NetVision

SVO SDK Manual

V1.21

2023.12.01

NetVision Corp.

目次

改版履歴

[19/11/21] 新規作成

[20/01/25] SVOCuiCtl サンプルを追加、SVO09USB30 ライブラリをアップデート

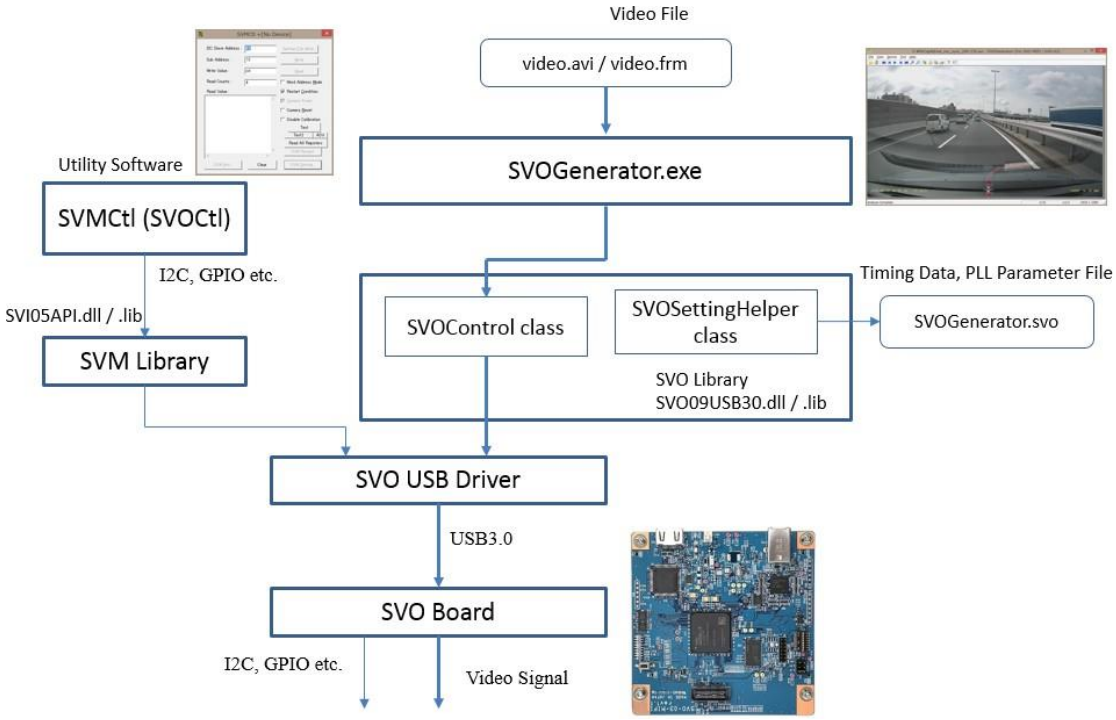
- ライブラリの C++ 関数は新しいクラス (CSVOControlMin) に移行しました。C 関数には変更ありません。

[21/06/10] 概要にシステム構成図などを追記

[21/07/30] SVOGenerator の説明と「新しい Visual Studio の使用」を追加、SVMCtl_I2C にデバッグ実行の注意事項を追加

[23/02/15] 各プロジェクトの内容を更新、NVFilePlayer を追加、「ビルドの構成」「ファイルパス」を追加
[23/12/01] 各プロジェクトの内容を更新

1. 概要



(SVO システム構成図)

NetVision SVO SDK は、NetVision 製 SVO-03 ボードや SVO-03-MIPI ボード等、弊社映像出力ボードをユーザプログラムから使用するためのライブラリ、サンプルソフトなどを格納した SDK です。

この SDK の内容物を下記に示します。SVO ボード付属の CD では、SVOGenerator ツールを使用して SVO ボードをコントロールすることを前提に設計されているため、外部ソフトウェアを使用してボードを操作するためのライブラリは含まれていませんでした。

SVO SDK では SVO09USB30Lib ライブラリを公開することで、SVOGenerator 以外のソフトウェアに SVO ボードの操作の組み込みが可能になっています。さらに、ライブラリの修正により外部ソフトウェアからの操作を容易にしたほか、C++ 言語で書かれたサンプルソフトも含まれるため、SVO ボードをよりフレキシブルに活用することができます。

ソフトウェア	SVO ボード付属 CD	SVO SDK
SVOGenerator バイナリ	○	○

ソフトウェア	SVO ボード付属 CD	SVO SDK
SVOGenerator ソースコード、VS2022	×	○
NVFilePlayer バイナリ	○	○
NVFilePlayer ソースコード、VS2022	×	○
SVO09USB30 ライブラリ	× (DLL のみ)	○
SVODirectController バイナリ＋ソースコード、VS2008	×	○
SVOGenExtCtl バイナリ＋ソースコード、VS2022	×	○
SVMCtl バイナリ	○	○
SVMCtl_I2C ソースコード、VS2022	× (SVM ボードのみサポート)	○ (SVO ボード対応)

SDK のライブラリやサンプルはすべて C++ 言語で書かれています。プロジェクトのビルドには MFC に対応した Microsoft Visual Studio 2008 または Visual Studio 2022が必要です。

SDK は開発途中につき、今後さまざまな新機能やサンプルの追加を行うことがあります。ドキュメントにも説明が不足している箇所があると思います。プログラム詳細についてはソースコード内の記述を参照いただきたく思いますが、ご不明な点がありましたら弊社サポート (sv-support@net-vision.co.jp) までご連絡いただけると幸いです。

1.1. 推奨動作環境

名前	値	備考
OS (Windows 版 SDK)	Windows 8.1 / 10 / 11	
CPU	Intel Core i5 以降	
Memory	3GB 以上	
記憶装置	SSD	

名前	値	備考
インタフェース	USB 3.0 ポート	
開発環境 (Windows) NVFilePlayer を除く	Microsoft Visual C++ 2008 Professional	要 MFC サポート
開発環境 (Windows) NVFilePlayer, 関連プラグイン	Microsoft Visual C++ 2019&2022 以降	要 MFC サポート

1.2. 新しい Visual Studio の使用

この SDK に含まれるいくつかのプロジェクトは Microsoft Visual C++ 2008 で開発されています。それより新しい Visual Studio でもビルドできますが、下記の注意が必要です。

Visual Studio 2019 や 2022 を使用する場合は、インストール時に、「C++ によるデスクトップ開発」をチェックして、規定のインストール項目に加えて「C++/CLI サポート」と「Windows 10 SDK」のオプションを選択してください。

1.3. DLL バージョン

このドキュメントに対応するライブラリ DLL (SVO09USB30.dll) のバージョンは 2.1.5.3 です。

1.4. ビルドの構成

SV シリーズのライブラリは x64, x86 で別れています。64bit OS で使用する場合、ビルドの構成は「x64」を選択してください。また、プロジェクトによっては Debug ビルドのパスが指定されておらず、ビルドできない可能性がありますので、基本的には「Release」ビルドを使用してください。

1.5. ファイルパス

一部のプロジェクトでは、各プロジェクトの「ビルド後のイベント」で生成されたファイルを別のプロジェクトにコピーするコマンドが入っています。使用中の環境とパスが合わない場合はエラーになることがあります。その場合は、「ビルド後のイベント」のコマンドを削除してください。

