Application Note

Sitara™ AM62Lx 基准测试

Divyansh Mittal, Andrew Shutzberg, Pekka Varis, Jonathan Bishop, and Qutaiba Saleh

摘要

本应用手册介绍了一系列基准，用于测量 AM62Lx 系列器件的各种元件的性能。一些标准基准包含在 Linux SDK 中，而其他基准可以从各自的托管网站下载。此外，本文还包含有关如何执行部分测试和分析测试结果的说明。

内容

商标

1 引言

AM62Lx 是一款片上系统 (SoC)，包含两个具有 64 位架构的 Arm®-Cortex®-A53 内核。AM62Lx 具有多种嵌入式功能，例如多媒体 DSI/DPI 支持、集成片上 ADC、高级低功耗管理模式以及用于 IP 保护的广泛安全选项。该器件包含多个支持系统级连接的外设，例如 USB、MMC/SD、OSPI、CAN-FD 和 ADC。图 1-1 是 AM62Lx 的功能方框图。有关详细信息，请参阅 AM62Lx Sitara 处理器数据表。

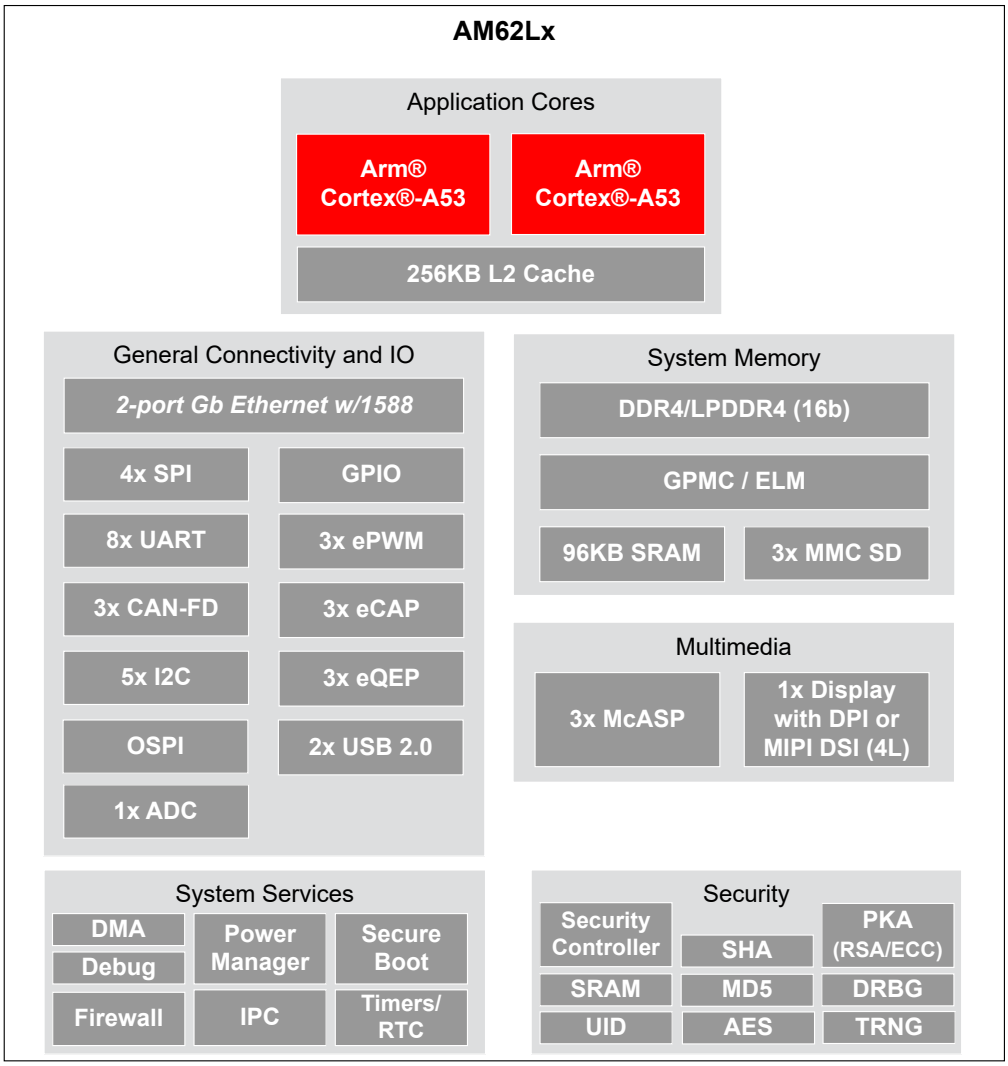

图 1-1. 功能方框图

本文档介绍了在 AM62Lx 处理器上测量的一系列业界通用的应用特定基准，这些测试侧重 Arm®-Cortex®-A53 内核和 LPDDR4 存储器的性能，以及一些特定应用的基准测试。评估板的关键参数是 Arm®-Cortex®-A53 内核的 1.25Ghz 时钟速度，LPDDR4 16 位宽和 1600MT/s 速度。大多数标准基准已经包含在 SDK 中，可以直接执行，而其他基准可以从各自的官方托管网站下载。

2 处理器内核和计算基准测试

本节介绍 Arm Cortex 处理器内核基准测试，包括合成基准测试，例如 Dhrystone。

2.1 Dhrystone

Dhrystone 基准测试侧重于处理器内核性能。该基准测试在所有现代处理器中均采用预加载 L1 高速缓存运行。该基准测试随时钟速度的增加而线性增加。虽然该基准测试于 1984 年由 Reinhold P. Weicker 引入，但 Dhrystone 至今仍用于嵌入式处理。业界已采用 VAX 11/780 作为参考 1 MIPS 机器。VAX 11/780 每秒可达到 1757 Dhrystones。计算分数时，通过参考 1MIPS 机器的分数 (1757)，对基准测试循环运行的时间进行归一化。由于分数随时钟速度的增加而线性增加，常见问题被进一步归一化为 DMIPS/MHz/内核。对于标准 Arm 内核，在相同的编译器和标志中，DMIPS/MHz 将是相同的。Dhrystone 是一个单核基准测试，有时会使用多个简单内核并行运行此基准测试。

