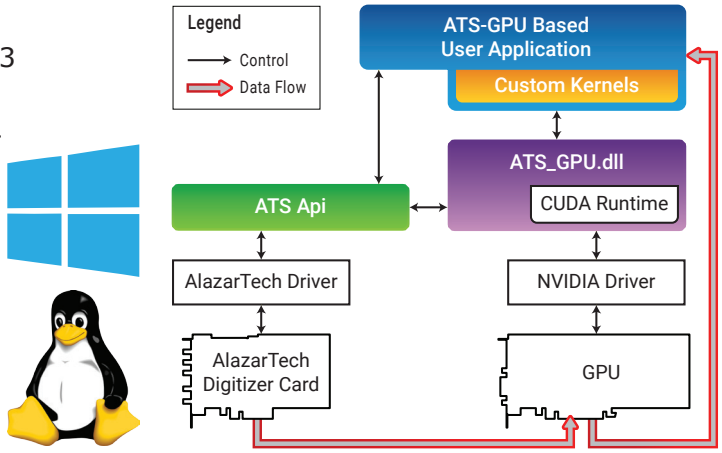
# ATS-GPU
## Real Time Signal Processing Software

- Transfer A/D data to GPU at high speed
- Up to 4 GB/s transfer rate for PCIe Gen 3 digitizer boards
- Supports CUDA compute capability 2.0+
- Designed to work with AlazarTech PCI Express waveform digitizers
- Optional OCT Signal Processing Module includes:
  - ◇ Very High-Speed Floating Point FFT
  - ◇ Dispersion Compensation and Windowing Functions
- Compatible with Windows & Linux




| Product | GPU Compatibility | Operating System | Throughput to GPU | †FFT Length | †Max. FFTs Per Second |
|---|---|---|---|---|---|
| ATS-GPU | CUDA compute capability 2.0+ | Windows (7/8/10) & Linux | Up to 4 GB/s | Up to 2 M Points | 1,000,000 (2048 pt FFTs, see benchmark table below for more details) |

### Overview

ATS-GPU is a software library developed by AlazarTech to allow users to do real-time data transfer from its PCI Express waveform digitizers to a CUDA-enabled Graphical Processing Unit (GPU) at rates up to 4 GB/s.

Modern GPUs include very powerful processing units and a very high speed graphical memory bus. This combination makes them perfectly suited for signal processing applications.

Unfortunately, it is not easy for other hardware devices, such as waveform digitizers, to DMA data directly to the GPU's on-board memory.

This forces users to manually copy data from the buffer returned by the waveform digitizer to the GPU. This copying process is relatively slow and causes the overall data throughput to be drastically reduced.

ATS-GPU solves this problem by transferring data to the GPU using highly optimized software routines implemented at the kernel level of the operating system and assisted by hardware. Rates up to 4 GB/s have been achieved.

The optional OCT Signal Processing module for ATS-GPU contains floating point FFT routines that have also been optimized to provide the maximum number of FFTs per second. Kernel code running on the GPU can do zero-padding, apply a windowing function, do a floating point FFT, calculate the amplitude and convert the result to a log scale. It is also possible to output phase information.

ATS-GPU includes an example program that demonstrates how to use the ATS-GPU library to transfer data from a waveform digitizer to a GPU. The example also shows how to do simple data processing on the GPU using CUDA kernels, and how to transfer the processed data back to host memory (RAM). Users can use this example program as a starting point to create their own kernels to do GPU-based DSP.

### GPU-Based Signal Processing

Graphical Processing Units (GPUs) were originally designed for rendering high quality video for gaming applications, which required being able to perform massive amount of real-time calculations. The highly parallel architecture of modern GPUs also makes them an ideal platform for digital signal processing (DSP) and high performance computing (HPC) systems.

In the past, complex real-time signal processing, such as FFT, correlation, FIR filtering etc. could only be achieved using dedicated DSP processors or by implementing the algorithms inside and FPGA or an ASIC. All these methods are non-trivial, expensive, time consuming and require highly specialized engineering skills.

