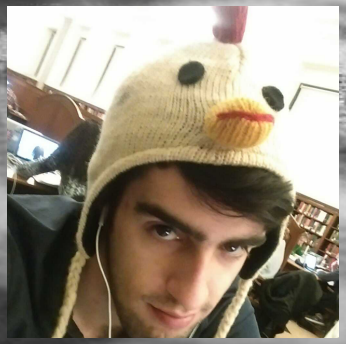
Sheets

What's Your Thread Count?

Ben Barg
Language Guru



Gabriel Blanco
Tester

Amelia Brunner
Project Manager

Ruchir Khaitan
System Architect

The Goal:

A high-level programming language for building GPU Accelerated Applications

CPU: Intel Xeon E5-2670	**GPU:** Nvidia Grid K520
8 Cores	3072 Cores
384 GB of Memory	8GB of Memory
48 GB per Core	2.6KB per Core
2.6 GHz Clock Speed	800 MHz Clock Speed

CPU and GPU found on Amazon E3 Instance

Language Features

Types	Operators	Functions	Comments
int int[] float float[]	+ - * / > < >= <= == != = .	funcs gfuncs Block	#~ block ~# ## inline

- Whitespace Delimited
- Statically Typed

Example Program

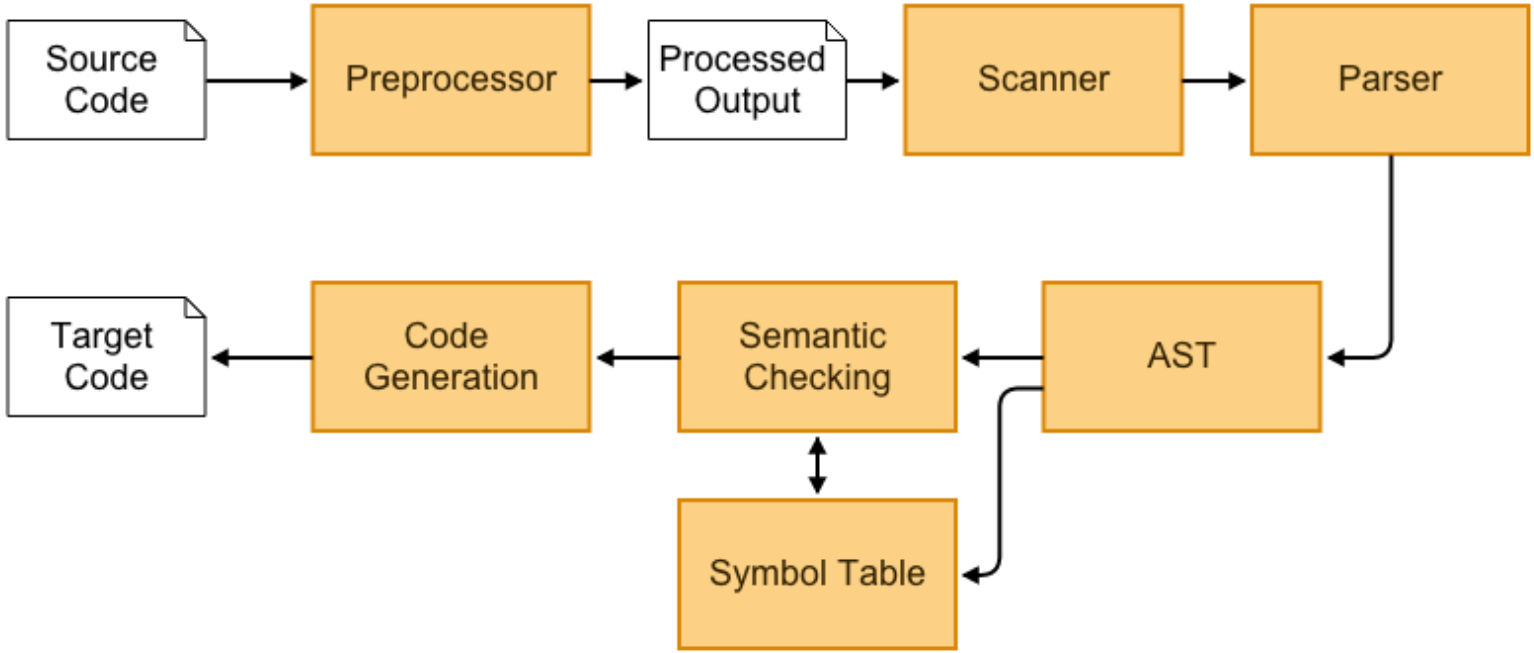
```
gfunc float[] gmultiply(float[] x, float[] y).[1]:
    for(int i=Block.start; i<Block.end; i=i+1;):
        Block.out[i] = x[i] * y[i]

func float[] snuggle():
    float[] x = [1.,2.,3.,4.,5.,6.]
    float[] y = [.5,.5,.5,.5,.5,.5]

    float[] result[6]
    return result = gmultiply(x,y)
```