Dhrystone (版本 2.1，C 语言) 基准测试包含在 SDK 中，只需运行命令 *dhrystone* 即可执行。由于执行时间短，TI 建议运行大量迭代测试以测量准确的结果。在为 Arm-Cortex-A53 实施的测试中使用了 10 亿次迭代。下面的代码块展示了用于 Dhrystone 基准执行的终端打印输出的简短版本。

```
root@am62lxx-evm:~# dhrystone

Dhrystone Benchmark, Version 2.1 (Language: C)

Program compiled without 'register' attribute

Please give the number of runs through the benchmark: 1000000000

Execution starts, 1000000000 runs through Dhrystone
Execution ends

Final values of the variables used in the benchmark:
.
.
.
Microseconds for one run through Dhrystone:    0.2
Dhrystones per Second:                      6410256.5
```

表 2-1 展示了此基准测试的结果，其中包含编译器和操作系统详细信息。具有两个 A53 内核的 AM62Lx 在 1.25GHz 下运行的汇总分数为 7,296 DMIPS。

表 2-1. Dhrystone 基准测试

	Arm-Cortex-A53 (1.25GHz)
Dhrystones/s	6,410,256
归一化 dhrystones (除以参考 1MIPS 机器的 1757)	3648
每个核心的 DMIPS/MHz	≈3
编译器和标志	GCC 13.3.0 -march=ARMv8 -O3
操作系统	Linux 6.12.0

2.2 Whetstone

Whetstone 是主要测量浮点运算性能的基准测试。

```
root@am62lxx-evm:~# runWhetstone


Whetstone running ...

Execution time approx 10 seconds

Loops: 100000, Iterations: 1, Duration: 2 sec.
C Converted Double Precision Whetstones: 5000.0 MIPS
```

