NetSys Technical Report No. NetSys-TR-2018-01, Jan 2018.

The Progress of the Energy-Efficiency of Single-board Computers

Fabian Kaup[1], Stefan Hacker[2], Eike Mentzendorff[2], Christian Meurisch[3], David Hausheer[1]
[1]Institute for Intelligent Cooperating Systems, Otto-von-Guericke Universität Magdeburg
[2]Peer-to-Peer Systems Engineering Group, Technische Universität Darmstadt
[3]Telecooperation Lab, Technische Universität Darmstadt
Email: {kaup|hausheer}@ovgu.de, sh@racksnet.com, eike.mentzendorff@stud.tu-darmstadt.de, meurisch@tk.tu-darmstadt.de

***Abstract*—The demand for high-bandwidth, low-latency services is rapidly increasing. Content Distribution Networks (CDNs) have addressed this by providing content from within or close to the Internet Service Provider (ISP). Still, the most common bottleneck for high service quality is the 'last mile' between ISP and end user. Serving content from small caches on end-user devices promises to increase service quality of the respective content. Similarly, fog computing promises to provide low-latency services from arbitrary nodes within the network. Both require additional functionality provided by network functions virtualization (NFV), redirecting traffic to the appropriate destinations. Still, the cost and performance of possible solutions are not well analyzed. Hence, this paper analyzes the forwarding and computing performance of a number of single-board computers (SBCs) from which models for the performance and energy cost of different loads are derived. Furthermore, the development of energy efficiency gains over the last years is analyzed, confirming Koomey's law also for SBCs, leading to an increase of computational efficiency by a factor of 5.5 to 7.5 over the course of four years, which relates to a doubling time of 1.62 years.**

I. Introduction

The demand for mobile data has increased exponentially and is predicted to further rise over the next years [1]. The majority of this traffic is caused by mobile video consumption [1]. CDNs provide the requested content from within or close to the end-user's ISP. Still, all traffic needs to be handled by cellular and WiFi networks in real-time, being the bottleneck between the end user and the ISP's network [2]. Considering the exponential increase in bandwidth demand, and the comparatively slow upgrade of networks, providing content and advanced network services requiring high bandwidth, low latency, or both from within the end-user promises becomes increasingly attractive, thus reducing the load on these bottleneck links.

Conventional approaches propose using resource rich servers at public locations [3], usually called cloudlets. Still, considering the number of clients, in particular at smaller public or private hotspots, smaller and more energy efficient machines are recommended. These nodes are expected to improve the quality of service (QoS) by e.g. re-routing traffic through a performance enhancing proxy (PEP) [2], thus increasing responsiveness of the network and improving perceived network quality, or serving locally cached content, for which also de-centralized traffic redirection is required. Also, the Internet of Things (IoT) is a prominent candidate for locating computing and storage resources at end-user premises. Large quantities of data and short update intervals suggest aggregating the collected data before sending them via public networks [4].

Currently available hardware at end-user premises is strictly limited in computation and storage capabilities. Extending these by attaching SBCs greatly enhances their capabilities for a reasonable price. Thus it becomes possible to provide virtual network functions (VNFs), serve cached content, or run other de-centralized computing and data aggregation tasks on fog nodes from within the end-users' premises. Still, the power consumption of these approaches, in particular under different loads, is not thoroughly evaluated yet.

Consequently, this publication addresses the need for fine-granular power models for decentralized network service delivery by measuring and modeling the power consumption of SBCs. The analysis is guided by the following questions:

- How can the power consumption of SBCs running VNFs and caches be modeled?
- How has the energy efficiency of SBCs changed since the release of the Raspberry Pi?
- How accurate can the energy cost of a deployed network service be derived using readily available system metrics?

Special emphasis is placed on generating versatile, but accurate power models. Thus, this paper builds the foundation for further analysis and optimization of the energy consumption of various distributed networking, caching, and data processing approaches. Using the presented models, the power consumption of a system can be derived in high accuracy based on readily available system monitoring values only. The practicality of this approach is shown by deriving the power consumption of the RB-HORST system [5] and HTTP streaming as described in Section VI. Furthermore, by analyzing the progress of computational efficiency and energy cost since the release of the Raspberry Pi, an 6-fold increase in energy efficiency over the course of four years is determined, confirming Koomey's law also in the area of ARM-based single board computers.

This paper is structured as follows. Section II discussed related work. Section III describes the Devices under Test (DUTs) and the measurement setup. The measurements executed are described in Section IV. The model generation and resulting models are described in Section V. Section VI compares the generated models. Finally, Section VII summarizes the results and concludes the paper.

II. Related Work

A power model of the Raspberry Pi was presented in [6]. As the Raspberry Pi established the age of low-power ARM computing, this area has seen enormous development over the last years. This trend is supported by the increasing computational capabilities of recent smartphones. Current SBCs often use systems on chip (SOCs) developed for smartphones. This is apparent in the graphical abilities of the platforms, which may not be required for embedded projects.

