DA1468x Power Measurements

AN-B-037

Abstract

This document provides information needed to perform power measurements with DA1468x family of chips.
For the measurements are used:
- the DA14681 ProDK
- the Agilent N6705 power analyzer with it's SW
- the SDK and SW tools provided by Dialog

Contents

Figures

Tables

1 Terms and definitions

LP clock	Low Power clock
HCLK	AMBA High Speed Bus clock
PCLK	AMBA Peripheral Bus clock
RCX	RC 10.5MHz internal LP clock
RC32K	RC 32KHz internal LP clock
RC16M	RC 16MHz internal clock
XTAL32K	32768Hz external crystal LP clock
XTAL16M	16MHz external crystal clock
PLL48	PLL 48MHz internal clock
PLL96	PLL 96MHz internal clock
MB	Mother Board
DB	Daughter Board
ProDK	Pro Development Kit
FSM	Finite State Machine
OOB	Out Of the Box

2 References

[1] DA1468x, Datasheet, Dialog Semiconductor

[2] Agilent Technologies, DC Power Analyser Model N6705 User's Guide, © Agilent Technologies, Inc. 2007 - 2012

[3] UM-B-060, User Manual DA1468x/DA1510x PRO-Development kit

[4] AN-B-061, DA1468x Application hardware design guidelines

[5] UM-B-047, User manual, DA1468x Getting Started with the Development Kit

[6] UM-B-056, User manual, DA1468x Software Developer's Guide

3 Introduction

This document describes the procedure for performing power consumption measurements on the DA1468x platforms using the DA1468x ProDK or just the DA1468x daughterboard.

Please refer to the User Manual for the DA1468x Pro-Development kit and related documentation for information about the ProDK.

The DA1468x includes a complete power management IC and battery charger. It is not just a SoC but SoC+PMU+Charger all in one package.

To compare the power consumption of the DA1468x, the reader must think of it like the complete PCB of a solution with all the power supplies needed, like DCDC, LDOs and the battery charger, all of them in a tiny package without any additional part requirements other than the decoupling capacitors and one inductor.

The measurements described in this document are performed by default on the DA1468x ProDK daughterboard only, detached from the motherboard, unless otherwise clearly stated.

For measurements accuracy, especially for the very low power states like extended sleep, a power analyser should be used. For the measurements presented in this document the Agilent N6705B power analyser is used. Please refer to the instrument documentation for details about it.

It is suggested to use the sampling at the N6705B instead of SW as the sampling is faster this way and the measurements more accurate.

The document also covers the preparation of the HW and equipment needed for the measurements, explains the parameters to set in the SW for accurate measurements.

Dialog provides also an embedded tool in the ProDK Motherboard which provides power measurements capabilities. This tool is called Power Profiler and it is very accurate for measuring the power consumption while the device is active, but it has lower accuracy when measuring the extremely low currents when the DA1468x is in sleep mode.

In chapter 4 is provided the information of the CPU power consumption alone (including the minimal of the subsystems needed to have it running). The chapter has tables and graphs for all possible clocks and clock dividers. This information allows the calculation of power consumption per MHz and the power consumption per MIPS. The CPU power consumption information is useful only in very special cases where need to estimate the energy cost of code consuming certain amount of MIPs.

The measurement using the power analyser will provide the most accurate and complete energy consumption information for a specific test case.

The power measurements in this document cover some typical cases and provide reference numbers. The reader can measure the power consumption on any scenario needed by modifying accordingly some of the example projects provided by Dialog and included in the SDK.

IMPORTANT NOTES:

1. The 16MHz XTAL must be properly trimmed in the DA1468x. If not the measurements will deviate significantly in some cases.
2. The bandgap is trimmed for each chip at production. This must not be changed for production chips.
3. Use mass production chips NOT engineering samples for measurements. The DA1468x hardware guidelines application note explains how to recognize the version on the package of the chip. [4]
4. Prefer to use ProDK motherboard and daughterboard rev.E. Early versions of the ProDK have known leakages which affect the measurements.

4 Preparation of the SW, HW, instruments and equipment

4.1 ProDK HW preparation

The standard DA1468x ProDK does not need any modifications to be used for getting the power measurements. If measuring with the daughterboard plugged on the Motherboard, have the motherboard USB2 DBG plugged to a PC USB port to avoid having any leakages.

The recommended approach is to program the FLASH and detach the daughterboard form the Motherboard. Then supply the daughterboard through the coin-cell battery connector at the bottom on the daughterboard and flip the switch on the daughterboard to position larked ‘COIN’.

If the shielded box is used then have the device for the connection measurements also placed in the box. It is recommended to use a shielded box as the environment can affect the power consumption results.

Figure 1: DA1468x ProDK with the default (OOB) jumper placement

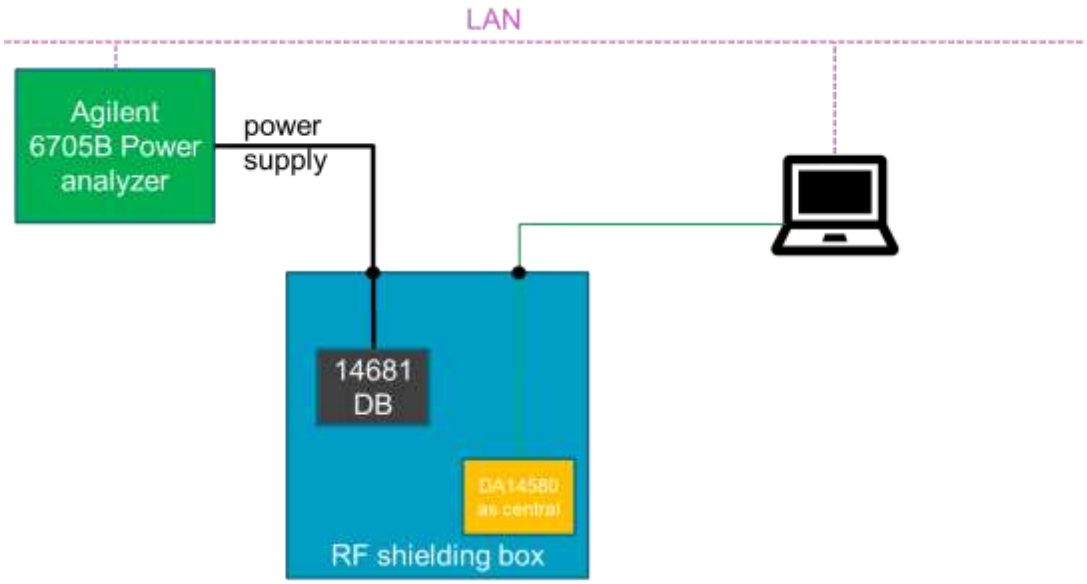

Figure 2: Topology block diagram

4.2 Power analyser use

There are many configurations in which the power analyser can be used. In this document we use it as power supply source for measuring the current and log its changes.

For the correct connection and use of the power analyser please refer to the instruments documentation.