表 2-2 展示了该基准测试的结果。

表 2-2. Whetstone 基准测试

	Arm-Cortex-A53 (1.25GHz)
Whetstone (MIPS)	5,000

2.3 Linpack

Linpack 测量峰值双精度（64 位）浮点性能，以解决密集线性系统问题。

```
root@am62lxx-evm:~# runLinpack


Linpack running ...

Execution time approx 10 seconds
Unrolled Single Precision Linpack

Unrolled Single Precision Linpack

     norm. resid      resid           machep         x[0]-1        x[n-1]-1
       2.0        4.69621336e-05  1.19209290e-07 -1.31130219e-05 -1.30534172e-05
    times are reported for matrices of order   100
      dgefa      dgesl      total       kflops     unit      ratio
 times for array with leading dimension of  201
       0.00       0.00       0.00     506018       0.00       0.02
       0.00       0.00       0.00     507890       0.00       0.02
       0.00       0.00       0.00     511293       0.00       0.02
       0.00       0.00       0.00     511331       0.00       0.02
 times for array with leading dimension of 200
       0.00       0.00       0.00     559630       0.00       0.02
       0.00       0.00       0.00     563766       0.00       0.02
       0.00       0.00       0.00     563304       0.00       0.02
       0.00       0.00       0.00     564925       0.00       0.02
Unrolled Single  Precision 511331 Kflops ; 10 Reps
```

表 2-3 展示了该基准测试的结果。

表 2-3. Linpack 基准测试

	Arm-Cortex-A53 (1.25GHz)
Linpack (Kflops)	511331

2.4 NBench

NBench 表示本地基准测试，是用于测量常用操作（如排序和分析算法）的宏基准测试。有关 NBench 的更多信息，请参阅 NBench Wiki 和 NBench Articles。

```
root@am62lxx-evm:~# cd /usr/bin
root@am62lxx-evm:/usr/bin# nbench

BYTEmark* Native Mode Benchmark ver. 2 (10/95)
Index-split by Andrew D. Balsa (11/97)
Linux/Unix* port by Uwe F. Mayer (12/96,11/97)

TEST                : Iterations/sec.  : Old Index   : New Index
                    :                  : Pentium 90* : AMD K6/233*
--------------------:------------------:-------------:------------
NUMERIC SORT        :          560.92  :      14.39  :       4.72
STRING SORT         :          146.35  :      65.39  :      10.12
BITFIELD            :      1.7834e+08  :      30.59  :       6.39
FP EMULATION        :          192.44  :      92.34  :      21.31
FOURIER             :           20381  :      23.18  :      13.02
ASSIGNMENT          :          12.971  :      49.36  :      12.80
IDEA                :          3075.1  :      47.03  :      13.96
HUFFMAN             :          1057.3  :      29.32  :       9.36
NEURAL NET          :           7.741  :      12.44  :       5.23
LU DECOMPOSITION    :          472.71  :      24.49  :      17.68
==========================ORIGINAL BYTEMARK RESULTS==========================
INTEGER INDEX       : 40.569
FLOATING-POINT INDEX: 19.182
Baseline (MSDOS*)   : Pentium* 90, 256 KB L2-cache, Watcom* compiler 10.0
============================LINUX DATA BELOW================================
CPU                 : Dual
L2 Cache            :
OS                  : Linux 6.12.0-ti-g1a4dee21b1eb-dirty
C compiler          : aarch64-oe-linux-gcc
libc                : static
MEMORY INDEX        : 9.390
INTEGER INDEX       : 10.711
FLOATING-POINT INDEX: 10.639
Baseline (LINUX)    : AMD K6/233*, 512 KB L2-cache, gcc 2.7.2.3, libc-5.4.38
* Trademarks are property of the respective holder.
```

表 2-4 展示了该基准测试的结果。

表 2-4. NBench 基准测试

基准测试	Arm-Cortex-A53 (1.25GHz)
任务（迭代次数）	12.97
傅里叶变换（迭代次数）	20381
浮点模拟（迭代次数）	192.44
霍夫曼编码（迭代次数）	1057.3
IDEA 加密算法（迭代次数）	3075.1
LU 分解（迭代次数）	472.71
神经网络（迭代次数）	7.74
数字排序（迭代次数）	560.92
字符串排序（迭代次数）	146.35

2.5 CoreMark®-Pro

CoreMark®-Pro 测试了整个处理器，增加了对多核技术，整数和浮点工作负载以及用于利用更大存储子系统的数据集的全面支持。CoreMark-Pro 的组件利用各种级别的高速缓存，数据存储器容量高达 3MB。许多但并非所有测试会使用 P 线程，以便允许执行多个内核。分数随内核数量的增加而增加，但总是低于线性增加（双核分数小于单核分数的 2 倍）。

