Launch the AWS provided FPGA Image (AMI) from AWS Marketplace.

↓

Launch the Instance and connect to it.

↓

Write code in Verilog or VHDL for RTL or optionally using open custom logic (OpenCL) framework e.g py-HDL, C, Java to describe the FPGA Logic.

↓

Test the logic using simulator e.g Xilinx Vivado

↓

Use vivado to synthesize and place/route the FPGA logic to create an FPGA check point (DCP).

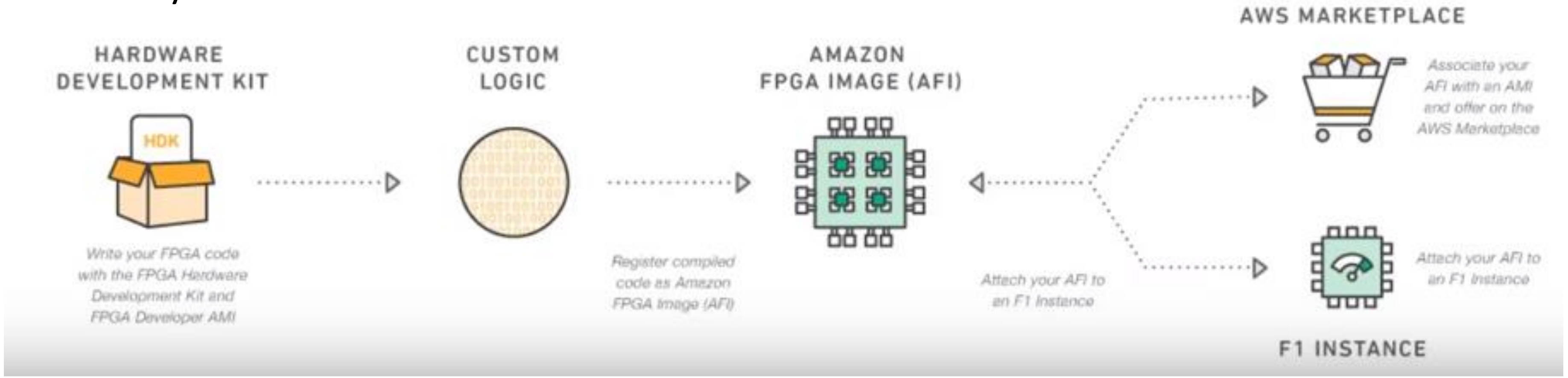
↓

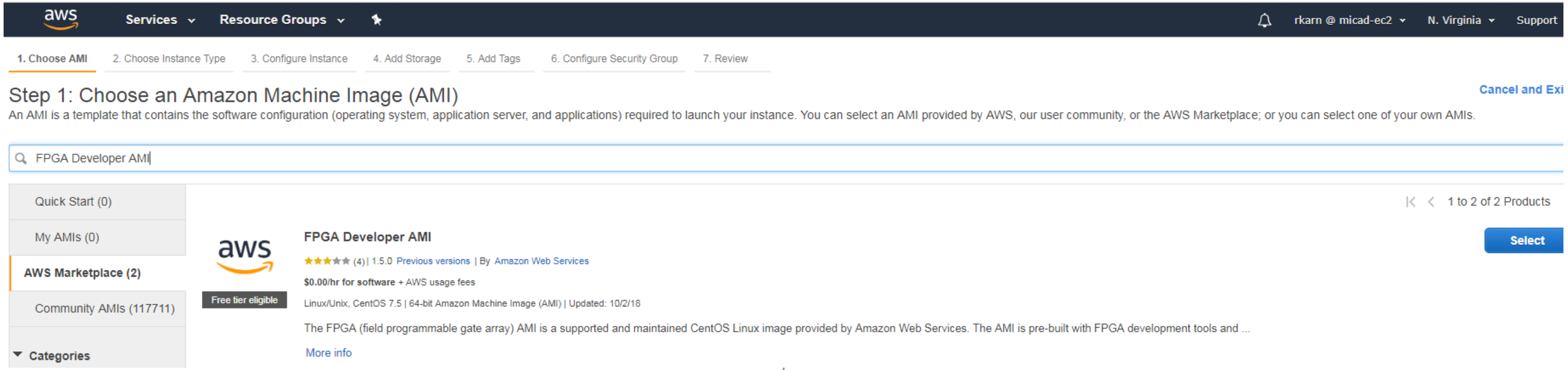
Create an encrypted Amazon FPGA Image (AFI) using the generated DCP.

