10. ケーススタディ1：ブルーレイディスク

10.1. 検証の概要

10.1.1. 検証対象システムの概要

検証対象システムは、再生専用の Blu-ray Disc プレイヤのディスク操作を制御するミドルウェア（以下、モード制御ソフトウェアと呼ぶ。）である。Blu-ray Disc プレイヤは、Blu-ray や CD、DVD などのディスクメディアを、プレイヤ本体の挿入口に入れ、ユーザのリモコン操作を通して入力される命令に応じて、メディアに記録されている映像の再生、停止、一時停止、早送り、スロー再生などを行う。

モード制御ソフトウェアは、図 10-1 に示す Blu-ray 系インタフェースと Blu-ray ミドルウェアモジュールで構成される。Blu-ray 系インタフェースは、Blu-ray アプリケーションと Blu-ray ミドルウェアモジュールのインタフェースであり、Blu-ray アプリケーションが要求する Blu-ray 用ドライブの制御（再生、停止等）を、Blu-ray 用ドライブの状況に応じて Blu-ray ミドルウェアモジュールに伝える。Blu-ray 系インタフェースから要求された制御命令を、Blu-ray 用ドライブの状態に応じて、ドライバを通して制御する。

CD、DVD など Blu-ray 以外のメディアについては、Red 系インタフェースを通して制御する。

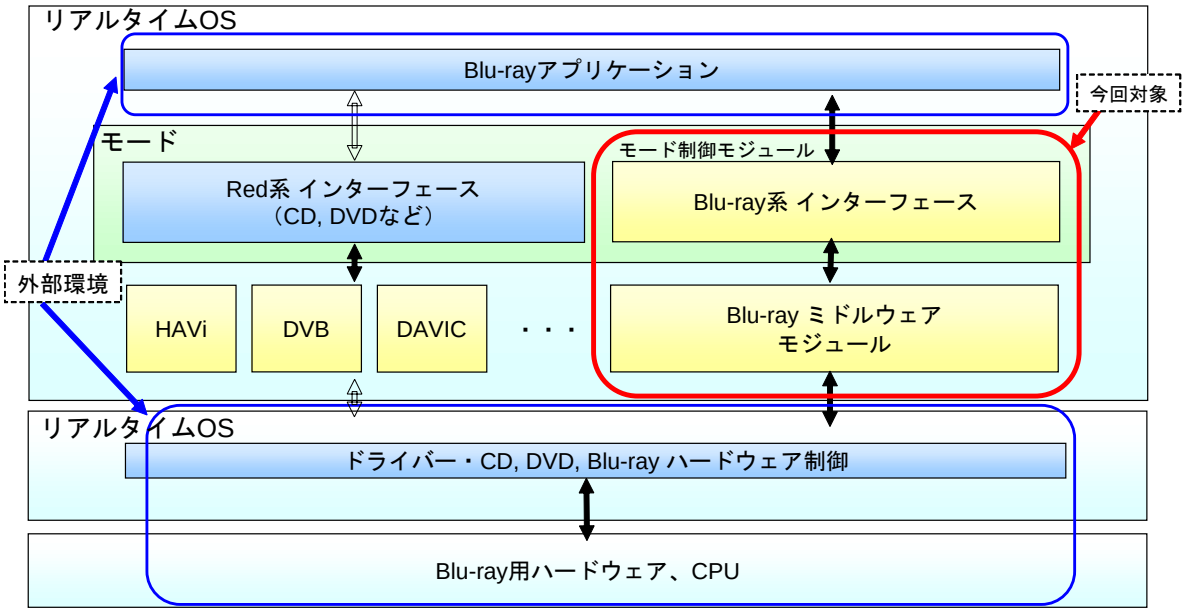

図 10-1：検証対象システムの構成

ユーザがリモコンなどを通して Blu-ray ディスクを操作する命令は、Blu-ray アプリケーションがインタフェースとなって受け付ける。Blu-ray アプリケーションは、モード制御ソフトウェアの API を通してユーザから受け付けた命令を、モード制御ソフトウェアに伝達する。モード制御ソフトウェアは、

Blu-ray アプリケーションから伝達されたユーザの命令を受信して、ディスクが現在再生中か、停止中かなどの状態を、ドライバを通して観測し、その状態を基にドライバに受け付けた命令を伝達する。例えば、ディスクが再生中の場合には、再生命令をモード制御モジュールが受け付けてもドライバには転送しないが、停止命令を受け付けた場合、モード制御モジュールはドライバに停止命令を要求する。また、モード制御モジュールは、Blu-ray ドライブから発せられる「再生終了」や「読み込みエラー」(ディスクに傷などがついていて、読み込みに失敗した場合に発せられるエラーイベント)などのイベント割り込みを受け付けて、停止状態への移行などモード制御を行う。

10.1.2. 検証の目的と検証内容の概要

モード制御ソフトウェアはBlu-ray系インタフェースとBlu-rayミドルウェアモジュールで構成されている。Blue-rayミドルウェアモジュールは、ディスクの制御状態を細かく把握しており、一方のBlue-ray系インタフェースはBlu-rayミドルウェアモジュールを通してディスクの制御状態をより抽象的に把握する。例えば、Blu-rayミドルウェアモジュールは、ディスクの制御状態を「再生移行中」、「再生」、「早送り」、「正スロー再生[161]」などのように詳細に把握しているが、Blu-ray系インタフェースはこれらの状態を1つの「再生」状態として把握する。このように 2 つのモジュールが異なる状態把握を行っていることに起因して、状態把握の不整合が生じ、結果としてソフトウェア設計時に意図していなかった制御やデッドロックが発生する可能性がある。そこで、今回の検証では、以下の点に着目して検証を行うこととした。

- モード制御ソフトウェア内の Blu-ray 系インタフェースと Blu-ray ミドルウェアモジュールのそれぞれが把握する状態の不整合などにより、意図しない制御やデッドロックが発生しないことを確認する。

SPIN によるモデル検査では、モード制御ソフトウェアをモデリングするとともに、それ以外の部分は外部環境としてモデル化した。

検証の結果、当初の設計において、API 要求とデバイスからの割込み処理が同時に発生した場合に、デッドロックが発生することが検出され、その修正を行うことができた。

10.2. 検証対象システムの仕様

Blu-ray プレイヤ全体のシステム構成を図 10-2 に示す。検証を行うソフトウェアは、図 10-2 中の Blu-ray 系インタフェースと Blu-ray ミドルウェアモジュールで構成されるモード制御ミドルウェアである。

