Application Note

Sitara™ AM62Ax Benchmarks

Qutaiba Saleh, Pekka Varis and Jonathan Bishop

ABSTRACT

This application note presents a series of benchmarks measuring performance of various components of AM62Ax family of devices. Some of the standard benchmarks are included in the Linux SDK while the others can be downloaded from their respective hosting websites. Instructions on how to execute the tests and analysis of the results are also included.

Table of Contents

List of Figures

List of Tables

Trademarks

1 Introduction

AM62Ax contains up to four Arm®-Cortex®-A53 cores with 64-bit architecture, a Cortex-R5F MCU core, a Cortex-R5F Device Management core, a single core C7x256v DSP with MMA, and various other accelerators such as Video and Vision processing, display interface with extensive peripheral and networking options. It supports LPDDR4 32-bit width with speed of 3200 MT/s. Figure 1-1 is a functional block diagram for AM62Ax. For details, see *AM62Ax Sitara Processors Data Sheet*.

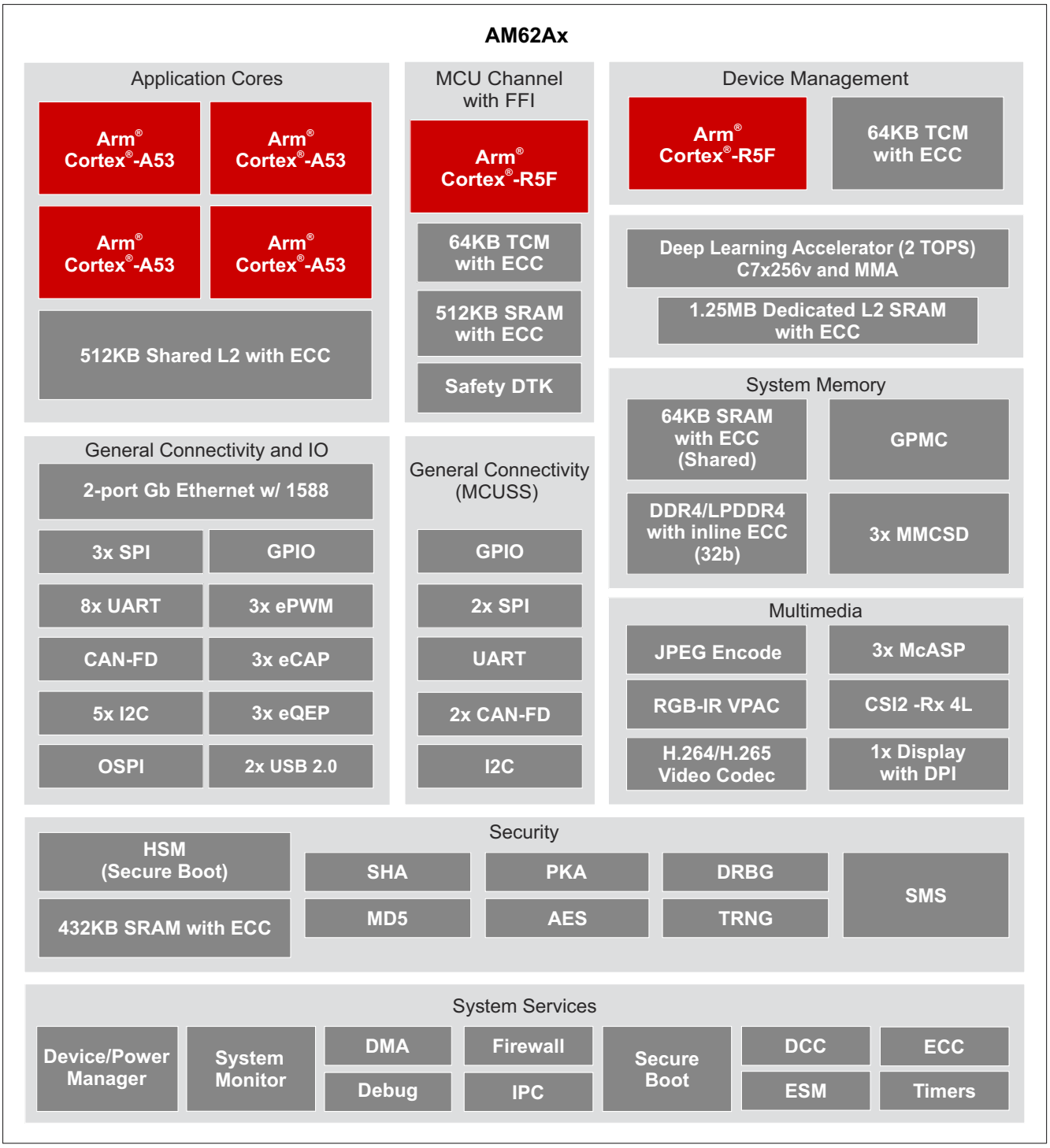

Figure 1-1. AM62Ax Functional Block Diagram

This document presents a number of industry standard and application specific benchmarks measured on the AM62Ax processor. The tests focus on the performance of the Arm-Cortex-A53 cors and the LPDDR4 memory with some application specific benchmarks for the Arm-Cortex-R5F MCU, C7x DSP core, and other memory components. The key parameters of the evaluation board are 1.25 GHz and 1.4 GHz clock speed for the Arm-Cortex-A53 cores, 1.0 GHz clock speed for the C7x DSP, and a 32-bit wide LPDDR4 at a speed of 3200MT/s. Most of the standard benchmarks are already included in the SDK and can be directly executed while the other benchmarks can be downloaded from their respective official host websites. All of the benchmarks are implemented using Linux SDK 8.06.

1.1 Change Cortex-A53 Clock Frequency

Maximum frequency of A53 cores on AM62Ax varies based on speed grade of the device and the VDD_CORE voltage applied. AM62A SK-LP EVM Rev E2 supports 0.75 V VDD_CORE mode operation which supports operation of maximum core frequency of 1.25 GHz. To obtain peak performance of the device, the setup was modified to run the A53 cores on AM62Ax at 1.4 GHz.