↓

Load the AFI to FPGA.

Summary:


HARDWARE DEVELOPMENT KIT
HDK
Write your FPGA code with the FPGA Hardware Development Kit and FPGA Developer AMI
CUSTOM LOGIC
Register compiled code as Amazon FPGA Image (AFI)
AMAZON FPGA IMAGE (AFI)
Attach your AFI to an F1 Instance
AWS MARKETPLACE
Associate your AFI with an AMI and offer on the AWS Marketplace
Attach your AFI to an F1 Instance
F1 INSTANCE



Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

FPGA Developer AMI

FPGA Developer AMI

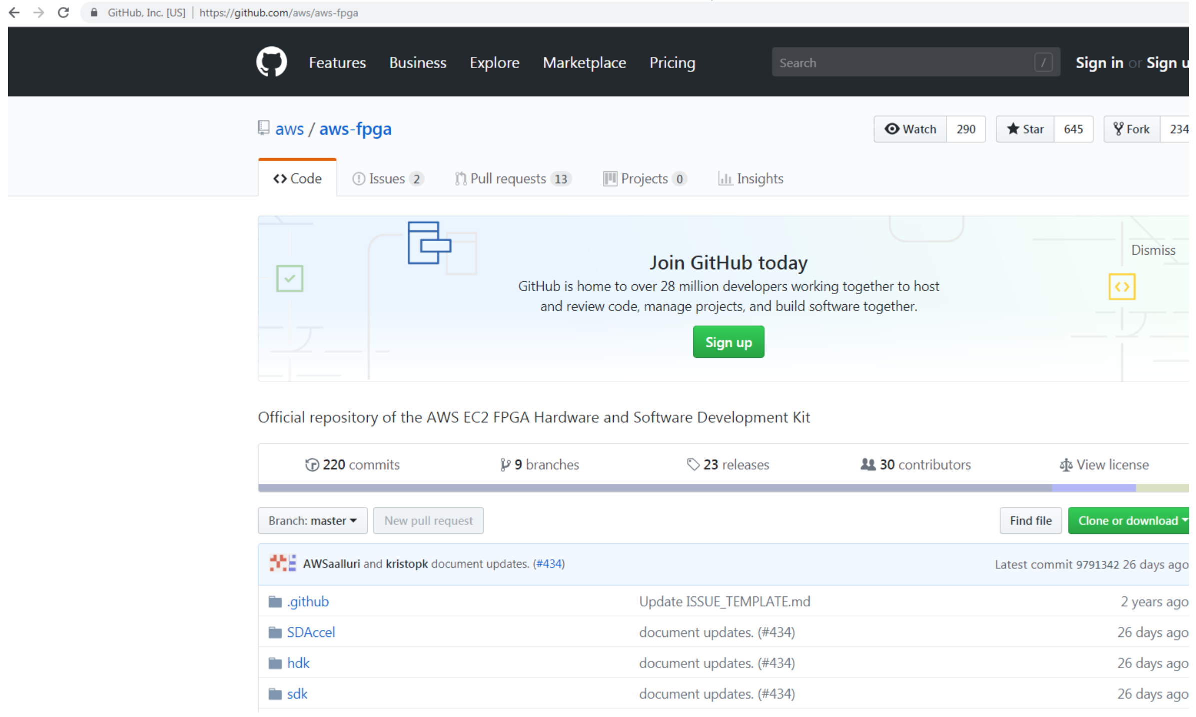
(4) | 1.5.0 Previous versions | By Amazon Web Services

$0.00/hr for software + AWS usage fees

Linux/Unix, CentOS 7.5 | 64-bit Amazon Machine Image (AMI) | Updated: 10/2/18

The FPGA (field programmable gate array) AMI is a supported and maintained CentOS Linux image provided by Amazon Web Services. The AMI is pre-built with FPGA development tools and ...

