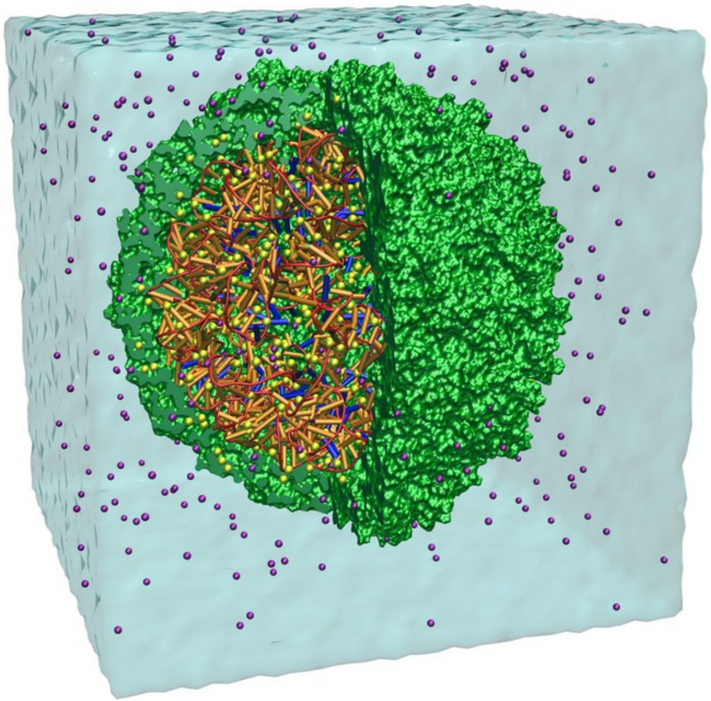
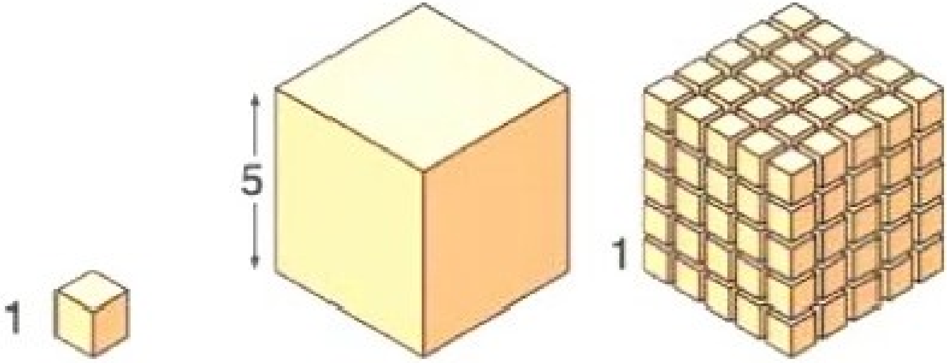
High Performance Computing in Web Browsers

CE Seminar WT14/15
Henning Lohse

High Performance Computing

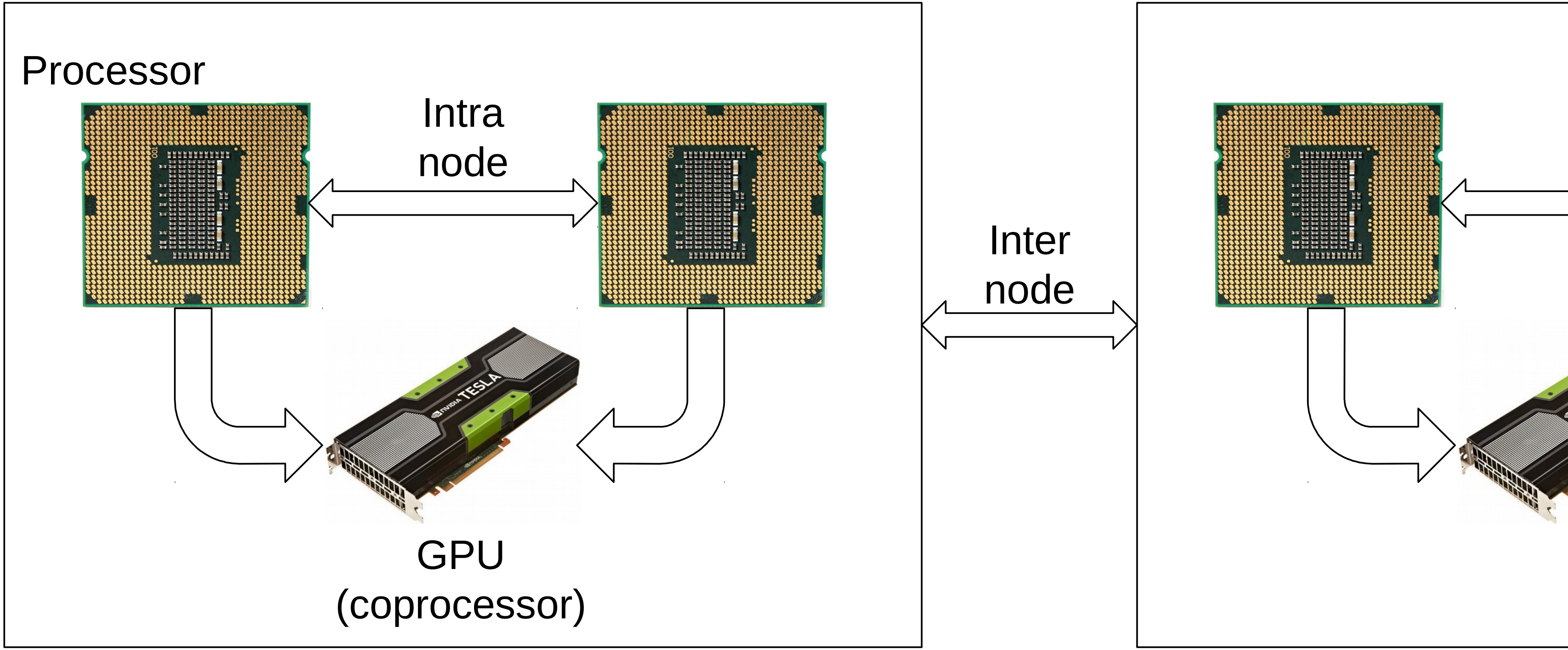


ks.uiuc.edu



voltagegate.scientopia.org

High Performance Computing



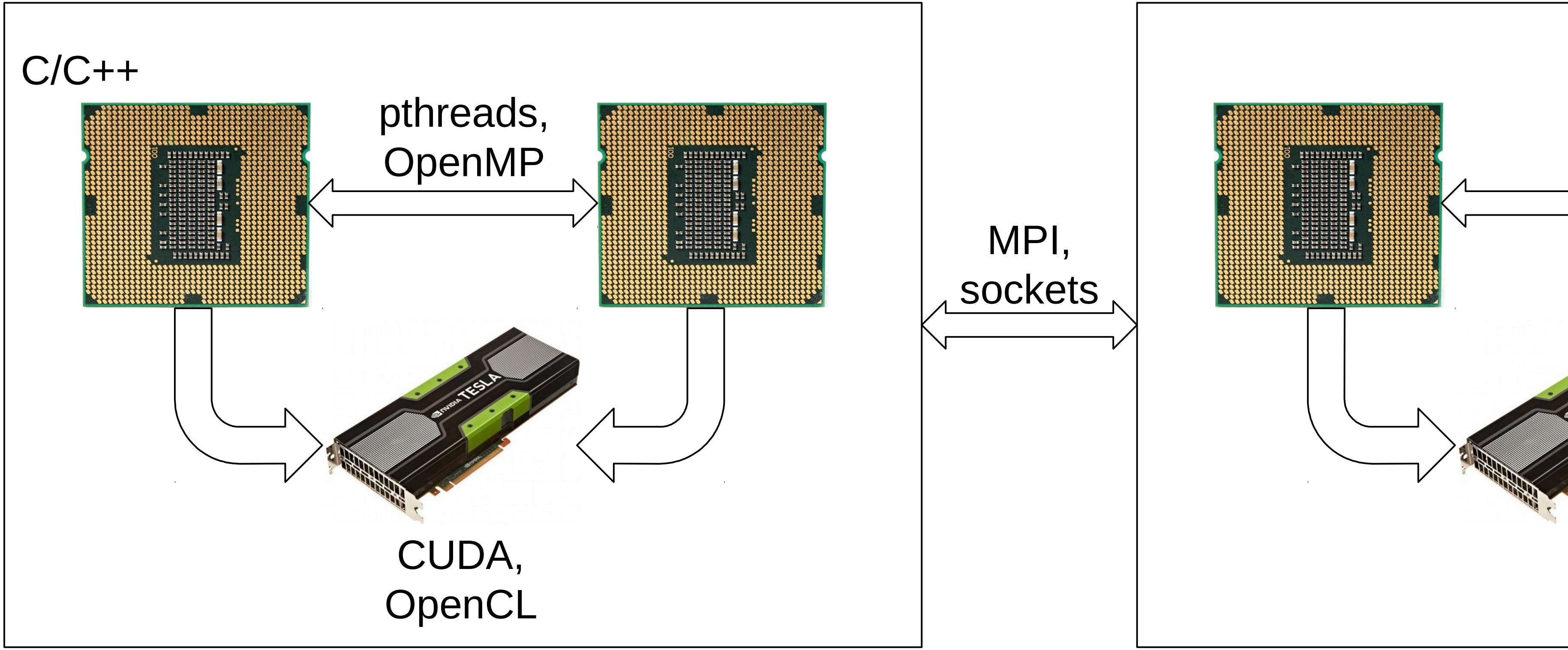
Processor

Intra node

Inter node

GPU (coprocessor)