2 Processor Core Benchmarks

This section contains benchmarks contained within an Arm Cortex processor core. Synthetic benchmarks included are for example Dhrystone.

2.1 Dhrystone

Dhrystone benchmark focuses on the processor core performance. It runs from warm L1 caches in all modern processors. It scales linearly with clock speed. Even though the benchmark was introduced in 1984 by Reinhold P. Weicker, Dhrystone still gets used in embedded processing. The industry has adopted the VAX 11/780 as the reference 1 MIPS machine. The VAX 11/780 achieves 1757 Dhrystones per second. The score calculated by normalizing the time it takes the benchmark loop to run by the reference 1 MIPS machine score of 1757. It is common to further normalize to DMIPS/MHz/core as the score scales linearly with clock speed. For standard Arm cores, the DMIPS/MHz is identical to the same compiler and flags. Dhrystone is a single core benchmark, a simple sum of multiple cores running the benchmark in parallel is sometimes used.

The Dhrystone (Version 2.1, C Language) benchmark is included in the SDK. It can be performed by simply running the command *dhrystone*. Due to its short execution time, it is suggested to run the test for high number of iterations in order to measure accurate results. More than 100 million iterations are used in the tests implemented for Arm-Cortex-A53. The code block below shows a short version of the terminal printout for Dhrystone benchmark execution.

```
root@am62axx-evm:~# dhrystone

Dhrystone Benchmark, Version 2.1 (Language: C)

Program compiled without 'register' attribute

Please give the number of runs through the benchmark: 100000000

Execution starts, 100000000 runs through Dhrystone
Execution ends

Final values of the variables used in the benchmark:
.
.
.
Microseconds for one run through Dhrystone:    0.1
Dhrystones per Second:                      7142857.0
```

Table 2-1 shows the results for this benchmark with the compiler and operating system details. The aggregate scores for AM62Ax with four A53 cores running at 1.25 GHz and 1.4 GHz are 14,880 DMIPS and 16,261 DMIPS, respectively.