不得将 CoreMark-Pro 与更小巧的 CoreMark 混淆，后者和 Dhrystone 一样，都是包含在现代处理器 L1 高速缓存中的微基准。

CoreMark-Pro 不包含在 SDK 中，可以从 CoreMark-Pro 下载。在此测试中，直接克隆代码并将其内置在 AM62Lx EVM 中。下面的步骤用于直接在目标上克隆、构建和运行 CoreMark-Pro：

1. 克隆存储库。

```
root@am62lxx-evm:~# git clone https://github.com/eembc/coremark-pro.git
```

2. 构建 CoreMark-Pro。

```
root@am62lxx-evm:~# cd coremark-pro/
root@am62lxx-evm:~/coremark-pro# make TARGET=linux64 build-all
```

3. 运行 CoreMark-Pro：使用 *“certify-all”* 运行 CoreMark-Pro 的所有 9 个基准测试并使用 *“XCMD”* 设置内核数量。

```
root@am62lxx-evm:~/coremark-pro# make TARGET=linux64 certify-all XCMD='-c2'
```

基准测试输出：

```
root@am62lxx-evm:~/coremark-pro# make TARGET=linux64 certify-all XCMD='-c2'
.
.
WORKLOAD RESULTS TABLE

                                              MultiCore SingleCore
Workload Name                                  (iter/s)   (iter/s)    Scaling
---------------------------------------------- ---------- ---------- ----------
cjpeg-rose7-preset                                71.43      37.04       1.93
core                                               0.52       0.27       1.93
linear_alg-mid-100x100-sp                         24.58      12.92       1.90
loops-all-mid-10k-sp                               0.72       0.43       1.67
nnet_test                                          1.96       1.01       1.94
parser-125k                                        6.85       7.09       0.97
radix2-big-64k                                    32.47      22.28       1.46
sha-test                                         138.89      73.53       1.89
zip-test                                          33.90      20.00       1.69

MARK RESULTS TABLE

Mark Name                                     MultiCore SingleCore    Scaling
---------------------------------------------- ---------- ---------- ----------
CoreMark-PRO                                    1189.42     710.32       1.67
```

所有正式的 CoreMark-Pro 规则都已得到满足，例如确保每个工作负载的执行时间至少是最小计时器分辨率的 1000 倍。表 2-5 展示了单核和双核 A53 在 1.25GHz 下的 CoreMark-Pro 结果。

表 2-5. CoreMark-Pro 结果

	Arm-Cortex-A53 (1.25GHz) [iter/s]	并行缩放
单核	710	1
双核	1,189	1.67

2.6 快速傅里叶变换

快速傅里叶变换 (FFT) 是常见的信号处理算法之一。表 2-6 展示了 Arm-Cortex-A53 上的 1024 点单精度浮点复数 FFT 执行时间。Arm-Cortex-A53 基准测试使用了 Ne10 库，该库利用了 Cortex-A53 的高级 SIMD 或 NEON 加速。此库不包含在 SDK 中，但可以从官方 Ne10 代码库下载。

表 2-6. NE10 CFFT 基准测试

	Arm-Cortex-A53 (1.25GHz) (单线程/内核)
1024 点复数 FFT 执行时间	37µs

2.7 加密基准测试

AM62Lx Processor SDK Linux 包括一个 openssl 加密库，可提供加密运算的优化实现。某些应用 (例如 HTTPS、ssh 和 netconf 实现) 采用了 AM62Lx Processor SDK Linux。为了获得优异的性能，必须使用 EVP 库提供的较高级别的接口。表 2-7 展示了在 AM62Lx 上运行的一组选定的软件观察到的性能的部分基准测试。运行的命令是 *openssl speed -elapsed -evp <cryptographic mode> -multi 2*。这利用了两个 A53 内核，每个内核使用两个线程。在这些测试中，Arm-Cortex-A53 的时钟频率为 1.25GHz。openssl 命令的输出以 KB/s 为单位。为了满足所需的行业标准，表 2-7 中所报告的结果被转换为 Mb/s。