[161] 正スロー再生とは、再生方向へのスロー再生のこと。

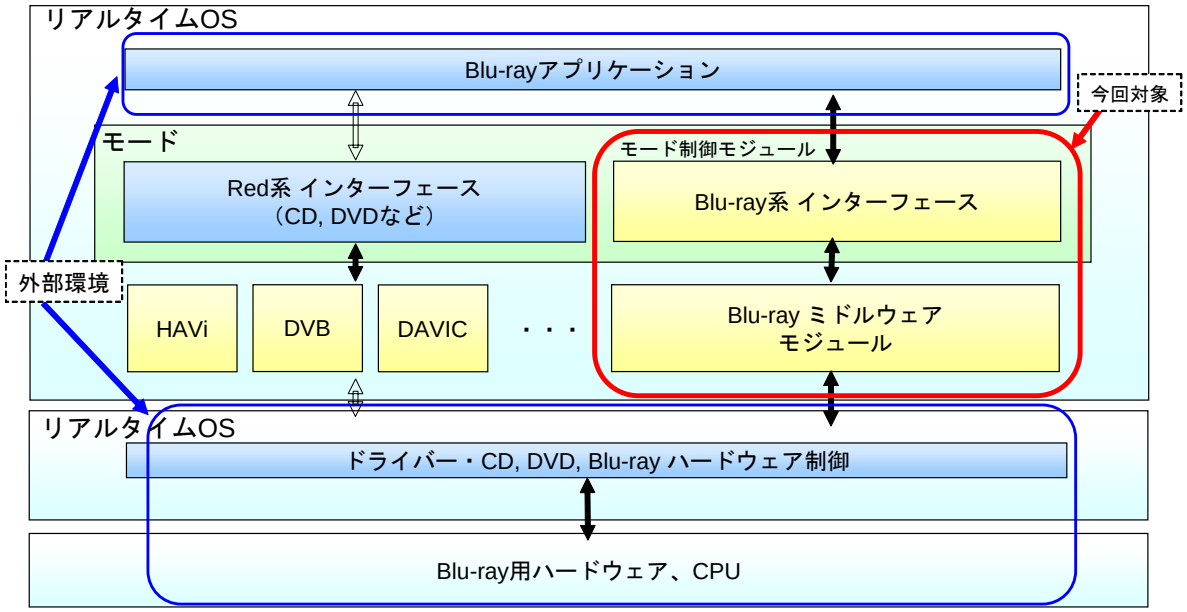

図 10-2:検証対象システムの構成(再掲)

Blu-ray プレイヤのシステムは、ハードウェアモジュールとソフトウェアモジュールで構成されている。Blu-ray や CD、DVD などのディスクメディアを扱う Blu-ray 用ドライブ、Blu-ray 用ドライブを制御する制御プロセッサはハードウェアで構成されている。制御プロセッサとのインタフェースとして、ドライバが実装されており、ドライバを通して、モード制御ミドルウェアが Blu-ray 用ドライブを制御する。また、Blu-ray ディスクが最後まで再生された場合や、Blu-ray ディスクに傷などがありディスク上に記録されているデータの読み込みができなかった場合には、それぞれ「再生終了」、「読み込みエラー」のイベントを制御プロセッサが発生させ、割り込み処理としてモード制御モジュールに伝達する。なお、Blu-ray 用ドライブおよび制御プロセッサ以外のコンポーネントは全てソフトウェアで構成されている。

リモコン操作などを通してユーザから Blu-ray プレイヤに送信されてくる命令は、Blu-ray アプリケーションと呼ばれるスレッドで蓄積、管理される。このスレッドは複数並列処理可能である。Blu-ray アプリケーションに蓄積された命令は、モード制御ミドルウェアに送信される。モード制御ミドルウェアは、Blu-ray 用ドライブの状態(例えば、「停止」や「再生中」など)の状態に応じて、Blu-ray アプリケーションから送信されてきた命令を、ドライバを通して、制御用プロセッサに伝達する。

ディスク操作モード制御ミドルウェアは、Blu-ray 系インタフェース(以下、「モードモジュール」と呼ぶ。)と Blu-ray ミドルウェアモジュール(以下、「ミドルモジュール」と呼ぶ)で構成されている。Blu-ray 系インタフェースと Blu-ray ミドルウェアモジュールは、それぞれ Blu-ray 用ドライブの現在の状態を記憶しているが、その詳細さが異なる。例えば、Blu-ray ミドルウェアモジュールは Blu-ray 用ドライブが「再生移行中」、「再生」、「早送り」、「正スロー再生」などのように詳細に把握しているが、Blu-ray 系インタフェースはこれらの状態を1つの「再生」状態として把握する。

Blu-ray 系インタフェースは、Blu-ray 用ドライブの現在の状態を Blu-ray ミドルウェアモジュールに問い合わせを行い、Blu-ray ミドルウェアからの戻り値によって Blu-ray 用ドライブの状態を把握する。つまり、Blu-ray インタフェースは Blu-ray 用ドライブの状態を直接観測できず、Blu-ray ミドルウェアモジュールを通して把握する。

Blu-ray 系インタフェースは、Blu-ray アプリケーションが呼び出すことを想定した API（アプリケーションプログラミングインタフェース）を提供しており、Blu-ray アプリケーションがユーザから受け付けた命令に対応する API 関数を呼び出すことで、ユーザの命令を Blu-ray 系インタフェースに伝達する。Blu-ray 系インタフェースは、Blu-ray アプリケーションから呼び出された API 関数と、自身が把握している Blu-ray 用ドライブの状態に応じて、Blu-ray ミドルウェアモジュールの関数を呼び出す。なお、Blu-ray 系インタフェースと Blu-ray アプリケーションは 1 対 1 に対応し、複数のスレッドが並列で動作するようになっている。

Blu-ray ミドルウェアモジュールは、自身が把握している Blu-ray 用ドライブの状態に応じて、呼び出された関数に応じて、Blu-ray 用ドライブの制御をドライバを通して行う。

10.2.1. 検証対象システムに求められる条件（要求仕様）

検証を行うモード制御ミドルウェアに求められる要求仕様を表 10-1 に示す。検証対象のモード制御ミドルウェアが「必ずしなければならないこと」を「機能要求」、「絶対にしてはならないこと」を「非機能要求」として、分類して表記してある。

表 10-1 に示した要求仕様が、後述するモデル検査における検証内容の出発点となる。

表 10-1：モード制御ミドルウェアの要求仕様

	要求仕様
機能要求	モードモジュールは、要求された API に対して、モード状態に基づき操作可能な API については、ミドルモジュールに対して、API に対応する要求を出す。
	ドライブから割り込みが発生した場合、ミドルモジュールは、割込みの種類に応じてミドル状態とモード状態を遷移させる。
非機能要求	デッドロックが発生しない。
	モードモジュールは、要求された API のうち、現在のモード状態で実行できない API は、ミドルモジュールに要求を出さない。
	モード状態、ミドル状態の間に不整合は生じない。

10.2.2. 検証対象システムの設計仕様

検証対象となるモード制御ミドルウェア（Blu-ray 系インタフェースと Blu-ray ミドルウェアモジュール）を中心として、その周辺のコンポーネントを含めた設計仕様をまとめる。

