Estimating Energy Efficiency of Data-Link Layer in System Level Performance Evaluation

Subayal Khan, Jukka Saastamoinen, Jyrki Huusko and Juha Korpi
VTT Technical Research Centre of Finland
Oulu, Finland
e-mail: {firstname.lastname}@vtt.fi

Jari Nurmi
Department of Computer Systems
Tampere University of Technology
Tampere, Finland
e-mail: jari.nurmi@tut.fi

Abstract—**Modern distributed embedded systems are composed of a number of mobile devices, which have limited battery life. The design of distributed embedded systems is therefore challenging and data-link layer plays an important role in end-user experience by ensuring reliable error free communications. Both the non-functional properties such as end-end delays and frame error rate, and energy consumption of the data-link and upper layer protocols must be thoroughly investigated in order to ensure optimal distributed system design. To achieve this goal, we propose a novel framework to estimate the energy consumption and non-functional properties of the Medium Access Control (MAC) protocols. In this article we elaborate the extensions made for Abstract workload based performance simulation (ABSOLUT) System Level Performance Evaluation (SLPE) approach and the corresponding methodology to estimate energy consumption of IEEE 802.11 family MAC protocol through a case-study in UDP/IP transmission.**

Keywords-Data-link; ABSOLUT; System Level Performance Evaluation; Distributed Systems; Energy Consumption

I. INTRODUCTION

System Level performance evaluation, has been approached in various ways. A detailed state of the art survey is provided by Khan et al. [1] and is therefore not presented here. In many embedded systems domains such as wireless sensor networks, devices are limited by battery and computation capacity. To operate successfully under these constraints, the wireless devices mostly employ highly specialized MAC protocols such as [1] and [2]. For highly accurate System Level Performance Exploration (SLPE), the system designer must be provided with highly accurate models of applications, middleware, transport, MAC and platform hardware components (such as processors, busses and memories). Different MAC protocol models can be used in the performance model in different iterations of the performance exploration. This will help to evaluate their feasibility before the design and deployment of the devices involved in the distributed system.

In lucrative mobile devices market segments, such as mobile phones and tablets the middleware technologies, application design methodologies, multi-core processors, cloud computing and application specific processors have played a key role in maintaining the good customer perception of the strongest brands like Samsung and Apple [3][4]. The reliable and highly accurate energy consumption of MAC protocols in different use-cases is of fundamental importance since most of the applications used by these devices are distributed [1] [2].

After the performance simulation, both the performance (non-functional properties such as end-end delays and frame error rate) [5] and energy consumption of the chosen MAC protocol must be reported simultaneously for a particular iteration during architectural exploration. This helps to evaluate the feasibility of a MAC protocol (non-functional properties, which must be satisfied by MAC protocol) under the energy and power constraints. After, the simulation, the values of non-functional properties (such as end to end delays and frame errors, etc.) are examined to ensure that their values are under the maximum allowable values for the current use-case. In case of energy consumption, the goal is to find the sole contribution of the employed MAC protocol in the energy consumed by the hardware components of the platforms of devices, which constitute the modelled distributed system.

The aim of the research presented in this article is to measure/estimate the energy consumption due to the MAC/transport protocols via highly accurate and functionally correct state machines models. The functionality of the MAC protocol models mimics the behaviour of modelled MAC accurately to provide reliable estimates of end-end frame delays, loss rates and delays. The highly accurate application workload models representing software implementation of MAC protocols are obtained via ABSINTH-2 [6], PAPI [7], or CORRINA [8]. These workload models [9] provide reliable estimates of busy times of the underlying platform entities due to modelled software implementations of corresponding MAC protocols.

The workload models are combined with the correct behavioural state machine models of contention resolution algorithms employed by MAC protocols [5]. The application models can be abstracted out in ABSOLUT via traffic generators to model a variety of use cases and helps to abstract out the workload of applications, thus providing the utilization (busy) times of platform components solely due to MAC protocols implementation [5].

Once the energy efficiency of the targeted MAC protocols is evaluated in isolation, the actual application workload models of applications are substituted instead of traffic generators. This results in the overall system level performance evaluation of complete distributed embedded systems, as described in [10] [11]. This way, the overall performance evaluation of the distributed system is performed. In this article, we only focus on the energy consumption of the MAC protocols. The methodology presented in this article is described via a case study.

The approach is not limited to any particular data-link protocol or application/system domain. The approach allows the system designer to freely choose between the analytical power models in research for different platform components or any/all of average power, minimum and maximum power values of the platform components.

The rest of the paper is organized as follows: Section II focuses on the related research and identifies the main focus areas of the previously conducted research. This helps to elaborate the importance and uniqueness of the research contribution presented in this article. Section III gives a brief overview of the ABSOLUT performance simulation approach. Section IV and Section V collectively describe the methodology and the analytical models used to estimate the energy consumption of the platform components due to the targeted MAC protocol. In Section VI, the approach is experimented via a case study. Finally the conclusions are presented in Section VII.

II. RELATED WORK

The deployment of energy efficient MAC protocols is important in many embedded systems domains such as wireless sensor networks and mobile devices. In wireless devices, the employed MAC protocols must deliver satisfactory performance in terms of for example end-to-end delays and frame loss rate [5] under the energy constraints due to limited battery [1][2].