High Performance Computing



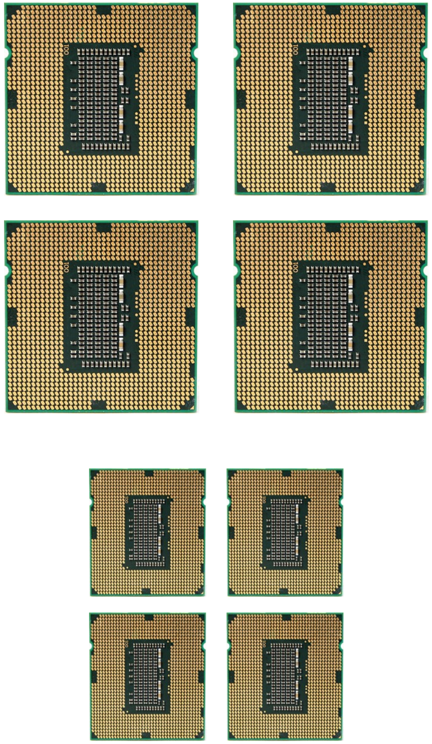
C/C++

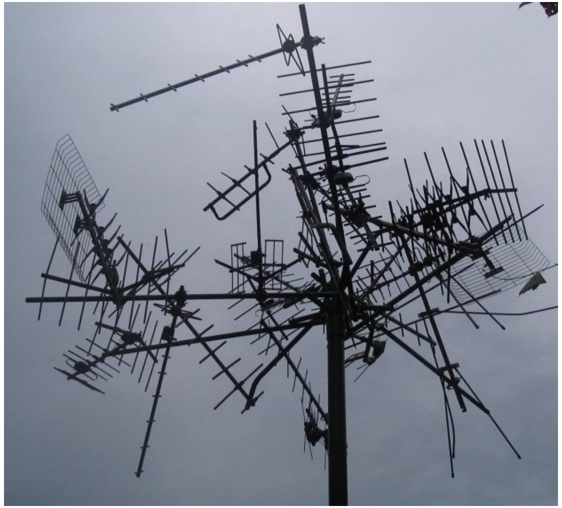
pthreads,
OpenMP

MPI,
sockets

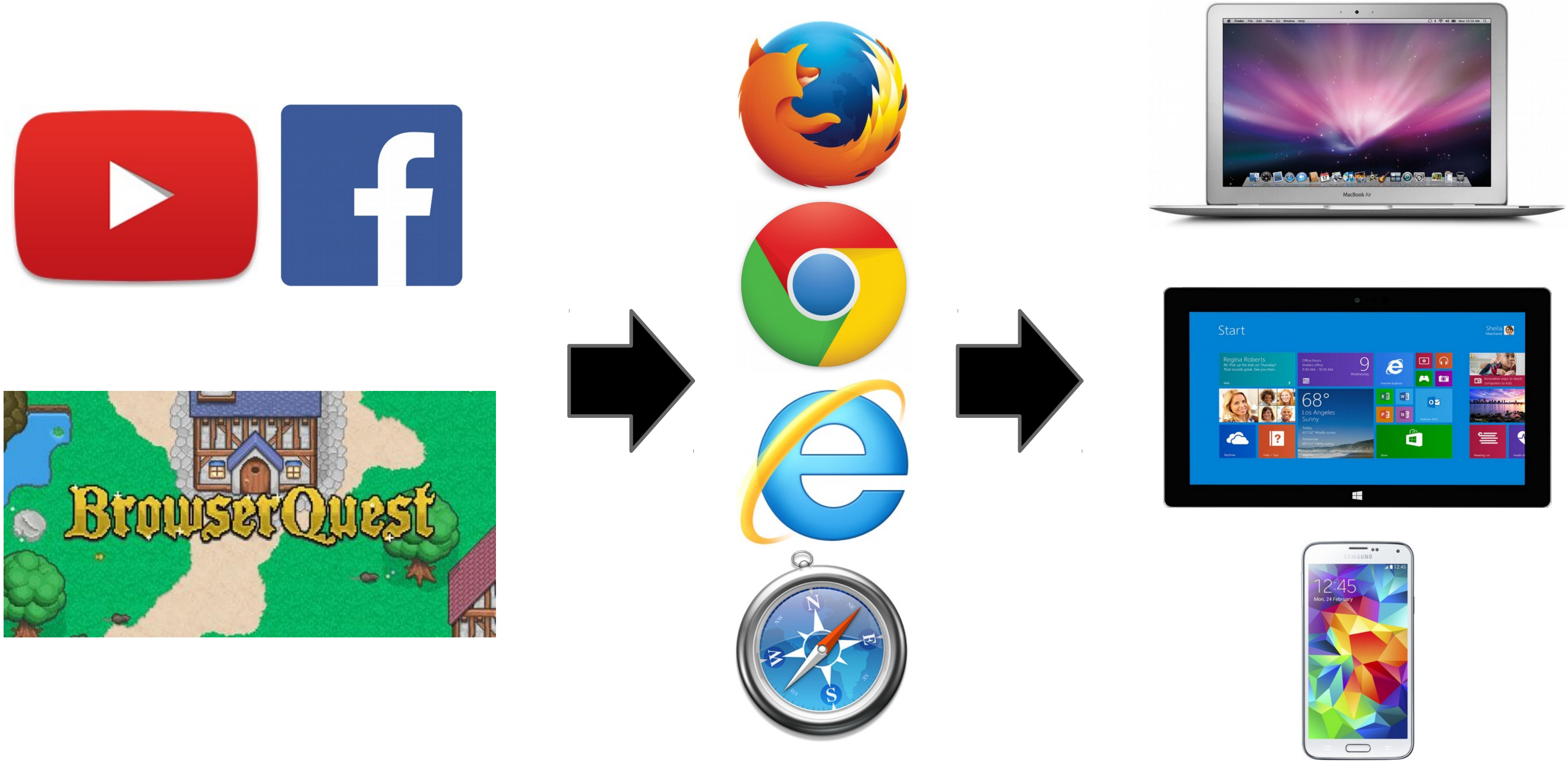
CUDA,
OpenCL

