MAC Design on Real 802.11 Devices: from Exponential to Moderated Backoff

Ilenia Tinnirello*, Menzo Wentink†, Domenico Garlisi*, Fabrizio Giuliano*, Giuseppe Bianchi‡,
*CNIT/Università di Palermo, Italy
†Qualcomm
‡CNIT/Università di Roma Tor Vergata, Italy

***Abstract*—In this paper we describe how a novel backoff mechanism called Moderated Backoff (MB), recently proposed as a standard extension for 802.11 networks, has been prototyped and experimentally validated on a commercial 802.11 card before being ratified. Indeed, for performance reasons, the time critical operations of MAC protocols, such as the backoff mechanism, are implemented into the card hardware/firmware and cannot be arbitrarily changed by third parties or by manufacturers only for experimental reasons. Our validation has been possible thanks to the availability of the so called Wireless MAC Processor (WMP), a prototype of a novel wireless card architecture in which MAC protocols can be programmed by using proper abstractions and a state-machine formal language, which enable easy modifications of legacy operations. Experimental results are in agreement with simulations and prove the effectiveness of Moderated Backoff, as well as the potentialities of the WMP platform.**

I. Introduction

In the last 20 years the original 802.11 CSMA/CA Medium Access Control (MAC) has been extensively amended for facing the breakthrough rate improvements made available by the latest PHY enhancements (802.11n, 802.11ac, 802.11ad), or the shortcomings related to novel application scenarios, such as ad hoc and mesh networks [1], vehicular environments, directional antennas, quality of service support and real time media streaming [2], and many others. The design of these extensions, as well as other non-standard solutions for adapting the 802.11 MAC to these new challenges, has been mostly supported by simulation and analytical results. The reason is that MAC protocol operations, such as backoff countdown, checksum verification, acknowledgment management, etc., are very time-critical and cannot be implemented via software (e.g. at the driver level) for allowing easy modifications. Only ratified extensions can be experimentally validated once included into a novel NIC release.

Obviously, vendors can implement new protocol features by accessing the hardware and firmware of their NIC products, while researchers can rely on FPGA-based or Software Defined Radio (SDR) platforms, for prototyping the whole MAC/PHY stack from scratch [3]–[5]. Specific software architectures are also available for facilitating the MAC protocol definition on SDR platforms [6], [7], while for some commercial cards it is possible to work on open firmware [8]. However, these solutions require to be familiar with the hardware platforms, are quite time-consuming and could not be considered viable for just testing a protocol proposal.

978-1-5090-2185-7/16/$31.00

In this paper we show an alternative design and prototype path for experimentally validating MAC protocols, based on a programmable architecture of wireless cards called Wireless MAC Processor (WMP) [9]. The architecture allows to define MAC protocols in software, by means of a specialized programming language based on state machines, without sacrificing the execution performance. The idea is decoupling the elementary hardware events and actions required by a MAC protocol (the WMP application programming interface) from the logic according to which the functions are sequentially composed (the MAC program) by a generic executor of protocols (the MAC engine). On top of this platform, MAC protocols can be implemented and modified even more easily than in simulation.

For demonstrating the effectiveness of our platform, we consider a recent extension proposal for the 802.11 MAC protocol, called Moderated Backoff (MB) [10], which requires to modify the NIC backoff process. Moderated backoff has been designed for being compatible with legacy EDCA, because it achieves a comparable throughput by working with an average contention window equal to the one used by legacy stations. The window is tuned as a function of the backoff freezes experienced by each station and exhibits small variations in comparison to exponential backoff, thus resulting in more stable throughput and lower short-term unfairness. The scheme has been implemented, tested and validated on the WMP architecture in a few days. Apart from demonstrating the performance benefits of the scheme, the aftermath is that novel programmable architectures for wireless NICs can overcome the need of protocol standardization, by allowing to move from the design of one-size-fits-all MAC protocols to the design of reconfigurable ad-hoc MAC programs.

The rest of this paper is organized as follows: in section II we introduce the WMP architecture and the high-level programming language for defining MAC protocols; in section III we describe the Moderated Backoff proposal, the general motivations of the scheme and the design rational; in section IV we specify how Moderated Backoff has been prototyped on top of the WMP architecture; experimental validation enabled by our prototype is described in section V and finally some conclusions are drawn in section VI.

