Radar Signal Processing on GPU's and performance comparison with vector processors

Exploring the Power of GPU's for real-time Radar Signal Processing

Peter Joseph Basil Morris, Bibhabasu Mondal, Saikat Roy Choudhury & Debasish Deb
Defense Research & Development Organisation (DRDO), India

Contact Information:
Electronics & Radar Development Establishment
Defense Research & Development Organisation
Bangalore,Karnataka,India
Email: peter.jbm@lrde.drdo.in

Abstract

We investigate the computing capabilities of GPU's for radar signal processing applications through the realisation of a radar signal processor on a GPU leveraging the inherent parallelization offered by the radar signal processing algorithms and the extensive computing capability of the GPU. A set of typical radar algorithms have been selected for the implementation. These algorithms includes a rich set of compute intensive operations viz:- matrix transposes, corner turns and Fast Fourier Transforms (FFT). The performance of these algorithms was bench-marked against the same application running on state of the art multiprocessor hardware with vector signal processing capabilities. The work clearly brings out the fact that an optimized implementation of the algorithms on GPU's has lead to a drastic speed-up in the computation time with respect to implementation of the same on a typical multi-core processor architecture. The performance results obtained on GPU's has been extremely promising which has ushered in the use of GPU's for a variety of radar signal processing applications. In-short, the work attempts to set the stage for newer fields of radar signal processing research on GPU based computing.

Introduction

Radar signal processing in itself signifies a complex signal processing challenge involving a plethora of advanced signal processing techniques and intense computational effort. Though the former has matured itself the latter still presents a profound challenge. It is no surprise that the computational load of modern radar signal processors is tremendous. Moreover, in most applications real time processing of radar data is required with the constraints of space ever haunting.

Modern radar signal processing benefits from parallel processing architectures due to the inherent parallel nature of the algorithms. The hardware multi-threading capabilities and the Single Instruction Multiple Thread (SIMT) model have made GPU's all the more promising for the next generation high performance radar signal processing applications due to the incredible levels of performance offered by GPU's on data parallel operations. Since SIMD execution model operates on multiple data, data parallelism is the key concept in leveraging the power of GPU's and radar data being inherently parallel offers much better scope for the deployment of the same on GPU's. However maximum performance on GPU calls for creative algorithm design. The work presents the implementation of radar signal processor chain on an nVidia based GPU,focusing on the real-world benefits of GPU's for radar signal processing applications.

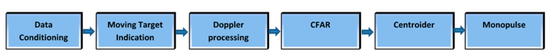

Figure 1: Algorithms Selected for Implementation

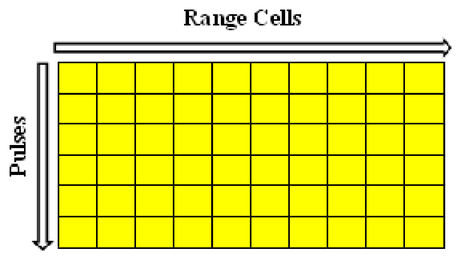

Figure 2: Radar Data Matrix

The radar signal processing algorithms implemented and tested on GPU includes (but not limited to) shown in Figure.1 are well studied and explained [3]. The cluster of high rate data samples from a single radar pulse may be viewed as being stored in a single row of a structure called the radar data matrix as shown in Figure.2. The cluster of samples taken from the next pulse forms the next row of the matrix and so on. This offers a unique advantage for parallel processing since each pulse data (row) of the data matrix could be processed in parallel independent of the other pulse data.

