University of Waterloo
Faculty of Engineering

Current Minimization Caching Techniques for LTE Video Streaming

Qualcomm Innovation Center, Inc.
San Diego, CA 92121-1714

Prepared by
Ziyad Mir
ID 20333385
2A Department of Systems Design Engineering
September 14, 2011

400 Joseph Street
Port Elgin, Ontario
N0H 2C2

September 14, 2011

Paul Fieguth, Chair
Department of Systems Design Engineering
University of Waterloo
Waterloo, Ontario
N2L 3G1

Dear Professor Fieguth,

I have prepared this report, "Current Minimization Caching Techniques for LTE Video Streaming," as my 2A Work Report for the Linux Android Power team at Qualcomm Innovation Center. This report is the second of four that I must submit as part of my degree requirements, and it has not received any previous academic credit. This report was entirely written by me and has not received any previous academic credit at this or any other institution.

Qualcomm Innovation Center, is a wholly owned subsidiary of Qualcomm, focused on open source and community based software initiatives. Within Qualcomm Innovation Center, the Linux Android Power team, of which I was a member, is responsible for the power management of mobile devices that run the Android mobile operating system.

This report explores the effects that cache modifications have on the consumption of electric current within mobile devices, and aims to determine cache optimizations which reduce average current consumption, over the duration of a video clip.

I would like to thank Sandeep Singhai, Engineering Director, Jun Zhang, Senior Staff Engineer and Harry Yang, Senior Engineer, for their guidance throughout my internship. I hereby confirm that I have received no further help, other than what is mentioned above, in writing this report.

Sincerely,

Ziyad Mir
ID 20333385

Abstract

The purpose of this report is to study cache size changes on mobile devices, in order to reduce the current consumption while streaming videos, over LTE networks.

Within this report, the significance of mobile energy conservation and current consumption, is covered. Introductory information regarding mobile video streaming, Long Term Evolution networks and the Android mobile operating system, is provided. A mathematical model is used to understand the current caching architecture and system behaviour. Cache size variations which minimize average current consumption are studied and their associated realizations are understood, tested, and analyzed. Optimal values for cache sizes are derived, and recommendations for future studies are suggested.

Table of Contents

List of Tables

List of Figures

1.0 Introduction

This report documents an exploratory study into the effects of cache modifications on the consumption of electric current within mobile devices. It culminates five months of study into the behaviour of mobile video caching and current consumption.

1.1 Background

Qualcomm is an American, wireless telecommunications research and development center and fabrication-less chip supplier [1]. Qualcomm Innovation Center (QuIC), is a wholly owned subsidiary of Qualcomm, focused on open source and community based software initiatives. Within QuIC, the Linux Android Power team is responsible for the power management of mobile devices that run the Android mobile operating system.

Chipset design and development serve as integral aspects of Qualcomm's business. A selection of embedded systems designed and developed at Qualcomm can be classified as System-on-a-Chip (SoC) technologies. A SoC combines electronic and computer system components into a single, integrated circuit, the high-end family of which, is known as Snapdragon. The Snapdragon SoC includes an application processor; a central processing unit which serves as the primary hardware component responsible for non-kernel related data processing.

1.2 Motivation and Significance

Android is an open-sourced, mobile software stack which includes an operating system, middleware and applications [2]. Within the last year, technological research and advisory firms have provided statistics indicating that between 33 and 54 percent of smartphones sold, worldwide, were powered using Android [3, 4, 5, 6, 7]. Qualcomm has broadened its product scope to include support for chipsets running Android as a result of Android's emergence as the platform of choice for many original equipment manufacturers. In addition, video streaming has emerged as a popular medium of data consumption, with video streaming websites, such as www.youtube.com, receiving over three billion views per day [8]. Finally, with the recent deployment and high data rate capabilities of Long Term Evolution (LTE) networks, the popularity of video streaming as a data consumption medium has been projected to increase [9, 10]. Collectively, these factors have motivated QuIC to focus software engineering efforts on enabling and optimizing Android video streaming over LTE networks.

Mobile power management refers to the engineering efforts related to the conservation of energy on mobile devices. The embedded systems designed and developed at Qualcomm have removable batteries, which serve as energy storage components. The energy stored in a mobile device's battery is depleted as it draws current, using it to perform various functions. A mobile device cannot function when its energy supplies are fully depleted, and its functionality can be significantly hindered as a result of low energy supplies. Naturally, the conservation of energy arises as a critical factor in determining the commercial success of a mobile device.

1.3 Video Streaming, Caching, and Long Term Evolution

The application processor has many computational duties, including the processing needed to stream, cache, and play videos. These videos are delivered to mobile devices using streaming technologies, which function on the premise that data can be simultaneously delivered to, and consumed by, a device. The temporal relationship inherent to both audio and video data allows streaming to suffice as a data delivery method. For example, a video clip's initial frames can be played while successive frames are streamed to the device, without performance drawbacks.

A cache is a collection of data-duplicating information. Generally, a cache is used to reduce the necessity for reacquisition of data, or to build up stores of data for quick, future access. Android uses video caching to store data on mobile devices after video requests have been made. The cache uses a single memory unit, or buffer. As data is streamed to the device, the buffer is filled, and as data is consumed by the device, the buffer is depleted. Low and high cache threshold levels, called $lowWaterThreshold$ and $highWaterThreshold$, are variables which govern the system's caching behaviour.

Long Term Evolution (LTE) is an optimized wireless connectivity solution, standardized by the 3rd Generation Partnership Project (3GPP), providing improved data rates over current 3G technologies [11]. Deployments of LTE networks, in the United States, began in late 2010.

1.4 Problem Description

This report aims to determine the values of $lowWaterThreshold$ and $highWaterThreshold$ which minimize I_T, the average current consumed by a mobile device, over the duration of a t-second long video clip. If multiple values of $lowWaterThreshold$ and $highWaterThreshold$ yield the same average current consumption, the pairwise combination which minimizes $CacheSize$ should be selected.