10.2.2.1. プロセス構成

検証を行うモード制御ミドルウェアに関連するコンポーネントは、Blu-ray系アプリケーション、ドライバ、および、割り込み処理を通知するコールバック[162]である。ここでは、各コンポーネントがBlu-rayディスクシステム内で実行時に起動されるスレッド数をプロセスとして考える。

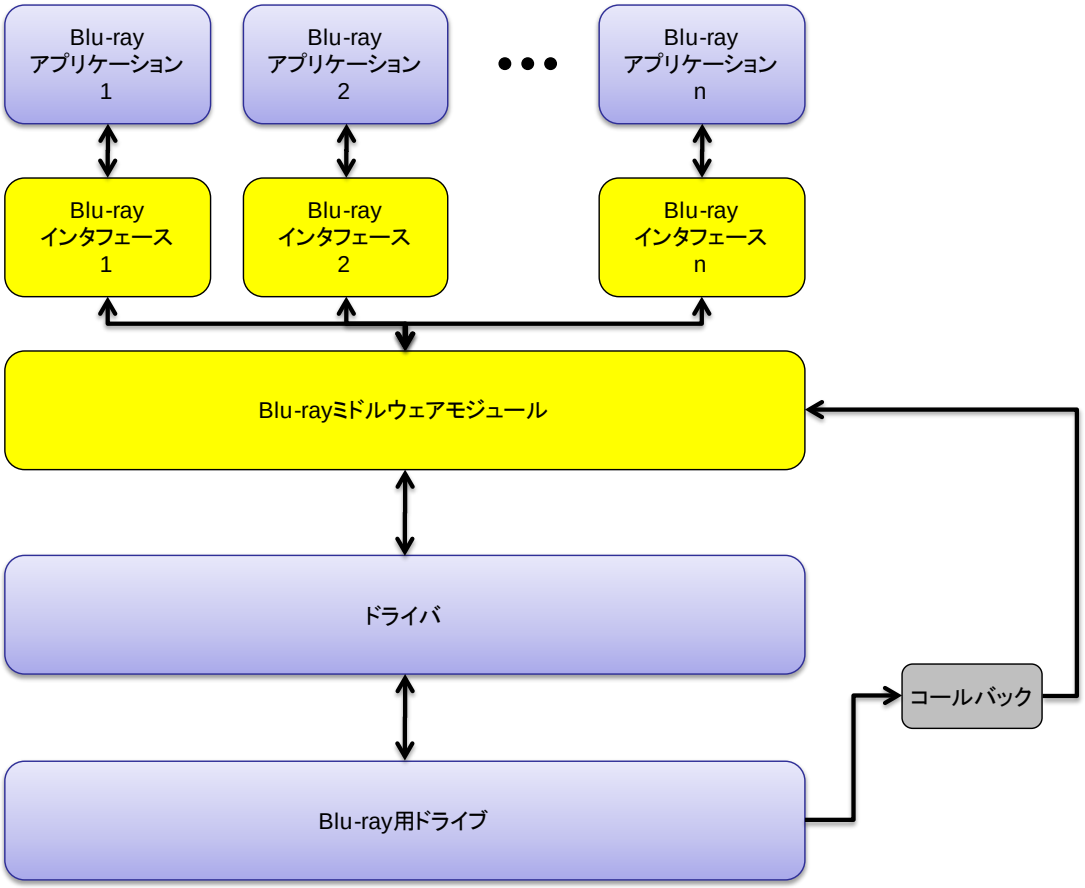

図 10-3:プロセスの構成図

Blu-ray アプリケーションは、複数のスレッドとして並列に動作するため、n 個のプロセスとなる。モード制御ミドルウェアの一部である Blu-ray 系インタフェースは、1 つの Blu-ray アプリケーションに対応して、1 つのスレッドが動作する。従って、Blu-ray 系インタフェースのプロセス数は、Blu-rayアプリケーションのプロセス数と同じ n 個である。1 つのBlu-rayアプリケーションプロセスとそのプロセスに対応する Blu-ray 系インタフェースがメッセージ通信を行うことで、ユーザ操作内容(命令)のモード制御ミドルウェアへの伝達や、Blu-ray 用ドライブなどのハードウェア部分の割り込み処理の結果等の Blu-ray アプリケーションプロセスへの伝達が行われる。

Blu-rayミドルウェアモジュールは、Blu-ray ディスクシステムに 1 つ付属する Blu-ray 用ドライブをドライバを通して観測する。このため、Blu-ray ミドルウェアモジュールのプロセスは、システム内

[162] プログラム中で、呼び出し先の関数の実行中に実行されるように、あらかじめ指定しておく関数。

で 1 つのスレッドが起動され、プロセスは 1 つとなる。同様にドライバのプロセスも 1 つである。

一方、コールバックは、Blu-ray 用ドライブから発生する予測不能なイベントをキャッチする。もし、イベントが発生した場合、当該イベントをキャッチした1つのコールバックのスレッドは、割り込み処理を行うように Blu-ray ミドルウェアモジュールに通知を行う。

10.2.2.2. モード制御モジュールの内部状態

モード制御モジュールを構成する Blu-ray 系インタフェースと Blu-ray ミドルウェアモジュールはそれぞれ異なる抽象度で、Blu-ray 用ドライブの状態を把握する。

10.2.2.2.1. Blu-ray系インタフェースの状態

Blu-ray 系インタフェースは、Blu-ray ミドルウェアモジュールに問い合わせを行い、その結果を基に Blu-ray 用ドライブの状態を把握して、自身の内部状態として管理する。

Blu-ray 系インタフェースの内部状態には、「停止中」と「再生中」の 2 種類である。

表 10-2：Blu-ray 系インタフェースの内部状態

内部状態	意味
停止中	Blu-ray 用ドライブは、挿入されている Blu-ray ディスクを回転させておらず、記録されているデータの読み込み等が不可能な状態である。
再生中	Blu-ray 用ドライブは、挿入されている Blu-ray ディスクを回転させており、記録されているデータの読み込み等が可能な状態である。

10.2.2.2.2. Blu-rayミドルウェアモジュールの内部状態

Blu-ray ミドルウェアモジュールは、ドライバを通して、Blu-ray 用ドライブの状態を把握し、自身の内部状態として記憶する。

Blu-ray ミドルウェアモジュールの内部状態は、「停止状態」、「再生移行中」、「再生状態」、「ポーズ中」、「早送り中」、「早戻し中」、「正スロー中」、「逆スロー中」の 8 種類がある。

表 10-3：Blu-ray ミドルウェアモジュールの内部状態

内部状態	意味
停止状態	Blu-ray 用ドライブに挿入されている Blu-ray ディスクは停止しており、回転していない状態。
再生移行中	Blu-ray 用ドライブに挿入されている、停止している Blu-ray ディスクの回転を開始し、再生状態になるまでの間の状態。
再生状態	Blu-ray 用ドライブに挿入されており、回転している Blu-ray

