

Dreamcast

ミドルウェアマニュアル

ミドルウェアプレイヤー外部仕様書

1998年 6月30日	Ver . 0.50
1998年 8月 3日	Ver . 0.55
1998年10月 2日	Ver . 1.00
1998年10月15日	Ver . 1.01
1998年10月29日	Ver . 1.03
1998年12月 8日	Ver . 1.10
1999年 3月17日	Ver . 1.26
1999年 6月11日	Ver . 1.30
1999年12月22日	Ver . 2.24
2000年 2月25日	Ver . 2.28
2000年 3月16日	Ver . 2.30

変 更 履 歴

年月日	バージョン	変 更 内 容
1998.06.30	0.50	新規作成。
1998.08.03	0.55	記述ミスを修正。(mwPlyStartFname の引数)
1998.10.02	1.00	上記記述ミスを再度修正 (mwPlyStartFname,mwPlyGetStat の引数) Wave 再生用関数仕様、サンプルを追加。
1998.10.15	1.01	記述ミスを修正。 Wave 用データ型仕様を追加。 共通関数に以下の関数を追加。 mwPlySetOutVol,mwPlyGetOutVol,mwPlySetOutPan,mwPlyGetOutPan, mwPlyPause
1998.10.29	1.03	記述ミスを修正。
1998.12.08	1.10	記述ミスを修正。
1999.03.17	1.26	TrueMotion と MPEG/Audio の説明 (データ型、関数仕様を含む) を追加。 MPEG Sofdec 用以下のデータ型仕様を追加。 MWS_PLY_INIT_SFD,MWS_PLY_OPT_SFD MPEG Sofdec 用以下のデータ型仕様を変更。 MWS_PLY_CPRM_SFD MPEG Sofdec 用として、以下の関数を追加。 mwPlyCalcWorkSofdec,mwPlySetDispMode,mwPlySetFastHalfpel mwPlySetAudioSw,mwPlySetVideoSw,mwPlyGetNumDropFrm 記述ミスを修正。
1999.06.11	1.30	DualSpeech の説明 (関数仕様) を追加。 音声コーデック用以下のデータ型仕様を変更。 MWS_PLY_CPRM 共通関数に以下の関数を追加。 mwPlyStartSj,mwPlyStartMem
1999.12.22	2.24	記述ミスを修正。 音声コーデック用以下のデータ型仕様を変更 MWS_PLY_CPRM WAVE 再生、MPEG/Audio 再生、DualSpeech 再生、ストリームジョイント を使用しての再生、メモリからの再生でのサンプルを MWS_PLY_CPRM の 仕様変更に伴い修正。 下記関数仕様を MWS_PLY_CPRM の仕様変更に伴い加筆。 mwPlyCreateWav,mwPlyCreateMpa,mwPlyCreateDisd mwPlyExecServer の記述を mwExecMainServer に変更。 以下の説明を追加。 システム構成、内部モジュール構成、コントロールフロー、 サウンドリソース、ビデオリソース mwPlyCalcWorkSofdec 関数を mwPlyCalcWorkCprmSfd 関数に変更。
2000.02.25	2.28	記述ミスを修正。
2000.03.16	2.30	ライブラリのバージョンアップに伴いバージョン番号を更新。 記述ミスを修正。

目 次

1. 概 要	1
1.1 目 的	1
1.2 モジュール構成	1
2. ミドルウェアライブラリの内部構成	2
3. ミドルウェアの使用方法	3
3.1 ミドルウェアライブラリの初期化	3
3.2 ミドルウェア再生ハンドル	3
3.3 ミドルウェアライブラリの使用方法	4
3.4 ミドルウェア再生ハンドルの動作状態	5
3.5 MPEG SofdecF/X の再生	6
3.6 WAVE の再生	8
3.7 TrueMotion の再生	9
3.8 MPEG/Audio の再生	10
3.9 DualSpeech の再生	11
3.10 ストリームジョイントによる再生	12
3.11 メモリからの再生	14
4. データ仕様	15
4.1 定 数	16
4.2 データ型	18
5. 関数仕様	21
5.1 MPEG SofdecF/X 用関数	23
5.2 WAVE 用関数	32
5.3 TrueMotion 用関数	34
5.4 MPEG/Audio 用関数	36
5.5 DualSpeech 用関数	38
5.6 共通関数	40
6. 付録	47
6.1 ドアオープンチェック	47
6.2 MPEG SofdecF/X と ADX との並行再生	48
6.3 システム構成	49
6.4 モジュール構成	49
6.5 コントロールフロー	50
6.6 サウンドリソース	51
6.7 ビデオリソース	52

1. 概 要

1.1 目 的

本ライブラリは、動画や音声を容易に再生するためのライブラリである。下記の形式のファイルをストリーム再生できる。

- (1) MPEG SofdecF/X データファイル
映像を MPEG/Video、音声を Sofdec/Audio により圧縮しマルチプレックスしたデータ。
- (2) WAVE データファイル
WAVE フォーマットの非圧縮 PCM 音声データ、Dreamcast ADPCM 圧縮された音声データ。
- (3) TrueMotion データファイル
映像を TrueMotion、音声を DK3/DK4 により圧縮しマルチプレックスしたデータ。
- (4) MPEG/Audio データファイル
MPEG1/Audio Layer 圧縮された音声データ。
- (5) DualSpeech データファイル
DualSpeech 圧縮された音声データ。

1.2 モジュール構成

ミドルウェアライブラリのモジュール構成図を以下に示す。

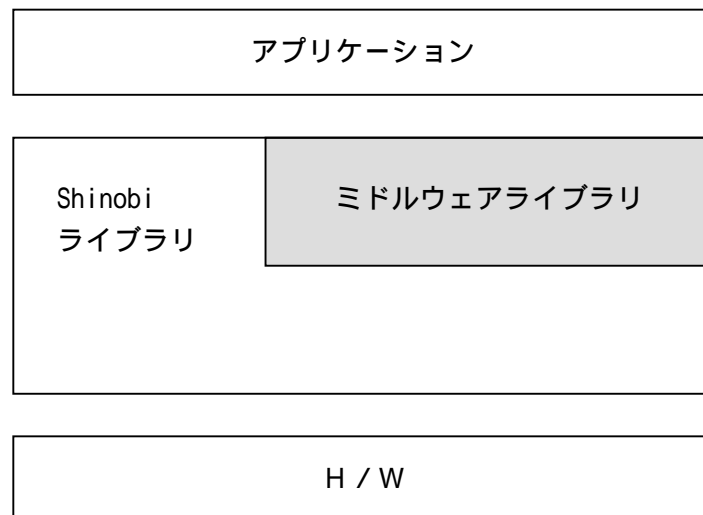


図 1 - 1 モジュール構成

2. ミドルウェアライブラリの内部構成

ミドルウェアライブラリは、以下のモジュールから構成されている。アプリケーションは、ミドルウェア API を用いることによって、容易に動画や音声を再生できる。

表 2 - 1 ミドルウェアライブラリの内部モジュール

モジュール名	説 明
モジュールマネージャ	デコーダに CPU タイムを割り当てる。
ストリームコントローラ	インターリーブされた映像と音声データを読み込む。
ストリームスプリッタ	インターリーブされた映像と音声データを分離し、デコーダに渡す。
ビデオデコーダ	圧縮された映像データを展開し、ビデオレンダラに渡す。
オーディオデコーダ	圧縮された音声データを展開し、オーディオレンダラに渡す。
ビデオレンダラ	展開された映像データを出力する。
オーディオレンダラ	展開された音声データを出力する。

以下に、ミドルウェアライブラリの内部構成図を示す。

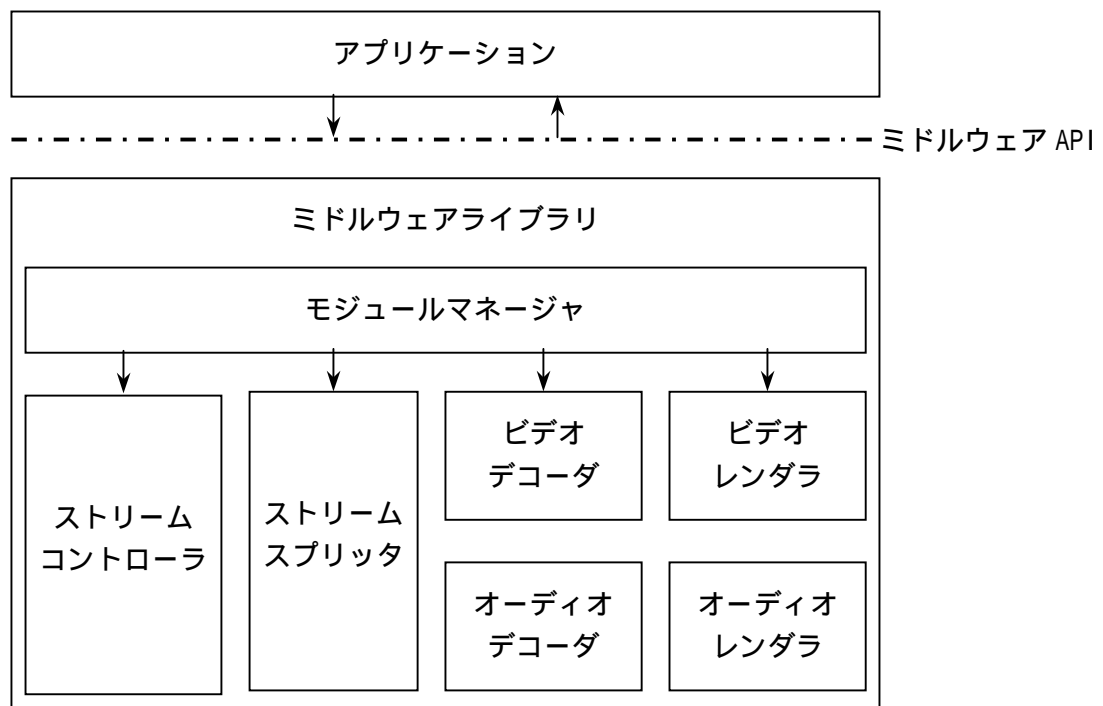


図 1.2 - 1 ミドルウェアライブラリの内部構成

3. ミドルウェアの使用方法

3.1 ミドルウェアライブラリの初期化

ミドルウェアライブラリの初期化は、各コーデックに対応した初期化関数を呼び出さなければならない。

< MPEG SofdecF/X の場合 >

```
mwPlyPreInitSofdec();  
sbInitSystem(NJD_RESOLUTION_640x480_NTSC_I, NJD_FRAMEBUFFER_MODE_ARGB8888, 1);  
mwPlyInitSofdec(&iprm);
```

< WAVE の場合 >

```
sbInitSystem(NJD_RESOLUTION_640x480_NTSC_I, NJD_FRAMEBUFFER_MODE_ARGB8888, 1);  
mwPlyInitWav(MWD_PLY_SVR_VSYNC);
```

< TrueMotion の場合 >

```
sbInitSystem(NJD_RESOLUTION_640x480_NTSC_I, NJD_FRAMEBUFFER_MODE_ARGB8888, 1);  
mwPlyInitTM();
```

