

Release Notes

NITRO-SDK

2004/06/16

任天堂企画開発本部 環境制作グループ

Version: NitroSDK-1.2

本パッケージについて

本パッケージはNITRO上でのアプリケーション開発効率を高めるために用意されたライブラリ群です。ハードウェアレジスタを抽象化し、視認性の高いソースコードを作成するお手伝いをいたします。またメモリや割り込みなどのシステムリソース管理の標準的な機構を提供します。

パッケージに含まれるもの

- 環境制作グループ作成版NITRO-SDKライブラリ (グラフィクス・OSシステム etc)
- オンライン版関数リファレンスマニュアル
- NITRO機能別デモプログラム
- 開発ターゲットの切り替えを統合したmakeシステム

今回の変更点

今回のリリースでの変更点をご紹介します。過去の変更点については、オンラインリファレンスマニュアルを参照してください。[\\$NitroSDK/man/ja_JP/changelog.html](#) にあります。

1 全体的な変更点

1.1 ARM7 側統合コンポーネントの収録

NITRO 実行バイナリファイル作成時に ARM9 用実行ファイルと結合する ARM7 用実行ファイル(コンポーネント)を新しく作成し収録しました。ビルド時に特に指定がない場合はこのコンポーネントが適用されます。

1.2 本体 NVRAM に保持されるデータの書き換えツール

本体内蔵 NVRAM に保存される各種情報を書き換えるための簡易ツールを収録しました。タッチパネルのキャリブレーション設定ツールとしてお使いください。実行ファイルは `$NitroSDK/bin/ARM9-TEG/Release/BootMenu.bin` です。これを TEG で実行することで、簡易メニューが起動します。タッチパネル以外の設定項目については最終品と異なる可能性があるため、現状では参考資料であるとお考えください。

2 開発環境の変更点

2.1 CodeWarrior for NITRO V0.5 への対応

CodeWarrior for NITRO V0.5 に対応しました。ライブラリファイル名などの変更に対応しています。NITRO-SDK 1.2 では、この V0.5 でのご利用を前提に作成されています。

2.2 プロファイル機能への対応

CodeWarrior の profile 機能を利用した関数コールトレースと、関数コスト計測の機構を追加しました。NITRO-SDK における profile 機能のサポートについての解説は \$NitroSDK/docs/TechnicalNotes/MechanismOfProfiler.doc をご覧ください。

2.3 ROM イメージ生成ツールの変更

ROM イメージの生成ツールを elftobin から makerom に変更しました。またオーバーレイへの対応を簡単にするために makelcf コマンドを追加しました。これらのツールに関する情報が、リファレンスマニュアルの[ツール]->[ROM イメージ作成関連] に収録されています。

以前の SDK で試験配布していた makerom にあったバグが修正されています。

2.4 開発環境へのその他の変更

その他にも以下の変更が行なわれています。

- FINALROM 版のビルド時に -g オプションを追加しました。
- キャッシュラインに関連する問題回避のためオーバーレイセグメントの先頭と最後尾で 32 バイトアラインメントをとるように lcf 関連ファイルを修正しました。
- 実行バイナリ生成規則に依存ファイルを追加できるようになりました。
- Visual C++/.NET などの他のコンパイラから NITRO-SDK の基本インクルードファイルを読み込めるように \$NitroSDK/include/nitro_win32.h を用意しました。
- IDE デモサンプルのサポートを終了しました。

3 ライブラリ (API) の変更点

3.1 ファイルシステムライブラリ(FS)の追加

makerom によって生成したアプリケーション内で ROM ファイルシステムを使用するためのライブラリを追加しました。ディレクトリの列挙/検索、ファイルの同期/非同期リード、オーバーレイなどが可能になります。

詳しくはリファレンスマニュアルの FS ライブラリ API のページをご覧ください。

3.2 タッチパネルライブラリ(TP)の追加

タッチパネル制御関連 API がサポートされました。

ARM7 側と ARM9 側との処理分担の変更にともない、今まで別パッケージで配布していた TP ライブラリと API が変更されていますので、ご注意ください。また、この変更により、タッチパネルのキャリブレーションパラメータが変更になっているため、もう一度タッチパネルのキャリブレーション処理を行なってください。

