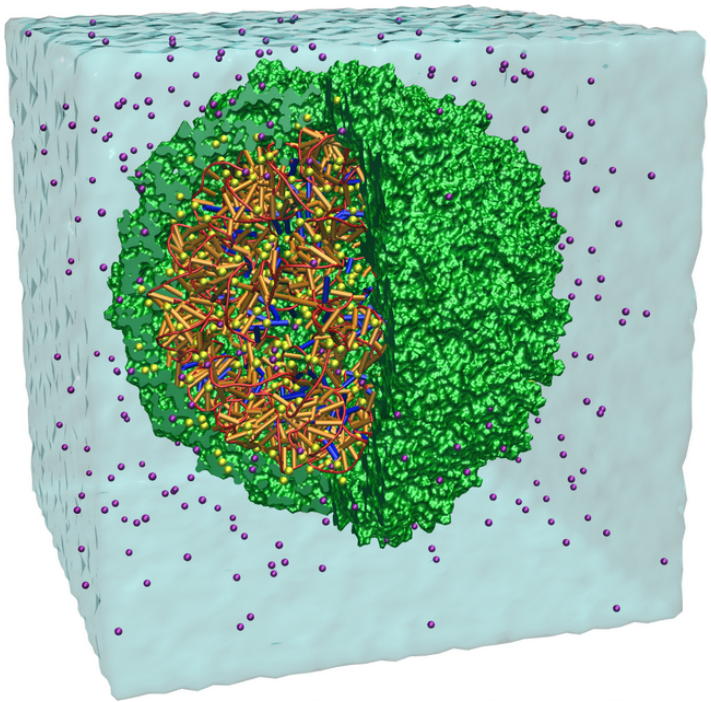
High Performance Computing in Web Browsers