< MPEG/Audio の場合 >

```
sbInitSystem(NJD_RESOLUTION_640x480_NTSC_I, NJD_FRAMEBUFFER_MODE_ARGB8888, 1);  
mwPlyInitMpa(MWD_PLY_SVR_VSYNC);
```

< DualSpeech の場合 >

```
sbInitSystem(NJD_RESOLUTION_640x480_NTSC_I, NJD_FRAMEBUFFER_MODE_ARGB8888, 1);  
mwPlyInitDlsd(MWD_PLY_SVR_VSYNC);
```

3.2 ミドルウェア再生ハンドル

各コーデックによって圧縮された音声や動画データは、ミドルウェア再生ハンドルを用いて再生する。まず、各コーデック用のミドルウェア再生ハンドルを生成する。このハンドルの生成には、使用するコーデックに対応した API を用いる。

```
MWPLY          plysfd, plywav, plytm, plympa, plydlsd;  
MWS_PLY_CPRM_SFD cprm_sfd;  
MWS_PLY_CPRM_TM  cprm_tm;  
MWS_PLY_CPRM     cprm_wav, cprm_mpa, cprm_dlsd;
```

```
plysfd = mwPlyCreateSofdec(&cprm_sfd); /* MPEG SofdecF/X 再生用ハンドルの生成 */  
plywav = mwPlyCreateWav(&cprm_wav);    /* WAVE データ再生用ハンドルの生成 */  
plytm  = mwPlyCreateTM(&cprm_tm);      /* TrueMotion データ再生用ハンドルの生成 */  
plympa = mwPlyCreateMpa(&cprm_mpa);    /* MPEG/Audio データ再生用ハンドルの生成 */  
plydlsd = mwPlyCreateDlsd(&cprm_dlsd); /* DualSpeech データ再生用ハンドルの生成 */
```

ハンドルの生成には、異なる API を使用するが、再生指示には同じ API を使用することができる。
GD-ROM 上のデータを再生する場合、mwPlyStartFname 関数により動画や音声の再生開始を制御する。

```
mwPlyStartFname(plysfd, "SAMPLE.SFD"); /* MPEG SofdecF/X データの再生開始 */
mwPlyStartFname(plywav, "SAMPLE.WAV"); /* WAVE データの再生開始 */
mwPlyStartFname(plytm, "SAMPLE.AVI"); /* TrueMotion データの再生開始 */
mwPlyStartFname(plympa, "SAMPLE.MP2"); /* MPEG/Audio データの再生開始 */
mwPlyStartFname(plydlsd, "SAMPLE.DUS"); /* DualSpeech データの再生開始 */
```

ストリームジョイントでストリーミングしながら再生する場合、mwPlyStartSj 関数により動画や音声の再生開始を制御する。

```
mwPlyStartSj(plysfd, sj); /* MPEG SofdecF/X データの再生開始 */
mwPlyStartSj(plywav, sj); /* WAVE データの再生開始 */
mwPlyStartSj(plympa, sj); /* MPEG/Audio データの再生開始 */
mwPlyStartSj(plydlsd, sj); /* DualSpeech データの再生開始 */
```

メモリ上のデータを再生する場合、mwPlyStartMem 関数により動画や音声の再生停止を制御する。

```
mwPlyStartMem(plysfd, addr, len); /* MPEG SofdecF/X データの再生開始 */
mwPlyStartMem(plywav, addr, len); /* WAVE データの再生開始 */
mwPlyStartMem(plympa, addr, len); /* MPEG/Audio データの再生開始 */
mwPlyStartMem(plydlsd, addr, len); /* DualSpeech データの再生開始 */
```

再生方法に関わらず、mwPlyStop 関数により動画や音声の再生停止を制御する。

3.3 ミドルウェアライブラリの使用方法

ミドルウェアライブラリの内部状態は、メイン処理用サーバ関数(mwExecMainServer)を呼び出すことで更新される。

```
while (1) {
    /* ペリフェラル情報の取得処理 */
    :
    mwExecMainServer(); // 動画データの描画処理など
    /* 描画処理 */
    :
    njWaitVSync();
}
```

3.4 ミドルウェア再生ハンドルの動作状態

ミドルウェア再生ハンドルの動作状態を以下に示す。生成した直後は、STOP 状態となる。再生開始後、状態は、PREP -> PLAYING -> PLAYEND と遷移する。GD リードエラーが発生すると、ERROR に遷移する。

表 3 - 1 ハンドルの状態

状 態	説 明
STOP	再生が停止している状態
PREP	再生の準備をしている状態
PLAYING	再生中の状態
PLAYEND	再生が終了した状態
ERROR	エラーが発生した状態

状態遷移図を以下に示す。

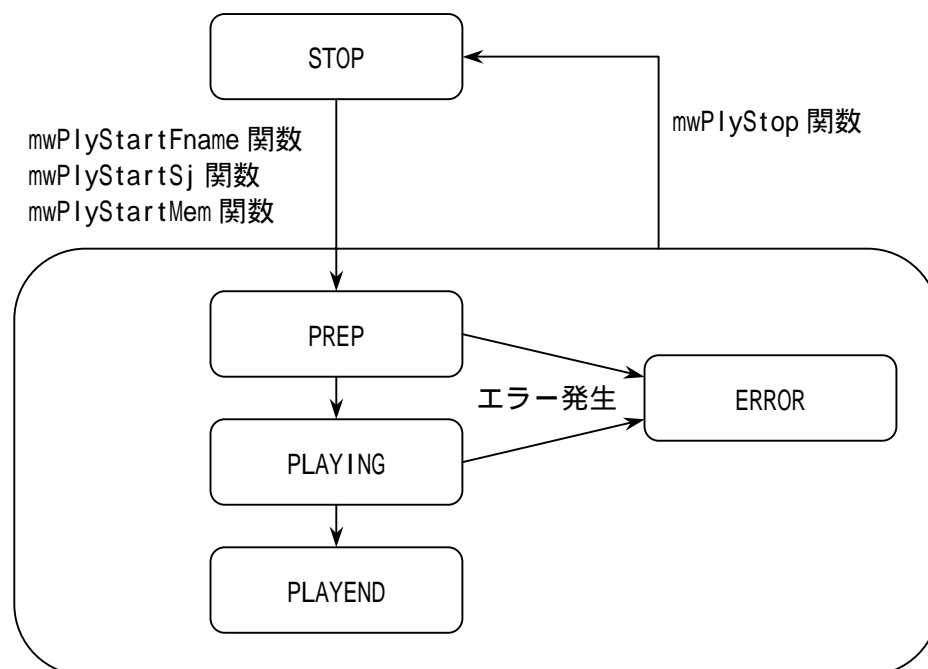


図 3 - 1 状態遷移図

3.5 MPEG SofdecF/X の再生

以下に、MPEG SofdecF/X を再生するサンプルプログラムを示す。

< サンプルプログラム >

```
/* 表示モード */
#define SYS_MODE      NJD_RESOLUTION_640x480_NTSC1
#define SYS_FRAME     NJD_FRAMEBUFFER_MODE_ARGB8888
#define SYS_COUNT     1

/* 頂点バッファサイズ(1つをマイナスにすると2Vレーテンシモード) */
#define SFD_VB_OP      -500000
#define SFD_VB_OM      0
#define SFD_VB_TP      20000
#define SFD_VB_TM      0
#define SFD_VB_PT      0

/* アプリケーションメイン関数 */
void main(void)
{
    MWPLY              ply;                /* ミドルウェア再生ハンドル */
    MWE_PLY_STAT       stat;               /* ハンドルの状態 */
    MWS_PLY_CPRM_SFD   cprm;              /* ハンドル生成のパラメータ構造体 */

    mwPlyPreInitSofdec();                  /* 割り込みスタックなどの設定 */
    sbInitSystem(SYS_MODE, SYS_FRAME, SYS_COUNT);
    njInitVertexBuffer(VB_OP, VB_OM, VB_TP, VB_TM, VB_PT);
    /*
     * 画面やサウンドの初期化
     */
    memset(&iprm, 0, sizeof(iprm));        /* 予約メンバのゼロ設定のため */
    iprm.mode= SYS_MODE;
    iprm.frame= SYS_FRAME;
    iprm.count= SYS_COUNT;
    iprm.latency= MWD_PLY_LATENCY(VB_OP, VB_OM, VB_TP, VB_TM, VB_PT);
    mwPlyInitSofdec(&iprm);                /* ライブラリの初期化 */
}
```

```

memset(&cprm, 0, sizeof(cprm));
cprm.ftype= MWD_PLY_FTYPE_SFD;
cprm.dtype= MWD_PLY_DTYPE_AUTO;
cprm.max_bps = 450*1024*8;
cprm.max_width = 320;
cprm.max_height = 480;
cprm.nfrm_pool_wk = 3;
cprm.wksize = mwPlyCalcWorkCprmSfd(cprm);
cprm.work = syMalloc(cprm.wksize);
ply = mwPlyCreateSofdec(&cprm);
mwPlyStartFname(ply, "SAMPLE.SFD");
for (;;) {
    mwExecMainServer();
    stat = mwPlyGetStat(ply);
    if ( stat == MWE_PLY_STAT_PLAYEND ) {
        break;
    }
    njWaitVSync();
}
mwPlyStop(ply);
mwPlyDestroy(ply);
syFree(cprm.work);
mwPlyFinishSofdec();
}

```

/* 予約メンバのゼロ設定のため */
 /* 映像 + 音声 */
 /* 画質優先 */
 /* ビットレート : 450 Kbyte/sec */
 /* 画像サイズ : 320x480 */
 /* フレームバッファの枚数 */
 /* 作業領域サイズの計算 */
 /* ハンドルの生成 */
 /* 再生開始 */
 /* ミドルウェアライブラリの実行 */
 /* ハンドルの状態の取得 */
 /* V-Sync 待 ち */
 /* 再生停止 */
 /* ハンドルの消去 */
 /* ライブラリの初期化 */

3.6 WAVE の再生

以下に、WAVE ファイルを再生するサンプルプログラムを示す。

< サンプルプログラム >

/* アプリケーションメイン関数 */

```
void main(void)
{
    MWPLY                ply;                /* ミドルウェア再生ハンドル */
    MWE_PLY_STAT         stat;               /* ハンドルの状態 */
    MWS_PLY_CPRM         cprm;              /* ハンドル生成のパラメータ構造体 */

    /*
     * 画面やサウンドの初期化
     */
    mwPlyInitWav(MWD_PLY_SVR_VSYNC);         /* ライブラリの初期化 */
    memset(&cprm, 0, sizeof(cprm));          /* 予約メンバのゼロ設定のため */
    cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
    cprm.buf = syMalloc(cprm.size);
    cprm.sample = MWD_PLY_SAMPLE_NUM;
    ply = mwPlyCreateWav(&cprm);              /* ハンドルの生成 */
    mwPlyStartFname(ply, "SAMPLE.WAV");      /* 再生開始 */
    for (;;) {
        mwExecMainServer();                  /* ミドルウェアライブラリの実行 */
        stat = mwPlyGetStat(ply);            /* ハンドルの状態の取得 */
        if ( stat == MWE_PLY_STAT_PLAYEND ) {
            break;
        }
        njWaitVSync();                       /* V-Sync 待 ち
    */
    }
    mwPlyStop(ply);                          /* 再生開始 */
    mwPlyDestroy(ply);                       /* ハンドルの消去 */
    syFree(cprm.buf);
    mwPlyFinishWav();                        /* ライブラリの初期化 */
}
```

