Simulator and Experimental Framework for the MAC of Power-Line Communications

Technical Report

Christina Vlachou
EPFL, Switzerland
christina.vlachou@epfl.ch

Julien Herzen
EPFL, Switzerland
julien.herzen@epfl.ch

Patrick Thiran
EPFL, Switzerland
patrick.thiran@epfl.ch

1 Introduction

Power-line communications (PLC) are employed in home networking as an easy way to provide high-throughput connectivity. IEEE 1901 [3] is the dominant MAC protocol for power-line networks, and its most popular implementation is HomePlug AV. This technical report introduces guidelines for measurements in HomePlug AV testbeds. We describe two tools that manage and configure PLC devices. These tools are useful for both users or researchers for measuring statistics or configuring HomePlug AV devices. Finally, this report presents a simple simulator for the CSMA/CA mechanism of the 1901 MAC protocol. We underline the complexity of the 1901 MAC and the features that challenge the simulation of the whole MAC stack of HomePlug AV.

The rest of the report is organized as follows. Section 2 presents the CSMA/CA mechanism studied through this report. Second 3 describes the experimental framework employed to measure statistics of HomePlug AV devices. Finally, Section 4 introduces a simple simulator for the HomePlug AV/IEEE 1901 CSMA/CA mechanism.

2 Background

In this section, we present the CSMA/CA procedure of 1901 [3]. This mechanism is the main difference between 1901 and 802.11. In CSMA/CA protocols such as 802.11, stations wait for a random number of time slots (determined by the *backoff counter*) before transmitting, in order to minimize the probability that some other station transmits at the same slot, which causes a collision. Nevertheless, a collision can still occur and when it does, the stations involved increase the range in which they select their backoff counter (called the *contention window CW*) to further reduce the collision probability. Clearly, there exists a tradeoff in CW: if CW is large, collision probability is small, but stations waste many slots on average before transmitting, which decreases throughput. As we explain later, 1901 aims at reducing CW – in particular, the minimum CW – to tackle this backoff inefficiency. To counterbalance the resulting large collision probability, 1901 introduces an additional mechanism that increases CW *before* a collision occurs: when a station senses a considerable number of transmissions in the channel, it increases CW. To count the number of times the station has to sense the medium busy before increasing CW, a new counter is introduced, called the *deferral counter*.

We now describe the technical details of the 1901 CSMA/CA procedure. It includes three counters: the backoff counter (BC), the deferral counter (DC) and the backoff procedure counter (BPC). Upon the arrival of a new packet, a transmitting station enters backoff stage 0. It then draws the backoff counter BC uniformly at random in $\{0, \ldots, CW_0 - 1\}$, where CW_0 denotes the contention window used at backoff stage 0. Similarly to 802.11, BC is decreased by 1 at each time slot if the station senses the medium to be idle (i.e., below the carrier-sensing threshold), and it is frozen when the medium is sensed busy. In case the medium is sensed busy, BC is also decreased by 1 once the medium is sensed idle again. When BC reaches 0, the station attempts to transmit the packet. Also similarly to 802.11, the station jumps to the next backoff stage if the transmission fails. In this case, the station increments the BPC counter and enters the next backoff stage. The station then draws BC uniformly at random in $\{0, \ldots, CW_i - 1\}$, where CW_i is the contention window used for backoff stage i, and repeats the process. For 802.11, the contention window is doubled between the successive backoff stages, i.e. $CW_i = 2^i CW_0$. For 1901, CW_i depends on the value of the BPC counter: there are four backoff stages which are mapped to the BPC counter, as given in Table 1.

The main difference between 1901 and 802.11 is that a 1901 station might enter the next backoff stage even if it did not attempt a transmission. This is regulated by the deferral counter DC, which works as follows. When the station enters backoff stage i, DC is set at an *an initial DC value* d_i, where d_i is given in Table 1 for each backoff stage i. After having sensed the medium busy, a station decreases DC by 1 (in addition to BC). If the medium is sensed busy and $DC = 0$, then the station jumps to the next backoff stage (or re-enters the last backoff stage, if it is already at this stage), and re-draws BC *without attempting a transmission.* An example of such a backoff process is shown in Figure 1.

		CA0/CA1		CA2/CA3	
backoff stage i	BPC	CW_i	d_i	CW_i	d_i
0	0	8	0	8	0
1	1	16	1	16	1
2	2	32	3	16	3
3	≥ 3	64	15	32	15

Table 1: IEEE 1901 parameters for the contention windows CW_i and the initial values d_i of deferral counter DC, for each backoff stage i. CA0/CA1 priorities are used for best-effort traffic and CA2/CA3 for delay sensitive traffic.

