



MODEL:

Mustang-F100-A10

Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA

Powered by Open Visual Inference & Neural Network Optimization (OpenVINO™) toolkit

User Manual

Rev. 1.05 - May 6, 2019



Revision

Date	Version	Changes
May 6, 2019	1.05	Added Section 4.2.4: Installation - Step by Step (OpenVINO Toolkit 2019 R1)
April 16, 2019	1.04	Added a warning message in Section 3.3: Hardware Installation
January 16, 2019	1.03	Added Section 4.2.3: Installation - Step by Step (OpenVINO Toolkit R5) Modified Appendix B: System Recovery
December 4, 2018	1.02	Modified Section 3.3: Hardware Installation Updated Chapter 4 ~ Chapter 7 Added Chapter 8: IEI Mustang Viewer Utility Added Appendix A: LED Indicators Added Appendix B: System Recovery
October 30, 2018	1.01	Added the following chapters: Chapter 4: Software Installation (OpenVINO™ Toolkit) Chapter 5: Configure and Use the Model Optimizer Chapter 6: Build the Sample Applications Chapter 7: Use the Sample Applications
October 12, 2018	1.00	Initial release

Copyright

COPYRIGHT NOTICE

The information in this document is subject to change without prior notice in order to improve reliability, design and function and does not represent a commitment on the part of the manufacturer.

In no event will the manufacturer be liable for direct, indirect, special, incidental, or consequential damages arising out of the use or inability to use the product or documentation, even if advised of the possibility of such damages.

This document contains proprietary information protected by copyright. All rights are reserved. No part of this manual may be reproduced by any mechanical, electronic, or other means in any form without prior written permission of the manufacturer.

TRADEMARKS

All registered trademarks and product names mentioned herein are used for identification purposes only and may be trademarks and/or registered trademarks of their respective owners.

Manual Conventions



WARNING

Warnings appear where overlooked details may cause damage to the equipment or result in personal injury. Warnings should be taken seriously.



CAUTION

Cautionary messages should be heeded to help reduce the chance of losing data or damaging the product.



NOTE

These messages inform the reader of essential but non-critical information. These messages should be read carefully as any directions or instructions contained therein can help avoid making mistakes.

Table of Contents

1 INTRODUCTION.....	1
1.1 INTRODUCTION.....	2
1.2 FEATURES.....	3
1.3 BLOCK DIAGRAM.....	3
1.4 I/O AND DIMENSIONS.....	4
1.5 TECHNICAL SPECIFICATIONS.....	5
2 UNPACKING.....	6
2.1 ANTI-STATIC PRECAUTIONS.....	7
2.2 UNPACKING PRECAUTIONS.....	7
2.3 PACKING LIST.....	8
3 HARDWARE INSTALLATION.....	9
3.1 ANTI-STATIC PRECAUTIONS.....	10
3.2 INSTALLATION CONSIDERATIONS.....	10
3.3 HARDWARE INSTALLATION.....	12
4 SOFTWARE INSTALLATION (OPENVINO™ TOOLKIT).....	16
4.1 SYSTEM REQUIREMENTS.....	17
4.2 INSTALLATION.....	17
4.2.1 <i>Installation via Website</i>	17
4.2.2 <i>Installation - Step by Step (OpenVINO Toolkit R4)</i>	18
4.2.2.1 Install the External Software Dependencies.....	18
4.2.2.2 Install the OpenVINO™ Core Components.....	19
4.2.2.3 Set the Environment Variables.....	19
4.2.2.4 Install Intel Vision Accelerator Design with Arria 10 FPGA Board Support Package.....	20
4.2.2.5 Verify Your Configuration.....	21
4.2.2.6 Intel® DLIA Bitstreams.....	21
4.2.2.7 Program the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA.....	22

4.2.3 <i>Installation - Step by Step (OpenVINO Toolkit R5)</i>	23
4.2.3.1 Install the External Software Dependencies	23
4.2.3.2 Install the OpenVINO™ Core Components	24
4.2.3.3 Set the Environment Variables	25
4.2.3.4 Program the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA	27
4.2.4 <i>Installation - Step by Step (OpenVINO Toolkit 2019 R1)</i>	30
4.2.4.1 Install the External Software Dependencies	30
4.2.4.2 Install the OpenVINO™ Core Components	30
4.2.4.3 Install FPGA driver	31
5 CONFIGURE AND USE THE MODEL OPTIMIZER	35
5.1 CONFIGURE THE MODEL OPTIMIZER	36
5.2 USE THE MODEL OPTIMIZER	37
6 BUILD THE SAMPLE APPLICATIONS	39
7 USE THE SAMPLE APPLICATIONS	41
7.1 CLASSIFICATION_ASYNC_SAMPLE WITH MAXIMUM OPTIMIZATION	42
7.2 OBJECT_DETECTION_SSD.....	45
7.3 OTHER DEMOS	45
8 IEI MUSTANG VIEWER UTILITY	46
8.1 INSTALLATION REQUIREMENT	47
8.2 DEVICE INFORMATION.....	48
8.3 DIAGNOSE	49
8.4 SAVING LOG FILE	53
8.5 TROUBLESHOOTING.....	54
A LED INDICATORS	55
A.1 ON-BOARD LED INDICATORS AND PURPOSE	56
B SYSTEM RECOVERY.....	57
B.1 REQUIRED HARDWARE	58
B.2 RECOVERY STEPS	59
C REGULATORY COMPLIANCE.....	64

Mustang-F100-A10

D PRODUCT DISPOSAL 66

E HAZARDOUS MATERIALS DISCLOSURE 68

List of Figures

Figure 1-1: Mustang-F100-A10	2
Figure 1-2: Block Diagram	3
Figure 1-3: Dimensions (mm)	4
Figure 3-1: Remove Two Blank Brackets	13
Figure 3-2: Change to Full-height Bracket	13
Figure 3-3: Change to Full-height Bracket	14
Figure 3-4: Power Connector Location	14
Figure 3-5: Assign a Card ID	15

List of Tables

Table 4-1:: Bitstream with Topology.....	22
Table 4-2: OpenVINO R5Bitstream with Topology	27
Table 4-3: OpenVINO 2019 R1Bitstream with Topology	34
Table 7-1: Other Sample Applications You can run with the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA	45

Chapter

1

Introduction

1.1 Introduction



Figure 1-1: Mustang-F100-A10

The Mustang-F100-A10 is a deep learning convolutional neural network acceleration card for speeding up AI inference, in a flexible and scalable way. Equipped with Intel® Arria® 10 FPGA, 8 GB DDR4 on board RAM, the Mustang-F100-A10 PCIe card can be used with the existing system, enabling high-performance computing without costing a fortune. FPGAs can offer reprogrammability that allows developers to implement algorithms in different applications to achieve optimal solutions. Algorithms implemented in FPGA provide deterministic timing, which achieved low latency real-time computation. Furthermore, compared to CPU or GPU, the power consumption of FPGA is extremely efficient. Those features make the Mustang-F100-A10 a great choice in edge computing.

"Open Visual Inference & Neural Network Optimization (OpenVINO™) toolkit" is based on convolutional neural networks (CNN), the toolkit extends workloads across Intel® hardware and maximizes performance. It can optimize pre-trained deep learning model such as Caffe, MXNET, Tensorflow into IR binary file then execute the inference engine across Intel®-hardware heterogeneously such as CPU, GPU, Intel® Movidius™ Neural Compute Stick, and FPGA.

Mustang-F100-A10

1.2 Features

Mustang-F100-A10 features are listed below:

- Intel® Arria® 10 GX1150 FPGA
- Interface: PCIe 3.0 x8
- Form factor: Standard half-height, half-length, double-slot
- Active fan
- Operating Temperature : 5°C~60°C (ambient temperature)
- Operation Humidity : 5% to 90% relative humidity
- Power consumption: 40 W typical
- Power connector: 12 V external power
- Rotary switch/LED indicator: Identify card number

1.3 Block Diagram

Figure 1-2 shows the block diagram of the Mustang-F100-A10.

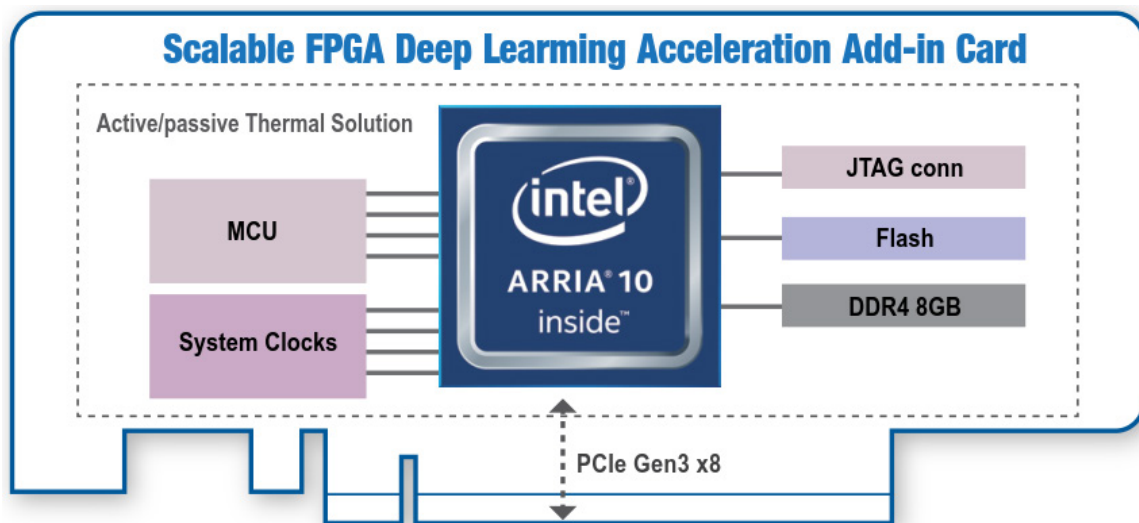


Figure 1-2: Block Diagram

1.4 I/O and Dimensions

The I/O interfaces and dimensions of the board are listed below:

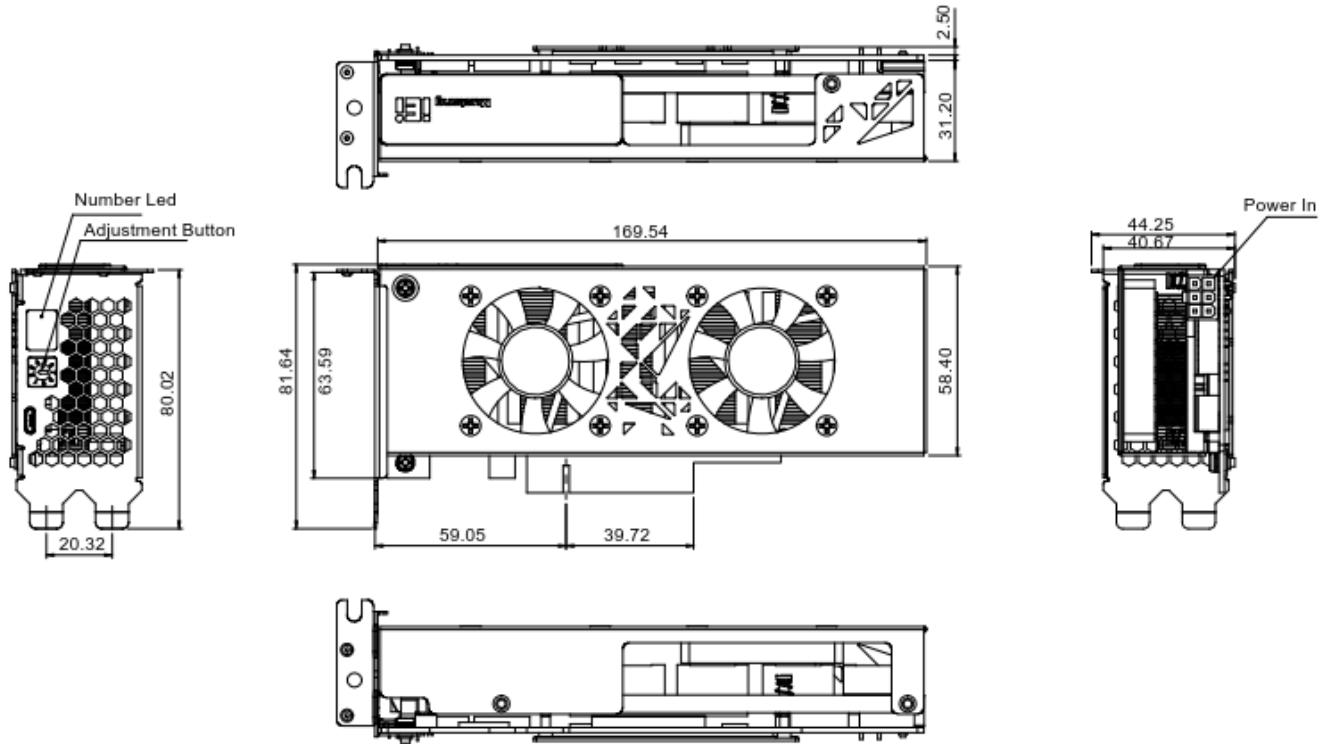


Figure 1-3: Dimensions (mm)

Mustang-F100-A10

1.5 Technical Specifications

Mustang-F100-A10 technical specifications are listed below.

