AN3012
Rev. 0, 10/2005

Statistical Considerations for Maximizing Channel Density on Embedded Systems

By Lúcio F. C. Pessoa

This application note provides an overview of statistical considerations for increasing average system usage of embedded applications with high channel density running on DSPs. This analysis applies to a large class of voice and video communications systems, which can typically tolerate some level of momentary degradation without significantly affecting the overall quality of the voice and video. An advantage of the statistical approach is that you can associate a level of confidence to a target channel density, with different channel densities depending on required levels of confidence. Experimental results at both the component and system levels support the underlying statistical analysis.

CONTENTS

© Freescale Semiconductor, Inc., 2005. All rights reserved.

1 Introduction

Many voice and video applications can tolerate certain losses of data without significant effects on quality of service (QoS) [1]. Therefore, statistical properties of traffic sources can be exploited to achieve a statistical multiplexing gain and increase average system usage. Allocating bandwidth in a multi-user communication system is a scheduling problem very similar to the problem of allocating processing time for individual components running on a DSP real-time application. For example, [2] in **Section 5**, *References*, on page 11 presents a statistical scheduling approach for periodic tasks with highly variable execution times and statistical QoS requirements. Here, QoS is defined as the probability that in an arbitrarily long execution history, a randomly selected execution of a given task will meet its deadline. More recently, complete operating systems have been proposed around the same ideas [3], indicating that voice and video communications systems can be more efficiently implemented on multi-channel devices by taking advantage of the inherent statistical properties of the software components for each channel.

Given a set of software components in a voice and video communication system, two approaches can be used to define the maximum number of channels per DSP. One approach relies on worst-case execution times and simple real-time scheduling (that is, worst-case provisioning). The other approach relies on statistical execution times and efficient scheduling for handling eventual (infrequent) peaks on execution time (that is, statistical provisioning). We would expect the statistical approach to lead to more efficient usage of the available DSP resources, as discussed in **Section 3**, *Experimental Results*, on page 7. An in-depth discussion of efficient scheduling schemes is beyond the scope of this note.

Based on classical results from probability theory, we can define an efficient method for maximizing channel density on embedded systems. We consider QoS from a statistical point of view by defining a target confidence level, P_o, to reflect the likelihood that a multi-channel system running several independent software components will not exceed its target execution time budget M. Next, we use the execution time average and variance of the different software components in the system to obtain a significant and predictable statistical multiplexing gain in channel density with respect to worst-case provisioning. We then explore the statistical multiplexing gain to increase average system usage. As a result of this analysis, we can answer the following questions:

1. What is the statistical confidence level P_s that N independent channels will not exceed the execution time budget M (that is, $P_s = f_1(N, M)$)?
2. What should be the execution time budget M_s for processing a given number N of independent channels with confidence level P_o (that is, $M_s = f_2(N, P_0)$)?
3. What is the maximum number N_s of independent channels that can be processed within a given execution time budget M with confidence level P_0 (that is, $N_s = f_3(M, P_0)$)?

Using independent test signals (vectors), we define a multi-channel loading scheme in **Section 3**, and we present experimental results for validating theoretical predictions. Regardless of the execution time profiles of the software components in the system, N independent channels of the same software component tend to have an execution time profile of a normal distribution as the number of channels increases. Furthermore, mixing multiple channels of different software components leads to execution time profiles with a normal distribution. Therefore, the total execution time profile of N independent channels composed of different software components is also expected to have a normal distribution. This result greatly simplifies the statistical analysis of expected execution time.

Most underlying software components in an embedded system can be extensively profiled using representative test vectors, so the average and variance execution times can easily be estimated. Here, we measure execution times in million of cycles per second (MCPS). Execution time profiles are characterized by MCPS histograms $h(m)$ so that the average and variance MCPS are estimated using the first and second moments of $h(m)$. When the average and

variance MCPS of each software component are known, the average MCPS μ_t and variance MCPS σ_t^2 of a set of different and independent software components are simply the sum of the individual averages and variances per component, respectively. Therefore, the total average and variance MCPS of a system composed of *N* independent channels is $N\mu_t$ and $N\sigma_t^2$, respectively. Estimating and using the total average and variance MCPS are the key steps for the analysis described in **Section 2**.

