XPU – A Programmable FPGA Accelerator for Diverse Workloads

Jian Ouyang,[1] (ouyangjian@baidu.com)
Ephrem Wu,[2] Jing Wang, [1]Yupeng Li, [1] Hanlin Xie[1]

[1]Baidu, Inc. [2]Xilinx

XILINX ALL PROGRAMMABLE

Baidu 百度

Outlines

- Background - FPGA for emerging applications
  - Potentials of FPGA for AI and big data
  - Overviews of FPGA accelerators in real system
  - Challenges of using FPGA for diverse workloads
- XPU
  - Motivation : A programmable FPGA Accelerator for diverse workloads
  - Architecture
  - Program model
  - Implementation
  - Evaluation
- Conclusion

Background - Potentials of FPGA for AI and big data

Kernels

- AI
  - Convolution
  - Matrix multiplication
  - Activations
  - Pooling…
- Data analysis
  - Compression/decompression
  - Filter
  - sort
  - Join
  - Aggregation …

Architecture support

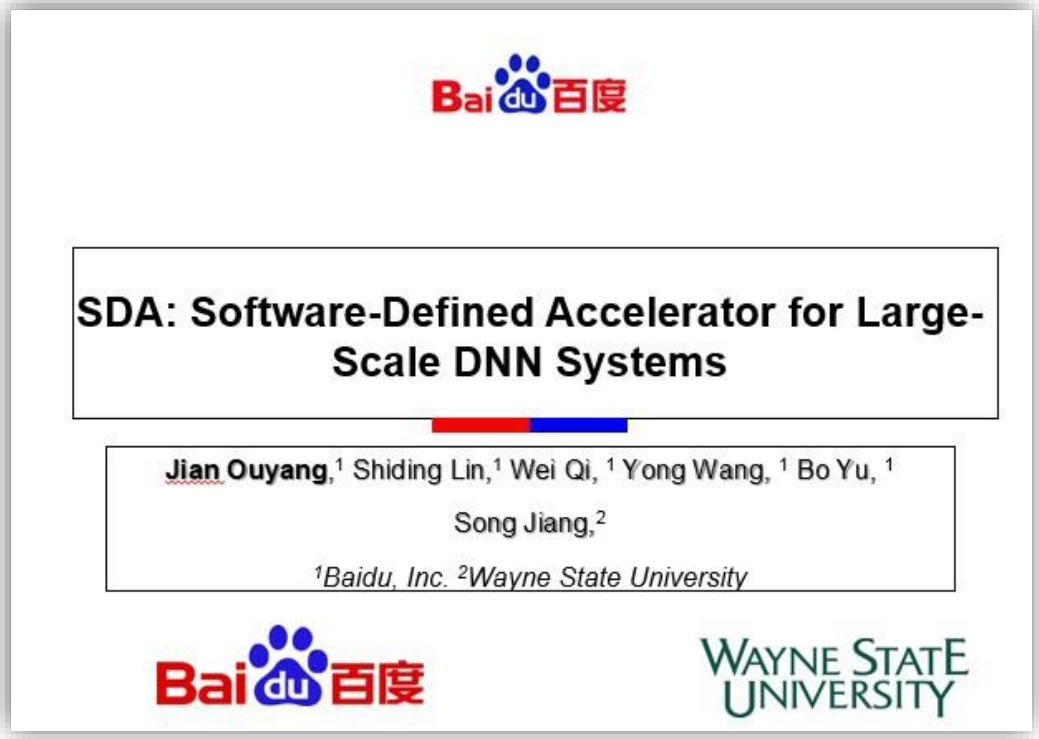
- Computing
  - Massive MAC arrays
  - Sophisticated logic operations
  - Math functions
- Memory access
  - high bandwidth off-chip memory
  - High bandwidth and low latency on-chip memory
  - Sophisticated access pattern
- IO
  - High bandwidth
  - Diversity

FPGA

- Computing
  - Thousands DSP
  - Millions LUT
  - Flex data path
- Memory access
  - off-chip : DDR4/HMB
  - on-chip : tens MB SRAM
  - Flex access pattern
- IO
  - PCIe
  - SERDES, GPIO…

