Vialux の DMD を用いた箱型ポテンシャルの構築

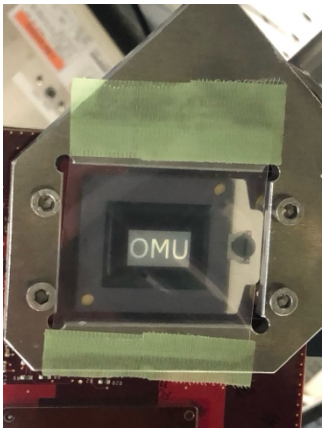
井上慎

2024 年 12 月

1. DMD の概要

1.1 DMD と SLM の比較

DMD(Digital Micromirror Device)は 100 万個以上の小さなミラーをソフトで制御できるデバイスである。DMD を直接原子の上に結像する、もしくは DMD をフーリエ空間で使う、などの方法で任意の形状のトラップを実現することができる。



ビームの整形、という用途では液晶を用いた空間位相変調器(SLM)も有力である。冷却原子の実験でもケンブリッジの Hadzibabic のグループを筆頭に、多くの実験で用いられている。最初に両者の比較をまとめておこう。SLM を冷却原子実験に用いる上で最初に問題になるのは交流ノイズの存在である。液晶に DC 電場をかけると化学的な損傷が発生するため、SLM の駆動には必ず交流電場が使われる。この周波数は装置によって異なるが、数 Hz から 1kHz 程度のことが多い。この周波数は冷却原子のトラップの周波数に近いことが多く、加熱が問題になる。Greiner らは「DMD をフーリエ空間で（ホログラムとして）用いればこの問題を回避できる」と主張して量子気体顕微鏡における DMD の使用を推し進めた。

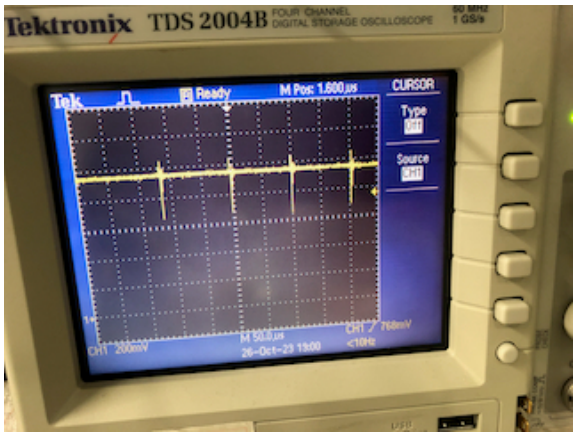
しかし DMD ではミラーの ON/OFF しか制御できないのに対し、SLM では 256 階調が実現可能なので、ホログラム設計の自由度としては SLM の方が秀でている。特にレーザーパワーが限られていて無駄にできない時や、複雑なビーム形状を実現したい時などは SLM が有利になる。実際、量子計算の実験で十分深い光ポテンシャルを何百個も並べたい時は SLM を用いる場合も多い。

本実験では比較的近い波長の光(755nm)で斥力ポテンシャルを作り、BEC を閉じ込める予定なのでレーザーのパワーには余裕がある。さらに光を上から照射して、2 次元箱型ポテンシャルを作るだけなので、穴の空いた形状のビームを作れれば十分である。この場合、

DMD にビームをあて、それをそのまま原子の位置に結像してしまうのが簡単である。Hadzibabic, Moritz, Dalibard らのグループも 2 次元箱型ポテンシャルの生成には DMD のイメージングを採用している。なお、この方式は奥行きが必要な 3 次元箱型ポテンシャルの生成には不向きである。焦点から外れると余計な縞模様などが生じてしまう。このような場合は DMD でも SLM でも良いので、ホログラムを用いてビーム整形することになる。