So far, the related research has been focused on three main areas in the development of new energy efficient MAC protocols [1][2] such as the software and hardware based energy consumption techniques [12], simulation frameworks for comparing energy saving of MAC protocols [13] and analytical models for estimating the energy consumption of specific MAC protocols such as Y-MAC [14]. The tools and methodologies developed as a result of research contributions do not provide a seamless progression between two steps, i.e., between the selection of energy efficient MAC protocols (under the different use cases) and the system-wide performance evaluation. In the first step, the Application workload models are abstracted out by traffic generators to focus solely on the performance of MAC protocols as described by Khan et al. [5]. This results in choosing the most efficient MAC protocol among different alternatives. Once the MAC protocol is selected, the second step proceeds. In the second step, the actual application workload models are employed (and traffic generators are removed) to evaluate the feasibility of the overall system. In order to obtain reliable performance numbers in both steps, the methodology employs the following important techniques/tools:

1. The highly accurate models of contention resolution techniques, as described by Khan et al. [5].
2. Automatic workload Extraction for MAC protocols via ABSINTH [9] or ABSINTH-2 [6] in the cases where a software implementation is available.
3. In the case where software implementations are not available, it can be estimated via transport layer workload models via CORRINA [8].
4. In order to focus on MAC protocols entirely, the workload models of application layer can be abstracted out, as described by Khan et al. [5].
5. The busy times of the platform hardware components such as processors, memories and busses can be obtained solely due to MAC layers and used to estimate energy consumption via the specifications of the hardware components provided by vendors, such as ARM Holdings [15].

The energy estimation of data-link/MAC protocols via ABSOLUT is fundamentally important to many domains of distributed embedded systems such as wireless sensor networks and mobile devices. During the early design phase, the energy consumption of MAC protocol can be estimated in a variety of use-cases, which helps in the selection of the most appropriate MAC protocol out of a number of available alternatives. This results in a more optimal distributed system design.

III. INTRODUCTION TO ABSOLUT

The abstract workload based performance simulation (ABSOLUT) approach has been extensively applied for the performance simulation of non-distributed and distributed embedded systems. It follows the Y-chart model [1], consisting of application workloads and platform model [9]. After mapping the workloads to the platform, the models are combined for transaction-level performance simulation in SystemC. Based on the simulation results, we can analyse e.g. processor utilization, memory traffic and execution time. The approach enables early performance evaluation, exhibits light modelling effort, allows fast exploration iteration and reuses application and platform models [9].

A. *Application Workload Model*

The application workload model has a layered architecture as explained by Kreku et al. [9]. The hierarchical structure of the application workload model is shown in Fig. 1.

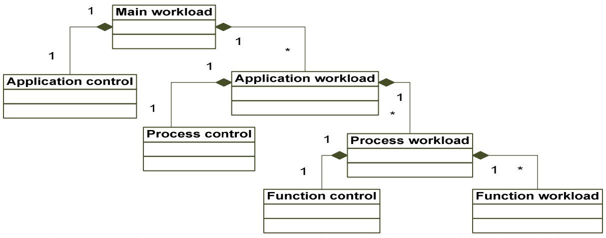

Figure 1. Hierarchical structure of application workload model.

B. *Platform Model*

The platform model is an abstract hierarchical representation of actual platform architecture. It is composed of three layers: component layer, sub-system layer and platform architecture layer, as shown in Fig 2.

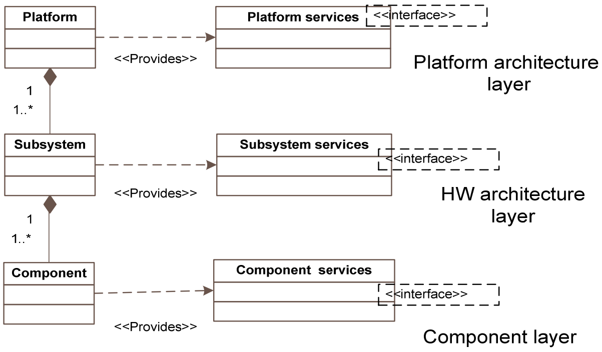

Figure 2. Platform model layers.

Each layer has its own services, which are abstract views of the architecture models. Services in sub-system and platform architecture layers are invoked by application workload models [9].

IV. ESTIMATING ENERGY CONSUMPTION OF DATA-LINK PROTOCOLS

The ABSOLUT performance evaluation approach allows the system designer to adjust the parameters for the platform components (prior to simulation) and also reports a variety of performance statistics after simulation. A subset of these parameters and statistics are listed below.

- Adjusting Clock rate of platform components.
- The percentage of platform component consumption (proportion of the total busy times of components) by the different OSI model layers and their protocols.
- Average Packets/sec.
- Average Frames/sec.
- Idle/Busy times of components.
- The number of words written to and read from the memory.
- The collision rate and the maximum and minimum frame delays.
- Maximum and minimum frame delays.
- The busy times of each and every core of a multi-core processor.
- Trace, which validates the functional correctness of the MAC contention resolution algorithms.

