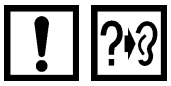
# Power Movie PCI Development Kit2

# Programmer's Manual

**Version 1.1/J**

canopus

## ■ご購入製品を使用される際の注意事項



ここでは、ご購入製品及びプログラムを行うハードウェア(Power Movie PCI)をご使用になられるときにご注意いただきたい事項について説明しています。ご使用方法や、この内容についてご不明な点、疑問点などございましたら、カノープス株式会社テクニカルサポートまでお問い合わせください。

カノープス株式会社
〒651-2241
神戸市西区室谷1-2-2
テクニカルサポート　　TEL.078-992-6830
(祝祭日を除く月～金 10:00～12:00　13:00～17:00)

### ●ご利用目的についての注意事項

医療機器や人命に関るシステムでは、絶対にご利用にならないでください。製品の性質上、これらのシステムへの導入は適しません。

### ●製品の取り付けおよび取り外しに関する注意事項

製品の取り付けや取り外しを行う場合、必ずパソコン本体および周辺機器の電源を切り、さらに電源ケーブルをコンセントから抜いた状態で行ってください。

パソコン本体および周辺機器の電源を入れたまま製品を取り付けたり取り外したりした場合、製品やパソコン本体、周辺機器および周辺機器に接続されている機器の一部が破壊される恐れがあります。 また、パソコン本体および周辺機器の電源ケーブルをコンセントから抜かずにパソコン本体や周辺機器の筐体（電源ユニットなど)、機器の金属部分をさわった場合には感電する恐れがあります。

### ●静電気に関する注意事項

製品に静電気が流れると製品上の部品が破壊される恐れがあります。各コネクタや部品面には直接手をふれないでください。

静電気は衣服や人体からも発生します。製品に触れる前に、一旦接地された金属製のものに触れてください(体内の静電気を放電することになります)。

### ●消費電流に関する注意事項

複数の拡張ボードをパソコンに取り付けるときは、ご購入製品を含めた全ての製品の消費電流の合計がパソコンの最大供給電流を越えていないことを必ず確認してください。全ボードの消費電流の合計がパソコンの最大供給電流を越えたりするなどの動作条件を満たさない環境で使用し続けると、システムが正常に動作しない場合やシステムに負荷がかかり、パソコンが故障する原因となる恐れがあります。

消費電流のわからない製品については、その製品の取扱説明書をご覧いただくか、メーカーに直接お問い合わせいただいてお確かめください。

### ●他社製品と併用されるときの注意事項

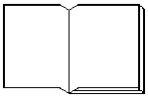
他社製品と併用されるとご購入製品は正常に動作しないことがあり、そのためにシステムが本来の目的を達成することができないこともあります。あらかじめ、 製品単体の環境でご購入製品が正常に動作することをご確認ください。また、他社製品との併用でご購入製品が正常に動作しないのであれば、その他社製品とご購入製品との併用はお止めください。

### ●その他の注意事項

製品は指定された位置に指示通り取り付けてください。指示通りに取り付けられてない場合、製品の金属部とパソコンの金属部が接触してショートするなどの要因で、製品やパソコン本体・周辺機器が破壊される恐れがあります。

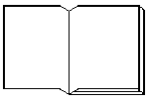
製品を取り扱うときは手など皮膚を傷つけないよう十分にご注意ください。ハードウェアの仕様上、製品のパネル、コネクタ、エッジ、裏面は金属のピンが突出していることがあります。製品を取り付けたり取り外したりするときは、製品全体を軽く包み込むようにお持ちください。

動作中の製品は熱により非常に熱くなります。長時間使用した製品に手を触れる際には、十分ご注意ください。



# ご注意

(1)本製品の一部または全部を無断で複製することを禁止します。
(2)本製品の内容や仕様は将来予告無しに変更することがあります。
(3)本製品は内容について万全を期して作成いたしましたが、万一ご不審な点や誤り、記載漏れなどお気付きの事がございましたら、当社までご連絡ください。
(4)運用した結果については、(3)項にかかわらず責任を負いかねますのでご了承ください。
(5)本製品付属のソフトウェア、マニュアル、その他添付物を含めたすべての関連品に関して、解析、リバースエンジニアリング、デコンパイル、ディスアッセンブリを禁じます。
(6)Power Movie PCI、Power Movie PCI Development Kitはカノープス株式会社の商標です。
(7)MS、Windowsは米国マイクロソフト・コーポレーションの登録商標です。またその他の商品名やそれに類するものは各社の商標または登録商標です。



# 表記について

■本書ではMicrosoft® Windows® operating systemおよびMicrosoft® Windows NT® operating systemを『Windows』、『Windows NT』と表記します。
■本書はWindows上でプログラムを作成することができる方を対象に書かれています。
■本書での説明と実際の運用方法とで相違点がある場合には、実際の運用方法を優先するものとします。プログラム上の基本的な事項、プログラムの方法などについては記載していません。
■本書に記載されていない情報が提供される場合がありますので、ディスクに添付のテキストファイルも必ずお読みください。

# ご留意事項

■ご使用上の過失の有無を問わず、本製品の運用において発生した逸失利益を含む特別、付随的、または派生的損害に対するいかなる請求があったとしても、当社はその責任を負わないものとします。
製品本来の使用目的及び当社が提供を行っている使用環境以外での動作は保証いたしかねます。

Power Movie PCI Development Kit2
Programmer's Manual
Version 1.1/J
September 13, 2001

# 目次

CHAPTER 0

# Preliminaries

# 開発キットの内容

Power Movie PCI Development Kit2(以下開発キットと記載します)は、ビデオの映像をパソコンのディスプレイにオーバーレイ表示させたり、表示されている映像を静止画として取り込んだりする Power Movie PCI の機能をアプリケーションに提供するコンポーネントです。

本開発キットのオーバーレイ表示は２種類用意されていますので使用目的に応じて指定してください。

Super Overlay　：グラフィックカードのオンスクリーンメモリに直接データを書き込むことでオーバーレイ表示します。
★ウィンドウサイズは 160 × 120 ～ 640 × 480 ピクセルとなります。

DirectDraw Overlay　：オフスクリーンメモリにデータを転送し、グラフィックカードの DirectDraw 機能を用いてオーバーレイ表示します。

この開発キットには以下に示すように、Power Movie PCI に準じる画像オーバーレイや静止画キャプチャ、ビデオ出力デバイス用のソフトウェア開発キット(MGEAPI)が含まれています。MGEAPI はインクルードファイル、ライブラリ、およびサンプルプログラムで構成されています。

(2001 年 8 月現在、MGEAPI に対応しているデバイスは Power Movie PCI のみです。)

Visual Basic 用コードモジュール

VB¥include¥movgear.bas
VB¥include¥mgeerror.bas

C/C++ 用インクルードファイル

VC¥include¥movgear.h
VC¥include¥mgeerror.h

C/C++ 用ライブラリファイル

VC¥lib¥mge.lib

Visual Basic サンプルプログラム

VB¥Samples¥capture　・・・ キャプチャ
VB¥Samples¥Overlay　・・・ オーバーレイ
VB¥Samples¥Videoout　・・・ ビデオ出力

C 言語サンプルプログラム

VC¥Samples¥capture　・・・ キャプチャ
VC¥Samples¥Overlay　・・・ オーバーレイ
VC¥Samples¥Videoout　・・・ ビデオ出力

# 開発キットのインストール

開発キットのインストールは専用のセットアッププログラムを用いて行います。以下に標準的な環境を想定したインストールの手順を説明しています。
お使いのパソコンの環境に合わせて適宜読み替えを行いながら、手順に従ってインストールを行ってください。

本開発キットは Windows 95/98/Me/NT 4.0 ServicePack3 以上 /2000 ServicePack1 以上環境に対応しております。また、作成されたプログラムの実行には前記Windows環境が必要です。

★Windows NT/2000 ドライバは Video for Windows 対応アプリケーションからは利用できません。
★Windows Me/2000 環境でお使いの場合の当社サポートは、提供製品の使用可能な範囲内のみとなります。
★Windows 98/Me/2000 環境でのマルチモニタはサポートしておりません。
★Windows NT/2000 Single CPU 環境にのみ対応しています。その他使用環境は Power Movie PCI 本体に準じます。

**〈Windows 95/98/Me 環境へのインストール手順〉**

1. Windows を起動します。
2. 本開発キット同梱の『Power Movie PCI Development Kit Development Kit』ディスクをフロッピーディスクドライブにセットします。
3. スタートメニューから『ファイル名を指定して実行』を選択します。
4. ディスクを挿入したドライブ名に続けて『SETUP.EXE』と入力し、セットアッププログラムを起動します。
5. 画面の指示に従ってインストールを継続します。

☆Windows Me環境でご使用時の当社サポートは提供製品の使用可能な範囲内のみとなります。

**〈Windows NT 4.0 環境へのインストール手順〉**

1. Windows を起動し、administrator もしくは administrator の権限を持ったユーザー ID でログオンします。
2. 本開発キット同梱の『Power Movie PCI Development Kit NT Driver Disk』ディスクをフロッピーディスクドライブにセットします。
3. マイコンピュータからフロッピードライブを開き、[Pmpnt.inf] のファイル（アイコン）上でマウスを右クリックし、「インストール」を選択します。
4. セットアップ完了のダイアログが表示されるので、設定を有効にするためにディスクを取り出し、コンピュータを再起動します。
5. 再起動後、『Power Movie PCI Development Kit Development Kit』ディスクをフロッピーディスクドライブにセットします。
6. スタートメニューから『ファイル名を指定して実行』を選択します。
7. ディスクを挿入したドライブ名に続けて『SETUP.EXE』と入力し、セットアッププログラムを起動します。
8. 画面の指示に従ってインストールを継続します。

☆Power Movie PCI 付属の『Application Disk(Ver1.02 以降)』をセットアップすることにより、「Video Magic」および、「Slide Theater」が Windows NT 4.0 環境でお使いいただけます。

## 〈Windows 2000 環境へのインストール手順〉

1. カノープスホームページ（http://www.canopus.co.jp/download/pmovpci.htm）からWindows 2000 用 INFファイル PMPW2KINF.EXE をダウンロードします。
2. Windows を起動し、administratorもしくはadministratorの権限を持ったユーザーIDでログオンします。
3. 1. のプログラムを解凍し作成されたファイルと本開発キット同梱の『Power Movie PCI Development Kit NT Driver Disk』ディスクのファイルをハードディスクに新しいフォルダを作成しコピーします。
4. ［マイコンピュータ］を右クリックして［管理］を選択します。
5. ［ツリー］から［デバイスマネージャー］をクリックします。
6. ［その他のデバイス］の［マルチメディア コントローラ］をダブルクリックします。
7. ［ドライバ］タブの［ドライバの更新］をクリックします。
8. ［デバイスドライバのアップグレードウィザードの開始］では［次へ］をクリックします。
9. ［ハードウェア デバイス ドライバのインストール］では［デバイスに最適なドライバを検索する］を選択して［次へ］ボタンをクリックします。
10. ［ドライバファイルの特定］では［検索場所のオプション］として［場所を指定］にのみチェックし、［次へ］をクリックします。
11. ［製造元のファイルのコピー元］に 3. で作成したフォルダ名を指定して［OK］をクリックします。［参照］ボタンをクリックして指定することもできます。
12. ［ドライバ ファイルの検索］では［次へ］をクリックします。
13. ここで、［デジタル署名が見つかりませんでした］というダイアログが出ますが［はい］ボタンをクリックします。
14. ［デバイスドライバのアップグレードウィザードの完了］では［完了］ボタンをクリックします。
15. ［閉じる］をクリックします。
16. ディスクを取り出し、指示にしたがって Windows 2000 を再起動します。

    ☆解凍ファイル中のReadme.htmに詳しいインストール手順が記載されておりますので、併せてご参照ください。
17. 再起動後、『Power Movie PCI Development Kit Development Kit』ディスクをフロッピーディスクドライブにセットします。
18. スタートメニューから『ファイル名を指定して実行』を選択します。
19. ディスクを挿入したドライブ名に続けて『SETUP.EXE』と入力し、セットアッププログラムを起動します。
20. 画面の指示に従ってインストールを継続します。

    ☆Power Movie PCI付属の『Application Disk(Ver1.02以降)』をセットアップすることにより、「Video Magic」および、「Slide Theater」がWindows 2000環境でお使いいただけます。当社サポートは提供製品の使用可能な範囲内のみとなります。

# Visual Basic でのアプリケーション開発

Visual Basic のプロジェクトに次のコードモジュールを追加してください。

movgear.bas（各種定義が書かれているファイルです。）
mgeerror.bas（エラーが定義されているファイルです。）

上記のコードモジュールでは、戻り値を持たない関数でも Function プロシージャとして宣言されています。

# Visual C++ でのアプリケーション開発

● 開発環境

プロジェクトで参照されるインクルードファイルおよびライブラリファイルのフォルダに以下のパスを追加してください。

（開発キットが組み込まれているフォルダを C:¥mgesdk¥ であると想定しています。）

| | |
|---|---|
| インクルードファイル | C:¥mgesdk¥vc¥include |
| ライブラリ | C:¥mgesdk¥vc¥lib |

● コンパイル

次のファイルをインクルードしてください。

movgear.h（各種定義が書かれているファイルです。）
mgeerror.h（エラーが定義されているファイルです。）

● リンク

次のライブラリをリンクしてください。

mge.lib

# サンプルプログラム

開発キットにはこのドキュメントに記載されているファンクションの具体的な使用例として Visual Basic および Visual C++ に対応したサンプルが含まれています。作成しようとしているアプリケーションに組み込む機能に応じて各サンプルを参照してください。
サンプルプログラムの内容は C:¥mgesdk¥SAMPLES.TXT を参照してください。
（開発キットが組み込まれているフォルダを C:¥mgesdk¥ であると想定しています。）

| | |
|---|---|
| Capture | キャプチャプログラム |
| Overlay | 2 画面 Overlay |
| Videoout | Video 出力プログラム |

