独習アルテラ OpenCL 基礎演習
（Atlas-SoC ボード編）

ver.15

2016 年 7 月 Rev.1

独習アルテラ OpenCL 基礎演習
（Atlas-SoC ボード編）

目次

1. はじめに

この資料は、アルテラ SoC を利用して OpenCL™ による FPGA ロジックの高位合成を行う方法を解説します。

アルテラが提供する SDK for OpenCL を使用した演習を体験することで、迅速かつ簡単にハードウェア・アクセラレータを構築する方法を理解することができます。演習では単純な配列加算の処理を OpenCL で実装します。

1-1. 演習環境

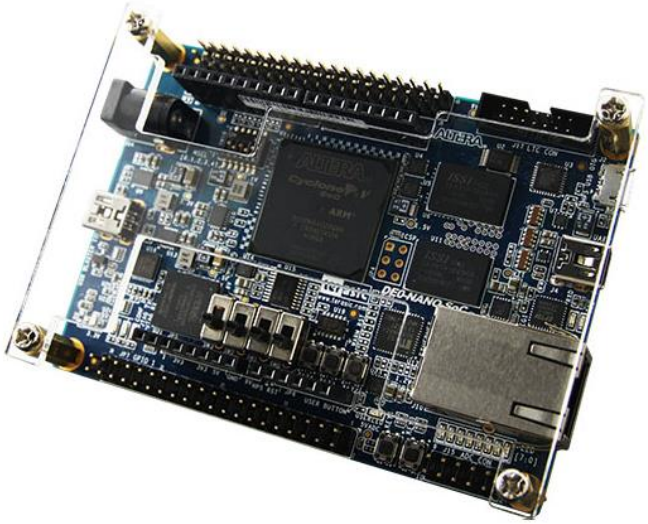
本演習では下記の開発環境を使用します。

【表 1-1.1】この資料の説明で使用している主な開発環境

項番	項目	内容
1	ホスト PC	Microsoft® Windows® 7 Professional sp1（64 bit）搭載の 64 bit マシン
2	アルテラ Quartus® Prime 開発ソフトウェア	アルテラ SoC FPGA のハードウェアを開発するためのツールです。 ソフトウェア開発に必要なハンドオフ・ファイルの生成も行います。 本説明書では、Quartus Prime 開発ソフトウェア・ライト・エディション v15.1 を使用しています。 ■ アルテラ Quartus Prime 開発ソフトウェア・ライト・エディション v15.1 http://dl.altera.com/15.1/?edition=lite
3	アルテラ SoC EDS（ARM® DS-5™ Altera Edition）	アルテラ SoC FPGA のソフトウェアを開発するためのツールです。 ホスト・アプリケーション・ソフトウェアをコンパイルすることができます。 本説明書では、アルテラ SoC EDS v15.1.1 を使用しています。 ■ アルテラ SoC エンベデッド・デザイン・スイート v15.1.1 http://dl.altera.com/soceds/15.1.1/?edition=standard
4	アルテラ SDK for OpenCL	アルテラ FPGA で OpenCL ソフトウェアを開発するためのツールです。 OpenCL カーネルをコンパイル・実行することができます。 本説明書では、アルテラ SDK for OpenCL v15.1 を使用しています。 ■ アルテラ SDK for OpenCL v15.1 http://dl.altera.com/opencl/15.1/?edition=standard
5	Atlas-SoC ボード	本資料の説明でターゲット・ボードとして使用する、アルテラ Cyclone® V SoC を搭載した Terasic Atlas-SoC ボードです。 ■ Atlas-SoC ボード http://rocketboards.org/foswiki/view/Documentation/AtlasSoCDevelopmentPlatform
6	演習デザイン・ファイル	本資料の演習で使用する、Atlas-SoC 上で動作する、単純な配列加算の処理を OpenCL で実装します。 ホスト・アプリケーションでは入力テーブルの確保と乱数の格納を行い、カーネル（FPGA ロジック）で演算された加算値が正しいかを比較します。 付属ファイルの Lab フォルダ の下に、Altera-SoCFPGA-OpenCL-vectorAdd.tar.gz があります。
7	UART ターミナル・ソフト	本演習では Tera Term を使用しています。 ダウンロードやインストールにつきましては、別途、作成者または関連サイトの情報を参考にしてください。

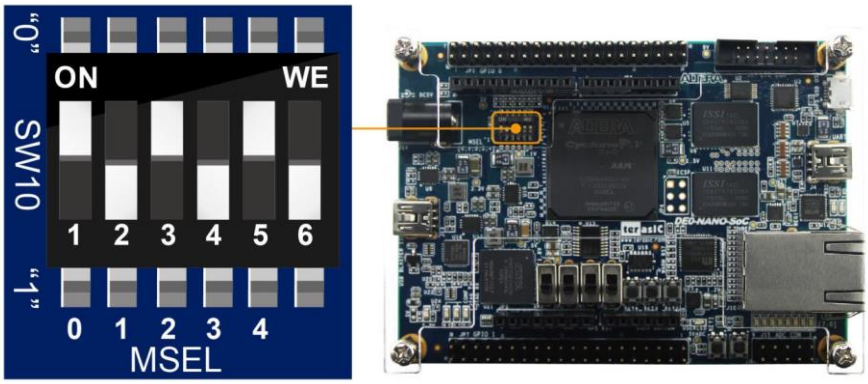
1-2.“Atlas-SoC”キット

このキットは、Cyclone V SoC デバイスが搭載されています。



【図 1-2.1】“Atlas-SoC”キット

(1) Atlas-SoC ボードのディップ・スイッチ設定を確認します。



【図 1-2.2】SW10 設定

【表 1-2.1】SW10 設定表

ボード・リファレンス	信号名	内容	設定
SW10-1	MSEL 0	これらのピンは、FPGA のコンフィギュレーション方法の選択に使用します	ON
SW10-2	MSEL 1		OFF
SW10-3	MSEL 2		ON
SW10-4	MSEL 3		OFF
SW10-5	MSEL 4		ON
SW10-6	N/A	N/A	N/A