The scope of this report is limited, strictly, to cache threshold level changes and the impact of those changes on a mobile device's average current consumption and memory usage. The purpose of restricting the domain of interest is two-fold: first, it serves to reduce the overhead of the project, enabling a complete study to be conducted within the allotted time frame; and second, it provides a suitable reference point from which further studies can be based.

1.5 Approach and Evaluation

In order to determine the values of $lowWaterThreshold$ and $highWaterThreshold$ which minimize the average current consumption, over the duration of a video clip, a representative sample of values will be generated, tested, and the results analyzed. The values will be analyzed with respect to the impact they have on average current consumption and memory usage. The test results will indicate which values of $lowWaterThreshold$ and $highWaterThreshold$ minimize I_T, without wasting memory, and the optimal threshold levels will be reported as results.

2.0 Design

In this section, it is assumed that all multimedia is streamed to mobile devices, powered by Qualcomm application processors, running the Android operating system, over LTE networks. Furthermore, it is assumed that a single video clip will be selected and subjected to a set of test cases, each with differing values of $lowWaterThreshold$ and $highWaterThreshold$. This section also includes a discussion of the test and system caching designs.

2.1 Video Caching Architecture

Android currently utilizes the Stagefright framework as a multimedia playback engine. One of the main entities within the Stagefright framework is AwesomePlayer, a component used to coordinate the connection of Stagefright's modules with the Android Media Framework [12]. Within the Stagefright framework, certain modules are responsible for data retrieval, encoding, decoding, rendering, and caching. The caching modules make extensive use of variables called $lowWaterThreshold$ and $highWaterThreshold$, which govern caching behaviour. The significance of the two variables is apparent after observing the typical data flow arising while streaming a video clip.

With the video data buffer initially empty, AwesomePlayer receives a message to begin video streaming. AwesomePlayer establishes a network connection between a mobile device requesting the video data, and the web server hosting the video data. It then checks whether the buffer contains data of cumulative size less than the value of $lowWaterThreshold$, and if so, requests video data. As this data is received, it is stored in the buffer, and checked against the value of $highWaterThreshold$. Video data requests continue until the buffer contains data of cumulatize size greater than or equal to the value of $highWaterThreshold$, at which point, the network connection is terminated. The video stored as data in the buffer is depleted while being played, until the buffer contains data of cumulative size less than the value of $lowWaterThreshold$. Once the buffer holds data of cumulative size less than the value of $lowWaterThreshold$, a network connection is reestablished, and the above procedure begins anew.

$CacheSize$, defined as $CacheSize = highWaterThreshold - lowWaterThreshold$, serves as a measure of the size of the buffer. $CacheSize$ represents the amount of data, in bytes, that, beginning at the termination of the network connection, can be depleted from the video data buffer, before the network connection is reestablished.

2.2 Processing States

For the context in which this study is being conducted, the application processor can assume three main states: a maximum current-consumption, active state; a high current-consumption, dormant state; and a low current-consumption, sleep state. The active state corresponds to all cases in which a network connection is live and a mobile device downloading, the dormant state corresponds to all cases in which a network connection is live and a mobile device not downloading, and the sleep state corresponds to all other cases.

2.3 Cache Threshold Values

Based on recommendations from senior members of the Linux Android Power team, a representative sample of values was selected for $lowWaterThreshold$, $highWaterThreshold$ and their associated $CacheSize$ values, as outlined in Table 1. These values are assumed to provide an accurate representation of the possible pairwise combinations of $lowWaterThreshold$

and $highWaterThreshold$.

Table 1: lowWaterThreshold, highWaterThreshold and CacheSize Values

Test Case Number	lowWaterThreshold	highWaterThreshold	CacheSize
1	524,288	20,971,520	20,447,232
2	524,288	31,457,280	30,932,992
3	524,288	41,943,040	41,418,752
4	524,288	52,428,800	51,904,512
5	524,288	62,914,560	62,390,272
6	524,288	73,400,320	72,876,032

2.4 System Design

By modelling the system caching mechanics, the effects on caching behaviour as a result of variations to $lowWaterThreshold$ and $highWaterThreshold$, can be understood.

Let R_i represent the rate at which data enters the cache (bytes/second).

Let R_o represent the rate at which data leaves the cache (bytes/second).

Let $\Delta R = R_i$ - R_o (bytes/second).

Let t_d represent the time taken for one dormancy period (seconds).

Let $t_{i,l}$ represent the time taken for $lowWaterThreshold$ bytes to enter the cache (seconds).

Let $t_{i,c}$ represent the time taken for $CacheSize$ bytes to enter the cache (seconds).

Let $t_{o,c}$ represent the time taken for $CacheSize$ bytes to leave the cache (seconds).

Assuming:

1. $\Delta R \neq 0$

2. Video playback begins immediately, once there is data in the cache.

3. Establishing and terminating a network connection takes a negligible amount of time.

For a video clip, of length z seconds, and size b bytes

$$z = t_{i,l} + k + j + n \cdot (t_{i,c} + t_{o,c}) + (n+1) \cdot t_d \tag{1}$$

for

$$n, z \geq 0 \tag{2}$$

where

$$-t_{i,l} \leq k \leq t_{i,c} \tag{3}$$

$$y(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & otherwise \end{cases}$$

and

$$j = \frac{lowWaterThreshold + k \cdot \Delta R}{R_o} \tag{4}$$

Analyzing the components of equation (1) provides insight into the system caching behaviour.

Generally, a mobile device:

1. Downloads $lowWaterThreshold$ bytes of data in $t_{i,l} = lowWaterThreshold/\Delta R$ sec-

onds

2. Downloads $CacheSize$ bytes of data in $t_{i,c} = CacheSize/\Delta R$ seconds, n times

3. Remains dormant for $(n+1) \cdot t_d$ seconds

4. Depletes $CacheSize$ bytes of data in $t_{i,c} = CacheSize/\Delta R$ seconds, n times

5. Downloads $k \cdot \Delta R$ bytes of data in k seconds

6. Depletes $lowWaterThreshold + k \cdot \Delta R$ bytes of data in j seconds

Here, $z = t_{i,l}+k+j+n\cdot(t_{i,c}+t_{o,c}) + (n+1)\cdot t_d$. $t_{i,l}$ represents the time taken to download $lowWaterThreshold$ bytes of data, k represents the time taken to download the final $k \cdot \Delta R$ bytes of data, and j represents the time taken to deplete the final $lowWaterThreshold{+}k{\cdot}\Delta R$ bytes of data. Between the initial $lowWaterThreshold$ bytes of data and the final k bytes of data that are downloaded, there are n downloads of $CacheSize$ bytes of data, n depletions of $CacheSize$ bytes of data, and $n+1$ dormancy periods of t_d seconds, each.