Malik et al. [7] compare the energy efficiency of an Intel Xeon processor with an Intel Atom processor. The workload consists of different standard big-data, scale-out, and traditional benchmarks. The power consumption of the Intel Atom processor is always lower, but also execution times increase. Still, the energy delay product for all except very large workloads favors the Intel Atom server. Blem et al. [8] analyze the difference in computational efficiency and power consumption between x86 and ARM based instruction set architectures (ISAs). Their analysis is based on a number of benchmarks, from which their execution time and resulting power consumption are derived. The authors conclude that the ISA does not have an influence on the task execution nor energy consumption. Using ARM platforms for high performance computing becomes increasingly popular. Jarus et al. [9] compare the performance and energy efficiency of different high-performance high-density computing platforms. They conclude that the performance per Watt was highest for the ARM Cortex A9, and execution time was lowest compared to the Intel Atom N2600 and the AMD Fusion G-T40N. Lorenzon et al. [10] detail the differences in energy efficiency of SBCs for different idle consumption and numbers of cores. Maqbool et al. [11] analyze the feasibility of using ARM based systems for scientific workloads, while also considering energy efficiency. They conclude that although computational performance is lower, the energy efficiency per calculation is between 2.6 to 4 times higher for ARM based systems. Similar results are confirmed by Zhao et al. [12] for data-center workloads. Contrary to these studies, the focus of this publication is a general purpose power model of SBCs with the goal of estimating the current power consumption based on system monitoring values. Furthermore, the influence of network activity on the power consumption is analyzed, which was not part of the presented studies.

Fajardo et al. [13] propose to augment the mobile edge with storage and computation facilities to form a distributed cloud serving special services with low latency. Their approach proposes to use energy efficient ARM architectures, which are to be extended using special purpose acceleration hardware. The approach also mentions energy efficiency, but lacks a detailed analysis. Here, the proposed models may be used to derive cost metrics. Thus it becomes also possible to assess the benefit over cloud services.

Meurisch et al. [14] analyze the energy efficiency of offloading computationally intensive tasks from the smartphone to the cloud or de-centralized cloudlets, being located at traffic and user-intensive locations. As cloudlet two different hardware platforms were used. One is a conventional PC, the other a high-end WiFi AP/router. Major focus of the evaluation was the energy consumption of the smartphone, which also depends on network bandwidth to the cloud/cloudlet and the respective processing time. Using low-power ARM platforms, which may be integrated into future Access Points (APs), may simultaneously improve performance and reduce latency, while at the same time reducing the system power consumption. Using the power models proposed here, the cost of different workloads (including network transfer) can be calculated.

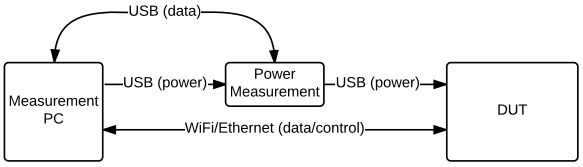
A similar approach is presented by Lareida et al. [5], using SBCs as intelligent caches. Their functionality extends traditional caching functionality by using information from social platforms to derive interest in particular content, which is then pre-loaded to the local device. By making content available locally, the quality of experience (QoE) of the end user is increased due to lower latency and higher local bandwidths. Further, the approach considers global popularity to determine content to be made available locally. The power model derived here is applied to their experimental data to derive the power consumption of the full system under operating conditions.

Another example of the application of Raspberry Pis is the *Glasgow Raspberry Pi Cloud* [15]. Tso et al. built a scale model of a cloud computing infrastructure for research and educational purposes. It emulates all functionality commonly found in large, commercial data centers (DCs), including clustering of machines and networking. Compared to conventional cloud-computing environments, this project allows conducting cloud experiments on a small, restricted environment. The authors evaluate the power consumption roughly, but don't describe a full power model. Using the power models derived within this paper, the scheduling algorithms for the cloud DC may also include the energy cost in their scheduling decisions. Applying the principles learned there, the optimization of full-grown cloud environments may be possible.

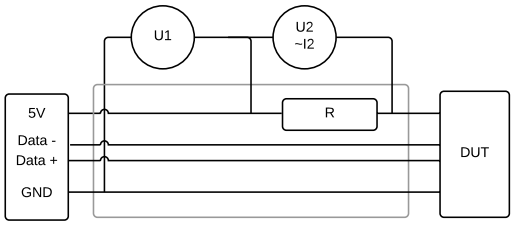
SBCs may also be used in the IoT. Currently, each device connects independently to its respective cloud service, sending measurements and system state for aggregation and receiving control commands. Papageorgiou et al. [4] have identified the potential of aggregating data within the customers' premises to reduce the load on the network and the cloud environment. This is possible, as usually data sensed by the devices is redundant (e.g. multiple temperature/light sensors, high sampling interval). Their approach may benefit from deriving the energy cost and savings from edge-processing compared to cloud processing using the power models presented here.

III. Measurement Setup

The measurement setup extends the one described in [6]. An schematic of the power measurements is given in Figure 1. Contrary to the previous study [6], the goal of this publication is analyzing the evolution of the energy efficiency of SBCs over the last years. Therefore, a number of different platforms were selected. In the following, the DUTs, measurement environment, test, and the accuracy of the measurements are discussed.



(a) Overview of the measurement setup



(b) Power measurement

Fig. 1. Overview of the power and performance measurement setup

A. Description and Comparison of the Devices under Test

While a power model for the Raspberry Pi B is already published in [6], no power models of other SBCs are known to the authors. Hence, the analysis is extended using a number of additional devices (e.g. Raspberry Pi 2, Cubieboard3, Odroid C1, Raspberry Pi 3, Odroid C2). These platforms were selected to represent a range of different vendors, and hence approaches and focus on the purpose and usage of these platforms. The Raspberry Pi 2 can be seen as the direct successor of the Raspberry Pi as a general purpose, low-cost, low-performance platform. Compared to the Pi 1, the processor frequency was slightly improved (700 MHz to 900 MHz), but instead of a single core, the Pi 2 features a quad-core processor. Also the memory was doubled to $1\,\mathrm{GB}$, while keeping the I/Os stable. The Raspberry Pi 3, released exactly one year later features a $64\,\mathrm{bit}$ CPU, but the default OS (which is used here, as it is most commonly run) is still a $32\,\mathrm{bit}$ OS, due to compatibility reasons. The other major difference compared to the Pi 2 is the increased processor frequency of $1.2\,\mathrm{GHz}$. The Ethernet interface of all these devices is Fast Ethernet with a maximum bandwidth of $100\,\mathrm{Mbps}$.