Using GPUs, users can implement any algorithm that can be parallelized in a GPU using well known software techniques and gain a better than 10-fold improvement over CPU based signal processing. The reason why GPUs perform so well for DSP applications is that they contain hundreds of processing cores (kernels) running in parallel, while sharing a very high speed graphical memory bank.

### Benchmarks

An ATS9373 in an X99 Deluxe machine using an Intel i7 5930K @ 3.5 GHz, 64GB DDR4 and NVIDIA GeForce GTX Titan X GPU had the following benchmarks:

| FFT Length | Single Channel Max. | | Dual Channel Max. | |
|---|---|---|---|---|
| | NPT Trigger Repeat Rate (sampled @2.4 GS/s) | Continuous Sample Rate | NPT Trigger Repeat Rate (sampled @1.2 GS/s) | Continuous Sample Rate |
| 1024 | 2000 kHz | 2 GS/s | 1000 kHz | 1 GS/s |
| 1536* | 1200 kHz | -- | 550 kHz | -- |
| 2048 | 1000 kHz | 2 GS/s | 500 kHz | 1 GS/s |
| 4096 | 525 kHz | 2 GS/s | 275 kHz | 1 GS/s |
| 6144* | 300 kHz | -- | 130 kHz | -- |
| 8192 | 275 kHz | 2 GS/s | 125 kHz | 1 GS/s |
| 16384 | 125 kHz | 2 GS/s | 55 kHz | 1 GS/s |
| 32768 | 65 kHz | 2 GS/s | 28 kHz | 1 GS/s |
| 1048576 | 1.7 kHz | 1.8 GS/s | 0.85 kHz | 0.8 GS/s |

† Available only with optional OCT Signal Processing Module

* Zero-padded to the next power of 2

Tests using the same X99 Deluxe system produced comparable results with both the NVIDIA Quadro P5000 GPU and Asus GTX980 GPU.

### A Typical ATS-GPU Application

A typical user application that uses ATS-GPU consists of the following minimum sections:

1) User application sets up waveform digitizer hardware (sample rate, input range, trigger parameters etc.).
2) User application allocates buffers and sets up the GPU.
3) User application starts data capture.
   ATS-GPU starts streaming data to GPU, one buffer at a time.
4) GPU kernels do the following:
   - Process a buffer;
   - Copy result buffer to user memory;
   - Get next buffer, and repeat.
5) User application running on CPU consumes result buffer.
   For highest performance, make sure data consumption is faster than the rate at which result buffers are supplied by GPU kernels.
6) This continues until the application has to be closed.

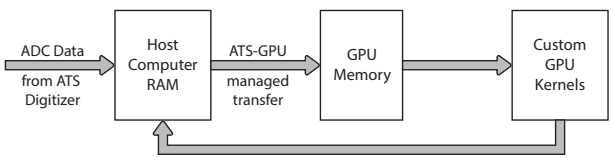
### ATS-GPU and CUDA Runtime Library

ATS-GPU is shipped with a specific version of CUDA runtime library and links statically to it.

Programmers are allowed to use a different version of CUDA runtime library for their custom kernel code. NVIDIA guarantees that the two versions of CUDA runtime libraries will be interoperable.

Note: ATS-GPU only supports Windows versions and Linux distributions that are supported by NVIDIA's CUDA Toolkit. 32-bit operating system support is also similarly limited by NVIDIA. In particular, ATS-GPU optional OCT Signal Processing library cannot be built as a 32-bit library. We currently use CUDA toolkit 8.0, older versions are untested.

### ATS-GPU Data Flow




**ATS-GPU Data Flow**

ATS-GPU is supplied with an example user application in source code. The application includes GPU kernels that use ATS-GPU to receive data, do very simple signal processing (data inversion), and copy the processed (inverted) data back to a user buffer. All this is done at the highest possible data transfer rate.