The aforementioned parameters and performance statistics are used for measuring the energy consumption of the platform components by different Layers of the Open Systems Interconnection (OSI) model [16]. The total energy consumption of the platform is shown as the sum of energy consumption of all the platform components

$$E_{platform\_total} = E_{comp1} + E_{comp2} + \ldots + E_{compN}, \qquad (1)$$

where the E_{compi} is the energy of the i^{th} platform component.

The energy of each platform component is mostly found by multiplying the average power of the component with the busy time. It should be noted that in the case of some components, the power consumed can be given by the analytical formulas as explained afterwards. If the average power consumption of the components is available via vendors, we can write

$$E_{platform\_total} = (Pwr_{comp1})(T_{comp1}) + (Pwr_{comp2})(T_{comp2}) + \ldots + (Pwr_{compN})(T_{compN})$$

$$E_{plaform\_total} = \sum_{n=1}^{N} \left(Pwr_{compi}\right) \left(T_{compi}\right). \qquad (2)$$

The total busy time of each component (T_{compi}) is due to the sum of busy times (utilization of the component) due to each OSI model layer. ABSOLUT provides the ability to separately report the consumption of data-link and transport layers by abstracting out the application workload models by traffic generators. Therefore the energy consumption of the data-link layer can be given by.

$$E_{plaform\_dl} = \sum_{n=1}^{N} \left(Pwr_{compi}\right) \left(T_{compi\,DL}\right). \qquad (3)$$

This equation requires the busy time and power consumption of each component in the platform. We now describe the methods for estimating the busy times and power consumption of platform components.

A. *Estimating Busy Times of Platform Components*

As shown by (3), in order to compute the busy times of the components, the workloads of the OSI model layers considered in the use case must be modelled. Just like the application layer ABSOLUT workload models, the workload models of the data-link layer are also composed of abstract instructions as described by Khan et al. [5] and are executed by the processor models (multi-core or single core) of the ABSOLUT platform models. After execution, the amount of time taken by the platform components to execute these instructions is automatically reported. During the simulation whenever the MAC frame of a transport layer packet is served by the functionally correct data-link layer

MAC protocols, the workload (consisting of abstract instructions) is send to the underlying processing elements of the ABSOLUT platform by the ABSOLUT operating system (OS) model as described in [5].

The data-link layer protocols (for example MAC protocols) and transport layer protocols are modelled as operating system (OS) services in ABSOLUT as explained in [5] The transport layer OS_Service [5] divides the application layer message into individual frames and sends it one by one to the data-link layer OS_Service for transmission over the channel. The higher layer ABSOLUT OS_Service uses the lower layer ABSOLUT OS_Service as described in [5] just as in case of real OSI model layers. The functional correctness of the ABSOLUT OS_Service provides reliable estimates of non-functional properties such as end-end packet and frame delays and loss rates, which play an important role in the end-user experience in a variety of distributed multimedia streaming applications [10]. In other words, in case of data-link layer, whenever an ABSOLUT channel model is sensed idle by the MAC protocol model, the frame is transmitted and the non-functional workload mimicking frame processing in actual platform is send to the ABSOLUT platform model.

The workload per frame transmission can be automatically extracted via ABSINTH-2 tool if the source code of the MAC protocol implementation is available. In cases where the source code of data-link protocols is not available, the workload can be estimated using CORRINA [8]. In this case, firstly the overall workload of message transmission at the transport layer i.e., the workload of TCP/IP BSD API functions is extracted first. Afterwards, this information is used to estimate the workload of a single frame transmission (data-link protocol). The obtained ABSOLUT workload models are non-functional but the handling of the frame (contention resolution and retransmissions etc.) is done in the functionally correct manner, which is confirmed by the trace generated by the corresponding probes [5].

The workload models of ABSOLUT can be estimated via run-time performance statistics based methods called CORRINA if the source code of the data-link protocol is not available. In such cases, firstly the overall workload models at the transport layer (sum of workload due to data-link and transport) is estimated and afterwards the workload of transport layer is subtracted from it to obtain the workload due to data-link layer. The workload due to the processing at the transport layer (excluding underlying MAC layer) is estimated by using the case studies conducted in research as shown in [5]. If the source code of the data-link protocol is available, ABSINTH-2, which is a compiler based technique, can be used to generate highly accurate workload models.

1) CORRINA: First of all, the TCP/IP or UDP socket API functions, which are meant for sending and receiving the messages in distributed applications are identified. In case of TCP/IP, the socket API functions are send() and receive(). Afterwards, a test bench consisting of a client and server is programmed. The messages exchanged between the client and server can have different lengths (sizes in bytes). The data size is limited to a few bytes bytes so that one MAC frame is transmitted by the MAC layer for each call to send() and receive() transport functions at the transport layer. TCP/IP API Functions tagged by CORRINA [8] and the ABSOLUT workloads models are extracted after exchanging a number of messages between the client and server. In this way the average workload of a TCP/IP send() and receive() API function corresponds to the workload per packet transmission.

The extracted workload models can be used as such and mimic the workload per frame transmission. The energy consumption values obtained as a result will always be pessimistic since they also contain the transport layer processing workload. If required, these workload models can be refined by subtracting the overheads of transport layer from the overall workloads obtained (for TCP/IP API functions send() and receive()). The information about the processing overheads of the transport layer is elaborated in [8]. The obtained workload models are mapped to the corresponding ABSOLUT MAC OS_Service and on each frame transmission, the workload/frame is executed by the processor models in the ABSOLUT platform models.