Each component of z can be grouped with its associated current consumption state: active, dormant or sleep. Furthermore, the average current consumption, over the duration of a video clip, I_T, can be computed as the sum of the cumulative current consumption during each of the current consumption states, divided by the total time spent in each of the states. That is,

Let T_a represent the time spent in the active current consumption state (seconds).
Let T_d represent the time spent in the dormant current consumption state (seconds).
Let T_s represent the time spent in the sleep current consumption state (seconds).
Let I_a represent the current drawn during the active current consumption state (Amperes).

Let I_d represent the current drawn during the dormant current consumption state (Amperes).

Let I_s represent the current drawn during the sleep current consumption state (Amperes).

Then,

$$I_T = \frac{T_a \cdot I_a + T_d \cdot I_d + T_s \cdot I_s}{T_a + T_d + T_s} \tag{5}$$

A mobile device must spend exactly $t_{i,b} = b/R_o$ seconds in the active state, downloading b bytes of data. However, it also spends $(n + 1) \cdot t_d$ seconds in the dormant state, and $z - t_{i,b} - (n + 1) \cdot t_d - j$ seconds in the sleep state. $T_a = t_{i,b}$, which is constant for a given video clip. Furthermore, in the current system design, $I_d \gg I_s$, and consequently, the largest savings, in regards to minimizing I_T, are seen by minimizing T_d. By selecting $n = 0$, the time spent in the dormant state is minimized, and consequently, the average current consumption, I_T, is also minimized.

2.5 Test Procedure

Using the combinations of low and high cache threshold levels, as outlined in Table 1, each cache threshold pair will be subjected to the same test case, and the average current consumption, I_T, will be measured and recorded. Through empirical measurement, it will be determined how I_T varies as a result of variations to $CacheSize$, $lowWaterThreshold$ and $highWaterThreshold$.

In order to minimize external variation between test cases, a consistent testing environment must be specified. For the waveform capture procedure, the constraints are as follows.

1. The same video clip will be used for all test cases.

2. All tests will be streamed through the YouTube Android application.

3. All tests will be streamed over QuIC's internal LTE network.

4. All tests will be streamed on the same mobile testing device.

5. The volume level will be set to half the allowed maximum.

6. The brightness level will be set to half the allowed maximum.

The waveform capture tests measure system-level current consumption. The tests provide high-level battery consumption measurements; the average current consumed by the device over the duration of a video clip. Having system-level current consumption measurements provide clear insight into the effects of individual component changes on the average current consumption of the device.

2.6 Test Evaluation

Each test will be evaluated based upon two criteria: first, minimizing I_T; and second, minimizing $CacheSize$.

Clearly, the main intent of the study is to determine the values of $lowWaterThreshold$ and $highWaterThreshold$, which minimize I_T. Through analyzing equation (1), it is evident that the minimum average current consumption, I_T, is reached when $n = 0$. However, setting $n = 0$, in equation (1), does not yield unique values of $CacheSize$. Any cache size larger than $CacheSize = highWaterThreshold - lowWaterThreshold$, when $n = 0$, will yield the minimum average current consumption, for a given video clip. It will also, however, use additional, unnecessary memory. Thus, for sets of $CacheSize$ values which yield the same average current consumption, I_T, the minimum $CacheSize$ value will be selected.

3.0 Analysis

In this section, the current consumption test results will be analyzed and evaluated, based on the two aforementioned criteria. Furthermore, the observed test results will be compared with the expected behaviour, and the optimal values of $lowWaterThreshold$ and $highWaterThreshold$ will be determined.

3.1 Observed System Behaviour

In order to determine whether the observed system behaviour reflects the expected system behaviour, two main trends should be considered while studying the test results: first, the average current consumption, I_T, should be a non-increasing function of the value of $CacheSize$; and second, smaller values of $CacheSize$ should have less lengthy, but more frequent, periods of time spent in the sleep current consumption state.

In order to determine whether the average current consumption, I_T, tends not to increase as the value of $CacheSize$ increases, the various $CacheSize$ values can be plotted against their associated average current consumption values. Some variance, with respect to the expected caching behaviour, is anticipated; this can largely be attributed to real-world data acquisition variations, which do not impact the overall result. For instance, adjacent data points might not accurately represent the expected system behaviour, however, the entire set of data points should, as a whole, largely reflect the expected behaviour. The pairwise combinations of cache threshold levels, as outlined in Table 1, and their associated average current consumptions, are outlined in Table 2.

The data from Tables 1 and 2 can be graphed, as in Figure 1, to explore the relationship between $CacheSize$ and I_T, the average current consumption.