3.3 マイクライブラリ(MIC)の追加

マイクデバイス进行操作する為のAPI、及びこれを使用したデモを追加しました。また、MICライブラリとタッチパネルやサウンドのライブラリを同時に使用するサンプルとして、“spiMonkey”デモを追加しました。

ただし現状の spiMonkey デモは全てのコントロールをARM9 側で自由に行なう都合上、あまりパフォーマンスは、良くありません。パフォーマンスを上げるためには、ある程度の使用される状況を想定し、自由度を減らした実装を行なう必要があります。この点は検討事項となっています。

3.4 CP ライブラリのスレッドセーフ化

除算器、平方根演算器を扱うための CP ライブラリをスレッドセーフ化しました。これによって複数のスレッドにおいて 明示的な排他処理を行なうことなく CP ライブラリを使用することができます。詳しくは CP 関連 API のリファレンスマニュアルをご覧ください。

3.5 スレッドシステムの変更および IRQ 割り込み待ち処理の変更

将来接続される様々なデバイスの制御およびミドルウェアなどとの親和性を考慮し、スレッドシステムの初期化をデフォルトで行なうように変更しました。OS_Init() で OS_InitThread() を呼んでいます。以前のように、スレッドシステムを使用しないライブラリを作成する場合は、NITRO_NO_THREAD を定義してライブラリを再コンパイルしてください。

この修正にともない、IRQの発生を待つための以下の関数を新設しました。

```
OS_WaitIrq  
OS_WaitAnyIrq()  
OS_WaitInterrupt()
```

スレッドシステムが有効である場合には OS_WaitIrq, OS_WaitAnyIrq() を使用します。これらの関数は IRQ の発生を待つ間、他のスレッドに処理権を譲ります。

スレッドシステムを使用しないときには OS_WaitInterrupt() を使用してください。これは、システムコールとして用意されていた SVC_WaitIntr() と同等の動作を行ないます。

またスレッドシステムに関して、他に以下の変更を行いました。

- 作成できる最大スレッド数を8から16に変更しました。
- 現在作成されているスレッド数を取得する関数 OS_GetNumberOfThread() を作成しました。
- スレッド切り替え時のコールバックが OS_SetSwitchThreadCallback() でセット可能になりました。
- OS_ExitThread() でスレッドを終了するときに、そのスレッドがロックしている Mutexを開放するようにしました。

3.6 メイン LCD, サブ LCD 出力切り替えフラグの変更

パワーコントロールレジスタの“LCD出力先切り換えフラグ”の設定フラグの GX_DISP_SELECT_*** を最終実機と一致するように定義値を入れ替え（0 と 1 の論理を反転）を行ないました。

同様の修正は、ensata でも行なわれています。SDK 1.2 以降の SDK で作成したアプリケーションを ensata 上で実行される場合は、ensata がこの修正に対応したバージョンであるかどうかご確認ください。

3.7 拡張パレット用VRAM設定 API 名称の変更

GXVRamBGExtPltt, GXVRamOBJExtPltt, GXVRamSubBGExtPltt, GXVRamSubOBJExtPltt で定義されている拡張パレット用のVRAM設定名称をスロット番号を含む表記へ統一しました。過去の名称はマニュアルには記載されませんが別名として残されます。

3.8 GX API の追加、および MtxFx22 型操作関数の追加

以下の 2D-OBJ データ値の取得関数 `G2_GetObj***()` をインライン関数として用意しました。
また、以下の 2x2 行列の `MtxFx22` 型変数を操作するための API が用意されました。

```
MTX_Identity22()  
MTX_Concat22()  
MTX_Inverse22()  
MTX_Transpose22()  
MTX_Rot22()  
MTX_Scale22()
```

3.9 scaleW を調整できる透視変換行列関連 API の追加

`scaleW` の値を任意に設定できる以下のような透視変換関連 API を作成しました。

```
G3_FrustumW(), G3_PerspectiveW(), G3_OrthoW()
```