	ディスクから、記録されているデータを読み出し、デコーダに転送している状態。
ポーズ状態	Blu-ray 用ドライブに挿入されている Blu-ray ディスクは回転しているが、データの読み出しておらず、デコーダへの伝送を停止している状態。
早送り中	Blu-ray 用ドライブに挿入されている Blu-ray ディスクは回転しており、データを単位時間ごとに読み出し、デコーダへの伝送している状態。
早戻し中	Blu-ray 用ドライブに挿入されている Blu-ray ディスクは回転しており、データを単位時間ごとに逆向きに読み出し、デコーダへの伝送している状態。
正スロー中	Blu-ray 用ドライブに挿入されている Blu-ray ディスクは回転しており、データの読み出しタイミングを通常の再生の場合より遅く行い、デコーダへの伝送している状態。
逆スロー中	Blu-ray 用ドライブに挿入されている Blu-ray ディスクは回転しており、データの読み出しタイミングを通常の再生の場合より遅く行いながら、データを逆向きに読み出して、デコーダへ伝送している状態。

10.2.2.2.3. Blu-ray系インタフェースとBlu-rayミドルウェアモジュールの内部状態の関係

Blu-ray 系インタフェースの内部状態とBlu-rayミドルウェアモジュールの内部状態の対応関係を表 10-4 に示す。

表 10-4:内部状態の対応関係

Blu-ray 系インタフェースの内部状態	Blu-ray ミドルウェアモジュールの内部状態
停止中	停止状態
再生中	再生移行中
	再生状態
	ポーズ状態
	早送り中
	早戻し中
	正スロー中
	逆スロー中

10.2.2.3. Blu-ray系インタフェースのAPI

Blu-ray 系インタフェースには、Blu-ray アプリケーションがユーザの操作命令を伝達するための API を実装している。

表 10-5：Blu-ray 系インタフェースの API

	API	内容
1	FirstPlay	Blu-ray ディスクに記録されているデータを最初から再生することを要求する
2	Resume	再生中の Blu-ray ディスクの一時停止を要求する。Blu-ray ディスクに記録されているデータ列の中で、停止位置を記録する。
3	Chapter_Search	ユーザが指定するチャプターの再生を要求する。チャプターはBlu-rayディスクに記録されているコンテンツごとに異なる。
4	Normal_Stop	再生中の Blu-ray ディスクの停止を要求する。
5	Switch	Blu-ray ディスクが再生中なら一時停止を要求する。一時停止中の場合は、一時停止の解除を要求する。
6	Fast_Play	早送りを要求する。
7	Fast_Back	早戻しを要求する。
8	Slow_Play	
9	Slow_Back	
10	Flame_Advance	コマ送りを要求する。
11	Cancel_Trick	一時停止・早送り・早戻し・正スロー・逆スローといった特殊再生状態から解除して、通常の再生をすることを要求する。
12	Get_Info	Blu-ray ディスクの再生状態の取得を要求する。
13	Change_Angle	メディアのアングルの切り替えを要求する。
14	Set_Code	音声の言語コード設定を要求する。
15	Get_CallBack_Info	この API は呼び出しが特殊である。コールバック部がデバイスからのコールバックがあったことをアプリケーションプロセスに通知するために用いる。この API では、モード状態の変更が行われる。

10.3. 検証対象システムのモデリング

10.3.1. モデリングの前提

10.3.1.1. ケーススタディの方針（モデリングの方針）

今回の事例では、検証対象となるモード制御モジュール（Blu-ray 系インタフェースおよび Blu-ray ミドルウェア）以外にも様々なコンポーネントがそれぞれ 1 つ以上のスレッドとして並行動作する。例えば、Blu-ray アプリケーションは *n* 個、コールバックは 1 個、ドライバおよび Blu-ray 用ドライブはそれぞれ 1 個のスレッドとして並行動作する。このため、状態爆発やモデルの複雑さに起因する検証結果のトレースが困難になるなどの問題が想定される。そこで、今回のケーススタディでは、検証の目的を明確化した上で、検証に必要な部分を詳細にモデル化する一方、その他の部分は簡略化および統合により、1 つのコンポーネントにするなどして、プロセスを減らすると共に検証対象のモデルを簡略化していく。

今回のケーススタディでは、複数並行動作する Blu-ray 系インタフェースと単一スレッドで動作する Blu-ray ミドルウェアモジュールの内部状態の間で不整合が生じることはないか、不整合が生じた場合に Blu-ray 用ドライブの制御やデッドロックは発生しないか、といった点を検証することを目的とする。このため、Blu-ray 系インタフェースのプロセスと Blu-ray ミドルウェアモジュールのプロセスの間の通信を検証対象の中心に据える。従って、Blu-ray 系インタフェースおよび Blu-ray ミドルウェアモジュールの内部状態とその制御をなるべく詳細にモデリングする。一方、それ以外の部分については、外部環境とみなして簡略化してモデリングする。

10.3.1.2. 検証対象の範囲

Blu-ray 用ドライブに関連する、Blu-ray 系アプリケーション、Blu-ray 系インタフェース、Blu-ray ミドルウェアモジュール、ドライバ、Blu-ray 用ドライブと、Blu-ray ディスクプレイヤを操作する人間を検証対象の範囲としてモデル化する。

Blu-ray ディスクプレイヤ内に付属する DVD、CD などを再生するため制御機構は除外する。

10.3.1.3. モデリングの前提条件

今回の検証では、その検証目的から、Blu-ray ディスクプレイヤに何らかの Blu-ray ディスクは挿入済みであるという前提で考える（前提1）。また、Blu-ray ディスクプレイヤが通常動作している際の、モード制御モジュールの検証を行うため、停電や不慮の電源コンセントの取り外しなどは起こらない前提で考える（前提2）。

10.3.1.4. モデリングの考え方

上述のとおり、本ケーススタディのモデリングの方針では、「Blu-ray 系インタフェースと Blu-ray ミドルウェアモジュールの内部状態と通信についてなるべく忠実にモデル化する」とともに、「それ以外のコンポーネントは、簡略化または統合によりプロセス数を削減する」ということであった。

ここでは、モデルを構成するために、実際の Blu-ray ディスクプレイヤのプロセス構成の単位を

図 10-4 に分けて考えることとした。図 10-4 の赤枠で囲まれた Blu-ray インタフェースと Blu-ray ミドルウェアモジュールは、今回の検証における主要なターゲット部分であるため、実際のシステムの仕様に対して、必要部分を選別後、なるべく忠実にモデル化する。一方で、その他の部分（図 10-4 の青枠部分）については、簡略化または統合してプロセス数を減らす。

