平成 14 年度　研究開発成果報告書

「ＰＣなどオープンアーキテクチャーデジタル放送受信機に対応する権利保護システムの研究開発」

目　　次

1　研究課題の背景

１９９９年末のＢＳデジタル放送開始を皮切りに、２００２年には１１０°ＣＳデジタル放送が開始され、図１－１のように２００２年は２００１年のほぼ倍となっている。

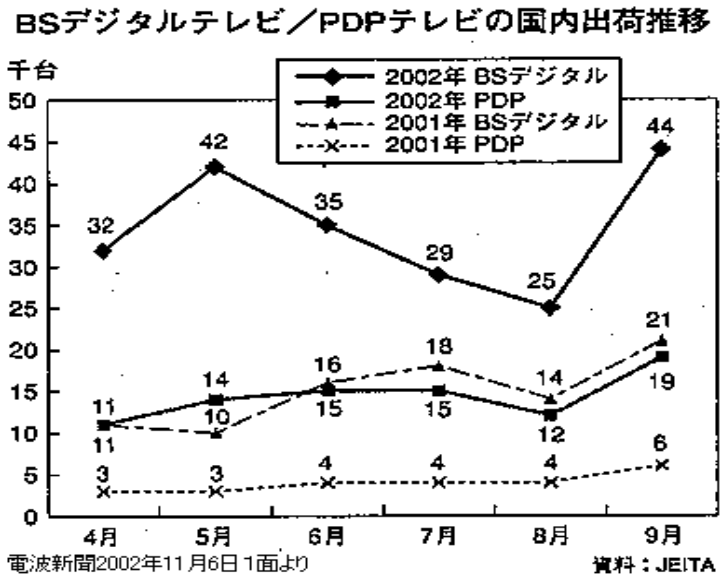


図１－１

また、ＢＳデジタル放送の普及率は２００３年２月末で約３８４万世帯（ＮＨＫ調べ）となり、受信機器の総出荷台数も２００３年３月末で約１９６万台（ＪＥＩＴＡ：日本技術産業協会調べ）と、順調に普及している。更に、２００３年末には一部地域から全国へと地上デジタル放送が開始･展開される予定である。

デジタル放送開始と共にもう一方で、これらデジタル放送受信機にハードディスクを搭載させ、受信したデジタルＡＶコンテンツをハードディスクに蓄積、コンテンツ課金やタイムシフト再生など行う新サービスが普及する。受信機でのハードディスクの蓄積媒体の存在を前提としたサーバー型放送の規格化も策定されている。これに伴いデジタルＡＶコンテンツ複製や移動を制限するコンテンツ権利保護システム研究開発が急務となっている。

過去の一例ではあるが、ＰＣから音楽をインターネット上に配信するナプスターは、コンテンツ複製や移動を制限する機能がないため著作権者からクレームが付き、サービスの停止を余儀なくされた。ＰＣなど内部が公開されたオープンアーキテクチャー機器上での権利保護機構の困難さを示した一例と言える。

また、現状のＰＣ上ソフトウェアベースのシステムでは、本来的にコンテン

ツなどの情報の複製／移動、その他制御が可能であることが前提で、これを制限することはＰＣシステム全体の構想に合わないといったこともある。

しかし、今後更にデジタル放送が一般化し、ＰＣ上でもデジタルＴＶ視聴機能が実現されるようになると、デジタルＡＶコンテンツもＰＣ上に保存されるようになり、ますますＰＣ上のコンテンツ権利保護が重要となる。デジタルＡＶコンテンツの著作権者は、ＰＣ上に強固なコンテンツ権利保護機構が構築されない限り、ＰＣ上でのデジタルＡＶコンテンツ保存を容認しないものと思われる（現在販売中のＢＳデジタル受信機能を持つＰＣであっても、デジタルＡＶコンテンツの保存機能を有していない）。

一方、ＰＣ上で保存されたデジタルＡＶコンテンツの権利保護が有効な方法で実現すると、ＰＣからインターネット経由で世界中にデジタルＡＶコンテンツが流通する市場が開かれる可能性がある。

本研究開発は、ＰＣでのデジタル放送受信とデジタルＡＶコンテンツ保存を前提に、ＰＣソフトウェア上でのコンテンツ権利保護を実現する技術に関するものである。現状ＰＣソフトウェア上でのコンテンツ権利保護は、基本的に保護アルゴリズムを秘匿化することで行われている。そのため、ソフトウェアの解析を困難にする「難読化」などが中心である。しかし、これでは一旦アルゴリズムが解析されてしまうとコンテンツ権利保護が困難になるため、セキュリティレベルが低くなり、その強化が必要である。本研究は、ＰＣなど（１）ソフトウェア処理が主体で、かつ（２）オープンアーキテクチャー構成のデジタル放送受信機のコンテンツ権利保護の研究開発に関するものである。

２　研究開発分野の現状

デジタルコンテンツの権利保護や流通に関しては、これまでにも各社、各団体により提案がいくつもされてきている。例えば，DVD video における著作権保護システムであるＣＳＳ方式や、リムーバブルメディア向けＣＰＲＭ方式、ＩＥＥＥ１３９４を用いた方式(DTCP)など存在する。

- CSS 方式(DVD における著作権保護システム)　詳細は非公開
- CPRM(Content Protection for Removable Media)方式　詳細は非公開
- DTCP(Digital Transmission Content Protection)
  http://www.dtcp.com/data/dtcp_tut.pdf など

これらの方式においては、ハードウェアやソフトウェアの耐タンパー性が重要とされている。ハードウェアの耐タンパー技術はある程度確立されてきた技術であり、コンテンツ提供者サイドにもある程度理解されているものと考えている。このため、ハードウェアによる専用装置（デジタルＴＶなど）の世界においては耐タンパー性を利用したコンテンツ保護システムの構築は可能となってきている。

しかし、ＰＣのようなオープンアーキテクチャー上でソフトウェアが動作するような世界では、全てをハードウェアで処理することへの抵抗や、全てをハードウェアで処理するアーキテクチャが受け入れられ難い状況である。この為、ソフトウェアの耐タンパー性が強く求められている。

一般的にＰＣのようなオープンアーキテクチャー上でのソフトウェアの耐タンパー性は、ハードウェアよりも低いレベルであり、ソフトウェアの解析から完全に守ることは難しいとされている。現在、各社で実施されているソフトウェア耐タンパーの技術には大きく分けると以下の２つの技術が用いられている。

【難読化】

ソフトウェアに余分なコードを付加したり、簡単な計算処理をわざと複雑な計算ルーチンを用いて計算させることで、逆アセンブルによるコード解析に時間がかかるようにすることで耐タンパー性を持たせようとする方式。

【暗号化手段を利用】

ソフトウェアを暗号化してハードディスク上に格納しておく方式。また、ソフトウェアを細かいモジュールに分割して実行させ、モジュール間の相互認証などをさせる方式も存在する。内容は一般的には公開されておらず（方式を未公開にしないと安全性を保てないため）内容は不明だが、公開されている技術論文としては、

- 「逆解析や改変からソフトを守る」(日経エレクトロニクス 1998.1.5(no706)) 耐タンパなソフトウェア構造の提案論文

が存在する。ただし、暗号化技術を用いた場合には、対象となるソフトウェアの攻撃だけでは解析することはできませんが、ソフトウェアを復号する別のソフトウェアが存在するわけで、これを含めて攻撃されるとソフトウェアの解析から守ることは困難となる。

更に近年になって、ＰＣそれ自体のアーキテクチャに手を入れ、ＰＣとその上で動作するＯＳ(Operating System)をセットにして、耐タンパー性を高くしようという動きがある。

米国 Microsoft 社は、“Palladium（パレイディアム：後に NGSCB: Next-Generation Secure Computing Base と改名)”を提唱している。

- http://www.microsoft.com/presspass/features/2002/jul02/0724palladiumwp.asp

簡単に書けば、Palladium は既存のＰＣ内に、もう一つ、高度なセキュリティを持ったＰＣを設け、例えば内部のセキュアなＰＣの隠しメモリにソフトウェアを置くなどの方式で、解析や攻撃からソフトウェアを守るといった方式である。

しかし、この方式では、ＰＣそのもののアーキテクチャの変更が必要であり、既存のＰＣでは対応できず、かつ、ＰＣのオープンアーキテクチャー性が失われていく可能性が大きい。

今回の研究では、上記のようなソフトオンリーのセキュリティーでは、現在の技術水準においての十分な安全性（耐タンパー性）の確保（方式が分かってしまうと、ソフトウェアの安全性を保てない）は困難であり、また現行のＰＣアーキテクチャの全面的な変更をしないという方針のもと、詳細な検討の上に抜き出した最小限のセキュリティー機能を搭載したハードウェア（ＴＲＭ:Tamper Resistant Module:化したセキュアハード）をベースに、ソフトウェアの安全性を確保する手段を研究開発するものである。

セキュアハードのコストを低くするため回路規模を小さくするよう検討すると同時に、セキュアハードの普及が容易なように、ＰＣＩバス等の公開されたバスにセキュアハードを接続することを前提に研究開発を進めている。

なお当社では、数年に渡って超流通やデジタルコンテンツ保護の研究開発を進めてきたという実績がある。

鳥居直哉，長谷部高行，武仲正彦，木島裕二：``超流通システムの試作",

電子情報通信学会技術研究報告，Vol.96, No.71(OFS96-10), pp.1--5 (1996)

- 長谷部高行，鳥居直哉，武仲正彦：``超流通システムの試作（課金サーバ型）"，電子情報通信学会　基礎・境界ソサイエティ大会講演論文集，SA-5-6, pp.283--284 (1996)
- 木島裕二，長谷部高行，鳥居直哉：``超流通におけるコンテンツ流通のための課金機構の開発"，創造的ソフトウェア育成事業及びエレクトロニック・コマース推進事業 最終成果 発表会論文集 創造的ソフトウェア育成事業編，pp.701--704 (1998)
- 長谷部高行, 木島裕二, 鳥居直哉：``超流通における課金機構の開発"，情報処理学会研究報告, Vol.98, No.85 (98-EIP-2),pp.15--19 (1998)
- 田平孝彦、瀬野尾晴海、小川清隆、小檜山清之、秋山良太, ``MPEG-TS（デジタルビデオ／オーディオ）セキュリティー方式の開発"、テレビジョン学会年次大会、No. 23-3（1996）

これらの研究がベースとなり、既に実際に製品運用されている以下の権利保護関連システムがある。それは、ＤＤＩポケット社が提供する Sound Market と呼ばれるサービスである。そのサービスで採用されている、ケータイ de ミュージック方式と呼ぶ方式は当社等により開発された方式である。

- http://www.keitaide-music.org/index_j.html

これは、音声の権利保護が目的でＰＣのようなソフト主体のオープンアーキテクチャーシステムではないが、権利保護やコンテンツ課金に必要な以下の全ての機能が揃っている。

- ライセンスとコンテンツの分離技術
- ライセンス管理技術（メディアベースランセンス管理）
- 暗号化実装技術（ハードウェア、ソフトウェア）
- 相手認証や再送防止等といったプロトコル技術
- ハードウェア耐タンパー技術、など

今回の研究開発における課題としては、さまざまなソフトウェアが動作するＰＣのようなオープンアーキテクチャー上において、ＰＣのアーキテクチャの変更無しに、詳細な検討の上に抜き出した最小限のセキュリティー機能を搭載したハードウェアを搭載した形のソフトウェアベースのコンテンツ保護が、いかなる形式ならば可能なのかを研究し、その実装技術を開発すること。また、実際のデジタル放送にこれらの技術を適用した場合にＰＣなどのオープンアーキテクチャー上での動作が可能かを実証することである。

３　研究開発の全体計画

３-１　研究開発課題の概要

１９９９年末のＢＳデジタル放送開始を皮切りに２００２年は１１０°ＣＳデジタル放送、２００３年には地上デジタルが開始される。デジタル放送開始と共にもう一方で、受信機にハードディスクを搭載し、受信したデジタルＡＶコンテンツをハードディスクに蓄積しコンテンツ課金やタイムシフトなど行う新サービスが普及する。受信機でのハードディスクの蓄積媒体の存在を前提としたサーバー型放送規格化も策定されている。これに伴いデジタルＡＶコンテンツ複製や移動を制限する権利保護システム研究開発が急務になる。

本研究は、ＰＣでのデジタル放送受信を前提にＰＣソフトウェア上で権利保護を実現する技術に関するものである。現状ＰＣソフトウェア上での権利保護は、基本的に保護アルゴリズムを秘匿化することで権利保護している。そのためソフトウェアの解析を困難にする「難読化」などが中心である。しかし、これではアルゴリズムが解析されると権利保護が困難になるため、セキュリティレベルが低く強化が必要である。本研究は、ＰＣなど（１）ソフトウェア処理主体かつ（２）オープンアーキテクチャー構成デジタル受信機の権利保護の研究開発に関するものである。

過去の例では、ＰＣで音楽をインターネット配信するナプスターは、コンテンツ複製や移動を制限する機能がないため著作権者からクレームが付きサービスを停止された。ＰＣなど内部が公開されたソフトウェア処理主体機器での権利保護機構実現の困難さを示した例と言える。これは現在のソフトウエアベースのシステムでは、本来的に情報の複製／移動、その他制御が可能であることが前提でこれを制限することはシステム全体の構想に合わないためである。

今後デジタル放送が一般化し、ＰＣ上でデジタルＴＶ機能が実現されるようになると音楽のみならずデジタルＡＶコンテンツもＰＣ上に存在するようになり、ますますＰＣ上のコンテンツ権利保護が重要になる。デジタルＡＶコンテンツの著作権者は、強固な権利保護機能を持たない限り、ＰＣ上でのデジタルＡＶコンテンツを容認しないものと思われる。

一方でＰＣ上でのデジタルＡＶコンテンツ権利保護が一旦実現するとＰＣからブロードバンドインターネット経由で世界中にデジタルＡＶコンテンツが流通する可能性が開ける。

図３－１は一般的な既存ＢＳデジタル放送受信機能搭載ＰＣの機能ブロック図であり、デジタルＡＶコンテンツをハードディスク蓄積すると想定している。信号の流れを概説すると以下のようになる。放送波は「デジタルチューナモジュール」より入力され、デジタルＡＶコンテンツを時分割多重の形で持つ

ＭＰＥＧ－ＴＳデジタルストリームが出力される。ＭＰＥＧ－ＴＳデジタルストリーム中の視聴者が選択した番組（特定デジタルＡＶコンテンツ）はＭＵＬＴＩ２暗号化されており、「ＭＵＬＴＩ２暗号復号モジュール」で復号される。復号されたデジタルＡＶストリームはハードディスク蓄積のため再暗号化され、再暗号化デジタルＡＶコンテンツがハードディスク等の蓄積媒体にＰＣＩバス等を経由し蓄積される。ＭＵＬＴＩ２暗号復号に必要な暗号鍵等のライセンス情報は「Ｂ－ＣＡＳカード等ライセンス保持モジュール」から出力される。また、再暗号化に必要なライセンス情報は、「ライセンス生成モジュール」で生成される。

ＨＤＤ蓄積された暗号化デジタルＡＶコンテンツを再生する場合は、「暗号復号モジュール」に転送され復号され、復号デジタルＡＶ情報は、「ＭＰＥＧ等デコードモジュール」で伸長処理される。伸長デジタルＡＶ情報は、「グラフィックモジュール」でグラフィックがオーバレイされ、その後アナログ出力の場合はアナログ著作権保護情報が付加され出力される。デジタル出力の場合は、デジタル著作権保護情報が付加され出力される。

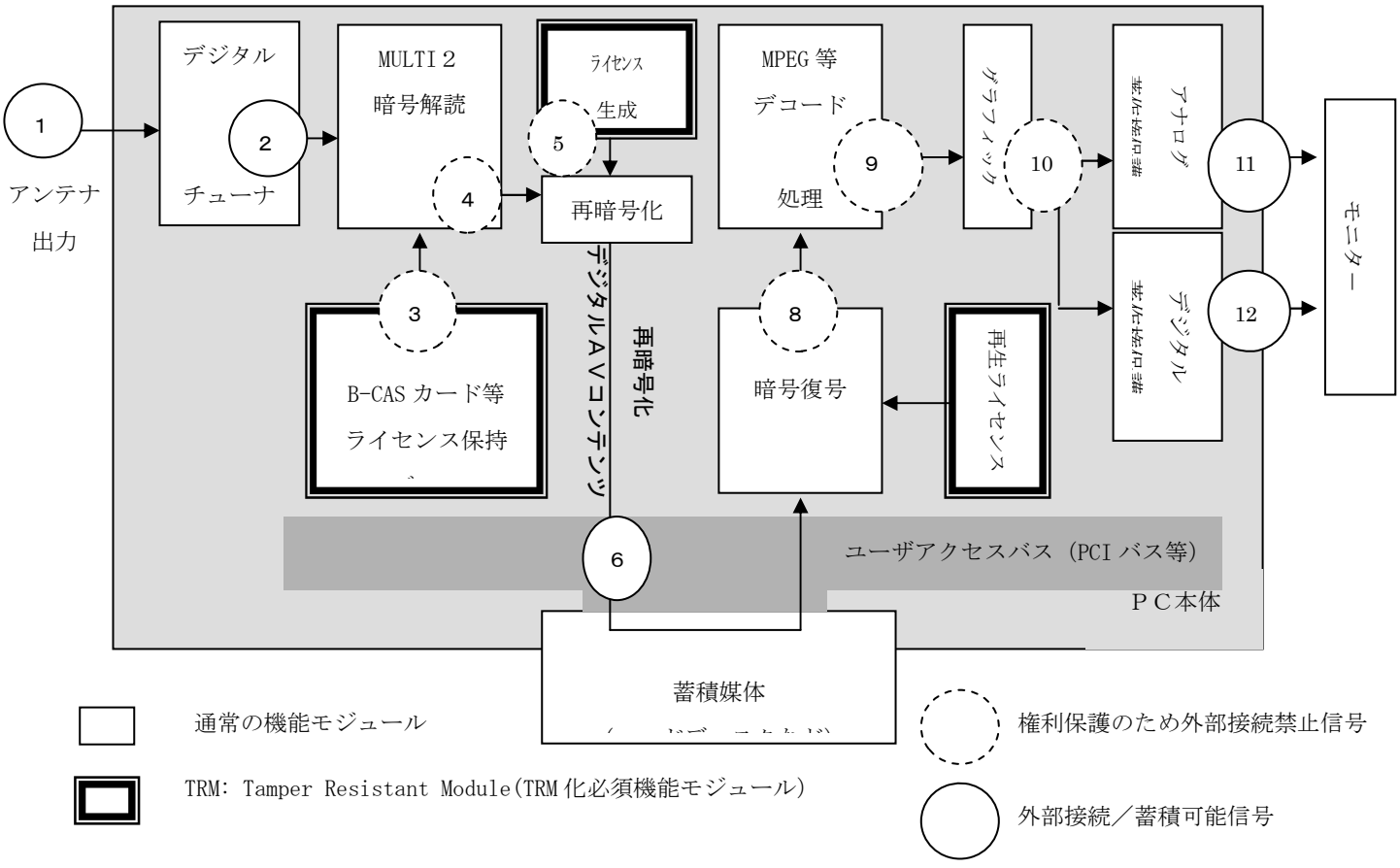


図３－１　ＰＣ等ユーザアクセスバスを有するデジタル放送受信機の機能ブロック図（例）

図３－１で暗号解読鍵等のライセンス情報を持つ機能モジュールは TRM（Tamper Resistant Module）である必要がある。また、その他のモジュールは TRM である必要はないが図１で点線の信号（信号４、５、８、９、１０など）は、権利保護上、外部流出禁止信号である。例えば、暗号復号後の信号８などが外に漏れると著作権者の権利を保護することができなくなる。

また、図３－１は機能モジュールであり、各モジュールともソフトウェアで実現してもハードウェアで実現しても機能的には同様に動作する。

図３－２は、ＬＳＩ（ハードウェア主体）で図３－１の機能を実現した場合のブロック図である。太線内がＬＳＩであり、ほとんどの機能がＬＳＩ内に搭載され、ＬＳＩ全体を TRM（Tamper Resistant）化することで外への信号流出が防げ、比較的容易に権利保護が実現する。しかし、ハードウェアを搭載することはＰＣのコストアップに繋がる。特にＭＰＥＧデコード機能は回路規模が大きく高価格である。

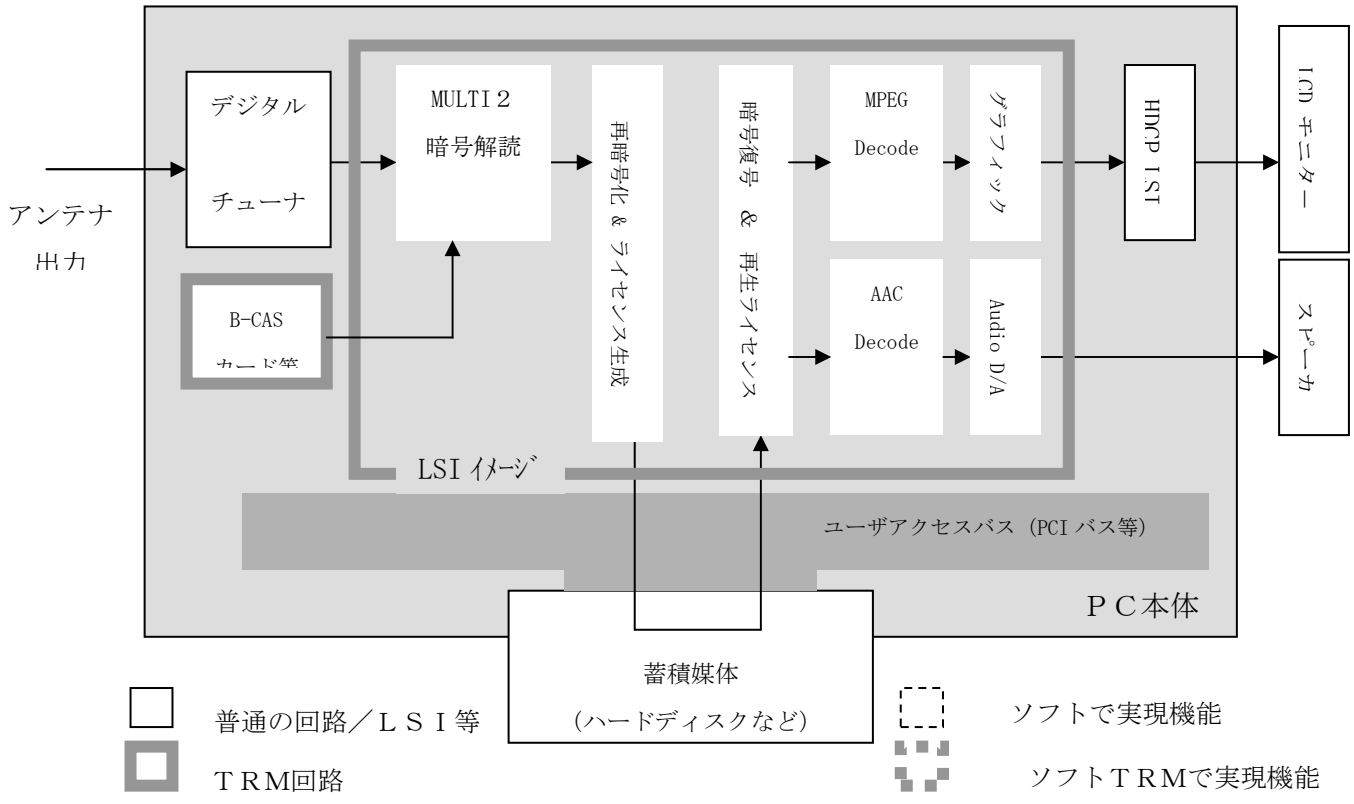


図３－２　想定されるハード処理主体の実施例

またＰＣはソフトウェア主体の装置であるが、この構成ではソフトウェア／プロセッサ等の資源が有効利用されていると言い難い。さらに、既に DVD プレイヤー（ＭＰＥＧ　ＭＰ@ＭＬ圧縮デジタル情報のデコードが必要）が一般のＰＣ上でソフトウェア処理されていることを考えるとプロセッサの処理能

力が今後増大した場合、デジタル放送（ＭＰＥＧ　ＭＰ＠ＨＬ圧縮デジタル情報のデコードが必要。ＭＰ＠ＨＬはＭＰ＠ＭＬの６倍の処理能力が必要）でも一般ＰＣでソフトウェア処理可能になるのは時間の問題と考えられる。

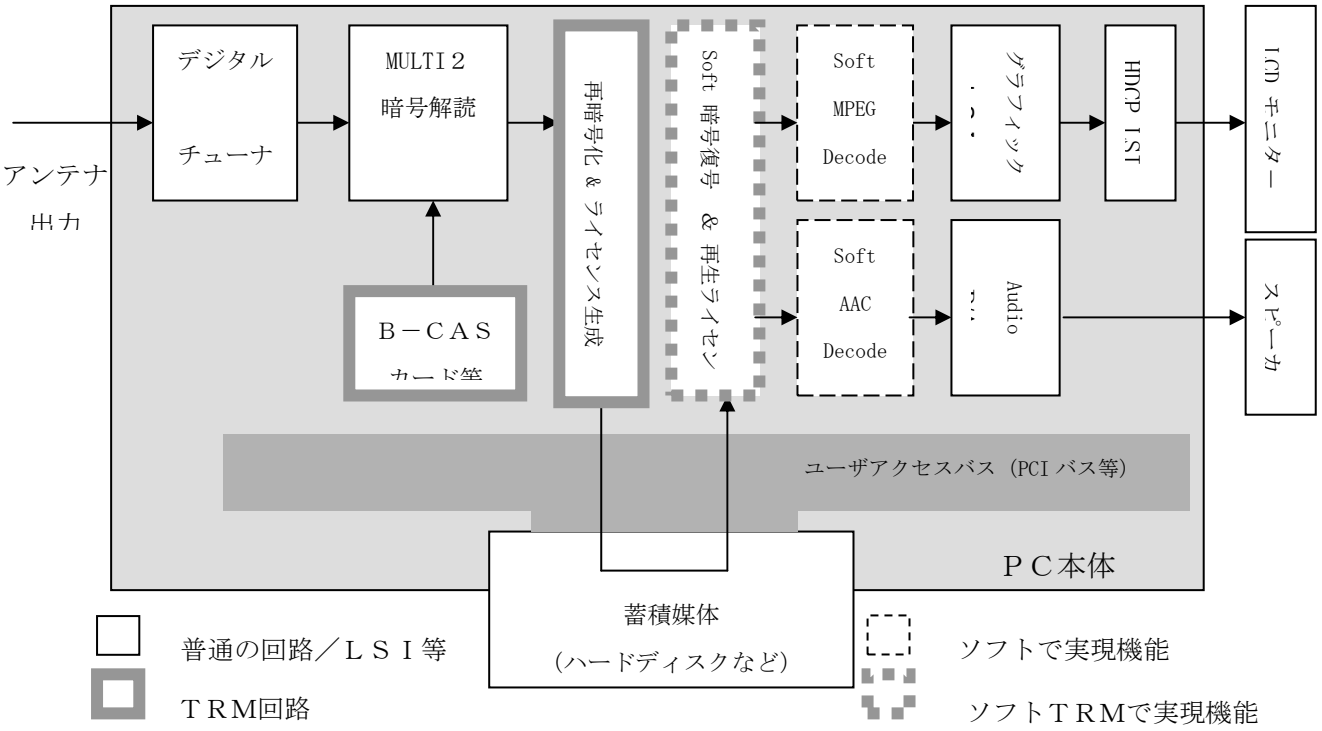


図３－３　想定されるソフト処理主体の実施例

図３－３は、ソフトウェア処理を主体とした場合のシステム構成例である。ハードウェア処理した場合に回路規模が大きくコストアップにつながる「ＭＰＥＧデコード」、「ＡＡＣ音声デコード」や「暗号復号」処理などがソフトウェア処理されている。従って、ソフトウェア処理が実現した場合、ハードウェアで実現したときと比較してかなりのコストダウンが図られると考えられる。

ソフトウェア処理した場合の課題は、「暗号復号」に必要な暗号解読鍵の保護や「ＭＰＥＧデコード」など、各ソフトウェア処理モジュールからの出力信号の保護である。

以上の検討でPCのようなソフトウェア主体装置の場合、ハードウェア処理を持ち込むことはソフトウェア主体装置の持つ本来の可能性が大幅に制約されることが分かる。

従って、本研究方針は上記検討で「ＰＣなどソフトウェア処理主体の装置の可能性を損なわないデジタルＡＶコンテンツの権利保護システム」に絞った方がより本質的であることも分かる。しかし、一方ですべてソフトウェアのみの

処理では、強固なデジタルＡＶの権利保護は困難と思われる。すべてソフトウェア処理とした場合、最終的に暗号鍵等の機密保持が非常に困難と考えられるためである。

現状の技術水準を考慮した場合、処理の大半をソフトウェア処理するが最低限の信号処理はハードウェア化（セキュアハード）したデジタル放送対応の権利保護システムの研究開発が現実的であると考える。

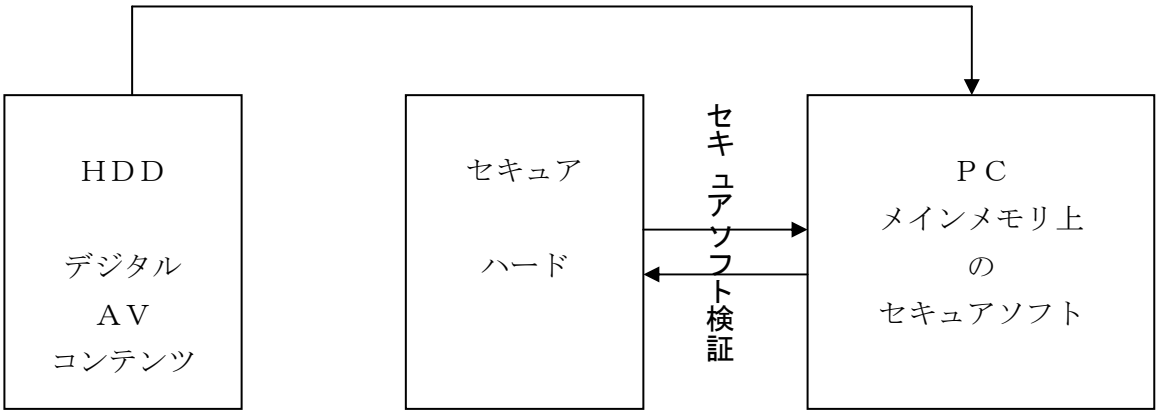


図３－４　想定されるセキュアソフトとハード

本研究で想定するデジタルＴＶ機能のシステム構成ブロック図を図３－４に示す。このシステム構成における課題は、以下のようなものである。

ハードディスク内のデジタルＡＶコンテンツは、ＰＣのメインメモリ上のセキュアなソフトウェアで処理される。一方でセキュアなソフトウェアの安全性を保障するのはセキュアハードになる。最終的な信頼点としてのセキュアハードを活用として、権利保護アルゴリズムを構築するという点が大きな研究開発課題になる。

またこの権利保護システム全体にリアルタイム処理性が要求される。処理するコンテンツは、基本的にハイビジョン動画像（ＭＰＥＧ　ＭＰ＠ＨＬレベル）であり、ＰＣ性能をギリギリまで使用しないと画像のリアルタイム処理は困難と考えられる。この中にセキュアなための処理をＰＣから見てあまり負担のかからない範囲でどう盛り込むか課題になる。安全性とＰＣ性能のトレードオフをどう考えるか、ハードディスク／セキュアハード／セキュアソフトウェア間の機能分担をどう捕らえるか、リアルタイム性というセキュリティーと別の視点から検討が必要である。

さらに別の視点としてコストがあげられる。２００４年以降店頭販売の一般ＰＣで活用されることを前提とした場合、コストを押さえるため、権利保護システム内の特殊部品はセキュアハードのみとして検討する必要がある。また、普及が容易なようにセキュアハードはＰＣカード搭載を前提とし、ＰＣ接続するときのインターフェースはＰＣＩバスなどユーザーに公開された汎用なものとする必要がある。

研究手法として、まずセキュアハード／ソフトの概略構成を初年度に検討し、大まかに権利保護システムの構想を固める。この時、図３－４の（１）ハードディスク／（２）セキュアハード／（３）PC 上のセキュアソフトの機能分担や想定されるセキュリティーホールを、特定し対策を検討する。１４年度は概略構成に従って、セキュアハード、セキュアソフトの基本仕様の検討を行い、機能試作を行う。

３-２　研究開発目標

３-２-１　最終目標

ＰＣ上、ソフト処理主体でデジタルＴＶの権利保護を実現する基盤技術を完成させることを最終目標とする。

具体的にはセキュアソフト実現のための（１）セキュアハード コアＬＳＩ、（２）セキュアハードを利用したセキュアＭＰＥＧビデオ処理ソフト／セキュアオーディオ処理ソフトなど、（３）「権利保護ＰＣカード」などを完成させる計画とする。「権利保護ＰＣカード」の中にセキュア コアＬＳＩ以外に地上デジタル放送受信機能も搭載を計画する。また、旧ＰＣから新ＰＣに買い替えた時に必要な蓄積コンテンツのセキュアな視聴権移動機能などＰＣ上でデジタルＴＶ受信機を構成するための基本機能を揃えるものとする。セキュリティー強度は、基準となる物差しがないので言及困難であるが、基本的にオールソフト処理より強い、放送など公共性の高い網で適用可能なレベルを目指す。またリアルタイム処理性能は、２００４年のＰＣ上でＭＰＥＧ　ＭＰ@ＨＬレベルの画像を権利保護しつつ一般の視聴者が違和感なく動画像を視聴出来るレベルとする。

３-２-２　中間目標

基本の機能レベルの完成目標を平成１５年３月とする。セキュアハードの回路、セキュアソフトを機能レベルで完成させ、実際にＰＣ上でデジタルテレビ画像を見られるようにすることを計画する。

但し、ＦＰＧＡ搭載実験ボードで製作されたハードウェアは物理的なサイズが大きく、ＰＣカード等に実装出来るレベルでない。またＦＰＧＡは高価であり、サイズ的にもコスト的にも製品展開は不可能なレベルに留まる。製品展開が期待出来るレベルにするためには、ＦＰＧＡ搭載実験ボードのＬＳＩ化が不可欠あり、またリアルタイム性もこの段階では十分なレベルに到達していないと想定する。リアルタイム性の目標として毎秒５フレーム（最終目標は一般視聴者が違和感なく動画像を視聴出来るレベル）の処理性能を想定する。

３-３　研究開発の年度別計画

（金額は非公表）

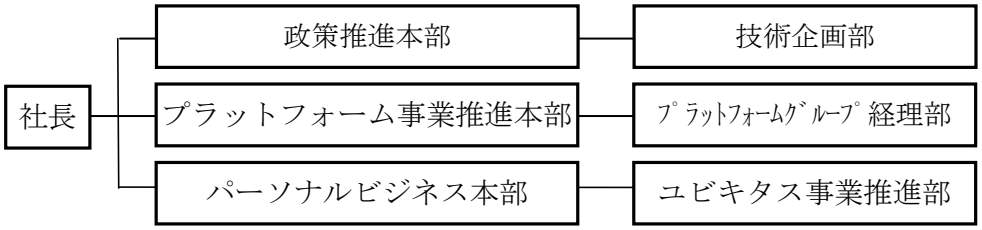
研究開発項目	１３年度	１４年度	１５年度	計	備　考
アーキ検討、実験ボード開発 → 設計、試作（ＦＰＧＡ）、 ソフトウェア開発 → 改良、ＬＳＩ化、ＰＣカード開発 →					
間接経費					
合　　計					