Consumer Electronics

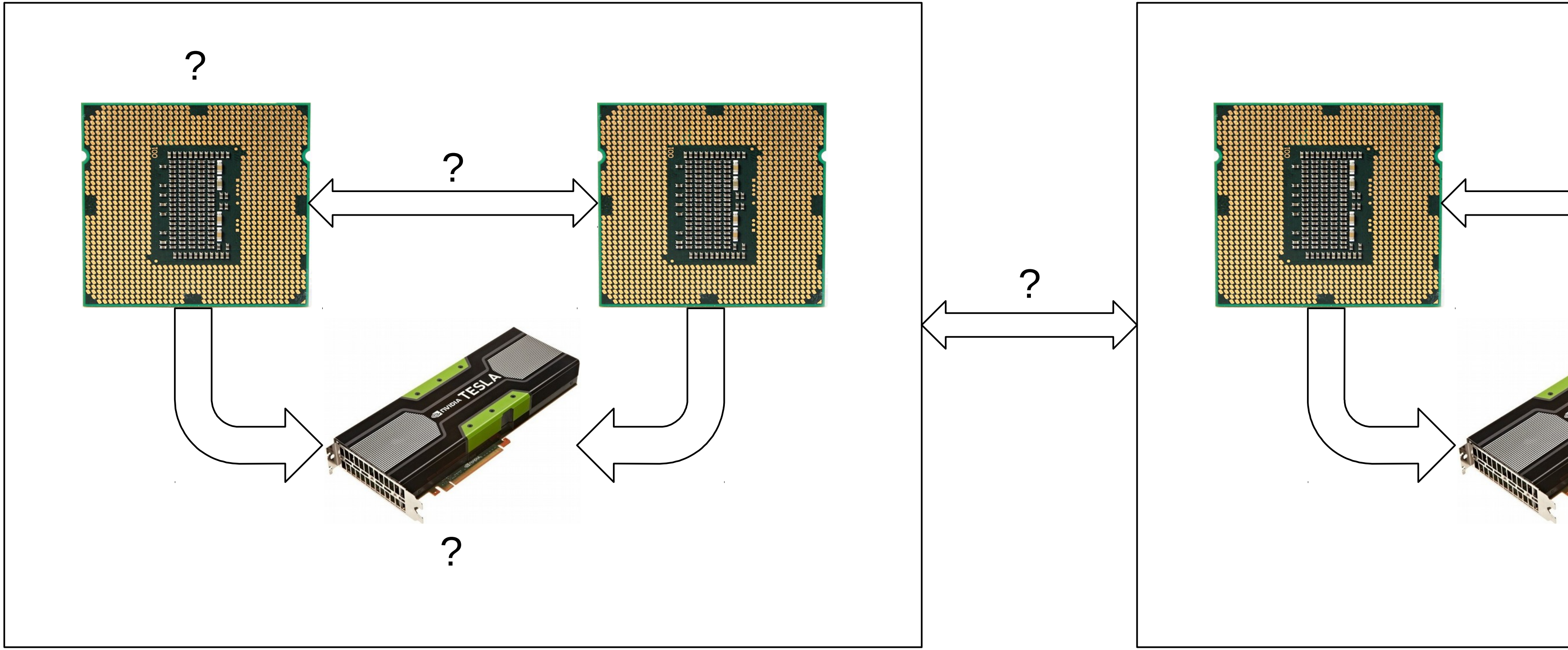




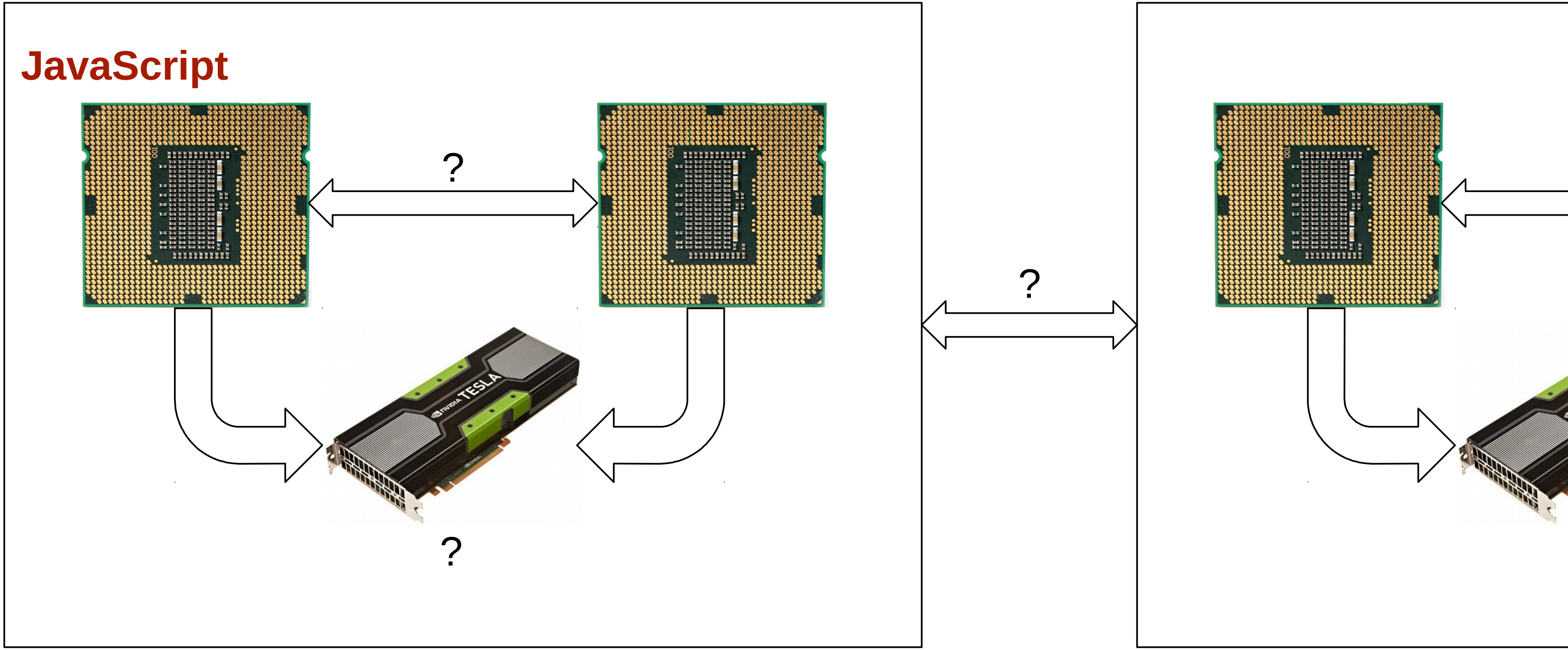
Web Browsers



HPC in Web Browsers



JavaScript



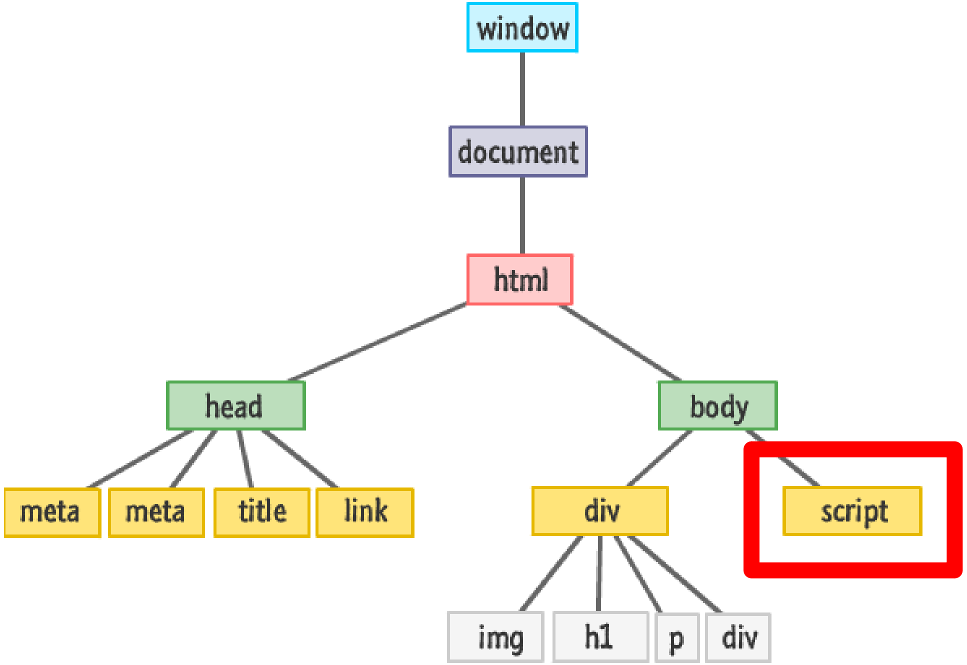
JavaScript

?

?

?