Specification	Mustang-F100-A10
Operating Systems	Ubuntu 16.04.3 LTS 64-bit, CentOS 7.4 64-bit (support Windows® 10 in the end of 2018 & more OS are coming soon)
Memory	8G on board DDR4
Mini USB Ports	USB 2.0 mini port for debugging
Physical PCIe Interface	PCI Express x8 Compliant with PCI Express Specification V3.0
External Power Connector	*Preserved PCIe 6-pin 12V external power
Indicator	7-segment LED display for card ID
Fan	Dual fan
Power Consumption	40 W typical
Operating Temperature	5°C ~ 60°C
Operating Humidity	5% ~ 90%
Dimensions (WxHxD)	169.5 mm x 68.7 mm x 33.7 mm

*Standard PCIe slot provides 75W power; this feature is preserved for user in case of different system configuration

Chapter

2

Unpacking

Mustang-F100-A10

2.1 Anti-static Precautions



WARNING!

Static electricity can destroy certain electronics. Make sure to follow the ESD precautions to prevent damage to the product, and injury to the user.

Make sure to adhere to the following guidelines:

- **Wear an anti-static wristband:** Wearing an anti-static wristband can prevent electrostatic discharge.
- **Self-grounding:** Touch a grounded conductor every few minutes to discharge any excess static buildup.
- **Use an anti-static pad:** When configuring any circuit board, place it on an anti-static mat.
- **Only handle the edges of the PCB:** Don't touch the surface of the motherboard. Hold the motherboard by the edges when handling.

2.2 Unpacking Precautions

When the Mustang-F100-A10 is unpacked, please do the following:

- Follow the antistatic guidelines above.
- Make sure the packing box is facing upwards when opening.
- Make sure all the packing list items are present.



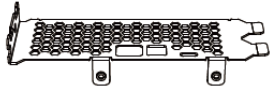
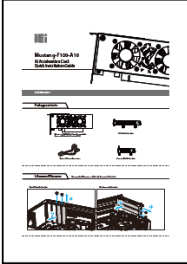
2.3 Packing List



NOTE:

If any of the components listed in the checklist below are missing, do not proceed with the installation. Contact the IEI reseller or vendor the Mustang-F100-A10 was purchased from or contact an IEI sales representative directly by sending an email to sales@ieiworld.com.

The Mustang-F100-A10 is shipped with the following components:

Quantity	Item and Part Number	Image
1	Mustang-F100-A10 AI acceleration card	
1	PCIe power adapter	
1	Full-height bracket	
1	Quick Installation Guide	

Chapter

3

Hardware Installation

3.1 Anti-static Precautions

**WARNING:**

Failure to take ESD precautions during the installation of the Mustang-F100-A10 may result in permanent damage to the Mustang-F100-A10 and severe injury to the user.

Electrostatic discharge (ESD) can cause serious damage to electronic components, including the Mustang-F100-A10. Dry climates are especially susceptible to ESD. It is therefore critical that whenever the Mustang-F100-A10 or any other electrical component is handled, the following anti-static precautions are strictly adhered to.

- ***Wear an anti-static wristband:*** Wearing a simple anti-static wristband can help to prevent ESD from damaging the board.
- ***Self-grounding*** Before handling the board, touch any grounded conducting material. During the time the board is handled, frequently touch any conducting materials that are connected to the ground.
- ***Use an anti-static pad:*** When configuring the Mustang-F100-A10, place it on an anti-static pad. This reduces the possibility of ESD damaging the Mustang-F100-A10.
- ***Only handle the edges of the PCB:*** When handling the PCB, hold the PCB by the edges.

3.2 Installation Considerations

**NOTE:**

The following installation notices and installation considerations should be read and understood before installation. All installation notices must be strictly adhered to. Failing to adhere to these precautions may lead to severe damage and injury to the person performing the installation.

Mustang-F100-A10



WARNING:

The installation instructions described in this manual should be carefully followed in order to prevent damage to the Mustang-F100-A10, Mustang-F100-A10 components and injury to the user.

Before and during the installation please **DO** the following:

- Read the user manual:
The user manual provides a complete description of the Mustang-F100-A10 installation instructions and configuration options.
- Wear an electrostatic discharge cuff (ESD):
Electronic components are easily damaged by ESD. Wearing an ESD cuff removes ESD from the body and helps prevent ESD damage.
- Turn off system:
When installing the Mustang-F100-A10, make sure that the system to be connected is disconnected from all power supplies and that no electricity is being fed into the system.

Before and during the installation of the Mustang-F100-A10 **DO NOT:**

- Remove any of the stickers on the PCB board. These stickers are required for warranty validation.
- Use the product before verifying all the cables and power connectors are properly connected.
- Allow screws to come in contact with the PCB circuit, connector pins, or its components.

3.3 Hardware Installation

**WARNING:**

DO NOT install the Mustang-F100-A10 into the TANK AIoT Dev. Kit before shipment. It is recommended to ship them in their original boxes to prevent the Mustang-F100-A10 from being damaged.

To install the Mustang-F100-A10, please follow the steps below.

Step 1: Prepare the computer. Turn off the computer, and remove the power cord from the rear of the power supply.

**WARNING:**

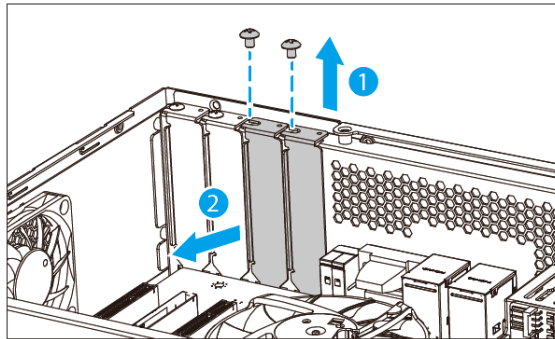
Disconnect the computer from the power supply and from any networks to which you will install the Mustang-F100-A10, or you risk damaging the system or experiencing electrical shock.

Step 2: Remove the cover from the chassis.

Step 3: Locate available PCIe slots and remove the blank brackets. The Mustang-F100-A10 is compatible with PCIe x8 and x16 slots, and needs two side-by-side PCIe slots for installation. Remove two blank bracket panels on the back of the computer that align with the PCIe slot (right side in **Figure 3-1**) for installing the Mustang-F100-A10. Save the bracket screws.

Mustang-F100-A10

Low-profile bracket



Full-height bracket

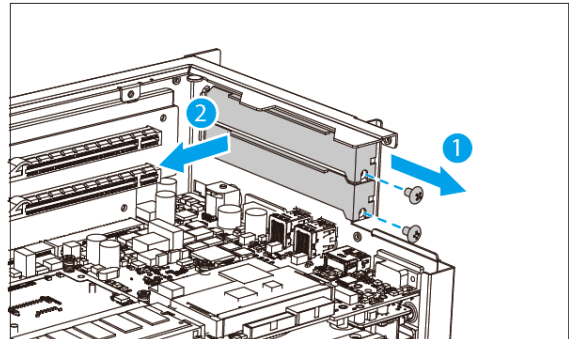


Figure 3-1: Remove Two Blank Brackets

Step 4: [Only needed for full-height installation] Change the bracket on the Mustang-F100-A10 from low-profile bracket to full-height bracket.

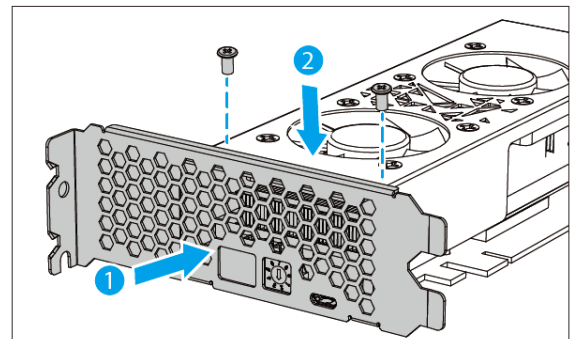
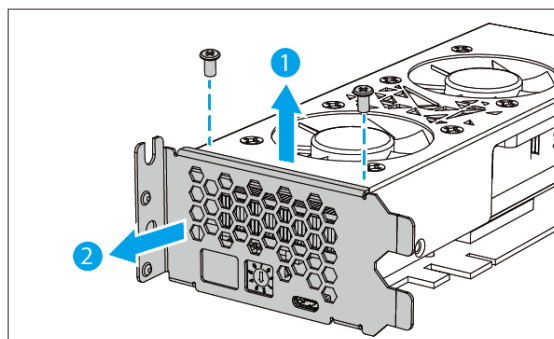
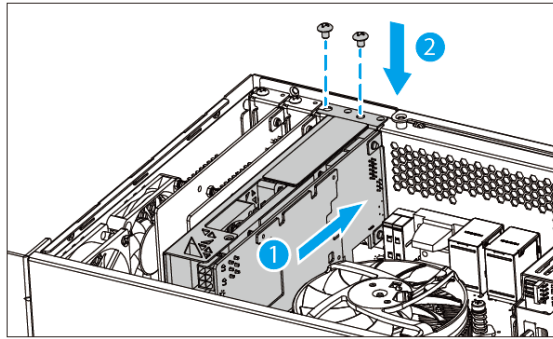


Figure 3-2: Change to Full-height Bracket

Step 5: Install and secure the Mustang-F100-A10 to the system. Align the Mustang-F100-A10 to the PCIe slot. Press down gently, but firmly, to seat the Mustang-F100-A10 correctly in the slot. Install two bracket screws to secure the Mustang-F100-A10 to the system's chassis.

Low-profile bracket



Full-height bracket

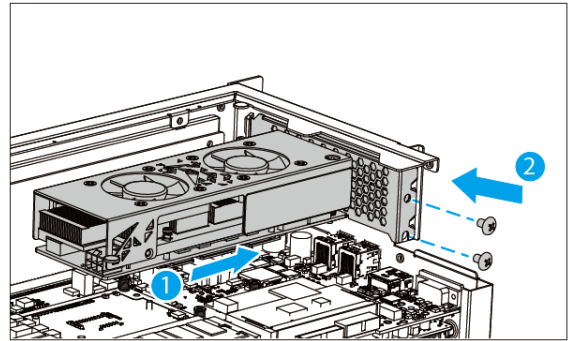


Figure 3-3: Change to Full-height Bracket

Step 6: Connect a power cable to the **Mustang-F100-A10**. The Mustang-F100-A10 requires 12V 5A DC power. Use a power cable with 6-pin connector from the system, if applicable, or add the 4-pin to 6-pin PCIe power adapter to connect to the power connector of the Mustang-F100-A10.

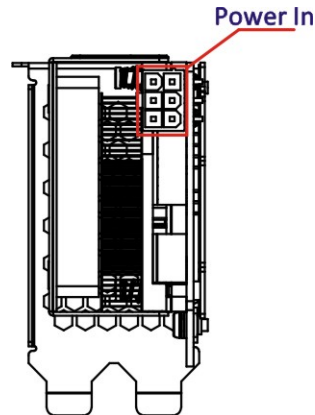


Figure 3-4: Power Connector Location

Step 7: Assign a card ID to the **Mustang-F100-A10** by adjusting the rotary switch. The card ID number assigned here will be shown on the LED display of the card after power-up.

Mustang-F100-A10

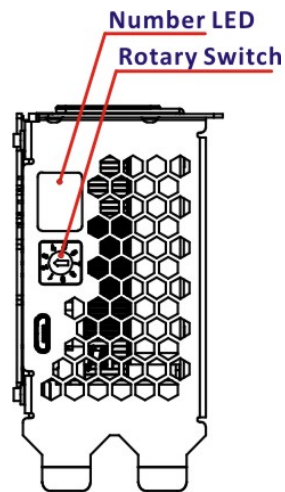


Figure 3-5: Assign a Card ID

- Step 8:** Repeat **Step 3 ~ Step 7** to install multiple Mustang-F100-A10 into the system if available.
- Step 9:** Replace the cover of the chassis.
- Step 10:** Reconnect any power cords and any network cables to the system. Power up the system.