II. MAC Prototyping Platform

For prototyping the Moderated Backoff scheme, we use an experimentation platform built on top of a cheap commodity NIC by Broadcom. The platform offers the possibility to easy program, load and execute customized MAC protocols, by using a platform-independent, high-level programming language. This capability is achieved by developing a firmware which does not implement a specific protocol, but rather a generic protocol executor called *MAC Engine*, able to load and run different MAC programs (from legacy DCF and TDMA). The programs specify the so called lower-MAC operations, i.e. the time constrained logic for accessing the medium and react to the reception of a frame. For supporting the upper-MAC operations (associations, scanning, and other management issues), we use the *b43* soft-MAC driver, which adapts the Linux internal *mac80211* interface to network card.

We now briefly review the Wireless Mac Processor general concept. The reader interested in technical details and implementation aspects is referred to the original work [9].

A. The Wireless MAC Processor Architecture

In designing a viable abstraction for wireless MAC protocols specification, the technical hurdle to face is how to formally model the MAC protocol behavior using an high level language, and meanwhile support operations which may require a precision in the order of microseconds (e.g. schedule a frame transmission), and which cannot hence be outsourced to software programs running outside the NIC (e.g. in the driver). Our proposed abstraction is based on the following *decoupling* compromise:

- NIC cards do support an hard-coded (not modifiable by the MAC protocol programmer) *instruction set*, namely an Application Programming Interface (API) comprising of *actions*, *events* and internal parameters upon which *conditions* can be evaluated.
- Third-party MAC programmers formally describe how actions are coordinated and triggered (by events and conditions) via eXtensible Finite State Machines (XFSM - [9]); in essence, the programmer can specify in a formal (executable) model, custom protocol *states*, state transitions and relevant triggering events and conditions, and actions invoked when state transitions occur and/or when a state is reached.
- To execute injected XFSM (suitably compiled into a byte-code-like language), the NIC further implements a generic XFSM processing engine, conceptually analogous to a Central Processing Unit (CPU) in ordinary computing systems, but technically differing in its operation, as its role is to fetch states, parse events, trigger state transitions, and invoke associated actions.

B. MAC Programming Interface

In our experimentation platform, MAC programs are specified as XFSMs, which are built by composing the platform primitive *actions*, in response of specific platform *events* and *conditions* of the internal registers. A revised version of the API originally proposed in [9] is summarized in table I.

events	*actions*	*data*
CH_UP	rx_header()	channel
CW_DOWN	rx_msdu()	antenna
RX_PLCP_END	start_timer(reg,prm)	power
RX_MAC_HEAD_END	extract_bk(reg, prm)	txrx_on
RX_END	tx_start(prm)	backoff_slot
RX_ERROR	update_cw(reg, prm)	rx_chks
QUEUE_OUT_UP	repor_to_host(prm)	busy_time
IFS_EXPIRED	start_ifs(prm)	backoff_value
TX_END	set(reg/var, var/prm)	bandwidth
TIMER_EXPIRED	get(reg/var, var/prm)	slot_time
ACK_TIMEOUT	write(queue, field, var/prm)	+ *protocol*
	read(queue, field, var/prm)	*variables*
	incr(var)	+ *payload*
	hw_reset()	*fields*

TABLE I
WMP application programming interface.

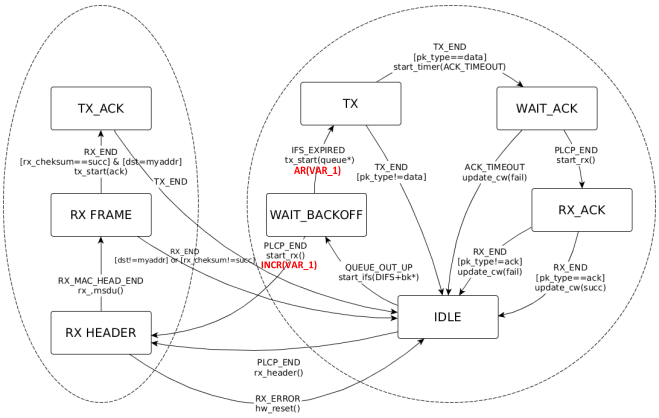


Fig. 1. An example of simplified DCF program defined in terms of XFSM and modifications (red actions) to support Moderated Backoff.