More info



Join GitHub today

GitHub is home to over 28 million developers working together to host and review code, manage projects, and build software together.

Official repository of the AWS EC2 FPGA Hardware and Software Development Kit

AWSaalluri and kristopk document updates. (#434) Latest commit 9791342 26 days ago

.github	Update ISSUE_TEMPLATE.md	2 years ago
SDAccel	document updates. (#434)	26 days ago
hdk	document updates. (#434)	26 days ago
sdk	document updates. (#434)	26 days ago

Synthesizing the Example with Xilinx Vivado

https://github.com/aws/aws-fpga/blob/master/hdk/cl/examples/cl_hello_world/software/runtime/test_hello_world.c

- Change into an example directory and set the CL_DIR environment variable to the path of the example. You will need to set this again if you change examples.

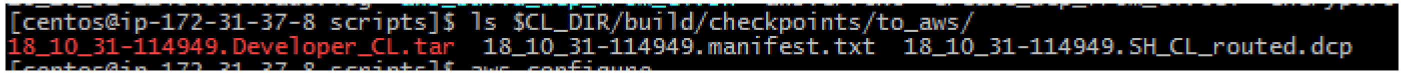
```
cd $HDK_DIR/cl/examples/cl_hello_world     # you can change cl_hello_world to any other examp
export CL_DIR=$(pwd)
```

- Verify Vivado is installed.

```
vivado -mode batch
```

- Run Vivado synthesis

```
cd $CL_DIR/build/scripts
./aws_build_dcp_from_cl.sh
```



```
[centos@ip-172-31-37-8 scripts]$ ls $CL_DIR/build/checkpoints/to_aws/
18_10_31-114949.Developer_CL.tar  18_10_31-114949.manifest.txt  18_10_31-114949.SH_CL_routed.dcp
```

Creating an Amazon FPGA Image (AFI)

Now that synthesis is done, we need to create an Amazon FPGA Image (AFI) from the specified design checkpoint (DCP). The AFI contains the FPGA bitstream that will be programmed on the FPGA F1 instance.

- To create an AFI, the DCP must be stored on S3. So we first need to create an s3 bucket. Make sure your credentials are set up correctly for this (aws configure).