Chapter

4

Software Installation (OpenVINO™ Toolkit)

Mustang-F100-A10

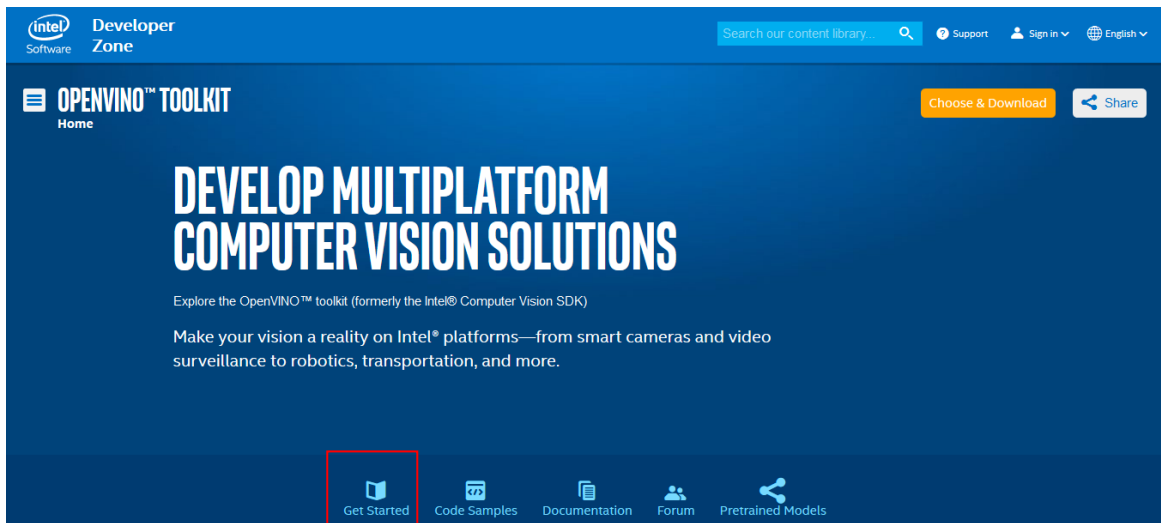
4.1 System Requirements

- Linux Ubuntu 16.04.3 LTS 64bit
- CentOS 7.4 64bit
- Windows 10 64bit (coming soon)
- OpenVINO™ Toolkit was pre-installed in TANK-870AI.

4.2 Installation

4.2.1 Installation via Website.

Go to <https://software.intel.com/en-us/openvino-toolkit>. Click “Get Started” then choose your configuration from “Development Environment Installation Guides & Videos”. Follow the instruction to complete the installation procedure.



INSTALLATION & SETUP GUIDES

Development Environment Installation Guides & Videos

[Linux*: Guide | Video](#)
[Windows*: Guide | Video](#)

[Linux* with FPGA: Guide](#)

Target Device Installation Guides

[Quick Start Guide for Intel® Programmable Acceleration Card with Intel® Arria® 10 FPGA GX](#)
[Acceleration Hub for Intel® FPGA Development Kit for Intel® Arria® 10 FPGA GX](#)

[Intel® Movidius™ Neural Compute Stick Quick Start Guide](#)

Intel® Deep Learning Deployment Toolkit References

[Model Optimizer Developer Guide](#)
[Inference Engine Developer Guide](#)

[Pretrained Models & Algorithms](#)

4.2.2 Installation - Step by Step (OpenVINO Toolkit R4)

If you have not done so already, download the [OpenVINO toolkit R4 release](#). Be sure to download the Linux version that includes FPGA support.

NOTE: Before beginning the installation, make sure you have the correct Linux kernel version:

```
cat /proc/version
```

If you have the correct kernel version, your output looks like this:

```
Linux version 4.13.0.45-generic (buildd@lgw01-12) (gcc version
5.4.0 20160609 (Ubuntu >>5.4.0-6ubuntu1~16.04.4) )
#32~16.04.2-Ubuntu SMP Thu Jul 20 10:19:48 UTC 2017
```

Run two commands to get specific kernel version if your kernel version is different with 4.13.0.45.:

```
sudo apt-get install linux-image-extra-4.13.0-45-generic
sudo apt-get install linux-headers-4.13.0-45-generic
```

4.2.2.1 Install the External Software Dependencies

1. Go to the directory to which you downloaded the OpenVINO toolkit.

The default directory is `~/Downloads`

The default filename is `l_openvino_toolkit_fpga_p_<version>.tgz`.

If you used a different directory or renamed the file, change the following instructions according to your naming conventions.

```
cd ~/Downloads
```

2. Unpack the `.tgz` file:

```
tar -xvf l_openvino_toolkit_fpga_p_<version>.tgz
```

A directory named `l_openvino_toolkit_fpga_p_<version>` is created.

3. Go to the `l_openvino_toolkit_fpga_p_<version>` directory:

```
cd l_openvino_toolkit_fpga_p_<version>
```

Mustang-F100-A10

4. Run a script named `install_cv_sdk_dependencies.sh`
`./install_cv_sdk_dependencies.sh`

This script downloads and installs the external software dependencies.

Continue with the next section to install the OpenVINO™ core components.

4.2.2.2 Install the OpenVINO™ Core Components

1. Choose between installing with or without a GUI. Only the visual aspects are different between these options. Choose ONE option:
 - If you want to use a GUI installation wizard to prompt you for input:
`./install_GUI.sh`
 - If you want to use command-line instructions to prompt you for input:
`./install.sh`

2. Follow the instructions on your screen.

The base installation is complete. Continue to the next section to set the environment variables.

OpenVINO™ installs in one of two locations, depending on how you install it:

- If you install as the root user, the software will install to
`/opt/intel/computer_vision_sdk_fpga_<VERSION>/`
- If you install as the the software will install to:
`/home/<USERNAME>/intel/computer_vision_sdk_fpga_<VERSION>/`

4.2.2.3 Set the Environment Variables

Run a script to set the environment variables that are required to run the OpenVINO™ toolkit for this session:

```
source /opt/intel/computer_vision_sdk_2018.3.<version>/bin/setupvars.sh
```

NOTE: The OpenVINO™ environment variables are removed when you close the shell. As an option, use your preferred method to permanently set the variables.

Continue to the next section to initialize the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA.

4.2.2.4 Install Intel Vision Accelerator Design with Arria 10 FPGA Board Support Package

The version of the OpenVINO® toolkit that you installed includes the Intel FPGA RTE for OpenCL Pro Edition software version.

The Board Support Package for Intel Vision Accelerator Design with Arria 10 FPGA will be available at <BSP_package>. **Please download this package from IEI website.**

<BSP_package> => hddlf_1150_sg1.tgz

Download procedure:

1. Go to: <http://download.ieiworld.com/>
2. Search: Mustang-F100
3. Download Mustang-F100-A10 Driver

Extract the bsp package:

```
1.tar -xvf hddlf_1150_sg1.tgz
```

Convert the BSP files from DOS to UNIX:

1. apt-get install dos2unix
2. chmod +x <BSP_package>
3. find <BSP_package> -type f -print0 | xargs -0 dos2unix

Make the script files from the BSP executable:

```
1.chmod +x /opt/altera/aocl-pro-rte/aclrte-linux64/board/<BSP_package>/linux64/libexec/*
```

The environment setup will be as follows:

1. `source /opt/intel/computer_vision_sdk_2018.4.420/bin/setupvars.sh`
2. Set the AOCL_BOARD_PACKAGE_ROOT environment variable with the command:
`export AOCL_BOARD_PACKAGE_ROOT=$HOME/<BSP_package>`
3. Run a script to temporarily set the Intel OpenCL runtime environment variables:
`source /opt/altera/aocl-pro-rte/aclrte-linux64/init_openc1.sh`
4. Install the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA drivers:
`aocl install`

Mustang-F100-A10

4.2.2.5 Verify Your Configuration

1. View the PCIe device on your system:

```
lspci | grep -i Altera
```

Success is indicated by a response similar to:

```
01:00.0 Processing accelerators: Altera Corporation Device 2494
(rev 01)
```

2. After configuring the board, run the AOCL diagnose command from a command line prompt on the machine that is connected to Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA.

```
aocl diagnose
```

3. If the configuration is successful, the command returns “Diagnostic PASSED”.

4.2.2.6 Intel® DLIA Bitstreams

You must set up the Intel Vision Acceleration Design with Intel Arria 10 FPGA before you program the bitstreams. Make sure that the board and environment are properly configured and set up before you program the bitstream.

Pre-compiled bitstream samples for Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA are available with the OpenVINO™ toolkit that you installed.

The table below lists all bitstreams available, with the associated supported topologies.

Bitstream
FP11
4-0_PL1_FP11_Generic_Alexnet.aocx
4-0_PL1_FP11_GoogleNet.aocx
4-0_PL1_FP11_SqueezeNet.aocx
4-0_PL1_FP11_MobileNet_ResNet_VGG_Clamp.aocx
4-0_PL1_FP11_TinyYolo_SSD300.aocx
4-0_PL1_FP11_ELU.aocx
FP16
4-0_PL1_FP16_Generic_Alexnet_GoogleNet_VGG.aocx
4-0_PL1_FP16_ResNet_MobileNet_SqueezeNet.aocx

4-0_PL1_FP16_TinyYolo_SSD300.aocx
4-0_PL1_FP16_ELU.aocx
4-0_PL1_FP16_Clamp.aocx

Table 4-1:: Bitstream with Topology

4.2.2.7 Program the Intel[®] Vision Accelerator Design with Intel[®] Arria[®] 10 FPGA

This step uses the Intel FPGA RTE for OpenCL. To program the AOCX file with FP11 or FP16 bitstreams:

```
aocl program acl0 $<BITSTREAM_DATA_TYPE>.aocx
```


Mustang-F100-A10

4.2.3 Installation - Step by Step (OpenVINO Toolkit R5)

If you have not done so already, download the [OpenVINO toolkit R5 release](#). Be sure to download the Linux version that includes FPGA support.

NOTE: Before beginning the installation, make sure you have the correct Linux kernel version:

```
cat /proc/version
```

Make sure you are using a Linux kernel version 4.14 and later.

For example:

- 4.15.0-38-Generic

Install kernel of 4.15.0-38

- `apt-get install linux-image-4.15.0-38-generic`
- `apt-get install linux-headers-4.15.0-38-generic`
- `apt-get install linux-modules-extra-4.15.0-38-generic`
- `apt-get remove linux-modules-extra-4.15.0-38-generic`

4.2.3.1 Install the External Software Dependencies

1. Go to the directory to which you downloaded [Quartus Pro Programmer 17.1.1](#).

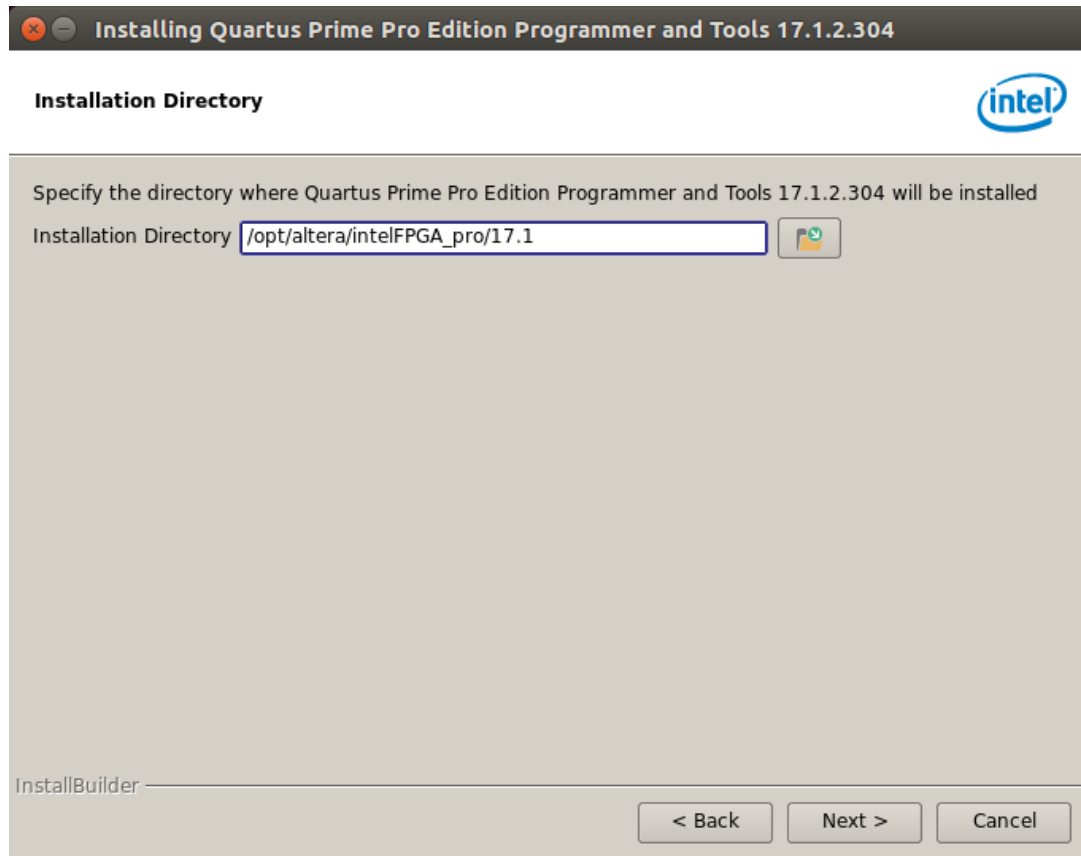
```
cd /home/<user>/Downloads.
```

2. Run the Quartus Pro Programmer Setup file:

```
sudo chmod +x QuartusProProgrammerSetup-17.1.2.304-linux.run
```

```
sudo ./QuartusProProgrammerSetup-17.1.2.304-linux.run
```

Installation Directory: `/opt/altera/intelFPGA_pro/17.1`



4.2.3.2 Install the OpenVINO™ Core Components

1. Go to the directory to which you downloaded the OpenVINO toolkit. The default directory is `~/Downloads`, and the default filename is `l_openvino_toolkit_fpga_p_<version>.tgz`.
If you used a different directory or renamed the file, change the following instructions according to your naming conventions.

```
cd ~/Downloads
```

2. Unpack the .tgz file:

```
tar -xf l_openvino_toolkit_fpga_p_<version>.tgz
```

A directory named `l_openvino_toolkit_fpga_p_<version>` is created.