Table 2: Average Current Consumption Values

Test Case Number	Average Current Consumption (milliamperes)
1	689.8306
2	578.5382
3	580.3882
4	580.1758
5	581.3250
6	581.2934

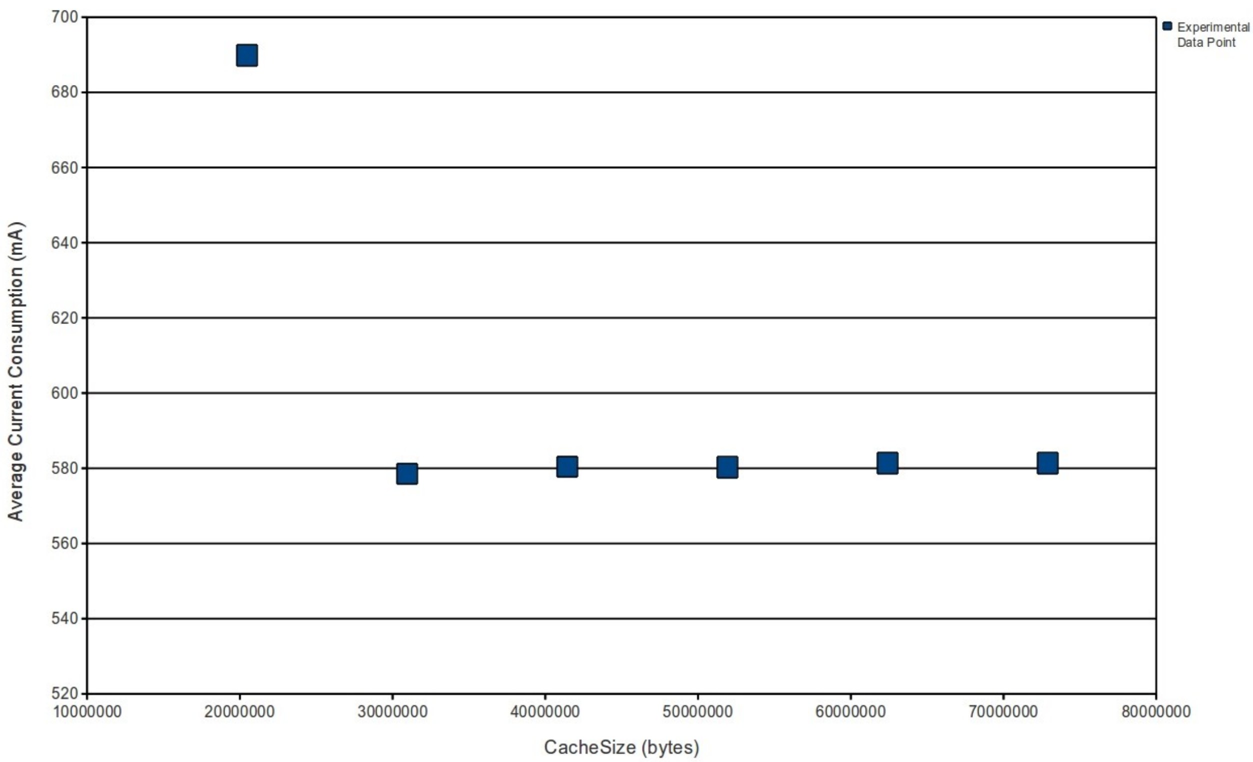

Figure 1: Current Consumption as a function of *CacheSize*

Graphically, it is clear that the observed system behaviour does, indeed, mimic the expected behaviour. That is, average current consumption is, in fact, a non-increasing function of *CacheSize*.

Next, by observing the waveforms captured for each of the testcases outlined in Table 1, the second trend can be verified. For the sake of brevity, three waveforms will be analyzed; those captured for tests 1, 2, and 3. These three waveforms are representative of the possible values $CacheSize$ can take. Supplementary waveforms, those for test cases 4, 5, and 6, are available in Appendix B.

In Test 1, $lowWaterThreshold$ has a value of $524,288$, $highWaterThreshold$ has a value of $20,971,520$, and $CacheSize$ has a value of $20,447,232$. From its waveform, as outlined in Figure 2, there are four periods, each of approximately four seconds in length, spent in the sleep state.

Figure 2: Current Consumption for $CacheSize = 20,447,232$

In Test 2, $lowWaterThreshold$ has a value of 524, 288, $highWaterThreshold$ has a value of 31, 457, 280, and $CacheSize$ has a value of 30, 932, 992. From its waveform, as outlined in Figure 3, there is one period, of approximately 50 seconds in length, spent in the sleep state.

