A Scalable Complex Event Processing Framework for Combination of SQL-based Continuous Queries and C/C++ Functions

Takashi Takenaka, Masamichi Takagi
Hiroaki Inoue
NEC Corporation

August 30, 2012

Today's Key Message

Synthesizing

SQL with C

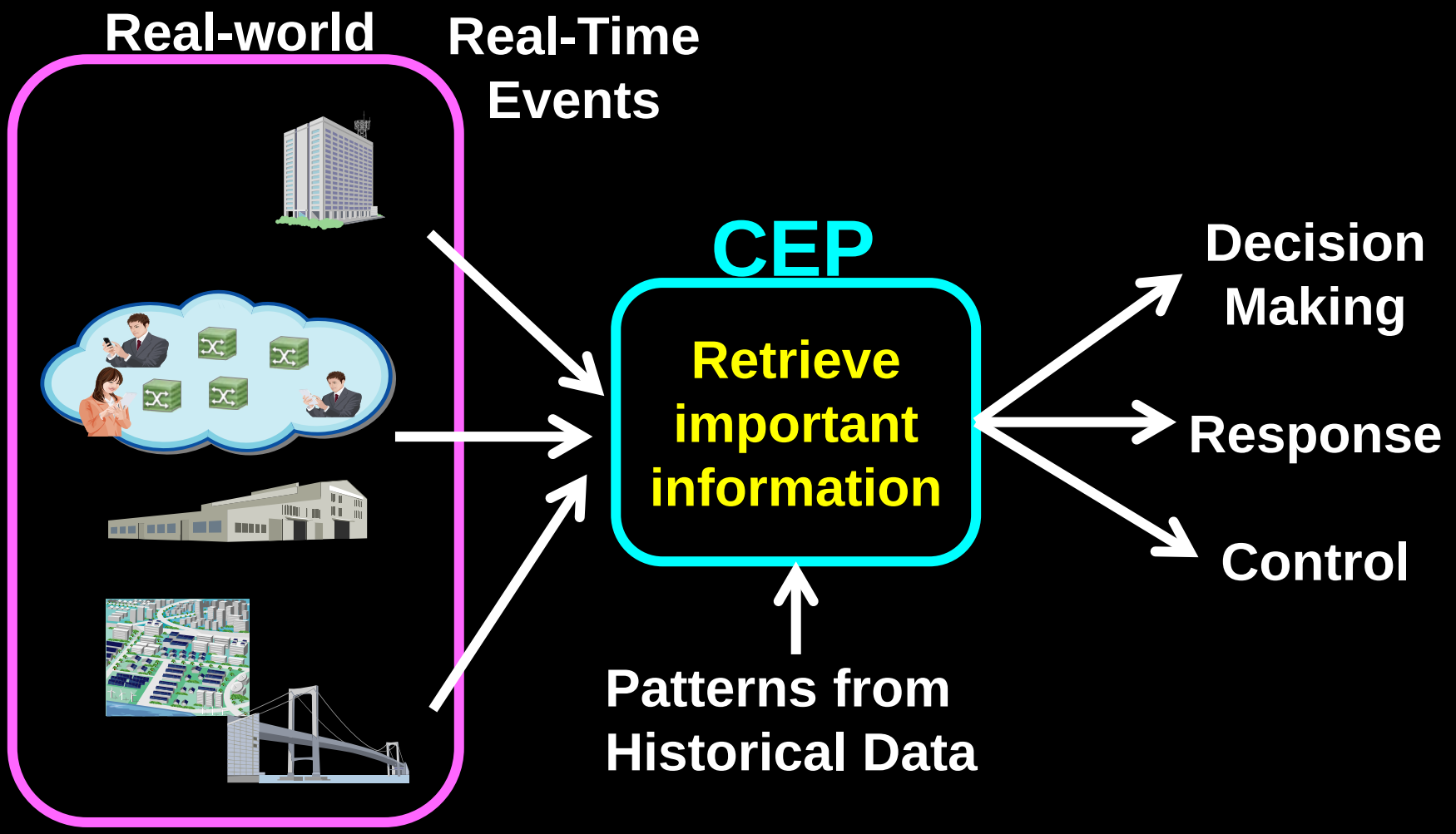
to event processing

on FPGAs

Outline

- Background
- Our work
- Evaluation
- Conclusion

Complex Event Processing(CEP)



IDC: $10B (software) by 2014

CEP Applications

Financial Algorithmic Trading

Active diagnostics of facilities

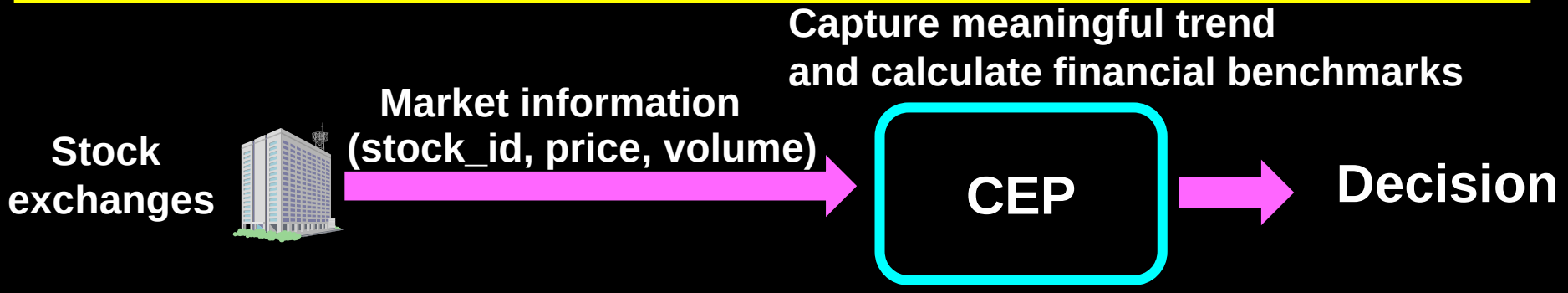
Fraud detection: web commerce, credit card