3. Go to the `l_openvino_toolkit_fpga_p_<version>` directory:

```
cd l_openvino_toolkit_fpga_p_<version>
```

Mustang-F100-A10

4. Run a script named `install_cv_sdk_dependencies.sh`

```
sudo ./install_cv_sdk_dependencies.sh
```

This script downloads and installs the external software dependencies.

5. Choose between installing with or without a GUI. Only the visual aspects are different between these options. Choose ONE option:

If you want to use a GUI installation wizard to prompt you for input:

```
sudo ./install_GUI.sh
```

6. Follow the instructions on your screen.

The base installation is complete. Continue to the next section to set the environment variables.

4.2.3.3 Set the Environment Variables

1. View the PCIe device on your system:

```
lspci | grep -i Altera
```

Success is indicated by a response similar to:

```
01:00.0 Processing accelerators: Altera Corporation Device 2494 (rev 01)
```

2. Download `fpga_support_files.tgz` from [Intel Resource Center](#). The files are required to ensure that the FPGA card and OpenVino work correctly.

3. Go to the directory where the file is being downloaded and unpack the file:

```
tar -xvzf fpga_support_files.tgz
```

4. Go to the `fpga_support_files` directory.

5. Switch to superuser:

```
sudo su
```

6. Change directory to `fpga_support_files`:

```
cd /home/<user>/Downloads/fpga_support_files/
```

7. Source the `setup_env.sh` script from the `fpga_support_files` to setup the environment variables.

```
source setup_env.sh
```

8. Run `fpga_dependencies` script to allow OpenCL to support Ubuntu and recent kernel versions:

```
./install_openvino_fpga_dependencies.sh
```

Then chose “3” Intel Vision Accelerator Design with Intel Altera 10 FPGA (IEI Mustang-F100-A10).

```
root@iei-SER0:/home/iei/Downloads/fpga_support_files# ./install_openvino_fpga_dependencies.sh
Would you like to setup an FPGA? Select an option below.
Select (1) for None, (2) Intel® Arria® 10 GX FPGA Development Kit, (3) for Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA (IEI Mustang-F100-A10):
1) none
2) Intel® Arria® 10 GX FPGA Development Kit
3) Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA (IEI Mustang-F100-A10)
#? 3
Selected:

The NEO OpenCL GPU driver is required for using the GPU with OpenVINO. Would you like to install it?
Please select (y/n)
y
```

Note: The OpenVINO™ environment variables are removed when you close the shell. As an option, use your preferred method to permanently set the variables

9. Check if the host system recognizes the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA board.

Confirm you can detect the PCIe card:

```
lspci | grep -i Altera
```

Your output is similar to:

```
01:00.0 Processing accelerators: Altera Corporation Device 2494
(rev
```

10. Run the command:

```
aocl install
```

Mustang-F100-A10

11. Run the command:

```
aocl diagnose
```

You should be seeing DIAGNOSTIC_PASSED before proceeding to the next steps.

This step uses the Intel FPGA RTE for OpenCL. To program the AOCX file with FP11 or FP16 bitstreams:

```
aocl program acl0 $<BITSTREAM_DATA_TYPE>.aocx
```

Bitstream
FP11
5-0_PL1_FP11_Alexnet_GoogleNet.aocx
5-0_PL1_FP11_ELU.aocx
5-0_PL1_FP11_Generic.aocx
5-0_PL1_FP11_MobileNet_Clamp.aocx
5-0_PL1_FP11_ResNet.aocx
5-0_PL1_FP11_RMNet.aocx
5-0_PL1_FP11_SqueezeNet.aocx
5-0_PL1_FP11_TinyYolo_SSD300.aocx
5-0_PL1_FP11_VGG.aocx
FP16
5-0_PL1_FP16_AlexNet_GoogleNet_SqueezeNet.aocx
5-0_PL1_FP16_MobileNet_Clamp.aocx

Table 4-2: OpenVINO R5Bitstream with Topology

4.2.3.4 Program the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA

This step uses the Intel FPGA RTE for OpenCL. To program the AOCX file with FP11 or FP16 bitstreams:

```
aocl program acl0 $<BITSTREAM_DATA_TYPE>.aocx
```

If the below error message appear "aocl program: Program failed.", that means you may have bitstreams versions compatible issue and have to update the bitstreams via FPGA download cable.

```
Call "aocl diagnose <device-names>" to run diagnose for specified devices
Call "aocl diagnose all" to run diagnose for all devices
lei@lei-SER0:~$ aocl program acl0 /home/iei/Downloads/2-0-1_PL1_FP11_Generic.aocx
aocl program: Running program from /opt/altera/aocl-pro-rte/aclrte-linux64/board/hddl1_1150_sg1/linux64/libexec
Programming device: a10gx_1ddrd : Intel Vision Accelerator Design with Intel Arria 10 FPGA (acla10_1150_sg1)
Reprogramming device [0] with handle 1
MMD INFO : [acla10_1150_sg10] PR base and import compile IDs do not match
MMD INFO : [acla10_1150_sg10] PR base ID currently configured is 0x389dd9fe
MMD INFO : [acla10_1150_sg10] PR import compile expects ID to be 0x25e14e25
MMD INFO : [acla10_1150_sg10] Falling back to JTAG programming instead of PR
MMD INFO : Autodetect Cable not found!!
MMD INFO : setting Cable to default value 1
MMD INFO : setting Device Index to default value 1
MMD INFO : executing "quartus_pgm -c 1 -m jtag -o "P;reprogram_temp.sof@1"
Error (213013): Programming hardware cable not detected
Error (213013): Programming hardware cable not detected
Error (213013): Programming hardware cable not detected
mmd program device: Board reprogram failed
OpenCL Notification Callback: Reprogram of device failed
Failed clCreateProgramWithBinary.
Error code: -2
aocl program: Program failed.
lei@lei-SER0:~$
```

This step uses FPGA download cable to program the AOCX file with FP11 or FP16 bitstreams:

1. Connect the FPGA download cable to USB connector and FPGA connector (please refer to Section B “Required Hardware”)
2. Update the FPGA aocx by the below command,
3. `aocl flash acl0 $<BITSTREAM_DATA_TYPE>.aocx`

Mustang-F100-A10

```

root@iel-SER0: /home/iel/Downloads/fpga_support_files
root@iel-SER0: /home/iel/Downloads/fpga_support_files# aocl flash acl0 /opt/intel/computer_vision_sdk/bitstreams/a10_vision_design_bitstreams/S-0_PL1_FP11_Generic.aocx
aocl flash: Running flash from /opt/altera/aocl-pro-rte/aclrte-linux64/board/hddl1_1150_sgl/linux64/libexec
Info:
Info: Running Quartus Prime Convert programming file
Info: Version 17.1.2 Build 304 01/31/2018 SJ Pro Edition
Info: Copyright (C) 2018 Intel Corporation. All rights reserved.
Info: Your use of Intel Corporation's design tools, logic functions
Info: and other software and tools, and its ANPP partner logic
Info: functions, and any output files from any of the foregoing
Info: (including device programming or simulation files), and any
Info: associated documentation or information are expressly subject
Info: to the terms and conditions of the Intel Program License
Info: Subscription Agreement, the Intel Quartus Prime License Agreement,
Info: the Intel FPGA IP License Agreement, or other applicable license
Info: agreement, including, without limitation, that your use is for
Info: the sole purpose of programming logic devices manufactured by
Info: Intel and sold by Intel or its authorized distributors. Please
Info: refer to the applicable agreement for further details.
Info: Processing started: Fri Jan 11 09:22:45 2019
Info: Command: quartus_cpf --convert flash.cof
Info: Quartus Prime Convert programming file was successful. 0 errors, 0 warnings
Info: Peak virtual memory: 1635 megabytes
Info: Processing ended: Fri Jan 11 09:23:03 2019
Info: Elapsed time: 00:00:18
Info: Total CPU time (on all processors): 00:00:25
Info:
Info: Running Quartus Prime Programmer
Info: Version 17.1.2 Build 304 01/31/2018 SJ Pro Edition
Info: Copyright (C) 2018 Intel Corporation. All rights reserved.
Info: Your use of Intel Corporation's design tools, logic functions
Info: and other software and tools, and its ANPP partner logic
Info: functions, and any output files from any of the foregoing
Info: (including device programming or simulation files), and any
Info: associated documentation or information are expressly subject
Info: to the terms and conditions of the Intel Program License
Info: Subscription Agreement, the Intel Quartus Prime License Agreement,
Info: the Intel FPGA IP License Agreement, or other applicable license
Info: agreement, including, without limitation, that your use is for
Info: the sole purpose of programming logic devices manufactured by
Info: Intel and sold by Intel or its authorized distributors. Please
Info: refer to the applicable agreement for further details.
Info: Processing started: Fri Jan 11 09:23:04 2019
Info: Command: quartus_pgh -c 1 flash.cdf
Info (213045): Using programming cable "USB-Blaster [1-7]"
Info (213011): Using programming file flash.jic with checksum 0xA7640AE4 for device 10AX115H20
Info (209060): Started Programmer operation at Fri Jan 11 09:23:24 2019
Info (209016): Configuring device index 1
Info (209017): Device 1 contains JTAG ID code 0x02E60000
Info (209007): Configuration succeeded -- 1 device(s) configured
Info (19845): Start Serial Flash Loader programming
Info (209018): Device 1 silicon ID is 0x20
Info (209044): Erasing ASP configuration device(s)
Info (209023): Programming device(s)
Info (19845): End Serial Flash Loader programming
Info (209011): Successfully performed operation(s)
Info (209001): Ended Programmer operation at Fri Jan 11 09:31:35 2019
Info: Quartus Prime Programmer was successful. 0 errors, 0 warnings
Info: Peak virtual memory: 1747 megabytes
Info: Processing ended: Fri Jan 11 09:31:35 2019
Info: Elapsed time: 00:08:31
Info: Total CPU time (on all processors): 00:00:59
root@iel-SER0: /home/iel/Downloads/fpga_support_files#

```

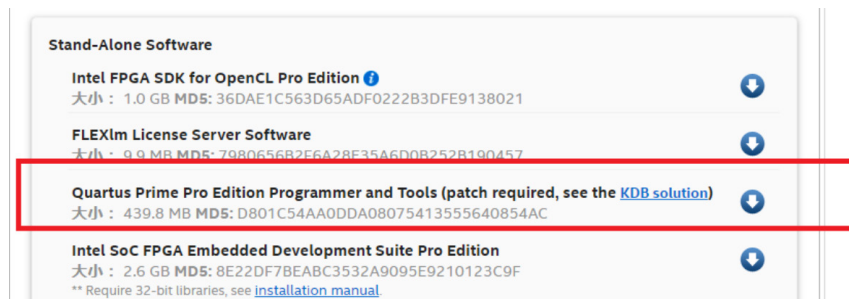
4.2.4 Installation - Step by Step (OpenVINO Toolkit 2019 R1)

If you have not done so already, download the [OpenVINO toolkit 2019 R1 release](#). Be sure to download the Linux version that includes FPGA support.

4.2.4.1 Install the External Software Dependencies

1. Go to the directory to which you downloaded [Quartus Pro Programmer 18.1](#).

```
cd /home/<user>/Downloads.
```



2. Run the Quartus Pro Programmer Setup file:

```
sudo chmod +x <YourQuartusFileName>.run  
sudo ./ <YourQuartusFileName>.run
```

Installation Directory: /home/<user>/intelFPGA_pro/18.1

4.2.4.2 Install the OpenVINO™ Core Components

1. Go to the directory to which you downloaded the OpenVINO toolkit. The default directory is ~/Downloads, and the default filename is

```
l_openvino_toolkit_fpga_p_<version>.tgz.
```

If you used a different directory or renamed the file, change the following instructions according to your naming conventions.

```
cd ~/Downloads
```


Mustang-F100-A10

2. Unpack the .tgz file:

```
tar -xf l_openvino_toolkit_fpga_p_<version>.tgz
```

A directory named l_openvino_toolkit_fpga_p_<version> is created.

