Transparent Offloading of Cloud FPGA Accelerators using OpenMP

Guido Araujo

Instituto de Computação – UNICAMP

Programming Clusters is Increasingly Complex

- Requires various programming languages and API
  - OpenMP / Multithreading for multicore processors
  - MPI for programming distributed architectures
  - CUDA for GPUs
  - Checkpoint library for fault tolerance
- High cost of development and maintenance

Our proposal

Programming heterogenous cluster with OpenMP only

- Programming using all OpenMP directives together (*parallel, task, target, etc.*)
- Need to deal with heterogeneity
- Extending the OpenMP runtime for distributed architectures

Dealing with GPUs

Extension for programming accelerators (v4.0+)

- New directives for local accelerators (e.g. GPU)
- Could it be extended to other devices?

```
int MatMul(float *A, float *B, float *C) {
  #pragma omp target device(GPU)
  #pragma omp map(to: A[:N*N], B[:N*N]) map(from: C[:N*N])
  #pragma omp parallel for
  for(int i=0; i < N; ++i)
    for(int j = 0; j < N; ++j)
      C[i * N + j] = 0;
      for(int k = 0; k < N; ++k)
        C[i * N + j] += A[i * N + k] * B[k * N + j];
  return 0;
}
```

Dealing with FPGAs (Intel HARP)

(FCCM 2018)

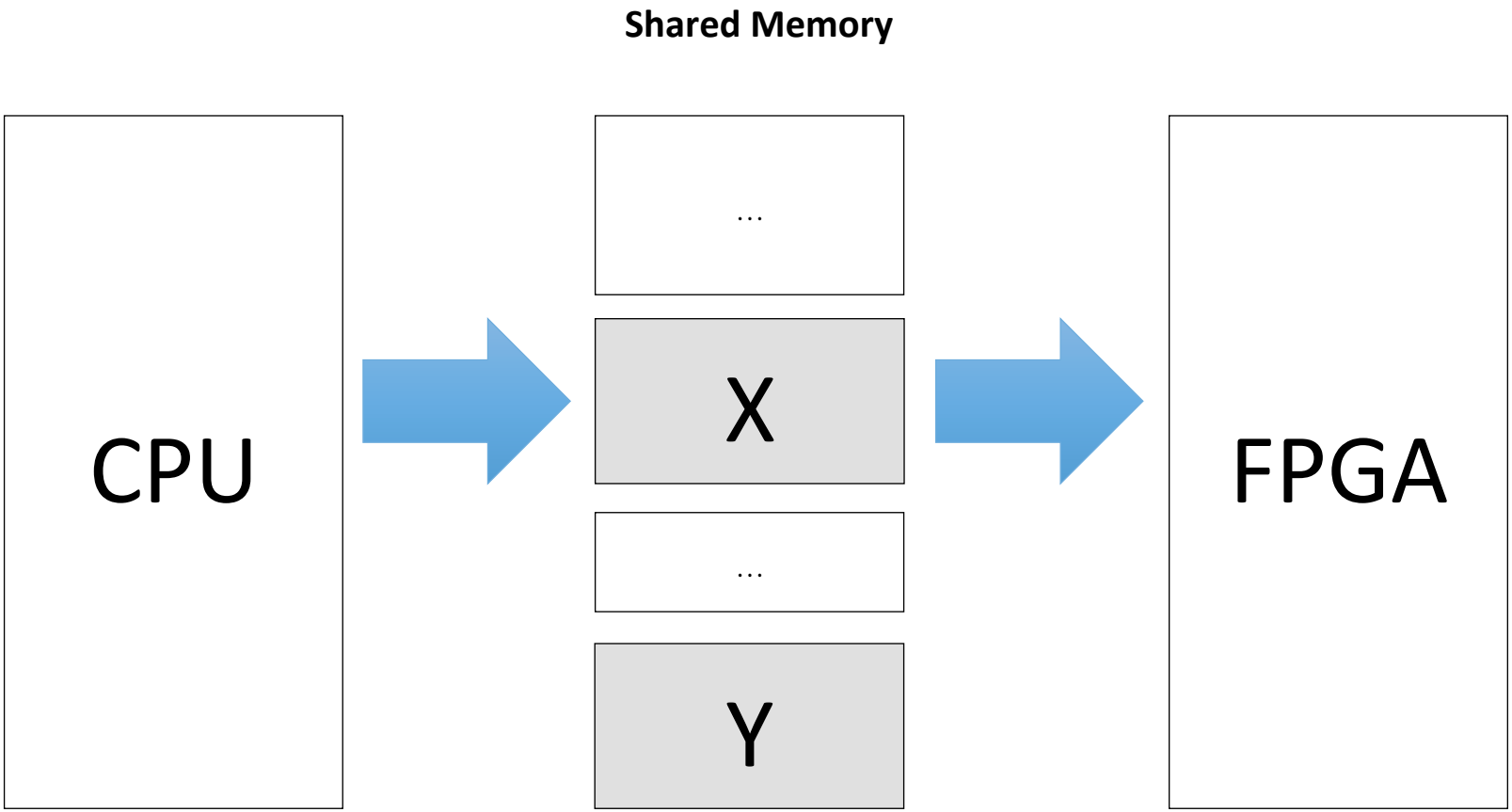
```
/* some code */

#pragma omp target device(HARP) map(to: X) map(from: Y)
#pragma omp parallel for use(hrw) module(loopback)
for (int i = 0; i < elements_size; i++) {
  Y[i] = X[i];
}

/* some code */
```

www.hardcloud.org

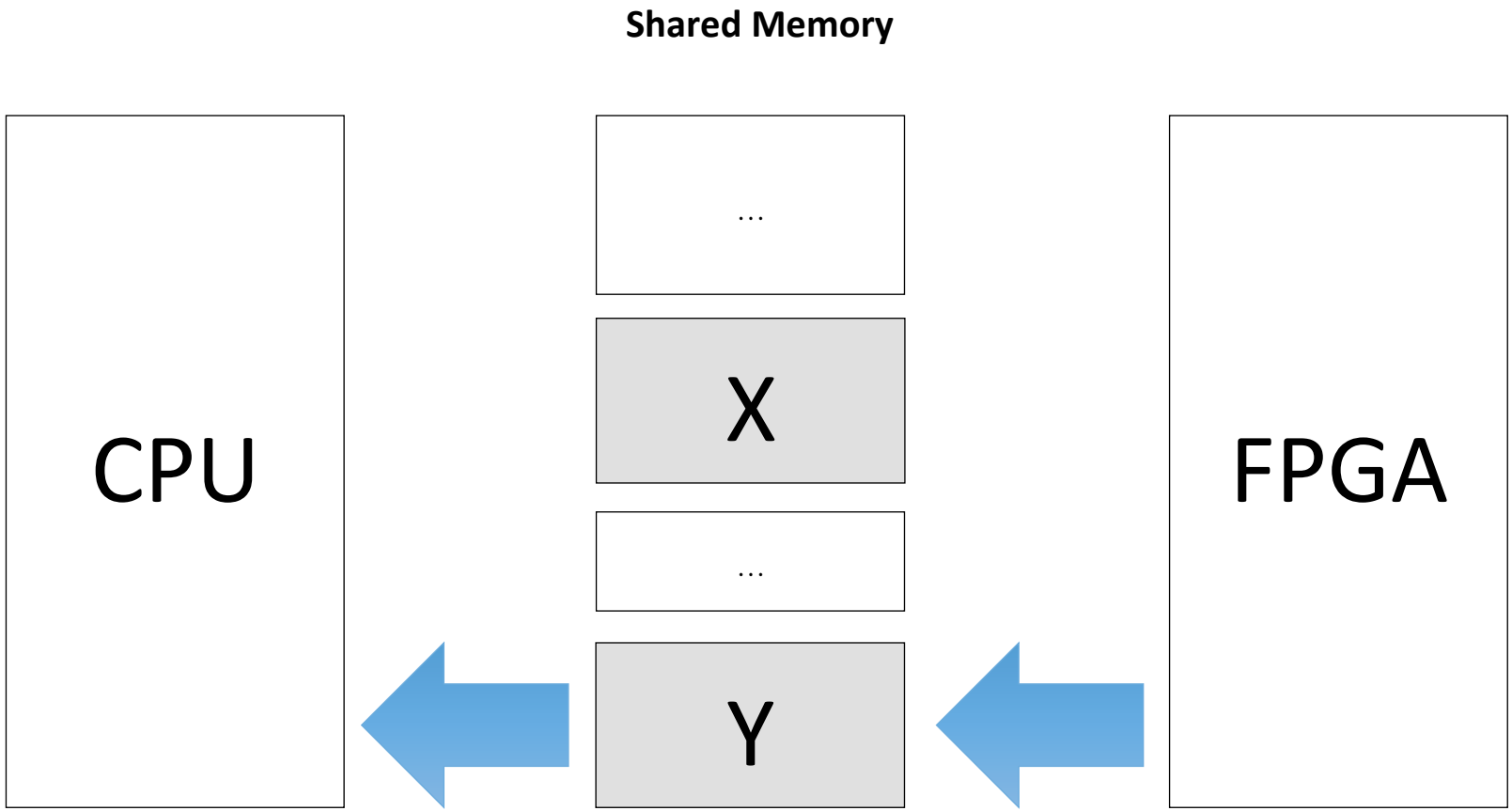
Supported by:

Intel

```
#pragma omp target device(HARP) map(to: X) map(from: Y)
```



```
#pragma omp target device(HARP) map(to: X) map(from: Y)
```



How to verify the IP-core against the software?

HW/SW co-validation (Intel HARP)

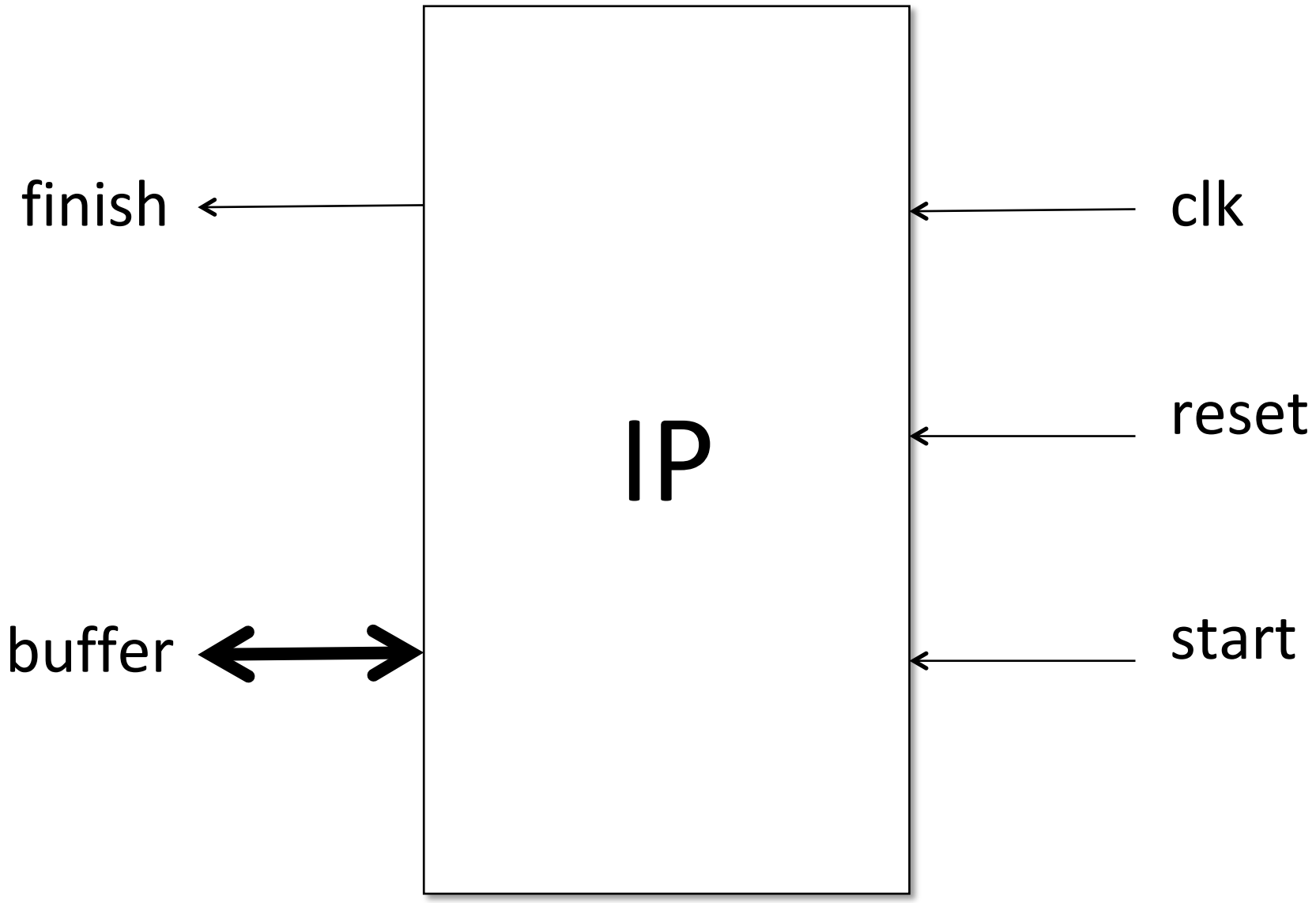
(FCCM 2018)

```
/* some code */

#pragma omp target device(HARP) map(to: X) map(from: Y)
#pragma omp parallel for use(hrw) module(loopback) check
for (int i = 0; i < elements_size; i++) {
  Y[i] = X[i];
}

/* some code */
```