As shown in Table 1, there are four different priority classes (CA0 to CA3), which correspond to different values for the CW_i's and the d_i's. The 1901 standard specifies that only the stations belonging to the highest contending priority class run the backoff process. In practice, the contending priority class is decided during a so-called *priority resolution phase*, using a simple system of busy tones. These busy tones are transmitted during the two so-called *priority slots*. The rest of the priority classes defer their transmission until the highest contending priority class does not transmit a busy tone during the corresponding slot.

3 HomePlug AV Experimental Framework

In this section, we give guidelines on how to measure collision probability and capture management and data frames in HomePlug AV networks. To conduct measurements we use two tools: *faifa* [2] and *Atheros Open PLC Toolkit* [1]. These tools enable a user to interact with the HomePlug AV firmware

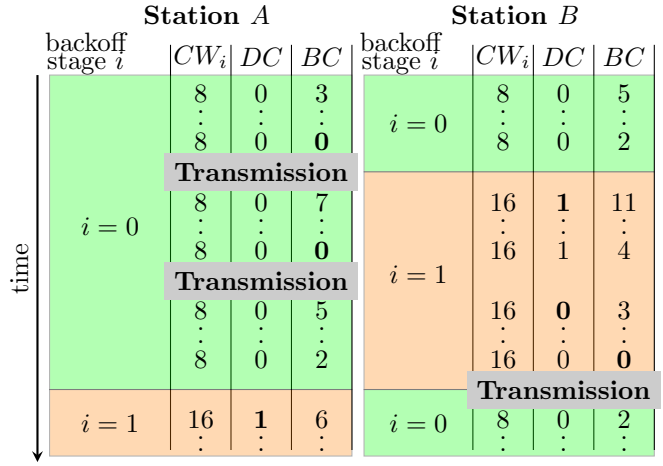

Figure 1: An example of the time evolution of the 1901 backoff process with 2 saturated stations A and B. Initially, both stations start at backoff stage 0. Station A wins the channel for two consecutive transmissions. Observe the change in CW_i when a station senses the medium busy and has $DC = 0$. This figures also exposes the short-term unfairness when there are 2 contending stations; a station that grabs the channel for a successful transmission moves to backoff stage 0, whereas the other station enters a higher backoff stage with larger CW and has lower probability to transmit.

by employing management messages (MMEs), as defined in the standard [3]. MMEs are used by the stations for network management and quality of service enhancements. In addition to the MMEs specified by [3], vendor-specific MMEs are used to configure the devices or measure statistics. The tools presented here, i.e., [1] and [2], employ the vendor-specific MMEs of PLC chips such as INT6300[1]. The PLC device distinguishes the MME requests using the field MMType of the MME header [3], and responds with the corresponding MME reply. Both tools presented here have functionalities such as measuring the PHY error rate, the number of collided and acknowledged frames, or the PHY data rates between the stations. However, we employ each tool to measure different metrics, as [3] has an option of reseting statistics for the frame transmission or reception, and [2] captures and prints the fields of the preambles of PLC frames.

In the following paragraphs, we assume that we have N saturated PLC stations transmitting UDP traffic to the same destination station called D. At each experiment, only the N stations are activated and plugged on the power-strip, to avoid hidden factors that might affect our measurements, such as MMEs exchanged between non-transmitting stations. Moreover, the channel conditions are ideal, as we attach all stations to the same power-strip.

3.1 Frame Aggregation and Frame Bursting

IEEE 1901 employs aggregation of multiple Ethernet frames in one PLC frame. The data are organized in physical blocks (PBs), which are blocks of 512 bytes. Then, the PBs are organized in a MAC protocol data unit (MPDU), which is the PLC frame. Now, stations can transmit multiple MPDUs in a burst through CSMA/CA access, when their backoff counter expires (see Section 2). Up to four MPDUs may be supported in a burst. While this number indicates the upper limit, the actual number of MPDUs per burst supported by a station depends on channel conditions and station capabilities [1]. Thus, in 1901, bursts contend for the medium and not individual MPDUs. As this result might affect our

[1]The testbed for which this report presents guidelines comprises devices with the INT6300 chip.

measurements, we measured the frequency of all the possible burst sizes. It turns out that the stations in the isolated experiments use bursts with 2 MPDUs (measured using the methods of Subsection 3.3). Thus, to evaluate the collision probability via testbed measurements we use the number of collided and acknowledged MPDUs, because this is the only information we can obtain from the INT6300 PLC chips.