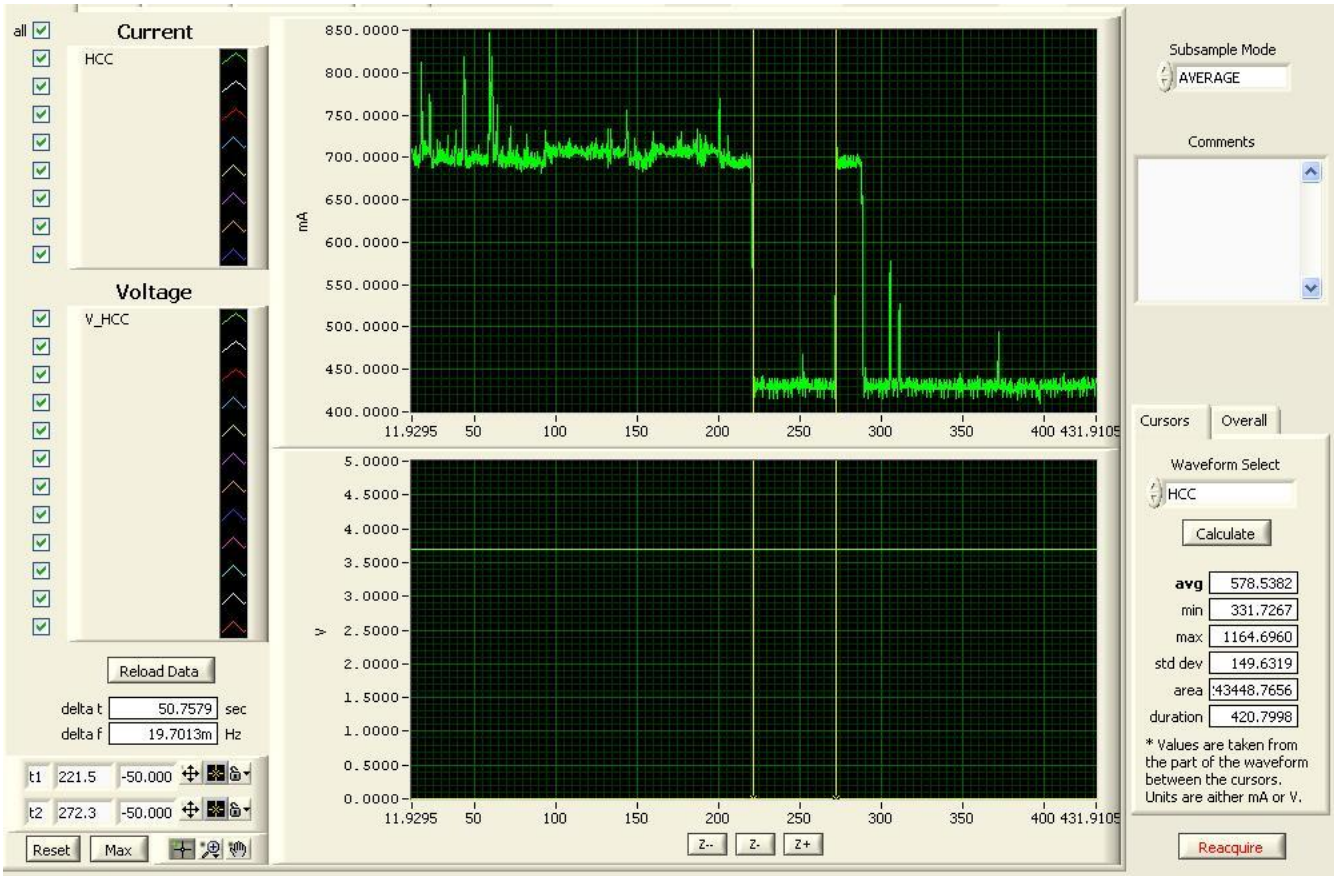

Figure 3: Current Consumption for $CacheSize = 30, 932, 992$

In Test 3, $lowWaterThreshold$ has a value of 524, 288, $highWaterThreshold$ has a value of 41, 943, 040, and $CacheSize$ has a value of 41, 418, 752. From its waveform, as outlined in Figure 4, there is one period, of approximately 50 seconds in length, spent in the sleep state.

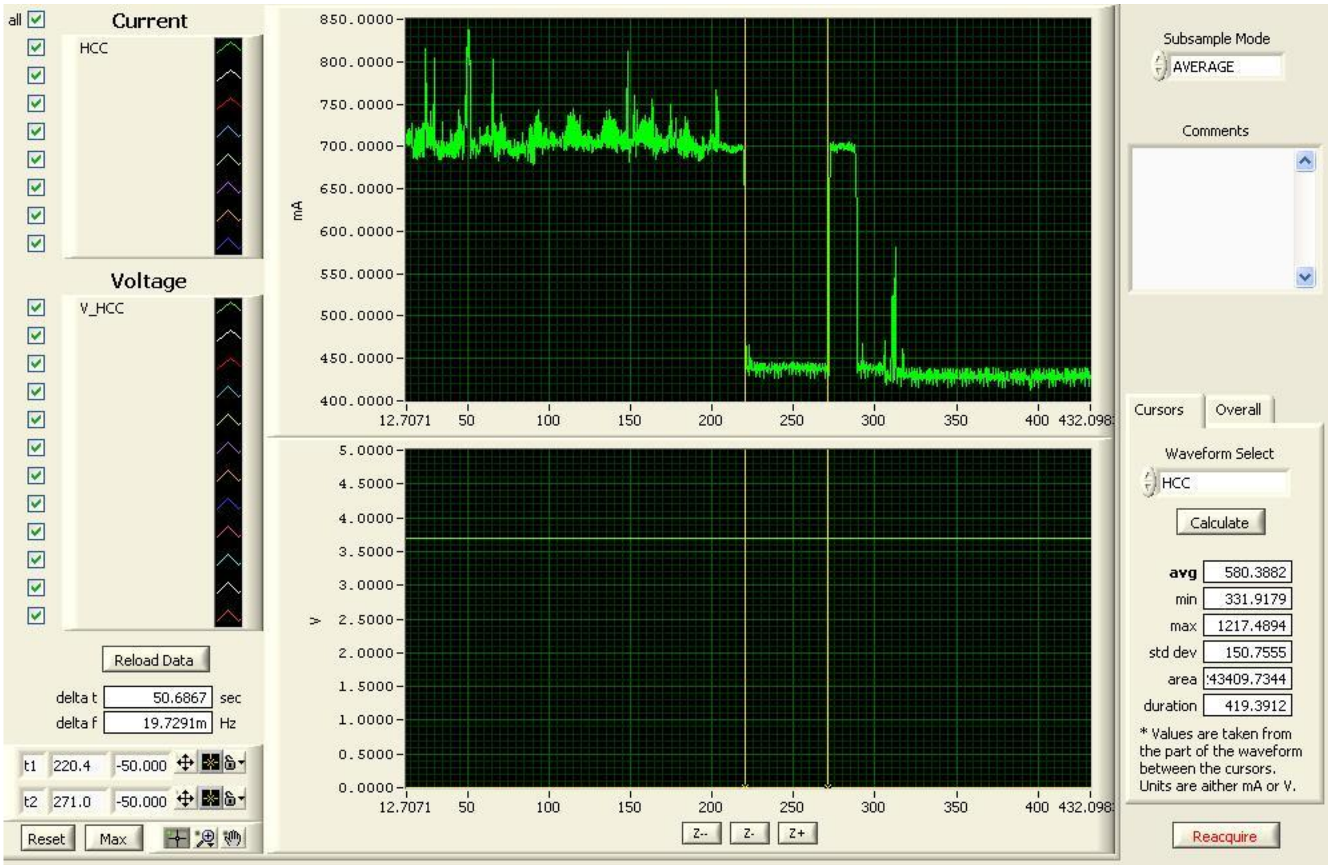

Figure 4: Current Consumption for $CacheSize = 41,418,752$

Graphically, it is clear that smaller values of $CacheSize$ result in less lengthy, but more frequent, periods of time spent in the sleep state. However, continuously increasing the value of $CacheSize$ does not guarantee an increase in T_S, the amount of time spent in the sleep state. This characteristic will be of interest when analyzing the memory usage associated with each $CacheSize$.