www.hardcloud.org

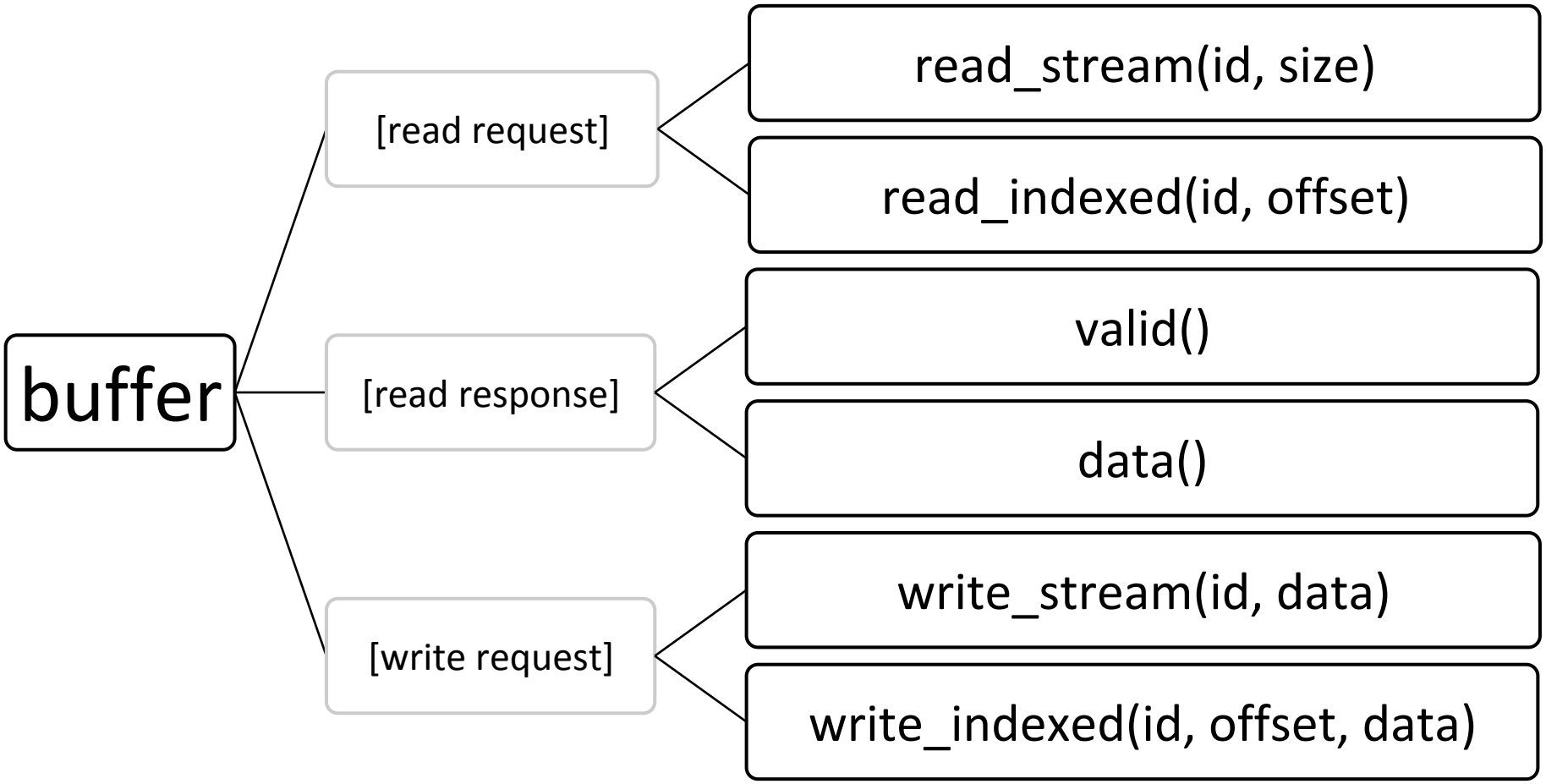
Supported by:
Intel

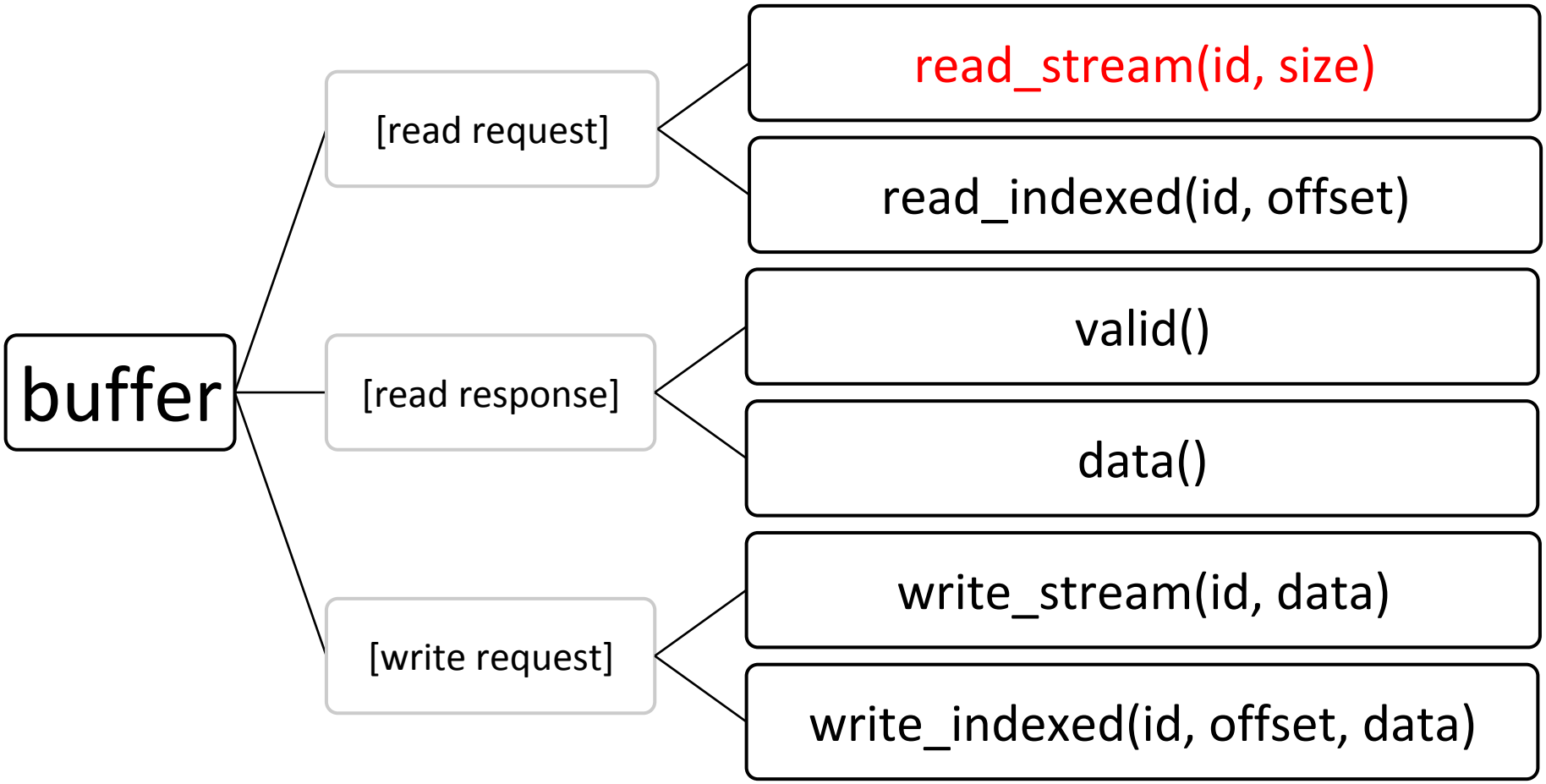
How to embed your IP-core?

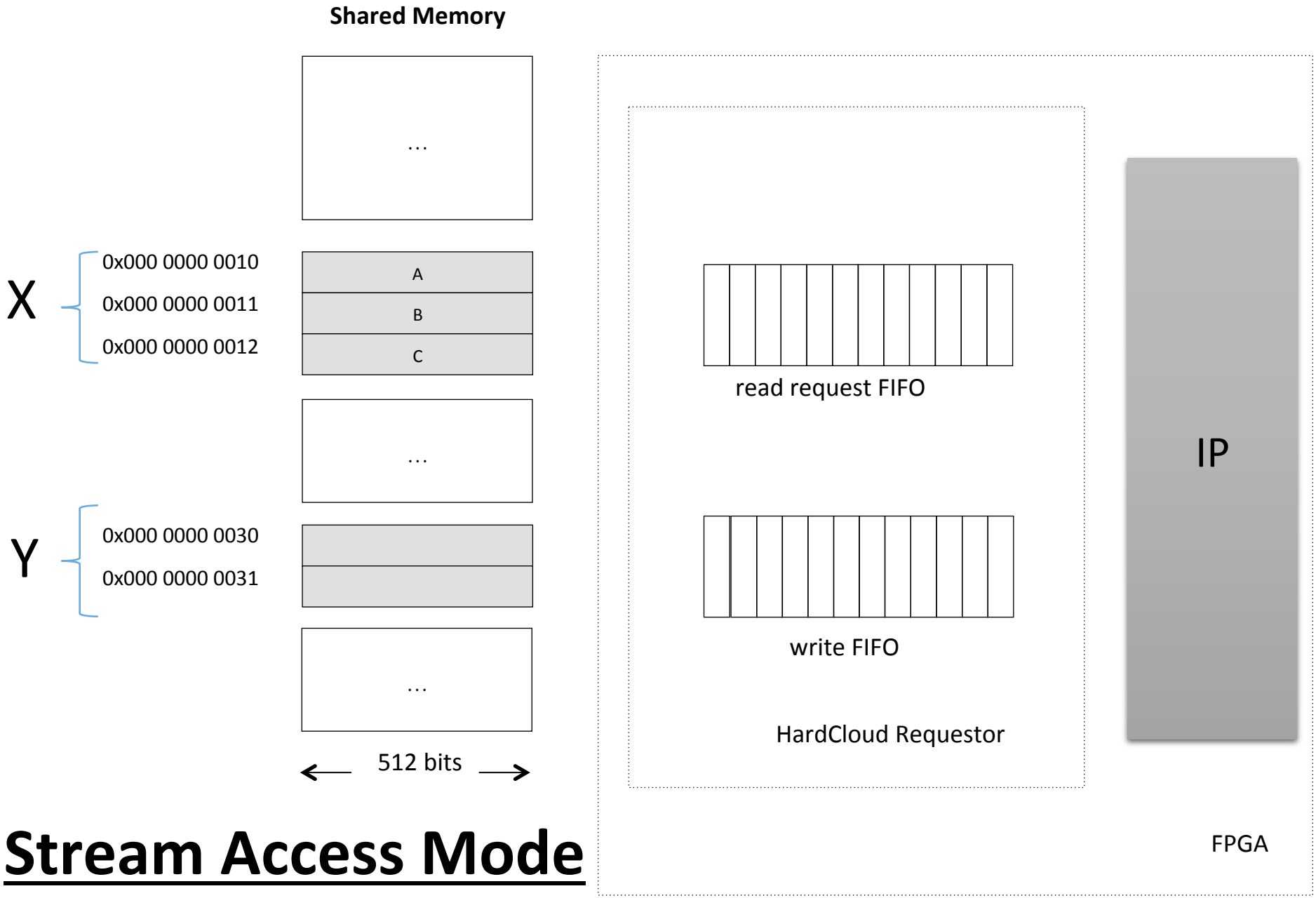

finish
clk
IP
reset
buffer
start

```
module IP
(
  input  logic  clk,
  input  logic  reset,
  input  logic  start,
  output logic  finish,
  hc_buffers_if buffer
);

/* IP implementation */

endmodule : IP
```

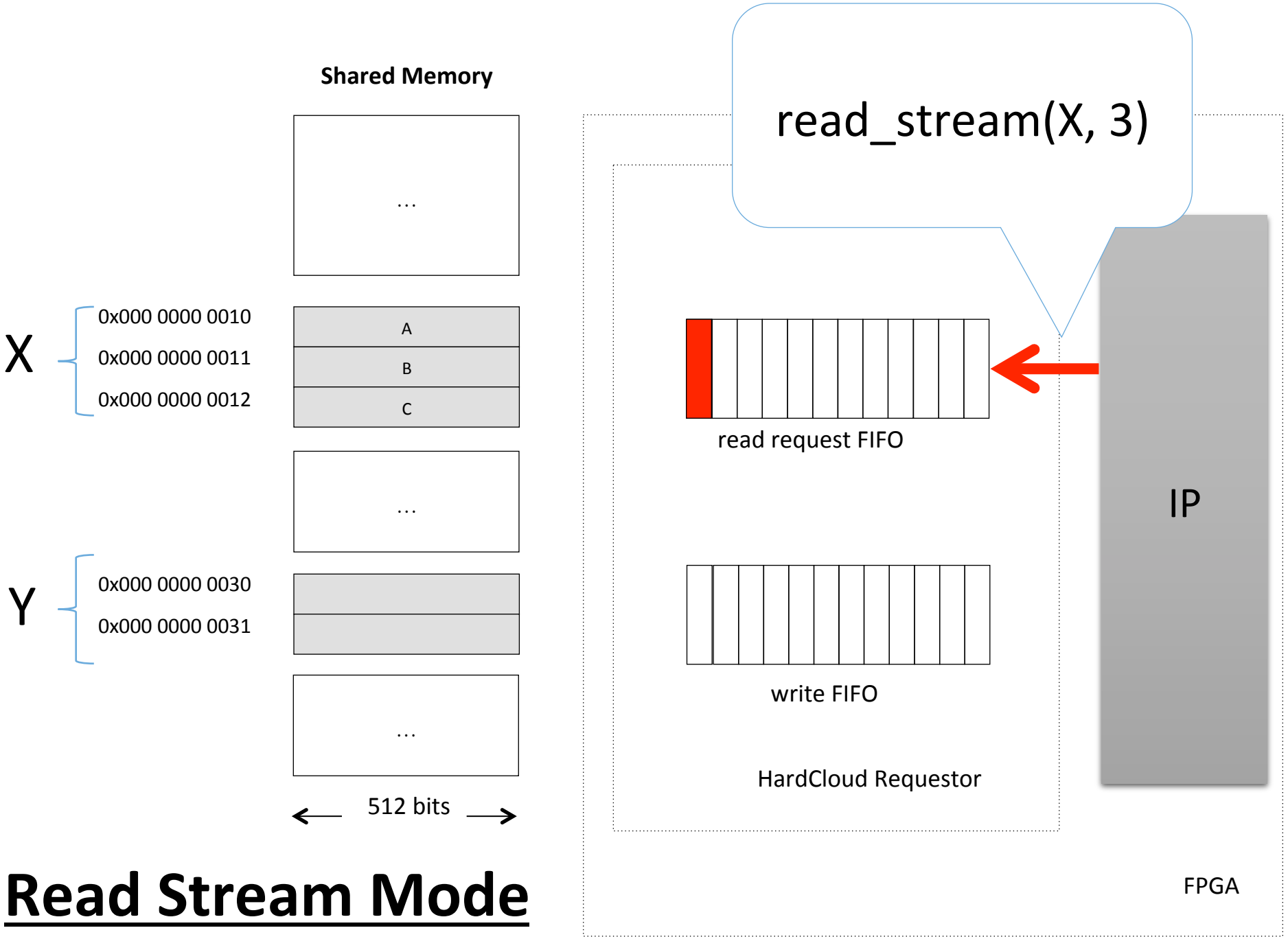
How to access the data?


buffer
[read request]
[read response]
[write request]
read_stream(id, size)
read_indexed(id, offset)
valid()
data()
write_stream(id, data)
write_indexed(id, offset, data)

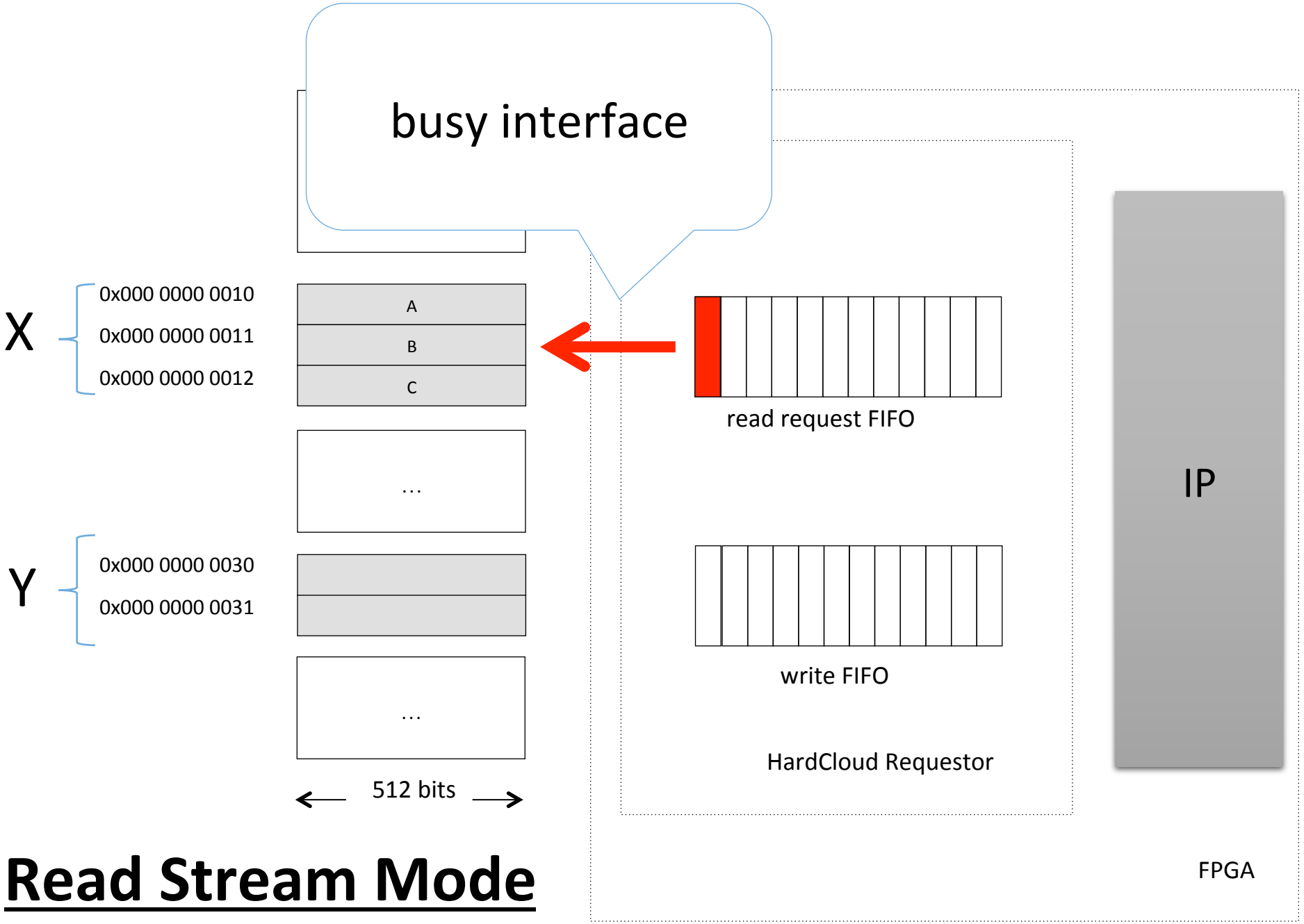

buffer
[read request]
[read response]
[write request]
read_stream(id, size)
read_indexed(id, offset)
valid()
data()
write_stream(id, data)
write_indexed(id, offset, data)



