Application Note

Sitara™ AM62P 基准测试

Qutaiba Saleh, Jonathan Bishop, Andrew Shutzberg, Krunal Bhargav, and Cooper Kelley

摘要

本应用手册介绍了一系列基准，用于测量 AM62Px 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

插图清单

表格清单

商标

1 引言

AM62Px 包含多达四个具有 64 位架构的 Arm®-Cortex®-A53 内核、一个 Cortex-R5F MCU 内核、一个 Cortex-R5F 器件管理内核以及各种其他特性（包括多高清显示支持、3D 图形加速、4K 视频加速和广泛的外设），非常适合各种汽车和工业应用（包括汽车数字仪表、汽车显示屏和工业 HMI 等）。AM62Px 支持 LPDDR4 32 位宽度，速度为 3200MT/s。图 1-1 是 AM62Px 的功能方框图。有关详细信息，请参阅 *AM62Px Sitara 处理器数据表*。

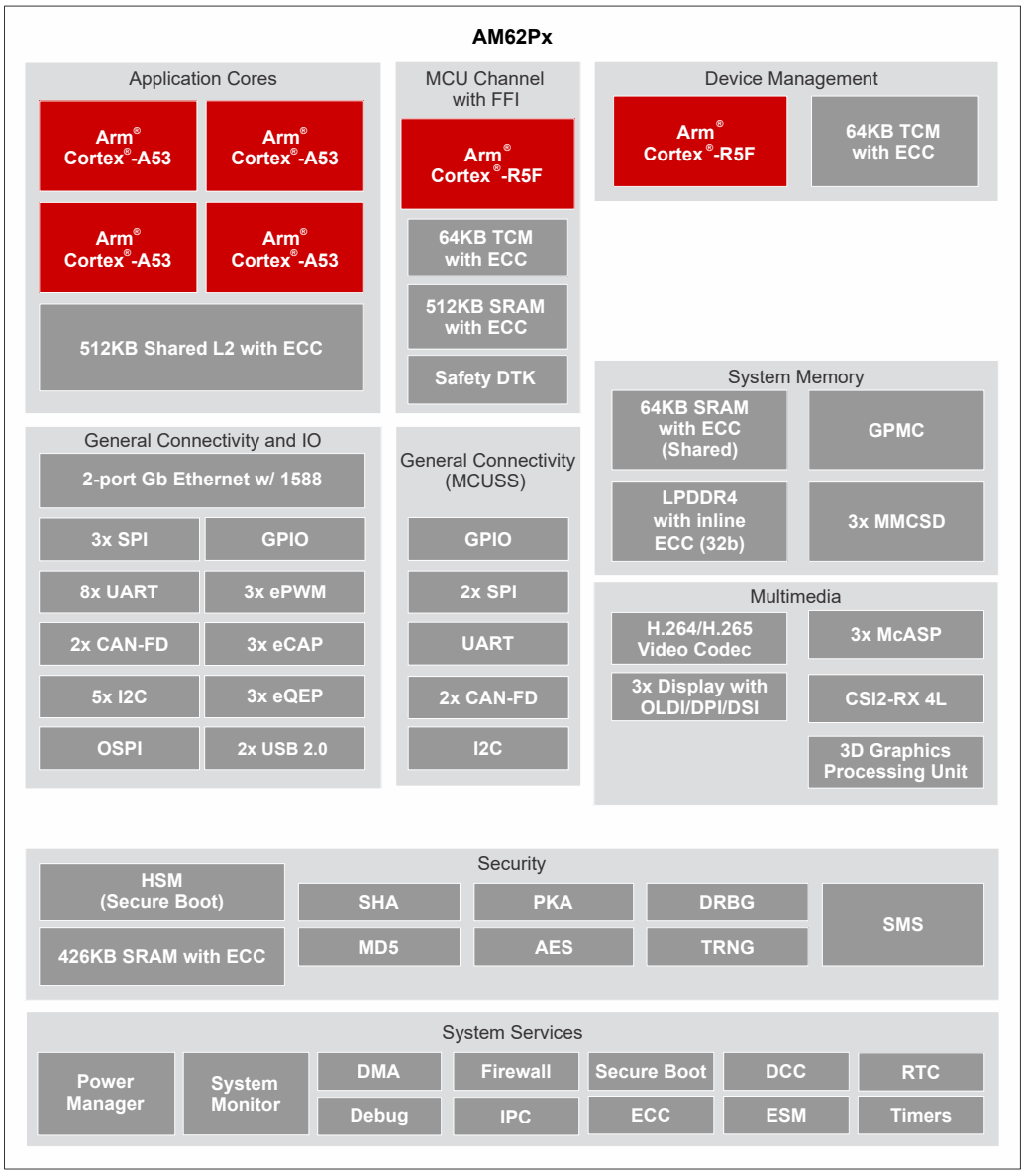

图 1-1. AM62Px 功能方框图

本文档介绍了在 AM62Px 处理器上测量的一系列业界通用的应用特定基准，这些测试关注 Arm-Cortex-A53 内核和 LPDDR4 存储器的性能，并一些具有面向 Arm-Cortex-R5F MCU、GPU 和其他元件的应用特定基准。评估板的关键参数是 Cortex-A53 内核有 1.25GHz 和 1.4GHz 时钟速度，LPDDR4 为 32 位宽和 3200MT/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。所有基准测试均使用 Linux SDK 9.02 实施。

1.1 更改 Cortex-A53 时钟频率

AM62Px 上 A53 内核的最大频率取决于器件的速度等级和施加的 VDD_CORE 电压。为了获得器件的峰值性能，在部分基准测试中，修改了设置以在 1.4GHz 下运行 AM62Px 上的 A53 内核。

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 1 亿多次迭代。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@am62pxx-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
```

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情。AM62Ax 具有四个 A53 内核，以 1.25GHz 和 1.4GHz 运行时的总分数分别为 14,229DMIPS 和 16,261DMIPS。

表 2-1. Dhrystone 基准测试

	Arm-Cortex-A53 (1.25GHz)	Arm-Cortex-A53 (1.4GHz)
Dhrystones/秒	6,250,000	7,142,857
归一化 dhrystones（除以参考 1MIPS 机器的 1757）	3,557	4,065
每个核心的 DMIPS/MHz	~3	~3
编译器和标志	GCC 11.4 -march=ARMv8 -O3	
操作系统	Linux 6.1.80 (2023)	