3.2 Measuring Collision Probability

To measure collision probability we use [1]. With the command *ampstat* of [1], we can reset to 0 or retrieve the number of acknowledged and collided PLC frames (MPDUs) given the destination MAC address, the priority, and the direction (transmission or reception) of a specific link.

To measure collision probability, we reset the statistics of the frames transmitted at all the stations at the beginning of each test. Then, at the end of the test we request the number of collided and acknowledged frames transmitted from all the stations given the MAC address of the destination station D. To obtain these statistics *ampstat* sends an MME with MMType 0xA030. We use the reply of the PLC interface. Specifically, the bytes 25-32 of this reply represent the number of acknowledged frames and the bytes 33-40 represent the number of collided frames.

Now, let C_i be the number of collided frames transmitted by station i, and let A_i be the number of acknowledged frames transmitted by station i. To evaluate the collision probability in the network, we compute $\sum_{i=1}^{N} C_i / \sum_{i=1}^{N} A_i$. Observe that we divide only by the sum of the acknowledged frames not including the collided ones in this sum, as the 1901 standard allows selective acknowledgments for all the physical blocks contained in a frame [3]. When a collision occurs and the preambles of the collided frames can be decoded (due to the robust modulation with which they are transmitted), the destination acknowledges the frame, with an indication that all the physical blocks are received with errors, which yields a collision. Indeed, we verified this 1901 feature with our testbed, as we observed that the total number of acknowledgment frames $\sum_{i=1}^{N} A_i$ increases with the number of stations N, which means that this number includes the collided frames also. If it was not including the collided frames also, then this number would be drastically decreasing with the number of contending stations due to collisions (given that all tests are run for the same duration). Note that if we run the simulator of Section 4, we also observe that the number of transmitted frames increases with N, because as N increases the backoff counters of the stations expire more often, yielding less total time spent in backoff. Table 2 presents the statistics $\sum_{i=1}^{N} C_i$, $\sum_{i=1}^{N} A_i$, of one test with duration 240s for $1 \leq N \leq 7$.

$\sum_{i=1}^{N} C_i$	$\sum_{i=1}^{N} A_i$
$2.5000 \cdot 10^1$	$1.6222 \cdot 10^5$
$1.2012 \cdot 10^4$	$1.6202 \cdot 10^5$
$2.1390 \cdot 10^4$	$1.5978 \cdot 10^5$
$2.8924 \cdot 10^4$	$1.6259 \cdot 10^5$
$3.5990 \cdot 10^4$	$1.6539 \cdot 10^5$
$4.1877 \cdot 10^4$	$1.7144 \cdot 10^5$
$4.6989 \cdot 10^4$	$1.7608 \cdot 10^5$

Table 2: Statistics $\sum_{i=1}^{N} C_i$, $\sum_{i=1}^{N} A_i$ measured in one test for $1 \leq N \leq 7$. The collision probability is evaluated as $\sum_{i=1}^{N} C_i / \sum_{i=1}^{N} A_i$.

Figure 2 presents the average collision probability of 10 tests, the collision probability obtained by our simulator introduced in Section 4.2, and the collision probability estimated by an analytical model published in [5]. We observe an excellent fit between measurements, simulation, and analysis.

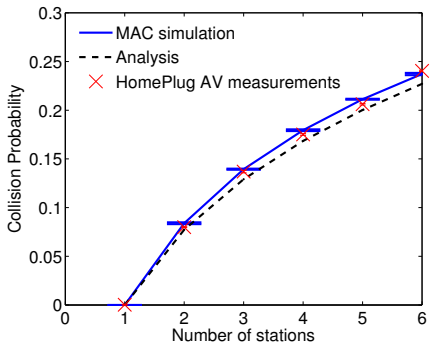

Figure 2: Collision probability obtained by simulation, our analytical model of [5], and experiments with HomePlug AV devices for the default configuration CA1 of 1901 given in Table 1.

