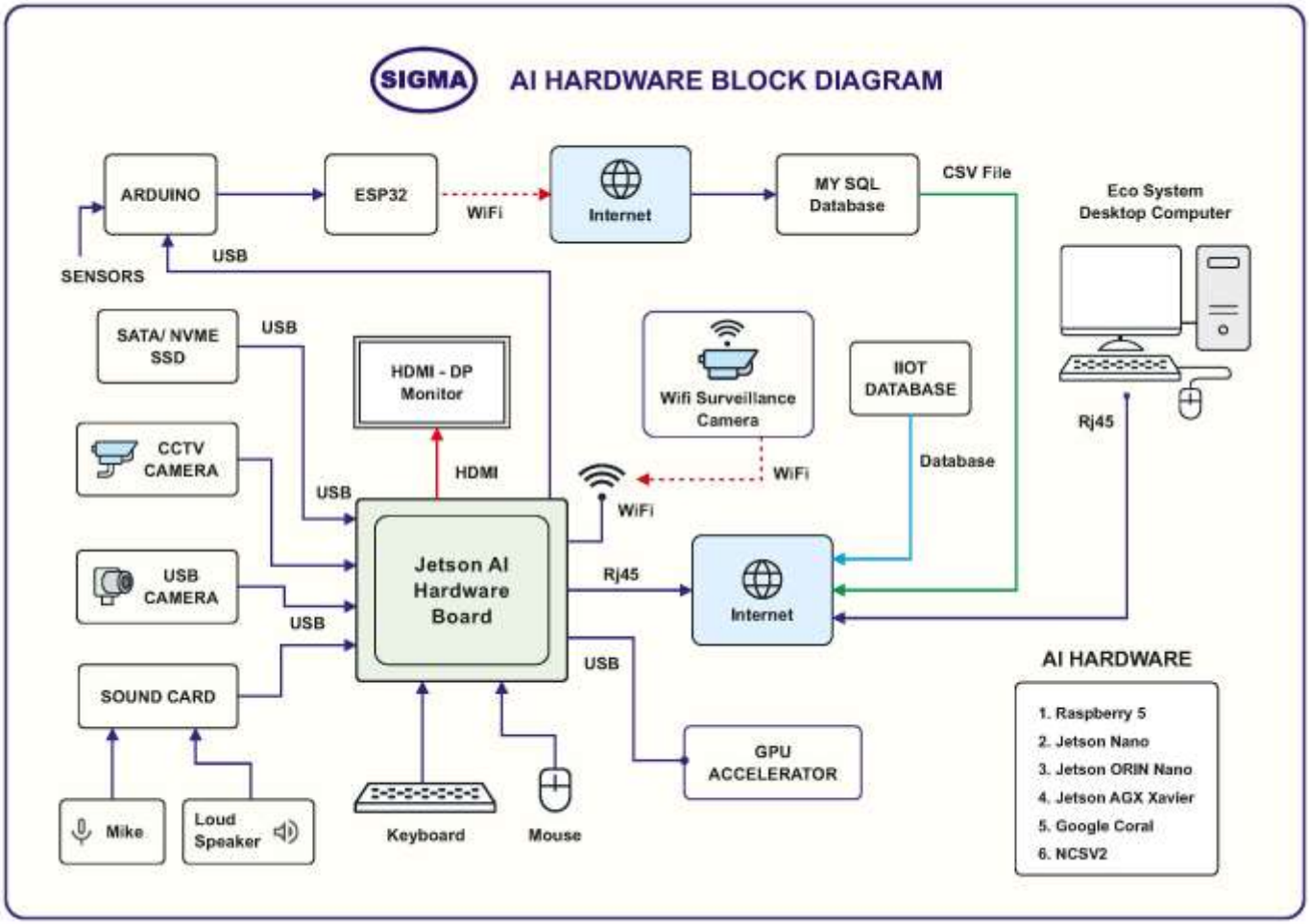
BASIC IMAGE AND VIDEO PROCESSING LAB

MODEL-AI-IMAGEVIDEO100

SPECIFICATIONS



This trainer has been designed with a view to provide practical and experimental knowledge of Machine Learning and Deep Learning Technology using Medium speed GPU AI Board.

SPECIFICATIONS

A. Main Specs

1. Following Parts and Modules are assembled on Single PCB of size - 18 Inch x 15 Inch.
2. The complete circuit diagram is screen printed on component side of the PCB with circuit and Parts at the same place.
3. The PCB with components on front side is fitted in elegant wooden box having lock and key arrangement.
4. Modules and Parts should be removable without desodlering for easy repair / replacement
5. The acrylic cover is fitted on PCB to safeguard main parts.