FPGA has huge potentials for AI and big data

Background - Overviews of FPGA accelerators in real system



Baidu 百度

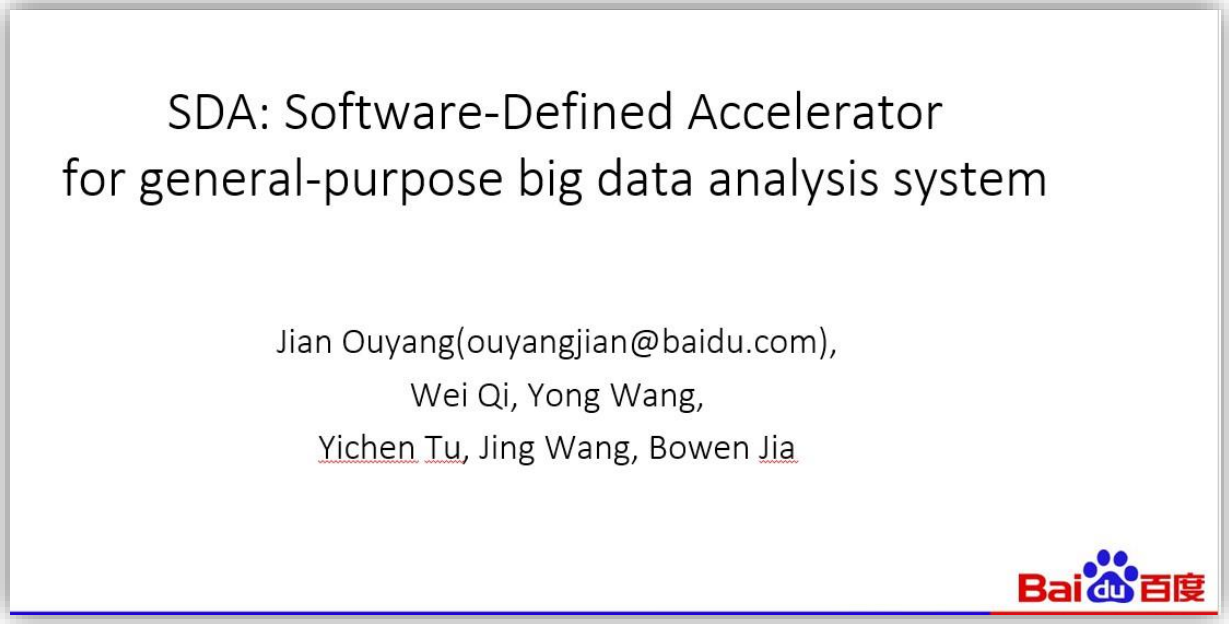
SDA: Software-Defined Accelerator for Large-Scale DNN Systems

Jian Ouyang,[1] Shiding Lin,[1] Wei Qi, [1] Yong Wang, [1] Bo Yu, [1]
Song Jiang,[2]
[1]*Baidu, Inc.* [2]*Wayne State University*

Baidu 百度

WAYNE STATE UNIVERSITY

Hot Chips 2014



SDA: Software-Defined Accelerator
for general-purpose big data analysis system

Jian Ouyang(ouyangjian@baidu.com),
Wei Qi, Yong Wang,
Yichen Tu, Jing Wang, Bowen Jia

Baidu 百度

Hot Chips 2016

Baidu 百度

Background - Overviews of FPGA accelerators in real system

Data center

Cloud

Autonomous driving

Background - Challenges of using FPGA for diverse workloads

- Most applications contain diverse workloads

- Diverse workloads
  - Computing intensive
  - Memory bound
  - Rule-based
- Variety of kernels
  - Several big kernels, such as convolution, matrix multiplications
  - Lots of small kernels, such as activations, element-wise operations, openCV kernels

Background - Challenges of using FPGA for diverse workloads

- FPGA has potential to support all kinds of workloads
  - Very low and predictable latency
  - Massive parallel computing
  - High memory bandwidth

- But, FPGA is not good at supporting diverse workloads
  - Hardware-reconfigurable
    - Dedicated logic for specific functionalities
  - Lacks flexible programmability
    - Specific circuit
    - Hardware reconfigurable