表 2-7. 对称加密和安全哈希 (单位为 Mbit/s)

	帧大小 (字节)					
	16	**64**	**256**	**1024**	**8192**	**16384**
aes-128-gcm	893	2,673	5,379	7,776	9,090	9,212
aes-256-gcm	848	2,470	4,760	6,895	8,001	8,105
aes-128-ctr	1,279	3,860	8,178	11,782	13,742	13,800
sha256	176	651	2,124	4,860	7,772	8,091
sha512	98	392	881	1,494	1,905	1,942
chacha20-poly1305	571	1,245	2,413	3,014	3,192	3,212

公钥加密的进一步基准测试如 表 2-8 中所示。使用命令 *openssl speed -elapsed -multi 2<algorithm>* 可运行测试。

表 2-8. 公钥加密基准测试

RSA	**大小**	**512**	**1024**	**2048**	**3072**	**4096**
	签名/秒	7,357	1,511	230	74	33
	验证/秒	86,983	29,394	8,515	3,888	2,232
	加密/秒	70,260	26,811	8,091	3,810	2,198
	解密/秒	6,248	1,450	226	74	33
ECDSA	**曲线**	**nistp224**	**nistp256**	**nistp521**	**nistk233**	**nistb233**
	签名/秒	802	8,218	103	638	607
	验证/秒	921	3,344	137	326	311

3 存储器系统基准测试

本节包含涉及 Arm-Cortex 处理器内核和片上系统 (SoC) 存储系统的基准测试。包括综合基准测试，例如 LMBench 和 CoreMark-Pro。

3.1 存储器带宽和延迟

存储器系统基准测试的一个子集是 LMBench 和 STREAM，用于测量软件实现的存储器带宽和延迟。

3.1.1 LMBench

LMBench 是一套适用于处理器内核和操作系统基元的微基准测试工具。存储器带宽和延迟相关测试非常适用于现代嵌入式处理器。每次运行的结果略有不同 (<10%)。

LMBench 基准测试 *bw_mem* 测量实现的存储器复制性能。通过使用参数 *cp*，基准测试执行数组复制，*bcopy* 参数使用运行时 glibc 版本的 *memcpy()* 标准函数。利用 SIMD 等实现更高性能，在实施高度优化的基础上进行 glibc 实践。等于或小于给定级别高速缓存大小的 size 参数可测量进行典型的 for 循环或 *memcpy()* type 操作的软件可实现的存储器带宽。通常用于计算外部存储器带宽。带宽根据字节读写 (每读写 1 字节计为 1) 计算，结果约为 STREAM 复制结果的一半。此基准测试还允许利用 -P 参数创建并行线程。为了获得较大多核存储器带宽，需要创建的线程数量应等同于操作系统可用的内核数量，即 2 个 (AM62Lx Linux (-P 2))。为了显示 AM62Lx 的完整性能特征，LMBench 测试是在内核数量和时钟频率的完整阶乘组合上实施的。下面的代码块展示了执行 LMBench 命令的终端打印输出。

```
root@am62lxx-evm:~# bw_mem 8M bcopy 8.00 1000.00 root@am62lxx-evm:~# bw_mem -P 2 8M bcopy 8.00
1127.87 root@am62lxx-evm:~# bw_mem 8M cp 8.00 579.54 root@am62lxx-evm:~# bw_mem -P 2 8M cp 8.00
645.71
```

表 3-1 展示了相对于理论线速测得的带宽和效率。使用的线速计算方式为：LPDDR4 MT/s 速率 x 宽度 ÷ 2 (构成复制的读取和写入均会消耗总线) 。

$$Efficiency = \frac{Measured\ Speed}{\frac{LPDDR4\ MT/s \times width}{2}} = \frac{Measured\ Speed}{\frac{1600 \times 2\ B}{2}} = \frac{Measured\ Speed}{1600} \quad (1)$$

