FPGA Accelerated Computing Using AWS F1 Instances

Applications and development environment

David Pellerin, Amazon Web Services
HotChips 2017
August 22, 2017
Why Accelerated Computing in the Cloud?

Parallelism increases throughout...

CPU: High speed, low efficiency
GPU/FPGA: High throughput, high efficiency

GPUs and FPGAs can provide massive parallelism and higher efficiency than CPUs for certain categories of applications
Compelling Use-Cases for Acceleration

- Deep Learning Training and Inference
- Video and Image Processing
- Engineering Simulations
- Financial Computing
- Molecular Dynamics
- VR Content Rendering
- Accelerated Search and Databases
- Many More
GPU and FPGA for Accelerated Computing

P2: GPU-accelerated computing
- Enabling a high degree of parallelism – each GPU has thousands of cores
- Consistent, well documented set of APIs (CUDA, OpenACC, OpenCL)
- Supported by a wide variety of ISVs and open source frameworks

F1: FPGA-accelerated computing
- Massively parallel – each FPGA includes millions of parallel system logic cells
- Flexible – no fixed instruction set, can implement wide or narrow datapaths
- Programmable using available, cloud-based FPGA development tools
AWS Compute Instance Types

General purpose:
- M4
- M3

Compute optimized:
- C5
- C4
- C3
- CC2

Storage and IO optimized:
- I3
- I2
- HS

Memory optimized:
- X1
- R4
- R3

GPU and FPGA accelerated:
- F1
- P2
- G2
- CG1

Years:
- 2011
- 2013
- 2016
- 2017
FPGA Acceleration in the AWS Cloud: Goals

- Make FPGAs available as standard AWS instances to a large community of developers, and to millions of potential end-customers
- **Simplify the development process** by providing cloud-based FPGA development tools
- **Allow developers to focus on algorithm design**, by abstracting FPGA I/O using well-defined interfaces
- **Provide a Marketplace for FPGA applications**, providing more choice and easy access for all AWS customers
How FPGA Acceleration Works on AWS

FPGA handles compute-intensive, deeply pipelined, hardware-accelerated operations.

Dedicated PCIe and ring connections also allow communication between up to 8 FPGAs, at up to 400Gbps.

```
module filter1 (clock, rst, strm_in, strm_out)
for (i=0; i<NUMUNITS; i=i+1)
    always@(posedge clock)
        integer i,j; //index for loops
        tmp_kernel[j] = k[i*OFFSETX];
```

CPU handles the rest.
Data is transferred to and from the FPGA via PCIe.
**Guiding principle:** allow FPGAs to be included in a customer’s deployment as easily as any other AWS instance type or service

**Important concepts:**
- Region
- Availability Zone (AZ)
- Virtual Private Cloud (VPC)
- Elastic Compute Cloud (EC2)
- Amazon Machine Image (AMI)
- EC2 Instance
- AWS Marketplace

**Additional for F1:**
- FPGA Developer AMI
- Amazon FPGA Image (AFI)
F1 Instances

- Up to eight Xilinx UltraScale Plus VU9P FPGAs per F1 instance
- Each FPGA includes
  - Local 64 GiB DDR4 ECC protected memory
  - Dedicated PCIe x16 connections, and an up to 400Gbps bidirectional ring connection for high-speed streaming
  - Approximately 2.5 million logic elements, and approximately 6,800 Digital Signal Processing (DSP) engines
FPGA Acceleration Using F1

An F1 instance can have any number of AFIs.

An AFI can be loaded into the FPGA in less than 1 second.
Developing Applications for F1

The F1 Development AMI

Use Xilinx Vivado and a hardware description language (Verilog or VHDL for RTL, or optionally using the OpenCL framework) with the HDK to describe and simulate your custom FPGA logic

Xilinx Vivado for custom logic development

Virtual JTAG for interactive debugging
FPGA Developer AMI

Sold by: Amazon Web Services

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 run time tools required to develop and use custom FPGAs for hardware acceleration. The FPGA developer AMI includes a prepackaged tool development environment, with scripts and tools for simulating your FPGA design, compiling code, building and registering your API (Amazon FPGA Image). Developers can deploy the FPGA developer AMI on an Amazon EC2 Instance and quickly provision the resources they need to write...