2.2 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，可以从官方托管网站下载。在此测试中，直接克隆代码并将其内置在 AM62Px 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@am62pxx-evm:~# git clone https://github.com/eembc/coremark-pro.git
```

2. 构建 CoreMark-Pro

```
root@am62pxx-evm:~# cd coremark-pro/
root@am62pxx-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@am62pxx-evm:~/coremark-pro# make TARGET=linux64 certify-all XCMD='-c4'
```

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 2-2 展示了单核、双核和四核 A53 在 1.25GHz 和 1.4GHz 下的 CoreMark-Pro 结果。

表 2-2. CoreMark®-Pro 结果

	Arm-Cortex-A53 (1.25GHz) [iter/s]	并行缩放	Arm-Cortex-A53 (1.4GHz) [iter/s]	并行缩放
单核	850	1	936	1
双核	1,531	1.82	1,700	1.82
四核	2,426	2.88	2,654	2.83

2.3 快速傅里叶变换

快速傅里叶变换 (FFT) 是最常见的信号处理算法之一。表 2-3 显示了 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 2-3. NE10 CFFT 基准测试

	Arm-Cortex-A53 (1.25GHz)（单线程/内核）	Arm-Cortex-A53 (1.4GHz)（单线程/内核）
1024 点复数 FFT 执行时间	35µs	31µs

2.4 加密基准测试

AM62Px 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 AM62Px 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 2-4 展示了在 AM62Px 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 4*。这利用了全部四个 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 1.4GHz。openssl 命令的输出以 KB/s 为单位。为了满足所需的行业标准，表 2-4 中所报告的结果被转换为 Mb/s。

表 2-4. 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	2,035	6,082	12,285	17,755	20,806	21,033
aes-256-gcm	1,945	5,652	10,906	15,271	18,231	18,485
aes-128-ctr	3,205	9,477	19,306	27,385	31,360	31,597
sha256	412	1,531	4,950	11,190	17,774	18,538
sha512	240	956	2,083	3,496	4,365	4,441