表 3-1. LMBench 结果

命令	说明	Arm-Cortex-A53 (1.25GHz)，LPDDR4（1600MT/s、16 位）[MB/s]	LPDDR4 效率 [%]
Bw_mem 8M bcopy	*单核，glibc memcpy*	1,000	62
bw_mem -P 2 8M bcopy	*双核，glibc memcpy*	1,127	70
Bw_mem 8M cp	*单核，内联复制循环*	579	36
bw_mem -P 2 8M cp	*双核，内联复制循环*	645	40

LMBench 基准测试 *lat_mem_rd* 用于测量外部存储器 (AM62Lx LPDDR4) 观察到的存储器存取延迟和高速缓存命中率。有两个参数，分别是事务大小（64，如以下代码块所示）和读取跨度 (512)。选择这两个数值来测量高速缓存和外部存储器的延迟，而不是处理器数据预取器或其他推测性执行的延迟。存取模式可实现预取，但此基准测试特别适用于无法实现预取的存取模式下的相关测量。

下面的代码块展示了执行 *lat_mem_rd* 命令的终端打印输出。左列是数据存取模式的大小（单位为兆字节），右侧是往返读取延迟（单位为纳秒）。此命令在 1.25GHz 的 Arm-Cortex-A53 时钟频率下执行。

```
root@am62lxx-evm:~# lat_mem_rd 64 512
"stride=512
0.00049 2.405
0.00098 2.404
0.00195 2.404
0.00293 2.405
0.00391 2.404
0.00586 2.404
0.00781 2.405
0.01172 2.404
0.01562 2.405
0.02344 2.525
0.03125 5.903
0.04688 7.172
0.06250 8.453
0.09375 9.674
0.12500 10.239
0.18750 10.838
0.25000 30.115
0.37500 106.020
0.50000 155.871
0.75000 178.770
1.00000 181.088
1.50000 182.258
2.00000 182.664
3.00000 182.958
4.00000 182.947
6.00000 183.280
8.00000 183.235
12.00000 183.265
16.00000 183.375
24.00000 183.149
32.00000 183.076
48.00000 183.513
64.00000 183.483
```

图 3-1 展示了 1.25GHz 下存储器延迟结果的连接散点图。基于存储器块大小（x 轴），该图可分为三个区域。第一个区域是被存取的存储器块小于 L1 高速缓存时。假定数据完全位于 L1 内部，因此该区域中的这种延迟是 L1 高速缓存延迟的近似估计值。第二个区域是被访问的存储器块大于 L1 但小于 L2 高速缓存时。该区域中的延迟是 L1、L2 和 LPDDR4 延迟的混合。可以假设该区域中间的延迟是 L2 延迟的近似表示。第三个区域是访问的存储器块大于 L2 高速缓存时。该区域中的最后一个读数反映了 LPDDR4 延迟。