Figure 1 shows a reference example of MAC program. It implements a simplified DCF version supporting only the basic access mode. For sake of readability, the figure also groups the protocol states into the two main reception (on the left) and transmission (on the right) macro states. The operation of the program is somehow straightforward, but we detail the description of the backoff mechanism that will be modified for supporting MB.

Starting from an IDLE state, the program switches to a WAIT_BACKOFF state when a frame is ready in the transmission queue. The availability of a frame in the queue is signaled by the QUEUE_OUT_UP event. The backoff waiting time is set to the value stored in the *bk* register (in case of resumption from a previously paused backoff count-down) or to a new value uniformly extracted in the range indicated by the contention window register. This register is modified at each transmission attempt according to the *update_cw()* function and to the minimum and maximum values configured by the driver. The backoff decrement is performed by the hardware action *start_ifs(DIFS+bk)*. It automatically waits for a DIFS idle time, decrements *bk* of one unit at each subsequent idle interval whose size is set in the *slot_time* register, and stops the decrement when the medium is busy.

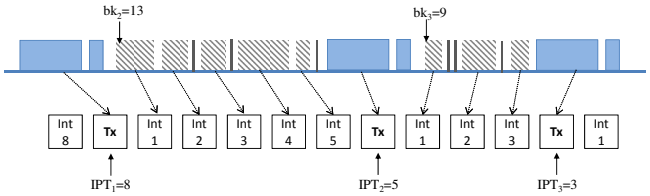


Fig. 2. IPT measurement example

III. Moderated Backoff Design

It has been largely demonstrated that exponential backoff, a mechanism for adaptively tuning the contention window used in random-access protocols (including 802.11 EDCA) suffers of critical performance problems, such as inefficiencies in error-prone channels and show-term unfairness. This last problem can indeed be solved by using a fixed contention window, which guarantees that each station extracts the backoff counters in the same range of values, regardless of the outcomes of the last transmissions. While several studies have proposed to tune such a fixed value for optimizing the network throughput, Moderated Backoff has been designed for being compatible with legacy EDCA, i.e. for working with a fixed contention window equal to the average value achieved by legacy EDCA.

Analytical studies of DCF and EDCA [11], [12] provide a closed form expression relating the average contention window experienced under exponential backoff to channel level measurements, such as the collision probability or the number of busy backoff slots observed during consecutive contentions. Moderated Backoff exploits this last parameter, also called Interruptions Per Transmission (IPT), for tuning the contention window. Figure 2 shows an example of three consecutive transmissions performed by a given stations, by enlightening station transmissions in blue and channel busy intervals due to other stations in gray. In the figure, the first transmission is performed after 8 backoff freezes (not shown in the figure), the second one after 5 freezes and the last one after three freezes, thus corresponding to an average IPT value of (8+5+3)/3=5.3. The example also shows the backoff countdown: for the second transmission, the station extracts a backoff counter bk equal to 13, which is decremented after each idle slot or at the end of each busy interval down to zero before performing the next transmission.

The IPT parameter can be related to the collision probability as discussed in [13]: since in each backoff interruption (busy slot) an eventual packet transmission would have failed, the conditional collision probability can be obtained by considering the total number of busy slots observed in a long time interval as potential collision events, and dividing this sum by the total number of backoff slots counted during the same time interval, i.e. $p \simeq \sum_i IPT_i / \sum_i bk_i$ and from renewal theory it is also:

$$p = \frac{2 \cdot E[IPT]}{E[CW]} \tag{1}$$

.