2) ABSINTH-2: This method is especially useful for the workload modelling of MAC protocols which are developed for specialized applications such as WSNs (Wireless Sensor Networks). The implementation of these protocols is usually done in “C” programming language. WSNs have more specific requirements, which include a local unicast or broadcast. The traffic flow is usually from many nodes towards one or a few sinks (most traffic is thus directed in one direction). The individual nodes have periodic or rare communication and must consider energy consumption as a major factor.

An effective MAC protocol for WSNs must have reduced power consumption, shall avoid collisions, should be implemented with a small code size and memory requirements, be efficient for a single application and be tolerant to changing radio frequency and networking conditions [17]. That is why many WSNs employ highly efficient MAC protocols for the transfer of frames over the wireless channels for example NANO MAC [2] and BMAC [18].

After the design and development of these novel MAC protocols, the source code of these protocols can be compiled with ABSINTH patched gcc compiler and executed. After executing the use-case, the profiling information, which is used to generate highly accurate workload models, is obtained. These workload models can be mapped to the ABSOLUT MAC OS_Service to obtain the energy consumption of these protocols on different platforms.

B. *Estimating Power Consumption of Platform Components*

The power consumption of the individual components (busses, processors and memories etc.) are obtained either via vendor specifications or analytical models. If the average power consumption of a platform component is provided by the vendor, the energy consumption is easily estimated by multiplying the power consumption value with the busy time of the component reported by ABSOLUT performance simulation. For example, if the ABSOLUT platform model contains models of ARM_Cortex A-9 processors, the energy consumption of the processor is simply a product of the busy times obtained by the ABSOLUT performance simulation and the power values available on the following website [15].

If the power values are not provided by the component manufacturer, the analytical models available from the literature are utilized. For example, the power consumption of an un-buffered DDR2 SDRAM (in idle state) is given by [19]:

$$P_{RAM\_idle} = \sum_{i=1}^{n}(S_i * P), \tag{4}$$

where *n* is the number of installed memory modules and *s* is the size (in GB) of the memory modules. The values of "p" for different vendors are mentioned in Table I.

TABLE I: VALUES OF P FOR UN-BUFFERED DDR2 SDRAMS MANUFACTURED BY DIFFERENT VENDORS

Vendor	**Value**
Kingston	*f/1000*
Samsung	*0.95*f/1000*
Hynix	*1.9*f/1000*
Generic	*1.45*f/1000*

The model of the dynamic power consumption has been derived in [19] by using the RAMSpeed benchmark [20] and is given by the following equation:

$$P_{RAM} = P_{RAM\_idle} + \beta, \tag{5}$$

where β = 7.347 as described in [19]. The equations listed in Table I require the value of frequency (*f*) of the SRDAM in Mega Hertz, which is obtained by accessing its value from the list of ABSOLUT platform component parameters values set by the system designer. Also, the different analytical models for the measurement of power consumption of the network interface (NIC) card are provided in [21]. We consider the linear model given by the following equation:

$$P_{NIC} = P_{idle} + (P_{max} - P_{idle}) * pps. \tag{6}$$

In (6), $P_{NIC,}$ P_{idle} and P_{max} denote the total, idle and maximum power of the network interface card whereas *pps* denotes packets per second, which is reported at the end of ABSOLUT simulation. In many cases, the values of P_{idle} and P_{max} can be obtained from vendor specifications . In cases where the values of P_{idle} and P_{max} are not provided by the vendors, they can be estimated by using the trace based method described by Ebert et al. [22]. The system designer might select a range of values for P_{idle} and P_{max} to determine the range (of allowable P_{idle} and P_{max}) of allowable values feasible for the use case. After the simulation the NIC cards, which fulfil the power and energy budgets can be integrated into the devices.

V. METHODOLOGY

The energy consumption of data-link protocols via ABSOLUT involves the following steps.

1. In the first step, the ABSOLUT workload models for data-link protocols are automatically generated by using ABSINTH-2 or CORRINA. These workload models are assigned to the ABSOLUT MAC OS_Service [5]. Therefore, whenever a MAC frame is transmitted, the estimated processing load is executed on the platform model.
2. After workload extraction of data-link layer, the application workload models are abstracted out by using traffic generators. This is important since we need only the workload of data-link protocols to load the ABSOLUT platform model. In this way, we get the platform utilization of the platform by data-link layer only [5] in isolation.
3. After simulation, the busy time of each platform component is automatically obtained and can be used directly in (3). The busy time of each platform component is then multiplied by the average power of the corresponding platform component to find the energy consumption of the component due to processing load of data-link layer. The power of different platform components can be found as described in this section.
4. In the next step, the use-case is modeled. The system designer initializes traffic generators [5] with desired parameters and employing the model of the simulated MAC protocol.
5. Once the ABSOLUT performance model is finalized by the system designer, the performance model is executed.
6. After simulation, the busy times of all the hardware components in the platform are reported automatically in a text file. The busy times of each platform component are used in (3) for estimating the energy consumption of the platform by the data-link layer.
7. The busy time of each platform component is then multiplied by the average power of the corresponding platform component to find the energy consumption of

the component due to processing load of data-link layer. The power of different platform components can be estimated in a variety of ways.