表 2-4. 对称加密和安全哈希（单位为 Mbit/s）（续）

	帧大小（字节）					
	16	64	256	1024	8192	16384
chacha20-poly1305	1,306	2,802	5,519	6,876	7,288	7,313

公钥加密的进一步基准测试如表 2-5 中所示。使用命令 *openssl speed -elapsed -multi 4 <algorithm>* 可运行测试。

表 2-5. 公钥加密基准测试

RSA	大小	512	1024	2048	3072	4096
	签名/秒	16,359	3,419	520	171	76
	验证/秒	202,801	67,920	19,394	8,950	5,124
ECDSA	曲线	nistp224	nistp256	nistp521	nistk233	nistb233
	签名/秒	1,884	23,010	240	1,467	1,395
	验证/秒	2,124	7,859	315	743	707

2.5 IPC 邮箱延迟

AM62Px 器件使用邮箱 IP 作为处理器间通信 (IPC) 的主要方法之一。邮箱模块通过提供排队的邮箱中断机制，促进器件各片上处理器之间的通信。

排队的邮箱中断机制允许软件通过一组寄存器和具有 32 位小有效载荷的相关中断信号，在多个处理器（用户）之间建立通信通道。[1] 邮箱由 8 组 FIFO（集群）组成，支持最多 4 个用户之间的双向通信。

每个集群包含一系列（16 个）FIFO，每个 FIFO 支持多达 4 个用户之间的单向通信。每个 FIFO 最多可保存 4 条 32 位消息。

测量是使用裸机芯片验证测试在 AM62Px 平台上进行的。R5F 内核通过本地 TCM 运行，而 A53 内核通过 DDR 运行。每次测试都包含 32 个发送/接收迭代的循环，并对其结果求平均值。使用了两种处理接收到的消息的方法：一种使用处理器中断（如表 2-6 所示），另一种使用轮询（如表 2-7 所示）。

表 2-6. 采用中断的 IPC 32 位延迟

32 位发送/接收平均延迟 (ns)			
发送内核	**接收内核**		
	A53	**R5F MCU**	**R5F WKUP**
A53	693	409	340
R5F MCU	745		309
R5F WKUP	695	331	

表 2-7. 采用轮询的 IPC 32 位延迟

32 位发送/接收平均延迟 (ns)			
发送内核	**接收内核**		
	A53	**R5F MCU**	**R5F WKUP**
A53	521	502	471
R5F MCU	445		497
R5F WKUP	508	361	

[1] 邮箱中断的静态路由如 AM62Px Sitara 处理器技术参考手册中的表 4-48 所示。

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。数学函数基准测试包括快速傅里叶变换 (FFT)。

3.1 存储器带宽和延迟

LMBench 和 STREAM 的子集是用于测量实现的存储器带宽和软件延迟的基准测试程序。

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 4 个 (AM62Px Linux (-P 4))。为了显示 AM62Px 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@am62pxx-evm:~# bw_mem 8M bcopy
8.00 1956.71
root@am62pxx-evm:~# bw_mem -P 2 8M bcopy
8.00 3122.66
root@am62pxx-evm:~# bw_mem -P 4 8M bcopy
8.00 3605.80

root@am62pxx-evm:~# bw_mem 8M cp
8.00 1012.27
root@am62pxx-evm:~# bw_mem -P 2 8M cp
8.00 1568.78
root@am62pxx-evm:~# bw_mem -P 4 8M cp
8.00 1874.32
```

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) 。

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

表 3-1. LMBench 结果

命令	说明	Arm-Cortex-A53 (1.25GHz)，LPDDR4（3200MT/s、32 位）[MB/s]	LPDDR4 效率 [%]	Arm-Cortex-A53 (1.4GHz)，LPDDR4（3200MT/s、32 位）[MB/s]	LPDDR4 效率 [%]
Bw_mem 8M bcopy	*单核，glibc memcpy*	1,911	30	1,956	31
bw_mem -P 2 8M bcopy	*双核，glibc memcpy*	3,052	48	3,122	49
bw_mem -P 4 8M bcopy	*四核，glibc memcpy*	3,571	56	3,605	56
Bw_mem 8M cp	*单核，内联复制循环*	1,003	16	1,012	16
bw_mem -P 2 8M cp	*双核，内联复制循环*	1,562	24	1,568	25
bw_mem -P 4 8M cp	*单核，内联复制循环*	1,862	29	1,874	29

LMBench 基准测试 *lat_mem_rd* 用于测量外部存储器 (AM62Px 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 和 1.4GHz 的 Arm-Cortex-A53 时钟频率下执行。