The focus of the Odroids is clearly more on processing and I/O performance, making it a suitable media center or casual gaming machine. The quad-core processor of the C1 with $1.5\,\mathrm{GHz}$ per core provides sufficient performance. The Gigabit Ethernet interface is suitable for streaming high bandwidth content from a remote server. The Odroid C2 shows similar system parameters, but instead of the $32\,\mathrm{bit}$ CPU, a $64\,\mathrm{bit}$ CPU is used. It is possible to run both an $32\,\mathrm{bit}$ and $64\,\mathrm{bit}$ system on these as done for the Raspberry Pi 3, but for the C2 only a $64\,\mathrm{bit}$ OS is available.

The Cubieboard 3, also known as Cubietruck, is more focused on communication, with built in Bluetooth and WiFi chip sets. Further, the Cubieboard3 has the largest memory of the analyzed devices and a SATA port, making a suitable low-power home-server. With a dual-core processor with $1\,\mathrm{GHz}$, a reasonable performance can be expected. Using Gigabit Ethernet or WiFi, the Cubieboard 3 is well connected and may also serve as local WiFi AP.

An overview of these platforms is given in Table I.

B. Measurement Environment

The measurement setup is an extended version of the setup described in [6]. The main differences are support for measuring and controlling the system utilization of multi-core systems, and the further automation of the tests.

The DUTs were connected to the power supply via a custom built power measurement board (cf. Fig 1a). The power consumption is measured using Measurement Computing's USB1608-FSPlus[1], a 16 bit, simultaneous sampling measurement card. Configuration and test execution are controlled via Ethernet or WiFi, depending on the test requirements.

Figure 1b details the schematics of the power measurement board. A precision $0.1\,\Omega$ resistor ($1\,\%$) is inserted into the $5\,\mathrm{V}$ line, and provides connectors to measure the supply voltage ($U1$) and the voltage $U2$ over the measurement shunt R, which is proportional to the current $I2$ consumed by the DUT. The measurements are recorded with a sampling rate of $10\,\mathrm{kS/s}$. The data is recorded using a custom software interfacing with the measurement library provided by Measurement Computing. Based on the electrical configuration and the resistance of the measurement shunt, this software calculates the power consumption of the DUT, averages the measurements over the course of one second, and writes a CSV-file for later analysis.

The measurement PC is a laptop with a Gigabit network interface, a quad-core processor, and sufficient memory. Hence, the influence of this machine on the measurements is expected to be negligible. This machine is used to control the load tests, measure and record the power consumption, and act as remote end-point for network throughput measurements. Test execution is controlled using SSH from the measurement PC. Still, as some tests require the absence of any network connection, the measurement and monitoring scripts were designed to run in the background and start only after a short interval, allowing to adjust the system configuration (e.g. remove Ethernet cable). The measurements were conducted in a residential neighborhood. This is particularly important for the WiFi measurements. There, an empty channel was selected and measurements run during night, further reducing the probability of interference.

C. Description of the Test Suite

The tests run on the DUT comprise CPU benchmarks, and Ethernet and WiFi throughput measurements, both in up- and down-link direction. Tests of the influence of RAM utilization on the power consumption showed no effect and are consequently left out of this analysis.

The CPU benchmarks run a loop adding numbers either on a single core, or starting the same task on multiple cores in parallel. The CPU utilization is limited using *cpulimit*. Tasks were further pinned to the respective CPU core. Possible

[1] http://www.mccdaq.com/usb-data-acquisition/USB-1608FS-Plus.aspx, accessed 2016-04-20

TABLE I
OVERVIEW OF THE DUTS

	Raspberry Pi B	Cubieboard3	Odroid C1	Raspberry Pi 2 B	Odroid-C2	Raspberry Pi 3 B
Release Year	2/2012	10/2012	12/2014	2/2015	02/2016	02/2016
CPU Type	ARM1176JZF-S	ARM Cortex-A7	ARM Cortex-A5	ARM Cortex-A7	ARM Cortex-A53	ARM Cortex-A53
Instruction Set	ARMv6	ARMv7	ARMv7	ARMv7	ARMv8	ARMv8
Number cores	1	2	4	4	4	4
Processor Frequency	700 MHz	1 GHz	1.5 GHz	900 MHz	1.5 GHz	1.2 GHz
RAM	512 MB	2 GB	1 GB	1 GB	2 GB	1 GB
Ethernet	FE	GbE	GbE	FE	GbE	FE
WiFi	TL-WDN3200	BCM43362	TL-WDN3200	TL-WDN3200	—	BCM43438

inaccuracies of the load-limiting script are mitigated by constantly monitoring the CPU utilization using the *proc* file system. Thus, the measured power consumption can be correlated to the actual CPU utilization in a given interval. The influence of the CPU utilization is measured in the range from $10\,\%$ to $100\,\%$ with a step size of $10\,\%$.