1-3. 付属ファイルの内容

独習アルテラ OpenCL 基礎演習（Atlas-SoC ボード版）の付属ファイル
Self-study_Altera_OpenCL_basic_v15.1 for Atlas-Soc.zip を解凍すると、以下のものが含まれています。

■ **BSP_Atlas フォルダ**（ボード・テンプレート）

- board

  c5soc_atlas

  arm32 ： ホスト・アプリケーション用ライブラリ等

  c5soc_atlas_sharedonly ： ボード・テンプレート・プロジェクト

  driver

  board_env.xml

- share

■ **Lab フォルダ**（演習ファイル）

- Altera-SoCFPGA-OpenCL-vectorAdd.tar.gz

■ **SD_Image フォルダ**（OpenCL 用 SD カード・イメージ）

- atlas_opencl_sdimage_v1511_r3.tgz

■ README.txt（アルテラ開発ツール 入手先 URL）

2. デザイン・フロー

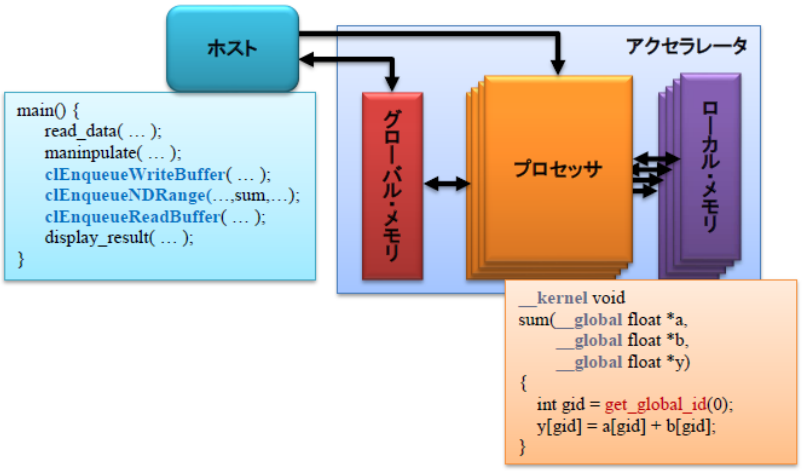
アルテラの SDK for OpenCL は、エンベデッド ARM Cortex-A9 プロセッサ・コアをホストとして、ANSI C ベースの言語である OpenCL C で記述したソフトウェア関数（カーネル）を、FPGA デザイン（ハードウェア・アクセラレータ）に合成し、実行することができます。

この章では、OpenCL 開発における、ホストとカーネル、FPGA デザインへの合成などデザイン・フローを理解することができます。

2-1. プログラミング・モデル

2-1-1. 一般的なプログラミング・モデル

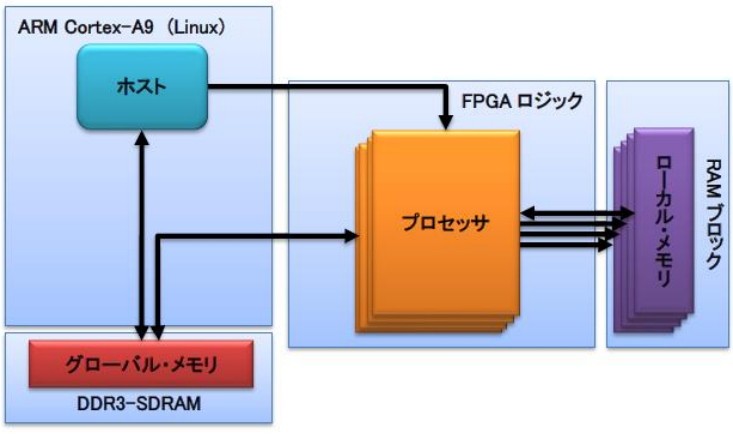
OpenCL の一般的なプログラミング・モデルを下図に示します。
ホストでは、グローバル・メモリを確保し、プロセッサで処理するデータを準備し、プロセッサに制御を移します。
プロセッサはそのデータを処理した後、再びホストに制御を戻し、処理が完了したことを通知します。



【図 2-1-1.1】一般的なプログラミング・モデル

2-1-2. アルテラ SDK for OpenCL での実装モデル

アルテラ SDK for OpenCL では、ホストを ARM Cortex-A9 プロセッサ・コアとして、グローバル・メモリを DDR3-SDRAM、プロセッサを FPGA ロジック、ローカル・メモリは FPGA 内の RAM ブロックに実装されます。



【図 2-1-2.1】アルテラ SDK for OpenCL での実装モデル

2-2. 開発フロー

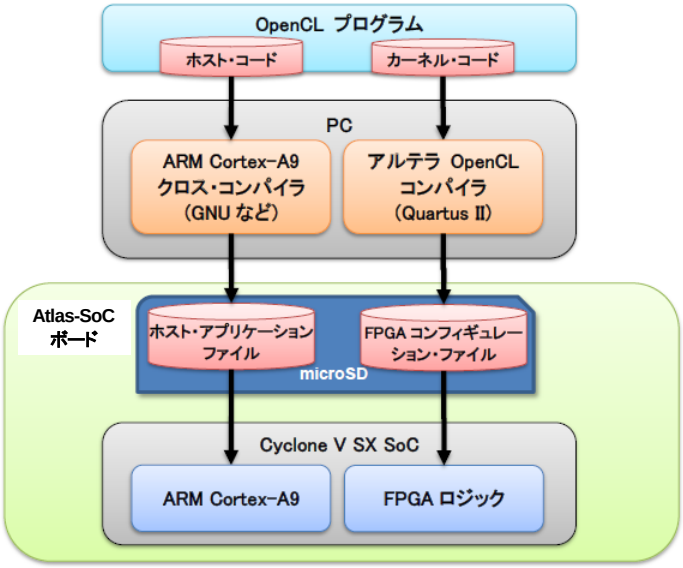
まず、OpenCL プログラムを記述します。 OpenCL プログラムには FPGA に実装する部分を記述したカーネル・コードと、そのカーネルを制御するホスト・コードを記述します。