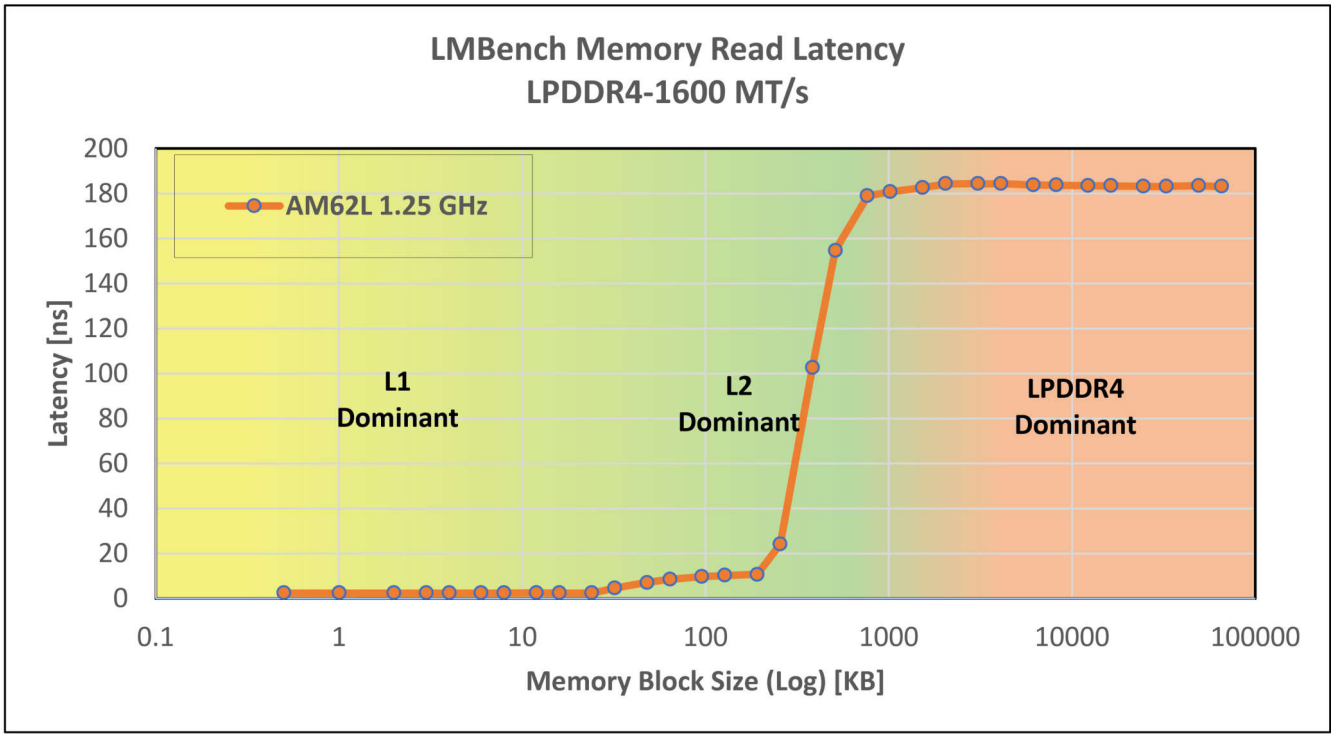

图 3-1. 存储器读取延迟

表 3-2 展示了 Arm-Cortex-A53 读取延迟的摘要。

表 3-2. 存储器读取延迟结果

存储器	Arm-Cortex-A53 (1.25GHz) [ns]
L1 高速缓存	2.4
L2 高速缓存	10.2
LPDDR4-1600 MT/s	183.5

延迟：lat_mem_rd-stride128-szN，其中 N 等于或小于给定级别的缓存大小，用于测量缓存未命中惩罚。至少是最后一级缓存两倍大小的 N 是外部存储器的延迟。

带宽：bw_mem_bcopy-N，其中 N 等于或小于给定级别的缓存大小，用于测量执行 memcpy() 类型操作的软件所能达到的存储器带宽。通常用于计算外部存储器带宽。带宽根据字节读写（每读写 1 字节计为 1）计算，结果约为 STREAM 复制结果的一半。

3.1.2 STREAM

STREAM 是测量数据存储器系统性能的微基准测试，无需重复使用任何数据。STREAM 旨在不命中高速缓存，执行数据预取和推测性存取。STREAM 使用双精度浮点（64 位），但在大多数现代处理器中，存储器访问是瓶颈。四个单项分数包括 copy（复制）、scale（乘常数）、add（数字相加）及 triad（乘法累加）。

- Copy：在不进行算术运算的情况下测量存储器传输速率，a[i] = b[i]
- Scale：包括一个简单的算术运算，a[i] = k × b[i]
- Add：除算术运算之外，还包含三个存储器存取，a[i] = b[i] + c[i]
- Triad：将 scale 和 add 组合到一个运算中，a[i] = b[i] + k × c[i]

对于带宽，每读取一个字节计数为 1，每写入一个字节计数为 1，得到的分数是 LMBench 带宽的两倍。表 3-3 展示了相对于理论线速测得的带宽和效率。使用的线速是 LPDDR4 MT/s 与宽度的乘积。为了获得总体最大吞吐量，使用命令 *stream -M 16M -P 2 -N 10*，这意味着两个并行线程和 10 次迭代。在此测试中，Arm-Cortex-A53 时钟频率设置为 1.25GHz。

```
root@am62lxx-evm:~# stream -M 16M -P 2 -N 10
STREAM copy latency: 13.64 nanoseconds
STREAM copy bandwidth: 2346.27 MB/sec
STREAM scale latency: 13.59 nanoseconds
STREAM scale bandwidth: 2354.55 MB/sec
STREAM add latency: 21.72 nanoseconds
STREAM add bandwidth: 2209.49 MB/sec
STREAM triad latency: 22.20 nanoseconds
STREAM triad bandwidth: 2162.58 MB/sec
```