Visual C++ 用のサンプルは C 言語で記述されています。

# 開発キットに含まれるソースコードの取り扱いについて

この開発キットに含まれるサンプルプログラムはドキュメント（本書）を補うための資料となっています。サンプルプログラムのソースコード（以下、「ソースコード」といいます）自体を改変したり、その一部をお客様のアプリケーションに組み込んで利用してください。ただし、お客様のアプリケーションとそこに組み込まれたソースコードの一部との適合性に関してサポートの範囲を限定させていただくこともございますのでご了承ください。

カノープス株式会社はソースコードの使用と変更に関して完全に自由な権利をお客様に許諾いたします。ただし、お客様の営利を目的としたソフトウェア製品にこれらのコードをソースコードもしくはこれを変更したものの形態のままで含めることはご遠慮願います。

また、最終的に作成されたアプリケーションの運用結果および目的への適合性につき、カノープス株式会社では一切の責任を負いかねますので予めご了承ください。

# プログラム作成時の注意事項

この開発キットでプログラムを作成される場合は、次の事項にご注意ください。

1．本開発キットの対応言語は次の通りです。
Visual Basic　Version 4.0 以降(32bit 版のみ)
Visual C++　　Version 4.0 以降

2．本開発キットの開発環境対応ＯＳは次の通りです。
マイクロソフト社製 Windows® 95 operating system 日本語版
マイクロソフト社製 Windows® 98 operating system 日本語版
マイクロソフト社製 Windows® Me operating system 日本語版
マイクロソフト社製 Windows NT® 4.0 operating system 日本語版+ServicePack3 以上
マイクロソフト社製 Windows® 2000 operating system 日本語版+ServicePack1 以上
★作成されたプログラムを Windows Me/2000 環境で動作させる場合の当社サポートは、提供製品の使用可能な範囲内のみとなります。

3．本開発キットには Windows® 98/95/Me 用ドライバは含まれておりません。Power Movie PCI 本体のフロッピーディスクに含まれるモジュールをお使いください。
Windows NT 4.0/2000 用ドライバは本開発キットに含まれております。

4．プログレッシブ JPEG でエンコードされたファイルは扱えません。

5．本開発キットを使用して複数枚の Power Movie PCI を制御することはできません。

6．複数のアプリケーションからの同時使用はできません。

7．MultiThread には対応していません。

8．ハードウェアの制限により、オーバーレイ表示の上部 2 ラインに同じ内容が繰り返し表示されます。

9．本開発キットは従来当社から販売しております Power Movie MP/V 開発キットおよび Power Movie 開発キットとの互換性はありません。

10. Unicode には、対応していません。

１０．Power Movie PCIを複数枚ご購入の上、本製品をWindows NT/2000環境でご使用いただく場合、Power Movie PCIボード本体の数量分だけ『Power Movie PCI Development Kit NT Driver Disk』ディスクを複製し、使用することができます。本製品を使用して開発したアプリケーションに、ドライバディスク(『Power Movie PCI Development Kit NT Driver Disk』)を添付する必要がある場合、配布に関する内容は当社までお問い合わせください。

１１．本開発キットに関するご質問はE-mailにて承っております。お電話、Faxでのお問い合わせについては受付できませんのであらかじめご了承ください。

インターネット E-mail SDK@canopus.co.jp

お問い合わせの際には発生現象と共に次の内容を必ず記載してください。

１．使用しているモジュールのタイムスタンプ

MGE.DLL / PVJPEG.DLL / PVMJPEG.DLL

PMP16.DLL / PMP32.DLL / PMP.VXD (Windows 95/98/Meの場合)

MGEHOOK.DLL / PAVAPI.DLL / PMPKRNL.SYS (Windows NT/2000の場合)

２．使用しておられる開発環境

・Windows のバージョン

・コンパイラのメーカー、バージョン

・使用されているその他の開発キット

３．ハードウェア環境

・PC本体メーカーおよび機種名

・周辺機器メーカーおよび型番

CHAPTER 1

# Tutorial

# 1．ビデオ表示アプリケーションの作成

ビデオ表示アプリケーションでは、

Power Movie PCI の使用開始
ビデオウィンドウの生成
ビデオ入力の確認 ＊
ビデオ映像の表示開始
入力画像のフィールド選択 ＊
入力画質の調整 ＊
ウィンドウの移動およびサイズの変更
ビデオ映像の取り込みおよび静止 ＊
ビデオ映像の表示終了
ビデオウィンドウの破棄
Power Movie PCI の使用終了

などの機能が必要となります。＊は必須項目ではありません。必要に応じて組み込みください。

次ページから DirectDraw Overlay を使用し、ビデオ映像をウィンドウ内に表示するアプリケーションの作成について説明します。本文中では **C** 言語を用いて説明していますが、**Visual Basic** でも同様のシーケンスでコーディングを行います。
また、エラーチェックは省いてあります。

# Power Movie PCIの使用開始

Power Movie PCIを使用するアプリケーションは、その使用開始をドライバに許可してもらう必要があります。アプリケーションの初期化時に **MGE_Initialize** および **MGE_SetCooperativeLevel** ファンクションを呼び出してPower Movie PCIの使用を開始します。

```
HWND g_hWndApp;     // Applicationのメインウインドウハンドル(グローバル変数)
HMGE g_hmge;        // Power Movie PCIの使用権を保持するハンドル(グローバル変数)
MGERESULT mr;

    mr = MGE_Initialize(0, g_hWndApp, &g_hmge);
    if(mr != MGE_S_OK) {
        // Power Movie PCIが使用できない

        エラー処理
    }
    mr = MGE_SetCooperativeLevel(g_hmge, MGE_SCL_EXCLUSIVE);
    if(mr != MGE_S_OK) {
        // Power Movie PCIが使用できない

        エラー処理
    }
```

# ビデオウィンドウの生成

アプリケーションのウィンドウ生成終了後( CreateWindow 実行後)、クライアント領域を Video Window として使用します。
ただし、DirectDraw Overlay の制限に触れるような場合は事前にウィンドウのサイズを変更しておく必要があります。

```
HMGEVSRC      g_hmgevsDDraw;          // Video Decoder のハンドル(グローバル変数)
HMGESURFACE   g_hmgesrfSurfaceDDraw;  // DDraw Overlay Surface のハンドル(グローバル変数)
HMGESTREAM    g_hmgestmStreamDDraw;   // DDraw Overlay Stream のハンドル(グローバル変数)

    RECT      rcCrop, rcFrame, rcWindow, rcView;
    RECT      rcClient;
    DWORD     dwWindowWidth;
    DWORD     dwViewWidth;
    DWORD     dwWindowHeight;
    DWORD     dwViewHeight;
    DWORD     dwCropWidth;
    DWORD     dwCropHeight;

        // 現在の Client 領域の大きさを取得
    GetClientRect(g_hWndApp, &rcClient);
    ClientToScreen(g_hWndApp, (LPPOINT)&rcClient);
    ClientToScreen(g_hWndApp, (LPPOINT)&rcClient + 1);

        // Crop 用の領域初期化
    rcCrop.left = 0;
    rcCrop.top = 0;
    rcCrop.right = 640;
    rcCrop.bottom = 480;

        // Frame 用の領域初期化
    rcFrame.left = 0;
    rcFrame.top = 0;
    rcFrame.right = 640;
    rcFrame.bottom = 480;

        // View 用の領域初期化
    rcView.left = 0;
    rcView.top = 0;
    rcView.right = 640;
    rcView.bottom = 480;
```

```
    // Window用の領域初期化, Client領域を使用
rcWindow = rcClient;

    // DirectDraw Overlayの制限を回避するために
    // 各矩形を補正する。
dwWindowWidth = rcWindow.right - rcWindow.left;
dwViewWidth = rcView.right - rcView.left;
dwCropWidth = rcCrop.right - rcCrop.left;

while(dwViewWidth > dwWindowWidth) {
    dwViewWidth /= 2;
    if(dwViewWidth * 4 < dwCropWidth) {
        /* この状況では実現できない */
        return;
    }
}
rcView.right = rcView.left + dwViewWidth;

dwWindowHeight = rcWindow.bottom - rcWindow.top;
dwViewHeight = rcView.bottom - rcView.top;
dwCropHeight = rcCrop.bottom - rcCrop.top;

while(dwViewHeight > dwWindowHeight) {
    dwViewHeight = dwViewHeight / 2;
    if(dwViewHeight * 8 < dwCropHeight) {
        /* この状況では実現できない */
        return;
    }
}
rcView.bottom = rcView.top + dwViewHeight;

//
// DirectDraw Overlay
//
MGE_CreateVideoDecoder(g_hmge, MGEChannel2, &g_hmgevsDDraw);
MGE_CreateVideoInStream(g_hmgevsDDraw, &g_hmgestmStreamDDraw);
MGE_CreateDDrawOvlSurface(g_hmge, &g_hmgesrfSurfaceDDraw, g_hWndApp, 640, 480);
MGE_VStreamConnect(g_hmgestmStreamDDraw, g_hmgesrfSurfaceDDraw);
MGE_VStreamSetRect(g_hmgestmStreamDDraw, MGEStreamRectCrop, &rcCrop, FALSE);
MGE_VStreamSetRect(g_hmgestmStreamDDraw, MGEStreamRectFrame, &rcFrame, FALSE);
MGE_VStreamSetRect(g_hmgestmStreamDDraw, MGEStreamRectView, &rcView, FALSE);
MGE_VStreamSetRect(g_hmgestmStreamDDraw, MGEStreamRectWindow, &rcWindow, TRUE);
```

# ビデオ入力の確認

ビデオ入力が正しく行われているかどうかを確認します。
本サンプルプログラムでは入力されていない場合に確認を促すメッセージを表示するのみで、再度の確認は行いません。

```
MGELINE       mgl;
MGEVDSTATUS   mgs;

MGE_VDGetLine(g_hmgevsDDraw, &mgl);
mgs.dwSize = sizeof(mgs);
MGE_VDGetStatus(g_hmgevsDDraw, mgl, &mgs);
if((mgs.dwSignalState & MGE_SS_M_STABLE) == MGE_SS_NOTSTABLE)
    MessageBox(NULL, "ビデオが入力されていません", "注意", MB_OK);
```

# ビデオ映像の表示開始

ビデオ映像をアプリケーションウィンドウのクライアント領域に表示させます。
他のウィンドウが上に重なった場合に、オーバーレイされるビデオ映像がそのウィンドウの上に表示されないようにキーカラー（Color Key）を有効にします。

キーカラーを有効にするためには、ウィンドウのクライアント領域をキーカラーで塗りつぶす必要があります。これはウィンドウのバックグラウンドカラーをキーカラーと同じ色にすることで簡単に実現できます。
ビデオオーバーレイの表示開始後にウィンドウ内のキーカラーで塗りつぶすようにすれば、キーカラーそのものの色を見えないようにすることもできます。

```
MGECOLORKEY    ck;
HBRUSH         hBrush;

    // キーカラーをマゼンタにして表示を開始する
ck.dwColorLow = 0x00ff00ff; // マゼンタ
ck.dwColorHigh = 0x00ff00ff;

MGE_SurfaceSetColorKey(g_hmgesrfSurfaceDDraw, &ck);
MGE_VStreamStart(g_hmgestmStreamDDraw);
MGE_SurfaceStartOverlay(g_hmgesrfSurfaceDDraw);

hBrush = CreateSolidBrush(RGB(255, 0, 255));
SetClassLong(g_hWndApp, GCL_HBRBACKGROUND, (LONG)hBrush);
InvalidateRect(g_hWndApp, NULL, TRUE);
UpdateWindow(g_hWndApp);
```

# 入力画像のフィールド選択

入力画像のフィールドを選択します。

偶数フィールドを選択する場合

```
MGE_VStreamSetVideoMode(g_hmgestmStreamDDraw, MGEVideoModeEvenField, TRUE);
```

奇数フィールドを選択する場合

```
MGE_VStreamSetVideoMode(g_hmgestmStreamDDraw, MGEVideoModeOddField, TRUE);
```

両フィールド(フレーム)を選択する場合

```
MGE_VStreamSetVideoMode(g_hmgestmStreamDDraw, MGEVideoModeFrame, TRUE);
```

# 入力画質の調整

明るさなど入力画質の調整を行います。

下記のコード片では現在入力されている明るさに 10 を加えています。
新しい明るさがシステムに対する設定可能値を超えている場合は、設定可能な最小の値を使用します。
このコード片自体には意味はありませんが、この応用によりアプリケーションから入力画質の調整ダイアログなどを作成することができます。

```
MGELINE   mgl;
DWORD     dwMin, dwMax, dwDefault, dwVal;

MGE_VDGetLine(g_hmgevsDDraw, &mgl);
MGE_VDGetParamRange(g_hmgevsDDraw, mgl, MGEVideoAmpBrightness,
                    &dwMin, &dwMax, &dwDefault);
MGE_VDGetParam(g_hmgevsDDraw, mgl, MGEVideoAmpBrightness, &dwVal);
dwVal += 10;
if(dwVal >= dwMax)
     dwVal = dwMin;
MGE_VDSetParam(g_hmgevsDDraw, mgl, MGEVideoAmpBrightness, dwVal);
```