注）１　経費は研究開発項目毎に消費税を含めた額で計上。また間接経費は直接経費の３０％で計上（消費税含む。）。
　　２　備考欄に再委託先機関名を記載。

３-４　研究開発体制

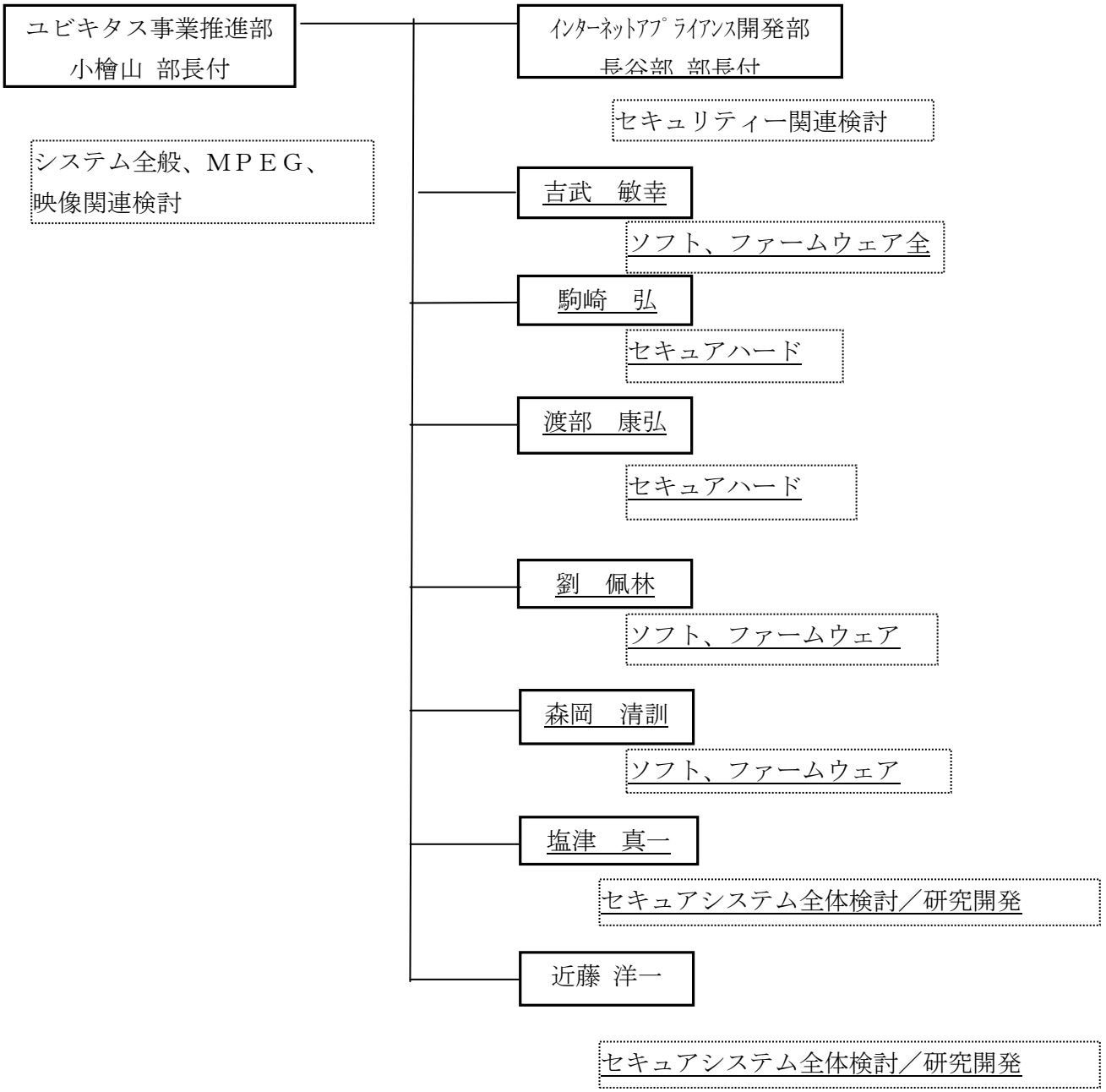
３-４-１　研究開発管理体制

（注　受託者の経理部門の体制、経理責任者（所属、氏名、電話、ＦＡＸ、Ｅ－メールの連絡先）を含む。）



３-４-２　研究開発実施体制

＜研究代表者＞



4　研究開発の概要

4－1　研究開発実施計画

4-1-1　研究開発の計画内容

4-1-1-1　平成１３年度

初年度の１３年度は、ＰＣ型のセキュアデジタル放送受信機のセキュアハード／ソフトの概略構成を検討し、大まかに権利保護システムの基本アーキテクチャを固める。またこの構想に従い、ＦＰＧＡ（Field Programmable Gate Array）を使用した構成でセキュアハード実験ボードの基本部分を製作する。この実験ボードは１４年度以降のセキュアデジタル放送受信機の性能や機能評価などを行うための実験プラットフォームという位置付けで製作する。１４年度以降の実験などで改良点や問題が発覚した場合、その時点でハードウェアの設計を容易に変更出来るようにＦＰＧＡ（Field Programmable Gate Array）構成とする。

なお、実験ボードは、各種改良を経たのち、本研究開発の最終年度にはＬＳＩ化しＰＣカード搭載を目標とし、本研究開発が目標とする権利保護システムの「核」になる部分とする。

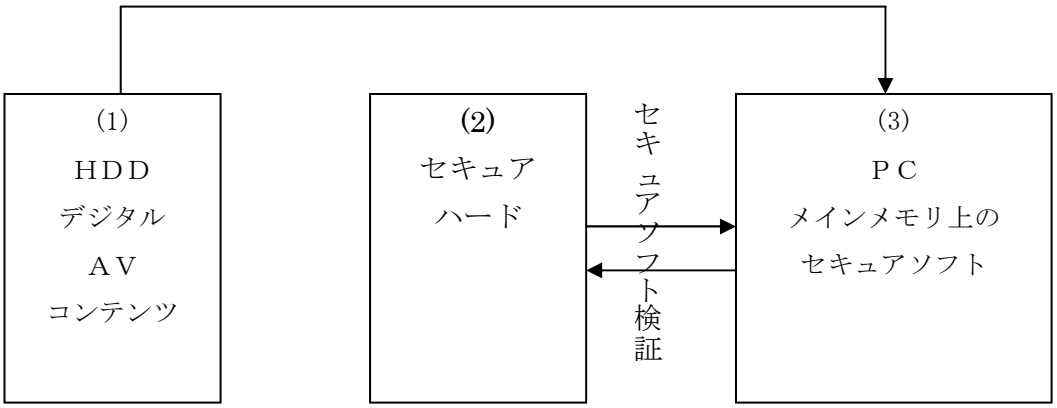


図４－１　想定されるセキュアソフトとハード

PC 型のソフトウェア処理主体のセキュアデジタル放送受信機は、セキュリティーという側面から見ると図４－１に示すブロック図のような構成が考えられる。本図に従い、平成１３年度はセキュアデジタル放送受信機の基本アーキの決定など今後の方向付けを行う。

図４－１でデジタル放送受信機は、（１）ハードディスク、（２）セキュアハ

ード、（３）ＰＣのメインメモリ上のセキュアソフトという基本要素から構成される。ハードディスク内に放送されたデジタルＡＶコンテンツが蓄積されており、これがＰＣメインメモリ上のセキュアソフトでデコード処理され、視聴される。一方でセキュアソフトの安全性（デコード処理後のデジタルＡＶコンテンツの著作権など権利保護機能）をセキュアハードが保証する。以上の基本構成を前提に１３年度研究課題として以下を検討する。

１）最終的な信頼点としてのセキュアハードの活用法の検討。

２）リアルタイム性とセキュリティーのトレードオフの検討。システム全体にリアルタイム性が要求される。処理コンテンツ（デジタルＡＶコンテンツ）は、基本的にハイビジョン動画像（ＭＰＥＧ　ＭＰ＠ＨＬレベル）であり、ＰＣ性能をギリギリまで使用しないとリアルタイムソフト処理は困難と考えられる。この中にセキュア処理をＰＣから見てあまり負担のかからない範囲でどう盛り込むか。セキュリティー（安全性）とＰＣ性能のトレードオフをどう考えるか、ハードディスク／セキュアハード／セキュアソフト間の機能分担をどう捕らえるか、リアルタイム性というセキュリティーと別の視点から検討する。

３）さらに別の視点としてのコストがある。本技術を２００４年以降店頭販売の一般ＰＣで活用されることを前提とした場合、コスト削減のためシステム内の特殊部品はセキュアハードのみとすることが望ましい。またセキュアハード普及を容易にするためセキュアハードはＰＣカード搭載とし、ＰＣ接続時のインタフェースはＰＣＩバスなどユーザーに公開された汎用なものが望ましい。

以上のようなことを前提とし、セキュアハード／ソフトの概略構成を１３年度に検討し、大まかなシステム構想を固める。

また本システム構想に従い、１３年度はＦＰＧＡ（Field Programmable Gate Array）構成でセキュアハード実験ボードを製作する。この実験ボードの最終目標はこれをＬＳＩ化し、図４－１のセキュアハードとすることにある。実験ボードにより１４年度以降、上記で検討したセキュアハードとセキュアソフト間で安全性検証のための特別な通信などを具体化し、検証実験を行う。実験ボードの基本部分をＦＰＧＡ構成とすることでセキュア化の研究が進み問題が発覚した時点でのセキュアハードの設計変更を容易にする。従って、このボードはセキュリティーシステム試作／実証のための基本プラットフォームとなるよう構成する。

なお１３年度はセキュアシステム構想を固めることが主眼であり、セキュアハード実験ボードの中のＦＰＧＡの具体設計は、１４年度以降とする。但し、セキュアハードにはＦＰＧＡ以外にプロセッサ、プロセッサ周辺回路、メモリ、ＰＣＩインタフェース回路などの搭載するため、プロセッサ周辺、ＰＣＩイン

タフェース回路の動作確認などは１３年度に行うものとする。

4-1-1-2　平成１４年度

１３年度の成果を踏まえ、基本機能レベル（基本デジタル放送受信機能）の完成目標を平成１４年度とする。セキュアハードの回路、およびセキュアソフトを機能レベルで完成させ、実際にＰＣ上でデジタルテレビ画像を見られるようにする。

但し、セキュアハードはＦＰＧＡ（Field Programmable Gate Array）搭載実験ボードで試作されるため、物理的なサイズが大きく、ＰＣカード等に実装出来るレベルでなく、また高価なものとなる。従って、サイズ的にもコスト的にも製品展開にはさらに開発が必要なレベルに留まる。製品展開が期待出来るレベルにするには、ＦＰＧＡ搭載実験ボードのＬＳＩ化が必要であり、これは１５年度の目標となる。またセキュアソフトなどのリアルタイム性も１４年度の段階ではさらに開発が必要なレベルと想定する。具体的なリアルタイム性の目標として毎秒約５フレーム（最終目標は一般視聴者が違和感なく動画像を視聴出来るレベル）の処理性能を想定する。

以上の計画を以下のような手順で実施する。１３年度に決定した基本アーキをベースに研究要素を特定し、それぞれの要素ごと検討を行う。

まず、図４－２に示すような基本アーキテクチャをセキュアシステムとして展開していく。

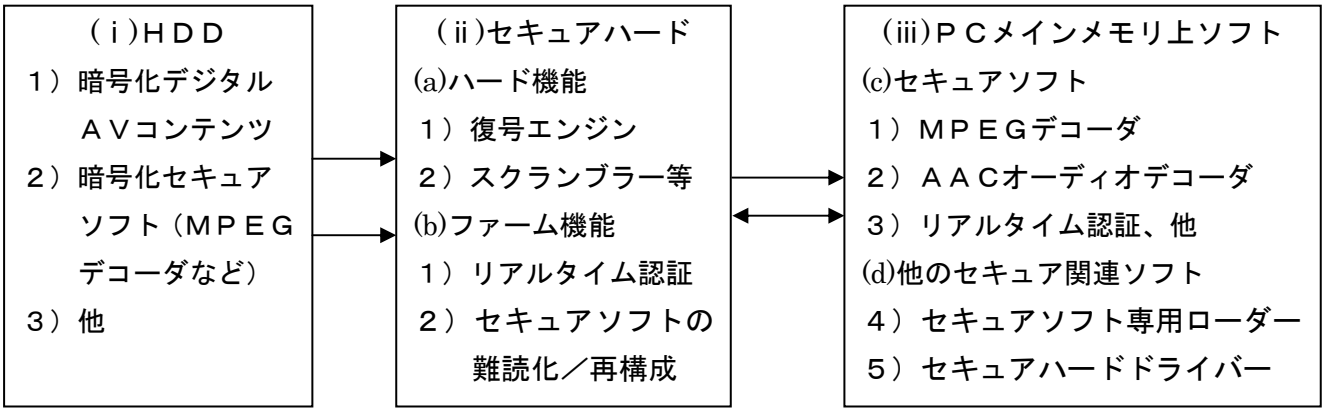


図４－２　１３年度の研究開発で想定されたセキュアソフトとハードの関係

全体基本アーキは以下のように考える。ＨＤＤ内に（１）暗号化デジタルＡＶコンテンツ、（２）暗号化セキュアソフト（ＭＰＥＧデコーダやＡＡＣオーディオデコーダなど）が存在する。デジタルＡＶコンテンツ再生には、暗号化セキュアソフトをセキュアハードに読込み、復号する。復号セキュアソフトに

対し、セキュアハード内で難読化／ソフト再構成などセキュア処理を加えた上でＰＣメインメモリ上に転送する。転送はセキュアソフト専用ローダーが行う。

ロードされたセキュアソフトは、セキュアハードがリアルタイムに認証する。常に同じ手順で認証したり、セキュアソフトが常に同じプログラムコードになっていたりするとＰＣ上のセキュアソフトや認証アルゴリズムを解析され、成りすましにより処理途中でデジタルＡＶコンテンツを盗まれる危険がある。これを防ぐため、セキュアハード内で毎回セキュアソフトを再構成する。再構成された毎回違うプログラムコードでセキュアソフトを実現し、さらに毎回（例えば１時間ごと）違う手順で認証することで、クラッカーに解析に必要な時間を与えないようにする。これにより、従来の難読化主体のソフトウェアと比較して、大幅に安全度の向上したソフトウェアを提供することを考える。セキュアハード上のファームは、ソフトを半ば自動的に再構成し、毎回違う認証手順を半ば自動発生させる処理を行う。ここが、研究開発のキーポイントとなると考える。

また、HDD 内に暗号化された状態で蓄積されているデジタル AV コンテンツは、セキュアハードで復号され、さらにスクランブルされて PC メモリ上のセキュアソフトに転送される。PC メモリ上のセキュアソフト（MPEG ビデオデコーダなど）は、デジタル AV コンテンツのデスクランブルや伸張処理などを行う。

１４年度は、上記のようにして図４－２に示す基本アーキをベースにして実際にセキュアソフト、セキュアハード等を試作し、検証を行う。この目標レベルは、最初に記載したとおり、基本機能レベルで完成を目指す。

１４年度で研究開発を要する項目は、図４－２で示した全体の基本アーキ、セキュアハードの（a）ハード機能、（b）ファーム機能、ＰＣメインメモリ上の（c）セキュアソフト、（d）その他のセキュア関連ソフトなどである。また、上で挙げた項目以外にも「基本デジタル放送受信機能」を実現するために必要な付属的機能は存在する。例えば GUI（Graphical User Interface）による簡単な全体制御機構が必要と考えられるが、それらも必要に応じ開発する。

実際の研究開発は、以下のように行う予定である。まず、全体基本アーキを前述のように想定し、想定に沿ってセキュアハード、ＰＣメインメモリ上のソフトなどセキュアシステムを実験的に構築する。最初に構築されたシステムは、必ずしも完全なものでなく、一部機能のみ持つ暫定的なものでも良いこととし、順次機能を追加していくことにより完成度を高める。つまり、上述の要素の中の幾つかのみが実装されている状態であり、各要素が未完成であっても、その要素に関してある程度の評価が可能であれば良いと考える。例えばセキュアソフトの中のＭＰＥＧビデオデコーダは、当初リアルタイムで動作しなくても、

高速化のためのチューニングなど実験可能なレベルであれば良いとする。このようにして出来るだけ、各要素を個別に評価実験出来るような体制にして研究開発を進める予定としている。評価・検証実験の過程で問題が発覚した場合は、基本的には要素ごとに改良研究を行う。また、基本アーキテクチャに関連した問題が発覚した場合は、セキュアハード等の構成まで戻って再検討を行う必要があるが、セキュアハード回路はＦＰＧＡ構成になっているため迅速な設計変更が可能である。より具体的には、以下のような手順で開発を進める。

１）まず、セキュアハードの暫定版を開発し、セキュアハードの機能が確認できるレベルのファームやソフトを立ち上げる。

２）その後、ソフトやファームの最低限の機能が評価できるようなレベルで立ち上げる。セキュアハード上のファーム（セキュアソフト再構成機能など）とＰＣメインメモリ上のソフト（ＭＰＥＧビデオデコード機能など）は、必ずしも同時に立ち上げる必要はなく個別に立ち上げることも可能である。すなわち、セキュアソフト再構成機能は、ＭＰＥＧビデオデコーダが動作していなくとも対象となる単純な評価用ソフトがあれば、再構成機能の評価実験は可能と考えられる。また、セキュアソフト（ＭＰＥＧビデオデコーダなど）は、セキュアハードがなくともある程度の評価実験は可能であるし、セキュアハードの機能の一部を使用しながら検証を行うことが可能である。

３）最終的に、各要素を組み合わせて、機能レベル（具体的な１４年度のリアルタイム性目標は毎秒約５フレーム）で「基本デジタル放送受信機能」を実現させる。

上記のように本研究開発は、図４－２の（ｉ）ＨＤＤ，（ii）セキュアハード、（iii）セキュアソフトの３基本要素をベースに如何にセキュアなシステム（セキュアＭＰＥＧビデオデコードソフトなどを含む基本デジタル放送受信システム）を構築するかが一つの課題となる。

一方で研究開発要素ごとの研究開発も必要である。

１）セキュアハードは、地上デジタル放送関連規格にどう対応するか検討する必要がある。さらに Multi2 復号、ローカル暗号／復号回路など、平成１５年度の LSI 化を前提に開発や機能レベルの実証を行う。

２）また、セキュアハード内のファームはセキュアソフトを再構成する機能やセキュアソフトの保護を行う機能を持ち、本権利保護システムの大きなポイントと位置付けられる。セキュアソフトにリアルタイム認証アルゴリズムを挿入するなど、「セキュアソフト再構成手法」を検討する予定である。ここでのポイントは、（１）「成りすまし」攻撃を防ぐような解析困難なレベルの再構成の実現、（２）セキュアハード内のメモリなど限られた資源の中での再構成機能の実現だと考えられる。セキュアソフトの保護や、リアルタイム認証の具体

手順など詳細検討を行っていく。詳細検討において、或いは評価実験の過程でセキュリティーホールなどの検討を行い、より安全な手法を追求する。そのため、前述のように可能な限り研究要素ごとに独立に研究開発できる体制を敷いて検討を進める。

３）セキュアソフトなどＰＣメインメモリ上の関連ソフトに関しても、以下のような検討が必要である。（１）地上デジタル放送対応高速ＭＰＥＧビデオデコーダ、（２）オーディオデコーダ、（３）リアルタイム認証などのセキュア機能、（４）セキュアソフト専用ローダ（５）セキュアハードドライバーである。また、伸張されたビデオ映像、オーディオの同期処理を含めた統合システムとユーザーが操作するための GUI（Graphical User Interface）が必要である。対象となるＯＳについては、１４年度も引き続き検討するが、目標はできる限りＯＳに依存しないシステムを考える。ＯＳを調査し、図４－２の専用ローダやセキュアハードドライバーなどでできる限りＯＳ依存部を全面的にカバーし、ソフト本体はＯＳに依存しないような手法を考える。ローダ、ドライバに関しては対象ＯＳを Microsoft Windows として検討を進める。ＭＰＥＧデコーダは、セキュアハードで再構成した上での、処理速度の向上がポイントとなる。今後の検討に依存するが、まずＭＰＥＧビデオデコードの機能的な動作を確認し、その後、再構成したことによる影響などを検討し、最後に処理性能向上を図ることを目標とする。１４年度は、５フレーム／秒の処理速度が目標であり、本格的な高速化は次年度以降とする予定である。再構成によりなるべく処理性能に影響が出ないようにすることが研究開発の一つのポイントとなると考えられる。オーディオデコードに関しても、一定の開発は必要だが、MPEG ビデオデコードに準じた開発が可能であると判断する。

４-１-２　研究開発課題実施計画

４-１-２-１　平成１３年度　　（金額は非公表）

研究開発項目	第１四半期	第２四半期	第３四半期	第４四半期	計	備　考
アーキ検討、実験ボード開発				→		
間接経費						
合　　計						

単位：百万円

注）１　経費は研究開発項目毎に消費税を含めた額で計上。また、間接経費は直接経費の３０％を上限として計上（消費税を含む。）。
（合計の計は、「３－１の研究開発課題必要概算経費」の総額と一致）
２　備考欄に再委託先機関名を記載。

４-１-２-２　平成１４年度　　（金額は非公表）

研究開発項目	第１四半期	第２四半期	第３四半期	第４四半期	計	備　考
１）セキュアハードの研究開発	→	→	→	→		
２）セキュアハード上のファームウエアの研究開発	→	→	→	→		
３）ＰＣメインメモリ上のソフトの研究開発	→	→	→	→		
間接経費						
合　　計						

単位：百万円

注）１　経費は研究開発項目毎に消費税を含めた額で計上。また、間接経費は直接経費の３０％を上限として計上（消費税を含む。）。
（合計の計は、「３－１の研究開発課題必要概算経費」の総額と一致）
２　備考欄に再委託先機関名を記載。

４－２　研究開発の実施内容

４-２-１　平成１３年度

初年度の１３年度は、ＰＣ型のセキュアデジタル放送受信機のセキュアハード／ソフトの概略構成を検討し、大まかに権利保護システムの基本アーキテクチャを固める。またこの大まかな構想に従い、ＦＰＧＡ（Field Programmable Gate Array）を使用した構成でセキュアハード実験ボードの基本部分を製作する。実験ボードを１４年度以降のセキュアデジタル放送受信機の性能や機能評価などを行うための実験プラットフォームという位置付けで製作し、１４年度以降の実験などで改良点や問題が発覚した場合、その時点でハードウェアの設計を容易に変更出来るようにＦＰＧＡ（Field Programmable Gate Array）構成とする。

なお、この実験ボードは、各種改良を行ったのち、本研究開発の最終年度にはＬＳＩ化し、ＰＣカード搭載を目標とし、本研究開発が目標とする権利保護システムの「核」になる部分とする。

PC 型のソフト処理主体のセキュアデジタル放送受信機は、セキュリティーという側面から見ると図４－３で示すブロック図のような構成が考えられる。本図に従い、平成１３年度はセキュアデジタル放送受信機の基本アーキの決定など今後の方向付けを行った。

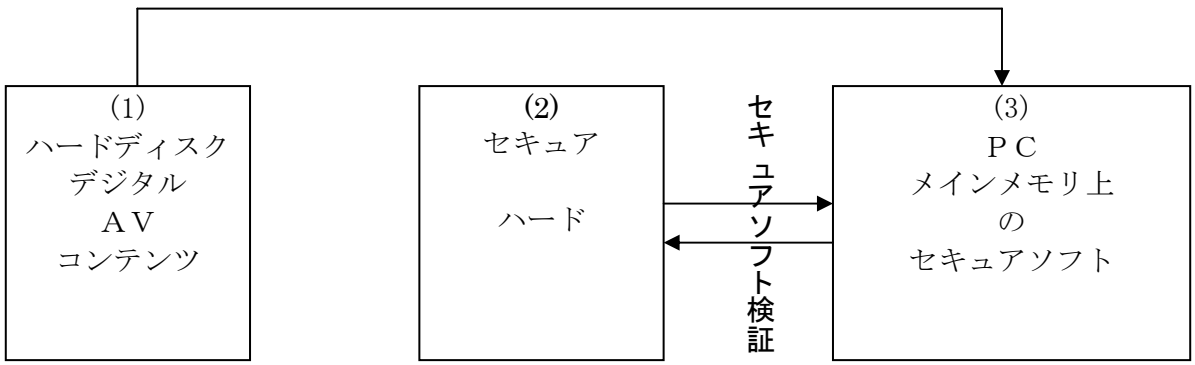


図４－３　想定されるセキュアソフトとハードの関係

図４－３ではデジタル放送受信機は、（１）ハードディスク、（２）セキュアハード、（３）ＰＣメインメモリ上のセキュアソフトという基本要素から構成される。ハードディスク内に放送されたデジタルＡＶコンテンツが蓄積されており、これがＰＣメインメモリ上のセキュアソフトでデコード処理され、視聴される。一方でセキュアソフトの安全性（デコード処理後のデジタルＡＶコンテンツの著作権など権利保護機能）をセキュアハードが保証する。以上の基本構成を前提に１３初年度研究課題として以下を検討した。

１）最終的な信頼点としてのセキュアハードの活用法を検討した。セキュア

ハードには固有のマスター鍵を持たせておき、デジタルＡＶコンテンツの復号にはこのマスター鍵が必要な構成とする。マスター鍵は外部から参照できない情報としてセキュアハード内部でしか使用しない。これによってセキュアハードがない状態でのデジタルＡＶコンテンツを不可能にすることができる。また、セキュアハードを経由したセキュアソフト（デコードプログラム）の実行や、セキュアハードが主体となった認証や改竄検出を行うことにより、プログラムの安全性をセキュアハードが保障する方法が実現可能であることが分かった。

２）リアルタイム性とセキュリティーのトレードオフを検討した。システム全体にリアルタイム性が要求されるが、処理コンテンツ（デジタルＡＶコンテンツ）は、基本的にハイビジョン動画像（ＭＰＥＧ　ＭＰ＠ＨＬレベル）であり、ＰＣ性能をギリギリまで使用しないとリアルタイムソフト処理は困難と考えられる。この中にセキュア処理をＰＣから見てあまり負担のかからない範囲で盛り込む方法を検討した。処理負担の少なくセキュリティー強度の高いデジタルＡＶコンテンツの復号化手法の必要性がわかった。また、セキュアハードとＰＣの間でのデータ転送の効率化を図る必要性があることが分かった。これらのことを１４年度のセキュアハード設計に生かしていくと同時に、リアルタイム性実現のための検討は続けていく必要がある。

３）さらに別の視点としてのコスト面での検討を行った。本技術を２００４年以降店頭販売の一般ＰＣで活用されることを前提とした場合、コスト削減のためシステム内の特殊部品はセキュアハードのみとすることが望ましい。セキュアハードにはシステムから抽出したセキュア機能のみを搭載する。またセキュアハード普及を容易にするためセキュアハードはＰＣカード搭載とし、ＰＣ接続時のインタフェースはＰＣＩバスなどユーザーに公開された汎用なバスを採用する。

以上のようなことを検討し、セキュアハード／ソフトの概略構成として、大まかなシステム構想を固めた。

また本構想に従い、初年度はＦＰＧＡ（Field Programmable Gate Array）構成でセキュアハード実験ボードを製作した。この実験ボードの最終目標はこれをＬＳＩ化し、図４－３のセキュアハードとすることにある。実験ボードの基本部分をＦＰＧＡ構成とすることで１４年度以降のセキュア化の研究開発の結果を柔軟に取り入れてセキュアハードの設計変更を容易にすることができる。従って、このボードはセキュリティーシステム試作／実証のための基本プラットフォームとなるよう構成する。この実験ボードにより次年度以降、上記で検討したセキュアハードとセキュアソフト間で安全性検証のための特別な通信などを具体化し、検証実験を行う。

１３年度はセキュアシステム構想を固めることが主眼であり、セキュアハード実験ボードの中のＦＰＧＡの具体設計は、次年度以降とするがセキュアハードにはＦＰＧＡ以外にプロセッサ、プロセッサ周辺回路、メモリ、ＰＣＩインタフェース回路など必要な回路を搭載する。これらのプロセッサおよび周辺回路の動作確認などを行った。

平成１３年度の成果を踏まえ、基本機能レベル（基本デジタル放送受信機能）で完成させることを平成１４年度の目標とした。具体的には、（１）セキュア

ハードの回路、（２）セキュアソフトを機能レベルで完成させ、実際にＰＣ上でデジタルテレビ画像を見られるようにすることを目標とした。

4-2-2　平成１４年度

平成１３年度の成果を踏まえ、基本機能レベル（基本デジタル放送受信機能）で完成させることを平成１４年度の目標とした。具体的には、（１）セキュアハードの回路、（２）セキュアソフトを機能レベルで完成させ、実際にＰＣ上でデジタルテレビ画像を見られるようにすることを目標とした。

但し、セキュアハードはＦＰＧＡ（Field Programmable Gate Array）搭載実験ボードで試作されるため、物理的なサイズが大きく、ＰＣカード等に実装できるレベルでなく、また高価なものとなっている。従ってサイズ的にもコスト的にも製品展開にはさらに開発が必要なレベルに留まる。製品展開が期待できるレベルまで進めるには、ＦＰＧＡ搭載実験ボードのＬＳＩ化が必要と考えられ、これは最終年度の目標となる。またセキュアソフトなどのリアルタイム性も平成１４年度の段階ではさらに開発が必要なレベルである。具体的にはリアルタイム性として毎秒約５フレーム程度の動画再生の達成を目標とした。最終年度での目標は一般視聴者が違和感なく動画像を視聴出来るレベルを目標とする。

上記の計画を以下のように実施した。平成１３年度は図４－３の（１）ＨＤＤ，（２）セキュアハード、（３）セキュアソフトの３つの基本要素をベースにセキュアシステムの基本アーキを決定した。平成１４年度は、上記基本アーキをベースに研究要素を特定し、それぞれの要素ごと検討した。

具体的には、平成１４年度は、図４－３の構成を具体的にセキュアシステムとして展開していく方法を検討し、全体アーキを図４－４で示すようなシステムとした。以下、図４－４の概要を説明する。

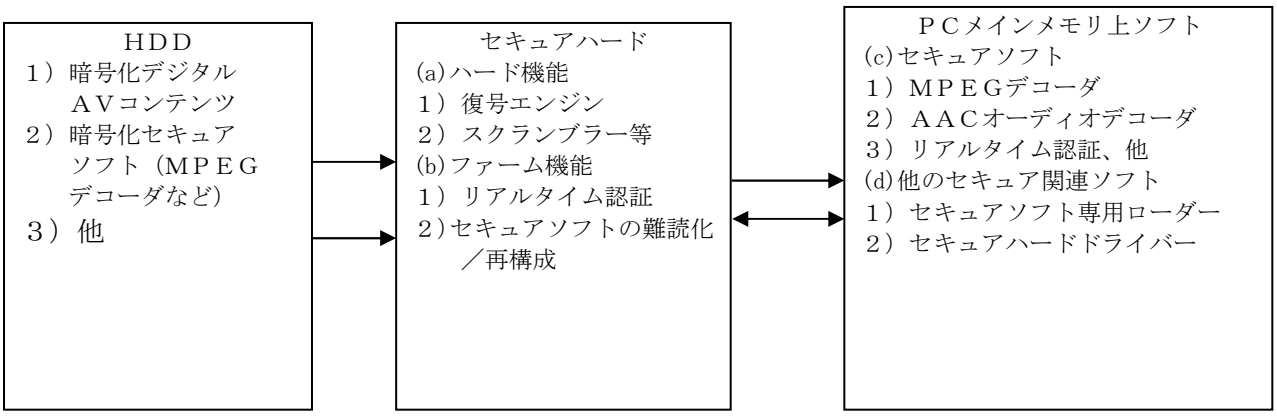


図４－４　１４年度研究開発したセキュアソフトとハードの概要

Ｄ上にある暗号化デジタルＡＶコンテンツは複製されることを防ぐために暗号化される。また、セキュアソフト等のプログラムも解析されることを防ぐた

めに暗号化された状態になっている。デジタルＡＶコンテンツの再生は、次の手順で行う。暗号化セキュアソフトを一旦セキュアハードに読込み復号化する。復号化されたセキュアソフトに対して、セキュアハード内でソフト再構成などのセキュア化処理を加えた上でＰＣメインメモリ上に転送する。転送領域はセキュアソフト専用ローダが確保し、転送はセキュアハードが行う。

ロードされたセキュアソフトは、セキュアハードがリアルタイムに認証する。常に同じ手順で認証したり、セキュアソフトが常に同じプログラムコードになっていたりするとＰＣ上のセキュアソフトや認証アルゴリズムを解析され、成りすましにより処理途中でデジタルＡＶコンテンツを盗まれる危険がある。これを防ぐため、セキュアハード内で毎回セキュアソフトを再構成する。再構成された毎回違うプログラムコードでセキュアソフトを実現し、さらに毎回（例えば１時間ごと）違う手順で認証することで、クラッカーに解析に必要な時間を与えないようにする。これにより、従来の難読化主体のソフトウェアと比較して、大幅に安全度の向上したソフトウェアを提供する。セキュアハード上のファームは、ソフトを半ば自動的に再構成し、毎回違う認証手順を半ば自動発生させる処理を行う。また、セキュアファームはセキュアソフトを常時監視し、セキュアソフトの改竄を検出するスキャン機能を持つ。

HDD 内に暗号化された状態で蓄積されているデジタル AV コンテンツは、セキュアハードで復号され、さらにローカルスクランブルされて PC メモリ上のセキュアソフトに転送される。PC メモリ上のセキュアソフト（MPEG ビデオデコーダなど）は、デジタル AV コンテンツのローカルデスクランブルや伸張処理などを行う。

上記の検討アーキテクチャをベースに実際に（セキュアソフト、セキュアハード）を試作し、機能検証実験を行いました。１４年度は、全体の基本アーキテクチャの検討、図４－４に示す、セキュアハードの（a）ハード機能、（b）ファーム機能、ＰＣメインメモリ上の（c）セキュアソフト、（d）その他の関連ソフトなどの開発を行った。

また、上で挙げた項目以外にも「基本デジタル放送受信機能」を実現するために必要な付属的機能、例えば GUI（Graphical User Interface）による簡単な全体制御機構も開発した。

動画再生能力はＤＶＤ動画像（ＭＰＥＧ　ＭＰ@ＭＬレベル）では、ほぼフレームレートの３０フレーム／秒を達成した。また、ハイビジョン動画像（ＭＰＥＧ　ＭＰ@ＨＬレベル）では当初目標を上回る毎秒１０フレーム程度の動画再生を達成することができた。最終年度では、一般視聴者が違和感なく動画像を視聴できるレベルを目標とする。

５　研究開発実施状況

５－１　セキュアハードの研究開発

５－１－１　序

ＡＶコンテンツやＰＣ上で実行されるソフトウェアのセキュア化を実現する上で、外部から「改ざん」や「覗き見」を行なうことができない耐タンパモジュールは、必要不可欠である。本システムにおいては、この耐タンパモジュールを「セキュアハード」と呼ぶ。本システムの製品化を考えた場合、セキュアハードは、耐タンパ性を備えたＬＳＩによって実現することを想定しているが、本研究開発においては、ハードウェアに搭載すべき機能やシステムとしてのパフォーマンスの検討を行なうことが目的であるため、機能の追加・修正が容易なＦＰＧＡ（Ｆｉｅｌｄ　Ｐｒｏｇｒａｍｍａｂｌｅ　Ｇａｔｅ　Ａｒｒａｙ）で実現している。

セキュアハードは耐タンパ性を備えており、その内部で実行する機能に関しては、「改ざん」や「覗き見」ができないため、安全性の高いものであるが、システムにおいてセキュアハードで実現しなければならない機能が拡大するにつれて、ハードウェアに要求される性能が高くなり、コストの増大につながる。本研究開発においては、低コストでデジタル放送受信機の権利保護を実現することが重要なテーマの一つとなっているため、ハードウェアに搭載する機能を最小限に抑えることが重要となる。本システムにおいて、セキュアハードで実現している機能としては、以下の項目が挙げられる。

(1)　デジタルＡＶコンテンツの暗号化・復号化機能
(2)　PC 上で実行されるソフトウェアを保護するための機能
(3)　システムのパフォーマンス向上のために必要な機能（ＤＭＡなど）

以下では、本プロジェクトにおいて開発したセキュアハードの構成・機能について解説する。セキュアハードの解説では、主にハードウェアとしての構成・機能に焦点をあて、(1)、(3)の機能を中心に解説する。(2)のＰＣ上ソフトウェアのセキュア化機能については、主として、セキュアハードに搭載されたコントロールプロセッサによって実現されている。セキュア化機能方式等の詳細については、次章以降に解説する。

５－１－２　セキュアハードモジュール構成

本プロジェクトに使用する試作セキュアハードモジュールの構成図を以下に示す。セキュアハードモジュールには、ＦＰＧＡおよび制御用プロセッサが搭載されており、暗号・復号処理などはＦＰＧＡ上に論理展開される

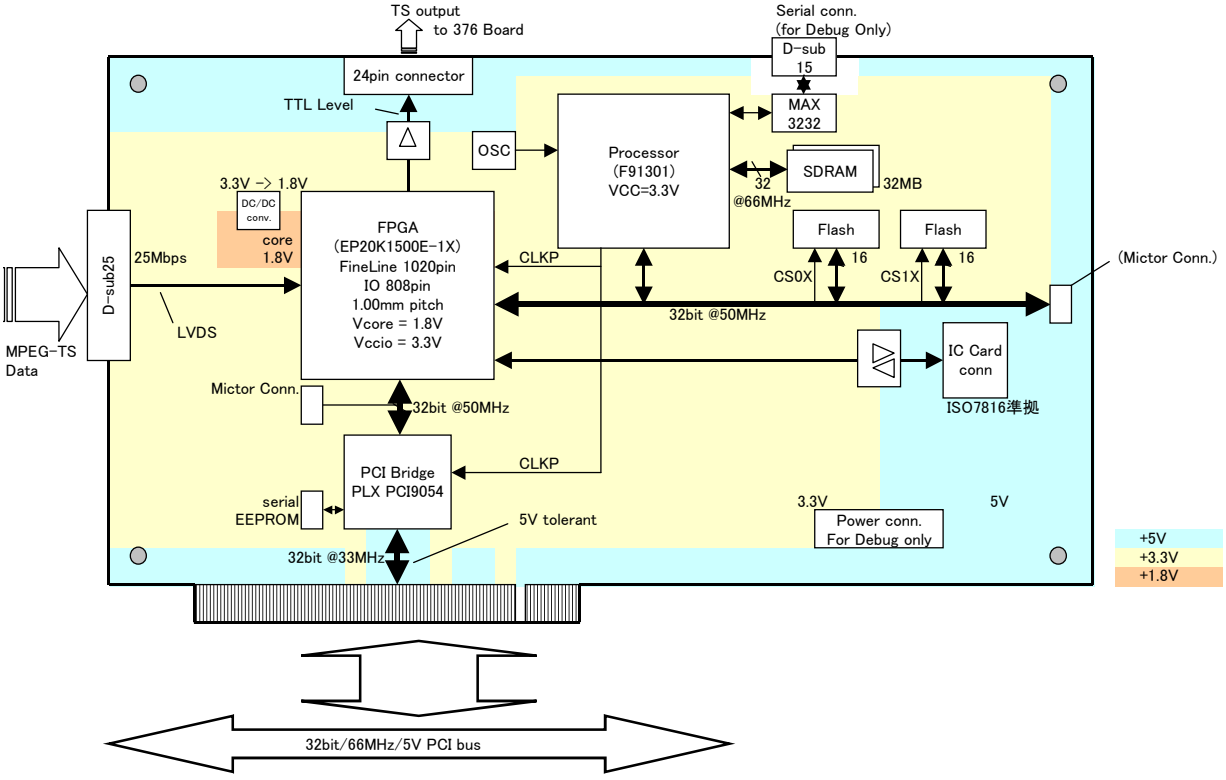


図　5-1　セキュアハードモジュール(試作ボード)構成

- FPGA　APEX20KE1500 (EP20K1500EBC652-1X)
- Processor　FR91301/FR91V301(Fujitsu)
- Flash　16Mbit FlashROM (1M x16bit) x2pcs (TSOP(I) socket 装着)
- Memory　32MB SDRAM (4bank x2M x16bit TSOP(II) x2pcs) (32bit width)
- PCI bridge　PCI9054 (PLX)
- Serial EEPROM　2Kbit (or4Kbit) Microwire devices
- MPEG I/F　MPEG Input 25pinD-sub x1
  MPEG Output　24pin connector x1
- IC Card Slot　ISO7816-1 準拠 IC Card Slot x1
- Serial IF　D-sub9 x1ch (for Debug)
- Level converter
- DC-DC Converter　MIC29302BU　(MICREL)
- Logic Analyzer pod (for Debug)　AMP Mictor connector