3.7 TrueMotion の再生

以下に、TrueMotion を再生するサンプルプログラムを示す。

< サンプルプログラム >

/* アプリケーションメイン関数 */

void main(void)

```
{
    MWPLY          ply;          /* ミドルウェア再生ハンドル */
    MWE_PLY_STAT    stat;        /* ハンドルの状態 */

    /*
     * 画面やサウンドの初期化
     */
    mwPlyInitTM();              /* ライブラリの初期化 */
    ply = mwPlyCreateTM(NULL);  /* ハンドルの生成 */
    mwPlyStartFname(ply, "SAMPLE.AVI"); /* 再生開始 */
    for (;;) {
        mwExecMainServer();     /* ミドルウェアライブラリの実行 */
        stat = mwPlyGetStat(ply); /* ハンドルの状態の取得 */
        if ( stat == MWE_PLY_STAT_PLAYEND ) {
            break;
        }
        njWaitVSync();          /* V-Sync 待 ち
    */
    }
    mwPlyStop(ply);             /* 再生開始 */
    mwPlyDestroy(ply);          /* ハンドルの消去 */
    mwPlyFinishTM();            /* ライブラリの初期化 */
}
```

3.8 MPEG/Audio の再生

以下に、MPEG/Audio を再生するサンプルプログラムを示す。

< サンプルプログラム >

```
#define NUM_HNDL    (1)                                /* 生成するハンドル数          */

/* アプリケーションメイン関数 */
void main(void)
{
    MWPLY            ply;                                /* ミドルウェア再生ハンドル    */
    MWE_PLY_STAT     stat;                              /* ハンドルの状態              */
    MWS_PLY_CPRM     cprm;                              /* ハンドル生成のパラメータ構造体 */

    /*
     * 画面やサウンドの初期化
     */
    mwPlyInitMpa(MWD_PLY_SVR_VSYNC);                    /* ライブラリの初期化          */
    memset(&cprm, 0, sizeof(cprm));                     /* 予約メンバのゼロ設定のため  */
    cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
    cprm.buf = syMalloc(cprm.size);
    cprm.libwork = syMalloc(MWD_MPA_CALC_WORK(NUM_HNDL));
    cprm.extsize[MWD_CH_L] = MWD_MPA_EXTRA_SIZE;
    cprm.extsize[MWD_CH_R] = MWD_MPA_EXTRA_SIZE;
    cprm.sample = MWD_PLY_SAMPLE_NUM;
    ply = mwPlyCreateMpa(&cprm);                         /* ハンドルの生成              */
    mwPlyStartFname(ply, "SAMPLE.MP2");                 /* 再生開始                    */
    for (;;) {
        mwExecMainServer();                             /* ミドルウェアライブラリの実行 */
        stat = mwPlyGetStat(ply);                       /* ハンドルの状態の取得        */
        if ( stat == MWE_PLY_STAT_PLAYEND ) {
            break;
        }
        njWaitVSync();                                  /* V-Sync 待ち                 */
    }
    mwPlyStop(ply);                                     /* 再生開始                    */
    mwPlyDestroy(ply);                                  /* ハンドルの消去              */
    syFree(cprm.libwork);
    syFree(cprm.buf);
    mwPlyFinishMpa();                                   /* ライブラリの初期化          */
}
```

3.9 DualSpeech の再生

以下に、DualSpeech を再生するサンプルプログラムを示す。

< サンプルプログラム >

```
#define NUM_HNDL    (1)                                /* 生成するハンドル数          */

/* アプリケーションメイン関数 */
void main(void)
{
    MWPLY            ply;                                /* ミドルウェア再生ハンドル    */
    MWE_PLY_STAT     stat;                               /* ハンドルの状態              */
    MWS_PLY_CPRM     cprm;                              /* ハンドル生成のパラメータ構造体 */

    /*
     * 画面やサウンドの初期化
     */
    mwPlyInitDlisd(MWD_PLY_SVR_VSYNC);                  /* ライブラリの初期化          */
    memset(&cprm, 0, sizeof(cprm));                     /* 予約メンバのゼロ設定のため  */
    cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
    cprm.buf = syMalloc(cprm.size);
    cprm.libwork = syMalloc(MWD_DLSD_CALC_WORK(NUM_HNDL));
    cprm.extsize[MWD_CH_L] = MWD_DLSD_EXTRA_SIZE;
    cprm.sample = MWD_DLSD_SAMPLE_NUM;
    cprm.limit = MWD_DLSD_MEMORY_LIMIT;
    ply = mwPlyCreateDlisd(&cprm);                       /* ハンドルの生成              */
    mwPlyStartFname(ply, "SAMPLE.DUS");                 /* 再生開始                    */
    for (;;) {
        mwExecMainServer();                             /* ミドルウェアライブラリの実行 */
        stat = mwPlyGetStat(ply);                       /* ハンドルの状態の取得        */
        if ( stat == MWE_PLY_STAT_PLAYEND ) {
            break;
        }
        njWaitVSync();                                  /* V-Sync 待ち                */
    }
    mwPlyStop(ply);                                     /* 再生開始                    */
    mwPlyDestroy(ply);                                  /* ハンドルの消去              */
    syFree(cprm.libwork);
    syFree(cprm.buf);
    mwPlyFinishDlisd();                                 /* ライブラリの初期化          */
}
```

3.10 ストリームジョイントによる再生

以下に、ストリームジョイントを使用して再生するサンプルプログラムを示す。

< サンプルプログラム >

```
#define NUM_HNDL    (1)                                /* 生成するハンドル数          */

/* アプリケーションメイン関数 */
void main(void)
{
    MWPLY            ply;                                /* ミドルウェア再生ハンドル    */
    MWE_PLY_STAT     stat;                               /* ハンドルの状態              */
    MWS_PLY_CPRM     cprm;                              /* ハンドル生成のパラメータ構造体 */
    SJ               sj;                                /* ストリームジョイントハンドル */
    SJCK             ck, ck2;                           /* チャンク                    */
    Sint8            *buf;                              /* ストリームジョイント作業領域 */
    Sint8            *data;                             /* データへのポインタ          */
    Sint32           size;                              /* データサイズ                */
    Sint32           pos;                               /* データ位置                  */
    Sint32           len;                               /* データ転送量                */
    Sint32           nroom;                             /* バッファの空き容量          */

    /*
     * 画面やサウンドの初期化
     */
    mwPlyInitWav(MWD_PLY_SVR_VSYNC);                    /* ライブラリの初期化          */
    buf = syMalloc(2048 * 24);
    sj = SJRBF_Create(buf, 2048 * 24, 0);
    memset(&cprm, 0, sizeof(cprm));
    cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
    cprm.buf = syMalloc(cprm.size);
    cprm.sample = MWD_PLY_SAMPLE_NUM;
    ply = mwPlyCreateWav(&cprm);                          /* ハンドルの生成              */
    /*
     * この間で再生するデータを準備する
     */
    mwPlyStartSj(ply, sj);                              /* 再生開始                    */
    pos = 0;
```

```

for (;;) {
    If (pos < size) {
        nroom = SJ_GetNumData(sj, SJ_LIN_FREE);
        SJ_GetChunk(sj, SJ_LIN_FREE, nroom, &ck); /* 空きチャンクを取得 */
        len = MIN(ck.len, (size - pos));
        memcpy(&ck.data, data[pos], ck.len);
        SJ_SplitChunk(&ck, len, &ck, &ck2);
        SJ_PutChunk(sj, SJ_LIN_DATA, &ck); /* データチャンクを渡す */
        SJ_UngetChunk(sj, SJ_LIN_FREE, &ck2); /* 空きチャンクを戻す */
        pos += len;
    }
    mwExecMainServer(); /* ミドルウェアライブラリの実行 */
    stat = mwPlyGetStat(ply); /* ハンドルの状態の取得 */
    if ( stat == MWE_PLY_STAT_PLAYEND ) {
        break;
    }
    njWaitVSync(); /* V-Sync 待 ち
*/
}
mwPlyStop(ply); /* 再生開始 */
mwPlyDestroy(ply); /* ハンドルの消去 */
syFree(buf);
SJ_Destroy(sj);
syFree(cprm.buf);
mwPlyFinishWav(); /* ライブラリの初期化 */
}

```


3.1 1 メモリからの再生

以下に、メモリ上のデータを再生するサンプルプログラムを示す。

< サンプルプログラム >

```
#define NUM_HNDL    (1)                                /* 生成するハンドル数          */

/* アプリケーションメイン関数 */
void main(void)
{
    MWPLY            ply;                                /* ミドルウェア再生ハンドル    */
    MWE_PLY_STAT     stat;                              /* ハンドルの状態              */
    MWS_PLY_CPRM     cprm;                              /* ハンドル生成のパラメータ構造体 */
    Sint8            *data;                             /* データへのポインタ          */
    Sint32           size;                              /* データサイズ                */

    /*
     * 画面やサウンドの初期化
     */
    mwPlyInitWav(MWD_PLY_SVR_VSYNC);                    /* ライブラリの初期化          */
    memset(&cprm, 0, sizeof(cprm));                     /* 予約メンバのゼロ設定のため  */
    cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
    cprm.buf = syMalloc(cprm.size);
    cprm.libwork = NULL;
    cprm.sample = MWD_PLY_SAMPLE_NUM;
    ply = mwPlyCreateWav(&cprm);                        /* ハンドルの生成              */
    /*
     * この間で再生するデータを準備する
     */
    mwPlyStartMem(ply, (void *)data, size);             /* 再生開始                    */
    for (;;) {
        mwExecMainServer();                            /* ミドルウェアライブラリの実行 */
        stat = mwPlyGetStat(ply);                     /* ハンドルの状態の取得        */
        if ( stat == MWE_PLY_STAT_PLAYEND ) {
            break;
        }
        njWaitVSync();                                /* V-Sync 待ち                */
    }
    mwPlyStop(ply);                                    /* 再生開始                    */
    mwPlyDestroy(ply);                                 /* ハンドルの消去              */
    syFree(cprm.buf);
    mwPlyFinishWav();                                  /* ライブラリの初期化          */
}
```

4. データ仕様

ライブラリのデータ一覧を以下に示す。

表 4 - 1 データ一覧

データ名	機 能	番号
定 数		
MWE_PLY_STAT_~	ハンドルの状態	1.1
MWE_PLY_FTYPE_~	再生するファイルのタイプ	1.2
MWE_PLY_DTYPE_~	動画の表示タイプ	1.3
MWD_PLY_COMPO_~	合成モード	1.4
データ型		
MWPLY	ミドルウェア再生ハンドル	2.1
MWS_PLY_INIT_SFD	初期化パラメータの構造体 (MPEG SofdecF/X 用)	2.2
MWS_PLY_CPRM_SFD	ハンドル生成のパラメータ構造体 (MPEG SofdecF/X 用)	2.3
MWS_PLY_CPRM_TM	ハンドル生成のパラメータ構造体 (TrueMotion 用)	2.4
MWS_PLY_CPRM	ハンドル生成のパラメータ構造体 (WAVE, MPEG/Audio, DualSpeech 用)	2.5