2. SVO09USB30 ライブラリ

場所: SVO_SDK/SVO09USB30Lib

2.1. 概要

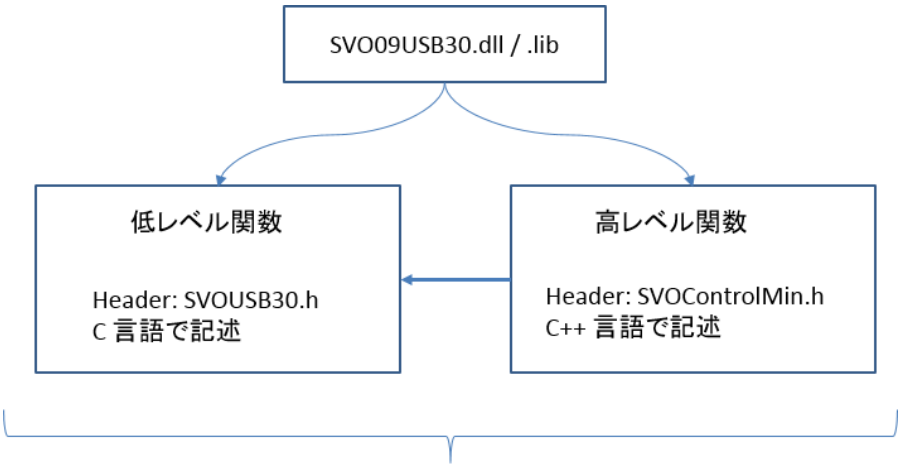
SVO09USB30 ライブラリは、C++ 言語で SVO ボードを操作するためのライブラリです。DLL ファイルとスタティックライブラリ、ヘッダファイルで構成されています。

下記のように、SVO09USB30 ライブラリには低レベル関数、高レベル関数の 2 レベルの関数群が存在し、それぞれのヘッダファイルが存在します。通常は高レベル関数のみ使用します。

低レベル関数は従来のライブラリとの互換用に残されており、ヘッダファイル「SVOUSB30.h」は C 言語で書かれています。

高レベル関数は外部ソフトから簡単に操作できるように設計されており、ヘッダファイル「SVOControlMin.h」は C++ 言語で書かれています。関数はボードの操作やデータ転送を行う SVOControl クラス(CSVOControlMin) と、タイミング設定ファイルの入出力などを行うための SVOSettingHelper クラス (CSVOSettingHelper) から構成されています。

高レベル関数は内部で低レベル関数を呼び出しています。すべてのボードの機能は高レベル関数のみで実行できるので、本ドキュメントでは低レベル関数の説明は行いません。



ライブラリには x64 版と x86 版が存在しますが、64bit OS 環境では x86 版ライブラリは動

作しませんのでご注意ください。

ライブラリ内の関数一覧は、ヘッダファイルを参照してください。

2.2. ファイル一覧

場所	ファイル名
SVO09USB30Lib¥Include¥	SVOControlMin.h
	SVOError.h
	SVOStruct.h
	SVOUSB30.h
SVO09USB30Lib¥bin_x64¥	SVO09USB30.dll
	SVO09USB30.lib
SVO09USB30Lib¥bin_x86¥	SVO09USB30.dll
	SVO09USB30.lib

2.3. ユーザソフトへの組み込み

ユーザソフトにライブラリを組み込むには、上記のヘッダファイルを同じフォルダにコピーした上で、 ヘッダファイル SVOControlMin.h をインクルードしてください。 ボード操作は CSVOControlMin クラス、ボードへの設定の管理（SVOGenerator でエクスポートした設定ファイルの読み書き）は CSVOSettingHelper クラスによって行います。

ユーザのプログラムから CSVOControlMin クラスのインスタンスを 1 つ作成し、1 つの SVO ボードを開きます。出力タイミングなどボードへの設定は CSVOSettingHelper クラスのインスタンスを作成し、.svo ファイルを読み込み、CSVOControlMin にの設定関数を呼び出します。

プログラム作成時はスタティックライブラリ SVO09USB30.lib を静的にリンクしてください。また、.exe ファイルと同じフォルダに SVO09USB30.dll ファイルを置いてください。ライブラリは 32bit (x86) と 64bit (x64) がありますが、64bit OS で 32bit ライブラリは正常動作しませんので、ご注意ください。64bit OS で使用する場合、x64 のライブラリを使用する必要があります。

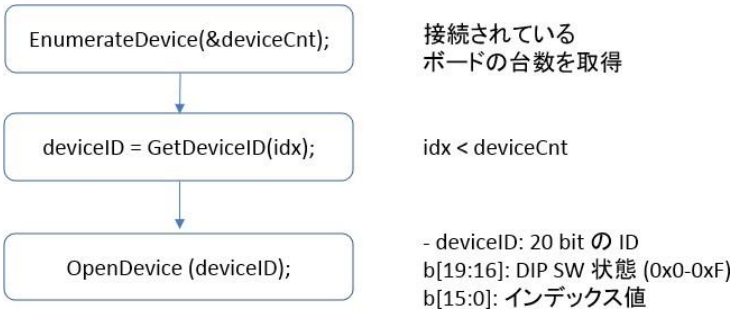
出力映像データの送出方法は 2 種類あります。1 つはボード上 DRAM に映像フレームを数十フレーム分転送し、再生操作を行うことで DRAM 内のデータをループして再生する方法です。この方法では再生中のソフト処理は必要ありません。もう 1 つは、ボード上 DRAM の領域を 2 つに分けて、ボードの転送状態をポーリングする方法です。片側の領域が空になったら、その領域に次のフレームデータを送出します。これにより、任意の長さの映像データを送信します。この方法では、再生中にソフトウェアによる処理が必要です。

実装例は「SVODirectController」サンプルや「SVOCuiCtl」サンプルも参照してください。

2.4. SVO ボードの操作方法

下記にライブラリの基本的な操作方法を説明します。なお、下記の説明ではクラス名(CSVOControlMin) や一部の定数の接頭辞を省略しています。

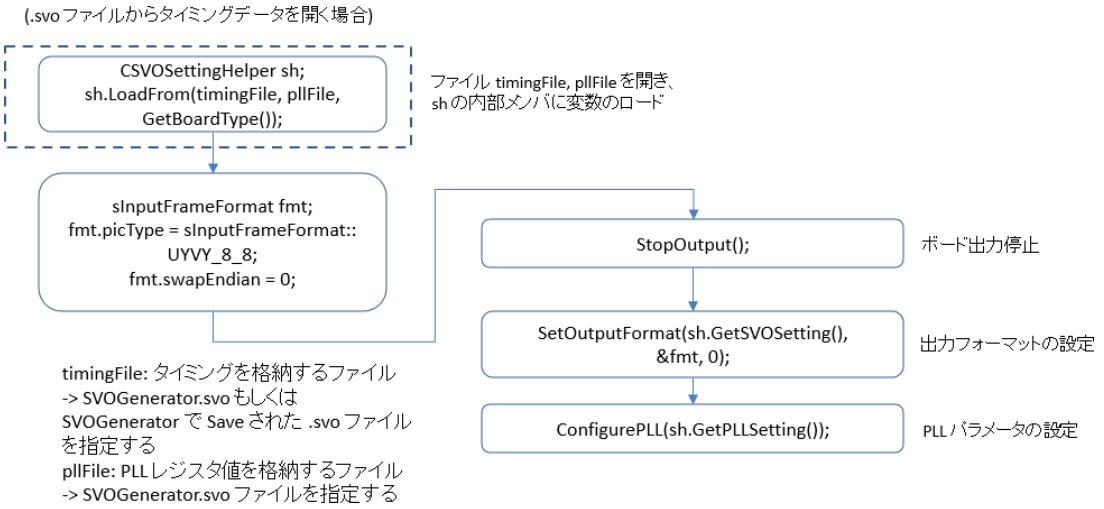
2.4.1. ライブラリ初期化とボードのオープン



ライブラリの初期化時は、上図の順番で関数を呼び出します。

ボードは GetDeviceID 関数で取得された deviceID によってライブラリと紐づけされます。deviceID の b[19:16] には ボード上 DIP SW の状態が反映されるため、これを利用して物理ボードと deviceID との間の関連付けを行うことができます。OpenDevice で deviceID を指定してボードを開きます。

2.4.2. レジスタの初期化



ボード上の FPGA レジスタの初期化（出力映像信号のタイミングデータや PLL データの FPGA レジスタへの書き込み）は、SetOutputFormat 関数と ConfigurePLL 関数によって行います。