The throughput measurements were conducted using *iperf* in UDP mode. This assures that no congestion control influences the accuracy of the network throughput measurements. Furthermore, as packages do not need to be acknowledged, the influence of the remote system can be neglected. Additionally, as the measurement PC is much more capable than the DUT, data rates received by the SBC are the maximum the network interface and the overall system can process.

The Ethernet measurements were conducted by directly connecting the network interfaces of the two devices. The measurement task is started with a short delay on both sides, eliminating the influence of control traffic and system utilization on the measurement accuracy. The power consumption of the Ethernet interface is evaluated in the range from $10\,\mathrm{Mbps}$ to $100\,\mathrm{Mbps}$ for the Raspberry Pis, and $1\,\mathrm{Gbps}$ for the other platforms, all with a granularity of $10\,\mathrm{Mbps}$ below, and $100\,\mathrm{Mpbs}$ above $100\,\mathrm{Mbps}$.

The WiFi measurements are conceptually similar, but one end must implement the AP functionality. The AP was configured to run on the measurement PC using *hostapd*, thus minimizing negative impacts on the DUT and thus on the measurement accuracy. Contrary to the Ethernet measurements, the WiFi measurements are controlled via WiFi and SSH, thus eliminating the influence of an active Ethernet interface on the power consumption of the DUT. The power consumption of the WiFi interface is evaluated in the range from $10\,\mathrm{Mbps}$ to $100\,\mathrm{Mbps}$ with a granularity of $10\,\mathrm{Mbps}$.

D. Description of the measurement procedure

The measurement procedure follows an incremental approach. First, the power consumption of the idle system is measured. Then, the influence of CPU activity is evaluated, allowing to later eliminate the influence of the CPU consumption from the component power consumption. Lastly, each component is stressed individually, while at the same time the system utilization (i.e. CPU) is monitored. Based on these measurements, the power consumption of the individual components is calculated using a regression based approach.

The idle power consumption is measured by just booting the DUT and measuring the power consumption. Care is taken that only the Linux kernel with the minimal number of required services is running. Also, the network interfaces are shut down and the Ethernet cable is disconnected, thus minimizing the power consumption of the idle system. Still, the system monitoring scripts are active, allowing to later calibrate the CPU model and component power consumption. This measurement runs for $900\,\mathrm{s}$.

The CPU power consumption is measured by running a script stressing the CPU while simultaneously limiting the CPU utilization using *cpulimit*. The load generator, load limiter, and system monitoring are started together using a remote shell. These are started as delayed, detached processes, allowing sufficient time to disconnect the remote shell and Ethernet cable. Thus, negative effects caused by automatic network requests can be avoided. Simultaneously, the power measurement on the measurement PC is started. These scripts measure each CPU utilization level for a duration of $900\,\mathrm{s}$. Thus, the inevitable inaccuracies of the load limiter can be limited to a minimum.

The Ethernet and WiFi measurements are conducted in a similar fashion. Instead of stressing the CPU, *iperf* is started as either client or server and data is sent to, or received by the measurement PC. The system monitoring is extended to also record the network utilization by reading the bytes received and transmitted on the active interface from the */proc* file system. The CPU utilization is recorded for all measurements, thus allowing to later eliminate the influence of the CPU utilization using the previously calibrated model. This step is required to create a simple, additive power model which is based on the system utilization only. Hence, no interdependency between network traffic and CPU utilization needs to be considered when applying the final models.

All measurement data is first collected in RAM, eliminating the influence of either network or SD-Card activity on the power measurements. Therefore, a RAM disk is created, where the CPU and network utilization logs are stored. Only after the measurement has finished, the collected data is written to the SD-Card for later collection and analysis.

The measurement duration of $900\,\mathrm{s}$ was chosen, as a compromise between accuracy and measurement duration. Iterating

over 10 system states in the case of the CPU measurements, or 40 in the case of Ethernet or WiFi up- and downloads results in a measurement duration of 2.5 and 10 hours respectively. By automating these, the measurements of a single device can be completed within a day by running the idle, CPU, and Ethernet measurements during the day, while restricting the WiFi measurements to the night where interference is expected to be low. Here it must be considered that after each group of measurements (e.g. idle, CPU, Ethernet, WiFi) manual interaction is required to adjust the desired system state (e.g. connect/disconnect Ethernet/WiFi).

E. Discussion of the measurement accuracy

The accuracy of the network measurements, and thus the elimination of error sources, was of considerable interest during design and execution of the experiments. As already discussed, considerable care was taken to assure that the system state, monitoring, and load-generating scripts are as accurate as possible. The accuracy of the final power models mainly depends on the measurement accuracy. Hence, a 16 bit measurement card was selected to sample the current and voltage at the power supply. Additionally the sampling rate was set to the maximum supported sampling rate (10 kHz). The accuracy is the same as derived in [6]. The maximum power drawn by the DUTs does not influence the measurement accuracy, as the same range is used and the measurement error is bound by the resolution supported by A/D converter and the minimum currents and voltages measured. Hence, the error of the measurements is expected to not exceed 2.5 %.

IV. Description of the Measurements

The following section first describes the measurements derived of the DUTs. Further, the power consumption of the USB WiFi dongle is evaluated as described in Section III.

A. Measurement of the SBCs

Figure 2 shows the power consumption of the SBCs as caused by different CPU utilization on the measured platforms. The power model of the Raspberry Pi B is taken from [6]. As the relative CPU utilization is plotted, no further adjustments are required to allow a comparison with other platforms. For low utilization, the Cubietruck show the lowest energy consumption, while for higher utilization the Raspberry Pi consumes the least energy. This is expected, as it also has the slowest processor of the DUTs. The highest energy consumption with 2.2 W is visible for the Odroid C1 under high utilization, while the original Raspberry Pi is most expensive when idle.