B. Microcontroller Board

1. Jetson Orin Nano Microcontroller Board
2. AI Performance : 40 TOPS
3. GPU : 1024-Core NVIDIA Ampere Architecture GPU with 32 Tensor Cores
4. GPU Frequency : 625 MHz
5. CPU : 6-Core Arm® Cortex®-A78AE v8.2 64-bit CPU 1.5MB L2 + 4MB L3
6. CPU Frequency : 1.5 GHz
7. Memory : 8 GB 128-bit LPDDR5 68 GB/s
8. Storage : SD Card Slot & External NVMe via M.2 Key M
9. Video Encode : 1080p30 Supported via CPU 1-2 Cores with Software
10. Video Decode : 1x 4K60 (H.265), 2x 4K30 (H.265),
11. 5x 1080p60 (H.265), 11x 1080p30 (H.265)
12. Camera Connectors : 2x MIPI CSI-2 - 22-pin Camera Connectors
13. Ethernet : Gigabit Ethernet, M.2 Key E
14. Display Port : 1x Display Port - DP 1.2 (+MST) Connector
15. USB : 4 x USB Type-A 3.2 Gen2, USBC- Supports Recovery Mode USB Type-C connector for UFP
16. PCIe : M.2 Key M slot with x4 PCIe Gen3
    M.2 Key M slot with x2 PCIe Gen3
    M.2 Key E slot
17. GPIO : 20x2 GPIO Interface Header 12-pin button Header
    GPIO, I2C, I2S, SPI, UART, PWM

18. Fan Header : Fan with 4 Pin Fan Header
19. SD Card Slot : microSD slot
20. DC power Jack : 19V DC, 2.4A Power Supply - Barrel Type 2.1 mm
21. Hard Disk : 128GB nVME SSD with pre-loaded Linux OS
22. With pre-configured image which is again used as edge computing setup With Tools like OpenCV, Tensor RT, CUDA

C. Peripherals Hardware

1. USB Camera : Logitech 270
2. Mouse: Logitech USB Mouse
3. Keyboard : TVSE Gold USB Keyboard
4. Monitor : 15.6 Inch LED HDMI and DP Port
5. Sound Card with Stereo Loudspeaker with Box.
6. 64 GB Micro SD Card
7. 128GB SSD
8. 4 Port USB Hub
9. 4 Port Ethernet Switch

D. Accessories:

1. All Cables and Adaptors
2. Pen Drive : 16 GB with All Codes and Soft copy of Manual
3. E-Books for AI, ML, DL Subject : 100 Nos. in PDF Format
4. Mp4 Video for AI, ML, DL Subject : 100 Nos
5. Online Cloud/Server Services : For 2 Years on Cloud Server
6. Live Training at College : For 2 Days for 4 Hours per Day
7. After Sale Training support : By Online Zoom Meeting or By Whatsapp Video Call

E. Software and Programs

1. Cuda
2. cuDNN
3. Tensor RT
4. Torch
5. PyTorch
6. TorchVision
7. Tensor Flow
8. Keras
9. Caffe
10. Caffe2
11. OpenCV
12. Computer Vision - CV
13. Yolov5
14. Open GL
15. Vulkan
16. NVidia Vision Works
17. DL4J
18. NVIDIA DIGITS
19. Vowpal Wabbit
20. Vulkan
21. Xgboost
22. Theano
23. CNTK
24. Kubernet
25. Dockers
26. Containers

EXPERIMENTS

1. Computer Vision – OpenCV Experiments

1. Write a program to display Hello World.
2. Write a program to Read image apply sobel filter and display output image.
3. Write a program to Read image apply sobel filter and display output image.
4. Write a program to Read Video file and display video.
5. Write a program to Read Video file apply sobel filter and write xvid video file.
6. Write a program to capture image from USB Webcam, apply sobel filter to it and write to image file
7. Write a program to capture video from USB Webcam, apply sobel filter to it and write to xvid video file.
8. Write a program to perform basic operation like resize over an image.
9. Write a program to perform Simple Canny Edge filter over an image.
10. Write a program to perform Canny Edge filter using blur technique to get desired result.
11. Write a program to perform simple feature detection using OrbFeatureDetector over a video.
12. Write a program to perform optical flow over feature detection to track features and show the tracking over a video.
13. Write a program to perform object detection by comparing unique points of an object to a video and find the object.
14. Write a program to perform object detection and match unique descriptors of object with video and draw lines to show match.
15. Write a program to perform object detection using matching with desired object and put a box around if the object is near to what is described.
16. Write a program to perform Face detection using Cascade Classifier.
17. Write a program to perform Face detection using Cascade Classifier with Histogram.
18. Write a program to perform Face detection using alternate Cascade Classifier profile.
19. Write a program to perform Background and foreground segmentation using CPU.
20. Write a program to perform Background and foreground segmentation using GPU.
21. Write a program to perform laplace point edge detection using USB Webcam capture.
22. Write a program to perform Houghlines detection over an art image using both CPU and GPU.
23. Write a program to perform grabcut segmentation over selected section.