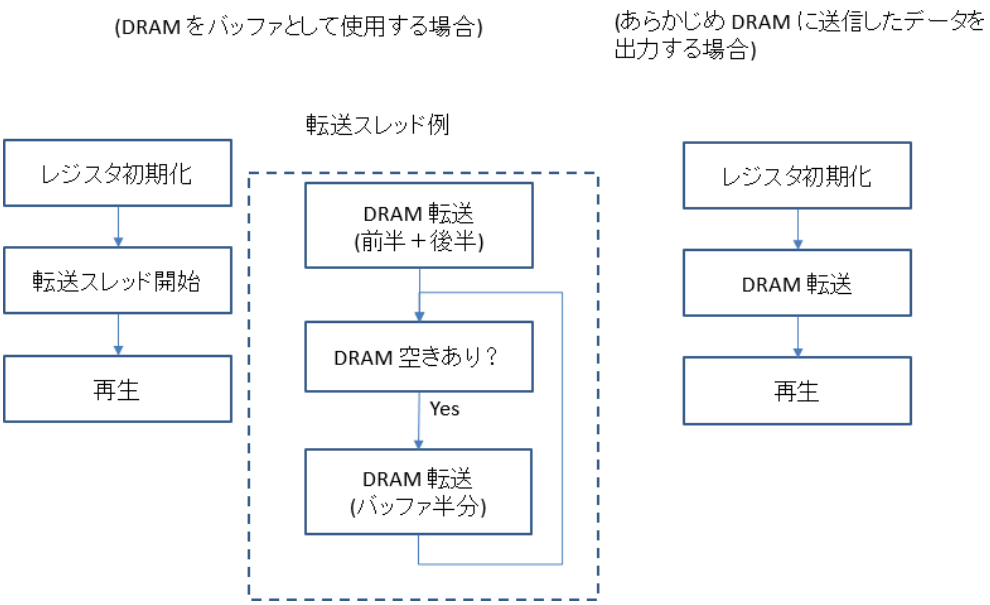
2.4.3. DRAM に映像データの転送

SVO ボード上には 128MB または 256MB の DRAM が搭載されており、この DRAM にホストから映像データを転送して、ボードは DRAM 上のデータを読み出して、映像信号を出力します。SVO には 2 つの動作モードがあり、

(1) DRAM をバッファとして使用して、随時ホストから映像データを転送するモード

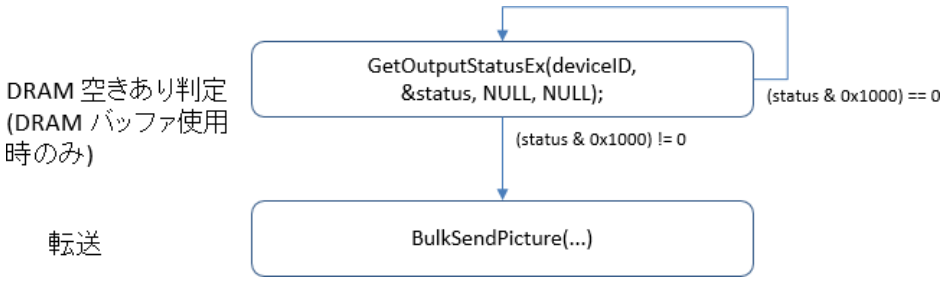
(2) DRAM に映像データを送信しておき、そのデータを繰り返し再生するモード

のどちらかを指定して動作することができます。このモードは StartOfflineOutPut 関数の playMode 引数として指定します。

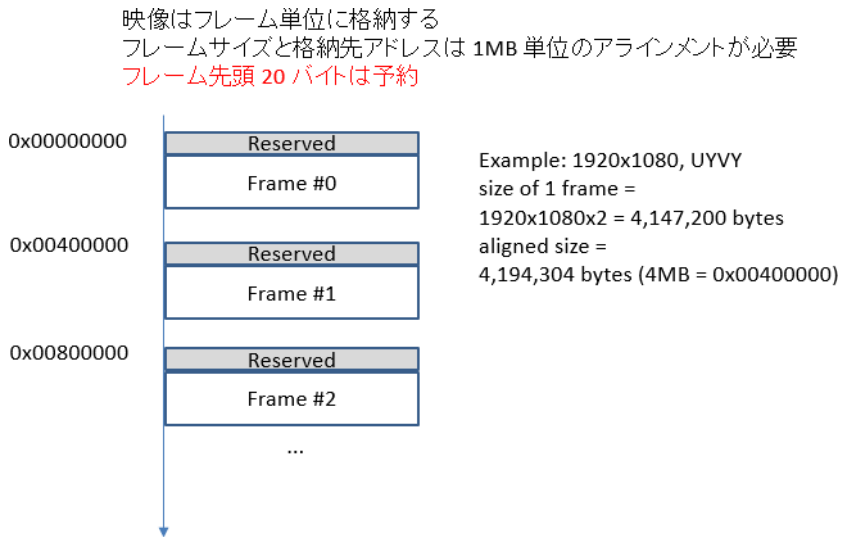


DRAM をバッファとして使用する場合、DRAM を 2 分割して、バッファの空き状態をポーリングして DRAM 転送を行います（上図左側）。この場合、ボードへのデータ転送スレッドを分ける必要があります。

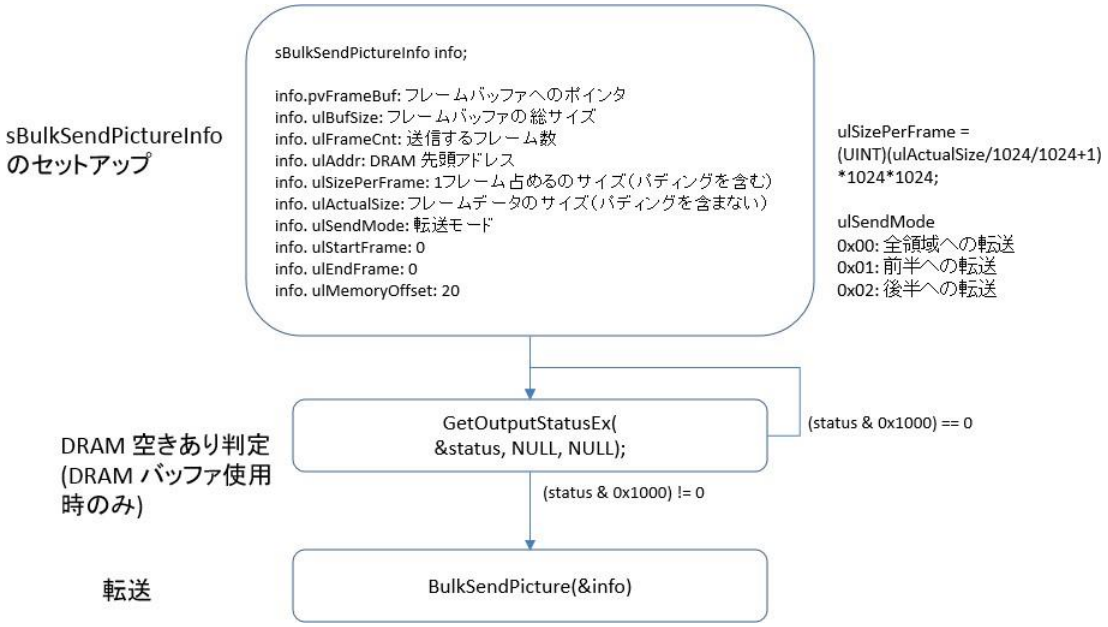
あらかじめ DRAM に映像データを送信して DRAM 内部のデータをループ再生する場合は、再生中のソフトウェアからの処理は不要です。



DRAM の転送は BulkSendPicture 関数によって行います。(1) のモードで使用する場合、GetOutputStatusEx 関数を使って DRAM バッファの空き状態を取得して、転送制御を行います。映像はフレーム単位に格納しますが、フレームサイズと格納先アドレスは 1MB 単位のアラインメントが必要です(下図参照)。実装は SVODirectController サンプルも参照してください。



