**Chapter 1**

**INTRODUCTION**

This paper entitled “PowerPi: Measuring and Modelling the Power Consumption of the Raspberry Pi” considers the world’s population of 7.1 billion and a fixed broadband subscription rate of 10.82% in 20121, there exist around 770 million home gateways worldwide. With a power consumption of around 10 W each, their power draw alone results in approximately 6.7T Wh of electrical energy per year. This corresponds to 0.03% of the world electricity consumption2, or between 2.6% and 5% of the Internet power consumption. Green networks are an emerging topic. estimate the carbon footprint of the Internet [1], which is extended by Hinton et al. to include the cost of content storage and delivery [11]. Chiaravigli focus on Internet Service Provider (ISP) networks and the reduction of their power consumption. Other, very active areas of energy improvements are cellular networks in general [9] and the optimization of 4G networks [5].

The development of energy models currently focuses mainly on servers [2], desktop PCs [6] or mobile handsets. Less work is done on profiling the power consumption of home gateways, access points or small servers. Due to the high number of these devices, these cannot be neglected when evaluating the power consumption of the full Internet infrastructure. 43% of these devices located in the developed world are always on [8] and often idle. The idle resources of such devices may be used to provide network access to other users or pre-load content for their owners, while they are away. The devices are built to be cheap and reasonably energy efficient. However, no detailed power model for this device class is available, which would allow for software based optimization. In most calculations, fixed power consumption is assumed, independent of the device utilization.

The Raspberry Pi is a popular platform for low-power and low-cost computational tasks, suitable for a large range of applications. It is used as a platform to model cloud computing, for home monitoring and automation, or to provide low cost computation to developing countries [10]. Applications for the Raspberry Pi range from enhanced Internet gateways (Nano Datacenters (NaDas) [24], supporting the intelligent caching of video content, over intelligent WiFi access points [21], future ICN applications [16], to applications in outer space [3].

* 1. **Overview**

The method proposed in tries to to accurately estimate the power consumption and possible improvements to each application, accurate power models of the devices are necessary.

To this end, this paper presents PowerPi, a power model focusing on the power consumption of the Raspberry Pi to derive possible power saving strategies. The hypotheses of the paper are:

* An accurate estimation of the Raspberry Pi’s power consumption can be obtained using the system utilization only.
* Knowing the power model, it is possible to reduce the power consumption by optimizing the software running on the platform.

PowerPi is based on hardware measurements of the Raspberry Pi. The remainder of this paper is structured as follows.

The setup and configuration of the platform is detailed in Chapter II, which gives an overview of tools and services used during the experiment and custom tools to generate load and monitor the system state.

The measurements are described in Chapter III, showing a linear dependency between the CPU power consumption and the utilization. Similarly, second to fourth order functions between the data rate on the interface and the power consumption are derived.

Based on these, the model generation and models approximating the behaviour of the device under load are described in Chapter IV.

The resulting model is compared to similar approaches in Chapter V.

Finally, chapter VI concludes the paper and gives an outlook on possible applications of the energy model.

**Chapter 2**

**LITERATURE SURVEY**

This chapter discusses about the measuring and modeling of the power consumption in Raspberry Pi.

1. **Minimizing Power Consumption in Raspberry Pi to use as remote WSN Gateway**

The reliability of Raspberry Pi minicomputer as a remote gateway in a wireless sensor network requires an analysis of power consumption. Due to the limited energy conditions, when a station is powered by a photovoltaic system, it is essential to reduce its energy consumption to maximize the lifetime of the network. This paper presents a comparative analysis of Raspberry Pi power consumption profile under multiple settings: reduced hardware elements, installation of different operating systems and, down clock CPU (700 *MHz* to 100 *MHz*), SDRAM (400 *MHz* to100 *MHz*) and GPU (250 *MHz* to 50 *MHz*) units. Based on the results, the energy consumption is reduced up to 20% which is equivalent to a saving of 0*.*25 *W*. The minimum Raspberry Pi power consumption reached is 0*.*97 *W*.