3. Go to the l_openvino_toolkit_fpga_p_<version> directory:

```
cd l_openvino_toolkit_fpga_p_<version>
```

4. Run a script named install_openvino_dependencies.sh

```
sudo ./install_openvino_dependencies.sh
```

This script downloads and installs the external software dependencies.

5. Choose between installing with or without a GUI. Only the visual aspects are different between these options. Choose ONE option:

If you want to use a GUI installation wizard to prompt you for input:

```
sudo ./install_GUI.sh
```

6. Follow the instructions on your screen.

The base installation is complete. Continue to the next section to set the environment variables.

4.2.4.3 Install FPGA driver

1. Download fpga_support_files.tgz from [Intel Resource Center](#). The files are required to ensure that the FPGA card and OpenVino work correctly.

2. Go to the directory where the file is being downloaded and unpack the file:

```
tar -xvzf fpga_support_files.tgz
```

3. Switch to superuser:

```
sudo su
```

4. Change directory to fpga_support_files:

```
cd /home/<user>/Downloads/fpga_support_files/
```

5. Source the setup_env.sh script from the fpga_support_files to setup the environment variables.

```
source /home/<user>/setup_env.sh
```

6. Run fpga_dependencies script to allow OpenCL to support Ubuntu and recent kernel versions:

```
./install_openvino_fpga_dependencies.sh
```

Then chose “3” Intel Vision Accelerator Design with Intel Altera 10 FPGA (IEI Mustang-F100-A10).

```
root@iei-SER0:/home/iei/Downloads/fpga_support_files# ./install_openvino_fpga_dependencies.sh
Would you like to setup an FPGA? Select an option below.
Select (1) for None, (2) Intel® Arria® 10 GX FPGA Development Kit, (3) for Intel® Vision Accelerator D
esign with Intel® Arria® 10 FPGA (IEI Mustang-F100-A10):
1) none
2) Intel® Arria® 10 GX FPGA Development Kit
3) Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA (IEI Mustang-F100-A10)
#? 3
Selected:
```

Then chose “y” to install OpenVINO GPU driver (optional)

```
The NEO OpenCL GPU driver is required for using the GPU with OpenVINO. Would you like to install it?
Please select (y/n)
y
```

Then chose “y” to install OpenVINO Movidius (VPU) driver (optional)

```
Do you want to install Movidius USB Rules?
Please select (y/n)
y
```

Note: The OpenVINO™ environment variables are removed when you close the shell.
As an option, use your preferred method to permanently set the variables

9. Run the command:

```
aocl install
```

Mustang-F100-A10

```

root@iei-SER0:/home/iei/Downloads/fpga_support_files# aocl install
Do you want to install /opt/altera/aocl-pro-rte/aclrte-linux64/board/hddlf_1150_sg1? [y/n] y
aocl install: Running install from /opt/altera/aocl-pro-rte/aclrte-linux64/board/hddlf_1150_sg1/linux64/libexec
Looking for kernel source files in /lib/modules/4.15.0-38-generic/build
Using kernel source files from /lib/modules/4.15.0-38-generic/build
Building driver for BSP with name a10_1150_sg1
make: Entering directory '/usr/src/linux-headers-4.15.0-38-generic'
  CC [M] /tmp/opencl_driver_vA72Vq/aclpci_queue.o
  CC [M] /tmp/opencl_driver_vA72Vq/aclpci.o
  CC [M] /tmp/opencl_driver_vA72Vq/aclpci_fileio.o
  CC [M] /tmp/opencl_driver_vA72Vq/aclpci_dma.o
  CC [M] /tmp/opencl_driver_vA72Vq/aclpci_pr.o
  CC [M] /tmp/opencl_driver_vA72Vq/aclpci_cmd.o
  LD [M] /tmp/opencl_driver_vA72Vq/aclpci_a10_1150_sg1_drv.o
Building modules, stage 2.
MODPOST 1 modules
  CC /tmp/opencl_driver_vA72Vq/aclpci_a10_1150_sg1_drv.mod.o
  LD [M] /tmp/opencl_driver_vA72Vq/aclpci_a10_1150_sg1_drv.ko
make: Leaving directory '/usr/src/linux-headers-4.15.0-38-generic'
root@iei-SER0:/home/iei/Downloads/fpga_support_files# █

```

11. Run the command:

```
aocl diagnose
```

You should be seeing DIAGNOSTIC_PASSED before proceeding to the next steps.

```

root@iei-SER0:/home/iei/Downloads/fpga_support_files# aocl diagnose
-----
Device Name:
acl0

Package Pat:
/opt/altera/aocl-pro-rte/aclrte-linux64/board/hddlf_1150_sg1

Vendor: Intel(R) Corporation

Phys Dev Name  Status  Information
acla10_1150_sg10Passed  Intel Vision Accelerator Design with Intel Arria 10 FPGA (acla10_1150_sg10)
PCIe dev_id = 2494, bus:slot.func = 01:00.00, Gen3 x8
FPGA temperature = 53.3438 degrees C.

DIAGNOSTIC_PASSED
-----

Call "aocl diagnose <device-names>" to run diagnose for specified devices
Call "aocl diagnose all" to run diagnose for all devices
root@iei-SER0:/home/iei/Downloads/fpga_support_files# █

```

This step uses the Intel FPGA RTE for OpenCL. To program the AOCX file with FP11 or FP16 bitstreams:

```
aocl program acl0 ${BITSTREAM_DATA_TYPE}.aocx
```

Bitstream
FP11
2019R1_PL1_FP11_AlexNet_GoogleNet
2019R1_PL1_FP11_ELU
2019R1_PL1_FP11_MobileNetCaffe
2019R1_PL1_FP11_MobileNet_Clamp
2019R1_PL1_FP11_ResNet_SqueezeNet_VGG
2019R1_PL1_FP11_RMNet
2019R1_PL1_FP11_SSD300_TinyYolo
FP16
2019R1_PL1_FP16_AlexNet_GoogleNet_SSD300_TinyYolo
2019R1_PL1_FP16_MobileNet_Clamp
2019R1_PL1_FP16_ResNet_SqueezeNet_VGG_ELU
2019R1_PL1_FP16_RMNet

Table 4-3: OpenVINO 2019 R1Bitstream with Topology

Chapter

5

Configure and Use the Model Optimizer

5.1 Configure the Model Optimizer

You must configure the Model Optimizer for the framework that was used to train your model. Follow the steps in this section to use scripts to configure the Model Optimizer for the Caffe framework.

NOTE: As an option, you can manually configure the Model Optimizer instead of following these steps. If this is your choice, see the Custom Layers section of the [Model Optimizer Developer Guide](#).

1. Go to the Model Optimizer prerequisites directory:

```
cd /deployment_tools/model_optimizer/install_prerequisites
```
2. If you want to run the script for the Caffe model framework:

```
sudo ./install_prerequisites_caffe.sh
```
3. If you want to run the script for the MXNet model framework:

```
sudo ./install_prerequisites_mxnet.sh
```
4. If you want to run the script for the TensorFlow model framework:

```
sudo ./install_prerequisites_tf.sh
```
5. There are some most popular public models created by the open developer community that are available in Model Downloader. Make sure you have `sudo pip install yaml` before running `downloader.py` file. Find the model downloader in the OpenVINO toolkit folder:

```
cd  
/opt/intel/computer_vision_sdk_fpga_<version>/deployment_tools  
/model_downloader/  
./downloader.py
```

5.2 Use the Model Optimizer

Before you use the Inference Engine APIs, you must use the Model Optimizer to create the Intermediate Representation (IR) files from your pre-trained Caffe model. For this conversion, the Model Optimizer Python script converts the prototxt and caffemodel files to generate .xml and .bin topology files that describe the network.

The result is two files:

- Topology file – a .xml file that describes the network topology
- Trained data file – a .bin file that contains the weights and biases binary data

NOTE: For information about the Model Optimizer command line arguments and options:

```
python3 mo_caffe.py --help.
```

1. Temporarily set the environment variables

```
source /opt/intel/computer_vision_sdk_<VERSION>/bin/setupvars.sh
```

NOTE: The OpenVINO™ environment variables are removed when you close the shell. As an option, use your preferred method to permanently set the variables.

2. Get the mean file for the AlexNet or ResNet topology. This file provides optimized performance.

- AlexNet mean file location:
http://dl.caffe.berkeleyvision.org/caffe_ilsvrc12.tar.gz
- ResNet mean file location:
<https://github.com/ry/tensorflow-resnet/tree/master/data>

3. Go to the Model Optimizer directory:

```
cd  
/opt/intel/computer_vision_sdk_fpga_<VERSION>/deployment_tools  
/model_optimizer
```

4. Run `mo_caffe.py` on the `caffemodel` and `prototxt` files that have the data type that you need. FP11 bitstreams use data type FP16 when generating the IR files:

-- For AlexNet or ResNet:

```
python3 mo_caffe.py --input_model $<CAFFEMODEL> --input_proto  
$<PROTOTXT_FILE> -n $<NAME_OUT> --data_type $<DATA_TYPE> --scale  
1 --mean_file $<MEAN_FILE> --output_dir $<XML_PATH>
```

-- For GoogleNet, SqueezeNet, VGG16, or SSD300 topology, provide the mean value for optimized performance:

```
python3 mo_caffe.py --input_model $<CAFFEMODEL> --input_proto  
$<PROTOTXT_FILE> -n $<NAME_OUT> --data_type $<DATA_TYPE> --scale  
1 --mean_value [104,117,123] --output_dir $<XML_PATH>
```

-- For MobileNet v1 and MobileNet v2 topology, provide the scale factor and mean value for optimized performance:

```
python3 mo_caffe.py --input_model $<CAFFEMODEL> --input_proto  
$<PROTOTXT_FILE> -n $<NAME_OUT> --data_type $<DATA_TYPE> --scale  
58.824 --mean_value [104,117,123] --output_dir $<XML_PATH>
```

Note:

For more information on model optimizer to convert to Caffe models, refer to [Model Optimizer to Convert Caffe* Models](#)

For more information on model optimizer to convert to MXNet models, refer to [Model Optimizer to Convert MXNet* Models](#)

For more information on model optimizer to convert to TensorFlow models, refer to [Model Optimizer to Convert TensorFlow* Models](#)

Chapter

6

Build the Sample Applications

This section uses cmake to build the sample applications.

1. Temporarily set the environment variables:

```
source
/opt/intel/computer_vision_sdk_<VERSION>/bin/setupvars.sh
```

NOTE: The OpenVINO™ environment variables are removed when you close the shell. As an option, use your preferred method to permanently set the variables.

2. Go to the Inference Engine samples directory:

```
cd
/opt/intel/computer_vision_sdk_fpga_<VERSION>/deployment_tools
/inference_engine/samples/
```

3. Create a build directory:

```
mkdir build
```

4. Go to the Inference Engine samples build directory:

```
cd
/opt/intel/computer_vision_sdk_fpga_<VERSION>/deployment_tools
/inference_engine/samples/build/
```

5. Run cmake to generate the Makefiles without debugging information:

```
sudo cmake -DCMAKE_BUILD_TYPE=Release
/opt/intel/computer_vision_sdk_fpga_<version>/deployment_tools
/inference_engine/samples/
```

6. Build the sample applications:

```
make

make install
```

7. Confirm the build exists. If this directory exists, your build was successful:

```
cd
/opt/intel/computer_vision_sdk_fpga_<VERSION>/deployment_tools
/inference_engine/samples/build/intel64/Release/
```

The existence of this directory confirms you successfully completed the steps in this section.

Chapter

7

Use the Sample Applications

**IMPORTANT:**

You must have completed the previous sections in this document before you will be successful using the sample applications.