2.4.4. 映像データの転送

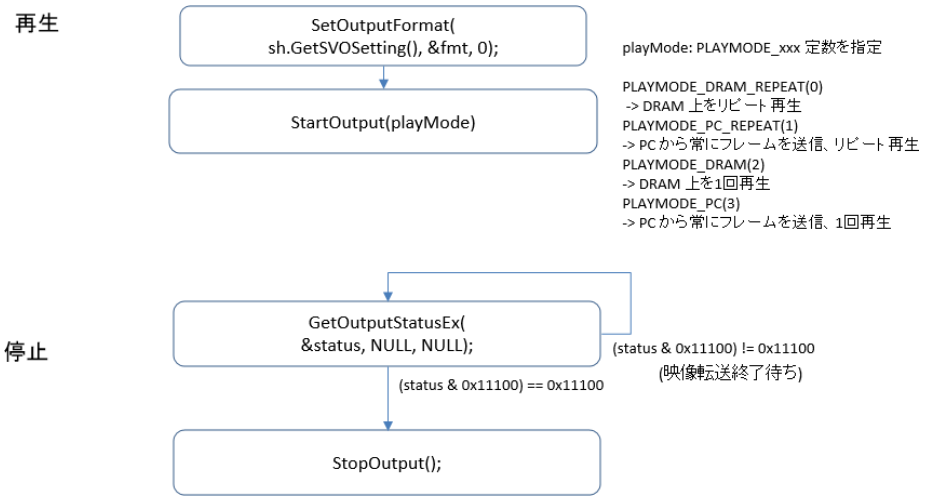


映像データは BulkSendPicture 関数によって転送します。初回は ulSendMode = 0 として、DRAM の全領域(サイズは GetDRAMSize 関数で取得する)に映像データを転送します。DRAM をバッファとして使用する場合は、DRAM の空きフラグを取得して、空きがあれば半分の領域に対してデータを転送します。2 回目以降の転送は ulSendMode = 1 or 2 として、ulAddr に転送先アドレスを適切に指定してフレームを転送します。

タイミングデータに格納されている出力フレームのサイズは、CSVOSettingInfo2::picWidth,

picHeight に格納されています。ただし picHeight はピクセル単位ではなくクロック単位となっています。CSVOSettingHelper::GetImageWidth 関数を使ってピクセル単位の解像度を取得することができます。

2.4.5. 映像出力の開始と停止



映像出力の開始と停止操作は上図に従ってください。再生モードは StartOutput 関数の引数によって指定します。現在の仕様では PLAYMODE_DRAM_REPEAT 以外の場合のボード側処理は同じとなっていますが、将来の拡張のため、適切なモードを設定してください。

再生モードに PLAYMODE_PC または PLAYMODE_PC_REPEAT を選択した場合、GetOutputStatusEx() で取得したボード情報をポーリングして、DRAM の前半/後半に分けて随時データを送信します。

外部からの F-SYNC 信号に同期させる場合、外部同期が有効なタイミングデータをロードした上で、SetOutputFormat の 3 番目の引数に XT_ENABLE を指定してください。SetOutputFormat の 3 番目の引数が 0 の場合、フリーランモードで動作します。

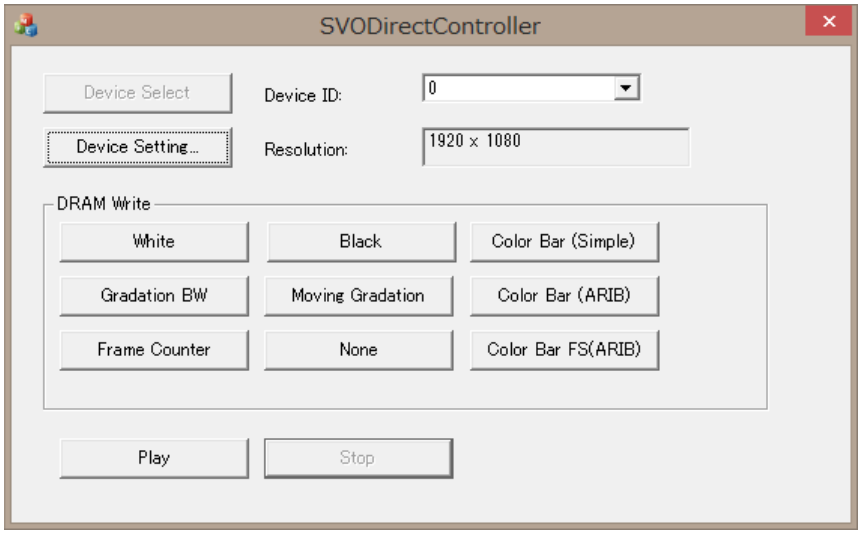
2.4.6. ライブラリの終了

ライブラリの終了時は CloseDevice() 関数を呼び出してください。その他の終了処理はクラスのデストラクタによって自動的に行われます。

3. SVODirectController

場所: SVO-SDK/SVODirectController

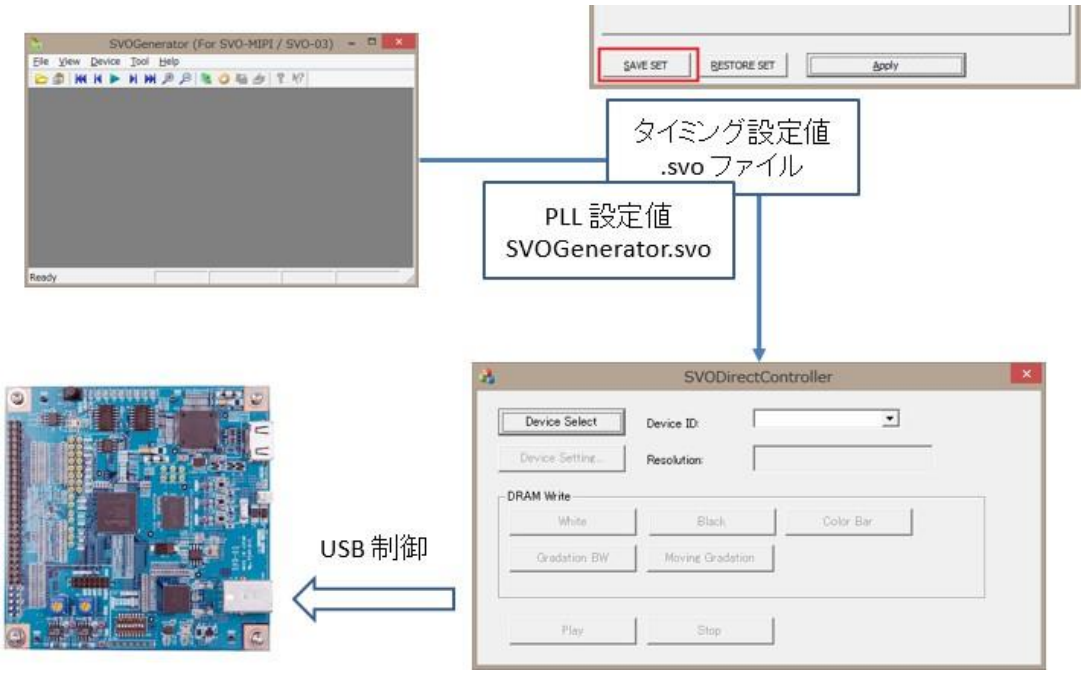
3.1. 概要



「SVODirectController」は、SVO09USB30 ライブラリを使用して直接 SVO ボードを操作するサンプルソフトです。このサンプルは C++ 言語で書かれており、VC2008 のプロジェクトファイル、x64 版、x86 版のソフトウェアとソースコードが含まれます。

このサンプルでは、SVO ボードの SDRAM に画像データの転送と、レジスタ設定、再生、停止操作という一通りの操作を行っています。SVO ボードは DRAM 上データを繰り返し再生するモードで動作します。SVO09USB30 ライブラリの仕様に関しては、本サンプルの内容も合わせてご参照ください。

SVO ボードの動作には、映像信号のタイミング設定とピクセルクロックなどの PLL 設定値を FPGA レジスタに書き込むことが必要です。「SVODirectController」では下図のように、映像信号のタイミング設定と PLL 設定値は「SVOGenerator」で生成したファイルを読み込み、レジスタ設定を行う仕組みになっています。 PLL 設定値は「SVOGenerator.exe」と同じフォルダに存在する「SVOGenerator.svo」ファイルを使用します。タイミング設定は、SVOGenerator の Control Dialog 画面で「SAVE SET」ボタンによりエクスポートした .svo ファイルを指定するか、最後に設定された設定値を使用する場合には「SVOGenerator.svo」ファイルを指定します。