1.2 Vialux と Texas Instruments の比較

世の中のほとんど全ての DMD チップの供給元はテキサス・インスツルメンツ("TI", https://www.ti.com)であり、TI 自身が"DLP Light crafter"の名前で評価ボードや制御ソフトを比較的安い値段で販売している。安価なのは有難いのだが、TI の評価ボードを用いた場合、"Flicker"と言う上記の SLM に似た問題が発生するのが難点である。具体的には評価ボードの電子回路によって自動的に$100\mu s$に 1 回ミラーがオフになってしまうのである。これはミラーが長い時間同じ位置にいると固着してしまうので、それを予防するためにあらかじめ TI が評価ボードに組み込んだ措置で、ユーザーは簡単に解除できない。ユーザーが強引に回避する手段として、ハンブルグ大学の Kluas Hueck らは制御基盤のある場所を MOSFET で一時的にショートして clock を止める方法を実装し、論文にもしている(K. Hueck, et al. "Suppression of kHz-frequency switching noise in digital micro-mirror devices". Rev. Sci. Inst. 88, 016103 (2017))。我々の研究室でも再現はできたのだが、安定的に動作しなかったため、不採用とした。その後、ハイデルベルグ大学の Heiderberg の Carl Heintze の修論の p.83 でも「改変で制御ボードを損傷した」「改善として、Vialux のモデルを発注した。Vialux の DMD はソフトで数秒間 Flicker を止められる」の記述があったため、Vialux の DMD を発注した。なお、Flicker によって「明るいところは一時的に暗くなる」が「暗いところは暗いまま」なので、斥力ポテンシャルで原子を閉じ込める場合は一瞬壁が消失するだけである。よってそこまで被害が大きくない可能性はある。



$100\mu s$に 1 回消える(Flickering)


MOSFET でショート

1.2 Vialux のモデル選定

我々が冷却原子実験で作りたいトラップの大きさは100μm程度である。よって DMD を縮小して投影するので、最初からピクセルサイズが小さい方が使いやすいこと、それほど画素数はいらないこと、755nm で使うので可視光モデルでも Window の反射率が大丈夫なこと(95%程度)、値段との兼ね合いで V-6501VIS を選択した。仕様は1920 × 1080pixel, 7.6μm ピッチ(鏡の傾きの軸は対角線)なので物理的なサイズは14.5mm × 8.2mmである。コントローラーの型番は V4390 である。

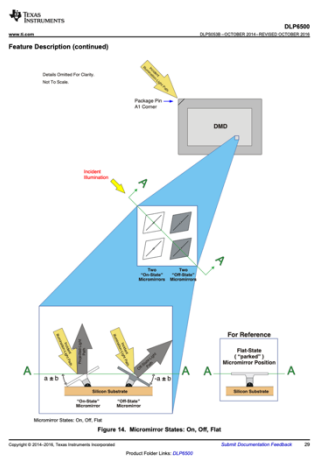
V-6501

Chipset	DLP6500 & DLPC910
DLP Format	0.65" 1 080p
Window Options	VIS
Micromirrors	1 920 x 1 080
Pitch	7.6 µm
DLP Area	14.5 x 8.2 mm²
Controller Board	V4390

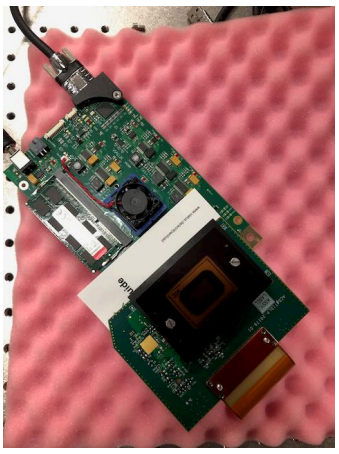
2. DMD の準備

2.1 ホルダーの作成

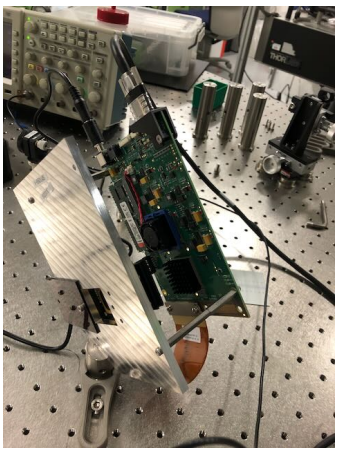
DMD のピクセルは斜め 45 度を軸に傾くので、DMD は定盤に対して 45 度傾けて設置する必要がある。そのためのホルダーを作成する。具体的にはメーカーが提供している "Interface plate"の step ファイルを取り寄せ、寸法を読み出して固定用治具を設計し、発注した。