JavaScript in HTML DOM



kirupa.com

JavaScript Type System

- Dynamically typed
- Object-oriented
- Classless
- Prototypes

JavaScript Object Notation (JSON)

```
var car_prototype = {
        "Brand":      "Audi",
        "Plate":      "MUSTER 00",
        "Max.km/h": 250,
        "Owner": {
                "Name":       "Max",
                              "Male":        true,
                              "Hobbies":    ["Driving",
        "Reading"],
                              "Age": 42,
                      }
                };

                var car_copy = Object.create(car_prototype);
                car_copy["Price"] = 10000;
                car_copy["Owner"]["Hobbies"].push("Sport");
```

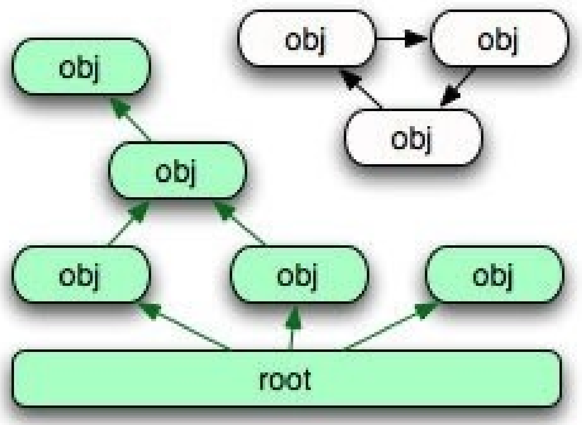
Garbage Collection

Provided by Browsers

Frees unused objects

- Implicit
- No leaks
- Undefined time

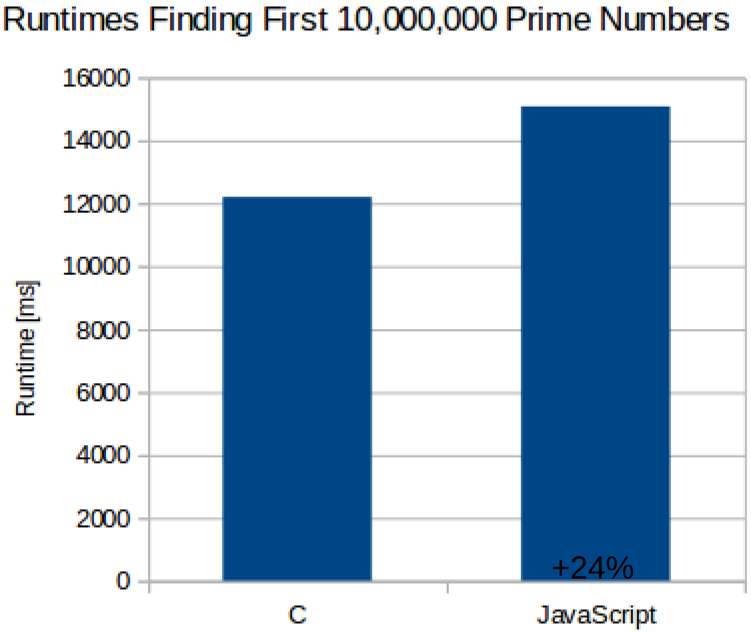
Mark and Sweep



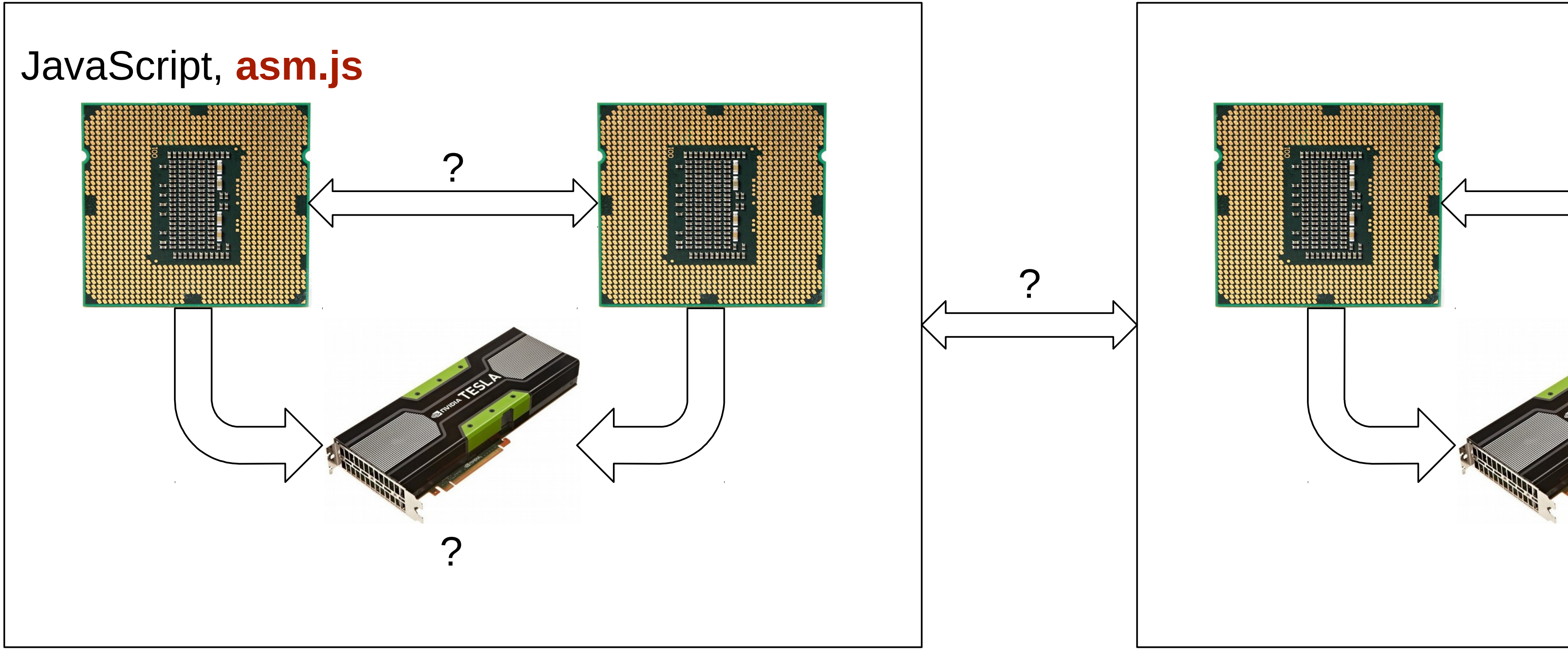
adobe.com

Performance Example

```
var primes = [];

for (var p = 2; p <= max; p++) {
 var is_prime = true;

 for (var i = 2; i <= Math.sqrt(max); i++)
  if (p % i == 0 && p != i) {
   is_prime = false;
   break;
  }

 primes[p] = is_prime;
}
```



asm.js



JavaScript, **asm.js**

?

?

?

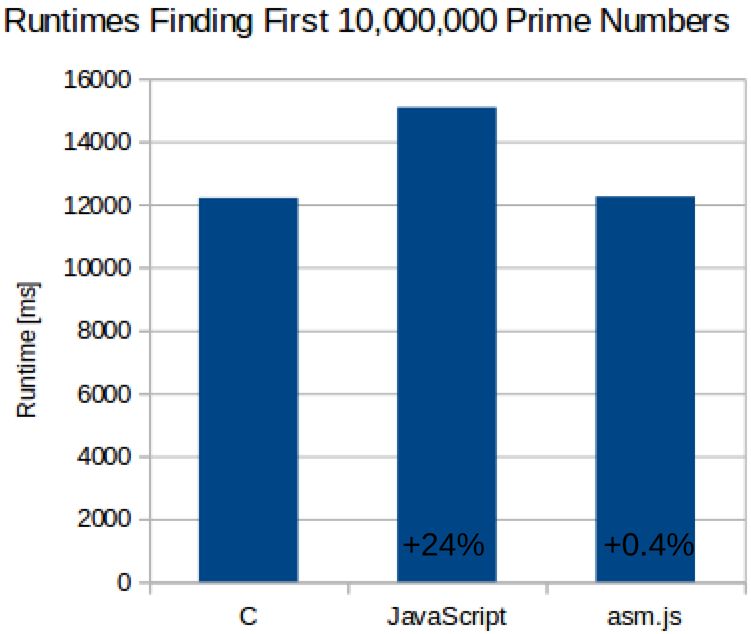
asm.js

“optimizable, low-level subset of JS” - Mozilla

```
var primes = [];

for (var p = 2; p <= max; p++) {
 var is_prime = true;

 for (var i = 2; i <= Math.sqrt(max); i++)
  if (p % i == 0 && p != i) {
   is_prime = false;
   break;
  }

 primes[p] = is_prime;
}
```

```
var primes = new Int32Array(max);

for (var p = (2|0); p <= max; p++) {
 var is_prime = (1|0);

 for (var i = (2|0); i <= (Math.sqrt(max)|0); i++)
  if (p % i == (0|0) && p != i) {
   is_prime = (0|0);
   break;
  }

 primes[p] = is_prime;
}
```

asm.js



```
var primes = new Int32Array(max);

for (var p = (2|0); p <= max; p++) {
 var is_prime = (1|0);

 for (var i = (2|0); i <= (Math.sqrt(max)|0); i++)
  if (p % i == (0|0) && p != i) {
   is_prime = (0|0);
   break;
  }

 primes[p] = is_prime;
}
```

