Exploring Portability and Performance of OpenCL FPGA Kernels on HARPv2

Anthony M. Cabrera, Roger D. Chamberlain

Washington University in St. Louis

{acabrera, roger}@wustl.edu

IWOCL '19

May 14, 2019
Motivation

Moore’s Law is "Dying"

Source: Courtland, IEEE Spectrum 2016
Motivation
Heterogeneous Systems

Source: Kharya, Forbes 2018

Source: Forrest, TechRepublic 2017
Motivation

How about FPGAs?

Intel’s $16.7 Billion Altera Deal Is Fueled by Data Centers

Source: King, Bloomberg 2015

Project Catapult

Source: Microsoft
Motivation
OpenCL to the Rescue!
Exploring HARPv2
Cabrera Chamberlain

Motivation
Intel’s Hardware Accelerator Research Program (HARP)

We address the following questions:

- How performant and portable are OpenCL FPGA kernels on the HARPv2 platform?

- What are the hardware knobs we can turn to get the best performance?

- What is the impact of the FPGA sharing the same memory as the CPU on the HARP system?
What’s an FPGA, anyway?

What’s an FPGA, anyway?

What’s an FPGA, anyway?

Programmable Routing Fabric

Logic Element

What’s an FPGA, anyway?

What’s an FPGA, anyway?

Exploring HARPv2

Cabrera Chamberlain

Introduction
Preliminaries
FPGAs
HARPv2
Path to Portability and Performance
Basic Kernel
Design Choices
Wavefront Parallelism
Hardware Design Space
SVM
Results
HW Design Space Search Comparison SVM Performance
Conclusion

Intel HARPv2

Socket

CPU
N Cores
LL$

FPGA
Interface
User Kernel

DRAM
Intel HARPv2 (top) vs. Discrete FPGA Card (bottom)
Introduction

Preliminaries
- FPGAs
- HARPv2

Path to Portability and Performance
- Basic Kernel
- Design Choices
- Wavefront Parallelism
- Hardware Design Space
- SVM

Results
- HW Design Space Search
- Comparison
- SVM Performance

Conclusion
Application Flavor
Dynamic Programming

```
__kernel void nw(__global int* ref_mat,
                 __global int* out_mat,
                 int num_rows,
                 int num_cols,
                 int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max( out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty );
        }
    }
}
```
Application Flavor

\( i = 1, j = 1 \)

```c
__kernel void nw(__global int* ref_mat,
                 __global int* out_mat,
                 int num_rows,
                 int num_cols,
                 int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max( out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty );
        }
    }
}
```
Application Flavor

\( i = 1, j = 2 \)

```c
__kernel void nw(__global int* ref_mat,
                __global int* out_mat,
                int num_rows,
                int num_cols,
                int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max( out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty);
        }
    }
}
```
Application Flavor

\( i = 1, j = 3 \)

```c
__kernel void nw(__global int* ref_mat,
                 __global int* out_mat,
                 int num_rows,
                 int num_cols,
                 int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max( out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty );
        }
    }
}
```
Application Flavor

$i = 2, j = 1$

```c
__kernel void nw(__global int* ref_mat,
                 __global int* out_mat,
                 int num_rows,
                 int num_cols,
                 int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max(out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty);
        }
    }
}
```
Exploring HARPv2
Cabrera Chamberlain

Introduction
Preliminaries
FPGAs
HARPv2
Path to Portability and Performance
Basic Kernel
Design Choices
Wavefront Parallelism
Hardware Design Space
SVM
Results
HW Design Space Search Comparison
SVM Performance
Conclusion

Application Flavor
$i = 2, j = 2$

```
out_mat

__kernel void nw(__global int* ref_mat,
                  __global int* out_mat,
                  int num_rows,
                  int num_cols,
                  int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
            max( out_mat[i-1][j] - penalty,
                 out_mat[i-1][j-1] + ref_mat[i][j],
                 out_mat[i][j-1] - penalty );
        }
    }
}
```
Application Flavor

Application Flavor

i = 2, j = 3

```
out_mat
```

```c
__kernel void nw(__global int* ref_mat,
                __global int* out_mat,
                int num_rows,
                int num_cols,
                int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max( out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty );
        }
    }
}
```
Application Flavor

\( i = 3, \ j = 1 \)

```c
__kernel void nw(__global int* ref_mat,
                 __global int* out_mat,
                 int num_rows,
                 int num_cols,
                 int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
            max( out_mat[i-1][j] - penalty,
                 out_mat[i-1][j-1] + ref_mat[i][j],
                 out_mat[i][j-1] - penalty );
        }
    }
}
```
Application Flavor