また、アプリケーション中に"規定値に戻す"ボタンなどを実装する場合は、下記の様に **MGE_VDGetParamRange** で返されるデフォルト値を使用することで実現できます。

```
MGELINE   mgl;
DWORD     dwMin, dwMax, dwDefault;

MGE_VDGetLine(g_hmgevsDDraw, &mgl);
MGE_VDGetParamRange(g_hmgevsDDraw, mgl, MGEVideoAmpBrightness,
                    &dwMin, &dwMax, &dwDefault);
MGE_VDSetParam(g_hmgevsDDraw, mgl, MGEVideoAmpBrightness, dwDefault);
```

# ウィンドウの移動およびサイズ変更

ユーザーの操作により、アプリケーションのウィンドウを移動させた時や、ウィンドウのサイズを変更した場合には、ビデオウィンドウを新しい位置に移動させたり、サイズを変更します。

```
case WM_WINDOWPOSCHANGED:
     {
     RECT      rcWindow;
     RECT      rcView;
     RECT      rcCrop;
     DWORD     dwWindowWidth;
     DWORD     dwViewWidth;
     DWORD     dwWindowHeight;
     DWORD     dwViewHeight;
     DWORD     dwCropWidth;
     DWORD     dwCropHeight;

     GetClientRect(g_hWndApp, &rcWindow);
     ClientToScreen(g_hWndApp, (LPPOINT)&rcWindow);
     ClientToScreen(g_hWndApp, (LPPOINT)&rcWindow + 1);

     MGE_VStreamGetRect(g_hmgestmStreamDDraw, MGEStreamRectCrop, &rcCrop);
     rcView = rcCrop;

          // DirectDraw Overlayの制限を回避するために
          // 各矩形を補正する。
     dwWindowWidth = rcWindow.right - rcWindow.left;
     dwViewWidth = rcView.right - rcView.left;
     dwCropWidth = rcCrop.right - rcCrop.left;

     while(dwViewWidth > dwWindowWidth) {
          dwViewWidth /= 2;
          if(dwViewWidth * 4 < dwCropWidth) {
               /* この状況では実現できない */
               goto do_other_things;
          }
     }
     rcView.right = rcView.left + dwViewWidth;
```

```
dwWindowHeight = rcWindow.bottom - rcWindow.top;
dwViewHeight = rcView.bottom - rcView.top;
dwCropHeight = rcCrop.bottom - rcCrop.top;

while(dwViewHeight > dwWindowHeight) {
     dwViewHeight = dwViewHeight / 2;
     if(dwViewHeight * 8 < dwCropHeight) {
     if(dwViewWidth * 4 < dwCropWidth) {
          /* この状況では実現できない */
          goto do_other_things;
     }
}
rcView.bottom = rcView.top + dwViewHeight;

     // View と Frame は同じ
MGE_VStreamSetRect(g_hmgestmStreamDDraw, MGEStreamRectFrame, &rcView, FALSE);
MGE_VStreamSetRect(g_hmgestmStreamDDraw, MGEStreamRectView, &rcView, FALSE);
MGE_VStreamSetRect(g_hmgestmStreamDDraw, MGEStreamRectWindow, &rcWindow, TRUE);
}
do_other_things:;
```

# ビデオ映像の取り込みおよび静止

ビデオ映像の取り込みおよび静止を行ないます。
この例では現在の状態を取得して、取り込みと静止を交互に切り替えます。

```
MGESTREAMSTATE ss;

MGE_VStreamGetState(g_hmgestmStreamDDraw, &ss);
if(ss == MGEStreamStatePause) {
    MGE_VStreamRestart(g_hmgestmStreamDDraw);
}
else if(ss == MGEStreamStateStarted) {
    MGE_VStreamPause(g_hmgestmStreamDDraw);
}
else {
    // ビデオがスタートしていない。
}
```

# ビデオ映像の表示終了

ビデオ映像の表示を終了させます。

ビデオオーバーレイの表示終了前にウィンドウ内を適当な色で塗りつぶすようにすれば、キーカラーそのものの色を見えないようにすることができます。

```
HBRUSH    hBrush;

    // バックグラウンドカラーを黒にする
hBrush = GetStockObject(BLACK_BRUSH);
hBrush = (HBRUSH)SetClassLong(g_hWndApp, GCL_HBRBACKGROUND, (LONG)hBrush);
InvalidateRect(g_hWndApp, NULL, TRUE);
UpdateWindow(g_hWndApp);

    // 表示を終了させる。
MGE_SurfaceStopOverlay(g_hmgesrfSurfaceDDraw);
MGE_VStreamStop(g_hmgestmStreamDDraw);

    // キーカラーのブラシを削除する
if(hBrush)
    DeleteObject(hBrush);
```

# ビデオウィンドウの破棄

ビデオウィンドウを破棄します。WM_DESTROY などで実行します。

```
MGE_VStreamDisconnect(g_hmgestmStreamDDraw, g_hmgesrfSurfaceDDraw);
MGE_DestroyVideoStream(g_hmgestmStreamDDraw);
MGE_DestroySurface(g_hmgesrfSurfaceDDraw);
MGE_DestroyVideoDecoder(g_hmgevsDDraw);
```

# Power Movie PCI の使用終了

Power Movie PCI を使用するアプリケーションは、その使用終了をドライバに通知する必要があります。アプリケーションの終了時などに **MGE_Uninitialize** ファンクションを呼び出して Power Movie PCI の使用を終了します。

```
MGE_Uninitialize(g_hmge);
```

# ２．ビデオ出力アプリケーションの作成

ビデオ出力アプリケーションでは

Power Movie PCI の使用開始
ビデオへの出力の準備
ビデオへ出力するものを描画する
ビデオへの出力の開始
ビデオへの出力を切り替える　＊
ビデオへの出力の終了
ビデオへの出力の後処理
Power Movie PCI の使用終了

などの機能が必要となります。＊は必須項目ではありません。必要に応じて組み込みください。

次ページから静止画像をビデオ出力するアプリケーションの作成について説明します。本文中では **C** 言語を用いて説明していますが、**Visual Basic** でも同様のシーケンスでコーディングを行います。
また、エラーチェックは省いてあります。

# Power Movie PCIの使用開始

Power Movie PCIを使用するアプリケーションは、その使用開始をドライバに許可してもらう必要があります。アプリケーションの初期化時に **MGE_Initialize** および**MGE_SetCooperativeLevel** ファンクションを呼び出してPower Movie PCIの使用を開始します。

```
HWND g_hWndApp;     // Applicationのメインウインドウハンドル(グローバル変数)
HMGE g_hmge;        // Power Movie PCIの使用権を保持するハンドル(グローバル変数)
MGERESULT mr;

     mr = MGE_Initialize(0, g_hWndApp, &g_hmge);
     if(mr != MGE_S_OK) {
          // Power Movie PCIが使用できない

          エラー処理
     }
     mr = MGE_SetCooperativeLevel(g_hmge, MGE_SCL_EXCLUSIVE);
     if(mr != MGE_S_OK) {
          // Power Movie PCIが使用できない

          エラー処理
     }
```

# ビデオへの出力の準備

静止画像をビデオ出力する準備を行ないます。

```
HMGEVDST      g_hmgevdVideoOut;       // Video Encoder へのハンドル(グローバル変数)
HMGESTREAM    g_hmgestmStreamVideoOut;// Video out Stream へのハンドル(グローバル変数)
HMGESURFACE   g_hmgesrfSurfaceMemory; // Memory Surface へのハンドル(グローバル変数)

    RECT      rcWindow, rcView;
    MGEPFINFO pfinfo;

    rcView.left = 0;
    rcView.top = 0;
    rcView.right = 640;
    rcView.bottom = 480;

    rcWindow.left = 0;
    rcWindow.top = 0;
    rcWindow.right = 640;
    rcWindow.bottom = 480;

        //
        // Video out するための各オブジェクトの生成
        //
    MGE_CreateVideoEncoder(g_hmge, MGEChannel1, &g_hmgevdVideoOut);
    MGE_CreateVideoOutStream(g_hmgevdVideoOut, &g_hmgestmStreamVideoOut);
    pfinfo.dwSize = sizeof(pfinfo);
    MGE_GetPixelFormatInfo(MGEPixelFormatYUY2, &pfinfo);
    MGE_CreateMemorySurface(g_hmge, &g_hmgesrfSurfaceMemory,
                            640, 480, 640 * pfinfo.dwBytesPerPixel,
                                    MGEPixelFormatYUY2);
    MGE_VStreamConnect(g_hmgestmStreamVideoOut, g_hmgesrfSurfaceMemory);
    MGE_VStreamSetRect(g_hmgestmStreamVideoOut,
                            MGEStreamRectView, &rcView, FALSE);
    MGE_VStreamSetRect(g_hmgestmStreamVideoOut,
                            MGEStreamRectWindow, &rcWindow, TRUE);
```

# ビデオへ出力するものを描画する

ビデオへ出力するものを用意します。ここで **MGE_SurfaceFillBuffer**、**MGE_SurfaceSetBitmapBits**、 **MGE_SurfaceLoadDIB** や **MGE_SurfaceLoadJPEG** などを使用して、オブジェクト（静止画）をサーフェスに描画します。
ここでは **MGE_SurfaceFillBuffer** を使用します。

```
DWORD     s_ardwColor[] = {
          0x00ffffff,
          0x00ff0000,
          0x0000ff00,
          0x000000ff,
          0x00000000,
          0x00ffff00,
};
RECT rc;
int  i;

rc.left = 0;
rc.top = 0;
rc.right = 640;
rc.bottom = 480;

for(i = 0;i < 6;i++) {
     MGE_SurfaceFillBuffer(g_hmgesrfSurfaceMemory, &rc, s_ardwColor[i]);
     rc.left += 40;
     rc.right -= 40;
     rc.top += 30;
     rc.bottom -= 30;
}
```

# ビデオへの出力の開始

ビデオ出力を開始します。

```
MGE_VStreamStart(g_hmgestmStreamVideoOut);
```

# ビデオへの出力を切り替える

Memory サーフェースを直接書き換えることでビデオの出力を変更することは可能ですが、この場合書き換えを行っている途中の絵が出力されてしまいます。

本サンプルプログラムでは別の Memory サーフェースを用意してビデオへの出力を切り替えています。

```
HMGESURFACE    g_hmgesrfSurfaceMemory2; // 切り替えの Memory Surface へのハンドル(グローバル変数)

    MGEPFINFO pfinfo;
    RECT      rc;

        //
        // 切り替えの Memory Surface を作成します。
        //
    pfinfo.dwSize = sizeof(pfinfo);
    MGE_GetPixelFormatInfo(MGEPixelFormatYUY2, &pfinfo);
    MGE_CreateMemorySurface(g_hmge, &g_hmgesrfSurfaceMemory2,
                        640, 480, 640 * pfinfo.dwBytesPerPixel,
                                MGEPixelFormatYUY2);

        //
        // このメモリにデータを格納します。
        //

        //
        // 事前に黒で塗っておきます。
        // (JPEG のデータが 640x480 で在れば必要ありません)
        //
    rc.left = 0;
    rc.top = 0;
    rc.right = 640;
    rc.bottom = 480;
    MGE_SurfaceFillBuffer(g_hmgesrfSurfaceMemory2, &rc, 0);

        //
        // データを JPEG ファイルからロードします。
        //
    rc.left = 0;
    rc.top = 0;
    rc.right = 640;
    rc.bottom = 400;
    MGE_SurfaceLoadJPEG(g_hmgesrfSurfaceMemory2, "pic.jpg", &rc);
```

```
    //
    // Video 出力を切り替えます。
    //
MGE_VStreamReplace(g_hmgestmStreamVideoOut, g_hmgesrfSurfaceMemory2);

    //
    // 何らかの処理をする。
    // 例えば所定の時間だけ待つ
    //      :
    //      :
    //      :
    //

    //
    // Video の出力を元に戻します。
    //
MGE_VStreamReplace(g_hmgestmStreamVideoOut, g_hmgesrfSurfaceMemory);

    //
    // 作成した Memory surface を削除します。
    //
MGE_DestroySurface(g_hmgesrfSurfaceMemory2);
```

# ビデオへの出力の終了

ビデオ出力を終了します。

```
MGE_VStreamStop(g_hmgestmStreamVideoOut);
```

# ビデオへの出力の後処理

ビデオ出力を行なったオブジェクト（静止画）を削除する。

```
MGE_VStreamDisconnect(g_hmgestmStreamVideoOut, g_hmgesrfSurfaceMemory);
MGE_DestroyVideoStream(g_hmgestmStreamVideoOut);
MGE_DestroySurface(g_hmgesrfSurfaceMemory);
MGE_DestroyVideoEncoder(g_hmgevdVideoOut);
```

# Power Movie PCI の使用終了

Power Movie PCI を使用するアプリケーションは、その使用終了をドライバに通知する必要があります。アプリケーションの終了時などに **MGE_Uninitialize** ファンクションを呼び出して Power Movie PCI の使用を終了します。

```
MGE_Uninitialize(g_hmge);
```

CHAPTER 2

# Quick Reference

# MGE API Quick Reference

## § 1.Generic Functions(初期化)

| | |
|---|---|
| **MGE_Initialize** | ライブラリの初期化 |
| **MGE_Uninitialize** | ライブラリの終了 |
| **MGE_SetCooperativeLevel** | ライブラリの使用形態を指定する |
| **MGE_SaveConfiguration** | ライブラリの設定を registry に格納する |
| **MGE_LoadConfiguration** | ライブラリの設定を registry からロードする |