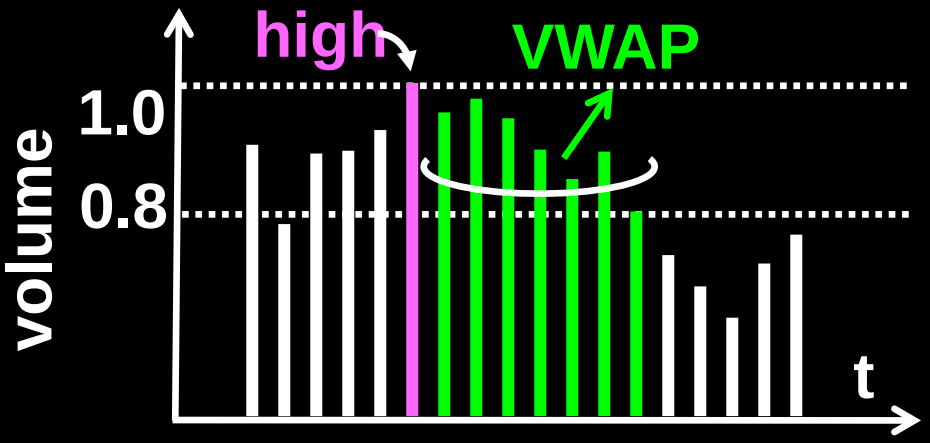
Compliance reporting and monitoring

Track and Trace: Patients, packages

…

CEP in financial treading

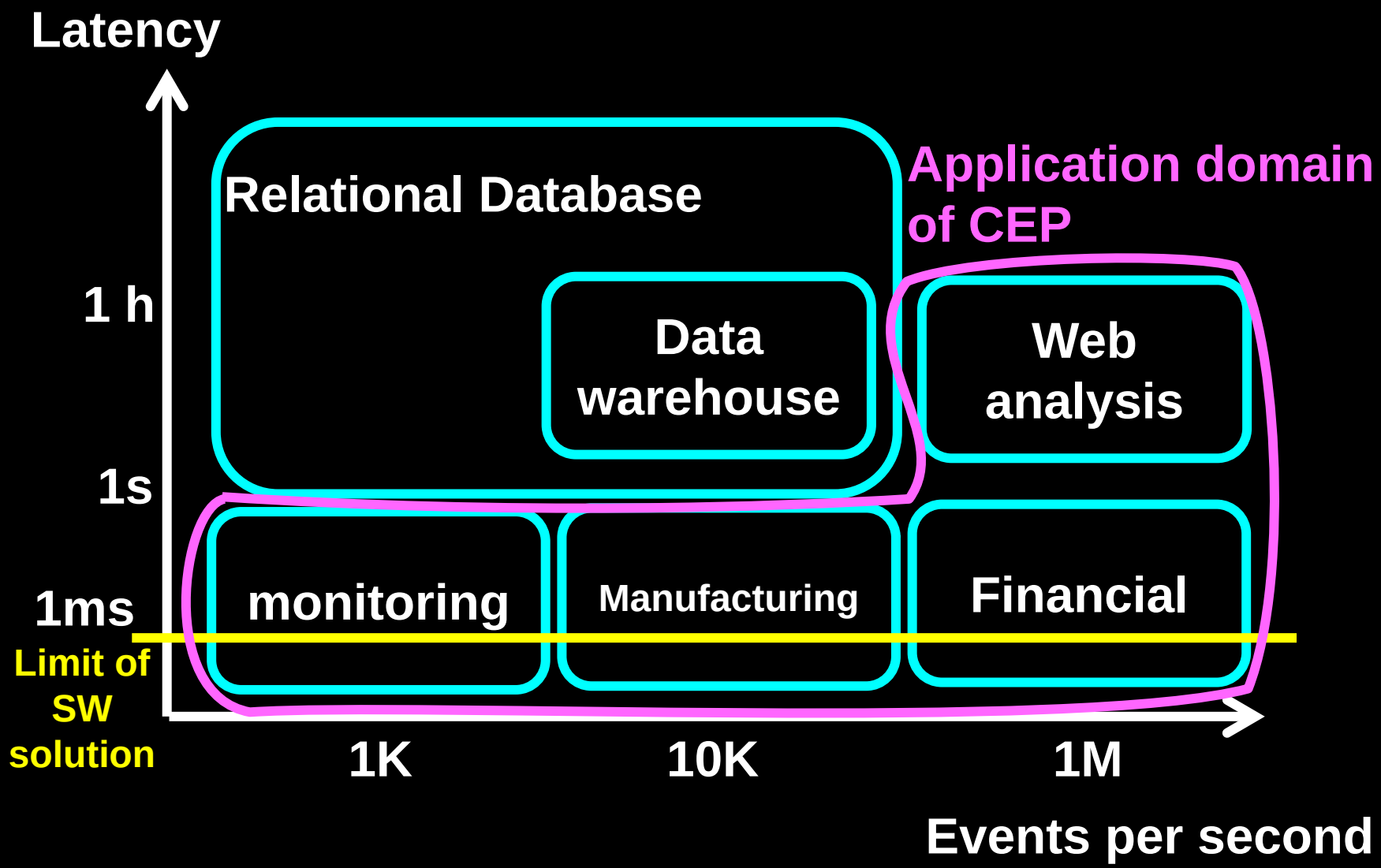




Capture a trend:

volume is starting high then it reaches 80%, then **Calculate *"volume-weighted average of price(VWAP)"* during the period.**

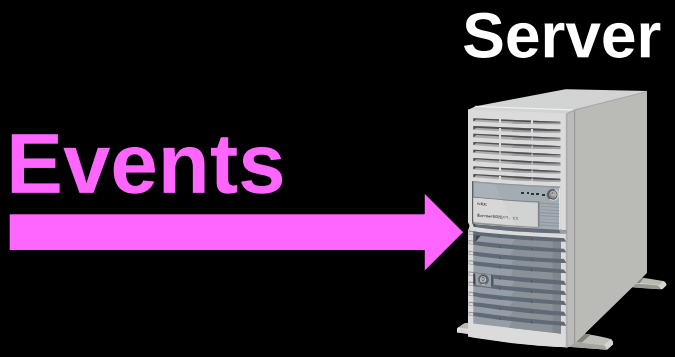
Performance requirement



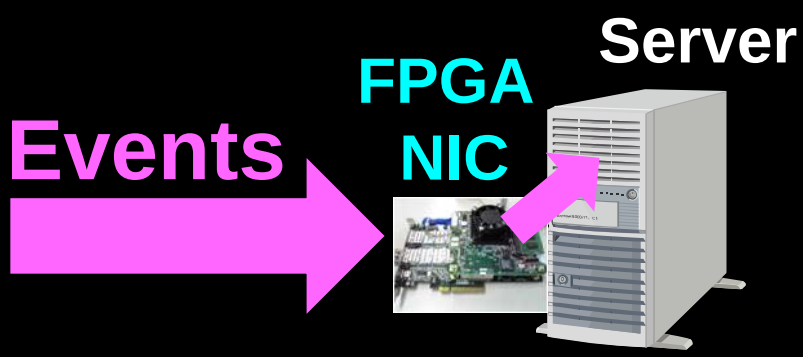
(Torsten Grabs, “Introduction to Microsoft SQL Server 2008 R2 StreamInsight”)

Hardware CEP

Software CEP



Hardware CEP [17,21]



	Software CEP	Hardware CEP
Performance	Low ☹ (0.12Gbps) [12]	High ☺ (20Gbps)
Application range	Broad ☺	Limited ☹

Major HW CEP Requirements

- Programmability
- Scalability

Programmability

Software CEPs

Sybase CCL

Oracle CQL

IBM SPL

EsperTech EPL

StreamBase StreamSQL

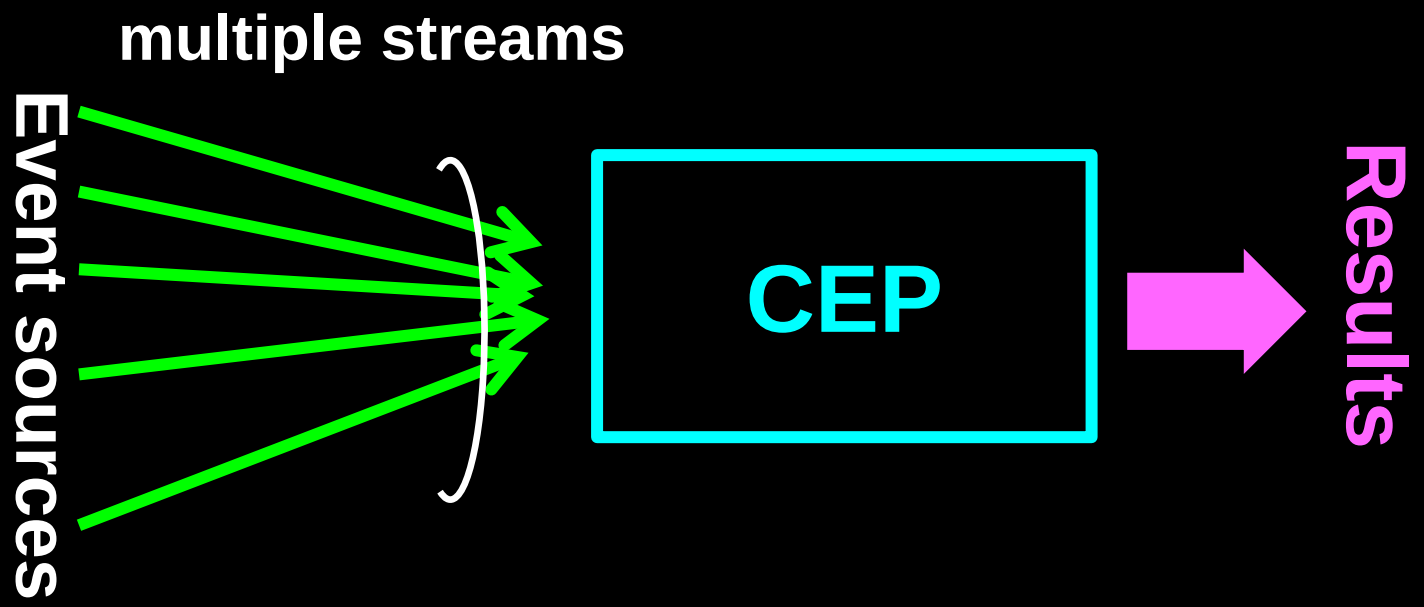
... etc.