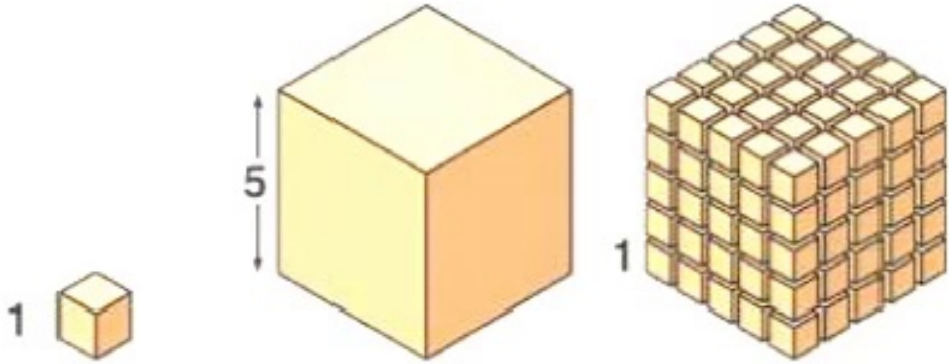
CE Seminar WT14/15

Henning Lohse

High Performance Computing

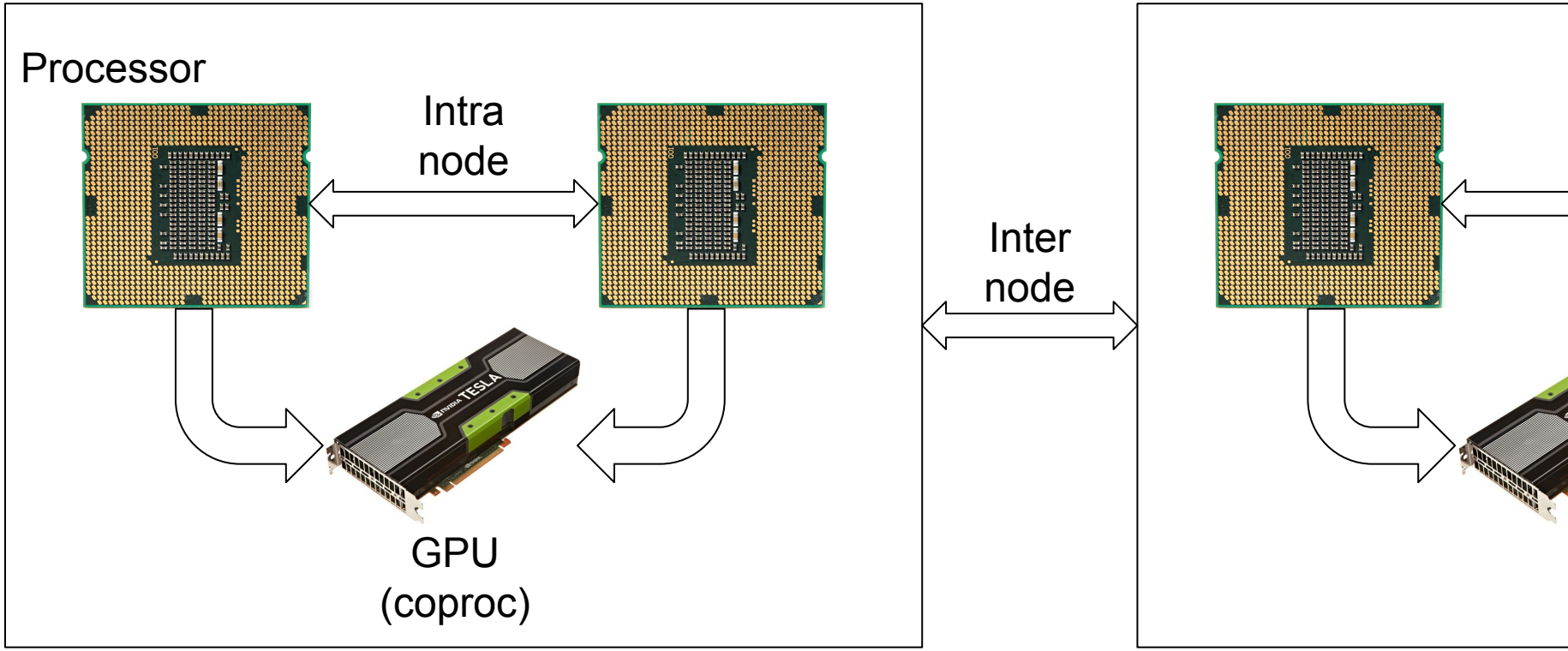


ks.uiuc.edu

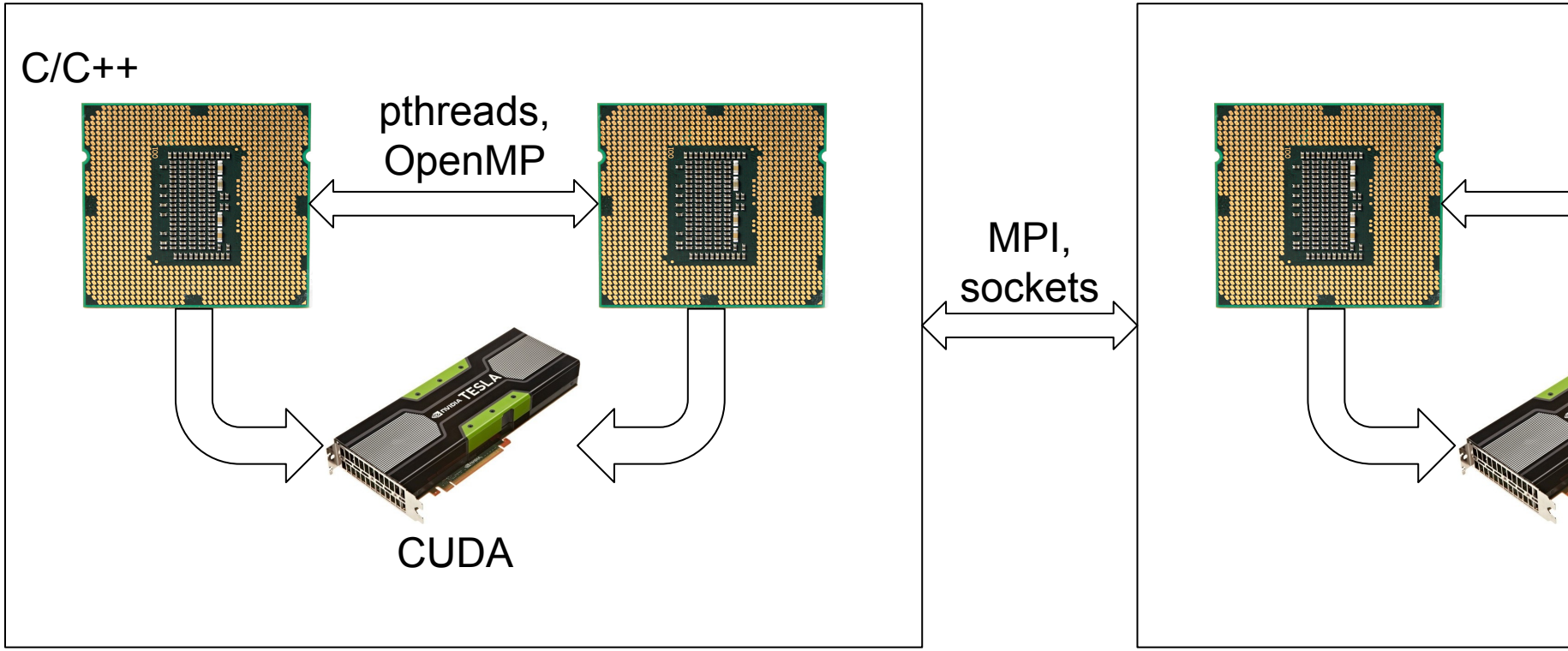


voltagegate.scientopia.org

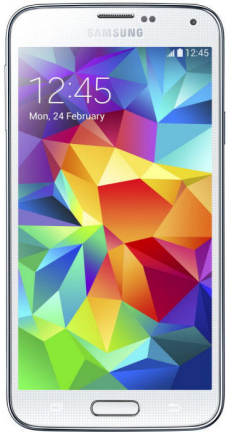
High Performance Computing

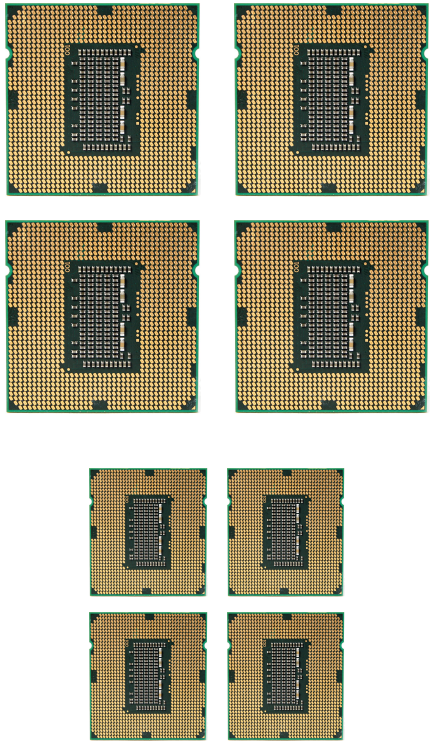


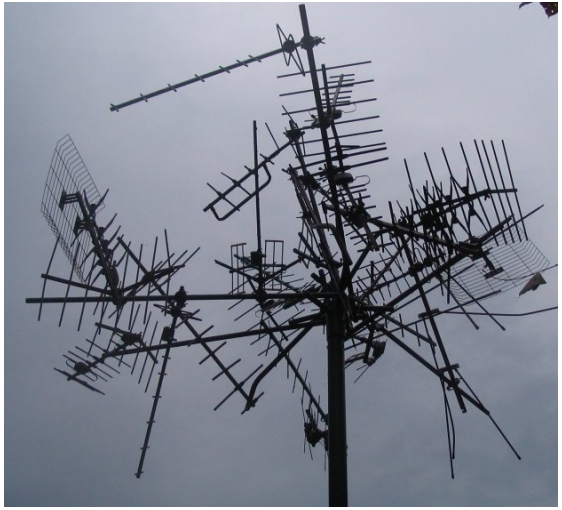
High Performance Computing



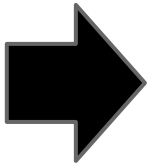
Consumer Electronics

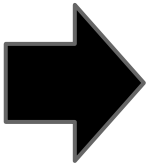