カーネル・コードは OpenCL C で記述しますので、そのままではプロセッサで動作させることはできません。最初にカーネル・コードを標準の C 言語で記述し、正しく動作することを確認した上で OpenCL C に変換することを推奨します。

続いて、カーネル・コードを FPGA デザインに合成、コンパイルし、.aocx（Altera Offline Compiler eXecutable file）を生成し、ホスト（Linux）に転送します。

ホスト・コードも、 PC のクロス・コンパイラ環境で実行可能なホスト・アプリケーションにコンパイルし、ホストに転送します。

ホスト・アプリケーションを実行すると、生成された .aocx ファイルで FPGA をコンフィギュレーションし、カーネル・コードに記述された機能と同等の処理が FPGA ロジックを使用して実行されます。



【図 2-2.1】アルテラ SDK for OpenCL で開発フロー

3. 事前準備

この章では、演習の実施に必要な機材を確認し、ツールをインストールします。

なお、後述のツールを既にインストール済みの場合は、新たにインストールする必要はありませんので読み飛ばしてください。

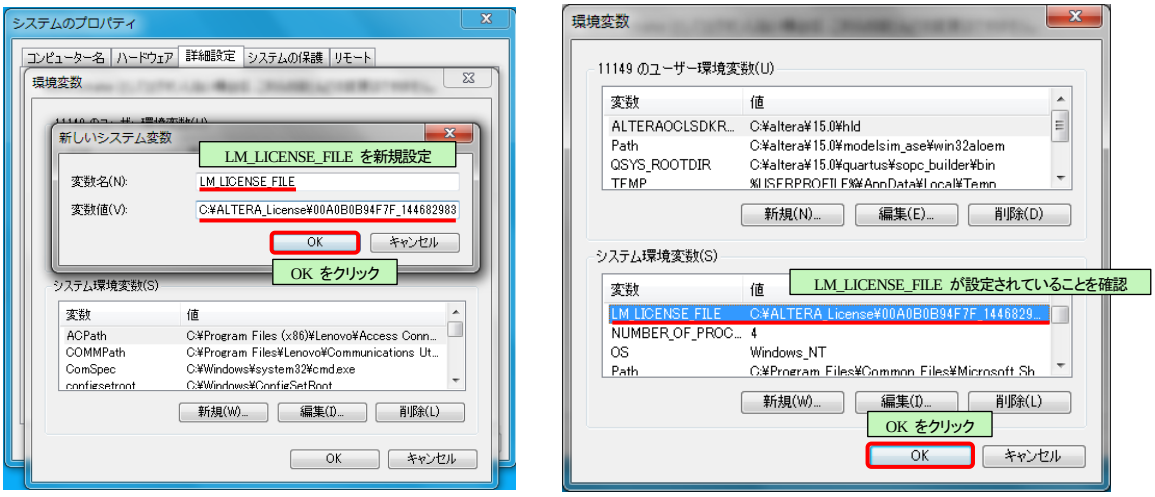
3-1. Quartus Prime 開発ソフトウェアのインストール

(1) 今回は、アルテラのサイトから、Quartus Prime v15.1（Windows 版）をダウンロードしインストールします。（有償ライセンスが必要なスタンダード・エディションでも、無償のライト・エディションでも構いません。）

(2) Windows の環境変数を開いて、**LM_LICENSE_FILE** を追加し、**OpenCL 評価用ライセンス・ファイル** のパスとファイル名を設定します。

［注意事項］

✓ **OpenCL 評価用ライセンス・ファイル** は、別途お客様担当の代理店にお問い合わせください。



【図 3-1.1】LM_LICENSE_FILE 環境変数の設定

3-2. SoC EDS のインストール

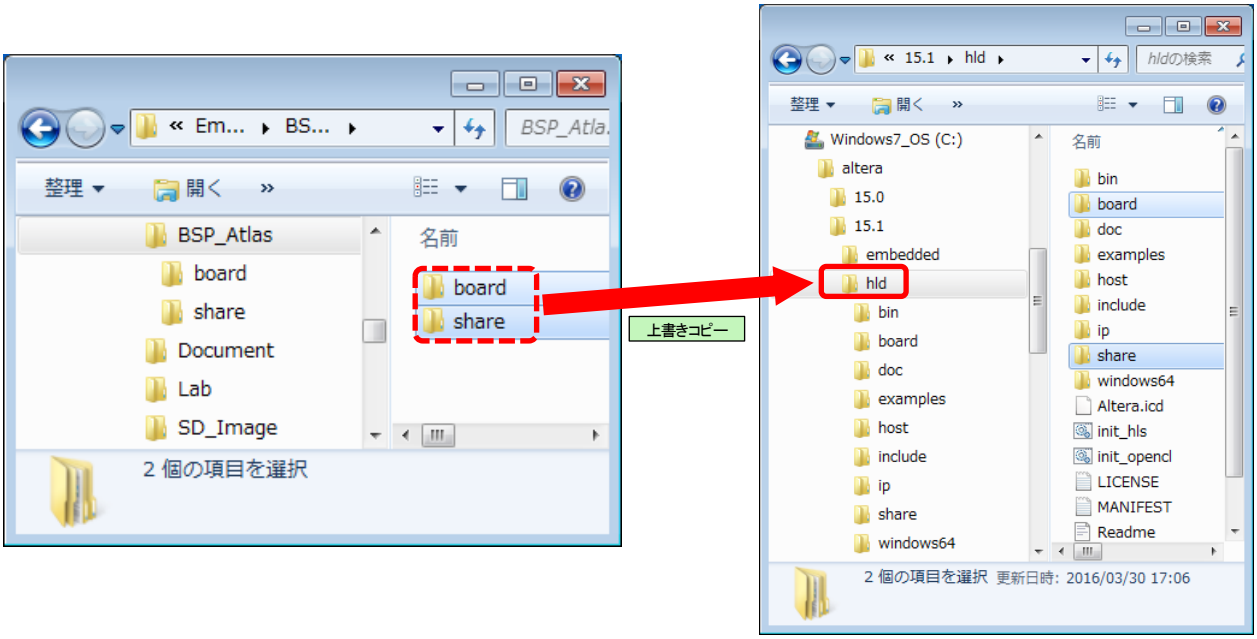
アルテラのサイトから、SoC EDS v15.1.1（Windows 版）をダウンロードしインストールします。