3.2 Minimizing Cumulative Current Consumption

Graphically, as observed in Figure 1, the average current consumption, I_T, is minimized for all values of $CacheSize$ greater than or equal to $30,932,992$, satisfying the first condition.

3.3 Minimizing Memory Usage

Since there are multiple $CacheSize$ values which minimize the value of I_T, the second constraint, namely, memory usage, needs to be considered. In order to determine the optimal value of $CacheSize$, the minimum value of $CacheSize$, which also minimizes the value of I_T, should be selected. Graphically, the optimal value of $CacheSize$ is $30,932,992$. Selecting this value for $CacheSize$ ensures that no additional memory is used, other than that which is necessary to minimize the average current consumption, I_T.

To determine the optimal values of $lowWaterThreshold$ and $highWaterThreshold$, consider the possible values each can assume. With the optimal value of $CacheSize$ determined, the only available option is to change the value of $lowWaterThreshold$, and consequently, $highWaterThreshold$.

Suppose that L_1 is the minimum possible value that $lowWaterThreshold$ can assume, and L_2 is the designated value of $lowWaterThreshold$, where $L_2 \geq L_1$. Clearly, I_T will be minimized and $highWaterThreshold = L_2 + CacheSize$. However, $L_2 - L_1$ bytes of extra memory will be used, without further reduction of I_T. Optimally, the designated value of $lowWaterThreshold$ should be the minimum allowed value it can assume; that is, $L_2 = L_1$. For the context of this study, $L_1 = 524,288$, and thus, the optimal values of $lowWaterThreshold$, $highWaterThreshold$ and $CacheSize$ are $524,288$, $31,457,280$ and $30,932,992$, respectively.

4.0 Conclusions

From the design and analysis conducted as part of the body of the report, the following conclusions can be drawn. First, it was shown that the observed system behaviour mimiced the expected behaviour, as derived from the system model. Specifically, the average current consumption, I_T, is a non-increasing function of the value of $CacheSize$. In addition, values of $CacheSize$ that are large enough to download the entire video clip, minimize I_T. As the values of $CacheSize$, which are not large enough to download the entire video clip, increase, the frequency of entering the sleep state tends to decrease, and the duration of the sleep state tends to increase. Furthermore, a value of $lowWaterThreshold$ larger than the minimum allowed value should not be used, as it introduces additional, unnecessary memory usage. By analyzing the captured waveforms, it was determined that cache size values equal to and larger than $CacheSize = 30,932,992$ minimize the average current consumption, and the value of $CacheSize$ which minimizes both average current consumption and memory usage, is $30,932,992$. Therefore, the optimal values of $lowWaterThreshold$ and $highWaterThreshold$ are $524,288$ and $31,457,280$, respectively.

5.0 Recommendations

Based on the direct conclusions of this report, it is recommended that $lowWaterThreshold$ and $highWaterThreshold$ takes values of $524,288$ and $31,457,280$, respectively.

Furthermore, it is recommended that a similar study be conducted for video clips of various lengths. The values of $lowWaterThreshold$, $highWaterThreshold$ and $CacheSize$, that are determined, should be recorded in a look-up table data structure. This data structure, in conjunction with simple logic within the threshold selection source code, should adaptively select the values of $lowWaterThreshold$ and $highWaterThreshold$, on a per video basis.

Appendix A - Glossary

1. **3GPP** stands for 3rd Generation Partnership Project, a collaborative group of telecommunications associations.

2. **LTE** stands for Long Term Evolution, an optimized wireless connectivity solution, standardized by the 3rd Generation Partnership Project, providing improved data rates over current 3G technologies.

3. **QuIC** stands for Qualcomm Innovation Center, a wholly owned subsidiary of Qualcomm, focused on open source and community based software initiatives.

4. **SoC** stands for System-on-a-Chip. A System-on-a-Chip combines electronic and computer system components into a single, integrated circuit.