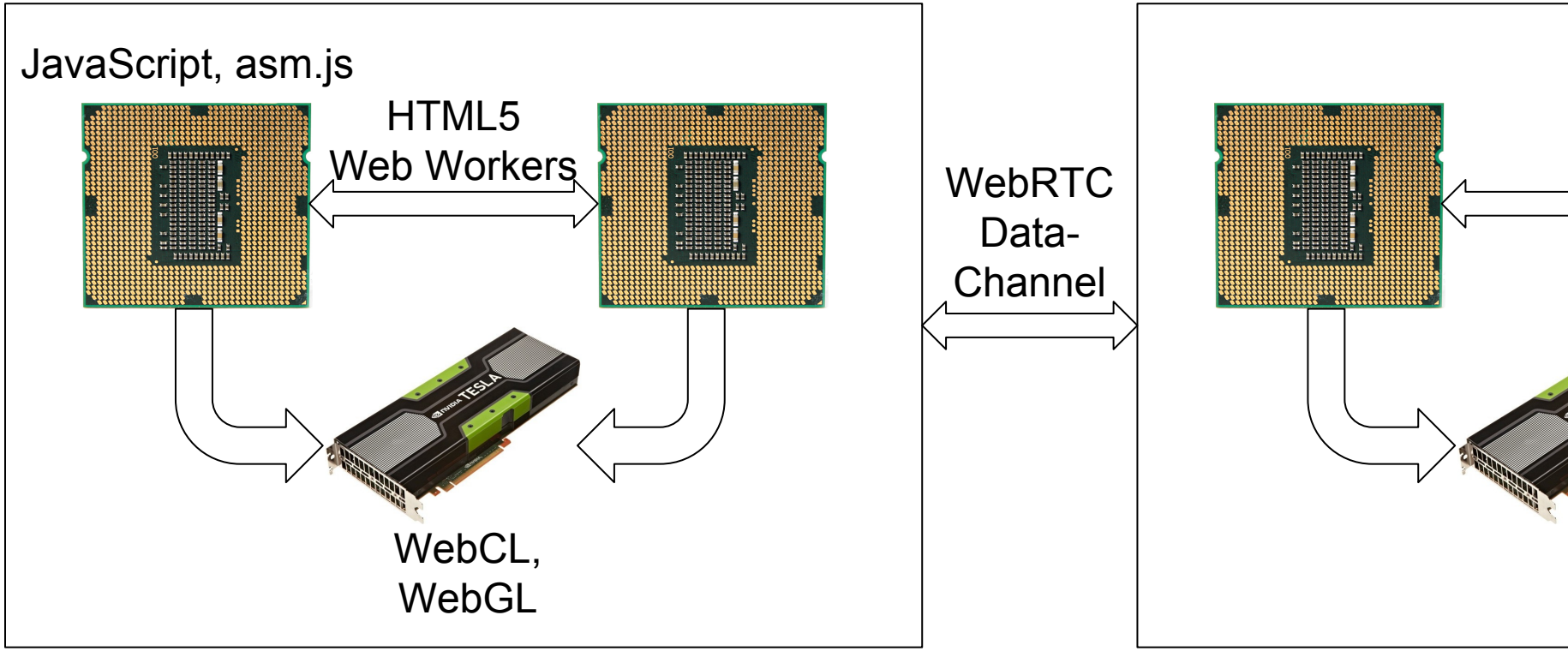


Web Browsers

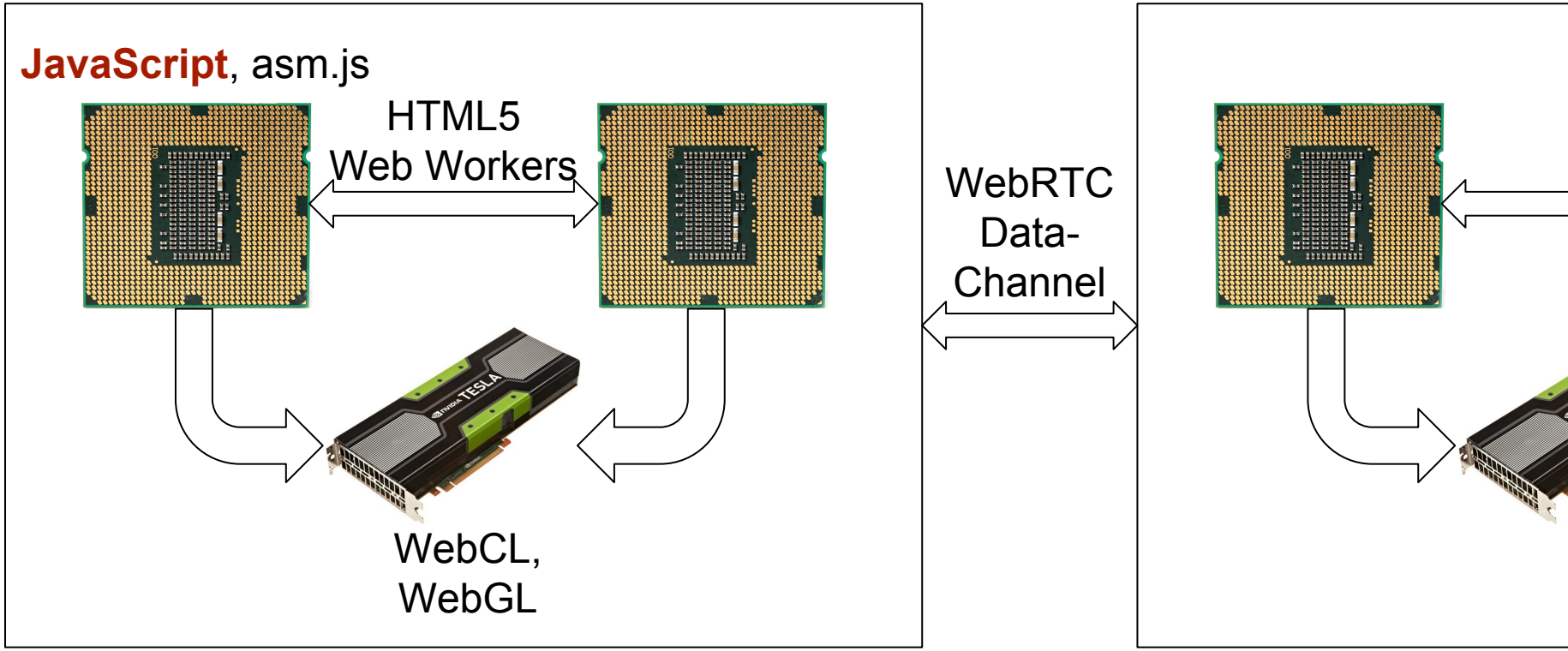




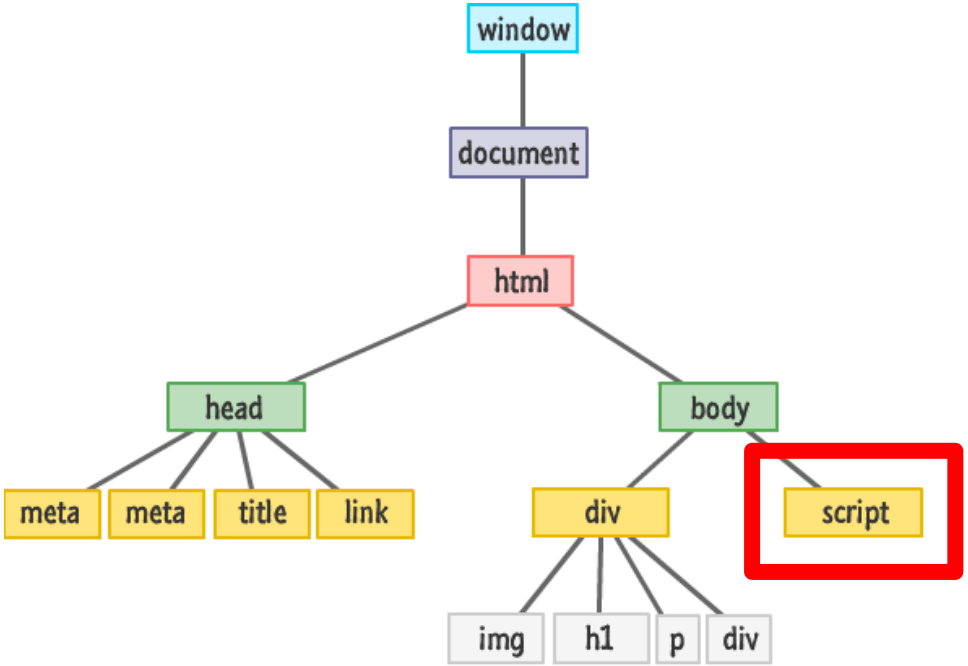
HPC in Web Browsers



JavaScript


JavaScript, asm.js
HTML5
Web Workers
WebCL,
WebGL
WebRTC
Data-
Channel

JavaScript in HTML DOM



JavaScript Object Notation: JSON

JavaScript

- Dynamically typed
- Object-oriented
- Classless
- Prototypes

```
var prototype_object = {
	"Herausgeber":	"Xema",
	"Nummer":	"1234-5678-9012-3456",
	"Deckung":	2e+6,
	"Inhaber": {
		"Name":	"Max",