3-3. SDK for OpenCL のインストール

アルテラのサイトから、SDK for OpenCL v15.1（Windows 版）をダウンロードしインストールします。

3-4. v15.1 用 Atlas-SoC ボード・パッケージ（ボード・テンプレート）の追加

付属ファイルの“BSP_Atlas”フォルダ の中身（board と share）を、SDK for OpenCL v15.1 インストール・フォルダ（例: C:¥altera¥15.1¥hld ）の下に上書きコピーしてください。



【図 3-4.1】Atlas-SoC ボード・パッケージ（ボード・テンプレート）の追加

これにより board フォルダの下に、c5soc_atlas フォルダが追加され、share¥models¥dm フォルダの下に、5csema4u23c6n_dm.xml ファイルが追加されます。

3-5. Windows 環境変数の確認・設定

Windows の環境変数を開いて下記の変数を追加してください。

(1) 以下の環境変数が存在し正しい値が設定されていることを確認します。

- 変数: ALTERAOCLSDKROOT
  値: （例: C:¥altera¥15.1¥hld ）
- 変数: QUARTUS_ROOTDIR
  値: （例: C:¥altera¥15.1¥quartus ）

(2) 以下の環境変数を追加します。

- 変数: AOCL_BOARD_PACKAGE_ROOT
  値: %ALTERAOCLSDKROOT%¥board¥c5soc_atlas
- 変数: PATH
  値: %QUARTUS_ROOTDIR%¥bin64;%ALTERAOCLSDKROOT%¥windows64¥bin

3-6. 演習ファイルの抽出

付属ファイル内の “Lab” フォルダ の中にある **Altera-SoCFPGA-OpenCL-vectorAdd.tar.gz** ファイルを PC の任意のワーク・フォルダ（この例では、C:¥Work）にコピーして解凍します。

解凍は、エンベデッド・コマンド・シェル からコマンドラインで実行できます。

Windows エクスプローラで、SoC EDS のインストール・フォルダ（この例では、C:¥altera¥15.1¥embedded）を開き、その下にある **Embedded_Command_Shell.bat** をダブルクリックしてエンベデッド・コマンド・シェルを起動し、以下のコマンドで Altera-SoCFPGA-OpenCL-vectorAdd.tar.gz ファイルを解凍します。

```
$ cd "C:¥Work"
$ tar -xzf Altera-SoCFPGA-OpenCL-vectorAdd.tar.gz
```

［注意事項］

✓ コピー＆解凍先のフォルダ・パス名にスペースや日本語（全角文字）が無いよう注意してください。

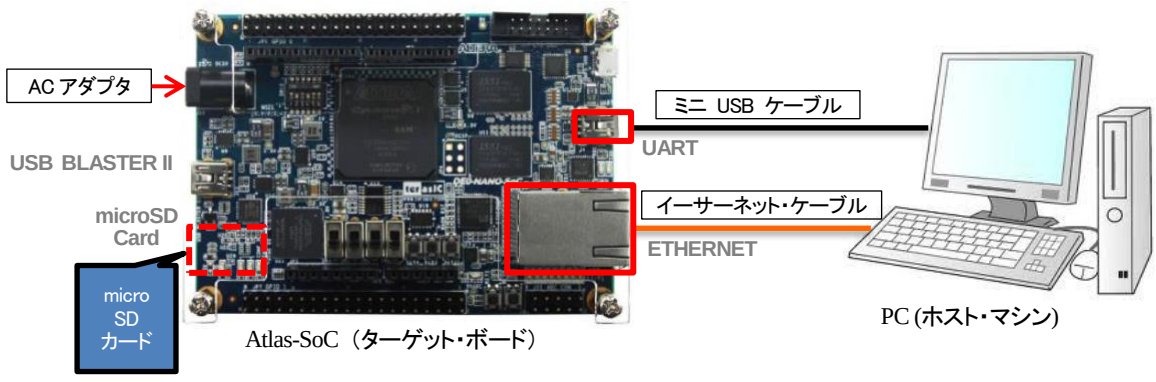
3-7. その他ツールのインストール

UART ターミナル・ソフトをインストールします。本資料では Tera Term を使用していますが、機能的に同等であれば別のツールを使っても構いません。

なお、ツールのダウンロードやインストールにつきましては、別途、作成者または関連サイトの情報を参考にしてください。

3-8. Atlas-SoC と PC の接続

PC と Atlas-SoC を接続し、microSD カードをカード・スロットに挿入し、電源を入れてください。



【図 3-8.1】Atlas-SoC と PC の接続

［注意事項］

✓ 付属ファイル内の “SD_Image” フォルダ の下に

atlas_opencl_sdimage_v1511_r3.tgz

ファイルがありますので、これを エンベデッド・コマンド・シェル から `tar -xzf` コマンドで解凍し、フリー・ソフト Win32DiskImager を使用して microSD カードに書き込みます。