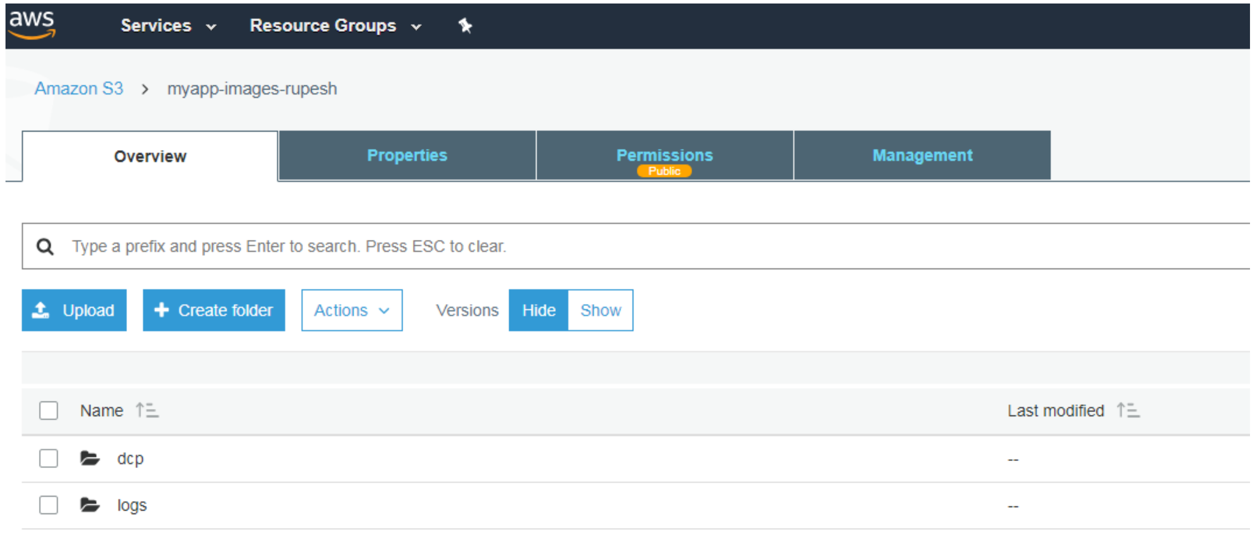
```
aws s3 mb s3://<bucket-name> --region <region-name> # Create an S3 bucket. Choose a unique b
aws s3 mb s3://<bucket-name>/<dcp-folder-name> # Create a folder for your tarball files (e.g
```

- Now copy the output files from synthesis to the new s3 bucket.

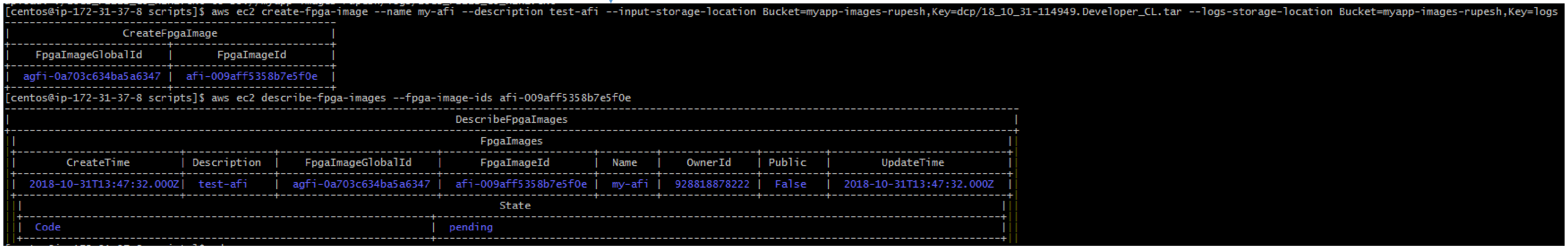
```
s3 cp $CL_DIR/build/checkpoints/to_aws/*.Developer_CL.tar s3://<bucket-name>/<dcp-folder-name>/
```

- Create a folder for your log files

```
aws s3 mb s3://<bucket-name>/<logs-folder-name>  # Create a folder to keep your logs
touch LOGS_FILES_GO_HERE.txt                     # Create a temp file
aws s3 cp LOGS_FILES_GO_HERE.txt s3://<bucket-name>/<logs-folder-name>/
```