		"maennlich": true,
		"Hobbys":	["Reiten", "Golfen"],
		"Alter":	42,
	}
};

var copy_object = prototype_object;
copy_object[“Waehrung”] = “EURO”;
copy_object[“Inhaber”][“Hobbys”].push(“Lesen”);
```

Garbage Collection

Provided by Browsers

Frees unused objects

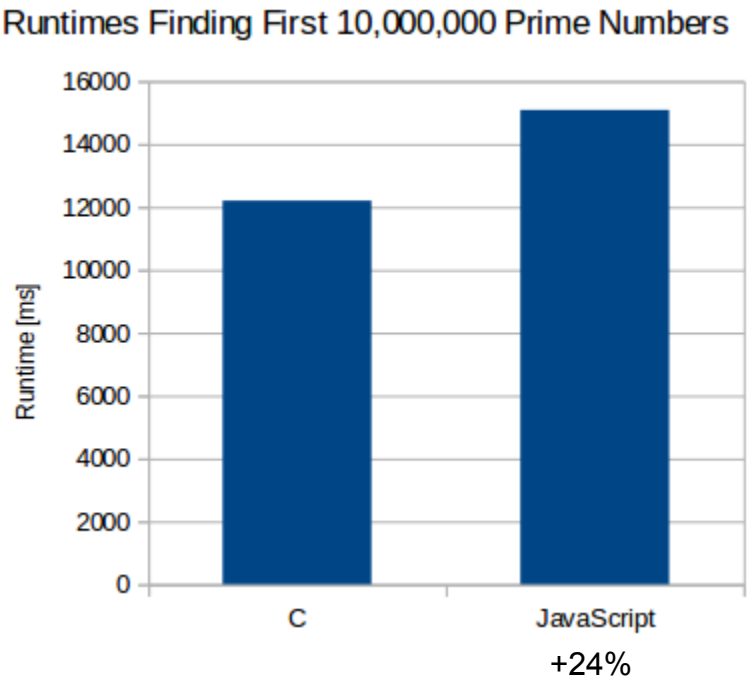
- Implicit
- No leaks
- Undefined time

Reference Counting

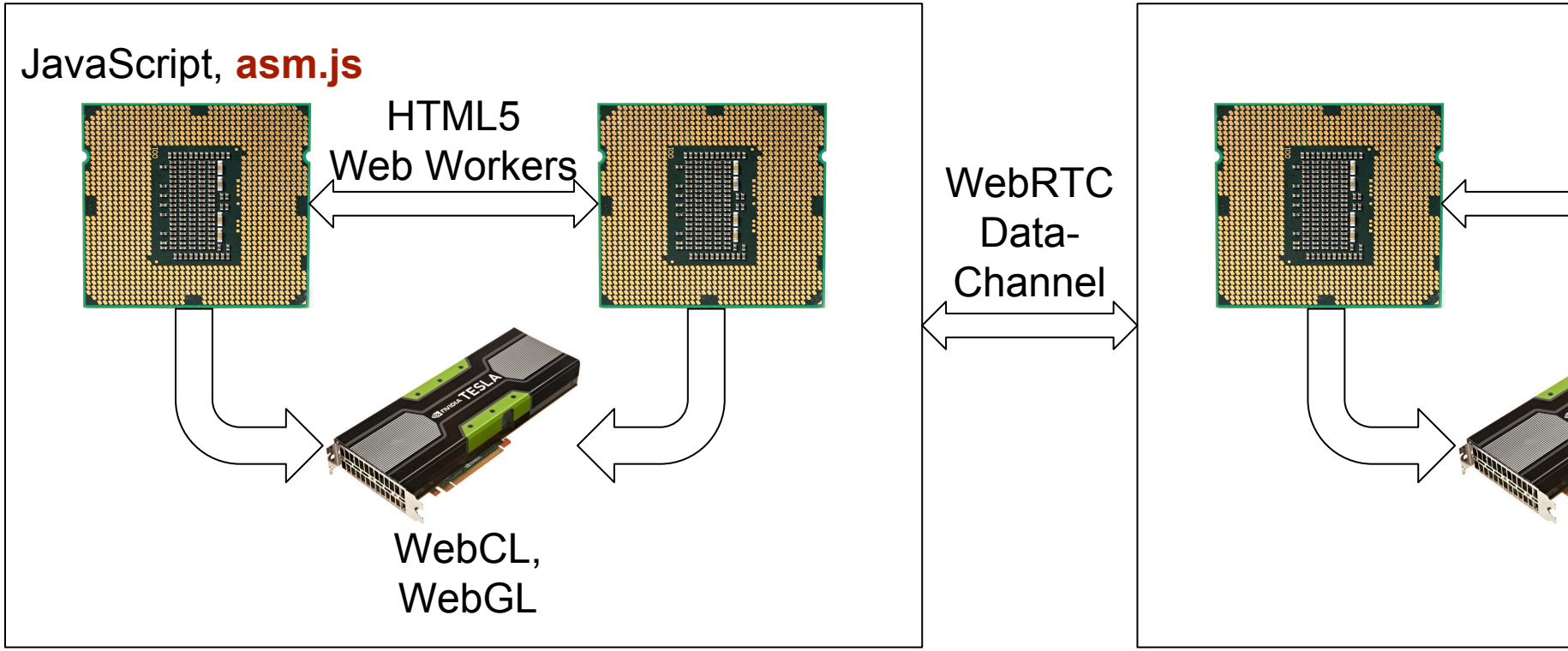
→ Mark and Sweep

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
HTML5
Web Workers
WebCL,
WebGL
WebRTC
Data-
Channel

