AN14449

i.MX 95 Power Consumption Measurement

Rev. 1.0 — 9 February 2026 **Application note**

Document information

Information	Content
Keywords	AN14449, i.MX 95, IMX95LPD5EVK-19, power consumption, i.MX 95 power domains
Abstract	This application note describes how to measure the current consumption of the i.MX 95 application processor on the NXP i.MX 95 EVK board across various use cases.

1 Introduction

This application note is intended to help system designers to create power-optimized systems. It describes how to measure the current drain of the i.MX 95 application processor on an NXP i.MX 95 19x19 EVK board through different use cases.

Note: *Some use case binaries can be found in AN14449SW. Copy the binaries (Linux application) to the board and use `chmod +x *` to grant the execution permissions.*

Users can choose the appropriate power supply domains for the i.MX 95 processor and become familiar with the expected processor power consumption in various scenarios.

Note: *The reported empirical results in this application note are based on a small sample size only and not to be statically representative.*

2 i.MX 95 power architecture

The power architecture of the chip is established with the presumption that the most affordable systems are built for the scenario in which the PMIC is used to supply all the power rails to the processor.

Figure 1 shows the power architecture diagram for the entire SoC.

Note: *Figure 1 shows only the power supplies and does not show the capacitors that can be required for internal LDO regulators.*

The values in Figure 1 are for reference purpose only. For actual data, see the i.MX 95 Applications Processors Data Sheet for Commercial Products (document IMX95CEC).

From an architectural standpoint, most SoC digital and analog logic can be power-gated in Low-power mode through an external PMIC supply or an internal power switch.

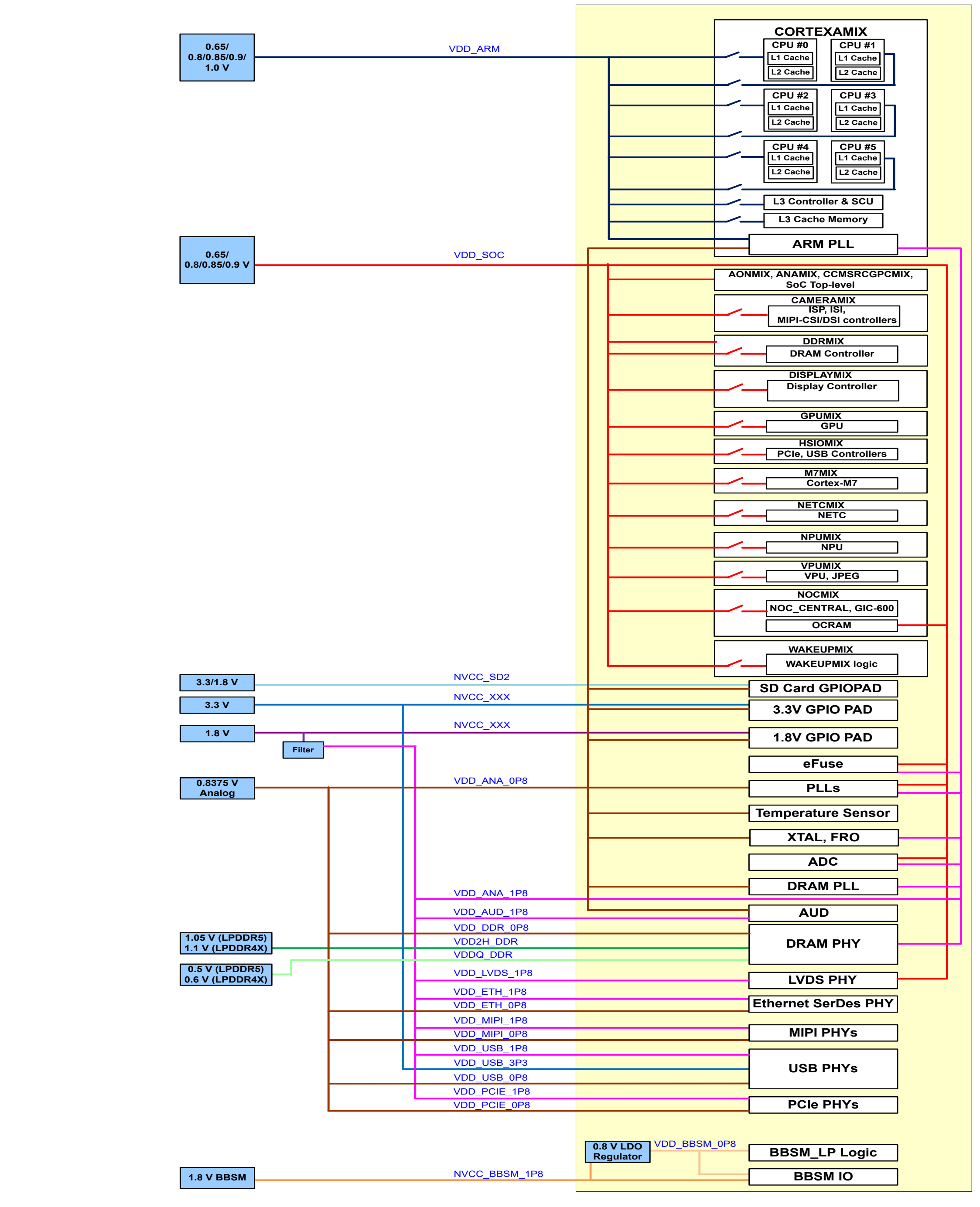

Figure 1. i.MX 95 power architecture

Table 1 summarizes the operating condition for all the external power rails.

Note: *The values in Table 1 are for reference purpose only. For actual values, see the i.MX 95 Applications Processors Data Sheet for Commercial Products (document IMX95CEC).*

Table 1. External power supply

Symbol	Description	Min.	Typ	Max.	Unit
VDD_SOC	Power supply for SoC logic[1], overdrive mode	0.85	0.90	0.955	V
	Power supply for SoC logic[1], nominal mode	0.80	0.85	0.905	V
	Power supply for SoC logic[1], low drive mode	0.76	0.80	0.85	V
	Power supply for SoC logic[1], suspend mode	0.61	0.65	0.7	V
VDD_ARM	Power supply for Cortex-A55 core[1], super overdrive mode	0.95	1.00	1.05	V
	Power supply for Cortex-A55 core[1], overdrive mode	0.85	0.90	0.955	V
	Power supply for Cortex-A55 core[1], nominal mode	0.80	0.85	0.905	V
	Power supply for Cortex-A55 core[1], low drive mode	0.76	0.80	0.85	V
	Power supply for Cortex-A55 core[1][2], suspend mode	0	0	0	V
NVCC_BBSM_1P8	IO supply for GPIO in BBSM bank	1.65	1.8	1.95	V
VDD_DDR_0P8	DDR supply for DDR PHY	0.795	0.8375	0.88	V
VDD_ETH_0P8	Digital supply for Ethernet PHY	0.795	0.8375	0.88	V
VDD_ETH_1P8	I/O voltage supply and analog high voltage power supply	1.71	1.8	1.89	V
VDD_ANA_0P8	Digital supply for PLLs, temperature sensor, and LVCMOS I/O	0.795	0.8375	0.88	V
VDD_ANA_1P8/VDD_ ANAVDET_1P8	1.8 V supply for PLLs, eFuse, Temperature sensor, LVCMOS voltage detect reference, ADC, 24 MHz XTAL, and supply voltage for voltage detect	1.71	1.8	1.89	V
VDD_MIPI_0P8	Digital supply for MIPI PHY	0.795	0.8375	0.88	V
VDD_USB_0P8	Digital supply for USB PHYs	0.795	0.8375	0.88	V
VDD_USB_1P8	1.8 V supply for USB PHYs	1.71	1.8	1.89	V
VDD_USB_3P3	3.3 V supply for USB PHY (Vmax consistent with Vmax supported by NVCC GPIO supplies)	3.069	3.3	3.45	V
VDD_PCI_0P8	Digital supply for PCIe PHY	0.795	0.8375	0.88	V
VDD_PCI_1P8	1.8 V supply for PCIe PHY	1.71	1.8	1.89	V
VDD_LVDS_1P8	1.8 V supply for LVDS	1.71	1.8	1.89	V
VDD_MIPI_1P8	1.8 V supply for MIPI PHYs	1.71	1.8	1.89	V

Table 1. External power supply...*continued*

Symbol	Description	Min.	Typ	Max.	Unit
VDD_AUD_1P8	1.8 V supply for audio transceiver	1.71	1.8	1.89	V
VDD2H_DDR	Voltage supply for LPDDR5/ LPDDR4X PHY, LPDDR5 mode	1.01	1.05	1.12	V
VDD2H_DDR	Voltage supply for LPDDR5/ LPDDR4X PHY, LPDDR4X mode	1.06	1.1	1.17	V
VDDQ_DDR	Voltage supply for LPDDR5/4 X PHY, LPDDR5 Mode, ODT enabled or disabled	0.47	0.5	0.57	V
VDDQ_DDR	Voltage supply for LPDDR5/4X PHY, LPDDR4X Mode	0.57	0.6	0.65	V
NVCC_AON, NVCC_SD2, NVCC_GPIO, NVCC_ WAKEUP, NVCC_CCM_DAP, NVCC_ENET	Power supply for GPIO when it is in 1.8 V mode	1.65	1.8	1.95	V
NVCC_AON, NVCC_SD2, NVCC_GPIO, NVCC_ WAKEUP, NVCC_CCM_DAP, NVCC_ENET	Power supply for GPIO when it is in 3.3 V mode	3	3.3	3.45	V

[1] Voltages > Vtyp x 1.05 but < Vmax are only supported if using a PMIC supporting Automatic Voltage Positioning (AVP).
[2] When the SoC is in a low power state, it can save more power when configuring PMIC to turn VDD_ARM off vs. having the PMIC continue to supply VDD_ARM and rely on the on-die power gating around the CORTEXAMIX components.

3 i.MX 95 processor power measurement

This document provides details of several use cases run by NXP on the NXP IMX95LPD5EVK-19 board to measure i.MX 95 power. These use cases are described in Section 4.

3.1 Hardware and software requirements

Table 2 provides details of the hardware and software used during the power measurement.

Table 2. Hardware and software used

Category	Description
Hardware	NXP IMX95LPD5EVK-19 system based on main board and CPU board revisions (87753 Rev B + 87754 Rev B) respectively
Software	Linux kernel version: LF6.12.49_2.2.0
	Yocto rootfs
	BCU tool is available at BCU version: 1.1.128

The following figure shows the power distribution on the IMX95LPD5EVK-19 board.

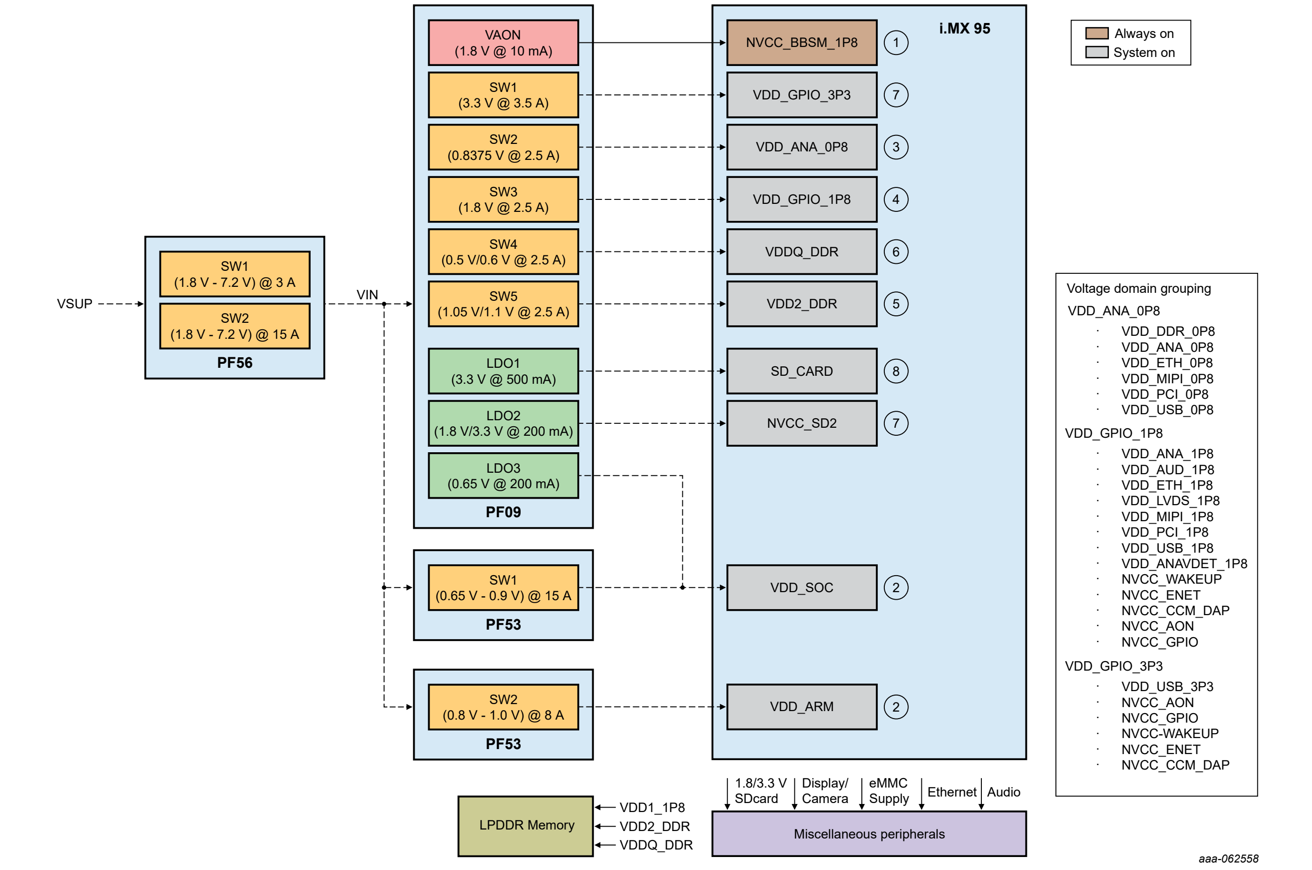

Figure 2. Power distribution on the IMX95LPD5EVK-19 board

Note: *In the used BCU software tool, measurements are performed using the on-board measurement circuitry and these measurements are taken at room temperature without thermal forcing equipment.*

3.2 Building the i.MX Yocto Project

To build the i.MX Yocto Project, perform the steps as follows:

1. To download and build the i.MX Yocto Project community BSP recipe layers, run the following commands:

```
repo init -u https://github.com/nxp-imx/imx-manifest \
-b imx-linux-walnascar -m imx-6.12.49-2.2.0.xml
repo sync
DISTRO=fsl-imx-xwayland
MACHINE=imx95-19x19-lpddr5-evk source imx-setup-release.sh \
-b build-imx95-19x19-lpddr5-evk
```

Note: *For more information on the i.MX Yocto Project, see the i.MX Yocto Project User's Guide (document UG10164).*

2. For some audio or video cases, `gstreamer1.0-libav` is necessary. Put the following commands at the end of the `build-imx95-19x19-lpddr5-evk/conf/local.conf` file:

```
LICENSE_FLAGS_ACCEPTED += "commercial"
IMAGE_INSTALL:append = "gstreamer1.0-libav"
PACKAGECONFIG:append_pn-gstreamer1.0-libav = "x264"
```

3. To build, run the following command:

```
bitbake imx-image-full
```

The build image can be found in `build-imx95-19x19-lpddr5-evk/tmp/deploy/image`.

3.3 Power consumption measurement

To measure the i.MX 95 power consumption, the steps are as follows:

1. Connect a Type-C cable between the host PC and the J31 USB port on the IMX95LPD5EVK-19 base board.
2. To start the monitor in the BCU path, run the following command on the host PC:

```
bcu monitor -board=imx95evk19
```

3. Run the related use cases. These use cases are described under Section 4.
4. To reset the value, press "3" once the use case starts.
5. (Optional) To switch measurement precision: mA/auto/uA, press "4".
6. Wait for 1 minute and record the data from the BCU.

The measurements are taken mainly for the power supply domains shown in Table 3. This table also provides a mapping between the power rails in BCU software, and the power supply domains in the i.MX 95 processor. For more information, download the latest *BCU.pdf*.

Table 3. Measured power supply domains

Power groups	Power supply domains	Description
GROUP_DRAM	lpd5_vdd1	LPDDR5 VDD1 power supply
	lpd5_vdd2	LPDDR5 VDD2 power supply
	lpd5_vddq	LPDDR5 VDDQ power supply
GROUP_SOC_FULL	nvcc_3v3	i.MX 95 3.3 V IO power supply
	nvcc_bbsm_1v8	i.MX 95 BBSM domain power supply
	nvcc_enet_ccm	i.MX 95 ENET and CCM IO power supply
	nvcc_sdio2	i.MX 95 SD2 IO power supply
	nvcc_wakeup	i.MX 95 WAKEUP domain power supply
	vdd2_ddr	i.MX 95 DRAM interface VDD2 power supply
	vdd_ana_0v8	i.MX 95 ETH/MIPI/PCI/USB/ANA 0.8 V analog domain power supply
	vdd_ana_1v8	i.MX 95 ETH/MIPI/PCI/USB/ANA/ ANAVDET/LVDS/AUD 1.8 V analog domain power supply
	vdd_arm	i.MX 95 A55 cores power supply
	vdd_ddr	i.MX 95 DRAM interface VDD power supply
	vdd_soc	i.MX 95 SOC power supply
	vdd_usb_3v3	i.MX 95 USB interface 3.3 V power supply

Table 3. Measured power supply domains*...continued*

Power groups	Power supply domains	Description
	vddq_ddr	i.MX 95 DRAM interface IO VDDQ power supply

Note: *`vddq_ddr` and `lpd5_vddq` are always similar in this version, as `vddq_ddr` and `lpd5_vddq` are short-circuited with SJ3-SJ9 on the back of the SOM board. Separately, the two values hold no significance, and the user is required to sum them to derive the total VDDQ power consumption (i.MX 95 + LPDDR5 device). To get the power consumption of `vddq_ddr` and `lpd5_vddq` separately, make sure SJ3-SJ9 (on the bottom side of the SOM board) are open.*

4 Use cases and measurement results

The main use cases and subcases that form the benchmarks for the i.MX 95 internal power measurements on the EVK platform are described in the following sections.

Note:

- *Before running a use case, `<configuration_script>.sh` must be run to configure the environment, see Section 5.*
- *For all use cases, the platform is booted from eMMC in the U-Boot stage.*
- *The current sample resistors on the power path create a drop in the voltage on each power rail.*
- *Rename the `flash95-a55.bin` occurrences mentioned in the steps from `imx-boot-imx95-19x19-lpddr5-evk-sd.bin-flash_a55` in the release.*
- *By default, the CM33 core enters idle state after power-on reset, but the user can still type "idle" and press **Enter** to make sure that CM33 enters idle state.*
- *When the case is running, the case clock summary can be obtained by using the following command: `cat /sys/kernel/debug/clk/clk_summary`.*

Table 4 summarizes the power measurement results of various use cases performed on the IMX95LPD5EVK-19 board.