The power consumption can be measured with the DB either unplugged from the MB or with the DB plugged on the MB.

Prefer to have the DB unplugged from the MB for measurements not requiring the breakout connectors of the MB. Have the DB plugged on the MB with all the jumpers of the MB removed when the test requires connecting either loads for testing or GPIOs for control/testing to DA1468x.

IMPORTANT NOTE:

Avoid using low sampling rates in the instrument settings. Low sampling rates may lead to aliasing. Sampling rate of at least 10 ksamples/s is suggested to avoid aliasing and probably wrong measurements especially for the sleep modes.

4.3 BLE advertisement and connection power consumption

4.3.1 HW Preparation

Voltage: 3.8V

Power supply of the daughterboard: through the coin cell contacts

SW project to use: *ble_adv*, included in the generally available SDK.

To prepare the daughterboard for the measurement use the Smartsnippets Studio to compile and write the code to the FLASH of the daughterboard.

Plug the daughterboard on the motherboard and switch to VBAT position the small switch on the daughterboard so it will get power from motherboard. The motherboard has all the circuits and the J-Link to enable the user to FLASH the device through SWD or UART interface.

Once the DB is programmed, unplug it from the MB and connect power supply at the coin cell contacts, using soldered leads or a coin cell adapter or clamps.

Flip the switch on the daughterboard to COIN and turn on the power supply.

NOTE: It is recommended to perform the measurements in environment without RF activity (like active Wi-Fi and BLE/BT devices). The use of a shielded box is highly recommended, to avoid increased power consumption due scanning from other devices, or attempts to connect, or re-transmissions caused by RF interference due to other protocols on same band.

4.3.2 SW preparation

For this measurement, is used the *ble_adv* project included in the SDK. The application is suitable as it is to measure power consumption for Advertising and for Connection without advertising in parallel.

The typical default settings of the application in **custom_config_qspi.h** are fine for the general case and they are the ones used for the measurements presented in this document.

Before start please verify that the following macro in the custom_config_qspi.h is as follows:

```
#define dg_configFLASH_POWER_DOWN                (1)
```

The desired BLE parameters can be easily set in the **ble_config.h** file under **sdk\ble\config** path of the SmartSnippets Studio project or by copying the parameters of interest from **ble_config.h** into **custom_config_qspi.h** and set the desired value in the **custom_config_qspi.h** without touching the **ble_config.h** at all.

It is recommended to copy the macros for the parameters in the custom_config_qspi.h and set the values there. Changing the macro value sin the SDK files will affect all the projects.

The main macros of interest to set and their default values are:

```
#define defaultBLE_STATIC_ADDRESS          { 0x01, 0x00, 0x80, 0xCA, 0xEA, 0x80 }
#define defaultBLE_ADDRESS_TYPE            PUBLIC_ADDRESS
#define defaultBLE_ADVERTISE_INTERVAL_MAX  (BLE_ADV_INTERVAL_FROM_MS(687.5))  // 687.5ms
#define defaultBLE_ADVERTISE_INTERVAL_MIN  (BLE_ADV_INTERVAL_FROM_MS(687.5))  // 687.5ms
#define defaultBLE_ADVERTISE_DATA_LENGTH   (28)
#define defaultBLE_PPCP_INTERVAL_MIN       (BLE_CONN_INTERVAL_FROM_MS(10))    // 10ms
#define defaultBLE_PPCP_INTERVAL_MAX       (BLE_CONN_INTERVAL_FROM_MS(20))    // 20ms
```

IMPORTANT NOTE: The above macro values must be set in the **custom_consgi_qspi.h** before including the bsp_defaults.h.

```
/* Include bsp default values */
#include "bsp_defaults.h"
```

The developer can also set the BLE parameters using the provided API calls.

To set the advertising interval, use the function:

```
ble_error_t ble_gap_adv_intv_set(uint16_t adv_intv_min, uint16_t adv_intv_max);
```

The function must be called in the main BLE task (`static void ble_adv_demo_task(void *pvParameters))` prior of starting advertising.

The connection parameters are set by the master (central) device. If there is no control to the connection parameters at the master it is possible to trigger an update of the connection parameters from the DA1468x. To do so, after the connection is established, use a timer to trigger the connection parameters update if needed. The technique is used in the example pxp_reporter and the developer should follow the pxp_reporter example as guide to add the update connection parameters functionality in the ble_adv example.

Having all the needed changes done, compile the application using the **DA146881-01-Release_QSPI (DA14680/1 Release build configuration for cached QSPI mode)** configuration and FLASH the device.

NOTE: Since the SDK v.1.0.8.1050.1, there is also a special project for power measurements which makes testing the various parameters easier. The project is named *power_demo*. The instructions for building and using the *power_demo* can be found in the readme.md text file under the project folder.

4.3.3 Measurement procedure

For the measurement, have the power analyser set at the voltage desired within the voltage of the DA1468x as those defined in the datasheet. The measurements in this document performed at 3.8V and the current supply capability at 0.1A as this represents a typical battery used in wearables. Have the power analyser supply/measurement channel to OFF.

For average current measurements, data logger mode should be used on N6705B. Select the current only trace to be captured or current and power if required and set an appropriate capture time. Suggested is to set 3-5 min capture time to result in more accurate and consistent results.

In order to measure individual parts of the event interval, such as current during sleep or during active state, use the SCOPE mode which provides higher sampling rate, thus increased accuracy.

By adjusting time per division and number of points, sampling period is altered. After capture is stopped, markers may be used to limit the area of interest and provide the required measurements within it. Measurement settings allow for various useful values to be selected (average, max/min, charge etc.). It is suggested that measurements are repeated for a couple of events, so that rough averaging may be performed. When using the SCOPE function, especially when sleep current measurements are intended, it is important that 'AUTO' is selected in the 'Ranges…' setting window, found or the lower part of the screen to the right. This way, the 'seamless range' option of the Power Analyser is employed which effectively increases the dynamic range, allowing mA and uA measurements at the same time, without having to adjust the range.

To measure, detach the programmed daughterboard from motherboard and connect the power analyser leads to the battery coin connectors at the bottom of the DB. Then power ON the supply/measurement channel of the power analyser.

On the power analyser use the Data Logger function to capture the current and power if required for 3-5 minutes to have realistic averaging.

Configure the golden unit using with the connection interval parameters for the specific test.

Figure 3, Figure 4 and Figure 5 show the power consumption footprint for advertising and averaging measurement over longer period of time while advertising.

The device will get in/out of sleep as needed based on the parameters set for the BLE. Developer does not have to change anything.

In case there is need to measure various power save modes of the DA1468x, just change accordingly the parameter in the `pm_set_sleep_mode(pm_mode_extended_sleep);` in the main() function of the project.