For command-line arguments and options used with the sample applications:

```
python3 mo_caffe.py -help
```

7.1 classification_async_Sample with Maximum Optimization

■ AlexNet topology example

```
cd
/opt/intel/computer_vision_sdk_fpga_<OPENVINO_VERSION>/deployment_tools/inference_engine/samples/build/intel64/Release/

export CL_CONTEXT_COMPILER_MODE_INTELFPGA=3

sudo cp
/opt/intel/computer_vision_sdk_fpga_<OPENVINO_VERSION>/deployment_tools/demo/squeezenet1.1.labels $<XML_PATH>

mv squeezenet1.1.labels alexnet_fp16.labels

./classification_sample_async -m $<XML_PATH>/alexnet_fp16.xml -i
$<IMAGE_PATH> -d HETERO:FPGA,CPU -ni $<ITERATION_NUMBER> -nireq
2
```

■ AlexNet topology example with a batch size of 96

```
cd
/opt/intel/computer_vision_sdk_fpga_<OPENVINO_VERSION>/deployment_tools/inference_engine/samples/build/intel64/Release/

export CL_CONTEXT_COMPILER_MODE_INTELFPGA=3

sudo cp
/opt/intel/computer_vision_sdk_fpga_<OPENVINO_VERSION>/deployment_tools/demo/squeezenet1.1.labels $<XML_PATH>

mv squeezenet1.1.labels alexnet_fp16.labels

./classification_sample_async -m $<XML_PATH>/alexnet_fp16.xml
`for i in {1..96}; do echo -n "<IMAGE_PATH>";done` -d
```

Mustang-F100-A10

```
HETERO:FPGA,CPU -ni $<ITERATION_NUMBER> -nireq 2
```

The output example shows the classification_async with data type FP16, 1000 iterations and nireq set to 2 for the AlexNet topology.

```
[ INFO ] InferenceEngine:
```

```
    API version ..... 1.4
```

```
    Build ..... 16050
```

```
[ INFO ] Parsing input parameters
```

```
[ INFO ] Parsing input parameters
```

```
[ INFO ] Files were added: 1
```

```
[ INFO ]
```

```
/opt/intel/computer_vision_sdk_2018.4.420/deployment_tools/demo/car.png
```

```
[ INFO ] Loading plugin
```

```
    API version ..... 1.4
```

```
    Build ..... heteroPlugin
```

```
    Description ..... heteroPlugin
```

```
[ INFO ] Loading network files
```

```
[ INFO ] Preparing input blobs
```

```
[ WARNING ] Image is resized from (787, 259) to (227, 227)
```

```
[ INFO ] Batch size is 1
```

```
[ INFO ] Preparing output blobs
```

```
[ INFO ] Loading model to the plugin
```

```
[ INFO ] Start inference (100 iterations)
```

```
[ INFO ] Processing output blobs
```

Top 10 results:

Image /opt/intel/computer_vision_sdk_2018.4.420/deployment_tools/demo/car.png

```
479 0.7527428 label car wheel
```

```
511 0.0757053 label convertible
```

```
436 0.0745316 label beach wagon, station wagon, wagon, estate car, beach waggon,
```

station waggon, waggon
817 0.0466407 label sports car, sport car
656 0.0310694 label minivan
661 0.0056141 label Model T
581 0.0031988 label grille, radiator grille
468 0.0030763 label cab, hack, taxi, taxicab
717 0.0023221 label pickup, pickup truck
627 0.0016857 label limousine, limo

Top 10 results:

Image /opt/intel/computer_vision_sdk_2018.4.420/deployment_tools/demo/car.png

479 0.7527428 label car wheel
511 0.0757053 label convertible
436 0.0745316 label beach wagon, station wagon, wagon, estate car, beach waggon,
station waggon, waggon
817 0.0466407 label sports car, sport car
656 0.0310694 label minivan
661 0.0056141 label Model T
581 0.0031988 label grille, radiator grille
468 0.0030763 label cab, hack, taxi, taxicab
717 0.0023221 label pickup, pickup truck
627 0.0016857 label limousine, limo

total inference time: 1048.9667654

Throughput: 95.3319050 FPS

[INFO] Execution successful

Mustang-F100-A10

7.2 object_detection_ssd

SSD300 topology:

```
cd
/opt/intel/computer_vision_sdk_fpga_<VERSION>/deployment_tools
/inference_engine/samples/build/intel64/Release/

export CL_CONTEXT_COMPILER_MODE_INTELFPGA=3

./object_detection_sample_ssd -m $<XML_PATH> -i $<IMAGE_PATH> -d
HETERO:FPGA,CPU -i
$<OPENVINO_INSTALLATION>/deployment_tools/inference_engine/sam
ples/build/intel64/Release/lib/libcpu_extension.so
```

7.3 Other Demos

Other sample application are available to run on the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA. For information on how to run the demos in OpenVINO™ toolkit, see the [Inference Engine Sample documentation](#).

For more information on pre-trained models available, refer to [Intel Pre-trained Models](#).

Sample Application	Model Used
classification_sample	Model downloader - AlexNet
classification_sample_async	Model downloader - AlexNet
hello_autoresize_classification	Model downloader - AlexNet
hello_request_classification	Model downloader - AlexNet
interactive_face_detection_sample	face-detection-retail-0004 age-gender-recognition-retail-0013 head-pose-estimation-adas-0001
security_barrier_camera_sample	vehicle-license-plate-detection-barrier-0007 vehicle-attributes-recognition-barrier-0010 license-plate-recognition-barrier-0001
object_detection_demo	faster_rcnn_vgg16
object_detection_sample_ssd	person-detection-retail-0013
object_detection_demo_ssd_async	person-detection-retail-0014
validation_app	Model downloader - AlexNet
segmentation_demo	fcn8_FP16
multi-channel-demo	face-detection-retail-0004
benchmark_app	person-vehicle-bike-detection-crossroad-0078

Table 7-1: Other Sample Applications You can run with the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA

Chapter

8

IEI Mustang Viewer Utility

Mustang-F100-A10

8.1 Installation Requirement

Hardware Requirement:

- Mustang-F100-A10 acceleration card
- Personal computer with PCIe x8 slot or above
- USB to Micro USB cable

Procedure:

1. Power off PC
2. Install a Mustang-F100-A10 into a PCIe 3.0 x8 slot (or above)
3. Plug the USB connector of the USB cable into the PC; plug the Micro USB connector into the Micro USB port of the Mustang-F100-A10
4. Power on the PC

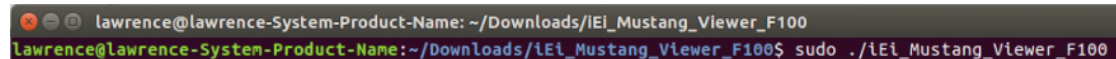
Software Requirement:

- OS: Ubuntu 16.04.03 with Kernel version 4.15.0.38
- iEi_Mustang_Viewer_F100: it is an utility program that uses USB HID interface to transfer data from Mustang-F100-A10 (ex. FPGA temperature) to PC for monitoring hardware status

Procedure:

1. Get the zip file - iEi_Mustang_Viewer_F100_V1.0.0.xxxxx.yyyyymmdd.tar.gz
2. Unzip the file. Open the Terminal window, and go to the directory of the unzipped file
3. Use the following command to launch the program

```
sudo ./iEi_Mustang_Viewer_F100
```

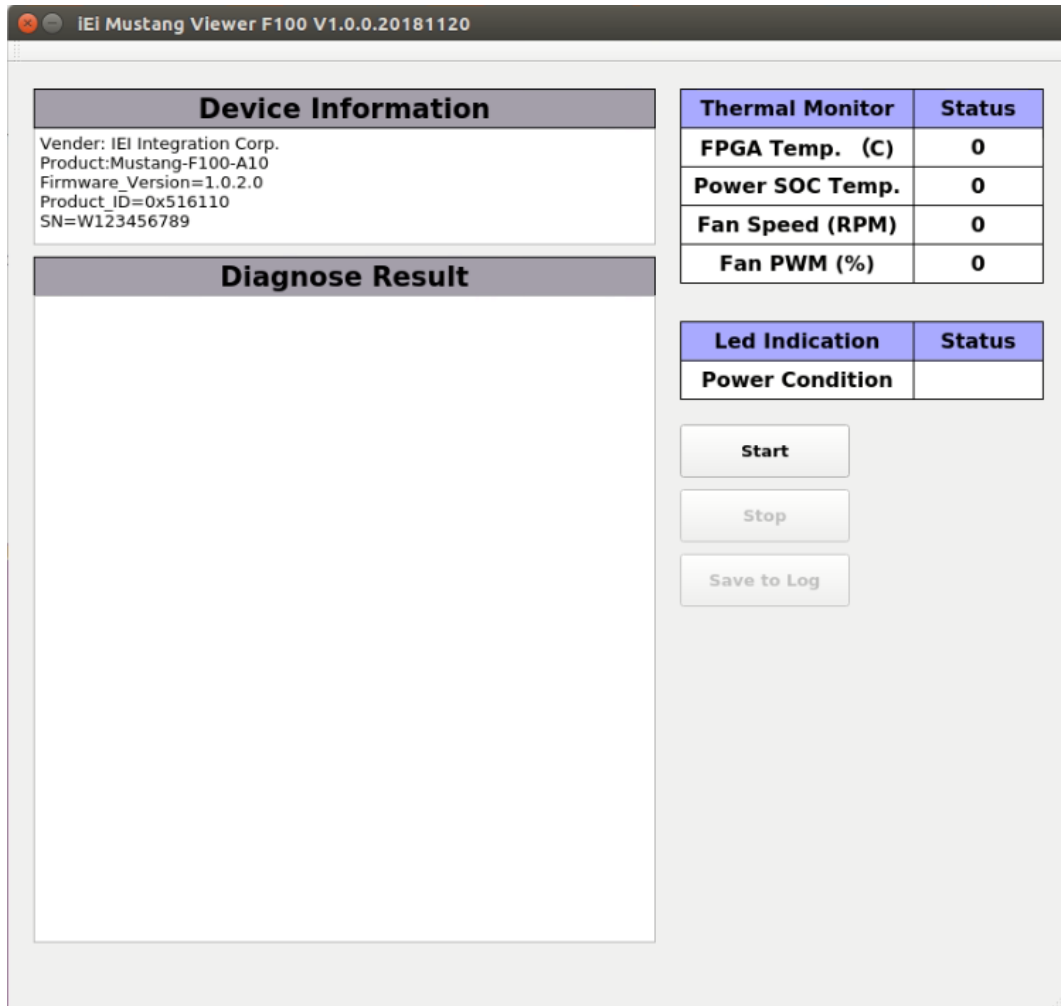


```
lawrence@lawrence-System-Product-Name: ~/Downloads/iEi_Mustang_Viewer_F100
lawrence@lawrence-System-Product-Name:~/Downloads/iEi_Mustang_Viewer_F100$ sudo ./iEi_Mustang_Viewer_F100
```

8.2 Device Information

The following screen appears after IEI Mustang Viewer is launched. The Device Information section provides information of the Mustang-F100-A10, including:

- **Vendor:** shows the manufacturer of the Mustang-F100-A10
- **Product:** shows the model name of the installed acceleration card
- **Firmware_Version:** shows the firmware version of the Mustang-F100-A10
- **Product_ID:** shows the product ID of the Mustang-F100-A10; this ID corresponds to the model name
- **SN:** shows the product serial number of the Mustang-F100-A10



Device Information	
Vender: IEI Integration Corp.	
Product: Mustang-F100-A10	
Firmware_Version=1.0.2.0	
Product_ID=0x516110	
SN=W123456789	

Diagnose Result	

Thermal Monitor	Status
FPGA Temp. (C)	0
Power SOC Temp.	0
Fan Speed (RPM)	0
Fan PWM (%)	0

Led Indication	Status
Power Condition	

Start

Stop

Save to Log

Mustang-F100-A10

8.3 Diagnose

The user can press the **Start** button on IEI Mustang Viewer to start diagnosing the Mustang-F100-A10. The program will update and display the result approximately every one second.