SQL

+

User-defined functions (C/Java)

Scalability



Bridge monitoring

100

sensor nodes

Factory monitoring

1k

sensor nodes

Financial Trading

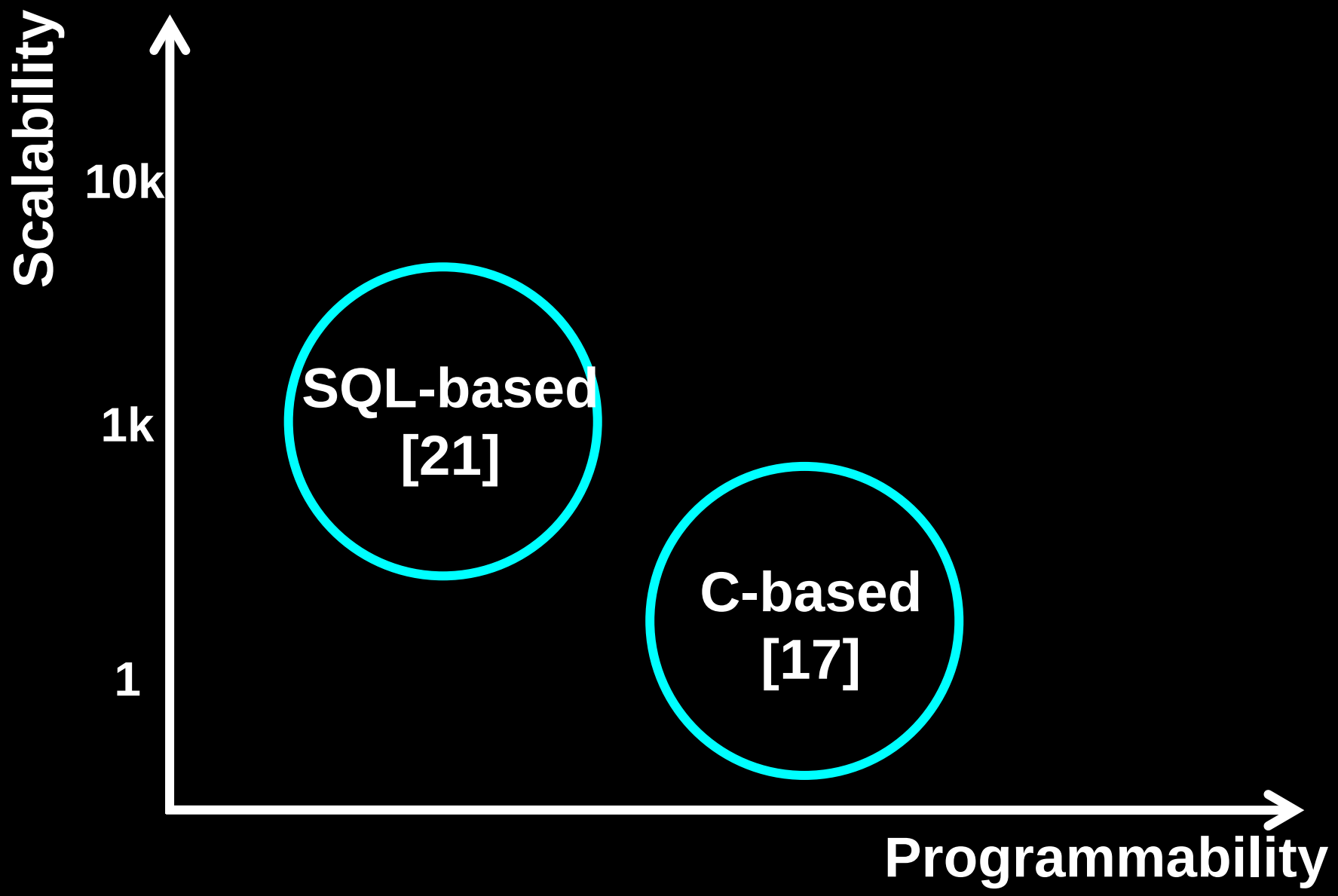
10k

Listed issues in NYSE

Existing HW CEPs

	Woods [21]	Inoue[17]
Language	SQL-based	C-based
Performance	Good (1Gbps)	Good (20Gbps)
Programmability	No user functions	No SQL interface
Scalability	Limited (< 1K)	No