反射の様子



DMD と基盤



作成した治具

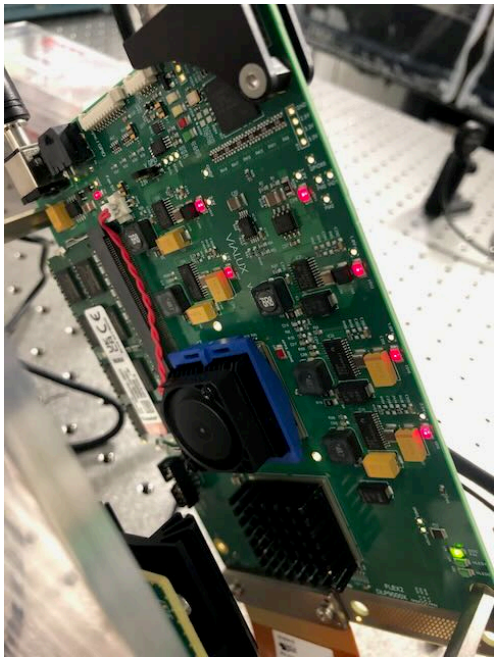
2.2 制御回路関係の準備

制御ボードの動作に必要なものは以下の通り。

・付属の特殊な USB ケーブル(本体はねじ込み、PC 側は USB-A)
・AC アダプタ（5V で 6A 必要、丸文推薦の秋月の gM-111-5, gC-00179 を購入）
・Windows PC
・USB ケーブルの金属部分を定盤に接地した方が動作が安定する。(接地しないとトリガを感知しなかったり誤作動する場合がある。)

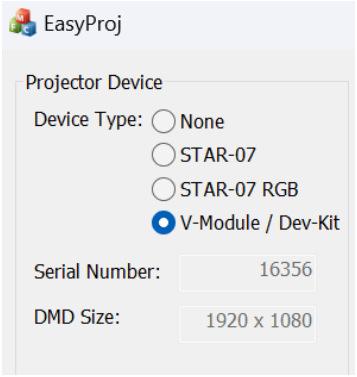
2.3 ソフトの準備
Vialux の HP(Vialux.de)の Support→Download で V-6501 の ALP-4.3 をダウンロードする。

3. 付属ソフトを用いた動作確認
3.1 まず、付属のソフト"EasyProj"で動作確認を行う
(1)電源を入れると回路基盤の赤い LED が 8 個くらい点灯する
(2)USB を繋ぐと USB ケーブルの近くの 3 個の LED が青黄赤に点灯する


電源を繋いだとき

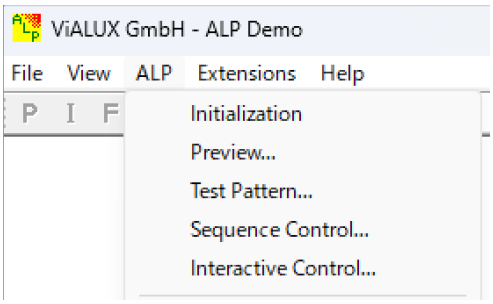

USB を繋いだ時

(3)Windows で全てのアプリ→ALP-4.3→EasyProj を開く。
(4)Projector Device→V-module を選択する。通信できていれば Serial Number(16356)、DMD Size(1920 x 1080)が表示される。
(5)Load で 1920x1080 の bitmap イメージをロードする。イメージのビット深さは何でも良い。
(6)Projection ボタンを押す。本体を覗き込み、イメージが投影されているのを確認する。
(7)Stop を押して終了、Free を押して DMD との通信を解放し、Quit でプログラムを終了する。