Appendix B - Waveforms

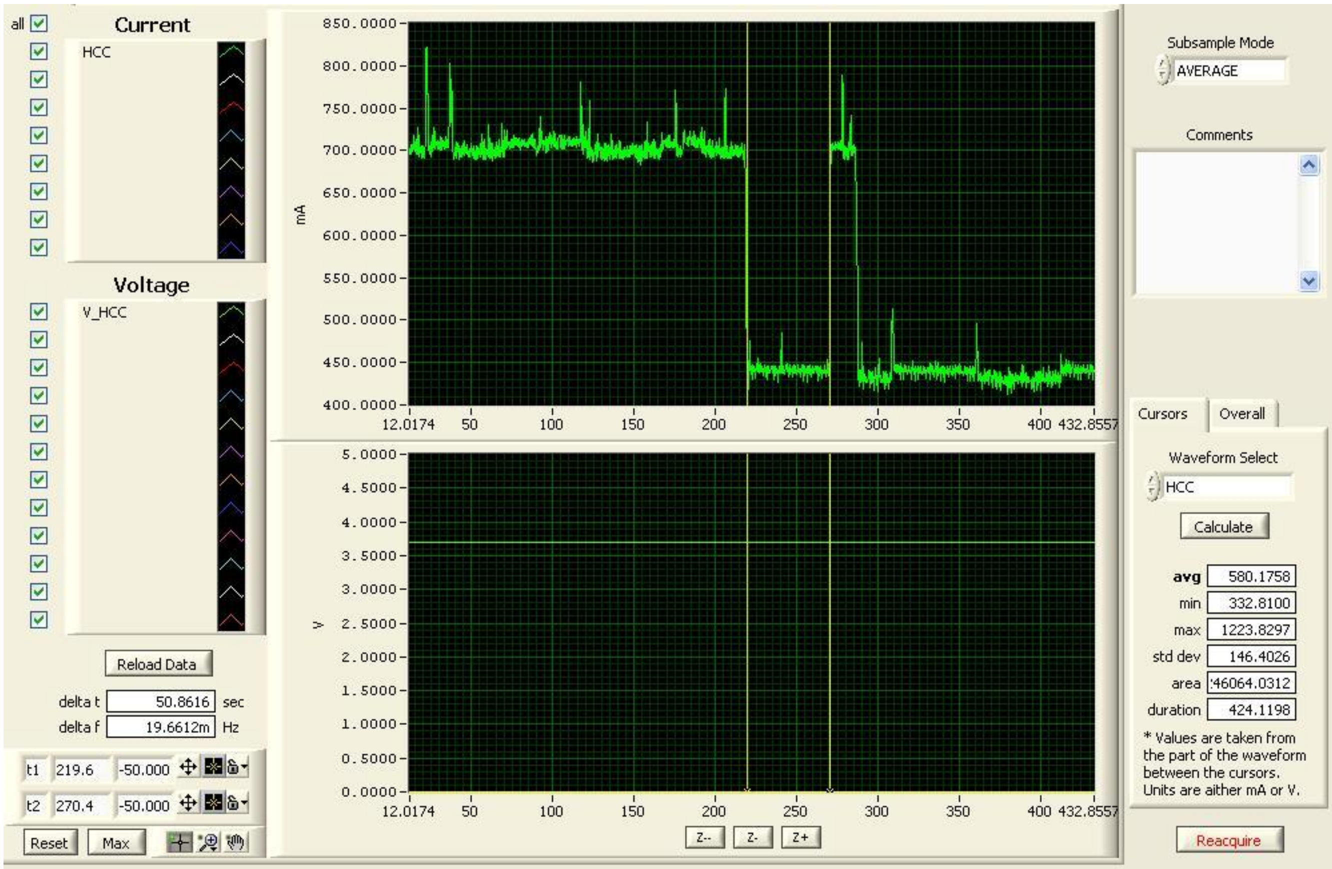

Figure 5: Current Consumption for $CacheSize = 51,904,512$

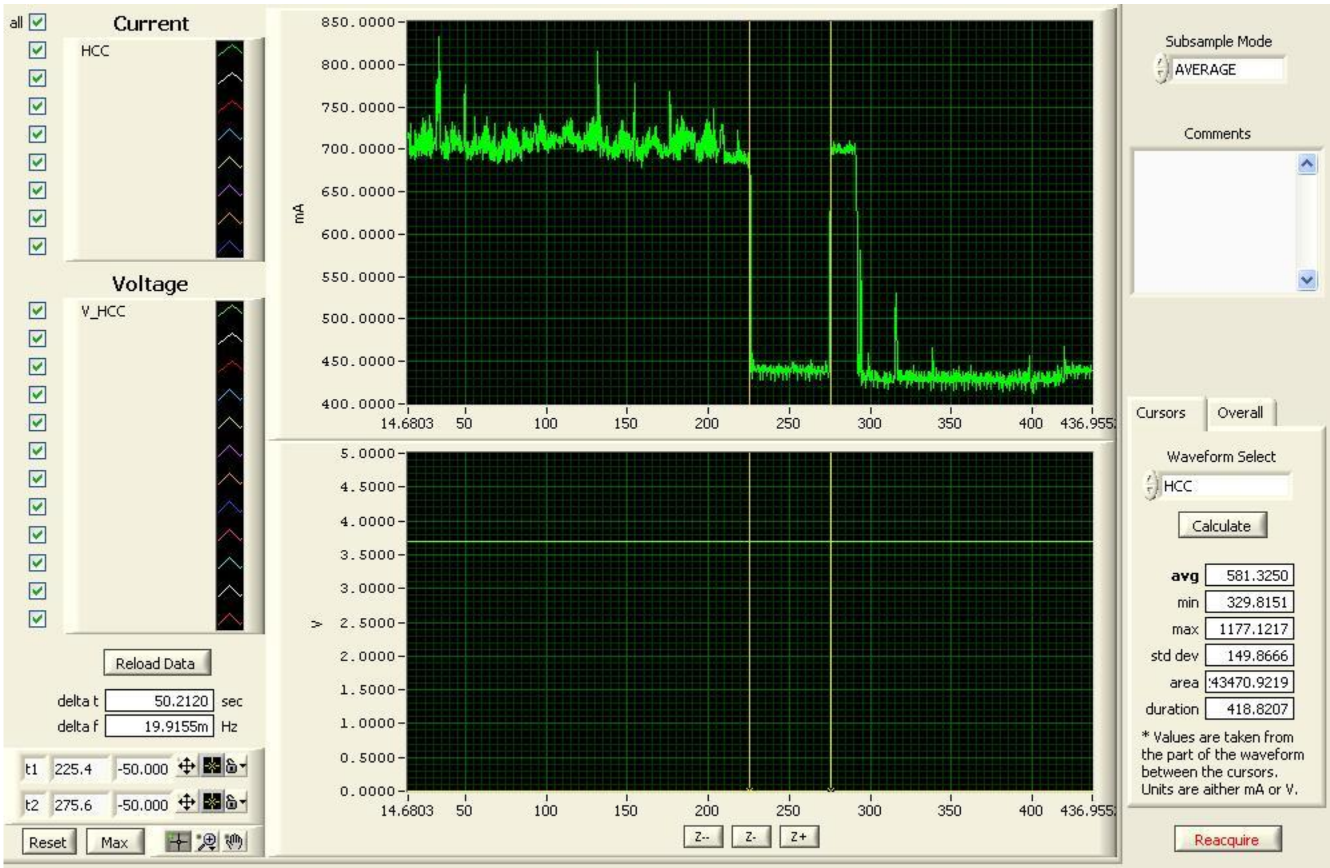

Figure 6: Current Consumption for $CacheSize = 62,390,272$

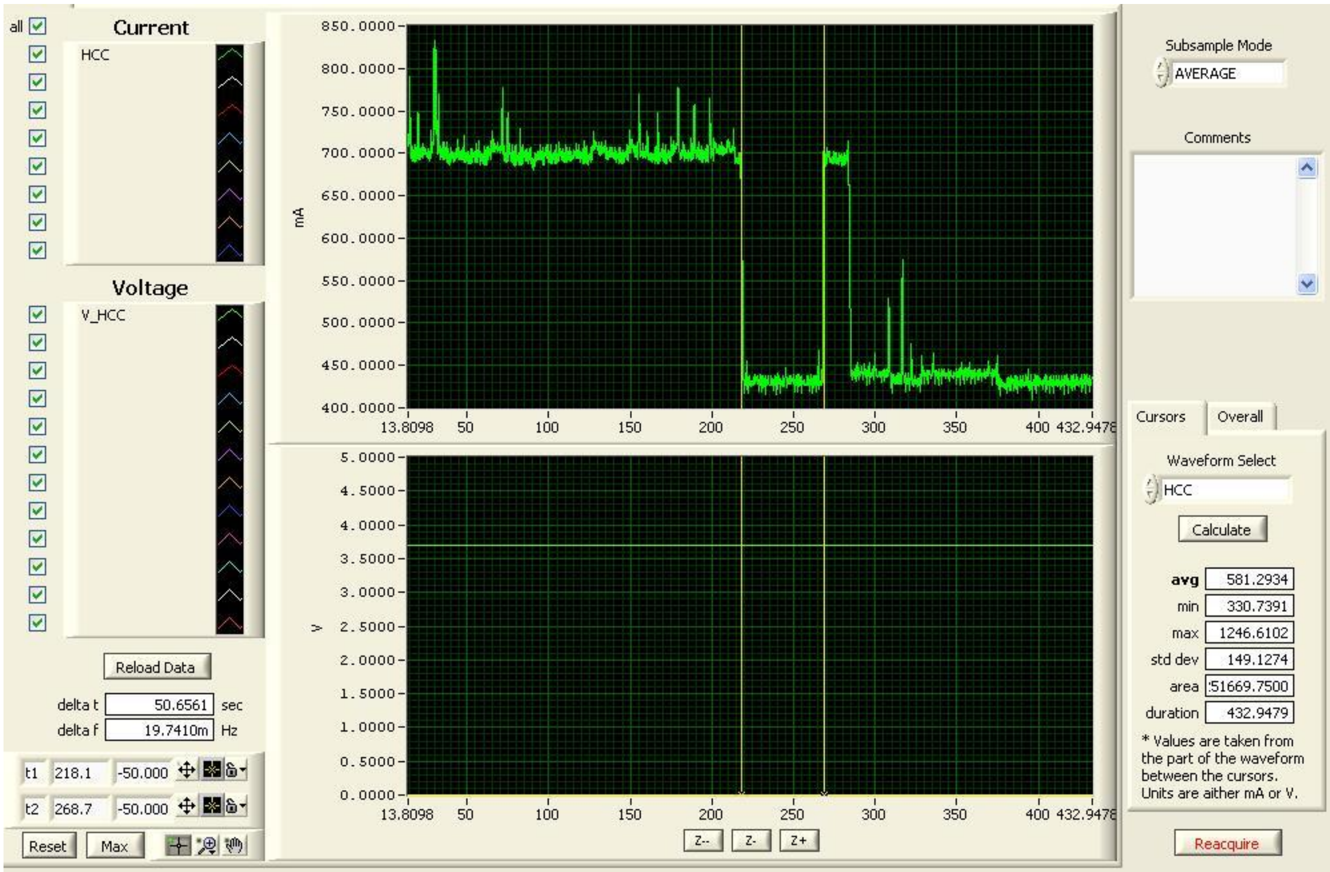

Figure 7: Current Consumption for $CacheSize = 72,876,032$

References

[1] Qualcomm, Incorporated. *Who We Are*, September 1st, 2011.
http://www.qualcomm.com/who-we-are/history

[2] Google, Incorporated. *What is Android?*, August 25th, 2011.
http://developer.android.com/guide/basics/what-is-android.html

[3] Gartner, Incorporated. *Gartner Says 428 Million Mobile Communication Devices Sold Worldwide in First Quarter 2011, a 19 Percent Increase Year-on-Year*, August 20th, 2011.
http://www.gartner.com/it/page.jsp?id=1689814

[4] Canalys. *Googles Android becomes the worlds leading smart phone platform*, July 30th, 2011.
http://www.canalys.com/newsroom/google%E2%80%99s-android-becomes-world%E2%80%99s-leading-smart-phone-platform

[5] The NPD Group, Incorporated. *Android Extends its Smartphone Market Share in the Third Quarter of 2010*, August 13th, 2011.
http://www.sfgate.com/cgi-bin/article.cgi?f=/g/a/2010/11/01/prwebprweb4726684.DTL