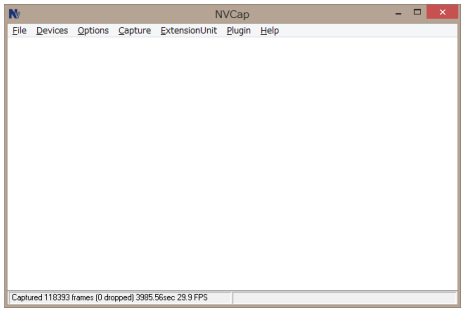


3.2. 操作方法

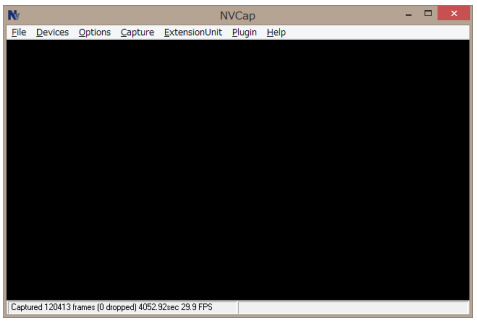
ボタン	説明
Device Select	クリックすると、「Device ID」で選択されたボードを開きます。正常に開くことができると、Device Setting... ボタンが有効になります。
Device Setting...	ボタンをクリックすると、ファイルを開くダイアログボックスが 2 回表示されます。 最初のダイアログでは、映像信号のタイミングデータを指定します。タイミングデータは SVOGenerator で出力した .svo ファイルを指定します。ただし、現時点ではピクセルフォーマット UYVY 形式にのみ対応しています。 2 回目のダイアログでは、PLL 設定の格納されている「SVOGenerator.svo」ファイルを開きます。通常は SVOGenerator.exe と同じフォルダに存在するファイルを開きます。 ファイルが選択されると、ファイルの内容をもとに SVO ボードのレジスタ設定が行われます。正常にレジスタ設定が完了すると、DRAM Write ボタンと Play ボタンが有効になります。また、設定ファイルから読み出された解像度が Resolution ボックスに表

	示されます。 
DRAM Write	DRAW Write 内のボタンをクリックすると、ボード上 SDRAM に映像データの転送を行います。映像出力が Stop した状態で映像データの転送を行ってください。 (1) Moving Gradation / Frame Counter 以外 固定パターンの映像フレームを 1 フレーム出力します。 (2) Moving Gradation / Frame Counter パターンが左に移動する映像を出力します。
Play	映像出力を開始します。
Stop	映像出力を終了します。

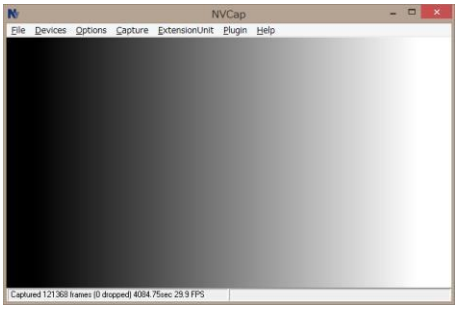
3.3. 出力映像例

White	Color Bar (Simple)
	

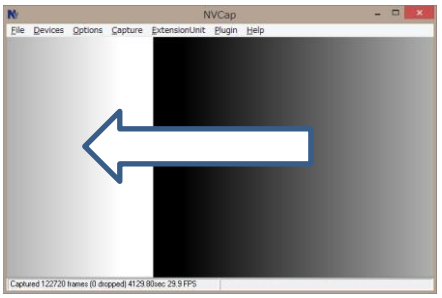
Black



Gradation BW



Moving Gradation



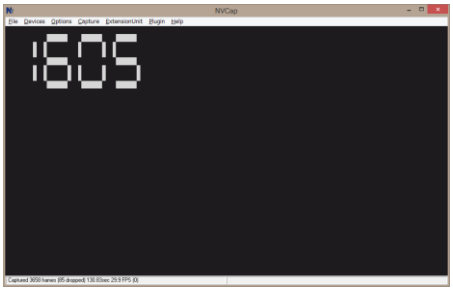
Color Bar(ARIB)

Limited Range のカラーバー

Color Bar FS(ARIB)

Full Range のカラーバー

Frame Counter

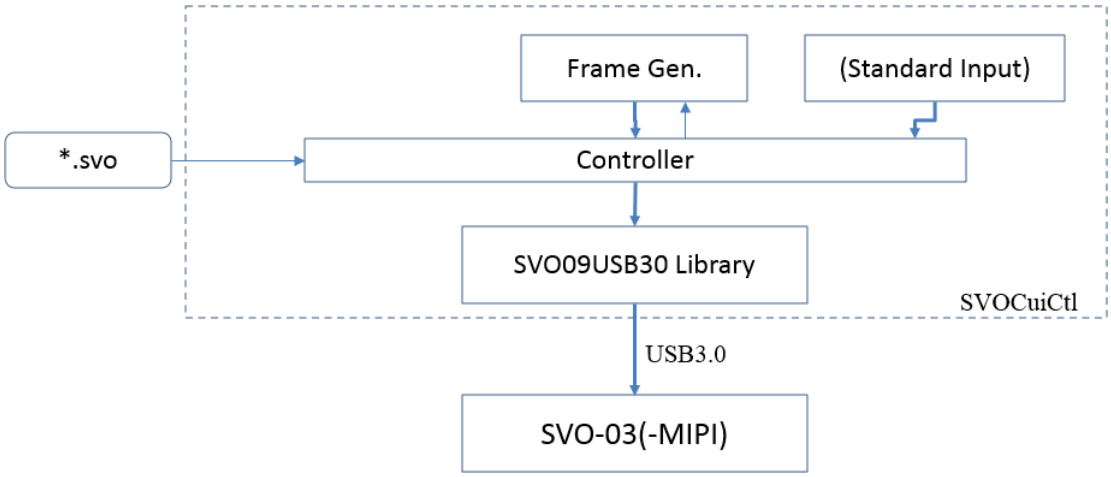


4. SVOCuiCtl

場所: SVO-SDK/SVOCuiCtl

4.1. 概要

SVOCuiCtl は、コマンドラインベースで SVO ボードのコントロールを行うサンプルソフトです。再生・停止などのボードの操作は標準入力によって行います。対応 OS は Windows 7 (x64) 以降 です。



4.2. 仕様

対応OS	Windows 7/8.1/10 x64
対応ボード	SVO-03, SVO-03-MIPI

4.3. コマンドラインオプション

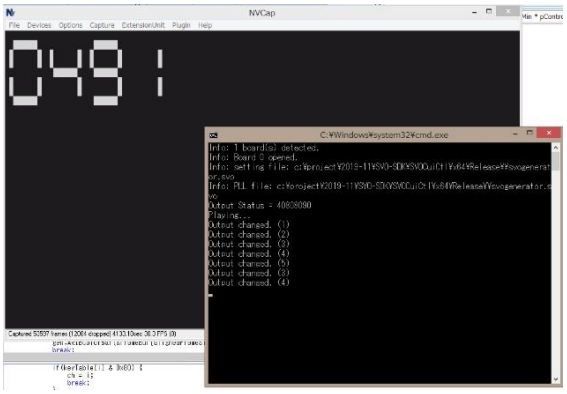
Required	Option	Description
	-id [ID]	開くボード ID (0-15) を指定します。 ボード ID には、ボード上 DIP SW で指定されている ID を指定します。 指定しない場合、最初に見つかったデバイスを開きます。
*	-s [filename]	出力タイミングファイル (.svo) を指定します。 タイミングファイルは SVOGenerator (ボード付属のコントロ

		ールソフト) より出力します。
	-pll [filename]	PLL 設定ファイルを指定します。 指定がない場合、-s オプションで指定したファイルを PLL 設定ファイルとみなして開きます。
	-mode [mode]	動作モードを指定します。 'avi': AVI ファイルを開いて出力するモードです。 (デフォルト) 'gen': フレームジェネレータで生成した映像を出力します。
	-pattern [number]	フレームジェネレータの出力パターンを数値で指定します。