The measurement numbers include:

- The DA1468x silicon which integrates
  - The DCDC
  - The LDOs
  - The Battery charger
  - The power supply circuitry for 1V8, 1V8P, 3V3, 1V4 and 1V2 rails supplying internal and external circuits
- The QSPI FLASH winbond W25Q80EW

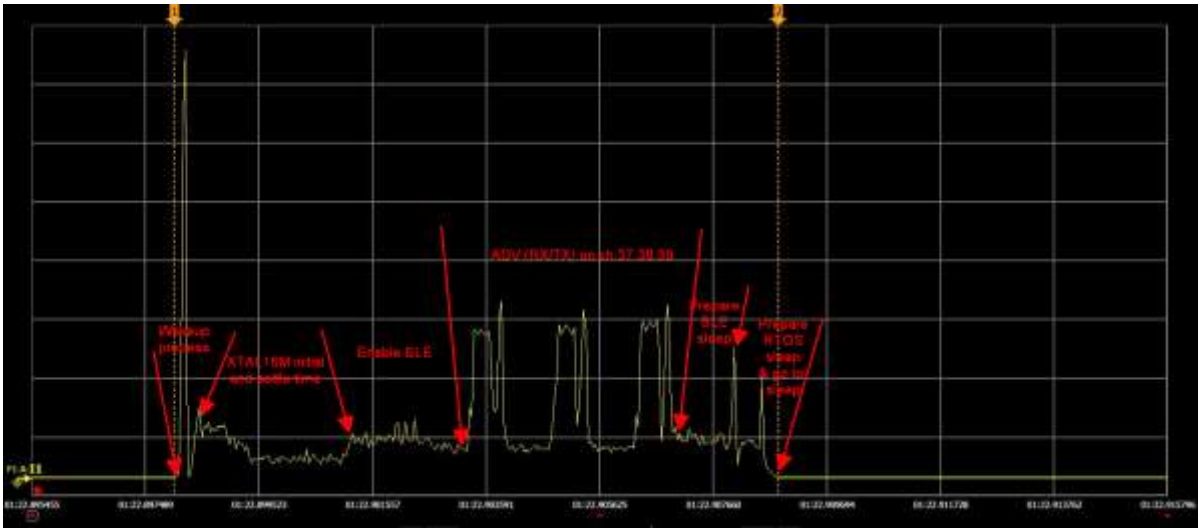

Figure 3: Advertising, footprint on data logger

In Figure 3 the first large spike during the wakeup period is due to capacitors charging. While the system is in sleep the charged capacitors will slowly discharge. When the system wakes up, the capacitors will recharge. The spike gets bigger for longer sleep periods as the capacitors will discharge more with time.

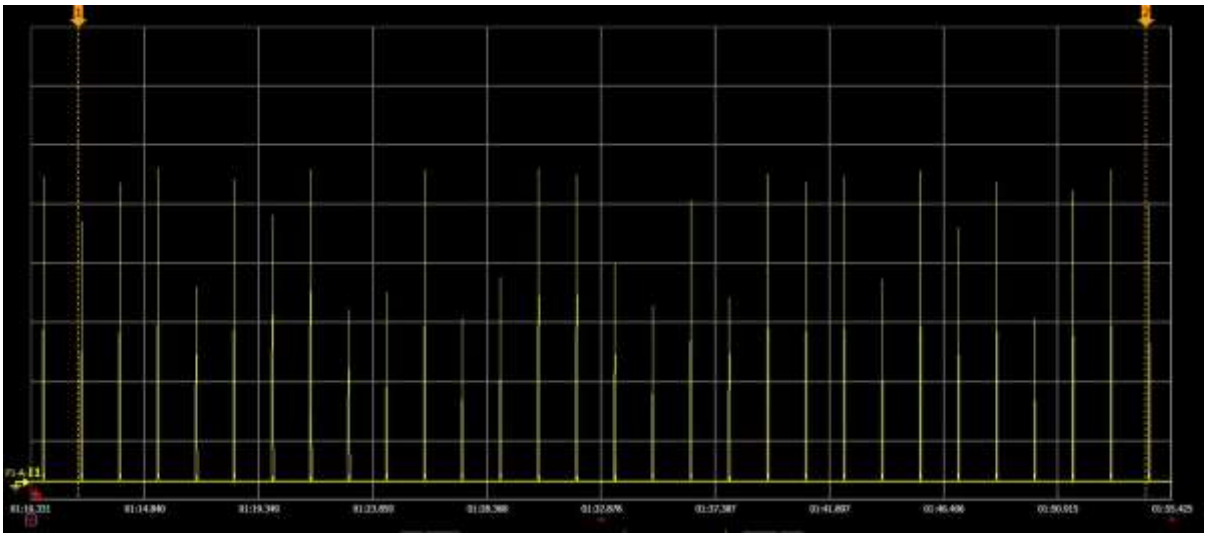

Figure 4: Advertising average over longer period of time.

Every vertical line is an advertising active state as shown on Figure 3

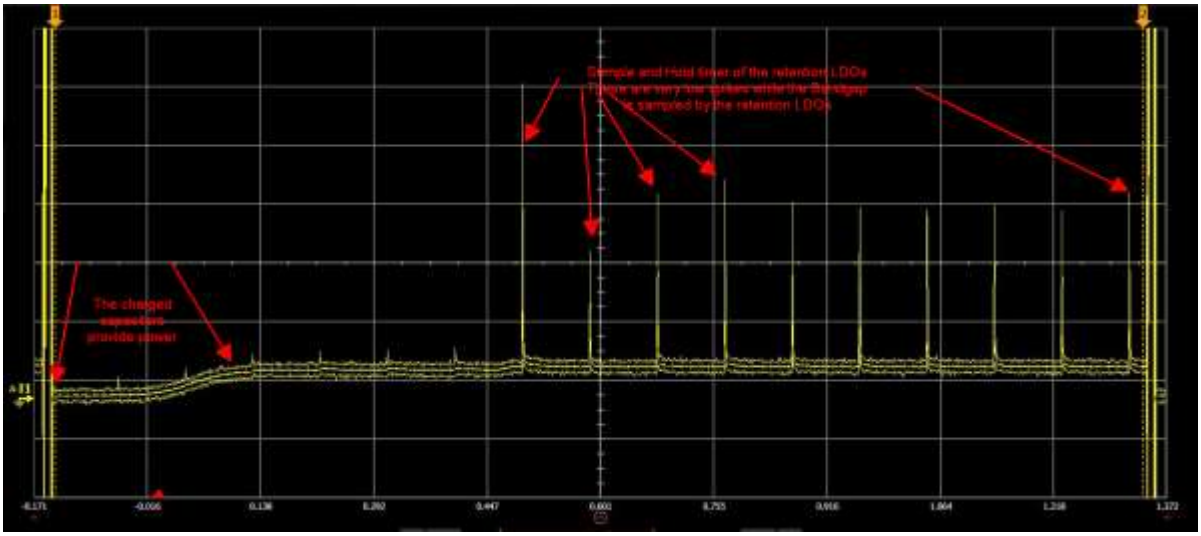

Figure 5: Sleep period between two advertising events

IMPORTANT NOTE: The device will not power save for the first 8 second since power up (or boot). Start getting the measurement after the 8 seconds to get the correct results. 8 Seconds is the maximum possible time in worst case to stabilize XTAL32K which is necessary for sleep. Usually much less time is needed, but the safe value is the 8 sec to allow sleep.