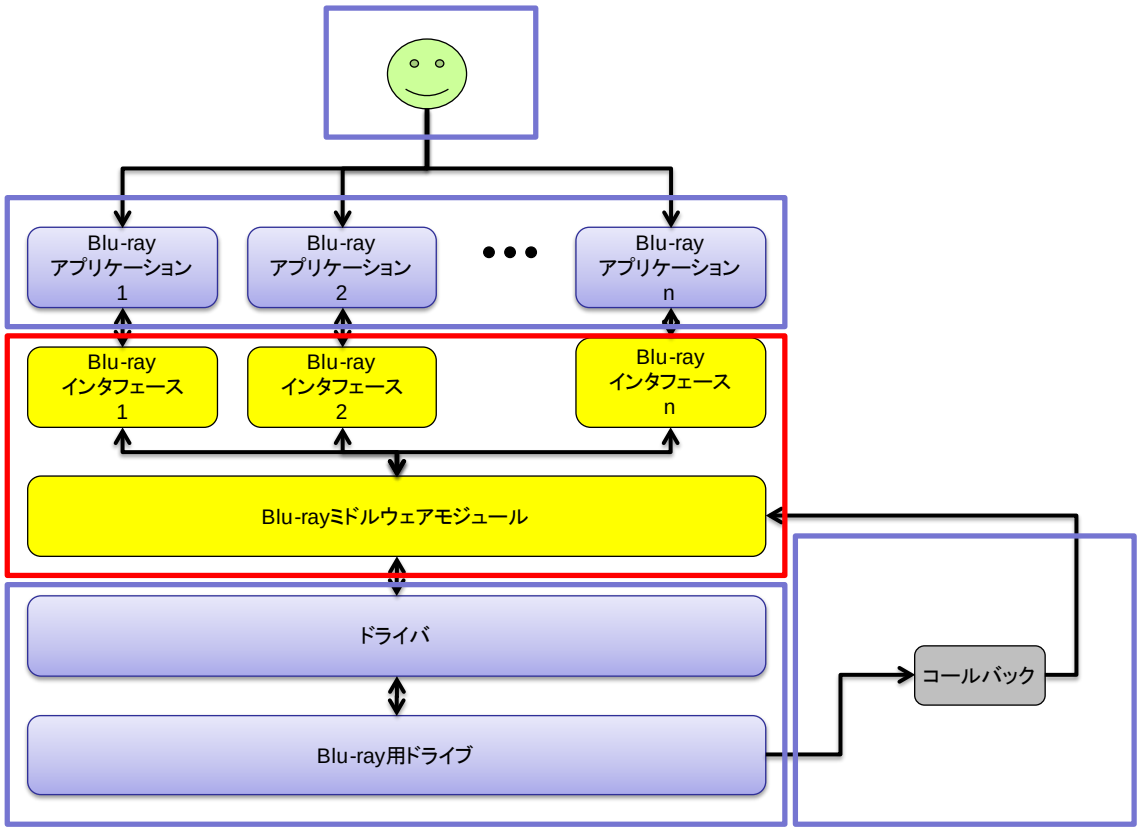

図 10-4：実際のシステムのプロセス構成とモデリングを行う単位

以下、表 10-6 に、図 10-4 のそれぞれの部分についてのモデリングの方針とその理由を概説した。

表 10-6：における各部分のモデリングの方針とその理由

部分	モデリングの方針	理由
ユーザ	外部環境 Blu-ray ディスクドライブに対して、任意の順番で選択可能な命令を送信する	あらゆる命令の組に対して検証を行うため
Blu-ray アプリケーション	Blu-ray インタフェースに統合する	Blu-ray アプリケーションは検証目的の主要部分を構成せず、Blu-ray インタフェースと

		Blu-ray アプリケーションのプロセスは 1 対 1 で対応しているため
Blu-ray インタフェースと Blu-ray ミドルウェア	必要部分を割り出した上で、実際のシステムの制御方法になるべく忠実に記述する	主要な検証対象部分であるため
ドライバと Blu-ray 用ドライブ	1 つのプロセスとして統合する	主要な検証対象部分ではなく、1 対 1 に対応したプロセスであため
コールバック	サイズ m のバッファを持つ 1 つのプロセスとする	割り込みイベントの発生確率は非常に小さいと考えられるため、発生した順番で逐次処理を行うことで十分と考えられるため

モデリングの結果と実際の Blu-ray ディスクプレイヤのプロセス構成の対比を図 10-5 に示す。

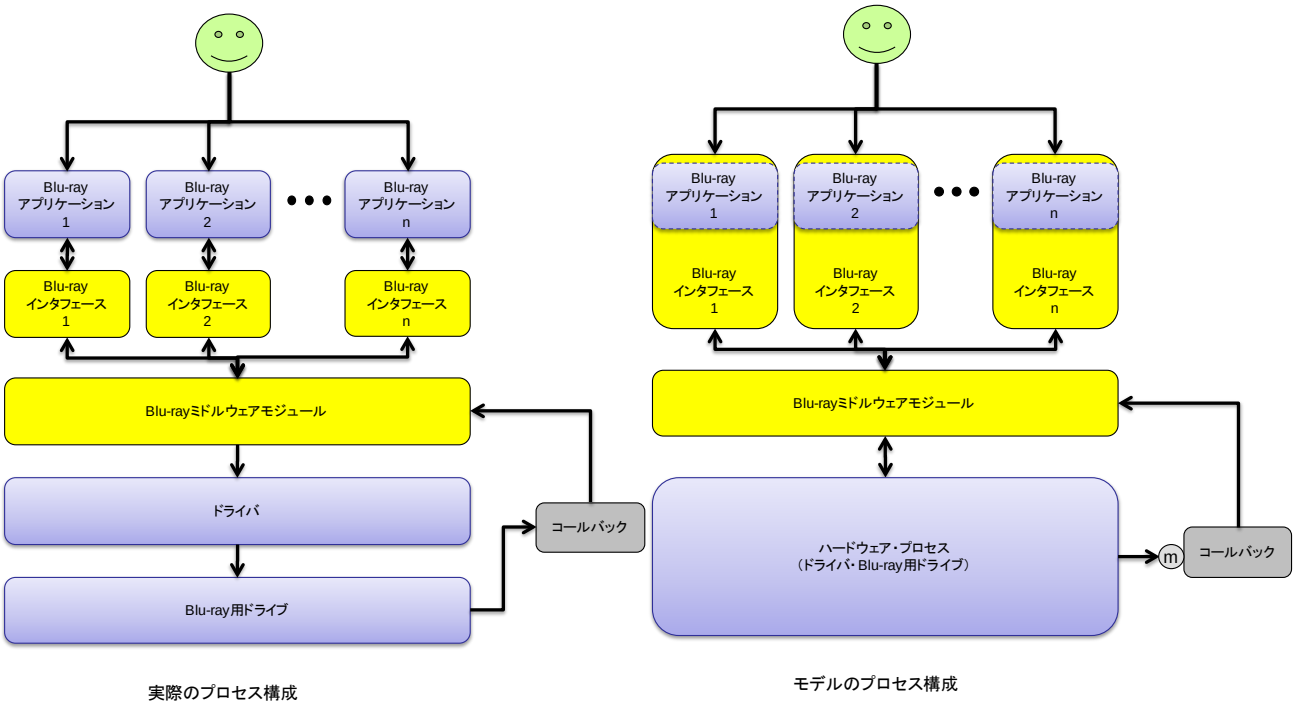

図 10-5:実際のシステム構成とモデルのプロセス構成

10.3.2. モデリングの詳細

10.3.2.1. ユーザのモデリング

Blu-ray ディスクプレイヤからみて、ユーザは自身を操作する命令を、(リモコンなどを通して)送信してくる主体である。ユーザは選択可能な命令を自分の好きな順番、任意のタイミングで選択して送信してくる。このため、検証対象であるモード制御モジュールからみると、ユーザは外部環境として捉えることが可能である。