4.1 定 数

Title	Data Name	Data	No
データ	MWE_PLY_STAT_~	ハンドルの状態	1.1

以下に示す定数は、ハンドルの状態を示す。

定数名	説 明
MWE_PLY_STAT_STOP	停止中
MWE_PLY_STAT_PREP	準備中
MWE_PLY_STAT_PLAYING	再生中
MWE_PLY_STAT_PLAYEND	再生終了
MWE_PLY_STAT_ERROR	エラー状態

Title	Data Name	Data	No
データ	MWE_PLY_FTYPE_~	再生するファイルのタイプ	1.2

以下に示す定数は、MPEG SofdecF/X で再生するファイルのタイプを示す。

定数名	説 明
MWE_PLY_FTYPE_SFD	MPEG SofdecF/X (音声 + 映像)
MWE_PLY_FTYPE_MPV	MPEG/Video (映像のみ)

Title データ	Data Name MWE_PLY_DTYPE ~	Data 動画の表示タイプ	No 1.3
--------------	------------------------------	------------------	-----------

以下に示す定数は、動画の表示タイプを示します。

定数名	説 明
MWE_PLY_DTYPE_AUTO	画面の表示サイズを自動的に調整 画質を優先させる時はこのモードを使用します。 テクスチャの UV 座標を調整しているため、バイ リニアフィルタを使用してもボケの少ない映像 が再生されます。 このモードを使用した場合は、mwPlySetDispPos, mwPlySetDispSize 関数を使用してはいけない。
MWE_PLY_DTYPE_FULL	全画面に表示されるように調整 Ver.1 との整合性のために残しています。
MWE_PLY_DTYPE_WND	表示位置とサイズをウィンドウレベルで調整 mwPlySetDispPos,mwPlySetDispSize 関数により 設定します。
MWE_PLY_DTYPE_SRF	表示位置とサイズをサーフェスレベルで調整 mwPlySetSrf ~ 関数を使用し、頂点後とに表示位 置や輝度を設定することが可能。

Title データ	Data Name MWD_PLY_COMPO ~	Data 合成モード	No 1.4
--------------	------------------------------	---------------	-----------

以下に示す定数は、合成モードを示します。

定数名	説 明
MWD_PLY_COMPO_OPEQ	不透明
MWD_PLY_COMPO_TRNSP	透明
MWD_PLY_COMPO_ADD	加算合成
MWD_PLY_COMPO_LUMI	ルミネッセンス合成
MWD_PLY_COMPO_ALPH3	3 値アルファ合成
MWD_PLY_COMPO_ALPH5	5 値アルファ合成
MWD_PLY_COMPO_ALPH256	フルアルファ合成
MWD_PLY_COMPO_MIX	混合モード 1 つのストリームに複数の合成モードが混在。

4.2 データ型

Title	Data Name	Data	No
データ	MWPLY	ミドルウェア再生ハンドル	2.1

動画や音声の再生を制御するためのハンドル。

Title	Data Name	Data	No
データ	MWS_PLY_INIT_SFD	初期化パラメータの構造体	2.2

MPEG SofdecF/X 用の初期化関数に設定するパラメータ構造体。Ninja に設定した表示パラメータと同じ値を設定してください。メンバーは以下の通り。

メンバー	型 名	説 明
mode	Sint32	画面モード
frame	Sint32	フレームバッファのカラーモード
count	Sint32	フレームカウント数
latency	Sint32	表示レイテンシ (2V または 3V)

Title データ	Data Name MWS_PLY_CPRM_SFD	Data ハンドル生成のパラメータ構造体	No 2.3
--------------	-------------------------------	-------------------------	-----------

MPEG SofdecF/X 用ミドルウェア再生ハンドルを生成する時に、設定するパラメータの構造体。メンバーは以下の通りです。

メンバー	型 名	説 明
f_type	Sint32	再生するファイルのタイプ MWE_PLY_FTYPE_~ から選択する。
max_bps	Sint32	最大のビットストリーム量（単位：bit/sec） 実際よりも少なくすることも可能。 再生が途中でフリーズする時は、この値を増やしてください。
max_width	Sint32	再生画像サイズの最大幅（単位：ピクセル）
max_height	Sint32	再生画像サイズの最大高さ（単位：ピクセル）
nfrm_pool_wk	Sint32	システム領域のフレームプール数（通常：3） 処理負荷の変動によりフレーム落ちが発生した場合は、この値を増やしてください。
work	Sint8*	使用するバッファへのポインタ
wksize	Sint32	確保したバッファサイズ（単位：バイト）
dtype	Sint32	動画の表示タイプ MWE_PLY_DTYPE_~ から選択する。
compo_mode	Sint32	合成モード MWD_PLY_COMP_~ から選択する。

Title データ	Data Name MWS_PLY_CPRM_TM	Data ハンドル生成のパラメータ構造体	No 2.4
--------------	------------------------------	-------------------------	-----------

TrueMotion を再生する場合、ミドルウェア再生ハンドル生成時に設定するパラメータ構造体。現在使用していないため、設定する項目はない。mwPlyCreateTM 関数の引数には NULL を設定してください。

Title データ	Data Name MWS_PLY_CPRM	Data ハンドル生成のパラメータ構造体	No 2.5
--------------	---------------------------	-------------------------	-----------

音声コーデック（WAVE,MPEG/Audio,DualSpeech）用ミドルウェア再生ハンドルを生成する時に、設定するパラメータ構造体。メンバーは以下の通り。

メンバー	型 名	説 明
buf	Uint8*	使用するバッファへのポインタ MWD_PLY_CALC_AWORK マクロにより計算された大きさのバッファを確保し、このメンバーに設定する。
size	Sint32	使用するバッファのサイズ（単位：バイト） MWD_PLY_CALC_AWORK マクロにより計算されたバッファサイズを設定する。MWD_PLY_MIN_AWORK=48 は、MWD_PLY_CALC_AWORK マクロに設定する最小値で、滞りなく音声を再生するために必要なセクタ数である。
libwork	Sint32*	デコーダが使用する作業領域 WAVE の再生では、このメンバーを使用していないので、NULLを指定する。 MPEG/Audio の再生では、MWD_MPA_CALC_WORK マクロにより計算されたサイズの領域を確保し、このメンバーに設定する。 MWD_MPA_CALC_WORK マクロに指定する引数には、生成するハンドルの個数を指定する。
extsize[2]	Sint32	エキストラバッファサイズ（単位：バイト） 内部で使用しているバッファにエキストラ領域を必要とする場合は、このメンバーにサイズを指定する。 (MPEG/Audio と DualSpeech は指定する必要がある。) 各コーデック毎の指定方法は、関数仕様・ハンドルの生成の項を参照のこと。
sample	Sint32	オーディオレンダラが再生を開始する為の再生サンプル数 標準では MWD_PLY_SAMPLE_NUM=8192 を指定する。 DualSpeech では MWD_DLSD_SAMPLE_NUM=1 を指定する。
limit	Sint32	メモリ再生において、自動停止する為の限界値 標準では MWD_PLY_MEMORY_LIMIT=0 を指定する。 DualSpeech では MWD_DLSD_MEMORY_LIMIT=1 を指定する。

5. 関数仕様

ライブラリの関数一覧を以下に示す。

表 5 - 1 関数一覧

関数名	機 能	番号
MPEG SofdecF/X 用関数		
mwPlyPreInitSofdec	システム初期化の設定	1.1
mwPlyInitSofdec	MPEG SofdecF/X ライブラリの初期化	1.2
mwPlyFinishSofdec	MPEG SofdecF/X ライブラリの終了処理	1.3
mwPlyCreateSofdec	MPEG SofdecF/X 用ハンドルの生成	1.4
mwPlyCalcWorkCprmSfd	作業領域サイズの計算	1.5
mwPlySetDispPos	表示位置の設定	1.6
mwPlySetDispSize	表示サイズの設定	1.7
mwPlySetBright	輝度の設定	1.8
mwPlyGetBright	輝度の取得	1.9
mwPlySetBrightOfst	輝度オフセットの設定	1.10
mwPlyGetBrightOfst	輝度オフセットの取得	1.11
mwPlySetDispZ	表示スクリーンの奥行き値の設定	1.12
mwPlyGetDispZ	表示スクリーンの奥行き値の取得	1.13
mwPlySetDispMode	表示モードの設定	1.14
mwPlySetFastHalfpel	高速ハーフペル処理の設定	1.15
mwPlySetAudioSw	音声出力スイッチの設定	1.16
mwPlySetVideoSw	映像表示スイッチの設定	1.17
mwPlyGetNumDropFrm	コマ落ちしたフレーム数の取得	1.18
WAVE 用関数		
mwPlyInitWav	WAVE 再生ライブラリの初期化	2.1
mwPlyFinishWav	WAVE 再生ライブラリの終了処理	2.2
mwPlyCreateWav	WAVE 再生用ハンドルの生成	2.3
TrueMotion 用関数		
mwPlyInitTM	TrueMotion ライブラリの初期化	3.1
mwPlyFinishTM	TrueMotion ライブラリの終了処理	3.2
mwPlyCreateTM	TrueMotion 用ハンドルの生成	3.3
MPEG/Audio 用関数		
mwPlyInitMpa	MPEG/Audio ライブラリの初期化	4.1
mwPlyFinishMpa	MPEG/Audio ライブラリの終了処理	4.2
mwPlyCreateMpa	MPEG/Audio 用ハンドルの生成	4.3
DualSpeech 用関数		
mwPlyInitDIsd	DualSpeech ライブラリの初期化	5.1
mwPlyFinishDIsd	DualSpeech ライブラリの終了処理	5.2
mwPlyCreateDIsd	DualSpeech 用ハンドルの生成	5.3

表 5 - 2 関数一覧

関数名	機 能	番号
共通関数		
mwPlyDestroy	ハンドルの消去	6.1
mwPlyStartFname	再生開始(GD-ROM からの再生)	6.2
mwPlyStartSj	再生開始(ストリームジョイントからの再生)	6.3
mwPlyStartMem	再生開始(メモリからの再生)	6.4
mwPlyStop	再生停止	6.5
mwPlyGetStat	ハンドルの状態の取得	6.6
mwPlyGetTime	再生サンプル数の取得	6.7
mwPlyPause	ポーズの設定	6.8
mwPlySetOutVol	ボリュームの設定	6.9
mwPlyGetOutVol	ボリュームの取得	6.10
mwPlySetOutPan	パンポットの設定	6.11
mwPlyGetOutPan	パンポットの取得	6.12
mwPlyEntryErrFunc	エラー発生時に呼ばれる関数の登録	6.13
mwExecMainServer	サーバ関数	6.14