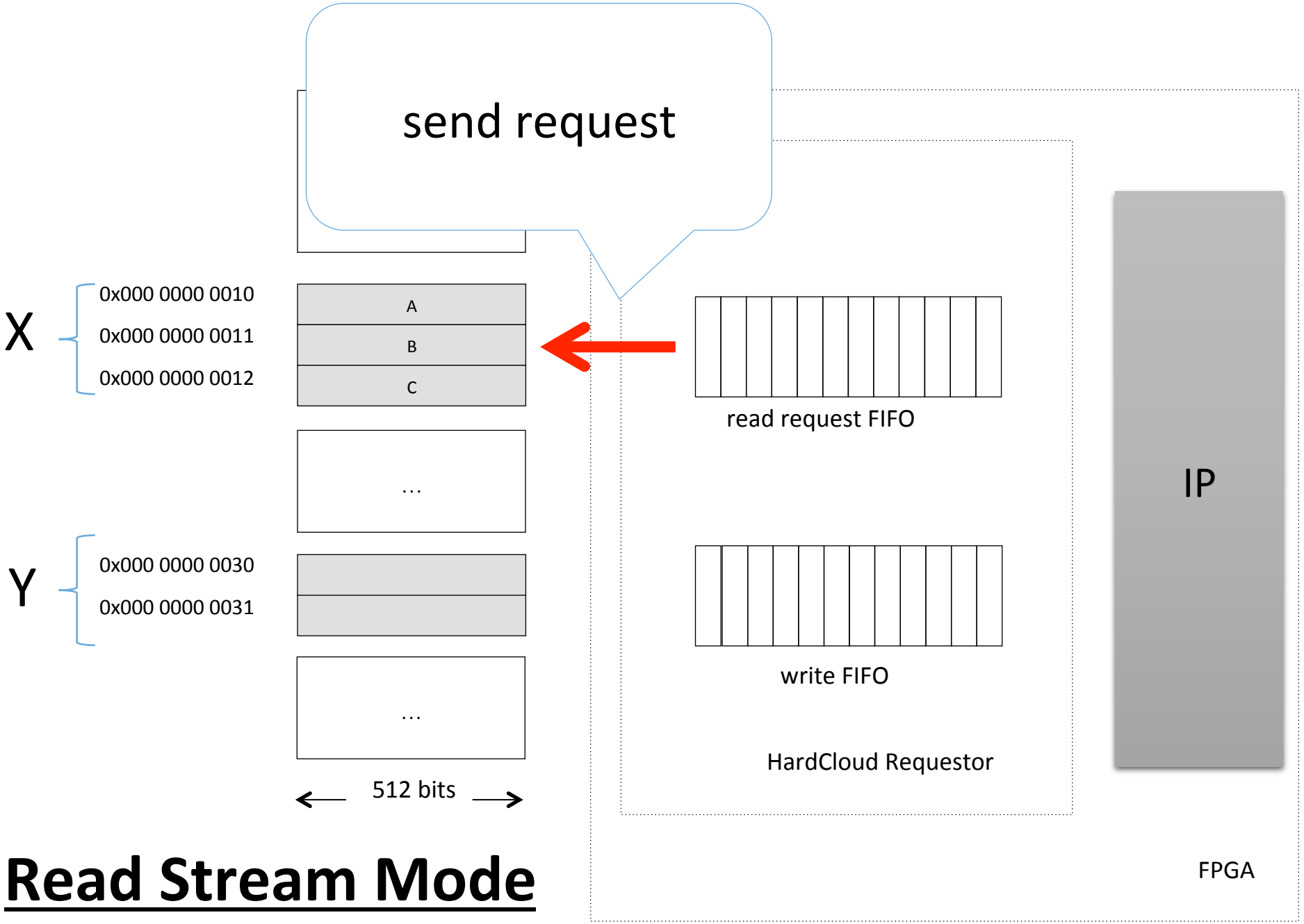
Stream Access Mode



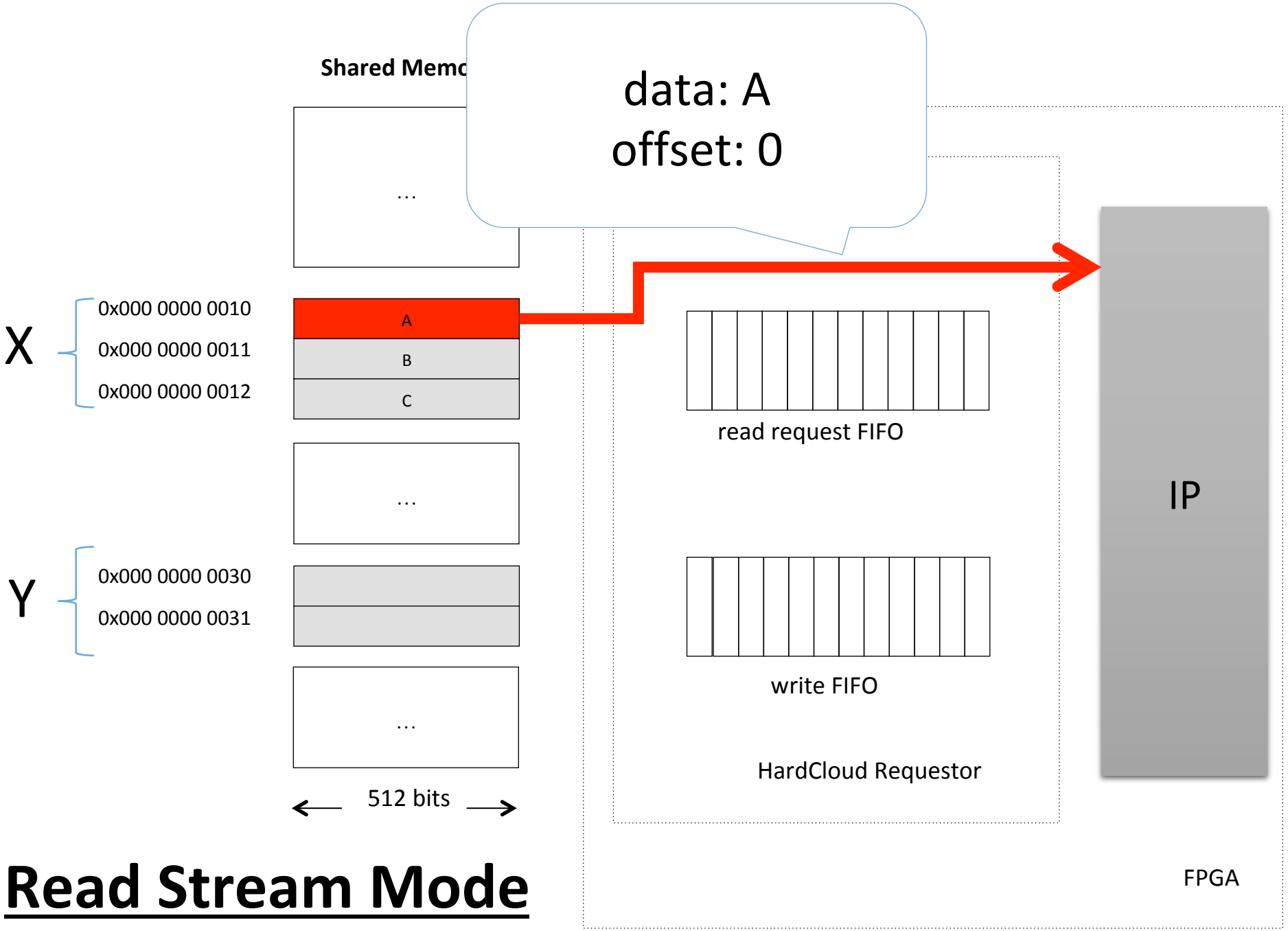
Read Stream Mode



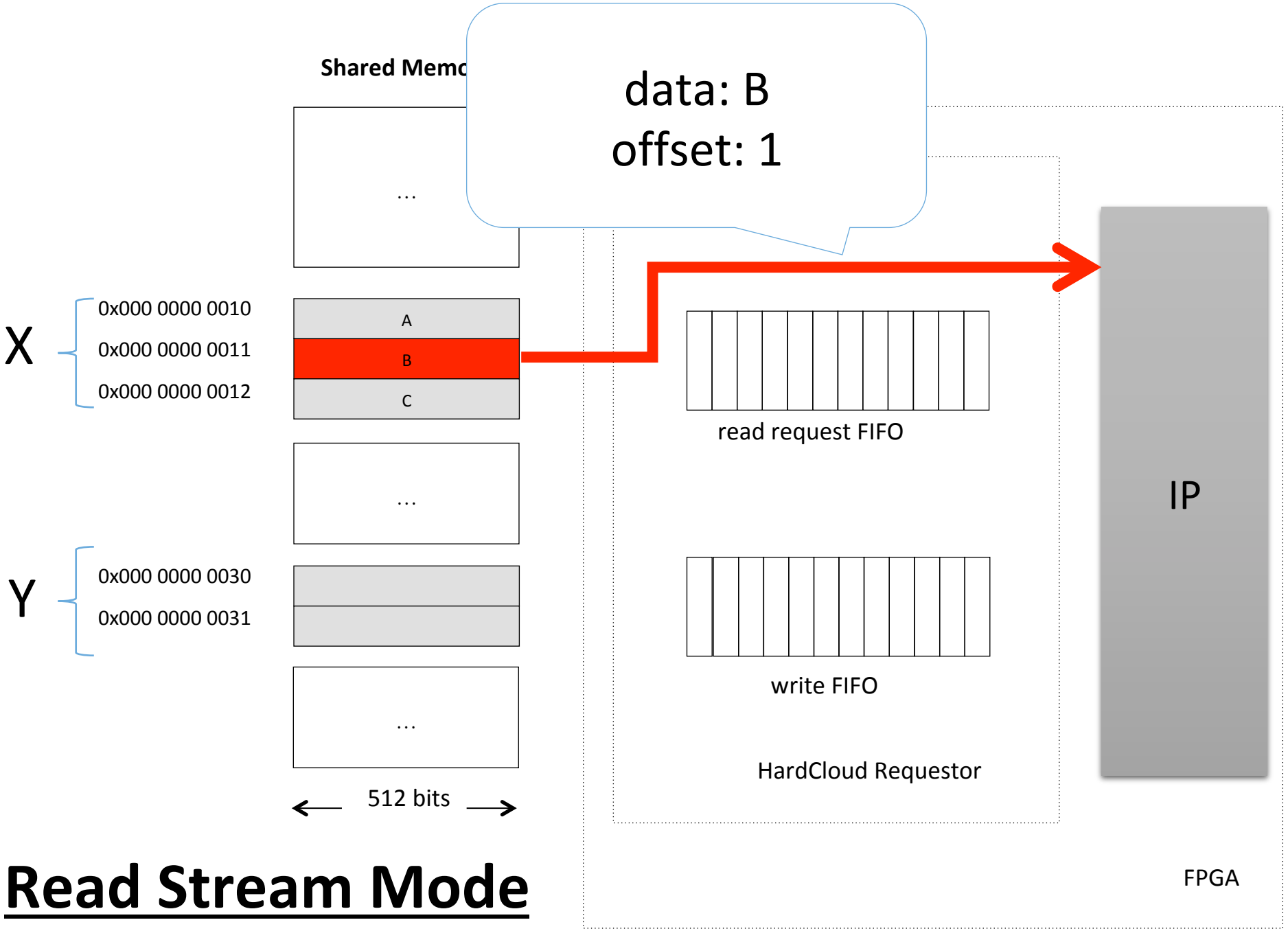
Read Stream Mode



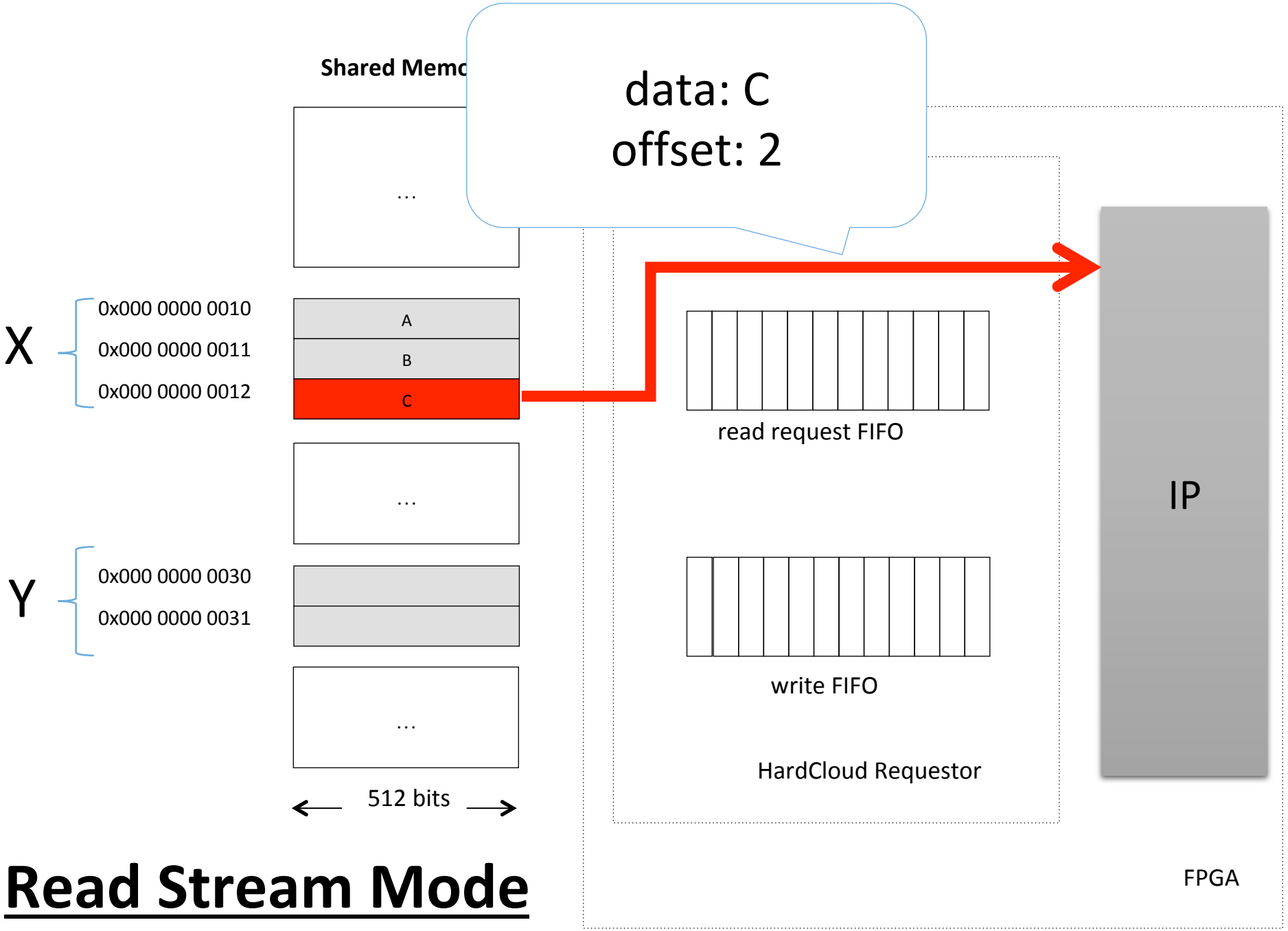
Read Stream Mode



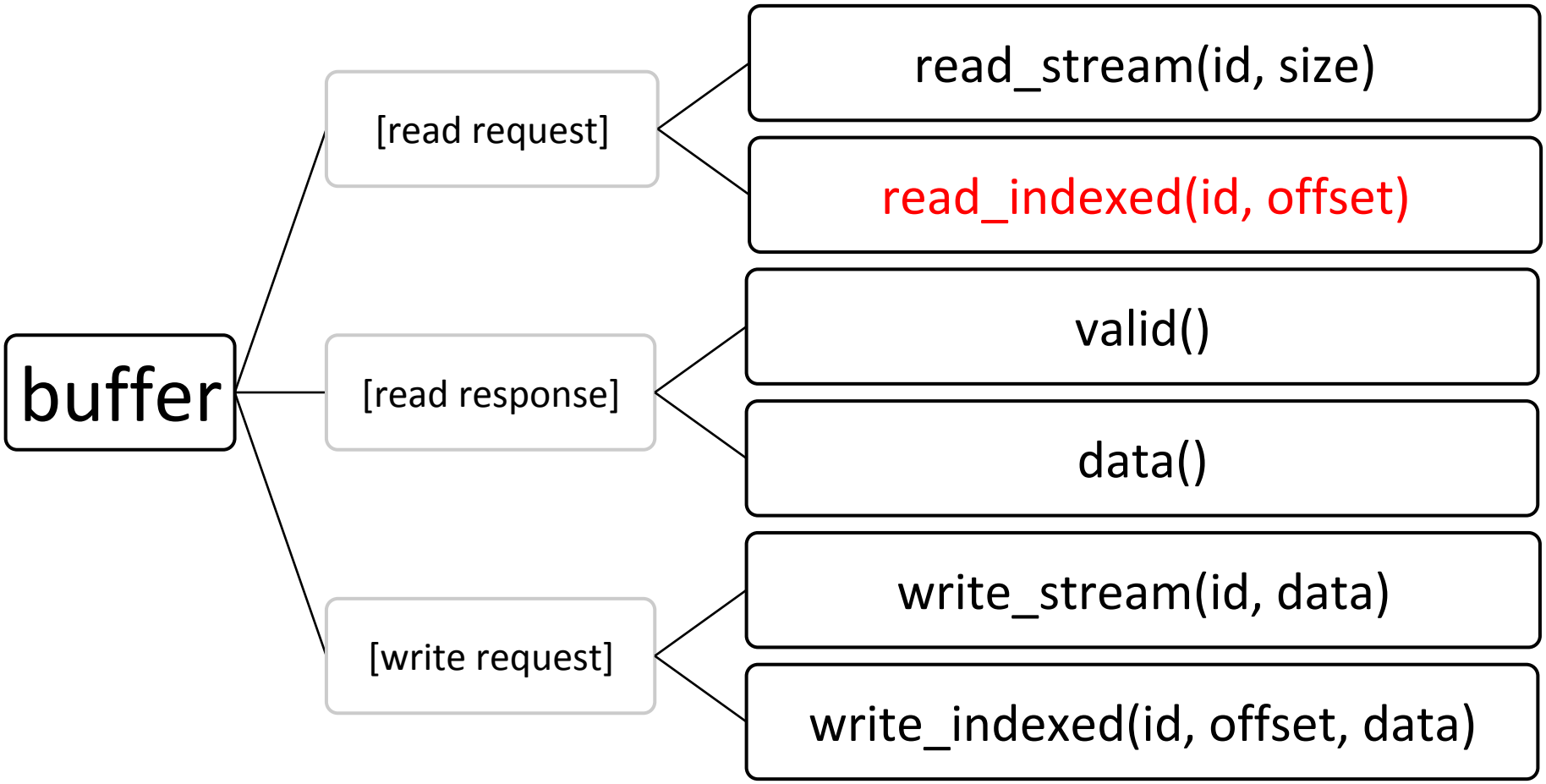
Read Stream Mode