1. **Power Consumption and Energy Efficiency in the Internet**

This article provides an overview of a network-based model of power consumption in Internet infrastructure. This model provides insight into how different parts of the Internet will contribute to network power as Internet access increase over time. The model shows that today the access network dominates the Internet’s power consumption and, as access speeds grow, the core network routers will dominate power consumption. The power consumption of data centers and content distribution networks is dominated by the power consumption of data storage for material that is infrequently downloaded and by the transport of the data for material that is frequently downloaded. Based on the model several strategies to improve the energy efficiency of the Internet are presented.

1. **Power Consumption of the Raspberry Pi: A Comparative Analysis**

Over the past few decades, human beings have increasingly adopted different types of personal computers including desktop computers, laptops, tablets and smart phones. More recently, there has been the emergence of the Raspberry Pi and since its release in 2012, this new type of computer has undergone rapid growth in adoption to even become the fastest selling British computer. The Raspberry Pi has often been referred as a computer designed to change the world since it is capable to do most things that a desktop computer can do. The growing concern is that all these computers utilize power in order to operate thereby turning ICT into a power drainer. The diverse functionalities present in modern computers including communication and web browsing, among others, were found to be important components that affect the power consumption of such devices. As such, this paper investigates how power consumption of the Raspberry Pi is affected by the key functionalities that could be performed by end-users on the platform. Moreover, this relationship is compared against other types of common personal computers before recommending on techniques and practices that could reduce the power consumption of this emerging type of computer.

1. **SMART MIRROR FOR AMBIENT HOME ENVIRONMENT**

This paper describes the design and development of a futuristic smart mirror that represents an unobtrusive interface for the ambient home environment. The mirror provides a natural means of interaction through which the residents can control the household smart appliances and access personalized services. Emphasis is also given to ensure convenience in accessing these services with a minimum amount of user intervention. For example, face recognition-based authentication is used to automatically identify the user facing the mirror and provide widget-based interface to access data feeds and other services. Aservice-oriented architecture has been adopted to develop and deploy the various services, where the mirror interface, the appliances, and the news and data feeds all use web service communication mechanisms. The smart mirror functionalities have been demonstrated bydeveloping an easily extendable home automation system that facilitates the integration of household appliances and various customized information services.

**Chapter 3**

**PROBLEM STATEMENT**

There are several related works which focus on the high performance, networking models for the Raspberry Pi. But there exists no power model for the Raspberry Pi or for any other ARM boards.

**3.1 Proposed System**

This paper presents PowerPi, a power model focusing on the power consumption of the Raspberry Pi to derive possible power saving strategies. The hypotheses of the paper are:

* An accurate estimation of the Raspberry Pi’s power consumption can be obtained using the system utilization only.
* Knowing the power model, it is possible to reduce the power consumption by optimizing the software running on the platform.



PowerPi is based on hardware measurements of the Raspberry Pi.

**3.2 Advantages of proposed system**

* Using the PowerPi model, and similar models of other devices connected to the network, it becomes possible to estimate the power consumption of the full network for a given load.
* Energy efficiency improvements of the full network infrastructure are possible based on the power models and the system utilization only.
* Eliminates the need for dedicated power measurement hardware, but allows derivation of the power consumption with an accuracy of lower than 3.3%.
* Allows optimizing the traffic flows to use the most energy-efficient paths, or redirect computations to the most energy-efficient location based on the current utilization.

**Chapter 4**

**METHODOLOGY**

PowerPi measures the power consumption of the Raspberry Pi using an external power meter. Simultaneously, scripts and custom tools on the platform generate load and monitor the device state. The following sections describe the measurement setup in detail.