## § 2.Generic Functions (全体)

| | |
|---|---|
| **MGE_Update** | パラメータを反映させる |

## § 3.Information Functions

| | |
|---|---|
| **MGE_GetDeviceInfo** | ハードウェアの情報を得る |
| **MGE_SetVideoStandard** | システムのビデオスタンダード（ビデオ方式）を設定する |
| **MGE_GetVideoStandard** | 現在のシステムのビデオスタンダードを取得する |
| **MGE_GetPixelFormatInfo** | ピクセルフォーマットの情報を取得する |
| **MGE_GetImageInfo** | イメージデータの情報を得る |

## § 4.Surface Functions

| | |
|---|---|
| **MGE_CreateSuperOvlSurface** | Super Overlay のサーフェスを作成する |
| **MGE_CreateDDrawOvlSurface** | DirectDraw Overlay のサーフェスを作成する |
| **MGE_CreateMemorySurface** | サーフェスをメモリ上に作成する |
| **MGE_DestroySurface** | サーフェスを破棄する |
| **MGE_SurfaceSetColorKey** | キーカラーを設定する |
| **MGE_SurfaceGetColorKey** | 現在のキーカラーを取得する |
| **MGE_SurfaceSetOverlayMode** | オーバーレイモードを設定する |
| **MGE_SurfaceGetOverlayMode** | 現在のオーバーレイモードを取得する |

| | |
|---|---|
| MGE_SurfaceStartOverlay | オーバーレイを開始する |
| MGE_SurfaceStopOverlay | オーバーレイを停止する |
| MGE_SurfaceSetBitmapBits | メモリからサーフェスへbitmapをコピーする |
| MGE_SurfaceGetBitmapBits | サーフェスからメモリへbitmapをコピーする |
| MGE_SurfaceFillBuffer | サーフェスを指定した色で塗る |
| MGE_SurfaceLoadDIB | サーフェスにDIBをロードする |
| MGE_SurfaceSaveDIB | サーフェスのデータをDIBとしてファイルにセーブする |
| MGE_SurfaceLoadJPEG | サーフェスにJPEGデータを展開してロードする |
| MGE_SurfaceSaveJPEG | サーフェスのデータをJPEGに圧縮してファイルにセーブする |

## § 5.Video Decoder Functions

| | |
|---|---|
| MGE_CreateVideoDecoder | ビデオデコーダオブジェクトを作成する |
| MGE_DestroyVideoDecoder | ビデオデコーダオブジェクトを破棄する |
| MGE_VDGetStatus | ビデオデコーダの状態を取得する |
| MGE_VDSetParam | ビデオデコーダのパラメータを調整する |
| MGE_VDGetParam | 現在のビデオデコーダのパラメータを取得する |
| MGE_VDGetParamRange | ビデオデコーダの調整可能パラメータの範囲を取得する |
| MGE_VDSetLine | ビデオデコーダへの入力を選択する |
| MGE_VDGetLine | 現在のビデオデコーダへの入力を取得する |
| MGE_VDSaveConfiguration | ビデオデコーダの設定をregistryに格納する |
| MGE_VDLoadConfiguration | ビデオデコーダの設定をregistryからロードする |

## § 6.Video Encoder Functions

| | |
|---|---|
| MGE_CreateVideoEncoder | ビデオエンコーダオブジェクトを作成する |
| MGE_DestroyVideoEncoder | ビデオエンコーダオブジェクトを破棄する |
| MGE_VESetLine | ビデオエンコーダの出力を選択する |
| MGE_VEGetLine | 現在のビデオエンコーダの出力ライン番号を取得する |
| MGE_VESaveConfiguration | ビデオデコーダの設定をregistryに格納する |
| MGE_VELoadConfiguration | ビデオデコーダの設定をregistryからロードする |

## § 7.Video Stream Functions

| | |
|---|---|
| **MGE_CreateVideoInStream** | 入力用のビデオストリームを作成する |
| **MGE_CreateVideoOutStream** | 出力用のビデオストリームを作成する |
| **MGE_DestroyVideoStream** | ビデオストリームを破棄する |
| **MGE_VStreamConnect** | ビデオストリームをサーフェスに結び付ける |
| **MGE_VStreamDisconnect** | ビデオストリームとサーフェスを切り離す |
| **MGE_VStreamReplace** | ビデオストリームのサーフェスを付け替える |
| **MGE_VStreamSetVideoMode** | ビデオストリーム上のビデオ信号モードを設定する |
| **MGE_VStreamGetVideoMode** | 現在のビデオストリーム上のビデオ信号モードを取得する |
| **MGE_VStreamSetRect** | 各矩形を指定する |
| **MGE_VStreamGetRect** | 現在の矩形情報を取得する |
| **MGE_VStreamGetRectRange** | 各矩形情報を取得する |
| **MGE_VStreamStart** | ストリームを開始する |
| **MGE_VStreamPause** | ストリームを一時停止させる |
| **MGE_VStreamRestart** | ストリームを再開させる |
| **MGE_VStreamStop** | ストリームを停止させる |
| **MGE_VStreamGetState** | 指定したストリームの状態を取得する |
| **MGE_VStreamSetEffect** | ストリームに特殊効果を付加する |
| **MGE_VStreamGetEffect** | ストリーム上の現在の特殊効果を取得する |

CHAPTER 3

# Generic Functions

# MGE_Initialize

ライブラリの初期化を行う。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_Initialize(DWORD dwReserved, HWND hWnd, LPHMGE lphmge);
```

〈Basic〉

```
Declare Function MGE_Initialize Lib "mge.dll" (ByVal dwReserved As Long, ByVal hWnd As Long,
lphmge As Long) As Long
```

## 引数

| | |
|---|---|
| dwReserved | 予約0をわたす |
| hWnd | Applicationの Window Handle<br>MGE_Uninitializeを呼ぶまで hWndが有効でなければならない |
| lphmge | ライブラリのハンドルを格納するアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_Uninitialize

ライブラリを終了する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_Uninitialize(HMGE hmge);
```

〈Basic〉

```
Declare Function MGE_Uninitialize Lib "mge.dll" (ByVal hmge As Long) As Long
```

## 引数

hmge　　MGE_Initialize で返された MGE ライブラリのハンドル

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_SetCooperativeLevel

ライブラリの使用形態を指定する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SetCooperativeLevel(HMGE hmge, DWORD dwLevel);
```

〈Basic〉

```
Declare Function MGE_SetCooperativeLevel Lib "mge.dll" (ByVal hmge As Long, ByVal dwLevel As
Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |
| dwLevel | 現在は MGE_SCL_EXCLUSIVE を指定すること |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊この関数でエラーが返ってきた場合には、MGE_Uninitialize 以外のライブラリは使用できない。
*エラーコード MGE_E_BUSY が返ってきた場合には他のクライアントがライブラリを使用しているので、MGE_Uninitialize を呼び出して、終了しなければならない。

# MGE_SaveConfiguration

ライブラリの設定を registry に格納する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SaveConfiguration(HMGE hmge, HKEY hKey, LPCSTR lpszBase);
```

〈Basic〉

```
Declare Function MGE_SaveConfiguration Lib "mge.dll" (ByVal hmge As Long, ByVal hKey As Long,
    ByVal lpszBase As String) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |
| hKey | registry の root key<br>HKEY_CURRENT_USER など |
| lpszBase | registry の key、実際の値はこの key の下に<br>以下の構造で作成される。 |

```
├─ VD¥
│  ├─ Channel1¥
│  │  ├─ Line1¥
│  │  │  └─ NTSC¥
│  │  │     ├─ Brightness=〈brightness value〉
│  │  │     ├─ Hue=〈hue value〉
│  │  │     ├─ Contrast=〈contrast value〉
│  │  │     ├─ Saturation=〈saturation value〉
│  │  │     └─ Sharpness=〈sharpness value〉
│  │  └─ CurrentLine = 〈current line number〉 1..4
│  └─ Channel2¥
│  .
│  .
└─ VE¥ ← 今は何も書かれない
```

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_LoadConfiguration

ライブラリの設定を registry からロードする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_LoadConfiguration(HMGE hmge, HKEY hKey, LPCSTR lpszBase);
```

〈Basic〉

```
Declare Function MGE_LoadConfiguration Lib "mge.dll" (ByVal hmge As Long, ByVal hKey As Long,
ByVal lpszBase As String) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |
| hKey | registry の root key<br>HKEY_CURRENT_USER など |
| lpszBase | registry の key<br>(MGE_SaveConfiguration 参照) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_Update

パラメータを反映させる。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_Update(HMGE hmge);
```

〈Basic〉

```
Declare Function MGE_Update Lib "mge.dll" (ByVal hmge As Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

CHAPTER 4

# Information Functions

# MGE_GetDeviceInfo

ハードウェアの情報を得る。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_GetDeviceInfo(HMGE hmge, LPMGEDEVICEINFO lpmgedi);
```

〈Basic〉

```
Declare Function MGE_GetDeviceInfo Lib "mge.dll" (ByVal hmge As Long, lpmgedi As Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |
| lpmgedi | Device Information を格納する領域 |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCI では下記の値が返ります。

| | |
|---|---|
| dwNumVideoDecoderChannel | 2 |
| dwNumVideoDecoderLine | 4 |
| dwNumVideoEncoderChannel | 1 |
| dwNumVideoEncoderLine | 1 |
| szBoardName | "Power Movie PCI" |

# MGE_SetVideoStandard

システムのビデオスタンダード（ビデオ方式）を設定する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SetVideoStandard(HMGE hmge, MGEVIDEOSTANDARD mgevs);
```

〈Basic〉

```
Declare Function MGE_SetVideoStandard Lib "mge.dll" (ByVal hmge As Long, ByVal mgevs As Long)
As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initializeで返されたMGEライブラリのハンドル |
| mgevs | ビデオスタンダード |
| | MGEVideoStandardNTSC ・・・ NTSC方式 |
| | MGEVideoStandardPAL ・・・ PAL方式 |
| | ※Power Movie PCIではNTSC方式のみをサポート |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCIはPALは非サポートなので、この関数を使用する必要はありません。

# MGE_GetVideoStandard

現在のシステムのビデオスタンダード（ビデオ方式）を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_GetVideoStandard(HMGE hmge, LPMGEVIDEOSTANDARD lpmgevs);
```

〈Basic〉

```
Declare Function MGE_GetVideoStandard Lib "mge.dll" (ByVal hmge As Long, lpmgevs As Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |
| lpmgevs | ビデオスタンダードを格納するアドレス |
| | MGEVideoStandardNTSC ・・・ NTSC 方式 |
| | MGEVideoStandardPAL ・・・ PAL 方式 |
| | ※Power Movie PCI では NTSC 方式のみをサポート |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCI では PAL は非サポートなので、常に MGEVideoStandardNTSC が返ります。

# MGE_GetPixelFormatInfo

ピクセルフォーマットの情報を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_GetPixelFormatInfo(MGEPIXELFORMAT mgepfFormat, LPMGEPFINFO lpmgpfi);
```

〈Basic〉

```
Declare Function MGE_GetPixelFormatInfo Lib "mge.dll" (ByVal mgepfFormat As Long, lpmgfi As
MGEPFINFO) As Long
```

## 引数

| | |
|---|---|
| mgepfFormat | 情報を取得するPixel Format |
| lpmgpfi | Pixel Format情報を返すアドレス<br>lpmgpfi->dwSizeは初期化されていなければならない |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊lpmgpf －>rgbMaskはdwFourCCがBI_BITFIELDSの時にのみ意味を持つ。

# MGE_GetImageInfo

イメージデータの情報を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_GetImageInfo(LPCSTR lpszImageFile, LPMGEIMAGEINFO lpmgii);
```

〈Basic〉

```
Declare Function MGE_GetImageInfo Lib "mge.dll"Alias "MGE_GetImageInfoA"(ByVal lpszImageFile
As String,lpmgii As MGEIMAGEINFO)As Long
```

## 引数

| | |
|---|---|
| lpszImageFile | 情報を取得するイメージのファイル名 |
| lpmgii | Image Info情報を返すアドレス<br>lpmgii->dwSizeは初期化されていなければならない |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊取得できるのはDIBファイルまたはJPEGファイルのみです。

CHAPTER 5

# Surface Functions

# MGE_CreateSuperOvlSurface

Super Overlayのサーフェスを作成する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_CreateSuperOvlSurface(HMGE hmge, LPHMGESURFACE lphmgesrf, HWND hWnd,
                          DWORD dwWidth, DWORD dwHeight);
```

〈Basic〉

```
Declare Function MGE_CreateSuperOvlSurface Lib "mge.dll" (ByVal hmge As Long, lphmgesrf As Long,
ByVal hWnd As Long, ByVal dwWidth As Long, ByVal dwHeight As Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initializeで返されたMGEライブラリのハンドル |
| lphmgesrf | サーフェスハンドルを格納するメモリアドレス |
| hWnd | Overlayする Windowハンドル<br>この Windowは Overlayを終了するまで有効でなければならない<br>この hWndの Client領域全体に overlayされる |
| dwWidth | サーフェスの幅(pixel単位) |
| dwHeight | サーフェスの高さ(pixel単位) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_CreateDDrawOvlSurface

DirectDraw Overlayのサーフェスを作成する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_CreateDDrawOvlSurface(HMGE hmge, LPHMGESURFACE lphmgesrf, HWND hWnd,
                          DWORD dwWidth, DWORD dwHeight);
```

〈Basic〉