2. Computer Graphics – OpenGL Experiments

1. Write a program to demonstrate generation of large number of slightly varying objects with bindless rendering.
2. Write a program to demonstrate blooming effect on rendered surfaces making it glow.
3. Write a program to demonstrate access to GL textures using both reading and writing to image.
4. Write a program to demonstrate particle expansion by accessing vertex shaders in parallel.
5. Write a program to demonstrate water simulation by using compute shaders.
6. Write a program to demonstrate use of vertex shaders to animate particles and write back result into vertex buffer.
7. Write a program to demonstrate use of high performance and quality approximation of anti-aliasing.
8. Write a program to demonstrate High Dynamic Range (HDR) imaging.
9. Write a program to implement instancing to tessellate objects in real time.
10. Write a program to implement instancing to accelerate drawing of simillar objects
11. Write a program to demonstrate multi-pass filtering for motion blur of fast moving objects.
12. Write a program to demonstrate motion blur using 2D multi-pass filter.
13. Write a program to demonstrate large number of drawcalls overhead using openGL extension.
14. Write a program to implement openGLPSI(Pixel Shader Interlock) feature to blend decals.
15. Write a program to implement Path rendering extension to draw 2D line art.
16. Write a program to use path rendering and animate live cursive writing.
17. Write a program to implement HarfBuzz text shaping engine library for strings of unicode.
18. Write a program to implement path rendering to draw text like spokes in a wheel with 3D effect.
19. Write a program to implement conventional 3D graphic of Tiger using path rendering.
20. Write a program to implement path rendering to wrap an artwork of Tiger with multiple paths and animate.
21. Write a program to implement optimization techniques to rendering process to improve app level CPU GPU timings.
22. Write a program to simulate a cloud of particles and render its shadow on model or floor object.
23. Write a program to implement skinned meshes over bones in vertex shaders for smooth deformation.
24. Write a program to demonstrate two methods of simulating soft shadows.
25. Write a program to implement terrain engine by using hardware tessellation.
26. Write a program to implement Terrain using Texture Array for high performance.
27. Write a program to implement rendering of OIT (Order Independent Transparency) using weighted blending.

3. Computer Graphics – Vulkan Experiments

1. Write a program to render colored triangle on the screen.
2. Write a program to demonstrate the use of pipeline state objects (pso) in one single renderpass.
3. Write a program to demonstrate the use of descriptor sets for passing data to shader stages.
4. Write a program to demonstrate the use of Dynamic uniform buffers for rendering multiple objects with multiple matrices stored in a single uniform buffer object.
5. Write a program to demonstrate the use of (push constants) small shader block accessed outside of uniforms for fast updates.
6. Write a program to demonstrate the use of Shader specialization constants to create multiple pipelines with different lighting paths from a single "uber" shader.
7. Write a program to demonstrate texture loading including mip maps.
8. Write a program to demonstrate the use of cube map textures.
9. Write a program to demonstrate the use of texture arrays to display 2D textures.
10. Write a program to generate a 3D texture.
11. Write a program to load Model and texture maps.
12. Write a program to demonstrate the use of sub pass to implements a deferred rendering setup with a forward transparency pass
13. Write a program to demonstrate the use of offsceen rendering to render mirror surface from the original image.
14. Write a program to implement a simple CPU based particle system.
15. Write a program to demonstrate the use of stencil buffer and it's compare functionality for rendering a 3D model with dynamic outlines.
16. Write a program to aenders a scene made of multiple parts with different materials and textures (Scene rendering).
17. Write a program to implements multisample anti-aliasing (MSAA) using a renderpass with multisampled attachments and resolve attachments that get resolved into the visible frame buffer.
18. Write a program to implements a high dynamic range rendering pipeline using 16/32 bit floating point precision for all internal formats, textures and calculations, including a bloom pass, manual exposure and tone mapping.
19. Write a program to rendering shadows for a directional light source. (Shadow mapping)
20. Write a program to implement projective cascaded shadow mapping for directional light sources (Cascaded shadow mapping)
21. Write a program to implement omni directional shadows using a dynamic cube map.
22. Write a program to demonstrate how to generate a complete texture mip-chain using texture mapping at runtime instead of loading offline generated mip-maps from a texture file.