またジオメトリエンジンへ送らずに行列作成のみを行なう以下の API も用意しました。

```
MTX_Frustum(), MTX_Perspective(), MTX_Ortho()
```

3.10 GX 関連のその他の追加と変更

グラフィクスライブラリ(GX) に関して、以下の変更が加えられました。

- ジオメトリエンジンの Swapbuffer 処理と同期を取るための手法のデモ `tips/SwapCrossOver` を追加しました。
- コマンド FIFO リセット処理を、NOP を 128 個送った後 `G3_End()` を発行するという実装に変更した。
- `GX_SetBankForARM7()` がサポートする引数として `GX_VRAM_ARM7_128_C` が追加されました。
- `GX_Init`の複数呼び出しの対策を追加しました。

3.11 割込みに関する変更

割り込み処理に関して、以下の変更が加えられました。

- ハードの仕様上、割込みの許可や禁止と実際のIRQの発生のタイミングの問題で IRQが発生して IRQハンドラに入ったときに IME が 0であったり、IEとIFの論理積が0 となることがありましたが、この時は何もせずに抜けるようにしました。
- 割込み要因として、SDKの割込み要因と同様にアプリケーションで自由に使用できる 2つのチェックフラグが用意されました。

3.12 メモリ保護(プロテクションリージョン)に関する変更

プロテクションリージョン設定関数 `OS_SetProtectionRegion()`をより直感的に設定出来るようにしました。
パラメータ設定の関数は `OS_SetProtectionReginParam()` です。

また、プロテクションリージョンの設定を取得する以下の関数を作成しました。

```
OS_GetProtectionRegionParam()
```

OS_GetProtectionRegionAddress()
OS_GetProtectionRegionSize()

3.13 OS 関連のその他の追加と変更

OS に関して、以下の変更を行いました。

- 変数配置のアラインメントを指定する ATTRIBUTE_ALIGN() マクロを作成しました。
- SpinLock で使用されるロックID関数 OS_GetLockID(), OS_ReleaseLockID() を作成しました。
- CPU クロック値 HW_CPU_CLOCK_ARM7, HW_CPU_CLOCK_ARM9, HW_CPU_CLOCK を定義しました。
- OS_Panic() の停止動作を OS_Halt()ではなく、OS_Terminate()としました。
- デモに waitIrq-1, waitIrq-2 が追加されました。
- 現在の動作環境を取得する関数 OS_GetConsoleType() を作成しました。
- FINALROM ビルドで OS_Printf() 等の表示関数が呼ばれないようにしました。
- 軽量な書式文字列関数 OS_SPrintf() (とその系列関数)を追加しました。
- メモリバイトアクセス対策のMI_ReadByte(), MI_WriteByte を用意しました。

4 バグ修正

4.1 TEG ボードにおけるコマンド FIFO の動作不良

TEG ボードにおけるコマンド FIFO 溢れ処理のハードの不具合により GX_LoadMtx44 などの処理が正常に行なわれないことがありました。これを、ライブラリ側で対策しました。このハードの不具合は製品版では修正されます。

4.2 GX 関連の不具合修正

グラフィクスに関して、以下の不具合が修正されました。

- BG拡張パレットへの GX_VRAM_BGEXTPLTT_01_F の指定時の不具合がありました。
- フォグモードの制御で fogMode を 1 から 0 に戻せませんでした。
- デモ AntiAlias で、ClearColorが設定されていなかったために背景色とブレンドされずアンチエイリアスがかかっていませんでした。
- デモ2D_Oam_5 と Sub_Oam_5 で、データキャッシュのFlushをしてなかったために FINALROM ビルドで正常動作していませんでした。

4.3 OS 関連の不具合修正

OSに関して、以下の不具合が修正されました。

- プロテクションリージョン7の操作を行なう API がリージョン6 に対して処理を行なっていました。
- 関数コールトレースの結果表示に不具合があり、4レベル以上のインデント表示を行うと表示が崩れていました。