５－１－３　セキュアハードモジュール機能

コンテンツの暗号・復号処理などのセキュア機能は、セキュアハードモジュールに搭載されたＦＰＧＡに論理展開される。以下では、ＦＰＧＡに論理展開される機能について解説する。

５－１－３－１機能概要と特徴

本 FPGA は、セキュアハードに入力された MPEG TS パケットデータをコンテンツの権利を保障した状態で自由に視聴し、ハードディスクへの保存等も行う事を実現するために TS パケットデータの暗号化/復号化や Routing、PC との DMA マスタ転送を行う。

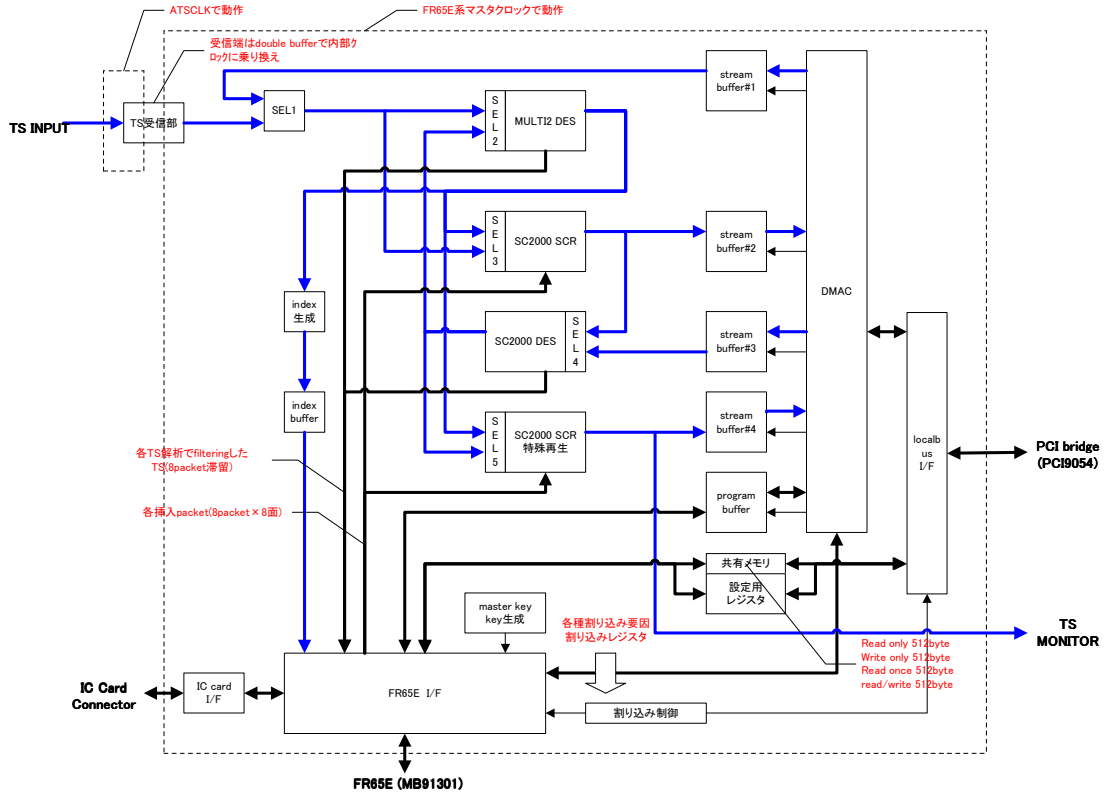


図　5-2　ＦＰＧＡ機能ブロック構成

●Multi2 Descrambler
　TSパケットデータをMulti2方式で復号化処理する。

●Local Scrambler
　TSパケットデータをSC2000方式で暗号化処理する。

●Local Descrambler
　TSパケットデータをSC2000方式で復号化処理する。

●Secure Scrambler
TSパケットデータをSC2000方式で暗号化処理する。

●ストリーム入力
メインメモリからMULTI2 Descrambler部にストリームを転送する。(DMA)

●ストリーム録画
Local Scrambler部からメインメモリにストリームを転送する。(DMA)

●録画ストリーム再生
メインメモリからLocal Scrambler部にストリームを転送する。(DMA)

●セキュアデコーダ入力
Secure Scrambler部からメインメモリにストリームを転送する。(DMA)

●Program転送
メインメモリ⇔FR65E間でProgramデータを転送する。(メインメモリ⇔Program Buffer間はDMA)

●共有メモリアクセス
PCのセキュアソフトとFR65E間の通信、及び秘密の番号通信で使用する乱数データ通信などに使用し、基本的にMaster転送より優先してアクセスする。

●PCI9054⇔FR65E間のレジスタアクセス
PCのセキュアソフトとFR65E間の通信を行うにあたってコマンド番号やアクノリッジの通信に使用する。

●FR65インタフェース
PCからの設定を含めて、本FPGAに対するすべての設定は直接的にはFR65Eから行われる。
ここでは、設定/Status通知に関する各機能ブロックとのインタフェースを行う。

●PCI9054インタフェース
PC側とのインタフェースをPLX社のPCI9054(PCI Bridge)経由で行う。通常はFPGAがBUS Masterとなって DMA転送を行い、PCからターゲットアクセス要求があった場合は、マスタ転送の切れ目でLocal BUSを開放し、ターゲットアクセスを優先させて実行する。

本項で使用するアルファベットによる略称を以下に挙げる。

略称		内容
EXTIN	(External Input)	ストリーム入力データ転送用 DMA パス
RECOUT	(Record Output)	ストリーム録画データ転送用 DMA パス

HDDIN	(HDD Input)	録画ストリーム再生データ転送用 DMA パス
PLAYOUT	(Play Output)	セキュアデコーダ入力データ転送用 DMA パス。
PRG	(Program Input / Output)	プログラムデータ転送用 DMA パス。

５－１－３－２　各ブロック機能概要

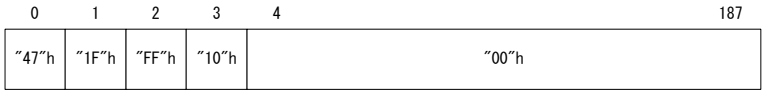
● TS パケット変換処理

TS パケット変換処理部では TS パケット廃棄処理、NULL パケット変換処理、PID 変換処理を実施する。未定義な TS パケットの処理は、廃棄 or NULL パケット変換の選択が可能となっている。(※初期状態では、廃棄設定となっている。)

補足) 表における定義パケットとは CPU 設定された 16 種類の PID を指す。

[廃棄処理]
対象となる TS パケットのスタートパルス IFP,データイネーブル IEN,
データ IDT[7:0]を全て ALL”0”に変換する。

[NULL パケット変換]
対象となる TS パケットのヘッダ情報を “471FFF10”h に、
残りの領域は ALL”0”に変換する。



[PID 変換処理]
対象となる TS パケットのヘッダ情報の PID を CPU 設定値に変換する。

[TS パケット変換処理対応表]

種類	a	b	c	d	TS パケット処理
未定義パケット				0	廃棄
				1	スルー
定義パケット	1				廃棄
	0	1			NULL パケット変換
	0	0	1		PID 変換
	0	0	0		スルー

a) Dell Enable
b) Null Enable
c) PID Convert Enable
d) Undefined Packet Control

● TS パケット挿入処理

TS パケット挿入処理部は、SC2000 暗号化処理部の後段に位置し、挿入要求に応じて、指定されたスロットに格納された TS パケットデータの挿入を行い、挿入が完了した時点で、挿入完了通知する。(1 回の挿入要求に対し、1 個の TS パケットを挿入する。)

挿入用 TS パケットバッファへの書き込みは、CPU から行い、アドレスを直接指定する事で、書き込み可能となる。

設定可能なスロット数は合計 8 個存在するが、一度に挿入要求を出せるスロットは 1 個だけであり、複数挿入したい場合は挿入完了通知を待ってから、再度挿入要求をだす必要がある。

主に、鍵情報パケットの送出を目的に使用する。

[TS パケット挿入用 RAM の構成]

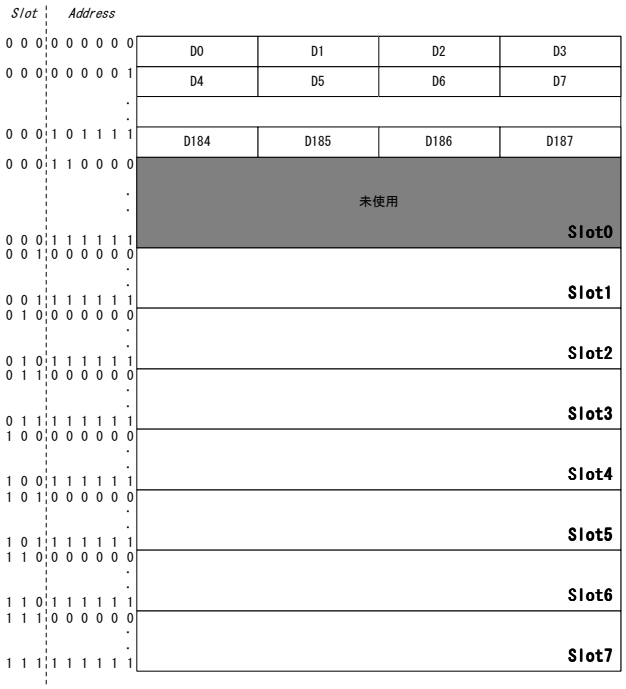


図 5-3 ＴＳパケット挿入用ＲＡＭの構成

● MULTI2 復号化処理

MULTI2 方式で暗号化された TS パケットの情報の復号処理を行う。(MULTI2 方式の復号アルゴリズムについては、限定受信実験方式提案書 A 方式を参照)

MULTI2 暗号によるデータの復号化には、256bit のシステム鍵と 64bit のデータ鍵が必要である。データ鍵は各 PID 毎に 2 種類(ODD 鍵、EVEN 鍵)が存在し、TS パケットのヘッダ情報(PID,TSC)より選択される。

復号化演算は TS パケットのペイロード領域にのみ実施し、ヘッダ情報、アダプテーション領域は、スルー出力される。

[MULTI2 復号化処理対応表]

種類	a	b	c	d	ペイロード処理
未定義パケット					スルー
定義パケット	0				スルー
	1	0			スルー
	1	1	0		スルー
	1	1	1	0	デスクランブル(EVEN 鍵)
	1	1	1	1	デスクランブル(ODD 鍵)

a) TSP Scramble Control
b) KEY Enable
c) TSC[1]
d) TSC[0]

[回路構成]

-システム鍵と PID,TSC 値から選択されたデータ鍵を使用して、ワーク鍵を生成。
-ワーク鍵は復号処理器へ送られる。
-復号処理は 64bit 単位で実施する為、8bit のデータを 64bit にシリアル/パラレル変換。
-復号処理は、通常 CBC で復号処理を行うが、アダプテーション領域の長さによっては 64bit に満たない長さになる場合がある。この時は OFB で復号処理を行う。
-復号処理後の 64bit データを 8bit にパラレル/シリアル変換
-通常データと復号処理データをマージする。

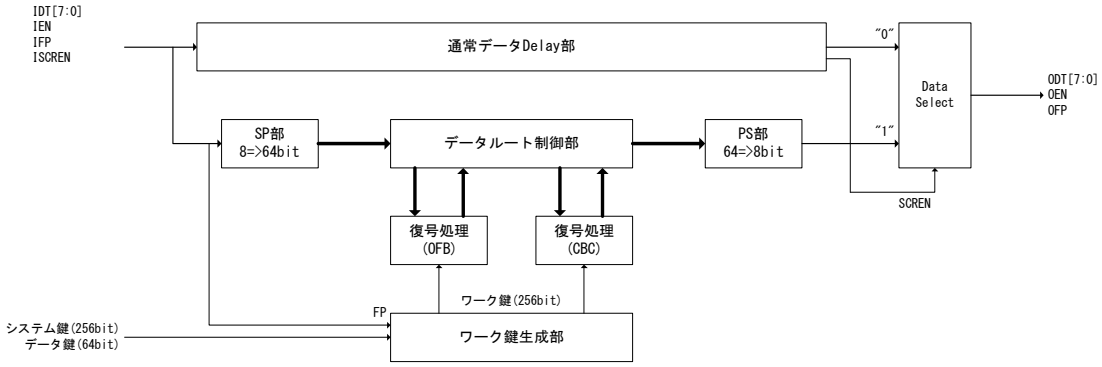


図 5-4 ＭＵＬＴＩ２復号回路ブロック図

[MULTI2 復号化手順]

ハードとしての処理手順は後述する SC2000 復号化手順と同じである。

● SC2000 暗号化処理

SC2000 方式で指定した TS パケットの暗号化処理を行う。暗号化処理をするかどうかは、CPU 設定値と入力 TS パケットのヘッダ情報(PID)を見て決定する。暗号化処理は、TS パケットのペイロード領域のみに実施し、ヘッダ情報、アダプテーション領域は、スルー出力する。

暗号用の鍵情報はデータ鍵(128bit)を使用する。データ鍵は各 PID 毎に 2 種類(ODD/EVEN) 存在し、CPU 設定値を見て、どちらを選択するかを決定する。

また、復号処理側では TSC コードから暗号化情報を認識する為、TSC コードの変換処理を行う。

[SC2000 暗号化処理対応表]

種類	a	b	c	d	TSC 処理	ペイロード処理
未定義パケット					スルー	スルー
定義パケット	1				スルー	スルー
	0	0			“00”	スルー
	0	1	0		“00”	スルー
	0	1	1	0	“10”	スクランブル(EVEN 鍵)
	0	1	1	1	“11”	スクランブル(ODD 鍵)

a) TSP Through
b) TSP Scramble Control
c) KEY Enable
d) Scramble Key Select

［SC2000 暗号処理手順］

１TS パケット暗号化処理

TS パケットの暗号化処理はハードのみで行い、ファームは介在しない。

１－１TS パケットから PID を取り出し、設定された even/odd を元に、鍵バッファから使用する鍵を取り出し、暗号化回路に設定する。

１－２使用する鍵(even/odd)に従って、TS ヘッダを書き換える。

１－３TS パケットを暗号化回路へ入力、及び暗号化回路からの取り出しを行う。

２スクランブル暗号鍵の切り替え処理

使用する暗号の even/odd を切り替えるタイミングはファームが決定し、鍵パケットの挿入、鍵バッファの設定、切り替え処理を行う。

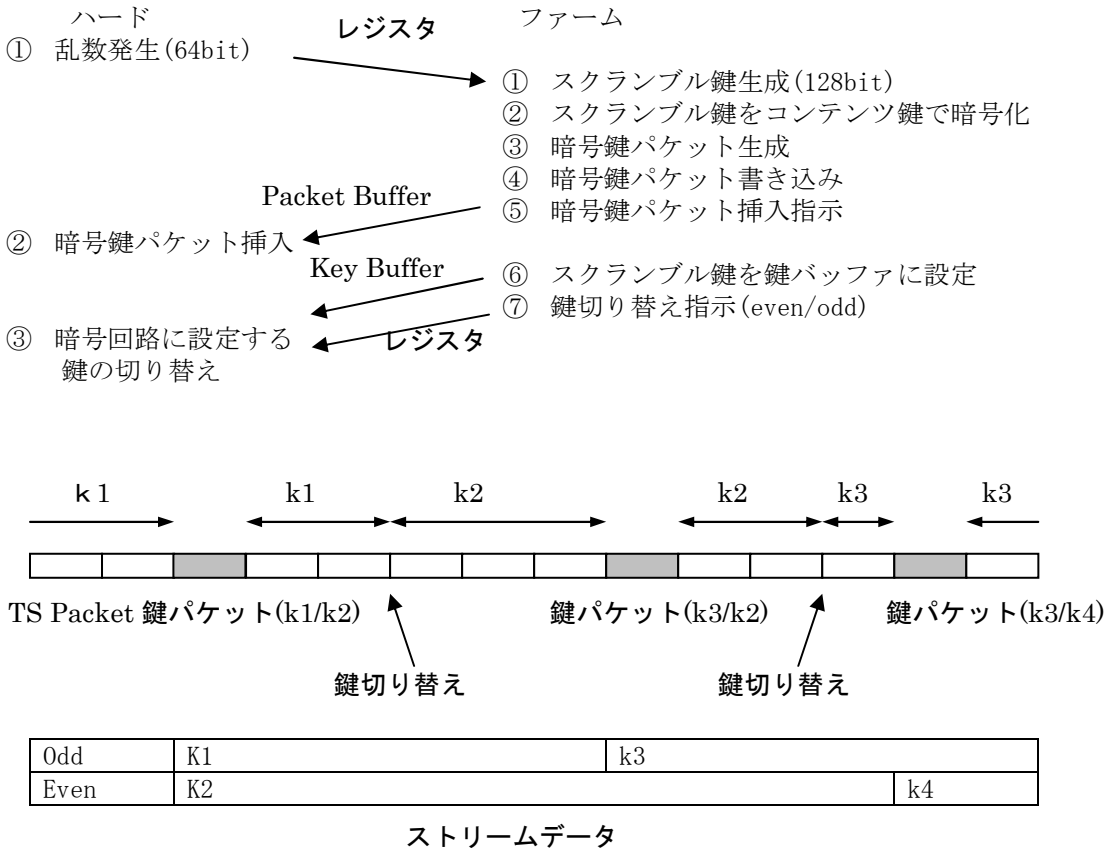


図 5-5 スクランブル鍵切り替えタイミング

● SC2000 復号化処理

SC2000 方式で暗号化された TS パケットの情報の復号化処理を行う。復号処理をするかどうかは、CPU 設定値と入力 TS パケットのヘッダ情報(PID,TSC)を見て決定する。
復号処理は、TS パケットのペイロード領域のみに実施し、ヘッダ情報、アダプテーション領域は、スルー出力する。

復号用の鍵情報はデータ鍵(128bit)を使用する。データ鍵は各 PID 毎に 2 種類(ODD/EVEN) 存在し、TS パケットのヘッダ情報(PID,TSC)より選択される。

[SC2000 復号化処理対応表]

種類	a	b	c	d	ペイロード処理
未定義パケット					スルー
定義パケット	0				スルー
	1	0			スルー
	1	1	0		スルー
	1	1	1	0	デスクランブル(EVEN 鍵)
	1	1	1	1	デスクランブル(ODD 鍵)

a) TSP Scramble Control
b) KEY Enable
c) TSC[1]
d) TSC[0]

[SC2000 復号化処理対応表(強制鍵選択モード)]
※Force Scramble Enable = “1”

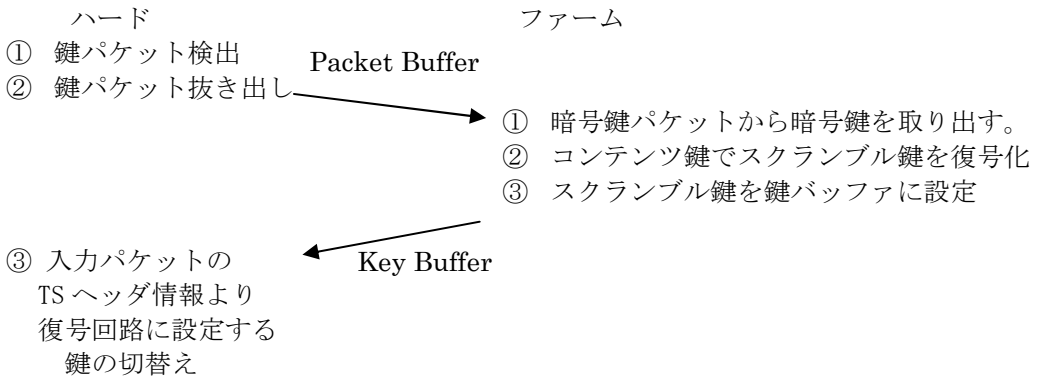
種類	a	b	c	ペイロード処理
未定義パケット				スルー
定義パケット	0			スルー
	1	0		スルー
	1	1	0	デスクランブル(EVEN 鍵)
	1	1	1	デスクランブル(ODD 鍵)

a) TSP Scramble Control
b) KEY Enable
c) Force Scramble Key Select

[SC2000 復号化手順]

１．TS パケット復号化処理
　TS パケットの暗号化処理はハードのみで行い、ファームは介在しない。
２－１TS パケットから PID, even/odd を取り出し、使用する鍵を決定する。
２－２鍵バッファから鍵を取り出し、復号化回路に設定する。
２－３TS パケットを復号化回路へ入力、及び復号化回路からの取り出しを行う。

２．スクランブル復号鍵の切り替え処理
ストリームに復号鍵パケットが含まれていた場合、ファームはパケットからスクランブル鍵を取り出し、鍵バッファに設定する。実際に鍵が切り替わるタイミングはストリームからハードが決定する。(TS ヘッダで決定)



● Master 転送

Master 転送では FPGA が Master となり、PCI バスを通して、主としてメインメモリへの Read/Write アクセスを行う際の機能を行う。Master 動作を行うアクセス要因を以下に示す。

NO	アクセス要因	内容
1	EXTIN 転送	ストリーム入力データ転送用 Read アクセス
2	RECOUT 転送	ストリーム録画データ転送用 Write アクセス
3	HDDIN 転送	録画ストリーム再生用データ転送用 Read アクセス
4	PLAYOUT 転送	セキュアデコーダ入力データ転送用 Write アクセス
5	PRG 転送	プログラムデータ転送用 Write/Read アクセス
6	転送テーブルアドレス転送	上記 1..5 転送テーブルアドレスデータ転送用 Read アクセス

これらの Master 転送は PCI bridge(PCI9054)の Master 動作を使用して行う。ただし、各転送における入力ビットレートなどに応じて、効率良くデータ転送を行うため、周辺回路で上記各 Master 転送要因の優先順位を制御する。

動作概要

(1) Master 動作シーケンス

・Master 転送は各アプリケーションの実行中しか動作せず、基本的には 8DW バーストのトランザクションに分けられて処理される。

・Master 転送の開始は LocalFIFO 内データ量の状態で制御される。

(2) Master 転送優先順位

・RECOUT ＞PLAYOUT ＞EXTIN ＞HDDIN ＞ PRG の順

・各 Master 転送要因に関して、転送アドレステーブル(※)の読み出しが必要な場合は、各 Master 転送要因に先立って、転送アドレステーブルの読み出しが行われる。

※ 転送アドレステーブルには、アクセスすべきメインメモリの物理アドレスがページ(4Kbyte)単位で登録されている。各 Master 転送要因について、ページ毎に転送アドレステーブルの読み出しが必要となる。転送アドレステーブルの詳細については後述する。

(3) 各 Master 転送の Request

・ストリーム録画/再生に関しては、LocalFIFO (RAM)内データ量の閾値による段階制御

・Program データ転送に関しては、S/W 設定による固定制御

<u>各 Master 転送要因の Request 制御</u>

(1)EXTIN 転送

3 段階のリクエストを設定(REQ2 ＞ REQ1 ＞ REQ0 の順でリクエストが高い)。各段階は LocalFIFO 内のデータ量によって決定される。各段階のデータ量閾値はレジスタ設定可能。

(2) RECOUT 転送

3 段階のリクエストを設定(REQ2 ＞ REQ1 ＞ REQ0 の順でリクエストが高い)。各段階は LocalFIFO 内のデータ量によって決定される。各段階のデータ量閾値はレジスタ設定可能。

(3)HDDIN 転送

EXTIN 転送と同様。

(4)PLAYOUT 転送

RECOUT 転送と同様。

(5) PRG 転送

Master Read (FR65E へのデータ Write)動作に関するリクエストアサート条件は、PRGDIR=’0’ ＋ FIFO に 8 個 (1 個は 32bit) 以上の空領域。Master Write (FR65E からの Program データ Read)動作に関するリクエストアサート条件は、PRGDIR=’1’ ＋ FIFO に 8 個 (1 個は 32bit) 以上のデータ領域。

なお、リクエストの段階制御は、レジスタ設定により 3 段階(REQ2、REQ1、REQ0)により選択可能。

(6) 転送アドレステーブル転送

転送アドレステーブル読み出しに関しては、各 Master 転送要因についてそれぞれリクエスト信号(TBLREQ)が割り当てられる。TBLREQ がアサートされた場合、その Master 要因のデータ転送に先立って、テーブルの転送が優先されるが、転送テーブルの先読みを円滑に行うために、以下のように TBLREQ のアサート条件を設定する。

(a) 先読みすべきアドレスがある(次の転送アドレステーブルは未読)場合に、TBLREQ の設定モードに移行する。
(b) 現ページアクセスのアドレス(4096byte 空間の転送数)により、リクエストレベルを判断する。ただしアプリケーション開始時は、レジスタ設定にもよるが、まずテーブルを読まなければならないので REQ2 となる。
(c) テーブルのリクエストレベルが該当のデータ転送リクエストレベル以上であれば、TBLREQ をアサートする (もしテーブルのリクエストが REQ2 なら必ず TBLREQ がアサートすることになる)

Master 転送優先順位制御

各 Master 転送要因の優先順位は、前項で示したように、RECOUT ＞ PLAYOUT ＞ EXTIN ＞ HDDIN ＞ PRG の順である。しかし、このように優先順位が固定されていると、優先順位の高い転送のみが実行され、他の転送が行われない可能性がある。これを回避する

ために、Master 転送制御部では、各転送要因のリクエストに 3 つの段階を設け、優先順位の低い転送要因であっても、リクエスト要求が高い場合は、この転送に対して優先して割り当てを行う。各 Master 転送要因からのリクエストにしたがって、どの Master 転送を実行するかを判断する。優先順位の判断は、リクエストが無い状態からリクエストが発生した時か、または各 Master 転送の終了時（LBLAST の次のクロック）で行う。その優先度は以下のように決定する。

(1) リクエストレベルの高い(REQ2 ＞ REQ1 ＞ REQ0 の順)要因が優先
(2) 同一レベルのリクエストが複数ある場合は、
    ① RECOUT
    ② PLAYOUT
    ③ EXTIN
    ④ HDDIN
    ⑤ PRG
    の順に優先付けする。
(3) 各Master 転送要因において､転送アドレステーブルの読み出しが必要な場合は、テーブル読み出しリクエスト信号がアサートされ､(1)～(2)で決定されたMaster 転送要因の転送アドレステーブルの読み出しが行われる。

転送アドレステーブル

OS のページ変換機構によって、不連続な物理アドレスにマッピングされたメモリ領域を順にアクセスするために、転送アドレステーブルを用いた Master 転送を行う。転送アドレステーブルのフォーマットは以下のとおり。

転送アドレステーブルは DWORD 境界に配置される必要がある。また、テーブル自体がページ(4096byte)境界をまたぐことはできないので､ページ境界を越えるテーブルを作成する場合は、Table Link Bit によって、テーブルのリンクを行わなければならない。

[Table Link Bit]
このビットが立っている場合､bit31-2 は次の転送アドレステーブルのアドレスを示す。

[Transfer Interrupt Bit]
このビットが立っているエントリーに対応するページ(4096byte)の最終バイトを含む TS パケットとの転送が完了した時に、Transfer Interrupt を発生する。転送はそのまま継続される。

[Transfer Abort Bit]
このビットが立っているエントリーに対応するページの最終バイトの転送が完了した時に、即時停止、Abort Interrupt が発生する。
パケットの途中でも転送を中止するため､中途半端なデータが転送される場合がある。

[Transfer Stop Bit]

このビットが立っているエントリーに対応するページの最終バイトを含むTS パケットの転送が完了した時に停止し、Stop Interrupt が発生する。

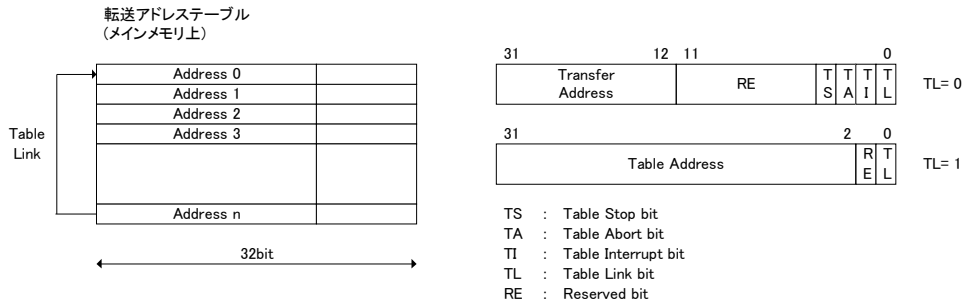


図 5-6 転送アドレステーブル

● 共有メモリ

セキュアコントローラとセキュアソフト間の通信に使用されるメモリ。512byte のエリアを４面持ち、個々にアクセス方法として、READ ONLY、WRITE ONLY、READ ONCE、READ WRITE の設定が可能である。これらアクセス方法の設定は、セキュアコントローラから設定する。セキュアコントローラからは常に READ WRITE 可能。通信レジスタと組み合わせて、セキュアシステムの全ての処理の通信を行う。

５－１－４　セキュアハードにおける暗号・復号動作

本項では、セキュアハードの重要な機能の一つであるＡＶコンテンツの暗号化・復号化機能について、動作例をもとに解説する。

５－１－４－１　暗号・復号ブロック概要

本システムは、セキュアハードに入力された MPEG TS データをコンテンツの権利を保障した状態で自由に視聴し、ハードディスクへの保存等も行う事を実現するものである。

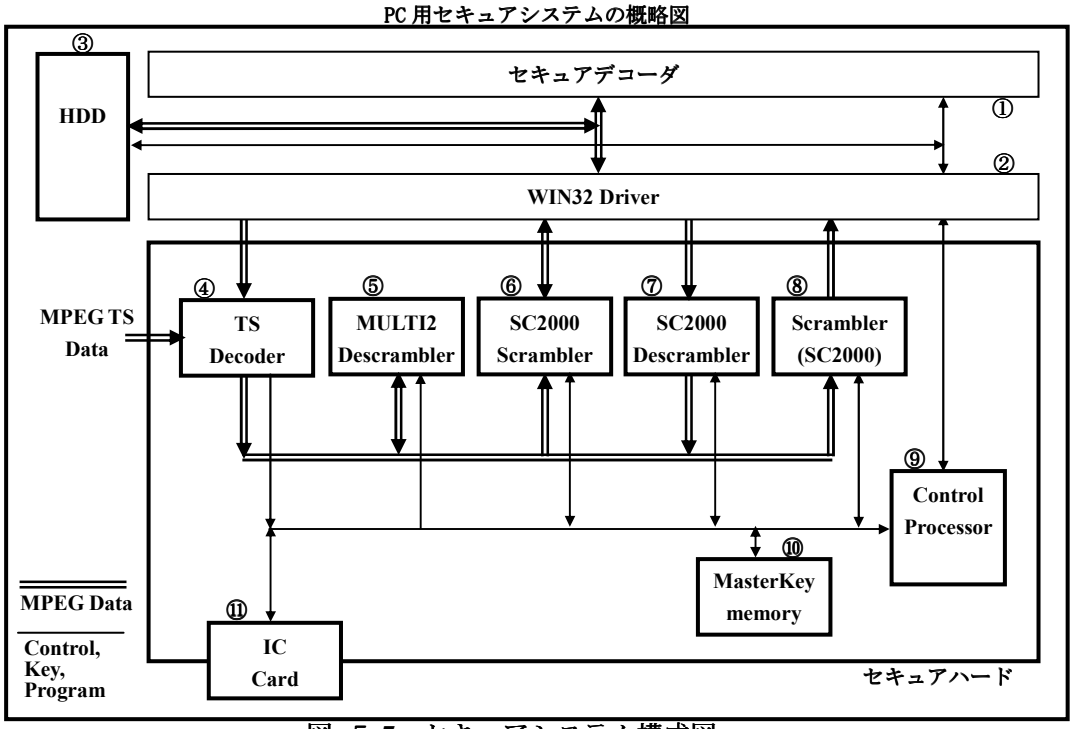


図 5-7　セキュアシステム構成図

①：セキュアデコーダ

セキュアソフトのうちドライバ、ローダを除いた、ＭＰＥＧデコードに関する部分。（ビデオとオーディオの両方をデコードする。）

②：Win32 Driver

セキュアソフトのドライバ部分が含まれる。

③：ハードディスク

PC の主要な構成パーツであるハードディスク。セキュアシステムにおいても一般的な使用方法から逸脱した使い方はしない。セキュアシステムでは、セキュアソフトパーツ、ローダ等が格納される。

④：TS Decoder

ECM、EMM などのパケットを取り出し、セキュアコントローラに割り込みによっ

て通知する機能がある。割り込みの対象パケットの設定は、PID を設定する事で行う。

⑤：MULTI2 Descrambler
MULTI2 で暗号化された MPEG TS データのペイロード部分を復号化する処理を行う。セキュアコントローラからは、EVEN、ODD の 2 つの鍵を設定するだけで、MULTI2 の復号処理が行われる。

⑥：SC2000 Scrambler
MPEG2 TS のペイロード部分を SC2000 で暗号化する処理を行う。セキュアコントローラからは、EVEN、ODD の 2 つの鍵を設定し、鍵の切替タイミングを指示する。実際の鍵の変更シークエンスでは、これ以外に、使用した鍵をパケットとしてデータに埋め込まなければならない。鍵を設定する鍵バッファと鍵の切替タイミングを指定するレジスタと鍵情報をデータに埋め込む為の鍵パケットバッファが用意されている。

⑦：SC2000 Descrambler
SC2000 で暗号化された MPEG TS データのペイロード部分を復号化する処理を行う。セキュアコントローラからは、EVEN、ODD の 2 つの鍵を設定するだけで、SC2000 の復号処理が行われる。SC2000 の鍵データパケットを抽出する為に指定した PID のパケットを取り出し、セキュアコントローラに割り込みによって通知する機能がある。

⑧：Scrambler(SC2000)
MPEG2 TS のペイロード部分を SC2000 で暗号化する処理を行う。セキュアコントローラからは、EVEN、ODD の 2 つの鍵を設定し、鍵の切替タイミングを指示する。実際の鍵の変更シークエンスでは、これ以外に、使用した鍵をパケットとしてデータに埋め込まなければならない。鍵を設定する鍵バッファと鍵の切替タイミングを指定するレジスタと鍵情報をデータに埋め込む為の鍵パケットバッファが用意されている。

⑨：Control Processor(FR60)
セキュアシステムの全ての機能の制御と通信を行うコントロールプロセッサ。実験システムでは、FR65 が使用されている。

⑩：MasterKey memory (Flash)
セキュアシステム全体のマスター鍵を記録する為のメモリ。実験システムでは、どの様に実装されるかは未定であるが、最終的なシステムでは、不揮発のメモリでセキュアハード外からは、アクセスできない状態で実装される事になる。

⑪：IC Card
MPEG2 TS データの視聴権利情報の管理を行う為の IC カード。MULTI2 の鍵情報等を CPU と通信する事によってデータ取り出すことができる。通信内容は、ほとんど明らかにされておらず、実際の通信は、ARIB STD-B25 に書かれている様に ECM、EMM などの情報をほぼそのまま使用することで必要なデータを取り出す。

MPEG-TS の各部での存在形態

セキュアシステムでは、データの存在位置によって基本的なデータの形態が決まっている。あらゆる状態においてもコンテンツの権利侵害が発生しない事を保証しなければならないので、データがどの様な位置に存在する場合でも何等かの暗号が施されている。セキュアシステムでは、データの存在位置（入力も含む）として大きく分けて

セキュアシステム外(データ入力)、ハードディスク、セキュアデコーダ、

セキュアハード内

が考えられる。この中でセキュアハード内だけは特殊で、それぞれのデータ存在位置へデータ移動させる為のデータ変換（暗号化、暗号復号化等）を行っている為、全てのデータ存在形態が混在する。それぞれの存在位置でのデータ形態は下表の様になっている。

データの存在位置とデータ形態

データの存在位置	データの形態
セキュアシステム外(データ入力)	MULTI2 で暗号化された MPEG TS
ハードディスク	１． SC2000 で暗号化された MPEG TS ２． SC2000 と MULTI2 で暗号化された MPEG TS Cf.ハードディスク内は、基本的には、SC2000 のみで暗号化された MPEG TS データが保存される。
セキュアデコーダ	SC2000(ローカル暗号)で暗号化された MPEG TS
セキュアハード内	１． 暗号化されていない MPEG TS ２． MULTI2 で暗号化された MPEG TS ３． MULTI2,SC2000 で暗号化された MPEG TS ４． SC2000 で暗号化された MPEG TS