LLVM

ISA of a virtual machine + compilation passes

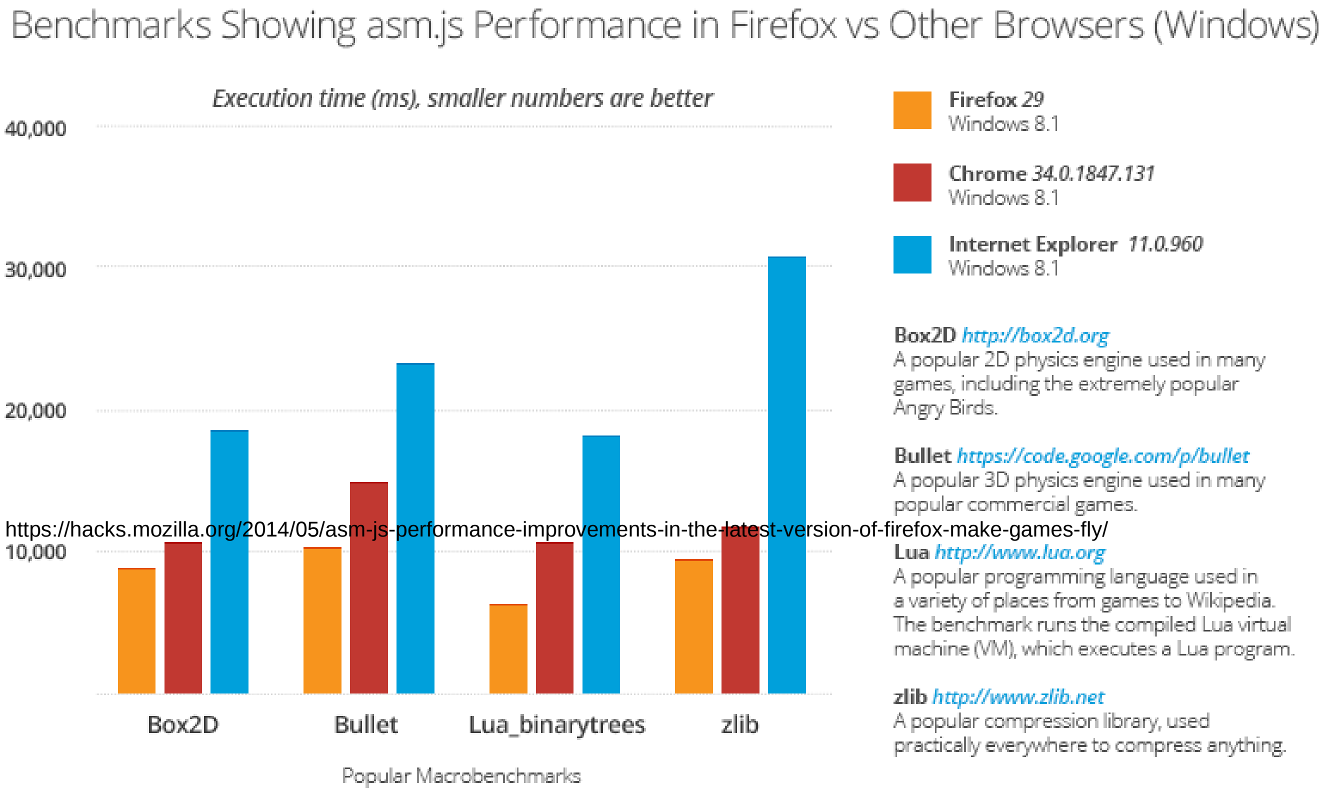
C, C++, Java, Python, … → LLVM IR → Binary

Emscripten

C/C++ → LLVM IR → asm.js

clang Emscripten

asm.js Support



Benchmarks Showing asm.js Performance in Firefox vs Other Browsers (Windows)

🟧 **Firefox** *29*
Windows 8.1

🟥 **Chrome** *34.0.1847.131*
Windows 8.1

🟦 **Internet Explorer** *11.0.960*
Windows 8.1

Box2D *http://box2d.org*
A popular 2D physics engine used in many games, including the extremely popular Angry Birds.

Bullet *https://code.google.com/p/bullet*
A popular 3D physics engine used in many popular commercial games.

Lua *http://www.lua.org*
A popular programming language used in a variety of places from games to Wikipedia. The benchmark runs the compiled Lua virtual machine (VM), which executes a Lua program.

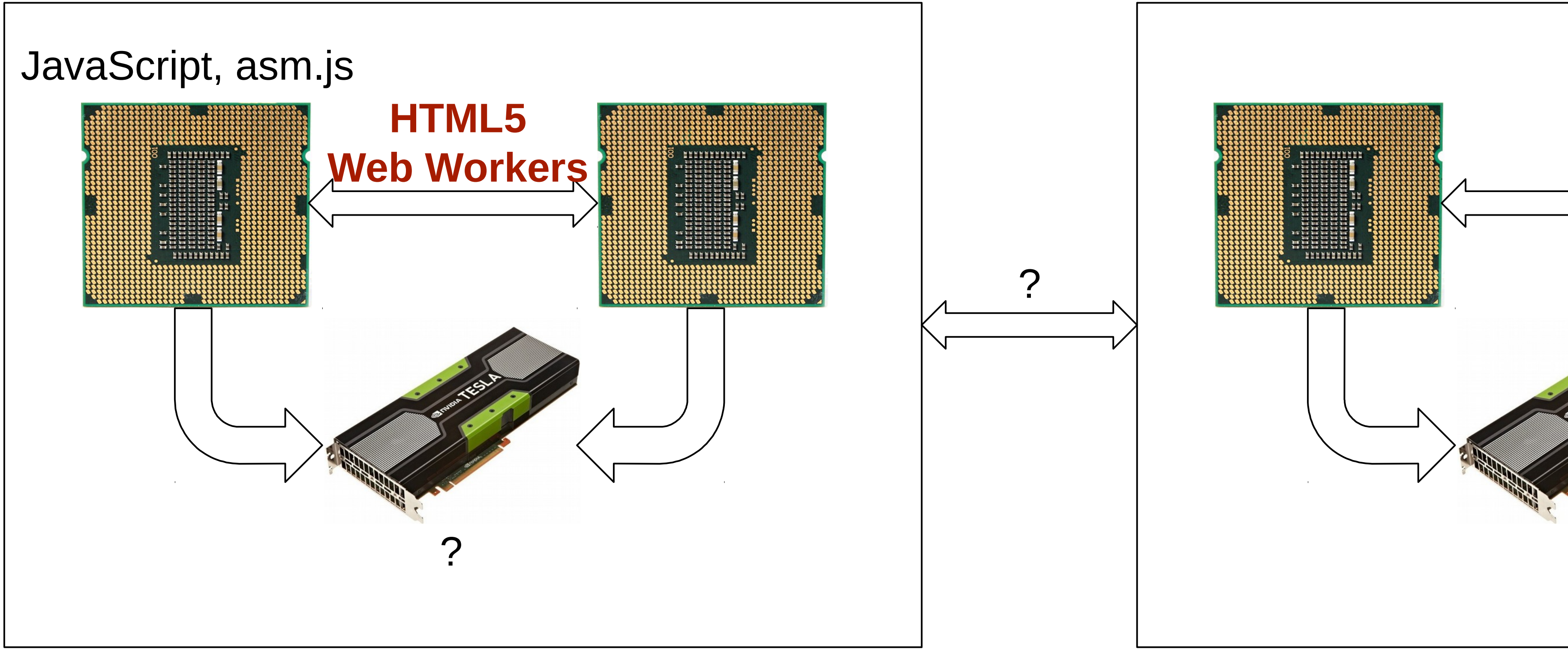
zlib *http://www.zlib.net*
A popular compression library, used practically everywhere to compress anything.

https://hacks.mozilla.org/2014/05/asm-js-performance-improvements-in-the-latest-version-of-firefox-make-games-fly/

asm.js Summary

- JavaScript subset
- Annotation-based optimization detection
- Aims at near-native performance
- Compilable from C/C++
- Growing browser support

HTML5 Web Workers



JavaScript, asm.js

HTML5
Web Workers

?

?