\( i = 3, j = 2 \)

```cpp
__kernel void nw(__global int* ref_mat,
                 __global int* out_mat,
                 int num_rows,
                 int num_cols,
                 int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max( out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty );
        }
    }
}
```
Application Flavor

\( i = 3, j = 3 \)

```c
__kernel void nw(__global int* ref_mat,
     __global int* out_mat,
     int num_rows,
     int num_cols,
     int penalty)
{
    for (int i = 1; i < num_rows; ++i)
    {
        for (int j = 1; j < num_cols; ++j)
        {
            out_mat[i][j] =
                max( out_mat[i-1][j] - penalty,
                     out_mat[i-1][j-1] + ref_mat[i][j],
                     out_mat[i][j-1] - penalty );
        }
    }
}
```
Design Choices
for authoring OpenCL FPGA kernels

- **Width vs Depth**
Design Choices
for authoring OpenCL FPGA kernels

- Compiler Directives

- `reqd_work_group_size(X, Y, Z)`
- `num_simd_work_items(NUM)`
- `#pragma ivdep` (ignore vector dependences)
- `#pragma unroll`
Design Choices
for authoring OpenCL FPGA kernels

- Expressing performant FPGA constructs in High Level Language

5 Clock Cycle Delay by Shift Register

Data Out → Data Out → Data Out → Data Out → Data Out → Data Out
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism

Multiple Work Item \{\text{Inter, Intra}\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

<table>
<thead>
<tr>
<th>Unprocessed</th>
<th>Currently Processing</th>
<th>Processed</th>
</tr>
</thead>
</table>

```plaintext
Exploring HARPv2
Cabrera Chamberlain
Introduction
Preliminaries
FPGAs
HARPv2
Path to Portability and Performance
Basic Kernel Design Choices
Wavefront Parallelism
Hardware Design Space SVM
Results
HW Design Space Search Comparison SVM Performance
Conclusion
```
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item {Inter, Intra}-Work Group
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group

- Unprocessed
- Currently Processing
- Processed
Wavefront Parallelism
Multiple Work Item \{Inter, Intra\}-Work Group
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked

PAR=4

BSIZE=8

Shift Registers
PAR=4

DRAM
Wavefront Parallelism
Single Work Item Blocked, Chunked

PAR=4

BSIZE=8

Shift Registers
PAR=4

DRAM
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked

PAR=4
BSIZE=8

Shift Registers PAR=4
DRAM
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked

PAR=4
BSIZE=8

Shift Registers
PAR=4

DRAM
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked

PAR=4

BSIZE=8

Shift Registers
PAR=4

DRAM
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked

BSIZE=8
PAR=4

Shift Registers
PAR=4

DRAM
Exploring HARPv2

Cabrera Chamberlain

Introduction
Preliminaries
FPGAs
HARPv2
Path to Portability and Performance
Basic Kernel Design Choices
Wavefront Parallelism
Hardware Design Space
SVM

Results
HW Design Space Search Comparison SVM Performance

Conclusion
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked
Wavefront Parallelism
Single Work Item Blocked, Chunked

<table>
<thead>
<tr>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
<th>17</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
</tr>
<tr>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>12</td>
<td>13</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
</tr>
</tbody>
</table>

BSIZE = 8
PAR = 4
Shift Registers
PAR = 4
DRAM

Exploring HARPv2
Cabrera Chamberlain
Introduction
Preliminaries
FPGAs
HARPv2
Path to Portability and Performance
Basic Kernel Design Choices
Wavefront Parallelism
Hardware Design Space
SVM
Results
HW Design Space Search
Comparison
SVM Performance
Conclusion
Wavefront Parallelism
Single Work Item Blocked, Chunked
Hardware Design Space

Example: $\text{BSIZE} = \{4, 8\}$, $\text{PAR} = \{2, 4\}$
Hardware Design Space
Example: BSIZE = \{ 4, 8 \}, PAR = \{ 2, 4 \}
Shared Virtual Memory (SVM)
Exploring HARPv2

Introduction

Preliminaries
- FPGAs
- HARPv2

Path to Portability and Performance
- Basic Kernel
- Design Choices
- Wavefront Parallelism
- Hardware Design Space
- SVM

Results
- HW Design Space Search
- Comparison
- SVM Performance

Conclusion
Design Space Search
Design Space Search

![Diagram showing execution time vs PAR value for different BSIZE values.](image-url)
It took 14 days to build all kernel configurations!
### Comparison Results