Implementation on GPU's

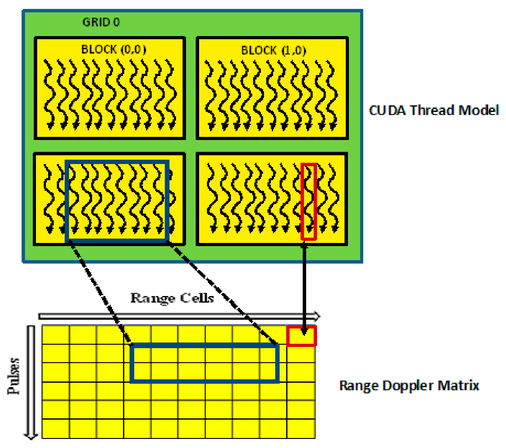

Figure 3: Mapping of GPU Threads to the Range Doppler Matrix

CUDA has been employed for the implementation of the various algorithms onto the GPU hardware. The dimension along which the radar data is processed is of prime concern to obtain the optimal performance for any hardware architecture. Radar data inherently being two dimensional (i.e., Range and Pulse dimension), requires the data to be re-arranged along the required processing dimension for each specific algorithm. The current implementation fixes the block size to 256 threads for better optimization [2]. The mapping of CUDA threads to Range Doppler matrix is shown in Figure.3. The kernel executes computations on a given dimension of the data matrix in parallel based on the processing dimension of the algorithm. Each thread executes computations for a single range bin or pulse as the case may be. To prevent memory latency due to host-device memory transfer, a single memory transfer to the device memory is performed at the start while the rest of the computations are carried out at the device with only the final results transferred back to the global memory. Several optimisation strategies were employed to achieve maximum efficiency of computation. Threads were identified corresponding to the computationally intensive algorithms (e.g.,Data Conversions, FFT, Complex Transpose etc) and kernels were written to execute these threads. Each kernel code executes actions of a single thread only. Prudent memory usage and data transfers were adopted to reduce the inherent memory latency due to host-device memory transfers and the computational density of the GPU's were leveraged by the reduced usage of memory accesses.

Results

The entire algorithms were tested using an nVidia QuadroFX 3800 on a workstation equipped with Intel Xeon X5675 six core processor operating at 3.06GHz with 48GB RAM. The CUDA kernels were tested for two Range-Doppler data matrices of sizes (530 x 258) and (1024 x 258) complex samples and the results compared with the same algorithms implemented on a standard vectorized multiprocessor hardware based on PowerPC architecture. The results are tabulated in Tables I, II.

Table 1: TIMING OF CUDA KERNELS FOR 530 X 258 COMPLEX SAMPLES

Algorithms	GPU Time(us)	PowerPC Time(us)	SpeedUp
Data Conditioning	0.37	9.125	24
MTI	0.098	1.655	16
Doppler Processing	0.08	5.5175	69
CFAR	0.46	29.18	63
Peak Filtering	1.4	61.68	44
Centroider	0.6	14.64	24
Monopulse Processing	0.097	7.7	79

Table 2: TIMING OF CUDA KERNELS FOR 1024 X 258 COMPLEX SAMPLES

Algorithms	GPU Time(us)	PowerPC Time(us)	SpeedUp
Data Conditioning	0.63	18.03	28
MTI	0.25	4.2	17
Doppler Processing	0.16	12.2	76
CFAR	0.7	54.9	78
Peak Filtering	2.4	110	46
Centroider	0.7	25	36
Monopulse Processing	0.18	14.7	82

Conclusions

The poster tries to demonstrate GPU computing and CUDA programming environment by bringing forward the step-by-step implementation of a radar signal processing application. The application was specifically chosen due to the computationally intensive challenges and time constraints it poses in exploiting the benefits of GPU. The results obtained clearly validates that an optimized parallel implementation has lead to a drastic reduction in computation time with respect to implementation of the same on a typical multi-core processor architecture. The performance achievements obtained clearly demonstrate the immense potential of GPU's for developing high performance radar signal processing applications.

Forthcoming Research

The speedup obtained in this application has spawned a great deal of interest in a variety of areas in radar signal processing research. Extensive work is being carried out in the use of GPU's for a number of radar applications including IQ data generation for Synthetic Aperture Radar (SAR), signal processing for the Strip-map mode of SAR , signal processing for the Ground Moving Target Indication (GMTI) mode of SAR. A plethora of SAR signal processing algorithms :- Global Back-projection, Local back-projection, Coherent Change Detection are being bench-marked with GPU's and have already exhibited considerable speedups. In-short the work has set the stage for GPU's to be used as the probable hardware candidate for the next generation high performance radar signal processing applications.

References

[1] *CUDA,CuFFT Library*.

[2] *NVidia CUDA Programming Guide*.

[3] Mark A Richards, *Fundamentals of Radar Signal Processing*, 6th Edn,New Delhi,India:Tata Mc-Graw Hill,2005.