The Ethernet measurements are shown in Figures 3a and 3b. As the Raspberry line is equipped 100 Mbps capable interfaces these graphs end earlier. Both Odroids, and the Cubietruck support Gbps interfaces, but cannot fully saturate these. The Cubietruck achieves data rates up to 80 Mbps, while the Odroids achieve rates of 400 Mbps to 1000 Mbps. Most expensive is the Ethernet interface on the Odroid C2. Still as this is a more energy efficient device, this effect cancels out when in use. The negative values of the Pi 3 network

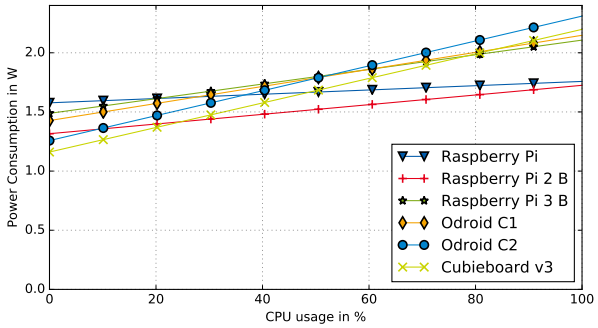


Fig. 2. Comparison of the influence of the CPU on the power consumption

model are caused by periodic CPU activity while no Ethernet is connected, but cancel out when the network is active.

B. Measurements of WiFi Dongles

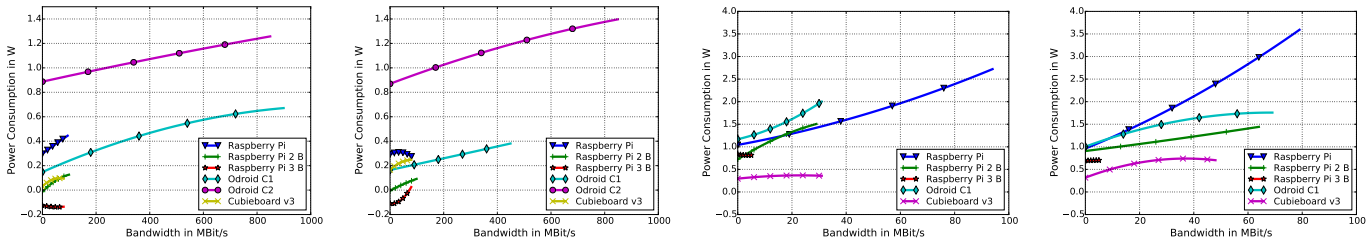
The WiFi throughput measurements for the Raspberry Pi 1 and 2, and the Odroid C1 were conducted using the same WiFi dongle (TP-Link TL-WDN3200). Hence, the influence of the power and throughput measurements must be attributed both to the SOC and the USB chip set. On the Cubietruck and Pi 3 the onboard WiFi is used.

A comparison of the power consumption of the different platforms is given in Figures 3c and 3d. The power consumption of WiFi connections is similar for the Raspberry Pi 1, Pi 2, and Odroid C1, using the TP-Link WiFi dongle. Here it must be noted that the WiFi measurements on the Raspberry Pi were conducted without encryption, thus explaining the high achieved data rates. On the Raspberry Pi 3 and the Cubietruck the integrated WiFi was used. The performance on the Cubietruck was good, also showing the smallest additional energy consumption. On the Raspberry Pi 3, the maximum measured data rate is 6 Mbps, with a cost similar to the external WiFi dongle.

V. Model Generation

The measurements as described in Section III are used to generate aggregate power models of the DUTs. For this, a regression based approach is used. All processing and plotting is conducted in Python using the *sklearn* library for a robust regression on the measurement data. The analysis is conducted analog to [6]. Similar to the measurements of the Raspberry Pi, each measurement and the corresponding regression was plotted and the resulting function, including the root mean square error (RMSE) written to a file.

Figure 4 shows an exemplary measurement used to model the device's power consumption. The bandwidth was measured for data rates from between 10 Mbps and 1 Gbps. The resulting power consumption over the measured network throughput is given as a heat map with the density of color increasing for a larger number of samples at a given point. Still, there are differences visible between target rates and measured rates. This is expected as *iperf* generates packets with a given payload, while the interface traffic stats return bytes on the wire. For



(a) Ethernet download (b) Ethernet upload (c) WiFi download (d) WiFi upload

Fig. 3. Comparison of the power consumption of data transfers

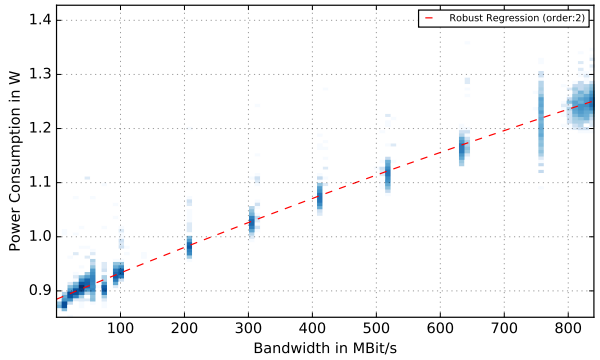


Fig. 4. Measurement and model (Odroid C2, Ethernet down, 2^{nd} order fit)

measurements with higher data rates an interesting effect is observed. A large cluster of values is visible in the range of $830\,\mathrm{Mbps}$. Here, the maximum transfer capacity was reached.