4.4. ソフトウェア操作方法

ソフトウェア起動後、ボードの設定が正常に終了すると、キーボード入力(標準入力)を受け付けます。キーと動作の関係は下表の通りです。

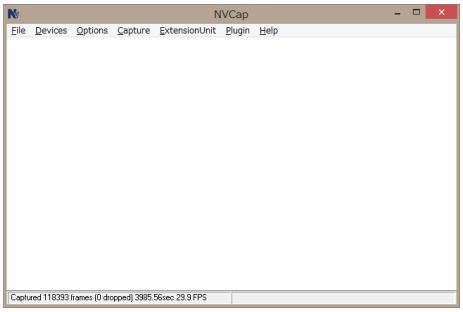
Input	Description
'q' or [ESC]	ソフトウェアを終了します。
'0' - '9'	ジェネレータの出力パターンを変更します。



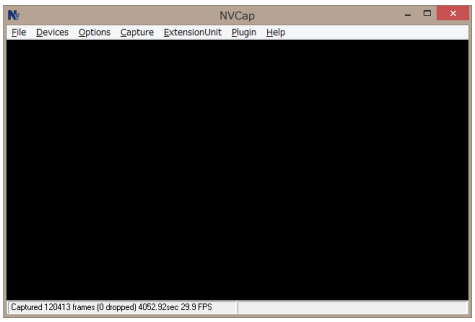
(実行中の画面)

4.5. 内蔵ジェネレータの出力パターン

0: White



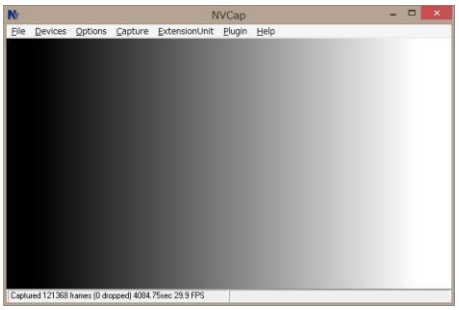
1: Black



2: Gradation BW

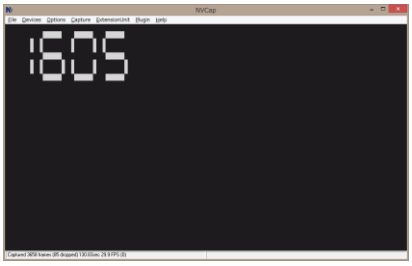
3: Moving Gradation

Moving Gradation はグラデーション画像を右から左に移動するパターンです。



4: Frame Counter

DRAM をバッファとして使うサンプルです。0000 - 9999 のフレームカウンタを表示します。



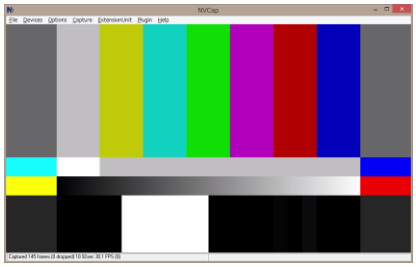
5: Color Bar (Simple)

6: Color Bar (ARIB)

Y　レンジはリミテッドスケールです。

7: Color Bar FS (ARIB)

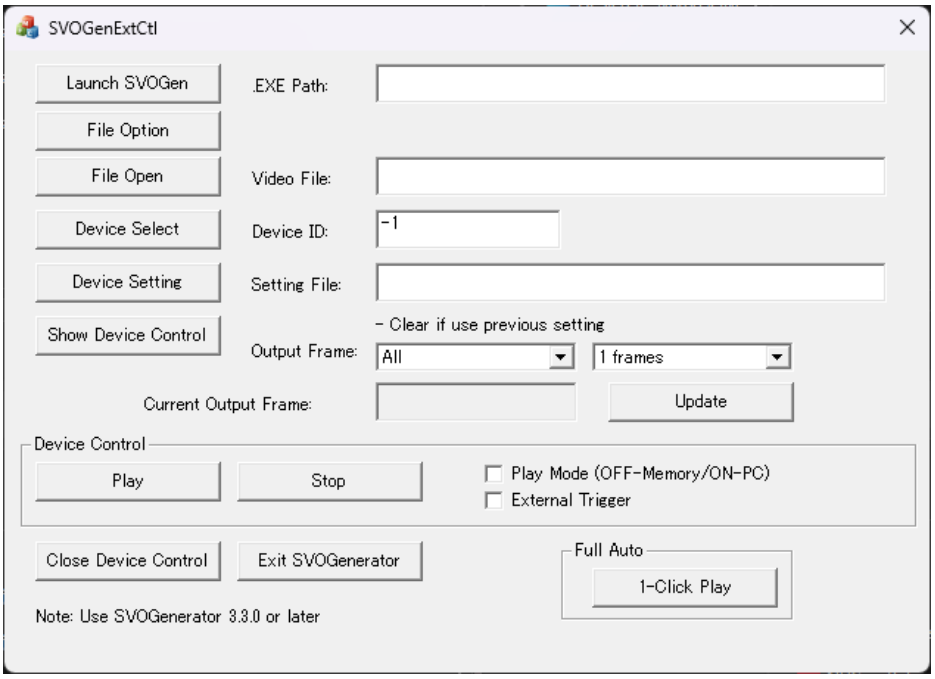
Y レンジはフルスケールです。



5. SVOGenExtCtl

場所: SVO-SDK/SVOGenExtCtl

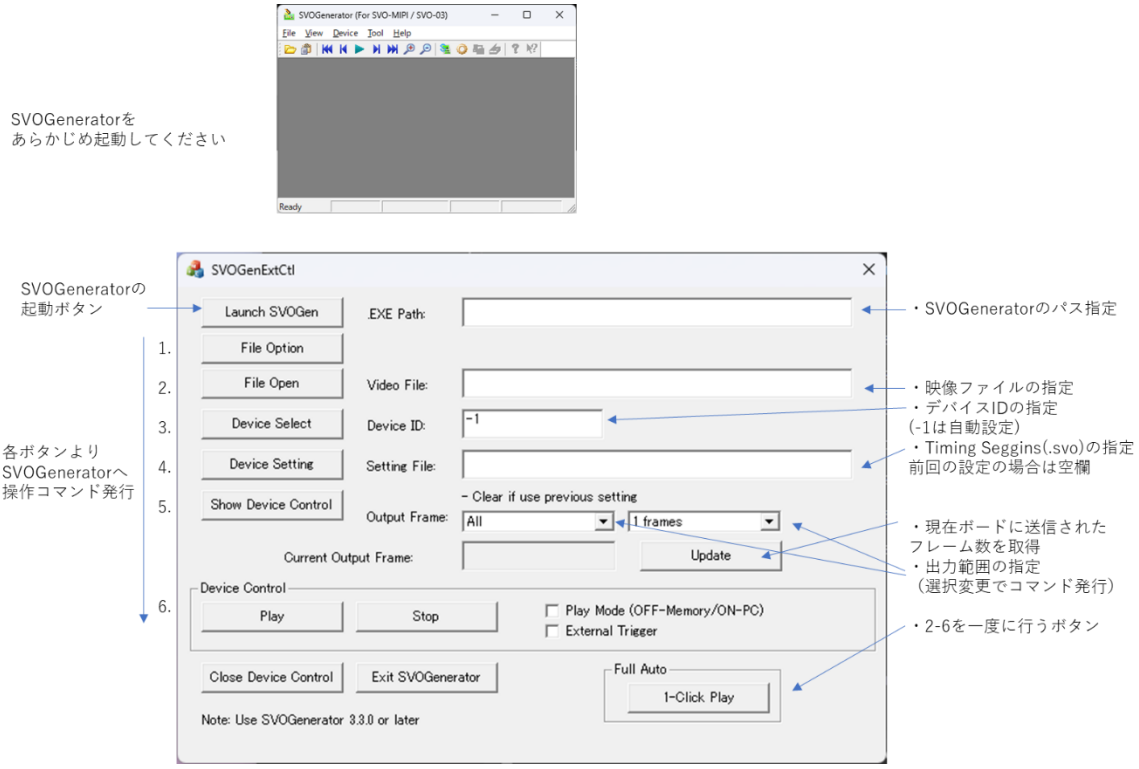
5.1. 概要



SVOGenExtCtl プロジェクトは、Windows メッセージを使用したプロセス間通信で SVOGenerator ツールを操作するためのサンプルソフトです。SVOGenerator は GUI で操作する以外にも、SVOGenerator のウィンドウに送られた特定の Window メッセージを受信することで一連の操作が行えるようになっています。Window メッセージで操作することで、任意の言語で書かれた Windows アプリケーションに SVOGenerator の操作を組み込むことができます。このサンプルは C++ 言語で書かれており、VC2022 のプロジェクトファイル、x64 版、x86 版のソフトウェアとソースコードが含まれます。