Table 2-1. Dhrystone Benchmarks

	Arm-Cortex-A53(1.25 GHz)	Arm-Cortex-A53 (1.4 GHz)
Dhrystones/s	6,535,947	7,142,857
Normalized Dhrystones (divide by 1757 reference for 1MIPS)	3,720	4,065
DMIPS/MHz each core	~3	~3
Compiler and flags	GCC 9.2 -march=ARMv8 -O3	
Operating System	Linux 5.10 (2021 LTS)	

3 Compute and Memory System Benchmarks

This section contains benchmarks involving the Arm-Cortex processor core and the memory system of the System-on-Chip (SoC). Synthetic benchmarks included are, for example, LMBench and CoreMark-Pro. Math function benchmarks include Fast Fourier Transforms (FFT).

3.1 Memory Bandwidth and Latency

A subset of LMBench and STREAM are benchmarks to measure achieved memory bandwidth and latency from software.

3.1.1 LMBench

LMBench is a suite of micro benchmarks for processor cores and operating system primitives. The memory bandwidth and latency related tests are most relevant for modern embedded processors. The results vary a little (< 10%) run to run.

LMBench benchmark *bw_mem* measures achieved memory copy performance. With parameter *cp* it does an array copy and *bcopy* parameter uses the runtime glibc version of *memcpy()* standard function. The glibc uses a highly optimized implementation that utilizes, for example, SIMD resulting in higher performance. The size parameter equal to or smaller than the cache size at a given level measures the achievable memory bandwidth from software doing a typical for loop or *memcpy()* type operation. Typical use is for external memory bandwidth calculation. The bandwidth is calculated as byte read and written counts as 1, which is roughly half of STREAM copy result. The benchmark further allows creating parallel threads with -P parameter. To get the maximum multi-core memory bandwidth, create the same amount of threads as there are cores available for the operating system, which is 4 for AM62x Linux (-P 4). To show full performance characterization of the AM62Ax, the LMBench tests are implemented on full factorial combinations of number of cores and clock frequency. The code block below shows terminal printout of executing the LMBench commands.

```
root@am62axx-evm:~# bw_mem 8M bcopy
8.00 2125.96
root@am62axx-evm:~# bw_mem -P 2 8M bcopy
8.00 3408.74
root@am62axx-evm:~# bw_mem -P 4 8M bcopy
8.00 3884.24

root@am62axx-evm:~# bw_mem 8M cp
8.00 1108.49
root@am62axx-evm:~# bw_mem -P 2 8M cp
8.00 1671.98
root@am62axx-evm:~# bw_mem -P 4 8M cp
8.00 1976.17
```

Table 3-1 shows the measured bandwidth and the efficiency compared to theoretical wire rate. The wire rate used is the LPDDR4 MT/s rate times the width divided by two (read and write making up a copy both consume the bus).

$$Efficiency = \frac{Measured\ Speed}{\frac{LPDDR4\ MT/s \times width}{2}} = \frac{Measured\ Speed}{\frac{3200 \times 4\ B}{2}} = \frac{Measured\ Speed}{6400} \quad (1)$$

Table 3-1. LMBench Results

Command	Description	Arm-Cortex-A53 at 1.25 GHz, LPDDR4-3200MT/s-32 Bit	LPDDR4 Efficiency	Arm-Cortex-A53 at 1.4 GHz, LPDDR4-3200MT/s-32 Bit	LPDDR4 Efficiency
Bw_mem 8M bcopy	*Single core, glibc memcpy*	2,058 MB/s	32%	2,125 MB/s	33%
bw_mem -P 2 8M bcopy	*Dual core, glibc memcpy*	3,300 MB/s	52%	3,408 MB/s	53%
bw_mem -P 4 8M bcopy	*Quad core, glibc memcpy*	3,816 MB/s	60%	3,884 MB/s	61%
Bw_mem 8M cp	*Single core, inline copy loop*	1,076 MB/s	17%	1,108 MB/s	17%
bw_mem -P 2 8M cp	*Dual core, inline copy loop*	1,659 MB/s	26%	1,671 MB/s	26%
bw_mem -P 4 8M cp	*Single core, inline copy loop*	1,952 MB/s	31%	1,976 MB/s	31%

LMBench benchmark *lat_mem_rd* is used to measure the observed memory access latency for external memory (LPDDR4 on AM62Ax) and cache hits. The two arguments are the size of the transaction (64 in the code block below) and the stride of the read (512). These two values are selected to measure the latency to caches and external memory, not the processor data prefetchers or other speculative execution. For access patterns, the prefetching will work, but this benchmark is most useful to measure the case when it does not.