XPU – motivations

Traditional FPGA Accelerator

- Aim at specific workloads
- High efficiency
- Lacks programmability

Traditional CPU

- Aim at general workloads, especially rule- base workloads
- High flexibility

GPU

- Aim at parallelism workloads
- High performance

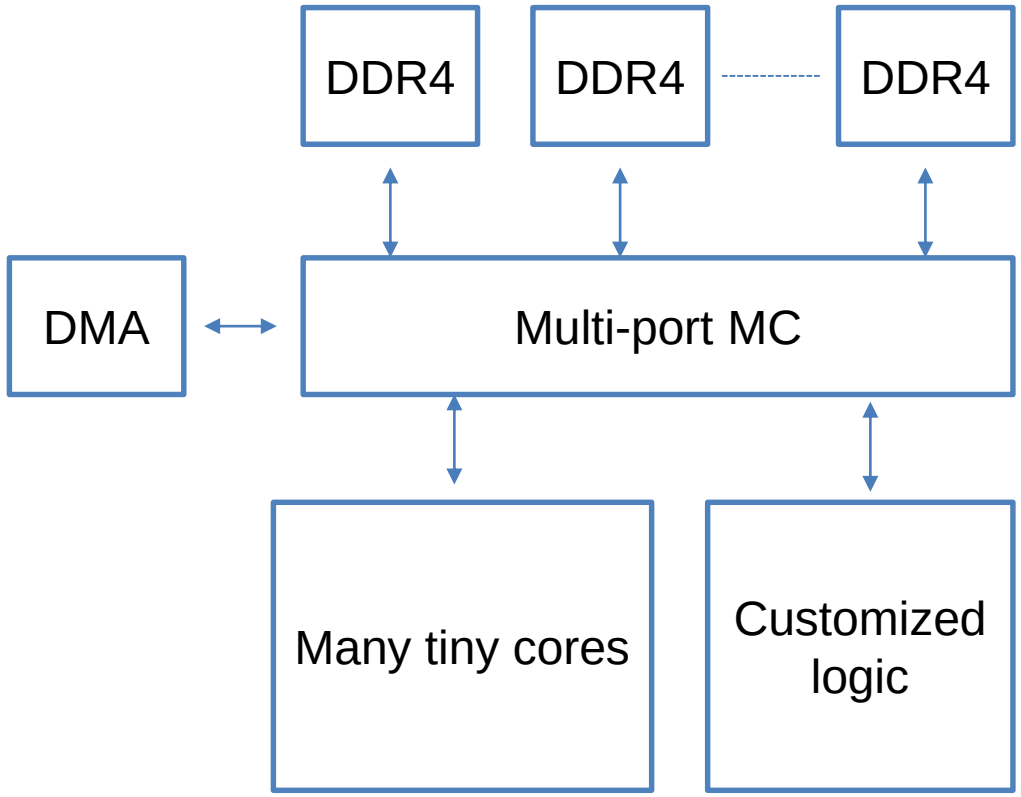
XPU

- Aimed at diverse workloads
  - computing intensive
  - rule-based
- High efficiency, flexible and performance

XPU – design choices

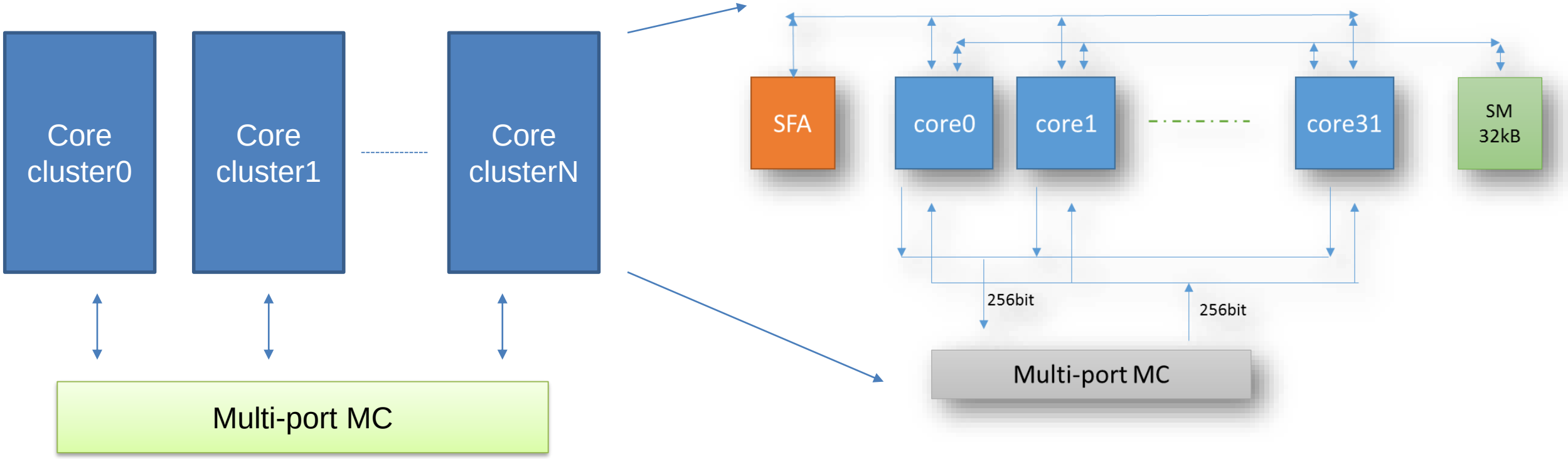
- High efficiency
  - Customized circuit for specified workloads
  - Example: SDA
- High flexibility
  - ISA based core
  - Customized for rule-based workloads
- High performance
  - Many cores for parallelism workloads