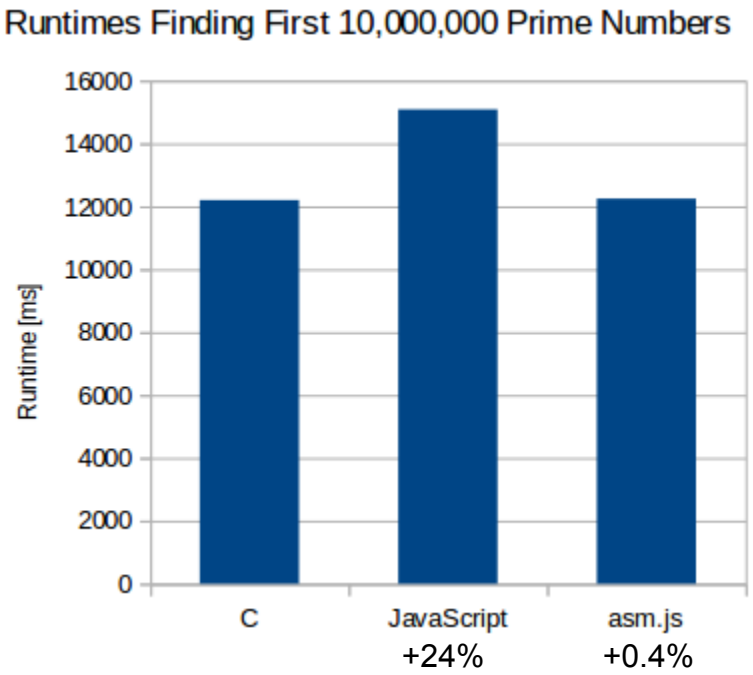
asm.js

“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

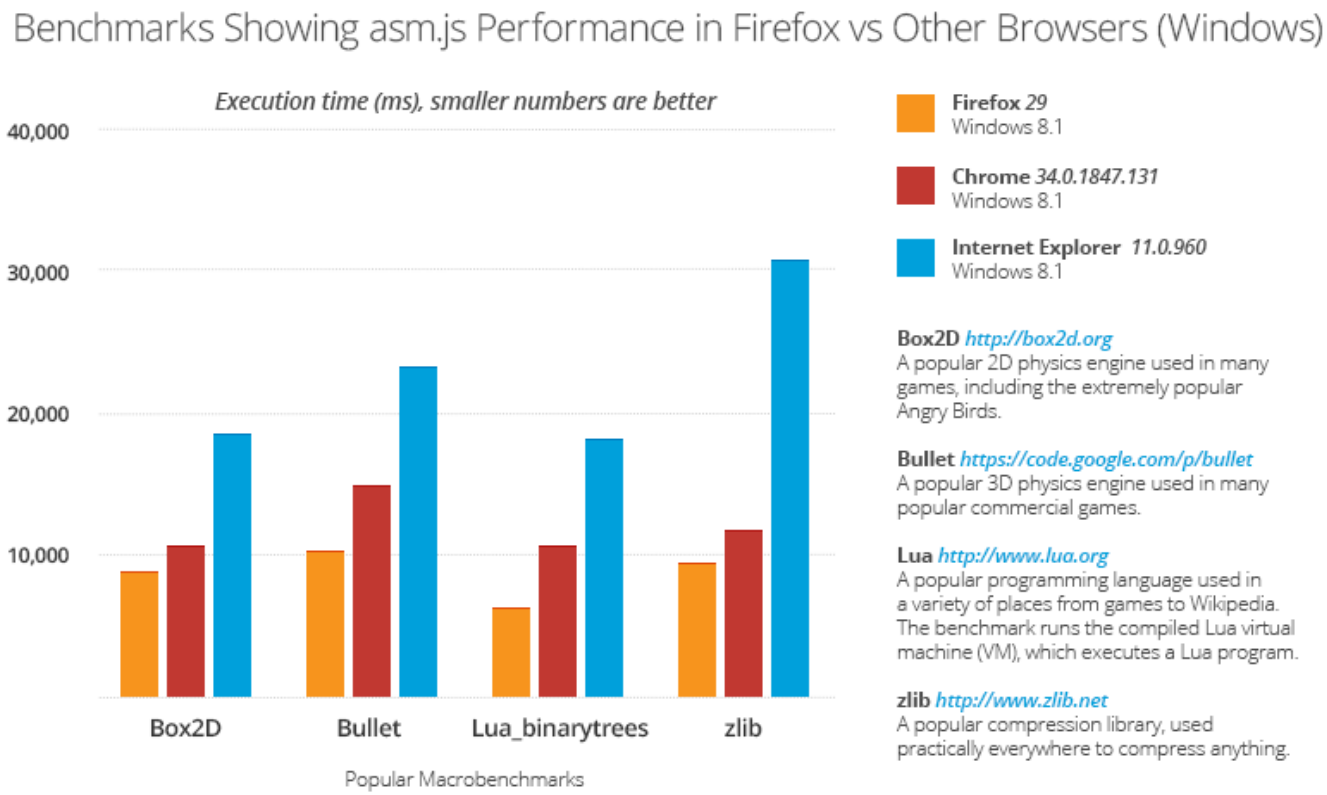
C, C++, Java, Python, … → LLVM IR → Binary