2 Statistical Analysis

The goal of the brief statistical analysis presented in this section is to answer the three basic questions outlined in **Section 1**. Any DSP has a maximum number of cycles per second for executing different software components. In a statistical analysis, the input signals processed by a given set of software components are viewed as independent signals. The execution times of these software components are inherently variable, so the processing load *m* per component, commonly measured in MCPS, can be modeled as a random variable ***m***. In a multi-channel system, a DSP is expected to handle a target number *N* of channels with a known number *C* of different software components. If statistically representative sets of test vectors are available for estimating MCPS histograms $h_c(m)$ for every component, statistical analysis can be used to predict the QoS of the system and associate a level of confidence that depends on the expected quality.

In the current analysis, *M* is the maximum available MCPS per DSP. Every histogram $h_c(m)$, $c = 1, \ldots, C$, is estimated as a function of *m* (MCPS), so that the following is true:

$$\sum_m h_c(m) = 1$$

The MCPS mean and variance estimates per component are easily computed as follows:

$$\mu_c = \sum_m m h_c(m)$$

$$\sigma_c^2 = \sum_m m^2 h_c(m) - \mu_c^2$$

Denoting by $\boldsymbol{m}_k$ the MCPS consumption of the k^{th} channel for a given component *c* in a system with *N* independent channels, the total MCPS required by a single component $\boldsymbol{m}_t = \boldsymbol{m}_1 + \boldsymbol{m}_2 + \ldots + \boldsymbol{m}_N$ is the sum of the MCPS required for every channel. Because of the inherent stochastic nature of MCPS and statistical independence of the measurements $\boldsymbol{m}_k$, the total MCPS $\boldsymbol{m}_t$ tends to have a normal distribution (Central Limit Theorem of Probability [4]), as is illustrated in **Section 3**. To assess this trend more effectively, we must look at the estimated probability density function (pdf) of the normalized MCPS as follows due to statistical independence (notice that the normalized MCPS has zero mean and unit variance):

$$\boldsymbol{m}_t' = (\boldsymbol{m}_t - \mu)/\sigma, \text{ where } \mu = N\mu_c \text{ and } \sigma = \sqrt{N}\sigma_c$$

The pdf is estimated from the MCPS histogram by simply adjusting its scale to represent $\boldsymbol{m}_t'$ (normalized MCPS), and normalizing the likelihood $h(m)$ by the smallest normalized MCPS increment. The resulting pdf estimates are then expected to converge to a normal distribution $N\ (0,\ 1)$ with zero mean and unit variance, that is, pdf as follows:

$$f(x) = e^{-x^2/2}/\sqrt{2\pi}$$

This in turn implies that the total MCPS pdf converges to a normal distribution, as follows, as the number of channels increases:

$$N(N\mu_c, N\sigma_c^2)$$

Therefore, a multi-channel system composed of independent components can be approximated as having an MCPS histogram $h_s(m)$ according to a normal distribution:

$$N(N\mu_t, N\sigma_t^2)$$

Where:

$$\mu_t = \sum_{c=1}^{C} \mu_c$$

$$\sigma_t^2 = \sum_{c=1}^{C} \sigma_c^2$$

The likelihood P_o that a multi-channel system running several independent software components will not exceed a target execution time budget M is then given by the probability that the total MCPS consumption in the system is smaller than or equal to M. That is it is $P_o = Prob(\boldsymbol{m} \le M)$, where $\boldsymbol{m}$ is modeled as a normal distribution:

$$N(N\mu_t, N\sigma_t^2)$$

Therefore, it becomes straightforward to answer the questions in **Section 1**.