XPU – architecture



- Many tiny cores
  - Instruction set based Software-programmable
  - No OS, No Cache, domain specific ISA
  - Flexible to serve diverse workloads
- Customized logic
  - Hardware-reconfigurable
  - Achieve high performance efficiency
- Resource allocation is reconfigurable
  - Configure the ratio of cores vs. custom logic depending on applications requirements

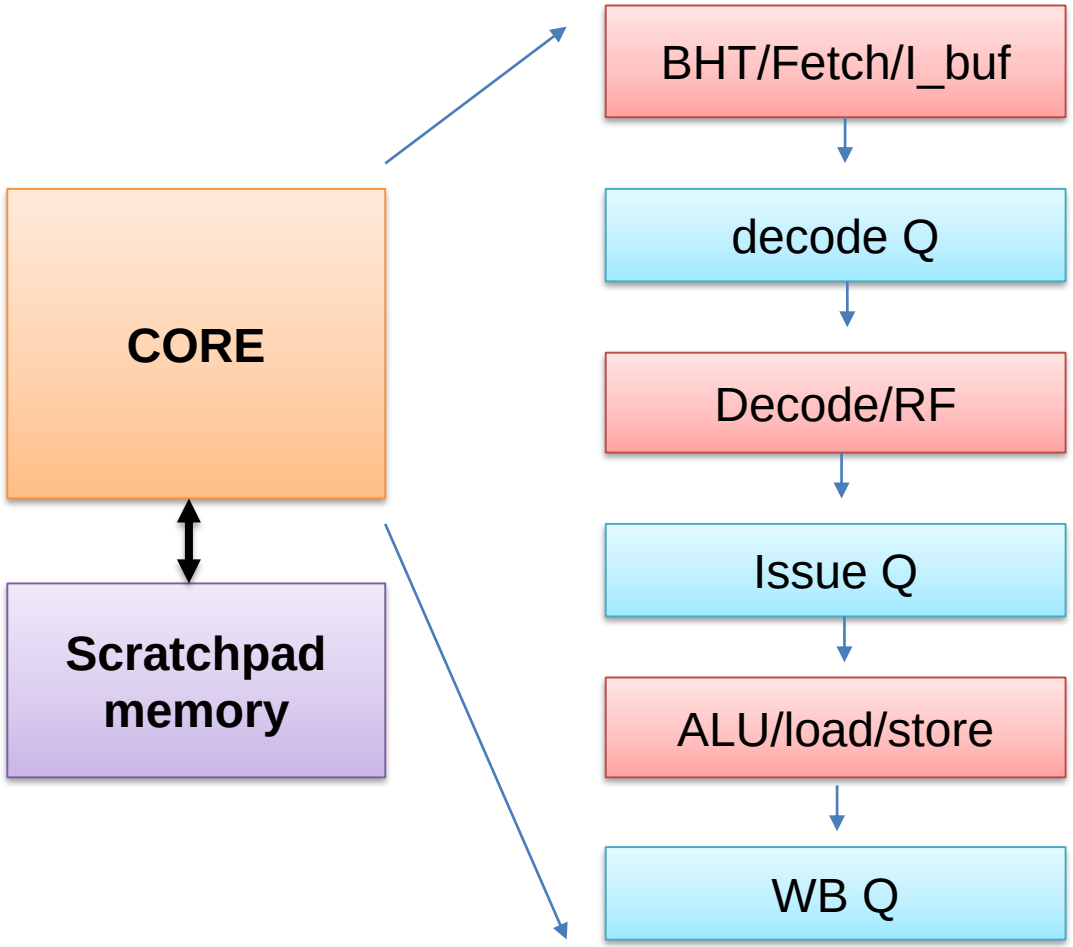
XPU – architecture of tiny cores



- Each 32 cores are clustered
  - Data locality and synchronization
  - Easy to route and place