```
Declare Function MGE_CreateDDrawOvlSurface Lib "mge.dll" (ByVal hmge As Long, lphmgesrf As Long,
ByVal hWnd As Long, ByVal dwWidth As Long, ByVal dwHeight As Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initializeで返されたMGEライブラリのハンドル |
| lphmgesrf | サーフェスハンドルを格納するメモリアドレス |
| hWnd | OverlayするWindowハンドル<br>このWindowはOverlayを終了するまで有効でなければならない |
| dwWidth | Overlayサーフェスの幅(pixel単位) |
| dwHeight | Overlayサーフェスの高さ(pixel単位) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_CreateMemorySurface

サーフェスをメモリ上に作成する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_CreateMemorySurface(HMGE hmge, LPHMGESURFACE lphmgesrf,
            DWORD dwWidth, DWORD dwHeight, LONG lPitch,
                    MGEPIXELFORMAT mgpfFormat);
```

〈Basic〉

```
Declare Function MGE_CreateMemorySurface Lib "mge.dll" (ByVal hmge As Long, lphmgesrf As Long,
ByVal dwWidth As Long, ByVal dwHeight As Long, ByVal lPitch As Long, ByVal mgefFormat As Long)
As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initializeで返されたMGEライブラリのハンドル |
| lphmgesrf | サーフェスハンドルを格納するメモリアドレス |
| dwWidth | Memoryサーフェスの幅(pixel単位) |
| dwHeight | Memoryサーフェスの高さ(pixel単位) |
| lPitch | Memoryサーフェスのピッチ(ストライド)(byte単位)<br>現在は正の値のみ |
| mgpfFormat | Memoryサーフェスのフォーマット<br>MGEPixelFormatRGB24 ・・・RGB24bit<br>MGEPixelFormatRGB16 ・・・RGB16bit<br>MGEPixelFormatYUY2 ・・・YUY2 |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊作成できるサーフェスは最大1024x768、最小80x60です。

# MGE_DestroySurface

サーフェスを破棄する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_DestroySurface(HMGESURFACE hmgesrfSurface);
```

〈Basic〉

```
Declare Function MGE_DestroySurface Lib "mge.dll" (ByVal hmgesrfSurface As Long) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | サーフェスハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_SurfaceSetColorKey

キーカラーを設定する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceSetColorKey(HMGESURFACE hmgesrfSurface, LPCMGECOLORKEY lpmgck);
```

〈Basic〉

```
Declare Function MGE_SurfaceSetColorKey Lib "mge.dll" (ByVal hmgesrfSurface As Long, lpmgck
As MGECOLORKEY) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | キーカラーを設定するサーフェスハンドル |
| lpmgck | 設定するキーカラー<br>24bit full color を指定する<br>00..07bit 青<br>08..15bit 緑<br>16..23bit 赤<br>24..31bit 0を指定する |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*DirectDraw Overlay サーフェスのみ有効となる。

＊画面(グラフィックボード)モードによっては適宜カラーキーの色変換を行うので、意図した色にならない場合がある。

(例)

24bit→15bit, 24bit→16bit 変換の場合

# MGE_SurfaceGetColorKey

現在のキーカラーを取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceGetColorKey(HMGESURFACE hmgesrfSurface, LPMGECOLORKEY lpmgck);
```

〈Basic〉

```
Declare Function MGE_SurfaceGetColorKey Lib "mge.dll" (ByVal hmgesrfSurface As Long, lpmgck
As MGECOLORKEY) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | キーカラーを取得するサーフェスハンドル |
| lpmgck | キーカラーを返すアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*DirectDraw Overlay サーフェスのみ有効となる。

# MGE_SurfaceSetOverlayMode

オーバーレイモードを設定する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceSetOverlayMode(HMGESURFACE hmgesrfSurface, DWORD dwMode);
```

〈Basic〉

```
Declare Function MGE_SurfaceSetOverlayMode Lib "mge.dll" (ByVal hmgesrfSurface As Long, ByVal
    dwMode As Long) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | オーバーレイモードを設定するサーフェスハンドル |
| dwMode | 設定するモード<br>MGE_SOM_COLORKEY |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*DirectDraw Overlay サーフェスのみ有効となる。

*Power Movie PCI は MGE_SOM_COLORKEY のみサポートしているのでこの関数は使用しない。

# MGE_SurfaceGetOverlayMode

現在のオーバーレイモードを取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceGetOverlayMode(HMGESURFACE hmgesrfSurface, LPDWORD lpdwMode);
```

〈Basic〉

```
Declare Function MGE_SurfaceGetOverlayMode Lib "mge.dll" (ByVal hmgesrfSurface As Long, lpdwMode
As Long) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | オーバーレイモードを取得するサーフェスハンドル |
| lpdwMode | モードを返すアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*DirectDraw Overlay サーフェスのみ有効となる。
*Power Movie PCI は MGE_SOM_COLORKEY のみサポートしているのでこの関数は使用しない。

# MGE_SurfaceStartOverlay

オーバーレイを開始する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceStartOverlay(HMGESURFACE hmgesrfSurface);
```

〈Basic〉

```
Declare Function MGE_SurfaceStartOverlay Lib "mge.dll" (ByVal hmgesrfSurface As Long) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | オーバーレイを開始するサーフェスハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Memory サーフェスにはオーバーレイできない。

# MGE_SurfaceStopOverlay

オーバーレイを停止する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceStopOverlay(HMGESURFACE hmgesrfSurface);
```

〈Basic〉

```
Declare Function MGE_SurfaceStopOverlay Lib "mge.dll" (ByVal hmgesrfSurface As Long) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | オーバーレイを停止するサーフェスハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Memory サーフェスにはオーバーレイできない。

# MGE_SurfaceSetBitmapBits

メモリからサーフェスへBitmapをコピーする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceSetBitmapBits(HMGESURFACE hmgesrfSurface, LPVOID lpData,
                         MGEPIXELFORMAT mgpfFormat,
                         LPMGEPALINFO lppi,
                         LONG lPitch, LPCRECT lprc);
```

〈Basic〉

```
Declare Function MGE_SurfaceSetBitmapBits Lib "mge.dll" (ByVal hmgesrfSurface As Long, lpData
As Any, ByVal mgpfFormat As Long, lppi As Long, ByVal lPitch As Long, lprc As RECT) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | Bitmapをコピーするサーフェスハンドル |
| lpData | Bitmapデータ<br>データの並びは一番最後のラインがデータの先頭にくる<br>(例)<br>縦が240ラインのデータは240line目のデータがlpDataの先頭にくる |
| mgpfFormat | Bitmapのフォーマット<br>MGEPixelFormatRGB24 ・・・RGB24bit<br>MGEPixelFormatRGB16 ・・・RGB16bit<br>MGEPixelFormatPAL8 ・・・8bitパレット |
| lppi | mgpFormatがMGEPixelFormatPAL8の時のパレット情報を指定する |
| lPitch | lpDataのピッチ<br>4の倍数でなければならない |
| lprc | Surface上の位置とサイズ |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Super Overlay サーフェスにはデータをコピーできない。

＊サーフェスがYUY2の場合にはRectは以下の関係を保つ必要がある。

1. Rectのスタート位置は偶数
2. Rectのサイズは4の倍数

# MGE_SurfaceGetBitmapBits

サーフェスからメモリへBitmapをコピーする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceGetBitmapBits(HMGESURFACE hmgesrfSurface, LPVOID lpData,
                         MGEPIXELFORMAT mgpfFormat,
                         LPMGEPALINFO lppi,
                         LONG lPitch, LPCRECT lprc);
```

〈Basic〉

```
Declare Function MGE_SurfaceGetBitmapBits Lib "mge.dll" (ByVal hmgesrfSurface As Long, lpData
As Any, ByVal mgpfFormat As Long, lppi As Long, ByVal lPitch As Long, lprc As RECT) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | Bitmapをコピーするサーフェスハンドル |
| lpData | Bitmapデータをコピーするメモリのアドレス<br>データの並びは一番最後のラインがデータの先頭にくる<br>(例)<br>縦が240ラインのデータは240line目のデータがlpDataの先頭にくる |
| mgpfFormat | Bitmapのフォーマット<br>MGEPixelFormatRGB24 ･･･RGB24bit<br>MGEPixelFormatRGB16 ･･･RGB16bit |
| lppi | 将来の拡張用、現在は無視される。 |
| lPitch | lpDataのピッチ<br>4の倍数でなければならない |
| lprc | サーフェス上の位置とサイズ |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊サーフェスがYUY2の場合にはRectは以下の関係を保つ必要がある。

1. Rectのスタート位置は偶数
2. Rectのサイズは4の倍数

*Super Overlay サーフェスからはStop状態での動作は保証されない。

*Super Overlay サーフェスは指定されたCropping windowが最大の大きさとなる。

# MGE_SurfaceFillBuffer

サーフェスを指定した色で塗る。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceFillBuffer(HMGESURFACE hmgesrfSurface, LPRECT lprc, DWORD dwFillColor);
```

〈Basic〉

```
Declare Function MGE_SurfaceFillBuffer Lib "mge.dll" (ByVal hmgesrfSurface As Long, lprc As RECT,
ByVal dwFillColor As Long) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | Bitmap をコピーするサーフェスハンドル |
| lprc | サーフェス上の位置とサイズ |
| dwFillColor | 塗る色、24bit color を指定する |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Super Overlay サーフェスには使用できない。

＊サーフェスが YUY2 の場合には Rect は以下の関係を保つ必要がある。

1. Rect のスタート位置は偶数
2. Rect のサイズは 4 の倍数

# MGE_SurfaceLoadDIB

サーフェスにDIBをロードする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceLoadDIB(HMGESURFACE hmgesrfSurface,
                   LPCSTR lpszFileName, LPCRECT lprc);
```

〈Basic〉

```
Declare Function MGE_SurfaceLoadDIB Lib "mge.dll" Alias "MGE_SurfaceLoadDIBA" (ByVal hmgesrfSurface
As Long, ByVal lpszFileName As String, lprc As RECT) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | DIBをロードするサーフェスハンドル |
| lpszFileName | DIBのファイル名 |
| lprc | DIBをロードするサーフェスの位置、大きさ |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Super Overlay サーフェスにはロードできない。

＊サーフェスがYUY2の場合にはRectは以下の関係を保つ必要がある。

1. Rectのスタート位置は偶数
2. Rectのサイズは4の倍数

*1024x768より大きいDIBはロードできない。

# MGE_SurfaceSaveDIB

サーフェスのデータをDIBとしてファイルにセーブする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceSaveDIB(HMGESURFACE hmgesrfSurface,
                   LPCSTR lpszFileName, LPCRECT lprc,
                   MGEPIXELFORMAT mgpfFormat);
```

〈Basic〉

```
Declare Function MGE_SurfaceSaveDIB Lib "mge.dll" Alias "MGE_SurfaceSaveDIBA" (ByVal hmgesrfSurface
As Long, ByVal lpszFileName As String, lprc As RECT, ByVal mgpfFormat As Long) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | DIBをセーブするサーフェスハンドル |
| lpszFileName | DIBのファイル名 |
| lprc | DIBをセーブするサーフェスの位置、大きさ |
| mgpfFormat | DIBのフォーマット |
| | MGEPixelFormatRGB24 ・・・RGB24bit |
| | MGEPixelFormatRGB16 ・・・RGB16bit |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊サーフェスがYUY2の場合にはRectは以下の関係を保つ必要がある。

1. Rectのスタート位置は偶数
2. Rectのサイズは4の倍数

*Super Overlay サーフェスからはStop状態での動作は保証されない。

*Super Overlay サーフェスは指定されたCropping windowが最大の大きさとなる。

# MGE_SurfaceLoadJPEG

サーフェスにJPEGデータを展開してロードする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceLoadJPEG(HMGESURFACE hmgesrfSurface,
                    LPCSTR lpszFileName, LPCRECT lprc);
```

〈Basic〉

```
Declare Function MGE_SurfaceLoadJPEG Lib "mge.dll" Alias "MGE_SurfaceLoadJPEGA" (ByVal
hmgesrfSurface As Long, ByVal lpszFileName As String, lprc As RECT) As Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | JPEGをロードするサーフェスハンドル |
| lpszFileName | JPEGのファイル名 |
| lprc | JPEGをロードするサーフェスの位置、大きさ |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Super Overlay サーフェスにはロードできない。

＊サーフェスがYUY2の場合にはRectは以下の関係を保つ必要がある。

1. Rectのスタート位置は偶数
2. Rectのサイズは4の倍数

# MGE_SurfaceSaveJPEG

サーフェスのデータを JPEG に圧縮してファイルにセーブする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_SurfaceSaveJPEG(HMGESURFACE hmgesrfSurface,
                    LPCSTR lpszFileName, LPCRECT lprc,
                    MGEJPEGQUALITY mgjqQuality);
```

〈Basic〉

```
Declare Function MGE_SurfaceSaveJPEG Lib "mge.dll" Alias "MGE_SurfaceSaveJPEGA" (ByVal
hmgesrfSurface As Long, ByVal lpszFileName As String, lprc As RECT, ByVal mgjqQuality As Long) As
Long
```

## 引数

| | |
|---|---|
| hmgesrfSurface | JPEG をセーブするサーフェスハンドル |
| lpszFileName | JPEG のファイル名 |
| lprc | JPEG をセーブするサーフェスの位置、大きさ |
| mgjqQuality | JPEG の圧縮率<br>1/n の n × 1000 を指定する<br>現在サポートしている値は以下の通り<br>5000(1/5)<br>7000(1/7)<br>10000(1/10)<br>12000(1/12)<br>15000(1/15)<br>17000(1/17)<br>20000(1/20)<br>25000(1/25)<br>30000(1/30)<br>40000(1/40) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Super Overlay サーフェスデータのセーブをサポートしています。

＊サーフェスが YUY2 の場合には Rect は以下の関係を保つ必要がある。

1. Rect のスタート位置は偶数
2. Rect のサイズは 4 の倍数

*Super Overlay サーフェスからは Stop 状態での動作は保証されない。

*Super Overlay サーフェスは指定された Cropping window が最大の大きさとなる。

CHAPTER 6

# Video Decoder Functions

# MGE_CreateVideoDecoder

ビデオデコーダオブジェクトを作成する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_CreateVideoDecoder(HMGE hmge, MGECHANNEL mgcChannel, LPHMGEVSRC lphmgevs);
```