Emscripten

C/C++ → LLVM IR → asm.js

clang Emscripten

asm.js Support

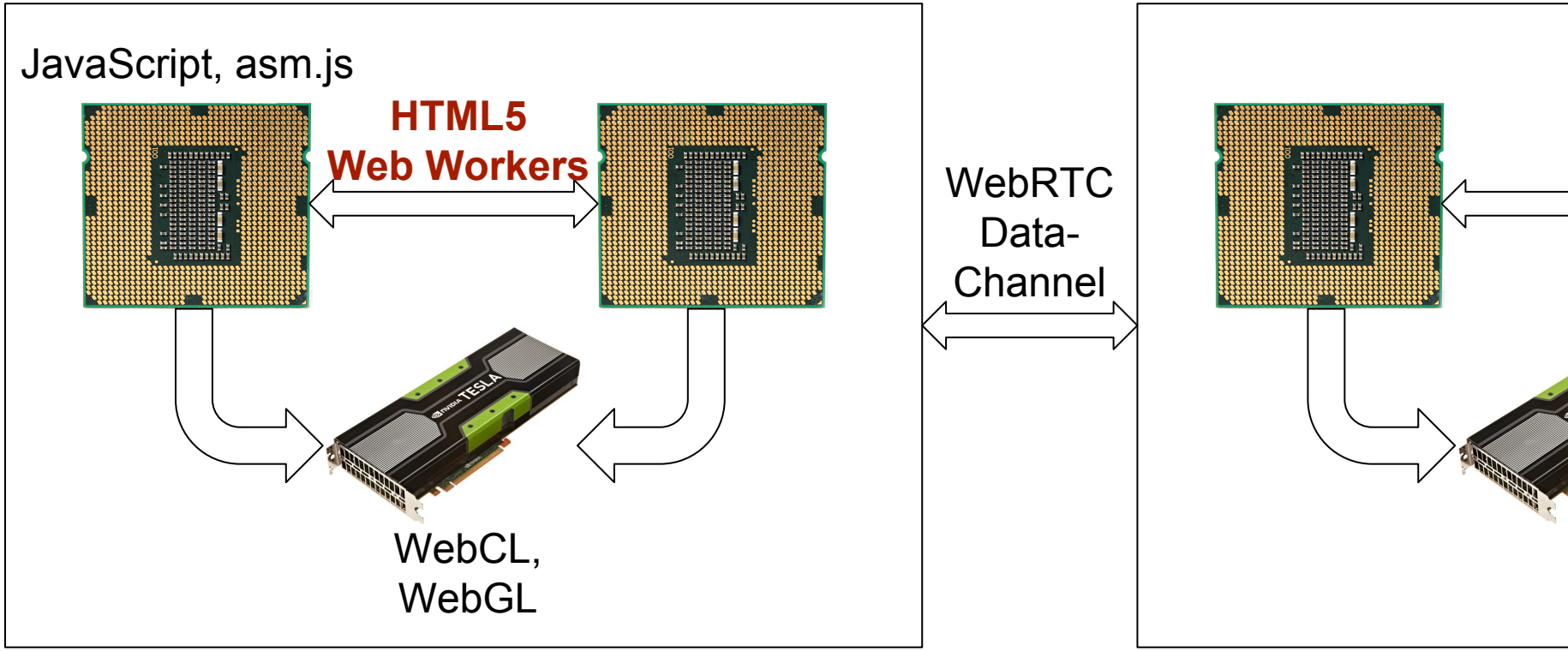


https://hacks.mozilla.org/2014/05/asm-js-performance-improvements-in-the-latest-version-of-firefox-make-games-fly/

asm.js Summary

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

HTML5 Web Workers



HTML5 Web Workers

Threads communicating via Message Passing

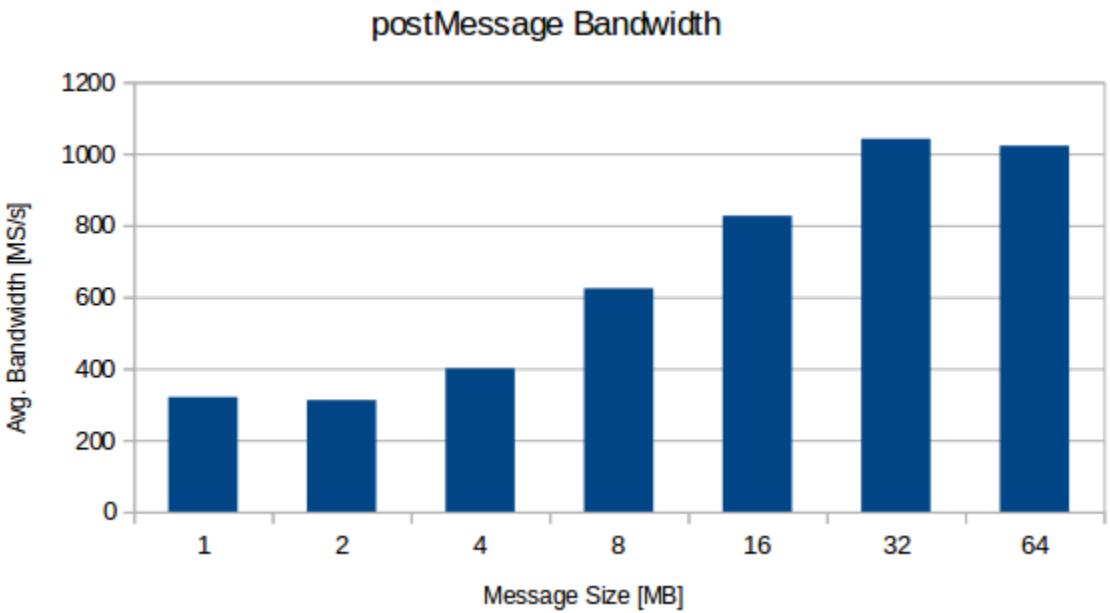
<script>

```
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]);
```

Latency: 53µs

Transferred data switch contexts!

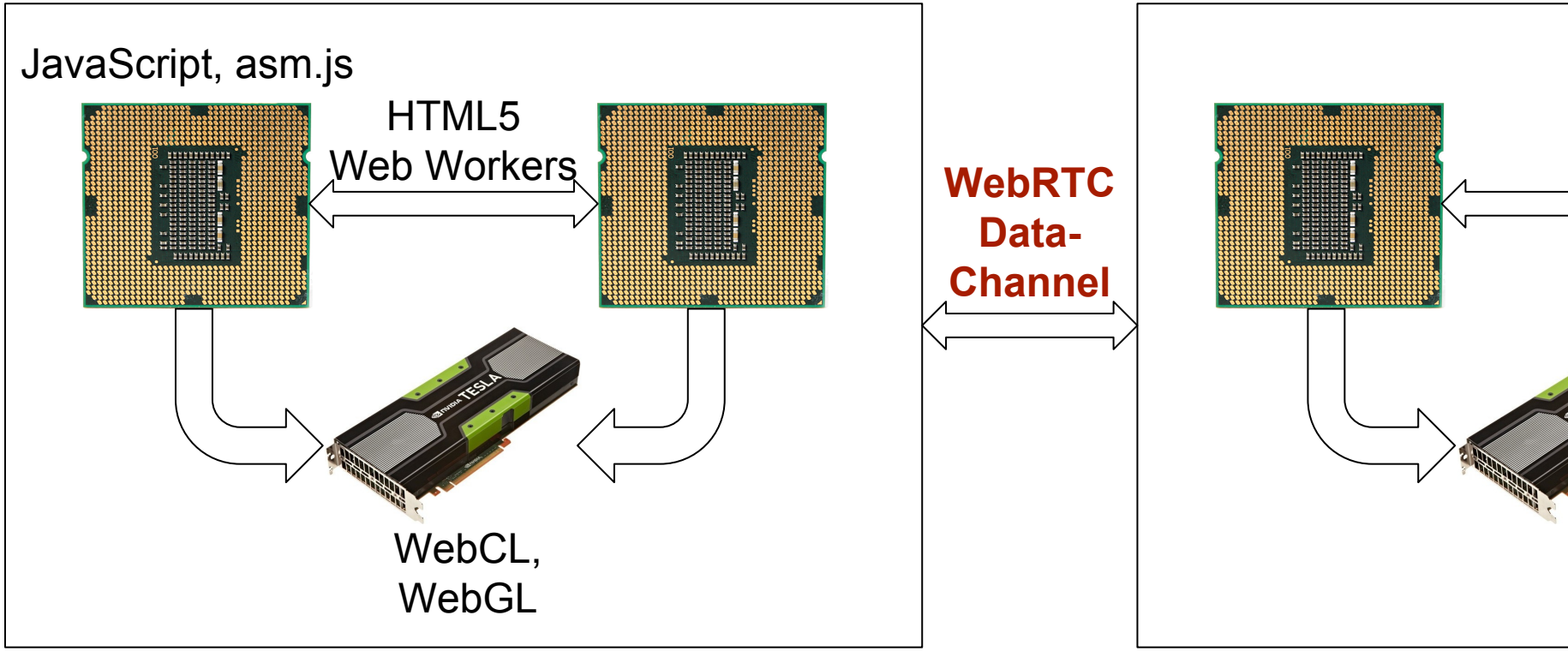
HTML5 Web Workers Summary

- No Shared Memory; Message Passing
- Cloning: Beware of GC, JSON
- Transfer: Beware of data context, JSON