3.3 Capturing Management and Data Frames

To capture management or data frame preables we use *faifa*. *faifa* activates the "sniffer" mode of the devices (using the option 0xA034 for the MMType of the MME), which is used to capture the start of frame (SoF) delimiters of all the PLC frames, including data frames, beacons, management. This tool prints the fields of the SoF that are defined in [3]. We can capture the SoF delimiters at the destination station D to measure the number of data frames and MMEs transmitted during the tests. To distinguish the management frames from the data frames, we use the *Link ID* field of SoF, which represents the priority of the frame, as we transmit UDP traffic at the default priority which is CA1. The UDP traffic is transmitted with CA1 priority, whereas the MMEs are transmitted with CA2 or CA3 priorities [3]. Note that with *faifa* we can only capture the SoF delimiters and not the frame content, hence unfortunately we are not able to unveil the content of the management messages that do not arrive in the Ethernet interface of the chip, in contrast to the data packets. We can use the information described above to measure the number of MMEs transmitted during the tests, hence compute the overhead of MMEs with respect to the data packets transmitted. This overhead is computed by dividing the number of bursts corresponding to MMEs by the number of bursts corresponding to data frames. We employ bursts and not individual MPDUs for this overhead, because as mentioned in subsection 3.1, bursts contend for the medium, hence each burst consumes some CSMA/CA time due to backoff, priority resolution, inter-frame spaces and ACKs. To identify the end of a burst we use the *MPDUCnt* field of the SoF, that denotes the number of remaining MPDUs for a specific burst. When this number is equal to 0, the corresponding MPDU is the last one in the burst.

Finally, the SoF contains the source identification of each frame, thus the "sniffer" mode of the devices can be used to capture a trace of the sources for all the transmitted data frames. Employing this, we can study the fairness of the PLC MAC layer by considering again bursts and not individual MPDUs. This method is used to produce the results of our prior work in [4].

4 PLC MAC layer Simulator

4.1 Challenges and IEEE 1901 MAC Layer Complexity

To the best of our knowledge, the firmware of HomePlug AV devices is not open-source, and it is encrypted. Hence, implementation details that are vendor-specific and are not mentioned in the standard [3], are unknown. These details prevent us from designing a simulator of the complete MAC stack and from estimating the achievable throughput in a real network by simulation. For instance,

the following parameters or mechanisms remain to be unknown and affect the throughput observed at the application layer:

Frame Aggregation Procedure and Bit Loading Some mechanisms in the aggregation procedure of Ethernet frames into a PLC frame are not defined by the standard. First, there is a timeout between the arrival of the first Ethernet frame inserted in the PLC frame and the last Ethernet frame inserted in the PLC frame. Second, the bit loading, thus the number of Ethernet frames inside a PLC frame, depends on the channel, and each frame can employ different modulation scheme. Thus, to simulate the full MAC stack, we need full information, or a model of the PHY layer. The algorithm that is employed to update the bit loading based on the channel conditions has not been published by the manufacturers of PLC devices.

Management Messages Apart from the data frames, there are a lot of management messages exchanged between the stations. These introduce some overhead as they consume also some backoff. Again, the standard does not describe the frequency of these management messages, and some of these messages are vendor-specific, hence their existence is not mentioned in the standard. Finally, some of these management messages are exchanged for updating the modulation scheme when the error rate of the channel changes. Hence, their arrival rate depends also on the channel conditions.

Channel errors In this work we assume that the channel is error-free, but in reality there are retransmissions due to errors. First, there is no model of the bit error probability for HomePlug AV devices, given the PHY layer mechanisms it employs. Second, the retransmissions can involve some physical blocks (PB) and not the entire frame. Hence, the number of PBs needed to be retransmitted and the frame length due to retransmissions remains unknown, and currently cannot be modeled. Third, there is a timeout mechanism at the reception side of the frames according to which if all the PBs of a frame are not received within a specific time interval, the PBs of this frame are discarded. So far, we could not find any information on the value of this timeout.

For the reasons mentioned above, we simulate the 1901 MAC using the finite state machine of the standard [3] ignoring the mechanisms described above. We focus on studying the MAC performance which is not affected by these mechanisms/overheads. The above mechanisms depend mostly on the PHY layer and affect the bit loading of the frames. Our simulator can be efficiently employed to evaluate the performance of different MAC configurations and investigate the dynamics of the MAC parameters described in Section 2.

4.2 A Simple, Finite-State-Machine Simulator

In this subsection we present our simulator, which evaluates the normalized throughput and the collision probability in the network. Note that the simulator does not include any PHY layer procedures, as we are only interested in the CSMA/CA protocol of 1901. To the best of our knowledge, there is no PHY layer simulator validated experimentally in a testbed that we could adopt to simulate both layers. Our simulator aims at studying the performance degradation as the number of station decreases. It can be modified to return the traces of successfully transmitted packets to study other metrics such as fairness. Table 3 summarizes the input variables of our simulator in the order they should be given.