〈Basic〉

```
Declare Function MGE_CreateVideoDecoder Lib "mge.dll" (ByVal hmge As Long, ByVal mgcChannel As
Long, lphmgevs As Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |
| mgcChannel | ビデオデコーダ番号 |
| lphmgevs | ビデオデコーダへのハンドルを格納するメモリアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCI の実装では MGEChannel1 と MGEChannel2 のみをサポートしています。
＊ビデオデコーダ番号と使用するサーフェスは密接に関係しています。MGE_VStreamConnect を参照ください。

# MGE_DestroyVideoDecoder

ビデオデコーダオブジェクトを破棄する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_DestroyVideoDecoder(HMGEVSRC hmgevs);
```

〈Basic〉

```
Declare Function MGE_DestroyVideoDecoder Lib "mge.dll" (ByVal hmgevs As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VDGetStatus

ビデオデコーダの状態を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDGetStatus(HMGEVSRC hmgevs, MGELINE mglLine, LPMGEVDSTATUS lpmgevds);
```

〈Basic〉

```
Declare Function MGE_VDGetStatus Lib "mge.dll" (ByVal hmgevs As Long, ByVal mglLine As Long, ByVal
lpmgevds As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| mglLine | 入力ライン番号 |
| lpmgevds | Video Decoder のステータスを格納するアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊現在選択されているLineと違うLineを指定した場合には、一度指定したLineに入力が変更された後に元のLineに戻る。

# MGE_VDSetParam

ビデオデコーダのパラメータを調整する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDSetParam(HMGEVSRC hmgevs, MGELINE mglLine,
                    MGEVIDEOAMPPROPERTY mgvap, DWORD dwValue);
```

〈Basic〉

```
Declare Function MGE_VDSetParam Lib "mge.dll" (ByVal hmgves As Long, ByVal mglLine As Long, ByVal
mgvap As Long, ByVal dwValue As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| mglLine | 入力ライン番号 |
| mgvap | 調整可能項目 |
| | MGEVideoAmpBrightness ･･･明るさ |
| | MGEVideoAmpHue ･･･色合い |
| | MGEVideoAmpContrast ･･･コントラスト |
| | MGEVideoAmpSaturation ･･･色の濃さ |
| | MGEVideoAmpSharpness ･･･輪郭強調 |
| | MGEVideoAmpColorEnable ･･･Power Movie PCI では使用しません |
| | MGEVideoAmpGamma ･･･Power Movie PCI では使用しません |
| | MGEVideoAmpWhiteBalance ･･･Power Movie PCI では使用しません |
| dwValue | 設定する値 |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VDGetParam

現在のビデオデコーダのパラメータを取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDGetParam(HMGEVSRC hmgevs, MGELINE mglLine,
                    MGEVIDEOAMPPROPERTY mgvap, LPDWORD lpdwValue);
```

〈Basic〉

```
Declare Function MGE_VDGetParam Lib "mge.dll" (ByVal hmgves As Long, ByVal mglLine As Long, ByVal
mgvap As Long, lpdwValue As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| mglLine | 入力ライン番号 |
| mgvap | 調整項目 |
| | MGEVideoAmpBrightness ・・・明るさ |
| | MGEVideoAmpHue ・・・色合い |
| | MGEVideoAmpContrast ・・・コントラスト |
| | MGEVideoAmpSaturation ・・・色の濃さ |
| | MGEVideoAmpSharpness ・・・輪郭強調 |
| | MGEVideoAmpColorEnable ・・・Power Movie PCI では使用しません |
| | MGEVideoAmpGamma ・・・Power Movie PCI では使用しません |
| | MGEVideoAmpWhiteBalance ・・・Power Movie PCI では使用しません |
| lpdwValue | 値を格納するアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VDGetParamRange

ビデオデコーダの調整可能パラメータの範囲を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDGetParamRange(HMGEVSRC hmgevs, MGELINE mglLine,
                    MGEVIDEOAMPPROPERTY mgvap,
                    LPDWORD lpdwMin, LPDWORD lpdwMax, LPDWORD lpdwDef);
```

〈Basic〉

```
Declare Function MGE_VDGetParamRange Lib "mge.dll" (ByVal hmgves As Long, ByVal mglLine As Long,
ByVal mgvap As Long, lpdwMin As Long, lpdwMax As Long, lpdwDef As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| mglLine | 入力ライン番号 |
| mgvap | 調整項目 |
| | MGEVideoAmpBrightness ･･･明るさ |
| | MGEVideoAmpHue ･･･色合い |
| | MGEVideoAmpContrast ･･･コントラスト |
| | MGEVideoAmpSaturation ･･･色の濃さ |
| | MGEVideoAmpSharpness ･･･輪郭強調 |
| | MGEVideoAmpColorEnable ･･･Power Movie PCI では使用しません |
| | MGEVideoAmpGamma ･･･Power Movie PCI では使用しません |
| | MGEVideoAmpWhiteBalance ･･･Power Movie PCI では使用しません |
| lpdwMin | 最小値を格納するアドレス |
| lpdwMax | 最大値を格納するアドレス |
| lpdwDef | default 値を格納するアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊VideoDecoder 用の調整パラメータの最小値、最大値、default 値を得ます。

＊この関数は VideoDecoder 用の調整 Dialog 等を作成するためにあります。Application はこの関数を使用し、各調整可能項目(もしくは調整させたい項目)の最小値、最大値、default 値を取得して記憶します。最小値、最大値は scroll bar 等に渡し、範囲を限定することに使用できます。また default 値は＂初期値＂などのボタンを実装した場合に使用できます。Application のユーザが値を指定したときに MGE_VDSetParam を通じて Power Movie PCI に通知できます。

# MGE_VDSetLine

ビデオデコーダへの入力を選択する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDSetLine(HMGEVSRC hmgevs, MGELINE mglLine);
```

〈Basic〉

```
Declare Function MGE_VDSetLine Lib "mge.dll" (ByVal hmgves As Long, ByVal mglLine As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| mglLine | 入力ライン番号 |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCIの実装はMGELine1, MGELine2, MGELine3, MGELine4に対応しています。
入力ライン番号はコネクタの番号に対応しています。MGELine1がIN1..MGELine4がIN4です。

# MGE_VDGetLine

現在のビデオデコーダへの入力を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDGetLine(HMGEVSRC hmgevs, LPMGELINE lpmglLine);
```

〈Basic〉

```
Declare Function MGE_VDGetLine Lib "mge.dll" (ByVal hmgves As Long, lpmglLine As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| lpmglLine | 入力ライン番号を格納するアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VDSaveConfiguration

ビデオデコーダの設定を registry に格納する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDSaveConfiguration(HMGEVSRC hmgevs, HKEY hKey, LPCSTR lpszBase);
```

〈Basic〉

```
Declare Function MGE_VDSaveConfiguration Lib "mge.dll" (ByVal hmgves As Long, ByVal hKey As Long,
ByVal lpszBase As String) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| hKey | registry の root key<br>HKEY_CURRENT_USER など |
| lpszBase | registry の key、実際の値は対応する channel 番号の下に作成される<br>(MGE_SaveConfiguration 参照) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊ビデオデコーダの設定を個別に保存したい場合に使用します。通常は MGE_SaveConfiguration を使用することになります。

# MGE_VDLoadConfiguration

ビデオデコーダの設定を registry からロードする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VDLoadConfiguration(HMGEVSRC hmgevs, HKEY hKey, LPCSTR lpszBase);
```

〈Basic〉

```
Declare Function MGE_VDLoadConfiguration Lib "mge.dll" (ByVal hmgves As Long, ByVal hKey As Long,
ByVal lpszBase As String) As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオデコーダへのハンドル |
| hKey | registry の root key<br>HKEY_CURRENT_USER など |
| lpszBase | registry の key<br>(MGE_VDSaveConfiguration 参照) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊ビデオデコーダの設定を個別にロードしたい場合に使用します。通常は MGE_LoadConfiguration を使用することになります。

CHAPTER 7

# Video Encoder Functions

# MGE_CreateVideoEncoder

ビデオエンコーダオブジェクトを作成する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_CreateVideoEncoder(HMGE hmge, MGECHANNEL mgcChannel, LPHMGEVDST lphmgevd);
```

〈Basic〉

```
Declare Function MGE_CreateVideoEncoder Lib "mge.dll" (ByVal hmge As Long, ByVal mgcChannel As
Long, lphmgevd As Long) As Long
```

## 引数

| | |
|---|---|
| hmge | MGE_Initialize で返された MGE ライブラリのハンドル |
| mgcChannel | ビデオエンコーダ番号 |
| lphmgevd | ビデオエンコーダへのハンドルを格納するメモリアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCI の実装は、MGEChannel1 のみをサポートしています。

# MGE_DestroyVideoEncoder

ビデオエンコーダオブジェクトを破棄する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_DestroyVideoEncoder(HMGEVDST hmgevd);
```

〈Basic〉

```
Declare Function MGE_DestroyVideoEncoder Lib "mge.dll" (ByVal hmgevd As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevd | ビデオエンコーダへのハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VESetLine

ビデオエンコーダの出力を選択する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VESetLine(HMGEVDST hmgevd, MGELINE mglLine);
```

〈Basic〉

```
Declare Function MGE_VESetLine Lib "mge.dll" (ByVal hmgevd As Long, ByVal mglLine As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevd | ビデオエンコーダへのハンドル |
| mglLine | 出力ライン番号 |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCI の実装は、MGEChannel1 のみをサポートしています。

# MGE_VEGetLine

現在のビデオエンコーダの出力ライン番号を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VEGetLine(HMGEVDST hmgevd, LPMGELINE lpmglLine);
```

〈Basic〉

```
Declare Function MGE_VEGetLine Lib "mge.dll" (ByVal hmgevd As Long, lpmglLine As Long) As Long
```

## 引数

| | |
|---|---|
| hmgevd | ビデオエンコーダへのハンドル |
| lpmglLine | 出力ライン番号を格納するアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

※Power Movie PCIの実装はMGELine1のみをサポートしているため、この関数を使用する必要はありません。

# MGE_VESaveConfiguration

ビデオエンコーダの設定を registry に格納する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VESaveConfiguration(HMGEVDST hmgevd, HKEY hKey, LPCSTR lpszBase);
```

〈Basic〉

```
Declare Function MGE_VESaveConfiguration Lib "mge.dll" (ByVal hmgevd As Long, ByVal hKey As Long,
ByVal lpszBase As String) As Long
```

## 引数

| | |
|---|---|
| hmgevd | ビデオエンコーダへのハンドル |
| hKey | registry の root key<br>HKEY_CURRENT_USER など |
| lpszBase | registry の key、実際の値は対応する channel 番号の下に作成される<br>(MGE_SaveConfiguration 参照) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊今は何も情報は書かれていない。

# MGE_VELoadConfiguration

ビデオエンコーダの設定を registry からロードする。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VELoadConfiguration(HMGEVDST hmgevd, HKEY hKey, LPCSTR lpszBase);
```

〈Basic〉

```
Declare Function MGE_VELoadConfiguration Lib "mge.dll" (ByVal hmgevd As Long, ByVal hKey As Long,
ByVal lpszBase As String) As Long
```

## 引数

| | |
|---|---|
| hmgevd | ビデオエンコーダへのハンドル |
| hKey | registry の root key<br>HKEY_CURRENT_USER など |
| lpszBase | registry の key<br>(MGE_VESaveConfiguration 参照) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊今は何も情報はロードされない。

CHAPTER 8

# Video Stream Functions

# MGE_CreateVideoInStream

入力用のビデオストリームを作成する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_CreateVideoInStream(HMGEVSRC hmgevs, LPHMGESTREAM lphmgestm);
```

〈Basic〉

```
Declare Function MGE_CreateVideoInStream Lib "mge.dll" (ByVal hmgevs As Long, lphmgestm As Long)
As Long
```

## 引数

| | |
|---|---|
| hmgevs | ビデオソースへのハンドル |
| lphmgestm | ビデオストリームハンドルを格納するメモリアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_CreateVideoOutStream

出力用のビデオストリームを作成する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_CreateVideoOutStream(HMGEVDST hmgevd, LPHMGESTREAM lphmgestm);
```

〈Basic〉

```
Declare Function MGE_CreateVideoOutStream Lib "mge.dll" (ByVal hmgevd As Long, lphmgestm As
Long) As Long
```

## 引数

| | |
|---|---|
| hmgevd | ビデオ出力へのハンドル |
| lphmgestm | ビデオストリームハンドルを格納するメモリアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊このストリームにConnectできるのはMGEPixelFormatYUY2のMemory サーフェスのみです。

# MGE_DestroyVideoStream

ビデオストリームを破棄する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_DestroyVideoStream(HMGESTREAM hmgestm);
```

〈Basic〉

```
Declare Function MGE_DestroyVideoStream Lib "mge.dll" (ByVal hmgestm As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestm | 破棄するビデオストリームハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VStreamConnect

ビデオストリームをサーフェスに結び付ける。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamConnect(HMGESTREAM hmgestmStream, HMGESURFACE hmgesrfSurface);
```