The code block below shoes the terminal printout of executing *lat_mem_rd* command. The left column is the size of the data access pattern in megabytes, right column is the round trip read latency in nanoseconds. This command is executed for Arm-Cortex-A53 clock frequency of 1.25 GHz and 1.4 GHz.

```
root@am62axx-evm:~# lat_mem_rd 64 512
"stride=512
0.00049 2.146
0.00098 2.146
0.00195 2.146
0.00293 2.146
0.00391 2.146
0.00586 2.146
0.00781 2.146
0.01172 2.146
0.01562 2.146
0.02344 2.146
0.03125 2.202
0.04688 6.725
0.06250 7.711
0.09375 8.725
0.12500 9.229
0.18750 9.748
0.25000 10.009
0.37500 22.217
0.50000 23.840
0.75000 88.270
1.00000 116.937
1.50000 133.405
2.00000 135.724
3.00000 137.268
4.00000 137.974
6.00000 138.512
8.00000 138.841
12.00000 139.033
16.00000 139.102
24.00000 139.097
32.00000 139.165
48.00000 139.163
64.00000 139.261
```

Figure 3-1 shows connected scatter plots of memory latency results for both 1.25 GHz and 1.4 GHz. Based on memory block size (x-axis), the plot can be divided into three regions. The first region is when the accessed memory block is smaller than L1 cache. It is safe to assume that the data is completely inside the L1 and such the latency in this region is a close estimation of L1 cache latency. The second region is when the accessed

memory block is bigger than L1 but smaller than L2 cache. The latency in this region is a mix of L1, L2, and LPDDR4 latency. The latency at the middle of that region can be assumed to be a close representation of L2 latency. The third region is when the access memory block is bigger than L2 cache. The last reading in this region reflects the LPDDR4 latency.

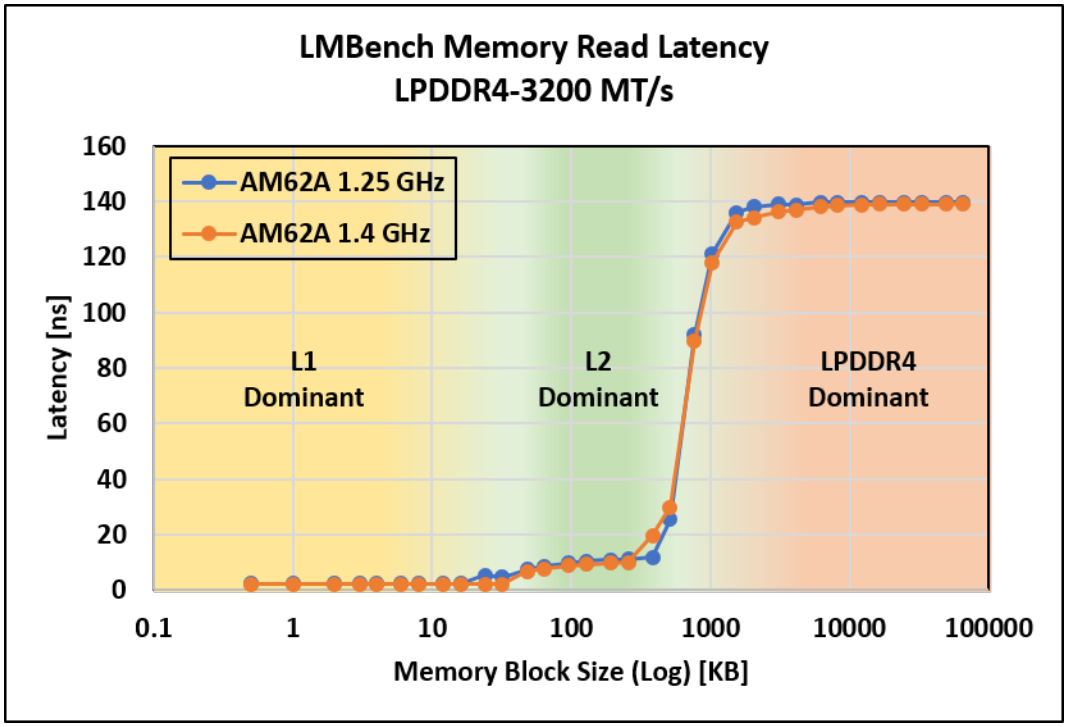

Figure 3-1. Memory Read Latency

Table 3-2 shows a summary for Arm-Cortex-A53 read latency.