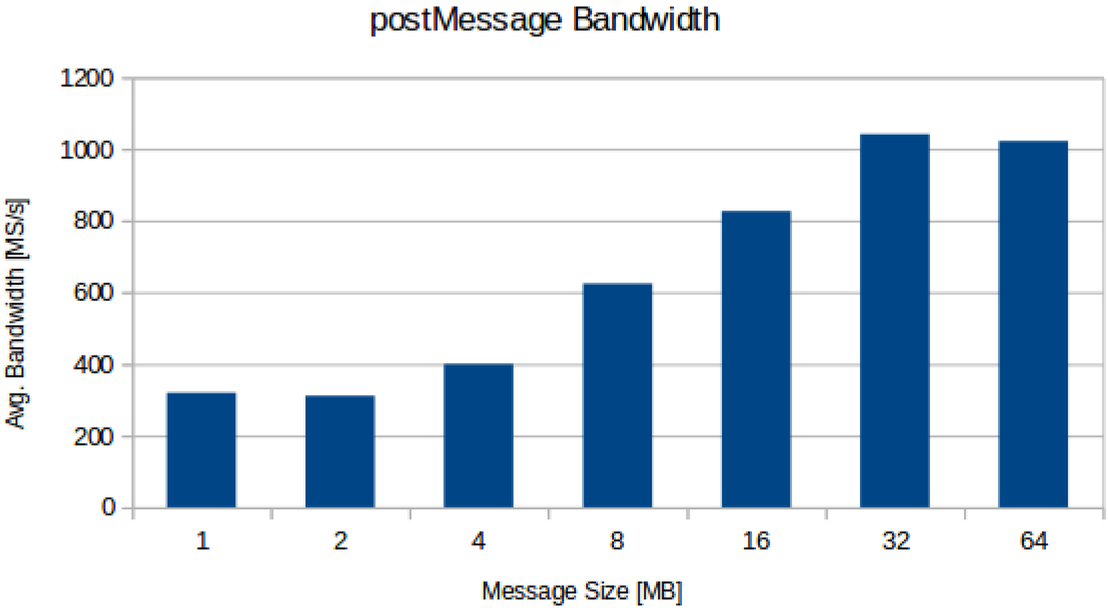
HTML5 Web Workers

Threads communicating via Message Passing

```
<script>
var worker = new Worker(“worker_script.js”);

worker.addEventListener(“message”, function(e) {
      console.log(e.data);
}, false);

worker.postMessage(“Hello!”);
</script>
```

```
worker_script.js:


self.addEventListener(“message”, function(e) {
      // Async computations go here
      self.postMessage(e.data);
}, false);
```

postMessage: Structured Cloning



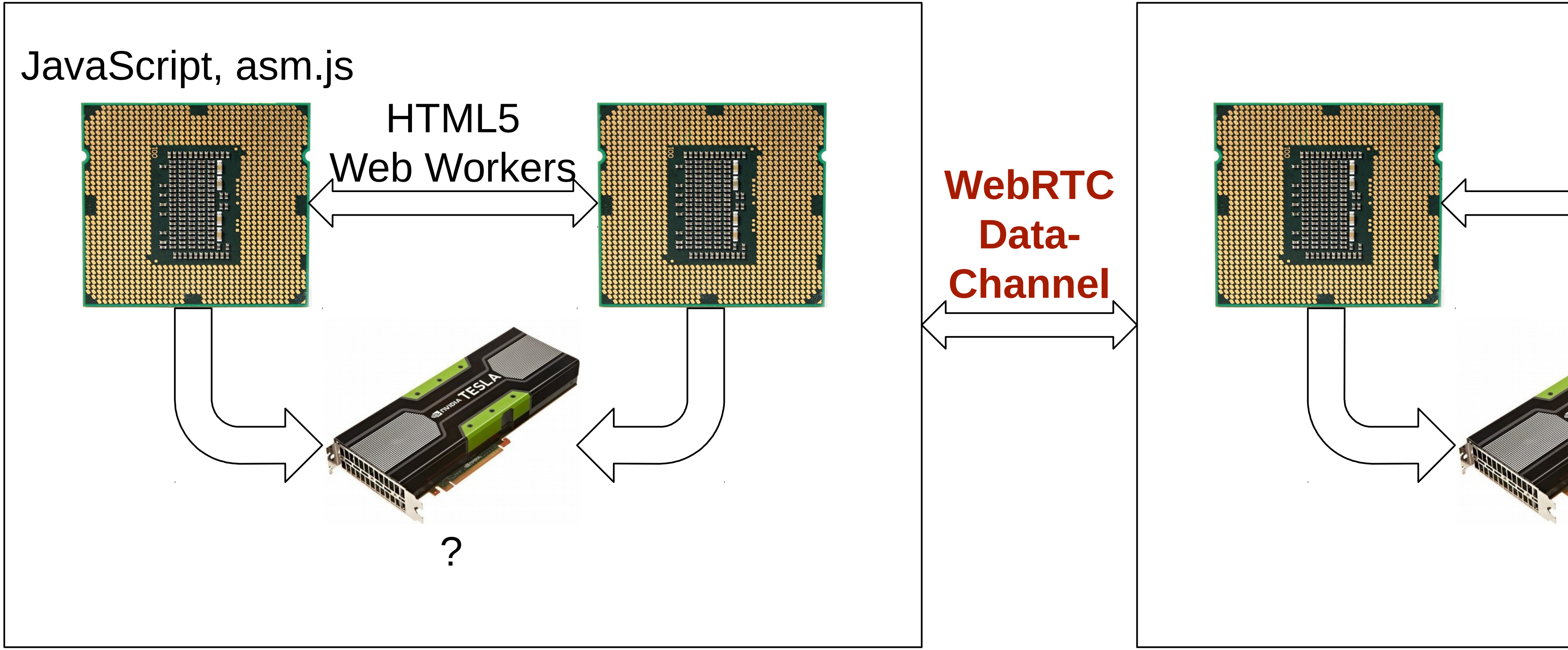
Garbage collection!

postMessage: Transferable Objects

```
var array = new ArrayBuffer(1024); // 1kB
worker.postMessage(array.buffer, [array.buffer]);
```

Transferred data switch contexts!

WebRTC DataChannel



JavaScript, asm.js

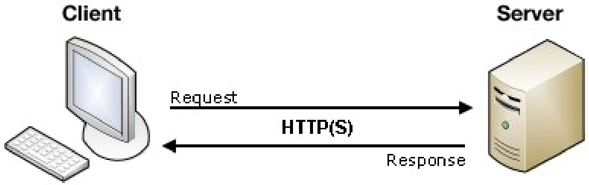
HTML5
Web Workers

?