- SVP = Stratix V, PCIe
- HARP = Arria 10, HARP
- \( vD = \) Dummy

<table>
<thead>
<tr>
<th>( V )</th>
<th>Kernel Type</th>
<th>FPGA</th>
<th>( f_{max} ) (MHz)</th>
<th>Logic</th>
<th>Speedup</th>
</tr>
</thead>
</table>

- Zohouri et al., 2018
- Our Work
### Comparison Results

- **SVP** = Stratix V, PCIe
- **HARP** = Arria 10, HARP
- **vD** = Dummy

<table>
<thead>
<tr>
<th>V</th>
<th>Kernel Type</th>
<th>FPGA</th>
<th>$f_{max}$ (MHz)</th>
<th>Logic</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>v0</td>
<td>MWI</td>
<td>SVP</td>
<td>267.52</td>
<td>27%</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>211.77</td>
<td>25%</td>
<td>0.74</td>
</tr>
<tr>
<td>v1</td>
<td>SWI</td>
<td>SVP</td>
<td>304.50</td>
<td>20%</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>256.6</td>
<td>26%</td>
<td>0.01</td>
</tr>
<tr>
<td>v2</td>
<td>MWI</td>
<td>SVP</td>
<td>164.20</td>
<td>38%</td>
<td>2.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>162.865</td>
<td>50%</td>
<td>3.90</td>
</tr>
<tr>
<td>v3</td>
<td>SWI</td>
<td>SVP</td>
<td>191.97</td>
<td>19%</td>
<td>3.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>178.12</td>
<td>25%</td>
<td>3.24</td>
</tr>
<tr>
<td>v5</td>
<td>SWI</td>
<td>SVP</td>
<td>218.15</td>
<td>53%</td>
<td>38.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>186.81</td>
<td>40%</td>
<td>34.27</td>
</tr>
<tr>
<td>vD</td>
<td>N/A</td>
<td>HARP</td>
<td>350.26</td>
<td>23%</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Zohouri et al., 2018

Our Work
## Comparison Results

- **SVP** = Stratix V, PCIe
- **HARP** = Arria 10, HARP
- **vD** = Dummy

<table>
<thead>
<tr>
<th>V</th>
<th>Kernel Type</th>
<th>FPGA</th>
<th>$f_{max}$ (MHz)</th>
<th>Logic</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>v0</td>
<td>MWI</td>
<td>SVP</td>
<td>267.52</td>
<td>27%</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>211.77</td>
<td>25%</td>
<td>0.74</td>
</tr>
<tr>
<td>v1</td>
<td>SWI</td>
<td>SVP</td>
<td>304.50</td>
<td>20%</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>256.6</td>
<td>26%</td>
<td>0.01</td>
</tr>
<tr>
<td>v2</td>
<td>MWI</td>
<td>SVP</td>
<td>164.20</td>
<td>38%</td>
<td>2.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>162.865</td>
<td>50%</td>
<td>3.90</td>
</tr>
<tr>
<td>v3</td>
<td>SWI</td>
<td>SVP</td>
<td>191.97</td>
<td>19%</td>
<td>3.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>178.12</td>
<td>25%</td>
<td>3.24</td>
</tr>
<tr>
<td>v5</td>
<td>SWI</td>
<td>SVP</td>
<td>218.15</td>
<td>53%</td>
<td>38.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>186.81</td>
<td>40%</td>
<td>34.27</td>
</tr>
<tr>
<td>vD</td>
<td>N/A</td>
<td>HARP</td>
<td>350.26</td>
<td>23%</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Zohouri et al., 2018

Our Work
## Comparison Results

- **SVP** = Stratix V, PCIe
- **HARP** = Arria 10, HARP
- **vD** = Dummy

<table>
<thead>
<tr>
<th>V</th>
<th>Kernel Type</th>
<th>FPGA</th>
<th>$f_{\text{max}}$ (MHz)</th>
<th>Logic</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>v0</td>
<td>MWI</td>
<td>SVP</td>
<td>267.52</td>
<td>27%</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>211.77</td>
<td>25%</td>
<td>0.74</td>
</tr>
<tr>
<td>v1</td>
<td>SWI</td>
<td>SVP</td>
<td>304.50</td>
<td>20%</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>256.6</td>
<td>26%</td>
<td>0.01</td>
</tr>
<tr>
<td>v2</td>
<td>MWI</td>
<td>SVP</td>
<td>164.20</td>
<td>38%</td>
<td>2.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>162.865</td>
<td>50%</td>
<td>3.90</td>
</tr>
<tr>
<td>v3</td>
<td>SWI</td>
<td>SVP</td>
<td>191.97</td>
<td>19%</td>
<td>3.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>178.12</td>
<td>25%</td>
<td>3.24</td>
</tr>
<tr>
<td>v5</td>
<td>SWI</td>
<td>SVP</td>
<td>218.15</td>
<td>53%</td>
<td>38.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>186.81</td>
<td>40%</td>
<td>34.27</td>
</tr>
<tr>
<td>vD</td>
<td>N/A</td>
<td>HARP</td>
<td>350.26</td>
<td>23%</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Zohouri et al., 2018