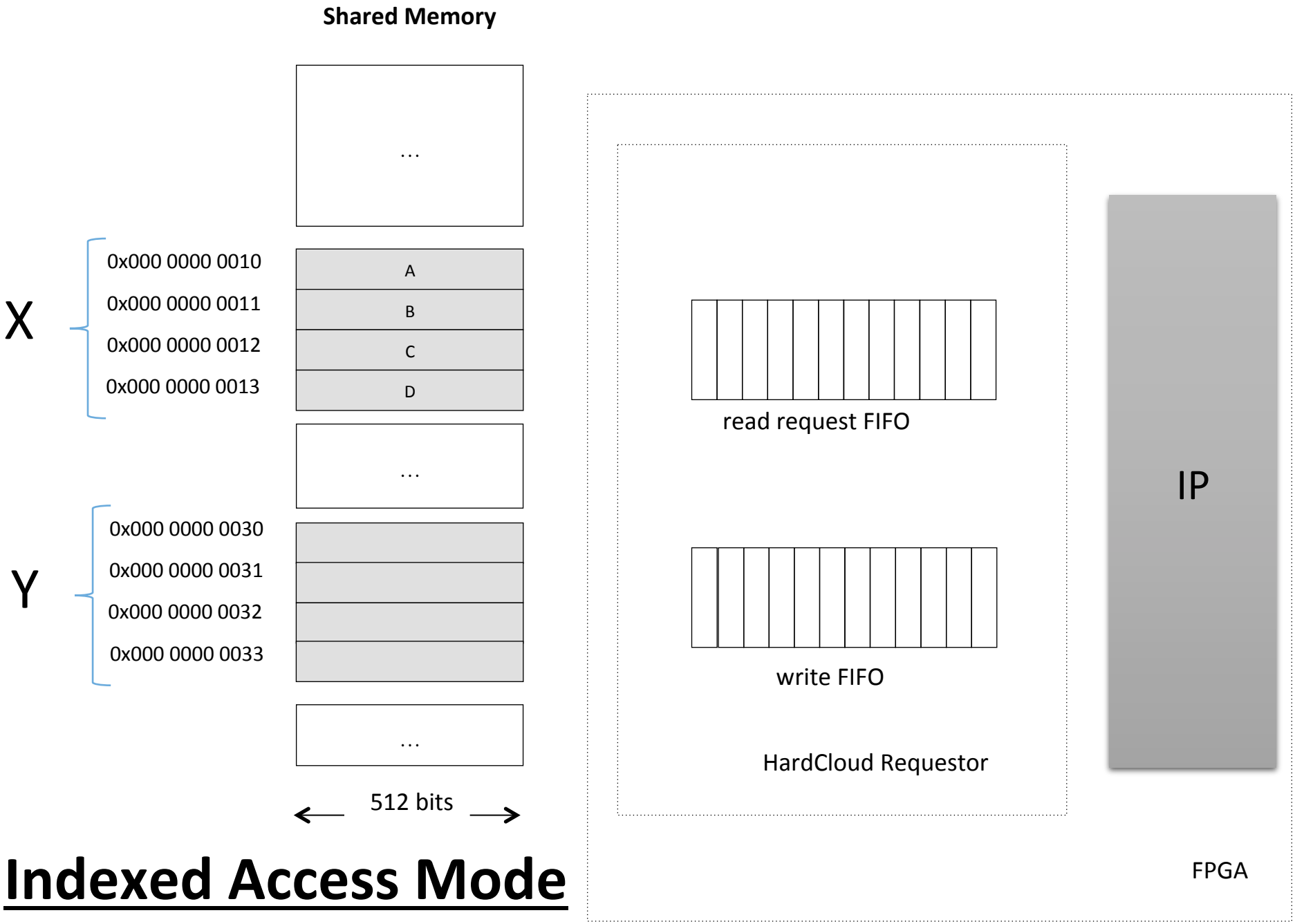


Read Stream Mode

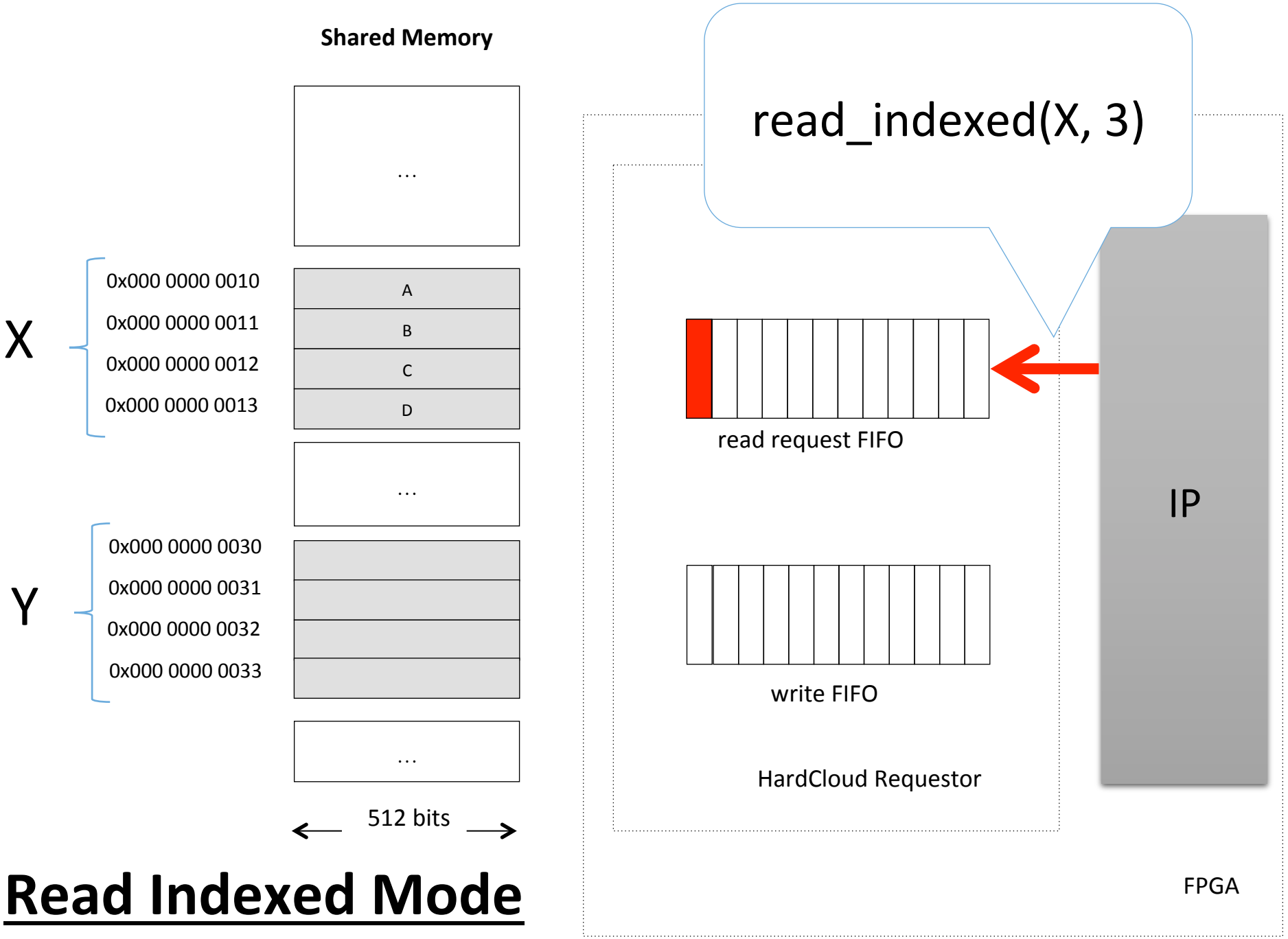


Read Stream Mode

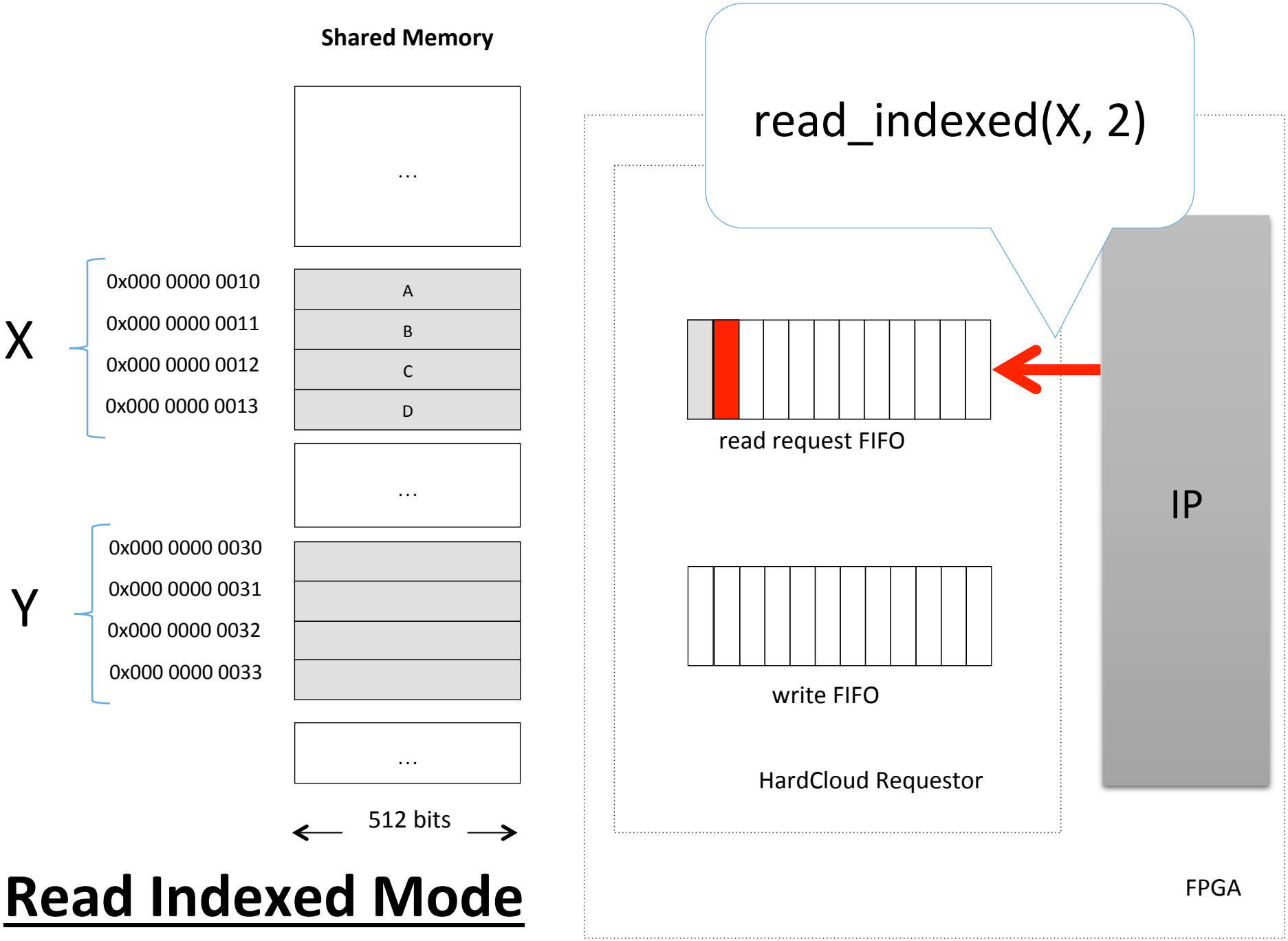

buffer
[read request]
[read response]
[write request]
read_stream(id, size)
read_indexed(id, offset)
valid()
data()
write_stream(id, data)
write_indexed(id, offset, data)