Transferable objects support:

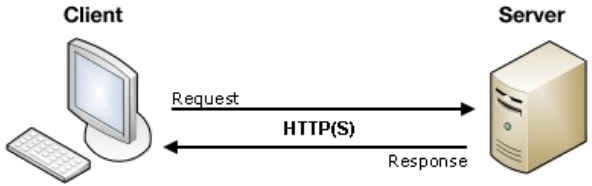


WebRTC DataChannel



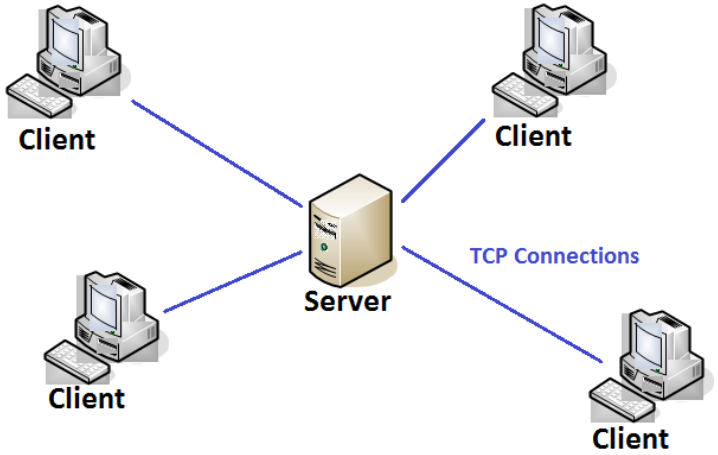
HTTP

- Stateless
- GET, POST, …
- Cookies

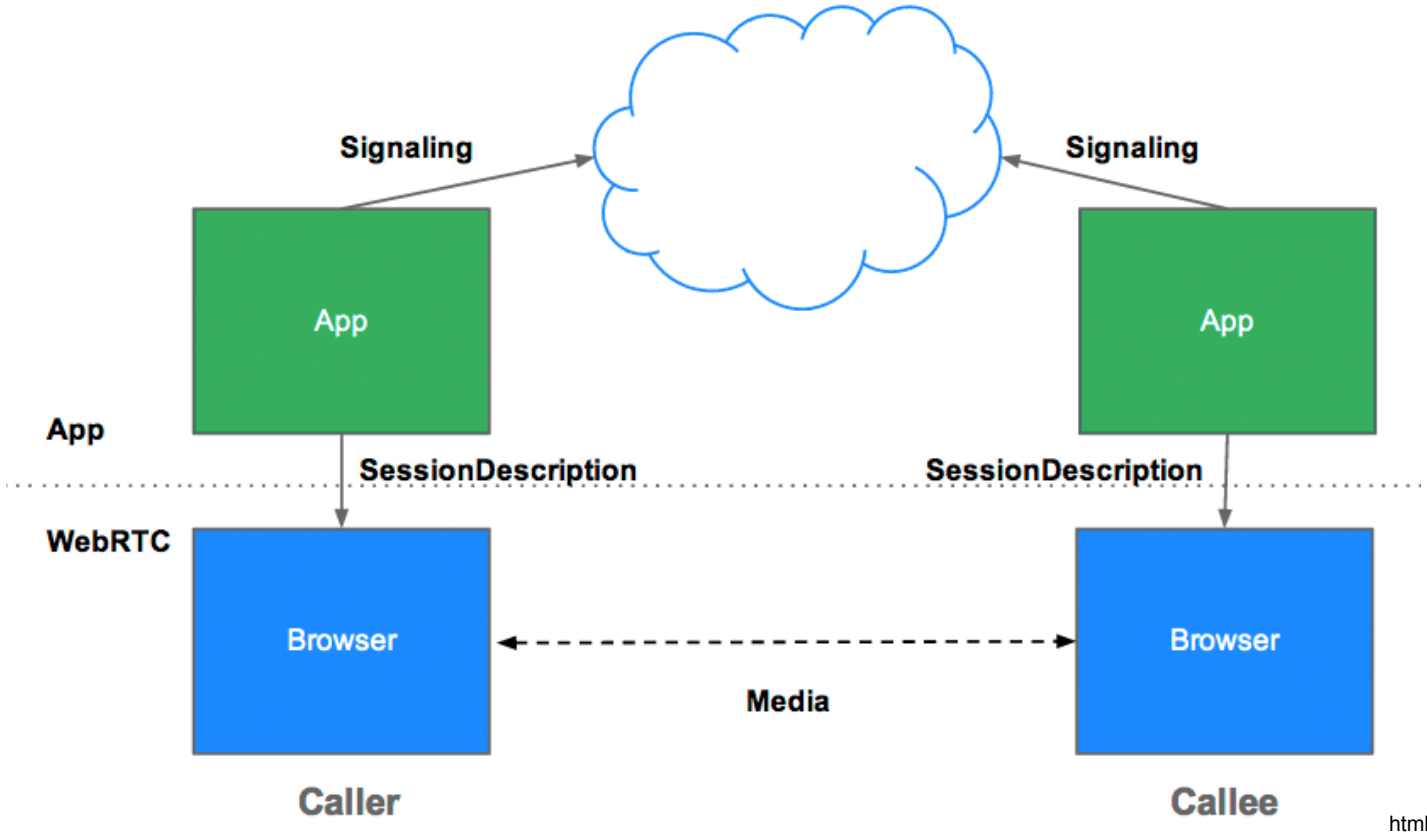


HTML5 WebSocket

- Client server arch.
- TCP only

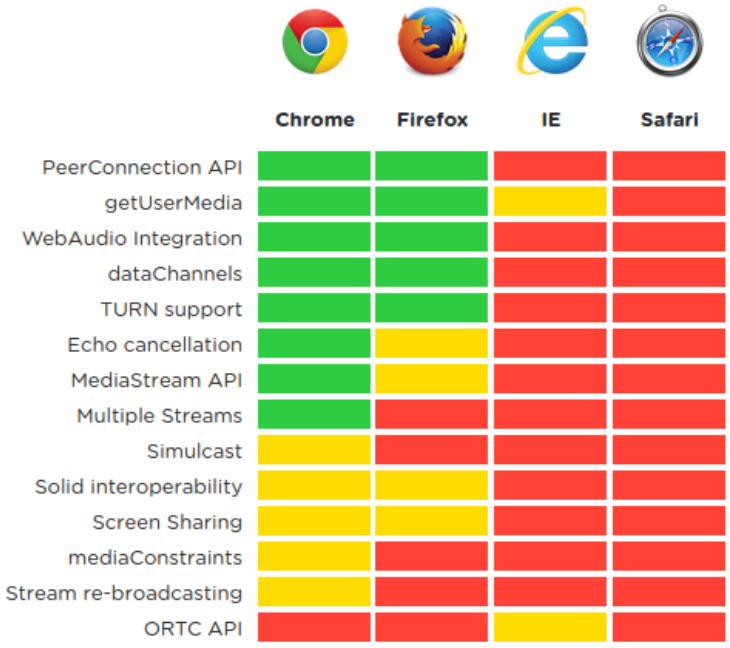


WebRTC


Signaling
Signaling
App
App
App
SessionDescription
SessionDescription
WebRTC
Browser
Browser
Media
Caller
Callee