Programmers can replace the data inversion code with application-specific signal processing kernels to develop custom applications.

### Performance Dependencies

Since the host CPU is involved in moving data to and from the GPU and in scheduling GPU kernels, CPU speed and motherboard's memory bandwidth can have a significant impact on the overall performance.

The optional OCT Signal Processing module was used to benchmark performance.

On an Asus X99 Deluxe motherboard that uses an Intel i7 5930K 3.5 GHz CPU and DDR4 memory (64 GB RAM), a combination of the ATS9373 and NVIDIA GeForce GTX Titan X (Maxwell) GPU was able to do a 2048 point FFT at a rate of 1000 kHz.

An older P9X79 machine with an Intel Xeon E5-2603 v2 CPU and 16 GB DDR3 1333 MHz memory performed approximately 35% slower.

Complexity of the kernel code running on the GPU can have a significant impact on the overall performance. Users should optimize their code to take advantage of the GPU's high speed memory.

### Computer Power Supply

GPUs are power hungry. Even consumer-grade models such as Asus GTX980 require a power supply that can provide at least 500 Watts of power. As such, users must make sure their computer's power supply has sufficient capacity.

### Compatible GPUs

ATS-GPU is been designed to be compatible with all compute capability 2.0 or higher CUDA-enabled GPUs. NVIDIA has deprecated compute capability 2.0 and 2.1 but they are still supported. Testing was done using:

- Asus GTX980
- NVIDIA GeForce GTX Titan X
- NVIDIA Quadro P5000
- NVIDIA Tesla P100
- NVIDIA Quadro 600
- NVIDIA GTX670

It should be noted that ATS-GPU supports only one GPU at a time. If you have multiple GPUs installed in your computer, ATS-GPU will let you select one of them for use.

### Deprecation of OpenCL-Based ATS-GPU

Due to lack of demand, ATS-GPU v.2.x was the last version to support OpenCL. ATS-GPU v3.5 programming manual includes instructions on how you can modify your existing programs to use the CUDA-based ATS-GPU.

### Data Throughput to GPU

The data transfer rate to GPU is dependent on the generation of PCI Express digitizer board used:

| Generation | Transfer Rate |
|---|---|
| Gen 3: ATS9373 | Up to 4 GB/s |
| Gen 2: ATS9360, ATS9416 | Up to 3.0 GB/s |
| Gen 1: ATS9870, ATS9350, ATS9351, ATS9625, ATS9626, ATS9440, ATS9462* | Up to 1.6 GB/s *(720 MB/s on ATS9462) |

### Compatible Waveform Digitizers

All AlazarTech PCI Express waveform digitizers are compatible with ATS-GPU. Only single-board configurations are supported at this time.

AlazarTech's PCI bus waveform digitizers are not supported, as the host CPU is more than capable of handling data rates generated by PCI bus boards.

ATS-GPU cannot directly be interfaced with non-AlazarTech waveform digitizers. However, users can always capture data from non-AlazarTech digitizers and pass it to the GPU using the software validation data path.

Note that this will probably not provide optimal throughput. Also note that AlazarTech will not support this type of software development.

## Software Licensing Policy

Users are allowed to freely distribute the ATS-GPU library as long as there is an AlazarTech PCI Express waveform digitizer present in the same computer. If an AlazarTech PCI Express waveform digitizer is not present in the computer, users must purchase a separate license for each computer on which ATS-GPU is installed.

In no case is the user allowed to distribute or share the source code of ATS-GPU with other users.

## Annual Subscriptions

The purchase of an ATS-GPU subscription provides customers with the following for a period of 1 year on ATS-GPU:

- Download ATS-GPU updates from the AlazarTech web site;
- Receive new example programs as they become available;
- Receive technical support on ATS-GPU.

Additional add-on modules for ATS-GPU, such as the *OCT Signal Processing Module* are not covered by the annual subscription, i.e. holders of an annual subscription will have to purchase subscriptions for additional modules separately.