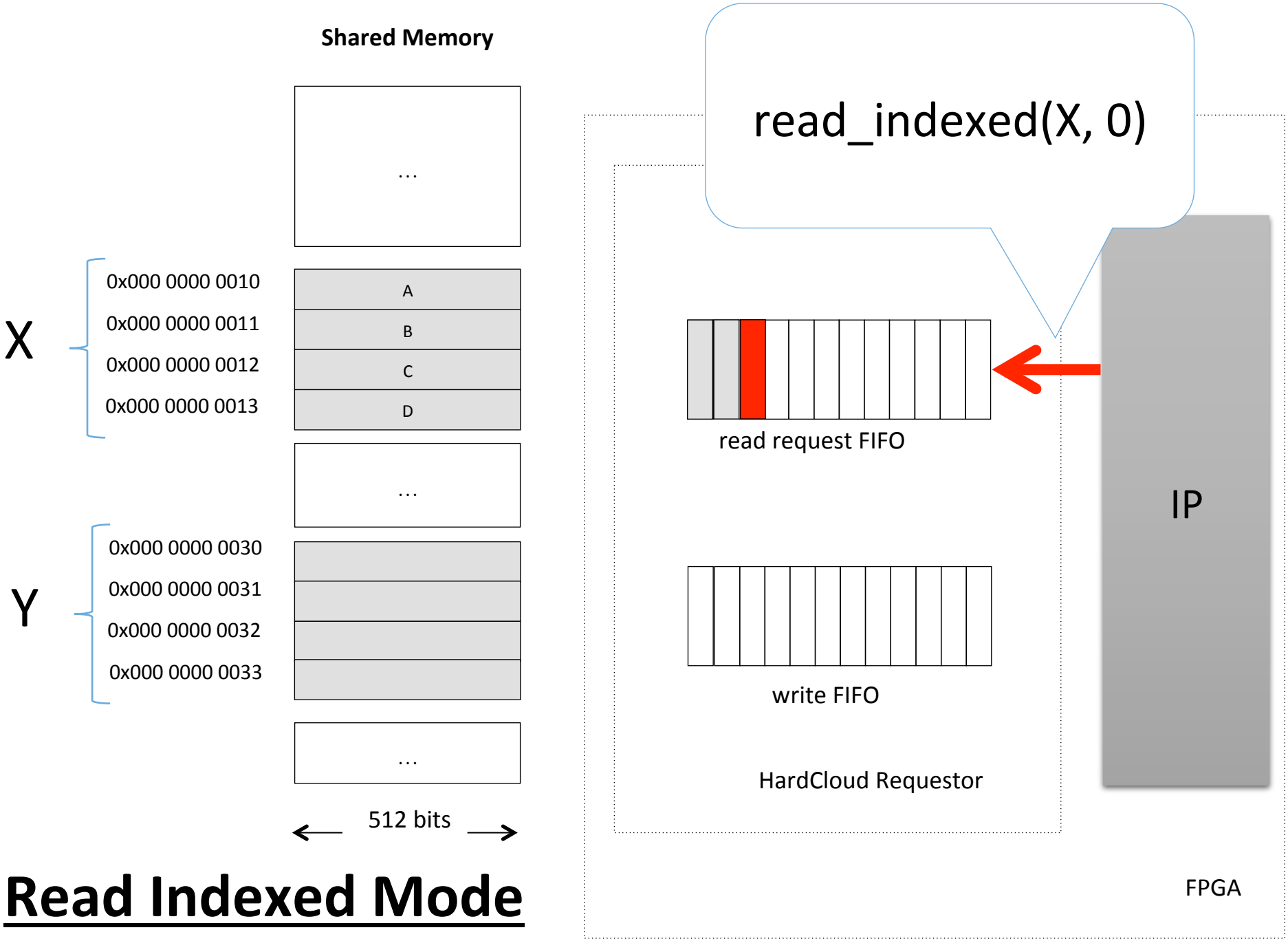
Indexed Access Mode



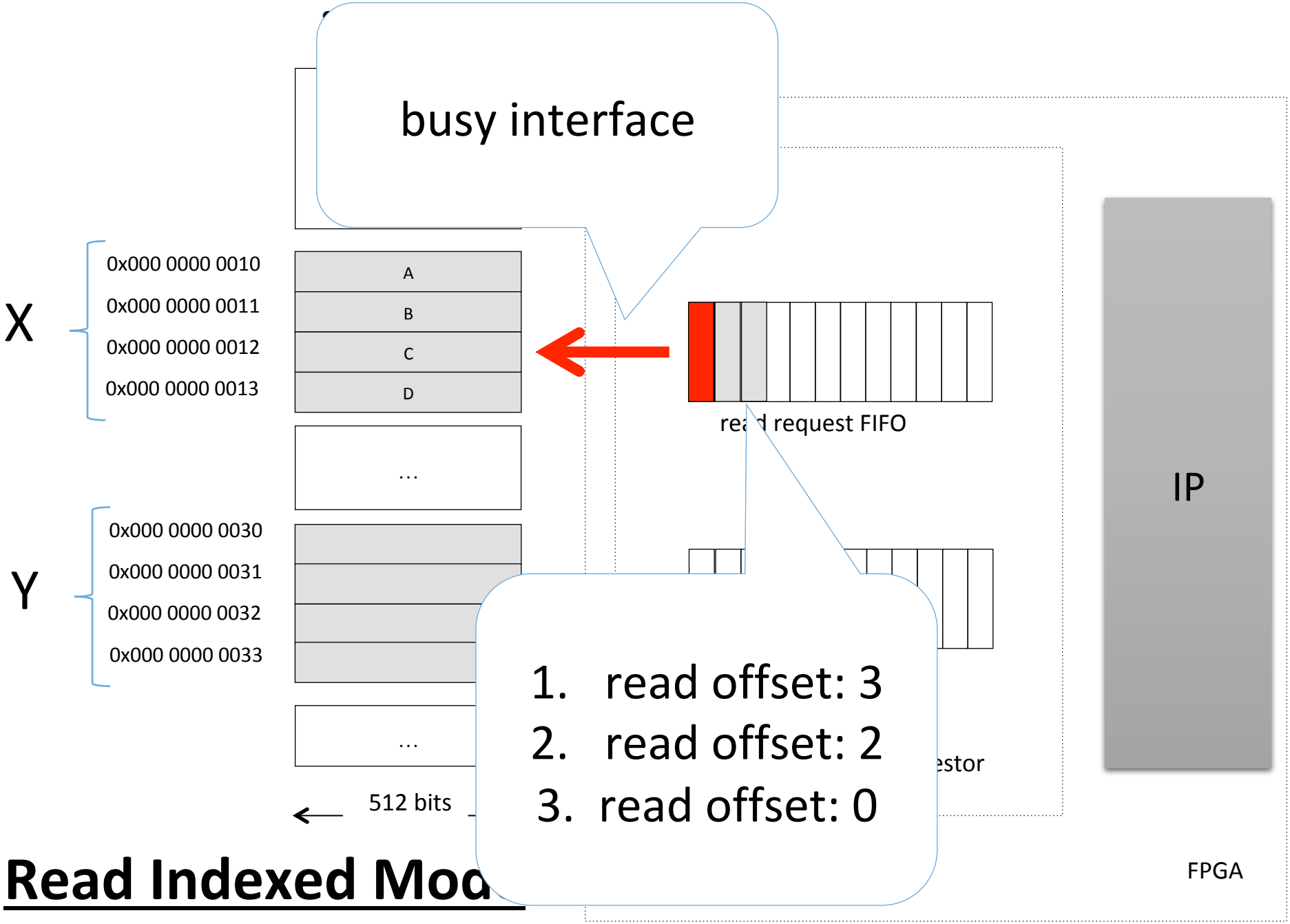
Read Indexed Mode



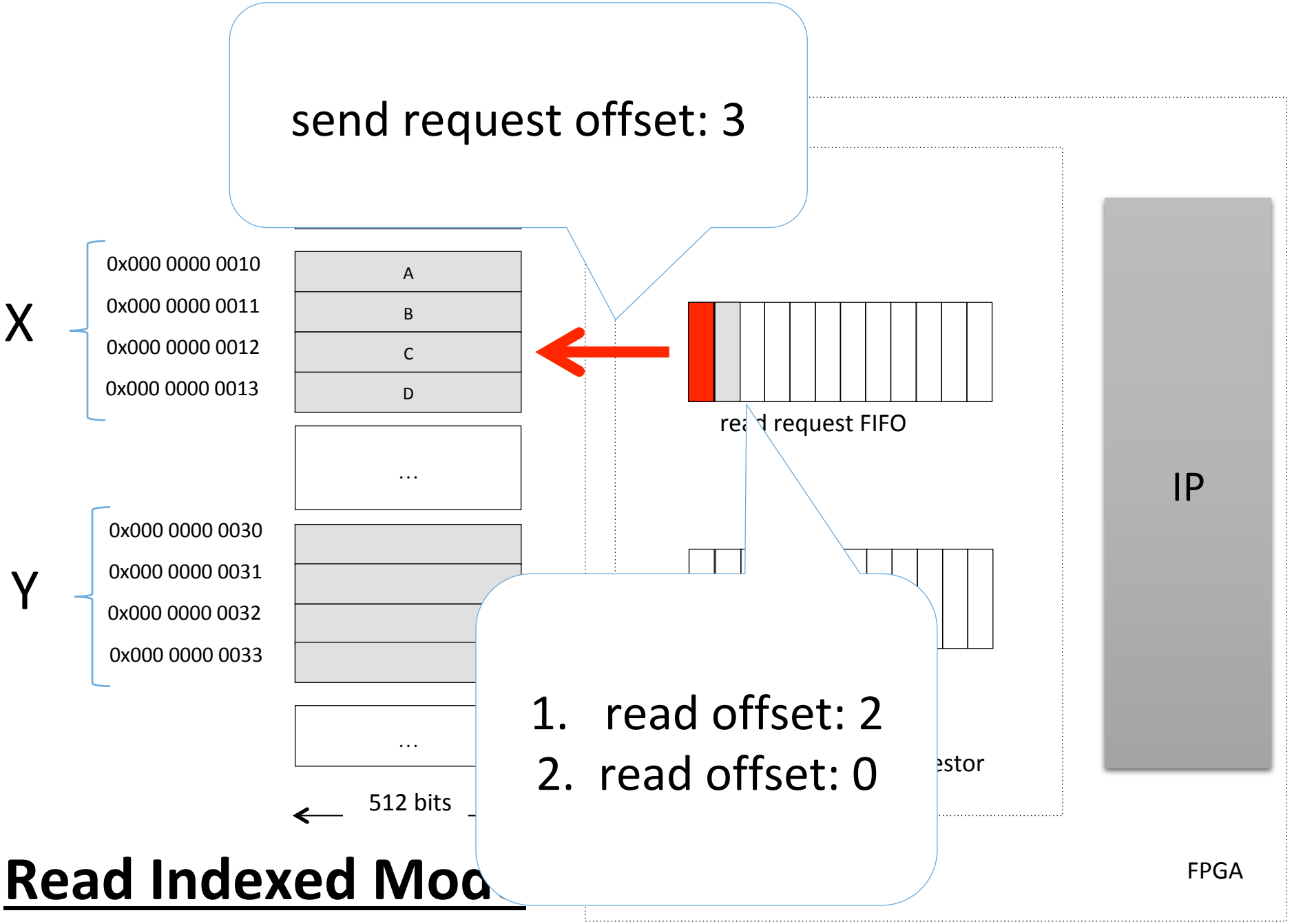
Read Indexed Mode



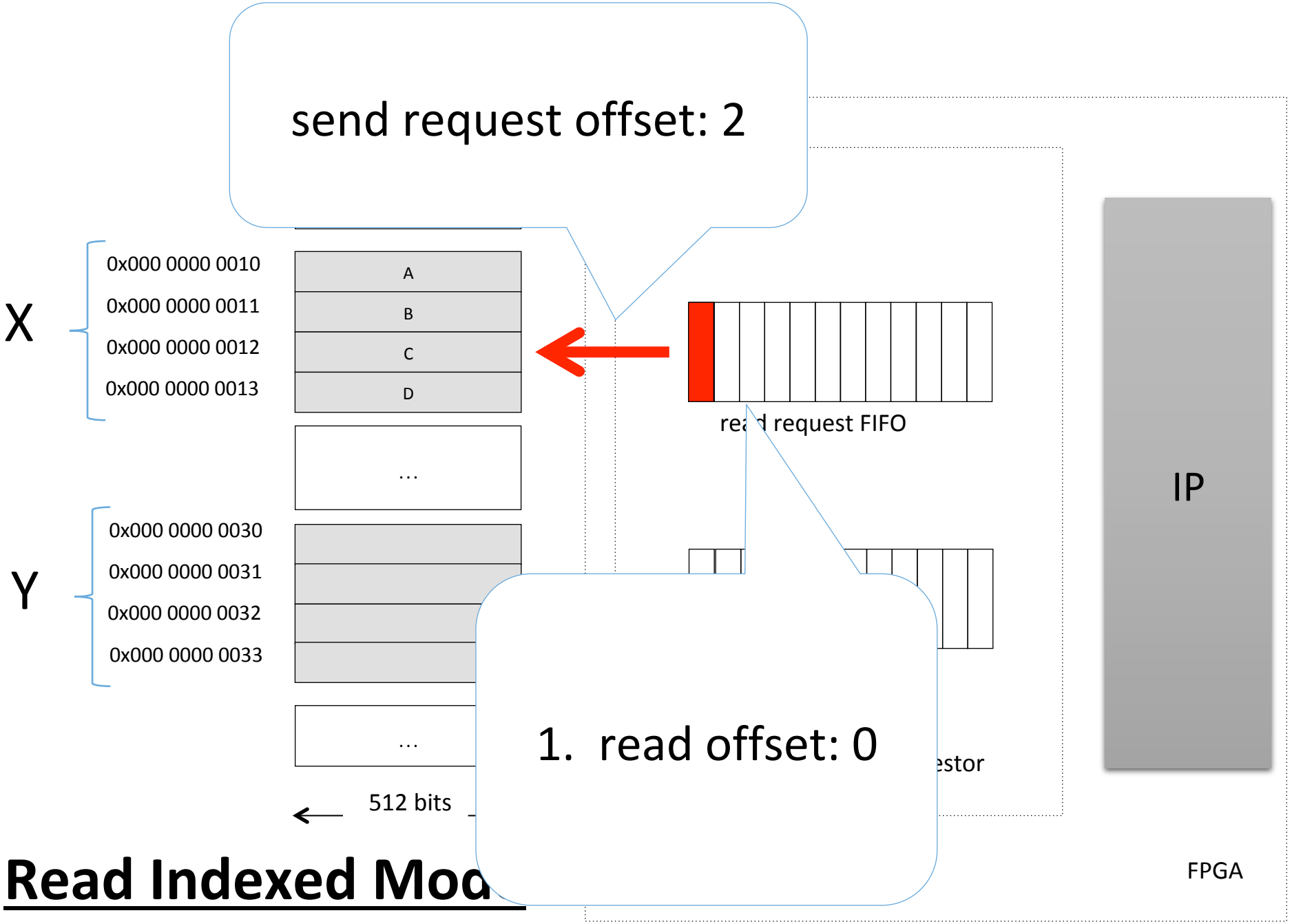
Read Indexed Mode



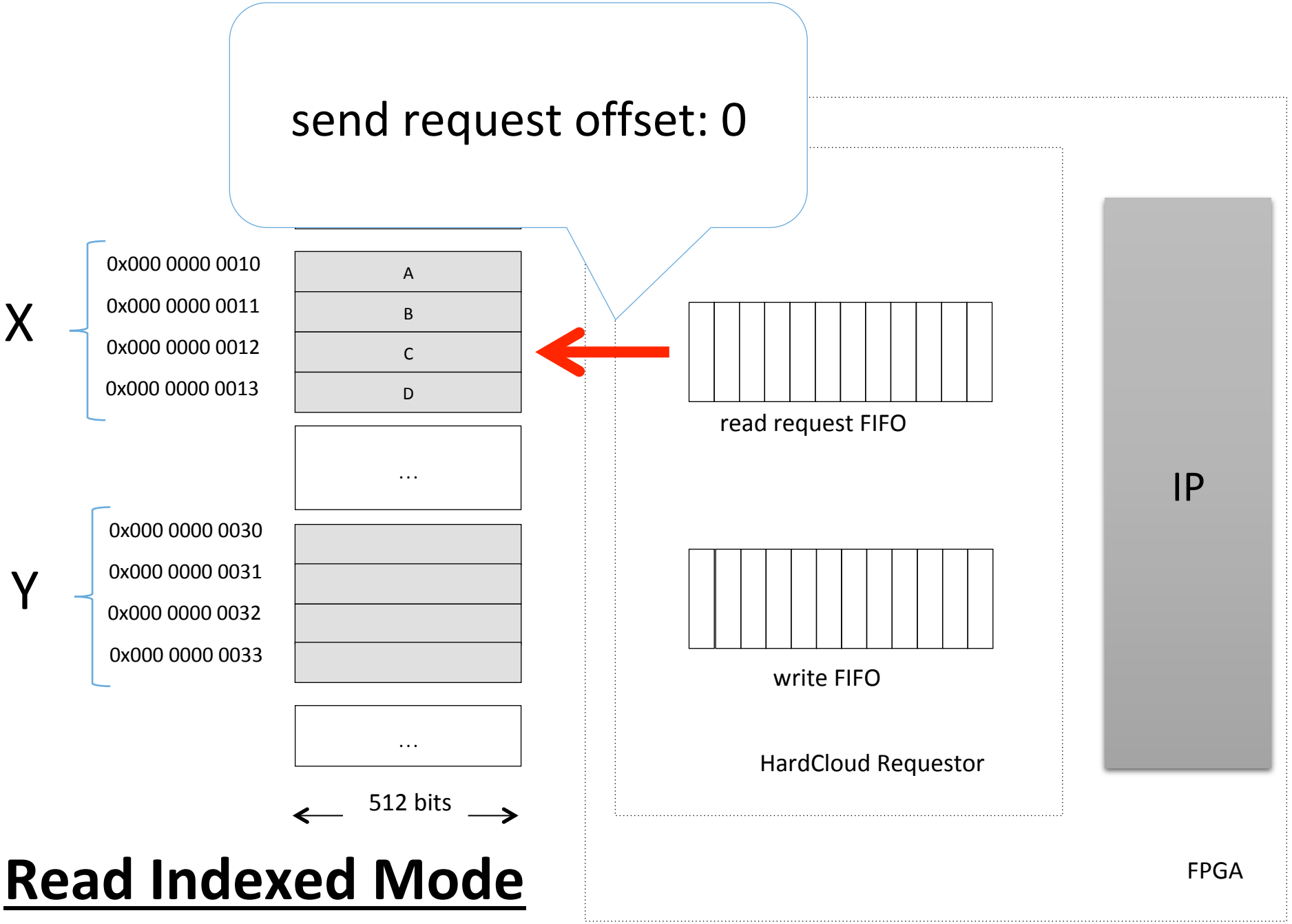
Read Indexed Mod



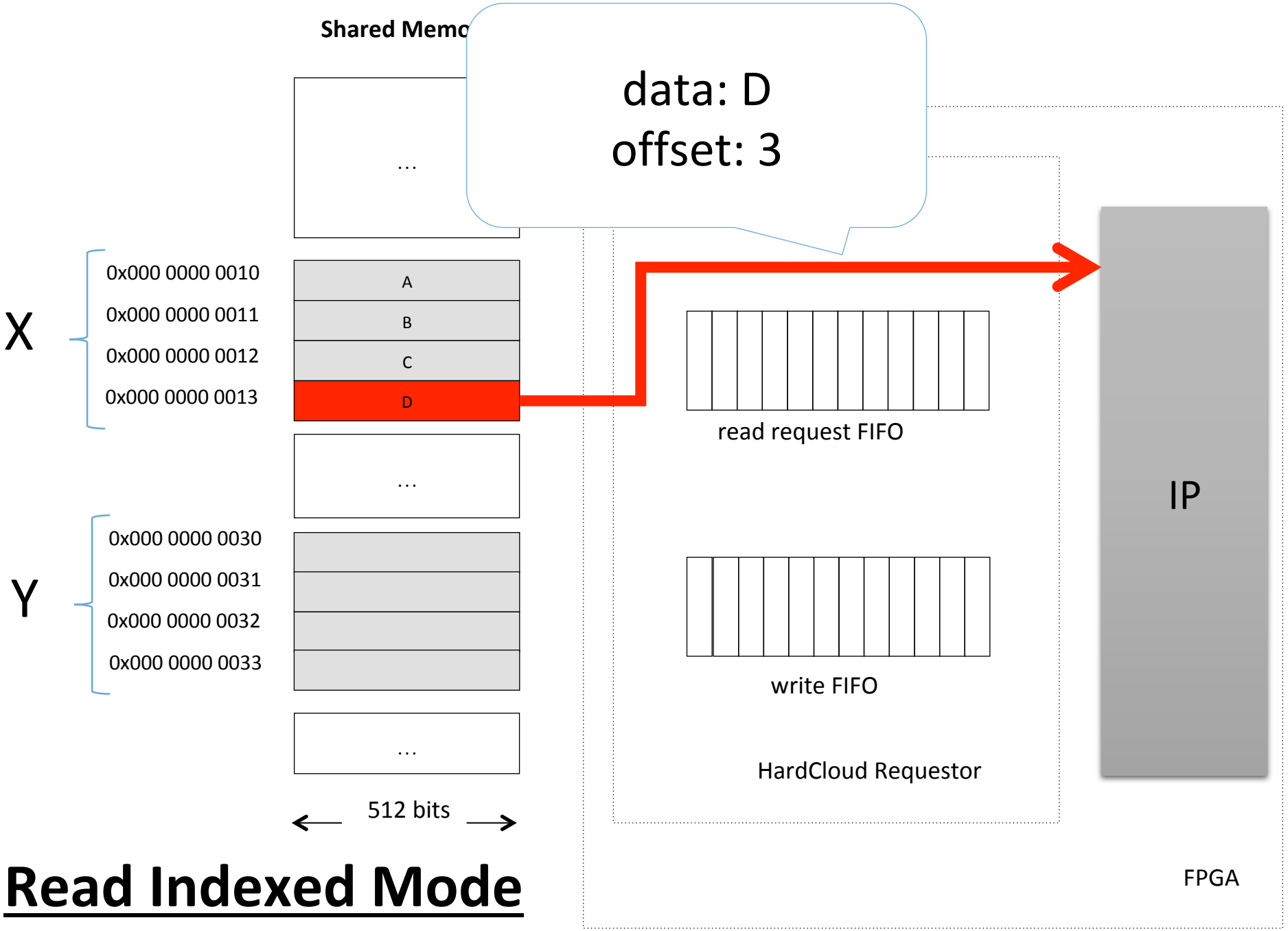
Read Indexed Mod



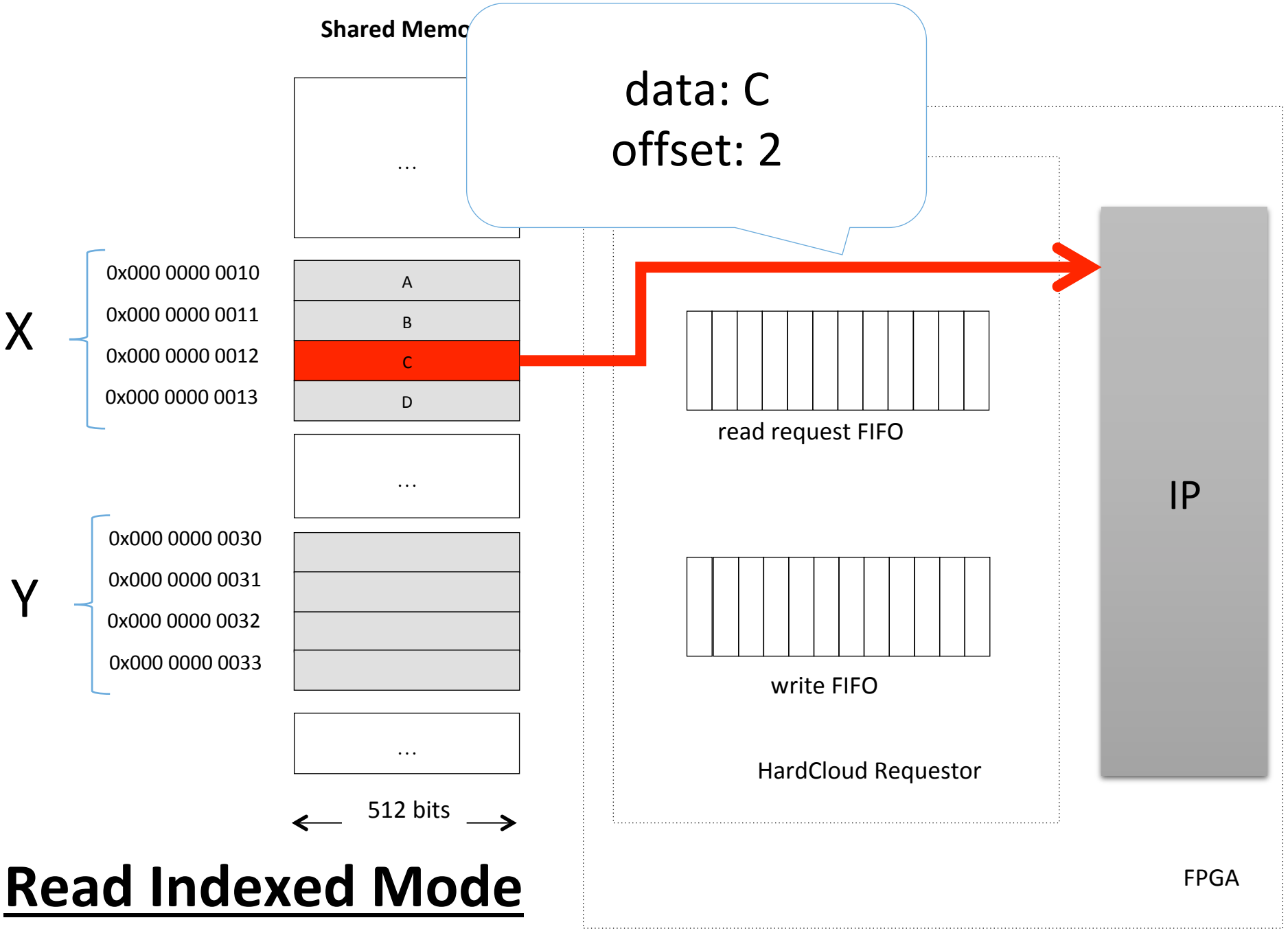
Read Indexed Mod



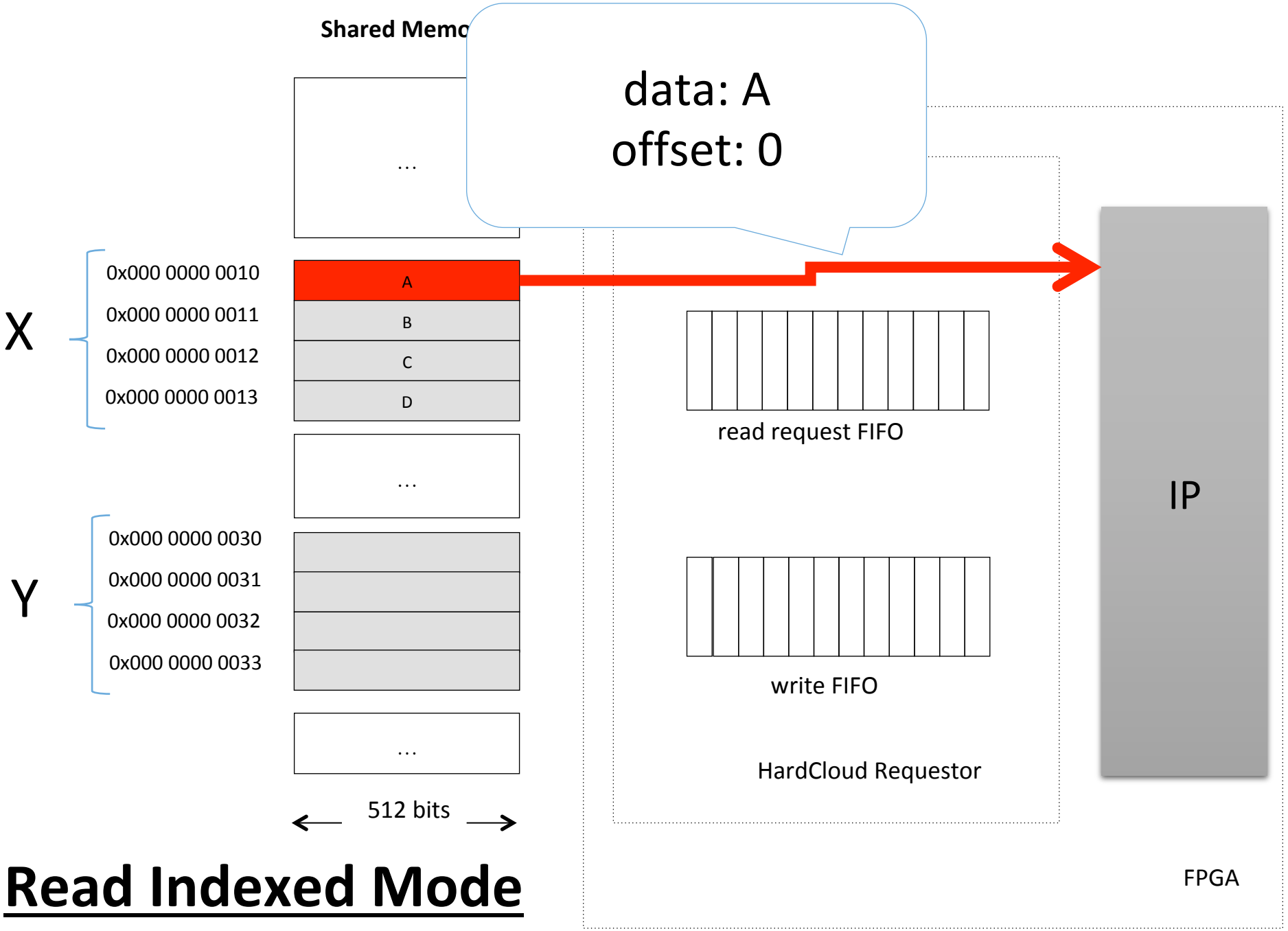
