宣昭　　東海ソフト株式会社
竹岡　尚三　　株式会社アックス
田向　忠夫　　株式会社ジェーエフピー
中村　憲一　　アップウィンドテクノロジー・インコーポレーテッド
萩原　秀和　　株式会社ミントウェーブ
前田　勝之　　株式会社ディー・ディー・エス
三輪　一義　　ＩＳＭＳ審査員
門田　浩　　日本電気株式会社

平成 19 年度

組込みシステムにおける機能安全に関する調査研究

ネット社会における組込みシステム、2つの課題
「情報セキュリティ」と「機能安全」

平成 20 年 3 月
社団法人　組込みシステム技術協会
〒103-0007　東京都中央区日本橋浜町 1-8-12
電話　03-5821-7973
FAX　03-5821-0444
http://www.jasa.or.jp