8. The sum of energy consumption of all platform components amounts to the overall energy consumption of the platform as described in (3).

The aforementioned methodology for estimating energy consumption of data-link protocols is summarised in Fig. 3.

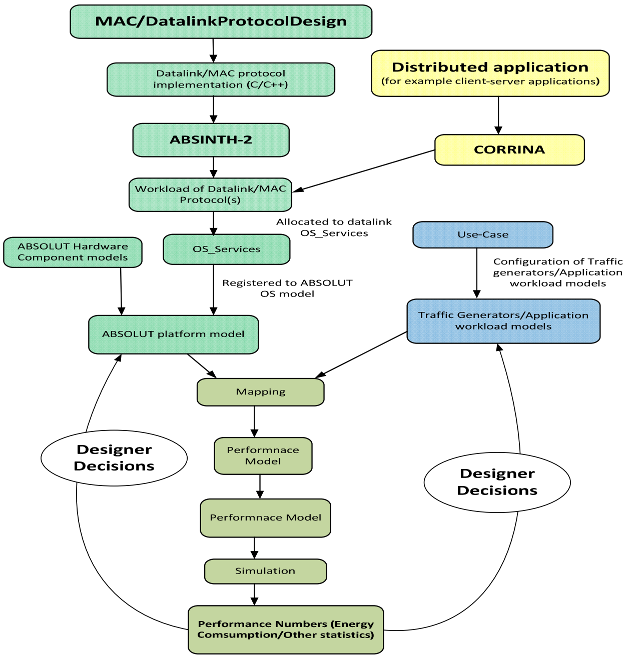

Figure 3. Measuring energy consumption of data-link layer via ABSOLUT

The methodology is not restricted to the energy consumption of data-link layer and can be used to estimate the energy consumption of other layers of the OSI model.

VI. CASE STUDY "IEEE WLAN (IEEE 802.11 DCF)"

After elaborating the methodology, we now describe two case studies, which validate the approach. In both the case studies, CORRINA was used to estimate the workload of the data-link layer as described in previous sections. The UDP/IP transport protocol was used in the test bench consisting of a client server application. The client and servers merely exchange messages of fixed lengths (in Bytes). The message size was limited to a few bytes to ensure the transmission of a single frame for each exchanged message. In this way, the workload per frame transmission is estimated. Therefore, whenever an ABSOLUT OS_Service transmits a frame, the corresponding workload is send to the underlying platform processors by ABSOLUT MAC model for processing. For pessimistic results, we included the overheads of transport layer, though more accurate (and less pessimistic) results can be obtained by studying the research articles focusing on the overheads of TCP/IP or UDP protocols [8]. The workloads used in both the case studies are shown in Table II.

TABLE II: PROCESSING TIMES OF TCP/IP API FUNCTIONS AND ABSTRACT INSTRUCTIONS OF CORRESPONDING ABSOLUT WORKLOAD MODELS. WORKLOADS OF ONLY SEND() AND RECEIVE() API FUNCTIONS ACT AS ESTIMATES OF ABSOLUT PER-FRAME TRANSMISSION WORKLOADS. THESE WORKLOADS ARE USED IN THE ABSOLUT DATA-LINK LAYER MODELS. CORRINA [8] WAS USED FOR AUTOMATIC WORKLOAD MODELING.

NoTA API funcitons	**Average Execution Times on Intel Core i5 Processor based platform. Message size=2 Bytes**	**ABSOLUT workload models (abstract instructions).Workload models of send() and receive functions represent the workload/frame**

socket	3196 usec	m_host->execute(58274432);
		m_host>read(Address(0),45582561,32);
		m_host->write(Address(0),0,32);

bind	446 usec	m_host->execute(1296);
		m_host->read(Address(0),3375,32);
		m_host->write(Address(0),0,32);

listen	374 usec	m_host->execute(1168);
		m_host->read(Address(0),3300,32);
		m_host->write(Address(0),228,32);

accept	592 usec	m_host->execute(372284);
		m_host->read(Address(0),4491,32);
		m_host->write(Address(0),772,32);

receive	442 usec	m_host->execute(283971);
		m_host->read(Address(0),85133,32);
		m_host->write(Address(0)23149,32);

send	429 usec	m_host->execute(67884);
		m_host->read(Address(0),391062 ,32);
		m_host->write(Address(0), 52653,32);

From this point onwards, we only describe the simulation scenarios and the results.

A. Overview of WLAN (IEEE 802.11 DCF)

In a wireless network where a number of stations contend for the wireless medium, if multiple stations sense the channel busy and defer their access, they will also virtually simultaneously find that the channel is released and then try to seize the channel. As a result, collisions may occur. In order to avoid such collisions, IEEE 802.11 Distributed Coordination Function (DCF) is employed, which requires a station wanting to transmit, to first listen to the channel to check its status (occupied or not) for a DCF Inter-frame Space (DIFS) interval. The IEEE 802.11 DCF [5] can be shown in the form of a flow chart, as shown in Fig. 4.