Table II summarizes the generated models and the resulting RMSE for the measured devices. Similar to [6], the variances in the collected data are low. Based on the power models of the Raspberry Pi B, first order models are chosen for the CPU utilization, while second order models are identified to be sufficient for modeling the network induced power consumption of SBCs under load.

To apply the power models as given in Table II, an additive power model is used. The model

$$P = P_{\mathrm{idle}} + P_{\mathrm{cpu}}(u) + \sum_{\mathrm{if}} \left(P_{\mathrm{if,idle}} + P_{\mathrm{if,up}}(x) + P_{\mathrm{if,dn}}(x)\right) \tag{1}$$

consists of the idle power P_{idle} of the respective device and its load dependent power consumption $P_{\mathrm{cpu}}(u)$, plus the consumption caused by any active interfaces. Here, the CPU utilization is relative to the system load in the range $[0, 1]$. P_{if} is the idle power of the respective interface as defined in Table II. The influence of upload traffic ($P_{\mathrm{if,up}}$) and download traffic ($P_{\mathrm{if,dn}}$) is defined by the polynomials in Table II. The possible data rates r (in Mbps) are defined by the graphs in Figure 3.

VI. Evaluation of the Power Models

Considering the measurements as derived in Section IV, the power consumption of the DUTs is derived. This section relates these to the performance as derived from the hardware specifications and measured during the study.

A. Assessment of the Model Accuracy

The accuracy of the generated models is exemplary assessed in a bursty video streaming scenario as is common for cloud based video streaming. The models are validated for the use case of a decentralized cache. Thus, a web server is installed, serving traffic to a remote device over the Ethernet interface. This interface is selected, as it is available on all DUTs without requiring additional hardware. The web server (nginx[2]) serves bursty traffic with a simple PHP script. This script creates traffic according to the request parameters absolute content size, average bit rate, and burst interval. Thus, the traffic characteristics of mobile video streaming are simulated.

Reference measurements emulate Full-HD video streaming with a bitrate of $1\,\mathrm{MB/s}$ The request interval was chosen to be $10\,\mathrm{s}$. In a conventional streaming setup, this would vary depending on the video bitrate. For simplicity the interval was chosen to be static, but no difference in the final outcome is expected. The size of the served content was configured to fill the measurement interval of $300\,\mathrm{s}$.

The test was repeated for all DUTs. These serve content from to a client requesting the content. During the test, both the external power consumption and the system performance monitoring are active. Based on these, the difference is estimated and measured power consumption calculated.

Table III summarizes the results of the reference measurements. For all devices, a relative error smaller than $10\,\%$ is achieved. For the Odroids, accuracies below $2\,\%$ are observed. Considering the modeling to be based on a comparatively coarse granularity of $1\,\mathrm{s}$, the results are quite accurate. Particularly, the effects of short variations in system load on the power consumption are difficult to include in any model. Thus, it can be argued that a good trade-off between accuracy, availability of system monitoring values, and ease of use was found.

B. Performance Developments of SBCs

The computational efficiency of the tested SBCs is in the following compared with numbers derived from literature. Here, Koomey's law [16] is a prominent example of analyzing and interpreting the development of computational efficiency.