23. Write a program to load and render an animated skinned 3D model.
24. Write a program to capturing and saving an image after a scene has been rendered.
25. Write a program to implement multi threaded command buffer generation.
26. Write a program to implement instanced mesh rendering.
27. Write a program to demonstrate the use of indirect draw commands.
28. Write a program to demonstrate the use of occlusion query for visibility testing.
29. Write a program to demonstrate the use of query pool objects to gather statistics from different stages of the pipeline
30. Write a program to demonstrate a basic specular BRDF implementation with solid materials and fixed light sources on a grid of objects with varying material parameters.
31. Write a program to demonstrate physical based rendering with image based lighting
32. Write a program to demonstrate physical based rendering with a textured object (metal/roughness workflow) with image based lighting
33. Write a program to demonstrate deferred shading with multiple render targets
34. Write a program to demonstrate multi sampling with explicit resolve for deferred shading
35. Write a program to demonstrate deferred shading with shadows from multiple light sources using geometry shader instancing
36. Write a program to add ambient occlusion in screen space to a 3D scene.
37. Write a program to demonstrate the use of a compute shader with different convolution kernels in realtime.
38. Write a program to demonstrate attraction based compute shader particle system.
39. Write a program to demonstrate compute shader N-body simulation using two passes and shared compute shader memory.
40. Write a program to demonstrate simple GPU ray tracer with shadows and reflections using a compute shader.
41. Write a program to demonstrate compute shader cloth simulation.
42. Write a program to demonstrate compute shader culling and LOD using indirect rendering.
43. Write a program to demonstrate geometry shader (vertex normal debugging).
44. Write a program to demonstrate viewport array with single pass rendering using geometry shaders.
45. Write a program to demonstrate tessellation shader PN triangles.
46. Write a program to demonstrate the use of tessellation to renders a terrain.
47. Write a program to demonstrate minimal headless rendering
48. Write a program to demonstrate minimal headless compute shader
49. Write a program to demonstrate text overlay rendering on-top of an existing scene using a

separate render pass.

50. Write a program to demonstrate font rendering using signed distance fields.
51. Write a program to generate and renders a complex user interface with multiple windows, controls and user interaction on top of a 3D scene.
52. Write a program to demonstrate the basics of fullscreen shader effects.
53. Write a program to demonstrate bloom effect with fullscreen shader effects.
54. Write a program to implement multiple texture mapping methods to simulate depth based on texture information (Normal mapping, parallax mapping, steep parallax mapping and parallax occlusion mapping).
55. Write a program to demonstrate the use of a spherical material capture texture array defining environment lighting and reflection information to fake complex lighting.
56. Write a program to demonstrate the use of push descriptors apply the push constants concept to descriptor sets.
57. Write a program to demonstrate the use of the VK_EXT_debug_marker extension to set debug markers, regions and to name Vulkan objects for advanced debugging in graphics debuggers like RenderDoc.
58. Write a program to demonstrate the use of animated gears using multiple uniform buffers
59. Write a program to render a Vulkan demo scene with logos and mascots.

4. High Performance Image/Video Processing - VisionWorks

1. Write a program to detect feature points and track them using Lucas-Kanade method.
2. Write a program to detect lines and circles using Hough Transform.
3. Write a program for video stabilization using Lucas-Kanade method of feature tracking.
4. Write a program to detect motion using Iterative Motion Estimation Algorithm.
5. Write a program to take stereo input/video and do stereo matching to provide merged output.
6. Write a program to do alpha blending between two images (use image mask) using interoperation between VisionWorks, OpenCV and NPP.
7. Write a program to take video input from camera or video and display output using interoperation between VisionWorks and OpenGL.
8. Write a program to take input video or camera input and display as it is to test I/O facilities and camera.
9. Write a program to track any rigid object using optical flow method and object can be selected real time by dragging cursor using mouse or touch.
10. Write a program to estimate 3D feature/structure using VisionWorks SFM pipeline over 2D images/video.