5.1 MPEG SofdecF/X 用関数

MPEG SofdecF/X の再生に必要な関数である。

Title 関 数	Function Name	Function	No
	mwPlyPreInitSofdec	システム初期化の設定	1.1

[書 式] void mwPlyPreInitSofdec(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] 割込みスタックの設定をする。割込みスタックサイズは 16Kbyte となる。

Title 関 数	Function Name	Function	No
	mwPlyInitSofdec	MPEG SofdecF/X ライブラリの初期化	1.2

[書 式] void mwPlyInitSofdec(MWS_PLY_INIT_SFD *iprm);

[入 力] iprm: 初期化パラメータ

[出 力] なし

[関数値] なし

[機 能] ライブラリを初期化する。

各種サーバ関数を設定する。サーバ関数には以下の種類のものがある。

(1) メインサーバ

メイン処理内で、mwExecMainServer が呼び出された時に実行される。

(2) V-Sync サーバ

V-Sync 割り込み時に実行される。

(3) アイドルサーバ

njWaitVsync 内で、次のゲームフレームまでの待ち時間に実行される。

MPEG SofdecF/X ライブラリは、このサーバ内でデコードを実行する。

[備 考] Ninja に、実際に設定した表示モードと同じ値を初期化パラメータに設定する。

[例] 使用例を以下に示す。

```
sbInitSystem(SYS_MODE, SYS_FRAME, SYS_COUNT);  
njInitVertexBuffer(VB_OP, VB_OM, VB_TP, VB_TM, VB_PT);  
iprm.mode= SYS_MODE;  
iprm.frame= SYS_FRAME;  
iprm.count= SYS_COUNT;  
iprm.latency= MWD_PLY_LATENCY(VB_OP, VB_OM, VB_TP, VB_TM, VB_PT);  
mwPlyInitSofdec(&iprm);
```

Title 関 数	Function Name	Function	No
	mwPlyFinishSofdec	MPEG SofdecF/X ライブラリの終了処理	1.3

[書 式] void mwPlyFinishSofdec(void);
[入 力] なし
[出 力] なし
[関数値] なし
[機 能] ライブラリの終了処理をする。

Title 関 数	Function Name	Function	No
	mwPlyCreateSofdec	MPEG SofdecF/X 用ハンドルの生成	1.4

[書 式] MWPLY mwPlyCreateSofdec(MWS_PLY_CPRM_SFD *cprm);
[入 力] cprm : ハンドルの生成パラメータ
[出 力] なし
[関数値] ミドルウェア再生ハンドル
[機 能] ハンドルを生成する。
[例] 使用例を以下に示します。

```

memset(&cprm, 0, sizeof(cprm));          /* 予約メンバのゼロ設定のため */
cprm.compo_mode = MWD_PLY_COMPO_OPEQ;    /* 合成モード */
cprm.ftype= MWD_PLY_FTYPE_SFD;           /* 再生ファイルタイプ */
cprm.dtype= MWD_PLY_DTYPE_AUTO;          /* 画質優先 */
cprm.max_bps = 450*1024*8;                /* ビットレート : 450 Kbyte/sec */
cprm.max_width = 320;                     /* 画像サイズ : 320x480 */
cprm.max_height = 480;
cprm.nfrm_pool_wk = 3;                    /* フレームバッファの枚数 */
cprm.wksize = mwPlyCalcWorkCprmSfd(cprm); /* 作業領域サイズの計算 */
cprm.work = syMalloc(cprm.wksize);        /* 作業領域の確保 */
ply = mwPlyCreateSofdec(&cprm);

```

Title 関 数	Function Name mwPlyCalcWorkCprmSfd	Function 作業領域サイズの計算	No 1.5
--------------	---------------------------------------	------------------------	-----------

[書 式] Sint32 mwPlyCalcWorkCprmSfd(MWS_PLY_CPRM_SFD *cprm);
[入 力] cprm : ハンドルの生成パラメータ
[出 力] なし
[関数値] 作業領域サイズ (単位: バイト)
[機 能] MPEG SofdecF/X の再生に使用する作業領域サイズを計算します。

Title 関 数	Function Name mwPlySetDispPos	Function 表示位置の設定	No 1.6
--------------	----------------------------------	---------------------	-----------

[書 式] void mwPlySetDispPos(MWPLY ply, float lx, float ly);
[入 力] ply : ミドルウェア再生ハンドル
lx : X座標
ly : Y座標
[出 力] なし
[関数値] なし
[機 能] 表示位置を設定する。

Title 関 数	Function Name	Function	No
	mwPlySetDispSize	表示サイズの設定	1.7

[書 式] void mwPlySetDispSize(MWPLY ply, float sx, float sy);

[入 力] ply: ミドルウェア再生ハンドル

sx : X座標

sy : Y座標

[出 力] なし

[関数値] なし

[機 能] 表示サイズを設定する。

Title 関 数	Function Name	Function	No
	mwPlySetBright	輝度の設定	1.8

[書 式] void mwPlySetBright(MWPLY ply, Sint32 val);

[入 力] ply: ミドルウェア再生ハンドル

val: 輝度 (0 ~ 255)

[出 力] なし

[関数値] なし

[機 能] 輝度を設定する。デフォルトでは、224 が設定されている。

Dreamcast の映像出力は、255 で 110IRE の輝度となるため、デフォルトで 224 を設定している。素材によってはこの値を調整する必要がある。

Title 関 数	Function Name mwPlyGetBright	Function 輝度の取得	No 1.9
--------------	---------------------------------	-------------------	-----------

[書 式] Sint32 mwPlyGetBright(MWPLY ply);

[入 力] ply: ミドルウェア再生ハンドル

[出 力] なし

[関数値] 輝度

[機 能] 輝度を取得する。

Title 関 数	Function Name mwPlySetBrightOfst	Function 輝度オフセットの設定	No 1.10
--------------	-------------------------------------	------------------------	------------

[書 式] void mwPlySetBrightOfst(MWPLY ply, Sint32 val);

[入 力] ply: ミドルウェア再生ハンドル

val: 輝度オフセット (0~255)

[出 力] なし

[関数値] なし

[機 能] 輝度オフセットを設定する。デフォルトでは、6 が設定されている。

CG ムービーなどでは、黒がつぶれる傾向があるため、デフォルトで 6 が設定されている。
素材によっては、この値を調整する必要がある。

この値を 255 から徐々に減少させることにより、フェードインすることができる。

また、反対に 255 まで徐々に増加させることにより、フェードアウトすることができる。

Title 関 数	Function Name mwPlyGetBrightOfst	Function 輝度オフセットの取得	No 1.11
--------------	-------------------------------------	------------------------	------------

[書 式] Sint32 mwPlyGetBrightOfst(MWPLY ply);

[入 力] ply: ミドルウェア再生ハンドル

[出 力] なし

[関数値] 輝度オフセット

[機 能] 輝度オフセットを取得する。

Title 関 数	Function Name mwPlySetDispZ	Function 表示スクリーンの奥行き値の設定	No 1.12
--------------	--------------------------------	-----------------------------	------------

[書 式] void mwPlySetDispZ(MWPLY ply, float z);

[入 力] ply: ミドルウェア再生ハンドル

z : 奥行き値 1.0(最手前) ~ 65536.0(最奥)

[出 力] なし

[関数値] なし

[機 能] 表示スクリーンの奥行き値を設定する。

Title 関 数	Function Name	Function	No
	mwPlyGetDispZ	表示スクリーンの奥行き値の取得	1.13

[書 式] float mwPlyGetDispZ(MWPLY ply);
[入 力] ply: ミドルウェア再生ハンドル
[出 力] なし
[関数値] 奥行き値 1.0(最手前) ~ 65536.0(最奥)
[機 能] 設定されている表示スクリーンの奥行き値を取得する。

Title 関 数	Function Name	Function	No
	mwPlySetDispMode	表示モードの設定	1.14

[書 式] void mwPlySetDispMode(Sint32 mode, Sint32 frame, Sint32 count, Sint32 latency);
[入 力] mode : 画面モード
frame : フレームバッファのカラーモード
count : フレームカウント数
latency : 表示レイテンシ
[出 力] なし
[関数値] なし
[機 能] 表示モードをミドルウェアに設定する。
[注 意] Ninja に、実際に設定した表示モードと同じ値を設定してください。
初期化時に MWS_PLY_INIT_SFD で設定した場合は、この関数は必要ない。
表示モードを変更した場合は、この関数で再設定する必要がある。
[備 考] (1) フレームカウント数について
インタレースの場合、表示更新 VSYNC 数は 2×count となります。例えば、インタレースでフレームカウント数 2 なら、4V 表示となります。
1V 表示は、24fps など、任意のフレームレートの動画をスムーズに表示します。
1V 表示は、29.97fps の NTSC 用ムービーを PAL で再生する時に、スムーズに表示します。
2V 表示は、より多くの CPU 時間を MPEG SofdecF/X に与え、コマ落ちの危険を軽減しますが、スムーズに再生できるフレームレートが限定されます。
(NTSC:29.97fps, VGA:30fps, PAL:25fps)
3V 以上の表示は、動画再生には推奨しません。
(2) レイテンシについて
320x240 を再生する場合、2V レイテンシの時で 500Kbyte、3V の時で 750Kbyte のテクスチャ領域が使用されます。

Title 関 数	Function Name	Function	No
	mwPlySetFastHalfpel	高速ハーフペル処理の設定	1.15

[書 式] void mwPlySetFastHalfpel(MWPLY ply, Sint32 sw);

[入 力] ply : ミドルウェア再生ハンドル
 sw : 高速ハーフペルスイッチ (0 : 高速処理しない、1 : 高速処理する)

[出 力] なし

[関数値] なし

[機 能] 高速ハーフペルの処理を設定する。

[備 考] 高速ハーフペルに設定した場合、CPU 負荷は軽減するが、多少画質が劣化する。

Title 関 数	Function Name	Function	No
	mwPlySetAudioSw	音声出力スイッチの設定	1.16

[書 式] void mwPlySetAudioSw(MWPLY ply, Sint32 sw);

[入 力] ply : ミドルウェア再生ハンドル
 sw : 音声出力スイッチ (0 : 音声出力しない、1 : 音声出力する)

[出 力] なし

[関数値] なし

[機 能] 音声出力スイッチを設定する。

[備 考] デフォルトでは、音声を出力する。
 音声付き動画ファイルでも映像のみを再生することが可能。

Title 関 数	Function Name mwPlySetVideoSw	Function 映像表示スイッチの設定	No 1.17
--------------	----------------------------------	-------------------------	------------

[書 式] void mwPlySetVideoSw(MWPLY ply, Sint32 sw);

[入 力] ply : ミドルウェア再生ハンドル

sw : 映像表示スイッチ (0: 映像を表示しない、1: 映像を表示する)

[出 力] なし

[関数値] なし

[機 能] 映像表示のスイッチを設定する。

Title 関 数	Function Name mwPlyGetNumDropFrm	Function コマ落ちしたフレーム数の取得	No 1.18
--------------	-------------------------------------	----------------------------	------------