[2]https://nginx.org/, accessed 2016-11-18

TABLE II
OVERVIEW OF THE GENERATED POWER MODELS

Function	Model	RMSE
	Rasberry Pi B	
P_{idle}	$= 1.577$	
$P_{eth,idle}$	$= 0.294$	
$P_{wlan,idle}$	$= 0.942$	
$P_{cpu}(u)$	$= 0.181u$	0.016
$P_{eth,up}(r)$	$= -15.2e^{-6}r^2 + 1.005e^{-3}r - 0.002$	0.008
$P_{eth,dn}(r)$	$= -6.531e^{-6}r^2 + 1.6344e^{-3}r + 0.003$	0.017
$P_{wlan,up}(r)$	$= 112.8e^{-6}r^2 + 24.386e^{-3}r + 0.020$	0.071
$P_{wlan,dn}(r)$	$= 71.988e^{-6}r^2 + 11.003e^{-3}r + 0.010$	0.026
	Raspberry Pi 2 B	
P_{idle}	$= 1.316$	
$P_{eth,idle}$	$= -0.019$	
$P_{wlan,idle}$	$= 0.899$	
$P_{cpu}(u)$	$= 0.409u$	0.038
$P_{eth,up}(r)$	$= 1.95e^{-06}r^2 + 1.17e^{-03}r + 0.014$	0.014
$P_{eth,dn}(r)$	$= -8.54e^{-06}r^2 + 2.25e^{-03}r + 0.006$	0.012
$P_{wlan,up}(r)$	$= 13.4e^{-03}r^2 + 7.50e^{-03}r + 0.008$	0.043
$P_{wlan,dn}(r)$	$= -0.37e^{-03}r^2 + 37.72e^{-03}r - 0.176$	0.081
	Raspberry Pi 3 B	
P_{idle}	$= 1.488$	
$P_{eth,idle}$	$= -0.1176$	
$P_{wlan,idle}$	$= 0.7645$	
$P_{cpu}(u)$	$= 0.6191u$	0.155
$P_{eth,up}(r)$	$= 26.2^{-06}r^2 + 0.357e^{-03}r + 0.007$	0.039
$P_{eth,dn}(r)$	$= -4.33e^{-06}r^2 + 0.485^{-03}r - 0.007$	0.007
$P_{wlan,up}(r)$	$= -0.25e^{-06}r^2 + 1.99e^{-03}r - 0.072$	0.013
$P_{wlan,dn}(r)$	$= 1.85e^{-03}r^2 + -13.5e^{-03}r + 0.072$	0.022
	Cubietruck	
P_{idle}	$= 1.161$	
$P_{eth,idle}$	$= 0.224$	
$P_{wlan,idle}$	$= 0.306$	
$P_{cpu}(u)$	$= 1.037u$	0.038
$P_{eth,up}(r)$	$= -17.6e^{-06}r^2 + 6.13e^{-03}r - 0.056$	0.057
$P_{eth,dn}(r)$	$= -20.9e^{-06}r^2 + 2.50e^{-03}r + 0.056$	0.024
$P_{wlan,up}(r)$	$= -0.307e^{-03}r^2 + 22.8e^{-03}r + 0.011$	0.178
$P_{wlan,dn}(r)$	$= 0.137e^{-03}r^2 + 6.33e^{-03}r - 0.011$	0.064
	Odroid C1	
P_{idle}	$= 1.427$	
$P_{eth,idle}$	$= 0.156$	
$P_{wlan,idle}$	$= 1.086$	
$P_{cpu}(u)$	$= 0.721u$	0.023
$P_{eth,up}(r)$	$= 20.1e^{-09}r^2 + 0.47e^{-03}r + 0.008$	0.062
$P_{eth,dn}(r)$	$= -0.44e^{-06}r^2 + 0.97e^{-03}r - 0.008$	0.019
$P_{wlan,up}(r)$	$= -0.16e^{-03}r^2 + 22.0e^{-03}r - 0.082$	0.126
$P_{wlan,dn}(r)$	$= 0.41e^{-06}r^2 + 14.0e^{-03}r + 0.082$	0.086
	Odroid C2	
P_{idle}	$= 1.258$	
$P_{eth,idle}$	$= 0.878$	
$P_{cpu}(u)$	$= 1.052u$	0.051
$P_{eth,up}(r)$	$= -0.24^{-06}r^2 + 0.83e^{-03}r - 0.009$	0.012
$P_{eth,dn}(r)$	$= 58.0e^{-09}r^2 + 0.48e^{-03}r + 0.009$	0.010

TABLE III
EVALUATION OF THE ACCURACY OF THE POWER MODELS

Device	$P_{e,max}$	$\overline{P_e}$	RMSE	$\overline{P_{e,rel}}$
Pi 1	0.282	-0.137	0.141	6.5%
Pi 2	0.098	0.031	0.036	2.5%
Pi 3	0.718	-0.195	0.275	9.3%
CT	0.346	0.002	0.112	6.0%
C1	0.121	-0.002	0.032	1.0%
C2	0.281	-0.016	0.076	1.7%

Historically, the number of computations achieved per kWh of consumed energy are compared. In his study Koomey derives an exponential dependency between year and computational efficiency, observing an average doubling time of $1.52\,\mathrm{years}$.

Hence, the computational efficiency of the DUTs is analyzed in a similar way. The power consumption during full CPU utilization periods (excluding network and other components) is noted and compared with the CPU frequency multiplied by the number of cores. The derived values are compared in Figure 5. Generally, an increasing trend is visible over the last 4 years. Most energy efficient is the Odroid C1, followed by the Odroid C2 as one of the latest models evaluated. Least energy efficient clearly is the original Raspberry Pi. Compared to the Odroids, the Raspberry Pi are less energy efficient, and considering Table I, less potent.

Similar to [16], Figure 5 shows an exponential dependency between year and computational efficiency. Compared to [16], the coefficient of determination is smaller, being caused by the smaller number of samples available in a still young field of devices. The exponential fit shows a growth rate of 0.428, which is quite close to the rate determined by Koomey (0.456), leading to a doubling time of 1.62 years (compared to 1.52 over all computers), or 19 months. Still, comparing the achieved values with the numbers given in [16], an offset between both functions is visible. Focus of both x86/64 and ARM processors currently is the improvement of energy efficiency. For x86/x64 processors, the majority of optimizations appears to be on increasing the computational efficiency during active periods, e.g. improving pipelining and and reducing energy consumption while idle. Contrary on ARM processors, energy efficiency is achieved by implementing deep sleep modes and disabling cores while idle. This is owed to their current main application in smartphones, where the processor is only active during comparatively short usage periods, while x64/64 processors are usually active for longer times, and consistently low response times are required. Still, all processors show improved energy efficiency over time, relating to lower device fabrication nodes.

VII. SUMMARY AND CONCLUSION

The power consumption of SBCs was measured, from which models mapping the system utilization to power are derived. Thus, the power consumption of live systems providing VNFs at end-user premises as proposed in fog computing, can be calculated based on monitored system utilization. These

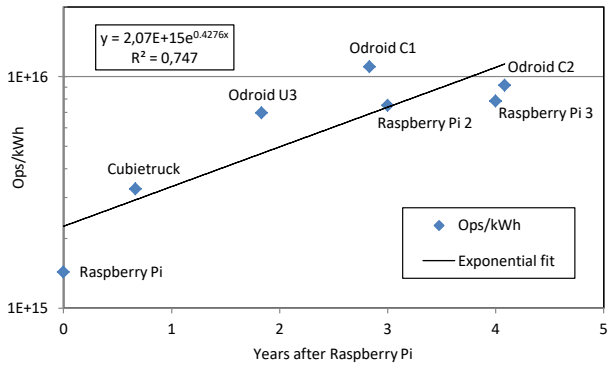


Fig. 5. Comparison of the energy efficiency of different SBCs over time

models are also applicable to emerging networking approaches, extending the information base for future applications.