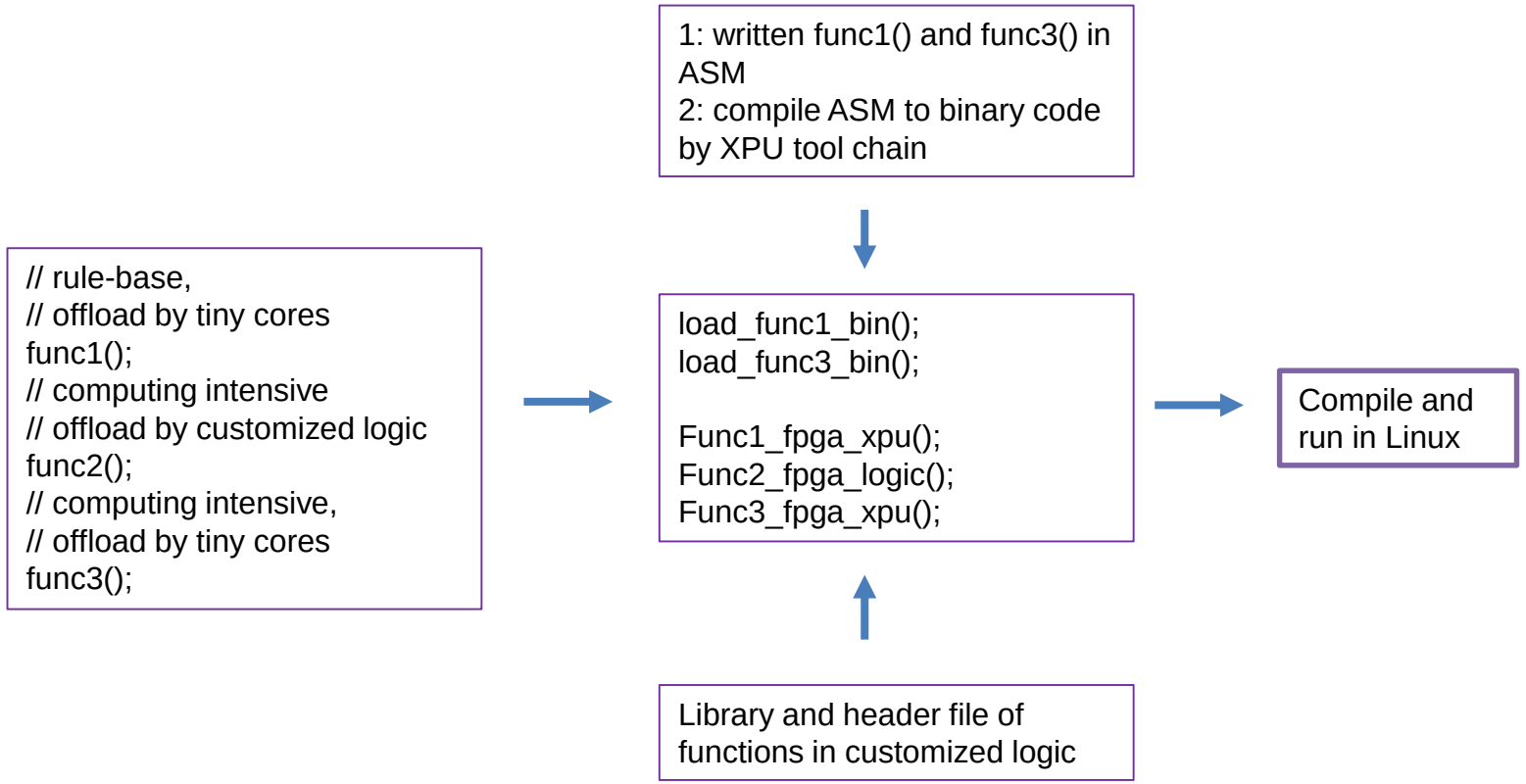
- 32 KB shared multi-bank memory
- Shared SFA (special function accelerator)

XPU –architecture of tiny cores



- MIPS-like instruction set
- Private scratchpad memory
  - 16 or 32KB
- Pipeline
  - Designed for low latency
  - 4 stage
  - BHT