Read Indexed Mode



Read Indexed Mode



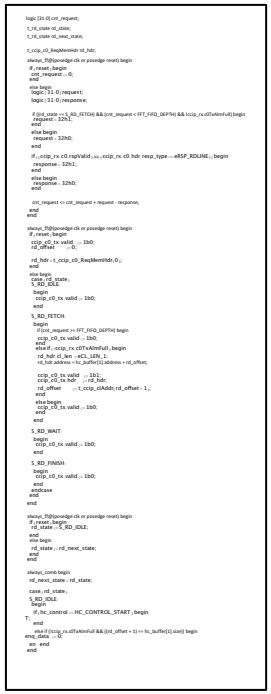
Read Indexed Mode



Read Indexed Mode

Does it help the programmer?

Before HardCloud



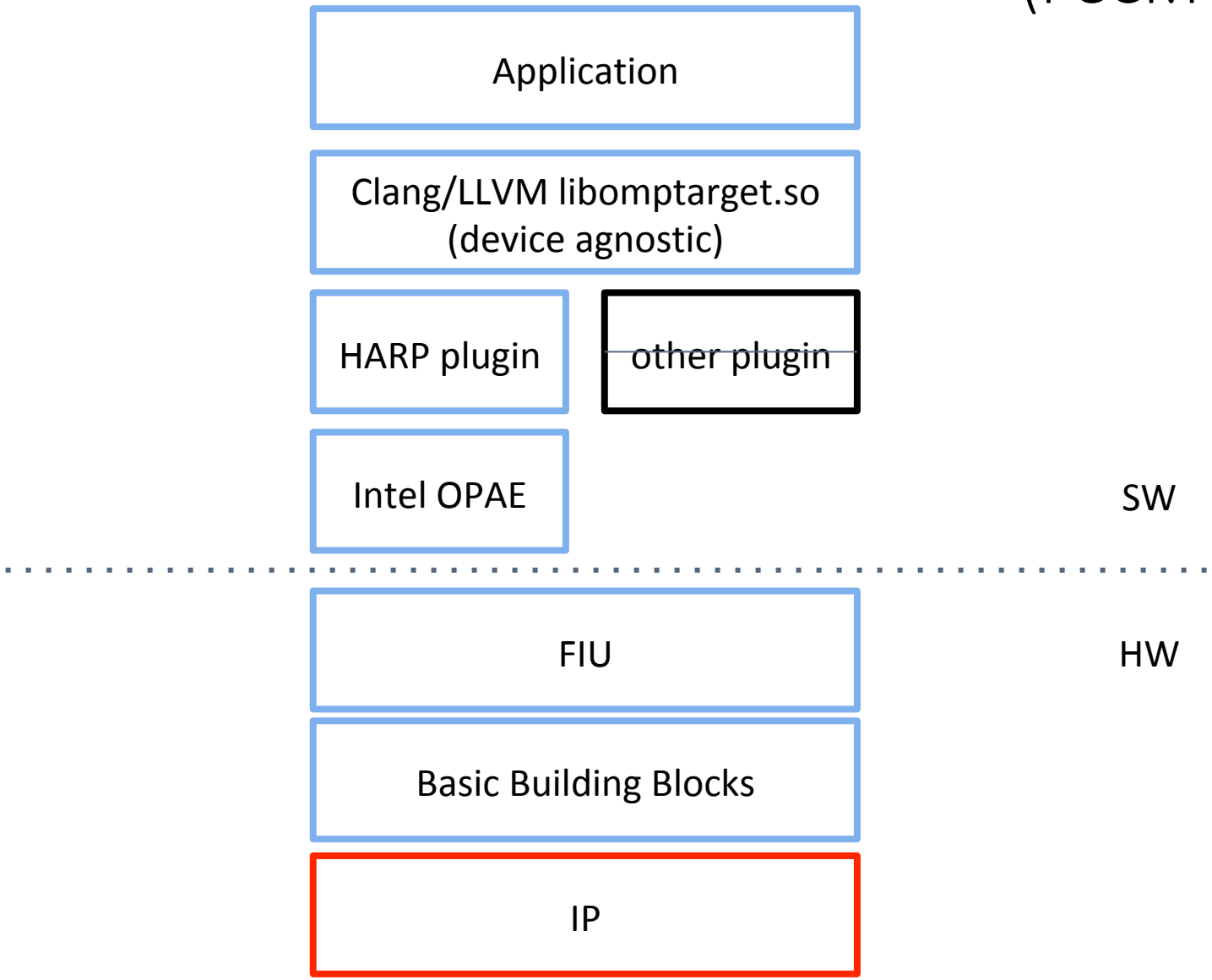
Using HardCloud

```
always_ff@(posedge clk or posedge rst) begin
  if (rst) begin
    buffer.read_idle();
  end
  else begin
    if (!buffer.read_fifo_is_full()) begin
      buffer.read_stream(0, 512);
    end
    else begin
      buffer.read_idle();
    end
  end
end
```