Serial Number まで認識するのに、イメージが投影されない場合は往々にして発生する。根本的な原因は分かっていない。ソフトを立ち上げ直す。それでもダメなら PC から USB を抜き、DMD 本体の電源を入れ直し、PC を立ち上げ直す、などを試しているうちに動作する場合もある。ただし、本体の温度が上がっている場合は立ち上げ直しを繰り返してもうまくいかない。"ALP Demo"を用いて PCB の温度を確認する。

3.2 "ALP Demo"を用いた基盤温度確認
(1) "ALP Demo"を立ち上げる
(2) ALP Initializion で ok
(3) Help>About ALP Device で PCB の温度が表示される。通常使用で 34 度程度。
(4) 終わったら ALP Initializing を再度クリックして DMD を通信から解放する。

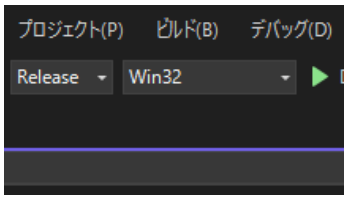


既存ソフトを使用して安定的な動作環境を確立に努める。

4. オリジナルソフトの作成
4.1 Microsoft Visual Studio のインストール
プログラムは C++で書かれているので Microsoft Visual Studio をインストールする。Visual Studio(Community Edition)は Microsoft のサイトから無料でダウンロードできる。

4.2 コンパイル環境の構築
外部のファイルにどのように依存するかを解明するのが面倒だが、幸い Vialux が Sample プログラムの多くでコンパイル前の.sln ファイルを提供してくれている。その一つを流用する。
具体的には
・Samples フォルダ(Program Files > ALP-4.3 > ALP-4.3API > Samples)にあるフォルダの

中で.sln ファイルが含まれているもの(例えば「BlackWhiteSample フォルダ」など)を見つけ、フォルダごと複製して新しい名前をつける
・複製したフォルダの中にある.sln をダブルクリック→「異なる資格情報でこのアプリケーションを再起動」→「このアプリがデバイスに変更を加えることを許可しますか？」で「はい」を選択→もう一度同じ sln を選択



→上部の drop box で「release」「Win32」を選択したのち、「ビルド」→「ソリューションのリビルド」でコンパイルを行う。なお、ここで「Win32」を選択するのは、用いる dll である「alp4395.dll」が Win32 用だからである。「x64」でもコンパイルできるが exe ファイルが alp4395.dll を読み込めない。

→"release"フォルダの中に exe ファイルが作成されている。同じ directory に alp4395.dll を持ってくれば動作するはず。

5. 自作プログラムによる flicker の回避
5.1 トリガが来るたびに同じイメージを 1 秒ぐらい表示させるプログラムの作成
元々Vialux の DMD を購入した理由は flicker が 10 秒に渡り回避できることにあった。しかし上記のプログラムでは連続投影を意味する"AlpProjStartCont"が使われているため、基盤のクロックのタイミングでランダムに flicker が入るのを回避できない。flicker 回避のためには投影の時間を決め、trigger でスタートする必要がある。どう変えれば良いかを知るために ALP-4.3>ALP-4.3 API フォルダにある"ALP-4.3 API description.pdf"を開くと
(1)トリガの受け付け：AlpProjControl(p.25)で ALP_SLAVE mode を指定する
(2) 1 回の表示時間の指定：AlpSeqTiming(p.15)で PictureTime をμs単位で$10^7 \mu s = 10$ sまで指定可能
(3) 繰り返し回数の設定：本当は無限回を指定し、やめたくなった時に中断するのが理想だが、「無限回」の指定ができない。よって、AlpSeqControl(p.12)で大きな数(最大約 10 万回)を指定しておく。開始後はキーボードの入力待ちとし、キーボードに入力があったら AlpDevHalt, AlpDevFree を実行する形をとる。
(4)投影開始の命令：AlpProjStart(p.29)を使用する。
まとめると、投影のタイミングに関する部分は以下のようになる：

```
if (ALP_OK != AlpProjControl(nDevId, ALP_PROJ_MODE, ALP_SLAVE))
        {
                printf("error(AlpProjControl)¥r¥n");
                return 1;
        }
```

```
        if (ALP_OK != AlpSeqControl(nDevId, nSeqId, ALP_SEQ_REPEAT, 1000000))
        {
                printf("error(AlpSeqControl)¥r¥n");
                return 1;
        }


        if (ALP_OK != AlpSeqTiming(nDevId, nSeqId, ALP_DEFAULT, PicTime,
                ALP_DEFAULT, ALP_DEFAULT, ALP_DEFAULT))
        {
                printf("error(AlpSeqTiming)¥r¥n");
                return 1;
        }

        if (ALP_OK != AlpProjStart(nDevId, nSeqId))
        {
                printf("error(AlpProjStartCont)¥r¥n");
                return 1;
        }

        //WaitForKeyStroke();
        printf("The program will accept down edges for million times. Press any key
to stop ALP projection..¥r¥n");
        do { _getch(); } while (_kbhit());
        //Done
        AlpDevHalt(nDevId);
        AlpDevFree(nDevId);
```