[書 式] Sint32 mwPlyGetNumDropFrm(MWPLY ply);

[入 力] ply : ミドルウェア再生ハンドル

[出 力] なし

[関数値] フレーム数

[機 能] コマ落ちしたフレーム数を取得する。

表示モードがインターレースで、29.97, 30fps の動画の時のみ有効である。

5.2 WAVE 用関数

WAVE の再生に必要な関数である。

Title 関 数	Function Name	Function	No
	mwPlyInitWav	WAVE 再生ライブラリの初期化	2.1

[書 式] void mwPlyInitWav(Sint32 mode);

[入 力] mode : サーバ関数呼び出しモード

MWD_PLY_SVR_VSYNC : V-Sync 割込み内で、サーバ関数を実行する。

MWD_PLY_SVR_MAIN : アプリケーション側でサーバ関数を呼び出して、サーバ処理を実行する。

[出 力] なし

[関数値] なし

[機 能] ライブラリを初期化する。

Title 関 数	Function Name	Function	No
	mwPlyFinishWav	WAVE 再生ライブラリの終了処理	2.2

[書 式] void mwPlyFinishWav(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリの終了処理をする。

Title 関 数	Function Name mwPlyCreateWav	Function WAVE 再生用ハンドルの生成	No 2.3
--------------	---------------------------------	-----------------------------	-----------

[書 式] MWPLY mwPlyCreateWav(MWS_PLY_CPRM *cprm);

[入 力] cprm : ハンドル生成のパラメータ

[出 力] なし

[関数値] ミドルウェア再生ハンドル

[機 能] ハンドルを生成する。

cprm.size には、MWD_PLY_CALC_AWORK マクロにより計算された値を設定する。

MWD_PLY_CALC_AWORK マクロの引数にはセクタ単位の値を設定する。MWD_PLY_MIN_AWORK=48 は MWD_PLY_CALC_AWORK マクロに設定する最小値で、滞りなく音声を再生するために必要なセクタ数である。

cprm.buf には、cprm.size 分の領域を確保して設定する。

WAVE 再生では、バッファ領域以外の作業領域を設定する必要がないので、cprm.libwork には NULL を指定してください。

cprm.extsize には、MWD_PLY_EXTRA_SIZE を指定する。

cpam.sample には、MWD_PLY_SAMPLE_NUM を指定する。

cpam.limit には、MWD_PLY_MEMORY_LIMIT を指定する。

[例] 使用例を以下に示す。

```
cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
```

```
cprm.buf = syMalloc(cprm.size);
```

```
cprm.libwork = NULL;
```

```
cprm.extsize[MWD_CH_L] = MWD_PLY_EXTRA_SIZE;
```

```
cprm.extsize[MWD_CH_R] = MWD_PLY_EXTRA_SIZE;
```

```
cprm.sample = MWD_PLY_SAMPLE_NUM;
```

```
cprm.limit = MWD_PLY_MEMORY_LIMIT;
```

```
ply = mwPlyCreateWav(&cprm);
```

5.3 TrueMotion 用関数

TrueMotion の再生に必要な関数である。

Title	Function Name	Function	No
関 数	mwPlyInitTM	TrueMotion ライブラリの初期化	3.1

[書 式] void mwPlyInitTM(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリを初期化する。

Title	Function Name	Function	No
関 数	mwPlyFinishTM	TrueMotion ライブラリの終了処理	3.2

[書 式] void mwPlyFinishTM(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリの終了処理をする。

Title 関 数	Function Name mwPlyCreateTM	Function TrueMotion 用ハンドルの生成	No 3.3
--------------	--------------------------------	---------------------------------	-----------

[書 式] MWPLY mwPlyCreateTM(MWS_PLY_CPRM_TM *cprm);

[入 力] cprm : ハンドル生成のパラメータ

[出 力] なし

[関数値] ミドルウェア再生ハンドル

[機 能] ハンドルを生成する。

5.4 MPEG/Audio 用関数

MPEG/Audio の再生に必要な関数である。

Title 関 数	Function Name	Function	No
	mwPlyInitMpa	MPEG/Audio ライブラリの初期化	4.1

[書 式] void mwPlyInitMpa(Sint32 mode);

[入 力] mode : サーバ関数呼び出しモード

MWD_PLY_SVR_VSYNC : V-Sync 割込み内で、サーバ関数を実行する。

MWD_PLY_SVR_MAIN : アプリケーション側でサーバ関数を呼び出して、サーバ処理を実行する。

[出 力] なし

[関数値] なし

[機 能] ライブラリを初期化する。

Title 関 数	Function Name	Function	No
	mwPlyFinishMpa	MPEG/Audio ライブラリの終了処理	4.2

[書 式] void mwPlyFinishMpa(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリの終了処理をする。

Title 関 数	Function Name mwPlyCreateMpa	Function MPEG/Audio 用ハンドルの生成	No 4.3
--------------	---------------------------------	---------------------------------	-----------

[書 式] MWPLY mwPlyCreateMpa(MWS_PLY_CPRM *cprm);

[入 力] cprm : ハンドル生成のパラメータ

[出 力] なし

[関数値] ミドルウェア再生ハンドル

[機 能] ハンドルを生成する。

cprm.size には、MWD_PLY_CALC_AWORK マクロにより計算された値を設定する。
MWD_PLY_CALC_AWORK マクロの引数にはセクタ単位の値を設定する。MWD_PLY_MIN_AWORK=48
は MWD_PLY_CALC_AWORK マクロに設定する最小値で、滞りなく音声を再生するために必要
なセクタ数である。

cprm.buf には、cprm.size 分の領域を確保して設定する。

cprm.libwork には、MWD_MPA_CALC_WORK マクロにより計算された値を設定する。

MWD_MPA_CALC_WORK マクロの引数には、生成するハンドルの個数を指定する。

cprm.extsize には、MWD_MPA_EXTRA_SIZE を指定する。

cpam.sample には、MWD_PLY_SAMPLE_NUM を指定する。

cpam.limit には、MWD_PLY_MEMORY_LIMIT を指定する。

[例] 使用例を以下に示す。

```
cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
cprm.buf = syMalloc(cprm.size));
cprm.libwork = syMalloc(MWD_MPA_CALC_WORK(1));
cprm.extsize[MWD_CH_L] = MWD_MPA_EXTRA_SIZE;
cprm.extsize[MWD_CH_R] = MWD_MPA_EXTRA_SIZE;
cprm.sample = MWD_PLY_SAMPLE_NUM;
cprm.limit = MWD_PLY_MEMORY_LIMIT;
ply = mwPlyCreateMpa(&cprm);
```


5.5 DualSpeech 用関数

DualSpeech の再生に必要な関数である。

Title 関 数	Function Name	Function	No
	mwPlyInitDlsd	DualSpeech ライブラリの初期化	5.1

[書 式] void mwPlyInitDlsd(Sint32 mode);

[入 力] mode : サーバ関数呼び出しモード

MWD_PLY_SVR_VSYNC : V-Sync 割込み内で、サーバ関数を実行する。

MWD_PLY_SVR_MAIN : アプリケーション側でサーバ関数を呼び出して、サーバ処理を実行する。

[出 力] なし

[関数値] なし

[機 能] ライブラリを初期化する。

Title 関 数	Function Name	Function	No
	mwPlyFinishDlsd	DualSpeech ライブラリの終了処理	5.2

[書 式] void mwPlyFinishDlsd(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ライブラリの終了処理をする。

Title 関 数	Function Name mwPlyCreateDlsd	Function DualSpeech 用ハンドルの生成	No 5.3
--------------	----------------------------------	---------------------------------	-----------

[書 式] MWPLY mwPlyCreateDlsd(MWS_PLY_CPRM *cprm);

[入 力] cprm : ハンドル生成のパラメータ

[出 力] なし

[関数値] ミドルウェア再生ハンドル

[機 能] ハンドルを生成する。

cprm.size には、MWD_PLY_CALC_AWORK マクロにより計算された値を設定する。

MWD_PLY_CALC_AWORK マクロの引数にはセクタ単位の値を設定する。MWD_PLY_MIN_AWORK=48 は MWD_PLY_CALC_AWORK マクロに設定する最小値で、滞りなく音声を再生するために必要なセクタ数である。

cprm.buf には、cprm.size 分の領域を確保して設定する。

cprm.libwork には、MWD_DLSD_CALC_WORK マクロにより計算された値を設定する。

MWD_DLSD_CALC_WORK マクロの引数には、生成するハンドルの個数を指定する。

cprm.extsize には、左チャンネル (MWD_CH_L) のみ MWD_DLSD_EXTRA_SIZE を指定する。

cpam.sample には、MWD_DLSD_SAMPLE_NUM を指定する。

cpam.limit には、MWD_DLSD_MEMORY_LIMIT を指定する。

[例] 使用例を以下に示す。

```
cprm.size = MWD_PLY_CALC_AWORK(MWD_PLY_MIN_AWORK);
```

```
cprm.buf = syMalloc(cprm.size));
```

```
cprm.libwork = syMalloc(MWD_DLSD_CALC_WORK(1));
```

```
cprm.extsize[MWD_CH_L] = MWD_DLSD_EXTRA_SIZE;
```

```
cprm.extsize[MWD_CH_R] = MWD_PLY_EXTRA_SIZE;
```

```
cprm.sample = MWD_DLSD_SAMPLE_NUM;
```

```
cprm.limit = MWD_DLSD_MEMORY_LIMIT;
```

```
ply = mwPlyCreateDlsd(&cprm);
```

5.6 共通関数

コーデックの種類に関わらず共通な関数群である。

Title	Function Name	Function	No
関 数	mwPlyDestroy	ハンドルの消去	6.1

[書 式] void mwPlyDestroy(MWPLY ply);
[入 力] Ply : ミドルウェア再生ハンドル
[出 力] なし
[関数値] なし
[機 能] ハンドルを消去する。

Title	Function Name	Function	No
関 数	mwPlyStartFname	再生開始 (GD-ROM からの再生)	6.2

[書 式] void mwPlyStartFname(MWPLY ply, Sint8 *fname);
[入 力] ply : ミドルウェア再生ハンドル
 fname : 再生する動画や音声のファイル名
[出 力] なし
[関数値] なし
[機 能] GD-ROM にある動画や音声の再生を開始する。

Title 関 数	Function Name	Function	No
	mwPlyStartSj	再生開始(ストリームジョイントからの再生)	6.3

[書 式] void mwPlyStartSj(MWPLY ply, SJ sj);
 [入 力] ply : ミドルウェア再生ハンドル
 sj : ストリームジョイントハンドル
 [出 力] なし
 [関数値] なし
 [機 能] ストリームジョイントに供給されている動画や音声の再生を開始する。

Ver.2.24 の TrueMotion では、この関数は実装されていない。

Title 関 数	Function Name	Function	No
	mwPlyStartMem	再生開始(メモリからの再生)	6.4

[書 式] void mwPlyStartMem(MWPLY ply, void *addr, Sint32 len);
 [入 力] ply : ミドルウェア再生ハンドル
 addr : 動画や音声データが展開されているデータへのポインタ
 len : データの長さ
 [出 力] なし
 [関数値] なし
 [機 能] メモリ上に展開されている動画や音声の再生を開始する。