3-9. PC 側ネットワークの設定

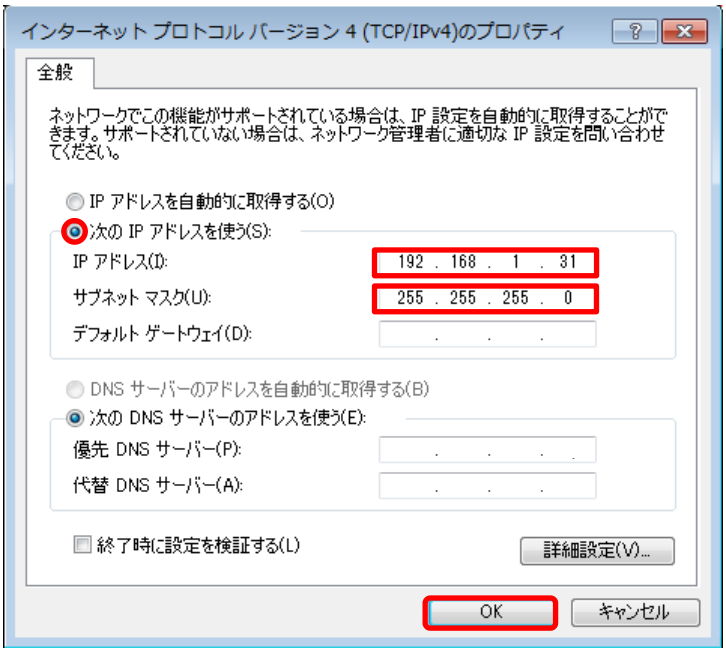
Atlas-SoC と PC を Ethernet で接続するための設定を行います。

(1) Windows のスタート・メニューから「**コントロールパネル**」を選択し、「**ネットワークと共有センター**」をクリックします。

(2) 「**アダプタの設定の変更**」を開いて、「**ローカルエリア接続**」を選択・右クリックして「**プロパティ**」を選択します。

(3) プロパティ・ダイアログが表示されたら、「**インターネットプロトコルバージョン 4 (TCP/IPv4)** 」を選択し、「**プロパティ**」ボタンをクリックします。

(4) 「**次の IP アドレスを使う**」にチェックを入れ、“**IP アドレス**”と“**サブネット マスク**”に任意のアドレスを入力します。この例では、IP アドレス : 192.168.1.31、サブネットマスク : 255.255.255.0 に設定します。

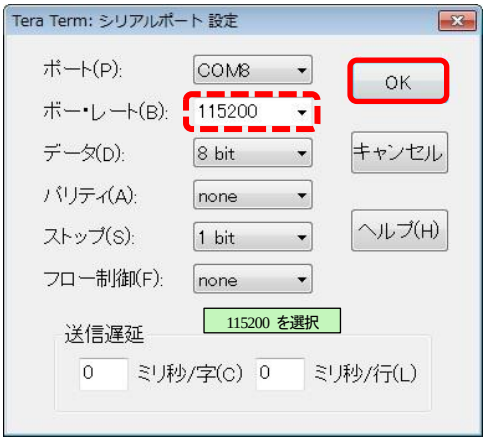


【図 3-9.1】PC 側ネットワークの設定

3-10.UART の接続確認と Linux の起動

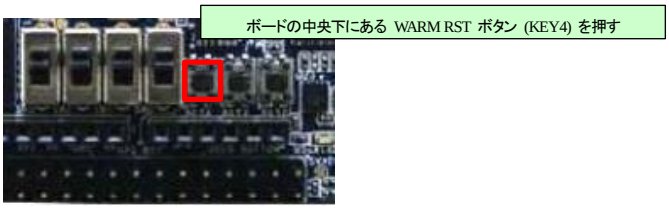
(1) Atlas-SoC と PC が UART ケーブルで接続されていることを確認します。