WebRTC
Data-
Channel

HTTP

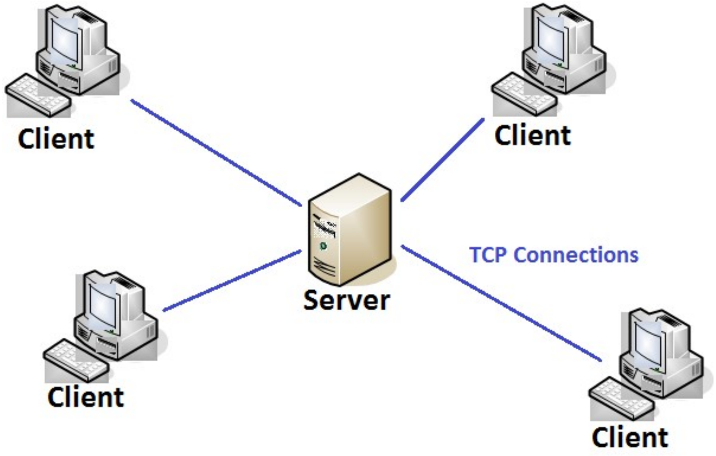
- Stateless
- GET, POST, …
- Cookies



developer.mozilla.org

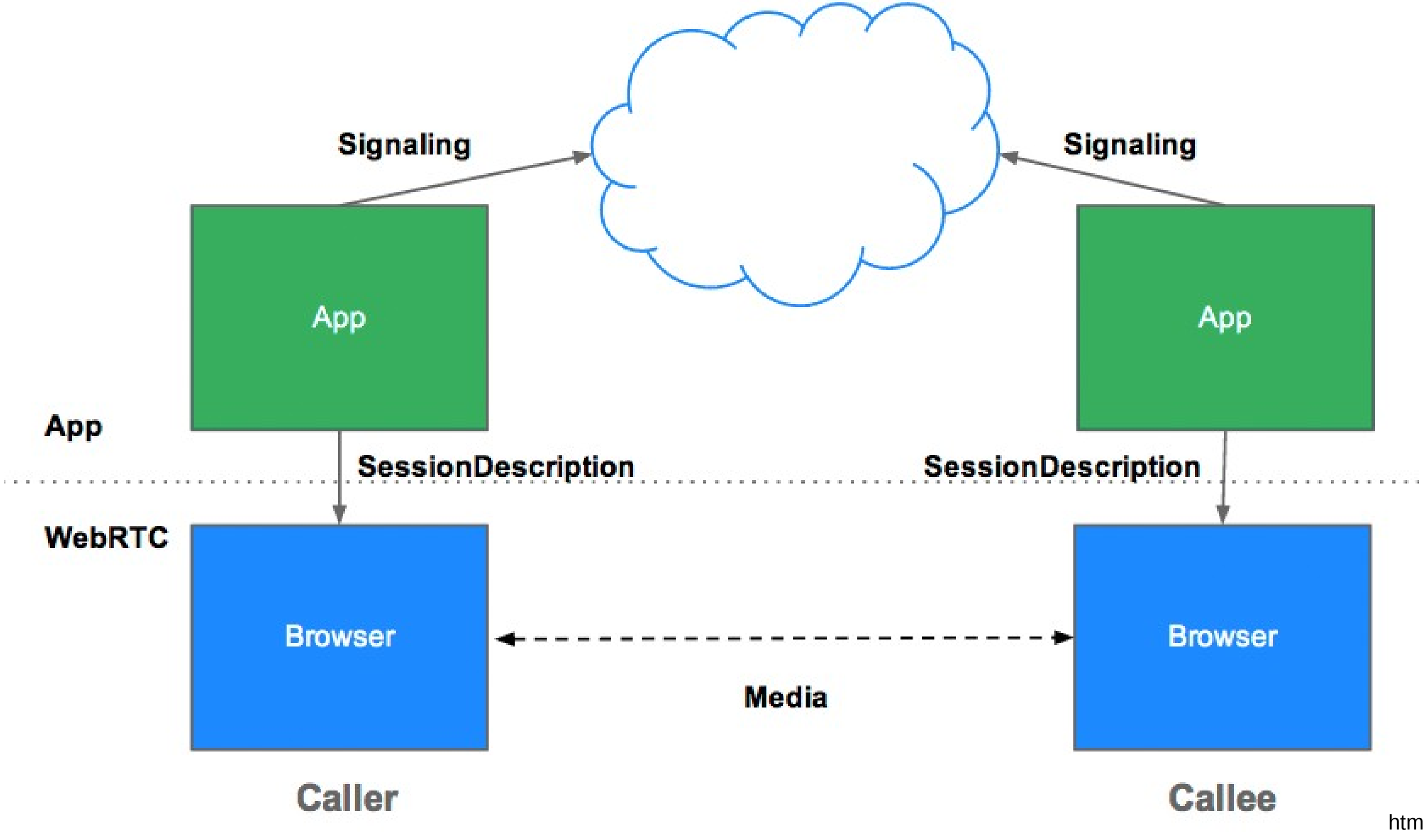
HTML5 WebSocket

- Client server arch.
- TCP only



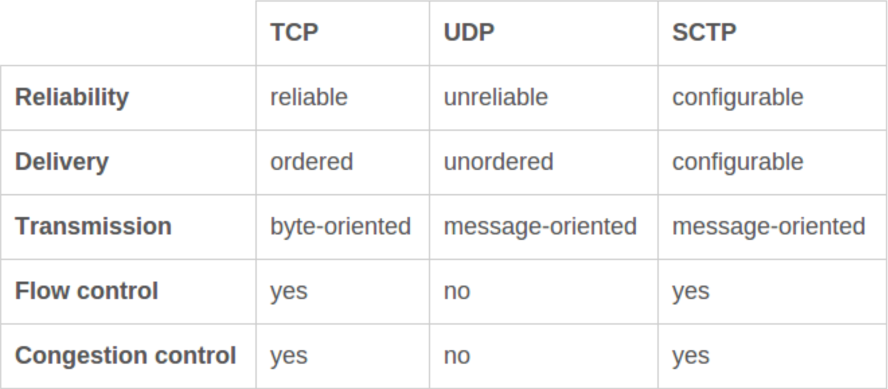
hilmanr.blogspot.com

WebRTC

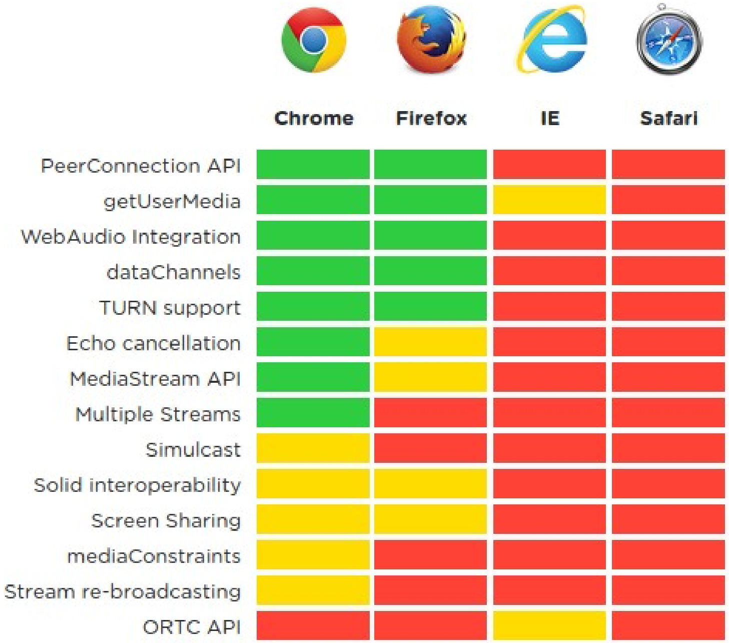


html5rocks.org

WebRTC DataChannel



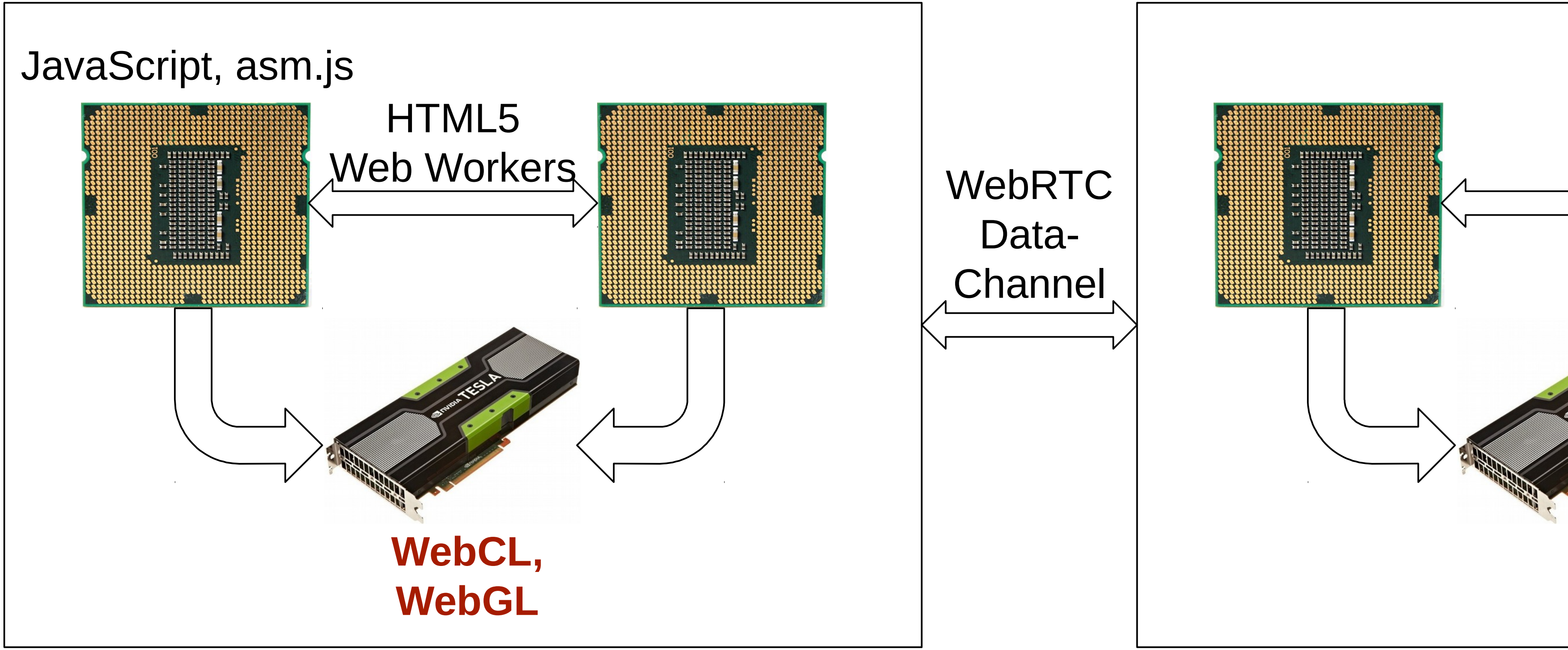
	TCP	UDP	SCTP
Reliability	reliable	unreliable	configurable
Delivery	ordered	unordered	configurable
Transmission	byte-oriented	message-oriented	message-oriented
Flow control	yes	no	yes
Congestion control	yes	no	yes