5. CUDA Experiments

1. Write a program to test Asynchronous data transfer using CPU to overlap execution over GPU.
2. Write a program to measure performance using clock functions.
3. Write a program to execute CUDA function into existing CPP application.
4. Write a Program to use CPP function overloading in CUDA API using attribute check.
5. Write a program to demonstrate use of OpenMP for using Multiple GPU.
6. Write a program to demonstrate use of inline PTX assembly language.
7. Write a program to do Matrix Multiplication using CUDA API and also run performance analysis.
8. Write a program to do Matrix Multiplication using high performance CUDA Library CUBLAS.
9. Write a program to do Matrix Multiplication and demonstrate CUDA programming principles and performance analysis.
10. Write a program to demonstrate use of Assert functions in CUDA programming.
11. Write a program to demonstrate use of global memory Atomic function for arithmetic operations.
12. Write a program to create heterogeneous CPU Callbacks for GPU CUDA streams and events.
13. Write a program to take 3D input array and fetch 2D cubemap texture data for each layer and write 3D output array.
14. Write a program to fetch texture from layed 2D texture input.
15. Write a program to demonstrate MPI programming using some calculation done on multiple nodes gpu
16. Write a program to demonstrate multiple memory copy overlap between host and device.
17. Write a program to do some arithmetic operations on multiple GPU and profile it against CPU.
18. Write a program to demonstrate occupancy calculated kernel launch again manual configured.
19. Write a program to demonstrate texture bound to pitch linear memory.
20. Write a program to show printf implementation on CUDA device.
21. Write a program to demonstrate how to create static library and use for compiling CUDA application.
22. Write a program to demonstrate use of CUDA streams to overlap memcopy to host(CPU) memory to improve performance.
23. Write a program to demonstrate write to texture using a simple program of rotating a provided image.
24. Write a program to demonstrate correct use of template using dynamically allocated shared memory arrays.
25. Write a program to demonstrate how to use texture fetches in CUDA.
26. Write a program to demonstrate how to fetch texture in CUDA using kernel launch driver API.
27. Write a program to use vote intrinsic instructions in CUDA kernels.

28. Write a program to use zero memcopy using pinned system memory access.
29. Write a program to provide template for CUDA project.
30. Write a program to use a CUDA runtime template.
31. Write a program to demonstrate unified memory streams access on GPU using OpenMP.
32. Write a program to do simple vector addition.
33. Write a program to do vector addition using driver API for kernel launch.
34. Write a program to measure the memcopy bandwidth of the GPU.
35. Write a program to query properties of CUDA devices.
36. Write a program to query CUDA device properties using kernel launch driver API.
37. Write a program to get peer to peer bandwidth latency.
38. Write a program to demonstrate Bindless Surface/Texture.
39. Write a program to demonstrate Mandelbrot or Julia Fractals set interactively
40. Write a program to demonstrate Marching Cubes Algorithm to extract isosurfaces.
41. Write a program to generate Sine wave using CUDA and generate geometry using OpenGL.
42. Write a program to demonstrate use of 3D textures in CUDA using OpenGL.
43. Write a program to demonstrate volumetric filtering using 3D Texture and Surface writes.
44. Write a program to demonstrate 3D Volumetric rendering with 3D textures.
45. Write a program to demonstrate how to efficiently implement bicubic texture filtering in CUDA.
46. Write a program to uses CUDA to perform a simple bilateral filter on an image and uses OpenGL to display the results.
47. Write a program to use CUDA to perform a simple box filter on an image and uses OpenGL to display the results.
48. Write a program to do 2D convolution using FFT Transformation.
49. Write a program to apply separable convolution filter to 2D signal using gaussian kernel.
50. Write a program to do texture based 2D convolution using gaussian kernel.
51. Write a program to demonstrate Discrete Cosine Transform (DCT) for blocks of 8 by 8 pixels
52. Write a program to demonstrate 1D discrete Haar wavelet decomposition.
53. Write a program to demonstrate DirectX Texture Compressor (DXTC) using CUDA.
54. Write a program to implement 64-bin and 256-bin histogram.
55. Write a program to demonstrate variational optical flow estimation.
56. Write a program to demonstrate two image denoising techniques KNN and NLM.
57. Write a program to post process an image in OpenGL using CUDA
58. Write a program to implement gaussian blur using recursive method.
59. Write a program to demonstrate CUDA and OpenGL interop for image.
60. Write a program to demonstrate Sobel Edge detection filter.