検証の目的である「Blu-ray 系インタフェースの内部状態とBlu-rayミドルウェアのモジュールの内部状態の不整合の有無」を考える場合、ユーザが選択可能な命令のあらゆる組み合わせにおいて検証することが求められる。後述のように、Blu-ray アプリケーションは Blu-ray 系インタフェースに組み込んで考えるため、ユーザのモデルでは Blu-ray 系インタフェースの API 関数を直接呼び出すことができるとみなし、これをユーザが選択可能な命令（操作）であると考える。

10.3.2.2. Blu-rayアプリケーションのモデリング

Blu-ray アプリケーションは、今回のケーススタディにおいて主要な検証部分ではないため、Blue-ray 系インタフェースに組込み、プロセス数の削減を行うこととした。

10.3.2.3. Blu-ray系インタフェースのモデリング

モードモジュールの状態遷移を定義する状態遷移表を以下に示す。モードモジュールが保持するモード状態は、設計仕様にある通り、「停止中」と「再生中」からなる。モードモジュールの状態遷移は、モード状態に加え、ミドル状態にも基づくため、状態遷移表において状態を示す列には、モード状態とミドル状態の一覧が記載される。状態遷移表の行には、イベントの種類を表す API の一覧が示される。

		status.b.disc_play		play_st（AVC_API.PLAY_ST系）ただし、BDMVの再生状態取得で扱わない状態は除く							
	状態に関わらず行う	停止中	再生中	停止状態	再生状態	再生移行中	ポーズ中	早送り中	正スロー中	早戻し中	逆スロー中
再生 FirstPlay再生	first_playback 再生処理 CALL disc_playを再生中にする アプリに成功を返す	（灰色部は状態によって処理に違いがないことを示す）									
再生 Resume再生 （Resume無し）	／（スラッシュは処理をしないことを示す）	first_playback 再生処理 CALL disc_playを再生中にする アプリに成功を返す	アプリにエラーを返す								
再生 Resume再生 （Resumeあり）	／	再生(再開)処理CALL disc_playを再生中にする アプリに成功を返す (ミドル関数からエラーが返ったときはdisc_playを変更しない・アプリにエラーを	アプリにエラーを返す								
再生 Chapterサーチ	／	アプリにエラーを返す	Chapter Search実行時処理CALL disc_playを再生中にする アプリに成功を返す (ミドル関数からエラーが返ったときはdisc_playを変更しない・アプリにエラーを								
停止 通常停止	／	アプリにエラーを返す	停止処理CALL disc_playを停止中にする アプリに成功を返す								
再生一時停止	／	アプリにエラーを返す	Pause押下処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
早送り	／	アプリにエラーを返す	順方向再生速度変更処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
早戻し	／	アプリにエラーを返す	逆方向再生速度変更処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
正スロー	／	アプリにエラーを返す	順方向再生速度変更処理（スロー）CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
逆スロー	／	アプリにエラーを返す	逆方向再生速度変更処理（スロー）CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
コマ送り	／	アプリにエラーを返す	順方向コマ再生処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
スキップアップ	／	アプリにエラーを返す	NextPoint(チャプターサーチ)処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
スキップダウン	／	アプリにエラーを返す	PrevPoint(チャプターサーチ)処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
トリックプレイ解除	／	アプリにエラーを返す	トリックプレイ解除処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエ								
リピート設定 Chapterリピート	／	アプリにエラーを返す	Chapterリピート関数CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
リピート解除	／	アプリにエラーを返す	リピートクリアCALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)								
レジュームポイントの有無	resume 情報存在チェックCALL アプリにレジューム有無を返す										
レジュームポイントの設定	／	Player 再生情報Registerセット処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)	アプリにエラーを返す								
レジュームポイントのクリア	／	Resume 情報クリア処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返す)	アプリにエラーを返す								
再生情報取得	ミドル情報pm_playから disc_play再設定			disc_playを停止中にする アプリに再生情報を返す	disc_playを再生中にする アプリに再生情報を返す	disc_playを再生中にする アプリに再生情報を返す	disc_playを再生中にする アプリに再生情報を返す	disc_playを再生中にする アプリに再生情報を返す	disc_playを再生中にする アプリに再生情報を返す	disc_playを再生中にする アプリに再生情報を返す	disc_playを再生中にする アプリに再生情報を返す
アングル切り替え	／	アプリにエラーを返す	play_st状態判定を行う	アングル変更処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返	アングル変更処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返	アングル変更処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返	アプリにエラーを返す	アングル変更処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返	アプリにエラーを返す	アングル変更処理CALL アプリに成功を返す (ミドル関数からエラーが返ったときはアプリにエラーを返	アプリにエラーを返す
音声言語コード設定	／	Audio Language セットCALL	アプリにエラーを返す								
コールバック状態取得関数（正常系通知用）	アプリに通知する	／	／	disc_playを停止中にする	disc_playを再生中にする	disc_playを再生中にする	disc_playを再生中にする	disc_playを再生中にする	disc_playを再生中にする	disc_playを再生中にする	disc_playを再生中にする
コールバック状態取得関数（異常系通知用）	アプリに通知する	／	／								

図 10-6：モードモジュールの状態遷移表

10.3.2.4. Blu-rayミドルウェアモジュールのモデリング

ミドルモジュールの状態遷移を定義するミドル状態遷移表を以下に示す。ミドルモジュールは、基本的には、ミドル状態に基づき、モードモジュールから呼出された関数をイベントとして、状態遷移表に示す通りモデリングされる。ただし、ミドルモジュールが受け取るイベントは、モードモジュールからの関数呼び出しに加え、外部環境に相当するドライバからの割込みがある。状態遷移表では、それをモデリングするために、イベントとして、割込みとモードモジュールからの関数呼出しの一覧を示している。