To run the simulation we can run the command *sim_1901(N,sim_time,Tc,Ts,frame_length,cw,dc)*. For example, for the default 1901 configuration we can run the command *sim_1901(2,*$5 * 10^8$*,2920.64,2542.64,2050,[8 16 32 64],[0 1 3 15])*.

Notation	Definition
N	Number of saturated stations
sim_time	Total simulation time in μs
T_c	Collision duration in μs
T_s	Successful transmission duration in μs
frame_length	Frame duration in μs
cw	Vector of contention window values at each backoff stage
dc	Vector of initial deferral counter values at each backoff stage

Table 3: Simulator input variables

```
function [collision_pr,norm_thoughput]=sim_1901(N,sim_time,Tc,Ts,frame_length,cw,dc)
% this function simulates the IEEE 1901 MAC layer under the assumptions that
% stations are saturated (i.e. always have a packet to transmit), that the retry
% limit is infinite (i.e they never discard a frame until is successfully transmitted)
% and finally, that the stations belong to a single contention domain.

% INPUTS: N (number of stations), sim_time (simulation time in \mu
% s), Tc (duration of a collision in \mu s), Ts (duration of a successful
% transmission in \mu s), frame_length (duration of the frame length not
% including overhead such as preamble or inter-frame spaces), cw (vector of
% cw values at each backoff stage), dc (vector of initial deferral counter
% values d_i at each backoff stage)

% 1901 time slot duration
slot = 35.84;

%State 0 is initialize (change backoff parameters), 1 is Tx, 2 is idle
State = zeros(1,N);

%initializing
t = 0;
% the backoff procedure counter or the backoff stage for all stations
BPC = zeros(1,N);
% the backoff counter for all stations
BC = zeros(1,N);
% the deferral counter for all stations
DC = zeros(1,N);
% the contention window for all stations
CWmin = cw(1);
CW = CWmin*ones(1,N);
next_state = zeros(1,N) + 2;
% number of backoff stages
m = size(cw,2);
if size(dc,2) ~= m

    return;
end

% for statistics
collisions = 0;
succ_transmissions = 0;

while t <= sim_time

    for i = 1:N
```

```
    if State(i) == 0

        if BPC(i) == 0 || BC(i) == 0 || DC(i) == 0

            if BPC(i) < m

                CW(i) = cw(BPC(i) + 1);
                DC(i) = dc(BPC(i) + 1);
            else

                CW(i) = cw(m);
                DC(i) = dc(m);
            end

            BC(i) = unidrnd(CW(i),1,1)-1;
            BPC(i) = BPC(i) + 1;
        else

            BC(i) = BC(i) - 1;
            DC(i) = DC(i) - 1;
        end

        if BC(i) == 0

            next_state(i) = 1;
        else

            next_state(i) = 2;
        end
    end

    if State(i) == 2

        BC(i) = BC(i) - 1;

        if BC(i) == 0

            next_state(i) = 1;
        else

            next_state(i) = 2;
        end
    end
end

counter = sum(next_state == 1);

% medium idle
if counter == 0

    t = t + slot;
end

% successful transmission
if counter == 1

    succ_transmissions = succ_transmissions + 1;
```

```
        % the station that transmits successfully restarts its backoff
        % process at backoff stage 0
        BPC(next_state == 1) = 0;
        % all stations move to initialize state because they sensed the medium busy
        next_state = zeros(1,N);
        t = t + Ts;
    end

    % collision
    if counter > 1

        collisions = collisions + counter;
        % all stations move to initialize state because either they
        % collided or they sensed the medium busy
        next_state = zeros(1,N);
        t = t + Tc;
    end

    State = next_state;

end

norm_thoughput = succ_transmissions*frame_length / t;
collision_pr = collisions / (collisions + succ_transmissions);
```

References

[1] Atheros Open Powerline Toolkit. `https://github.com/qca/open-plc-utils`.

[2] Faifa. `http://github.com/ffainelli/faifa`.

[3] IEEE Standard for Broadband over Power Line Networks: Medium Access Control and Physical Layer Specifications. *IEEE Std 1901-2010*, 2010.

[4] C. Vlachou, J. Herzen, and P. Thiran. Fairness of MAC protocols: IEEE 1901 vs. 802.11. In *2013 17th IEEE International Symposium on Power Line Communications and Its Applications*.

[5] Christina Vlachou, Albert Banchs, Julien Herzen, and Patrick Thiran. On the MAC for Power-Line Communications: Modeling Assumptions and Performance Tradeoffs. In *IEEE International Conference on Network Protocols (ICNP)*, 2014.