On Figure 5, initially the current is very low even negative in some cases. This is because the charged capacitors are discharging providing energy back to the system, thus for a short time the current on VBat drops.

The periodic low spikes are the Sample & Hold timer function for the retention LDOs. A part of the system is waking periodically by the sleep timer, samples the Bandgap and corrects the “Sample & Hold Retention LDOs” voltage.

4.3.4 Results

Measuring as described in paragraphs 4.3.1 to 4.3.3, the expected power consumption is as presented in Table 1. The numbers refer to the averaging of many sleep and active intervals and overall the average over 3-5 minutes measurement for more realistic results.

Table 1: BLE operation power consumption, VBat=3.8V

Test Case	Average **charge** when **active** (uC)	Average **current** when **active** (mA)	Average **current** when **sleep** (uA)	Overall measurement Average **current** (uA)	Overall measurement Peak **current** (mA)
Advertising: 400 ms interval XTAL32K	17.17	1.53	2.00	44.40	27.27
Advertising: 400 ms interval RCX	18.07	1.47	2.19	46.79	27.82
Advertising: 1500 ms interval XTAL32K	17.52	1.55	3.66	15.29	30.27
Advertising: 1500 ms interval RCX	18.28	1.49	4.00	16.14	28.94
Connected: 400 ms interval XTAL32K	11.84	1.39	1.99	31.60	26.95
Connected: 400 ms interval XTAL32K, Master ppm: 50	11.04	1.31	1.97	29.57	26.87

Connected: 400 ms interval RCX	13.89	1.30	2.19	36.92	27.28
Connected: 2000 ms interval XTAL32K	16.10	1.71	3.84	11.89	29.21
Connected: 2000 ms interval XTAL32K, Master ppm: 50	12.03	1.40	3.90	9.91	29.81
Connected: 2000 ms interval RCX	18.47	1.63	4.09	13.33	28.40
Connected: 4000 ms interval XTAL32K	21.35	2.01	4.11	9.45	29.33
Connected: 4000 ms interval XTAL32K, Master ppm: 50	12.92	1.48	4.21	7.44	29.52
Hibernation				1.35	

Note 1 Measured on DA14681-01 silicon with daughterboard unplugged from motherboard of the ProDK.

Note 2 The charge per active state per interval (uC) differs because the decoupling capacitors discharging level differ. The longer the sleep period the more the capacitors will discharge. This results in higher current peaks on wake-up. This is also why the peak current differs for each case.

Note 3 **Average charge when active:** This is how much charge is used only for the active part of the interval. The active state is from wake to sleep where the system is awake and there is BLE RX/TX activity like advertising event.
The fact that the shorter advertising period cases shows less uC is due to less discharging of the capacitors comparing to longer advertising period cases. This is even more obvious in the connection cases.

The number is an average over many packets.

Note 4 **Average current when active:** Is the average current over many packets during the active state of the interval.

Note 5 **Average current when sleep:** It is the average current over many packets for the sleep state of an interval. Sleep state is from sleep to wake during which the device is at low power and is not having any activity.
At the beginning of the sleep state the current is very low, even negative in some cases. This is because the capacitors discharge and provide energy to the system.
For longer sleep periods the current is getting higher, up to its max value for the sleep mode. This is because as the time is passing the capacitors are drained of energy and there is need for energy from the power supply rails.

Note 6 **Overall measurement average current:** Is the average current of a longer time capture for more intervals (sleep + active). This is the average current expected over time for the selected operation mode.

4.4 CPU power measurement

This is a special measurement performed by Dialog characterizing the power consumption of the CPU in the chip. The table 6 provides all the related information including the uA/MIPs.

This information is useful if one need to fine tune the CPU power consumption for specific piece of code demanding specific amount of MIPS (e.g. a complex calculation algorithm).

In this section the measurements refer to the power consumption of the CPU alone, isolating the power consumption of all the surrounding logic and circuitry in the chip and outside of it.

This measurement also includes the power consumption on bandgap on RC16M which is always running, the power consumption on DCDC and any possible leakages exist in the chip.

The MIPS/Frequency for the M0 CPU core used in the DA1468x is shown in Table 2.