	Chrome	Firefox	IE	Safari
PeerConnection API	🟩	🟩	🟥	🟥
getUserMedia	🟩	🟩	🟨	🟥
WebAudio Integration	🟩	🟩	🟥	🟥
dataChannels	🟩	🟩	🟥	🟥
TURN support	🟩	🟩	🟥	🟥
Echo cancellation	🟩	🟨	🟥	🟥
MediaStream API	🟩	🟨	🟥	🟥
Multiple Streams	🟩	🟥	🟥	🟥
Simulcast	🟨	🟥	🟥	🟥
Solid interoperability	🟨	🟨	🟥	🟥
Screen Sharing	🟨	🟨	🟥	🟥
mediaConstraints	🟨	🟥	🟥	🟥
Stream re-broadcasting	🟨	🟥	🟥	🟥
ORTC API	🟥	🟥	🟨	🟥

webrtc.org

WebCL, WebGL



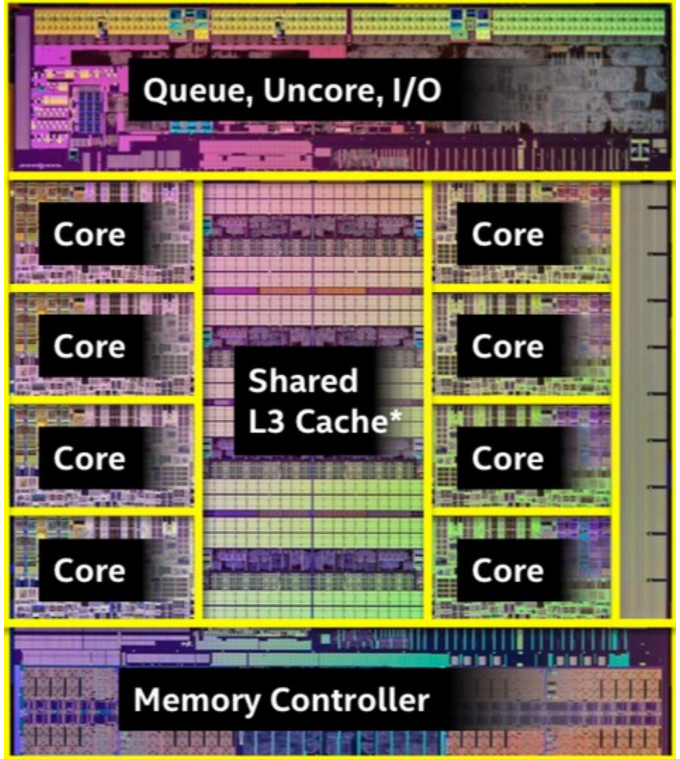
JavaScript, asm.js

HTML5
Web Workers

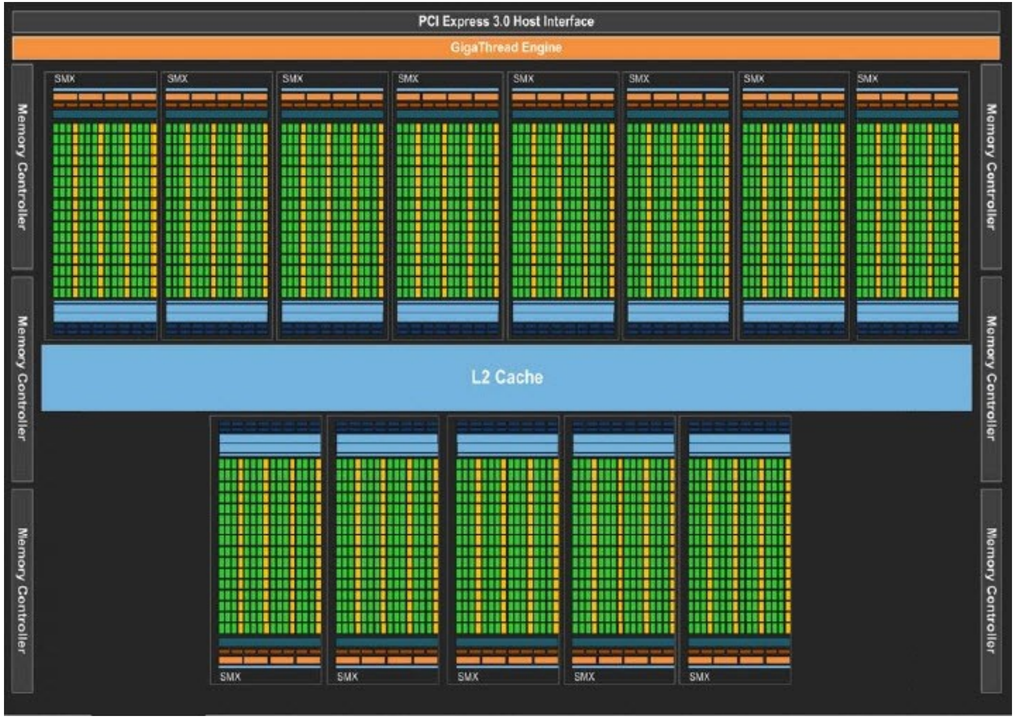
WebRTC
Data-
Channel

WebCL,
WebGL

GPU Computing



arstechnica.net

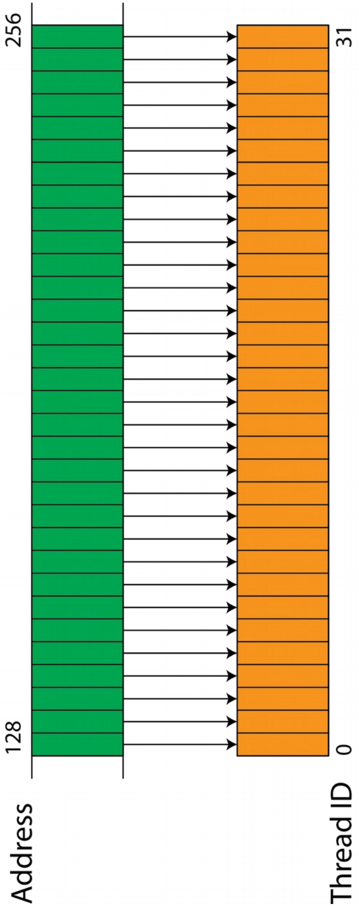


wccftech.com

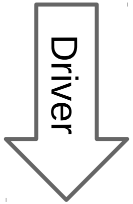
WebCL

- Like OpenCL
- Hardware exposure
- IEEE 754 float
- Heterogeneous

- Drivers
- Adaption



```
“__kernel void vectorAdd
   __global const float* x,
   __global const float* y,
   __global float* restrict z)
{
   int index = get_global_id(0);
   z[index] = x[index] + y[index];
}”
```



Compute Shaders

- Since 4.3 (ES 3.1)
- GLSL, adaption

- Graphics abstraction
- No IEEE 754 float

- WebGL 1.0/2.0

```
layout(local_size_x = 16, local_size_y = 16) in;
uniform readonly image2D fromTex;
uniform writeonly image2D toTex;

void main() {
   ivec2 texelCoords = ivec2(gl_GlobalInvocationID.xy);
   vec4 pixel = imageLoad(fromTex, texelCoords);
   pixel.rg = pixel.gr;
   imageStore(toTex, texelCoords, pixel);
}
```

```
L
[2014.03.11-00.51.24:930][498]LogScript:Warning: Accessed None 'BP Holo Scanner Effect ExecuteUbergraph LV Soul Cave JM 00 RefProperty'
```

Summary

asm.js	😊	😊	😕	😐
Web Workers	😊	😊	😕	😊
DataChannel	😊	😊	😐	😐
WebCL/GL CS	😐	😐	😐	😐