Table 3-2. Memory Read Latency Results

Memory	Arm-Cortex-A53 at 1.25 GHz	Arm-Cortex-A53 at 1.4 GHz
L1 cache	2.4 ns	2.1 ns
L2 cache	10.3 ns	9.2 ns
LPDDR4-3200 MT/s	140.2 ns	139.2 ns

3.1.2 STREAM

STREAM is a microbenchmark for measuring data memory system performance without any data reuse. It is designed to miss on caches and exercise the data prefetcher and speculative accesses. It uses double precision floating point (64 bit), but in most modern processors the memory access is the bottleneck. The four individual scores are copy, scale as in multiply by constant, add two numbers, and triad for multiply accumulate.

- Copy: Measures memory transfer rate without arithmetic operation, a[i] = b[i]
- Scale: Includes a simple arithmetic operation, a[i] = k × b[i]
- Add: Includes three memory access in addition to arithmetic operation, a[i] = b[i] + c[i]
- Triad: Combines scale and add in one operation, a[i] = b[i] + k × c[i]

For bandwidth, a byte read counts as one and a byte written counts as one resulting in a score that is double the bandwidth LMBench. Table 3-3 shows the measured bandwidth and the efficiency compared to theoretical wire rate. The wire rate used is the LPDDR4 MT/s rate times the width. To get overall maximum achieved throughput the command used is *stream -M 16M -P 4-N 10*, which means four parallel threads and 10 iterations. The Arm-Cortex-A53 clock frequency is setup to 1.4 GHz in this test.

Table 3-3. Stream Benchmarks

	LPDDR4-3200MT/s-32-Bit Bandwidth	LP DDR4-3200MT/s-32-Bit Efficiency
copy	7,780 MB/s	61%
scale	7,815 MB/s	61%
add	6,868 MB/s	54%
triad	6,871 MB/s	54%

3.1.3 Critical Memory Access Latency

This section provides read memory access latency measurements from processors in AM62Ax to various memory destinations in the system. The measurements where made on the AM62Ax platform using bare-metal silicon verification tests that are not currently included in the SDK. The tests executed on A53, C7x and R5F processor out of the LPDDR4. Each test includes a loop of 8192 iterations to read a total of 32 KiB of data. The number of cycles for each test were counted and divided by the respective processor clock frequency to obtain latency time. Table 3-4 shows the average latency results.

Table 3-4. Critical Memory Access Latency of A53, C7x, R5F MCU, and R5F WKUP

Memory	Arm-Cortex-A53 (Avg ns)	C7x DSP (Avg ns)	Arm-Cortex-R5F MCU (Avg ns)	Arm-Cortex-R5F WKUP (Avg ns)
LPDDR4	137	154	202	172
OCSRAM MAIN	59	57	122	77
OCSRAM MCU	120	118	58	85
OCSRAM WKUP	210	189	203	156
C7X SRAM - Local Path	NA	20	NA	NA
C7X SRAM - External Path	80	NA	151	103
R5F MCU TCM - Local Path	NA	NA	1	NA
R5F MCU TCM - External Path	143	144	NA	120
R5F WKUP TCM - Local Path	NA	NA	NA	1
R5F WKUP TCM - External Path	112	108	120	NA

Tests were done at 0.75V VDD_CORE settings (A53 : 1.25 GHz, C7x DSP: 1.0 GHz and R5: 800 MHz) and LPDDR4 @3200MT/s.

3.2 CoreMark-Pro

CoreMark®-Pro tests the entire processor, adding comprehensive support for multi-core technology, a combination of integer and floating-point workloads, and data sets for utilizing larger memory subsystems. The components of CoreMark-Pro utilizes all levels of cache with an up to 3MB data memory footprint. Many, but not all of the tests, are also using P threads to allow utilization of multiple cores. The score scales with the number of cores but is always less than linear (dual core score is less than 2x single core).