ここで、"PicTime"はユーザーが別に指定した値(int)である。

5.2 コマンドラインで指定した BMP ファイルを読み込む

5.2.1 毎回 C++で表示したいイメージを作成していては大変である。他のソフト（例えば MATLAB）などで BMP ファイルを作成し、C++のプログラムに読み込んで表示できた方が良い。そのようなプログラムを作成する。

5.2.2. BMP ファイルのビット深さ

実験で使うのはミラーのオンとオフだけであり、「必要なビットの深さは 1 」である。しかし BMP ファイルは 1byte=8bit ごとに情報をまとめてしまう規格である。よって「ビット深さ 1 」で BMP ファイルを作ると情報は 1 回 8bit ごとにまとめられ、また C++のプログラムでそれを分解してメモリにロードし直すコードを書かなければならない。この際にエラーが起こる可能性を取り除くため、

・BMP ファイルのビット深さは 1byte=8bit とする

・「0 が書き込まれたピクセルはミラーオフ、それ以外はオン」と解釈する

とした。

5.2.3 プログラムの作成

ChatGPT で

「c++で深さ 8 bit の白黒ビットマップファイルで横 1920px、高さ 1080px のファイルだけ開くプログラムを書いて」

「読み取ったデータは変数"bImage"に格納して」

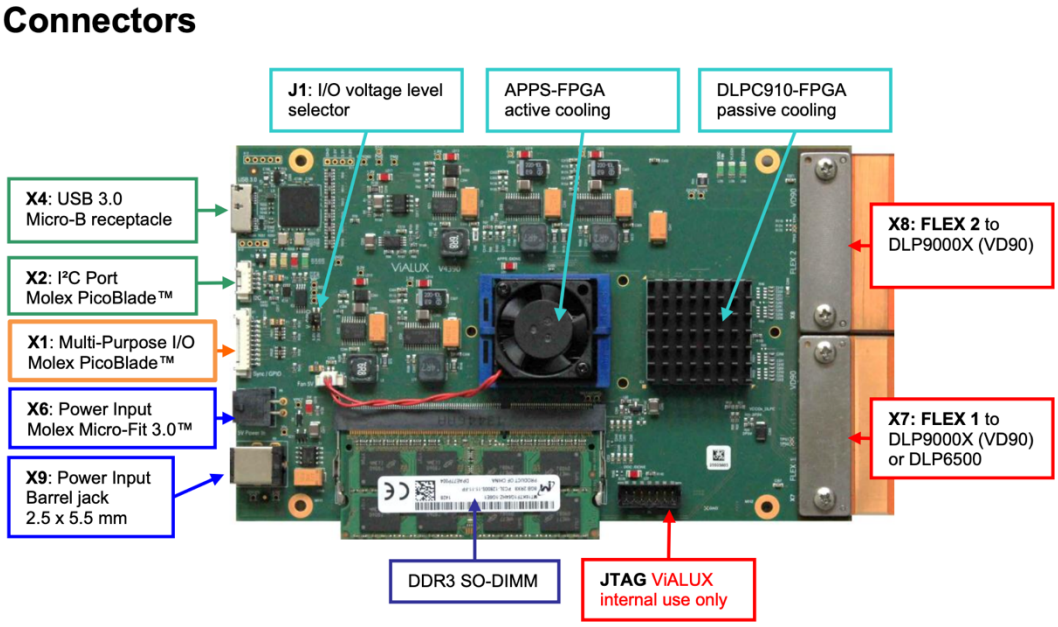
「ファイル名はコマンドラインから入力したいです」

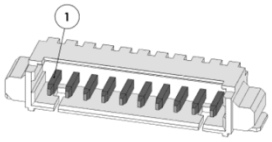
と命令して作成した。最終的なプログラムは付録を参照のこと。

5.3 トリガの配線

5.3.1 コネクタと電圧レベル

X1 コネクタの 7 番ピンからトリガを入力する。電圧レベルは 2.5V と 5V から選べる。Jumper J1 で 3 番と 4 番を繋いで 5V を選択する。トリガーのデフォルトは Falling Edge。変更したい場合は AlpDevControl で指定する。





X1: Multi-Purpose I/O connector

The connector is a Molex PicoBlade™ header, part number 53261-1071. It mates with Molex part number 51021-1000. Use crimp contacts 50058-8000 or 50079-8000.