[6] Nielson Company. *In U.S. Smartphone Market, Android is Top Operating System, Apple is Top Manufacturer*, September 3rd, 2011.
http://blog.nielsen.com/nielsenwire/online_mobile/in-u-s-smartphone-market-android-is-top-operating-system-apple-is-top-manufacturer/

[7] comScore. *2 out of 5 Smartphone Subscribers Now Use Android Platform*, August 17th, 2011.

http://www.comscore.com/Press_Events/Press_Releases/2011/8/comScore_Reports_June_2011_U.S._Mobile_Subscriber_Market_Share

[8] Google, Incorporated. *Thanks, YouTube community, for two BIG gifts on our sixth birthday!*, August 20th, 2011.
http://googleblog.blogspot.com/2011/05/thanks-youtube-community-for-two-big.html

[9] Qualcomm, Incorporated. *LTE boosts data capacity with new, wider bandwidth spectrum*, August 5th, 2011.
http://www.qualcomm.com/products-services/wireless-networks/lte

[10] Netflix, Incorporated. *The Future Is Streaming*, August 11th, 2011.
http://gigaom.com/video/netflix-the-future-is-streaming/

[11] Qualcomm, Incorporated. *LTE - An Optimized OFDMA Solution for Wider Bandwidth Spectrum*, August 28th, 2011.
http://www.qualcomm.com/documents/files/lte-an-optimized-ofdma-solution-for-wider-bandwidth-spectrum.pdf

[12] Google, Incorporated. *Android 2.2 Platform Highlights*, August 8th, 2011.
http://developer.android.com/sdk/android-2.2-highlights.html