Table 4. i.MX 95 EVK power summary report

Use cases category	Use cases	Total power (sum of average powers in `GROUP_SOC_FULL`) (mW)
Core benchmark use cases	CA55 CoreMark	2390.34
	CA55 + CM7 CoreMark	2468.57
	Dhrystone	2810.06
Memory use cases	Memcpy	2462.33
	Memset	2677.53
	Stream	3825.18
Audio/video playback use cases	Audio Playback	1384.48
	Video playback local 1080P	1888.64
	Video playback local 4K	2059.34
	Video playback streaming 1080P	1929.38
	Video playback streaming 4K	2094.23
GPU use case	GLMark	2625.42

Table 4. i.MX 95 EVK power summary report*...continued*

Use cases category	Use cases	Total power (sum of average powers in GROUP_SOC_FULL) (mW)
	Vkmark	1882.65
Machine learning use cases	eIQ benchmark (CPU)	2094.49
	eIQ benchmark (GPU)	2546.60
	eIQ benchmark (NPU)	2792.36
Storage use cases	DD_READ_eMMC	1960.51
	DD_WRITE_eMMC	1788.86
	DD_READ_SD	1809.81
	DD_WRITE_SD	1673.73
	DD_READ_SSD	2091.17
	DD_WRITE_SSD	2056.15
	DD_READ_USB	1903.62
	DD_WRITE_USB	1517.27
Low-power mode use cases	System Idle with display OD mode	1624.21
	System Idle without display OD mode	1185.52
	System in DSM	25.65
	BBSM	0.13
Stress test use cases	5 x CA55 CPU Inference + 1 x CA55 Dhryst + NPU + GPU + CM7 CoreMark	5145.35
	6xCA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU	4295.14
Product use cases	CM33 + CM7 CoreMark on DDR	921.69
	Telematics	2139.94
	Smart Home Appliance	3439.39
	Remote Camera (AP1302)	1796.58
	Surround view (OX03C10)	3155.11
	Digital Connected Cluster (DCC)	3080.79
Linux Suspend use cases	Linux Suspend + CM7 CoreMark (TCM)	197.24
	Linux Suspend + CM7 WFI	178.48
	Linux Suspend + CM7 FlexCAN transaction	659.71
	Linux Suspend + CM7 NETC	956.17
	Linux Suspend + WOL (Wake-on-LAN)	342.72
ISP use case	ISP Dual Camera (4K)	2433.56
	ISP Single Camera (4K)	2031.06

4.1 Core benchmark use cases

The following use cases scenarios have been tested with Cortex-A55 cores:

- CA55 CoreMark
- CA55 + CM7 CoreMark
- Dhrystone

4.1.1 CA55 CoreMark

This CA55 CoreMark use case measures power consumption when the CA55 CoreMark is running. The CoreMark is a modern and sophisticated benchmark used to accurately measure the processor performance and is intended to replace the older Dhrystone benchmark. Arm recommends to use the CoreMark over Dhrystone. The test is performed in a loop at a frequency of 1.8 GHz, with each core running CoreMark. The CPU frequency governor is set to the performance, and the DDR operates at a data rate of 6400 MT/s. The CM7 image is not loaded.

Table 5 shows the system clock and the usage for this use case.

Table 5. Summary for CA55 CoreMark

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Running CoreMark	Clock gating	Power Down	Active

To measure the power consumption of CoreMark, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`, see Section 5.
5. Run `coremark_loop.sh`.
Note: *To run `coremark_loop.sh`, the `coremark` file is needed. Obtain the file from `AN14449SW.zip`.*
```
while true;
do
  ./coremark > /dev/null 2>&1
done
```
6. Measure the power and record the result.
For the best performance, compile as follows:
```
make XCFLAGS="-DMULTITHREAD=6 -DUSE_PTHREAD -pthread"
```
Table 6 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 6. Measurement results for i.MX 95 19x19 EVK A55 CoreMark loop (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	1.77	3.18	43.52	31
	lpd5_vdd2	1.04	38.56	40.22		
	lpd5_vddq	0.50	0.22	0.11		
GROUP_SOC_FULL	nvcc_3v3	3.30	2.63	8.67	2390.34	

Table 6. Measurement results for i.MX 95 19x19 EVK A55 CoreMark loop (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.69	1.23		
	nvcc_sdio2	3.29	0.22	0.71		
	nvcc_wakeup	1.80	0.82	1.47		
	vdd2_ddr	1.04	0.17	0.17		
	vdd_ana_0v8	0.83	100.20	83.63		
	vdd_ana_1v8	1.79	91.74	164.41		
	vdd_arm	0.92	1385.02	1268.11		
	vdd_ddr	0.83	128.31	106.93		
	vdd_soc	0.92	807.64	742.55		
	vdd_usb_3v3	3.29	1.67	5.49		
	vddq_ddr	0.50	13.46	6.69		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.1.2 CA55 + CM7 CoreMark

This CA55 + CM7 Coremark use case measures power consumption when the CA55 CoreMark and CM7 CoreMark are running. The Coremark is a modern and sophisticated benchmark used to accurately measure the processor performance and is intended to replace the older Dhrystone benchmark. Arm recommends to use CoreMark over Dhrystone. The test is performed in a loop at a frequency of 1.8 GHz, with each CA55 core running Coremark. The CPU frequency governor is set to the performance, and the DDR operates at a data rate of 6400 MT/s. The CM7 core is also running CoreMark at 800 MHz core frequency.

Table 7 shows the system clock and the usage for this use case.

Table 7. Summary for CA55 + CM7 CoreMark

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	800 MHz	6400 MT/s
Usage	Running CoreMark	Clock gating	CoreMark	Active

To measure the power consumption of the CA55 + CM7 CoreMark, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_coremark.bin` to eMMC with `uuu`:

```
uuu -b emmc flash95-a55-m7_coremark.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. On the CA55 console:
   a. Run `setup.sh`. See Section 5.

b. Run `coremark_loop.sh`.
Note: *To run `coremark_loop.sh`, the `coremark` file is needed. Obtain the file from [AN14449SW.zip](AN14449SW.zip).*

```
while true; do
./coremark > /dev/null 2>&1
done
```

5. Open the CM7 serial port. The CoreMark is running in loop automatically.
6. Measure the power and record the results.
Table 8 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 8. Measurement results for i.MX 95 19x19 EVK A55 M7 CoreMark loop (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_ DRAM	lpd5_vdd1	1.79	1.78	3.19	43.09	30
	lpd5_vdd2	1.04	38.15	39.80		
	lpd5_vddq	0.50	0.19	0.09		
GROUP_SOC_ FULL	nvcc_3v3	3.30	2.69	8.86	2468.57	
	nvcc_bbsm_ 1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.66	1.19		
	nvcc_sdio2	3.29	0.23	0.76		
	nvcc_wakeup	1.80	0.82	1.47		
	vdd2_ddr	1.04	0.16	0.17		
	vdd_ana_0v8	0.83	100.45	83.84		
	vdd_ana_1v8	1.79	91.69	164.31		
	vdd_arm	0.92	1386.31	1269.48		
	vdd_ddr	0.83	129.68	108.07		
	vdd_soc	0.92	889.56	817.98		
	vdd_usb_3v3	3.29	1.66	5.48		
	vddq_ddr	0.50	13.47	6.69		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.1.3 Dhrystone

This Dhrystone use case measures power consumption when Dhrystone is running. Dhrystone is a synthetic benchmark used to measure the integer computational performance of processors and compilers, which is executed on six Cortex-A55 cores. The test is performed in a loop at a frequency of 1.8 GHz, with each core running a separate instance of the benchmark since Dhrystone is single-threaded. The CPU frequency governor is set to the performance, and the DDR operates at a data rate of 6400 MT/s. The CM7 image is not loaded.

Table 9 shows the system clock and the usage for this use case.

Table 9. Summary for Dhrystone

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Running Dhrystone	Clock gating	Power Down	Active

To measure the power consumption of Dhrystone, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`. See Section 5.
5. Run `dhrystone_loop.sh`.
   Note: *To run `dhrystone_loop.sh`, the `dhry2` file is needed. Obtain the file from `AN14449SW.zip`.*

```
while true;
do
   taskset -c 0 ./dhry2 &
   taskset -c 1 ./dhry2 &
   taskset -c 2 ./dhry2 &
   taskset -c 3 ./dhry2 &
   taskset -c 4 ./dhry2 &
   taskset -c 5 ./dhry2
done
```

6. Measure the power and record the result.
   Table 10 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 10. Measurement results for i.MX 95 19x19 EVK Dhrystone loop (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	2.05	3.68	45.38	33
	lpd5_vdd2	1.04	39.86	41.58		
	lpd5_vddq	0.50	0.25	0.12		
GROUP_SOC_FULL	nvcc_3v3	3.30	2.59	8.53	2810.06	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.66	1.19		
	nvcc_sdio2	3.29	0.24	0.78		
	nvcc_wakeup	1.80	1.01	1.81		
	vdd2_ddr	1.04	0.18	0.19		
	vdd_ana_0v8	0.83	100.41	83.81		
	vdd_ana_1v8	1.79	92.17	165.21		
	vdd_arm	0.92	1837.10	1680.96		
	vdd_ddr	0.83	133.23	111.03		
	vdd_soc	0.92	809.03	743.89		

Table 10. Measurement results for i.MX 95 19x19 EVK Dhrystone loop (average value)...*continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vdd_usb_3v3	3.29	1.67	5.49		
	vddq_ddr	0.50	13.90	6.91		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after Table 3 for more information.*

4.2 Memory use cases

The following memory-centric use case scenarios have been tested:

- Stream
- Memset
- Memcpy

The memset and memcpy are part of a perf-bench, which is a general framework for benchmark suites.

4.2.1 Stream

This Stream use case focuses on measuring the sustainable memory bandwidth and computation rate of simple vector kernels using the Stream benchmark. It runs all benchmark phases, such as Copy, Scale, Add, and Triad, on the system, which operates with the CPU frequency set to 1.8 GHz in performance mode, a DDR data rate of 6400 MT/s, and the CM7 core powered down.

Table 11 shows the system clock and the usage for this use case.

Table 11. Summary for Stream

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Running Stream	Clock gating	Power Down	Active

To measure the power consumption of the stream, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`. See Section 5.
5. Run `streamcpy_loop.sh`:

```
while true; do
    stream -M 40M -N 5 -P 6
done
```

6. Measure the power and record the results.
   Table 12 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 12. Measurement results for i.MX 95 19x19 EVK stream loop (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.77	56.41	99.97	593.08	34
	lpd5_vdd2	1.04	464.29	482.01		
	lpd5_vddq	0.50	22.30	11.10		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.57	8.48	3825.18	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.83	1.49		
	nvcc_sdio2	3.29	0.23	0.76		
	nvcc_wakeup	1.80	0.81	1.46		
	vdd2_ddr	1.04	7.95	8.27		
	vdd_ana_0v8	0.84	102.51	86.09		
	vdd_ana_1v8	1.79	92.00	164.87		
	vdd_arm	0.91	2072.24	1895.99		
	vdd_ddr	0.83	705.47	587.93		
	vdd_soc	0.92	1121.05	1030.65		
	vdd_usb_3v3	3.29	1.66	5.48		
	vddq_ddr	0.50	67.35	33.45		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.2.2 Memset

This Memset is a suite to evaluate the performance of a memory set in various ways. It runs with the CPU frequency set to 1.8 GHz in performance mode, a DDR data rate of 6400 MT/s, and the CM7 core powered down.

Table 13 shows the system clock and the usage for this use case.

Table 13. Summary for Memset

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Running Memset	Clock gating	Power Down	Active

To measure the power consumption of memset, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot the Linux image with `imx95-19x19-evk.dtb`.

4. Run `setup.sh`, see Section 5.
5. Run `memset_loop.sh`:

```
while true; do
 buff_size=`cat /proc/meminfo | grep CmaFree | awk '{print$2}'`
 perf bench -f simple mem memset -l 20000 -s ${buff_size}KB
done
```

6. Measure the power and record the results.
Table 14 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 14. Measurement results for i.MX 95 19x19 EVK Memset loop (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	38.71	68.88	354.23	32
	lpd5_vdd2	1.04	274.18	285.18		
	lpd5_vddq	0.50	0.36	0.18		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.60	8.57	2677.53	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.78	1.39		
	nvcc_sdio2	3.29	0.23	0.77		
	nvcc_wakeup	1.80	0.82	1.48		
	vdd2_ddr	1.04	6.94	7.22		
	vdd_ana_0v8	0.84	101.39	85.14		
	vdd_ana_1v8	1.79	91.36	163.70		
	vdd_arm	0.92	844.74	774.58		
	vdd_ddr	0.83	684.17	570.22		
	vdd_soc	0.92	1087.90	1000.22		
	vdd_usb_3v3	3.29	1.66	5.46		
	vddq_ddr	0.50	117.76	58.50		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.2.3 Memcpy

This Memcpy is a suite to evaluate the performance of a memory copy in various ways. It runs with the CPU frequency set to 1.8 GHz in performance mode, a DDR data rate of 6400 MT/s, and the CM7 core powered down.

Table 15 shows the system clock and the usage for this use case.

Table 15. Summary for Memcpy

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Running Memcpy	Clock gating	Power Down	Active

To measure the power consumption of the memcpy, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`. See Section 5.
5. Run `memcpy_loop.sh`:

```
while true; do
    buff_size=`cat /proc/meminfo | grep CmaFree | awk '{print$2}'`
    perf bench -f simple mem memcpy -l 20000 -s ${buff_size}KB
done
```

6. Measure the power and record the results.
   Table 16 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 16. Measurement results for i.MX 95 19x19 EVK Memcpy loop (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	37.98	67.60	361.41	31
	lpd5_vdd2	1.04	279.42	290.65		
	lpd5_vddq	0.50	6.34	3.15		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.62	8.64	2462.33	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.70	1.26		
	nvcc_sdio2	3.29	0.21	0.70		
	nvcc_wakeup	1.80	0.85	1.53		
	vdd2_ddr	1.04	5.38	5.60		
	vdd_ana_0v8	0.84	101.95	85.56		
	vdd_ana_1v8	1.79	91.92	164.72		
	vdd_arm	0.92	753.32	690.51		
	vdd_ddr	0.83	638.76	532.45		
	vdd_soc	0.92	1022.41	940.01		
	vdd_usb_3v3	3.29	1.66	5.46		
	vddq_ddr	0.50	51.59	25.63		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.3 Audio/video playback use cases

The following audio use case scenarios have been tested:

- Audio playback (gplay)

- Video playback local 1080p (gplay)
- Video playback local 4K (gplay)
- Video playback streaming 1080p (gplay)
- Video playback streaming 4K (gplay)

4.3.1 Audio playback (gplay)

The Audio Playback (gplay) use case measures the power consumption during MP3 audio playback. The system is configured with the CPU frequency governor set to 1.8 GHz, DDR data rate at 6400 MT/s, and the CA55 handling the audio decoding, I2S, and codec functions. The MP3 file used has a bit rate of 128 kbit/s and a sample rate of 44 kHz, with the CM7 powered down.

Table 17 shows the system clock and the usage for this use case.

Table 17. Summary for Audio playback (gplay)

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Playing Audio	Clock gating	Power Down	Active

To measure the power consumption of the audio playback, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`. See Section 5.
5. Run `gplay_audio.sh`:
```
systemctl --user --now enable pipewire wireplumber pipewire-pulse
gplay-1.0 Mpeg1L3_44kHz_128kbps_s_Ed_Rush_Sabotage_mplayer.mp3
```
Note: *Prepare your own MP3 file. To obtain similar results to those in this document, ensure that the audio bit rate is close to 128 kbit/s.*
6. Measure the power and record the results.
Table 18 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 18. Measurement results for i.MX 95 19x19 EVK Audio playback (gplay) (average value)

<table>
<tr><th>Group</th><th>Rail Label</th><th>Average Voltage (V)</th><th>Average Current (mA)</th><th>Average Power (mW)</th><th>Sum of average powers (mW)</th><th>Zone 0 die temperature (°C)</th></tr>
<tr><td rowspan="3">GROUP_ DRAM</td><td>lpd5_vdd1</td><td>1.79</td><td>3.28</td><td>5.89</td><td rowspan="3">53.03</td><td rowspan="7">28</td></tr>
<tr><td>lpd5_vdd2</td><td>1.04</td><td>44.96</td><td>46.90</td></tr>
<tr><td>lpd5_vddq</td><td>0.50</td><td>0.51</td><td>0.25</td></tr>
<tr><td rowspan="4">GROUP_SOC_ FULL</td><td>nvcc_3v3</td><td>3.29</td><td>3.00</td><td>9.89</td><td rowspan="4">1384.48</td></tr>
<tr><td>nvcc_bbsm_ 1v8</td><td>1.78</td><td>0.15</td><td>0.26</td></tr>
<tr><td>nvcc_enet_ccm</td><td>1.80</td><td>0.70</td><td>1.26</td></tr>
<tr><td>nvcc_sdio2</td><td>3.29</td><td>0.22</td><td>0.71</td></tr>
</table>

Table 18. Measurement results for i.MX 95 19x19 EVK Audio playback (gplay) (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	nvcc_wakeup	1.80	0.88	1.58		
	vdd2_ddr	1.04	0.20	0.21		
	vdd_ana_0v8	0.84	99.72	83.27		
	vdd_ana_1v8	1.79	92.12	165.08		
	vdd_arm	0.92	221.57	203.33		
	vdd_ddr	0.83	186.97	155.80		
	vdd_soc	0.92	815.50	749.82		
	vdd_usb_3v3	3.29	1.66	5.47		
	vddq_ddr	0.50	15.72	7.81		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.3.2 Video playback local (1080P) (gplay)

The Video Playback Local (1080P) (gplay) use case measures power consumption during the playback of a 1080P video, with the i.MX 95 EVK board connected to an HDMI display through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).

The system is configured with the CPU frequency governor set to default, DDR data rate at 6400 MT/s, and the CM7 powered down. The video file used is an MP4 compressed with H.264/265 at 1080P resolution and 60 fps, with audio encoded in AAC at 44.1 kHz, 2-channel.

Table 19 shows the system clock and the usage for this use case.

Table 19. Summary for Video playback local (1080P) (gplay)

	CA55	CM33	CM7	DDR	VPU
Frequency	1800 MHz	333 MHz	0	6400 MT/s	666 MHz
Usage	Playing Video	Clock gating	Power Down	Active	Decoding

To measure the power consumption of the video playback local (1080p) (gplay), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
4. Boot the Linux image with `imx95-19x19-evk-adv7535.dtb`.
5. To put the system into the Idle mode, run `setup_video.sh`. See Section 5.
6. Run `gplay_videoplayback.sh`:

```
gplay-1.0 ./1080P60.mp4
```

Note: *Prepare your own MP4 file. To obtain similar results, ensure that this file has 1080p with a frame rate of 60, bit rate of about 10 Mbit/s, and is encoded in the H.264 format.*

7. Measure the power and record the results.

Table 20 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 20. Measurement results for i.MX 95 19x19 EVK gplay videoplayback 1080P (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	21.32	38.06	182.69	30
	lpd5_vdd2	1.04	136.94	142.61		
	lpd5_vddq	0.50	4.06	2.02		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.96	9.75	1888.64	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.76	1.36		
	nvcc_sdio2	3.29	0.24	0.79		
	nvcc_wakeup	1.80	2.94	5.28		
	vdd2_ddr	1.04	1.28	1.33		
	vdd_ana_0v8	0.84	122.63	102.79		
	vdd_ana_1v8	1.79	96.27	172.49		
	vdd_arm	0.82	182.14	148.88		
	vdd_ddr	0.83	536.97	447.53		
	vdd_soc	0.92	1056.46	971.31		
	vdd_usb_3v3	3.29	1.67	5.50		
	vddq_ddr	0.50	43.02	21.37		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after Table 3 for more information.*

4.3.3 Video playback local (4K) (gplay)

The Video Playback Local (4K) (gplay) use case measures the power consumption during the playback of a 4K video, with the i.MX 95 EVK board connected to an HDMI display through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).

The system is configured with the CPU frequency governor set to default, DDR data rate at 6400 MT/s, and the CM7 powered down. The video file used is an MP4 compressed with H.264/265 at 4K resolution and 60 fps, with audio encoded in AAC at 44.1 kHz, 2-channel.

Table 21 shows the system clock and the usage for this use case.

Table 21. Summary for Video playback local (4K) (gplay)

	CA55	CM33	CM7	DDR	VPU
Frequency	1800 MHz	333 MHz	0	6400 MT/s	666 MHz
Usage	Playing Video	Clock gating	Power Down	Active	Decoding

To measure the power consumption of video playback local (4K) (gplay), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
   Note: *The card only supports a 4K display and up to 30 fps.*
4. Boot the Linux image with `imx95-19x19-evk-adv7535.dtb`.
5. To put the system into the Idle mode, run `setup_video.sh`. See Section 5.
6. Run `gplay_videoplayback.sh`:
```
gplay-1.0 ./4K60.mp4
```
   Note: *Prepare your own MP4 file. To obtain similar results, ensure that this file is in a 4K resolution with a frame rate of 60, bit rate of about 15 Mbit/s, and encoded in the H.264 format.*
7. Measure the power and record the results.
   Table 22 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 22. Measurement results for i.MX 95 19x19 EVK Video playback local (4K) (gplay) (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	29.56	52.70	269.15	30
	lpd5_vdd2	1.04	204.44	212.81		
	lpd5_vddq	0.50	7.32	3.64		
GROUP_SOC_FULL	nvcc_3v3	3.29	3.00	9.89	2059.34	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.74	1.33		
	nvcc_sdio2	3.29	0.21	0.69		
	nvcc_wakeup	1.80	0.87	1.57		
	vdd2_ddr	1.04	2.54	2.64		
	vdd_ana_0v8	0.84	122.91	103.05		
	vdd_ana_1v8	1.79	96.31	172.57		
	vdd_arm	0.82	188.79	154.56		
	vdd_ddr	0.83	575.85	479.88		
	vdd_soc	0.92	1199.78	1103.27		
	vdd_usb_3v3	3.29	1.65	5.44		
	vddq_ddr	0.50	48.69	24.19		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.3.4 Video playback streaming (1080p) (gplay)

The Video Playback Streaming (1080P) use case measures the power consumption during the streaming of a 1080P video over a network, with the i.MX 95 EVK board connected to an HDMI display through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).

The system is configured with the CPU frequency governor set to default, DDR data rate at 6400 MT/s, and the CM7 powered down. The video is streamed from a server PC to the i.MX 95 EVK board through the local network. The video is an MP4 file encoded in H.264 at 1080P resolution and 60 fps. The audio encoding is AACL with a 44.1 kHz sample rate in a 2-channel configuration.

Table 23 shows the system clock and the usage for this use case.

Table 23. Summary for Video playback streaming (1080p) (gplay)

	CA55	CM33	CM7	DDR	VPU	NET
Frequency	1800 MHz	333 MHz	0	6400 MT/s	666 MHz	666 MHz
Usage	Playing Video	Clock gating	Power Down	Active	Decoding	Network

Note: *NET refers to* `enet` *in* `clk_summary`*.*

To measure the power consumption of video playback streaming (1080P) (gplay), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
   ```
   uuu -b emmc flash95-a55.bin
   ```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect your PC and the board to the same local network.
4. On the server PC, perform the following steps:
   a. For Windows, download `Node.js` from https://nodejs.org/en and install it.
   b. To install `http-server`, use the following command:
   ```
   npm install http-server -g
   ```
5. Enter the target folder that contains the target video in the terminal.
6. Use `http-server -c-1`. Then you obtain the `<ip_server>` along with the port.
7. On the board, perform the following steps:
   a. Boot the Linux image with `imx95-19x19-evk-adv7535.dtb`.
   b. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
   c. Run `setup_video_stream.sh` on the board. See Section 5.
   d. Run the following command on the board:
   ```
   gplay-1.0 http://<ip_server:port>/1080P60.mp4
   ```
   Note: *Prepare your own MP4 file. To obtain similar results, ensure that this file has a resolution of 1080p with 60 fps, a bit rate of roughly 10 Mbit/s, and it is encoded in the H.264 format.*
8. Measure the power for the board and record the results.
   Table 24 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 24. Measurement results for i.MX 95 19x19 EVK Video playback streaming (1080p) (gplay) (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	21.24	37.92	183.74	29
	lpd5_vdd2	1.04	138.01	143.74		
	lpd5_vddq	0.50	4.19	2.08		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.97	9.77	1929.38	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	7.48	13.43		
	nvcc_sdio2	3.29	0.23	0.74		
	nvcc_wakeup	1.80	0.88	1.58		
	vdd2_ddr	1.04	1.28	1.33		
	vdd_ana_0v8	0.84	142.64	119.52		
	vdd_ana_1v8	1.79	100.50	180.05		
	vdd_arm	0.82	188.21	153.99		
	vdd_ddr	0.83	538.00	448.39		
	vdd_soc	0.92	1058.87	973.58		
	vdd_usb_3v3	3.29	1.64	5.42		
	vddq_ddr	0.50	42.91	21.32		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.3.5 Video playback streaming (4K) (gplay)

The Video Playback Streaming (4K) use case measures the power consumption during the streaming of a 4K video over a network, with the i.MX 95 EVK board is connected to an HDMI display through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).

The system is configured with the CPU frequency governor set to default, DDR data rate at 6400 MT/s, and the CM7 powered down. The video is streamed from a server PC to the i.MX 95 EVK board through the local network. The video is an MP4 file encoded in H.264 at 4K resolution and 60 fps. The audio encoding is AACL with a 44.1 kHz sample rate in a 2-channel configuration.

Table 25 shows the system clock and the usage for this use case.

Table 25. Summary for Video playback streaming (4K) (gplay)

	CA55	CM33	CM7	DDR	VPU	NET
Frequency	1800 MHz	333 MHz	0	6400 MT/s	666 MHz	666 MHz
Usage	Playing Video	Clock gating	Power Down	Active	Decoding	Network

Note: *NET refers to `enet` in `clk_summary`.*

To measure the power consumption of video playback streaming (4K) (gplay), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect your PC and board to the same local network.
4. On the server PC, perform the following steps:
   a. For Windows, download `Node.js` from https://nodejs.org/en and install it.
   b. To install `http-server`, use the following command:

   ```
   npm install http-server -g
   ```

   c. Enter the target folder that contains the target video in the terminal.
   d. Use `http-server -c-1`. Then, you obtain the `<ip_server>` along with the port.
5. On the board, perform the following steps:
   a. Boot the Linux image with `imx95-19x19-evk-adv7535.dtb`.
   b. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
   Note: *The card only supports a 4K display and up to 30 fps.*
   c. Run `setup_video_stream.sh` on the board. See Section 5.
   d. Run the following command on the board:

   ```
   gplay-1.0 http://<ip_server:port>/4K60.mp4
   ```

   Note: *Prepare your own MP4 file. To obtain similar results, ensure that this file has a resolution of 1080p, a frame rate of 60 fps, a bit rate of roughly 10 Mbit/s, and is encoded in the H.264 format.*
6. Measure the power for the board and record the results.
   Table 26 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 26. Measurement results for i.MX 95 19x19 EVK Video playback streaming (4K) (gplay) (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	29.16	51.98	262.31	30
	lpd5_vdd2	1.04	198.87	207.00		
	lpd5_vddq	0.50	6.69	3.33		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.96	9.77	2094.23	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	7.46	13.40		
	nvcc_sdio2	3.29	0.23	0.74		
	nvcc_wakeup	1.80	0.90	1.62		
	vdd2_ddr	1.04	2.44	2.54		
	vdd_ana_0v8	0.84	142.51	119.46		
	vdd_ana_1v8	1.79	100.91	180.78		
	vdd_arm	0.82	193.47	158.93		
	vdd_ddr	0.83	573.69	478.11		
	vdd_soc	0.92	1195.60	1099.35		

Table 26. Measurement results for i.MX 95 19x19 EVK Video playback streaming (4K) (gplay) (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vdd_usb_3v3	3.29	1.64	5.40		
	vddq_ddr	0.50	48.02	23.85		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after Table 3 for more information.*

4.4 GPU use case

The following GPU use case scenarios have been tested:

- GLMark
- Vkmark

4.4.1 GLMark

The GLMark use case measures the power consumption when the GLMark is running. The GLMark is an OpenGL benchmark used to evaluate the GPU performance on the system. The system is configured with the CPU frequency governor set to default, DDR data rate at 6400 MT/s, and the CM7 powered down. The GLMark test is run in fullscreen mode, continuously executing the "terrain" sub-case in benchmark.

Table 27 shows the system clock and the usage for this use case.

Table 27. Summary for GLMark

	CA55	CM33	CM7	DDR	GPU
Frequency	1800 MHz	333 MHz	0	6400 MT/s	1000 MHz
Usage	Running GLMark	Clock gating	Power Down	Active	GLMark

To measure the power consumption of the GPU, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect the display to the board through the HDMI interface.
4. Boot the Linux image with `imx95-19x19-evk-adv7535.dtb`.
5. Run `setup_video.sh` to put the system into the idle mode. See Section 5.
6. Run the following command to set the environment to obtain the best performance:
```
cp /etc/environment /etc/environment_backup
echo "WESTON_FORCE_RENDERER=1" >> /etc/environment
systemctl restart weston.service
```
7. Run the following command on the board:
```
glmark2-es2-wayland -b terrain --fullscreen --run-forever
```
8. Measure the power and record the result.

9. Run the following command to reset the environment to avoid affecting other cases:

```
mv /etc/environment_backup /etc/environment
systemctl restart weston.service
```

Table 28 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 28. Measurement results for i.MX 95 19x19 EVK GLMark (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	29.38	52.38	283.00	31
	lpd5_vdd2	1.04	217.05	225.87		
	lpd5_vddq	0.50	9.55	4.75		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.63	8.66	2625.42	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.77	1.39		
	nvcc_sdio2	3.29	0.21	0.70		
	nvcc_wakeup	1.80	0.87	1.56		
	vdd2_ddr	1.04	2.41	2.51		
	vdd_ana_0v8	0.84	123.67	103.69		
	vdd_ana_1v8	1.79	96.62	173.11		
	vdd_arm	0.83	189.83	161.80		
	vdd_ddr	0.83	582.29	485.31		
	vdd_soc	0.92	1802.18	1656.57		
	vdd_usb_3v3	3.29	1.65	5.45		
	vddq_ddr	0.50	49.12	24.40		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.4.2 Vkmark

The Vkmark use case measures the power consumption when the Vkmark is running. The Vkmark is an Vulkan benchmark used to evaluate the GPU performance on the system. The system is configured with the CPU frequency governor set to default, DDR data rate at 6400 MT/s, and the CM7 powered down. The Vkmark test is run in fullscreen mode, continuously executing the "cube" sub-case in benchmark.

Table 29 shows the system clock and the usage for this use case.

Table 29. Summary for Vkmark

	CA55	CM33	CM7	DDR	GPU
Frequency	1800 MHz	333 MHz	0	6400 MT/s	1000 MHz
Usage	Running Vkmark	Clock gating	Power Down	Active	Vkmark

To measure the power consumption of the GPU, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to eMMC with `uuu`:

   ```
   uuu -b emmc flash95-a55.bin
   ```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect the display to the board through the HDMI interface.
4. Boot the Linux image with `imx95-19x19-evk-adv7535.dtb`.
5. Run `setup_video.sh` to put the system into the idle mode. See Section 5.
6. Run the following command to set the environment to obtain the best performance:

   ```
   cp /etc/environment /etc/environment_backup
   echo "WESTON_FORCE_RENDERER=1" >> /etc/environment
   systemctl restart weston.service
   ```

7. Run the following command to run the case:

   ```
   vkmark -b cube --fullscreen --run-forever
   ```

8. Measure the power and record the result.
9. Run the following command to reset the environment to avoid affecting other cases:

   ```
   mv /etc/environment_backup /etc/environment
   systemctl restart weston.service
   ```

Table 30 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 30. Measurement results for i.MX 95 19x19 EVK VKMark (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	22.17	39.57	190.48	31
	lpd5_vdd2	1.04	142.52	148.40		
	lpd5_vddq	0.50	5.04	2.51		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.62	8.62	1882.65	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.70	1.26		
	nvcc_sdio2	3.29	0.22	0.71		
	nvcc_wakeup	1.80	0.91	1.64		
	vdd2_ddr	1.04	1.44	1.50		
	vdd_ana_0v8	0.84	123.10	103.17		
	vdd_ana_1v8	1.79	96.03	172.05		
	vdd_arm	0.82	189.15	156.37		
	vdd_ddr	0.83	548.80	457.38		
	vdd_soc	0.92	1035.10	951.67		
	vdd_usb_3v3	3.29	1.67	5.49		
	vddq_ddr	0.50	45.35	22.52		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.5 Machine learning use cases

The tested use case scenarios for machine learning are as follows:

- eIQ-benchmark (NPU)
- eIQ-benchmark (CPU)
- eIQ-benchmark (GPU)

4.5.1 eIQ benchmark (NPU)

The eIQ benchmark use case intends to run machine learning on NPU with eIQ `benchmark_tool`. Take model `mobilenet_v1_1.0_224_int8_converted.tflite` as an example. The CA55 runs in performance and the CM7 is powered down.

Table 31 shows the system clock and the usage for this use case.

Table 31. Summary for eIQ benchmark (NPU)

	CA55	CM33	CM7	DDR	NPU
Frequency	1800 MHz	333 MHz	0	6400 MT/s	1000 MHz
Usage	Running machine learning	Clock gating	Power Down	Active	Machine Learning

To measure the power consumption of the eIQ benchmark (NPU), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`, see Section 5.
5. Change the directory to `/usr/bin/tensorflow-lite-2.16.2/examples` in `rootfs` and run it:
```
cd /usr/bin/tensorflow-lite-2.16.2/examples
./ML_95_NPU.sh
```
6. Start the power measurement and record data.
Note: *The power data results are derived from an inference process. During the execution of the script, there is an approximately 60-second initialization process, which is not encompassed in the measurement. Start measurement after the log "INFO: Running benchmark for at least 220000 iterations and at least 1 second but terminate if exceeding 150 seconds."*
Table 32 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 32. Measurement results for i.MX 95 19x19 EVK eIQ NPU (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_ DRAM	lpd5_vdd1	1.79	17.10	30.56	216.28	31
	lpd5_vdd2	1.04	170.22	177.20		
	lpd5_vddq	0.50	17.14	8.53		
GROUP_SOC_ FULL	nvcc_3v3	3.30	2.58	8.51	2792.36	

Table 32. Measurement results for i.MX 95 19x19 EVK eIQ NPU (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.73	1.31		
	nvcc_sdio2	3.29	0.22	0.71		
	nvcc_wakeup	1.80	0.83	1.50		
	vdd2_ddr	1.04	2.11	2.20		
	vdd_ana_0v8	0.84	102.06	85.56		
	vdd_ana_1v8	1.79	91.60	164.15		
	vdd_arm	0.92	319.05	292.64		
	vdd_ddr	0.83	533.10	444.29		
	vdd_soc	0.92	1925.32	1769.88		
	vdd_usb_3v3	3.29	1.66	5.48		
	vddq_ddr	0.50	31.90	15.84		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after Table 3 for more information.*

4.5.2 eIQ benchmark (CPU)

The eIQ benchmark use case intends to run machine learning on CPU with eIQ `benchmark_tool`. Take model `mobilenet_v1_1.0_224_quant.tflite` as an example. The CA55 runs in performance and the CM7 is powered down.

Table 33 shows the system clock and the usage for this use case.

Table 33. Summary for eIQ benchmark (CPU)

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Running machine learning	Clock gating	Power Down	Active

To measure the power consumption of the eIQ benchmark (CPU), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`. See Section 5.
5. Change the directory to `/usr/bin/tensorflow-lite-2.16.2/examples` in `rootfs` and run it:

```
cd /usr/bin/tensorflow-lite-2.16.2/examples/
./ML_95_CPU.sh
```

6. Start the power measurement and record data.
   Note: *The power data results are derived from an inference process. During the execution of the script, there is an approximately 1-second initialization process, which is not encompassed in the measurement. Start measurement after the log "INFO: Running benchmark for at least 220000 iterations and at least 1 second but terminate if exceeding 150 seconds."*
   Table 34 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 34. Measurement results for i.MX 95 19x19 EVK eIQ CPU (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	17.55	31.37	138.04	31
	lpd5_vdd2	1.04	101.49	105.77		
	lpd5_vddq	0.50	1.79	0.89		
GROUP_SOC_FULL	nvcc_3v3	3.30	2.59	8.54	2094.48	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.71	1.28		
	nvcc_sdio2	3.29	0.22	0.73		
	nvcc_wakeup	1.80	0.84	1.51		
	vdd2_ddr	1.04	0.58	0.60		
	vdd_ana_0v8	0.84	100.48	84.23		
	vdd_ana_1v8	1.79	91.64	164.23		
	vdd_arm	0.92	612.48	561.46		
	vdd_ddr	0.83	520.38	433.70		
	vdd_soc	0.92	884.68	813.33		
	vdd_usb_3v3	3.29	1.67	5.50		
	vddq_ddr	0.50	38.49	19.12		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.5.3 eIQ benchmark (GPU)

The eIQ benchmark use case intends to run machine learning on GPU with eIQ `benchmark_tool`. Take model `mobilenet_v1_1.0_224_quant.tflite` as an example. The CA55 runs in performance and the CM7 is powered down.

Table 35 shows the system clock and the usage for this use case.

Table 35. Summary for eIQ benchmark (GPU)

	CA55	CM33	CM7	DDR	GPU
Frequency	1800 MHz	333 MHz	0	6400 MT/s	1000 MHz
Usage	Running machine learning	Clock gating	Power Down	Active	Machine Learning

To measure the power consumption of the eIQ benchmark (GPU), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

   ```
   uuu -b emmc flash95-a55.bin
   ```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`, see Section 5.
5. Change the directory to `/usr/bin/tensorflow-lite-2.16.2/examples` in `rootfs` and run it.

   ```
   cd /usr/bin/tensorflow-lite-2.16.2/examples/
   ./ML_95_GPU.sh
   ```

6. Start the power measurement and record data.
   Note: *The power data results are derived from an inference process. During the execution of the script, there is an approximately 1-second initialization process, which is not encompassed in the measurement. Start measurement after the log "INFO: Running benchmark for at least 220000 iterations and at least 1 second but terminate if exceeding 150 seconds."*
   Table 36 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 36. Measurement results for i.MX 95 19x19 EVK eIQ GPU (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	24.91	44.46	223.36	30
	lpd5_vdd2	1.04	169.15	176.15		
	lpd5_vddq	0.50	5.51	2.74		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.58	8.50	2546.60	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.64	1.15		
	nvcc_sdio2	3.29	0.22	0.71		
	nvcc_wakeup	1.80	0.85	1.53		
	vdd2_ddr	1.04	1.50	1.56		
	vdd_ana_0v8	0.84	101.17	84.84		
	vdd_ana_1v8	1.79	91.33	163.65		
	vdd_arm	0.92	252.69	231.81		
	vdd_ddr	0.83	550.96	459.17		
	vdd_soc	0.92	1704.00	1566.34		
	vdd_usb_3v3	3.29	1.66	5.47		
	vddq_ddr	0.50	43.49	21.60		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.5.4 Performance and power efficiency

Comparing the eIQ benchmark results of the test cases above, the NPU delivers nearly 16x the Inference Per Second (IPS) performance of the GPU and 32x that of the CPU. Its power efficiency, measured in IPS per watt, is 15x higher than the GPU and 27x higher than the CPU.

4.6 Storage use cases

The tested use case scenarios for storage are as follows:

- DD_WRITE_eMMC
- DD_READ_eMMC
- DD_WRITE_SD
- DD_READ_SD
- DD_WRITE_SSD
- DD_READ_SSD
- DD_WRITE_USB
- DD_READ_USB

Table 37 shows the system clock and the usage for all the storage use cases.

Table 37. Summary for all the storage use cases

	CA55	CM33	CM7	DDR
Frequency	1800 MHz	333 MHz	0	6400 MT/s
Usage	Running Storage test	Clock gating	Power Down	Active

4.6.1 DD_WRITE_eMMC

The DD_WRITE_eMMC use case measures the power consumption during a write operation to the eMMC storage. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_write_emmc`, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
   ```
   uuu -b emmc flash95-a55.bin
   ```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`.
5. Copy `dd_write_bs4096.sh` (see Section 5) on the eMMC partition (`/root`) and run it.
6. Measure the power and record the result.
   Table 38 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 38. Measurement results for i.MX 95 19x19 EVK DD WRITE eMMC (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	12.93	23.13	107.15	29
	lpd5_vdd2	1.04	80.29	83.70		
	lpd5_vddq	0.50	0.66	0.33		

Table 38. Measurement results for i.MX 95 19x19 EVK DD WRITE eMMC (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_SOC_FULL	nvcc_3v3	3.30	2.59	8.53	1788.86	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.89	1.61		
	nvcc_sdio2	3.29	0.23	0.74		
	nvcc_wakeup	1.80	10.00	17.95		
	vdd2_ddr	1.04	0.41	0.43		
	vdd_ana_0v8	0.84	100.96	84.53		
	vdd_ana_1v8	1.79	92.48	165.73		
	vdd_arm	0.92	368.99	338.38		
	vdd_ddr	0.83	418.18	348.54		
	vdd_soc	0.92	871.20	800.97		
	vdd_usb_3v3	3.29	1.66	5.47		
	vddq_ddr	0.50	31.61	15.70		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.6.2 DD_READ_eMMC

The DD_READ_eMMC use case measures the power consumption during a read operation to the eMMC storage. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_read_emmc`, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`.
5. Copy `dd_read_bs4096.sh` (see Section 5) on the eMMC partition (`/root`) and run it.
6. Measure the power and record the result.
   Table 39 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 39. Measurement results for i.MX 95 19x19 EVK DD_READ_eMMC (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	18.53	33.12	146.01	31
	lpd5_vdd2	1.04	107.82	112.35		

Table 39. Measurement results for i.MX 95 19x19 EVK DD_READ_eMMC (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	lpd5_vddq	0.50	1.08	0.53		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.62	8.65	1960.50	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	1.01	1.82		
	nvcc_sdio2	3.29	0.22	0.72		
	nvcc_wakeup	1.80	13.18	23.68		
	vdd2_ddr	1.04	0.71	0.74		
	vdd_ana_0v8	0.84	101.23	84.87		
	vdd_ana_1v8	1.79	91.78	164.46		
	vdd_arm	0.92	404.28	370.81		
	vdd_ddr	0.83	524.67	437.26		
	vdd_soc	0.92	916.13	842.41		
	vdd_usb_3v3	3.29	1.66	5.46		
	vddq_ddr	0.50	38.97	19.36		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.6.3 DD_WRITE_SD

The DD_WRITE_SD use case measures the power consumption during a write operation to the SD storage. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_write_sd`, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`. See Section 5.
5. Copy `dd_write_bs4096.sh` (see Section 5) to the SD partition (`/run/media/root-mmcblk1p2`) and run it.
6. Measure the power and record the result.
   Table 40 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 40. Measurement results for i.MX 95 19x19 EVK DD_WRITE_SD10 (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	9.32	16.68	84.27	29
	lpd5_vdd2	1.04	64.61	67.37		
	lpd5_vddq	0.50	0.46	0.23		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.68	8.83	1673.73	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.71	1.28		
	nvcc_sdio2	1.79	8.10	14.54		
	nvcc_wakeup	1.80	1.07	1.92		
	vdd2_ddr	1.04	0.30	0.31		
	vdd_ana_0v8	0.84	100.53	84.12		
	vdd_ana_1v8	1.79	91.59	164.11		
	vdd_arm	0.92	272.51	249.96		
	vdd_ddr	0.83	373.24	311.05		
	vdd_soc	0.92	890.96	819.28		
	vdd_usb_3v3	3.29	1.64	5.39		
	vddq_ddr	0.50	25.54	12.69		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.6.4 DD_READ_SD

The DD_READ_SD use case measures the power consumption during a read operation to the SD storage. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_read_sd`, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb`.
4. Run `setup.sh`. See Section 5.
5. Copy `dd_read_bs4096.sh` to the SD partition (`/run/media/root-mmcblk1p2`) and run it.
6. Measure the power and record the result.
   Table 41 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 41. Measurement results for i.MX 95 19x19 EVK DD_READ_SD (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	14.52	25.97	110.63	29
	lpd5_vdd2	1.04	80.95	84.39		
	lpd5_vddq	0.50	0.55	0.27		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.68	8.84	1809.81	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.72	1.29		
	nvcc_sdio2	1.79	9.23	16.56		
	nvcc_wakeup	1.80	1.24	2.24		
	vdd2_ddr	1.04	0.38	0.39		
	vdd_ana_0v8	0.84	101.15	84.78		
	vdd_ana_1v8	1.79	91.86	164.61		
	vdd_arm	0.92	263.55	241.75		
	vdd_ddr	0.83	504.75	420.63		
	vdd_soc	0.92	919.65	845.62		
	vdd_usb_3v3	3.29	1.64	5.39		
	vddq_ddr	0.50	35.15	17.46		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.6.5 DD_WRITE_SSD

The DD_WRITE_SSD use case measures the power consumption during a write operation to the SSD storage through the M.2 interface. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_write_sd(PCIe)`, perform the following steps:

1. Connect the SSD storage device through the M.2 interface.
2. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

3. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
4. Boot up the Linux image with `imx95-19x19-evk.dtb`.
5. Run `setup.sh`. See Section 5.
6. Initialize the SSD as follows:

```
# Use the following command to check for new device
fdisk -l
# Format the device, take new device is /dev/nvme0n1 as an example
mkfs.ext4 /dev/nvme0n1
# create mount directory and mount the device
```

```
mkdir nvme0n1
mount /dev/nvme0n1 /root/nvme0n1
```

7. Copy `dd_write_bs4096.sh` (see Section 5) to the SSD partition and run it.
8. Measure the power and record the result.
   Table 42 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 42. Measurement results for i.MX 95 19x19 EVK DD_WRITE_SSD (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	19.04	34.03	149.16	28
	lpd5_vdd2	1.04	110.00	114.62		
	lpd5_vddq	0.50	1.04	0.52		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.61	8.59	2056.15	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.82	1.47		
	nvcc_sdio2	3.29	0.22	0.74		
	nvcc_wakeup	1.80	1.24	2.23		
	vdd2_ddr	1.04	0.70	0.73		
	vdd_ana_0v8	0.84	131.29	109.99		
	vdd_ana_1v8	1.79	98.15	175.85		
	vdd_arm	0.92	477.12	437.50		
	vdd_ddr	0.83	521.68	434.77		
	vdd_soc	0.92	934.61	859.14		
	vdd_usb_3v3	3.29	1.66	5.48		
	vddq_ddr	0.50	39.06	19.40		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.6.6 DD_READ_SSD

The DD_READ_SSD use case measures the power consumption during a read operation to the SSD storage through the M.2 interface. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_read_sd(PCIe)`, perform the following steps:

1. Connect the SSD storage device through the M.2 interface.
2. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

3. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
4. Boot up the Linux image with `imx95-19x19-evk.dtb`.
5. Run `setup.sh`. See Section 5.

6. Initialize the SSD as follows:

```
# Use the following command to check for new device
fdisk -l
# Format the device, take new device is /dev/sda1 as an example
mkfs.ext4 /dev/nvme0n1
# create mount directory and mount the device
mkdir nvme0n1
mount /dev/nvme0n1 /root/nvme0n1
```

7. Copy `dd_read_bs4096.sh` (see Section 5) to the SSD partition and run it.
8. Measure the power and record the result.
   Table 43 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 43. Measurement results for i.MX 95 19x19 EVK DD_READ_SSD (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	19.87	35.49	161.47	30
	lpd5_vdd2	1.04	120.21	125.23		
	lpd5_vddq	0.50	1.51	0.75		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.60	8.58	2091.17	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.82	1.47		
	nvcc_sdio2	3.29	0.23	0.77		
	nvcc_wakeup	1.80	1.71	3.07		
	vdd2_ddr	1.04	0.99	1.04		
	vdd_ana_0v8	0.84	131.11	109.84		
	vdd_ana_1v8	1.79	98.15	175.86		
	vdd_arm	0.92	493.38	452.45		
	vdd_ddr	0.83	526.61	438.86		
	vdd_soc	0.92	950.90	874.18		
	vdd_usb_3v3	3.29	1.66	5.47		
	vddq_ddr	0.50	38.90	19.32		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.6.7 DD_WRITE_USB

The DD_WRITE_USB use case measures the power consumption during a write operation to the USB storage. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_write_sd(USB3.0)`, perform the following steps:

1. Connect Udisk through the USB 3.0 interface (J7) on i.MX 95 EVK.

2. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

3. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
4. Boot up the Linux image with `imx95-19x19-evk.dtb`.
5. Initialize the USB as follows:

```
# Use the following command to check for new device
fdisk -l
# Format the device, take new device is /dev/sda1 as an example
mkfs.ext4 /dev/sda1
# create mount directory and mount the device
mkdir sda1
mount /dev/sda1 /root/sda1
```

6. Run `setup.sh`. See Section 5.
7. Copy `dd_write_bs4096.sh` (see Section 5) to the Udisk partition (`/root/sda1`) and run it.
8. Measure the power and record the result.
Table 44 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 44. Measurement results for i.MX 95 19x19 EVK DD_WRITE_USB (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	4.68	8.39	61.19	29
	lpd5_vdd2	1.04	50.39	52.55		
	lpd5_vddq	0.50	0.50	0.25		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.65	8.74	1517.27	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.67	1.21		
	nvcc_sdio2	3.29	0.22	0.72		
	nvcc_wakeup	1.80	0.91	1.64		
	vdd2_ddr	1.04	0.27	0.28		
	vdd_ana_0v8	0.83	139.48	116.46		
	vdd_ana_1v8	1.79	91.76	164.44		
	vdd_arm	0.92	189.71	174.04		
	vdd_ddr	0.83	214.14	178.45		
	vdd_soc	0.92	843.76	775.75		
	vdd_usb_3v3	3.28	26.42	86.75		
	vddq_ddr	0.50	17.21	8.55		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.6.8 DD_READ_USB

The DD_READ_USB use case measures the power consumption during a read operation to the USB storage. The system is configured with the CPU frequency governor set to performance, the kernel's read-ahead for a single file set to 512 kB, DDR data rate at 6400 MT/s, and the CM7 powered down.

To measure the power consumption of `dd_read_sd(USB3.0)`, perform the following steps:

1. Connect Udisk through the USB 3.0 interface (J7) on i.MX 95 EVK.
2. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

3. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
4. Boot up the Linux image with `imx95-19x19-evk.dtb`.
5. Initialize the USB as follows:

```
# Use the following command to check for new device
fdisk -l
# Format the device, take new device is /dev/sda1 as an example
mkfs.ext4 /dev/sda1
# create mount directory and mount the device
mkdir sda1
mount /dev/sda1 /root/sda1
```

6. Run `setup.sh`. See Section 5.
7. Copy `dd_write_bs4096.sh` (see Section 5) to the Udisk partition (`/root/sda1`) and run it.
8. Measure the power and record the result.
   Table 45 shows the measurement results when this use case applies to the i.MX 95 processor.

Table 45. Measurement results for i.MX 95 19x19 EVK DD_READ_USB (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	10.67	19.10	95.31	29
	lpd5_vdd2	1.04	72.78	75.87		
	lpd5_vddq	0.50	0.68	0.34		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.64	8.69	1903.62	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.72	1.29		
	nvcc_sdio2	3.29	0.21	0.70		
	nvcc_wakeup	1.80	1.03	1.85		
	vdd2_ddr	1.04	0.38	0.40		
	vdd_ana_0v8	0.84	140.02	117.22		
	vdd_ana_1v8	1.79	91.77	164.45		
	vdd_arm	0.92	324.74	297.87		
	vdd_ddr	0.83	463.19	385.99		
	vdd_soc	0.92	897.82	825.37		

Table 45. Measurement results for i.MX 95 19x19 EVK DD_READ_USB (average value)...*continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vdd_usb_3v3	3.28	26.37	86.58		
	vddq_ddr	0.50	26.06	12.94		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after Table 3 for more information.*

4.7 Low-power mode use cases

The following low-power mode use case scenarios have been tested:

- System Idle with display in OD mode with DDRC auto clock gating
- System Idle without display in OD mode with DDRC auto clock gating
- System in DSM
- BBSM

Note: *The i.MX 95 low power modes include: Idle mode, Suspend mode, and BBSM mode. For more information, see the i.MX 95 Applications Processors Data Sheet for Commercial Products (document IMX95CEC).*

4.7.1 System Idle with display in OD mode with DDRC auto clock gating

The System Idle with Display in OD mode with DDRC auto clock gating use case measures the power consumption while the system is idle with the display in OD (OverDrive) mode, and DDRC (DDR Controller) auto clock gating enabled. The CM7 image is not loaded.

Table 46 shows the system clock and the usage for this use case.

Table 46. Summary for System Idle with display in OD mode with DDRC auto clock gating

	CA55	CM33	CM7	DDR
Frequency	900 MHz	333 MHz	0	6400 MT/s
Usage	Running with idle	Clock gating	Power Down	Active with gating

Note: *Most of the time, the CPU is in idle state and the DDR enters the clocking gating state.*

To measure the power consumption for the system Idle with display in OD mode with DDRC auto clock gating, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
   ```
   uuu -b emmc flash95-a55.bin
   ```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
4. Boot the Linux image with `imx95-19x19-evk-adv7535.dtb`.
5. Run `setup_video.sh`. See Section 5.
6. The default mode is the OD mode.
7. Measure the power and record the results.
   Table 47 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 47. Measurement results for i.MX 95 19x19 EVK System Idle with display in OD mode with DDRC auto clock gating (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_ DRAM	lpd5_vdd1	1.79	15.17	27.13	118.04	29
	lpd5_vdd2	1.04	86.23	89.89		
	lpd5_vddq	0.50	2.06	1.03		
GROUP_SOC_ FULL	nvcc_3v3	3.29	2.61	8.60	1624.21	
	nvcc_bbsm_ 1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.72	1.30		
	nvcc_sdio2	3.29	0.22	0.72		
	nvcc_wakeup	1.80	0.85	1.53		
	vdd2_ddr	1.04	0.46	0.48		
	vdd_ana_0v8	0.84	123.14	103.16		
	vdd_ana_1v8	1.79	97.05	173.90		
	vdd_arm	0.82	71.62	58.53		
	vdd_ddr	0.83	504.12	420.18		
	vdd_soc	0.92	904.75	831.86		
	vdd_usb_3v3	3.29	1.66	5.48		
	vddq_ddr	0.50	36.66	18.21		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.7.2 System Idle without display in OD mode with DDRC auto clock gating

The System Idle without Display in OD mode with DDRC auto clock gating use case measures the power consumption while the system is idle in OD (OverDrive) mode, and DDRC (DDR Controller) auto clock gating enabled. The CM7 image is not loaded.

Table 48 shows the system clock and the usage for this use case.

Table 48. Summary for System Idle without display in OD mode with DDRC auto clock gating

	CA55	CM33	CM7	DDR
Frequency	900 MHz	333 MHz	0	6400 MT/s
Usage	Running with idle	Clock gating	Power Down	Active with gating

Note: *Most of the time, the CPU is in idle state and the DDR enters the clocking gating state.*

To measure the power consumption for the system Idle without display in OD mode with DDRC auto clock gating, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot the Linux image with `imx95-19x19-evk.dtb`.
4. Run `idle_screen_off.sh`. See Section 5.
5. The default mode is the OD mode.
6. Measure the power and record the results.
   Table 49 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 49. Measurement results for i.MX 95 19x19 EVK System Idle without display in OD mode with DDRC auto clock gating (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	2.04	3.66	44.73	28
	lpd5_vdd2	1.04	39.25	40.95		
	lpd5_vddq	0.50	0.26	0.13		
GROUP_SOC_FULL	nvcc_3v3	3.30	2.57	8.48	1185.52	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.69	1.23		
	nvcc_sdio2	3.29	0.23	0.76		
	nvcc_wakeup	1.80	0.86	1.55		
	vdd2_ddr	1.04	0.17	0.17		
	vdd_ana_0v8	0.83	96.75	80.77		
	vdd_ana_1v8	1.79	91.08	163.25		
	vdd_arm	0.82	74.46	60.86		
	vdd_ddr	0.83	143.92	119.95		
	vdd_soc	0.92	800.26	735.77		
	vdd_usb_3v3	3.29	1.68	5.52		
	vddq_ddr	0.50	13.95	6.93		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.7.3 System in DSM

The System in DSM (Deep Sleep Mode) use case measures the power consumption when the system enters a suspend state, with Linux and the CM7 suspend, the CM33 in idle mode, and DDR in retention.

Table 50 shows the usage for this use case.

Table 50. Summary for system in DSM

	CA55	CM33	CM7	DDR
Usage	Suspend	Clock gating	Suspend	Retention

Note: *As CA55 and CM33 are not working, clocks cannot be detected.*

To measure the power consumption of the system in the DSM, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_pms.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55-m7_pms.bin
```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk.dtb` and run `echo mem > /sys/power/state` to put the system into the suspend (deep-sleep) mode.
4. Switch to the CM7 console and input the letters "D" and "S" to enter the suspend mode.
5. By default, the CM33 core should be in idle state after POR. You can also type "idle", and then press **Enter** to make sure its state for those low-power cases.
6. Measure the power and record the result.
   Table 51 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 51. Measurement results for i.MX 95 19x19 EVK DSM (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.80	1.72	3.09	9.40	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED, CM33 CORE IS IN IDLE STATE.
	lpd5_vdd2	1.05	6.01	6.32		
	lpd5_vddq	0.50	-0.02	-0.01		
GROUP_SOC_FULL	nvcc_3v3	3.31	2.17	7.20	25.65	
	nvcc_bbsm_1v8	1.79	0.10	0.18		
	nvcc_enet_ccm	1.80	0.43	0.77		
	nvcc_sdio2	3.29	0.11	0.36		
	nvcc_wakeup	1.80	0.74	1.34		
	vdd2_ddr	1.05	0.16	0.16		
	vdd_ana_0v8	0.80	3.02	2.41		
	vdd_ana_1v8	1.80	1.27	2.28		
	vdd_arm	0.00	-0.64	0.00		
	vdd_ddr	0.80	7.99	6.37		
	vdd_soc	0.65	5.93	3.85		
	vdd_usb_3v3	3.31	0.25	0.84		
	vddq_ddr	0.50	-0.19	-0.09		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.7.4 BBSM

The BBSM mode use case measures the power consumption when the system enters a low-power state, where only the BBSM domain remains powered. In this mode, all clocks and PLLs in CA55 are turn off, CM7 and CM33 are powered down, except for essential functions such as the Secure Real-Time Clock and tamper logic are maintained.