```
root@am62pxx-evm:~# lat_mem_rd 64 512
"stride=512
0.00049 2.145
0.00098 2.145
0.00195 2.145
0.00293 2.145
0.00391 2.145
0.00586 2.145
0.00781 2.145
0.01172 2.145
0.01562 2.145
0.02344 2.289
0.03125 4.179
0.04688 6.494
0.06250 7.748
0.09375 8.626
0.12500 9.283
0.18750 9.764
0.25000 9.976
0.37500 11.156
0.50000 33.705
0.75000 94.007
1.00000 126.437
1.50000 142.957
2.00000 145.089
3.00000 146.336
4.00000 147.029
6.00000 147.626
8.00000 147.867
12.00000 148.112
16.00000 148.169
24.00000 148.243
32.00000 148.219
48.00000 148.284
64.00000 148.291
```

图 3-1 展示了 1.25GHz 和 1.4GHz 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。

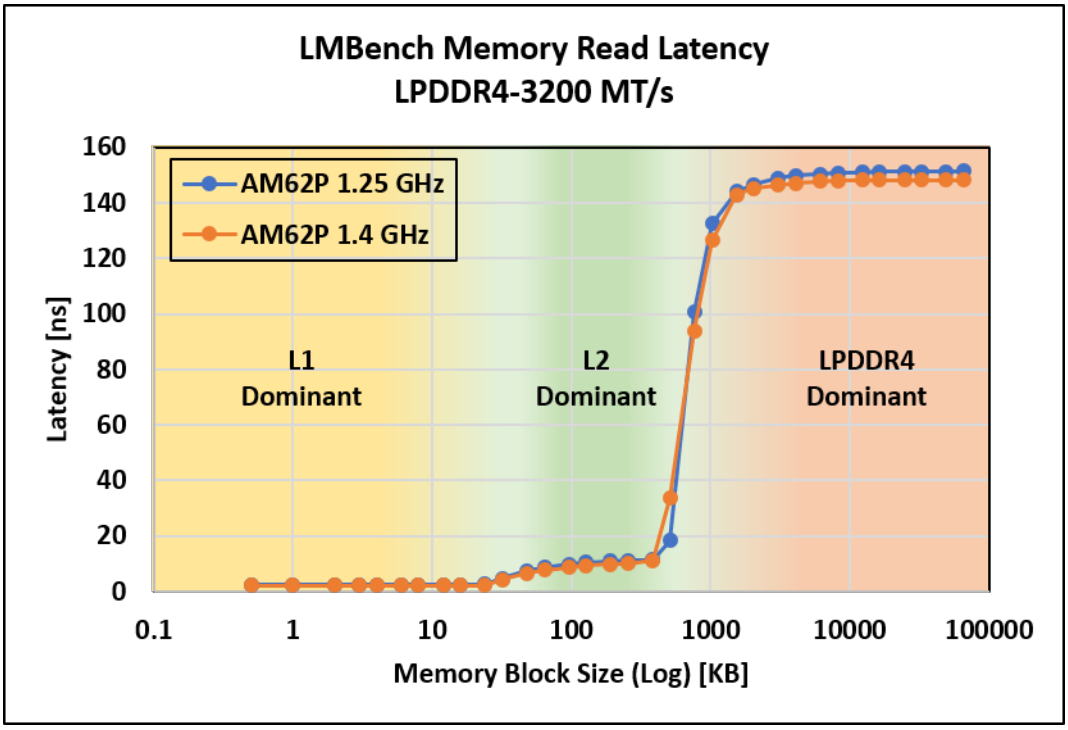

图 3-1. 存储器读取延迟

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]	Arm-Cortex-A53 (1.4GHz) [ns]
L1 高速缓存	2.4	2.1
L2 高速缓存	10.4	9.2
LPDDR4-3200MT/s	151.3	148.2

3.1.2 STREAM

STREAM 是测量数据存储器系统性能的微基准，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 4-N 10*，这意味着四个并行线程和 10 次迭代。在此测试中，Arm-Cortex-A53 时钟频率设置为 1.4GHz。

表 3-3. 流基准测试

	LPDDR4-3200MT/s-32 位带宽 [MB/s]	LPDDR4-3200MT/s-32 位效率 [%]
copy	7,316	57
scale	7,274	57
add	6,401	50

表 3-3. 流基准测试（续）

	LPDDR4-3200MT/s-32 位带宽 [MB/s]	LPDDR4-3200MT/s-32 位效率 [%]
triad	6,059	47

3.2 临界存储器访问延迟

本节提供从 AM62Px 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 AM62Px 平台上进行的。测试在使用 LPDDR4 的 A53 和 R5F 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。表 3-4 展示了平均延迟结果。

表 3-4. A53、R5F MCU 和 R5F WKUP 的临界存储器访问延迟

存储器	Arm-Cortex-A53（平均 ns）	Arm-Cortex-R5F MCU（平均 ns）	Arm-Cortex-R5F WKUP（平均 ns）
LPDDR4	129	207	173
OCSRAM MAIN	56	120	76
OCSRAM MCU	120	55	80
OCSRAM WKUP	207	193	153
R5F MCU TCM	140	1	180
R5F WKUP TCM	104	111	1

测试在以下条件下完成：0.75V VDD_CORE、1.25GHz A53 内核、800MHz R5F 内核和 3200MT/s LPDDR4。紧耦合存储器，即 TCM，是直接与 ARM Cortex 内核相连的 RAM。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。