	ミドル状態（表現はモードが解釈したと仮定したもの）							
	停止状態	再生状態	再生移行中	ポーズ中	早送り中	正スロー中	早戻し中	逆スロー中
ミドル側処理（モードから見るといつ発生するかわからない）	／	停止状態に遷移(再生終了)	再生状態に遷移	／	停止状態に遷移	停止状態に遷移	停止状態に遷移	停止状態に遷移
first_playback 再生処理CALL	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す	／	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す
再生(再開)処理CALL	再生移行中状態に遷移 モードに成功を返す	×	×	×	×	×	×	×
Chapter Search実行時処理CALL	×（エラーを返す）	再生移行中状態に遷移 モードに成功を返す	／	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す	再生移行中状態に遷移 モードに成功を返す
停止処理CALL	／（状態遷移なしで成功を返す）	停止状態に遷移 モードに成功を返す	停止状態に遷移 モードに成功を返す	停止状態に遷移 モードに成功を返す	停止状態に遷移 モードに成功を返す	停止状態に遷移 モードに成功を返す	停止状態に遷移 モードに成功を返す	停止状態に遷移 モードに成功を返す
Pause押下処理CALL	×	ポーズ中に遷移 モードに成功を返す	×	再生状態に遷移 モードに成功を返す	ポーズ中に遷移 モードに成功を返す	ポーズ中に遷移 モードに成功を返す	ポーズ中に遷移 モードに成功を返す	ポーズ中に遷移 モードに成功を返す
順方向再生速度変更処理CALL	×	早送り中に遷移 モードに成功を返す	×	早送り中に遷移 モードに成功を返す	／	早送り中に遷移 モードに成功を返す	早送り中に遷移 モードに成功を返す	早送り中に遷移 モードに成功を返す
逆方向再生速度変更処理CALL	×	早戻し中に遷移 モードに成功を返す	×	早戻し中に遷移 モードに成功を返す	早戻し中に遷移 モードに成功を返す	早戻し中に遷移 モードに成功を返す	／	早戻し中に遷移 モードに成功を返す
順方向再生速度変更処理（スロー）CALL	×	正スロー中に遷移 モードに成功を返す	×	正スロー中に遷移 モードに成功を返す	正スロー中に遷移 モードに成功を返す	／	正スロー中に遷移 モードに成功を返す	正スロー中に遷移 モードに成功を返す
逆方向再生速度変更処理（スロー）CALL	×	逆スロー中に遷移 モードに成功を返す	×	逆スロー中に遷移 モードに成功を返す	逆スロー中に遷移 モードに成功を返す	逆スロー中に遷移 モードに成功を返す	逆スロー中に遷移 モードに成功を返す	／
順方向コマ再生処理CALL	×	／	×	／	／	／	／	／
トリックプレイ解除処理CALL	×	／	×	再生状態に遷移 モードに成功を返す	再生状態に遷移 モードに成功を返す	再生状態に遷移 モードに成功を返す	再生状態に遷移 モードに成功を返す	再生状態に遷移 モードに成功を返す
アングル変更処理CALL	×	／	×	×	／	×	／	×
Audio Language セットCALL	／	×	×	×	×	×	×	×

図 10-7:ミドルモジュールの状態遷移表

10.3.2.5. ドライバとBlu-ray用ドライブのモデリング

Blu-ray ミドルウェアモジュールは、ドライバを通して制御する Blu-ray 用ドライブを制御する。つまり、ドライバが適切に動作しているのであれば、Blu-ray 用ドライブはドライバに従って動作すると考えられる。このため、ドライバと Blu-ray 用ドライブを一体とみなし、1 つのプロセスとする。
Blu-ray 用ドライブから発生する割り込みイベントは、挿入されている Blu-ray ドライブに傷がある、Blu-ray ディスクに記録されているデータを終端まで読み込んだなどにより発生する。つまり、上位の Blu-ray 系インタフェースや Blu-ray ミドルウェアモジュールが現在の状態について下位のモジュールに問い合わせても把握できないものである。従って、割り込みイベントは Blu-ray 用ドライブから任意のタイミングで発生するように、つまり、非決定的に発生するものとしてモデリングする。

通常の Blu-ray ディスクでは、1 枚のディスクにおいて最後までデータを読み込んだ場合に「再生終了」の割り込みイベントが発生する。つまり、Blu-ray ディスクに書き込まれているデータ系列の最後まで到達した場合に発生し、Blu-ray ディスクプレイヤが動作している時間軸で考えると発生確率は非常に小さいと考えられる。また、ディスクに傷などがある場合、Blu-ray 用ドライブがデータの読み込みを行っている途中で、ディスクの傷などによりデータが破損している箇所に到達した時に「読み込みエラー」のイベントが発生する。つまり、ディスクに傷があり、データ破損個所に到達した場合に「読み込みエラー」の割り込みイベントが発生し、その発生確率は非常に小さいと考えられる。従って、割り込みイベントの発生確率はいずれの場合も非常に低いと考えてモデリングすることができる。

10.3.2.6. コールバックのモデリング

Blu-ray 用ドライブから発生する割り込みイベントは、予測不能な任意のタイミングで発生する。コールバックは、この割り込みイベントを捕捉し、上位の Blu-ray 用ミドルウェアモジュールに通知する。ここで、上述の通り「個々の割り込みイベントは非常に小さな確率でしか発生しない」というようにモデリングしているため、同時に複数の割り込みイベントが発生する確率も非常に小さくなる。従って、同時に複数の割り込みイベントが発生する場合を捨象しても大きく現実のシステムと乖離するわけではない。そこで、割り込みイベントを発生した順番に m 個格納して逐次処理するようにコールバックをモデル化する。つまり「m 個のバッファを持つ 1 つのプロセス」としてモデル化する。

10.4. 検証性質の形式記述

10.4.1. 検証性質の抽出

本ケーススタディでは、モード制御ソフトウェアが API リクエスト等を処理する過程で、モードモジュールとミドルモジュールの2つのモジュールが独自に保持する状態の不整合などが原因で、意図しない制御やデッドロックなどの発生が懸念されていた。

しかし、設計工程では、どのような不具合が発生するか想定することが難しく、検証すべき性質を明確化することが出来なかった。そのため、形式モデリングの後に、検証性質の形式記述を実施した。

10.4.2. 検証性質の形式記述

設計に関する問題が懸念される点を検証性質の形で記述し、モデル検査により検証する。具体的に検証性質を以下に示す。

10.4.2.1. 状態のズレによる不具合の存在について

モードモジュールとミドルモジュールの状態がずれることにより、予期しない制御の発生やデッドロックが発生しないかどうか検証したい。

このような性質、モデル検査により検証するためには、LTL 式として記述しなければならない。そのために、上記の性質の一部を示すものとして、「ミドルが処理できない要求をモードが流しても、いつかは待機状態に戻る。」を考える。さらにこれを、ミドル状態と、モードからミドルへの要求イベ