Table 52 shows the usage for this use case.

Table 52. Summary for BBSM

	CA55	CM33	CM7	DDR
Usage	Power Down	Power Down	Power Down	Power Down

Note: *As CA55 and CM33 are not working, clocks cannot be detected.*

To configure and run the use case, perform the following steps:

1. Boot the Linux image with `imx95-19x19-evk.dtb` in the CA55.
2. Press and hold the ON/OFF key (SW3) on the baseboard for 5 seconds.
3. Measure the power and record the results.
   Table 53 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 53. Measurement results for i.MX 95 19x19 EVK BBSM (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	0.00	-0.01	0.00	0.00	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED, CM33 CORE IS IN IDLE STATE..
	lpd5_vdd2	0.00	-0.22	0.00		
	lpd5_vddq	0.00	-0.22	0.00		
GROUP_SOC_FULL	nvcc_3v3	0.00	-0.02	0.00	0.13	
	nvcc_bbsm_1v8	1.79	0.07	0.13		
	nvcc_enet_ccm	0.00	-0.03	0.00		
	nvcc_sdio2	0.00	-0.02	0.00		
	nvcc_wakeup	0.00	-0.03	0.00		
	vdd2_ddr	0.00	-0.01	0.00		
	vdd_ana_0v8	0.00	-0.23	0.00		
	vdd_ana_1v8	0.00	-0.06	0.00		
	vdd_arm	0.00	-0.49	0.00		
	vdd_ddr	0.00	-0.25	0.00		
	vdd_soc	0.00	-0.48	0.00		
	vdd_usb_3v3	0.00	-0.01	0.00		
	vddq_ddr	0.00	-0.31	0.00		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.8 Stress test use cases

The following stress test use case scenarios have been tested:

- 6 x CA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU
- 5 x CA55 CPU inference + 1 x CA55 Dhrystone + NPU + GPU + CM7 Coremark

4.8.1 6 x CA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU

The 6 x CA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU use case measures the power consumption under a high-load scenario, where multiple system components are actively running. This includes six CA55 cores performing Dhrystone, the GPU running GLMark, the VPU handling video encode, the Display controller (DC) performing 2D graphics tasks, the CM7 running Coremark, and the NPU performing inference tasks.

Table 54 shows the system clock and the usage for this use case.

Table 54. Summary for 6 x CA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU

	CA55	CM33	CM7	DDR	GPU	VPU	DC	NPU
Frequency	1800 MHz	333 MHz	800 MHz	6400 MT/s	800 MHz	666 MHz	800 MHz	1000 MHz
Usage	Running stress Test	Clock gating	CoreMark	Active	GLMark	Encode	Display	Machine Learning

To measure the power consumption of the 6 x CA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU, perform the following steps:

1. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
2. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_coremark.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55-m7_coremark.bin
```

3. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
4. Boot up the Linux image with `imx95-19x19-evk.dtb`.
5. Press any key in the CM7 console to start CoreMark.
6. Run `setup_video.sh`.
7. Run `cpufreq-set -g performance` to set the system to the performance mode.
8. Run `max_pwr_dhryst_gpu_vpu_dpu.sh`.

```
./dhrystone_loop.sh &
./gpu_glmark_loop.sh &
./vpu_loop.sh &
./g2d_loop.sh &
cd /usr/bin/tensorflow-lite-2.16.2/examples
./ML_95_NPU.sh
```

9. Measure the power and record the result.
   Table 55 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 55. Measurement results for i.MX 95 19x19 EVK 6 x CA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	34.48	61.38	349.79	37
	lpd5_vdd2	1.04	270.64	281.44		
	lpd5_vddq	0.50	14.00	6.97		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.69	8.87	4295.13	
	nvcc_bbsm_1v8	1.78	0.15	0.27		

Table 55. Measurement results for i.MX 95 19x19 EVK 6 x CA55 Dhrystone + GPU + VPU + DC + CM7 CoreMark + NPU (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	nvcc_enet_ccm	1.80	7.57	13.59		
	nvcc_sdio2	3.29	0.22	0.72		
	nvcc_wakeup	1.80	7.78	13.97		
	vdd2_ddr	1.04	3.34	3.48		
	vdd_ana_0v8	0.84	140.31	117.62		
	vdd_ana_1v8	1.79	101.56	181.92		
	vdd_arm	0.92	1701.82	1558.11		
	vdd_ddr	0.83	607.03	505.95		
	vdd_soc	0.92	2022.54	1859.56		
	vdd_usb_3v3	3.29	1.61	5.30		
	vddq_ddr	0.50	51.88	25.77		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after* *Table 3* *for more information.*

4.8.2 5 x CA55 CPU inference + 1 x CA55 Dhrystone + NPU + GPU + CM7 CoreMark

The 5 x CA55 CPU inference + 1 x CA55 Dhrystone + NPU + GPU + CM7 Coremark use case measures power consumption under a high-load scenario, where multiple system components are actively running. This includes five CA55 cores running the ResNet model for CPU inference, one CA55 core running Dhrystone, the NPU running the ResNet model for NPU inference, the GPU running GLMark, and the CM7 running Coremark.

Table 56 shows the system clock and the usage for this use case.

Table 56. Summary for 6 x CA55 Stream + GPU + VPU + DC + CM7 CoreMark + NPU

	CA55	CM33	CM7	DDR	GPU	DC	NPU
Frequency	1800 MHz	333 MHz	800 MHz	6400 MT/s	800 MHz	800 MHz	1000 MHz
Usage	Running stress Test	Clock gating	CoreMark	Active	GLMark	Display	Machine Learning

To measure the power consumption of the 5 x CA55 CPU inference + 1 x CA55 Dhrystone + NPU + GPU + CM7 CoreMark, perform the following steps:

1. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
2. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_coremark.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55-m7_coremark.bin
```

3. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
4. Boot up the Linux image with `imx95-19x19-evk.dtb`.
5. Check the CM7 console to start CoreMark.
6. Run `setup_video.sh`.
7. Run `cpufreq-set -g performance` to set the system to the performance mode.

8. Run `max_pwr.sh`:

```
taskset -c 0 ./benchmark_model --graph=resnet_v2_50_int8_cpu_95.tflite --
num_runs=220000 &
taskset -c 1 ./benchmark_model --graph=resnet_v2_50_int8_cpu_95.tflite --
num_runs=220000 &
taskset -c 2 ./benchmark_model --graph=resnet_v2_50_int8_cpu_95.tflite --
num_runs=220000 &
taskset -c 3 ./benchmark_model --graph=resnet_v2_50_int8_cpu_95.tflite --
num_runs=220000 &
taskset -c 4 ./benchmark_model --graph=resnet_v2_50_int8_cpu_95.tflite --
num_runs=220000 &
./benchmark_model --graph=resnet_v2_50_int8_npu_95.tflite --
external_delegate_path=/usr/lib/libneutron_delegate.so --num_threads=1 --
min_secs=3000 --use_xnnpack=true &
taskset -c 5 glmark2-es2-wayland -b terrain --fullscreen --run-forever &
while true; do taskset -c 5 ./dhry2; done
```

9. Measure the power and record the result.
Table 57 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 57. Measurement results for i.MX 95 19x19 EVK 5 x A55 CPU inference + 1 x A55 Dhrystone + NPU + GPU + M7 CoreMark (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	47.44	84.22	581.21	35
	lpd5_vdd2	1.04	458.73	476.19		
	lpd5_vddq	0.50	41.81	20.80		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.68	8.83	5145.35	
	nvcc_bbsm_1v8	1.78	0.16	0.28		
	nvcc_enet_ccm	1.80	0.78	1.40		
	nvcc_sdio2	3.29	0.23	0.76		
	nvcc_wakeup	1.80	1.02	1.83		
	vdd2_ddr	1.04	6.50	6.75		
	vdd_ana_0v8	0.84	128.32	107.72		
	vdd_ana_1v8	1.79	96.24	172.43		
	vdd_arm	0.92	1843.99	1687.93		
	vdd_ddr	0.83	706.24	588.56		
	vdd_soc	0.92	2756.42	2533.58		
	vdd_usb_3v3	3.29	1.66	5.48		
	vddq_ddr	0.50	59.97	29.78		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.9 Product use cases

The following product use case scenarios have been tested:

- CM33 + CM7 running on DDR
- Telematics (downlink simulation)
- Smart home (appliance/hub)
- Remote camera (AP1302)
- Surround view (OX03C10)
- Digital Connected Cluster (DCC)

4.9.1 CM33 + CM7 Run CoreMark on DDR

The CM33 + CM7 Run CoreMark on DDR use case measures the power consumption when the CM33 is idle and the CM7 core is running the CoreMark benchmark on DDR, while the CA55 image is not loaded.

Table 58 shows the usage for this use case.

Table 58. Summary for CM33 + CM7 run CoreMark on DDR

	CA55	CM33	CM7	DDR
Usage	Power down	Clock gating	CoreMark	Active

Note: *As CA55 and CM33 are not working, clocks cannot be detected.*

To measure the power consumption of CM33 + CM7 run CoreMark on DDR, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-m7_coremark_ddr.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55.bin flash95-m7_coremark_ddr.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Check the CM7 console to start CoreMark.
4. By default, the CM33 core should be in idle state after POR. You can also type "idle", and then press **Enter** to make sure its state for those low power cases.
5. Measure the power and record the results.
   Table 59 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 59. Measurement results for i.MX 95 19x19 EVK CM33 + CM7 Run CoreMark on DDR (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	1.67	2.99	42.14	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE IS POWERED DOWN, CM33 CORE IS IN IDLE STATE.
	lpd5_vdd2	1.04	37.46	39.08		
	lpd5_vddq	0.50	0.13	0.06		
GROUP_SOC_FULL	nvcc_3v3	3.29	4.02	13.25	921.69	
	nvcc_bbsm_1v8	1.78	0.14	0.26		
	nvcc_enet_ccm	1.80	4.38	7.86		
	nvcc_sdio2	3.29	0.11	0.38		
	nvcc_wakeup	1.80	0.86	1.54		
	vdd2_ddr	1.04	0.17	0.17		

Table 59. Measurement results for i.MX 95 19x19 EVK CM33 + CM7 Run CoreMark on DDR (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vdd_ana_0v8	0.83	60.67	50.65		
	vdd_ana_1v8	1.80	37.94	68.12		
	vdd_arm	0.00	-0.44	0.00		
	vdd_ddr	0.83	119.61	99.68		
	vdd_soc	0.92	679.54	624.86		
	vdd_usb_3v3	3.29	14.69	48.31		
	vddq_ddr	0.50	13.31	6.61		

4.9.2 Telematics (downlink simulation)

The Telematics (Downlink Simulation) use case measures the power consumption during a scenario that simulates various network traffic types, including a 5G download, Wi-Fi hotspot, Ethernet gateway traffic, and CPU load for packet filtering.

Table 60 shows the system clock and the usage for this use case.

Table 60. Summary for Telematics (downlink simulation)

	CA55	CM33	CM7	DDR	NET
Frequency	1800 MHz	333 MHz	800 MHz	6400 MT/s	666 MHz
Usage	Packet filtering	Clock gating	CoreMark	Active	Network

Note: *NET refers to `enet` in `clk_summary`.*

When the use case is running, the state of the system is as follows:

1. 310 Mbit/s receive data to simulate a 5G download (2.5 Gbit/s)
2. 200 Mbit/s send data through Wi-Fi to simulate a hotspot (2.5 Gbit/s)
3. 110 Mbit/s traffic on Ethernet to simulate a gateway
4. 1x CA55 CPU loading for packet filtering

To measure the power consumption of Telematics (downlink simulation), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
   ```
   uuu -b emmc flash95-a55.bin
   ```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Prepare one i.MX 95 EVK board, named "board A", and then:
   a. Connect JODY-W377-00 Wi-Fi module to the board via an M.2 interface.
   b. Connect the Ethernet cable to the board on J11.
   c. Connect a network card (for example, e1000e) to the board through the PCI-E interface (J25) and connect another Ethernet cable to the network card.
4. Use the following commands to enable the Wi-Fi module:
   ```
   modprobe moal mod_para=nxp/wifi_mod_para.conf
   wpa_passphrase abc 12345678 >> /etc/wpa_supplicant_abc.conf
   wpa_supplicant -B -i mlan0 -c /etc/wpa_supplicant_abc.conf -D nl80211
   ```

```
udhcpc -i mlan0
# Getting the ethernet ip address if they disappear after configuring the
# WiFi:
udhcpc -i eth0
udhcpc -i eth2
```

***Note:** `abc` is the Wi-Fi name and `12345678` is the Wi-Fi password.*

5. Prepare another i.MX 95 EVK board, named "board B" and connect the Ethernet cable to J11 under the same local network as board A.
6. Run the `iperf` server on both boards:
   a. On board A, run:
   ```
   iperf -s -p 5202 &
   ```
   b. On board B, run:
   ```
   iperf -s -p 5201 &
   ```
7. Run the telematics simulation on both boards:
   a. On board B:
   ```
   # network card client
   iperf -c [board A network card ip] -B [board B ip] \
   -t 300 -i 10 -p 5202 -b 310M &
   ```
   b. On board A:
   ```
   # ethernet client
   iperf -c [board B ip] -B [board A ethernet ip] \
   -t 300 -i 10 -p 5201 -b 110M &
   # WiFi client
   iperf -c [board B ip] -B [board A WiFi ip] -t 300 -i 10 -p 5201 -b 200M &

   # Run single-core coremark to simulate the cpu loading
   while true; do
   ./coremark_singlecore
   done
   ```
8. Measure the power of board A and record the results.
   Table 61 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 61. Measurement results for i.MX 95 19x19 EVK Telematics (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	11.69	20.91	107.95	30
	lpd5_vdd2	1.04	82.83	86.31		
	lpd5_vddq	0.50	1.45	0.72		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.66	8.76	2139.94	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	9.11	16.36		
	nvcc_sdio2	3.29	0.23	0.75		
	nvcc_wakeup	1.80	0.91	1.63		
	vdd2_ddr	1.04	0.51	0.53		

Table 61. Measurement results for i.MX 95 19x19 EVK Telematics (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vdd_ana_0v8	0.84	137.31	114.90		
	vdd_ana_1v8	1.79	103.34	185.11		
	vdd_arm	0.92	658.59	603.60		
	vdd_ddr	0.83	432.73	360.67		
	vdd_soc	0.92	900.89	828.24		
	vdd_usb_3v3	3.29	1.61	5.30		
	vddq_ddr	0.50	27.84	13.83		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.9.3 Smart Home (Appliance/Hub)

The Smart Home (Appliance/Hub) use case simulates multiple smart home tasks, such as video streaming, camera preview, save images, object detection, appliance control, and safety checks. The system handles Wi-Fi video playback with VPU decoding, camera input for food classification through the NPU, and running CoreMark on CM7 for safety protocol simulation.

For this use case, A smart home case is simulated by multiple applications, including video playback streaming from Wi-Fi by VPU decoding, camera previewing, food classification, and object detection by the NPU, CoreMark on CM7 to simulate the IEC60730 safety library, Weston for GUI, and saving the JPEG from a camera into the eMMC.

Table 62 shows the system clock and the usage for this use case.