(2) Atlas-SoC に接続されている COM ポートを選択し「**ボーレート**」を “**115200**”に設定します。



【図 3-10.1】Tera Term でのシリアル・ポート設定

(3) Atlas-SoC の WARM RESET ボタンを押下します。

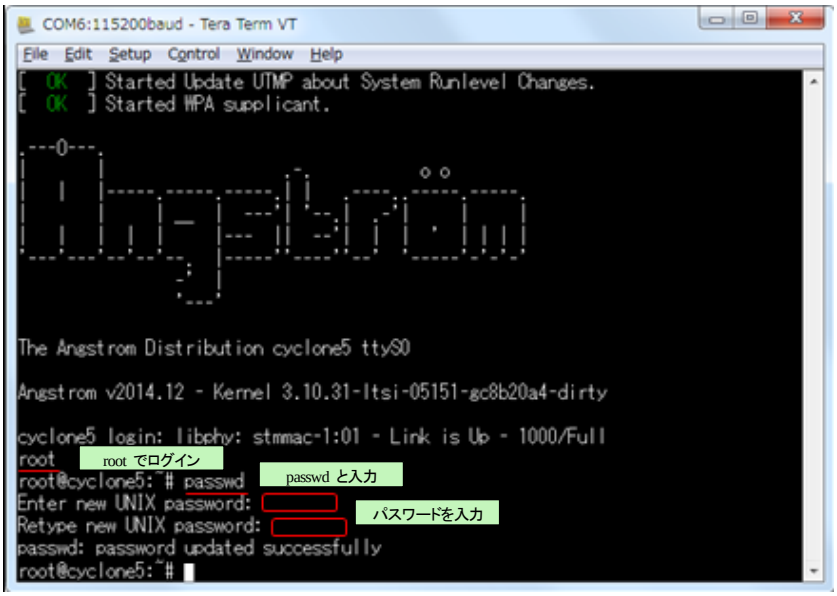


【図 3-10.2】Atlas-SoC の WARM RESET ボタンを押下

(4) Linux コンソールに Linux の起動ログが表示されます。ログインするには、login プロンプトに `root` ↵ と入力します。

(5) `passwd` ↵ と入力して root のパスワードを設定します。

(6) 新規パスワードが要求された場合は、この例では `altera.123` ↵ を入力した後、もう一度 `altera.123` ↵ を入力します（入力の際、パスワードは表示されません）。



【図 3-10.3】ログインとパスワードの設定

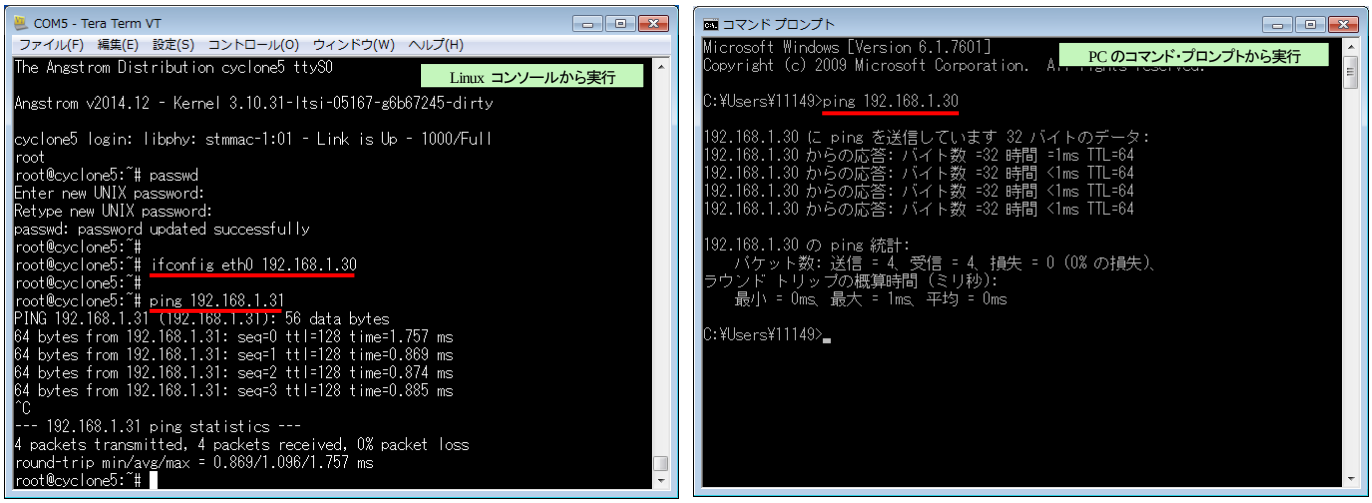
3-11.Ethernet の接続確認

Atlas-SoC と PC が Ethernet ケーブルで接続されていることを確認します。

(1) Linux コンソールから 任意の IP アドレスを設定します。この例では、192.168.1.30 を設定しますので、以下のように入力してください。

```
# ifconfig  eth0  192.168.1.30
```

(2) 続いて、Linux および PC 双方から Ping を送り、正しく接続できていることを確認します。その際、どちらかの Ping が通らない場合は Windows のファイア・ウォール設定を解除するなどしてください。



【図 3-11.1】Ethernet の接続確認

4. 演習

演習には、前述「3-6. 演習ファイルの抽出」で解凍した Altera-SoCFPGA-OpenCL-vectorAdd を使用します。単純な配列加算の処理を OpenCL で実装します。

ホスト・アプリケーションでは入力テーブルの確保と乱数の格納を行い、カーネル（FPGA ロジック）で演算された加算値が正しいかを比較します。

4-1. エンベデッド・コマンド・シェルの起動

後ほど実行する、「カーネルのコンパイル」と「ホスト・アプリケーションのコンパイル」は、このエンベデッド・コマンド・シェルからコマンドラインで実行します。

(1) Windows エクスプローラで、SoC EDS のインストール・フォルダ（この例では、C:¥altera¥15.1¥embedded）を開き、その下にある **Embedded_Command_Shell.bat** をダブルクリックしてエンベデッド・コマンド・シェルを起動します。

(2) エンベデッド・コマンド・シェルから 以下を実行します。

```
$ export  ALTERAOCLSDKROOT="C:¥altera¥15.1¥hld"
$ export  PATH=${ALTERAOCLSDKROOT}/bin:$PATH
$ export  AOCL_BOARD_PACKAGE_ROOT=${ALTERAOCLSDKROOT}/board/c5soc_atlas
```

(3) 現在認識されているボードを確認します。

```
$ cd  "C:¥altera¥15.1¥hld¥bin"
$ aoc  —list-boards
Board list:
  c5soc_atlas_sharedonly
```

4-2. カーネルのコンパイル

カーネルのコンパイルでは、バック・グランドで Quartus Prime 開発ソフトウェアが実行されますので、Quartus Prime 開発ソフトウェアが起動中であれば終了させてください。