1. Statistical confidence level: $P_s = f_1(N, M)$

$$P_o = Prob(\boldsymbol{m} \le M) = Prob\left(\frac{\boldsymbol{m} - N\mu_t}{\sqrt{N}\sigma_c} \le \frac{M - N\mu_t}{\sqrt{N}\sigma_c}\right) = Prob(\boldsymbol{m}' \le \alpha_o)$$

where $\boldsymbol{m}'$ is a random variable with a distribution $N\ (0,\ 1)$ and:

$$\alpha_o = \frac{M - N\mu_t}{\sqrt{N}\sigma_c}$$

This implies that

$$P_s = P_o = \frac{1}{2}[1 + erf(\alpha_o / \sqrt{2})]$$

where:

$$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$$

When the normal distribution approximation is not good enough, we can use the Tchebysheff's inequality [4] to estimate an upper bound on the likelihood of exceeding the available MCPS budget, as follows:

$$Prob\left(\left| \frac{\boldsymbol{m} - N\mu_t}{\sqrt{N}\sigma_c} \right| \geq \alpha_o \right) \leq \frac{1}{\alpha_o^2}$$

2. Statistical MCPS budget: $M_s = f_2(N, P_0)$

From the preceding result (1),

$$\alpha_o = \sqrt{2} \cdot erf^{-1}(2 \cdot P_o - 1)$$

so that

$$M_s = M = N(\mu_t + \alpha \cdot \sigma_t)$$

$$\alpha = \alpha_o / \sqrt{N}$$

For the particular selection $P_o = 1 - 10^{-k}$, $K = 1, 2, \ldots$, some corresponding values of α_o (that is, normalized MCPS values associated with the selected confidence levels) are listed in **Table 1**.

Table 1. Values of α_o Corresponding to $P_o = 1 - 10^{-k}$

k	1	2	3	4	5	6	7	8
α_o	1.2816	2.3263	3.0902	3.7190	4.2649	4.7534	5.1993	5.6120

3. Statistical channel density: $N_s = f_3(M, P0)$

We combine the results of (1) and (2) as follows:

$$\alpha_o = \frac{M - N\mu_t}{\sqrt{N}\sigma_c} = \frac{\mu_t}{\sigma_t}\left(\frac{N_{mean} - N}{\sqrt{N}}\right) = \sqrt{2} \cdot erf^{-1}(2 \cdot P_o - 1)$$

$$N_{meam} = \frac{M}{\mu_t}$$

so that:

$$\frac{N_{mean} - N}{\sqrt{N}} = \sqrt{2 \cdot \beta}$$

$$\beta = \left[\frac{\sigma_t}{\mu_t} \cdot erf^{-1}(2 \cdot P_o - 1)\right]^2 = \frac{1}{2}\left(\alpha_o \cdot \frac{\sigma_t}{\mu_t}\right)^2$$

Solving this equation in terms of N and imposing the fact that $N < N_{mean}$ is an integer results in the following statistical channel density estimation:

$$N_s = N = \lfloor N_{mean} - N_{margin} \rfloor$$

$$N_{margin} = \beta(\sqrt{2/\beta \cdot N_{mean} + 1} - 1)$$

N_{mean} is the channel density based on the total average MCPS required for executing one instance of every software component, and N_{margin} is a statistical margin (in terms of number of channels) that depends on the expected level of confidence P_0 that N channels will not exceed the total MCPS budget. If the available MCPS and the number of channels are large, using statistical channel density estimation can provide a multiplexing gain per channel proportional to the difference between the peak and average MCPS.

To be more specific, let m_p denote the peak MCPS required to process a single channel of a given number of independent components. Using worst-case provisioning to handle N channels requires an MCPS budget $M_p = N \cdot m_p = N\,[\mu_t + (m_p - \mu_t)]$. On the other hand, using statistical provisioning for some target confidence level P_o requires an MCPS budget, as shown in question (2):

$$M_s = N(\mu_t + \alpha \cdot \sigma_t),\ \alpha = \alpha_o / \sqrt{N}$$

The worst-case provisioning MCPS margin with respect to the average MCPS is a straight-line $r_p = (m_p - \mu_t) \cdot N$, whereas the statistical provisioning MCPS margin with respect to the average MCPS is a square-root curve:

$$r_s = \alpha_o \cdot \sigma_t \cdot \sqrt{N}$$

Therefore, the margin with respect to the average MCPS consumption can differ greatly depending on whether statistical or worst-case processing loads is used. The expected multiplexing gain of statistical provisioning versus worst-case provisioning is:

$$M_{gain} = M_p - M_s = r_p - r_s = (m_p - \mu_t) \cdot N - \alpha_o \cdot \sigma_t \cdot \sqrt{N}$$