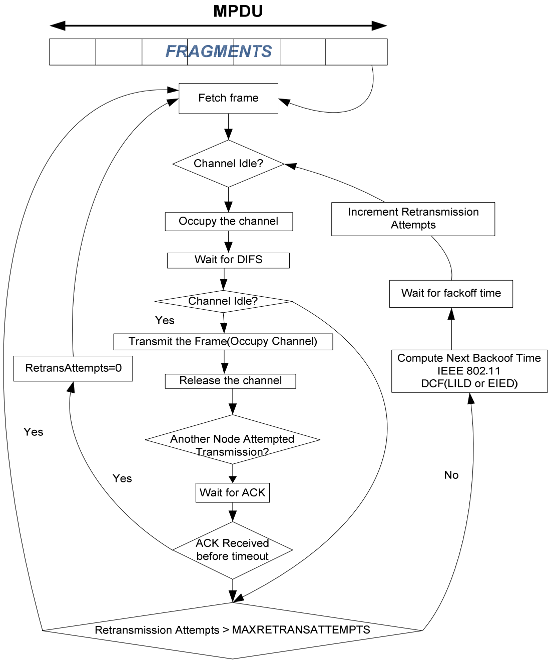

Figure 4: Flow chart of IEEE 802.11 DCF

In ABSOLUT, the MAC protocols, as well as transport layer protocols [5], such as TCP and UDP, are also modelled as services by deriving them from the same base class OS_Service, which provides the scheduling and synchronization mechanism. The transport layer services then request the Layer-2 services such as IEEE 802.11 DCF for transmission of individual frames of a transport layer packet. The do_service() method of the OS_Service base class is implemented by the derived class to provide the functionality of IEEE 802.11 DCF, as explained by Khan et al. [5]. The do_service() method spams a separate frame transmission function for handling each request of frame transmission from the transport as described in [5]. The simulation parameters used for the conducted case study are mentioned in Table III.

TABLE III: EXPERIMENT PARAMETERS (IEEE 802.11B)

Parameters	Values
SIFS	10 µs
DIFS	50 µs
Slot Interval(SLOT)	20 µs
Preamble Length	144 bits
PLCP header Length	48 bits
Channel bit rate	2 Mbps
CW_{min}	31
CW_{max}	1023
CW_o	32
EW	16

Whenever a frame transmission is serviced by ABSOLUT MAC protocol model, the workload per frame transmission estimated by CORRINA is executed by the ABSOLUT platform model. After simulation, the busy times and other performance statistics (for example, busy times of components and packets/sec, etc.) are obtained and used to automatically estimate the energy utilization of modelled platform.

B. Simulation Scenario

The simulations are carried out in WLAN environment and consist of 11 nodes, i.e., 10 wireless nodes sending data to an access point. Each node transmits 100 packets to the access point. The transmission of packets occurs after intervals obtained by configuring a Constant Bit Rate (CBR) Traffic Generator. The exact configuration of the traffic generator is shown in Fig. 5. All the network nodes are within the transmission range (form a collision domain) of each other. The access point acts as the only destination for the clients. 802.11 has a variety of standards, the standard simulated in the case study was 802.11b. If the channel is sensed busy, the nodes enter the collision avoidance phase in which each node executes the exponential back-off algorithm [23], as shown in Fig. 4. The nodes wait for a random time interval distributed uniformly between [0,CW] × SLOT. The Contention Window (CW) values can vary between CW_{min}(31) and CW_{max} (1023). The slot value (SLOT) for 802.11b is 20µs. The aforementioned parameters for 802.11b are listed in Table III. Typically, 802.11b products degrade the bit rate from 11Mbps to 5.5,2 or 1 Mbps [23]. We use the bit rate of 2Mbps during the simulation. The bit rate can be changed dynamically during simulation if desired with minor modelling effort. The CBR traffic generator available in ABSOLUT simulation framework is derived by the ns-2 CBR traffic generator [5]. The traffic generators can be configured with different bit rates and packet sizes via simple interface functions, as shown in Fig. 5.

```
//Decide the simulation parameters
double AverageBurstTime=.0025;      //In seconds, means 2.5 milliseconds
double InterFrameTime  = 1   ;      //Means 1 seconds
double SinglePktSize   = 512 ;      //Sinlge packet lent
double Bitrate         =2000000;    //2 Mega bits per second


//Make three generators.
SSConfig::Instance()->SetTrafficGenerator(CBR_TRAFFIC_GENERATOR,
AverageBurstTime,InterFrameTime, Bitrate,SinglePktSize);
```

Figure 5: An example configuration of the CBR traffic generator. Packet Length = 512 Bytes. Data rate = 2 Mbps. Average Burst Time = 0.0025 seconds. Inter Frame Space = 1 second.

C. *Platform Model*

Each ABSOLUT platform model used in the case study (all nodes and access point) consists of an ARM Cortex-A9 multi-core processor [10] model including four processing cores along with SDRAM, a POWERVR SGX40 graphics accelerator and an Image signal processor. This is shown in Fig. 6. These component models are connected via an AMBA bus model. The processor cores are clocked at 1 GHz and the DDR2 RAM operates at 800 MHz. The Bus model operates at 800MHz. In ABSOLUT methodology, the application models contain approximate timing information. Thus the execution platform is modelled at transaction level following OSCI TLM2.0 standard [9].