OpenCL

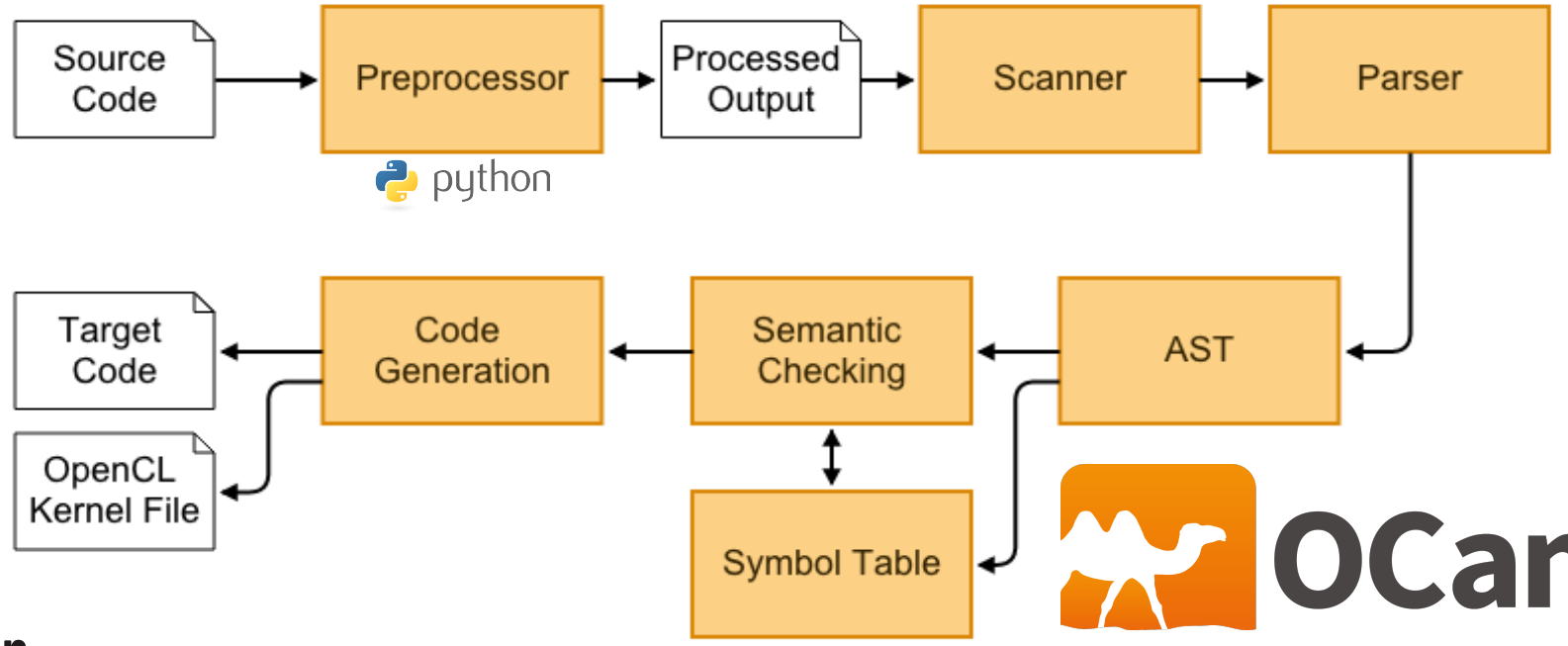
```
__kernel void image_filter(__global uchar4* src,
                           __global uchar4* dst,
                           int row_width)
{
   int x = get_global_id(0);
   int y = get_global_id(1);

   //My location in the image
   int position = x + y * row_width;

   //Read Input pixel
   uchar4 in  = src[position];

   //Convert to greyscale
   uchar out = in.x * 0.299f + in.y * 0.587f +
in.z * 0.114f;

   /*For Negative of the image*/
   //uchar4 maxpixel = (uchar4)(255,255,255,0);
   //uchar4 out = maxpixel - in;

   //Write out result to same location in
destination image
    dst[position] = (uchar4)(out, out, out, 0);

   //dst[position] = out;
}
```

```
#include"cl_util.h"

int main(int argc, char** argv)
{
  cl_device_id device_id;
  cl_context context;
  cl_kernel kernel;

  cl_mem cl_src;
  cl_mem cl_dst;
  cl_command_queue queue;
  cl_context_properties *properties = NULL;
  cl_event event;

  int w;
  int h;

  int err = CL_SUCCESS;


  cl_uint num_platforms;
  cl_platform_id clPlatformID;

  err = clGetPlatformIDs (1, &clPlatformID, NULL);
  CHK_ERROR(err, "clGetPlatformIDs");

  device_id = getDeviceId(&clPlatformID);

  //Create Context
  context = clCreateContext(properties, 1, &device_id, NULL, NULL, &err);
  CHK_ERROR(err, "clCreateContext");

   //Create Command Queue
  queue = clCreateCommandQueue(context, device_id, CL_QUEUE_PROFILING_ENABLE, &err);
  CHK_ERROR(err, "clCreateCommandQueue");

  //Query Capabilities - TBD
  int* src = readBmp("sample.bmp", &w, &h);
  int size = w*h*sizeof(int);

  int* dst = (int*)malloc(size);
  cl_src = clCreateBuffer(context, CL_MEM_USE_HOST_PTR, size, src, &err);
  CHK_ERROR(err, "clCreateBuffer source buffer");
  cl_dst = clCreateBuffer(context, CL_MEM_USE_HOST_PTR, size, dst, &err);
  CHK_ERROR(err, "clCreateBuffer destination buffer");

  kernel = getKernel(context, device_id);

   //set kernel arguments
  err = clSetKernelArg(
                    kernel,
                    0,
                    sizeof(cl_mem),
```

Compiler Architecture



Compiler Architecture



NVIDIA

OpenCL®

amazon web services™


The Demo