3.3 UDMA：DDR 至 DDR 数据复制

本节提供了使用高容量 (HC) 和正常容量 (NC) UDMA 通道进行 DDR 至 DDR 块复制的测试结果和观察结论。有关详细信息，请参阅表 3-5。

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。
高容量 (HC)	提供更多的描述符和 TR 预取以及自定义 Tx/Rx 控制和数据缓冲。非常适用于需要中等每通道带宽和显著增加的数据吞吐量的应用。由于缓冲区大小增加为 512B，此 FIFO 深度允许每个传输周期进行 8 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、1.25GHz A53 内核、800MHz R5F 内核和 3200MT/s LPDDR4。传输尺寸范围为 1KiB 至 512KiB。

表 3-6. UDMA：DDR 至 DDR 块复制

缓冲区大小 (KiB)	HC 通道带宽 (MiB/s)	NC 通道带宽 (MiB/s)	HC 通道延迟 (µs)	NC 通道延迟 (µs)
1	121.92	96.21	8.01	10.15
2	188.16	157.51	10.38	12.40
4	369.56	237.32	10.57	16.46
8	542.16	312.75	14.41	24.98
16	711.20	381.94	21.97	40.91
32	895.93	426.91	34.88	73.20
64	985.03	452.31	63.45	138.18
128	1049.36	464.93	119.12	268.86
256	1087.10	472.64	229.97	528.94
512	1105.71	476.06	452.20	1050.29

表 3-6 显示了 HC 和 NC 通道的传输容量，并表明高容量通道比正常容量通道获得的带宽高多达 2.3 倍。

4 图形处理单元基准测试

AM62P 配备了一个具有 3D 图形能力的 GPU，如表 4-1 所示。

表 4-1. AM62P GPU 规格

特性	值
GPU 内核	带有 256KB 高速缓存的 IMG BXS-4-64
GFLOPS	50
3D API	OpenGL ES 3.2 和 Vulkan 1.2

4.1 Glmark2

Glmartk2 用于对 AM62P 上的 GPU 性能进行基准测试。该测试包含在 SDK 中。该测试针对 DRM 和 Wayland 显示类型实施。表 4-2 显示了用于运行测试的命令及测试结果。

表 4-2. Glmark2 结果

测试	分数
glmark2-es2-drm --off-screen	298
glmark2-es2-wayland --off-screen	744

4.2 GFXBench5

表 4-3 中显示的结果是从 GFXBench5 性能测试中收集的。

表 4-3. GFXBench5 结果

测试	FPS
Manhattan Offscreen	14.72
Trex Offscreen	27.56
Egypt Offscreen	49.15

5 视频编解码器

本节介绍适用于 AM62P 中视频编解码器的基准测试。具体而言，该基准测试侧重于使用 H264 编码器和解码器。结果是从接近实际用例场景中收集的，场景中包括使用 gstreamer 管道通过 UDP 传输实时视频流。此设置使用分辨率为 1920x1080、帧率为 30FPS 的 USB 摄像头。发送器侧从摄像头采集实时视频流，使用 H264 格式编解码器加速器对视频流进行编码，然后通过 UDP 传输视频。另一侧通过 UDP 接收视频流，进行解码并将其显示在屏幕上。发送器和接收器侧在同一 AM62P 器件上执行。这显示了编解码器同时执行编码器和解码器的能力。管道中每个元件的延迟可以使用 gstreamer 跟踪器来测量，gstreamer 跟踪器将测量的实时日志输出到您指定的文件。

以下是编码器侧和 UDP 发送器侧的 gstreamer 管道，其中 gstreamer 跟踪器配置为在“/run/trace_encode.log”中记录延迟测量值。