**4.1 Power Measurement**

The Raspberry Pi is a low-power device, which supports being powered via USB. Its power consumption is measured by interrupting the power lines of the USB connection and inserting a measurement shunt in the 5 V line. The wiring of the setup, as shown in Figure 1a, is detailed in Figure 1b. The current flowing through R1 causes a voltage U1, proportional to the current drawn by the Raspberry Pi. The requirements for the measurement shunt are twofold. First, it must be large enough to create a voltage that can easily be measured. Secondly, it must be small enough to reduce the voltage drop to a minimum, allowing the connected device to start. A resistance of 100 m, with a maximum current of 1.2 A creates a voltage drop of 120 mV, which reduces the voltage on the +5 V line to 4.88 V. This is still sufficient to operate a USB device. Still, the voltage U1 when only small currents are drawn is around 30 mV, allowing a sufficient accuracy. A 12 bit A/D converter with an absolute error of 6 mV results in a relative error of 20 %. Therefore, Measurement Computing’s USB1608-FSPlus is used, which has a resolution of 16 bits, allowing the measurement of voltages of a few mV with an absolute accuracy of 0.68 mV, thus reducing the error to 2.3

% for idle measurements. The voltage U2 between the 5 V line and GND is measured directly.

A custom built software based on Measurement Computing’s FlexDAQ API3 constantly measures the voltage drop U1 over this shunt and the voltage U2 of the 5 V line. The power consumption of the Raspberry Pi is calculated with



The software then writes the measurements together with a time-stamp to a local file.

The error of the power measurement depends on the accuracy of the two voltage measurements, which depend on the accuracy of the A/D conversion and the accuracy of the measurement shunt. Mathematically, the maximum error is defined as:



The maximum error of the voltage measurement  can be calculated directly from the above considerations as:



Similarly, the maximum error of the current measurement is calculated.



The maximum error for the voltage U1 is,



Combining the error of the measurement with the tolerance of the shunt of 1% results in an error of



Which results in a maximum absolute measurement error of



This error is the upper bound of the errors introduced by the measurement setup. The actual accuracy of the measured samples is expected to be better. Furthermore, averaging the measurements over the evaluation period reduces the error of the final power measurements to even lower values.

**4.2 Measurement PC Setup**

These measurements are run from a conventional laptop. The only hardware requirements are a USB 2.0 interface for the measurement card and a Gigabit Ethernet interface to run the network tests. The Gigabit interface is recommended, as it ensures that the bottleneck of the throughput tests is located on the platform’s network chip and not on the measurement PC. The Wireless Fidelity (WiFi) measurements can be run over a conventional WiFi Access Point (AP), or by creating a software AP on the laptop. Here, the over-provisioning of bandwidth on the remote side is difficult, as the WiFi interface selected supports the 802.11n standard. The measurements are best run from a Linux PC, as most software required for the measurements is readily available for this platform. Still, the power measurement software is written in Java, hence running measurements is possible on each OS.

Before running the tests, the measurement PC is prepared by installing and configuring a number of applications. A custom built software measures the voltages on the USB connection and writes the values to a CSV file. This is later evaluated and correlated with the measured system and network utilization using MATLAB scripts.

Furthermore, the Precision Time Protocol (PTP) [14] is run in server mode on the PC to allow the Raspberry Pi to synchronize its clock before executing measurements. iPerf is installed on both the measurement PC and the Raspberry Pi and run either in server or client mode to generate traffic in the respective direction. It is run as a daemon in UDP mode, as only this mode allows configuration of the target data rates. The system parameters such as current CPU utilization or consumed bandwidth are monitored on the Raspberry Pi itself.

**4.3 Setup of Raspberry Pi**

The measured platform is a Raspberry Pi Model B running a software image based on Raspbian4. It is a Debian-based Linux distribution specifically designed for the Raspberry Pi. For the wireless tests, a USB WiFi dongle, the D-Link DWLG122 is used. It was selected because Linux drivers are available and high data rates (802.11n) are supported.

The default Raspbian image is extended by running PTP in client mode and a number of scripts monitoring the system utilization. The hardware monitors are detailed in Section II-D. Each monitor stores the collected measurements in the RAM until the end of the experiment. This minimizes the influence of the logging on the host system. The results are written to the SD-card only after the tests have finished.

**4.4 System State Monitoring**

During the tests, only required services run, minimizing side effects. These services are udev, dhcp client, ssh server, and dbus. All other services are stopped after the boot process completes.

The system is monitored using custom scripts tracking state and utilization of the platform. As the CPU monitoring application also causes CPU utilization, special consideration was paid to reduce its influence to a minimum.