In the beginning, three questions were posed, which are answered in the following:

How can the power consumption of SBCs running VNFs and caches be modeled? The power consumption of SBCs can be modeled by simultaneously recording the system utilization and the power consumption of the DUTs, to then run a regression analysis on the collected data. Here, special consideration must be paid to the accuracy of the measurement setup, running processes, and general system configuration, thus minimizing the influence of errors, and consequently maximizing the accuracy of the generated model.

How has the energy efficiency of SBCs changed since the release of the Raspberry Pi? The absolute idle and network related power consumption of SBCs has remained comparable over the last years. Still, the computational efficiency has doubled every 1.62 years and network throughput has increased considerably over the course of four years. The increase in computational efficiency correlates well with Koomey's law, describing a doubling time of 1.52 years. Based on these observations, and the ongoing trend of increasing capabilities of modern smartphones, further improvements in computational efficiency are expected.

How accurate can the energy cost of a deployed network service be derived using readily available system metrics? The power consumption of the modeled SBCs can be derived by monitoring the system utilization using simple calls to the unix */proc* file system and converting these to power consumption using the presented models. Reference measurements show errors in the range of 1 % to 10 %. The energy efficiency is mainly influenced by CPU capabilities. Depending on the used instructions, the gain of using a more recent CPU is expected to be even larger than shown in Section VI.

Concluding, the evolution of the power consumption of the Raspberry Pi, Pi 2, and Pi 3 was compared to alternative platforms. The measurements show considerable increases in energy efficiency over the last years, approximately doubling every 1.62 years. Thus, by adding computational resources at end-user premises, the promise of fog computing can be put into practice, and performance gains and respective costs accurately determined using the presented power models.

Future work will focus on mapping the requirements of various types of network services to system utilization values, thus providing means to determine their energy consumption based on end-user or network requirements only.

Acknowledgments

This work has been supported in parts by the European Union (FP7/#317846, SmartenIT) and the LOEWE initiative (Hessen, Germany) within the NICER project. The authors would also like to acknowledge valuable comments by their colleagues and project partners.

References

[1] "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015-2020," Cisco, Tech. Rep., 2016. [Online]. Available: http://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html

[2] K. Liu, J. Y. B. Lee, and S. Member, "On Improving TCP Performance over Mobile Data Networks," *IEEE Transactions on Mobile Computing*, vol. 15, no. 10, pp. 2522–2536, 2016.

[3] M. Satyanarayanan, P. Bahl, R. Cáceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[4] A. Papageorgiou, B. Cheng, and E. Kovacs, "Real-Time Data Reduction at the Network Edge of Internet-of-Things Systems," in *IEEE/IFIP Conference on Network and Service Management (CNSM)*, 2015.

[5] A. Lareida, G. Petropoulos, V. Burger *et al.*, "Augmenting Home Routers for Socially-Aware Traffic Management," in *IEEE Conference on Local Computer Networks (LCN)*, 2015.

[6] F. Kaup, P. Gottschling, and D. Hausheer, "PowerPi: Measuring and Modeling the Power Consumption of the Raspberry Pi," in *IEEE Conference on Local Computer Networks (LCN)*, 2014.

[7] M. Malik and H. Homayoun, "Big Data on Low Power Cores - Are Low Power Embedded Processors a good fit for the Big Data Workloads?" in *IEEE International Conference on Computer Design (ICCD)*, 2015.

[8] E. Blem, J. Menon, and K. Sankaralingam, "Power Struggles: Revisiting the RISC vs . CISC Debate on Contemporary ARM and x86 Architectures," in *IEEE International Symposium on High Performance Computer Architecture (HCPA)*, 2014.

[9] 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, vol. 8046, pp. 182–200.

[10] A. F. Lorenzon, M. C. Cera, and A. C. S. Beck, "On the influence of static power consumption in multicore embedded systems," in *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2015.

[11] J. Maqbool, S. Oh, and G. C. Fox, "Evaluating ARM HPC clusters for scientific workloads," *Concurrency and Computation: Practice and Experience*, vol. 27, pp. 5390–5410, dec 2015.

[12] Y. Zhao, S. Li, S. Hu *et al.*, "An experimental evaluation of datacenter workloads on low-power embedded micro servers," *Proceedings of the VLDB Endowment*, vol. 9, no. 9, pp. 696–707, 2016.

[13] J. O. Fajardo, F. Liberal, I. Giannoulakis *et al.*, "Introducing Mobile Edge Computing Capabilities through Distributed 5G Cloud Enabled Small Cells," *Mobile Networks and Applications*, vol. 21, pp. 564–574, 2016.

[14] C. Meurisch, A. Seeliger, B. Schmidt *et al.*, "Upgrading Wireless Home Routers for Enabling Large-Scale Deployment of Cloudlets," in *EAI International Conference on Mobile Computing, Applications and Services (MobiCase)*, 2015.

[15] F. P. Tso, D. R. White, S. Jouet *et al.*, "The Glasgow Raspberry Pi Cloud: A Scale Model for Cloud Computing Infrastructures," in *IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW)*, 2013.

[16] J. G. Koomey, S. Berard, M. Sanchez, and H. Wong, "Implications of Historical Trends in the Electrical Efficiency of Computing," *IEEE Annals of the History of Computing*, vol. 33, no. 3, pp. 46–54, 2011.