so that the multiplexing gain per channel $M_{gain} / N \rightarrow m_p - \mu_t$ as $N \rightarrow \infty$, assuming that the MCPS budget M per DSP can support it. We can also express this gain in terms of number of channels by looking at the difference between the statistical channel density and the worst-case channel density, that is, $N_p = \lfloor M / m_p \rfloor$, so that

$$N_{gain} = N_s - N_p$$

If the difference between the total peak and average MCPS per channel is not small and the number of channels is large, then it is simply impractical in many applications involving voice and video communications to assume worst-case execution times when assessing the number of channels to

process within a maximum execution time budget. The predictable statistical MCPS gain M_{gain} should not be ignored and can be used effectively to increase average system usage. **Table 2** summarizes the results of exploring questions (1), (2), and (3).

Table 2. Answers to Questions 1, 2, and 3

Qustion (1): Statistical confidence level	$P_s = \frac{1}{2}[1 + erf(\alpha_o/\sqrt{2})]$ $\alpha_o = \frac{M - N\mu_t}{\sqrt{N}\sigma_c}$
Question (2): Statistical MCPS budget	$M_s = N(\mu_t + \alpha \cdot \sigma_t)$ $\alpha = \alpha_o/\sqrt{N}$ $\alpha_o = \sqrt{2} \cdot erf^{-1}(2 \cdot P_o - 1)$
Question (3): Statistical channel density	$N_s = \lfloor N_{mean} - N_{margin} \rfloor$ $N_{mean} = M/\mu_t$ $N_{margin} = \beta(\sqrt{2/\beta \cdot N_{mean} + 1} - 1)$ $\beta = \frac{1}{2}\left(\alpha_o \cdot \frac{\sigma_t}{\mu_t}\right)^2$ $\alpha_o = \sqrt{2} \cdot erf^{-1}(2 \cdot P_o - 1)$
μ_t and σ_t are the average and standard deviation, respectively, of the total MCPS per channel to support all software components.	

3 Experimental Results

This section presents experimental results for both the component and system levels, and it uses real-life test cases to demonstrate agreement with the statistical analysis in **Section 2**.

The first experiment is performed with a single software component in a voice-over-Internet Protocol (VoIP) system, namely an echo canceller selected for its relatively large execution time variability. **Figure 1** illustrates the MCPS histogram of one representative channel of the echo canceller, which was implemented on a Freescale StarCore™ DSP. Similar results were observed for other components. Notice that there is a relatively large MCPS variability ($\sigma / \mu \approx 29\%$). This MCPS histogram was estimated from 48 representative and independent test vectors ($\boldsymbol{V}_k$). To illustrate the expected statistical gain from use of this software component, a multi-channel loading scheme is defined according to **Figure 2**. Different channels are processed in a typical time-division multiplex scheme using the same frame size, and the total MCPS is measured per frame. Every test vector is viewed as a different telephone call, so the echo canceller is initialized before each test vector is processed. Because the MCPS histogram is not affected by the order in which the test vectors are processed, all channels have

the same MCPS histogram. The total MCPS of multiple channels, that is, the sum of the required MCPS per channel over time, is then measured and analyzed. For example, a given channel may require six MCPS for a time frame, but another channel may require only two MCPS for the same time frame as they process uncorrelated signals.