Table 2: M0 MIPS per operation Frequency

Frequency (MHz)	ARM M0 MIPS
8	6.72
12	10.08
16	13.44
24	20.16
48	40.32
96	80.64

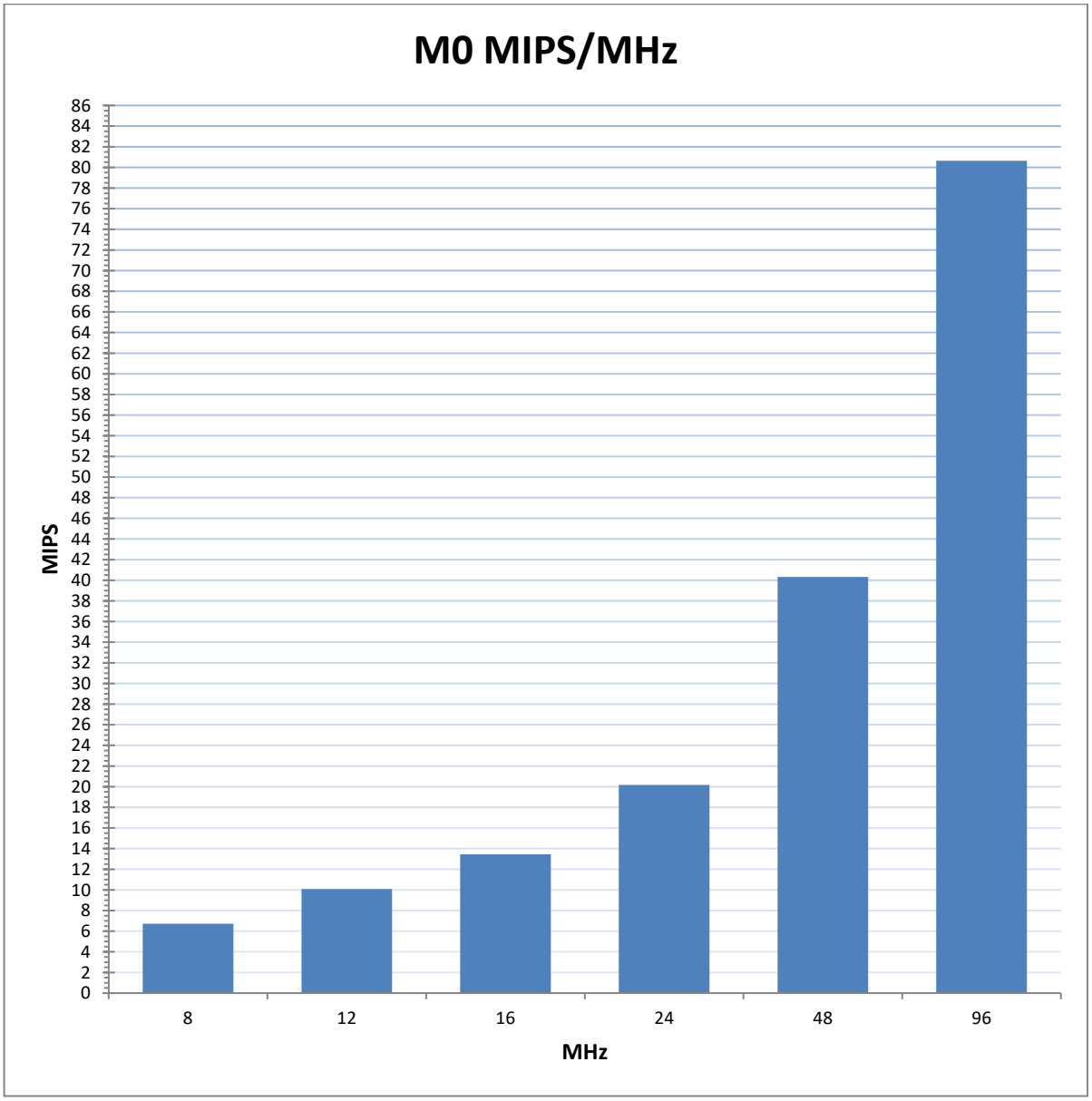

Figure 6: M0 MIPS per operation Frequency

4.4.1 Results

Table 3: MIPS, Clock & CPU Power consumption with HCLK/1

		HCLK/1			
		PCLK/1	**PCLK/2**	**PCLK/4**	**PCLK/8**
RC32K	*mA*	0.275103	N/A	N/A	N/A
	KHz	32			
	uA/MHz	0.041	N/A	N/A	N/A
	MIPS	0.0287			
	uA/MIPS	46.092	N/A	N/A	N/A
XTAL32K	*mA*	0.275341	N/A	N/A	N/A
	KHz	32			
	uA/MHz	0.049	N/A	N/A	N/A
	MIPS	0.0287			
	uA/MIPS	54.396			
RC16	*mA*	0.562	0.553	0.531	0.525
	MHz	16			
	uA/MHz	18.004	17.470	16.046	15.713
	MIPS	13.44			
	uA/MIPS	21.433	20.797	19.103	18.706
XTAL16M	*mA*	1.208	1.193	1.184	1.181
	MHz	16			
	uA/MHz	58.383	57.424	56.909	56.702
	MIPS	13.44			
	uA/MIPS	69.504	68.362	67.748	67.502
PLL48M	*mA*	3.413	3.345	3.301	3.278
	MHz	48			
	uA/MHz	65.400	63.993	63.059	62.595
	MIPS	40.32			
	uA/MIPS	77.857	76.182	75.070	74.518
PLL96M	*mA*	5.331	5.193	5.105	5.061
	MHz	96			
	uA/MHz	52.678	51.238	50.324	49.872
	MIPS	80.64			
	uA/MIPS	62.712	60.998	59.910	59.371