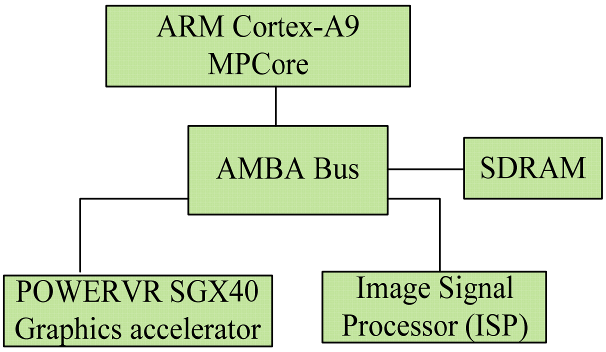

Figure 6: ABSOLUT platform model

Each processor core in the processor model has an L1 instruction and L1 data cache as shown in Fig. 7.

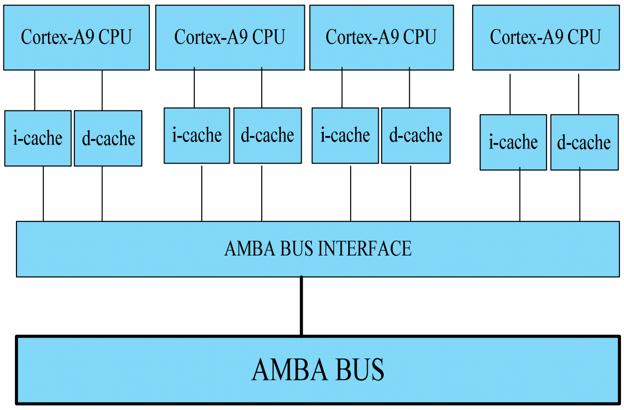

Figure 7: Diagram showing the quad-core processor (ARM Cortex A9 multi-core processor) models used in the performance simulation

D. *Simulation Results*

The energy and average power consumption of the platforms hosting the access point and the network nodes are shown in the following Table IV.

TABLE IV: ENERGY CONSUMPTION OF ACCESS POINT AND AVERAGE ENERGY CONSUMPTION OF THE WIRELESS NODES DUE TO IEEE 802.11B.

Platform Component	**Access Point**	**Wireless Nodes(Average)**
ARM-CORTEX-A9 MPCore	1341.35 kJ	711.694 kJ
NIC (Network Interface Card)	85.6 J	86.2 J
ISP	0 J	0 J
POWERVR SGX40	0 J	0 J
SDRAM	101.013 J	93.064 J
BUS	0.012984 J	0.011809 J
Total Energy of Platform	***1341.5366 kJ***	***711.8732 kJ***

Table V and Table VI present the transport layer and data link layer statistics, which are were collected with the ABSOLUT simulation framework.

TABLE V: UDP PROTOCOL STATISTICS

Average Transport Delay	284.51 ms
Maximum Transport Delay	2456.05 ms
Minimum Transport Delay	0 ms

TABLE VI: 802.11B MAC STATISTICS

Average CW Size	849.008
Maximum CW Size	2048
Average Backoff Time	34.908 ms
Maximum Backoff Time	102.4 ms
Minimum Backoff Time	0 ms
Average MAC Transmission Delay	54.2947 ms
Maximum MAC Transmission Delay	330.33 ms
Minimum MAC Transmission Delay	0 ms
Average Retransmission Attempts per Transmission	2.39828
Minimum Retransmission Attempts per Transmission	0
Maximum Retransmission Attempts per Transmission	7
Total Transmissions	25623
Total Collisions	18083
Collision Probability	0.705733
Average Collisions Per 100 Sec	2804

In our simulation scenario, the detailed statics where collected from UDP protocol and IEEE 802.11b medium access control layer.

VII. CONCLUSION AND FUTUREWORK

The data-link and transport layer models in ABSOLUT offer enormous flexibility and employ separation of concerns, which helps the system designer to modify the models according to modeling objective.

For example, in order to implement a new contention resolution scheme, the system designer only needs to implement or modify the state machine of the algorithm [5]. The energy consumption as well as performance of a particular OSI model layer can be obtained in isolation by probes and abstracting the workload models of other layers by delays. The performance statistics of the platform components can also be obtained at various levels of detail, for example in [10], the authors have demonstrated the performance of individual cores of the processor models. Also, the other component specific performance statistics such as cached hits/misses for cache models can also be obtained. Due to the availability of different types of traffic generators in ABSOLUT, various scenarios can be simulated. The ABSOLUT component library [9] allows the system designer to model a variety of widely used platforms [10]. In this work, the performance (in terms of energy consumption) of a widely used data-link protocol was demonstrated via ABSOLUT methodology. The same methodology can be used for the performance and energy evaluation of other data-link protocols. In future, the methodology can be further enhanced by providing a library of state machines ABSOLUT models for widely used data-link protocols. This will further shorten the time needed for selecting the appropriate data-link protocol.