データの存在位置の中でも、セキュアシステム外(データ入力)とハードディスクは、データのコピーを避ける事は、不可能である。しかしこれら部分では、少なくとも MULTI2 あるいは、SC2000 で暗号化された状態にあるので、権利侵害が発生しないものと考える。またセキュアハード内のデータについては、暗号化されていないデータも存在するので非常に危険ではあるが、セキュアハード内のモジュールは、最終的には、大部分が一つのＴＲＭにまとめられるので、権利侵害が発生しないものと考える。最も、危険であると思われるのは、SC2000（ローカル暗号）で暗号化されただけのデータが存在するセキュアデコーダ内とセキュアハードとセキュアデコーダ間の転送の場合である。この部分については、データの転送方法を変えたり、データ転送位置を変更したりする工夫が必要である。

５-１-４-２　暗号・復号ブロックにおける処理パターン

セキュアシステム上で実現する基本的な処理の詳細を以下に記す。
本システムは、セキュアハードに入力された MPEG TS データをコンテンツの権利を保障した状態で自由に視聴し、ハードディスクへの保存等も行う事を実現するものである。

本システムで実現する基本的な処理は、

1. セキュアハードに入力された TS データを視聴する。
2. セキュアハードに入力された TS データを HDD に保存する。
3. セキュアハードに入力された TS データを視聴しながら HDD に保存する。
4. HDD に保存された TS データを視聴する。
5. Net 等で入手した TS データを視聴する。
6. Net 等で入手した TS データを HDD に保存する。
7. Net 等で入手した TS データを視聴しながら HDD に保存する。

の７通りである。

視聴時の状態としてタイムシフト機能を使用する場合、HDD を介在させ自由な再生時間調整

を実現する。これにより基本的な処理は、

1. セキュアハードに入力された TS データを視聴する。(タイムシフト機能無)
2. セキュアハードに入力された TS データを視聴する。(タイムシフト機能有)
3. セキュアハードに入力された TS データを HDD に保存する。
4. セキュアハードに入力された TS データを視聴しながら HDD に保存する。(タイムシフト機能無)
5. セキュアハードに入力された TS データを視聴しながら HDD に保存する。(タイムシフト機能有)
6. HDD に保存された TS データを視聴する。
7. Net 等で入手した TS データを視聴する。(タイムシフト機能無)
8. Net 等で入手した TS データを視聴する。(タイムシフト機能有)
9. Net 等で入手した TS データをを HDD に保存する。
10. Net 等で入手した TS データを視聴しながら HDD に保存する。(タイムシフト機能無)
11. Net 等で入手した TS データを視聴しながら HDD に保存する。(タイムシフト機能有)

の１１通りになる。

これに HDD を介在、或いは、保存する場合に、MULTI2 復号済みか、否かにより（蓄積後課金）基本的な処理は、

1. セキュアハードに入力された TS データを視聴する。(タイムシフト機能無)
2. セキュアハードに入力された TS データを視聴する。(タイムシフト機能有)(MULTI2 復号未)
3. セキュアハードに入力された TS データを視聴する。(タイムシフト機能有)(MULTI2 復号済)
4. セキュアハードに入力された TS データを HDD に保存する。(MULTI2 復号未)
5. セキュアハードに入力された TS データを HDD に保存する。(MULTI2 復号済)
6. セキュアハードに入力された TS データを視聴しながら HDD に保存する。(タイムシフト機能無)(MULTI2 復号未)
7. セキュアハードに入力された TS データを視聴しながら HDD に保存する。(タイムシフト機能無)(MULTI2 復号済)
8. セキュアハードに入力された TS データを視聴しながら HDD に保存する。(タイムシフト機能有)(MULTI2 復号未)
9. セキュアハードに入力された TS データを視聴しながら HDD に保存する。(タイムシフト機能有)(MULTI2 復号済)
10. HDD に保存された TS データを視聴する。(MULTI2 復号未)
11. HDD に保存された TS データを視聴する。(MULTI2 復号済)
12. Net 等で入手した TS データを視聴する。(タイムシフト機能無)
13. Net 等で入手した TS データを視聴する。(タイムシフト機能有)(MULTI2 復号未)
14. Net 等で入手した TS データを視聴する。(タイムシフト機能有)(MULTI2 復号済)
15. Net 等で入手した TS データを HDD に保存する。(MULTI2 復号未)
16. Net 等で入手した TS データを HDD に保存する。(MULTI2 復号済)
17. Net 等で入手した TS データを視聴しながら HDD に保存する。(タイムシフト機能無)(MULTI2 復号未)
18. Net 等で入手した TS データを視聴しながら HDD に保存する。(タイムシフ

ト機能無）（MULTI2 復号済）
19. Net 等で入手した TS データを視聴しながら HDD に保存する。（タイムシフト機能有）（MULTI2 復号未）
20. Net 等で入手した TS データを視聴しながら HDD に保存する。（タイムシフト機能有）（MULTI2 復号済）

の２０通りになる。

以上の基本的な処理、１～２０をそれぞれ

1. Play Only, IN: SecureHard, TimeShift: OFF
2. Play Only, IN: SecureHard, TimeShift: ON, MULTI2: ON
3. Play Only, IN: SecureHard, TimeShift: ON, MULTI2: OFF
4. Save Only, IN: SecureHard, MULTI2: ON
5. Save Only, IN: SecureHard, MULTI2: OFF
6. Play & Save, IN: SecureHard, TimeShift: OFF, MULTI2: ON
7. Play & Save, IN: SecureHard, TimeShift: OFF, MULTI2: OFF
8. Play & Save, IN: SecureHard, TimeShift: ON, MULTI2: ON
9. Play & Save, IN: SecureHard, TimeShift: ON, MULTI2: OFF
10. Play Only, IN: HDD, MULTI2: ON
11. Play Only, IN: HDD, MULTI2: OFF
12. Play Only, IN: Net, TimeShift: OFF
13. Play Only, IN: Net, TimeShift: ON, MULTI2: ON
14. Play Only, IN: Net, TimeShift: ON, MULTI2: OFF
15. Save Only, IN: Net, MULTI2: ON
16. Save Only, IN: Net, MULTI2: OFF
17. Play & Save, IN: Net, TimeShift: OFF, MULTI2: ON
18. Play & Save, IN: Net, TimeShift: OFF, MULTI2: OFF
19. Play & Save, IN: Net, TimeShift: ON, MULTI2: ON
20. Play & Save, IN: Net, TimeShift: ON, MULTI2: OFF

と表記し、例として、1,3,10,14,16,18 の詳細処理を記述する。

<1> Play Only, IN: SecureHard, TimeShift: OFF

セキュアハードに入力されたMPEG TSデータをタイムシフト機能無しで、ＰＣ上のセキュアデコーダで視聴を行う場合の処理。

Play Only, IN: SecureHard, TimeShift: OFF の概略図

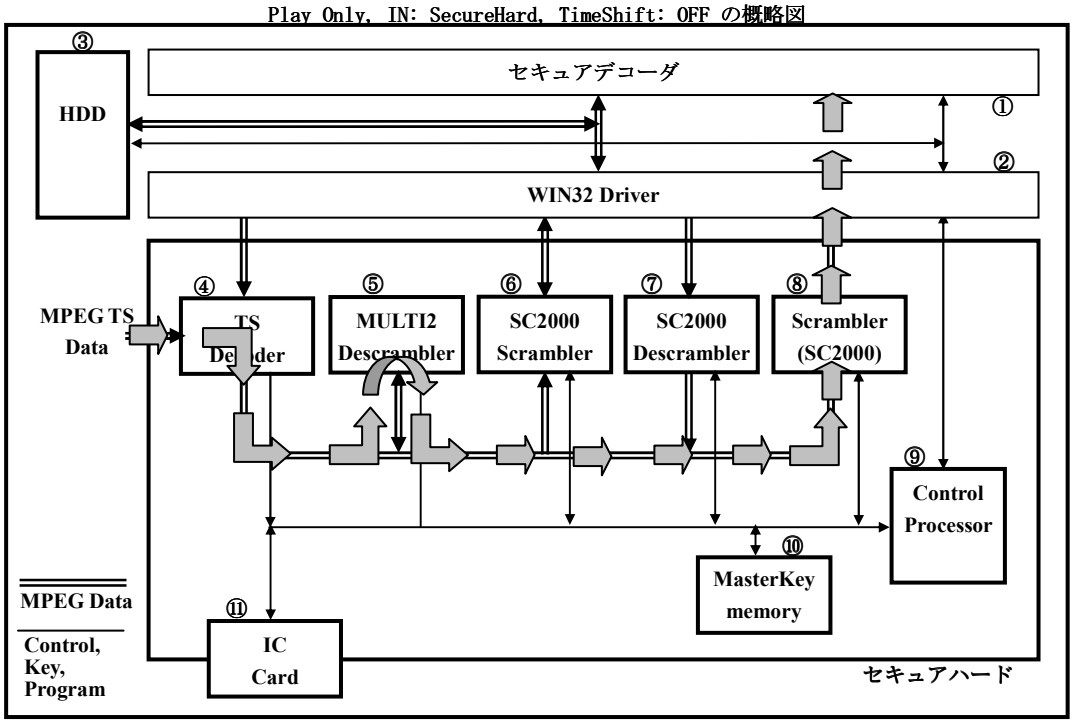


図 5-8 処理パターン１

１．TS decoder

処理：

入力されたMPEG TSデータのヘッダ部から鍵情報(EMM／ECM)等を抽出する。(MPEG TSデータはPayload部のみMULTI2により暗号化されている。ヘッダ部は暗号化されていない。ヘッダ部のECMにはOdd KeyとEven Keyが格納されている。)

- TS decoderで処理されたMPEG TSデータはTSパケットのまま、またはPayload部のみがFPGA内のRAMエリアに格納される
- TS decoderはFPGA内のRAMエリアにTSデータを書き込むとFR65に割込み要求を出す。
- 割込み要求を受け取ったFR65は、FPGA内のRAMエリアのTSデータからバージョンナンバー（version_number：５bit）を取得し、過去に取得したバージョンナンバーと比較する。もしバージョンナンバーが異なれば、FR65はFPGA内のRAMエリア格納されているTSデータをSDRAMへ転送する。バージョンナンバーが同値であればFR65は何もしない。
- FR65はSDRAMに転送されたTSデータからOdd/Even鍵を取り出し、IC Cardにセットする
- IC CardにOdd/Even鍵をセットするとIC CardからK_S－Mを取り出すことが可能となる。
- FR65はIC CardからK_S－Mを取り出し、MULTI2 Descramblerにセットする。

２．MULTI2 Descrambler
処理：

MPEG TS データの Payload 部にかけられた MULTI2 を復号する。

３．Scrambler(SC2000)
処理：

MULTI2 が解かれた TS データの Payload 部のみ SC2000 で暗号化する。SC2000 で暗号化された TS データは Win32 Driver を経由してセキュアデコーダに送られる。

- Scrambler(SC2000)での暗号化には、セキュアデコーダ暗号スクランブル鍵（K_S－D）が用いられる。K_S－D は K_C－D により暗号化されており、K_C－D は Km により暗号化されている。このため、K_S－D を使用するためには K_S－D と K_C－D の復号処理が必要である。

４．セキュアデコーダ
処理：

SC2000 を復号し、ストリームを MPEG２デコーダにより再生する。

- セキュアデコーダ暗号スクランブル鍵（KS－D）を用いて SC2000 を復号する。
- ストリームの復号化
- PAT/PMT 解析

<3> Play Only, IN: SecureHard, TimeShift: ON, MULTI2: OFF

セキュアハードに入力されたMPEG TSデータをタイムシフト機能有りで、ＰＣ上のセキュアデコーダで視聴を行う場合の処理。タイムシフト機能を実現する為のハードディスクへのバッファリング時のデータ形式はSC2000。

Play Only, IN: SecureHard, TimeShift: ON, MULTI2: OFF の概略図

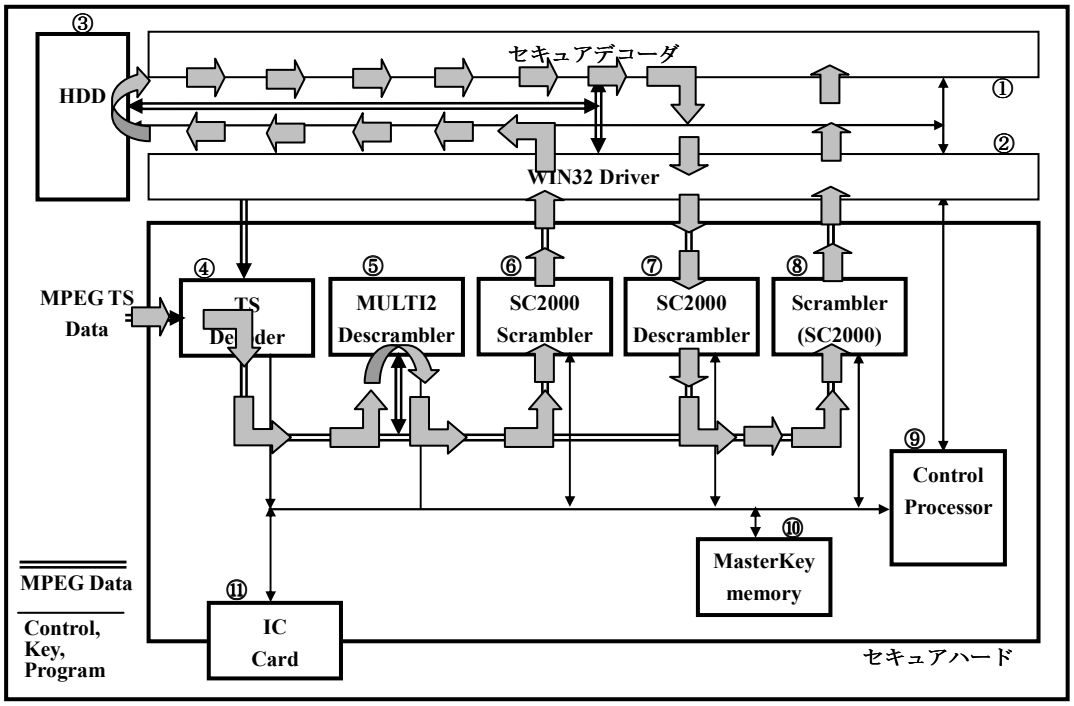


図 5-9 処理パターン3

１． TS decoder

処理：

入力されたMPEG TSデータのヘッダ部から鍵情報(EMM／ECM)等を抽出する。(MPEG TSデータはPayload部のみMULTI2により暗号化されている。ヘッダ部は暗号化されていない。ヘッダ部のECMにはOdd KeyとEven Keyが格納されている。)

- TS decoderで処理されたMPEG TSデータはTSパケットのまま、またはPayload部のみがFPGA内のRAMエリアに格納される。
- 割込み要求を受け取ったFR65は、FPGA内のRAMエリアのTSデータからバージョンナンバー（version_number：５bit）を取得し、過去に取得したバージョンナンバーと比較する。もしバージョンナンバーが異なれば、FR65はFPGA内のRAMエリア格納されているTSデータをSDRAMへ転送する。バージョンナンバーが同値であればFR65は何もしない。
- FR65はSDRAMに転送されたTSデータからOdd/Even鍵を取り出し、IC Cardにセットする
- IC CardにOdd/Even鍵をセットするとIC CardからK_S−Mを取り出すことが可能となる。
- FR65はIC CardからK_S−Mを取り出し、MULTI2 Descramblerにセットする。

２．MULTI2 Descrambler
処理：

MPEG TS データの Payload 部にかけられた MULTI2 を復号する。

３．SC2000 Scrambler
処理：

MULTI2 により暗号化されている MPEG TS データを Payload 部のみ SC2000 により暗号化する。

- ここで用いられる暗号鍵は KS－SC2K である。KS－SC2K は KC－SC2K により暗号化されており、KC－SC2K は Km により暗号化されている。このため、KS－SC2K を使用するためには KS－SC2K と KC－SC2K の復号処理が必要である。

４．HDD
処理：

SC2000 により暗号化されている MPEG TS データを HDD にバッファリングする。HDD は、セキュアソフトのタイムシフト機能が指定する MPEG TS データを掃き出す。

- HDD でのデータ形式は SC2000
- バッファの容量、バッファが確保できない場合（HDD に空き容量がない等）について検討する必要がある。

５． SC2000 Descrambler
処理：
SC2000 を復号する。

- ここで用いられる暗号鍵は KS－SC2K である。KS－SC2K は KC－SC2K により暗号化されており、KC－SC2K は Km により暗号化されている。このため、KS－SC2K を使用するためには KS－SC2K と KC－SC2K の復号処理が必要である。

６．Scrambler(SC2000)
処理：

MULTI2 が解かれた TS データの Payload 部のみ SC2000 で暗号化する。SC2000 で暗号化された TS データは Win32 Driver を経由してセキュアデコーダに送られる。

- Scrambler(SC2000)での暗号化には、セキュアデコーダ暗号スクランブル鍵（KS－D）が用いられる。KS－D は KC－D により暗号化されており、KC－D は Km により暗号化されている。このため、KS－D を使用するためには KS－D と KC－D の復号処理が必要である。

７．セキュアデコーダ
処理：

SC2000 を復号し、ストリームを MPEG２デコーダにより再生する。

- セキュアデコーダ暗号スクランブル鍵（KS－D）を用いて SC2000 を復号する。
- ストリームの復号化
- PAT/PMT 解析

<10> Play Only, IN: HDD, MULTI2: ON

ハードディスクに保存されたデータをＰＣ上のセキュアデコーダで視聴する場合の処理。ハードディスクの保存データ形式は SC2000+MULTI2。

Play Only, IN: HDD, MULTI2: ON の概略図

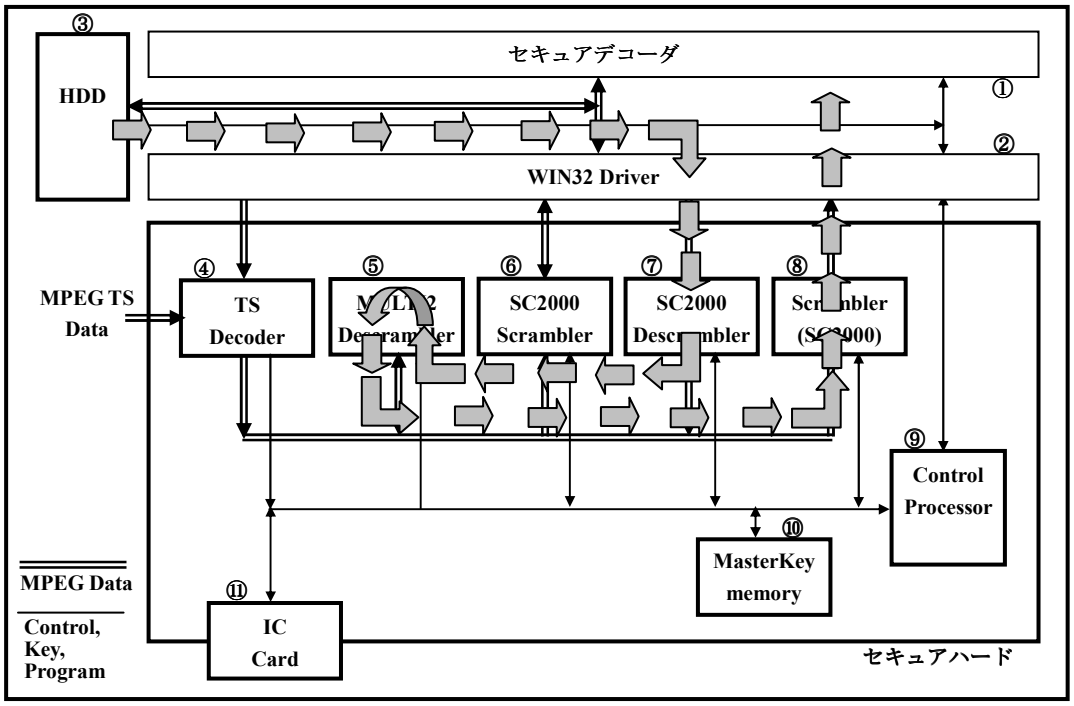


図 5-10 処理パターン 10

１．HDD

処理：

MULTI2 と SC2000 により暗号化されて HDD に保存されている MPEG TS データを送出する。

- HDD でのデータ形式は MULTI2+SC2000

２．SC2000 Descrambler

処理：

SC2000 を復号する。

- ここで用いられる暗号鍵は K_S－SC2K である。K_S－SC2K は K_C－SC2K により暗号化されており、K_C－SC2K は Km により暗号化されている。このため、K_S－SC2K を使用するためには K_S－SC2K と K_C－SC2K の復号処理が必要である。

３．MULTI2 Descrambler

処理：

MPEG TS データの Payload 部にかけられた MULTI2 を復号する。

- 復号時に用いられる K_S－M は、IC Card に格納されている。
- MULTI2 Descrambler は TS decoder で抽出された ECM を読込み、Control Processor に通知する。
- Control Processor は MULTI2 Descrambler が読込んだ ECM から Odd Key と Even Key を IC Card に転送して K_S－M の読み出し／解読を行う。（TS Decoder での処理済の TS データは SDRAM か FPGA 内の RAM エリアに格納されている。）

４．Scrambler(SC2000)

処理：

MULTI2 が解かれた TS データの Payload 部のみ SC2000 で暗号化する。SC2000 で暗号化された TS データは Win32 Driver を経由してセキュアデコーダに送られる。

- Scrambler(SC2000)での暗号化には、セキュアデコーダ暗号スクランブル鍵（K_S－D）が用いられる。K_S－D は K_C－D により暗号化されており、K_C－D は Km により暗号化されている。このため、K_S－D を使用するためには K_S－D と K_C－D の復号処理が必要である。

５．セキュアデコーダ

処理：

SC2000 を復号し、ストリームを MPEG２デコーダにより再生する。

- セキュアデコーダ暗号スクランブル鍵（K_S－D）を用いて SC2000 を復号する。
- ストリームの復号化
- PAT/PMT 解析

<14> Play Only, IN: Net, TimeShift: ON, MULTI2: OFF

Net 等で入手した MPEG TS データをタイムシフト機能有りで、ＰＣ上のセキュアデコーダで視聴する場合の処理。タイムシフト機能を実現する為のハードディスクへのバッファリング時のデータ形式は SC2000。

Play Only, IN: Net, TimeShift: ON, MULTI2: OFF の概略図

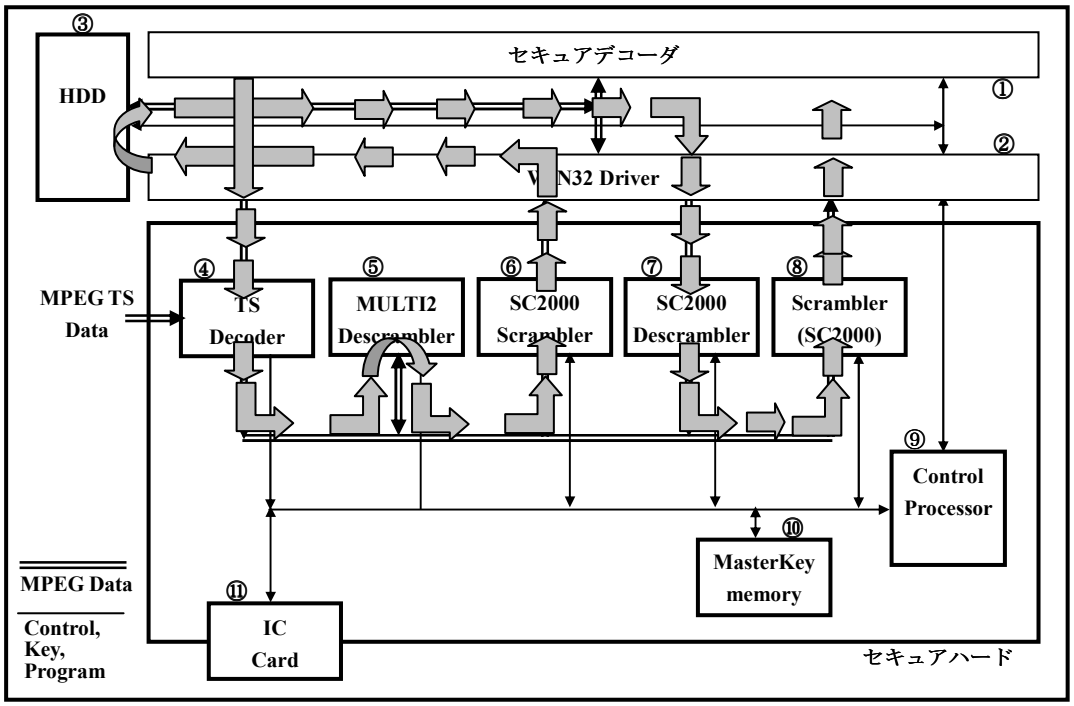


図 5-11 処理パターン 14

１．セキュアデコーダ

処理：

Net 等に接続し、MPEG TS データを受信する。

２．TS decoder

処理：

入力された MPEG TS データのヘッダ部から鍵情報(EMM／ECM)等を抽出する。(MPEG TS データは Payload 部のみ MULTI2 により暗号化されている。ヘッダ部は暗号化されていない。ヘッダ部の ECM には Odd Key と Even Key が格納されている。)

- TS decoder で処理された MPEG TS データは TS パケットのまま、または Payload 部のみが FPGA 内の RAM エリアに格納される。
- 割込み要求を受け取った FR65 は、FPGA 内の RAM エリアの TS データからバージョンナンバー（version_number：５bit）を取得し、過去に取得したバージョンナンバーと比較する。もしバージョンナンバーが異なれば、FR65 は FPGA 内の RAM エリア格納されている TS データを SDRAM へ転送する。バージョンナンバーが同値であれば FR65 は何もしない。
- FR65 は SDRAM に転送された TS データから Odd/Even 鍵を取り出し、IC Card にセットする
- IC Card に Odd/Even 鍵をセットすると IC Card から K_S－M を取り出すことが可能と

なる。
- FR65 は IC Card から K_S－M を取り出し、MULTI2 Descrambler にセットする。

３．MULTI2 Descrambler
処理：
MPEG TS データの Payload 部にかけられた MULTI2 を復号する。

４．SC2000 Scrambler
処理：
MULTI2 により暗号化されている MPEG TS データを Payload 部のみ SC2000 により暗号化する。

- ここで用いられる暗号鍵は K_S－SC2K である。K_S－SC2K は K_C－SC2K により暗号化されており、K_C－SC2K は Km により暗号化されている。このため、K_S－SC2K を使用するためには K_S－SC2K と K_C－SC2K の復号処理が必要である。

５．HDD
処理：
SC2000 により暗号化されている MPEG TS データを HDD にバッファリングする。HDD は、セキュアソフトのタイムシフト機能が指定する MPEG TS データを掃き出す。

- HDD でのデータ形式は SC2000

６．SC2000 Descrambler
処理：
SC2000 を復号する。

- ここで用いられる暗号鍵は K_S－SC2K である。K_S－SC2K は K_C－SC2K により暗号化されており、K_C－SC2K は Km により暗号化されている。このため、K_S－SC2K を使用するためには K_S－SC2K と K_C－SC2K の復号処理が必要である。

７．Scrambler(SC2000)
処理：
MULTI2 が解かれた TS データの Payload 部のみ SC2000 で暗号化する。SC2000 で暗号化された TS データは Win32 Driver を経由してセキュアデコーダに送られる。

- Scrambler(SC2000)での暗号化には、セキュアデコーダ暗号スクランブル鍵（K_S－D）が用いられる。K_S－D は K_C－D により暗号化されており、K_C－D は Km により暗号化されている。このため、K_S－D を使用するためには K_S－D と K_C－D の復号処理が必要である。

８．セキュアデコーダ
処理：
SC2000 を復号し、ストリームを MPEG２デコーダにより再生する。

- セキュアデコーダ暗号スクランブル鍵（K_S－D）を用いて SC2000 を復号する。
- ストリームの復号化
- PAT/PMT 解析

<16> Save Only, IN: Net, MULTI2: OFF

Net 等で入手した MPEG TS データをハードディスクに保存する場合の処理。ハードディスクへの保存データ形式は SC2000。

Save Only, IN: Net, MULTI2: OFF の概略図

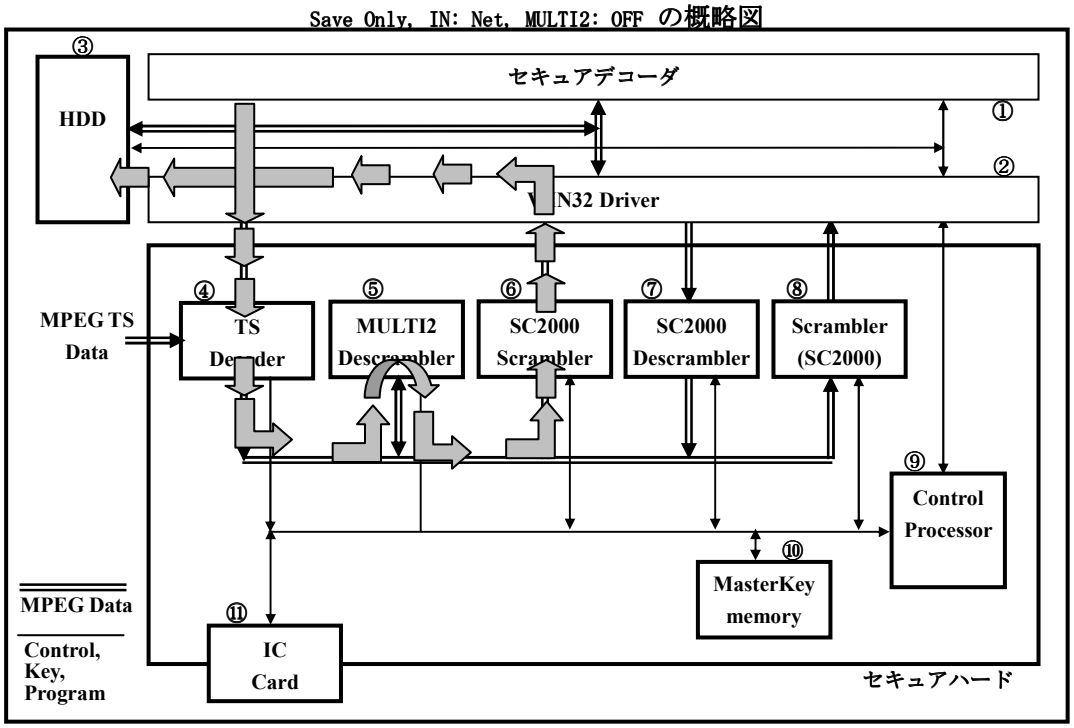


図 5-12 処理パターン 16

１．セキュアデコーダ.

処理：

Net 等に接続し、MPEG TS データを受信する。

２．TS decoder

処理：

入力された MPEG TS データのヘッダ部から鍵情報(EMM／ECM)等を抽出する。(MPEG TS データは Payload 部のみ MULTI2 により暗号化されている。ヘッダ部は暗号化されていない。ヘッダ部の ECM には Odd Key と Even Key が格納されている。)

- TS decoder で処理された MPEG TS データは TS パケットのまま、または Payload 部のみが FPGA 内の RAM エリアに格納される。
- 割込み要求を受け取った FR65 は、FPGA 内の RAM エリアの TS データからバージョンナンバー（version_number：５bit）を取得し、過去に取得したバージョンナンバーと比較する。もしバージョンナンバーが異なれば、FR65 は FPGA 内の RAM エリア格納されている TS データを SDRAM へ転送する。バージョンナンバーが同値であれば FR65 は何もしない。
- FR65 は SDRAM に転送された TS データから Odd/Even 鍵を取り出し、IC Card にセットする
- IC Card に Odd/Even 鍵をセットすると IC Card から K_S−M を取り出すことが可能となる。

・ FR65 は IC Card から K_S－M を取り出し、MULTI2 Descrambler にセットする。

３．MULTI2 Descrambler
処理：
MPEG TS データの Payload 部にかけられた MULTI2 を復号する。

４．SC2000 Scrambler
処理：
MULTI2 により暗号化されている MPEG TS データを Payload 部のみ SC2000 により暗号化する。

・ ここで用いられる暗号鍵は K_S－SC2K である。K_S－SC2K は K_C－SC2K により暗号化されており、K_C－SC2K は Km により暗号化されている。このため、K_S－SC2K を使用するためには K_S－SC2K と K_C－SC2K の復号処理が必要である。

５．HDD
処理：
SC2000 により暗号化されている MPEG TS データを HDD に保存する。

・ 保存データ形式は SC2000

<18> Play & Save, IN: Net, TimeShift: OFF, MULTI2: OFF

Net 等で入手した MPEG TS データをタイムシフト機能無しで、ＰＣ上のセキュアデコーダで視聴しながらハードディスクに保存する場合の処理。ハードディスクへの保存データ形式は SC2000。

Play & Save, IN: Net, TimeShift: OFF, MULTI2: OFF の概略図

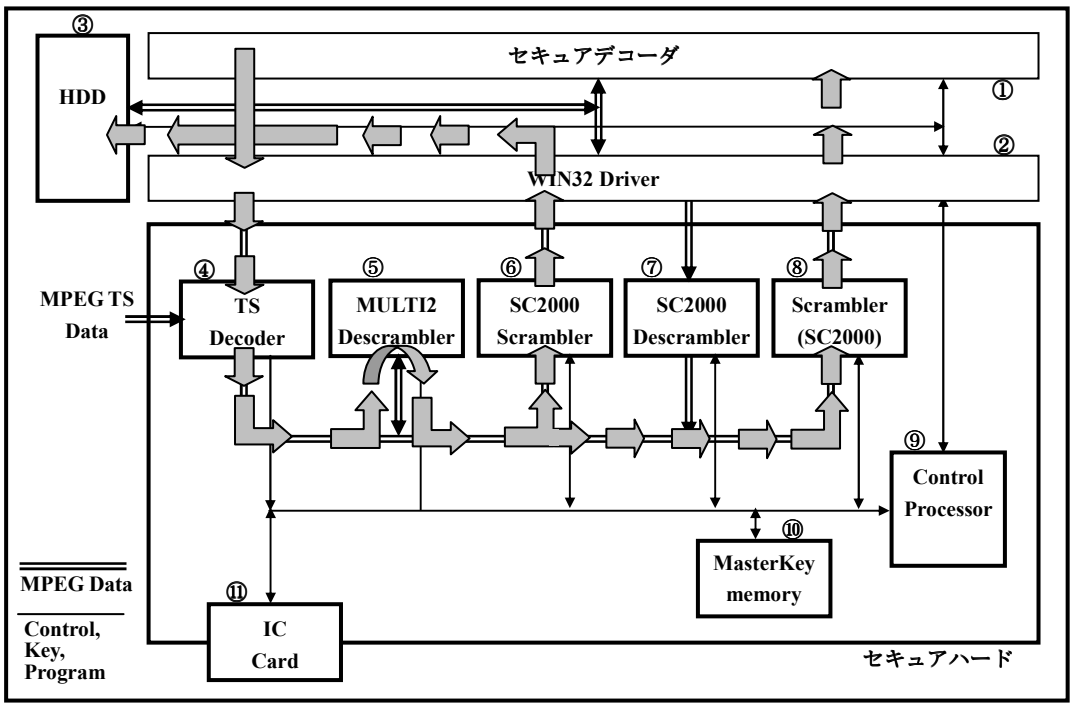


図 5-13 処理パターン 18

１．セキュアデコーダ.