Ver.2.24 の TrueMotion では、この関数は実装されていない。

Title 関 数	Function Name mwPlyStop	Function 再生停止	No 6.5
--------------	----------------------------	------------------	-----------

[書 式] void mwPlyStop(MWPLY ply);
[入 力] ply : ミドルウェア再生ハンドル
[出 力] なし
[関数値] なし
[機 能] 動画や音声の再生を停止する。

Title 関 数	Function Name mwPlyGetStat	Function ハンドルの状態の取得	No 6.6
--------------	-------------------------------	------------------------	-----------

[書 式] MWE_PLY_STAT mwPlyGetStat(MWPLY ply);
[入 力] ply : ミドルウェア再生ハンドル
[出 力] なし
[関数値] ハンドルの内部状態
[機 能] ミドルウェア再生ハンドルの内部状態を取得する。

定数名	説 明
MWE_PLY_STAT_STOP	停止中
MWE_PLY_STAT_PREP	準備中
MWE_PLY_STAT_PLAYING	再生中
MWE_PLY_STAT_PLAYEND	再生終了
MWE_PLY_STAT_ERROR	エラー状態

Title 関 数	Function Name mwPlyGetTime	Function 再生サンプル数の取得	No 6.7
--------------	-------------------------------	------------------------	-----------

[書 式] void mwPlyGetTime(MWPLY ply, Sint32 *ncount, Sint32 *tscale);

[入 力] ply : ミドルウェア再生ハンドル

[出 力] ncount : 再生サンプル数 (時刻)

tscale : サンプル周波数 (時刻の単位) (単位 : ヘルツ)

[関数値] なし

[機 能] サンプル単位で再生時刻を取得する。

[備 考]

(a) 実時間(runtime)は、以下の式で計算される。

$$runtime = ncount / tscale;$$

Title 関 数	Function Name mwPlyPause	Function ポーズの設定	No 6.8
--------------	-----------------------------	--------------------	-----------

[書 式] void mwPlyPause (MWPLY ply, Sint32 sw);

[入 力] ply : ミドルウェア再生ハンドル

sw : ポーズスイッチ (0 : ポーズ解除、1 : ポーズ)

[出 力] なし

[関数値] なし

[機 能] ポーズの切り換えを行う。

ポーズ中に、もう一度ポーズする (1を設定する) とコマ送りする。

Title 関 数	Function Name mwPlySetOutVol	Function ボリュームの設定	No 6.9
--------------	---------------------------------	----------------------	-----------

[書 式] void mwPlySetOutVol (MWPLY ply, Sint32 vol);

[入 力] ply : ミドルウェア再生ハンドル

vol : ボリューム (-999 ~ 0) (単位 : 1/10dB)

[出 力] なし

[関数値] なし

[機 能] ボリュームを設定する。デフォルトでは0が設定されている。

この値を-999~0まで増加させることにより、フェードインすることができる。

また、反対に0~-999まで減少させることにより、フェードアウトすることができる。

Title 関 数	Function Name mwPlyGetOutVol	Function ボリュームの取得	No 6.10
--------------	---------------------------------	----------------------	------------

[書 式] Sint32 mwPlyGetOutVol (MWPLY ply);

[入 力] ply : ミドルウェア再生ハンドル

[出 力] なし

[関数値] ボリューム (単位 : 1/10dB)

[機 能] 現在設定されているボリューム値を取得する。

Title 関 数	Function Name	Function	No
	mwPlySetOutPan	パンポットの設定	6.11

[書 式] void mwPlySetOutPan (MWPLY ply, Sint32 chno, Sint32 pan);

[入 力] ply : ミドルウェア再生ハンドル
chno : 設定するチャンネル
モノラル/ステレオ: 左 = MWD_CH_L(0)
ステレオ: 右 = MWD_CH_R(1)
pan : パンポット (-15 ~ 15, -128)
MWD_PAN_LEFT=-15, MWD_PAN_RIGHT=15
MWD_PAN_CENTER=0, MWD_PAN_AUTO=-128

[出 力] なし

[関数値] なし

[機 能] 出力チャンネル毎のパンポットを設定する。
デフォルトでは MWD_PAN_AUTO となり、モノラルの場合は MWD_PAN_CENTER 、
ステレオの場合は左が MWD_PAN_LEFT 、右が MWD_PAN_RIGHT に設定されている。

Title 関 数	Function Name	Function	No
	mwPlyGetOutPan	パンポットの取得	6.12

[書 式] Sint32 mwPlyGetOutPan (MWPLY ply, Sint32 chno);

[入 力] ply : ミドルウェア再生ハンドル
chno : 取得するチャンネル
モノラル/ステレオ: 左 = MWD_CH_L(0)
ステレオ: 右 = MWD_CH_R(1)

[出 力] なし

[関数値] チャンネルに対応したパンポット値 (-15 ~ 15)

[機 能] 現在設定されているパンポット値を取得する。

Title 関 数	Function Name mwPlyEntryErrFunc	Function エラー発生時に呼ばれる関数の登録	No 6.15
--------------	------------------------------------	------------------------------	------------

[書 式] void mwPlyEntryErrFunc(MW_PLY_ERRFN errfn, void *obj);

[入 力] errfn :エラー関数
obj :エラーオブジェクト

[出 力]

[関数値] なし

[機 能] エラー発生時に呼ばれる関数を登録する。

[備 考] エラー関数を登録すると、エラーが発生と共に、登録した関数が呼び出される。
エラー関数は、第2引数に文字列が渡される。この文字列によりエラーの内容が確認できる。

[例] 使用例を以下に示す。

```
/* ユーザエラー関数 */
void user_error(void *user_obj, char *msg)
{
    for (;;) {
        njPrintC(NJM_LOCATION(3, 3), msg);
    }
}

void main(void)
{
    mwPlyEntryErrFunc(user_error, user_obj);
}
```

Title 関 数	Function Name mwExecMainServer	Function サーバ関数	No 6.13
--------------	-----------------------------------	-------------------	------------

[書 式] void mwExecMainServer(void);

[入 力] なし

[出 力] なし

[関数値] なし

[機 能] ハンドルの内部状態を更新する。

6. 付録

6.1 ドアオープンチェック

以下に、ドアオープンチェックのサンプルプログラムを示す。

< サンプルプログラム >

```
/* GD ファイルシステムのエラー発生時に起動する関数 */
void UsrGdErrFunc(void *obj, Sint32 errcode)
{
    if (errcode == GDD_ERR_TRAYOPEN || errcode == GDD_ERR_UNITATTENT) {
        /* ミドルウェアの終了処理 */
        mwPlyFinishSofdec();
        mwPlyFinishTM();
        mwPlyFinishMpa();
        mwPlyFinishWav();

        sbExitSystem(); /* 忍ライブラリの終了処理 */
        syBtExit();     /* シンプルプレイヤーへジャンプ */
    }
}

/* ユーザが V-SYNC 割り込みに登録する関数 */
void UsrVsyncFunc(void)
{
    Sint32 dstat;

    /* ドアオープンチェック */
    dstat = gdFsGetDrvStat();
    if (dstat == GDD_DRVSTAT_OPEN || dstat == GDD_DRVSTAT_BUSY) {
        gdFsReqDrvStat();
    }
}

/* アプリケーションメイン */
void main(void)
{
    :
    :
    /* GD ファイルシステムエラーコールバック関数の登録 */
    gdFsEntryErrFuncAll((void *)UsrGdErrFunc, NULL);
    njSetVSyncFunction(UsrVsyncFunc);
    :
    :
}
```

6.2 MPEG SofdecF/X と ADX との並行再生

ADX 再生ライブラリを使用することにより、MPEG SofdecF/X と ADX との並行再生が実現できる。

BGM を CD ストリーム再生しながらムービーを非同期に再生することにより、ゲームシーンからムービーシーンへのシームレスな移り変わりを実現できる。しかしながら、音声と並行再生するときに ADX ハンドルの入力バッファ容量を増やさなければならない。同様に、ムービー用のバッファ容量を増やす必要がある。MPEG SofdecF/X 再生用ハンドルを生成する時に、最大ビットストリーム量を実際のストリーム量よりも大きい値を設定してください。目安としては、約 1.5 倍の数値を指定してください。これは、シークタイムを約 1 秒として見込んでいるので、データの配置の仕方によっては、減らせる可能性がある。また、ムービーが途切れる場合は、この値を増やしてください。

```
/* 44KHz ステレオを 2 ストリーム並行再生する場合の作業領域サイズ */
/* 2 ストリームは、BGM とムービーという意味 */
#define WKSIZ44S ADXT_CALC_WORK(2, ADXT_PLY_STM, 2, 44100)

char    wk44[WKSIZ44S];          /* 作業領域 */
MWPLY   ply;                     /* MWPLY ハンドル */
ADXT     adxt_bgm;               /* ADXT ハンドル */

/* アプリケーション */
:
:
adxt_bgm = ADXT_Create(2, wk44, WKSIZ44S); /* ADXT ハンドルの生成 */
ADXT_SetReloadSct(adxt_bgm, 25);          /* シーク音の軽減のため */

/* MWPLY ハンドルの生成 */
memset(&cprm, 0, sizeof(cprm));           /* 予約メンバのゼロ設定のために必要 */
cprm.ftype      = MWD_PLY_FTYPE_MPV;
cprm.dtype      = MWD_PLY_DTYPE_AUTO;
cprm.max_bps    = 900*1024*8;             /* 通常 600Kbyte/sec の 1.5 倍を指定 */
cprm.max_width  = 320;
cprm.max_height = 240;
cprm.nfrm_pool_wk = 3;
cprm.wksize     = mwPlyCalcWorkCprmSfd(cprm);
cprm.work       = syMalloc(cprm.wksize);
ply = mwPlyCreateSofdec(&cprm);

ADXT_StartFname(adxt_bgm, "BGM.ADX");     /* 音声再生開始 */

if ( /* イベント発生 */ )
    mwPlyStartFname(ply, "SCENE01.M1V");  /* ムービー再生 */
:
:
```