(1) エンベデッド・コマンド・シェルから 以下のコマンドを入力します。

```
$ cd "C:¥Work¥Altera-SoCFPGA-OpenCL-vectorAdd¥device"
$ aoc -v --report vectorAdd.cl -o vectorAdd.aocx --board c5soc_atlas_sharedonly
aoc: Environment checks are completed successfully.
You are now compiling the full flow!!
aoc: Selected target board c5soc_atlas_sharedonly
aoc: Running OpenCL parser....
c:/Work/Altera-SoCFPGA-OpenCL-vectorAdd/device/vectorAdd.cl:23:47: warning: decl
aring kernel argument with no 'restrict' may lead to low kernel performance
__kernel void vectorAdd(__global const float *x,
                                              ^

c:/Work/Altera-SoCFPGA-OpenCL-vectorAdd/device/vectorAdd.cl:24:47: warning: decl
aring kernel argument with no 'restrict' may lead to low kernel performance
                        __global const float *y,
                                              ^

2 warnings generated.
aoc: OpenCL parser completed successfully.
aoc: Compiling....
aoc: Linking with IP library ...

+--------------------------------------------------------------------+
; Estimated Resource Usage Summary                                  ;
+------------------------------------+-------------------------------+
; Resource                          + Usage                         ;
+------------------------------------+-------------------------------+
; Logic utilization                 ;   65%                         ;
; Dedicated logic registers         ;   27%                         ;
; Memory blocks                     ;   24%                         ;
; DSP blocks                        ;    0%                         ;
+------------------------------------+-------------------------------;
aoc: First stage compilation completed successfully.
aoc: Hardware generation completed successfully.
```

(2) ご使用の PC スペックにもよりますが、カーネルのコンパイルには 13 分程度かかります。エンベデッド・コマンド・シェルのプロンプトが表示されるまで待ちます。

(3) コンパイルが成功すると、以下のように **vectorAdd.aocx** ファイルが生成されます。

```
$ ls -l
合計 2276
d---------+ 1 elsfae    mkpasswd       0 5 月  28 18:34 vectorAdd
----------+ 1 elsfae    mkpasswd   14192 5 月  28 18:21 vectorAdd.aoco
----------+ 1 elsfae    mkpasswd 2281600 5 月  28 18:34 vectorAdd.aocx
-rwx------+ 1 ???????? mkpasswd    1703 6 月  30  2014 vectorAdd.cl
```

4-3. ホスト・アプリケーションのコンパイル

(1) エンベデッド・コマンド・シェルから 以下のコマンドを入力します。

```
$ cd "C:¥Work¥Altera-SoCFPGA-OpenCL-vectorAdd"
$ make all
arm-linux-gnueabihf-g++ host/src/main.cpp common/src/AOCL_Utils.cpp -o vector_a
dd -IC:/altera/15.1/hld/host/include -Icommon/inc -LC:¥altera¥15.1¥hld/board/
c5soc_atlas¥arm32¥lib -LC:/altera/15.1/hld/host/arm32/lib -Wl,--no-as-needed -la
lteracl -lalterahalmmd -lalterammdpcie -lelf -lrt -ldl -lstdc++
```

(2) コンパイルが成功すると、以下のように **vector_add** ファイルが生成されます。

```
$ ls -l
合計 52
drwx------+ 1 ???????? mkpasswd     0 5 月  28 18:03 common
drwx------+ 1 ???????? mkpasswd     0 5 月  28 18:34 device
drwx------+ 1 ???????? mkpasswd     0 5 月  28 18:03 host
-rwx------+ 1 ???????? mkpasswd  1406 12 月 10 19:59 Makefile
----------+ 1 elsfae    mkpasswd 43296 5 月  28 18:37 vector_add
```

4-4. ファイルの転送

ホスト・アプリケーションとコンパイル済みのカーネルを Atlas-SoC の microSD カードに転送します。

(1) SCP(セキュア・コピー)を使用してファイルを転送することができます。

SCP を経由してホスト PC から Atlas-SoC の microSD カードにファイルを転送するには、ホスト PC から scp <source_filename> root@<board_ip_address>:<target_filename> コマンドを使用します

(2) エンベデッド・コマンド・シェルから 次のように入力し vector_add ホスト・アプリケーションを Atlas-SoC に転送します。接続を続けるかどうかを聞かれた場合は、yes ↵ を入力します。この例ではパスワードは altera.123 ↵ を使用します。

```
$ scp  vector_add  root@192.168.1.30:/home/root ↵
Could not create directory '/home/11149/.ssh'.
The authenticity of host '192.168.1.30 (192.168.1.30)' can't be established.
ECDSA key fingerprint is SHA256:SZhkriFhnzX/arvfUwpBVVAG57AUnFXGLCSwRCnHAOE.
Are you sure you want to continue connecting (yes/no)? yes ↵   yes を入力
Failed to add the host to the list of known hosts (/home/11149/.ssh/known_hosts)
.
Password:        パスワードを入力
vector_add                                    100%   42KB  42.3KB/s   00:00
```

(3) 次のように入力し vectorAdd.aocx ファイルを Atlas-SoC に転送します。再度、パスワード altera.123 ↵ を使用します。

```
$ scp  ./device/vectorAdd.aocx  root@192.168.1.30:/home/root ↵
Could not create directory '/home/11149/.ssh'.
The authenticity of host '192.168.1.30 (192.168.1.30)' can't be established.
ECDSA key fingerprint is SHA256:SZhkriFhnzX/arvfUwpBVVAG57AUnFXGLCSwRCnHAOE.
Are you sure you want to continue connecting (yes/no)? yes ↵   yes を入力
Failed to add the host to the list of known hosts (/home/11149/.ssh/known_hosts)
.
Password:        パスワードを入力
vectorAdd.aocx                                100% 1663KB   1.6MB/s   00:00
```