Windows メッセージを使用したプロセス間通信の仕様については、別資料「SVOGenerator_プロセス間通信仕様.pdf」を参照してください。

5.2. 機能説明



本サンプルは SVOGenerator のウィンドウを検索し、検出されたウィンドウに対してウィンドウメッセージを送出します。したがって、ソフトウェアの操作を行う前に、あらかじめ SVOGenerator を起動する必要があります。

「Launch SVOGen」ボタンをクリックすると、「EXE Path」に指定されたパスの SVOGenerator が起動します。

「File Option」ボタンをクリックすると、SVOGeneratorのFile-Optionと同等のダイアログが開きますので、映像ファイルの属性指定を行います。

「File Open」ボタンをクリックすると、「Video File」に指定された映像ファイルを SVOGenerator で開くよう、ウィンドウメッセージが送られます。SVOGenerator で映像ファイルを開いた場合と同じ挙動となります。

「Device Select」ボタンは、右側の Device ID に指定されたデバイス ID にもとづき、SVO ボードを開きます。

「Device Setting」ボタンをクリックすると、Setting File が空白の場合、以前の起動時に設定された SVOGenerator のデバイス設定が引き続きセットされます。Setting File に .svo ファイルを指定すると、このファイルの設定を読み込み、セットします。

「Show Device Control」「Play」「Stop」「Close Device Control」ボタンは、それぞれ SVOGenerator の Device Control ダイアログの表示や、ダイアログの操作と同じ動きをします。

「1-Click Play」をクリックすると、上図 2. - 6. の操作を順次自動的に行います。

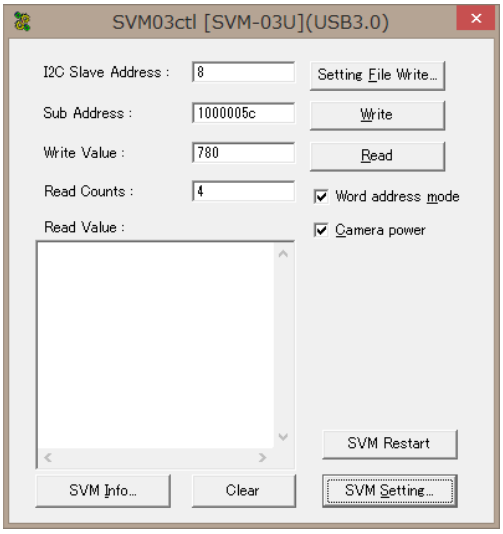
6. SVMCtl_I2C

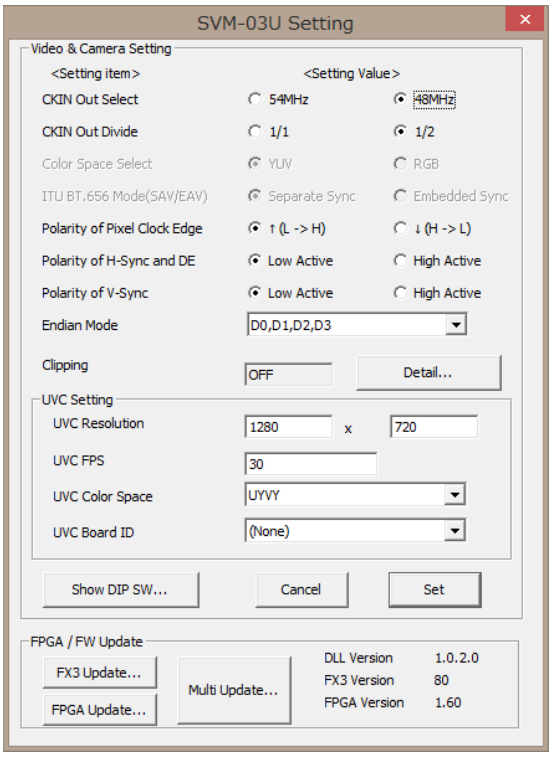
場所: SDK_Windows/SVMCtl_I2C/

6.1. 概要

SVMCtl は、SVM / SVO など弊社キャプチャボードを使用した I2C 通信や、SVO ボードのアップデート、センサの仕様設定、UVC 解像度設定等を行うための Windows 用ユーティリティソフトです。本 SDK に付属の「SVMCtl_I2C」は SVO ボードの CD に付属された標準版「SVMCtl」の I2C 通信部分のみを抜き出したプロジェクトです。

下記は標準版 SVMCtl の画面です。



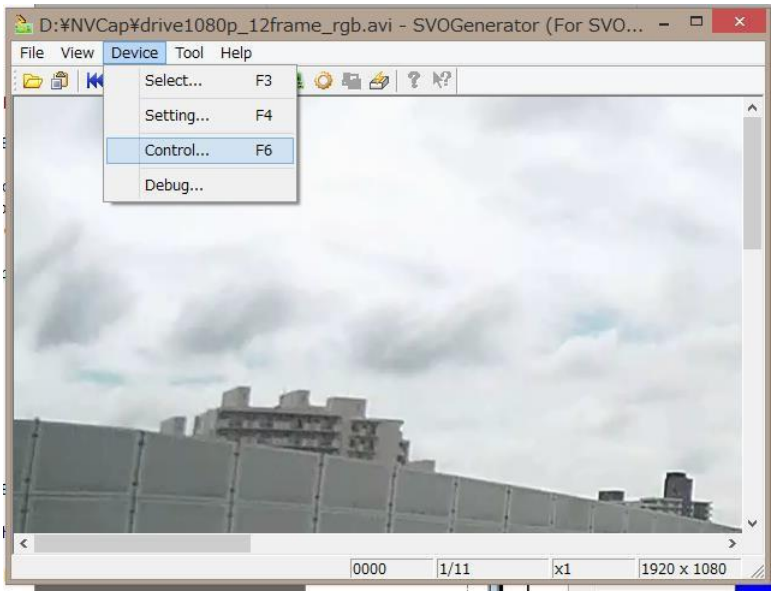


SVMCtl の機能詳細や使用方法については、ソフトウェアマニュアル「SVMCtl ソフトウェアマニュアル_9.9.pdf」を参照してください。

プロジェクトの設定で出力ファイル名を上書きしているため、ビルドは成功しても Visual Studio から実行できないことがあります。その場合、出力ファイル名を変更するか、デバッグコマンドを実際に生成される .exe ファイル名に上書きしてください。