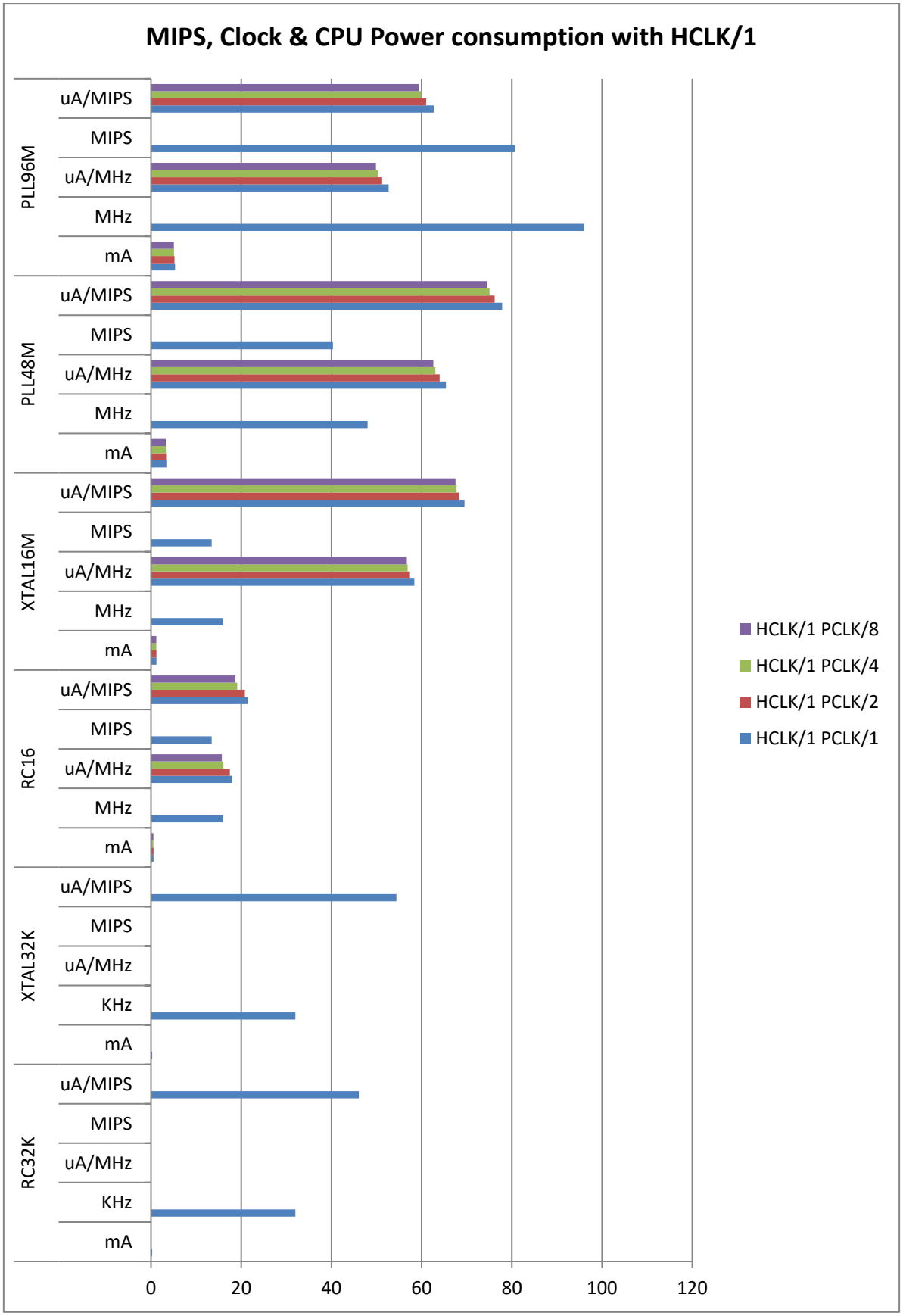

Figure 7: MIPS, Clock & CPU Power consumption with HCLK/1

Table 4: MIPS, Clock & CPU Power consumption with HCLK/2

		HCLK/2			
		PCLK/1	**PCLK/2**	**PCLK/4**	**PCLK/8**
RC16	*mA*	0.411	0.406	0.403	0.402
	MHz	8			
	uA/MHz	17.138	16.538	16.188	16.047
	MIPS	6.72			
	uA/MIPS	20.403	19.688	19.271	19.104
XTAL16M	*mA*	0.845	0.831	0.818	0.814
	MHz	8			
	uA/MHz	71.394	69.696	68.040	67.541
	MIPS	6.72			
	uA/MIPS	84.992	82.972	81.000	80.406
PLL48M	*mA*	2.523	2.479	2.457	2.447
	MHz	24			
	uA/MHz	93.736	91.900	90.960	90.532
	MIPS	20.16			
	uA/MIPS	111.591	109.404	108.285	107.776
PLL96M	*mA*	3.742	3.653	3.608	3.585
	MHz	48			
	uA/MHz	72.250	70.405	69.453	68.983
	MIPS	40.32			
	uA/MIPS	86.012	83.815	82.682	82.123

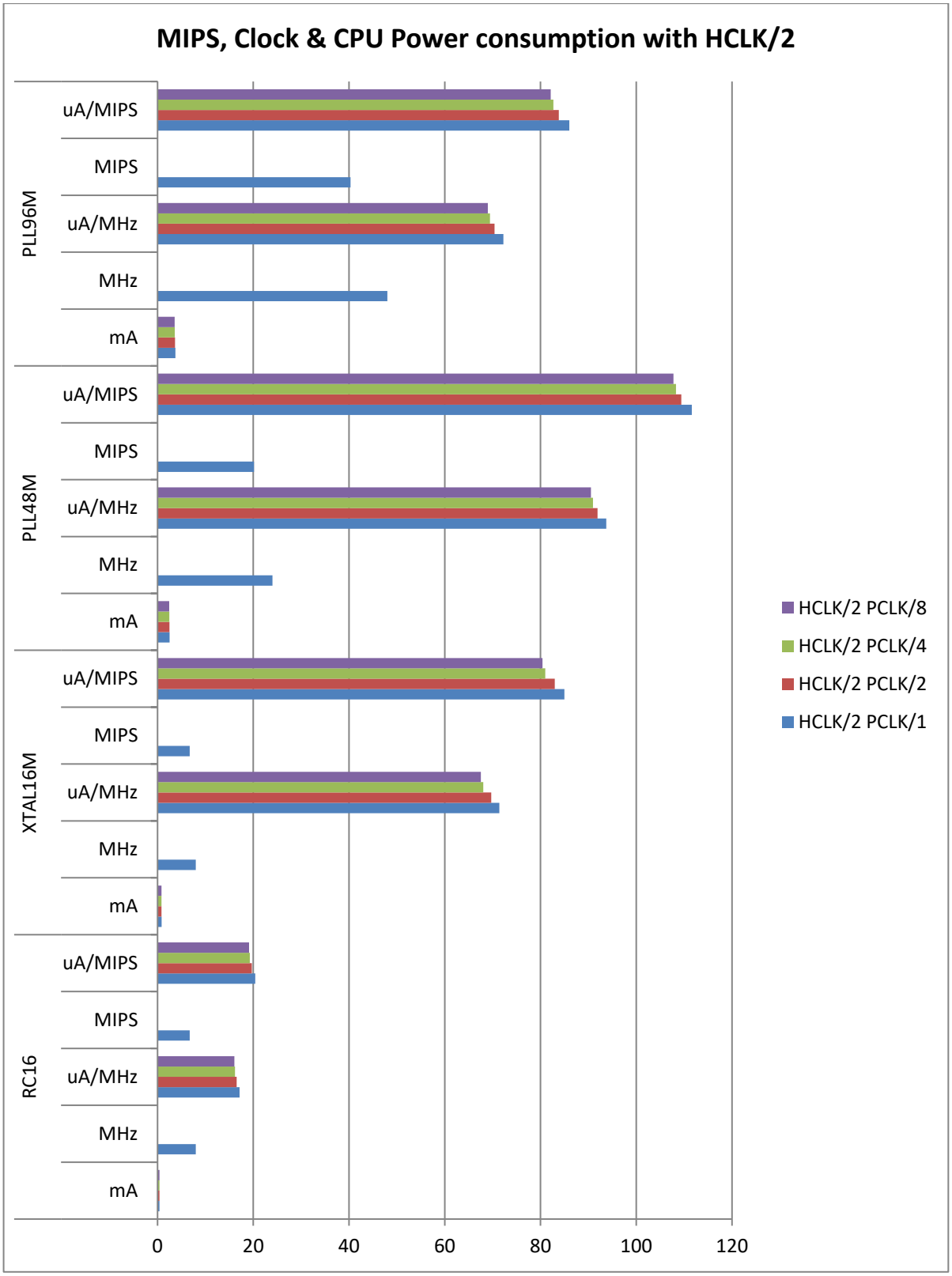

Figure 8: MIPS, Clock & CPU Power consumption with HCLK/2

Table 5: MIPS, Clock & CPU Power consumption with HCLK/4

		HCLK/4			
		PCLK/1	**PCLK/2**	**PCLK/4**	**PCLK/8**
RC16	*mA*	0.355	0.353	0.352	0.351
	MHz	4			
	uA/MHz	20.357	19.799	19.499	19.299
	MIPS	3.36			
	uA/MIPS	24.234	23.570	23.213	22.975
XTAL16M	*mA*	0.696	0.688	0.683	0.680
	MHz	4			
	uA/MHz	105.623	103.572	102.249	101.549
	MIPS	3.36			
	uA/MIPS	125.742	123.300	121.725	120.892
PLL48M	*mA*	1.989	1.967	1.956	1.951
	MHz	12			
	uA/MHz	142.974	141.074	140.189	139.796
	MIPS	10.08			
	uA/MIPS	170.207	167.945	166.892	166.424
PLL96M	*mA*	2.671	2.627	2.604	2.593
	MHz	24			
	uA/MHz	99.881	98.036	97.077	96.639
	MIPS	20.16			
	uA/MIPS	118.906	116.710	115.567	115.046