表 3-3. 流基准测试

	LPDDR4-1600MT/s-16 位延迟 [ns]	LPDDR4-1600MT/s-16 位带宽 [MB/s]	LPDDR4-1600MT/s-16 位效率 [%]
copy	13.64	2,346	73
scale	13.59	2,354	73
add	21.72	2,209	69
triad	22.20	2,162	67

3.2 临界存储器访问延迟

本节提供从 AM62Lx 中的处理器到系统中的各种存储器目标的往返读取延迟测量。测量是使用裸机芯片验证测试在 AM62Lx 平台上进行的。测试在使用 LPDDR4 的 A53 处理器上执行。每个测试包括一个由 8192 次迭代组成的循环，可读取总计 32KiB 的数据。每次访问的周期数被计数并除以相应的处理器时钟频率以获得延迟时间。

出于进行此延迟测量的目的，请禁用所有自动时钟门控。默认情况下，自动时钟门控（不启用时钟门控）已启用，以节省互连系统的功耗，但会对延迟性能造成轻微影响（约十几纳秒）。自动时钟门控以分布式方式完成，每个 IP 都有一个或多个不启用时钟门控的控制。这些参数通常由器件管理在初始化时以及进入和退出低功耗模式时进行配置。表 3-4 展示了平均延迟结果。

测试在以下条件下完成：1.25Ghz A53 内核和 1600MT/s LPDDR4。ARM 架构提供本地内部低延迟路径，还允许通过 SoC 总线基础设施对外部存储器进行访问。

表 3-4. A53 的关键存储器访问延迟

存储器	Arm-Cortex-A53（平均 ns）	SoC 地址
LPDDR4	155	0x80000000
OCSRAM MAIN	42	0x70800000
OCSRAM WKUP	108	0x707f0000

3.3 UDMA：DDR 至 DDR 数据复制

本节提供了使用正常容量 (NC) UDMA 通道进行 DDR 至 DDR 块复制的测试结果和观察结论。表 3-5 详述了此操作。

表 3-5. UDMA 通道类别

	说明
正常容量 (NC)	提供了基本数量的描述符和 TR 预取以及 Tx/Rx 控制和数据缓冲。非常适用于与片上存储器和 DDR 通信的大多数外设传输。缓冲区大小为 192B 时，此 FIFO 深度允许每个传输周期进行 3 次数据突发为 64B 的读取事务。

以下测量结果是通过使用 DDR 的 A53 上的裸机芯片验证测试收集的。传输描述符和环位于 DDR 中。测试在以下条件下完成：0.75V VDD_CORE、1.25Hz A53 内核和 1600MT/s LPDDR4。传输尺寸范围为 1KiB 至 512KiB。

NC UDMA 通道在缓冲区大小高达 512KiB 时的传输容量和延迟如 表 3-6 所示。

表 3-6. UDMA：DDR 至 DDR 块复制

缓冲区大小 (KiB)	NC 通道带宽 (MiB/s)	NC 通道延迟 (µs)
1	204.73	4.77
2	283.47	6.89
4	349.40	11.18
8	387.91	20.14
16	420.48	37.16
32	436.33	71.62
64	444.49	140.61
128	448.77	278.54
256	450.82	554.54
512	452.10	1105.96

4 总结

本文档介绍了 AM62L 处理器的一组基准测试。本文档中提供的基准测试侧重于处理器内核、存储器和关键外设以及延迟性能。大多数性能基准测试也会在 SDK 文档中发布。本文档还包括用于重现结果的步骤。

5 参考资料

- EEMBC、CoreMark-Pro、网页
- STREAM McCalpin, John D. (1991-2007) *STREAM: "高性能计算机中的可持续存储器带宽"*，持续更新技术报告。资料来源：http://www.cs.virginia.edu/stream/
- Github、Ne10 数学库、网页
- OpenSSL、OpenSSL、网页
- 德州仪器 (TI)，AM62Lx Sitara 处理器数据表

重要通知和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的相关应用。 严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265