Parsing the output of tools such as top or ps is resource intensive. A cause of this is the high amount of text output generated by these tools. Therefore, a very lightweight utilization monitor was written in C and compiled with the -O3 option of gcc to optimize the generated code. To avoid disk I/O, the monitors do not directly write to the memory card. The measurement script mounts a ram-disk to the /tmp folder and copies the content after the measurement is finished.

The CPU monitor reads the /proc/stat file , which includes information about the number of cycles the

CPU was busy (user, nice, system), in idle state, interrupted, and some more states since the last boot. The monitor takes the busy and idle states and computes the utilization according to Equation 9. Since the available values are incremental over time, the system utilization must be calculated from the difference of the counters. The system utilization u[t] is calculated by dividing the busy cycles cbusy[t] by the total

number of cycles ctotal[t] during the last evaluation interval.



The total number of busy cycles up to time t is defined as:



Here, cuser[t] are the user generated CPU cycles, while cnice[t] and csystem[t] are the cycles created by low priority processes and the system respectively. As we are interested in the full system load, these processes must be included in the calculation. The total number of cycles ctotal[t] is the number of busy cycles cbusy[t] plus the number of idle cycles cidle[t]

This leads to:



Hence, the utilization at time t is calculated based on the difference of utilization cycles during the last measurement period.

Similar to the CPU monitor, a network monitor was written to keep track of the network utilization. The advantages of this approach are the low overhead, as the proc file system is used, the possibility to use any traffic generator and the elimination on parsing the output of bandwidth measurement tools. The drawback is the reduced accuracy of the first and last sample of an experiment. The influence of these is eliminated by running each test for a considerable time. The /proc file system is read at /proc/net/dev, returning the processed packets on kernel level. This is advantageous, as it contains the raw number of bytes sent and received via the interface. Similar to the /proc/stat file, the counters are incremental. The current bandwidth (in B/s) is calculated by:



where  is the time interval between t - 1 and t and B[t] is the absolute amount of data transmitted or received on the interface.

**4.5 Load Generation**

To measure the different operating points of the CPU and the network interfaces, it is required to generate a configurable load. For this a combined approach of a load generator and load limiter is chosen. The CPU utilization is limited using a tool called cpulimit. It is available from the Raspbian repositories but has one major drawback for the measurements. It is designed to limit the CPU utilization of a specific process (including its children), but cannot control the overall CPU utilization. Hence, the cpulimit source code was modified to measure the full system utilization. The unmodified version of cpu limit reaches the targeted utilization very accurately because it has no external disturbance. After modifying the source code, all other processes also influence the measured CPU utilization. Hence the variance is higher. The load is generated by running an infinite loop adding numbers as shown in Listing 2, filling up the CPU load to the desired limit.



The load on the network interfaces may be generated by a number of tools. The basic differentiation of these is between TCP and UDP connections. TCP is the most widely used protocol in the Internet, hence, its performance is of high interest. Still, UDP has a number of advantages considering the measurements. The most important aspect is the missing traffic control, which allows configuring a fixed data rate beforehand. This further eliminates the errors introduced by TCP’s slow start and congestion avoidance algorithms. As UDP uses no back channel, the measurement of only the incoming or outgoing traffic counter is sufficient. Furthermore, only the bytes on the wire are used to calculate the power consumption.

Considering the final model, the selection of UDP has no influence on the applicability of the final power model. Contrary, the accuracy of the measurements is improved. As the kernel file system is evaluated during the model generation, the power consumption generated by TCP traffic can be modeled as well.

As a traffic generator, iPerf was selected. It allows configuration of the bandwidth for UDP connections. The measurements are automated using a script running iPerf in different configurations. iPerf also returns traffic statistics during and after each run, but these miss the required accuracy. Furthermore, they only contain the self generated traffic and require parsing of the command line output.