処理：

Net 等に接続し、MPEG TS データを受信する。

２．TS decoder

処理：

入力された MPEG TS データのヘッダ部から鍵情報(EMM／ECM)等を抽出する。(MPEG TS データは Payload 部のみ MULTI2 により暗号化されている。ヘッダ部は暗号化されていない。ヘッダ部の ECM には Odd Key と Even Key が格納されている。)

- TS decoder で処理された MPEG TS データは TS パケットのまま、または Payload 部のみが FPGA 内の RAM エリアに格納される。
- 割込み要求を受け取った FR65 は、FPGA 内の RAM エリアの TS データからバージョンナンバー（version_number：５bit）を取得し、過去に取得したバージョンナンバーと比較する。もしバージョンナンバーが異なれば、FR65 は FPGA 内の RAM エリア格納されている TS データを SDRAM へ転送する。バージョンナンバーが同値であれば FR65 は何もしない。
- FR65 は SDRAM に転送された TS データから Odd/Even 鍵を取り出し、IC Card にセッ

トする
- IC Card に Odd/Even 鍵をセットすると IC Card から K_S－M を取り出すことが可能となる。
- FR65 は IC Card から K_S－M を取り出し、MULTI2 Descrambler にセットする。

３．MULTI2 Descrambler
処理：
MPEG TS データの Payload 部にかけられた MULTI2 を復号する。

４．SC2000 Scrambler
処理：
MULTI2 により暗号化されている MPEG TS データを Payload 部のみ SC2000 により暗号化する。

- ここで用いられる暗号鍵は K_S－SC2K である。K_S－SC2K は K_C－SC2K により暗号化されており、K_C－SC2K は Km により暗号化されている。このため、K_S－SC2K を使用するためには K_S－SC2K と K_C－SC2K の復号処理が必要である。

５．Scrambler(SC2000)
処理：
MULTI2 が解かれた TS データの Payload 部のみ SC2000 で暗号化する。SC2000 で暗号化された TS データは Win32 Driver を経由してセキュアデコーダに送られる。

- Scrambler(SC2000)での暗号化には、セキュアデコーダ暗号スクランブル鍵（K_S－D）が用いられる。K_S－D は K_C－D により暗号化されており、K_C－D は Km により暗号化されている。このため、K_S－D を使用するためには K_S－D と K_C－D の復号処理が必要である。

６．HDD
処理：
SC2000 により暗号化されている MPEG TS データを HDD に保存する。

- 保存データ形式は SC2000

７．セキュアデコーダ
処理：
SC2000 を復号し、ストリームを MPEG２デコーダにより再生する。

- セキュアデコーダ暗号スクランブル鍵（K_S－D）を用いて SC2000 を復号する。
- ストリームの復号化
- PAT/PMT 解析

５−１−５　まとめ

本項では、セキュアハードモジュールの構成、機能について解説した。セキュアシステムを考えた場合、外部からの「改ざん」や「覗き見」ができない耐タンパモジュールの役割は重要であり、本研究開発のシステムにおいても、ＡＶコンテンツやＰＣ上のソフトウェアを保護するための重要な情報や機能は、「セキュアハード」内で保護されている。本項では、ハードウェアの機能として、ＡＶコンテンツの著作権保護に使用する暗号・復号処理制御を中心に解説したが、セキュアなシステムを実現する上で、ＰＣ上で実行されるソフトウェアを保護するための機能が極めて重要である。ＰＣ上のソフトウェアの保護は、主に「セキュアハード」内のコントロールプロセッサから行なわれおり、方式等の詳細については、次項以降で解説する。

今年度までの研究開発により、ＰＣでデジタル放送受信機を実現する上で「セキュアハード」に必要とされる機能を確認した。今後は、製品化に向け、高機能・小型・低コスト化を目指した検討が必要である。以下に、製品化に向けた「セキュアハード」の検討項目を挙げる。

(1)　セキュア機能性能評価
- 暗号鍵の変更頻度などの検討(性能評価)

(2)　プロセッサ／専用ハードウェアの機能分担
- 低コスト化を考慮し、専用ハードウェアで実現すべき機能とプロセッサで実現すべき機能の切り分けの検討

(3)　必要なプロセッサ性能・メモリ容量の評価
- 機能を実現する上で必要なプロセッサ性能・メモリ容量の検討。性能/コストのトレードオフとなる

(4)　製品化に必要な機能検討
- 暗号コンテンツにおけるトリックプレイ(早送り・巻き戻し)方式
- 権利移動機能の搭載(公開鍵暗号方式)

これらの項目は、相互に関連しており、セキュア強度・性能・コストなどを総合的に検討する必要がある。今後、開発したＦＰＧＡボードでの評価・検討を進めるとともに、製品化に向けたＬＳＩを開発する予定である。

５－２セキュア機能の研究開発

５-２-１概要

Secure PC System は、Secure PC Card と MPEG2 ソフトデコーダにより構成され、難読化,暗号化,プログラムコード/データの再構成,リアルタイム認証等により、BS デジタル放送などで使用される MPEG2 データ(MP@HL)の復号処理をセキュアに実施することを目的とした装置である。Secure PC System は、MPEG2 TS データを暗号化・復号化を行う secure hard,プログラムコード再構成を行う secure firm を搭載する Secure PC Card と MPEG2 ソフトデコードおよび Secure PC Card 制御を行う secure soft より構成される。

ストリームデータの復号を行うのは secure soft モジュールであるが、この secure soft を一般的なソフトウェアとして構築すると、悪意を持つ攻撃者に容易に解析されてしまう。そこで、これらの攻撃に対して容易に解析されない、耐タンパ性を持つシステムを構築する必要がある。

本システムでは、secure soft は暗号化してディスクに置き、実行時に暗号化データを Secure PC Card に渡してメモリ上に展開して実行する。暗号化された secure soft は、実行時に Loader/Core Driver によりセキュアファームに渡され、セキュアファームがコードをメモリに展開する（

図 ５-２-1)。また、セキュアファームはセキュアソフトのリアルタイム認証を行ったり、メモリ上のコードデータをスキャンして改変されていないかチェックすることで、動作しているソフトウェアの正当性を保証する。

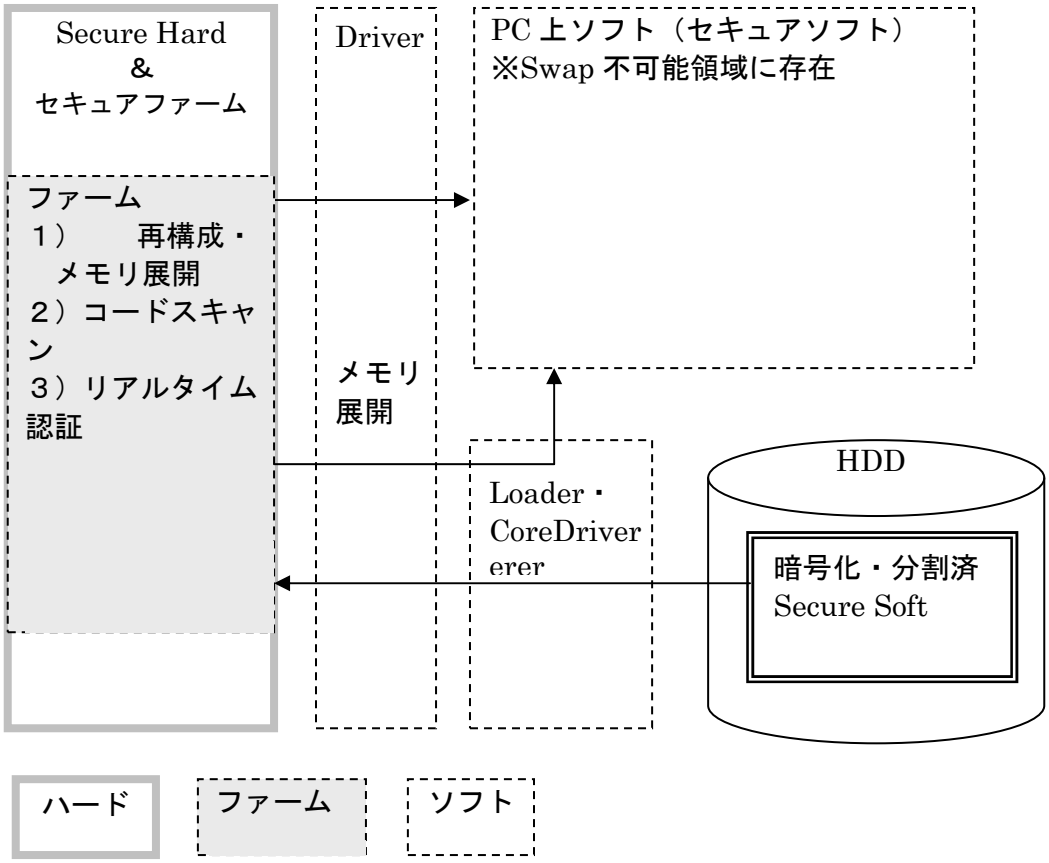


図 5-2-1 Secure System における Secure Soft のロード

また、単純に暗号化されたものをメモリ展開するだけでは、メモリ上に展開される実行イメージは常に同じとなるため、メモリスナップショットから、パターンマッチなどを使用して解析することが比較的容易に可能である。そこで、本システムではコードの分割・再構成を行うことでメモリ展開イメージがロードのたびに異なるようにする。

下図にコードの再構成の概要を示す。まず、ソフトウェア暗号化時にコードを分割し、それを再構成が可能なように各種情報を付加した形で保存する。Loader がセキュアファームに暗号化・分割済ソフトデータを渡す(1)と、セキュアファームはセキュアハードにあるマスター鍵とコード鍵からコードを復号する(2)。そして復号したコード部品をコード展開用バッファに展開する。このとき、コードはメモリ上にどのように配置してもよい(3)。メモリ上に配置したら、セキュアファームはコード部品同士を付加情報を元にリンクし、元のコード通りに動作するようにする(4)。

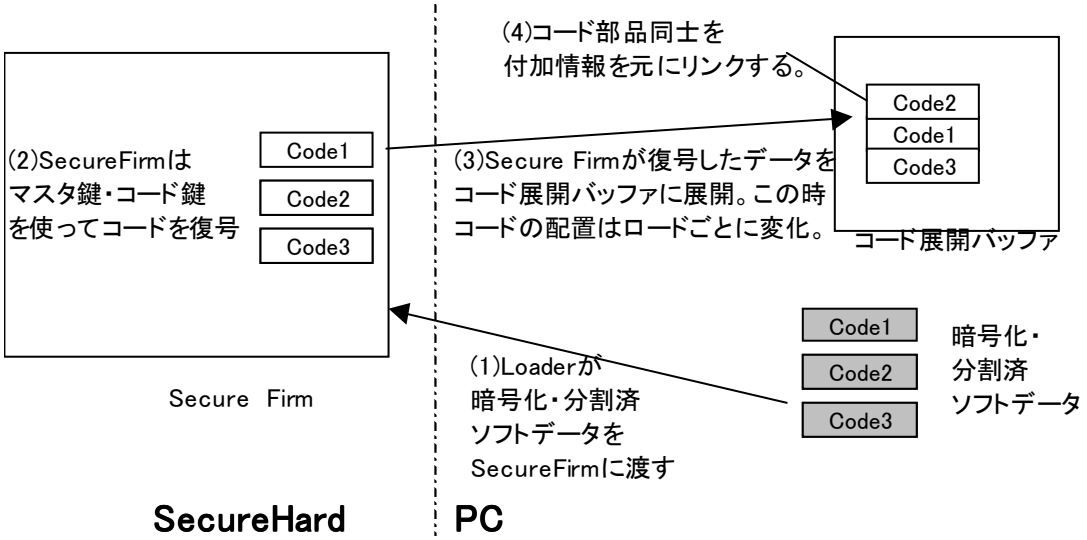


図 5-2-2 セキュアファームによるコードの復号・再構成

このシステムを運用するには、Secure Soft を HDD 上に置く暗号化された形式にエンコードするツールが必要となる。このエンコード処理を行うのが Secure Program Encoder である。

Secure Program Encoder のエンコード対象は実行されるソフトウェアモジュール、例えば実行形式ファイル（EXE）や、動的リンクライブラリ（DLL）である。Secure Program Encoder はこれらのソフトウェアモジュールを解析し、暗号化・分割済ソフトウェアデータに変換する。

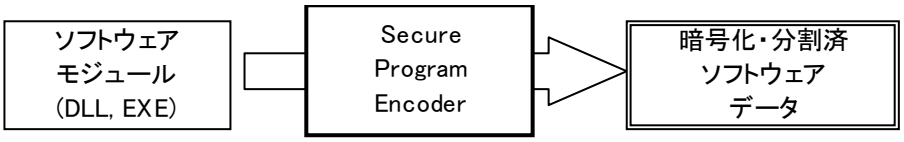


図 5-2-3 SecureProgramEncoder の機能

この Secure Program Encoder の開発では、secure soft に特化せず、一般的なソフトウェアモジュールについてエンコードできるようなツールを作成する。本書は、ソフトウェアモジュールを、Secure PC Card に渡す形に暗号化・分割するツールである Secure Program Encoder について説明する。

5-2-2 セキュアファームによる再構成について

Secure Program Encoder は、最終的にはセキュアファームに渡すデータを作成することが目的のツールである。よって、セキュアファームがどのように再構成を行うか、どういった情報が必要か、についてここで言及しておく。セキュアファームは、各ソフトウェアモジュールに対し、コード情報バッファという領域を通し、以下で説明する情報を受け取ることで再構成を行う。

５-２-２-１コード情報バッファヘッダ

Address	Bit 31-0
00-03 h	MagicNumber マジックナンバ。常に、"FJSC"を入れる。
04-07 h	DataLength コード情報バッファ全体の長さ(byte 単位)
08-0B h	NameOffset このソフトウェアモジュールの名前を格納した文字列に対するオフセット値。ソフトモジュール同士のリンク時に、ここから示される名前をベースにリンクを行う。
0C-0F h	ProcessID このコードを利用するプロセスのID。プロセスIDが一致した範囲でのみ DLL 同士をリンクする。-1 ならプロセス間共有コード。
10-13 h	CodeBufferMemoryAddressTableAddress コード展開バッファを示すメモリアドレステーブルの物理アドレス。
14-17 h	CodeBufferLogicalAddress コード展開バッファの論理アドレス
18-1B h	CodeBufferLength コード展開バッファの長さ
1C-1F h	SystemLogicalMemoryAddress
20-23 h	SystemLogicalMemorySize
24-33 h	CodeKey(128bit) コード暗号化キー（SC2000 鍵をマスター鍵で暗号化）
34-37 h	Characterisitic コードデータの種類を記述する。
38-4B h	Reserved
4C-5F h	SHA-1 コードの SHA-1 値格納領域。ファームで使用。
60-7B h	ファーム使用領域
7C-7F h	ImportTableOffset インポートテーブルに対するオフセット値。
80-83 h	ImportTableLength インポートテーブルの長さ。
84-87 h	ExportTableOffset エクスポートテーブルに対するオフセット値。
88-8B h	ExportTableLength エクスポートテーブルの長さ。
8C-8F h	SystemCallTableOffset システムコール解決テーブルに対するオフセット値。
90-93 h	SystemCallTableLength システムコール解決テーブルの長さ。
94-97 h	CodeTableOffset コードテーブルに対するオフセット値。
98-9B h	CodeTableLength コードテーブルの長さ。

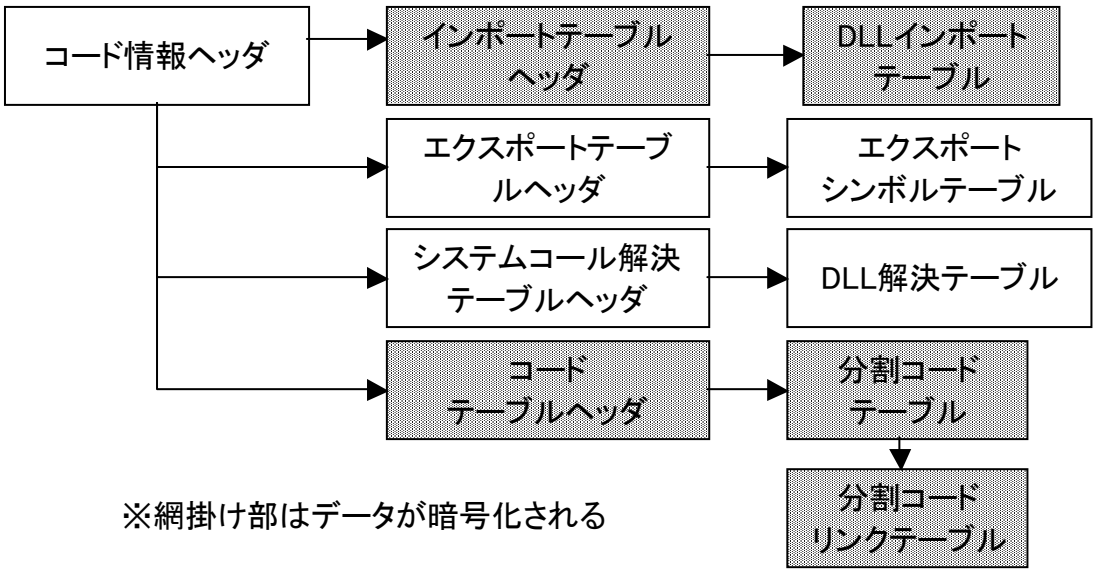
コード情報バッファは各ソフトウェアモジュール（EXE,DLL）ごとに一つ存在する。コード情報バッファは、バッファの先頭に以下のヘッダを持つ。

アドレス 00h-0Fh では他のソフトウェアモジュールとリンクするための基本情報を格納する。

アドレス 10h-1Bh の部分は、最終的にプログラムコードデータを展開する先であるコード展開バッファについての情報を記述している。

アドレス 7Ch-9Bh の部分はコード情報に含まれるテーブルがコード情報バッファのどこに存在するかを特定できるようにするための情報となっている。このヘッダ、及びそれぞれのテーブルヘッダから、それぞれのテーブルは芋づる式に取得可能である。

テーブルヘッダの参照における関係は下図のようになっている。



アドレス 24h-33h には、コードを復号するための暗号鍵を格納する。この暗号鍵は、コードを暗号化した鍵を、セキュアハード内にあるマスター鍵（外部に見せない鍵）で暗号化したものとする。

この暗号化は、インポートテーブル及び分割コードに対して行われる（上図網掛け部）。

5-2-2-2インポートテーブル

ソフトウェアモジュールは、自身で実装していないコードを外部のソフトウェアモジュールを参照する形で実行する。例えばあるプログラムでメモリを確保しようとした場合、プログラムは外部のソフトウェアモジュールである、システムのメモリ管理ライブラリを呼び出す形でメモリを確保する。現在のPC-Windows 上のソフトウェアは、このように外部のソフトウェアモジュールに依存する形で動作しており、そのためプログラムを動かすにはプログラムが参照する（インポートする）外部のソフトウェアモジュールがメモリのどこにあるかを解決する必要がある。

インポートテーブルは、ソフトウェアモジュールが他のソフトウェアモジュールからどんなシンボルを参照しているかを記述するテーブルである。ファームはインポートテーブルを元にジャンプテーブルを作成し、それをコード展開バッファに配置し、他の分割コードと結合する。

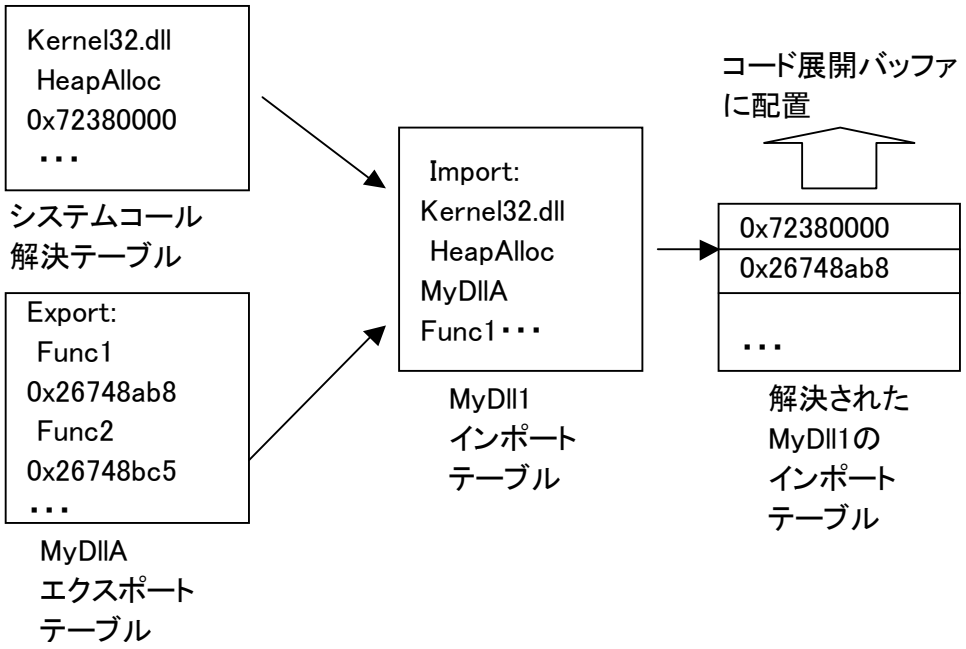


図 5-2-4 インポートテーブルの概要

5－2－2－2－1 インポートテーブルヘッダ

インポートテーブルは、テーブルバッファの先頭に以下のヘッダフィールドを持つ。

Address	Bit 31-0
00-03 h	SymbolID インポートテーブルに対応するシンボル ID
04-07 h	NumOf DLLUmportTable DLL インポートテーブルの数
08h～	DLL インポートテーブルへのオフセット。NumOfDLLImportTable の数存在。

インポートテーブルは、インポートするソフトウェアモジュール（DLL）ごとに DLL インポートテーブルを持つ。作成されたインポートテーブルは、コード再構成時にコードの一部となる。そのため、インポートテーブルは再構成時に必要なシンボル ID を持つ。

このヘッダでインポートテーブルに含まれる DLL インポートテーブルの数を特定する。また、それぞれの DLL インポートテーブルへのオフセットを特定する。

５－２－２－２－２ DLL インポートテーブル

Address	Bit 31-0
00-03 h	NameOffset インポートする DLL の名前へのオフセット
04-07 h	CodeOffset ジャンプテーブルを作成するとき、この DLL 分のジャンプテーブルをジャンプテーブルのどこに配置するかを指定する。ジャンプテーブルの先頭からのオフセット値で指定される。
08-0B h	NumOfImports この DLL からインポートするシンボルの数。
0C～	ImportOffsets インポートするシンボル名文字列へのオフセット。NumOfImports の数だけ存在。

インポートされる一つの DLL につき、一つの DLL インポートテーブルが存在する。ファームは ImportOffsets に含まれる文字列の論理アドレスを並べることでジャンプテーブルを作成する。

下図に DLL インポートテーブルの例を示す。NameOffset 先の文字列データから、インポートするのが Kernel32.dll ということがわかる。ファームはシステムコール解決テーブルまたは他に展開済みの DLL から Kernel32.dll を探す。見つかったら ImportOffset から得られる文字列データのシンボルを先頭から順に解決し、順番にジャンプテーブルに格納していく。ジャンプテーブルの作成が終わったら、インポートテーブルの先頭から CodeOffset バイトだけ進めたアドレスに、この DLL の分のジャンプテーブルをコピーする。

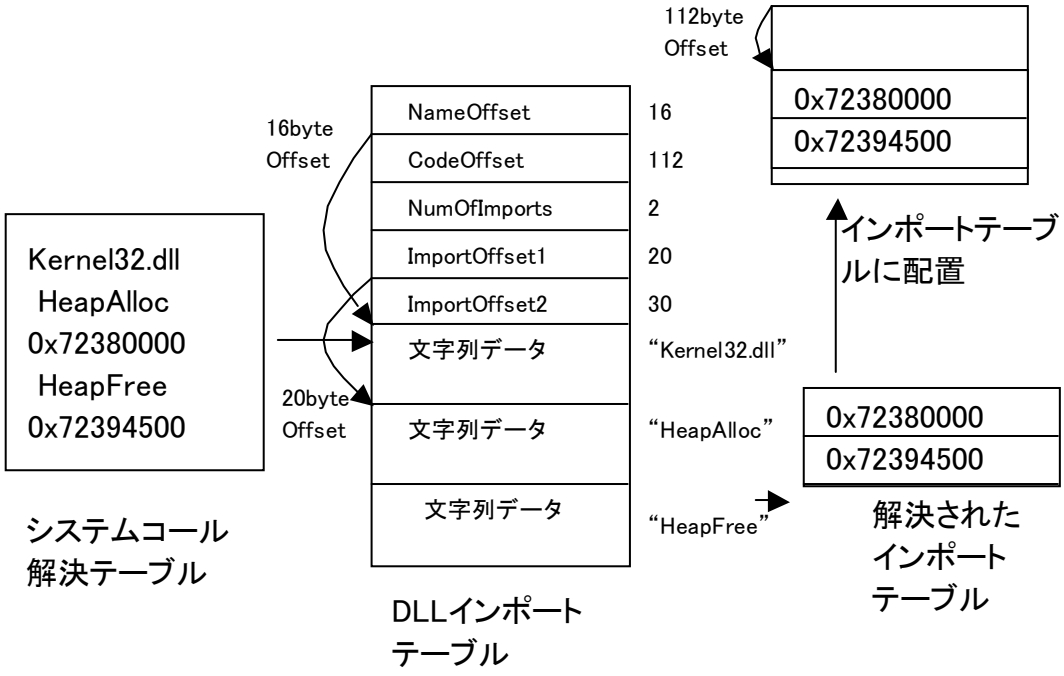


図 5-2-5 インポートテーブルの解決

5-2-2-3エクスポートテーブル

エクスポートテーブルは、このソフトウェアモジュールが他のコードに対しどんなシンボルを公開（エクスポート）するかを記述するテーブルである。ファームはコードの再構成が終わった後、エクスポート情報に基づきエクスポートするシンボルの論理アドレスを計算し、エクスポートテーブルに含まれる論理アドレスフィールドに格納する。

他のソフトウェアモジュールからこのソフトウェアモジュールをインポートする場合、このエクスポートテーブルを使ってアドレス解決を行う。

5-2-2-3-1エクスポートテーブルヘッダ

エクスポートテーブルは、テーブルバッファの先頭に以下のヘッダフィールドを持つ。

Address	Bit 31-0
00-03 h	NumOfExports エクスポートするシンボルの数
04h～	エクスポートシンボルテーブルへのオフセット。NumOfExports の数存在。

エクスポートテーブルは、エクスポートするシンボルごとにエクスポートシンボルテーブルを持つ。このヘッダでエクスポートテーブルに含まれるエクスポートシンボルテーブルの数と、それぞれのエクスポートシンボルテーブルへのオフセットを特定する。

5-2-2-3-2エクスポートシンボルテーブル

Address	Bit 31-0
00-03 h	NameOffset エクスポートするシンボルの名前へのオフセット
04-07 h	ExportAddress エクスポートするシンボルの論理アドレス。このフィールドはファームが埋める。
08-0B h	SymbolID エクスポートするシンボルを含むコードのシンボル ID。
0C-0F h	EntryOffset エクスポートするシンボルのエントリポイントアドレスの、 SymbolID のコードの先頭からのオフセット値

エクスポートされる一つのシンボルにつき、一つのエクスポートシンボルテーブルが存在する。ファームはコード再構成後、それぞれの ExportAddress フィールドを SymbolID と EntryOffset から計算し、それぞれのフィールドに格納する。

下図にエクスポートシンボルテーブルの例を示す。SymbolID と EntryOffset から論理アドレスがわかる。この例では SymbolID は 112 で、このコードは 0x72345000 に展開されている。ここに EntryOffset0x612 を足し

た値が ExportAddress になる。ファームはこれを格納する。他のソフトウェアモジュールとリンクを行う場合、ファームはこのエクスポートシンボルテーブルからシンボル名と論理アドレスを取得し、解決する。

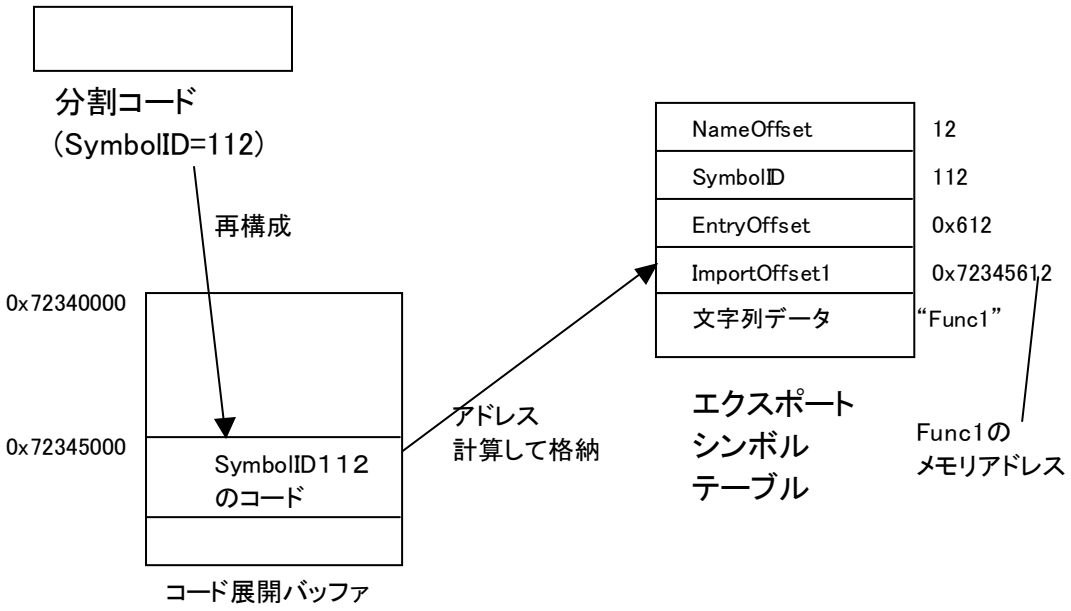


図 5-2-6 エクスポートテーブルの解決

5-2-2-4 システムコール解決テーブル

システムコール解決テーブルは、ソフトウェアモジュールがインポートするシステムコールを解決するためのテーブルである。

ファームはシステムコール解決テーブルを基に、インポートテーブルによりインポート指定されるシステム DLL アドレスを解決する。

5-2-2-4-1 システムコール解決テーブルヘッダ

システムコール解決テーブルは、テーブルの先頭に以下のヘッダフィールドを持つ。

Address	Bit 31-0
00-03 h	NumOf DLLResolveTable DLL 解決テーブルの数
04h～	DLL 解決テーブルへのオフセット。NumOfDLLIResolveTable の数存在。

システムコール解決テーブルは、解決する DLL ごとに DLL 解決テーブルを持つ。このヘッダでシステムコール解決テーブルに含まれる DLL 解決テーブルの数を特定する。また、それぞれの DLL 解決テーブルへのオフセットを特定する。

5-2-2-4-2 DLL 解決テーブル

<table>
<tr><th>Address</th><th>Bit 31-0</th></tr>
<tr><td>00-03 h</td><td>NameOffset 解決する DLL の名前へのオフセット</td></tr>
<tr><td>04-07 h</td><td>NumOfExports この DLL からエクスポートするシンボルの数。</td></tr>
<tr><td>08～</td><td>ExportInfos
エクスポートするシンボル情報へのオフセット。NumOfExportts の数だけ存在。
エクスポートするシンボル情報は、一つにつき 8byte の構造体で記述される。
構造体は以下の構造をもつ（エクスポートシンボルテーブルの先頭 8byte と等価）。
<table>
<tr><td>00-03 h</td><td>NameOffset
エクスポートするシンボルの名前へのオフセット</td></tr>
<tr><td>04-07 h</td><td>ExportAddress
エクスポートするシンボルの論理アドレス。</td></tr>
</table>
</td></tr>
</table>

DLL 解決テーブルのシンボル情報は、すべて PC 側で解決する。セキュアファームはインポートテーブルでインポート指定されているソフトウェアモジュールが Loader から渡された場合はそちらを、渡されなかった場合はこのシステムコール解決テーブルを使うことでインポートの解決を行う。

5-2-2-5 コードテーブル

コードテーブルは、ソフトウェアモジュールから実際にメモリに展開されるプログラムデータを格納するテーブルである。コードテーブル、それぞれの分割されたプログラムデータ、リンク情報はコード情報バッファヘッダに含まれる暗号鍵により暗号化された状態で渡される。

5-2-2-5-1 コードテーブルヘッダ

コードテーブルは、テーブルバッファの先頭に以下のヘッダフィールドを持つ。

Address	Bit 31-0
00-03 h	NumOf SegmentCodeTable 分割コードテーブルの数
04 h～	分割コードテーブルへのオフセット。NumOf SegmentCodeTable の数存在。

コードテーブルは、コードの分割単位ごとに分割コードテーブルを持つ。このヘッダでコードテーブルに含まれる分割コードテーブルの数を特定する。

5-2-2-5-2 分割コードテーブル

Address	内容
00-01 h	SubCodeStatus ファームで利用。
02-03 h	SubCodeControlId ファームで利用。
04-07 h	ScanFlag コードがスキャン対象かどうかを示すフラグ
08-0B h	Reserved1
0C-0F h	Reserved2
10-13 h	SymbolID 分割コードのシンボル ID
14-17 h	DataOffset 分割コードの実データへのオフセット
18-1B h	DataLength 分割コードの実データの長さ
1C-1F h	NumOfLinkTables 分割コードに含まれるリンク情報の数
10h～	リンクテーブルへのオフセット。NumOfLinkTables の数だけ存在。

コードの分割単位ごとに分割コードテーブルが存在する。分割コードテーブルは、それぞれ異なるシンボル ID を持つ。ファームはこのシンボル ID から分割コード同士のリンクを行う。ファームは DataOffset の位置にある実データを、コード展開メモリに置く。そして、リンクテーブルを基に分割コード同士のリンクを行う。

ScanFlag はこの分割コード領域をスキャン対象とするかどうかを示すフラグである。ScanFlag が 0 ならば、この分割コード領域はスキャン対象としない。ScanFlag が 1 ならば、この分割コード領域をスキャン対象とし、変更が行われたときはセキュアハードが不正検知時の処理を行う。

５－２－２－５－３ 分割コードリンクテーブル

Address	Bit 31-0
00-03 h	LinkType　リンクに使うアドレス種別。 0 ならば 絶対アドレスリンク、1ならば相対アドレスリンク。
04-07 h	PositionOffset コードデータの先頭から何 byte 目にリンクすべきアドレスデータがあるかを示すオフセット。
08-0B h	LinkSymbolID リンク対象分割コードのシンボル ID
0C-0F h	LinkOffset リンク対象分割コードの、先頭から何 byte 目のアドレスにリンクさせるかを示す。

一つのリンクごとに一つのリンクテーブルが存在する。LinkType には、そのリンクに使うアドレス種別が入る。0 ならば絶対アドレスリンクで、対象の絶対アドレスをアドレスデータに格納する。1 ならば相対アドレスリンクで、アドレスデータからの対象の相対アドレスをアドレスデータに格納する。PositionOffset には、コードの実体の先頭から何 byte 目にリンクすべきアドレスデータがあるかを示す。リンクすべきアドレスデータは、絶対アドレス、相対アドレスともに、常に 32bit とする。LinkSymbol にはリンクする分割コードのシンボル ID、LinkOffset にはリンクする分割コードの先頭からのオフセットを入れる。

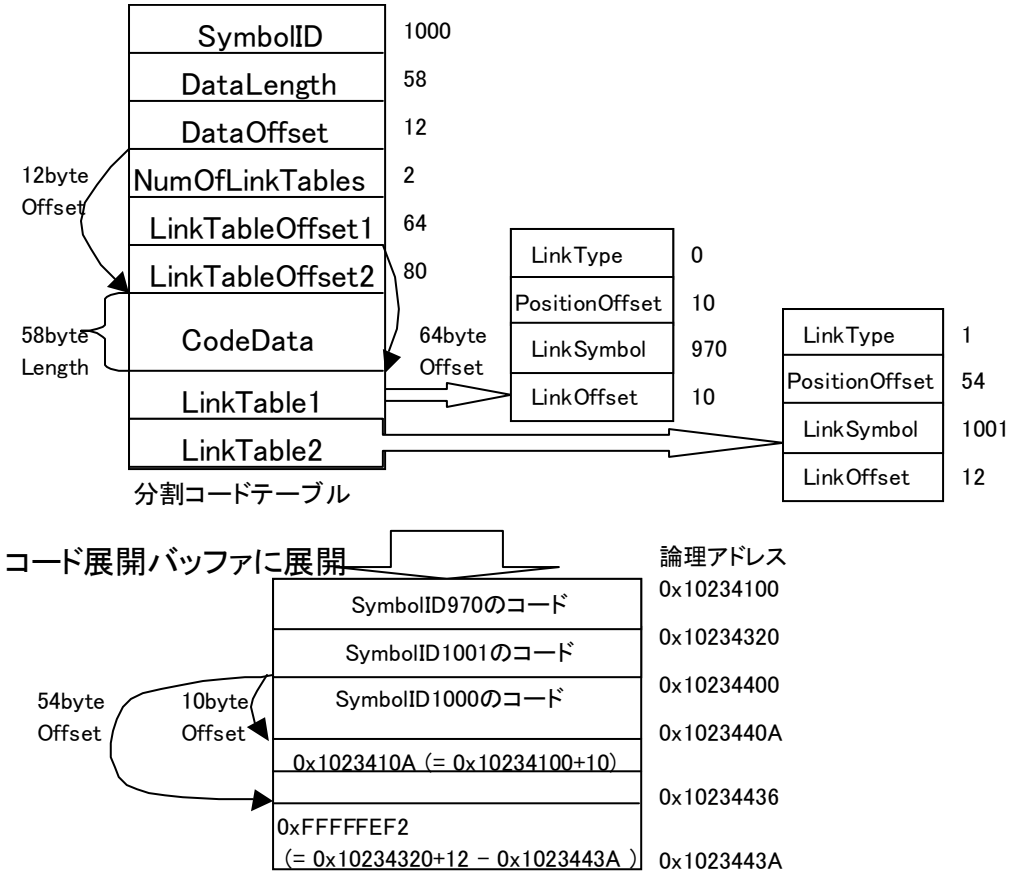


図 5-2-7 分割コードデータのリンク

図 5-2-7 に分割コードのリンク例を示す。ファームはまず、分割コードテーブルから得られるすべての分割コードデータをコード展開バッファに置き、続いて分割コードデータ同士をリンクする。

分割コードテーブルは、SymbolID が 1000 のものであり、二つのリンクテーブルを持つ。一つ目のリンクテーブルは LinkType が 0 なので絶対アドレスリンクであり、シンボル ID1000 のコードの先頭から 10byte 目のところにアドレスフィールドがある。リンク先はシンボル 970 の分割コードの 10byte 目である。二つ目のリンクテーブルは LinkType が 1 なので相対アドレスリンクであり、リンク先はシンボル 1001 の 12byte 目である。

シンボル ID1000 のコードのうち、一つ目のリンクはシンボル 970 の分割コードの 10byte 目である。この論理アドレス 0x1023410A を、指定されたアドレスフィールド、すなわちシンボル ID1000 のコードの 10byte 目に格納する。アドレス値は 32bit である。二つ目のリンクは、シンボル 1001 の 12byte 目に対する相対アドレスである。シンボル 1001 の 12byte 目の論理アドレスは 0x1023432C であり、一方呼び出しもとのアドレスフィールドの次のアドレスは、0x1023443A である。相対アドレスは 0x1023432C - 0x1023443A = 0xFFFFFEF2 なのでこれを格納する。