ACKNOWLEDGEMENTS

This work was performed in the EU-ECONET project funded by the European Union.

REFERENCES

[1] S. Khan, E. Ovaska, K. Tiensyrjä, and J. Nurmo Nurmi, ”From Y-chart to seamless integration of application design and performance simulation,” Proc. International Symposium on System-on-Chip - SOC. Tampere, Finland, 29-30 Sept. 2010. IEEE. Piscataway, NJ, USA, 2010, pp. 18-25

[2] J. Haapola, “NanoMAC: A Distributed MAC Protocol for Wireless Ad Hoc Sensor Networks,” Proc. XXVIII Convention on Radio Science & IV Finnish Wireless Communication Workshop, 2003, pp. 17-20.

[3] http://www.samsung.com/fi/#latest-home, [retrieved: August, 2013].

[4] http://www.apple.com, [retrieved: August, 2013].

[5] S. Khan, J. Saastamoinen, M. Majanen, J. Huusko and J. Nurmi, “Analyzing Transport and MAC Layer in System-Level Performance Simulation,” Proc. International Symposium on System-on-Chip 2011, Tampere, Finland, October 31 - November 2, 2011.

[6] J. Saastamoinen and J. Kreku, ”Application workload model generation methodologies for system-level design exploration,” Proc. IEEE Conference on Design and Architectures for Signal and Image Processing, DASIP 2011. Tampere, Finland, 2-4 Nov. 2011, pp. 254-260

[7] http://icl.cs.utk.edu/papi/, [retrieved: August, 2013].

[8] S. Khan, J. Saastamoinen, J. Huusko, J.-P. Soininen and J. Nurmi, “Application Workload modelling via Run-Time Performance Statistics,” IJERTCS 2013. Int. J. Embedded and Real-Time Communication Systems *(Accepted). To appear.*

[9] J. Kreku et al., “Combining UML2 Application and SystemC Platform Modelling for Performance Evaluation of Real-Time Embedded Systems,” Hindawi Publishing Corporation. EURASIP Journal on Embedded Systems. Volume 2008, Article ID 712329, 18 pages, doi:10.1155/2008/712329.

[10] S, Khan, J. Saastamoinen and J. Nurmi, “System-Level Performance Evaluation of distributed multi-core NoTA systems,” Proc. 2nd IEEE International Conference on Networked Embedded Systems for Enterprise Applications. NESEA 2011, Fremantle, Perth, Australia. 8th - 9th December 2011.

[11] S. Khan, J. Saastamoinen, K. Tiensyrjä and J. Nurmi, “System Level Performance Simulation of distributed GENESYS Applications on multi-core platforms.” Proc. 9th IEEE Int. Conf. on Dependable, Autonomic and Secure Computing (DASC), 2011.

[12] P. Hurni, B. Nyffenegger, T. Braun and A. Hergenroeder, “On the accuracy of software-based energy estimation techniques,” in Wireless Sensor Networks (pp. 49-64). Springer Berlin Heidelberg, 2011, pp. 49-64.

[13] G. P. Halkes, T. van Dam and K. G. Langendoen, “Comparing energy-saving MAC protocols for wireless sensor networks,” in Mobile Networks and Applications, 10(5), 2005. pp. 783-791.

[14] V. Ramchand and D. K. Lobiyal, “An analytical model for Energy Consumption in Y-MAC Protocol,” International Journal of Computer Science, 9, 2012.

[15] http://www.arm.com/products/processors/cortex-a/cortex-a9.php, [retrieved: August, 2013].

[16] H. Zimmermann, “OSI Reference Model – The ISO Model of Architecture for Open Systems Interconnection,” IEEE Transactions on Communications, vol. COM-28, no. 4, April 1980, pp. 425-432.

[17] I. Lee, J. Y.-T. Leung and S. H. Son (ed.), “Handbook of Real-Time and Embedded Systems,” Chapman and Hall/CRC (July 23, 2007) p. 800, ISBN-10: 1584886781, ISBN-13: 978-1584886785.

[18] J. Polastre, J. Hill, and D. Culler, “Versatile Low Power Media Access for Wireless Sensor Networks,” Proc. ACM SenSys, November 2004.

[19] O. Mämmelä, M. Majanen, R. Basmadjian, H. De Meer, A. Giesler and W. Homberg, “Energy-aware job scheduler for high-performance computing,” Computer science – Research and Development, vol. 27, issue 4, November, 2012, pp. 265-275, DOI: 10.1007/s00450-011-0189-6

[20] http://alasir.com/software/ramspeed/, [retrieved: August, 2013].

[21] J. C. Cardona Restrepo, C. Gruber and C. M. Machuca, “Energy Profile Aware Routing,” Proc. First International Workshop on Green Communications, IEEE International Conference on Communications (ICC), June 2009. Dresden, Germany.

[22] J.-P. Ebert, B. Burns, and A. Wolisz, “A trace-based approach for determining the energy consumption of a wlan network interface,” Proc. European Wireless, Florence, Italy, February 2002, pp. 230-236.

[23] G. Berger-Sabbatel, F. Rousseau, M. Heusse and A. Duda, “Performance anomaly of 802.11b,” Proc. The 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), San Francisco, CA, USA, March 30 - April 3, 2003.