CoreMark-Pro should not be confused with the smaller CoreMark which, like Dhrystone, is a microbenchmark contained in L1 caches of a modern processor.

CoreMark-Pro is not included in the SDK. It can be downloaded from the official host website. In this tests, the code is directly cloned and built in the AM62Ax EVM. All official CoreMark-Pro rules have been satisfied such as making sure that the execution time of each workload is at least 1000 times the minimum timer resolution. Table 3-5 shows the CoreMark-Pro results for single, dual, and quad A53 cores at both 1.25 GHz and 1.4 GHz.

Table 3-5. CoreMark®-Pro Results

	Arm-Cortex-A53 At 1.25 GHz (iter/s)	Parallel Scaling	Arm-Cortex-A53 At 1.4 GHz (iter/s)	Parallel Scaling
Single Core	837	1	965	1
Dual Core	1,548	1.81	1,723	1.8
Quad Core	2,465	2.9	2,694	2.82

3.3 Fast Fourier Transform

Fast Fourier Transform (FFT) is on of the most common signal processing algorithms. This section shows FFT performance results for both Arm-Cortex-A53 and C7x processors.Table 3-6 shows a 1024-point single precision floating point complex FFT execution time on Arm-Cortex-A53 and C7x DSP. The benchmark on Arm-Cortex-A53 uses the implementation from Ne10 library, which leverages the Advanced SIMD or NEON acceleration of Cortex-A53. This library is not included in the SDK but it can be downloaded from the official Ne10 repository. On the C7x, the FFT library available within the SDK was employed to benchmark the performance. The code executed on bare-metal with warm L2 cache.

Table 3-6. NE10 CFFT Benchmark

	Arm-Cortex-A53 at 1.25 GHz (single thread / core)	Arm-Cortex-A53 at 1.4 GHz (single thread / core)	C7x at 1.0 GHz
1024-point Complex FFT Execution Time	21.7 µs	19.4 µs	1.8 µs

3.4 Cryptographic Benchmarks

The AM62Ax Processor SDK Linux includes an openssl cryptographic library which provides optimized implementations of cryptographic functions. It is employed by some applications such as HTTPS, ssh, and netconf implementations. For the highest performance, the higher-level interface provided by the EVP library should be used. Table 3-7 shows a set of selected benchmarks of software observed performance run on AM62Ax. Command run was *openssl speed -elapsed -evp <cryptographic mode> -multi 4*. This is utilizing all four A53 cores using four threads. In these tests, the Arm-Cortex-A53 is clocked at 1.4 GHz. The output of the openssl command is in KB/s. To meet desired industry standard, the results reported in the Table 3-7 are transformed to Mb/s.

Table 3-7. Symmetric Cryptography and Secure Hash in Mbit/s

	Frame Size (bytes)					
	16	**64**	**256**	**1024**	**8192**	**16384**
aes-128-gcm	2,348	6,796	12,960	16,89	18,512	18,556
aes-256-gcm	2,227	6,242	11,372	14,538	15,703	15,634
aes-128-ctr	246	546	2,059	6,902	21,045	24,888
sha256	16	64	257	988	5,820	8,705
sha512	16	63	22	756	1,943	2,175
chacha20-poly1305	1,366	2,933	5,79	6,665	7,018	7,020

Further benchmarks for public key cryptography are shown in Table 3-8. Tests can be run with command *openssl speed -elapsed -multi 4 <algorithm>* .

Table 3-8. Public Key Cryptography Benchmarks

RSA	size	512	1024	2048	3072	4096
	sign/second	15,589	3,357	515	169	75
	verify/second	192,415	66,789	19,230	8,898	5,097
ECDSA	curve	nistp224	nistp256	nistp521	nistk233	nistb233
	sign/second	1,056	4,457	213	801	782
	verify/second	1,701	7,786	302	430	420

4 Application Benchmarks

This section contains an application level benchmark for machine learning inference.