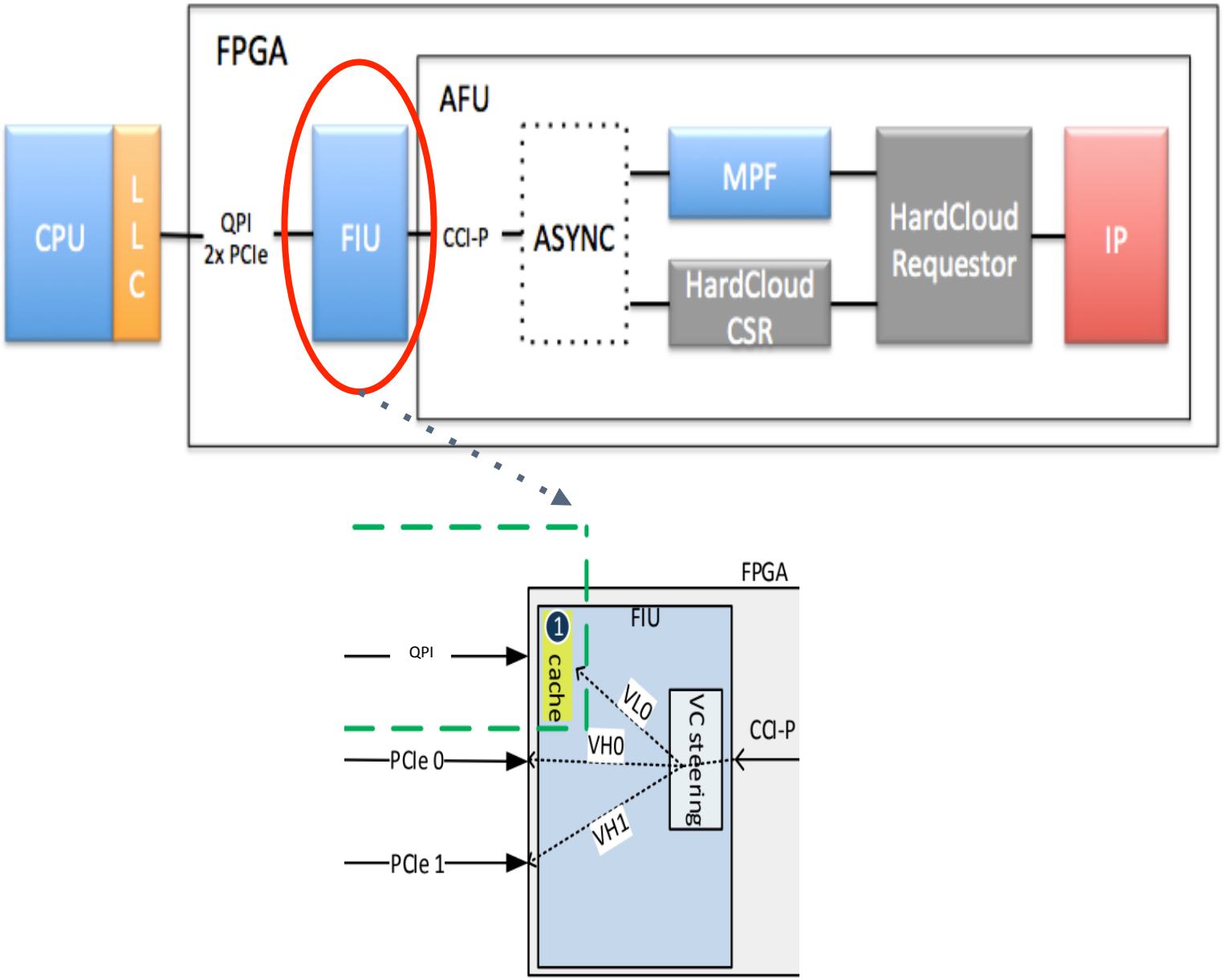
~100 lines of code

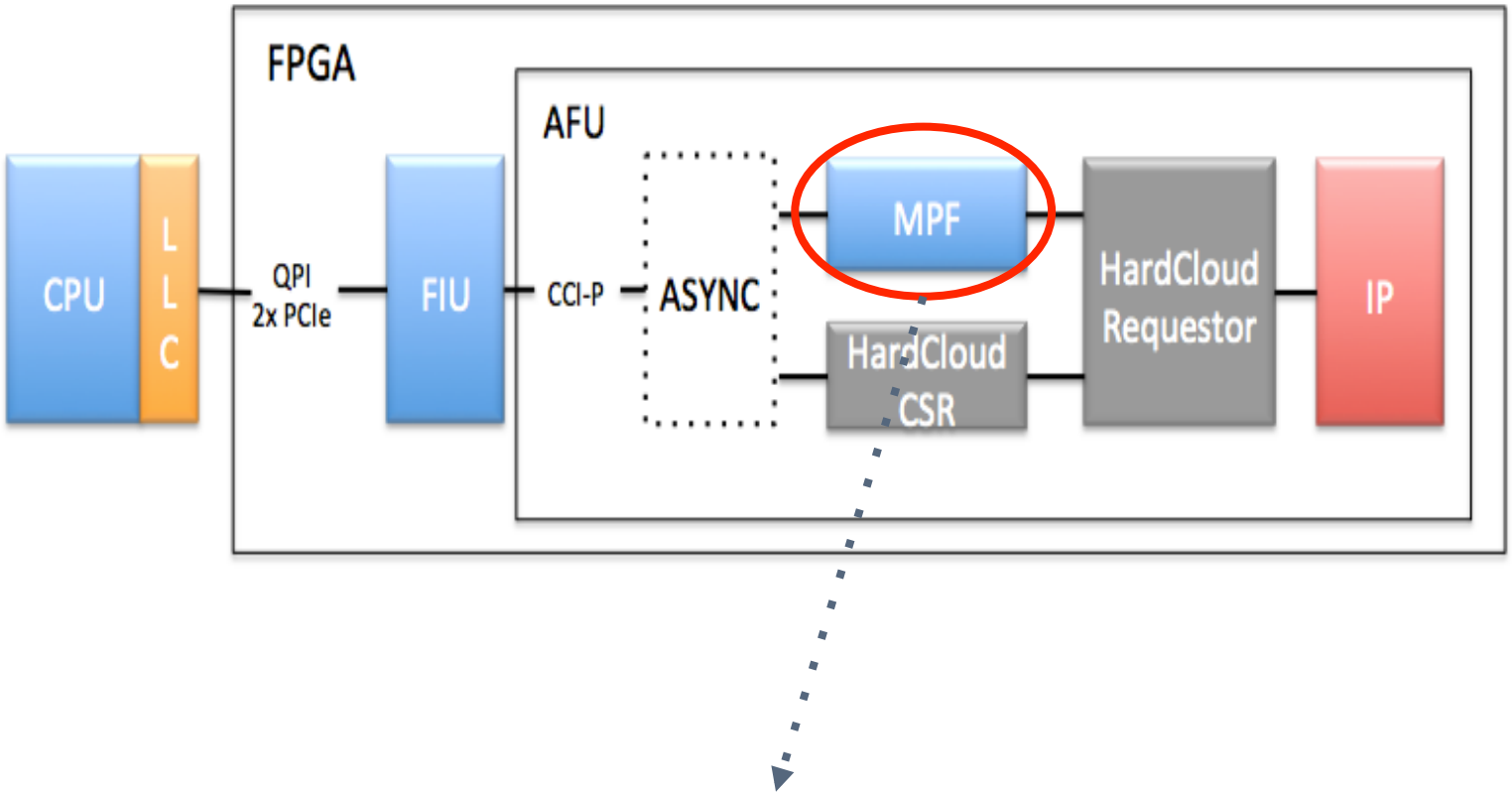
The software behind….

Compatible with LLVM/Clang 4.X (FCCM 2018)

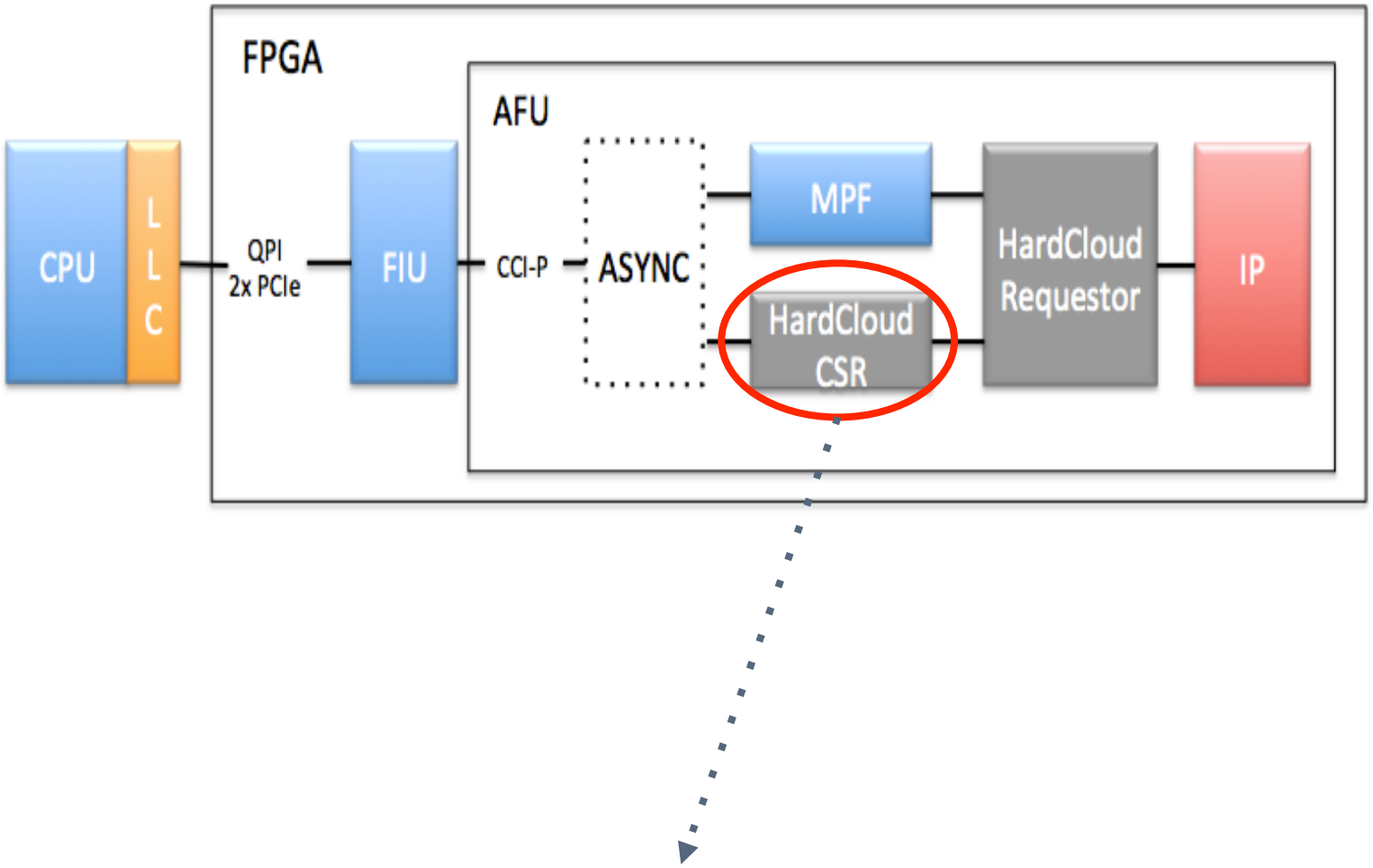


The hardware behind…

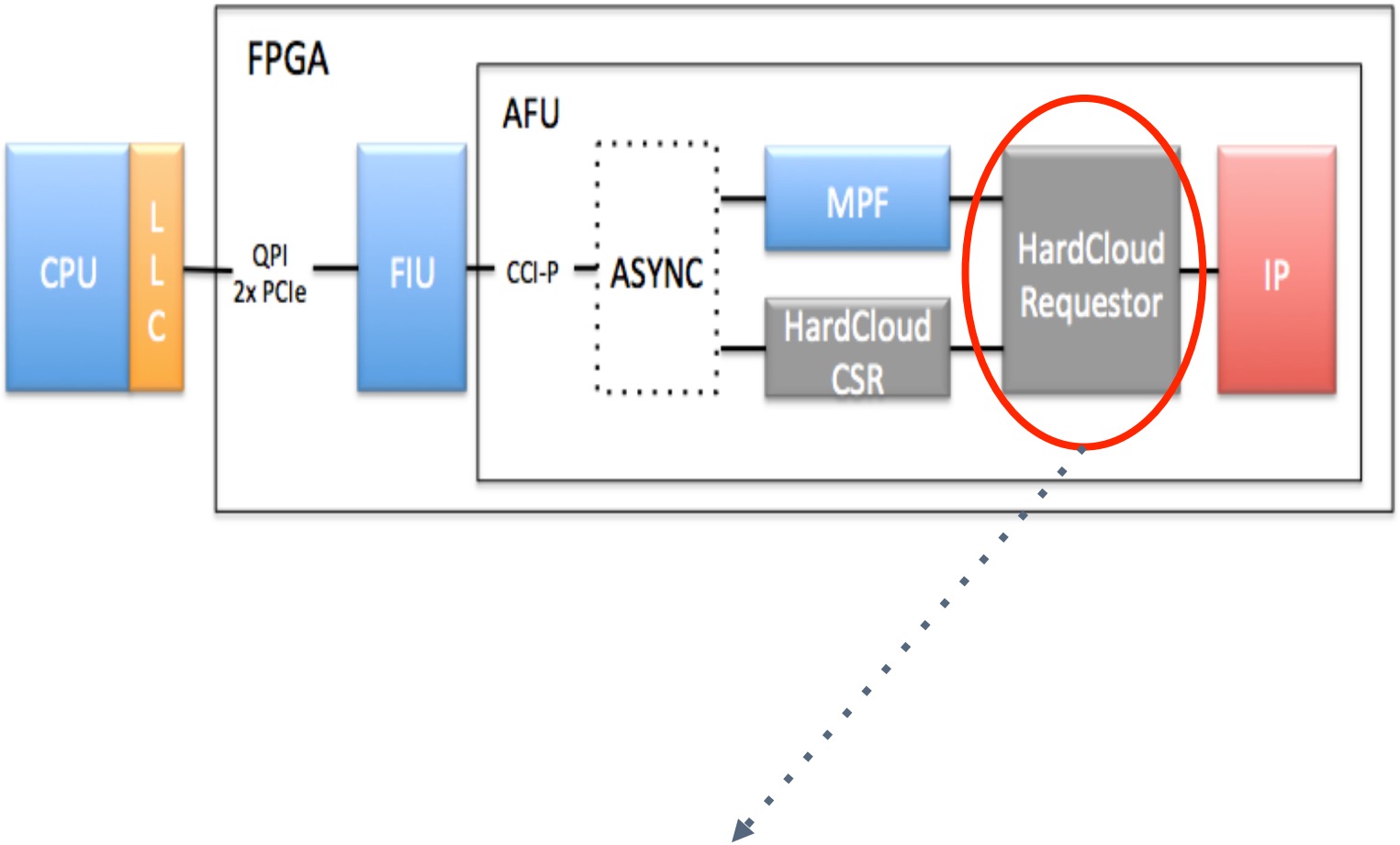

FPGA
AFU
CPU
L
L
C
QPI
2x PCIe
FIU
CCI-P
ASYNC
MPF
HardCloud
CSR
HardCloud
Requestor
IP
FPGA
FIU
1
cache
QPI
VL0
VH0
VH1
VC steering
CCI-P
PCIe 0
PCIe 1