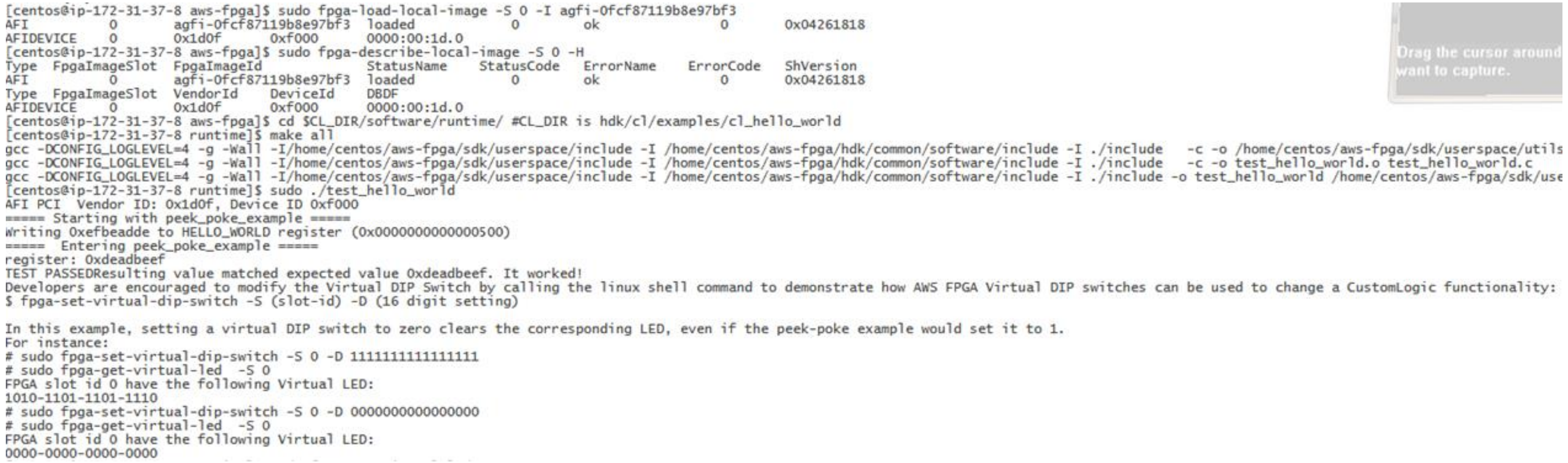


Name	Last modified
dcp	--
logs	--



```
[centos@ip-172-31-37-8 scripts]$ aws ec2 create-fpga-image --name my-afi --description test-afi --input-storage-location Bucket=myapp-images-rupesh,Key=dcp/18_10_31-114949.Developer_CL.tar --logs-storage-location Bucket=myapp-images-rupesh,Key=logs
-------------------------------------------------------
|                  CreateFpgaImage                    |
+-------------------------+---------------------------+
|   FpgaImageGlobalId     |       FpgaImageId         |
+-------------------------+---------------------------+
|  agfi-0a703c634ba5a6347 |  afi-009aff5358b7e5f0e    |
+-------------------------+---------------------------+
[centos@ip-172-31-37-8 scripts]$ aws ec2 describe-fpga-images --fpga-image-ids afi-009aff5358b7e5f0e
-------------------------------------------------------------------------------------------------------------------------------------------------------
|                                                           DescribeFpgaImages                                                                        |
+-----------------------------------------------------------------------------------------------------------------------------------------------------+
||                                                             FpgaImages                                                                            ||
|+--------------------------+--------------+------------------------+------------------------+----------+--------------+---------+--------------------------+|
||        CreateTime        | Description  |   FpgaImageGlobalId    |      FpgaImageId       |  Name    |   OwnerId    | Public  |        UpdateTime        ||
|+--------------------------+--------------+------------------------+------------------------+----------+--------------+---------+--------------------------+|
||  2018-10-31T13:47:32.000Z|  test-afi    |  agfi-0a703c634ba5a6347 |  afi-009aff5358b7e5f0e |  my-afi  |  928818878222 |  False  |  2018-10-31T13:47:32.000Z ||
|+--------------------------+--------------+------------------------+------------------------+----------+--------------+---------+--------------------------+|
|||                                                              State                                                                              |||
||+------------------------------------------------------------+--------------------------------------------------------------------------------------+||
|||  Code                                                       |  pending                                                                            |||
||+------------------------------------------------------------+--------------------------------------------------------------------------------------+||
```