```
GST_TRACERS="latency(flags=pipeline+element)" GST_DEBUG=GST_TRACER:7 GST_DEBUG_FILE="/run/
trace_encode.log" \
gst-launch-1.0 \
v4l2src device=/dev/video2 ! image/jpeg, width=1920, height=1080, framerate=30/1 ! jpegdec !
videoconvert ! v4l2h264enc ! h264parse ! rtph264pay ! udpsink host=$1 port=5000 sync=false
```

以下是解码器、UDP 接收器和显示侧的 gstreamer 管道，其中 gstreamer 跟踪器配置为在“/run/trace_decode.log”中记录延迟测量值。

```
GST_TRACERS="latency(flags=element+pipeline)" GST_DEBUG=GST_TRACER:7
GST_DEBUG_FILE=/run/trace_decode.log \
gst-launch-1.0 -v \
udpsrc port=5000 ! 'application/x-rtp, encoding-name=H264, payload=96' !
rtpjitterbuffer latency=50 ! rtph264depay ! h264parse ! v4l2h264dec !
queue ! kmssink driver-name=tidss sync=false plane-id=31
```

/opt/edgai-gst-apps/seripts/gst_tracers/parse_gst_tracers.py 提供的 Python 脚本可用于计算 .log 文件中记录的管道中每个元件延迟测量值的平均值。该脚本可以与 gstreamer 管道并行执行，以显示延迟测量值的实时更新。例如，如下是解码器侧的打印输出。

```
root@am62pxx-evm:/opt/edgeai-gst-apps/scripts/gst_tracers/# parse_gst_tracers.py /run/
trace_decode.log
.
.
+-------------------------------------------------------------------------------+
|element            latency        out-latancy   out-fps     frames |
+-------------------------------------------------------------------------------+
|capsfilter0         0.16              27.54        36        30938  |
|rtpjitterbuffer0    0.36              27.53        36        30938  |
|rtph264depay0       0.20              35.86        27        23752  |
|h264parse0          0.26              35.86        27        23752  |
|v4l2h264dec0        48.12             35.86        27        23751  |
|udpsrc0             49.73             35.86        27        23751  |
|queue0              0.75              35.86        27        23751  |
+-------------------------------------------------------------------------------+
```

编码器和解码器的平均延迟测量值如 视频编解码器延迟 所示。

表 5-1. 视频编解码器延迟

编解码器 H264	分辨率	延迟 [ms]
编码器	1920x1080	10.58
解码器	1920x1080	48.12

6 参考资料

- 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
- OpenSSL
- 德州仪器 (TI)： *AM62Px Sitara 处理器数据表* 。

7 修订历史记录

注：以前版本的页码可能与当前版本的页码不同

Changes from Revision * (April 2024) to Revision A (August 2024)	Page
• 更新了节 3.1.2。	9
• 更新了 *临界存储器访问延迟* 中的拼写错误	10
• 更新了以下部分中的拼写错误：*DMA：DDR 至 DDR 数据复制*	10

重要声明和免责声明

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 产品的应用。严禁对这些资源进行其他复制或展示。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