The purchase of an ATS-GPU OCT Signal Processing Module subscription provides customers with the following for a period of 1 year on the ATS-GPU OCT Signal Processing Module:

- Download OCT Signal Processing Module updates from the AlazarTech web site;
- Receive enhanced/improved OCT Signal Processing functionality as they become available;
- Receive technical support on the OCT Signal Processing Module.

Note that support is provided for product bugs, and not for writing custom GPU kernels or for learning GPU programming.

## Writing Custom GPU Kernels

ATS-GPU includes an example program in C/C++ source code, which implements very simple GPU kernels that invert data and write it back to a buffer in computer memory.

Users who need to write their own kernels should start with the included source code, add CUDA code in the appropriate place, and compile their libraries.

The example program is provided with a Visual Studio project and a CMake build file. We use more recent C++ features, and Visual Studio 2015 and later is required. On Linux, a C++11 compiler is required and can be accessed on older distributions via a devtoolset (RHEL and CentOS 6 for example).

Writing, testing, and debugging modified kernels will be the sole responsibility of the user and AlazarTech will not be responsible for assisting the user with such custom modifications.

Users must have expert programming knowledge of CUDA development in order to customize ATS-GPU kernels.

### ATS-GPU main API functions

```
ATS_GPU_SetCUDAComputeDevice
ATS_GPU_Setup
ATS_GPU_AllocBuffer
ATS_GPU_PostBuffer
ATS_GPU_StartCapture
ATS_GPU_GetBuffer
ATS_GPU_AbortCapture
ATS_GPU_FreeBuffer
```

## ORDERING INFORMATION

| | | |
|---|---|---|
| ATS-GPU: | 1 Year Subscription | ATSGPU-001 |
| ATS-GPU: | OCT Signal Processing Module 1 Year Subscription | ATSGPU-101 |
| ATS-GPU: | OCT Signal Processing Module Source Code | ATSGPU-104 |

**Manufactured By:**

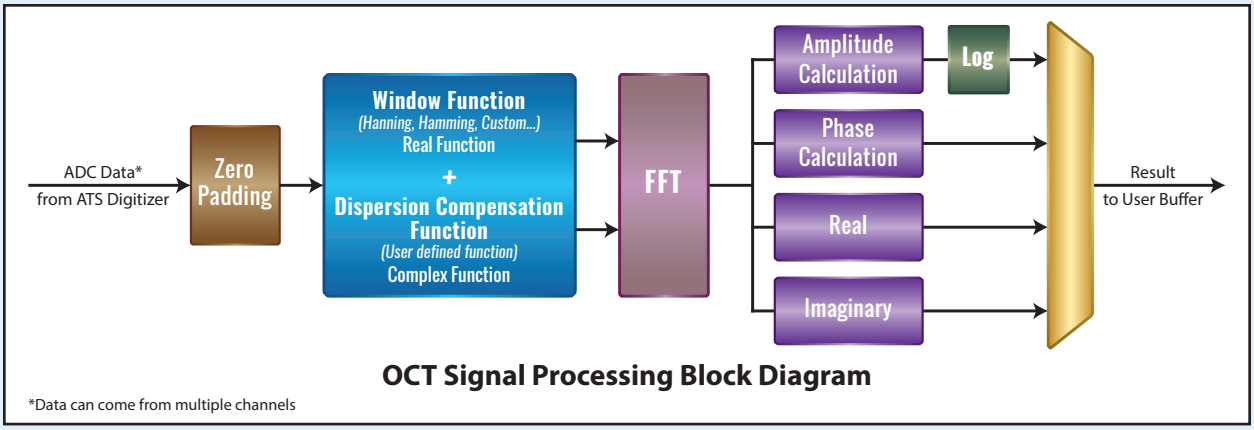
**Alazar Technologies, Inc.**

6600 TRANS-CANADA HIGHWAY, SUITE 310
POINTE-CLAIRE, QC, CANADA H9R 4S2

TOLL FREE: 1-877-7-ALAZAR OR 1-877-725-2927
TEL: (514) 426-4899 FAX: (514) 426-2723

E-MAIL: info@alazartech.com

# ATS-GPU
## Real Time Signal Processing Software




**OCT Signal Processing Block Diagram**

*Data can come from multiple channels

### Optional OCT Signal Processing module

Customers can purchase an optional OCT Signal Processing module that provides very high speed floating point FFT capability for data acquired by AlazarTech's PCI Express waveform digitizers or for user-supplied data.