- FPGA_protection_Alert_temp: the temperature value that triggers FPGA alert
- FPGA_protection_ShutDown_temp: the temperature value that triggers FPGA shutdown
- FPGA_Diode_Temperature: current operating temperature of FPGA
- EM2280_protection_Alert_temp: the temperature value that triggers EM2280 alert
- EM2280_protection_Shutdown_temp: the temperature value that triggers EM2280 shutdown
- EM2280_Power_Train_Temp: current operating temperature of EM2280 Power SOC
- EM2280_VIN: input voltage of EM2280 Power SOC
- EM2280_VOUT: output voltage of EM2280 Power SOC
- EM2280_IOUT: output current of EM2280 Power SOC
- EM2280_Controller_Temp: current operating temperature of EM2280 Power SOC controller
- EM2130_protection_Alert_temp: the temperature value that triggers EM2130 alert
- EM2130_protection_Shutdown_temp: the temperature value that triggers EM2130 shutdown
- EM2130_VIN: input voltage of EM2130 Power SOC
- EM2130_VOUT: output voltage of EM2130 Power SOC
- EM2130_IOUT: output current of EM2130 Power SOC
- EM2130_Temp: current operating temperature of EM2280 Power SOC
- Fan_PWM: current fan PWM value
- Fan_Speed_RPM: fan speed RPM value
- Card_ID: card ID value of the card; the card ID is adjusted by the DIP switch on the card
- LED_Status_PowerLed: LED power status; the value is 1 in the normal state

- POWER_CONDITION_VTT_0V6: VTT power status; it shows “Good” in the normal state
- POWER_CONDITION_VCC_12V: VCC 12V power status; it shows “Good” in the normal state
- POWER_CONDITION_VCC_05V: VCC 0.5V power status; it shows “Good” in the normal state
- POWER_CONDITION_VCC_3V3: VCC 3.3V power status; it shows “Good” in the normal state
- POWER_CONDITION_FPGA_CORE_0V9: FPGA Core power status; it shows “Good” in the normal state
- POWER_CONDITION_VCCT_1V03: VCCT 1.03V power status; it shows “Good” in the normal state
- POWER_CONDITION_VCCH_GXB_1V8: VCCH GXB 1.8V power status; it shows “Good” in the normal state
- POWER_CONDITION_VCC_1V8: VCC 1.8V power status; it shows “Good” in the normal state
- POWER_CONDITION_VDDQ_1V2: VDDQ 1.2V power status; it shows “Good” in the normal state

Mustang-F100-A10

iEi Mustang Viewer F100 V1.0.0.20181120

Device Information
Vender: IEI Integration Corp. Product: Mustang-F100-A10 Firmware_Version=1.0.2.0 Product_ID=0x516110 SN=W123456789

Diagnose Result
<pre> FPGA_protection_Alert_temp=105 FPGA_protection_ShutDown_temp=110 FPGA_Diode_Temperature=53 EM2280_protection_Alert_temp=110.00 EM2280_protection_Shutdown_temp=115.00 EM2280_Power_Train_Temp=46.37 EM2280_VIN=11.76 EM2280_VOUT=0.95 EM2280_IOUT=14.78 EM2280_Controller_Temp=47.25 EM2130_protection_Alert_temp=110.00 EM2130_protection_Shutdown_temp=115.00 EM2130_VIN=11.71 EM2130_VOUT=3.29 EM2130_IOUT=4.22 EM2130_Temp=52.75 Fan_PWM=69 Fan_Speed_RPM=5278 Card_ID=0 LED_Status_PowerLed=1 POWER_CONDITION_VTT_0V6=Good POWER_CONDITION_VCC_12V=Good POWER_CONDITION_VCC_05V=Good POWER_CONDITION_VCC_3V3=Good POWER_CONDITION_FPGA_CORE_0V9=Good POWER_CONDITION_VCCT_1V03=Good POWER_CONDITION_VCCH_GXB_1V8=Good POWER_CONDITION_VCC_1V8=Good POWER_CONDITION_VDDQ_1V2=Good </pre>

Thermal Monitor	Status
FPGA Temp. (C)	53
Power SOC Temp.	46.37
Fan Speed (RPM)	5278
Fan PWM (%)	69

Led Indication	Status
Power Condition	


Start

Stop

Save to Log

Some of the diagnose results will be listed simultaneously in the table on the right.

Thermal Monitor	Status
FPGA Temp. (C)	53
Power SOC Temp.	46.37
Fan Speed (RPM)	5278
Fan PWM (%)	69

Led Indication	Status
Power Condition	

Thermal Monitor

- FPGA Temp: shows FPGA_Diode_Temperature (current operating temperature of FPGA). The background color shows different states:
 - Green: normal
 - Yellow: over alert temperature
 - Red: over shutdown temperature
- Power Soc Temp: shows EM2280Parameter_Power_Train_Temp. The background color shows different states:
 - Green: normal
 - Yellow: over alert temperature
 - Red: over shutdown temperature
- Fan_Speed (RPM): fan speed RPM value
- Fan_PWM: current fan PWM value

Led Indication

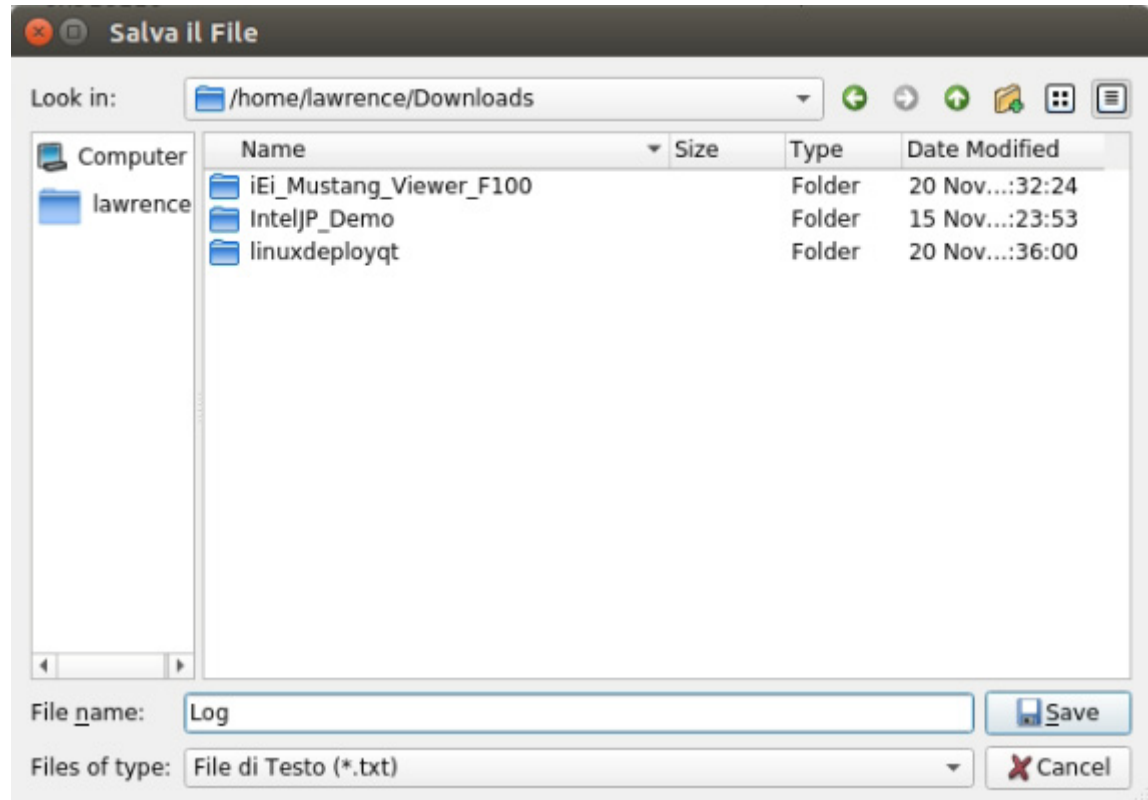
- Power Condition: shows LED power status (LED_Status_PowerLed).
 - Blue: normal
 - Gray: abnormal

NOTE: To stop diagnosing the card, press the **Stop** button on IEI Mustang Viewer.

Mustang-F100-A10

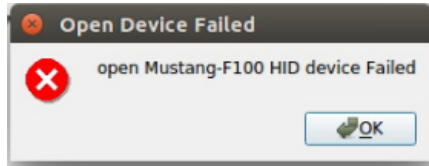
8.4 Saving Log File

Device information and diagnose result can be saved as a log file when the program is not updating information (in Stop mode). To do this, press the **“Save to Log”** button and the **Save File** window will appear. Select a directory, enter a file name, and press the **“Save”** button to save the file.



8.5 Troubleshooting

This section provides a troubleshooting suggestion when you are prompted with the “Open Device Failed” window (as shown below).



The “Open Mustang-F100 HID device failed” issue might be caused by the following reasons:

1. Using the `./iEi_Mustang_Viewer_F100` command to open the program in the unzipped folder (insufficient permissions)
2. Using a mouse to click on `iEi_Mustang_Viewer_F100` to open the program (insufficient permissions)
3. Failing to connect the USB-to-microUSB cable to the PC or the accelerator card

How to solve this issue:

1. Open the Terminal window
2. Go to the unzipped folder
3. Open the program with the following command

```
sudo ./iEi_Mustang_Viewer_F100
```

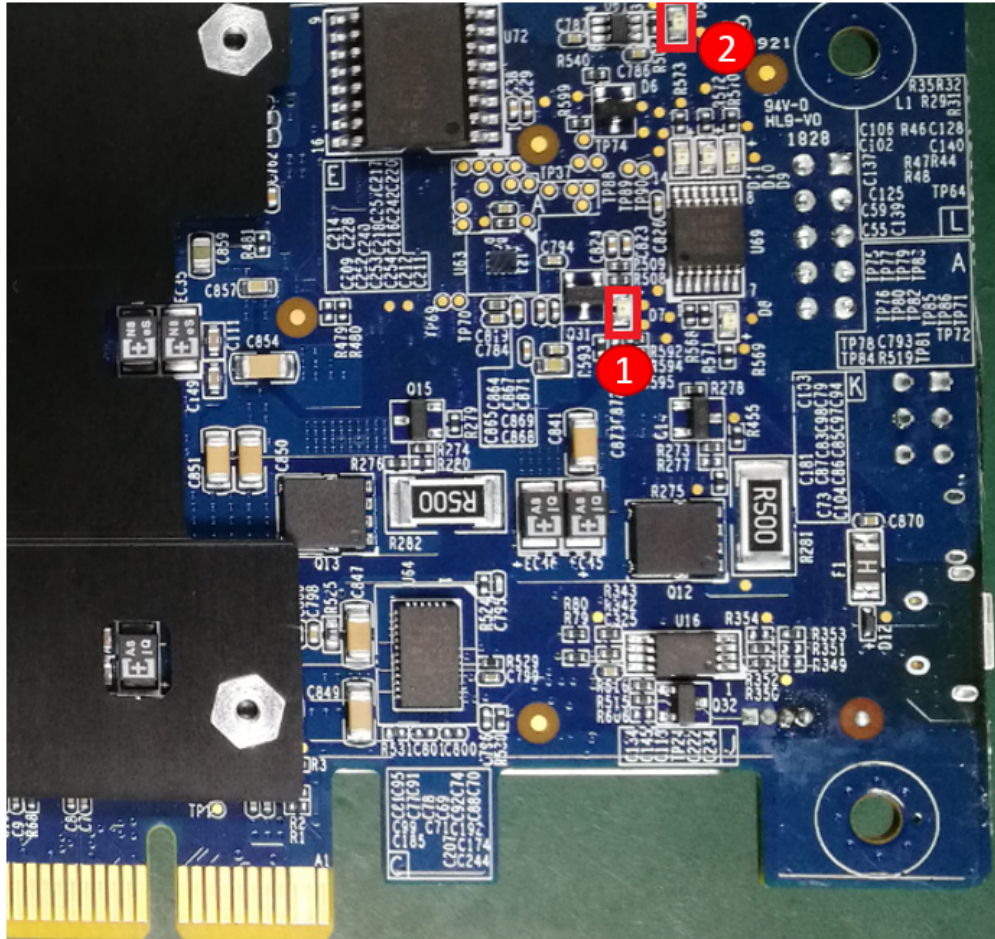

Appendix

A

LED Indicators

A.1 On-board LED Indicators and Purpose

Users can use the on-board LEDs to troubleshoot the board.



LED 1: Green light indicates power-good status after boot-up

LED 2: Green light indicates FPGA configuration has completed

Appendix

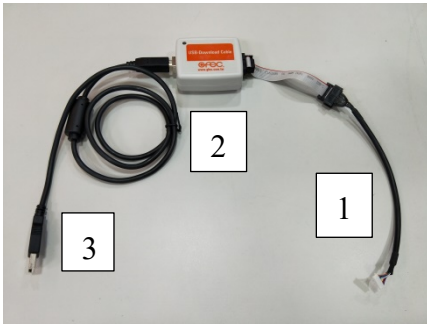
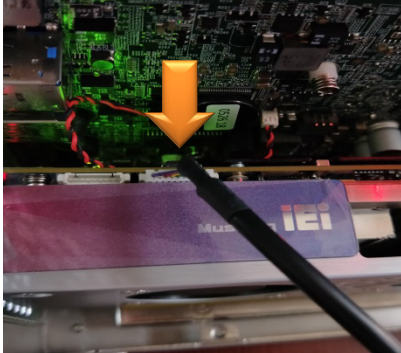

B

System Recovery

B.1 Required Hardware

If PCIe is not detected and using lspci and aocl to diagnose is failed or the command aocl program aocl0 <.aocx> is failed to update bitstreams , you can use a recovery procedure to re-program the board. The required hardware devices include:

Note: It is mandatory for user to implement FPGA programmer kit to update FPGA bitstreams if your target OpenVINO toolkit version does not match the Mustang-F100-A10 card's bitstreams.

Description	Image
<p>FPGA programmer kit</p> <p>Package is included:</p> <ul style="list-style-type: none"> - Intel FPGA download cable - USB cable - USB Blaster 	
<p>Step1</p> <ul style="list-style-type: none"> - Connect Intel FPGA download cable to Mustang-F100-A10 FPGA connector 	
<p>Step2</p> <ul style="list-style-type: none"> - Connect USB cable to host PC USB connector 	

Mustang-F100-A10

B.2 Recovery Steps



NOTE:

boardtest_2ddr base.sof and boardtest_2ddr_top.aocx are not supported in this release. See the README file for more information.