XPU – program model



Step1:
- **partition the workloads**

Step2:
- **Write the XPU code**
- **Call the dedicated logic functions**

Step3:
- **compile**
- **run**

- Program model is similar to traditional PCIe accelerator
  - GPU or SDA
- Customized logic
  - Informative commands
- Tiny cores
  - Similar to traditional CPU
  - Controlled by host

XPU – implementation

- One tiny core
  - 1252 LUT
  - 1230 FF
  - 4 DSP
  - 5 BRAM
- Many tiny core
  - Resource scales linear as core number
  - 256 core consume 25% LUT and 15% DSP on VU9P
- Customized logic
  - SDA-II, 5120 DSP, 16bit fixed point, 600Mhz
  - 6.144Tops

XPU evaluation - setup

- Host
  - E5-2670, 2.60GHz, 128GB memory
  - Linux system

- XPU
  - VU9P
  - 256 tiny core, 600MHz
  - SDA-II for deep learning, 16bit fixed point, 600MHz, 6.144Tops
  - PCIE 3.0x16
  - 4x72bit DDR4, 2400MHz

XPU evaluation – setup

- Case 1: simple micro benchmark
- Case 2: computing intensive
- Case 3: regular memory access
- Case 4: random memory access
- Case 5: rule-based

XPU evaluation – case 1

- 1 to 100 accumulation
  - CPU code, gcc -O2 compiling, ~310 CPU cycle
  - XPU single core, ~300 cycle
  - XPU has same pipeline efficiency as X86 for simple program

```
sum=0;
for(i=0;i<100;i++)
    sum = sum+i ;
```

XPU evaluation – case 2 and case 3

- Case 2: Softmax
  - channel=2，height=640， width=640
  - Data format : [height][width][channel]
  - CPU single core: 20.4413ms
  - XPU single core : 12266100 cycles, @600Mhz, ~ 20.5ms
- Case 3: Slice
  - channel_in=4， channel_out_0=2， channel_out_1=2， height=640， width=640
  - Data format : [height][width][channel]
  - CPU single core : 3.77981ms
  - XPU single core : 1054100 cycle, @600MHz, ~1.756ms
- conclusion
  - XPU has similar efficiency as X86 core for computing intensive and regular memory access workload

XPU evaluation – case 4

- Kernels from computer vision
- Input: 100k pixels(each has x, y, z and pi)
- CPU single core : 18ms
- XPU single core : 433990000 cycles, ~720ms
- This workload is random memory access bound
  - need to improve the XPU memory control for this case

Pseudocode:

```
for( each pixel)
{
    int idx = y * W +x;
    data_A[idx] = z;
    data_B[idx] = atan2(y,x);
    data_C[idx] = pi;
    data_D[idx] = sqrt(x*x,y*y);
}
```

XPU evaluation – case 5

- A kernel from computer vision applications
  - Rule based
- Performance of one data set
  - CPU single core : 5 K cycles
  - XPU single core
    - 4.4K cycles
  - Similar efficiency for rule based workload

```
for ( i=0; i<h;i++ ) {
    for (j=0; j<w; j++  ) {
        float dis = *(vectors+i*w+j);
        if (dis > CONSTANT_A) {
            continue;
        } else if (dis <= CONSTANT_B) {
            res += CONSTANT_C;
        } else if (dis < CONSTANT_D) {
            float offset = CONSTANT_E - dis;
            res += CONSTANT_F +
                        CONSTANT_H * pow(offset, 5.0);
        } else {
            res += CONSTANT_I;
        }
    }
}
```

XPU evaluation – scalability discussion

- Case 2: Softmax
  - 1 core : 12266100 cycles
  - 8 core: 2104300 cycles
  - 5.82x
- Case 3: Slice
  - 1 core : 1054100 cycles
  - 4 core: 468100 cycles
  - 2.25x
- Case 5
  - Task level parallelism without synchronization among tasks
  - 256 core can achieve about 64x faster than CPU core,
  - and 8x than 8 core XEON,
  - 25x power efficiency than CPU (25w XPU vs. 80w CPU)
- Conclusion
  - The scalability of XPU for workloads with data synchronization should be improved further
  - The scalability of XPU for workloads without data synchronization is linear as core number

Conclusion

- Motivations
  - Traditional FPGA accelerator is only for specific workloads
  - Diverse workloads are common for data center, cloud and autonomous driving
- Key features of XPU
  - The XPU provides software programmable by instruction set based architecture and guarantees the high efficiency by custom logic.
- Status
  - Demonstrated in autonomous car and cloud applications
  - Proven that it can support diverse workloads without degrading efficiency