５-２-３ Secure Program Encoder の構成

Secure Program Encoder は概要で述べたとおり、５-２-２章で述べたセキュアファームに渡す暗号化プログラムデータ､及びその付加情報を生成するソフトウェアである。

SecureProgramEncoder の全体構成図を図 ５-２-8 に示す。

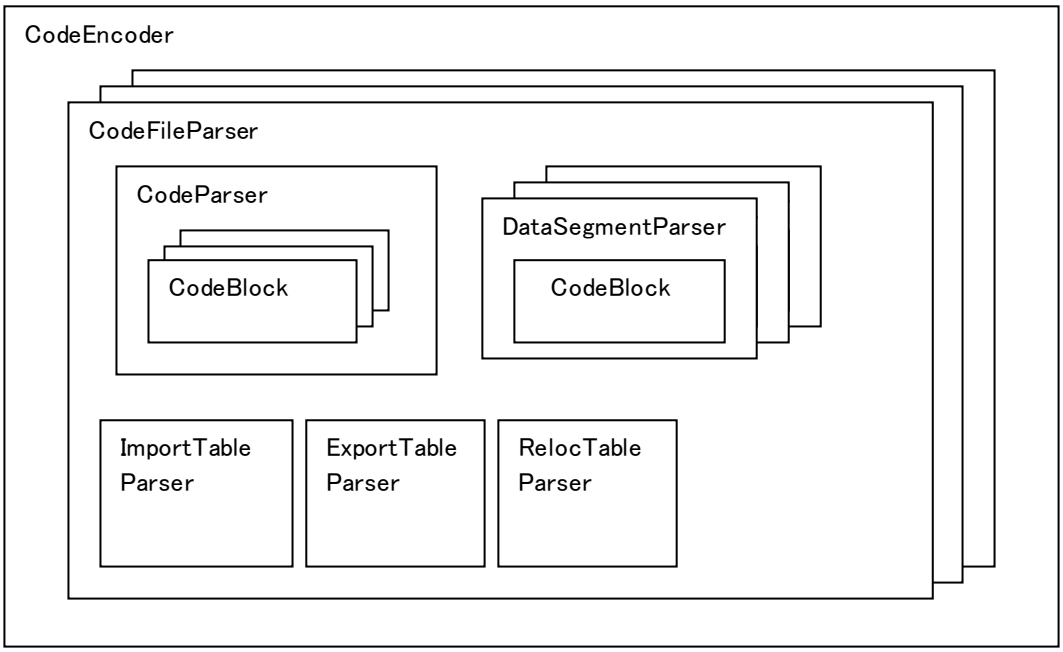


図 ５-２-8 Secure Program Encoder の構成

それぞれのオブジェクトの簡単な説明は以下のとおりである。、

CodeFileParser	ソフトウェアモジュールファイル（DLL,EXE ファイル）を解析するオブジェクト
ImportTableParser	ソフトウェアモジュールファイルに含まれるインポート情報を解析するオブジェクト。
ExportTableParser	ソフトウェアモジュールファイルに含まれるインポート情報を解析するオブジェクト。
RelocTableParser	ソフトウェアモジュールファイルに含まれる再配置情報を解析するオブジェクト
CodeParser	ソフトウェアモジュールに含まれる、テキストセグメント（プログラム実行部分）を解析するオブジェクト。プログラム実行部の分割も行う。
DataSegmentParser	ソフトウェアモジュールに含まれる、データセグメント（プログラムデータ部分）を管理するオブジェクト
CodeBlock	最終的な分割後のコードデータを保持するオブジェクト

5-2-3-1 SecureProgramEncoder のクラス構成

図 5-2-9 に、SecureProgramEncoder のクラス構成を示す。

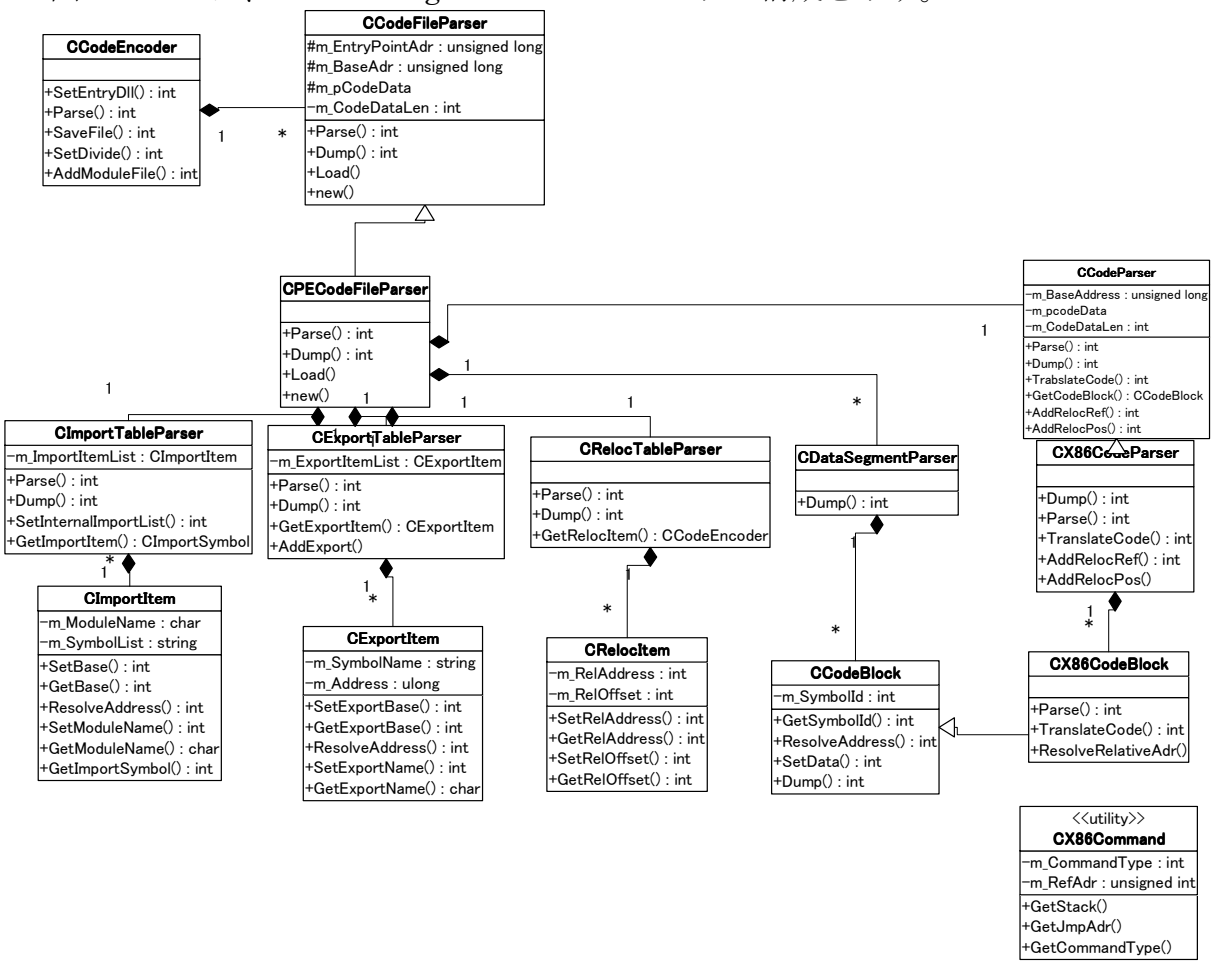


図 5-2-9 SecureProgramEncoder のクラス構成

なお、それぞれフォーマットに依存する部分を分離し、子クラスとして実装することで、将来的に他のフォーマットや命令セットにも対応できるようにしたものである。形式に依存したクラスとそのベースクラスの関係を下表に示す。

実装依存クラス	ベースクラス	概要
PECodeFileParser	CodeFileParser	CodeFileParser の機能のうち、Windows の PE 形式に依存した解析部分を実装したクラス
X86CodeParser	CodeParser	CodeParser の機能のうち、X86 命令セットに依存した解析部分を実装したクラス
X86CodeBlock	CodeBlock	CodeParser の機能のうち、X86 命令セットに依存した解析部分を実装したクラス

また、ユーティリティクラスとして以下のクラスを実装する。

クラス名	機能
CImportItem	インポート情報を管理するオブジェクト。インポートする DLL 一つにつき、一つの CimportItem オブジェクトが存在する。
CExportItem	エクスポート情報を管理するオブジェクト。エクスポートするシンボル一つにつき、一つの CexportItem オブジェクトが存在する。
CRelocItem	再配置情報を管理するオブジェクト。一つの再配置につき、一つの CrelocIem オブジェクトが存在する、
CX86Command	X86 コマンドを 1 命令単位で解析するユーティリティクラス。

5-2-3-2 SecureProgramEncoder の動作シーケンス

この Secure Program Encoder の動作シーケンスについて説明する。

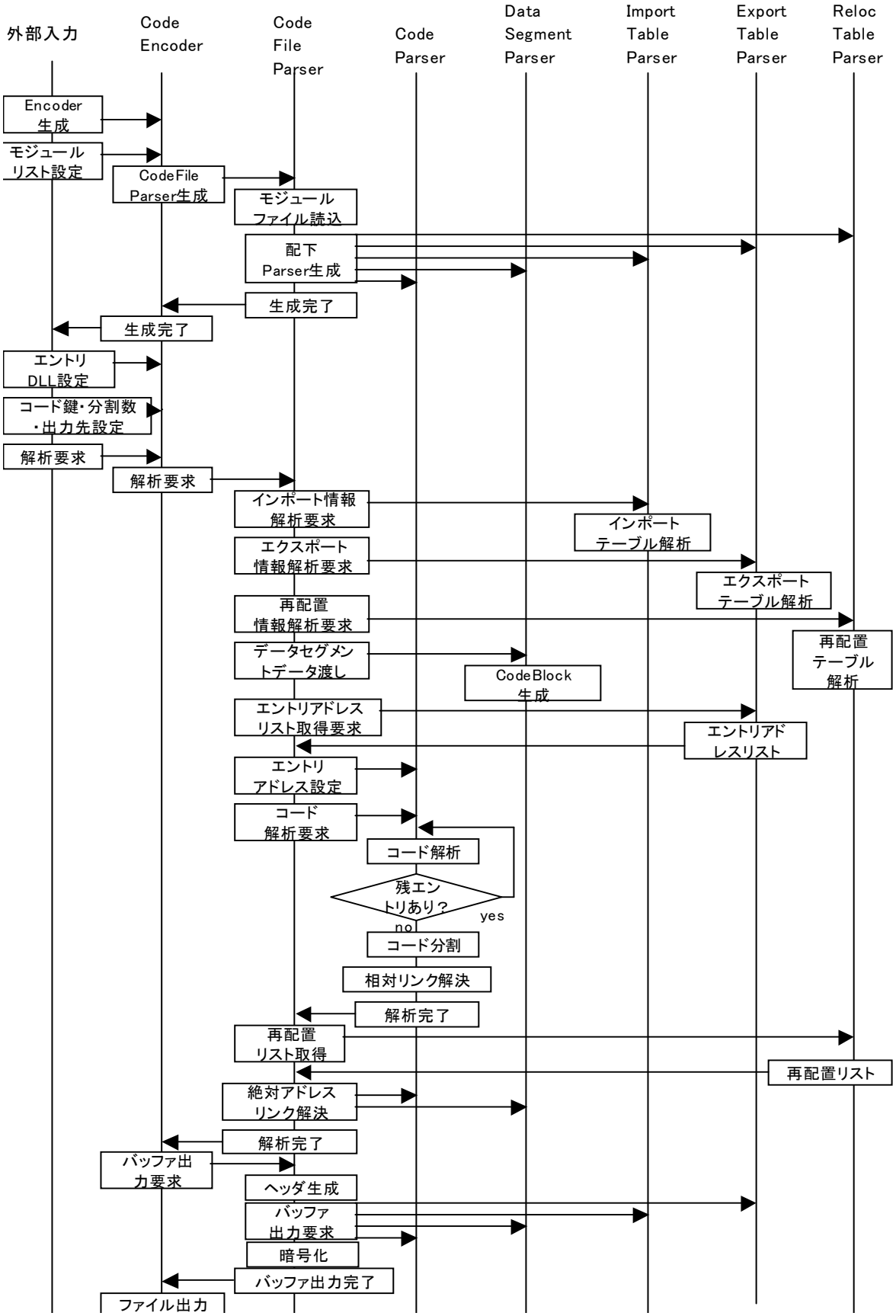
一般的に、PC のソフトウェアプログラムは複数のソフトウェアモジュールファイルから構成される。すなわち、コードの実行を開始する実行形式のソフトウェアモジュール（EXE）と、そこから利用するライブラリソフトウェアモジュール（DLL）である。Secure Program Encoder は、エンコードする単位として、一つのソフトウェアをエンコード単位としている。そこで Secure Program Encoder は、一つのソフトウェアで用いる複数のソフトウェアモジュールファイルを入力とする。また、ソフトウェアモジュールのうち一つを、プログラム実行開始時に呼び出すエントリ DLL として指定する。その他に、各コードの分割数、出力先ファイル、暗号化用コード鍵を外部からパラメータとして与える。

Secure Program Encoder は、それぞれのソフトウェアモジュールファイルに対してファイル解析を行う CodeFileParser を生成する。CodeFileParser は生成されると同時に配下の Parser を生成する。

すべてのソフトウェアモジュールに対して CodeFileParser の生成をおこなったら、続いて SecureProgramEncoder は順次それぞれの CodeFileParser に解析要求を出していくことによりソフトウェアモジュールファイルの解析を行っていく。

各ソフトウェアモジュールの解析が成功したら、各々のソフトウェアモジュールに関するコード情報バッファデータをそれぞれの CodeFileParser にバッファダンプさせていく。CodeFileParser は配下の Parser にそれぞれ必要なテーブルのバッファダンプを要求する。それぞれの Parser の出力をさせた後、それに対応したヘッダを生成し、結果を Secure Program Encoder に返す。Secure Program Encoder は、すべての CodeFileParser にバッファダンプをさせたら、最後にバッファをセキュアファイルとして保存し、処理を完了する。

次ページの図に Secure Program Encoder の動作シーケンスを示す。


外部入力
Code Encoder
Code File Parser
Code Parser
Data Segment Parser
Import Table Parser
Export Table Parser
Reloc Table Parser
Encoder 生成
モジュール リスト設定
CodeFile Parser生成
モジュール ファイル読込
配下 Parser生成
生成完了
生成完了
エントリ DLL設定
コード鍵・分割数 ・出力先設定
解析要求
解析要求
インポート情報 解析要求
インポート テーブル解析
エクスポート 情報解析要求
エクスポート テーブル解析
再配置 情報解析要求
再配置 テーブル 解析
データセグメン トデータ渡し
CodeBlock 生成
エントリアドレス リスト取得要求
エントリアド レスリスト
エントリ アドレス設定
コード 解析要求
コード解析
残エン トリあり？
yes
no
コード分割
相対リンク解決
解析完了
再配置 リスト取得
再配置リスト
絶対アドレス リンク解決
解析完了
バッファ出 力要求
ヘッダ生成
バッファ 出力要求
暗号化
バッファ出力完了
ファイル出力

５-２-４ CodeFileParser

５-２-４-１ 概要

CodeFileParser は、一つ一つのソフトウェアモジュールファイル（DLL,EXE ファイル）に対応して生成される。CodeFileParser の機能は、ソフトウェアモジュールファイルを解析し、暗号化・分割済データに変換することである。なお、今回の開発では、ソフトウェアモジュールファイルとしては Windows で用いられる PortableExecution(PE)フォーマットを対象とする。

ここでソフトウェアモジュールファイルの解析について説明する。今回対象としている Windows-PE のフォーマットも含めて、一般的にソフトウェアモジュールファイルは以下のようなフォーマットをとっている。

図 ５-２-10　ソフトウェアモジュールの一般的なフォーマット

ヘッダ情報にはソフトウェアモジュールそのものの形式や、それぞれのセグメント・情報テーブルがどこにあるかが記述される。CodeFileParser は、それぞれの情報及びセグメントに対して図 ５-２-11 のような、それぞれの情報・セグメント解析を行う Parser を生成する。

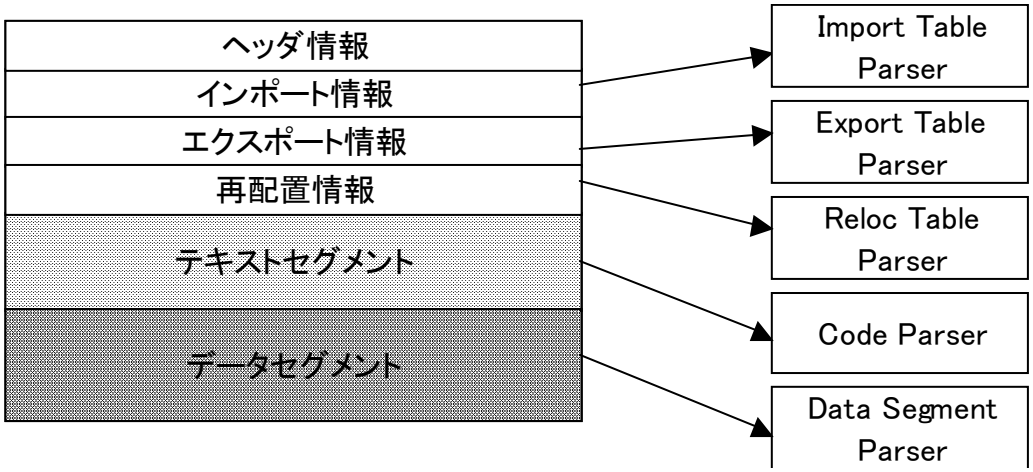


図 5-2-11 ソフトウェアモジュールファイルと各 Parser の関係

メモリ上に展開すべきプログラムデータは、テキストセグメント及びデータセグメントに置かれる。テキストセグメントには CPU に渡す命令データが記述される。一方データセグメントには、CPU に直接渡す命令データではない、プログラムの実行に必要なデータが格納される。CodeFileParser はこれらのセグメントに対し、

テキストセグメントを解析し分割処理を行う CodeParser

データセグメントを管理する DataSegmentParser

を生成する。ソフトウェアモジュール内のテキストセグメントにあるデータは CodeParser に渡され、解析及び変換処理が行われる。ここで行われる処理については 5-2-5.CodeParser の章で説明する。データセグメントのデータは DataSegmentParser が管理する。

ここまではメモリに展開すべきセグメントについて書いたが、ソフトウェアモジュールファイル内にはメモリに展開すべきデータ以外に、メモリに展開するためやソフトウェアモジュール同士をリンクするための情報が記述されている。インポート情報とエクスポート情報は、ソフトウェアモジュール同士をリンクするのに使用される(図 5-2-12)。

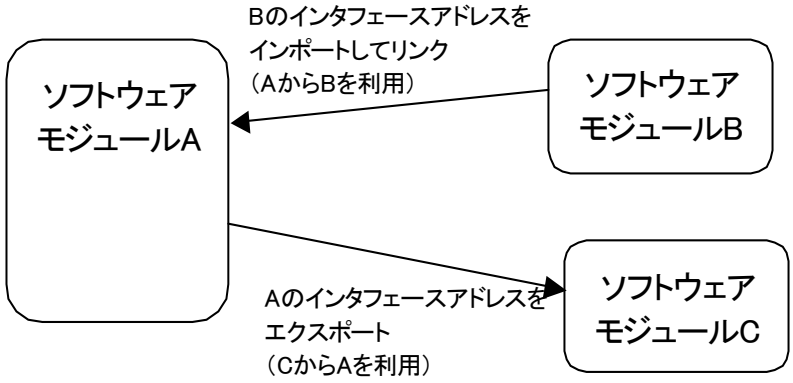


図 5-2-12 ソフトウェアモジュール同士のリンク

また、一般的にソフトウェアモジュールではメモリのどのアドレスにロードされても動作させられるように、コードの展開されるアドレスに応じてコードの一部を再配置する（Relocation）ことができるような機構が設けられている。

ソフトウェアモジュールをメモリにロードする際には、正しくメモリ上に展開し、ソフトウェアモジュール同士をリンクするためにインポート・エクスポート・再配置の情報が必要となる。そこでこれらの情報は暗号化・分割済ソフトウェアデータでも保存する（詳細は５-２-２章の再構成方法とデータ構造を参照）。

CodeFileParser は、内部的に、

・インポート情報を解析する ImportTableParser

・エクスポート情報を解析する ExportTableParser

・再配置情報を解析する RelocTableParser

を持つ。ソフトウェアモジュール内のこれらの情報は、各々の解析モジュールに渡され、Parse()メソッドを呼ぶことにより解析が行われる。

５-２-４-２ ImportTableParser

ImportTableParser は、CodeFileParser から与えられたインポート情報テーブルデータを解析するオブジェクトである。ImportTableParser は解析した各々のインポート情報をリストとして保持する。

５-２-２-２-１のインポートテーブルヘッダにあるように、エンコードされたソフトの実行時には、インポートテーブルはメモリに展開される必要があり、インポートデータも一つのシンボル ID を持つデータとして扱われる必要がある。そこで ImportTableParser は一つの CodeBlock を生成し、そのシンボル ID をインポートテーブルのシンボル ID とする。

また、SecureCodeEncoder は、CodeFileParser を通し、ImportTableParser にエンコード対象となるソフトウェアモジュールのリストを渡す。ImportTableParser は、エンコードする対象となるソフトウェアモジュール以外はシステムコールとみなし、ファイル保存時にはシステムコール解決テーブルとして保存する。

なお、システムコールの論理アドレス（DLL 解決テーブルの ExportAddress フィールド）については ImportTableParser では解決しない。この論理アドレスは、プログラム実行時に Loader により解決される。

ImportTableParser は DumpImport()メソッドにより、セキュアファームに渡すコード情報フォーマットに準拠したインポートテーブルをバッファにダンプする。同様に、DumpSysCall()メソッドによりシステムコール解決テーブルをバッファにダンプする。

５-２-４-３ ExportTableParser

ExportTableParser は、CodeFileParser から与えられたエクスポート情報テーブルデータを解析するオブジェクトである。

エクスポート情報テーブルを解析することで、どのアドレスが外部に公開さ

れたシンボルのアドレスかを知ることができる。外部に公開されたシンボルアドレスは、基本的には関数のエントリアドレスとみなすことができる。

ExportTableParser の役割は、この外部に公開されたシンボルアドレスのリストを生成することである。外部に公開されたシンボルアドレスのリストのうち、コード領域に含まれるアドレスのリストは、CodeFileParser を通して CodeParser に渡され、コード解析に利用される（詳細５-２-５-１）。

外部に公開されたシンボルアドレスは CodeParser によるコード解析・分割後、どの CodeBlock にシンボルがあるかという情報と、その CodeBlock の先頭からのオフセットという形に変換する。また、シンボルをエクスポートするにあたって、いくつか特殊なアドレスを実行時のために外部公開する。表 1 にそのリストを示す。

エクスポート名	エクスポートするアドレス
__ENTRY__	プログラムのエントリアドレス
__HEADER__	プログラムヘッダのアドレス
__RESOURCE__	リソースデータのアドレス

表 5-2-1 特殊なエクスポートアドレスのリスト

これら特殊なエクスポートアドレスについては、エクスポートテーブルを解析するだけでは知ることができない。そこで、CodeFileParser がこれらの特殊なアドレスを解決し、AddExport()メソッドにより ExportTableParser に解決したアドレスを設定する。

公開されたシンボルのアドレスリストは、DumpExport()メソッドによりセキュアファームに渡すコード情報フォーマット(に準拠した形でメモリにダンプされる。

５-２-４-４ RelocTableParser

RelocTableParser は、CodeFileParser から与えられた再配置情報テーブルデータを解析するモジュールである。ソフトウェアモジュールにおいて、絶対アドレスに基づいたコード、例えば関数コールやジャンプ、データ参照が行われた場合、必ずこの再配置情報を利用した再配置が行われる。

RelocTableParser は再配置情報を解析し、再配置が行われるコードアドレス、及びそこから参照されるコードアドレスのリストを生成する。

CodeFileParser は、この RelocTableParser から得られる再配置情報リストから、テキストセグメント・データセグメント双方に含まれる各々の CodeBlock の絶対アドレス参照を、参照する CodeBlock とその先頭からのオフセットという形のテーブルに変換する。この解決された絶対アドレス参照は、各々の CodeBlock に分割コードリンクテーブルの形で保持される。

ここで簡単にコードのリンク方法について説明する。X86 アーキテクチャでは、相対アドレスは JMP 及び Jcc(条件付ジャンプ)の時しか使用されない。よって、相対アドレスリンクは必ずテキストセグメント内を参照する。すなわ

ち、CodeParser 内で完全に解決することができる。

一方、絶対アドレスリンクはコード内のデータ領域、あるいはインポートテーブルなどを指すケースがあり、テキストセグメント内の情報だけでは解決することができない。そこで、絶対アドレスリンクについては CodeFileParser で全体を参照しながらリンク解決を行う。絶対アドレスは、ソフトウェアモジュールがロードされるアドレスにより異なってくる特性上、必ず再配置情報を必要とする。そこで絶対アドレスリンクを解決していくときは再配置情報を利用して行う。

・

5-2-5 CodeParser

CodeParserはCPUに渡す実行形式データを解析するオブジェクトである。CodeParser には、CodeFileParser からテキストセグメントのデータが渡される。プログラムコードを分割しようとした場合、データ部はどのように使用されるかを自動解析することは非常に困難であり、例えば一つの配列データを複数に分割してしまったらそのプログラムを正常に動作させることは不可能である。そこで、今回の開発ではテキストセグメントの実行命令部分に絞って分割処理を行う。

今回の開発対象となるCPUはx86アーキテクチャであるので、CodeParserでは、Intelのx86命令セット、及びその拡張命令であるMMX/SSE/SSE2の命令セットに基づいた解析を行う。CodeParserにおける解析は、

・コードの解析とエントリアドレスリスト生成
・コードの分割
・コードの変換
・絶対アドレス参照の解決

という手順で行われる。5-2-5章ではコードの変換についてまで述べる。絶対アドレス参照の解決については5-2-7章の CodeBlock に関する解説部分で述べる。

5-2-5-1 コードの解析とエントリアドレスリスト生成

5-2-5-1-1 エントリアドレスコードとエクスポートされるコードの解析

テキストセグメントは実行命令データを含むが、すべてが実行命令データというわけではない。中には本来データセグメントに置くべきデータや、ダミーデータが配置されるケースが存在する。よって、実行命令データであることが保証される部分のみ解析を行い、誤解析を防ぐ。

実行命令データであることが保証されるのは、ソフトウェアモジュールのエントリポイント及びソフトウェアモジュールがエクスポートしているアドレスである。そこで、まずこれらのアドレスから実行命令データの解析を行っていく。エントリポイントについてはCodeFileParserが、エクスポートアドレスについてはExportTableParserがそれぞれ情報を解析し、CodeParserにこれらのアドレスをデフォルトのエントリアドレスリストとして与える。CodeParserはこのエントリアドレスからデータを読み込み、順次命令の解析を行っていく。

x86 命令セットではコード読み込み位置が変化する命令が CALL、JMP、RET の３つ存在する。これらの命令を検出したら、どのアドレスにジャンプするかを解析する。解析時のルールは以下のものとする。

・ CALL あるいは JMP 命令を検出したら、CALL によりコールされるアドレス、JMP によりジャンプするアドレスを解析する。アドレス解析に成功したら、エントリアドレスリストにジャンプ先・コール先を追加する
・ ジャンプ先、あるいはコール先アドレスがコード解析で判明しなかった場合、

エントリアドレスリストへの追加は行わない。

- RET、あるいは無条件 JMP 命令を検出したらそのコードブロックは最後まで解析し終わったと判断し、与えられたエントリアドレスリストに関する解析を完了する。
- これを未処理のエントリアドレスがなくなるまで繰り返す。

解析の様子を図 5-2-13 に示す。

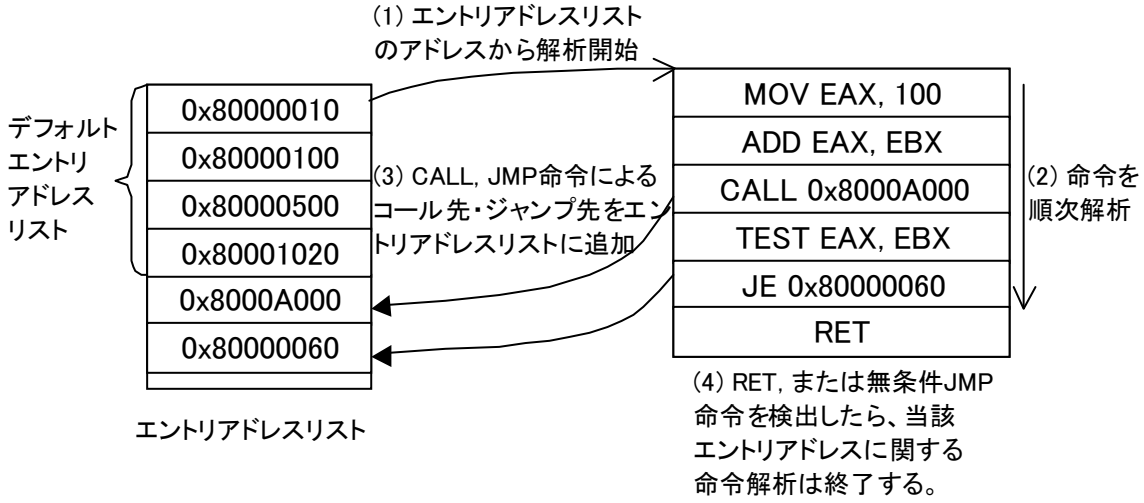


図 5-2-13 コード解析の動作

5－2－5－1－2 内部参照コードの解析

これらの解析が完了したら、次にまだ解析が完了しておらず、かつコード領域の可能性がある部分のコード解析を行う。これは、5-2-5-1 で解析したエントリアドレスあるいはエクスポートアドレスから参照されるコードの解析だけでは、コールバック関数など、アドレスを一旦変数として扱い、後でそれを呼び出すような部分のコード領域が解析できないためである。

エントリポイントの可能性があるかどうかの判定条件は、

・再配置テーブルにより、参照されるアドレスである。

・コード領域に含まれ、かつまだコードの解析が済んでいない領域に含まれる。

という条件で行う。これは、再配置可能なコードの場合、関数の呼び出し先アドレスに関しては再配置が必要であり、その場合再配置テーブルにより参照されるアドレスとなるためである。

これらのコード領域候補に対し、CodeParser は順次コード解析を行っていく。解析を行った際、

- コードとしての解析結果がおかしい
- 先にコードと特定した内容と一致しない

場合は、解析を行った部分をコード領域とみなさない。逆にこれを満足したら、解析した領域をコード領域と判定し、エントリアドレスリストに解析した部分を

本開発におけるコード解析は呼び出し先コードのアドレスが再配置情報により参照されることを前提としているため、参照されるアドレスに対し、何らかの演算をほどこして呼び出し先コードのアドレスを決めるような難読化が行われた場合、正常にコードの解析が行えない可能性がある。この対応については今後の課題とする。

5-2-5-2 コードの分割

コードの解析が終わったら、次にコードの分割を行う。コードをいくつに分割するかは、外部から与えられるパラメータに従う。CodeParser は 5-2-5-1 での解析結果から、

- コード解析が行われている領域の
- 一つの命令を二つに分断しない

アドレスで、コードを分割していく。分割時は、それぞれの分割されたデータに対し CodeBlock オブジェクトを生成していく。

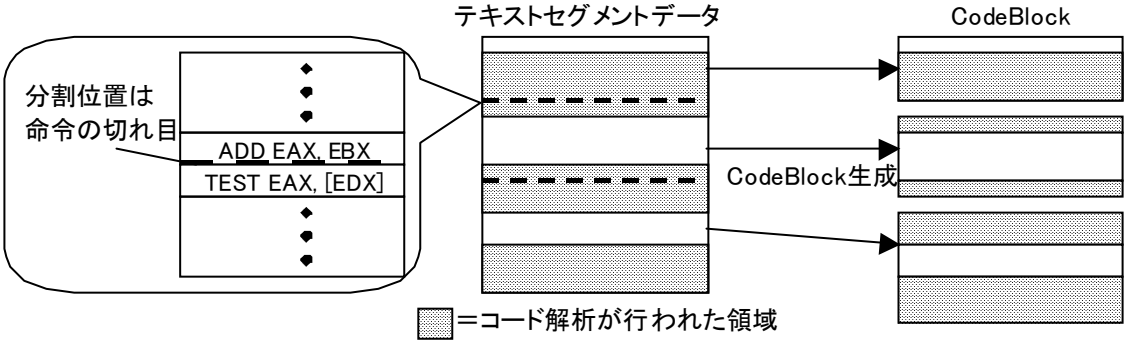


図 5-2-14 コードの分割

分割されたコードは、そのままでは分割されたコード間でのジャンプやコールがうまく処理できない。そこで次にコード変換を行って正しいコードに変換する。

5-2-5-3 コードの変換

分割された各々の CodeBlock は、プログラム実行時にはセキュアファームによりメモリ上にばらばらに配置される。よって、CodeBlock のコードデータはそのままでは元コード通りに動作しない。そこで、元コード通りに動作させるため、相対アドレスリンクの解決とコード変換を行う。なお、絶対アドレスリンクについては、後に CodeFileParser から各 CodeBlock の絶対アドレス変換を行う。

セキュアファームは相対アドレスリンク情報（5-2-2-5-3 参照）を利用して分割された CodeBlock 同士の相対アドレスを解決することができる。そこでここでは、最終的にファームで必要とする相対アドレスリンクテーブルを作成する。コード変換処理と相対アドレスリンクの解決処理の様子を図 5-2-15 に示す。まず必要なのは、分割された CodeBlock 同士を連結すること

である。そこで、分割されたコードの末尾に、次のコードブロックへの JMP 命令を追加し、同時にその JMP 命令に関する相対アドレスリンク情報をテーブルに追加する。また、分割された CodeBlock をまたぐ相対アドレスジャンプがあった場合、それを相対アドレスリンク情報として保存していく。

セキュアファームによる相対アドレスリンクは 32bit 相対値によるリンクのみサポートされる。よって CodeBlock をまたぐような 8bit/16bit 相対アドレスジャンプ命令は、32bit 相対アドレスを用いるように 32bit 相対アドレスジャンプに変換する。また、このコードの変換によりコード長が変化する場合があるので、コード長の変化情報を保存すると共に、コード長の変化のために相対アドレスが変化した場合、新しい相対アドレスに合わせて命令を変換する。

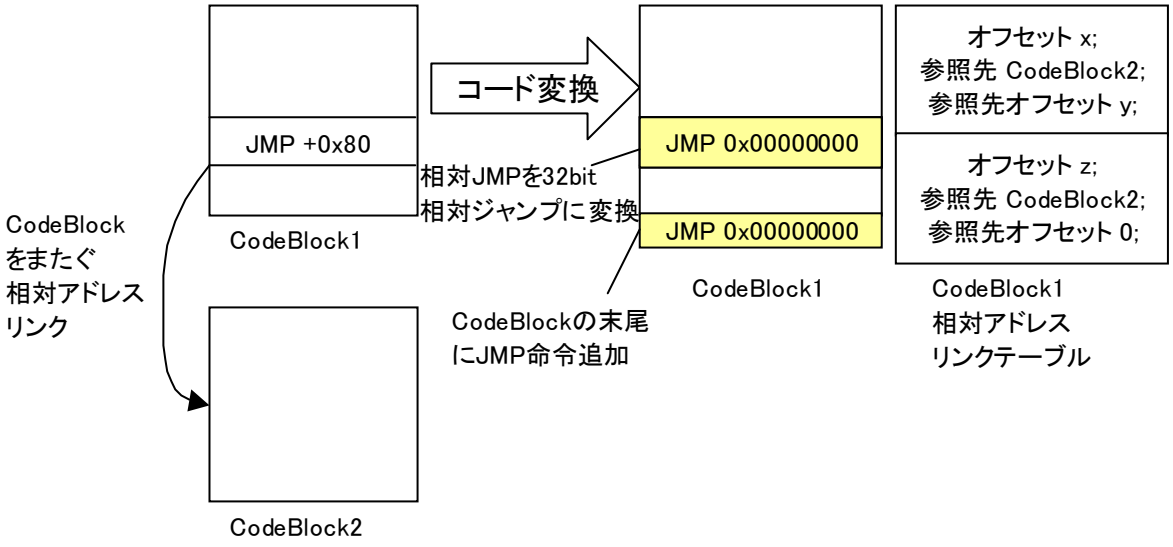


図 5-2-15 コード変換と相対アドレスリンクの解決処理

5-2-6 DataSegmentParser

DataSegmentParser は各々のデータセグメントに対応して生成される。データセグメントにおいては、データが置かれるが、そのデータがどのように利用されるかを解析するのは難しく、基本的に分割は不可能である。よって、データセグメントのデータは解析・分割は行わない。

DataSegmentParser は渡されたデータセグメントデータに対し、一つの CodeBlock を生成し、データセグメントのデータをそのまま CodeBlock に渡す。すなわち、一つのデータセグメントに対して一つの CodeBlock が生成されることになる。

データセグメントには絶対アドレスによって変化するような情報も置かれるが、それについては後に CodeFileParser が再配置情報を利用してまとめて変換・解決する。