Read more

Customer Rating

Latest Version 1.2.1

Operating System Linux/Unix, CentOS 7.3

Delivery Method 64-bit Amazon Machine Image (AMI) (Read more)

Support See details below

AWS Services Required Amazon EC2, Amazon EBS

Highlights
- Xilinx Vivado 2017.1 and 2016.4 SDx - Free license for F1 FPGA development
- AWS Integration - includes packages and configurations that provide tight integration with Amazon Web Services

Product Description

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 run time tools required to develop and use custom FPGAs for hardware acceleration. The FPGA developer AMI includes a prepackaged tool development environment, with scripts and tools for simulating your FPGA design, compiling code, building and registering your API (Amazon FPGA Image). Developers can deploy the FPGA developer AMI on an Amazon EC2 Instance and quickly provision the resources they need to write and debug FPGA designs in the cloud. The AMI is designed to provide a stable, secure, and high performance development environment. The FPGA AMI is provided at no additional charge to Amazon EC2 users.

Software Pricing

The data below shows pricing per instance for services hosted in US East (N. Virginia).

<table>
<thead>
<tr>
<th>EC2 Instance Type</th>
<th>Software /hr</th>
<th>EC2 /hr</th>
<th>Total /hr</th>
</tr>
</thead>
<tbody>
<tr>
<td>c4.large</td>
<td>$0.00</td>
<td>$0.796</td>
<td>$0.796</td>
</tr>
<tr>
<td>c4.xlarge</td>
<td>$0.00</td>
<td>$1.591</td>
<td>$1.591</td>
</tr>
<tr>
<td>m4.large</td>
<td>$0.00</td>
<td>$0.431</td>
<td>$0.431</td>
</tr>
<tr>
<td>m4.xlarge</td>
<td>$0.00</td>
<td>$0.862</td>
<td>$0.862</td>
</tr>
<tr>
<td>m4.10xlarge</td>
<td>$0.00</td>
<td>$2.155</td>
<td>$2.155</td>
</tr>
<tr>
<td>m4.16xlarge</td>
<td>$0.00</td>
<td>$3.447</td>
<td>$3.447</td>
</tr>
<tr>
<td>t2.large</td>
<td>$0.00</td>
<td>$0.376</td>
<td>$0.376</td>
</tr>
<tr>
<td>t2.xlarge</td>
<td>$0.00</td>
<td>$1.65</td>
<td>$1.65</td>
</tr>
<tr>
<td>f1.2xlarge</td>
<td>$0.00</td>
<td>$15.20</td>
<td>$15.20</td>
</tr>
<tr>
<td>r4.large</td>
<td>$0.00</td>
<td>$0.266</td>
<td>$0.266</td>
</tr>
<tr>
<td>r4.xlarge</td>
<td>$0.00</td>
<td>$0.532</td>
<td>$0.532</td>
</tr>
<tr>
<td>r4.8xlarge</td>
<td>$0.00</td>
<td>$1.064</td>
<td>$1.064</td>
</tr>
<tr>
<td>r4.16xlarge</td>
<td>$0.00</td>
<td>$2.128</td>
<td>$2.128</td>
</tr>
<tr>
<td>r4.4xlarge</td>
<td>$0.00</td>
<td>$4.256</td>
<td>$4.256</td>
</tr>
</tbody>
</table>
AWS FPGA Shell
FPGA I/O is provided using standard, pre-tested, and secure I/O components, allowing FPGA developers to focus on their differentiating value.

The FPGA Shell allows for faster coding of core acceleration functions by removing the need to develop I/O related FPGA hardware.
Some of the DRAM interface controllers are implemented in the CL rather than the Shell for optimized resource utilization of the FPGA (Allowing higher utilization for the CL place and route region to maximize usable FPGA resources). For those interfaces, the designs and the constraints are provided by AWS and must be instantiated in the CL (by instantiating sh_ddr.sv in the CL design).
Hardware Simulation on AWS

Run RTL simulation using the simulator of your choice, either using the AWS-provided FPGA Developer AMI, or using your choice of simulation tools.
Using Build Strategies to Accelerate Development

**Strategy descriptions:**

**BASIC**
The basic flow in Vivado, designed to provide a good balance between runtime and Quality of Results (QOR)

**EXPLORE**
This is a high-effort flow which is designed to give improved QOR results at the expense of runtime

**TIMING**
This flow is designed for more aggressive timing optimization at the expense of runtime and congestion

**CONGESTION**
This flow is designed to insert more aggressive whitespace to alleviate routing congestion

**DEFAULT**
This is an additional high-effort flow that results in improved QOR results for the example design at the expense of runtime
Create the Amazon FPGA Image (AFI)

Generate an encrypted AFI using the generated DCP

```
$ aws ec2 create-fpga-image \
  --name <afi-name> \
  --description <afi-description> \
  --input-storage-location Bucket=<dcp-bucket-name>,Key=<path-to-tarball> \
  --logs-storage-location Bucket=<logs-bucket-name>,Key=<path-to-logs> \
  [ --client-token <value> ] \n  [ --dry-run | --no-dry-run ]
```
AWS FPGA SDK