61. Write a program to implement Stereo Disparity Computation (SAD SIMD Intrinsics).
62. Write a program to evaluate fair call price for a given set of European options under binomial model.
63. Write a program to evaluate fair call and put prices for a given set of European options by Black-Scholes formula.
64. Write a program to evaluate fair call price for a given set of European options using Monte Carlo approach.
65. Write a program to demonstrate a NiederreiterQuasirandom Sequence Generator.
66. Write a program to demonstrate Sobol Quasi-random Number Generator.
67. Write a program to simulate fluid using openGL and CUFFT library.
68. Write a program to simulate N-Body using CUDA.
69. Write a program to Simulate Ocean using CUFFT and OpenGL library.
70. Write a program to simulate large set of particles and their physical interaction over a fixed grid.
71. Write a program to implement high performance method for adding volumetric shadowing to particle systems.
72. Write a program to demonstrate the access speed difference when using aligned and misaligned data structure.
73. Write a program to implement concurrent execution of multiple kernels.
74. Write a program to compute all eigen values using bisectional algorithm.
75. Write a program to implement Fast Walsh Transform.
76. Write a program to apply time domain progression stencil on a 3D surface.
77. Write a program to demonstrate use of function pointers using Sobel Edge Detection application.
78. Write a program to demonstrate use of recursive computation over interval arithmetic operation.
79. Write a program to demonstrate line of sight algorithm.
80. Write a program to do matrix multiplication using Just in Time (JIT) compilation using PTX code.
81. Write a program to demonstrate merge sort algorithm.
82. Write a program to demonstrate global dynamic allocation using new and delete operator.
83. Write a program to simply demonstrate JIT (Just in Time) compilation using PTX code kernel.
84. Write a program to demonstrate radix sort algorithm using CUDA API with Thrust library.
85. Write a program to implement CUDA Parallel Reduction over large arrays.
86. Write a program to demonstrate scalar product of vector pair.
87. Write a program to demonstrate CUDA Parallel Prefix Sum (Scan).
88. Write a program to demonstrate CUDA segmentation tree thrust library.
89. Write a program to demonstrate CUDA Parallel Prefix Sum with Shuffle Intrinsics (SHFL_Scan).
90. Write a program to implement concurrent CUDA streams using HyperQ.

91. Write a program to demonstrate sorting networks algorithm.
92. Write a program to demonstrate reduction using thread fence intrinsic operation.
93. Write a program to demonstrate CUDA Context Management use for Multi-threading.
94. Write a program to implement Matrix Transpose using different performance algorithm.
95. Write a program to get performance by using batch of CUBLAS API calls.
96. Write a program to perform BoxFilter using NPP library functions.
97. Write a program for conjugate gradient solver using CUBLAS and CUSPARSE Library.
98. Write a program for preconditioned Conjugate Gradient using CUDA libraries.
99. Write a program to implement interoperability between FreeImage and NPP library.
100. Write a program to implement grabcut algorithm using NPP library.
101. Write a program for image histogram equilization using NPP.
102. Write a program to show image segmentation using NPP.
103. Write a program to JPEG encode/decode and resize using NPP.
104. Write a program to perform Monte Carlo estimation of PI (inline PRNG) using CURAND library.
105. Write a program to perform Monte Carlo estimation of PI (inline QRNG) using CURAND library.
106. Write a program to perform Monte Carlo estimation of PI (batch PRNG) using CURAND library.
107. Write a program to perform Monte Carlo estimation of PI (batch QRNG) using CURAND library.
108. Write a program to perform Monte Carlo simulation for single asian option using CURAND library.
109. Write a program to perform Mersenne Twister GP11213 random number generator using CURAND.
110. Write a program to demonstrate use of random number by generating Random Fog.
111. Write a program to show use of CUBLAS library using CPU GPU test.
112. Write a program to demonstrate 1D-Convolution using CUFFT library.
113. Write a program to solve 2D-Poisson equation using CUFFT library.

Contact us

Registered Office

SIGMA TRAINERS AND KITS
E-113, Jai Ambe Nagar,
Near Udgam School,
Drive-in Road,
Thaltej,
AHMEDABAD-380054. INDIA.

Factory

SIGMA TRAINERS AND KITS
B-6, Hindola Complex,
Below Nishan Medical Store,
Lad Society Road,
Near Vastrapur Lake,
AHMEDABAD-380015. INDIA.

Contact Person

Prof. D R Luhar – Director

Mobile : 9824001168

Whatsapp : 9824001168

Phones:

Office : +91-79-26852427

Factory : +91-79-26767512

+91-79-26767648

+91-79-26767649

E-Mails :

sales@sigmatrainers.com

drluhar@gmail.com