データセグメントには、基本的には相対アドレスを利用したデータは基本的には含まれない。厳密に言えばデータセグメントに相対アドレスを利用したデータを置いておき、アドレスをコード部で計算し、ジャンプやコール、データ参照を行うことは可能ではある。しかし、それがただのデータなのか、相対アドレスを使ったデータなのかを判別することは非常に困難である。そこで今回はこのような相対アドレスに基づいた計算が行われた場合は Secure Program Encoder の適用対象外とする。

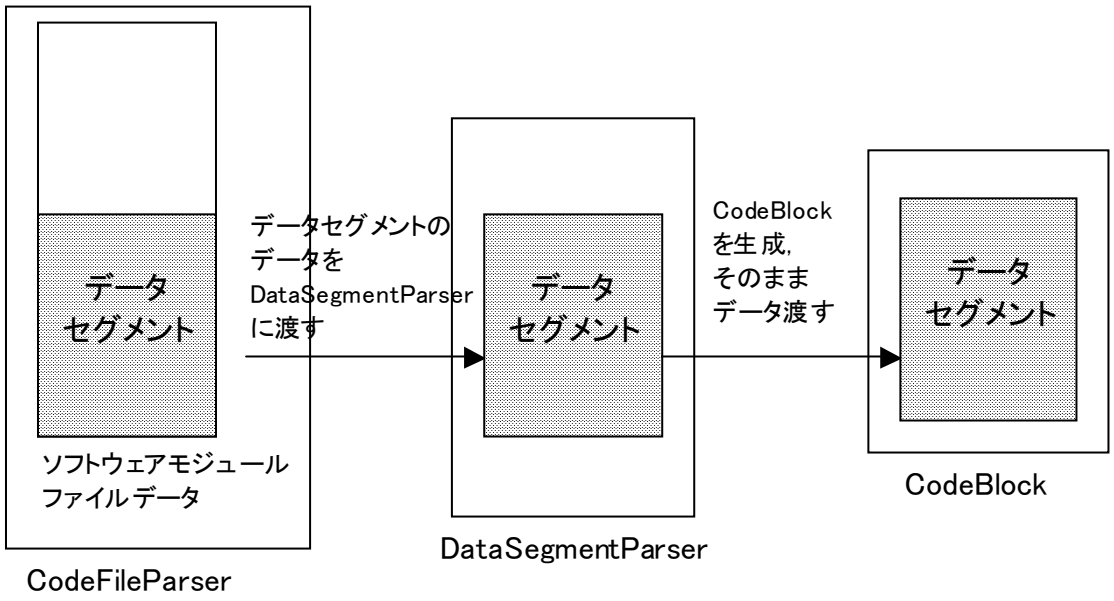


図 5-2-16 データセグメントに対応する CodeBlock の生成

5 - 2 - 7 CodeBlock

CodeBlock は、最終的にメモリに展開されるプログラムイメージ及びリンク情報を保持するオブジェクトである。ファームに渡す情報に対応付ければ、2.5 で述べたコードテーブルに対応する情報を保持する。

CodeBlock は各々独立したシンボル ID を持つ必要がある。そこで静的変数を一つ用意し、CodeBlock が生成されるたびにそれをインクリメントしていくことで、各々の CodeBlock に別々のシンボル ID が割り当てられるようにする。

CodeBlock は最終的にセキュアファームに渡されるプログラムデータを保持しており、これら分割されたコードデータは相互に参照される場合リンク情報によりその解決を行う必要がある。リンク情報のうち、5 章で述べた CodeParser の解析により相対アドレス参照の解決はできている(DataSegment には相対アドレス参照は基本的に存在しない)。そこであとは絶対アドレスを解決すればコードは実行可能となる。

絶対アドレス参照については、Windows がプログラムをロードして実行する時にもロードされるアドレスにより絶対アドレスが変わってくるため、必ず再配置情報を伴う。そこで、Reloc Table Parser の持つ再配置情報リストを使用して絶対アドレス参照の解決を行う。

まずそれぞれの CodeBlock は、本来自身が配置されるべきメモリアドレスと、ブロック長を属性として持つ。一方で、それぞれの再配置情報は、再配置が行われる位置と、その絶対アドレス参照先を情報として持っている。そこで、一つ一つの再配置情報について、まずその再配置が行われるアドレスがどの CodeBlock に含まれているかを調べる(1)。次に、その参照先がどの CodeBlock に含まれるか、またその CodeBlock の先頭からどれだけのオフセットの位置にあるかを調べる(2)。そこで得られた参照先のシンボル ID とオフセットを計算し(3)、再配置を含む CodeBlock に絶対アドレス参照として追加する(4)。これをすべての再配置情報について繰り返すことですべての絶対アドレス参照を解決する。

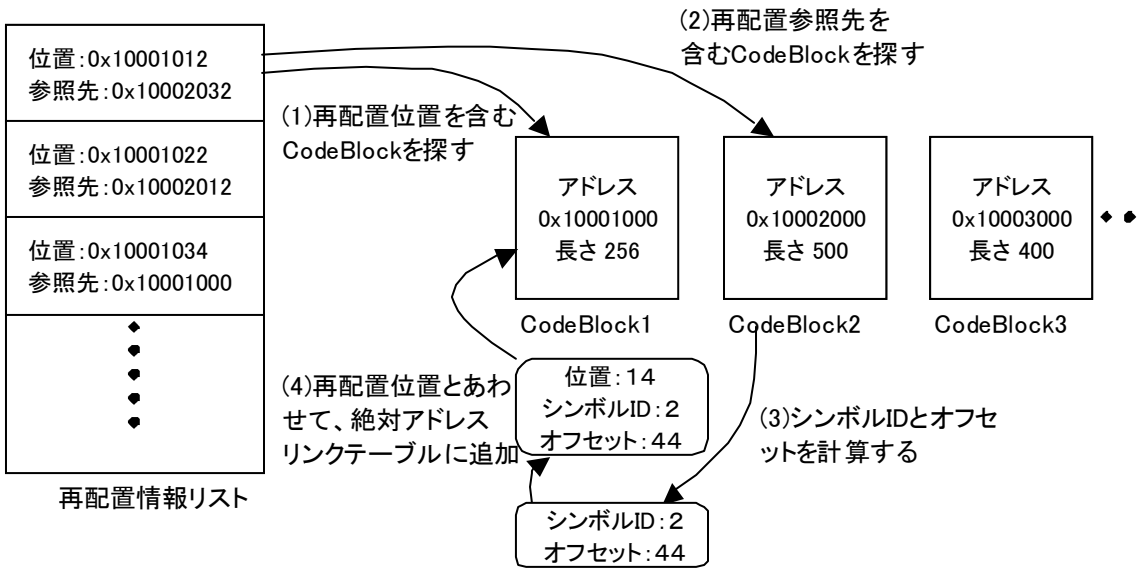


図 5-2-17 CodeBlock に含まれる絶対アドレス参照の解決

5-2-8 出力ファイルフォーマット

SecureProgram Encoder は、最終的に以下の形式のファイルを出力する。出力フォーマットは、以下のとおりである。

Address	Bit 31-0
00-03 h	MagicNumber マジックナンバ。常に、"FJSP"を入れる。
04-07 h	EntryDLLNameOffset エントリ DLL の名前を格納した文字列に対するオフセット値。
08-0B h	NumOfSoftModules このプログラムに含まれるソフトモジュールの数
0C-0F h	Reserved
10 h ～	SoftModuleOffsets 各々のソフトモジュールに対応するコード情報へのオフセット値。 NumOfSoftModules 個存在

それぞれのソフトモジュールのコード情報は、CodeFileParser が作成する。コード情報のフォーマットは、5-2-2 章で述べたコード情報フォーマットに従う。基本的にそれぞれのフィールド・テーブルは CodeFileParser により正しい値が設定され、暗号化が必要な部分は暗号化される。ただし、コード情報のうちでも以下のフィールドについては、ファイルにダンプした時点では情報を格納しない（Loader がロード時にそれらのフィールドを埋める）。

● ヘッダ情報内
  - ProcessID
  - CodeBufferMemoryAddressTableAddress

・ CodeBufferLogicalAddress

● システムコール解決テーブル内 DLL 解決テーブル

・ ExportInfos に含まれる、ExportAddress。

これら以外のフィールドについては CodeFileParser、あるいはその配下の Parser オブジェクトが埋め、暗号化をかけた形でデータをファイルに保存する。

5-2-9 コードスキャン

セキュアハードは、ＰＣメモリ上のコードデータをスキャンして改変されていないかチェックすることで、動作しているソフトウェアの正当性を保証する。

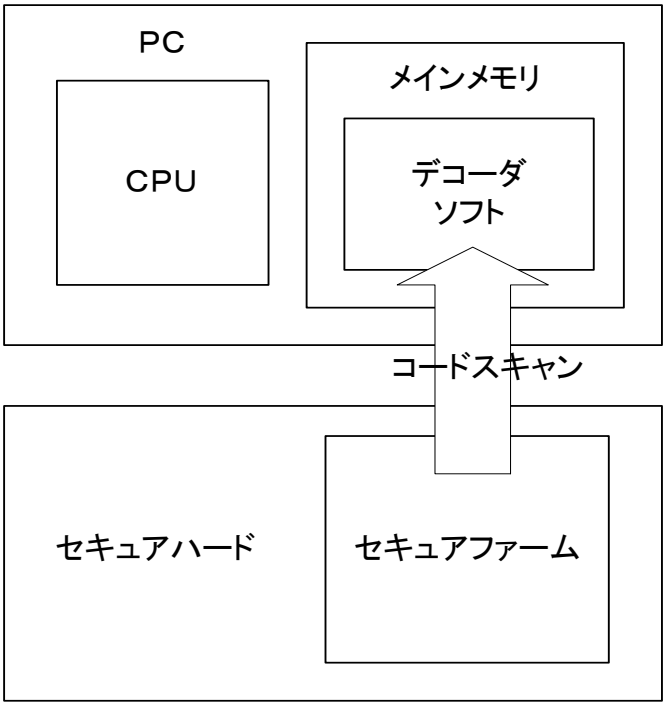


図 5-2-18 コードスキャン

図 5-2-18 コードスキャン にコードスキャンの概要を示す。セキュアハード上のセキュアファームは、ＰＣメモリ上のプログラムコードを常時監視しており、プログラムコードに対する改竄を検出する。具体的には、セキュアファームがＤＭＡにより、実行中のプログラムをセキュアハード側に取り込み、そのプログラムコードが起動時のプログラムコードと一致しているかどうかを調べることによって行う。起動時のプログラムコードと一致しない場合は改竄があったと判断でき、セキュアハードの機能を停止させる。

５-２-１０リアルタイム認証

セキュアハードとセキュアソフトは定期的に認証データのやり取りを行い、ＰＣ上で動作しているソフトウェアの正当性を確認する。リアルタイム認証に関しては次章で述べる。

５－３　ソフト研究開発の概要

５-３-１はじめに

Secure PC System は、Secure PC Card と MPEG2 ソフトデコーダにより構成され、難読化,暗号化,プログラムコード/データの再構成,リアルタイム認証等により、BS デジタル放送などで使用される MPEG2 データ(MP@HL)の復号処理をセキュアに実施することを目的とした装置である。Secure PC System は、MPEG2 TS データを暗号化・復号化を行う secure hard,プログラムコード再構成を行う secure firm を搭載する Secure PC Card と MPEG2 ソフトデコードおよび Secure PC Card 制御を行う secure soft より構成される。

secure soft は、Secure PC Card の制御を行い、secure decoder とのインタフェースを行うドライバ,secure decoder プログラムをメモリ(スワップ不可領域)に展開し、secure decoder を起動する指示を行うローダ,ユーザとのインタフェースを行う GUI および MPEG2 TS デコード,映像/音声出力を行う secure decoder より構成される。

本章では、MPEG2 ソフトデコーダ機能を搭載する secure decoder について説明する。

５-３-２　構成

secure decoder は MPEG2 TS デコード,映像/音声出力を行う機能を実現する。図 5－3－1 に secure decoder 構成図を示す。

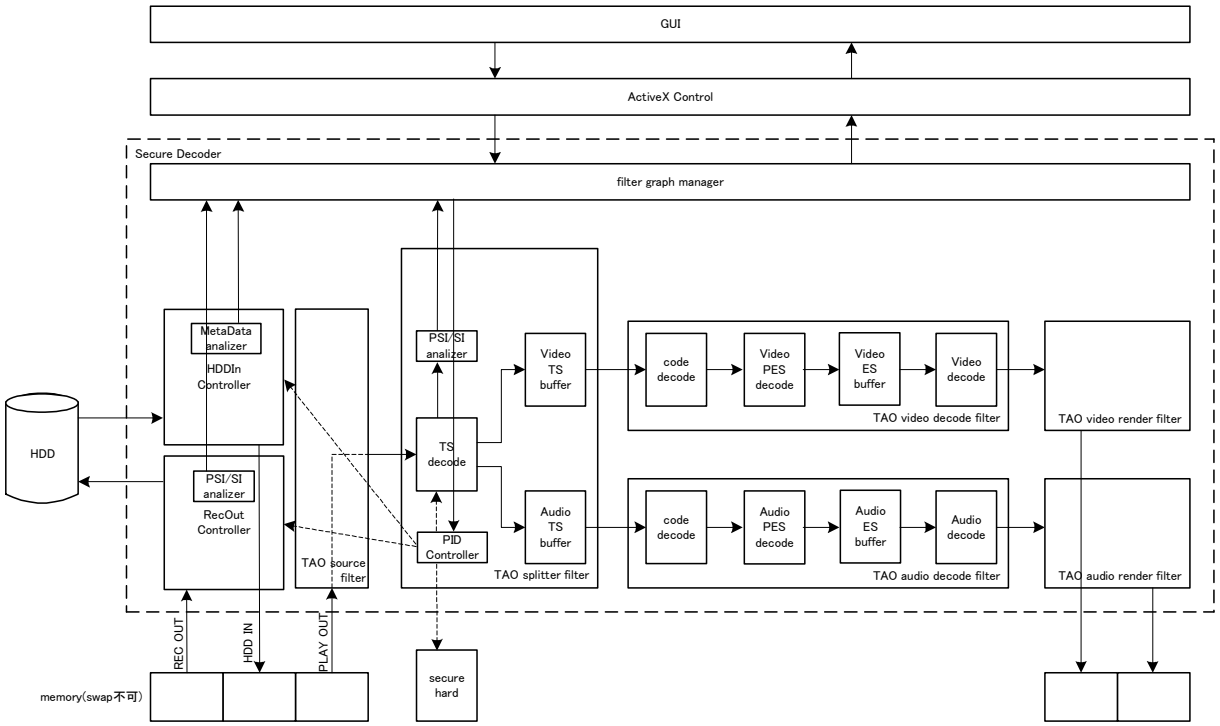


図 5－3－1　　secure decoder 全体構成

図 5－3－1 に示すように、MPEG2 デコーダ機能は、direct show を用いたフィルタグラフにより構成される。以下に、各フィルタの詳細について説明する。

5-3-2-1　TAO source filter

TAO source filter は、TAO splitter filter からのデータリード要求により、secure hard から MPEG2 TS データ(PLAY OUT)の読出処理を行う。

REC OUT,HDD IN は、secure controller 管理の REC OUT thread, HDD IN thread により制御する。

5-3-2-2　TAO splitter filter

TAO splitter filter は、MPEG2 TS データの TS ヘッダ解析および PSI/SI 情報の収集を行う。(PES データ解析は TAO video decode filter, TAO audio decode filter にて行う。)

TS ヘッダ解析を行うことにより、対象となる暗号化された Video TS payload,　Audio TS payload(TS ヘッダ, Key 情報付き)をそれぞれ TAO video decode filter,　TAO audio decode filter へ送出する。

PSI/SI 解析により、PAT,CAT,PMT,ECM,EMM 等の番組情報を収集し、GUI, secure hard とインタフェースする。

図５－３－２に、TAO splitter filter のクラス構成を以下に示す。

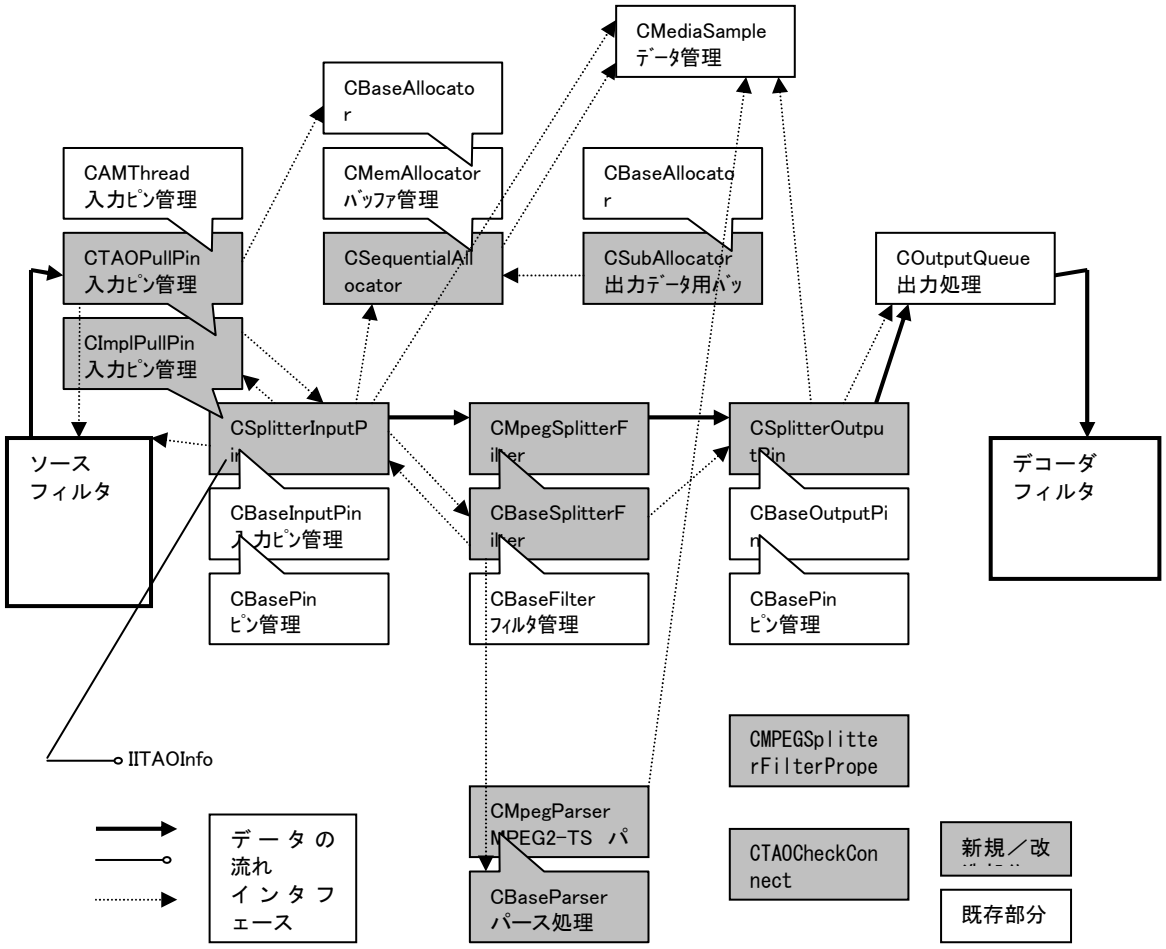


図５－３－２　TAO splitter filter クラス構成

【TS ヘッダ解析】

TS ヘッダ解析において、sync byte, transport error indicator, payload unit start indicator, continuity counter 等よりエラー検出した際には、復帰処理(警報通知処理含む)を行い、復帰の際にはダウンストリームフィルタ内に残っているメディアデータはフラッシュする。尚、TAO splitter filter 内 CsequentialAllocator にて管理されている入力用メディアサンプル情報は、TS ヘッダエラー発生した TS パケットを含む入力用メディアサンプル情報のみフラッシュする。

【PSI/SI 解析】

PSI/SI 解析は、PAT,SDT,EIT より番組テーブルを作成/GUI 表示し、ユーザが GUI 上で選択した番組に対する TS パケットの PID を生成/設定する。尚、secure decoder は、起動時に secure hard に対して PAT,CAT,SDT,EIT はスルー設定を行う。

以下に、PSI/SI による番組テーブル生成,番組選択および加入時/契約更新における解析フローを以下に示す。

尚、PLAY OUT に対する PSI/SI 解析は TAO splitter filter, REC OUT に対する PSI/SI 解析は、RecOut thread にて実施し、その解析結果を基にして、TAO secure controller にて GUI に対して、放送番組テーブル表示を行う。(蓄積番組テーブルは、HDDIn thread にて HDD 内メタデータより生成する。)

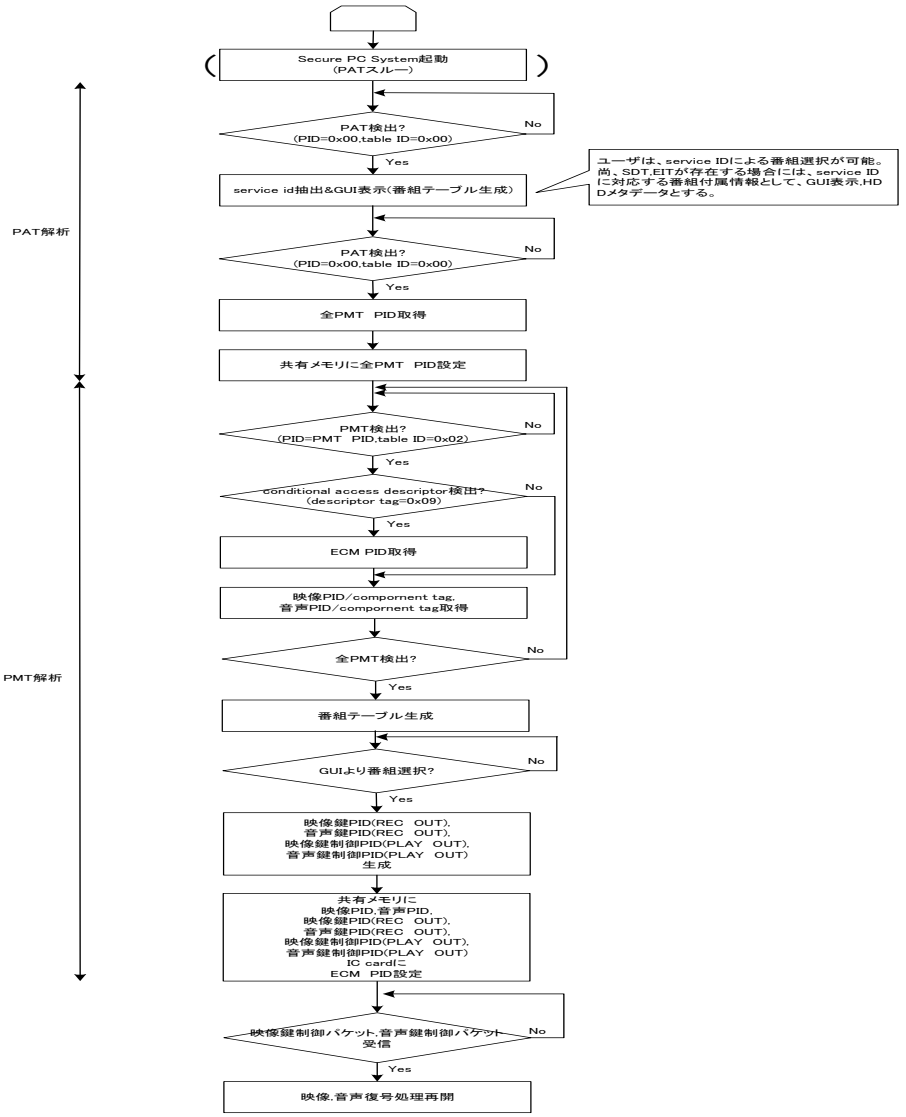


図５－３－３　番組テーブル生成,番組選択における PSI/SI 解析フロー

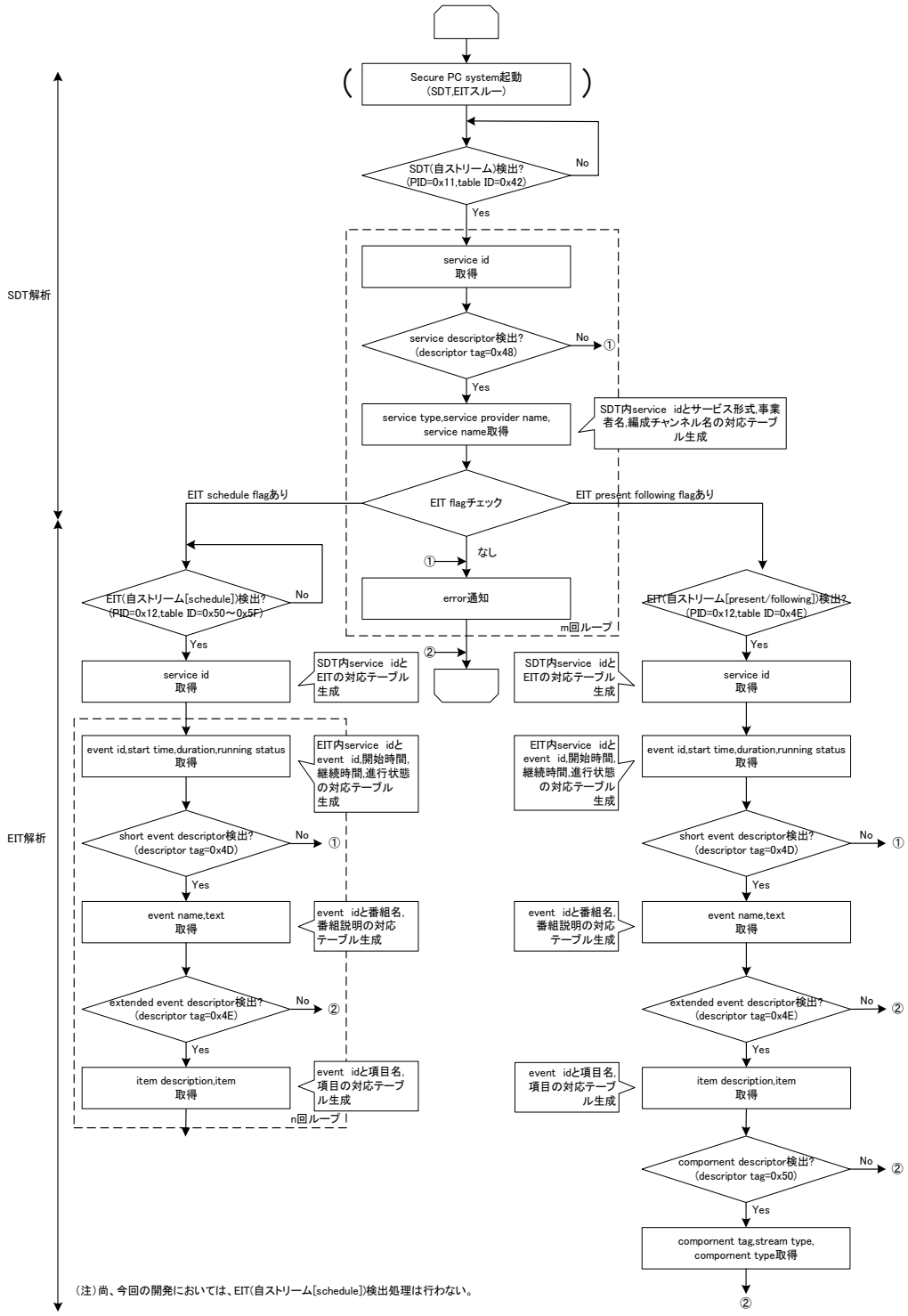


図５－３－４　番組テーブル(付属)生成における PSI/SI 解析フロー

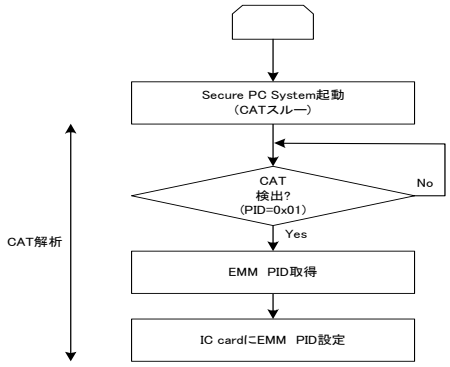


図５－３－５　加入/契約更新における PSI/SI 解析フロー

【PID 制御】

Secure PC System は、現状では 10 種類の処理ルートをサポートする。TAO splitter filter による PSI/SI 解析または、HDD 内メタデータ情報をもとにして、各処理ルートにおける暗号化/暗号復号化,PID フィルタの PID 制御を行う。詳細を下記に示す。

TS パケット種類	PID 生成方法	設定箇所	処理ルート										
PAT PID	0x00 固定。	H/S											
SDT PID	0x11 固定。	H/S											
EIT[p/f] PID	0x12 固定。	H/S											
CAT PID	0x01 固定。	H/S											
PMT PID	secure decoder が PAT パケット内の選択番組の program number に対応する PID 抽出。	H/S											
ECM PID	secure decoder が PMT パケット内 Conditional access descriptor より PID 抽出。	H/S											
EMM PID	secure decoder が CAT パケット内 Conditional access descriptor より PID 抽出。	H/S											
MULTI2 PID(映像)	secure decoder が secure hard,HDD から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、MULTI2 暗号復号化する選択番組の映像 PID を設定。	H/S											
MULTI2 PID(音声)	secure decoder が secure hard,HDD から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、MULTI2 暗号復号化する選択番組の音声 PID を設定。	H/S											
HS PID(映像) (REC OUT)	secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、SC2000 暗号化/HDD 格納する選択番組(save)の映像 PID を設定。	H/S											

HS PID(音声) (REC OUT)	secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、SC2000 暗号化/HDD 格納する選択番組(save)の映像 PID を設定。	H/S											
HS_key PID(映像) (REC OUT)	secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成。	H/S											
HS_key PID(音声) (REC OUT)	secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成。	H/S											
HD PID(映像) (HDD IN)	＜Play Only＞ secure decoder が secure hard,HDD から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析または､HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。	H/S											
HD PID(音声) (HDD IN)	＜Play Only＞ secure decoder が secure hard,HDD から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析または､HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。	H/S											
HD_key PID(映像) (HDD IN)	＜Play Only＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成または、HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像鍵 PID を抽出	H/S											
HD_key PID(音声) (HDD IN)	＜Play Only＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,	H/S											

	音声および reserved PID 以外の値をユニークな値を生成。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成または、HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の音声鍵 PID を抽出												
SS/SD PID(映像) (PLAY OUT)	＜Play Only＞ secure decoder が secure hard,HDD から入力された TS ストリームの PSI/SI 解析により、SC2000 暗号化/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。 ＜Play&Save＞ secure decoder が secure hard,HDD から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、SC2000 暗号化/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。	H/S											
SS/SD PID(音声) (PLAY OUT)	＜Play Only＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。	H/S											
SS/SD_keycont PID(映像) (PLAY OUT)	＜Play Only＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成または、HDD 内メタデータにより、SC2000 暗号化/暗号復号化する選択番組(play)の映像鍵制御 PID を抽出	H/S											
SS/SD_keycont PID(音声) (PLAY OUT)	＜Play Only＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値をユニークな値を生成。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、PMT,ECM,EMM,映像,音声および reserved PID 以外の値	H/S											

	をユニークな値を生成または、HDD 内メタデータにより、SC2000 暗号化/暗号復号化する選択番組(play)の音声鍵制御 PID を抽出												
DEC PID(映像)	＜Play Only＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、HDD 読出/SC2000 暗号復号化する選択番組(録画)の映像 PID を設定。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。	S											
DEC PID(音声)	＜Play Only＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析により、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。 ＜Play&Save＞ secure decoder が secure hard から入力された TS ストリームの PSI/SI 解析または、HDD 内メタデータにより、HDD 読出/SC2000 暗号復号化する選択番組(play)の映像 PID を設定。	S											

① Play Only, IN:SecureHard, TimeShift:OFF
この処理モードは、secure hard に入力された TS データをタイムシフト機能なしで、secure decoder で視聴を行う際に用いられる。
尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

③ Play Only, IN:SecureHard, TimeShift:ON, MULT2:OFF
この処理モードは、secure hard に入力された TS データをタイムシフト機能ありで、secure decoder で視聴を行う際に用いられる。
尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

④ Save Only, IN:SecureHard, MULTI2:ON
この処理モードは、secure hard に入力された TS データを HDD へ保存する際に用いられる。(保存データ形式：SC2000+MULTI2)

⑤ Save Only, IN:SecureHard, MULTI2:OFF
この処理モードは、secure hard に入力された TS データを HDD へ保存する際に用いられる。(保存データ形式：SC2000)

⑥ Play&Save, IN:SecureHard, TimeShift:OFF, MULTI2:ON
この処理モードは、secure hard に入力された TS データをタイムシフト機能なしで、secure decoder で視聴しながら、HDD に保存する際に用いられる。(保存データ形式：SC2000+MULTI2)

尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

⑦ Play&Save, IN:SecureHard, TimeShift:OFF, MULTI2:OFF
この処理モードは、secure hard に入力された TS データをタイムシフト機能なしで、secure decoder で視聴しながら、HDD に保存する際に用いられる。(保存データ形式 : SC2000)
尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

⑧ Play&Save, IN:SecureHard, TimeShift:ON, MULTI2:ON
この処理モードは、secure hard に入力された TS データをタイムシフト機能ありで、secure decoder で視聴しながら、HDD に保存する際に用いられる。(保存データ形式 : SC2000+MULTI2)
尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

⑨ Play&Save, IN:SecureHard, TimeShift:ON, MULTI2:OFF
この処理モードは、secure hard に入力された TS データをタイムシフト機能ありで、secure decoder で視聴しながら、HDD に保存する際に用いられる。(保存データ形式 : SC2000)
尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

⑩ Play Only, IN:HDD, MULTI2:ON
この処理モードは、HDD に格納された TS データを secure decoder で視聴を行う際に用いられる。(保存データ形式 : SC2000+MULTI2)
尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

⑪ Play Only, IN:HDD, MULTI2:OFF
この処理モードは、HDD に格納された TS データを secure decoder で視聴を行う際に用いられる。(保存データ形式 : SC2000)
尚、SS PID,SD PID を設定しないことにより、暗号なし TS データの復号処理を実現する。(デバッグ時のみ)

【鍵パケット/制御パケット】

Secure PC system は、TS ストリームはスクランブル鍵による暗号化,スクランブル鍵はコンテンツ鍵による暗号化により、データセキュア化を行っている。また、コンテンツ鍵は、セキュアマスタ鍵による暗号化を行っている。TS ストリームを暗号化するスクランブル鍵は、REC OUT,HDD IN においては、TS ストリーム内鍵パケットとして多重され、PLAY OUT においては、共有メモリ経由にて渡される。また、スクランブル鍵を暗号化するコンテンツ鍵は、共有メモリ経由にて渡される。(REC OUT,HDD IN のみ)

【TS バッファ管理】

TS バッファは、TAO source filter からの TS パケット入力,TS 解析処理および TAO video decode filter/TAO audio decode filter への出力に用いられる。また、鍵 TS パケット/共有メモリより、再生用スクランブル鍵(KPLS_O,KPLS_E),スクランブル制御値(odd/even)を抽出し、Key バッファに格納し、対象となる出力用メディアサンプル情報に格納ポインタを設定する。TS バッファサイズは、(188B×n 個)×128 面(VirtualAlloc でメモリ確保)とする。

5-3-2-3 TAO video decode filter

TAO video decode filter は、暗号化された MPEG2 TS payload の暗号復号化,PES 復号,video 復号処理を行う。

図５－３－６に、TAO video decode filter のクラス構成を以下に示す。

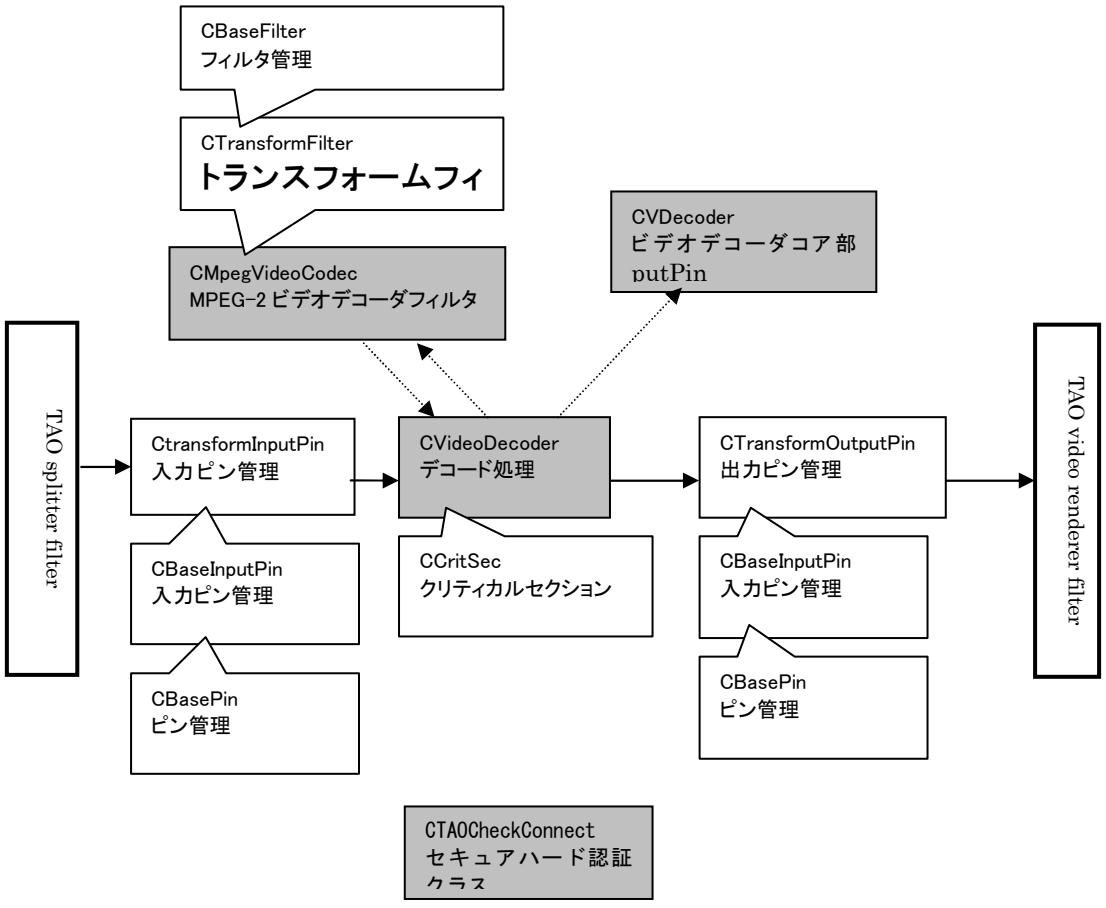


図５－３－６　TAO video decode filter クラス構成

【PES 復号】

PES 復号において、packet start code prefix,stream id,PES packet length,PES header data length,PES packet data byte 等によりエラー検出した際には、復帰処理(エラー通知処理含む)を行い、復帰の際にはダウンストリームフィルタに残っているメディアデータをフラッシュする。PES ヘッダ検出トリガとなる packet start code prefix,stream id は、TS ヘッダに実装されている payload unit start indicator 検出の際に行われる。また、PES パケットのデータ長情報である PES packet length,PES header data length,PES packet data byte により、データ長の妥当性を確認し、異常時には復帰処理を行う必要がある。