• SDK includes the software runtime environment required to deploy on F1 instances and perform FPGA debugging

• Includes the drivers and tools to manage deployment of the AFIs to the F1 FPGAs, and to manage I/O from the software side

• APIs can be used to load different AFIs onto the F1 instance, without requiring an instance reboot
SDK

Management options:
[A] Shell FPGA Management Tools
[B] C-library FPGA Management
[C] OpenCL runtime library

Runtime code for I/O:
[D] FPGA PCIe Lib
[E] DMA Interface
[F] Interrupt/Event notification
[I] OpenCL Installable Client Driver

Linux Kernel Driver:
[G] DMA Kernel Driver

Linux Userspace:
[C] awshllib

AWS FPGA Mgmt Tools (Linux Shell)

OpenCL App

AWS OpenCL ICD

C/C++ App

Read()
Write()
Fsnc() User Interrupts Poll()

FPGA Mgmt Lib

FPGA PCIe Lib

AWS Shell

MgmtPF

DMA

AppPF

CL
AWS FPGA SDK - APIs

Management APIs

fpga-load-local-image,
fpga-clear-local-image,
fpga-describe-local,
fpga-start-virtual-jtag,
fpga-get-virtual-led,
fpga-set-virtual-dip-switch

Runtime driver library APIs

```c
write_buffer = (char *)malloc(buffer_size);
read_buffer = (char *)malloc(buffer_size);
if (write_buffer == NULL || read_buffer == NULL) {
    rc = ENOMEM;
goto out;
}
rand_string(write_buffer, buffer_size);
for (channel=0; channel < 4; channel++) {
    rc = pwrite(fd, write_buffer, buffer_size, 0x10000000 + channel*MEM_16G);
    fail_on((rc == (rc < 0)? 1:0), out, "call to pwrite failed.");
}
```
F1 Now Supports OpenCL

Application Code has two parts:

**Host code**
- Initializes platform
- Moves data to/from device global memory
- Launches kernels on device
- C/C++
- OpenCL APIs

**Kernel code**
- Computation to be accelerated
- Synthesized to the FPGA
- OpenCL (.cl), C/C++
- HDL (verilog/vhdl) using RTL wizard
F1 OpenCL Design Flow

1. Software Emulation
   - Verify the functional behavior of the host code and kernel operation.

2. Hardware Emulation
   - Generate exact performance values and confirm kernel compiled in hardware.

3. Hardware
   - Implement the hardware in the FPGA and confirm real-time operation.

---

**Steps:**

- **A:** --platform
- **B:** --output
- **C:** --compile
- **D:** --link
- **E:** gcc

**Files:**

- .cl (kernel)
- .c (function)
- .RTL (module)
- main.c
- emconfigutil
- emconfig.json
- .xcbin
- .xo

**Tools:**

- RTL Wizard
- Vivado HLS
Developing Applications for F1 – AFI Creation

FPGA Development using Xilinx Vivado on C4 or M4 instance

Generate an Amazon FPGA Image (AFI)

Securely deploy AFI on one or more F1 instances

```bash
$ aws ec2 create-fpga-image \
  --name <afi-name> \
  --description <afi-description> \
  --input-storage-location Bucket=<dcp-bucket-name>,Key=<path-to-tarball> \
  --logs-storage-location Bucket=<logs-bucket-name>,Key=<path-to-logs> \
  [ --client-token <value> ] \
  [ --dry-run | --no-dry-run ]
```
Virtual JTAG for Runtime Debugging

$ sudo fpga-start-virtual-jtag -P 10201 -S 0
Starting Virtual JTAG XVC Server for FPGA slot id 0, listening to TCP port 10201. Press CTRL-C to stop the service.
Delivering FPGA Partner Solutions via AWS Marketplace

Amazon Machine Image (AMI)

AWS Marketplace

Amazon FPGA Image (AFI)

AFI is secured, encrypted, dynamically loaded into the FPGA - can’t be copied or downloaded

Amazon EC2 FPGA Deployment via Marketplace
The Amazon FPGA development environment provides developers with an end-to-end solution for using a cloud-based FPGA Developer AMI and Hardware Developer Kit that includes all components needed by a developer to describe, simulate, debug, and compile hardware acceleration code to create an Amazon FPGA Image (AFI), deploy it to an F1 instance, and, if desired, offer the resulting FPGA application on the AWS Marketplace for distribution and monetization.
Thank you!

David Pellerin dpelleri@amazon.com