Table 62. Summary for Smart Home (Appliance/Hub)

	CA55	CM33	CM7	DDR	DC	NPU	VPU
Frequency	1800 MHz	333 MHz	800 MHz	6400 MT/s	800 MHz	1000 MHz	666 MHz
Usage	Running Smart home	Clock gating	Coremark	Active	Display	Machine Learning	Decoding

To measure the power consumption of Smart Home (Appliance/Hub), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_coremark.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55-m7_coremark.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Connect the HDMI display to the board through the MIPI-to-HDMI converter card (IMX-MIPI-HDMI).
4. Connect the JODY-W377-00 Wi-Fi module to the board via an M.2 interface, using the following commands to enable the Wi-Fi module:

```
modprobe moal mod_para=nxp/wifi_mod_para.conf
wpa_passphrase abc 12345678 >> /etc/wpa_supplicant_abc.conf
wpa_supplicant -B -i mlan0 -c /etc/wpa_supplicant_abc.conf -D nl80211
udhcpc -i mlan0
```

Note: *`abc` is the Wi-Fi name, `12345678` is the Wi-Fi password.*

5. Connect the server PC and the board to the same hotspot. Prepare a video with a 4K resolution, 60 frames per second, and 15-Mbit/s bit rate, named `4K60fps.mp4` on the PC.
6. On the server PC, perform the following steps:
   a. Download the `Node.js` from `https://nodejs.org/en` and install it.
   b. To install the `http-server`, use the following command:

   ```
   npm install http-server -g
   ```

   c. Enter a target folder that contains the target video in the terminal.
7. Use `http-server -c-1`. Then, you obtain `<ip_server>` along with the port.
8. On the board, run the following commands (see Section 5):

```
gplay-1.0 http://<server_ip:port>/4K60fps.mp4 &
cd /usr/bin/tensorflow-lite-2.16.2/examples/
./ML_95_NPU.sh &
./preview.sh
```

9. On the board, switch to the CM7 console and input the letter "A" to run CoreMark (using CoreMark to simulate the power consumption of the safety library).
10. Measure the power and record the result.
    Table 63 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 63. Measurement results for i.MX 95 19x19 EVK Smart Home (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	37.09	66.00	404.90	32
	lpd5_vdd2	1.04	314.63	327.09		
	lpd5_vddq	0.50	23.74	11.81		
GROUP_SOC_FULL	nvcc_3v3	3.29	3.09	10.19	3439.39	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.95	1.70		
	nvcc_sdio2	3.29	0.23	0.77		
	nvcc_wakeup	1.80	3.41	6.13		
	vdd2_ddr	1.04	4.52	4.71		
	vdd_ana_0v8	0.84	151.88	127.37		
	vdd_ana_1v8	1.79	105.24	188.51		
	vdd_arm	0.91	458.45	419.03		
	vdd_ddr	0.83	638.79	532.26		
	vdd_soc	0.92	2302.39	2116.44		
	vdd_usb_3v3	3.29	1.65	5.43		
	vddq_ddr	0.50	53.54	26.58		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.9.4 Remote camera (AP1302)

The Remote Camera (AP1302) use case involves using the AP1302 MIPI camera to record 1080p video, which is then encoded by the VPU in H.264 format and stored in the eMMC. The CM7 is powered down during this process. This simulates a scenario involved in FPGA streaming, where video data is processed and stored.

Table 64 shows the system clock and the usage for this use case.

Table 64. Summary for remote camera (AP1302)

	CA55	CM33	CM7	DDR	VPU	GPU	ISI
Frequency	1800 MHz	333 MHz	0	6400 MT/s	666 MHz	800 MHz	666 MHz
Usage	Running Remote camera	Clock gating	Power down	Active	Encode	Display	Camera

For this use case, AP1302 MIPI camera is used to record video. The recorded video resolution is 1080P, encoded by the VPU in a H.264 format. The average bit rate is 10 Mbit/s. The video is saved to eMMC to simulate FPGA streaming.

To measure the power consumption of the remote camera, perform the following steps:

1. Connect the MIPI-CSI converter card (the sensor is AP1302) to J13 on the i.MX 95 EVK baseboard.
2. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
3. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
4. Download the `ap1302` firmware from ON Semiconductor, and rename it to `ap130x_ar0144_single_fw.bin`.
5. Copy `ap130x_ar0144_single_fw.bin` to the target board under the `/lib/firmware/` path.
6. Reboot the board.
7. Boot the Linux image with `imx95-19x19-evk-ap1302.dtb`.
8. Run `setup_video.sh`.
9. To record and store the video, run `remote_camera.sh`. See Section 5.
10. Measure the power and record the result.
Table 65 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 65. Measurement results for i.MX 95 19x19 EVK Remote Camera (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	16.30	29.14	130.34	30
	lpd5_vdd2	1.04	96.18	100.24		
	lpd5_vddq	0.50	1.92	0.95		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.60	8.55	1796.58	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.75	1.34		
	nvcc_sdio2	3.29	0.22	0.72		
	nvcc_wakeup	1.80	1.24	2.22		

Table 65. Measurement results for i.MX 95 19x19 EVK Remote Camera (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vdd2_ddr	1.04	0.61	0.64		
	vdd_ana_0v8	0.84	104.33	87.44		
	vdd_ana_1v8	1.79	94.12	168.63		
	vdd_arm	0.82	98.75	80.75		
	vdd_ddr	0.83	514.02	428.29		
	vdd_soc	0.92	1081.06	993.92		
	vdd_usb_3v3	3.29	1.67	5.49		
	vddq_ddr	0.50	36.91	18.33		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.9.5 Surround view (OX03C10)

The Surround View (OX03C10) use case involves using a 4-lane CSI MIPI camera (OX03C10) to capture video at a resolution of 1920x1282 per lane. The recorded video is dewarped using the GPU 2D and rendered using the GPU 3D. The CM33 remains in idle mode while the CM7 is powered down during this process.

For this use case, a 4-lane CSI MIPI camera OX03C10 module is used to record the video. The recorded video resolution is 1920 x 1282 for each lane. The video is then dewarped by GPU 2D and rendered by GPU3D.

Table 66 shows the system clock and the usage for this use case.

Table 66. Summary for surround view (OX03C10)

	CA55	CM33	CM7	DDR	GPU	DC	ISP
Frequency	1800 MHz	333 MHz	0	6400 MT/s	800 MHz	800 MHz	666 MHz
Usage	Running Surround view	Clock gating	Power down	Active	Rendering	Display	Camera

To measure the power consumption of the remote camera, perform the following steps:

1. Connect the OX03C10 camera module to connector J13 on the i.MX 95 EVK baseboard.
2. Connect the display to the board through the LVDS adapter to connector J15 on the i.MX 95 EVK baseboard.
3. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with `uuu`:
```
uuu -b emmc flash95-a55.bin
```
4. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
5. Reboot the board, and boot the Linux image with `imx95-19x19-evk-ox03c10-isp-it6263-lvds0.dtb`.
6. Copy the `Buffer_63dc20ab1c.bin` calibration file from the attachment to the `/root` directory.
7. Run `setup_video.sh`. See Section 5.
8. To record and display the video, run `surround_view.sh`. See Section 5.
9. Measure the power and record the result.

Table 67 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 67. Measurement results for i.MX 95 19x19 EVK Surround View (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	42.21	75.02	457.47	32
	lpd5_vdd2	1.04	358.19	372.21		
	lpd5_vddq	0.50	20.58	10.25		
GROUP_SOC_FULL	nvcc_3v3	3.29	3.20	10.54	3155.11	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.87	1.56		
	nvcc_sdio2	3.29	0.23	0.74		
	nvcc_wakeup	1.80	0.91	1.64		
	vdd2_ddr	1.04	4.81	5.00		
	vdd_ana_0v8	0.84	115.97	97.32		
	vdd_ana_1v8	1.79	134.84	241.35		
	vdd_arm	0.89	519.58	469.32		
	vdd_ddr	0.83	648.72	540.56		
	vdd_soc	0.92	1905.64	1751.37		
	vdd_usb_3v3	3.29	1.66	5.47		
	vddq_ddr	0.50	60.36	29.97		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after Table 3 for more information.*

4.9.6 Digital Connected Cluster (DCC)

The Digital Connected Cluster (DCC) use case measures the power consumption for typical Vision inference and DMS use cases on a 2-wheeler motorbike. The camera system captures three fisheye OX03C10 camera sensors, and GPU2D does the dewarp. Two sensors act as the front/rear camera, and GPU3D renders the captured images to the display for preview. Meanwhile, feeding resized captured images into NPU for object classification by MobileNet model, the inference results are displayed together with preview. The other sensor acts as the DMS camera, the image data is pushed directly into the NPU for helmet detection.

The following table shows the system clock and the usage for this use case.

Table 68. Summary for DCC

	CA55	CM33	DDR	ISP	ISI	DC	GPU	NPU
Frequency	1800 MHz	333 MHz	6400 MT/s	800 MHz	666 MHz	800 MHz	800 Mh	1000 MHz
Usage	Running preview	Clock gating	Active	Camera	Camera	Display/ Dewarp	Rendering	ML

To measure the power consumption of the Digital Connected Cluster, perform the following steps:

1. Connect the 1xMX95MBDES10001 module with the 2xMX95MBCAM10001 module to the MIPI CSI MINISAS connector.
2. Connect the display to the board through the MIPI adapter to connector J14 on the i.MX 95 EVK baseboard.
3. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to the eMMC with UUU:

```
uuu -b emmc flash95-a55.bin
```

4. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
5. Make the DTB compatible to OX03C10 and MIPI display:

```
cd /run/media/boot-mmcblk1p1
/usr/bin/fdtoverlay -o imx95-19x19-evk-ox03c10-adv7535.dtb -i imx95-19x19-evk.dtb imx95-19x19-evk-ox03c10.dtbo imx95-19x19-evk-neoisp.dtbo imx95-19x19-evk-adv7535.dtbo
```

   Reboot the board, choose DTB as `imx95-19x19-evk-ox03c10-adv7535.dtb` in U-Boot.
6. Copy the files `Buffer_63dc20ab1c.bin`, `dcc_vision.py`, `mobilenet_v2_ssd_quant_int8_convert_25Q4.tflite`, and `mobilenet_v3_helmet_int8_convert_25Q4.tflite` from the attachment to the `/usr/share/dcc` directory.
7. Run `setup_video.sh`. See Section 5.
8. Run the command:

```
export LIBCAMERA_PIPELINES_MATCH_LIST='nxp/neo,imx8-isi,uvc'
pip3 install posix_ipc
python3 dcc_vision.py
```

9. Measure the power and record the result.
   Table 69 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 69. Measurement results for i.MX 95 19x19 EVK Digital Connected Cluster (DCC)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	34.72	61.81	358.49	32
	lpd5_vdd2	1.04	278.24	289.30		
	lpd5_vddq	0.50	14.85	7.39		
GROUP_SOC_FULL	nvcc_3v3	3.29	3.20	10.53	3080.79	
	nvcc_bbsm_1v8	1.78	0.15	0.27		
	nvcc_enet_ccm	1.80	0.80	1.43		
	nvcc_sdio2	3.29	0.22	0.72		
	nvcc_wakeup	1.80	0.99	1.78		
	vdd2_ddr	1.04	3.47	3.60		
	vdd_ana_0v8	0.84	134.97	113.18		
	vdd_ana_1v8	1.79	99.78	178.75		
	vdd_arm	0.91	835.83	762.39		
	vdd_ddr	0.83	610.73	508.88		
	vdd_soc	0.92	1596.96	1467.99		
	vdd_usb_3v3	3.29	1.49	4.92		

Table 69. Measurement results for i.MX 95 19x19 EVK Digital Connected Cluster (DCC)...continued

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vddq_ddr	0.50	53.04	26.34		

4.10 Linux Suspend use cases

The following low-power run use case scenarios have been tested:

- Linux Suspend + CM7 CoreMark (TCM)
- Linux Suspend + CM7 in WFI
- Linux Suspend + CM7 FlexCAN transaction
- Linux suspend + CM7 Ethernet (NETC)
- Linux Suspend + WoL (Wake-on-LAN)

4.10.1 Linux Suspend + CM7 CoreMark (TCM)

The Linux Suspend + CM7 CoreMark (TCM) use case measures the power consumption when the Linux operating system (running on CA55) is in suspend mode while the CM7 core runs CoreMark in 400 MHz underdrive mode, DDR is in retention, NOCMIX and WAKEUPMIX are powered off, and the CM33 is idle.

Table 70 shows the system clock and the usage for this use case.

Table 70. Summary for Linux Suspend + CM7 CoreMark (TCM)

	CA55	CM33	CM7	DDR
Frequency	0	0	400 MHz	0
Usage	Suspend	Clock gating	CoreMark	Retention

To measure the power consumption of Linux Suspend + CM7 CoreMark (TCM), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_coremark.bin` to the eMMC with `uuu`:

```
uuu -b emmc flash95-a55-m7_coremark.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot the Linux image with `imx95-19x19-evk.dtb` and run `echo mem > /sys/power/state` to put the system into the suspend (deep-sleep) mode.
4. Switch to the CM7 console and press any key to start CoreMark.
5. By default, the CM33 core should be in idle state after POR. You can also type "idle", and then press **Enter** to make sure its state for those low power cases.
6. Measure the power and record the results.
   Table 71 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 71. Measurement results for i.MX 95 19x19 EVK Linux Suspend + CM7 CoreMark (TCM) (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	1.68	3.02	9.45	DIE TEMPERATURE
	lpd5_vdd2	1.04	6.06	6.32		

Table 71. Measurement results for i.MX 95 19x19 EVK Linux Suspend + CM7 CoreMark (TCM) (average value)...*continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	lpd5_vddq	0.50	0.22	0.11		CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED, CM33 CORE IS IN IDLE STATE.
GROUP_SOC_FULL	nvcc_3v3	3.30	2.24	7.37	197.24	
	nvcc_bbsm_1v8	1.78	0.14	0.25		
	nvcc_enet_ccm	1.80	0.33	0.59		
	nvcc_sdio2	1.80	1.31	2.36		
	nvcc_wakeup	1.80	0.76	1.37		
	vdd2_ddr	1.04	0.17	0.17		
	vdd_ana_0v8	0.83	27.86	23.24		
	vdd_ana_1v8	1.80	5.31	9.54		
	vdd_arm	0.00	-0.45	0.00		
	vdd_ddr	0.83	12.20	10.17		
	vdd_soc	0.82	172.57	141.36		
	vdd_usb_3v3	3.30	0.26	0.87		
	vddq_ddr	0.50	-0.10	-0.05		

Note: *The power values of* `vddq_ddr` *and* `lpd5_vddq` *are not the actual ones. See the Note after* *Table 3* *for more information.*

4.10.2 Linux Suspend + CM7 in WFI

The Linux Suspend + CM7 in WFI use case measures the power consumption when the Linux operating system (running on CA55) is in suspend mode while the CM7 core is in the Wait for Interrupt (WFI) mode, DDR is in retention, NOCMIX and WAKEUPMIX are powered off, and the CM33 is idle.

Table 72 shows the system clock and the usage for this use case.

Table 72. Summary for Linux Suspend + CM7 in WFI

	CA55	CM33	CM7	DDR
Frequency	0	0	400 MHz	0
Usage	Suspend	Clock gating	WFI	Retention

To measure the power consumption of the Linux Suspend + CM7 in WFI, perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_pms.bin` to the eMMC using `uuu`:

```
uuu -b emmc flash95-a55-m7_pms.bin
```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot the Linux image with `imx95-19x19-evk.dtb` and run `echo mem > /sys/power/state` to put the system into the suspend (deep-sleep) mode.
4. Switch to the CM7 console and input the letters "B" and "S" to enter the Wait mode.

5. By default, the CM33 core should be in idle state after POR. You can also type "idle", and then press **Enter** to make sure its state for those low power cases.
6. Measure the power and record the results.
   Table 73 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 73. Measurement results for i.MX 95 19x19 EVK Linux Suspend + CM7 in WFI (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	1.71	3.06	9.29	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED, CM33 CORE IS IN IDLE STATE.
	lpd5_vdd2	1.04	5.88	6.14		
	lpd5_vddq	0.50	0.18	0.09		
GROUP_SOC_FULL	nvcc_3v3	3.30	2.25	7.43	178.48	
	nvcc_bbsm_1v8	1.78	0.14	0.25		
	nvcc_enet_ccm	1.80	0.33	0.60		
	nvcc_sdio2	1.80	1.32	2.37		
	nvcc_wakeup	1.80	0.71	1.28		
	vdd2_ddr	1.04	0.17	0.18		
	vdd_ana_0v8	0.83	27.88	23.26		
	vdd_ana_1v8	1.80	5.29	9.50		
	vdd_arm	0.00	-0.44	0.00		
	vdd_ddr	0.83	12.20	10.17		
	vdd_soc	0.82	149.62	122.57		
	vdd_usb_3v3	3.30	0.27	0.88		
	vddq_ddr	0.50	-0.02	-0.01		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.10.3 Linux Suspend + CM7 FlexCAN transaction

The Linux Suspend + CM7 FlexCAN transaction use case measures power consumption when the Linux operating system (running on CA55) is in suspend mode while the CM7 core is sending/receiving CAN frames (FlexCAN `interrupt_transfer` driver example in SDK), DDR is powered on, and the CM33 is idle.

Table 74 shows the system clock and the usage for this use case.

Table 74. Summary for Linux Suspend + CM7 FlexCAN transaction

	CA55	CM33	CM7	DDR
Frequency	0	0	800 MHz	6400 MT/s
Usage	Suspend	Clock gating	FlexCAN	Active

To measure the power consumption of the Linux Suspend + CM7 FlexCAN transaction, perform the following steps:

1. Prepare two i.MX 95 EVK boards, namely boards A and B.

2. Connect the CAN connector of the two boards together.
3. On both boards, perform the following steps:
   a. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_flexcan.bin` to the eMMC using `uuu`:

   ```
   uuu -b emmc flash95-a55-m7_flexcan.bin
   ```

   b. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
   c. Stop the U-Boot on the CA55 console.
   d. Run:

   ```
   fatload mmc ${mmcdev}:${mmcpart} ${fdt_addr_r} ${fdtfile};
   fdt addr ${fdt_addr_r};
   fdt rm /soc/bus@44000000/can@443a0000;
   fdt rm /soc/bus@42000000/i2c@426c0000;
   run mmcargs;
   fatload mmc ${mmcdev}:${mmcpart} ${loadaddr} ${image};
   run boot_os
   ```

4. On board B in the CM7 console, input the letter "B" to start receiving the CAN message from board A.
5. On board A, perform the following steps:
   a. In the CM7 console, input the letter "A" to start transmitting and receiving the CAN message. Both CM7 consoles start to print the log.
   b. To suspend the CA55 cores, run the following command:

   ```
   echo mem > /sys/power/state
   ```

   c. By default, the CM33 core should be in idle state after POR. You can also type "idle", and then press **Enter** to make sure its state for those low power cases.
   d. Measure the power numbers from board A only.
6. On board B in the CM7 console, input the letter "B" to start receiving the CAN message from board A.
7. Measure the power and record the results.
   Table 75 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 75. Measurement results for i.MX 95 19x19 EVK Linux Suspend + CM7 FlexCAN transaction (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	1.70	3.04	42.37	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED, CM33 CORE IS IN IDLE STATE.
	lpd5_vdd2	1.04	37.63	39.25		
	lpd5_vddq	0.50	0.13	0.07		
GROUP_SOC_FULL	nvcc_3v3	3.30	2.34	7.72	659.71	
	nvcc_bbsm_1v8	1.78	0.14	0.26		
	nvcc_enet_ccm	1.80	0.37	0.66		
	nvcc_sdio2	3.29	0.23	0.77		
	nvcc_wakeup	1.80	0.68	1.22		
	vdd2_ddr	1.04	0.17	0.18		
	vdd_ana_0v8	0.84	41.31	34.50		

Table 75. Measurement results for i.MX 95 19x19 EVK Linux Suspend + CM7 FlexCAN transaction (average value)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	vdd_ana_1v8	1.80	17.01	30.55		
	vdd_arm	0.82	4.11	3.36		
	vdd_ddr	0.83	118.50	98.75		
	vdd_soc	0.92	515.92	474.21		
	vdd_usb_3v3	3.30	0.26	0.84		
	vddq_ddr	0.50	13.47	6.69		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.10.4 Linux suspend + CM7 Ethernet (NETC)

The Linux Suspend + CM7 Ethernet (NETC) use case measures power consumption when the Linux operating system (running on CA55) is in suspend mode while the CM7 core runs an Ethernet (NETC) DHCP client (`netc_share` demo example from SDK), DDR is powered on, and the CM33 is idle.

Table 76 shows the system clock and the usage for this use case.

Table 76. Summary for Linux suspend + CM7 Ethernet (NETC)

	CA55	CM33	CM7	DDR
Frequency	0	0	800 MHz	6400 MT/s
Usage	Suspend	Clock gating	NETC	Active

To measure the power consumption of Linux suspend + CM7 Ethernet (NETC), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55-m7_netc.bin` to eMMC with `uuu`:
   ```
   uuu -b emmc flash95-a55.bin flash95-a55-m7_netc.bin
   ```
2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.
3. Boot up the Linux image with `imx95-19x19-evk-netc-rpmsg.dtb`.
4. After enter Linux, run `$ echo 1 > /sys/bus/pci/devices/0002\:00\:10.0/sriov_numvfs` to enable a single VF on the Linux side, make sure `eth0` has obtained the IP address, and then run `echo mem > /sys/power/state` to suspend.
5. NETC DHCP client automatically runs on the CM7 console. The DHCP client IP request log is then displayed.
6. By default, the CM33 core should be in idle state after POR. You can also type "idle", and then press **Enter** to make sure its state for those low power cases.
7. Measure the the power and record the results.
   Table 77 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 77. Measurement results for i.MX 95 19x19 EVK Linux suspend + CM7 Ethernet (NETC) (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	1.65	2.97	42.27	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED, CM33 CORE IS IN IDLE STATE.
	lpd5_vdd2	1.04	37.55	39.18		
	lpd5_vddq	0.50	0.24	0.12		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.81	9.25	956.17	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	8.09	14.53		
	nvcc_sdio2	3.29	0.20	0.66		
	nvcc_wakeup	1.80	0.77	1.38		
	vdd2_ddr	1.04	0.16	0.17		
	vdd_ana_0v8	0.83	106.91	89.20		
	vdd_ana_1v8	1.79	58.10	104.24		
	vdd_arm	0.82	3.63	2.96		
	vdd_ddr	0.83	121.97	101.63		
	vdd_soc	0.92	679.02	624.32		
	vdd_usb_3v3	3.30	0.26	0.85		
	vddq_ddr	0.50	13.52	6.71		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after Table 3 for more information.*

4.10.5 Linux Suspend + WoL (Wake-on-LAN)

The Linux Suspend + WakeupOnLine use case measures the power consumption under the following configuration: The Linux operating system based on CA55 enters suspend mode with WoL (Wake-on-LAN) activated, DDR memory remains powered on, the CM7 image is not loaded, and the CM33 core is idle.

Table 78 shows the system clock and the usage for this use case.

Table 78. Summary for Linux Suspend + WoL (Wake-on-LAN)

	CA55	CM33	CM7	DDR
Frequency	0	0	800 MHz	6400 MT/s
Usage	Suspend	Clock gating	Power Down	Active

To measure the power consumption of Linux Suspend + WoL (Wake-on-LAN), perform the following steps:

1. Change the boot mode `SW7[1:4] = 1001`, and program `flash95-a55.bin` to eMMC with `uuu`:

   ```
   uuu -b emmc flash95-a55.bin
   ```

2. Turn off the board on SW4, change the boot mode `SW7[1:4] = 1010`, and then turn on the board on SW4.

3. Switch to the CA55 console and enable WoL on eth0 and suspend by typing:

```
ethtool -s eth0 wol g
echo mem > /sys/power/state
```

4. By default, the CM33 core should be in idle state after POR. You can also type "idle", and then press **Enter** to make sure its state for those low power cases.
5. Measure the power and record the results.
[Table 79](#) shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 79. Measurement results for i.MX 95 19x19 EVK Linux Suspend + WoL (Wake-on-LAN) (average value)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.80	1.67	3.00	9.37	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED.
	lpd5_vdd2	1.04	6.00	6.26		
	lpd5_vddq	0.50	0.22	0.11		
GROUP_SOC_FULL	nvcc_3v3	3.30	2.17	7.15	342.72	
	nvcc_bbsm_1v8	1.78	0.14	0.26		
	nvcc_enet_ccm	1.80	7.36	13.23		
	nvcc_sdio2	3.29	0.11	0.37		
	nvcc_wakeup	1.80	0.81	1.46		
	vdd2_ddr	1.04	0.16	0.17		
	vdd_ana_0v8	0.83	74.97	62.51		
	vdd_ana_1v8	1.80	42.10	75.58		
	vdd_arm	0.00	-0.44	0.00		
	vdd_ddr	0.83	14.85	12.38		
	vdd_soc	0.82	206.08	168.82		
	vdd_usb_3v3	3.30	0.25	0.83		
	vddq_ddr	0.50	-0.06	-0.03		

Note: *The power values of `vddq_ddr` and `lpd5_vddq` are not the actual ones. See the Note after [Table 3](#) for more information.*

6. (Optional) Use some WoL APPs to wake up the i.MX 95 EVK.

4.11 ISP use cases

The following ISP use case scenarios have been tested:

- ISP Single-Camera (4K)
- ISP Dual-Camera (4K)

4.11.1 ISP Single-Camera (4K)

The ISP 4K single-camera use case measures the power consumption during the single 4K camera (OS08A20 8MP@60fps sensor) preview on the display by using the ISP to do noise reduction, lens shading, debayer, and output YUV images.

The following table shows the system clock and the usage for this use case.

Table 80. Summary for ISP 4K single-camera

	CA55	CM33	CM7	DDR	ISP	ISI	DC
Frequency	1800 MHz	333 MHz	0	6400 MT/s	667 MHz	667 MHz	800 MHz
Usage	Running preview	Clock gating	Power down	Active	Camera	Camera	Display

To measure the power consumption of the ISP single-camera (4K), perform the following steps:

1. Use `fdtoverlay` to generate the DTB file:

```
/usr/bin/fdtoverlay -o imx95-19x19-evk-os08a20-isp-it6263-lvds0.dtb -i \
imx95-19x19-evk-it6263-lvds0.dtb \
imx95-19x19-evk-neoisp.dtbo \
imx95-19x19-evk-os08a20.dtbo
```

2. Connect IMX95-OS08A20 to the MIPI CSI MINISAS connector.
3. Connect the display to the board through the LVDS adapter to connector J15 on the i.MX 95 EVK baseboard.
4. Boot up the Linux image with `imx95-19x19-evk-os08a20-isp-it6263-lvds0.dtb`.
5. Run `setup_video.sh`. See Section 5.
6. Run the following command to capture, encode, and save the video stream:

```
export LIBCAMERA_PIPELINES_MATCH_LIST='nxp/neo,imx8-isi,uvc'
CAMERA0=/base/soc/bus@42000000/i2c@42530000/os08a20_mipi@36
DISPLAY_W=1920
DISPLAY_H=1080
gst-launch-1.0 \
    libcamerasrc camera-name="${CAMERA0}" ! \
    video/x-raw, format=YUY2 ! \
    queue ! \
    waylandsink window-width=${DISPLAY_W} window-height=${DISPLAY_H}
```

7. Measure the power and record the result for the board.
   Table 81 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 81. Measurement results for i.MX 95 19x19 EVK ISP Single-Camera (4K)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.79	21.79	38.92	187.98	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED.
	lpd5_vdd2	1.04	140.78	146.63		
	lpd5_vddq	0.50	4.90	2.44		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.63	8.66	2031.06	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.79	1.42		

Table 81. Measurement results for i.MX 95 19x19 EVK ISP Single-Camera (4K)*...continued*

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
	nvcc_sdio2	3.29	0.23	0.76		
	nvcc_wakeup	1.80	0.94	1.69		
	vdd2_ddr	1.04	1.34	1.40		
	vdd_ana_0v8	0.84	113.08	94.80		
	vdd_ana_1v8	1.79	132.08	236.45		
	vdd_arm	0.82	135.93	111.10		
	vdd_ddr	0.83	550.07	458.41		
	vdd_soc	0.92	1183.92	1088.19		
	vdd_usb_3v3	3.29	1.66	5.47		
	vddq_ddr	0.50	45.19	22.45		

4.11.2 ISP Dual-Camera (4K)

The ISP 4K dual-camera use case measures the power consumption during the dual 4K cameras (OS08A20 8MP@60fps sensor) preview on the display by using the ISP to do noise reduction, lens shading, debayer, and output YUV images.

The following table shows the system clock and the usage for this use case.

Table 82. Summary for ISP 4K dual-camera

	CA55	CM33	CM7	DDR	ISP	ISI	DC
Frequency	1800 MHz	333 MHz	0	6400 MT/s	666 MHz	666 MHz	800 MHz
Usage	Running preview	Clock gating	Power down	Active	Camera	Camera	Display

To measure the power consumption of the ISP dual-camera (4K), perform the following steps:

1. Use `fdtoverlay` to generate the DTB file:

```
/usr/bin/fdtoverlay -o imx95-19x19-evk-dual-os08a20-isp-it6263-lvds0.dtb -i \
imx95-19x19-evk-it6263-lvds0.dtb \
imx95-19x19-evk-neoisp.dtbo \
imx95-19x19-evk-os08a20.dtbo \
imx95-19x19-evk-os08a20-combo.dtbo
```

2. Connect IMX95-OS08A20 to the MIPI CSI MINISAS connector.
3. Connect the other IMX95-OS08A20 to the MIPI CSI/DSI combo MINISAS connector.
4. Connect the display to the board through the LVDS adapter to connector J15 on the i.MX 95 EVK baseboard.
5. Boot up the Linux image with `imx95-19x19-evk-dual-os08a20-isp-it6263-lvds0.dtb`.
6. Run `setup_video.sh`. See Section 5.
7. Run the following command to capture, encode, and save the video stream:

```
export LIBCAMERA_PIPELINES_MATCH_LIST='nxp/neo,imx8-isi,uvc'
CAMERA0=/base/soc/bus@42000000/i2c@42530000/os08a20_mipi@36
CAMERA1=/base/soc/bus@44000000/i2c@44350000/os08a20_mipi@36
DISPLAY_W=1920
```

```
DISPLAY_H=1080
gst-launch-1.0 -v \
    imxcompositor_g2d name=comp \
            sink_0::xpos=0 sink_0::ypos=0 sink_0::width=${DISPLAY_W}
 sink_0::height=${DISPLAY_H} \
            sink_1::xpos=0 sink_1::ypos=${DISPLAY_H} sink_1::width=
${DISPLAY_W} sink_1::height=${DISPLAY_H} ! \
        waylandsink \
    libcamerasrc camera-name="${CAMERA0}" ! \
        video/x-raw,format=YUY2 ! queue ! comp.sink_0 \
    libcamerasrc camera-name="${CAMERA1}" ! \
        video/x-raw,format=YUY2 ! queue ! comp.sink_1
```

8. Measure the power and record the result for the board.
   Table 83 shows the measurement results when this use case is applied to the i.MX 95 processor.

Table 83. Measurement results for i.MX 95 19x19 EVK ISP Dual-Camera (4K)

Group	Rail Label	Average Voltage (V)	Average Current (mA)	Average Power (mW)	Sum of average powers (mW)	Zone 0 die temperature (°C)
GROUP_DRAM	lpd5_vdd1	1.78	32.79	58.42	321.48	DIE TEMPERATURE CANNOT BE MEASURED AS THE CA55 CORE HAS BEEN SUSPENDED.
	lpd5_vdd2	1.04	248.40	258.41		
	lpd5_vddq	0.50	9.35	4.66		
GROUP_SOC_FULL	nvcc_3v3	3.29	2.60	8.58	2433.56	
	nvcc_bbsm_1v8	1.78	0.15	0.26		
	nvcc_enet_ccm	1.80	0.76	1.36		
	nvcc_sdio2	3.29	0.23	0.74		
	nvcc_wakeup	1.80	0.94	1.68		
	vdd2_ddr	1.04	3.77	3.92		
	vdd_ana_0v8	0.84	124.81	104.69		
	vdd_ana_1v8	1.79	132.05	236.40		
	vdd_arm	0.83	201.30	169.72		
	vdd_ddr	0.83	619.52	516.30		
	vdd_soc	0.92	1474.64	1355.33		
	vdd_usb_3v3	3.29	1.66	5.47		
	vddq_ddr	0.50	58.62	29.12		

5 Important commands

Before running a use case, the `<configuration_script>.sh` script must be run to configure the environment. Details for these scripts are as follows:

- `setup.sh`: The CPU frequency is set to the maximum value of 1.8 GHz to achieve the best performance. Disable the Ethernet, stop the Weston service, and blank the display. Set 512 kB as the maximum amount of data the kernel reads ahead for a single file.

Note: *512 kB is a value under a performance and resource balancing strategy. The value may not be the most suitable for i.MX 95. We provide a certain test condition.*

```
#!/bin/sh
systemctl stop weston.service
cpufreq-set -g performance
    echo 1 > /sys/class/graphics/fb0/blank
partitions=`lsblk |awk '$1 !~/-/{print $1}' |grep 'blk\|sd'`
for partition in $partitions; do
    echo 512 > /sys/block/$partition/queue/read_ahead_kb
done
eth_int=`ifconfig -a | grep 'eth[0-9]'|awk {'print substr($1, 0, 4)'}`
for eth in $eth_int; do
    ifconfig $eth down
done
```

- `setup_video.sh`: Disable the Ethernet and awake the display. Set 512 kB as the maximum amount of data the kernel reads ahead for a single file.

```
#!/bin/bash
partitions=`lsblk |awk '$1 !~/-/{print $1}' |grep 'blk\|sd'`
for partition in $partitions; do
    echo 512 > /sys/block/$partition/queue/read_ahead_kb
done
eth_int=`ifconfig -a | grep 'eth[0-9]'|awk {'print substr($1, 0, 4)'}`
for eth in $eth_int; do
    ifconfig $eth down
done
echo 1 > /sys/class/graphics/fb0/blank
echo 0 > /sys/class/graphics/fb0/blank
```

- `setup_video_stream.sh`: Enable the Ethernet and awake the display. Set 512 kB as the maximum amount of data the kernel reads ahead for a single file.

```
#!/bin/bash
partitions=`lsblk |awk '$1 !~/-/{print $1}' |grep 'blk\|sd'`
for partition in $partitions; do
    echo 512 > /sys/block/$partition/queue/read_ahead_kb
done
eth_int=`ifconfig -a | grep 'eth[0-9]'|awk {'print substr($1, 0, 4)'}`
for eth in $eth_int;do
    ifconfig $eth up
done
echo 1 > /sys/class/graphics/fb0/blank
echo 0 > /sys/class/graphics/fb0/blank
```

- `idle_screen_off.sh`: Disable the Ethernet, stop the Weston service, and blank the display. Set 512 kB as the maximum amount of data the kernel reads ahead for a single file.

```
#!/bin/sh
systemctl stop weston.service
echo 1 > /sys/class/graphics/fb0/blank
partitions=`lsblk |awk '$1 !~/-/{print $1}' |grep 'blk\|sd'`
for partition in $partitions; do
    echo 512 > /sys/block/$partition/queue/read_ahead_kb
done
eth_int=`ifconfig -a | grep 'eth[0-9]'|awk {'print substr($1, 0, 4)'}`
for eth in $eth_int; do
    ifconfig $eth down
done
```

- `dd_read_bs4096.sh`: This script is used to run the `dd read` command on the memory device.

```
#!/bin/bash
# Since we're dealing with dd, abort if any errors occur
set -e
TEST_FILE=${1:-dd_obs_testfile}
```

```
if [ $EUID -ne 0 ]; then
echo "NOTE: Kernel cache will not be cleared between tests without sudo. \
This will likely cause inaccurate results." 1>&2
fi
count=$COUNT conv=fsync > /dev/null 2>&1
# Header
PRINTF_FORMAT="%8s : %s\n"
printf "$PRINTF_FORMAT" 'block size' 'transfer rate'
while true
BLOCK_SIZE=4096
do
# Clear kernel cache to ensure more accurate test
[ $EUID -eq 0 ] && [ -e /proc/sys/vm/drop_caches ] && \
echo 3 > /proc/sys/vm/drop_caches
# Read test file out to /dev/null with specified block size
DD_RESULT=$(dd if=$TEST_FILE of=/dev/null bs=$BLOCK_SIZE 2>&1 1>/dev/null)
# Extract transfer rate
TRANSFER_RATE=$(echo $DD_RESULT | \
grep --only-matching-E '[0-9.]+ ([MGk]?B|bytes)s(ec)?')
printf "$PRINTF_FORMAT" "$BLOCK_SIZE" "$TRANSFER_RATE"
done
```

- `dd_write_bs4096.sh`: This script is used to run the `dd write` command on the memory device.