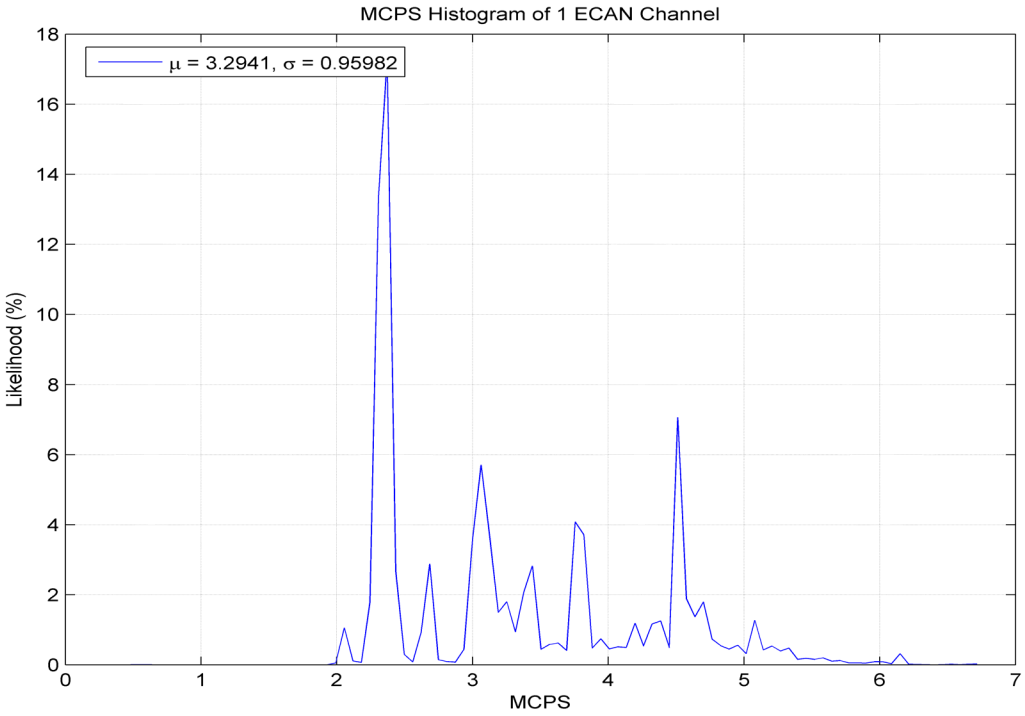

Figure 1. MCPS Histogram of a Single Echo Canceller Channel

Channel 1	V_1	V_2	▪ ▪ ▪	V_{48}
Channel 2	V_2	V_3	▪ ▪ ▪	V_1
▪ ▪ ▪				
Channel 48	V_{48}	V_1	▪ ▪ ▪	V_{47}
	Time Frames →			

Figure 2. Echo Canceller Multi-Channel Loading Scheme

Based on the statistical analysis in **Section 2**, the multi-channel pdf of the normalized MCPS is estimated for an increasing number of channels, namely 6, 12, 24, and 48. **Figure 3** presents all the resulting pdfs and compares the estimated pdf of $N = 48$ channels with a normal distribution of zero mean and unit variance. It is apparent that the total MCPS pdf converges to a normal distribution $N(N\mu, N\sigma^2)$ as the number of channels increases. μ and σ^2 are the average and variance MCPS of a single echo canceller channel. The predicted multi-channel pdf of the normalized MCPS (**Figure 4**) is based on 47 convolutions of the MCPS histogram of one echo canceller channel (**Figure 1**).