Running the Example on an Amazon EC2 F1 Instance



```
[centos@ip-172-31-37-8 aws-fpga]$ sudo fpga-load-local-image -S 0 -I agfi-0fcf87119b8e97bf3
AFI          0       agfi-0fcf87119b8e97bf3  loaded            0        ok               0       0x04261818
AFIDEVICE    0       0x1d0f        0xf000      0000:00:1d.0
[centos@ip-172-31-37-8 aws-fpga]$ sudo fpga-describe-local-image -S 0 -H
Type  FpgaImageSlot  FpgaImageId             StatusName    StatusCode   ErrorName    ErrorCode   ShVersion
AFI          0       agfi-0fcf87119b8e97bf3  loaded            0        ok               0       0x04261818
Type  FpgaImageSlot  VendorId    DeviceId    DBDF
AFIDEVICE    0       0x1d0f      0xf000      0000:00:1d.0
[centos@ip-172-31-37-8 aws-fpga]$ cd $CL_DIR/software/runtime/ #CL_DIR is hdk/cl/examples/cl_hello_world
[centos@ip-172-31-37-8 runtime]$ make all
gcc -DCONFIG_LOGLEVEL=4 -g -Wall -I/home/centos/aws-fpga/sdk/userspace/include -I /home/centos/aws-fpga/hdk/common/software/include -I ./include   -c -o /home/centos/aws-fpga/sdk/userspace/utils
gcc -DCONFIG_LOGLEVEL=4 -g -Wall -I/home/centos/aws-fpga/sdk/userspace/include -I /home/centos/aws-fpga/hdk/common/software/include -I ./include   -c -o test_hello_world.o test_hello_world.c
gcc -DCONFIG_LOGLEVEL=4 -g -Wall -I/home/centos/aws-fpga/sdk/userspace/include -I /home/centos/aws-fpga/hdk/common/software/include -I ./include -o test_hello_world /home/centos/aws-fpga/sdk/use
[centos@ip-172-31-37-8 runtime]$ sudo ./test_hello_world
AFI PCI  Vendor ID: 0x1d0f, Device ID 0xf000
===== Starting with peek_poke_example =====
Writing 0xefbeadde to HELLO_WORLD register (0x0000000000000500)
=====  Entering peek_poke_example =====
register: 0xdeadbeef
TEST PASSEDResulting value matched expected value 0xdeadbeef. It worked!
Developers are encouraged to modify the Virtual DIP Switch by calling the linux shell command to demonstrate how AWS FPGA Virtual DIP switches can be used to change a CustomLogic functionality:
$ fpga-set-virtual-dip-switch -S (slot-id) -D (16 digit setting)

In this example, setting a virtual DIP switch to zero clears the corresponding LED, even if the peek-poke example would set it to 1.
For instance:
# sudo fpga-set-virtual-dip-switch -S 0 -D 1111111111111111
# sudo fpga-get-virtual-led  -S 0
FPGA slot id 0 have the following Virtual LED:
1010-1101-1101-1110
# sudo fpga-set-virtual-dip-switch -S 0 -D 0000000000000000
# sudo fpga-get-virtual-led  -S 0
FPGA slot id 0 have the following Virtual LED:
0000-0000-0000-0000
```

Useful URLs:

https://www.youtube.com/watch?v=y5hatX7j-E4

https://kivantium.net/fpga-aws

https://www.legupcomputing.com/blog/index.php/2017/08/10/step-by-step-guide-on-running-two-examples-on-the-amazon-fpga-cloud-amazon-ec2-f1/

https://www.youtube.com/watch?v=R_Wxc8y7lb0

Useful Command:

```
aws ec2 create-fpga-image --name my-afi --description test-afi --input-storage-location Bucket=myapp-images-rupesh,Key=dcp/18_10_31-114949.Developer_CL.tar --logs-storage-location Bucket=myapp-images-rupesh,Key=logs
```

```
 FpgaImageGlobalId    |      FpgaImageId      |
+------------------------+------------------------+
|  agfi-0a703c634ba5a6347 |  afi-009aff5358b7e5f0e
```