これで、Atlas-SoC の microSD カードにファイルが転送できました。

4-5. OpenCL の実行

転送されたカーネルで FPGA をコンフィギュレーションし、ホスト・アプリケーションを実行することで OpenCL の動作を確認することができます。

(1) Linux コンソールから 以下のように入力し、ターゲットの Linux 上で ドライバ・モジュールをロードします。

```
# source  ./init_opencl.sh
```

(2) 次のように入力して、ボードが正常に動作していることを確認します。

```
# aocl  diagnose
aocl diagnose: Running diagnostic from /home/root/aocl-rte-15.1.0-1.arm32/board/c5soc/arm32/bin

Verified that the kernel mode driver is installed on the host machine.

Using platform: Altera SDK for OpenCL
Board vendor name: Altera Corporation
Board name: DE0nanoSoc : Cyclone V SoC Development Kit

Buffer read/write test passed.

DIAGNOSTIC_PASSED
```

(3) 次のように入力して、転送したファイルを実行可能にします。

```
# chmod  777  vector_add
# chmod  777  vectorAdd.aocx
```

(4) ターゲットの Linux 上で OpenCL カーネルプログラムのプログラミングを行います。リコンフィギュレーションが成功すると、LED の点滅パターンが変化します。

```
# aocl  program /dev/acl0 vectorAdd.aocx
aocl program: Running reprogram from /home/root/aocl-rte-15.1.0-1.arm32/board/c5soc/arm32/bin
Reprogramming was successful!
```

(5) ターゲットの Linux 上で ホスト(ARM)プログラムを実行します。

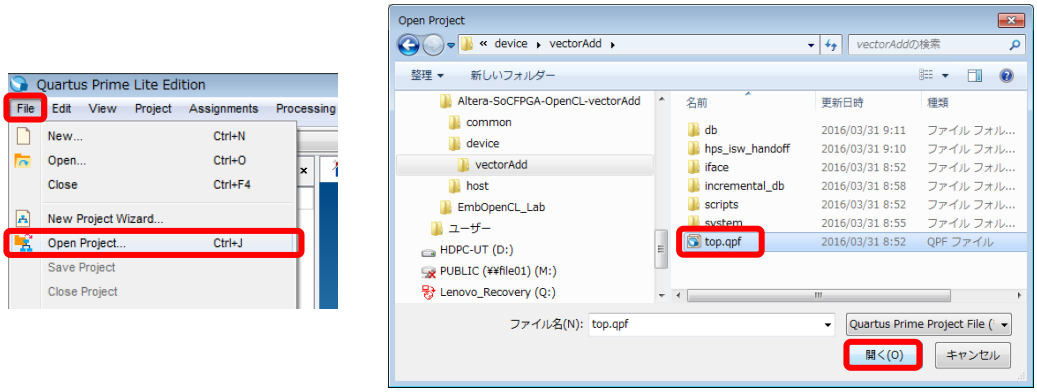
```
# ./vector_add
Initializing OpenCL
Platform: Altera SDK for OpenCL
Using 1 device(s)
  c5soc_atlas_sharedonly : Cyclone V SoC Development Kit
Using AOCX: vectorAdd.aocx
Reprogramming device with handle 1
Launching for device 0 (1000000 elements)

Time: 318.877 ms
Kernel time (device 0): 177.687 ms

Verification: PASS
```

4-6. カーネルのコンパイル結果の確認

コンパイルされたカーネルのデザインを確認します。

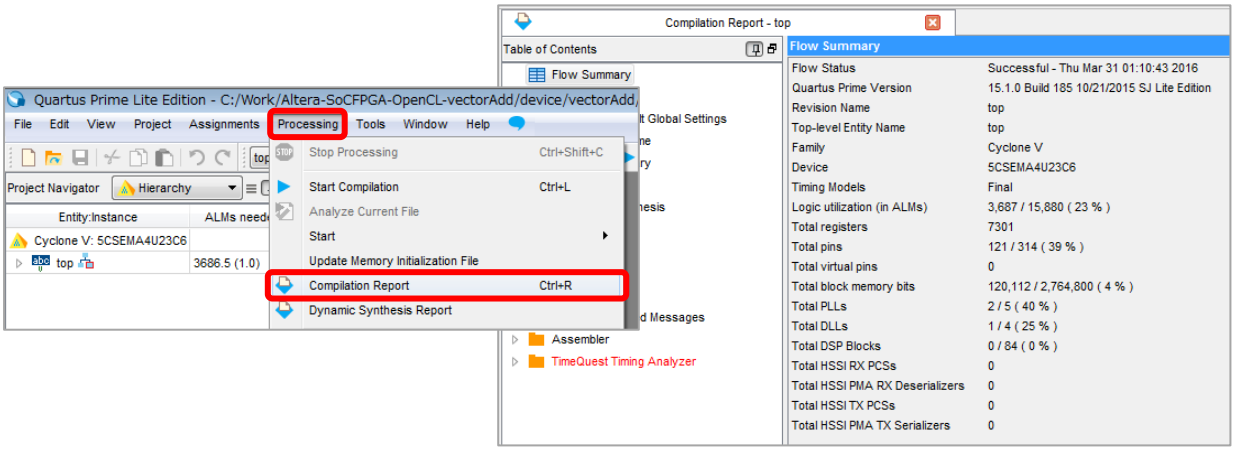
(1) Quartus Prime 開発ソフトウェアを起動して、File メニュー ⇒ Open Project から、

“C:¥Work¥Altera-SoCFPGA-OpenCL-vectorAdd¥device¥vectorAdd¥top.qpf” を開きます。



【図 4-6.1】Quartus Prime 開発ソフトウェアを起動して、File メニュー ⇒ Open Project

(2) Processing メニュー ⇒ Compile Report を選択します。



【図 4-6.2】Processing メニュー ⇒ Compile Report

改版履歴

Revision	年月	概要
1	2016 年 7 月	初版