Step 1: Download and install the [Intel® Quartus® Prime Pro Edition Programmer, version 17.1.1.](#)

Software Update Only

Use this option if you already have the Quartus Prime software installed and just want the updates.

Software and IP Updates (Latest)

Quartus Prime Software v17.1 Update 2

*You must have the base software installed before installing the update.

***Important Note: For Winzip users, you need to download version 22 or newer if your existing WinZip software cannot unzip the tar file, you also need to turn the TAR file smart CR/LF conversion option off to make it work correctly. The option can be found through settings menu -> WinZip options -> Advanced or Miscellaneous tab.**

Size: 16.0 GB MD5: 3AF6DF9B2B78A03F28ABDB29C84666A4

Intel FPGA SDK for OpenCL Pro Edition v17.1 Update 2

Size: 893.7 MB MD5: 0E7438C385BBE79D92E9CF2B32EE64A2

Intel FPGA Runtime Environment for OpenCL Pro Edition v17.1 Update 2

Size: 9.9 MB MD5: DC23E1A08EF5F9F1055BBB620C16C36C

DSP Builder Pro Edition v17.1 Update 2

Size: 56.3 MB MD5: 97FECC882AFD41CBFFBCA720134D013

Quartus Prime Pro Edition Programmer and Tools v17.1 Update 2

Size: 437.6 MB MD5: B6A32565EDDB60AC077D560D2196EF89

Intel FPGA Runtime Environment for OpenCL Linux x86-64 RPM

Size: 2.0 MB MD5: B44D9DCF7BDAF882633F9F76A2A9BBE4

Intel FPGA Runtime Environment for OpenCL Linux Cyclone V SoC TGZ

Size: 1.0 MB MD5: 1EAD0EBAB7557DD95E06B1C6C0A7E001

Intel FPGA Runtime Environment for OpenCL Linux x86-64

Size: 9.9 MB MD5: DC23E1A08EF5F9F1055BBB620C16C36C

Intel FPGA Runtime Environment for OpenCL Windows x86-64

Size: 11.9 MB MD5: 6EB99E69155945FAF64AF0D79E7BEE7

Complete Download

Use this option if you do not have the latest version of the Quartus Prime software installed and want to download the software and the update together. **Please note, this complete package only includes update to Quartus software. If you need updates to other products such as Intel FPGA SDK for OpenCL or DSP Builder, you need to download and install them individually. Also note, you need to run the update installer (17.1.2) after you install the base version (17.1).**

Quartus Prime Pro Edition Software Update 2 (Device support included)

Quartus-pro-17.1.2.304-linux-complete.tar

Size: 59.0 GB MD5: C2F0962175A16C626CF8CBDE0AF6FBF0

[Show Archived Software Updates](#)

[Hide Archived Software Updates](#)

Software and IP Updates (Archived)	
Quartus Prime Software v17.1 Update 1 *You must have the base software installed before installing the update. Size: 15.8 GB MD5: 0881721E5ED8CF03D4497D3E80EDBC11	↓
Intel FPGA SDK for OpenCL Pro Edition v17.1 Update 1 Size: 893.7 MB MD5: 23C0B92F59CBB12F0C19FA69D68BCE0A	↓
Intel FPGA Runtime Environment for OpenCL Pro Edition v17.1 Update 1 Size: 9.9 MB MD5: 08C11B2CCB81DFADAA815FE25E2B58DB	↓
DSP Builder Pro Edition v17.1 Update 1 Size: 56.3 MB MD5: E31D4E498415D25973695E5C1EF6112F	↓
Quartus Prime Pro Edition Programmer and Tools v17.1 Update 1 Size: 437.7 MB MD5: 11725D84FE7773D82C06CC56CEE104B7	↓
Intel FPGA Runtime Environment for OpenCL Linux x86-64 RPM Size: 2.0 MB MD5: BE0FF9AE7EFAB27F552621EC56470545	↓
Intel FPGA Runtime Environment for OpenCL Linux Cyclone V SoC TGZ Size: 1.0 MB MD5: 0E658B6133AA902F60B475777BA994CF	↓
Intel FPGA Runtime Environment for OpenCL Linux x86-64 Size: 9.9 MB MD5: 08C11B2CCB81DFADAA815FE25E2B58DB	↓
Intel FPGA Runtime Environment for OpenCL Windows x86-64 Size: 11.9 MB MD5: A759C4300F503A77C54B736E0C3A1F0A	↓
Quartus Prime Pro Edition Software Update 1 (Device support included) ⓘ Size: 58.8 GB MD5: C8231E3B97EC856099F6484630249677	↓

Figure B-1: Intel Software Downloads

Step 2: Add Intel® Quartus® Prime Pro Programmer to your environment variables:

```
export PATH=/opt/intelFPGA_pro/17.1/qprogrammer/bin:$PATH
```

Step 3: Connect the cable between the board and the host system. Use the letter codes in the diagram below for the connection points.

- Connect the B end of the cable to point B on the board.
- Connect the F end of the cable to point F on the FPGA download cable.

Mustang-F100-A10

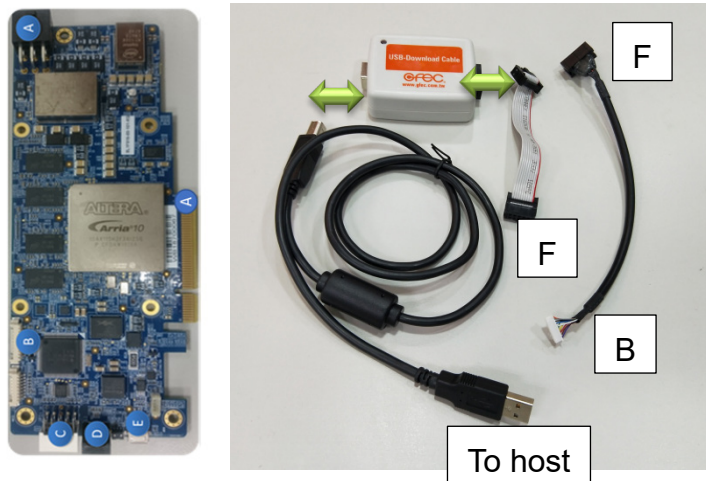


Figure B-2: Connection from JTAG Port to Cable to Intel® FPGA Download Cable

- Step 4:** Update the Intel FPGA Download Cable rules to program the board without root permissions and to flash the initialization bitstreams so that the Intel FPGA Download Cable can communicate with the board.

```
sudo cp 51-usbblaster.rules /etc/udev/rules.d
```

- Step 5:** Download the [51-usbblaster.rules](#) file from the Intel download center.

- Step 6:** Disconnect and reconnect the Intel FPGA Download Cable to enable JTAG connection.

- Step 7:** The BSP files can be found at <BSP_package>.

- Step 8:** Download the .tgz file and decompress the following file:

```
tar -xvf <BSP_package>.tgz
```

These files uncompress to the bringup folder:

- boardtest_1ddr_base.sof
- boardtest_1ddr_top.aocx

Step 9: Run `jtagconfig` to ensure that your Intel FPFA download cable driver is ready to use.

```
jtagconfig
```

Your output is similar to:

```
1) USB-Blaster [1-6]
02E660DD 10AX115H1(.|E2|ES)/10AX115H2/..
```

Step 10: Program the FPGA on the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA board:

```
quartus_pgm -c 1 -m JTAG -o "p;boardtest_1ddr_base.sof
```

Step 11: Reset the host system. The host system recognizes the Intel® Vision Accelerator Design with Intel® Arria® 10 FPGA board.

Confirm you can detect the PCIe card:

```
lspci | grep -i Altera
```

Your output is similar to:

```
01:00.0 Processing accelerators: Altera Corporation Device
2494 (rev 01)
```

Step 12: Run the command:

```
aocl diagnose
```

Optional: Permanently program the flash. Doing so removes the necessity to reprogram `boardtest_1ddr_base.sof` into the FPGA at each reboot.

To accomplish this option, use JTAG and Intel® Quartus® Prime Pro Edition software, version 17.1.1 to program the flash memory. You will need to download the [full Intel® Quartus® Prime Pro Edition software, version 17.1.1](#).

Mustang-F100-A10

```
export QUARTUS_ROOTDIR=${QUARTUS_PATH}/quartus  
aocl flash acl0 boardtest_1ddr_top.aocx
```

Step 13: Power down your host system, and then power it back on.

Step 14: Run aocl diagnose to confirm that the initialization completed successfully.

Success is indicated by a pass status.

Appendix

C

Regulatory Compliance

Mustang-F100-A10

DECLARATION OF CONFORMITY



This equipment has been tested and found to comply with specifications for CE marking. If the user modifies and/or installs other devices in the equipment, the CE conformity declaration may no longer apply.

FCC WARNING



This equipment complies with Part 15 of the FCC Rules. Operation is subject to the following two conditions:

- This device may not cause harmful interference, and
- This device must accept any interference received, including interference that may cause undesired operation.

This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his own expense.

Appendix

D

Product Disposal

Mustang-F100-A10

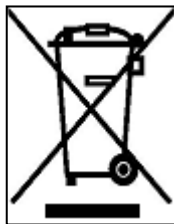


CAUTION:

Risk of explosion if battery is replaced by an incorrect type. Only certified engineers should replace the on-board battery.

Dispose of used batteries according to instructions and local regulations.

- Outside the European Union – If you wish to dispose of used electrical and electronic products outside the European Union, please contact your local authority so as to comply with the correct disposal method.
- Within the European Union – The device that produces less waste and is easier to recycle is classified as electronic device in terms of the European Directive 2012/19/EU (WEEE), and must not be disposed of as domestic garbage.



EU-wide legislation, as implemented in each Member State, requires that waste electrical and electronic products carrying the mark (left) must be disposed of separately from normal household waste. This includes monitors and electrical accessories, such as signal cables or power cords. When you need to dispose of your device, please follow the guidance of your local authority, or ask the shop where you purchased the product. The mark on electrical and electronic products only applies to the current European Union Member States.

Please follow the national guidelines for electrical and electronic product disposal.

Appendix

E

Hazardous Materials Disclosure

Mustang-F100-A10

The details provided in this appendix are to ensure that the product is compliant with the Peoples Republic of China (China) RoHS standards. The table below acknowledges the presences of small quantities of certain materials in the product, and is applicable to China RoHS only.

A label will be placed on each product to indicate the estimated “Environmentally Friendly Use Period” (EFUP). This is an estimate of the number of years that these substances would “not leak out or undergo abrupt change.” This product may contain replaceable sub-assemblies/components which have a shorter EFUP such as batteries and lamps. These components will be separately marked.

Please refer to the following table.

Part Name	Toxic or Hazardous Substances and Elements					
	Lead (Pb)	Mercury (Hg)	Cadmium (Cd)	Hexavalent Chromium (CR(VI))	Polybrominated Biphenyls (PBB)	Polybrominated Diphenyl Ethers (PBDE)
Housing	O	O	O	O	O	O
Display	O	O	O	O	O	O
Printed Circuit Board	O	O	O	O	O	O
Metal Fasteners	O	O	O	O	O	O
Cable Assembly	O	O	O	O	O	O
Fan Assembly	O	O	O	O	O	O
Power Supply Assemblies	O	O	O	O	O	O
Battery	O	O	O	O	O	O

O: This toxic or hazardous substance is contained in all of the homogeneous materials for the part is below the limit requirement in SJ/T11363-2006 (now replaced by GB/T 26572-2011).

X: This toxic or hazardous substance is contained in at least one of the homogeneous materials for this part is above the limit requirement in SJ/T11363-2006 (now replaced by GB/T 26572-2011).

此附件旨在确保本产品符合中国 RoHS 标准。以下表格标示此产品中某有毒物质的含量符合中国 RoHS 标准规定的限量要求。

本产品上会附有“环境友好使用期限”的标签，此期限是估算这些物质“不会有泄漏或突变”的年限。本产品可能包含有较短的环境友好使用期限的可替换元件，像是电池或灯管，这些元件将会单独标示出来。

部件名称	有毒有害物质或元素					
	铅 (Pb)	汞 (Hg)	镉 (Cd)	六价铬 (CR(VI))	多溴联苯 (PBB)	多溴二苯 醚 (PBDE)
壳体	○	○	○	○	○	○
显示	○	○	○	○	○	○
印刷电路板	○	○	○	○	○	○
金属螺帽	○	○	○	○	○	○
电缆组装	○	○	○	○	○	○
风扇组装	○	○	○	○	○	○
电力供应组装	○	○	○	○	○	○
电池	○	○	○	○	○	○

○: 表示该有毒有害物质在该部件所有物质材料中的含量均在 SJ/T 11363-2006 (现由 GB/T 26572-2011 取代) 标准规定的限量要求以下。

X: 表示该有毒有害物质至少在该部件的某一均质材料中的含量超出 SJ/T 11363-2006 (现由 GB/T 26572-2011 取代) 标准规定的限量要求。