**Chapter 5**

**RESULTS AND DISCUSSIONS**

**5.1 Measurements**

The measurements were conducted in a home environment during the night to reduce potential interference of the WiFi measurements. The power measurements are conducted with a sampling rate of 1 kS/s, while the maximum update rate of the bandwidth and utilization measurements is one sample per second. Hence, a block-wise average is applied to the power measurement samples. The start of the blocks was determined based on the beginning of the utilization samples. All measurement values between two system or network utilization samples are averaged and then mapped to the second value.



The time difference between the utilization and bandwidth measurement points is always below 80 ms. Hence, the time difference may be neglected. Each test runs for 900 seconds, resulting in 900k power measurements, which are reduced to 900 combined utilization and power measurements. For each experiment 10 different operating points are configured and measured, resulting in 90k independent samples for each approximation.

The CPU utilization measurement is executed without network access to reduce external influences to a minimum. The collected measurements are plotted in heat map to allow a visualization of the density of the measurements.



This is advantageous compared to scatter plots, as the high number of measurements reduces the visibility of the individual data points. The heat map is logarithmically weighted to visualize the full range of measurements. On top of the heat map, the models derived in Section IV are plotted.

****

**5.2 Model Generation**

The power model for the different measurements is generated by fitting a linear function to the measured data, minimizing the remaining root mean square error (RMSE).

For this purpose, Matlab’sTM robustfit function is used. This function is based on an iterative process fitting the linear function to the data, minimizing the RMSE. The detailed process is described in [13]. The underlying data are weighted with a bi-square function to reduce the effect of outliers on the final fit.

**5.2.1 Description of the Power Model**

The measurements of the CPU utilization in Figure 2 show a clear linear dependency. This observation is confirmed, by calculating the 1st order regression. The resulting function for the platform including CPU utilization is



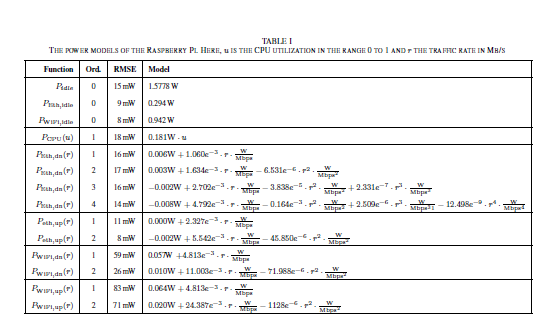
Here, u is the CPU utilization in the range 0 to 1 as defined in Equation 9. The resulting RMSE is 18.9 mW, which corresponds to an error of 1.2%. The measurements below 10% CPU utilization represent the idle state of the Raspberry Pi without additional load.

Figure 3 shows the fourth order approximation to the Ethernet download measurements, for which Matlab’s robust-fit function was used. The resulting RMSE of the 4th order function is 14 mW. This is only slightly lower than the RMSE of the first order approximation of 16 mW. For practical reasons, it might be sufficient to use the first order function when estimating the power.

**5.2.2 Combined Power Model and Usage**

Table I shows the approximations of the power consumption of the Raspberry Pi for different utilization. The first column is the symbol used in the text to refer to this function. The second column indicates the approximation order, while the third column gives the RMSE, which is the mean error to be expected when using the model. The last column lists the formula describing the dependency between utilization and power consumption.

The table is grouped to distinguish the different measurement categories. The first group gives the idle power consumption of the Raspberry Pi for different power states. PEth;idle and PWiFi;idle denote the power draw when the platform is idle. The second group describes the power consumption of the platform depending on the CPU utilization. This is included as the variable u, giving the CPU utilization as defined in Equation 9. The remaining groups show the power consumption of the data transfers on both network interfaces. The first term of each model in these groups is a correction term necessary to fit the model to the constants determined before. The other terms are modeled to depend on the transferred data rate r in Mb/s or the CPU utilization u as defined in Equation 9.



This results in an additive model, where the absolute power consumption of the Raspberry Pi can be modeled on a per-component basis. The model can be expressed as where the constants Pidle and Pif;idle and the approximations PCPU(u) and Pif;d(r) are defined in Table I. Here, the interface if is either WiFi or Eth. The RMSEs of the built in components are generally quite low (<18 mW). Only the WiFi measurement shows a larger error. This is explained by a higher power consumption of the USB dongle, leading to a higher variance of the combined measurements. The power consumptionof the USB WiFi dongle with 2 W is double the platform’s power consumption, and close to the maximum allowed power draw of a USB 2.0 device of 2.5W.