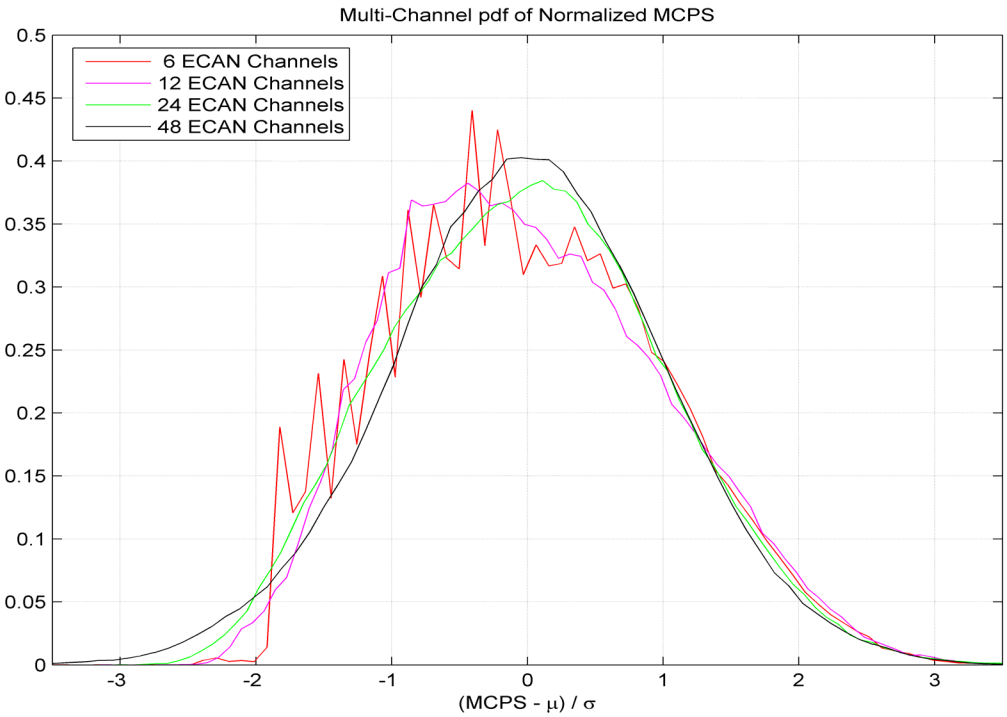

Figure 3. Multi-Channel PDF of Normalized MCPS

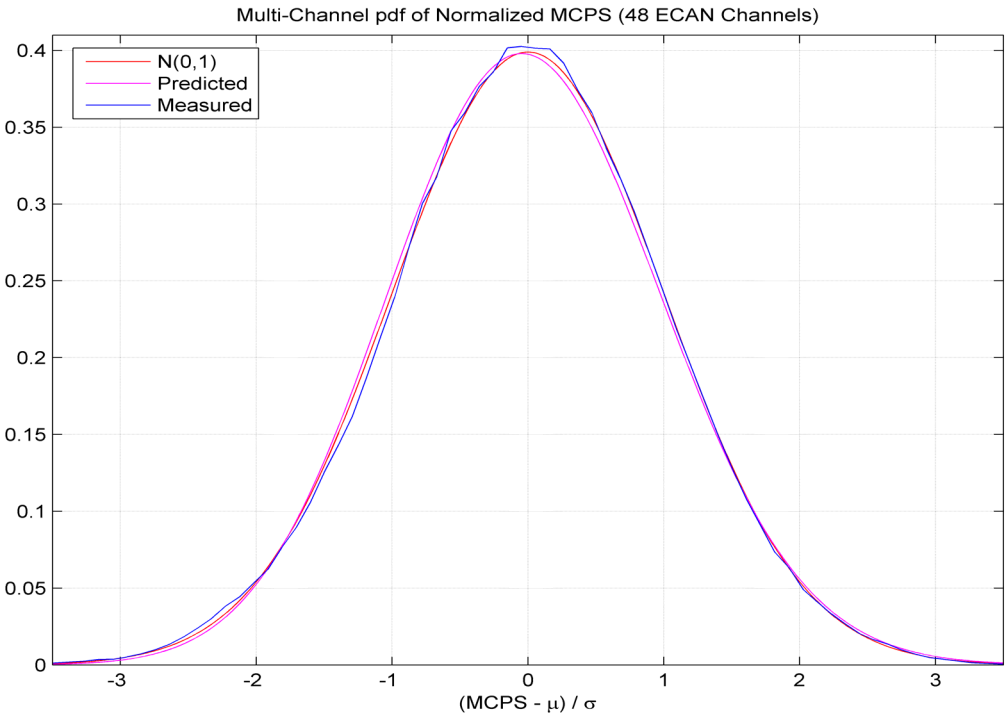

Figure 4. Estimated PDF Measurement of 48 Echo Canceller Channels Versus Theoretical Predictions

Figure 5 depicts the MCPS statistical gain (M_{gain}) with respect to the peak MCPS for different statistical confidence levels P_o. As expected, the MCPS gain can be significant as the number of channels increases. The predicted MCPS gain for a small number of channels may not represent a good approximation with respect to a normal distribution, and it should be used with care.