6.3 システム構成

ミドルウェア再生ライブラリのシステム構成は以下の通りです。

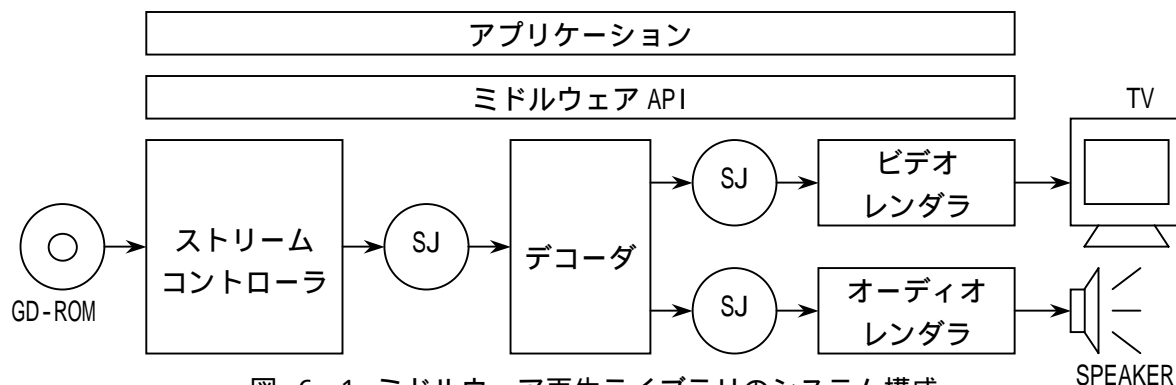


図 6-1 ミドルウェア再生ライブラリのシステム構成

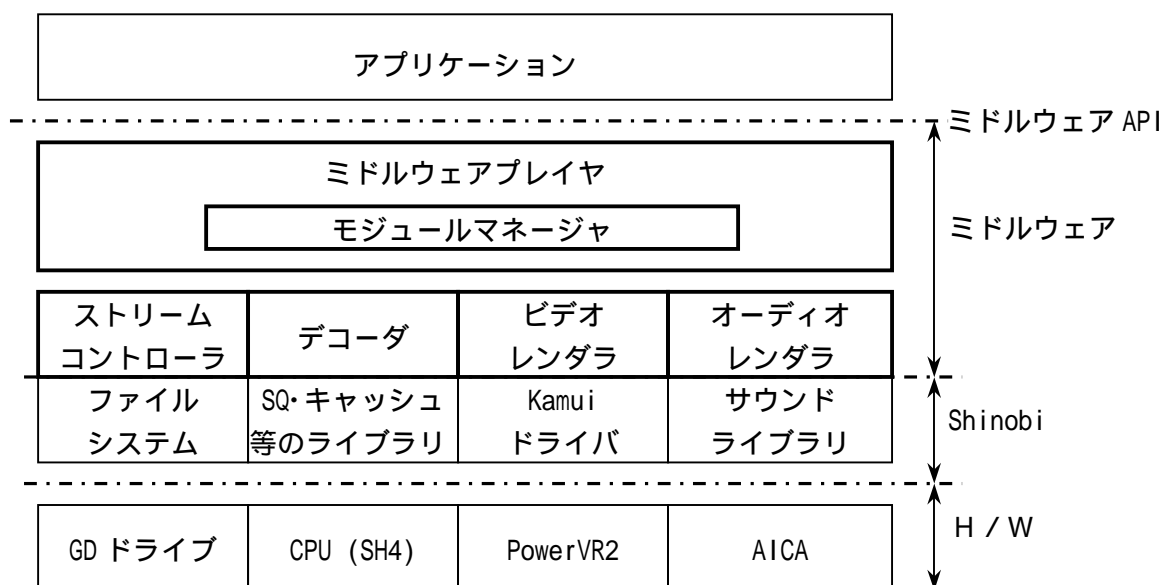
6.4 モジュール構成

ミドルウェア再生ライブラリは、以下のモジュールから構成されています。

表 6-1 ミドルウェア再生ライブラリの内部モジュール

モジュール	略称	説明
ミドルウェアプレイヤー	MWPLY	データの読み込みから出力までを管理し、映像や音声の再生を行います。
モジュールマネージャ	MWMNG	デコーダのCPU タイムを割り当てます。
ストリームコントローラ	MWGSC	映像や音声のデータを読み込みます。
デコーダ		圧縮されたデータを展開します。
ビデオレンダラ	MWRNV	展開された映像データを出力します。
オーディオレンダラ	MWRNA	展開された音声データを出力します。

以下に、ミドルウェア再生ライブラリの内部構成を示します。



SQ : ストアキュー

図 6-2 ミドルウェア再生ライブラリの内部構成

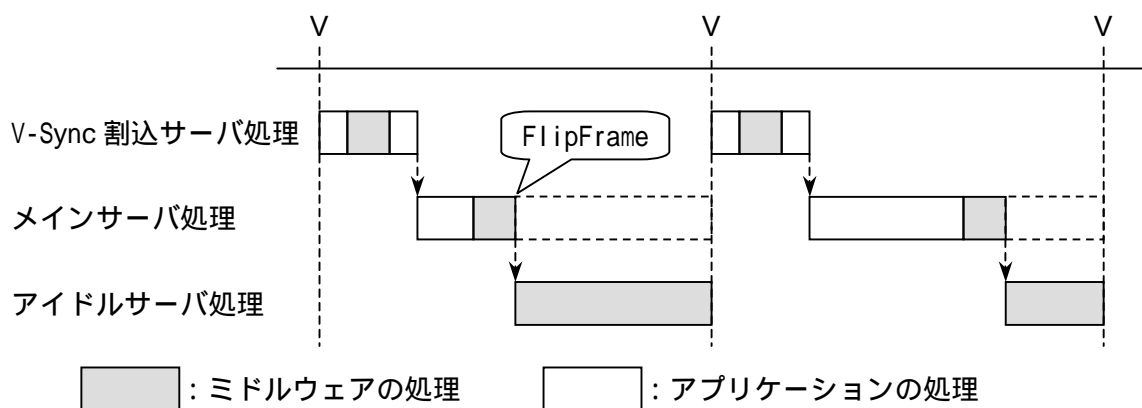
6.5 コントロールフロー

ミドルウェアの処理は、以下の3つのタイミングで行われます。

表 6-2 サーバ処理の種類

サーバ処理	説 明
V-Sync 割込サーバ処理	V-Sync 割込み時に実行されます。 音声データのデコードや音声の出力など、必ず定期的に行う処理を実行します。
メインサーバ処理	メイン処理内で、mwExecMainServer 関数が呼び出された時に実行されます。 映像データの転送や描画を行います。
アイドルサーバ処理	フレームフリップ関数内で、次のゲームフレームまでの時間に行われます。Ninja ライブラリでは、njWaitVSync 関数内で実行されます。 映像データのデコードなど、CPU 負荷が変動する処理を行います。

以下に、コントロールフローを示します。



コーデックによって、処理の行われているタイミングが異なります。

表 6-3 各コーデックによるミドルウェアの処理

	Sofdec	TrueMotion	WAVE	ADX
V-Sync 割込サーバ処理				
メインサーバ処理			×	×
アイドルサーバ処理		×	×	×

MPEG/Audio、DualSpeech は、WAVE と同じです。

Sofdec における各サーバ処理の内訳は以下の通りです。

- ：音声データのデコードと音声出力処理
- ：映像データのテクスチャメモリへの転送と描画処理
- ：映像データのデコード

TrueMotion における各サーバ処理の内訳は以下の通りです。

- ：音声出力処理
- ：映像データのデコードと映像出力処理

6.6 サウンドリソース

ミドルウェア再生ライブラリにおける音声出力には、サウンドドライバとサウンドライブラリを使用しています。ミドルウェア再生ライブラリ用にサウンドリソースを下記の通り確保して下さい。メモリブロックハンドルの最大値は 64 個なので、ミドルウェアを使用することにより、アプリケーション側で利用できるメモリブロックハンドルは減少します。

表 6-4 サウンドリソース

サウンドリソース	必要量
PCM ストリームポート	1
メモリブロックハンドル	4
サウンド RAM	4040H(16448)byte × チャンネル数

ミドルウェア再生ライブラリにより音声データを再生中、サウンド RAM 内の効果音や MIDI シーケンスを同時に再生できます。セガ・ライブラリ環境においてミドルウェア再生ライブラリは、サウンドライブラリの PCM ストリーム再生の機能を用いて実現されています。ミドルウェア再生用 PCM ストリームバッファは、サウンド RAM の最後から 4040H (16448) byte 単位で割り当てられます。従って、ミドルウェア再生ライブラリの使用するチャンネル数から PCM ストリームバッファのサイズを求め、その領域が重ならないようにマルチユニットファイルを作成する必要があります。

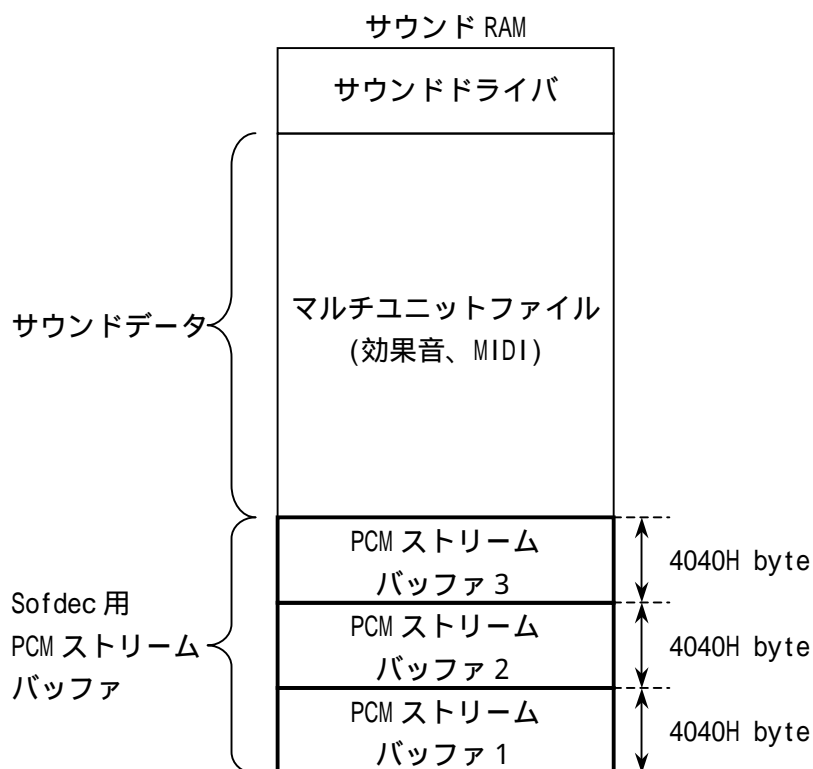


図 6-4 ミドルウェア再生用 PCM ストリームバッファの位置

6.7 ビデオリソース

グラフィックライブラリとの関係はコーデックによって異なります。

(1) Sofdec

Sofdec は、Kamui ドライバ上に実装されています。初期化時に、Ninja ライブラリにアクセスし、頂点バッファなどの情報を取得しています。初期化時以外では、Ninja ライブラリと独立に動作します。

以下に、グラフィックライブラリとの関係を示します。



図 6 - 5 Sofdec とグラフィックライブラリとの関係

(2) TrueMotion

TrueMotion は、Ninja ライブラリ上に実装されています。

以下に、グラフィックライブラリとの関係を示します。

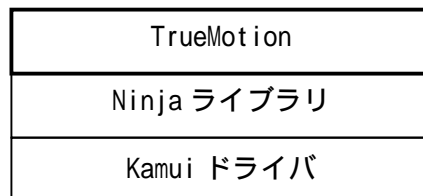


図 6 - 6 TrueMotion とグラフィックライブラリとの関係

動画の再生には、YUV422 テクスチャを使用します。Sofdec は、ハンドル作成時に Kamui ドライバを使用して確保します。TrueMotion は、ユーザが確保したテクスチャ領域に展開します。