ントの具体例について考えることで、検証式を記述する。たとえば、「いつでも、ミドル状態が「停止中」で、かつ、ミドルに「ポーズ」要求が来ても、いつかは待機状態に戻る」を検証すればよい。この性質を構成する要素となる命題には、「ミドル状態が「停止中」」、「ミドルに「ポーズ」要求が来る」、「待機状態に戻る」の3つの条件がある。これらは順に、Promela で記述したモデルにおいては、ミドル状態(Mid_state)が、「停止中」(mid_stop)、モードモジュールからの関数呼出が一時停止(midf_pause)、モード状態(Mode_state)は、「停止中」(mode_stop)になる、ということで記述できるため、それぞれ、Mid_state==mid_stop、Mode_req ? [midf_pause(ch1)、Mode_state == mode_stop のように記述される。また、性質全体のうち、「いつでも」や「いつかは」などのように、時間の順序を表す性質は、時相演算子[]、<>を用いて記述できる。したがって、この性質は形式的には以下のような式で記述することができる。

```
[]((Mid_state==mid_stop) && (Mode_req ? [midf_pause(ch1)])  -> <>(Mode_state == mode_stop))
```

10.4.2.2. モードからミドルへのリクエストの制約

次の検証性質は、モードモジュールは、ミドルモジュールで処理できない関数呼出しを行わないかどうか検証するものである。実際には、設計時に、反例が存在することが予想されていたため、ここでは反例を発見することを目的とした検証である。具体的には、「ミドル状態(mid_state)が「再生移行中」(mid_to_playing)で、かつ、モードモジュールへの要求がポーズである Mode_req ? [midf_pause(ch1)]、ことがない。」ことを検証すればよい。

```
[]!((mid_state == mid_to_playing) && (Mode_req ? [midf_pause(ch1)]))
```

10.4.2.3. モード状態とミドル状態の一致性

モード状態には、「停止中」と「再生中」の 2 種類がある。一方、ミドル状態には、「停止状態」、「再生移行中」、「再生状態」、「ポーズ中」、「早送り中」、「早戻し中」、「正スロー中」、「逆スロー中」の 8 種類がある。モード状態とミドル状態の一致とは、設計上は、モード状態が停止中の時、ミドル状態が停止状態にあり、かつ、モード状態が再生中の時、ミドル状態は、停止状態以外の状態であることと定義する。実際には、設計時点、このような性質には反例があることが予想されており、本検証では、その反例を見つけることが目的である。LTL 式は以下のように記述される。

```
#define s1 (Mode_state == mode_stop)
#define s2 (Mid_state == mid_stop)

#define p1 (Mode_state == mode_play)
#define p2 (Mid_state == mid_play)
#define p3 (Mid_state == mid_to_playing)
#define p4 (Mid_state == mid_pause)
#define p5 (Mid_state == mid_forward)
#define p6 (Mid_state == mid_fslow)
#define p7 (Mid_state == mid_rewind)
#define p8 (Mid_state == mid_bslow)
/*** LTL ***/
![]((s1 && s2) || (p1 && (p2 || p3 || p4 || p5 || p6 || p7 || p8)))
```

s1, s2 は、モード状態とミドル状態がそれぞれ停止状態であることを示す命題である。また、p1 は、モード状態が再生中である命題を示し、p2～p8 は、ミドル状態が停止状態以外の各状態であることを定義する。これらの命題を用いて、LTL式は、最終行のように定義できる。

10.4.2.4. 機能の実行

ある操作を実行したときに、いずれはその機能が実行されるという、通常の機能要求を検証したい。この性質は、状態やイベントを具体化してそれらの組合せとしてインスタンスかされる性質を順に検証する。具体化した一つの例は、「モード状態が「停止中」の時、いつでも「再生」を要求すれば、いずれは再生ミドル関数が実行される。」となる。LTL式で記述すると以下のようになる。

```
[]((Mode_state == mode_stop) && (Mode_req ? [midf_play(ch1)]) -> <>(Mid_state == mid_play))
```

LTL 式の読下すと以下の通りである。モード状態(Mode_state)が、停止状態(mode_stop)でかつ、モードモジュールからミドルモジュールへの関数呼出しが再生(midf_play)である時、いずれはミドルの状態(Mid_state)は、再生(mid_play)となる。

10.4.2.5. その他

モデル検査では、以上のように LTL 式で記述したもの以外に、到達性解析によりデッドロックやデッドロックの存在について検証する。

10.5. モデル検査と結果

10.5.1. 検証結果

検証の結果、以下の(1)～(5)の項目のうち、(4), (5)に関して想定外の問題を発見することができた。(1)～(3)に関しては、予想通りの結果を示すことができた。

10.5.1.1. 状態のズレによる不具合の存在について

以下の性質に関してモデル検査を行った。
「ミドル状態(Mid_state)が、「停止中」(mid_stop)で、かつ、モードモジュールからの関数呼出が一時停止(midf_pause)の時、いずれは、モード状態(Mode_state)は、「停止中」(mode_stop)になる」

モデル検査を実施したところ、反例が見つかったが、それは、LTL 式におけるイベントと状態の値の記述ミスであることが判明した。LTL 式を修正することにより、上記の性質を検証することができた。

10.5.1.2. モードからミドルへのリクエストの制約

予想通りの反例の存在を確認することができ、具体的な反例に至る過程を特定できた。

10.5.1.3. モード状態とミドル状態の一致性

予想通りの反例の存在を確認することができ、具体的な反例に至る過程を特定できた。
この LTL 式に関しても、記述に誤りがあることが判明し、それを修正することにより、期待通りの検証が行われた。

上記の(1), (3)の2つの性質に関する LTL 式の誤りは、要求イベントと状態変数の値の混同によるもので、Promela の記述において状態やイベントを型で区別できないために、構文チェックが行えないことによる。このような点は注意が必要である。

10.5.1.4. 機能の実現性

機能要求として成り立つことが期待されるものであるが、ある操作を要求した際に、いずれそれが実行されることを保証する性質に反例を発見された。これにより、ユーザ操作の系列によっては、ライブロックの問題が発生することが判明した。ユーザの操作イベントによっては、期待した機能に到達しない。具体的には、停止状態から「再生移行中」に移行した段階で、別の操作リクエストが発

生した場合に、再生状態に移行することのなく停止状態になる。この状態が繰り返されることにより、永久に再生に至らないことが発見された。

10.5.1.5.その他の検証

10.5.1.5.1. 到達性解析によるデッドロックの発見

到達性解析によりにデッドロックを発見した。デッドロックは、ミドル処理とデバイス割込み処理の並行プロセスに関して共有変数に対する排他的なアクセスにより発生していることが判明した。

これは、一方が、共有変数をロックし、それを解除する前に、そのプロセスから呼出された別のプロセスが、共有変数のロックを必要とし、デッドロックが発生する。この点の設計仕様を修正することによりデッドロックを解決することができた