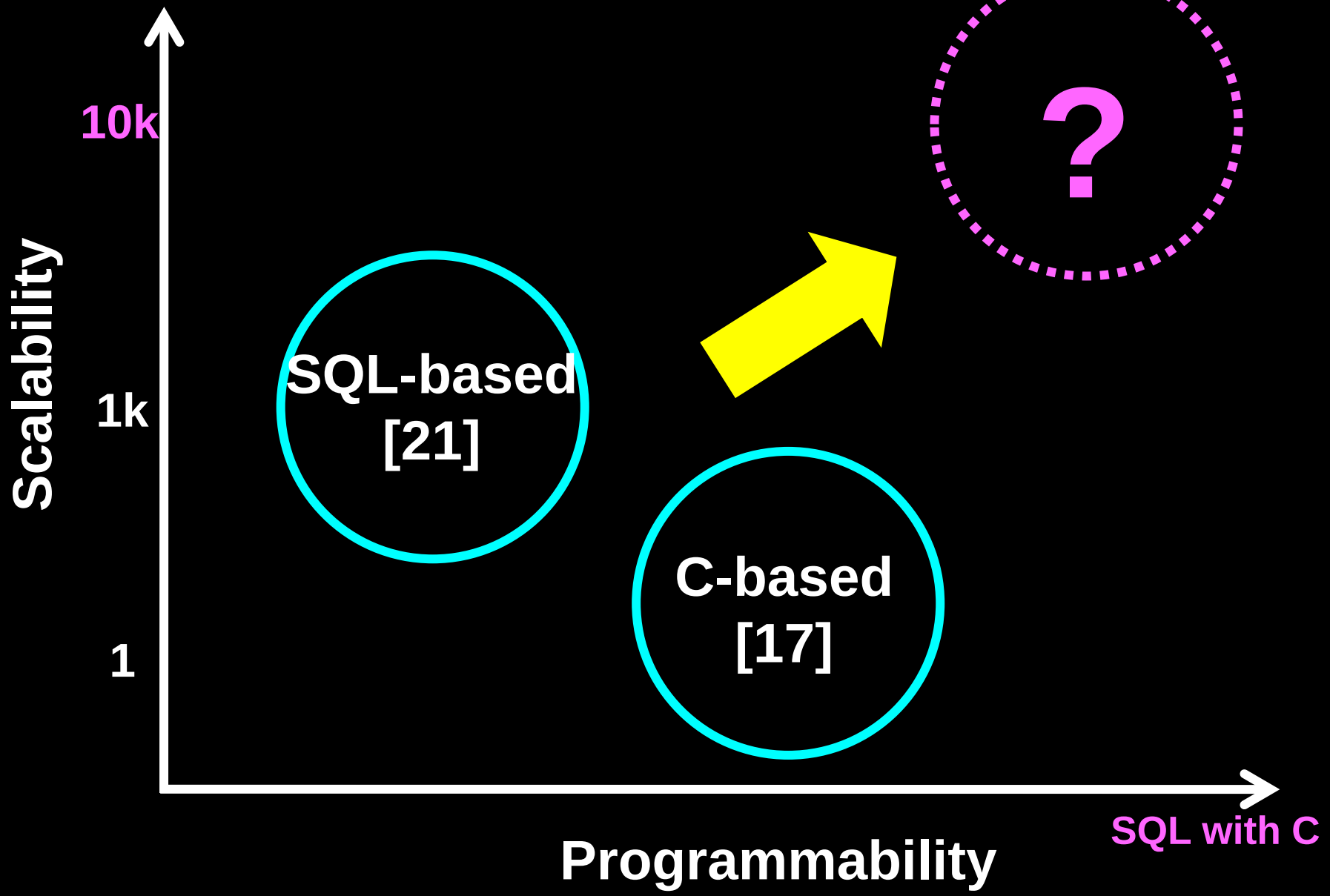
Summary of background



Outline

- Background
- Our work
- Evaluation
- Conclusion

Our Goal


10k
1k
1
Scalability
SQL-based
[21]
C-based
[17]
?
Programmability
SQL with C

Two Technical Points

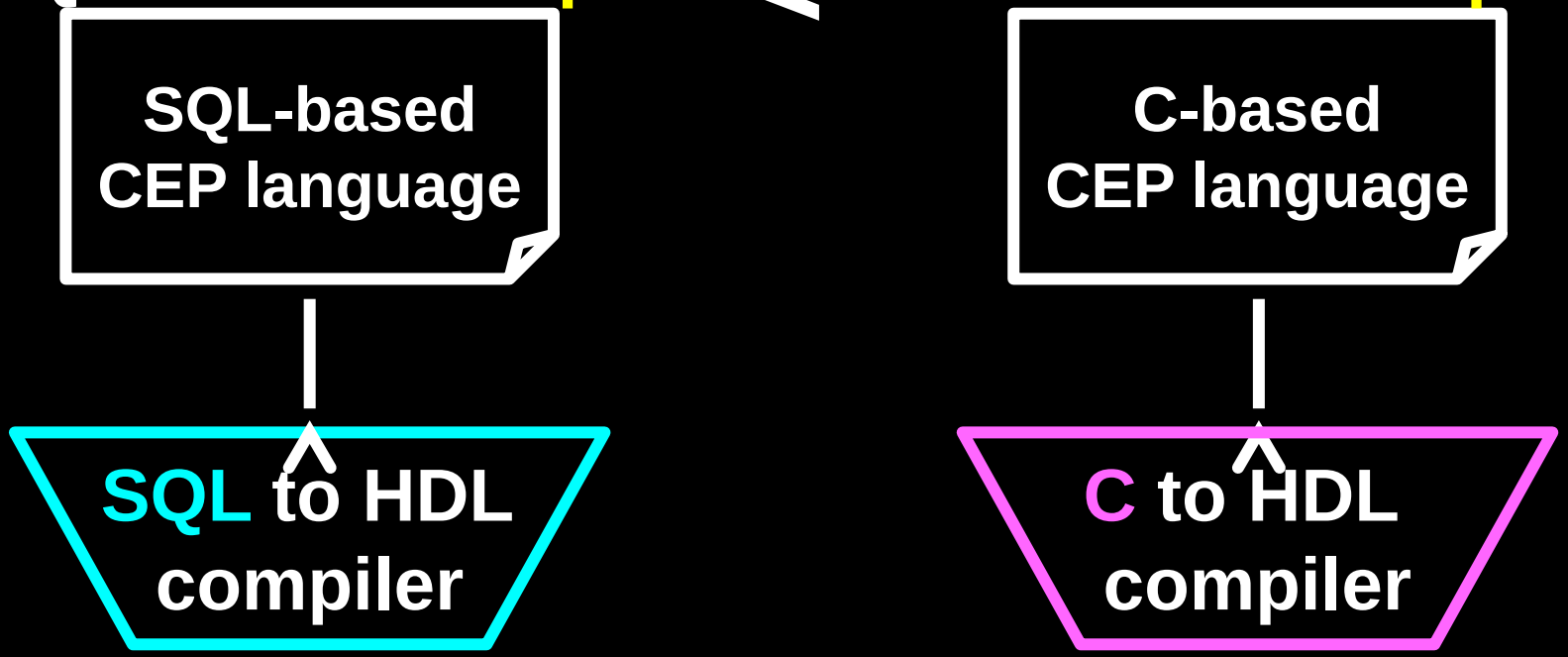
1. SQL interface on the top of C-to-HDL compiler

2. Scalable architecture for multiple streams

Performance came from …

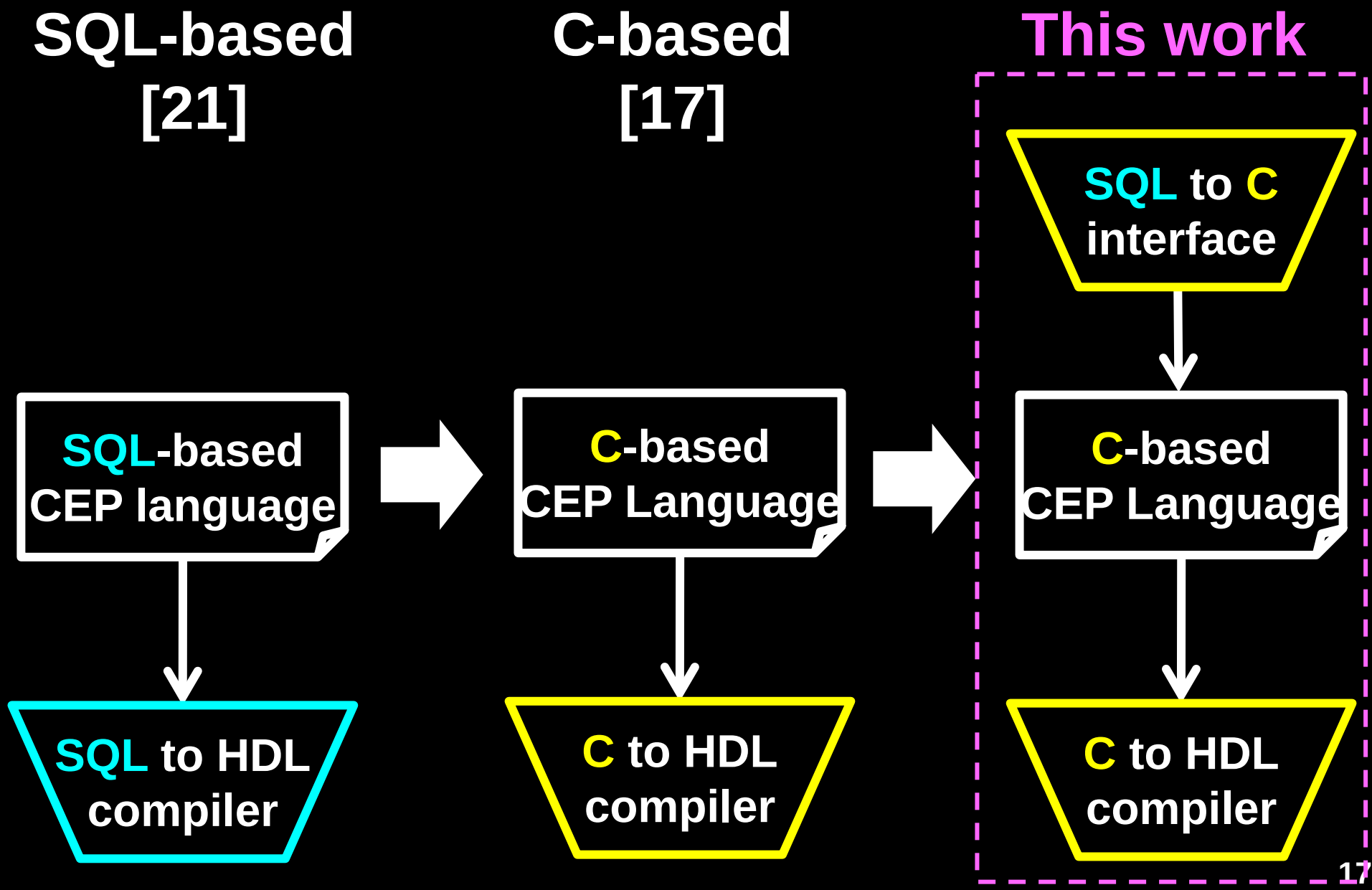
Current "C-to-HDL compiler" technology generates well-optimized circuits

SQL-based: 1Gbps **<** **C-based: 20Gbps**



No industry tool **Use industry tools**

Basic Strategy



SQL's primitives

Selection

```
SELECT *
WHERE volume > 3200
From Stock
```

Window, Aggregation

```
SELECT stock_id, SUM(volume) AS sum
From Stock [ROWS 4 PRECEDING]
```

User functions

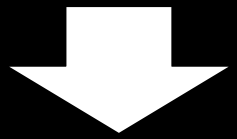
```
SELECT stock_id, calc_vwap() AS vwap
From Stock [ROWS 4 PRECEDING]
```

Translation rule: selection

Finding events whose volume is greater than 3,200.