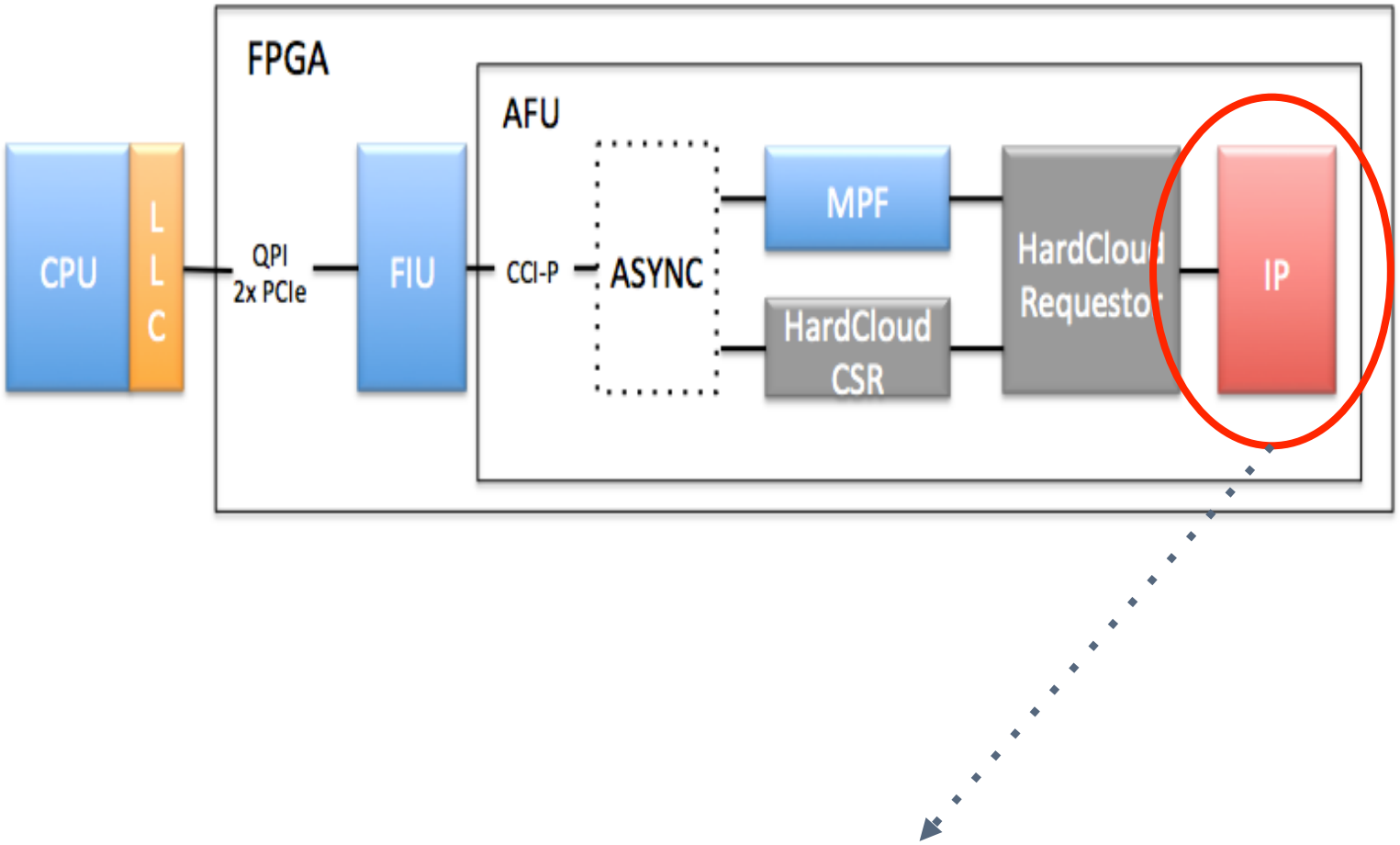
- VTP (Virtual to Physical): Read and write from virtual addresses
- ROB (Reorder Buffer): Return read responses in request order



- Manages communication with the host



- Controls the flow of data to/from the shared memory


FPGA
AFU
CPU
L
L
C
QPI
2x PCIe
FIU
CCI-P
ASYNC
MPF
HardCloud
CSR
HardCloud
Requesto
IP

- Your IP-core

How generic is HardCloud?

It took us 2 months to do Xilinx AWS....

```
/* some code */

#pragma omp target device(AWS:F1) map(to: X) map(from: Y)
#pragma omp parallel for use(hrw) module(loopback) check
for (int i = 0; i < elements_size; i++) {
  Y[i] = X[i];
}

/* some code */
```

www.hardcloud.org

Supported by:

Xilinx

Could it be expanded for many cloud-based FPGAs?

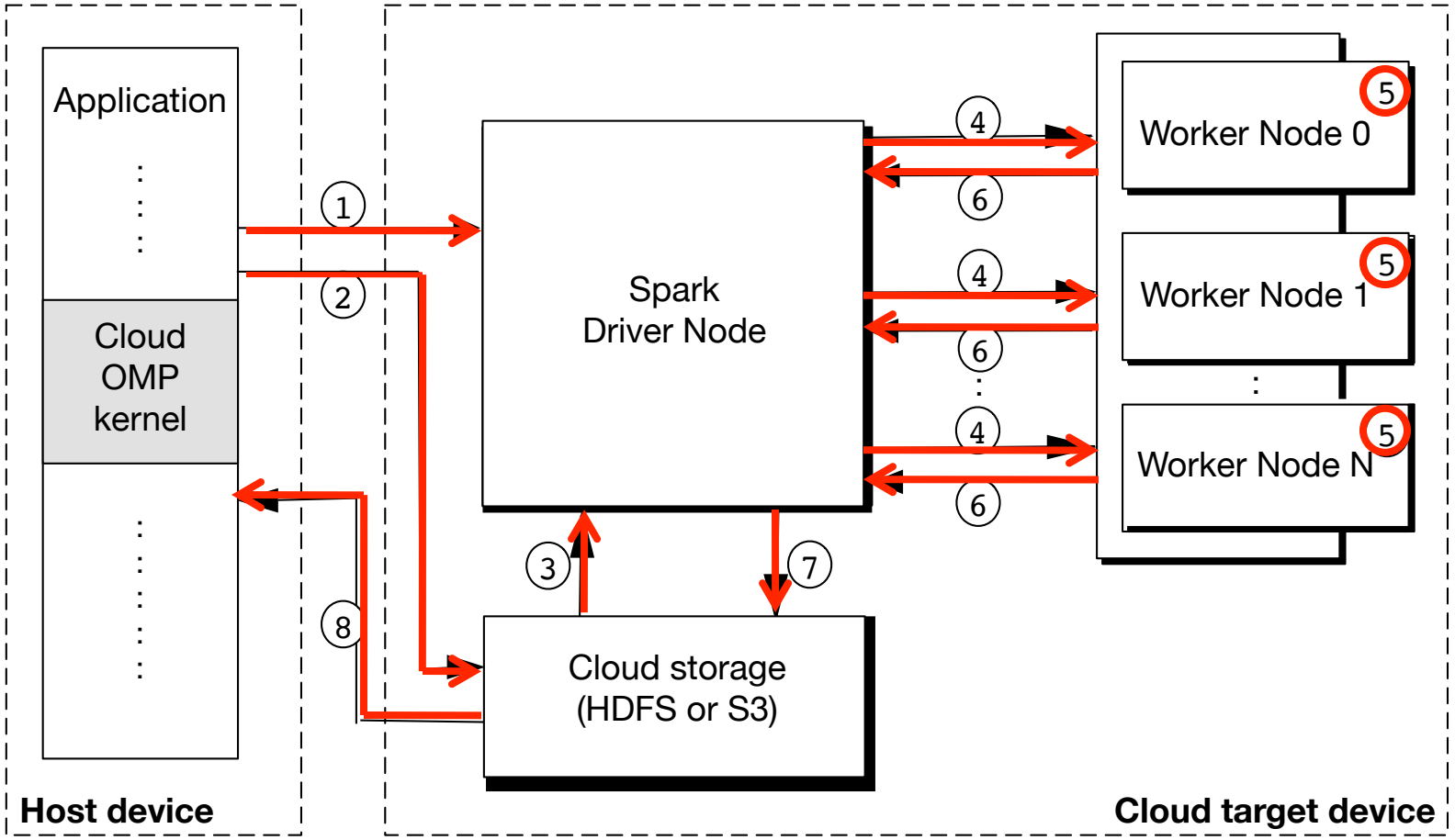
OpenMP + Cloud = **OmpCloud** (ACM TACO 2018)

Programming the Cloud with OpenMP

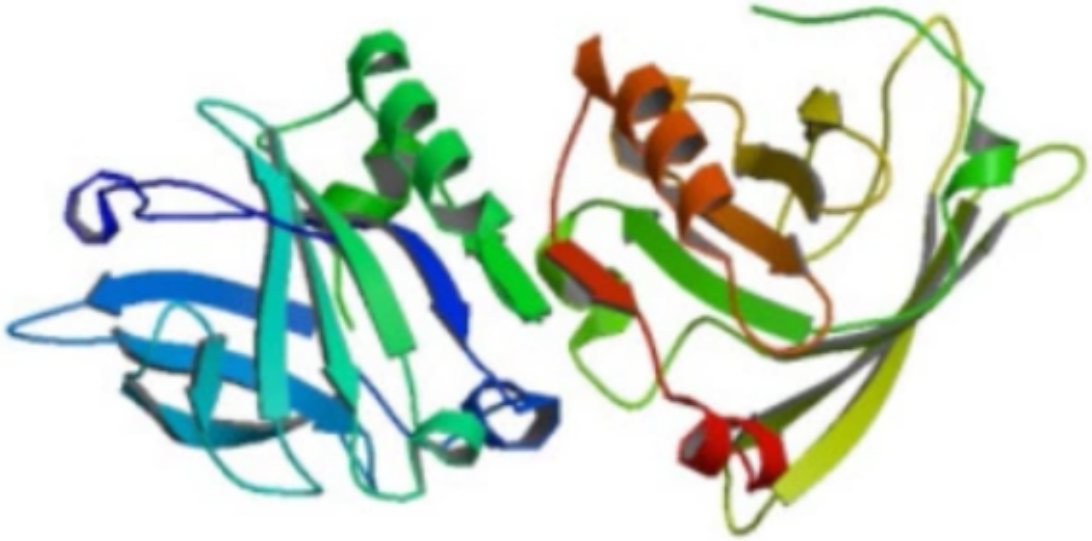
- Cloud is just another accelerator available in your computer
- Transformed into MapReduce Spark job by our custom LLVM compiler

```
int MatMul(float *A, float *B, float *C) {
  #pragma omp target device(AWS | Azure)
  #pragma omp target map(to: A[:N*N], B[:N*N]) map(from: C[:N*N])
  #pragma omp parallel for
  for(int i=0; i < N; ++i)
    for(int j = 0; j < N; ++j)
      C[i * N + j] = 0;
      for(int k = 0; k < N; ++k)
        C[i * N + j] += A[i * N + k] * B[k * N + j];
  return 0;
}
```

OmpCloud Workflow



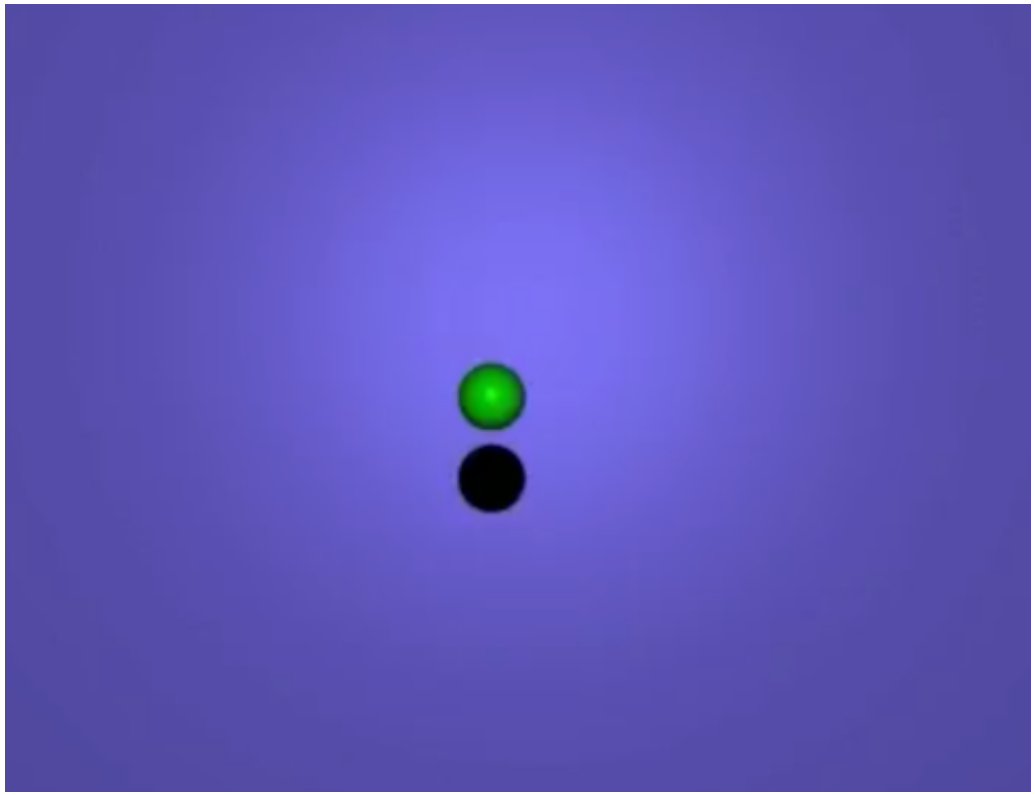
OmpCloud in Molecular Dynamics (ACM TACO 2018)



Single core (6h)
Microsoft Azure (5min)

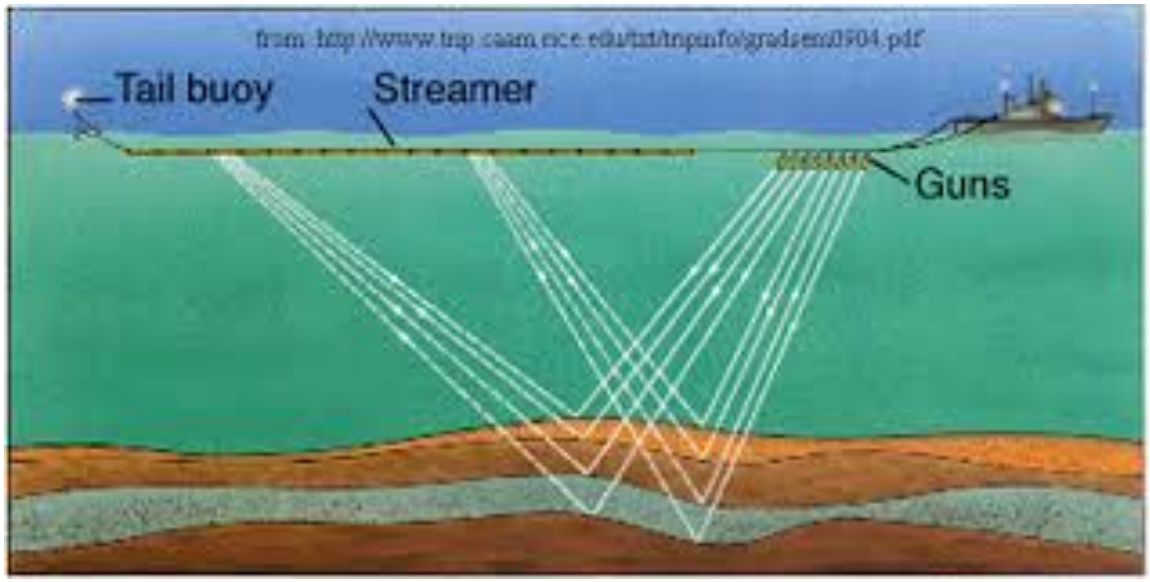
Supported by:
Microsoft Azure
Amazon AWS

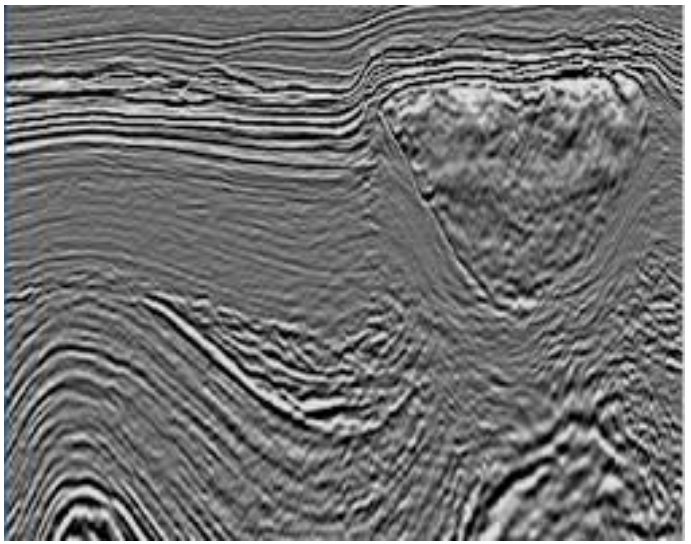
OmpCloud in Computer Graphics
(WAMCA 2018)



Supported by:
Microsoft Azure
Amazon AWS

Motivation: Seismic Imaging (RTM and FWI Problems)





Gracias!
Thanks!
Obrigado!

Any questions ?

Contact: guido@ic.unicamp.br

UNICAMP

eScience UNICAMP

FAPESP