4.1 Machine Learning Inference

Comprehensive Machine Learning performance numbers on 2TOPS Deep Learning Accelerator (C7x 256v with MMA) on AM62Ax will be available on EdgeAI cloud : Edge AI (ti.com).

TensorFlow Lite is used to test the performance of the Arm-Cortex-A53 processors in deep learning inference at the edge. As examples below are two runs of TensorFlow Lite models for image classification (224x224 pixels 3 bytes for colors) based on imagenet database and 1000 object classes. A quantized Mobilenetv1 and floating point Mobilenetv2 were chosen as common benchmarks that can be used to interpolate the performance of an inference application. These models are not available in the SDK. The TensorFlow Lite classifier and models (1.15-r5.0) were downloaded from the official host website at tensorflow.org . The example image of Rear Admiral Grace Hopper is installed in the file system (available at). The example label_image program will crop and resize the bmp image to the 224 x 224 pixels before calling the TensorFlow Lite. The below code block shows the terminal printout of Mobilenetv1 (mobilenet_v1_1.0_224_quant.tflite) and Mobilenetv2 (mobilenet_v2_1.0_224.tflite) models inference execution of the same image resolutions (224x224x3).

```
root@am62axx-evm:/usr/share/tensorflow-lite/examples# ./label_image -i grace_hopper.bmp -l
labels.txt -m mobilenet_v1_1.0_224_quant.tflite
Loaded model mobilenet_v1_1.0_224_quant.tflite
resolved reporter
invoked
average time: 56.945 ms
0.780392: 653 military uniform
0.105882: 907 Windsor tie
0.0156863: 458 bow tie
0.0117647: 466 bulletproof vest
0.00784314: 835 suit

root@am62axx-evm:/usr/share/tensorflow-lite/examples# ./label_image -i grace_hopper.bmp -l
labels.txt -m mobilenet_v2_1.0_224.tflite
Loaded model mobilenet_v2_1.0_224.tflite
resolved reporter
invoked
average time: 178.05 ms
0.911345: 653 military uniform
0.014466: 835 suit
0.0062473: 440 bearskin
0.00296661: 907 Windsor tie
0.00269019: 753 racket
```

For full performance evaluation of the Arm-Cortex-A53 processors, the benchmarks were executed at both 1.25 GHz and 1.4 GHz. Table 4-1 shows the average time for all tests executions.

Table 4-1. Average Time for All Tests Executions

Model	Arm-Cortex-A53 At 1.25 GHz	Arm-Cortex-A53 At 1.4 GHz
Mobilenetv1 *mobilenet_v1_1.0_224_quant.tflite*	63.35 ms	56.94 ms
Mobilenetv2 *mobilenet_v2_1.0_224.tflite*	192.10 ms	178.05 ms

5 References

- CoreMark-Pro
- STREAM McCalpin, John D. "STREAM: Sustainable Memory Bandwidth in High Performance Computers", a continually updated technical report (1991-2007), available at: http://www.cs.virginia.edu/stream/
- Ne10 math library
- hosted models at tensorflow.org
- OpenSSL
- Texas Instruments: *AM62Ax Sitara Processors Data Sheet*

IMPORTANT NOTICE AND DISCLAIMER

TI PROVIDES TECHNICAL AND RELIABILITY DATA (INCLUDING DATA SHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS AND IMPLIED, INCLUDING WITHOUT LIMITATION ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for skilled developers designing with TI products. You are solely responsible for (1) selecting the appropriate TI products for your application, (2) designing, validating and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, regulatory or other requirements.

These resources are subject to change without notice. TI grants you permission to use these resources only for development of an application that uses the TI products described in the resource. Other reproduction and display of these resources is prohibited. No license is granted to any other TI intellectual property right or to any third party intellectual property right. TI disclaims responsibility for, and you will fully indemnify TI and its representatives against, any claims, damages, costs, losses, and liabilities arising out of your use of these resources.

TI’s products are provided subject to TI’s Terms of Sale or other applicable terms available either on ti.com or provided in conjunction with such TI products. TI’s provision of these resources does not expand or otherwise alter TI’s applicable warranties or warranty disclaimers for TI products.

TI objects to and rejects any additional or different terms you may have proposed.

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265