SQL

```
SELECT *
WHERE volume > 3200
From Stock
```



C

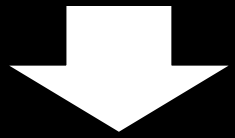
```
while(true) {
  entry = event_fifo_in.read();
  if ( entry.volume > 3200 ) {
    event_fifo_out.write(entry);
  }
}
```

Translation rule: window

Calculating sum of volume of latest 4 events.

SQL

```
SELECT stock_id, SUM(volume) AS sum
From Stock [ROWS 4 PRECEDING]
```



C

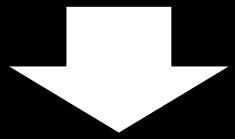
```
#define WINDOW_SIZE 4
evin_t win[WINDOW_SIZE];
while(true) {
  for(i=1;i<WINDOW_SIZE;i++)
    win[i] = win[i-1];
  win[0] = event_fifo_in.read();
  result.sum = calc_sum_volume(win);
  event_fifo_out.write(result);
}
```

Translation rule: User-function

Calculating "volume-weighted average of price" of latest 4 events.

SQL

```
SELECT stock_id, calc_vwap() AS vwap
From Stock [ROWS 4 PRECEDING]
```



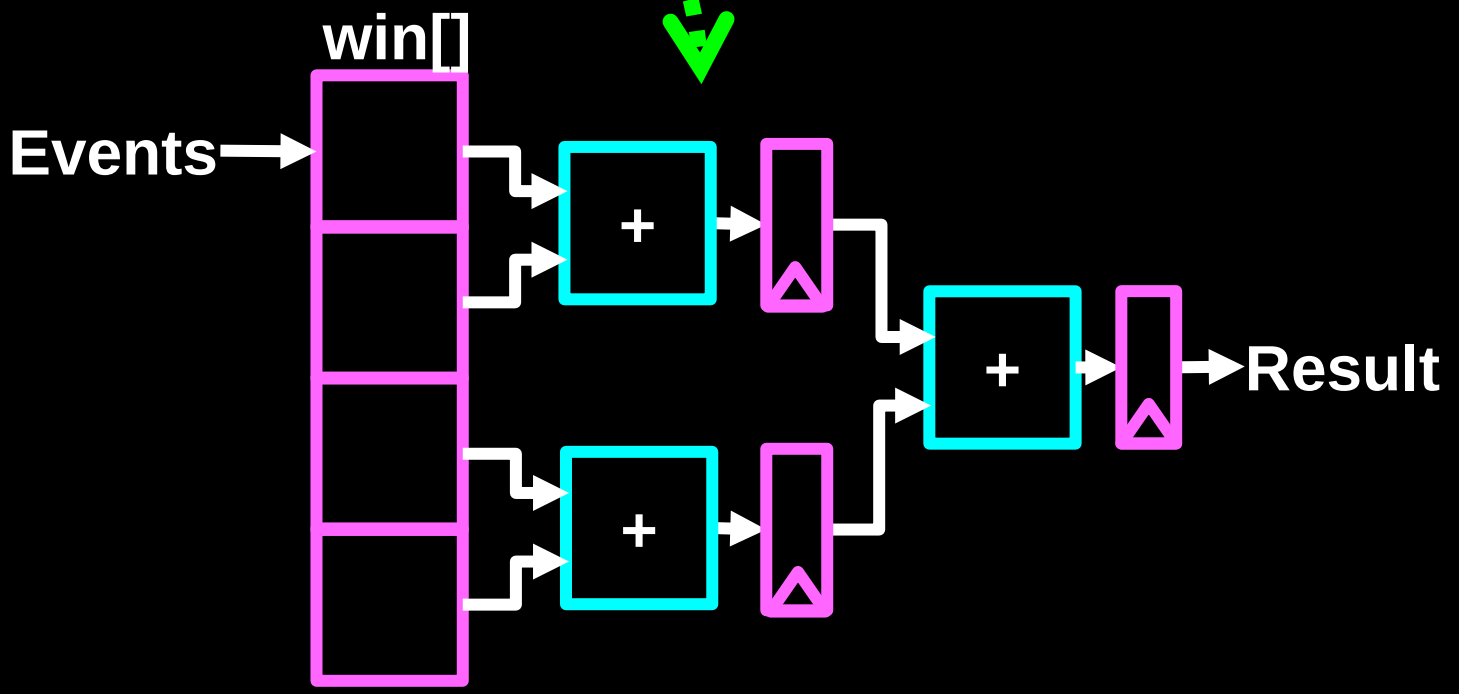
C

```
while(true) {
   ...
   result.vwap = calc_vwap(win);
   event_fifo_out.write(result);
}
```

After translated to C, …

C-to-HDL compilers generate well-parallelized and pipelined circuits.

```
for(i=0;i<WINDOW_SIZE;i++)
  sum += win[i];
result = sum;
```



Summary of SQL-to-C interface

	SQL-based [21]	C-based [17]	Ours
Selection	Yes	No	Yes
Window	Yes	No	Yes
Matching	Yes	Yes	Yes
Aggregation	Limited	Yes	Yes
User function	No	Yes	Yes
Multiple streams	Limited	No	?

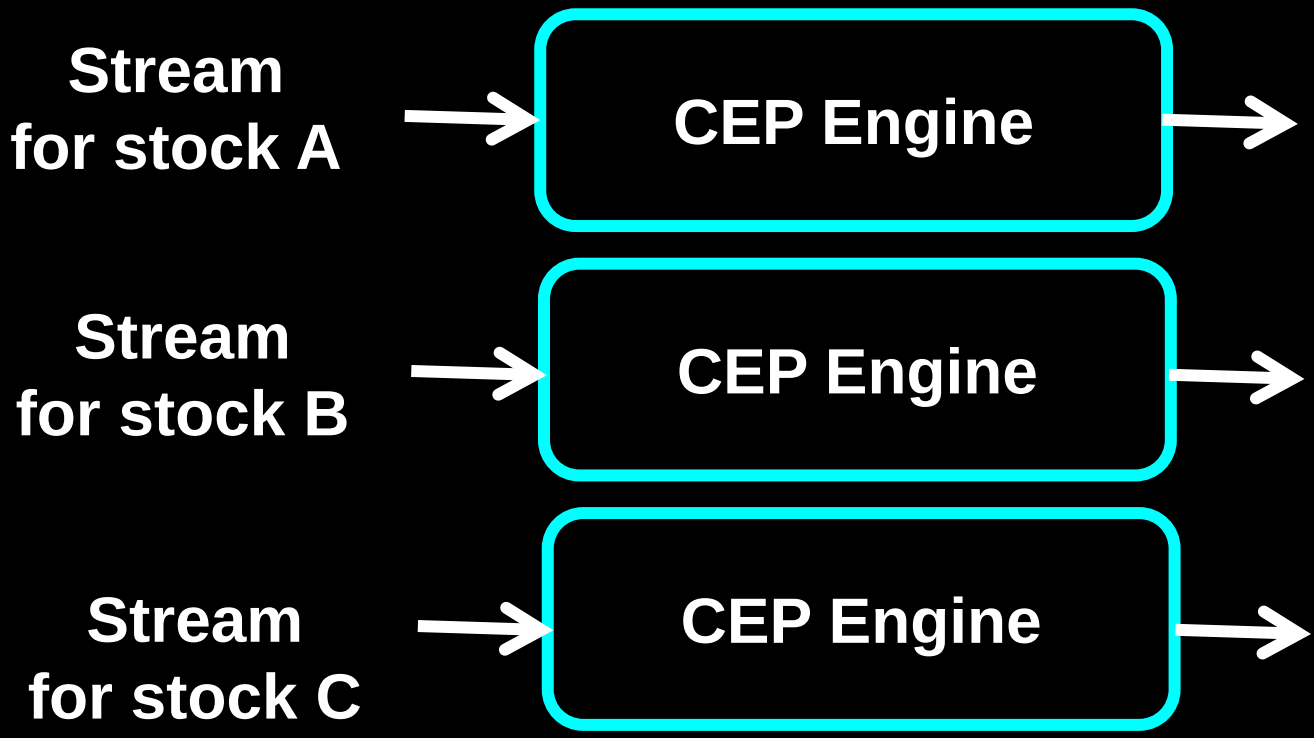
Two Technical Points

1. SQL interface on the top of a C-to-HDL compiler

2. Scalable architecture for multiple streams

Multiple streams

CEP is required to receive multiple streams and to perform event processing.