Obviously, in each backoff countdown the observed IPT_i value is a random variable, which depends on the length of the backoff countdown and on the transmissions performed by the other stations. In case all the stations employ a single access category (i.e. uniform CW_{min} and CW_{max} values), the average contention window and the resulting channel access rate experienced by the stations are all the same. Therefore, it is possible to find the same expression relating the average contention window to the collision probability:

$$E[CW] = \frac{1-p}{1-p^{R+1}} \sum_{i=0}^{R} p^i W_i \tag{2}$$

where W_i is the contention window given by the exponential backoff rules after i collisions and W_0 is the minimum contention window value. Since the average collision rate can also be expressed as a function of average IPT values as in equation 1, it exists an equation $E[CW] = f(E[IPT])$. Such an equation can be approximated in a *calibration curve* by means of a polynomial fitting of some couples of $(E[CW], E[IPT])$ values found in simulation. For example, for $CW_{min} = 15$ and $CW_{max} = 1023$, the second degree calibration curve results equal to $E[CW] = -0.01 \cdot E[IPT]^2 + 3.21 \cdot E[IPT] + 13.92$.

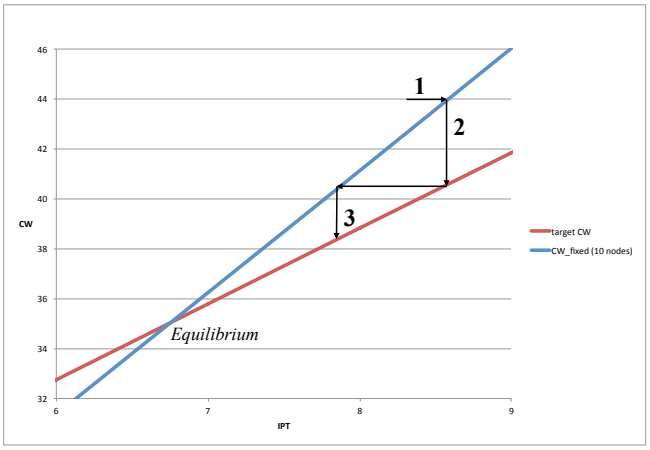


Fig. 3. IPT convergence curve

MB can be implemented by applying the calibration curve to an estimate of the expected $E[IPT]$ value, which is obtained by filtering the random IPT_i measurements performed at each channel access. For example, a first order auto-regressive filter can be used for smoothing the random fluctuations of the IPT measurements while tracking the network load dynamics.

Clearly, when the calibration curve is applied to time-varying filtered values of IPT, the resulting contention window is also time-varying. However, as also shown in the numerical results, the contention window dynamics are much smaller than exponential backoff.

Note that, under MB, each station performs the tuning of the contention window W_{MB} according to its own IPT measurements (which depend on the channel access probability of the other stations and monitoring station). To guarantee that the channel access rate of each station does not change abruptly, another autoregressive filter is used for adjusting the W_{MB} value when a novel IPT sample is available.

In the ideal case when all the stations use the same W_{MB}

value, the convergence of the scheme to the $E[CW]$ value of exponential backoff is intuitively explained in figure 3. The figure shows the $E[IPT]$ values resulting in a network with 10 stations when a fixed contention window is used (blue curve) and the calibration curve of exponential backoff (red curve). If all the stations work with a fixed window equal to 44, after filtering the IPT measurements and applying the calibration curve, the window is updated to 41; now the IPT filtered values will be reduced and the novel contention window value of the calibration curve will be 39. The adjustments are repeated until the equilibrium point is reached.

Summarizing, MB is based on the following equations:

$$ipt = ipt + \alpha \cdot (IPT - ipt); \quad (3)$$

$$targetcw = -0.01 \cdot ipt^2 + 3.21 \cdot ipt + 13.92; \quad (4)$$

$$cw = cw + \beta \cdot (targetcw - cw); \quad (5)$$

which represent i) filtering of the integer number IPT of backoff freezes measured in the last contention; ii) evaluation of the target contention window by means of the calibration curve performing floating-point operations on the filtered ipt values; iii) tuning of the contention window to a filtered value of the target window.

We want to remark that we are not interested in proving that MB is better than other adaptive tunings of the contention windows, but rather we want to focus on the challenges that have to be solved for implementing these simple non-standard rules on commercial 802.11 NICs. .

IV. Moderated Backoff Prototyping

Moderated Backoff is a simple variant of EDCA, in which only the contention window update law is different from the standard one. However, despite the fact that the difference with the standard protocol is confined to a precise function, there are two main problems that prevent its implementation on commercial cards. First, the contention window tuning depends on the IPT measurements which, in principle, can be available only in the lower-MAC by working on the firmware. Second, assuming that it is possible to access the card firmware, the calibration curve includes floating point operations that cannot be performed at this level.