Our Work
## Comparison Results

<table>
<thead>
<tr>
<th>V</th>
<th>Kernel Type</th>
<th>FPGA</th>
<th>$f_{max}$ (MHz)</th>
<th>Logic</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>v0</td>
<td>MWI</td>
<td>SVP</td>
<td>267.52</td>
<td>27%</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>211.77</td>
<td>25%</td>
<td>0.74</td>
</tr>
<tr>
<td>v1</td>
<td>SWI</td>
<td>SVP</td>
<td>304.50</td>
<td>20%</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>256.6</td>
<td>26%</td>
<td>0.01</td>
</tr>
<tr>
<td>v2</td>
<td>MWI</td>
<td>SVP</td>
<td>164.20</td>
<td>38%</td>
<td>2.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>162.865</td>
<td>50%</td>
<td>3.90</td>
</tr>
<tr>
<td>v3</td>
<td>SWI</td>
<td>SVP</td>
<td>191.97</td>
<td>19%</td>
<td>3.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>178.12</td>
<td>25%</td>
<td>3.24</td>
</tr>
<tr>
<td>v5</td>
<td>SWI</td>
<td>SVP</td>
<td>218.15</td>
<td>53%</td>
<td>38.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>186.81</td>
<td>40%</td>
<td>34.27</td>
</tr>
<tr>
<td>vD</td>
<td>N/A</td>
<td>HARP</td>
<td>350.26</td>
<td>23%</td>
<td>N/A</td>
</tr>
</tbody>
</table>

- **SVP** = Stratix V, PCIe
- **HARP** = Arria 10, HARP
- **vD** = Dummy

Zohouri et al., 2018

Our Work
## Comparison Results

- **SVP** = Stratix V, PCIe
- **HARP** = Arria 10, HARP
- **vD** = Dummy

<table>
<thead>
<tr>
<th>V</th>
<th>Kernel Type</th>
<th>FPGA</th>
<th>$f_{max}$ (MHz)</th>
<th>Logic</th>
<th>Speedup</th>
</tr>
</thead>
<tbody>
<tr>
<td>v0</td>
<td>MWI</td>
<td>SVP</td>
<td>267.52</td>
<td>27%</td>
<td>1.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>211.77</td>
<td>25%</td>
<td>0.74</td>
</tr>
<tr>
<td>v1</td>
<td>SWI</td>
<td>SVP</td>
<td>304.50</td>
<td>20%</td>
<td>0.05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>256.6</td>
<td>26%</td>
<td>0.01</td>
</tr>
<tr>
<td>v2</td>
<td>MWI</td>
<td>SVP</td>
<td>164.20</td>
<td>38%</td>
<td>2.48</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>162.865</td>
<td>50%</td>
<td>3.90</td>
</tr>
<tr>
<td>v3</td>
<td>SWI</td>
<td>SVP</td>
<td>191.97</td>
<td>19%</td>
<td>3.55</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>178.12</td>
<td>25%</td>
<td>3.24</td>
</tr>
<tr>
<td>v5</td>
<td>SWI</td>
<td>SVP</td>
<td>218.15</td>
<td>53%</td>
<td>38.22</td>
</tr>
<tr>
<td></td>
<td></td>
<td>HARP</td>
<td>186.81</td>
<td>40%</td>
<td>34.27</td>
</tr>
<tr>
<td>vD</td>
<td>N/A</td>
<td>HARP</td>
<td><strong>350.26</strong></td>
<td>23%</td>
<td><strong>N/A</strong></td>
</tr>
</tbody>
</table>

Zohouri et al., 2018
SVM Results

![Diagram showing execution time, device write, and device read for Explicit R/W and SVM.]
Conclusion

- Design space search necessary to find most performant kernel
- OpenCL design practices for PCIe Card FPGAs hold for HARPv2
- Intel HARPv2 FPGA-CPU interface requires a lot of FPGA resources
- SVM implementation alleviates data movement problem

For snapshot of artifacts:  
https://openscholarship.wustl.edu/data/17/

For most recent updates:  
https://github.com/cabreraam/iwocl2019_artifacts