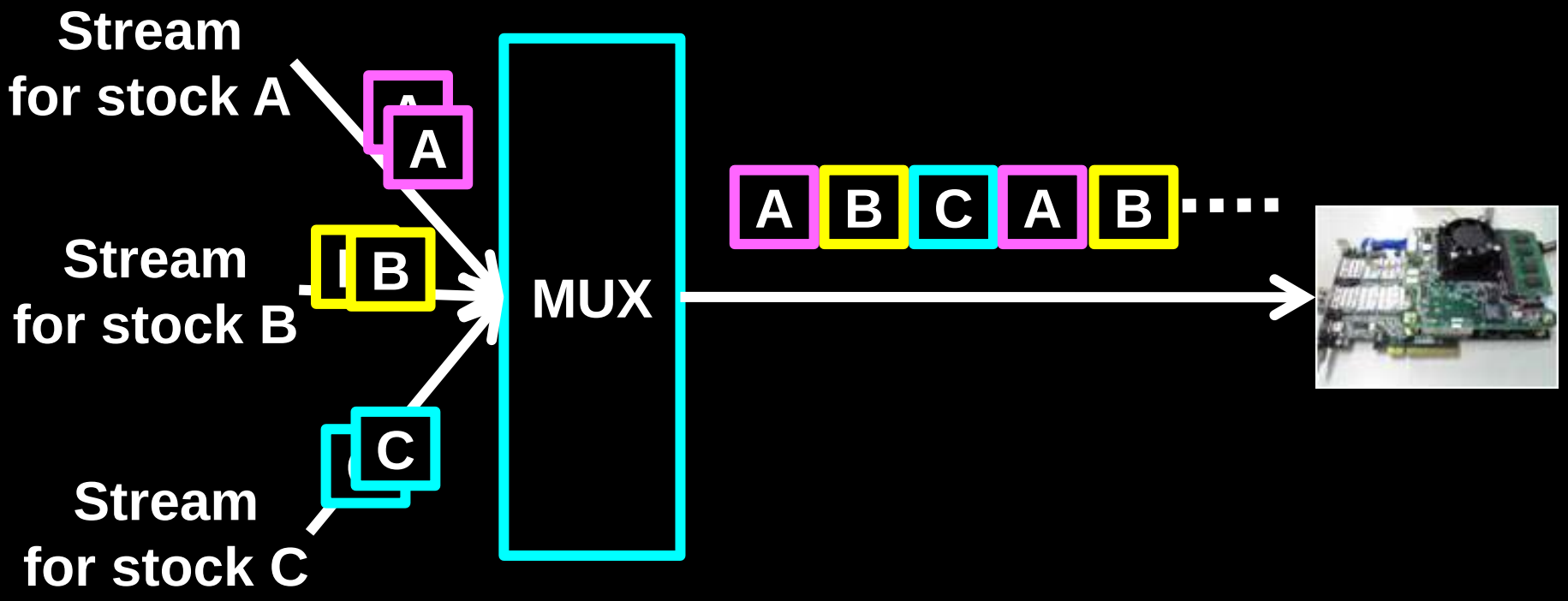


Naive replication is not applicable for > 100 streams

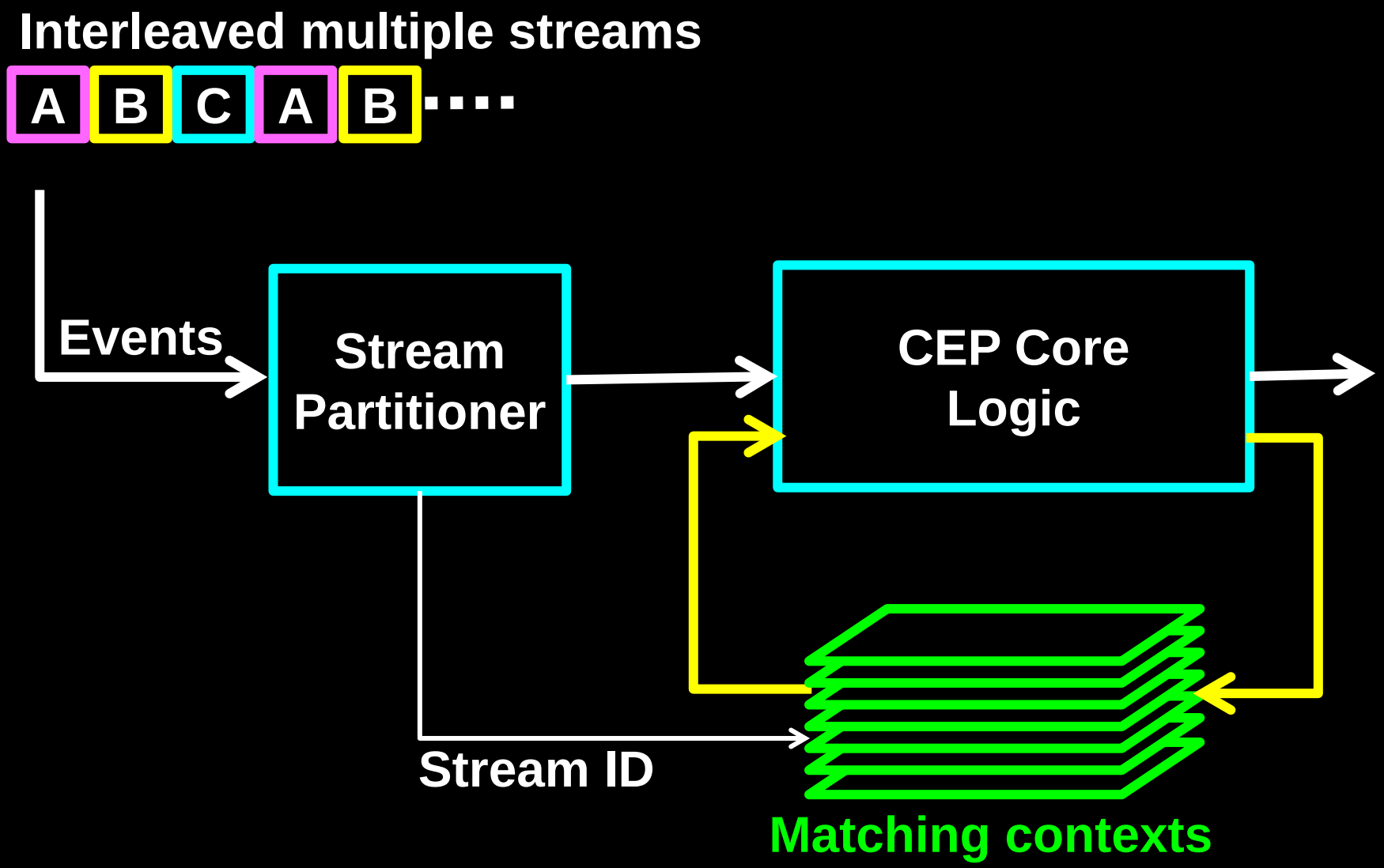
Observation

Multiple streams are usually interleaved into a stream on a high-speed link

→ an event arrives at a time



Interleaved multiple-context architecture



Summary of our work

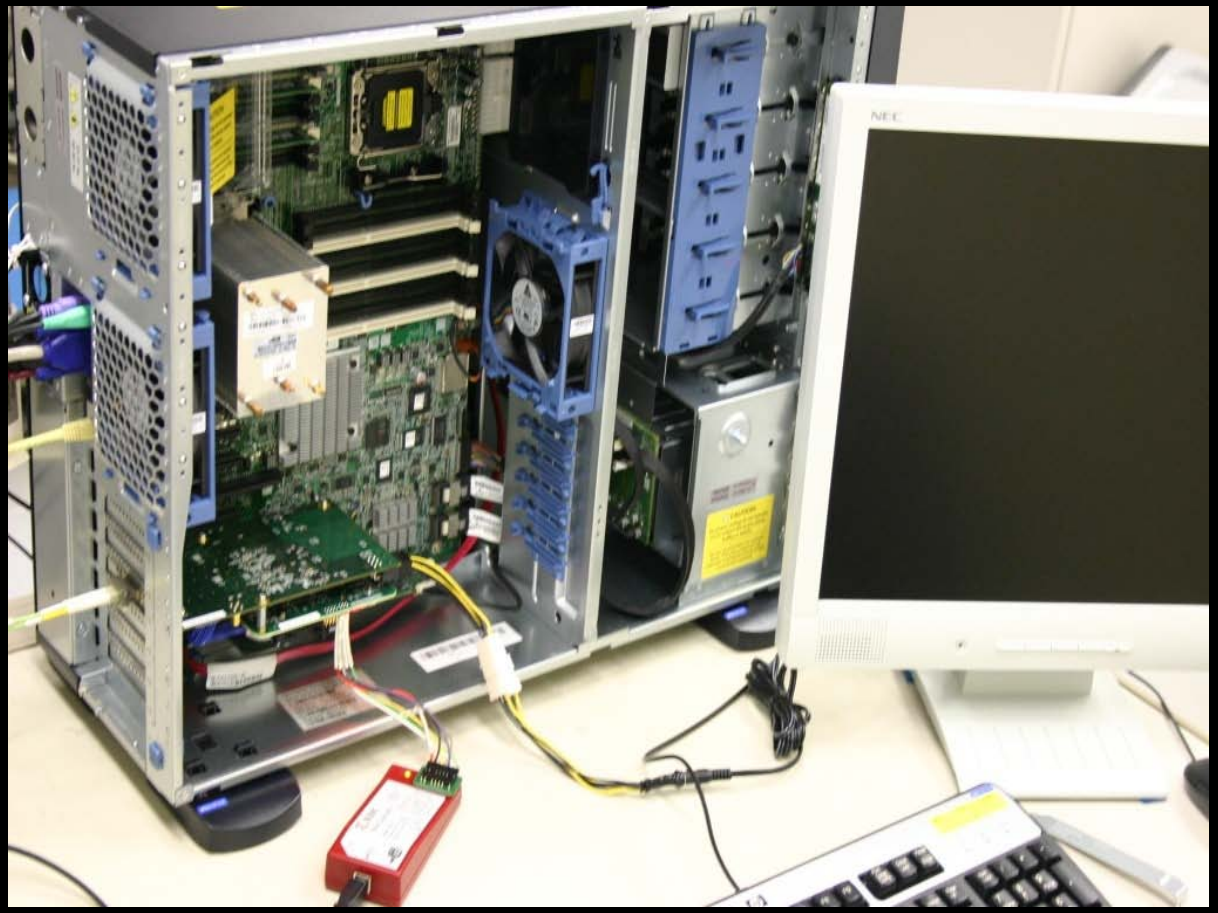
	SQL-based [21]	C-based [17]	Ours
Selection	Yes	No	Yes
Window	Yes	No	Yes
Matching	Yes	Yes	Yes