For your convenience the package includes a set of prepared cables and a connector housing (see picture left).

Pin	Signal Name	Direction	Group	Purpose
1	VCC_GPIO	IN/OUT		I/O Voltage (see configuration of jumper J1)
2	SYNCH_OUT1_GATE	OUT	A	Dynamic Gate Synch Output 1, refer to the ALP-4 API description, ALP_DEV_DYN_SYNCH_OUT1_GATE
3	SYNCH_OUT2_GATE	OUT	A	ALP_DEV_DYN_SYNCH_OUT2_GATE
4	SYNCH_OUT3_GATE	OUT	B	ALP_DEV_DYN_SYNCH_OUT3_GATE
5		OUT	B	Reserved for future use
6		IN	C	Reserved for future use
7	TRIGGER_IN	IN	C	Trigger Input
8	SYNCH_OUT	OUT	D	Frame Synchronization Output
9	PWM	OUT	D	ALP_PWM_LEVEL: output a pulse-width modulated value
10	GND	–		Power ground

Jumper J1: I/O voltage selection

Jumper J1 has 1.27 mm pitch and is used to select either 2.5 V or 5.0 V voltage supply for the Multi-Purpose I/O connector.

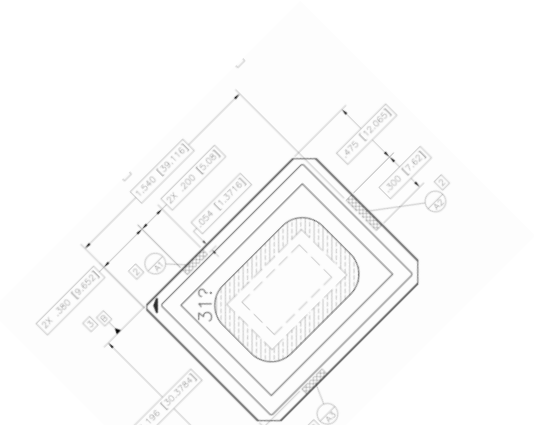
The attached hardware can supply the voltage level, when the jumper is left open. VCC_GPIO is connected to pin 1 of X1 (Multi-Purpose I/O) via a resettable fuse.