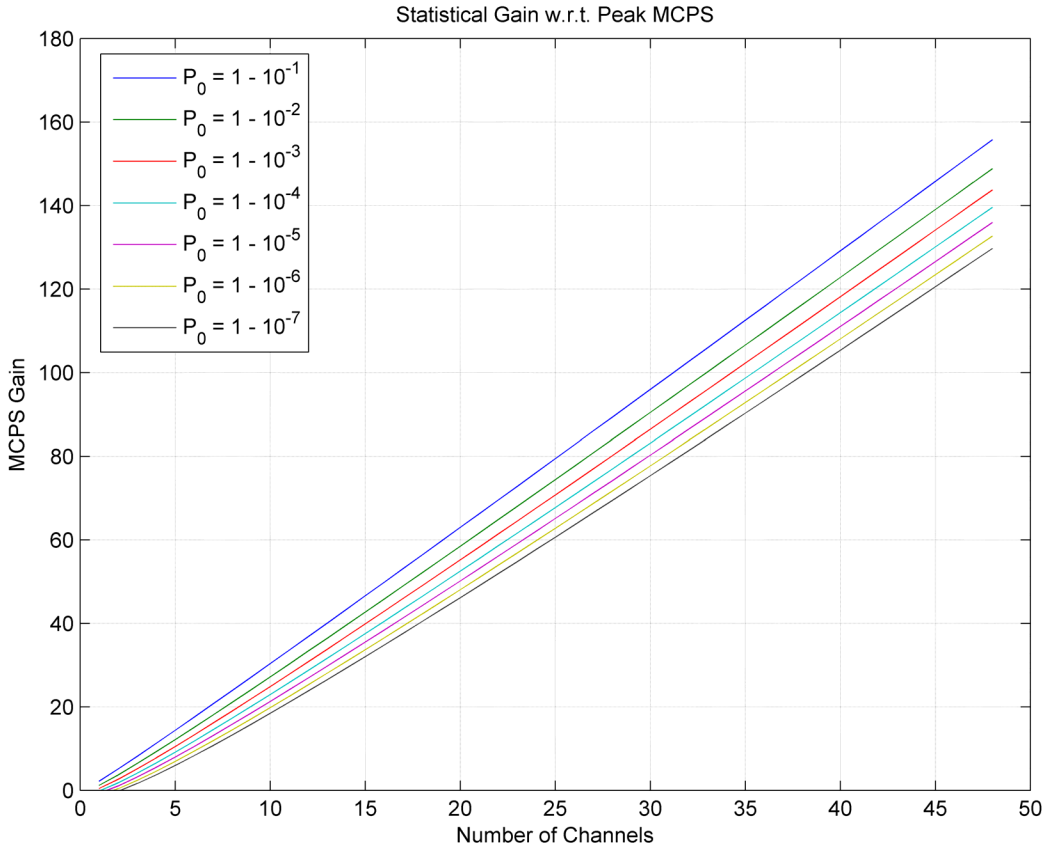

Figure 5. Statistical Gain with Respect to Peak MCPS

The second experiment was conducted on a complete VoIP system with different configurations with respect to the type of voice encoding transmitted through the IP network. This example focuses on one type of voice encoding processing 10 ms frames per function call of the various software components in the system. The total average, standard deviation, and peak MCPS are estimated as $\mu_t = 6.476$, $\sigma_t = 0.449$, and $m_p = 9.173$ on a Freescale StarCore-based DSP. A reference QoS was defined with a statistical confidence level $P_o = 1 - 10^{-7}$. This value corresponds to one failure because the MCPS budget is exceeded every 27h:46m:40s (that is, $10 \cdot 10^{-3} \cdot 10^{7} = 10^{5}$ s), which is more than adequate for most VoIP connections. An MCPS budget of $M = 400$ per DSP core is selected, the channel density based on worst-case provisioning is $N_p = 43$ channels. On the other hand, the corresponding channel density based on statistical provisioning is $N_s = 58$ channels (see **Table 2**), which implies a gain of $N_{gain} = N_s - N_p = 15$ additional channels with respect to worst-case provisioning. Notice that $N_s \cdot \mu_t \approx 376$ MCPS, such that $M - N_s \cdot \mu_t \approx 24$ MCPS are required to guarantee the target level of confidence. This is less than 0.42 MCPS per channel instead of the $m_p - \mu_t \approx 2.7$ MCPS per channel required for worst-case provisioning. Therefore, using worst-case provisioning unnecessarily reduces channel density. Other deterministic factors, such as memory usage, cache effect (if used), and fixed MCPS consumption for other supporting components of the system had to be accounted for (for example, by reserving a portion of the total MCPS budget) before this analysis was performed.