Aggregation	Limited	Yes	Yes
User function	No	Yes	Yes
Multiple streams	Limited	No	Yes

Outline

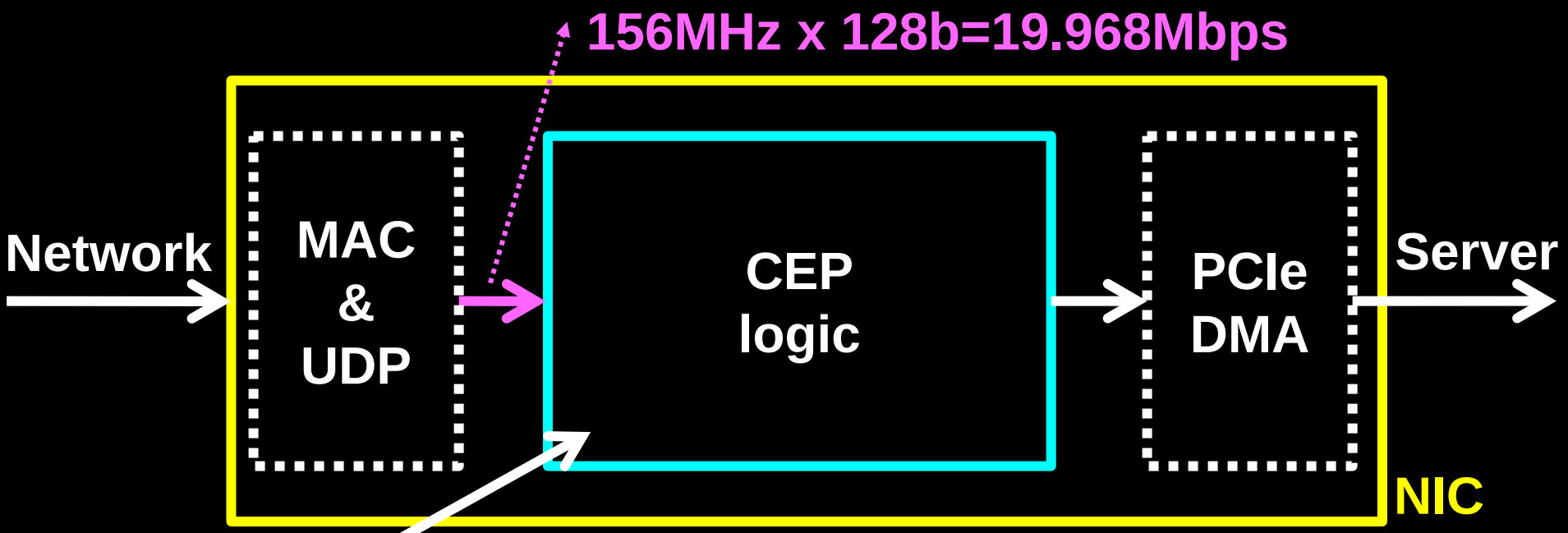
- Background
- Our work
- Evaluation
- Conclusion

Evaluation platform

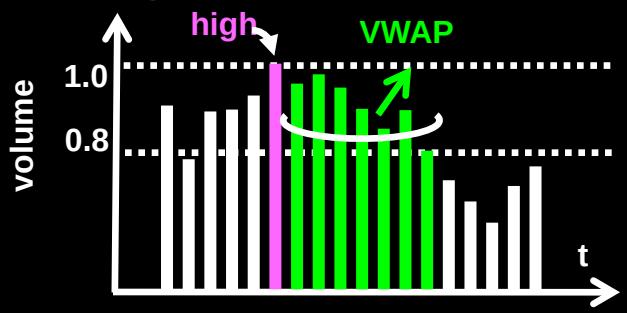
FPGA	Xilinx XC5VLX330T-2
CAD	NEC CyberWorkBench, Xilinx ISE 12.2



Setup



Query



Captures a trend where volume is starting high then it reaches 80%, then calculates "volume-weighted average of price(VWAP)" during the period.