〈Basic〉

```
Declare Function MGE_VStreamConnect Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal hmgesrfSurface
As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 結び付けるストリームハンドル |
| hmgesrfSurface | 結び付けるサーフェスハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Power Movie PCI の実装では下記の制限があります。

1. MGE_CreateVideoInStream で作成したストリームとサーフェスを結び付けた場合。
   1-1. MGEChannel1 で作成された(MGE_CreateVideoDecoder 参照)ビデオデコーダを使用して作成されたストリームは下記のサーフェスにしか結び付けられません。
      1-1-1 Super Overlay サーフェス
      1-1-2 下記のフォーマットの Memory サーフェス
         1-1-2-1 MGEPixelFormatRGB32
         1-1-2-2 MGEPixelFormatRGB24
         1-1-2-3 MGEPixelFormatRGB15
   1-2. MGEChannel2 で作成された(MGE_CreateVideoDecoder 参照)ビデオデコーダを使用して作成されたストリームは下記のサーフェスにしか結び付けられません。
      1-2-1 DirectDraw Overlay サーフェス
2. MGE_CreateVideoOutStream で作成したストリームとサーフェスを結び付けた場合。
   MGEPixelFormatYUY2 で作成された Memory surface にしか結び付けられません。

# MGE_VStreamDisconnect

ビデオストリームとサーフェスを切り離す。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamDisconnect(HMGESTREAM hmgestmStream, HMGESURFACE hmgesrfSurface);
```

〈Basic〉

```
Declare Function MGE_VStreamDisconnect Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal
hmgesrfSurface As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 切り離すストリームハンドル |
| hmgesrfSurface | 切り離すサーフェスハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VStreamReplace

ビデオストリームのサーフェスを付け替える。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamReplace(HMGESTREAM hmgestmStream, HMGESURFACE hmgesrfSurface);
```

〈Basic〉

```
Declare Function MGE_VStreamReplace Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal hmgesrfSurface
As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 付け替えるストリームハンドル |
| hmgesrfSurface | 付け替えるサーフェスハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊付け替えるサーフェスは付け替える前のサーフェスと同様の属性を持っている必要がある。

＊現在 Power Movie PCI の実装では MGE_CreateVideoOutStream で作成されたストリームに結び付けられたサーフェスのみを付け替えることができる。

# MGE_VStreamSetVideoMode

ビデオストリーム上のビデオ信号モードを設定する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamSetVideoMode(HMGESTREAM hmgestmStream,
                        MGEVIDEOMODE mgevm, BOOL flmmediate);
```

〈Basic〉

```
Declare Function MGE_VStreamSetVideoMode Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal
mgevm As Long, ByVal flmmediate As Boolean) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 設定するストリームハンドル |
| mgevm | ビデオモード |
| | MGEVideoModeFrame ・・・両フィールドモード |
| | MGEVideoModeOddField ・・・奇数フィールドモード |
| | MGEVideoModeEvenField ・・・偶数フィールドモード |
| flmmediate | すぐに反映させるかどうかを指定 |
| | (MGE_Update 参照) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*DirectDraw サーフェスへのストリーム以外はエラーとなる。

# MGE_VStreamGetVideoMode

現在のビデオストリーム上のビデオ信号モードを取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamGetVideoMode(HMGESTREAM hmgestmStream, LPMGEVIDEOMODE lpmgevm);
```

〈Basic〉

```
Declare Function MGE_VStreamGetVideoMode Lib "mge.dll" (ByVal hmgestmStream As Long, lpmgevm As
Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 取得するストリーム |
| lpmgevm | ビデオモードを格納するアドレス |
| | MGEVideoModeFrame ･･･両フィールドモード |
| | MGEVideoModeOddField ･･･奇数フィールドモード |
| | MGEVideoModeEvenField ･･･偶数フィールドモード |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*DirectDraw サーフェスへのストリーム以外はエラーとなる。

# MGE_VStreamSetRect

次の各矩形を指定する。

1. ビデオ信号の取り込み
2. フレームバッファへの書き込み
3. フレームバッファからの読み出し
4. 画面の位置(screen coordinate)

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamSetRect(HMGESTREAM hmgestmStream, MGESTREAMRECT mgestmrc,
                   LPCRECT lprc, BOOL flmmediate);
```

〈Basic〉

```
Declare Function MGE_VStreamSetRect Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal mgestmrc As
Long, lprc As RECT, ByVal flmmediate As Boolean) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | ストリームへのハンドル |
| mgestmrc | 各矩形の識別子 |
| | MGEStreamRectCrop ・・・ビデオ信号の取り込み |
| | MGEStreamRectFrame ・・・フレームバッファへの書き込み |
| | MGEStreamRectView ・・・フレームバッファからの読み出し |
| | MGEStreamRectWindow ・・・画面の位置(screen coordinate) |
| lprc | 矩形のアドレス |
| flmmediate | すぐに反映させるかどうかを指定 |
| | (MGE_Update 参照) |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

補足

＊ flmmediate が FALSE の時には rect の整合性はチェックされない。
＊ ストリームが停止 / 一時停止しているときには画像内容は更新されない。
＊ サーフェスが SuperOverlay の場合は以下の関係を保つ必要がある。
1. 各 Rect の関係は RectCrop 〉= RectWindow
2. Frame と View は無視される。
3. マップされた hWnd の Window が動くと内部的に RectWindow が調整される。ただし、RectCrop 〉= RectWindow を保つ必要があるので、hWnd の Owner はその関係を保つ必要がある。
＊ サーフェスが DDrawOvlSurface の場合には各 Rect は以下の関係を保つ必要がある。
1. 各 rect のスタート位置は偶数
2. 各 rect のサイズは偶数
3. RectCrop のサイズ 〉= RectFrame のサイズ
RectFrame の幅は RectCrop の 1/1, 1/2, 1/4 でなければならない。
RectFrame の高さは RectCrop の 1/1, 1/2, 1/4, 1/8 でなければならない。
4. RectWindow のサイズ 〉= RectView のサイズ
＊ サーフェスが YUY2 の場合には各 Rect は以下の関係を保つ必要がある。
1. 各 rect のスタート位置は偶数
2. Rect のサイズは 4 の倍数
＊ サーフェスが Memory Surface の場合は以下の関係を保つ必要がある。
1. ストリームが Video in では Frame と Crop のみ有効となる。
2. ストリームが Video out では View と Window のみ有効となる。

# MGE_VStreamGetRect

現在の矩形情報を取得する。

1. ビデオ信号の取り込み
2. フレームバッファへの書き込み
3. フレームバッファからの読み出し
4. 画面の位置(screen coordinate)

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamGetRect(HMGESTREAM hmgestmStream, MGESTREAMRECT mgestmrc, LPRECT lprc);
```

〈Basic〉

```
Declare Function MGE_VStreamGetRect Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal mgestmrc As
Long, lprc As RECT) As Long
```

## 引数

| | | |
|---|---|---|
| hmgestmStream | ストリームへのハンドル | |
| mgestmrc | 各矩形の識別子 | |
| | MGEStreamRectCrop | ･･･ビデオ信号の取り込み |
| | MGEStreamRectFrame | ･･･フレームバッファへの書き込み |
| | MGEStreamRectView | ･･･フレームバッファからの読み出し |
| | MGEStreamRectWindow | ･･･画面の位置(screen coordinate) |
| lprc | 情報を返す矩形のアドレス | |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VStreamGetRectRange

各矩形情報を取得する。

1. ビデオ信号の取り込み
2. フレームバッファへの書き込み
3. フレームバッファからの読み出し
4. 画面の位置(screen coordinate)

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamGetRectRange(HMGESTREAM hmgestmStream,
                        MGESTREAMRECT mgestmrc,
                        LPRECT lprcMin,
                        LPRECT lprcMax,
                        LPRECT lprcDef);
```

〈Basic〉

```
Declare Function MGE_VStreamGetRectRange Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal
mgestmrc As Long, lprcMin As RECT, lprcMax As RECT, lprcDef As RECT) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | ストリームへのハンドル |
| mgestmrc | 各矩形の識別子 |
| | MGEStreamRectCrop ・・・ビデオ信号の取り込み |
| | MGEStreamRectFrame ・・・フレームバッファへの書き込み |
| | MGEStreamRectView ・・・フレームバッファからの読み出し |
| | MGEStreamRectWindow ・・・画面の位置(screen coordinate) |
| lprcMin | 最小の矩形サイズを返す矩形のアドレス |
| lprcMax | 最大の矩形サイズを返す矩形のアドレス |
| lprcDef | デフォルトの矩形サイズを返す矩形のアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VStreamStart

ストリームを開始する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamStart(HMGESTREAM hmgestmStream);
```

〈Basic〉

```
Declare Function MGE_VStreamStart Lib "mge.dll" (ByVal hmgestmStream As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 開始するストリームハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

# MGE_VStreamPause

ストリームを一時停止させる。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamPause(HMGESTREAM hmgestmStream);
```

〈Basic〉

```
Declare Function MGE_VStreamPause Lib "mge.dll" (ByVal hmgestmStream As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 一時停止させるストリームハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Super Overlay は一時停止できない。

# MGE_VStreamRestart

ストリームを再開させる。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamRestart(HMGESTREAM hmgestmStream);
```

〈Basic〉

```
Declare Function MGE_VStreamRestart Lib "mge.dll" (ByVal hmgestmStream As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 再開させるストリームハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

＊Super Overlay は再開できない。

# MGE_VStreamStop

ストリームを停止させる。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamStop(HMGESTREAM hmgestmStream);
```

〈Basic〉

```
Declare Function MGE_VStreamStop Lib "mge.dll" (ByVal hmgestmStream As Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 停止させるストリームハンドル |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

## 補足

*Super Overlay に対して停止をかけた場合の画面更新は、アプリケーションが行う。

# MGE_VStreamGetState

指定したストリームの状態を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamGetState(HMGESTREAM hmgestmStream, LPMGESTREAMSTATE lpmgess);
```

〈Basic〉

```
Declare Function MGE_VStreamGetState Lib "mge.dll" (ByVal hmgestmStream As Long, lpmgess As Long)
As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 状態を取得するストリーム |
| lpmgess | 返ってくるストリームの状態を格納するアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK を返す。それ以外の場合はエラーコードを返す。

lpmgess に返ってくるストリームの状態

| | |
|---|---|
| MGEStreamStateNotComplete | ・・・ 未完成のストリーム |
| MGEStreamStateStarted | ・・・ ストリームは動いている |
| MGEStreamStatePause | ・・・ ストリームは一時停止している |
| MGEStreamStateStop | ・・・ ストリームは停止している |

# MGE_VStreamSetEffect

ストリームに特殊効果を付加する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamSetEffect(HMGESTREAM hmgestmStream, DWORD dwEffect, BOOL flmmediate);
```

〈Basic〉

```
Declare Function MGE_VStreamSetEffect Lib "mge.dll" (ByVal hmgestmStream As Long, ByVal dwEffect
As Long, ByVal flmmediate As Boolean) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 特殊効果を付加するストリームハンドル |
| dwEffect | 各 effect を指定する。<br>MGEEF_MIRROR ･･･ 左右反転<br>MGEEF_UPSIDEDOWN ･･･ 上下反転 |
| flmmediate | すぐに反映させるかどうかを指定<br>(MGE_Update 参照) |

## 戻り値

正常に終了した場合は MGE_S_OK を、指定したストリームが無効な場合は MGE_INVALIDVSTREAM を返す。それ以外の場合はエラーコードを返す。

## 補足

*Effect が加えられるのは MGEChannel1 につながっているストリームのみ。
Power Movie PCI ドライバ Ver1.00、または Ver1.01 使用時、MGEChannel2 につながっているストリームを指定してもエラーを返さない。

# MGE_VStreamGetEffect

ストリーム上の現在の特殊効果を取得する。

## 書式

〈C/C++〉

```
MGEAPI MGERESULT WINAPI
MGE_VStreamGetEffect(HMGESTREAM hmgestmStream, LPDWORD lpdwEffect);
```

〈Basic〉

```
Declare Function MGE_VStreamGetEffect Lib "mge.dll" (ByVal hmgestmStream As Long, lpdwEffect As
Long) As Long
```

## 引数

| | |
|---|---|
| hmgestmStream | 特殊効果を取得するストリームハンドル |
| lpdwEffect | 現在の特殊効果を返すアドレス |