```
#!/bin/bash
# Since we're dealing with dd, abort if any errors occur
set -e
TEST_FILE=${1:-dd_obs_testfile}
TEST_FILE_EXISTS=0
if [ -e "$TEST_FILE" ]; then TEST_FILE_EXISTS=1; fi
TEST_FILE_SIZE=1024000000
if [ $EUID -ne 0 ]; then
echo "NOTE: Kernel cache will not be cleared between tests without sudo. \
This will likely cause inaccurate results." 1>&2
# Header
PRINTF_FORMAT="%8s: %s\n"
printf "$PRINTF_FORMAT" 'block size' 'transfer rate'
while true
BLOCK_SIZE=4096
do
# Calculate number of segments required to copy
COUNT=$(($TEST_FILE_SIZE / $BLOCK_SIZE))
if [ $COUNT -le 0 ]; then
echo "Block size of $BLOCK_SIZE estimated to require $COUNT blocks, \
aborting further tests."
break
fi
# Clear kernel cache to ensure more accurate test
[ $EUID -eq 0 ] && [ -e /proc/sys/vm/drop_caches ] && \
echo 3 > /proc/sys/vm/drop_caches
# Create a test file with the specified block size
DD_RESULT=$(dd if=/dev/zero of=$TEST_FILE bs=$BLOCK_SIZE \
count=$COUNT conv=fsync 2>&1 1>/dev/null)
# Extract the transfer rate from dd's STDERR output
TRANSFER_RATE=$(echo $DD_RESULT | \
grep --only-matching -E '[0-9.]+ ([MGk]?B|bytes)/s(ec)?')
# Output the result
printf "$PRINTF_FORMAT" "$BLOCK_SIZE" "$TRANSFER_RATE"
done
```

- `dhrystone_loop.sh`: The script starts the Dhrystone example:
  Note: *To run `dhrystone_loop.sh`, the `dhry2` file is needed. Obtain the file from [AN14449SW.zip](AN14449SW.zip).*

```
#!/bin/sh

while true;
do
```

```
    sudo taskset -c 0 ./dhry2 &
    sudo taskset -c 1 ./dhry2 &
    sudo taskset -c 2 ./dhry2 &
    sudo taskset -c 3 ./dhry2 &
    sudo taskset -c 4 ./dhry2 &
    sudo taskset -c 5 ./dhry2
done
```

- `vpu_loop.sh`: The script starts one of the stress test examples:
  Note: *To run `vpu_loop.sh`, the `test.yuv` file is needed. Obtain the file from [AN14449SW.zip](AN14449SW.zip).*

```
#!/bin/sh

while true; do
 /unit_tests/V4L2_VPU/mxc_v4l2_vpu_test.out \
  ifile --key 0 --name test.yuv  --fmt nv12 --size 1920 1080 \
  --framenum 10 \
  encoder --key 1 --source 0 --size 1920 1080 --framerate 30 \
  --bitrate 4194304 --lowlatency 0 \
  ofile --key 2 --source 1 --name test.h264
done
```

- `g2d_loop.sh`: The script starts one of the stress test examples:

```
#!/bin/sh

while true; do
 /opt/g2d_samples/g2d_basic_test
done
```

- `gpu_glmark_loop.sh`: The script starts one of the stress test examples:

```
#!/bin/sh
glmark2-es2-wayland -b terrain --fullscreen --run-forever
```

- `ML_95_CPU.sh`: The script starts the machine learning (CPU) example:

```
#!/bin/bash
while true; do
./benchmark_model --graph=mobilenet_v1_1.0_224_quant.tflite --num_runs=220000
done
```

- `ML_95_NPU.sh`: The script starts the machine learning (NPU) example:

```
#!/bin/bash
while true; do
./benchmark_model --graph=mobilenet_v1_1.0_224_int8_converted.tflite \
--num_runs=220000 --external_delegate_path=/usr/lib/libneutron_delegate.so
done
```

- `ML_95_GPU.sh`: The script starts the machine learning (GPU) example:

```
#!/bin/bash
while true; do
./benchmark_model --graph=mobilenet_v1_1.0_224_quant.tflite \
--num_runs=220000 --use_gpu=true
done
```

- `remote_camera.sh`: The script starts the remote camera example:

```
#!/bin/sh

echo "Setup the media pipeline..."
media-ctl -l "'ap130x 2-003c':2->'csidev-4ad30000.csi':0 [1]"
media-ctl -l "'csidev-4ad30000.csi':1 -> '4ac10000.syscon:formatter@20':0 [1]"
media-ctl -V "'ap130x 2-003c':2 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'csidev-4ad30000.csi':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'4ac10000.syscon:formatter@20':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
```

```
media-ctl -V "'crossbar':2 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.0':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.1':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.2':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.3':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.4':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.5':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.6':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
media-ctl -V "'mxc_isi.7':0 [fmt: UYVY8_1X16/1920x1080 field:none]"
gst-launch-1.0 -v v4l2src device=/dev/video0 ! \
"video/x-raw,format=NV12,width=1920,height=1080,framerate=60/1" ! \
v4l2h264enc extra-controls="encode,video_bitrate_mode=0,\
frame_level_rate_control_enable=1,\
h264_mb_level_rate_control=1,video_bitrate=10000000" ! \
queue ! filesink location=camera-h264-1080p.mp4
```

- `streamcpy_loop.sh`: The script starts the stream example:

```
#!/bin/sh

while true; do
stream -M 40M -N 5 -P 6
done
```

- `preview.sh`: The script starts one of the Smart Home examples:

```
#!/bin/bash
media-ctl -l "'ap130x 2-003c':2->'csidev-4ad30000.csi':0 [1]"
media-ctl -l "'csidev-4ad30000.csi':1 -> '4ac10000.syscon:formatter@20':0 [1]"
media-ctl -V "'ap130x 2-003c':2 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'csidev-4ad30000.csi':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'4ac10000.syscon:formatter@20':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'crossbar':2 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.0':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.1':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.2':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.3':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.4':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.5':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.6':0 [fmt: UYVY8_1X16/1280x800 field:none]"
media-ctl -V "'mxc_isi.7':0 [fmt: UYVY8_1X16/1280x800 field:none]"

gst-launch-1.0 --no-position v4l2src device=/dev/video0 ! video/x-
raw,width=1280,height=800 ! tee name=t ! queue ! waylandsink t. ! queue max-size-
buffers=2 leaky=2 ! videoconvert ! jpegenc ! multifilesink location=frame.jpg
```

- `os08a20_preview.sh`: This script starts the os08a20 camera example:

```
export LIBCAMERA_PIPELINES_MATCH_LIST='nxp/neo,imx8-isi,uvc'
gst-launch-1.0 \
libcamerasrc camera-name="/base/soc/bus@42000000/i2c@42530000/os08a20_mipi@36" ! \
video/x-raw, format=YUY2 ! \
queue ! waylandsink
```

- `surround_view.sh`: This script starts the surround_view example:

```
gst-launch-1.0 glvideomixer name=comp \
sink_0::xpos=0 sink_0::ypos=0 sink_0::width=960 sink_0::height=640 \
sink_1::xpos=0 sink_1::ypos=640 sink_1::width=960 sink_1::height=640 \
sink_2::xpos=960 sink_2::ypos=0 sink_2::width=960 sink_2::height=640 \
sink_3::xpos=960 sink_3::ypos=640 sink_3::width=960 sink_3::height=640 ! \
glimagesink render-rectangle="<0,0,1920,1280>" \
libcamerasrc camera-name="/base/soc/bus@42000000/i2c@42530000/max96724@27\
/i2c-mux/i2c@0/mx95mbcam@40" ! video/x-raw,format=YUY2 ! \
imxvideoconvert_g2d video-warp-enable=true \
video-warp-coord-file=/root/Buffer_63dc20ab1c.bin ! \
video/x-raw,format=YUY2,width=1920,height=1280 ! queue ! comp.sink_0 \
libcamerasrc camera-name="/base/soc/bus@42000000/i2c@42530000/max96724@27\
```

```
/i2c-mux/i2c@1/mx95mbcam@40" ! video/x-raw,format=YUY2 ! \
imxvideoconvert_g2d video-warp-enable=true \
video-warp-coord-file=/root/Buffer_63dc20ab1c.bin ! \
video/x-raw,format=YUY2,width=1920,height=1280 ! queue ! comp.sink_1 \
libcamerasrc camera-name="/base/soc/bus@42000000/i2c@42530000/max96724@27\
/i2c-mux/i2c@2/mx95mbcam@40" ! video/x-raw,format=YUY2 ! \
imxvideoconvert_g2d video-warp-enable=true \
video-warp-coord-file=/root/Buffer_63dc20ab1c.bin ! \
video/x-raw,format=YUY2,width=1920,height=1280 ! queue ! comp.sink_2 \
libcamerasrc camera-name="/base/soc/bus@42000000/i2c@42530000/max96724@27\
/i2c-mux/i2c@3/mx95mbcam@40" ! video/x-raw,format=YUY2 ! \
imxvideoconvert_g2d video-warp-enable=true \
video-warp-coord-file=/root/Buffer_63dc20ab1c.bin ! \
video/x-raw,format=YUY2,width=1920,height=1280 ! queue ! comp.sink_3
```

6 Reducing power consumption

The overall system power consumption depends on the software optimization and the system hardware implementation. The following list of suggestions can help reduce system power consumption. Some of these suggestions are already implemented in the Linux BSP and/or SDK. The system of each individual user can undergo further optimizations.

Note: *Further power optimizations are planned in future software releases.*

- Apply clock gating by configuring registers in the CCM, whenever clocks or modules are not used.
- For Run modes, use the slowest frequency that can still meet the application requirements.
- Minimize the number of operating PLLs. Enabled PLLs can consume a few milliamps of current.
- Put the SoC into Low-power modes whenever possible, as long as it can still support the application requirements. Consider the following example:
  - Put the system into Suspend mode when it can enter deep sleep.
  - Power off the CA55 cores and other domains for low-load use cases.
- For each operating mode, use the lowest voltage (with the power supply tolerance) that can still meet the requirements of voltage specifications in the data sheet.
- DDR interface optimization:
  - Use board routing of the DDR memories carefully, maintaining PCB trace lengths as short as possible.
  - Use the proper output driver impedance for DDR interface pins that provides good impedance matching. To save current through DDR I/O pins, select the lowest possible drive strength that provides the required performance.
  - Use of LPDDR5/LPDDR4x memory offerings in the latest process technology can significantly reduce the power consumption of the DDR devices and the DDR I/O.

The following sections provide more details for system optimization. These sections are not exhaustive lists of features that can provide power reductions, but they are the easiest and most common ones.

6.1 Run fast and idle

NXP testing and various research have shown that for most customer use cases, the best power/energy management protocol is to run the cores at maximum speeds for the workload and then drop to the lowest power mode as soon as possible. This strategy cannot provide optimal energy savings for the use cases where constant data is being processed, for example, low-latency audio playback. However, this strategy does work for other standard workloads. Consider this trade-off for each application to quantify the overall effect on the system power/energy consumption.

Users must place the i.MX 95 into the Low-power mode as far as possible.

6.2 Clock gating

The CCM inside the i.MX 95 provides a programmable method to disable the clock sources for modules when the modules are not used. To reduce energy waste, always configure the CCM registers. Driving any inactive signal, whether on the SoC or the PCB, is simply charging and discharging the line and the load capacitance of this signal. The NXP BSP-released software implements clock gating by default.

6.3 DDRC auto clock gating

When the bus is idle after the number of cycles configured in the `ssi_idle_strap` field in the DDR BLK_CTRL module, the DDRC does auto clock gating to save power. This feature can be used to balance DDR subsystem performance and power significantly. The number of idle cycles before clock gating can be adjusted dynamically based on the actual use case to fine-tune the power saving.

In the i.MX 95, `auto_clk_gating` is used to enable the DDRC auto clock gating. Therefore, power is saved when there is no access to the DDR after the programmed idle count expires. "Write 0" disables the auto clock gating and the "write non-zero" value sets the `ssi_idle_strip` to this non-zero value and enable the auto clock gating. A value < 256 has some significant side effort for DDR performance, so a value >=256 is suggested when the user wants to enable it. When the auto clock gating is enabled, a high-resolution display like 4K 60 fps can flicker at lower DDR frequency. It is recommended not to adjust the `auto_clk_gating` when the display/NPU is running.

6.4 PLL reduction

Each PLL block consumes significant energy when active. Each application has unique requirements, but, if possible, reduce the number of operating PLLs. The CCM within the i.MX 95 provides Root Clock mux and programmable control for each PLL either by direct control mode or CPU Low-power mode. As a result, the Root Clocks source is allowed to modify to limit the PLL source and reduce the number of active PLLs when operating. Ensure that the application considers the PLL relock time when transitioning back to full operation.

6.5 DDR interface optimization

To optimize the DDR interface, the suggestions are as follows:

- Employ careful board routing of the DDR memories, maintaining the PCB trace lengths as short as possible. Longer trace lengths and more vias create more PCB capacitance for the signal, resulting in more energy wastage along the signal path.
- Keep the on-die termination (ODT) value as low as possible. The termination used greatly influences the power consumption of the DDR interface pins. To ensure the ODT variance does not reduce the bus signal integrity, simulate the DDR interface.
- Use an appropriate output driver impedance for the DDR interface pins that provide good impedance matching. Select the lowest possible drive strength that provides the required performance to reduce the current flowing through the DDR I/O pins. Remember that simulation must be done to ensure signal integrity.
- The use of the DDR memory offerings in the latest process technology can significantly reduce the power consumption of the DDR devices and the DDR I/O.
- Sizing DDR memory is important. For example, if you select 4 GB memory when only 2 GB is used, you are wasting the refresh current for the unused 2 GB of DDR.
- Sizing of ECC DDR regions is important as they use more energy for this feature.

6.6 Power gating of PHYs

The PHYs of unused modules often get overlooked when searching for power savings. Many PHYs contain local PLLs or clocking circuits and voltage references, which consume power even when not in use. As a result, high-speed PHYs like Ethernet, MIPI, and USB get affected.

6.7 Distribution of workloads

The concept of distributed workloads involves reviewing the system requirements to identify the most suitable SoC block for each task. By distributing the workload, the system can return to the "Idle" state sooner. System designers must ensure that the design uses the optimal cores for the specific workloads or tasks on the i.MX 95 efficiently. It provides significant power savings if the system can return to the low-power state faster (run fast and idle).

6.8 Thermal management to reduce leakage

Thermal management is also a key element of power reduction. As the temperature rises, so does the SoC gate leakage current for each gate within the device. Millions of high-gate leakages add up when looking for the lowest power consumption. As explained earlier, with any power savings, the temperature of the SoC reduces, and the lifetime reliability of the device improves.

As each system is unique, the system designer must ensure that the operating temperature of the SoC is as low as possible to reduce the leakage current loss. If this temperature cannot be achieved from software controls, the designer must include a heat sink or other thermal management methods to remove the heat from the SoC.

7 Enhancing precision in power measurements

Fine-tuning shunt resistors for specific applications can significantly improve measurement accuracy, typically by about 10-15%.

If your goal is to achieve performance comparable to a laboratory-grade instrument with extremely low offset and error (around ±0.02% or better) across all current ranges, consider using a higher-value shunt resistor in the PAC1934T environment for low-load measurements. This approach increases the sense voltage, helping to minimize offset-related errors.

8 Note about the source code in the document

Example code shown in this document has the following copyright and BSD-3-Clause license:

TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE

9 Acronyms

Table 84 lists the acronyms used in this document.

Table 84. Acronyms

Acronym	Meaning
ADC	Analog-to-Digital Converter
AHB	Arm AMBA High-performance Bus
AON	Always-On
APLL	Auxiliary Phase-Locked Loop
Arm	Advanced RISC machine processor architecture
AXI	Arm Advanced eXtensible Interface
BBNSM	Battery-Backed Non-Secure Module
BBSM	Battery-Backed Security Module
BSP	Board Support Package
CA55	Arm Cortex-A55 processor
CCM	Clock Controller Module
CM33	Arm Cortex-M33 processor
CM7	Arm Cortex-M7 processor
DAC	Digital-to-Analog Converter
DC	Display Controller
DDR	Dual Data Rate DRAM
DMA	Direct Memory Access controller
DRAM	Dynamic Random-Access Memory
DVFS	Dynamic Voltage and Frequency Scaling
EVK	Evaluation Kit
FIRC	FAST Internal Reference Clock
GND	Ground
GPIO	General-Purpose Input/Output
GPU	Graphics Processing Unit
High-Z	High-impedance
HSIO	High-Speed I/O
I/Os	Inputs/Outputs
IOMUX	Input/Output Multiplexing
IOMUXC	Input/Output Multiplexing Controller

Table 84. Acronyms*...continued*

Acronym	Meaning
LDO	Low Drop-Out regulator
LPAV	Low-Power Audio/Video domain
LPDDR4X	Low-Power DDR4X SDRAM
LPDDR5	Low-Power DDR5 SDRAM
LPTMR	Low-Power Timer
LVD	Low-Voltage Detector
MIPI-CSI	MIPI - Camera Serial Interface controller
MIPI-DSI	MIPI - Display Serial Interface controller
MU	Messaging Unit
ND	Nominal Drive
NOC	Network-On-Chip
NPU	Neural Processing Unit
OD	OverDrive
OTP	One-Time Programmable
PCB	Printed-Circuit Board
PLL	Phase-Locked Loop
PMC	Power Management Controller
PMIC	Power Management-Integrated Circuit
POR	Power On Reset
RAM	Random Access Memory
ROM	Read-Only Memory
RTC	Real-Time Clock
SCU	Snoop Control Unit
SDK	Software Development Kit
SIM	System Integration Module
SIRC	Slow Internal Reference Clock
SoC	System on Chip
SM	System Manager
SPLL	System Phase-Locked Loop
SRAM	Static Random Access Memory
TCM	Tightly Coupled Memory
UART	Universal Asynchronous Receiver/Transmitter
USB	Universal Serial Bus
USB 2.0	USB version 2.0 peripheral
USB OTG	USB On The Go
VPU	Video Processing Unit

Table 84. Acronyms...*continued*

Acronym	Meaning
WFI	Wait For Interrupt
WoL	Wake-on-LAN
WUU	Wake-Up Unit

10 Revision history

Table 85 summarizes the revisions to this document.

Table 85. Revision history

Document ID	Release date	Description
AN14449 v.1.0	9 February 2026	Initial public release

Legal information

Definitions

Draft — A draft status on a document indicates that the content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included in a draft version of a document and shall have no liability for the consequences of use of such information.

Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Terms and conditions of commercial sale — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at https://www.nxp.com/profile/terms, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Suitability for use in non-automotive qualified products — Unless this document expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

HTML publications — An HTML version, if available, of this document is provided as a courtesy. Definitive information is contained in the applicable document in PDF format. If there is a discrepancy between the HTML document and the PDF document, the PDF document has priority.

Translations — A non-English (translated) version of a document, including the legal information in that document, is for reference only. The English version shall prevail in case of any discrepancy between the translated and English versions.

Security — Customer understands that all NXP products may be subject to unidentified vulnerabilities or may support established security standards or specifications with known limitations. Customer is responsible for the design and operation of its applications and products throughout their lifecycles to reduce the effect of these vulnerabilities on customer's applications and products. Customer's responsibility also extends to other open and/or proprietary technologies supported by NXP products for use in customer's applications. NXP accepts no liability for any vulnerability. Customer should regularly check security updates from NXP and follow up appropriately.

Customer shall select products with security features that best meet rules, regulations, and standards of the intended application and make the ultimate design decisions regarding its products and is solely responsible for compliance with all legal, regulatory, and security related requirements concerning its products, regardless of any information or support that may be provided by NXP.

NXP has a Product Security Incident Response Team (PSIRT) (reachable at PSIRT@nxp.com) that manages the investigation, reporting, and solution release to security vulnerabilities of NXP products.

NXP B.V. — NXP B.V. is not an operating company and it does not distribute or sell products.

Trademarks

Notice: All referenced brands, product names, service names, and trademarks are the property of their respective owners.

NXP — wordmark and logo are trademarks of NXP B.V.

AMBA, Arm, Arm7, Arm7TDMI, Arm9, Arm11, Artisan, big.LITTLE, Cordio, CoreLink, CoreSight, Cortex, DesignStart, DynamIQ, Jazelle, Keil, Mali, Mbed, Mbed Enabled, NEON, POP, RealView, SecurCore, Socrates, Thumb, TrustZone, ULINK, ULINK2, ULINK-ME, ULINK-PLUS, ULINKpro, μVision, Versatile — are trademarks and/or registered trademarks of Arm Limited (or its subsidiaries or affiliates) in the US and/or elsewhere. The related technology may be protected by any or all of patents, copyrights, designs and trade secrets. All rights reserved.

CoreMark — is a registered trademark of SPEC.

EdgeLock — is a trademark of NXP B.V.

eIQ — is a trademark of NXP B.V.

Synopsys & Designware — are registered trademarks of Synopsys, Inc.

Contents

Date of release: 9 February 2026
Document identifier: AN14449