7. SVOGenerator

場所: SDK_Windows/SVOGenerator_SVO-MIPI/



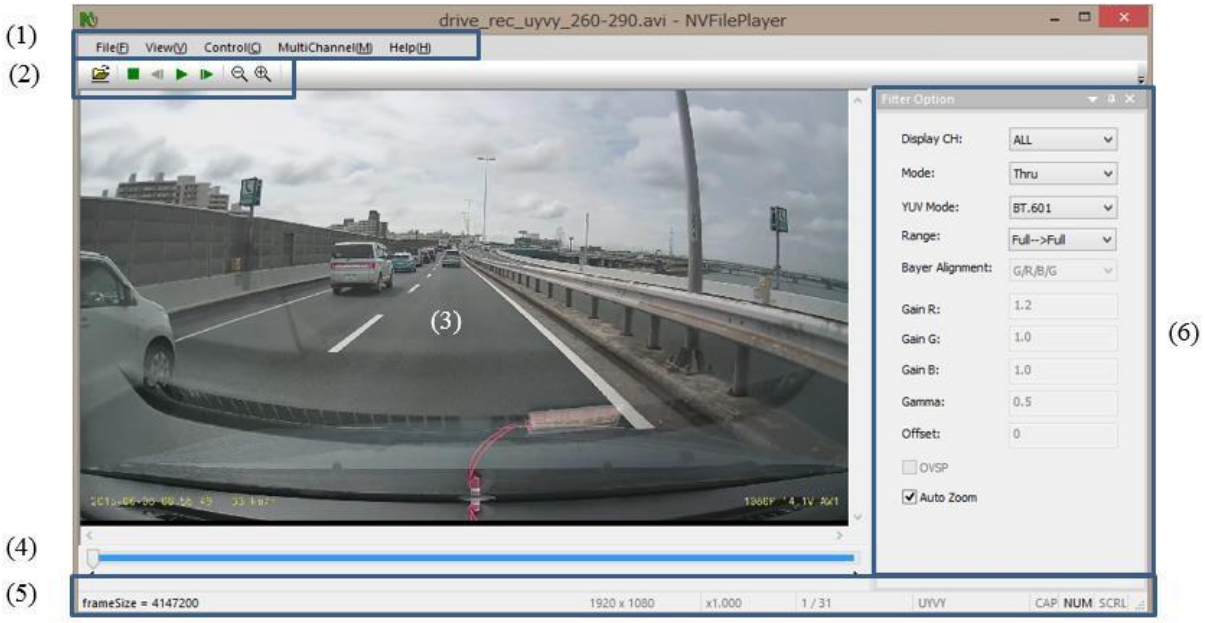
7.1. 概要

SVOGenerator は、SVO-03 ボード、SVO-03-MIPI ボードを使用して映像信号をボードから出力するための映像出力用ソフトウェアです。SVOGenerator は各 SVO ボードに標準で付属しているソフトウェアであり、本 SDK にはこの SVOGenerator のソースコードを格納しています。

SVOGenerator は 1 つの映像ファイルまたは映像ファイルのリストを読み込み、ボードに映像データを転送します。映像ファイルは弊社キャプチャソフトで録画した .avi ファイルまたは .frm ファイルに対応しています。ffmpeg などのソフトを使用して出力した、非圧縮 (YUV4:2:2 8-bit または RGB24) の .avi ファイルを読み込ませることも可能です。

SVOGenerator の操作方法などの詳細は、「SVOGenerator_ソフトウェアマニュアル」を参照してください。

8. NVFilePlayer



(1) メニューバー

(2) ツールバー

(3) 映像表示画面

(4) シークバー

(5) ステータスバー

(6) Filter Option 画面

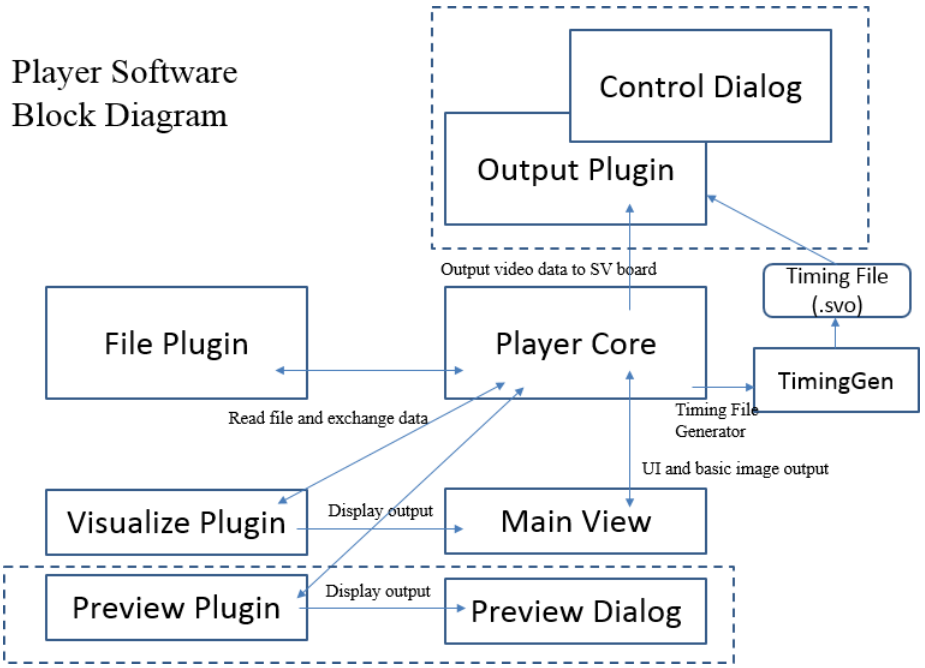
8.1. 概要

NVFilePlayer は、SVO-03 ボード、SVO-03-MIPI ボード、SVP-01-G など (SV シリーズ) を使用して映像信号をボードから出力するための映像出力用ソフトウェアです。また、SV シリーズを接続していない状態でも使用できるため、UYVY / RGB 形式の非圧縮動画ファイルの再生ソフトとしても使用することができます

NVFilePlayer は 1 つの映像ファイルまたは映像ファイルのリストを読み込み、ボードに映像データを転送します。映像ファイルは弊社キャプチャソフトで録画した .avi ファイルまたは .frm ファイルに対応しています。ffmpeg などのソフトを使用して出力した、非圧縮 (YUV4:2:2 8-bit または RGB24) の .avi ファイルを読み込ませることも可能です。

NVFilePlayer の操作方法などの詳細は、「NVFilePlayer_ソフトウェアマニュアル」を参照してください。

8.2. プロジェクト構成



「NVFilePlayer」はファイル読み込みや画面表示をプラグイン化しており、ソフトウェア本体といくつかのプラグイン、タイミングジェネレータ (TimingGen) によって構成されます。標準版の「NVFilePlayer」では、File Plugin に AVI と RAW 形式の動画ファイルを読み込むためのプラグイン、Visualize Plugin にこれらの動画ファイルを表示するためのプラグインを同梱しています。また、SVO シリーズのボードを制御するための Output Plugin にあたる機能はソフトウェア本体に組み込まれています。「NVFilePlayer」はカスタマイズされたプラグインを提供することで、ユーザの要求にフレキシブルに対応することができる構成となっています。

ボードから出力する映像信号のタイミングは、ボードごとの「TimingGen」によって生成された SVOGenerator.svo ファイルを経由して設定されます。NVFilePlayer の「Board Setting」メニューから、認識されたボードに応じた「TimingGen*.exe」を呼び出す形になっています。

この SDK の「NVFilePlayer」フォルダに NVFilePlayer 本体、プラグイン、各ボードの TimingGen を格納しています。

8.3. 付属プラグイン一覧

SDK に付属するプラグイン（SVO 標準仕様）は下記の通りです。

プラグイン名	格納フォルダ名	機能
NVFilePlugin_RAW.dll	NVFilePlayer_FilePlugin_RAW	RAW ファイル (ヘッダ、フッタのないフレームデータのみのデータ)を読み込むためのプラグインです。複数フレームが格納された RAW ファイルにも対応します。
NVFilePlugin_AVI.dll	NVFilePlayer_FilePlugin_6ch	AVI ファイルを読み込むためのプラグインです。圧縮された AVI ファイルには対応しません。フォーマットは一般的な AVI 2.0 フォーマットとなります。 サイズ 0 のフレームは無視されます。
NVFilePlugin_FRM.dll	NVFilePlayer_FilePlugin_FRM	SVI-06 など、SVI シリーズで録画された FRM ファイルを読み込むためのプラグインです。圧縮された AVI ファイルには対応しません。フォーマットは SVI シリーズ付属の説明書を参照してください。
NVVisualizePlugin.dll	NVFilePlayer_VisualizePlugin_avi	画面表示を行うための表示用プラグインです。 RAW->Grayscale / RGB 変換、YUV->RGB 変換を行います。

8.4. タイミングジェネレータ (TimingGen) 一覧

ボード種類	格納フォルダ名
SVO-03-MIPI	TimingGenMIPI
SVO-03	TimingGenParallel
SVO-06	TimingGenSVO06
SVP-01-G	TimingGenSVP01