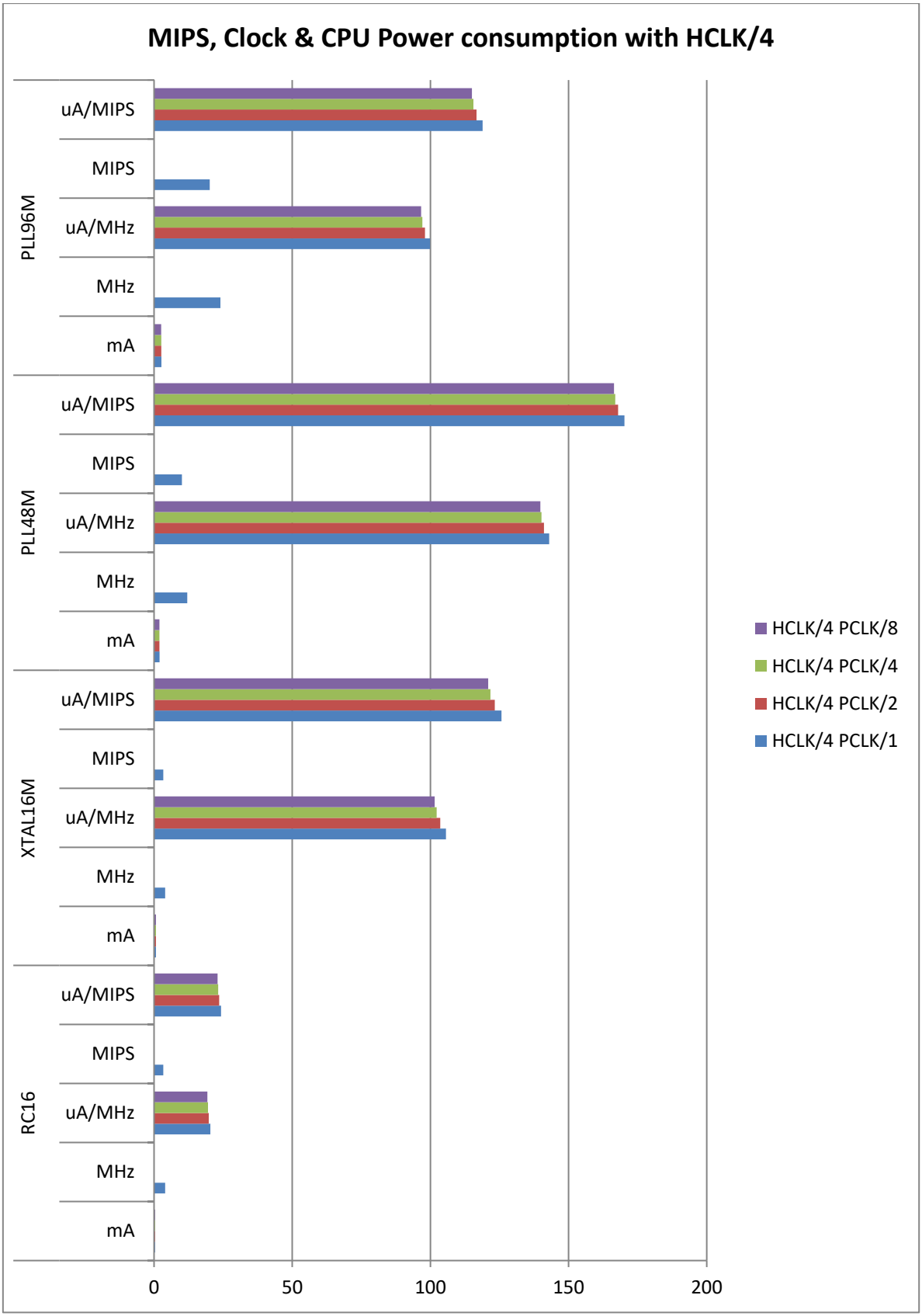

Figure 9: MIPS, Clock & CPU Power consumption with HCLK/4

Table 6: MIPS, Clock & CPU Power consumption with HCLK/8

		HCLK/8			
		PCLK/1	**PCLK/2**	**PCLK/4**	**PCLK/8**
RC16	*mA*	0.328	0.328	0.326	0.326
	MHz	2			
	uA/MHz	27.129	27.223	26.203	25.986
	MIPS	1.68			
	uA/MIPS	32.296	32.408	31.194	30.935
XTAL16M	*mA*	0.592	0.588	0.586	0.585
	MHz	2			
	uA/MHz	158.972	156.945	155.905	155.445
	MIPS	1.68			
	uA/MIPS	189.252	186.839	185.601	185.053
PLL48M	*mA*	1.721	1.711	1.705	1.703
	MHz	6			
	uA/MHz	241.246	239.484	238.614	238.155
	MIPS	5.04			
	uA/MIPS	287.198	285.100	284.065	283.518
PLL96M	*mA*	2.142	2.122	2.112	2.105
	MHz	12			
	uA/MHz	155.689	154.019	153.146	152.635
	MIPS	10.08			
	uA/MIPS	185.344	183.356	182.317	181.709