Position	0R (0603)	I/O voltage	VCC_GPIO source
open	-	2.5 V to 5.0 V	external (X1.1)
1 – 2	R184	2.5 V	Internal
3 – 4	R186	5.0 V	Internal

6. 光の入れ方と回折効率

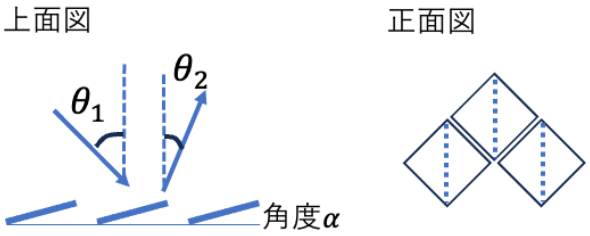
6.1

「31?」とマーキングのある方向から光を入射する。鏡が±12 度傾くので、基本的には正面から 24 度、「31?」の向きに傾いた方向から光を入れ、「ON」の時に正面に反射させるようにして用いる。そうすると以降の Optics で収差を小さくできる。



6.2 回折効率の計算

基本的には$\theta_1 = 2\alpha = 24^\circ$かつ$\theta_2 = 0^\circ$で使いたいのだが、波長によっては隣り合うピクセルからの反射が弱めあってしまい、回折効率が下がってしまう。波長をλ、ピクセルのピッチを$d = 7.6\,\mu m$とすると、その条件は回折の次数



$$n = 2\alpha \frac{d}{\sqrt{2}\lambda}$$

を計算することで分かる。nが半整数$(= 2.5, 3.5, \cdots)$だと回折効率が悪いが、整数に近ければ心配ない。例えば$\lambda = 750nm$の場合、

$$n = 0.419(\mathrm{rad}) \frac{7.6\ \mu\mathrm{m}}{\sqrt{2} \times 0.750\mathrm{nm}} \sim 3.003$$

なので全く問題ない。

7. 日々の運用手順の実際

すでに第 5 章でコンパイルした exe ファイル(ShinDMD.exe)があるとして、使い方を説明する。

(1)ファンジェネを用意し、5V のトリガ信号を出せるようにする。

(2)MATLAB などで投影したい bitmap ファイルを作る。大きさは 1920x1080、深さ 8 ビット。off にしたいピクセルにゼロを入れる。ゼロ以外の値は on になる。

(3) ALP-4.3 API のフォルダの中に新しいフォルダ（例えば、Shin_BlackWhiteSample）を作り、用意したオリジナルソフト（ShinDMD.exe）と alp4395.dll と bitmap イメージファイル(Shin100.bmp)を置く。

(4)コマンドプロンプトを開く（「ターミナル」や PwerShell は使っていない。セキュリティ関係で文句が出る。）

(5)(3)のフォルダに移動する

(6)トリガが来たら 1 秒間=1,000,000 マイクロ秒表示させたい場合、次のように入力する
>ShinDMD.exe　Shin100.bmp 1000000
と入力してリターン。
(7)5V の trigger を入れると falling edge で表示が開始する。

8. 複数のイメージの切り替え
8.1 付属のソフト使用
付属のソフト"EasyProj.exe"で複数のイメージをロードし、"Illuminate Time"と"Picture Time"を両方 1,000,000 μsに設定すると、"Illuminate Time"が 999,903 μs に訂正された上で 1 秒ごとに 2 つのイメージが切り替わって表示される。これは、2 つのイメージの間に 97 μsの dark time が必要、ということを意味している。

8.2 複数のイメージをトリガーで切り替えたい場合
" ALP-4.3 API description.pdf "の p.50 にある"3.8 Externally Triggered Frame Transition"によれば、同じく X1 conector の Pin7 に外部トリガーを入力することで、frame 間のトリガーをすることが可能である。
・AlpProjControl において、
　ALP_PROJ_MODE = ALP_SLAVE（←すでに使っている）
　ALP_PROJ_STEP = ALP_LEVEL_HIGH、もしくは ALP_EDGE_RISING
などとすれば良いらしい。

9. MATLAB で半径 150pixel の BMP ファイルを作成するサンプルプログラム

```
p=150;q=1920;r=1080;
[x,y] = meshgrid(1:q,1:r);
z=((x-960).*(x-960)/p/p+(y-540).*(y-540)/p/p-1);
z=1.-1./(1+exp(z));
z=round(z);
filename=['ShinDMD_',num2str(p),'.bmp'];
imwrite(z,filename);
```

1920x1080 の bmp ファイルで大きさ約 2MB のものが生成できていれば ok。

（以上）