PES ヘッダに格納されている PTS 情報は、CMediaSample クラスのメンバ変数である m_MediaStart,,m_MediaEnd に変換することにより、TAO video render filter, TAO audio render filter からの映像,音声出力タイミングを指定する。TAO video render filter と TAO audio render filter から同期がとれた映像,音声出力を保証するために、secure decoder 起動後または、エラー復帰後の一番初めの PTS/DTS 受信時に現在のメディアタイムとの位相関係(acum_DTS のデフォルト値)は、TAO video decode filter, TAO audio decode filter は、

① TAO video decode filter, TAO audio filter は算出した PTS/DTS とメディアタイムとの位相差情報を TAO splitter filter に設定要求
② TAO splitter filter は、secure decoder 起動後または、エラー復帰後に一番初めに設定要求があったフィルタ(TAO video decode filter, TAO audio decode filter)の位相差情報を acum_DTS のデフォルト値とする。
③ TAO video decode filter, TAO audio filter は、決定された acum_DTS のデフォルト値を取得し、自身の acum_DTS に設定する。

を行うことにより、TAO video decode filter と TAO audio decode filter にて独立に設定されるメディアタイムの同期性を保証する。

【video 復号】

video 復号において、各レイヤの start code 等によりエラーを検出した再には、復帰処理(エラー通知処理含む)を行い、ダウンストリームフィルタ内に残っているメディアデータはフラッシュする。また、TAO video render filter, TAO audio render filter からの quality message(Late 使用)により、ピクチャのスキップ処理を行う。

video レイヤにおいて、sequence header code, sequence end code, group start code, picture start code といった start code エラーが発生した場合、次の I ピクチャより再引き込み処理を行う。但し、P ピクチャ内において、intra slice を検出した場合には次 I ピクチャではなく、P ピクチャより再引き込み

処理を行う。
MPEG2 デコード処理において、TAO video render filter, TAO audio render filter といったレンダラフィルタのオーバフロー/アンダーフローが発生した場合には、レンダラフィルタよりアップストリームフィルタに対して quality message を送る。TAO video render filter では、quality message を受信した際には、負荷の大きさに応じて、ピクチャをいくつかスキップすることで、負荷の調整を行う。

クオリティコントロールは、quality 構造体のメンバ変数である Late(遅延時間)により、video 復号処理が遅れている分のピクチャ数を決定し、復号スキップ処理を行う。但し、スキップ可能なピクチャは参照画とならない B ピクチャのみであり、IBBP 構造である MPEG2 は一度に復号スキップ処理を行うことができるピクチャ数は最大 2 枚である。(レジストリの値を変更することにより P ピクチャの復号スキップ処理も可能)

【PES ヘッダバッファ,ES バッファ管理】

PES ヘッダバッファ,ES バッファは、TAO splitter filter(COutputQueue)からの映像 TS パケット/映像鍵制御パケット入力,暗号復号化,PES ヘッダ解析,ピクチャの生成,PTS 生成および video decoder への出力に用いられる。
PES ヘッダバッファサイズは、1KB(＞最大 PES ヘッダ長),ES バッファは、1.8MB(VirtualAlloc でメモリ確保)とする。

5-3-2-4 TAO audio decode filter

TAO audio decode filter は、暗号化された MPEG2 TS payload の暗号復号化,PES 復号,audio 復号処理を行う。図５－３－７に、TAO video decode filter のクラス構成を以下に示す。

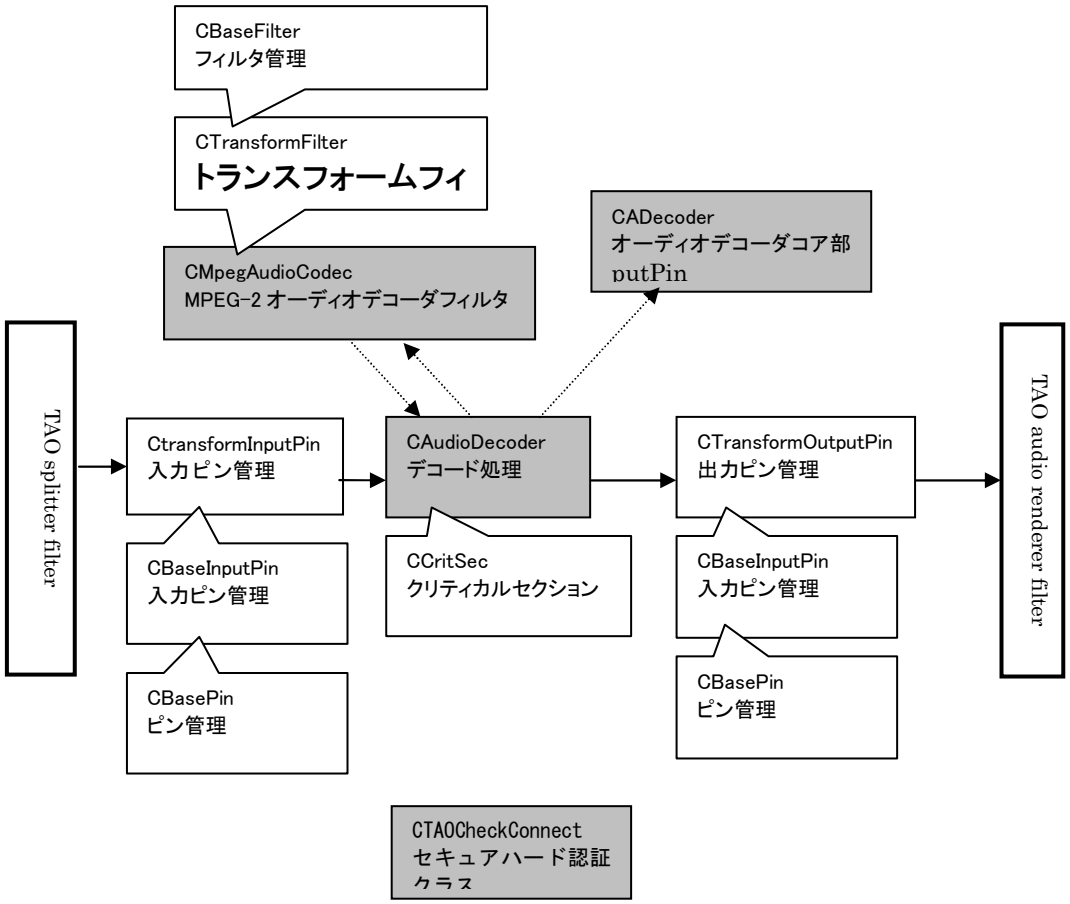


図５－３－７　TAO audio decode filter クラス構成

5-3-2-5　TAO video render filter

TAO video render filter は、TAO video decode filter にて復号された映像データの再生処理を行う。

詳細については検討中であり、今回の開発では、MS の Video Renderer Filter **を使用することとする。**

5-3-2-6　TAO audio render filter

TAO audio render filter は、TAO audio decode filter にて復号された音声データの再生処理を行う。

詳細については検討中であり、今回の開発では、MS の DirectSound Devices **を使用することとする。**

5-3-3　GUI

secure decoder におけるユーザとのインタフェースを行う GUI は以下示す六つの画面から構成されている。

図５－３－８に示すメイン画面から、

視聴ボタン押下すると、図５－３－９に示す放送番組テーブルがポップアップ。

再生ボタン押下すると、図５－３－１０に示す蓄積番組テーブルがポップアップ。

録画ボタン押下すると、図５－３－９放送番組テーブルがポップアップ。

設定/読出ボタン押下すると、図５－３－１１設定/読出画面がポップアップ。

統計ボタン押下すると、図５－３－１２統計データ画面がポップアップ。

通知ボタン押下すると、図５－３－１３通知画面がポップアップ。

Close ボタン押下すると、GUI 終了。

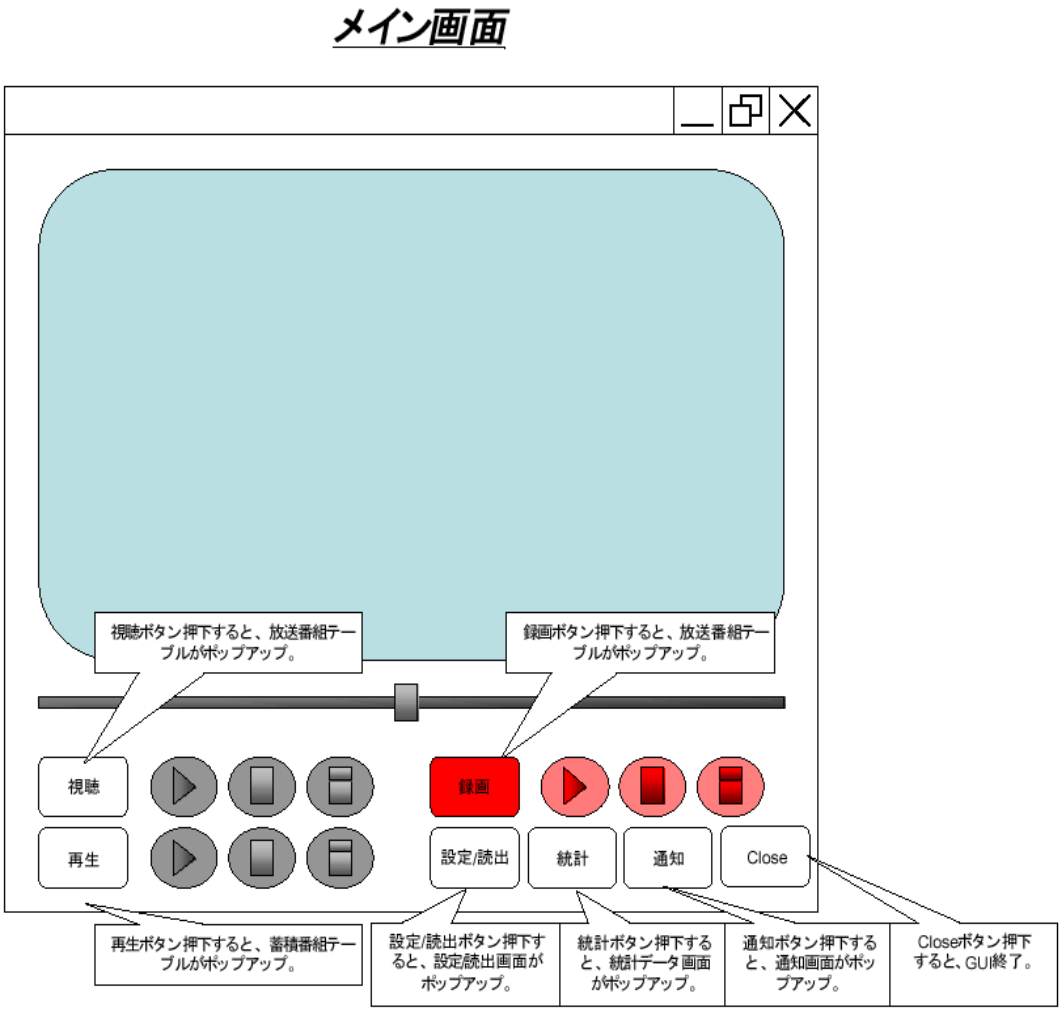

メイン画面
視聴ボタン押下すると、放送番組テーブルがポップアップ。
録画ボタン押下すると、放送番組テーブルがポップアップ。
視聴
録画
再生
設定/読出
統計
通知
Close
再生ボタン押下すると、蓄積番組テーブルがポップアップ。
設定/読出ボタン押下すると、設定読出画面がポップアップ。
統計ボタン押下すると、統計データ画面がポップアップ。
通知ボタン押下すると、通知画面がポップアップ。
Closeボタン押下すると、GUI終了。

図５－３－８　メイン画面

放送番組テーブル

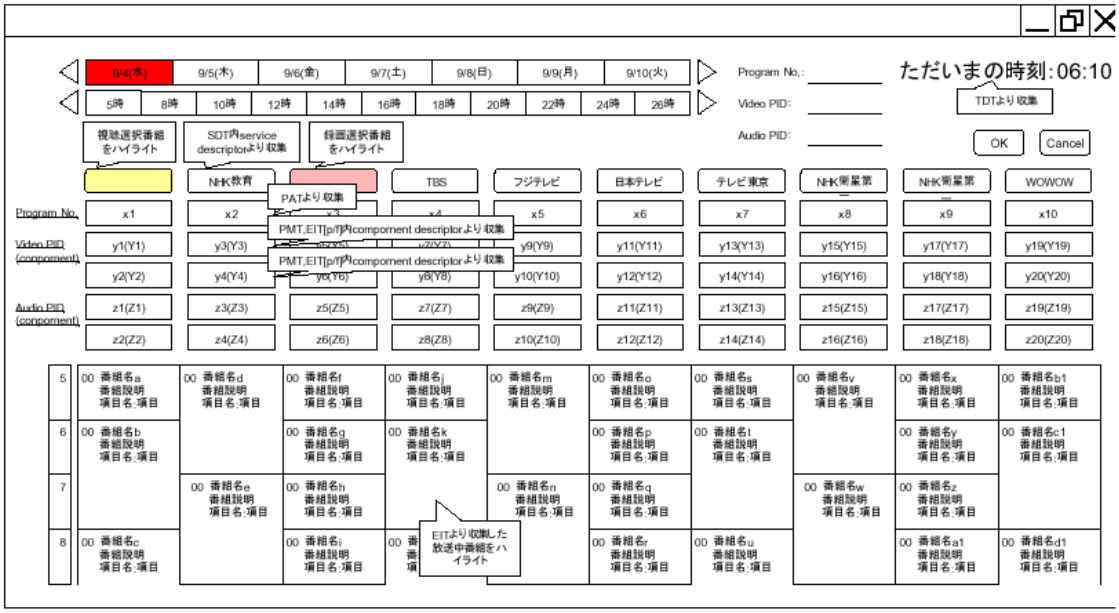


図５－３－９　放送番組テーブル

蓄積番組テーブル

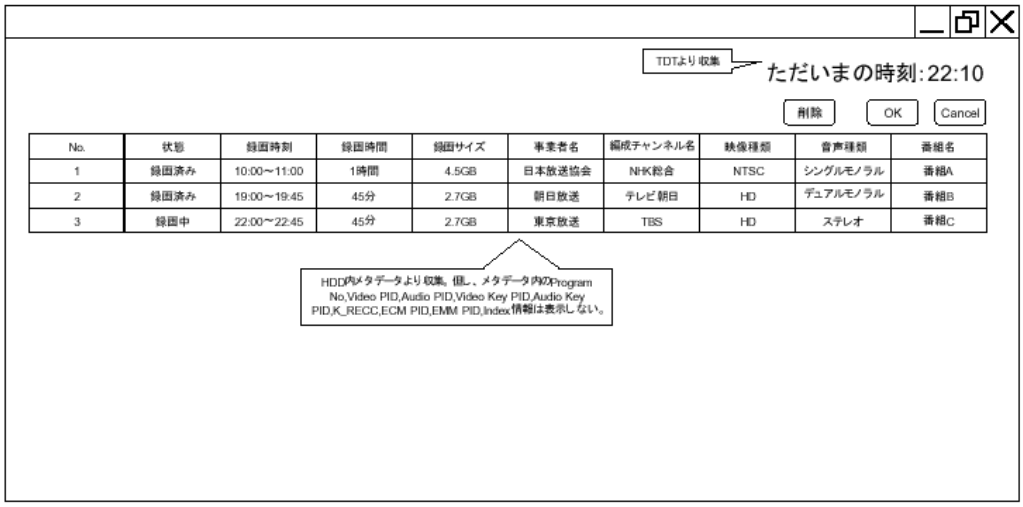


図５－３－１０　蓄積番組テーブル

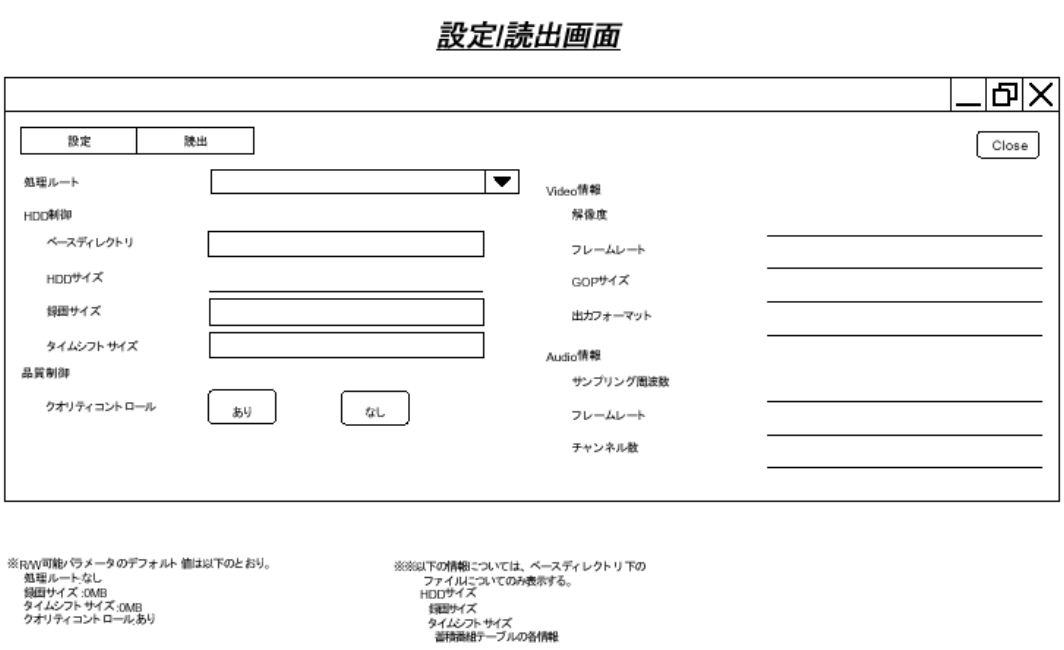


図５－３－１１　設定/読出画面

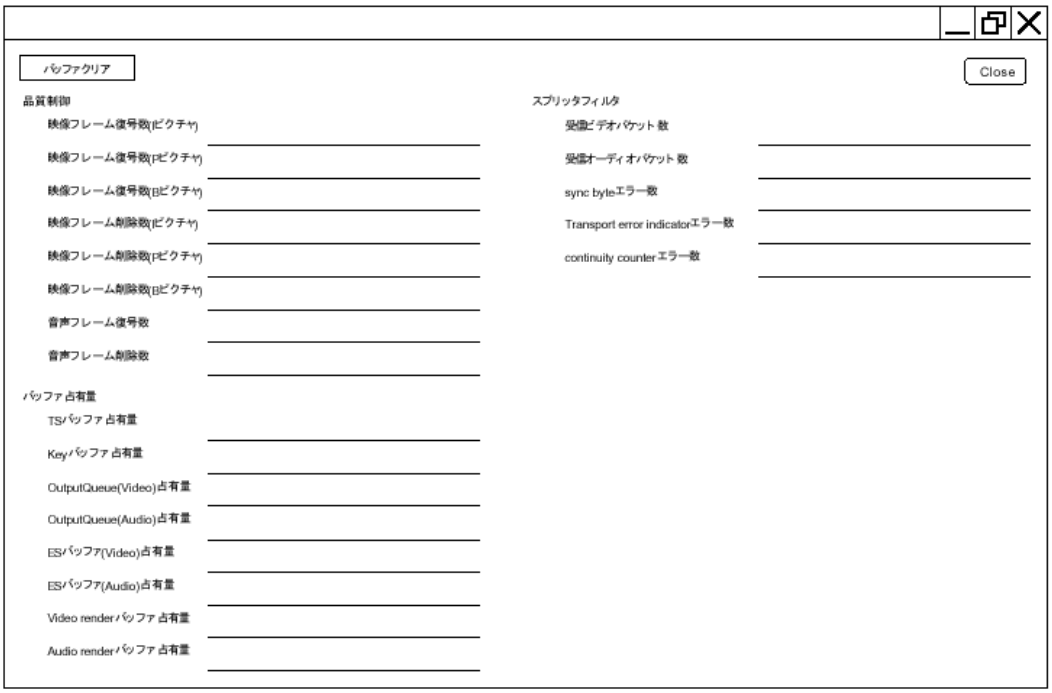


図５－３－１２　統計データ画面

図５－３－１３　通知画面

５-３-４ セキュア化

セキュア化は、ファイル等からディスアセンブラなどを用いた静的解析･攻撃,デバッガや別プログラムからの動的解析･攻撃に対して、プログラムコードおよびデータの保護を行うことを目的とする。セキュア化は基本的に以下に示す要素技術を組み合わせることにより実現する。

- 難読化(プログラムコード)
- 実行時暗号復号(プログラムコード, データ)
- リアルタイム認証(プログラムコード)
- プログラムスキャン(プログラムコード)
- ダミーコード挿入(プログラムコード)
- 再構成(プログラムコード)

secure decoder は、上記の”実行時暗号復号(データ)”,”リアルタイム認証”の機能を実現する。

【実行時暗号復号(データ）】

secure decoder は、secure hard により出力された TS ストリームは、TAO source filter⇒TAO splitter filter⇒TAO video decode filter/TAO audio decode filter までは、SC2000 暗号化された状態で処理されるため、データの解析に対するセキュア化が実現できる。

SC2000 暗号復号化は、TAO splitter filter にて鍵制御パケット解析を行い、

TAO video decode filter, TAO audio decode filter にて暗号復号化処理が行われる。

また、TS バッファ,ES バッファは一定処理完了ごとに、新たな別領域を確保することにより、一定領域でのデータ常駐を避けることにより、スナップショットに対するセキュア化を実現する。

【リアルタイム認証】

リアルタイム認証では、HW 認証とフィルタ認証を行う。

HW 認証では、One Time Password 認証での Challenge&Response 方式である秘密の番号通信を用いる。また、フィルタ認証では、独自インタフェースによる接続先フィルタ情報の取得を行う。

- HW 認証

PC 上に確保された swap されないメモリ領域に存在する secure decoder が実際に動作していることを確認するために secure decoder と secure hard 間で One Time Password 認証での Challenge&Response 方式である秘密の番号通信により、secure decoder が確実に PC 上のメモリ上で動作していることを確認する。図 5－3－14 に秘密の番号通信の構成,図 5－3－15 に秘密の番号通信における処理フローについて示す。

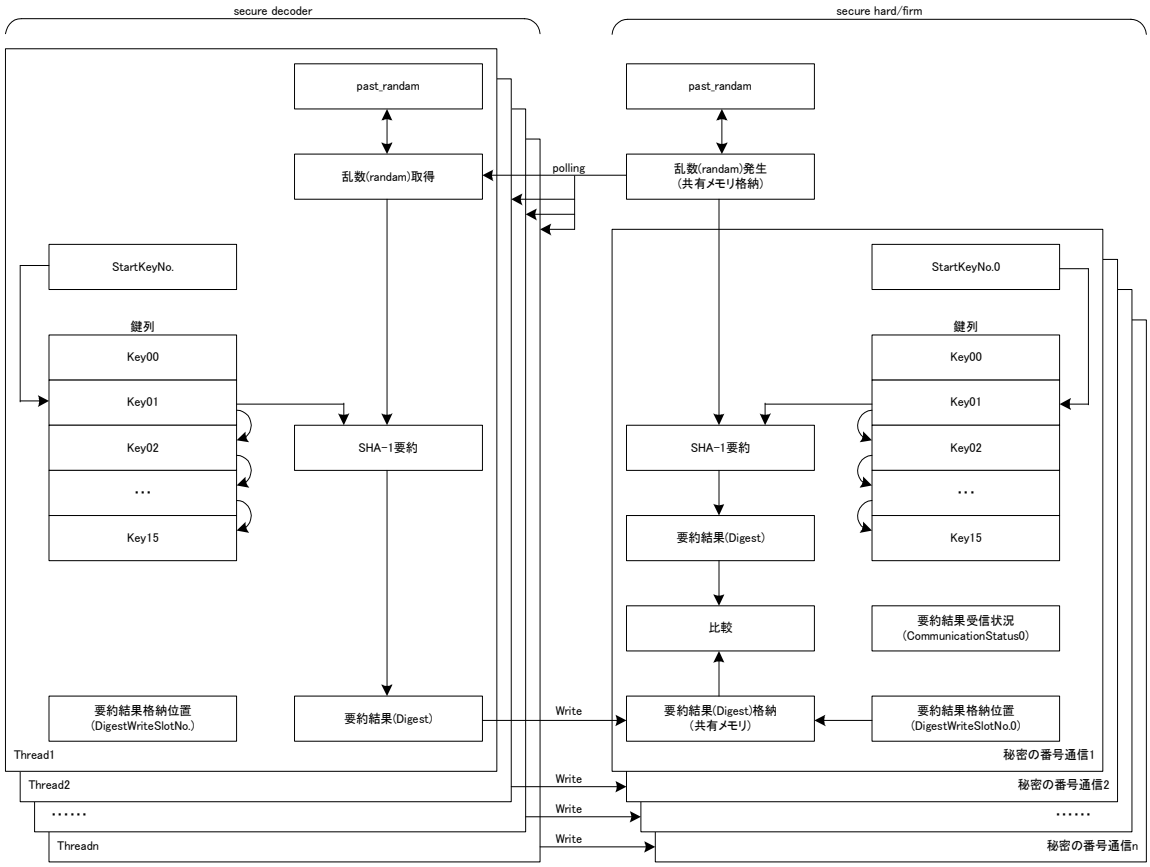


図５－３－１４　秘密の番号通信の構成

尚、key00～15,StartKeyNo.,DigestWriteSlotNo.は、再構成時に secure firm が、プログラムコードに埋め込む。

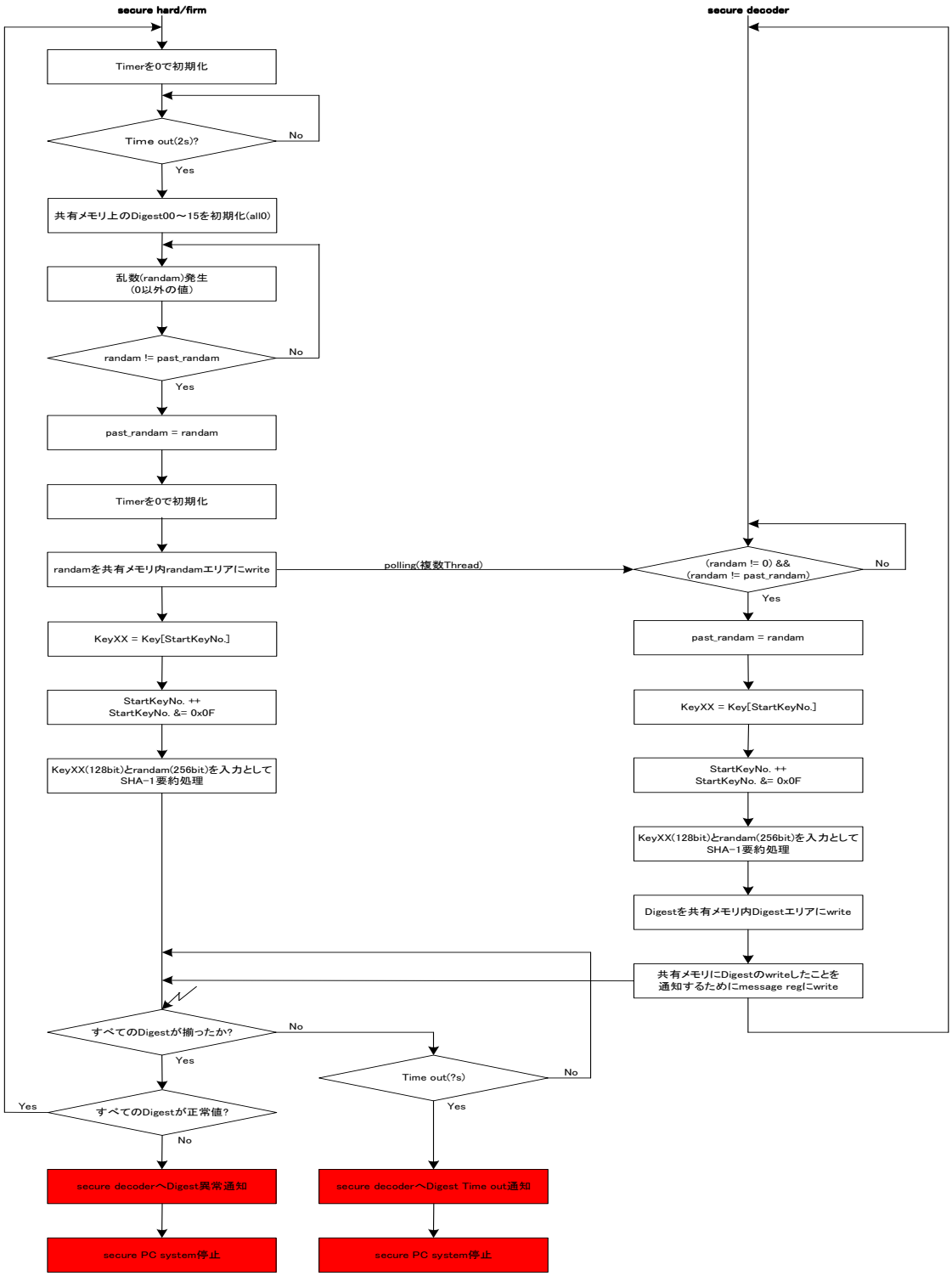


図５－３－１５　秘密の番号通信における処理フロー

尚、リアルタイム認証(HW 認証,フィルタ認証)は、別スレッドにて実現する

ため、GUI からの stop, pause 指示や secure decoder エラー検出による停止時においても、影響されることなく、処理を継続する。

- フィルタ認証

フィルタグラフにより構築される secure decoder において、フィルタ間に動的なフィルタ挿入によるデータの盗み見ができないように、隣接するフィルタ間で独自インタフェース(ITAOFilterInfo)を用いたフィルタ情報を受け渡すことにより、隣接するフィルタが正しいことを確認する。

５－４　総括

これまで述べたように、平成１４年度は前年度に開発したセキュア試作ボードのＦＰＧＡに搭載するセキュア回路の開発、セキュアハードからソフトウェアを保護するためのセキュア機能の検討と機能試作、そして実際にセキュアハードを使用してＭＰＥＧ－ＴＳストリームを伸張／表示するためのデコーダソフトウェアの開発を行った。

１）セキュア回路では、Multi2 復号化回路と B-CAS カードインタフェースによって、放送されるデジタルＡＶコンテンツの暗号を解く機能を実現した。さらに、SC2000 暗号化／復号化回路によるローカル暗号化機能を実現するとともに、これらの暗号化回路で使用される暗号鍵を安全に提供する方法を搭載した。これにより、ユーザアクセス可能なバスであるＰＣＩバス上を流れるコンテンツは必ず暗号化された状態で転送することを可能とした。また、各ボードは固有のマスター鍵を持っており、マスター鍵を基にしてハードディスク保存時のコンテンツの暗号化が行われる。そのため、ハードディスク上のコンテンツの複製を防ぐことが可能となった。セキュア回路にはＰＣ上のデコーダソフトウェアとボード上の制御用ファームウェアがコマンドやデータを通信するための共有メモリを設けて柔軟なデータの交換ができるような機能を実現した。これにより、レジスタに比較して大量のデータの交換が可能になった上、機能の強化やアップデートが適宜行えるようになっている。また、共有メモリはセキュリティ強化のため、ＰＣ上のソフトからのアクセスとして Read/Write 可能な領域に加えて、Read Only, Write Only, Read Once という機能を持たせることができ、他のソフトからのアクセスに対してデータの保護を行うことができる構成となっている。

セキュア回路は、コンテンツの保存や保存されたコンテンツの再生表示、タイムシフト再生、放送される番組を保存しながら保存されている別の番組を視聴するといったさまざまな操作が可能な構成を実現した。

セキュア回路のデータ処理能力は、ボードとＰＣの間のデータ転送ルート数が２系統のとき 67 Mbps、４系統のとき 34 Mbps である（ＦＰＧＡ，12.5MHz 動作で測定）。LSI 化に際してはセキュア回路の動作周波数を上げることを勘案して、デジタルハイビジョンの処理に十分な能力を有していると判断する。

２）デジタル放送受信システム全体の構成を検討し、セキュアハードに搭載するセキュア機能の検討を行った。セキュアハードに搭載する機能としては、セキュリティ維持に必要な機能だけを抽出した。そして、セキュアハードからデコーダなどのソフトウェアを保護する機能を実現することとし、機能の検討を行った。その結果、ソフトウェア解析を防止するためのソフトウェア再構成機能、改竄を防止するためのソフトウェア監視機能、成りすましを防止するためのリアルタイムの認証機能を考案し、機能試作を行った。

ソフトウェアの再構成機能では、Windows の実行プログラム、ダイナミックリンクライブラリを複数のブロックに分割し、再構成に必要な情報を取り出す機能を実現するとともに、分割されたブロックをランダムに再配置して実行

させる機能を実現した。ここで、ランダムな再配置はプログラムが実行されるたびに行われ、そのたびに見かけ上異なったプログラムが実行されているようにすることができる。本システムでは、ランダムに再配置し実行させる機能はセキュアボード上のファームウェアで行い、ファームから実行プログラムをＰＣ上に転送することを実現した。これにより、実行中のプログラムを解析することは困難になり、仮に解析されたとしても、次に起動されたときは異なった配置のプログラムが動作していることから、解析結果が無駄になる。さらに、分割されたソフトウェア自体は暗号化された上でハードディスク上に保存されており、セキュアハード上で復号化されるため、分割ブロック自体の解析も不可能である。

ソフトウェアの監視機能は、セキュアハードが常時ＰＣのメインメモリ上の実行プログラムを監視する機能である。ソフトウェアの改竄が行われた場合には、セキュアハードがその改竄を検出する機能を実現した。これはプログラム起動時の情報を基にセキュアハードがソフトウェアとは独立して行う機能であり、監視処理の実行タイミング等をＰＣ側で知ることはできないため、この監視を回避して改竄を行うのは困難な構成となっている。

リアルタイム監視は、ソフトウェアとセキュアハード上のファームウェアが一定間隔おきに認証データの交換を行う機能である。交換された認証データが誤っていた場合や、一定時間内に交換が行われなかった場合は不正が行われていると判断する。認証データや認証方式はセキュアハードが適宜変更しながら行うことにより解析を防止する。

このようにセキュアハードが主体となった複数のセキュア技術を組み合わせることにより、強固なセキュリティを実現することができる。今後は、実行速度を含めた技術の評価を進める予定である。

３）ＭＰＥＧ－２ＴＳストリームの再生を行うソフトウェアデコーダの開発を行った。このデコーダは暗号化されたストリームデータの復号化、ＭＰＥＧ－２の伸張処理、Windows 画面への表示を行う。また、GUI 機能を持っており、複数のコンテンツが含まれたマルチストリームから番組表の表示や視聴する番組の選択を行うことができる。処理性能は、ＭＰＥＧ－２ＭＰ@ＭＬストリームに対しては、秒３０フレームのリアルタイム処理が可能であり、ハイビジョン映像のＭＰＥＧ－２ＭＰ@ＨＬストリームに対しては１４年度目標である秒５フレームを上回る、秒１０フレームの再生が可能となっている。今後は来年度のリアルタイム処理化に向けて、処理ブロックごとの処理時間の分析や、アルゴリズムの見直し、MMX/MMX2/SSE/SSE2 といったＣＰＵのマルチメディア命令の使用等を検討する。

以上のように、セキュア回路の開発、セキュリティ機能の開発、ＭＰＥＧ－２デコーダの開発とも予定通り進んでおり、１４年度の目標は概ね達成している。今後は、セキュリティ機能の評価を行うとともに、セキュア回路のＬＳＩ化、デコーダのリアルタイム化を進める。