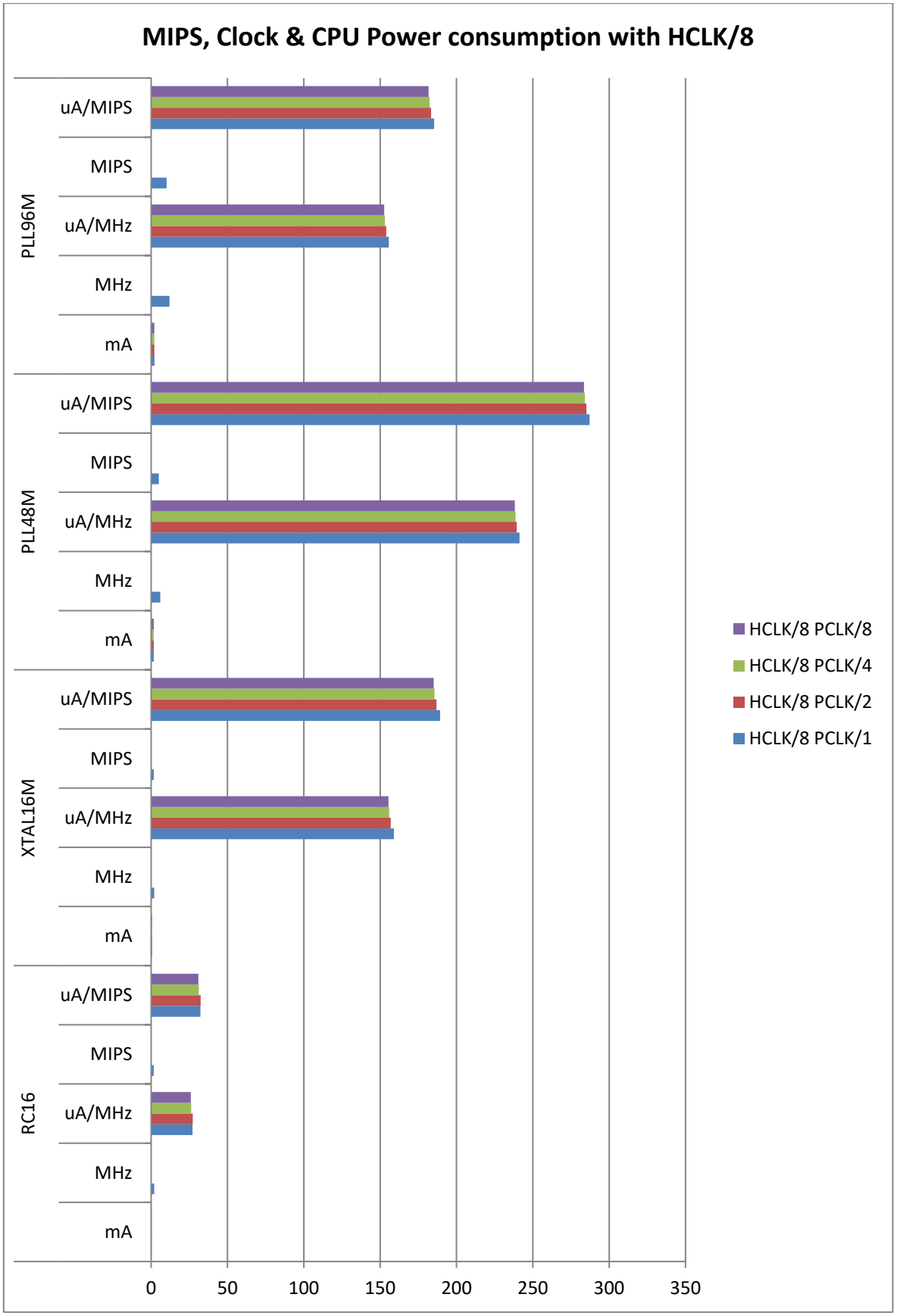

Figure 10: MIPS, Clock & CPU Power consumption with HCLK/8

5 Useful tips

Each product and each application code is unique. Having said that the reader should realize that there is no single rule to achieve the optimal power consumption for all products, all applications and all configurations.

However there are some key factors to help choose the most appropriate resource to use for the product design.

Retention RAM, CACHE, QSPI FLASH, CLOCKS

Retention RAM, CACHE and QSPI FLASH use are very tightly interconnected between them. Some common questions and their answers about this are:

1) How much retention RAM to use?

   There is not a golden rule here. The use case can even change this number. In general keep in the RAM the data, literals and code which is used very often and the access to QSPI FLASH will result in higher power consumption and slight degradation of the overall performance. In general the contribution of the retained RAM is small overall to the system and it is the last thing one should optimize when the application code is final and knows that there is some RAM not needed to be retained in sleep.

   The default setting for most in the SDK example projects is to have all RAM retained. For the ble_adv which is used for the measurements the default settings are already optimized.

   The power budget for each block of retained RAM is shown in Table 7.

2) When is it better to increase the use of retention RAM?

   When there are pieces of the code critical for the execution (e.g. ISR sensitive to delays), or used all the time (e.g. a timer with very short period) then these pieces of the code might me better to put them in retention RAM.

   Also if there are constants and lookup tables which the code has to use often, then these might be a good idea also to keep them in the retention RAM.

3) Should I bypass the CACHE?

   No, never. Although it is possible to put the CACHE in bypass mode and have the code running directly from FLASH, the chip is designed to work with CACHE. Running code without cache results to 15-20 times slower code execution due to delays inserted by the QSPI FLASH speed. This reflects to higher power consumption and slower code execution.

4) Use Power Down modes for the QSPI FLASH.

   This will minimize the power consumption on the QSPI FLASH when not used. Just set the macro in the custom_config_qspi.h to (1).The Power Down mode is more efficient power mode for FLASH.

5) For cases where the system need to wake-up periodically and read a sensor for example, regardless of the BLE activity, it is possible to choose either to wait for the 16MHz XTAL to settle on wakeup or continue with the current clock in use. The choice is easy to make by selecting the wakeup mode like this:

   `pm_set_wakeup_mode(true);` → `wait for the XTAL16M`

   `pm_set_wakeup_mode(false);` → `Do not wait for the XTAL16M`

6) The lowest power save mode for the unused GPIOs is Input Pull-Down. This is the default state of the GPIOs on power up. Select this state for the GPIOs when not in use if possible to minimize leakages and unnecessary power consumption.

Table 7: Retained RAM power budget per cell

RAM cell	Deep Sleep Mode (No clock)	Extended Sleep Mode (RCX)
RAM1 (8 kB)	190	260
RAM2 (24 kB)	431	543
RAM3 (32 kB)	570	698
RAM4 (32 kB)	623	714
RAM5 (32 kB)	790	980

Note 1 Measured on DA14680-AD silicon. The values are in nA

Extended Sleep or Hibernation modes

Extended Sleep and Hibernation power-save modes refer to the system blocks of DA1468x, not the BLE block. The BLE block power saves independently of the rest of the system even if the system is active.

For normal operation the mode of interest is the Extended Sleep.

In Extended sleep the BLE is available and the rest of the system is sleeping to lower the power consumption. The system wakes periodically and runs scheduled tasks or wakes on interrupt (BLE or external HW source) and runs the ISR.

Hibernation mode is a special mode to be used for shipping the final product to market without draining the battery. This is what is used for shipping and storing the final product.

Low Power (LP) Clocks: RCX, XTAL32K, RC32K

There are three LP clocks available. The RC32K is not an accurate clock and mainly is used when waking from clockless sleep modes like Deep Sleep to run the HW FSM. RC32K is the only clock which will be enabled upon event while in Deep Sleep and the only running clock if waking from VBUS interrupt (#define dg_configLOW_VBAT_HANDLING 1).

RCX and XTAL32K are accurate clocks to be used with extended sleep modes where we need to have accuracy on wake-sleep cycles in order to serve on time the BLE operations upon wakeup. The provided SDK is taking care of the clock use and the developer should just select the one to use in the custom_config_qspi.h or custom_config_ram.h files.

RCX Accuracy is less than 200ppm.

Fast clocks: RC16M, XTAL16M, PLL48MHz and PLL96MHz

RC16M is a fast internal clock, but not very accurate. Is used primarily by the DCDC, and the ROM-Booter to start the device. If it is disabled the DCDC cannot be used.

The XTAL16M is an accurate clock dependent on the external 16MHz crystal. It is used for normal operation with the BLE protocol enabled.

PLL48 and PLL96 are PLL clocks internally generated. They require the XTAL16M clock and they offer higher performance but higher power consumption too.

Avoid current leakage on peripheral GPIOs

Make sure that all the unused peripheral GPIOs are configured as input-pulldown which is the default power-up mode for the GPIOs.

Revision history

Revision	Date	Description
1.0	18-May-2016	Initial version.
1.1	25-May-2016	Added note for suggested sampling rate in chapter 4.2
1.2	20-Jul-2016	Updated measurements for DA14681-01 silicon Updated data logger screenshots Added explanations for the measurements in Table 1
1.3	29-Sept-2016	Updates on advertising and connection measurements. Added tips for saving power.
1.4	4-Apr-2017	Updated for SDK1.0.8
1.5	19-Mar-2018	Correction of typos with respect to decimal numbers.
1.6	18-Jan-2022	Updated logo, disclaimer, copyright.

Status Definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

RoHS Compliance

Dialog Semiconductor's suppliers certify that its products are in compliance with the requirements of Directive 2011/65/EU of the European Parliament on the restriction of the use of certain hazardous substances in electrical and electronic equipment. RoHS certificates from our suppliers are available on request.