4 Conclusion

This application note demonstrates that, under conditions of statistical independence, we can maximize channel density on embedded systems using statistical QoS, which is based on the likelihood that a multi-channel system running several independent software components will not exceed a target execution time budget. A brief statistical analysis and supporting experimental results show that if the average and variance MCPS per software component are known, the system can handle more attractive channel densities based on statistical provisioning rather than worst-case provisioning. If the available MCPS and the number of channels are large enough, statistical provisioning can provide a multiplexing gain per channel proportional to the difference between the total peak and average MCPS. We can similarly define more efficient real-time schedulers to take advantage of the inherent statistical MCPS gain of multi-channel systems and handle infrequent conditions of over-budget MCPS by gracefully degrading the system in a way that is controlled and predictable.

5 References

[1] Y. Xie and T. Yang, “Efficient Admission Control for EDF Scheduler with Statistical QoS Guarantee,” *Proceedings of the IEEE International Conference on Computer Communications and Networks*, pp. 242-247, 1997.

[2] A. Atlas and A. Bestavros, “Statistical Rate Monotonic Scheduling,” *Proceedings of the IEEE Real-Time Systems Symposium*, pp. 123-132, 1998.

[3] R. Kroger, *Admission Control for Independently-authored Realtime Applications*. Ph.D. Thesis, University of Waterloo, 2004.

[4] A. Papoulis and S. U. Pillai, *Probability, Random Variables and Stochastic Processes*. McGraw-Hill, 4th Edition, 2001.

How to Reach Us:

Home Page:
www.freescale.com

E-mail:
support@freescale.com

USA/Europe or Locations not listed:
Freescale Semiconductor
Technical Information Center, CH370
1300 N. Alma School Road
Chandler, Arizona 85224
+1-800-521-6274 or +1-480-768-2130
support@freescale.com

Europe, Middle East, and Africa:
Freescale Halbleiter Deutschland GMBH
Technical Information Center
Schatzbogen 7
81829 München, Germany
+44 1296 380 456 (English)
+46 8 52200080 (English)
+49 89 92103 559 (German)
+33 1 69 35 48 48 (French)
support@freescale.com

Japan:
Freescale Semiconductor Japan Ltd.
Headquarters
ARCO Tower 15F
1-8-1, Shimo-Meguro, Meguro-ku,
Tokyo 153-0064, Japan
0120 191014 or +81 3 5437 9125
support.japan@freescale.com

Asia/Pacific:
Freescale Semiconductor Hong Kong Ltd.
Technical Information Center
2 Dai King Street
Tai Po Industrial Estate
Tai Po, N.T. Hong Kong
+800 2666 8080

For Literature Requests Only:
Freescale Semiconductor Literature Distribution Center
P.O. Box 5405
Denver, Colorado 80217
1-800-441-2447 or 303-675-2140
Fax: 303-675-2150
LDCForFreescaleSemiconductor@hibbertgroup.com

Information in this document is provided solely to enable system and software implementers to use Freescale Semiconductor products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits or integrated circuits based on the information in this document.

Freescale Semiconductor reserves the right to make changes without further notice to any products herein. Freescale Semiconductor makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale Semiconductor assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. “Typical” parameters which may be provided in Freescale Semiconductor data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including “Typicals” must be validated for each customer application by customer’s technical experts. Freescale Semiconductor does not convey any license under its patent rights nor the rights of others. Freescale Semiconductor products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Freescale Semiconductor product could create a situation where personal injury or death may occur. Should Buyer purchase or use Freescale Semiconductor products for any such unintended or unauthorized application, Buyer shall indemnify and hold Freescale Semiconductor and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Freescale Semiconductor was negligent regarding the design or manufacture of the part.

Freescale™ and the Freescale logo are trademarks of Freescale Semiconductor, Inc. StarCore is a licensed trademark of StarCore LLC. All other product or service names are the property of their respective owners.

© Freescale Semiconductor, Inc. 2005.

Document Order No.: AN3012
Rev. 0
10/2005