Query

SQL

```
SELECT stock_id, vwap
FROM Stock
MATCH_RECOGNIZE (
  PARTITION BY stock_id
  MEASURES C.stock_id  AS stock_id
           C.vwap      AS vwap
  PATTERN (A B+ C)
  DEFINE A AS vol_high()
         B AS pri_stbl()
         C AS vol_down()
)
```

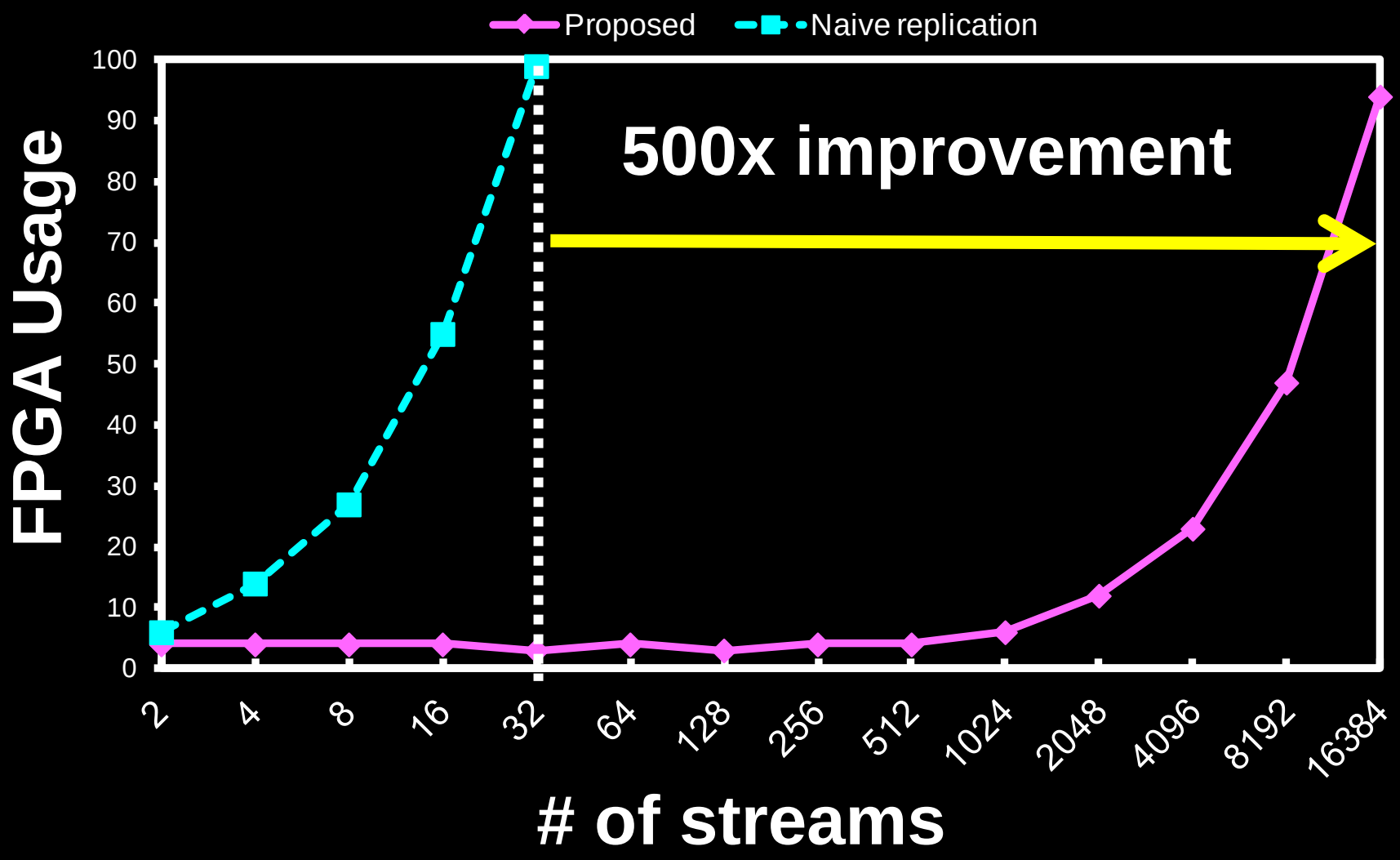
User functions

```
bool vol_high(evin_t ev, evarg_t &arg) {
  arg.stock_id = ev.stock_id;
  arg.volume = ev.volume
  arg.sum_volume = ev.volume;
  arg.sum_w_price  = ev.price * ev.volume;
  return ev.volume > 1000;
}
```

```
bool pri_stbl(evin_t ev, evarg_t &arg) {
  arg.sum_volume += ev.volume;
  arg.sum_w_price += ev.price * ev.volume
  arg.vwap=arg.sum_w_price/arg.sum_volume;
  return (ev.price > vwap);
}
```

```
bool vol_down(evin_t ev, evarg_t &arg) {
  return ev.volume < 0.8 * arg.volume;
}
```

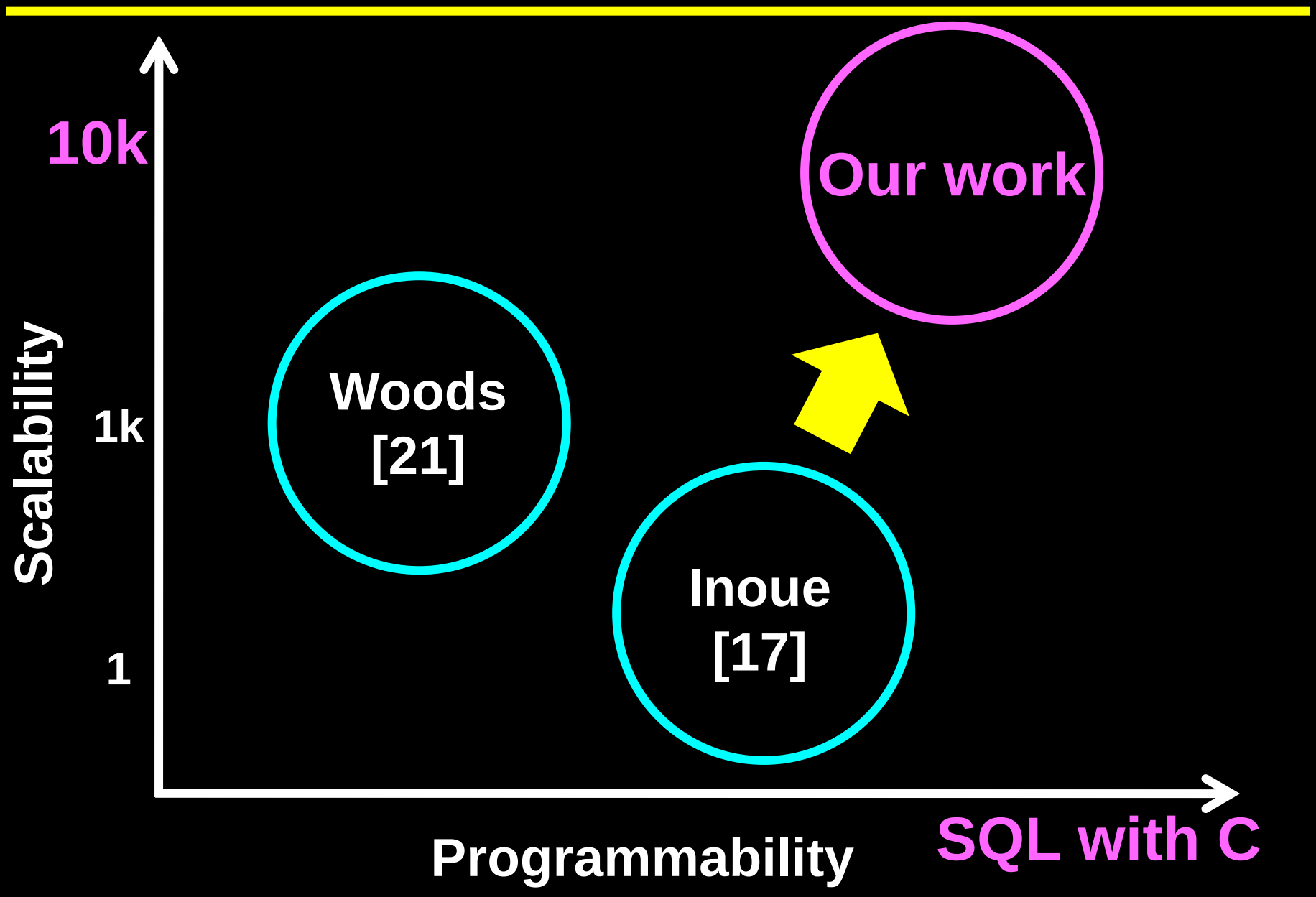
Scalability



FPGA Usage = max(block mem usage/block mem avail, slice usage/slice avail) * 100

Outline

- Background
- Our work
- Evaluation
- Conclusion


10k
1k
1
Scalability
Our work
Woods
[21]
Inoue
[17]
Programmability
SQL with C