**Chapter 6**

**CONCLUSION AND OUTLOOK**

This paper presented PowerPi, a power model for the Raspberry Pi, which includes the CPU and Ethernet power consumption as well as the power consumption of an external USB WiFi dongle. The power model is modular to incorporate all measured components.

At the beginning of the paper, the following hypotheses

were made:

* An accurate estimation of the Raspberry Pi’s power consumption can be obtained using the system utilization only.
* Knowing the power model, it is possible to reduce the power consumption by optimizing the software running on the platform.

The error of the model has been evaluated based on the measurement accuracy and the error introduced by the model.

The resulting errors for the built-in components are in the order of tens of mW only. PowerPi can be used to improve the energy footprint of software running on the Raspberry Pi, using system traces only. Similar power models can easily be generated for other devices by repeating the same measurements.

These power models are valuable, as future Internet services may likely run on the end-user’s premises. Examples of possible future distributed services are NaDas [24], Multiservice Home Gateways [20] or HORST [21]. These aim at improving the Quality of Service (QoS) of the specific service and reducing the dependency on the up-link bandwidth, while requiring local computation and storage.

Using the PowerPi model, and similar models of other devices connected to the network, it becomes possible to estimate the power consumption of the full network for a given load. Hence, energy efficiency improvements of the full network infrastructure are possible based on the power models and the system utilization only. This eliminates the need for dedicated power measurement hardware, but allows derivation of the power consumption with an accuracy of lower than 3.3%. For a networked system specialized on traffic forwarding, the traffic statistics alone are sufficient to generate accurate predictions of the network state and power consumption. This allows optimizing the traffic flows to use the most energy-efficient paths, or redirect computations to the most energy-efficient location based on the current utilization.

Thus, both hypotheses are supported by this paper.

**Chapter 7**

**REFERENCES**

[1] J. Baliga, K. Hinton, R. Ayre, and R. S. Tucker, “Carbon footprint of the Internet,”

[2] L. A. Barroso and U. Hölzle, “The Case for Energy-Proportional Computing,” Computer, vol. 40, pp. 33–37, 2007.

[3] J. Berk, J. Straub, and D. Whalen, “The Open Prototype for Educational NanoSats: Fixing the Other Side of the Small Satellite Cost Equation,” in IEEE Aerospace Conference, 2013.

[4] L. Chiaraviglio, M. Mellia, and F. Neri, “Minimizing ISP Network Energy Cost: Formulation and Solutions,” IEEE/ACM Transactions on Networking, 2012.

[5] P. Dini, M. Miozzo, N. Bui, and N. Baldo, “A Model to Analyze the Energy Savings of Base Station Sleep Mode in LTE HetNets,” in IEEE International Conference on Green Computing and Communications and IEEE Internet of Things and IEEE Cyber, Physical and Social Computing, 2013.

[6] M. E. M. Diouri, M. F. Dolz, O. Glück, L. Lefèvre, P. Alonso, S. Catalán, R. Mayo, and E. S. Quintana-Ortí, “Solving Some Mysteries in Power Monitoring of Servers: Take Care of Your Wattmeters!” in Energy Efficiency in Large Scale Distributed Systems, ser. Lecture Notes in Computer Science, J.-M. Pierson, G. Da Costa, and L. Dittmann, Eds. Springer Berlin Heidelberg, 2013.

[7] K. Gomez, R. Riggio, T. Rasheed, D. Miorandi, and F. Granelli, “Energino: a Hardware and Software Solution for Energy Consumption Monitoring,” in International Workshop on Wireless Network Measurements, 2012.

[8] S. Grover, M. S. Park, S. Sundaresan, S. Burnett, H. Kim, and N. Feamster, “Peeking Behind the NAT: An Empirical Study of Home Networks,” in IMC, 2013.