## 戻り値

正常に終了した場合、MGE_S_OK　を、指定したストリームが無効な場合はMGE_INVALIDVSTREAM を返す。それ以外の場合はエラーコードを返す。

# APPENDIX

# ファンクション使用時の戻り値一覧

MGEAPI 使用時の戻り値は次のとおりです。

| 戻り値 | 意　　味 |
|---|---|
| MGE_S_OK | 正常に終了。 |
| MGE_S_FALSE | ファンクションの実行自体は成功したが、ファンクションの意味的には失敗した。 |
| MGE_S_MEANINGLESS | ファンクションの実行自体は成功したが、実行ファンクションとしては意味なし。 |
| MGE_S_DISABLED | ファンクションの実行自体は成功したが、実行ファンクションとしては無効。 |
| MGE_E_GENERALERROR | その他のエラーが発生。 |
| MGE_E_INVALIDPARAMETER | 設定されたパラメータは無効。 |
| MGE_E_NOMEMORY | メモリが不足している。 |
| MGE_E_BUSY | オブジェクト（またはデバイス）がビジー状態である。 |
| MGE_E_NOTEQUIPPED | デバイスが実装されていない。 |
| MGE_E_NORESOURCE | 使用するリソースがない。 |
| MGE_E_NOTSUPPORTED | サポートされていないファンクションを実行した。 |
| MGE_E_SURFACE_CONNECTED | サーフェスは既に接続されている。 |
| MGE_E_CHANNEL | 無効なチャネルを指定した。 |
| MGE_E_INVALIDSURFACE | 無効なサーフェスを指定した。 |
| MGE_E_VSRC_CONNECTED | ビデオ入力オブジェクト（ビデオデコーダオブジェクト）は既に接続されている。 |
| MGE_E_NOT_IMPLEMENT | ファンクションは実装されていない。 |
| MGE_E_INVALIDSTREAM | 指定したストリームは無効。 |
| MGE_E_HW_NOTFOUND | Power Movie PCI が見つからない。 |
| MGE_E_IO | Power Movie PCI が動作していない。 |
| MGE_E_VSRC_NOTCONNECTED | ビデオ入力オブジェクト（ビデオデコーダオブジェクト）が接続されていない。 |
| MGE_E_VDST_CONNECTED | ビデオ出力オブジェクト（ビデオエンコーダオブジェクト）は既に接続されている。 |

| 戻り値 | 意　　味 |
|---|---|
| MGE_E_VDST_NOTCONNECTED | ビデオ出力オブジェクト（ビデオエンコーダオブジェクト）が接続されていない。 |
| MGE_E_STREAMNOTCOMPLETED | ストリームがサーフェスに結びついていない。 |
| MGE_E_NODIRECTDRAW | DirectDraw によるオーバーレイが行なえない。<br>もしくは DirectDraw がない。 |
| MGE_E_INVALIDSURFACEFORMAT | サーフェスのフォーマット指定が間違っている。 |
| MGE_E_INVALIDRECT | rect の指定が無効。 |
| MGE_E_SUPEROVLSURFACE | Super Overlay 時には実行できない。 |
| MGE_E_FILE | ファイルの入出力で失敗した。 |
| MGE_E_BITMAPFORMAT | ビットマップ形式の指定が無効。 |
| MGE_E_NOJPEGCODEC | JPEG コーデックが見つからない。 |
| MGE_E_JPEGSTREAM | 無効な JPEG データ。 |
| MGE_E_JPEGENCODE | JPEG エンコード時にエラーが発生。 |
| MGE_E_DIRECTDRAW | DirectDraw API がエラーを返した。 |
| MGE_E_NOSCLCALLED | MGE_SetCooperativeLevel を呼び出していない。 |
| MGE_E_REGISTRY | 指定された Registry にアクセス中にエラーが発生した。 |
| MGE_E_STREAMNOTSTARTED | ストリームが開始されていないか一時停止中である。 |
| MGE_E_UNICODE | UNICODE はサポートしていない。 |
| MGE_E_INVALIDPIXELCOMBINATION | 無効な Pixel Format の組を指定した。 |
| MGE_E_INTERNAL | 内部エラーです。サポート窓口まで連絡してください。 |

備考：戻り値は mgeerror.h（Visual C++）/mgeerror.bas（Visual Basic）に定義されています。

# 定　数　・　定　義　一　覧

| 信号状態の定義 | 意　　味 |
|---|---|
| MGE_SS_STABLE | ビデオ信号が安定して供給されている。<br>または |
| MGE_SS_NOTSTABLE | ビデオ信号が安定してないか、供給されていない。 |
| MGE_SS_COLOR_UNKNOWN | ビデオ信号のカラー信号が判別不能。<br>(MGE_SS_NOTSTABLE 状態の時に返ります。)<br>または |
| MGE_SS_COLOR_COLOR | ビデオ信号のカラー信号を検出した。<br>または |
| MGE_SS_COLOR_MONO | ビデオ信号のカラー信号を検出できないか、白黒信号を検出した。 |
| MGE_SS_TYPE_UNKNOWN | ビデオ信号の信号方式が判別不能。<br>(MGE_SS_NOTSTABLE 状態の時に返ります。)<br>または |
| MGE_SS_TYPE_NTSC | ビデオ信号が NTSC 方式。<br>または |
| MGE_SS_TYPE_PAL | ビデオ信号が PAL 方式(Power Movie PCI ではサポートされない)。 |

| カラーキーに使用するための<br>代表的な色の定義 | 意　　味 |
|---|---|
| MGE_COLORKEY_BLACK | 黒 |
| MGE_COLORKEY_RED | 赤 |
| MGE_COLORKEY_GREEN | 緑 |
| MGE_COLORKEY_YELLOW | 黄色 |
| MGE_COLORKEY_BLUE | 青 |
| MGE_COLORKEY_MAGENTA | 赤紫 |
| MGE_COLORKEY_CYAN | 水色 |
| MGE_COLORKEY_WHITE | 白 |

| 特殊効果を指定するための定数 | 意　　味 |
|---|---|
| MGE_EF_MIRROR_ON | 左右反転します。 |
| MGE_EF_MIRROR_OFF | 左右反転をしません。 |
| MGE_EF_UD_ON | 上下反転をします。 |
| MGE_EF_UD_OFF | 上下反転をしません。 |
| MGE_EF_BALL_ON | Power Movie PCI ではサポートされません。 |
| MGE_EF_BALL_OFF | Power Movie PCI ではサポートされません。 |

備考：各定数は or(|)で複数指定できます。xx_ON と xx_OFF を同時に指定した場合は、xx_ON を指定したのと同じ扱いとなります。
例えば、(MGE_EF_MIRROR_ON|MGE_EF_MIRROR_OFF)は MGE_EF_MIRROR_ON となります。

# データ構造解説

PixelFormat の詳細は次のとおりです。

### 8bit パレット

7 6 5 4 3 2 1 0

| Palette Index |
|---|

*Paletteデータへのインデックスです。

### RGB555

1 1 1 1 1 1
5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0

| U | Red | Green | Blue |
|---|---|---|---|

＊U は未定義ビットです。

### RGB565

1 1 1 1 1 1
5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0

| Red | Green | Blue |
|---|---|---|

### RGB24

2 2 2 2 1 1 1 1 1 1 1 1 1 1
3 2 1 0 9 8 7 6 5 4 3 2 1 0 9 8 7 6 5 4 3 2 1 0

| Red | Green | Blue |
|---|---|---|

## RGB32

| 3 1 | 3 0 | 2 9 | 2 8 | 2 7 | 2 6 | 2 5 | 2 4 | 2 3 | 2 2 | 2 1 | 2 0 | 1 9 | 1 8 | 1 7 | 1 6 | 1 5 | 1 4 | 1 3 | 1 2 | 1 1 | 1 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Undefined | | | | | | | | | | Red | | | | | | | | Green | | | | | | | | Blue | | | | | |

## YUY2

| 3 1 | 3 0 | 2 9 | 2 8 | 2 7 | 2 6 | 2 5 | 2 4 | 2 3 | 2 2 | 2 1 | 2 0 | 1 9 | 1 8 | 1 7 | 1 6 | 1 5 | 1 4 | 1 3 | 1 2 | 1 1 | 1 0 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| V | | | | | | | | | | Y1 | | | | | | | | U | | | | | | | | Y0 | | | | | |

2pixel 分で色を表現します。

- Y0　1 pixel 目の輝度信号
- Y1　2 pixel 目の輝度信号
- U　輝度信号と赤色成分の差
- V　輝度信号と青色成分の差

# 用　語　解　説

## オーバーレイ(Overlay)

Power Movie PCI ではある画面とある画面を重ね合わせることをオーバーレイすると表現します。通常キーカラーを使用して重ね合わせる方法（キーカラーに相当する部分を別のデータで表示します。DirectDraw Overlay サーフェスがこれに相当します)で行うことを指します。
Power Movie PCI ではDirectDraw Overlay の他にグラフィックボードに直接外部入力を表示する Super Overlay も用意しています。Super Overlay はキーカラーは使用できませんが、Window handle の矩形ベースで表示することができます。

## DirectDraw Overlay

Microsoft DirectDraw の機能を使用してオーバーレイを実現します。キーカラーが使用できます。

## Super Overlay

グラフィックボードに直接外部入力映像を表示しオーバーレイを実現します。キーカラーは使用できませんが、Window handle で表される矩形のクリッピングが可能です。

## サーフェス

イメージを保持する領域のことを表し、３種類のサーフェスがあります。

Super Overlay サーフェス

グラフィックボード画面上にサーフェスとして領域を確保します。
グラフィックボード画面に作成されるため、このサーフェスはRGB 系のサーフェスとなります。接続できるVideo デコーダーのチャンネルはMGEChannel1 で作成されたもののみとなります。詳しくはMGE_VStreamConnect を参照ください。

DirectDraw Overlay サーフェス

DirectDraw のオーバーレイ機能を利用し、オフスクリーン(グラフィックボードの画面外)にサーフェスとして領域を確保します。
DirectDraw Overlay サーフェスはYUV 系のサーフェスとなります。接続できるVideo デコーダーのチャンネルはMGEChannel2 で作成されたもののみとなります。詳しくはMGE_VStreamConnect を参照ください。
このサーフェスはキーカラーを使用することによってグラフィックボード上にオーバーレイ表示することができます。

メモリサーフェス

メインメモリ上に領域を確保します。入力(Video デコーダー)から静止画を得るために使用したり、出力(Video エンコーダー)への表示のために使用します。

## ストリーム

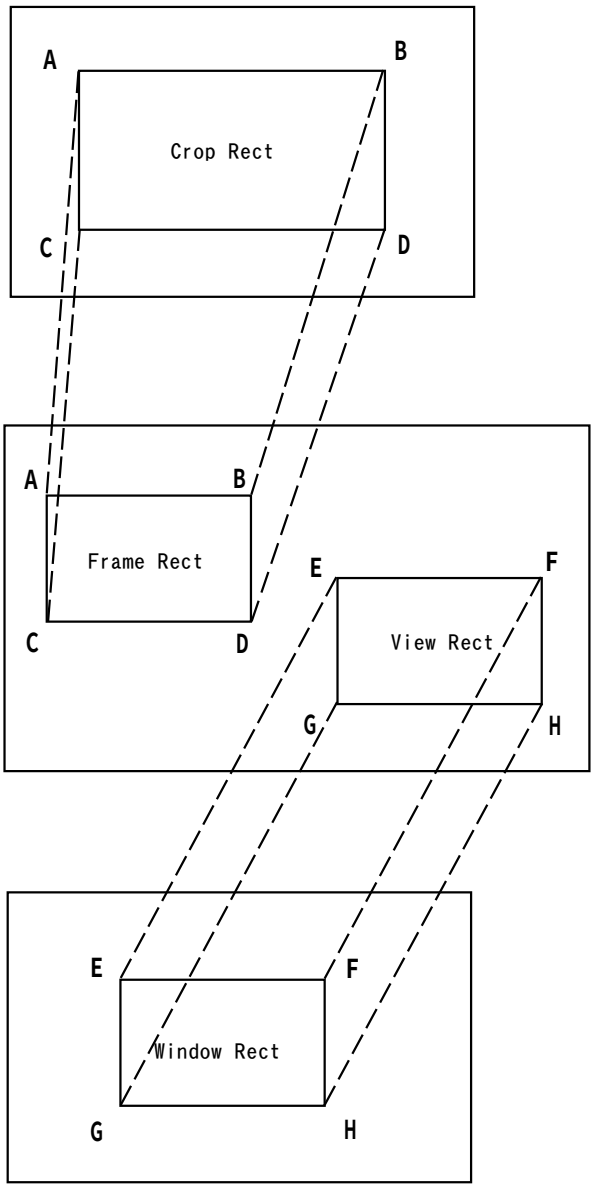
ビデオデータの流れを表す仮想的なもの。例えば、ビデオデコーダ(入力)からサーフェスへのデータの流れを結びつけるために使用します。

## レクト

レクトは下記４種類のレクト(矩形)が存在します。

| | |
|---|---|
| Crop Rect | Data Producer(例えばVideo デコーダー) の取り込み位置を指定します。 |
| Frame Rect | Crop Rectで指定された矩形をFrame bufferのどこに置くかを指定します。<br>この時(可能ならば)拡大縮小が行われます。<br>Power Movie PCIでは拡大は行なえません。 |
| View Rect | Frame buffer上のどこの領域を表示するかを指定します。 |
| Window Rect | View Rectで指定された矩形をData Consumer(例えばVideo エンコーダー)の<br>どこに置くかを指定します。この時(可能ならば)拡大縮小が行われます。<br>Power Movie PCIでは拡大縮小は行なえません。 |

下記の図のCrop RectのA点、B点、C点、D点はFrame RectのA点、B点、C点、D点に
対応します。また、View RectのE点、F点、G点、H点はWindow Rectの
E点、F点、G点、H点に対応します。



サーフェスがSuper Overlayの場合は、Frame bufferが存在しませんのでFrame RectとView Rectは指定できません。(指定するとMGE_S_FALSEが返ります。)
Data Source(Video デコーダー)とData Consumer(Super Overlay)が直結し、Crop RectとWindow Rectの関係で縮小が行われます。
(拡大はできません。MGE_E_INVALIDRECTが返ります。)

サーフェスがDirectDraw Overlayの場合は、すべてのRectが存在します。

Memory SurfaceがVideoInStreamに接続されている場合View RectとWindow Rectは指定できません。
(指定するとMGE_S_FALSEが返ります)
Memory SurfaceがVideoOutStreamに接続されている場合Crop RectとFrame Rectは指定できません。
(指定するとMGE_S_FALSEが返ります)
現在のPower Movie PCIの実装ではVideoInStreamとVideoOutStreamをMemory Surface経由で混合することはできません。

# INDEX

# INDEX