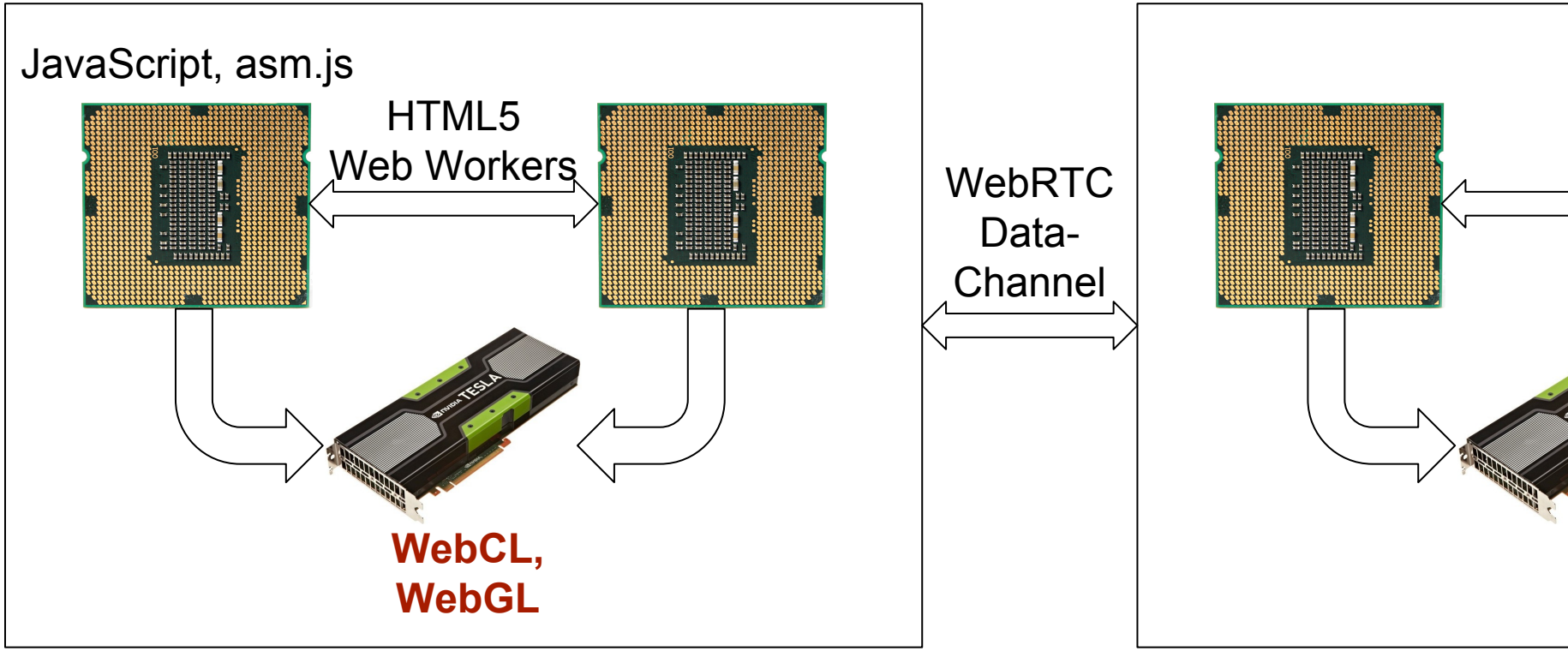
WebRTC DataChannel

	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



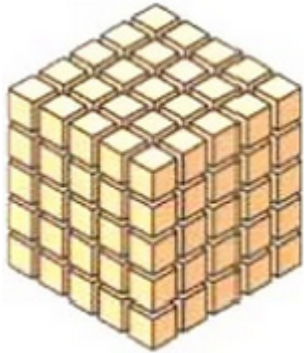
webrtc.org

WebCL, WebGL

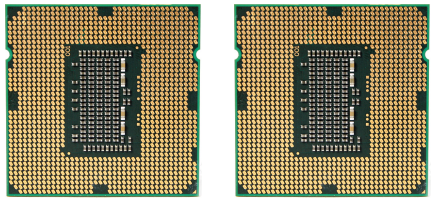

JavaScript, asm.js
HTML5
Web Workers
WebRTC
Data-
Channel
WebCL,
WebGL

GPU Computing

OpenCL/WebCL

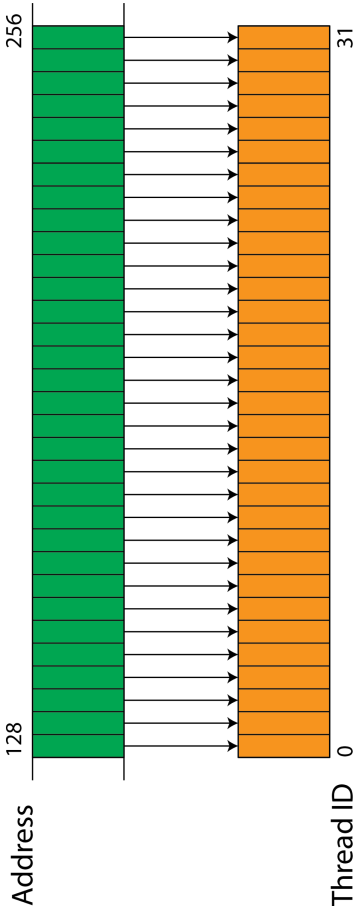


CUDA,
OpenCL/WebCL,
OpenGL/WebGL

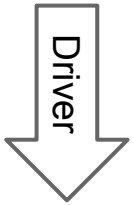


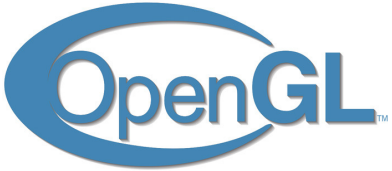
- Like OpenCL
- Hardware exposure
- IEEE 754 float
- Heterogeneous

- 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);
}
```

Soul

file:///C:/Users/Desktop/SOULbu/soul-firefox/Soul.html

fullscreen Pause Resume Quit

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	JS subset, near-native speed
Web Workers	Messages: Cloning vs. transfer
DataChannel	Signalling, TCP/UDP/SCTP
WebCL/GL CS	Specs vs. adaption

Summary




asm.js


Web Workers



DataChannel


WebCL/GL CS