[9] Z. Hasan, H. Boostanimehr, and V. K. Bharghava, “Green Cellular Networks: A Survey, Some Research Issues and Challenges,” IEEE Communications Surveys and Tutorials, 2011.

[10] R. Heeks and A. Robinson, “Ultra-low-cost computing and developing countries,” Communications of the ACM, Aug. 2013.

[11] K. Hinton, J. Baliga, M. Feng, R. Ayre, and R. S. Tucker, “Power Consumption and Energy Efficiency in the Internet,” IEEE Network, no. April, pp. 6–12, 2011.

[12] H. Hlavacs, G. Da Costa, and J.-M. Pierson, “Energy Consumption of Residential and Professional Switches,” in International Conference on Computational Science and Engineering, 2009.

[13] P. W. Holland and R. E. Welsch, “Robust regression using iteratively reweighted least-squares,” Communications in Statistics - Theory and Methods, pp. 813–827, 1977.

[14] “IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems,” IEEE Std 1588-2008.

[15] M. Jarus, S. Varette, A. Oleksiak, and P. Bouvry, “Performance Evaluation and Energy Efficiency of High-Density HPC Platforms Based on Intel, AMD and ARM Processors,” in Energy Efficiency in Large Scale Distributed Systems, ser. Lecture Notes in Computer Science, J.-M. Pierson, G. Da Costa, and L. Dittmann, Eds. Springer Berlin Heidelberg, 2013.

[16] U. Lee, I. Rimac, D. Kilper, and V. Hilt, “Toward Energy-Efficient Content Dissemination,” IEEE Network, 2011.

[17] J. Nunez-Yanez and G. Lore, “Enabling Accurate Modeling of Power and Energy Consumption in an ARM-based System-on-Chip,” Microprocessors and Microsystems, May 2013.

[18] K. Okiah, C. Finn, P. Leal, and T. Mirabito, “Power Pi - The Green Revolution,” University of Massachusetts Amherst, Tech. Rep., 2012.

[19] B. Raghavan and J. Ma, “The Energy and Emergy of the Internet,” in HotNets, 2011.

[20] Y. Royon and S. Frénot, “Multiservice Home Gateways: Business Model, Execution Environment, Management Infrastructure,” IEEE Communications Magazine, 2007.

[21] M. Seufert, V. Burger, and T. Hoß feld, “HORST - Home Router Sharing based on Trust,” in SETM Workshop, 2013.

[22] F. Tso, D. T. White, S. Jouet, J. Singer, and D. Pezaros, “The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing Infrastructures,” in International Workshop on Resource Management of Cloud Computing, 2013.

[23] B. M. Tudor and Y. M. Teo, “On understanding the energy consumption of ARM-based multicore servers,” in International Conference on Measurement and Modeling of Computer Systems (SIGMETRICS). New York, New York, USA: ACM Press, 2013. [24] V. Valancius, N. Laoutaris, L. Massoulié, C. Diot, and P. Rodriguez, “Greening the Internet with Nano Data Centers,” in CoNEXT, 2009.

[25] L. Zhang, B. Tiwana, R. P. Dick, Z. Qian, Z. Mao, Z. Wang, and L. Yang, “Accurate Online Power Estimation and Automatic Battery Behavior Based Power Model Generation for Smartphones,” in ACM, 2010.

**DECLARATION**

I, **VINAY KARTHIK M B** student of eighth semester BE, Computer Science & Engineering, City Engineering College hereby declare that the project work entitled  **“PowerPi: Measuring and Modeling the Power Consumption of the Raspberry Pi*”***  has been carried out by us at City Engineering College, Bangalore and submitted in partial fulfillment of the course requirements for the award of the degree of **Bachelor of Engineering in Information Science and Engineering of Visvesvaraya Technological University, Belgaum**, during the academic year 2016-2017.

We also declare that, to the best of our knowledge and belief, the work reported here does not from part of any other dissertation on the basis of which a degree or award was conferred on an earlier occasion on this by any other student.

**DATE:**

**PLACE: BANGALORE**

**VINAY KARTHIK M B (1CE13CS124)**