



Instigate

Parallel Systems Development

Design Services ◦ SW acceleration and parallelization

Introduction

Instigate provides vast spectrum of parallel programming services. Having huge experience with FPGA/ASIC, NVIDIA CUDA, ATI Stream, AMD NUMA architecture programming and more, we offer thorough consulting and R&D services to cover the whole work-flow, starting from problem classification and corresponding platform assortment to development, optimization and testing. Due to the experienced developers team and in-house methodologies/techniques, we offer our Customers reliable and efficient solutions.

Methodology

- Application exploration, problem classification
- Algorithmic level parallelization
- Target platform selection
- Target platform specific decomposition
- Implementation
- Target platform specific optimization
- Verification

Massive parallel reconfigurable architecture programming

GPGPU based acceleration

- Experience in NVIDIA CUDA and ATI Stream
- Up to 200x acceleration over CPU
- Wide range of applications
 - Financial forecasting
 - Video encoding/decoding
 - Image processing
 - E-mail Spam filter
 - Linear/Quadratic solver
 - Basic math functions
 - etc.

FPGA based HW acceleration

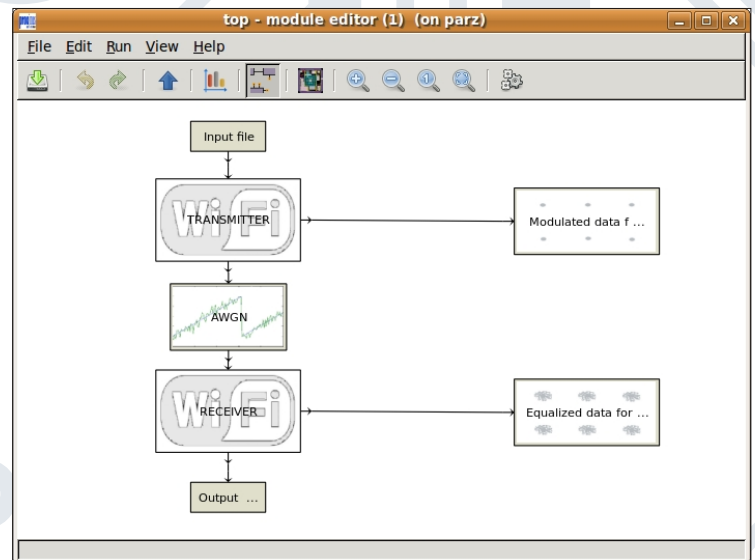
- Research of reference implementation
- Parallelization of reference implementation
- Profiling and identification of active part
- HW-SW decomposition and co-design
- Example: decomposing Viterbi decoder from Wi-Fi 802.11g and mapping to FPGA

```
#include <ostream>

// CUDA code
#define BLOCK_SIZE 16
__global__ void Muld(int* A, int* B, int ha, int wa, int wb, int* C)
{
    int size;
    int* Ad;
    size = ha * wa * sizeof(int);
    cudaMalloc((void**)&Ad, size);
    cudaMemcpy(A, A, size, cudaMemcpyHostToDevice);
    int* Bd;
    size = wa * wb * sizeof(int);
    cudaMalloc((void**)&Bd, size);
    cudaMemcpy(B, B, size, cudaMemcpyHostToDevice);
    int* Cd;
    size = ha * wb * sizeof(int);
    cudaMalloc((void**)&Cd, size);
    dim3 dimBlock(BLOCK_SIZE, BLOCK_SIZE);
    dim3 dimGrid(wB / dimBlock.x, ha / dimBlock.y);
    Muld<<<dimGrid, dimBlock>>>(Ad, Bd, wa, wb, Cd);
    cudaMemcpy(C, Cd, size, cudaMemcpyDeviceToHost);
    cudaFree(Ad);
    cudaFree(Bd);
}
```

TCL Log

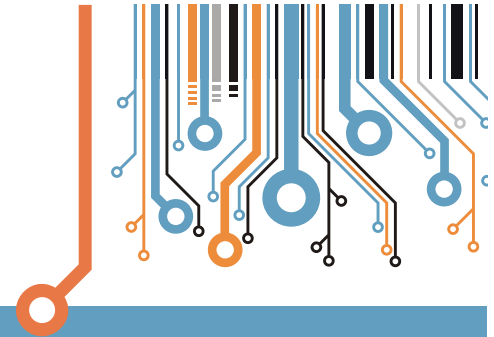
```
framework::deselect -object [cuda_implementation::get -name "cuda_matrix_mult" -project [project::get]]
framework::select -object [cuda_implementation::get -name "cuda_matrix_mult" -project [project::get]]
proximus::preferences
framework::select -object [cuda_implementation::get -name "cuda_matrix_mult" -project [project::get]]
framework::deselect -object [cuda_implementation::get -name "cuda_matrix_mult" -project [project::get]]
framework::select -object [cuda_implementation::get -name "cuda_matrix_mult" -project [project::get]]
# framework::show_property_editor -object [cuda_implementation::get -name "cuda_matrix_mult" -project [project::get]]
```





Instigate

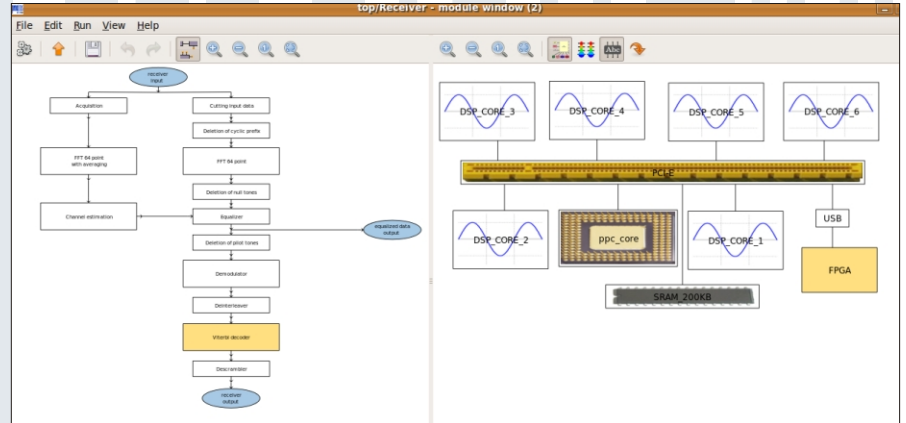
Parallel Systems Development



Design Services ◦ SW acceleration and parallelization

Proprietary multi-cpu device programming

- Complex real-life designs parallelization and acceleration
 - FFTs (with 64 - 256 M transform size)
 - Wi-Fi 802.11 g (PHY layer, 54 Mb/s)
- System Level modeling
- Model level parallelization
- Target platform virtual design
- Simulation and bottlenecks detection
- Platform based optimization
- Export to architecture specific proprietary language



Multi-core NUMA architecture CPU programming

- Efficient use of multi-core/cpu Non-Uniform-Memory-Architecture capabilities
- Reducing the number of processing units competing for access to a shared memory bus
- Algorithm specific scheduling on available computing units
- Custom memory distribution considering neighborhood between cores and memory units
- Used NUMA C library (libnuma)
- Example: accelerating TLM simulation engine using NUMA benefits