Users can use example programs in C/C++, Python, LabVIEW, or MATLAB to set-up the waveform digitizer parameters, set-up FFT parameters in the GPU, do the acquisition, and receive the FFT result buffer.

C/C++ example programs are provided with Visual Studio projects and CMake build files. Python code is tested under Python 2.7 and 3.6. LabVIEW 2009 or newer is necessary to use LabVIEW example codes. MATLAB code is developed under MATLAB 2015A, but is expected to work with most MATLAB versions.

Waveform digitizer data is transferred to the GPU in a buffer that will contain many records. This number, RecordsPerBuffer, is specified by the user. Users should make sure that they choose this number such that the buffer size is in the order of 1 MByte or larger. Smaller buffers can reduce overall data throughput.

For software validation purposes, ATS-GPU OCT Signal Processing module allows the GPU to operate on user-supplied data. It should be noted that the overall throughput may be significantly reduced.

If the number of samples per record is not a power-of-2, ATS-GPU FFT will perform zero-padding to the next power of 2. It will then apply a complex windowing function, do a single-precision floating point FFT, calculate the amplitude and phase, and convert the amplitude to logarithmic values.

### Very Long FFTs

For some applications, it is necessary to perform very long FFTs (e.g. one million points).

Even if a waveform digitizer has an on-board FPGA, such very long FFTs do not fit inside an FPGA due to resource limitations of the FPGA.

With the optional OCT Signal Processing module, ATS-GPU is fully capable of calculating such very long FFTs. Our benchmarks using Intel i7 5930K CPU and NVIDIA GeForce GTX Titan X GPU have shown that ATS-GPU is capable of doing 1800 one million point FFTs per second in single channel mode (keep up with sample rate of up to 1800 MS/s). Even longer FFTs are possible. We have not tested the limits of FFT length with ATS-GPU.

### Using ATS-GPU on File Based Data

In many circumstances, users have previously captured raw data on file that they would like to process using a GPU.

ATS-GPU provides example programs in C/C++, Python, MATLAB, and LabVIEW that show how to read data from file, format it according to the GPU's requirements, and transfer it to a GPU for FFT processing. This is done using the software validation datapath.

### Zero Padding

If the number of samples per record (A-scan) is not a power of 2, the OCT Signal Processing Module will perform zero-padding to the closest power of 2 before doing further signal processing.

### Dispersion Compensation Function

Dispersion compensation is an essential part of any OCT signal processing system. ATS-GPU Optional OCT Signal Processing Module allows users to multiply the zero-padded data with a user-specified Dispersion Compensation Function (DCF). The DCF is a complex function.

### Windowing Function

The windowing function in the Optional OCT Signal Processing Module is used to ensure that there are no discontinuities in the FFT. Note that the length of the window function should be the same as the length of the A-Scan, e.g. if the A-scan is 1536 points long, the window function should also be 1536 points long, even though the FFT length will be 2048.

### Amplitude and Phase Output

The FFT algorithm implemented in OCT Signal Processing Module is capable of calculating both amplitude and phase outputs. All outputs are provided as single-precision floating point data (32 bits per data point).

### Source Code License

For users who want to customize the OCT Signal Processing Module for their own use, a source code license is available. A Non-Disclosure Agreement must be executed before source code can be released. Note that the source code (including GPU kernels) are provided on an as-is basis and are meant to be used by expert level GPU programmers. In other words, AlazarTech will not be responsible for explaining to the user what the code does.