The WMP platform allows to easily solve both the problems and quickly prototyping the novel MB mechanism by exploiting the usual splitting between lower-MAC time critical operations (performed inside the card) and upper-MAC functionalities (performed in the host) and the card registers that can be accessed by the host as a communication interface. Specifically, the IPT measurements can be managed by a customized MAC program, while a control program running on the host can sample these measurements, apply the calibration curve and tune accordingly the contention window registers.

MB Program. The state machine implementing the low-level MB scheme can be obtained as a straightforward variant of the DCF state machine. Figure 1 shows the additional operations performed by the MB program with red actions. On one side, the program adds the action $incr(VAR\_1)$ to the transition from the WAIT_BACKOFF state to RX_HEADER state which is triggered by the reception of a novel frame. This action allows to count consecutive backoff freezes. On the other side, some arithmetic operations are added to the transition from the WAIT_BACKOFF state to the TX state, i.e. at the end of the backoff countdown, for filtering the last IPT measurement and resetting the VAR_1 counter. MB program provide the action $AR(VAR\_1)$ which implements a generic auto-regressive filter. The filter α coefficient is set to $1/8$ for performing the multiplication as a register shift.

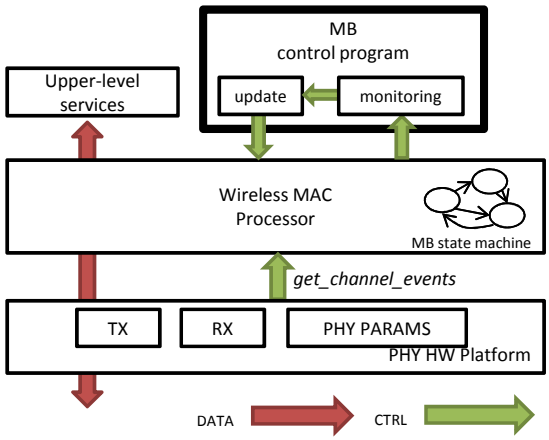
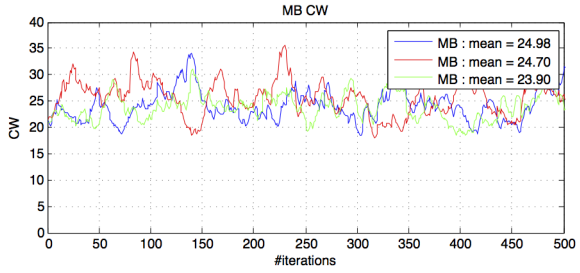


Fig. 4. Implementation of Moderated Backoff on the WMP Platform.

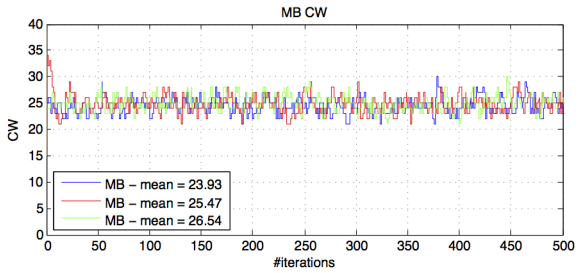
CW Tuning. The evaluation of the target and filtered contention windows are performed by an host control program. The interaction between the program and the WMP platform is shown in figure 4. The control program samples the filtered value ipt of the IPT measurements at regular time intervals of 100 ms (completely decoupled from the updates of ipt values, which are performed at every transmission) by *reading* a given shared register of the WMP platform. After the execution of equations 4 and 5, it tunes the W_{MB} value by *writing* the shared registers corresponding to the CW, CW_MIN and CW_MAX variables. Note that using the same values for the minimum and maximum contention windows allows to practically disenabling the exponential backoff without changing the *update_cw* action.

V. Experimental Validation

Our main goal is demonstrating that our MB implementation works in agreement with the simulation results and, specifically, it is able to achieve throughput performance comparable with EDCA nodes. To this purpose, we set up a mixed network topology in our lab, with three nodes running legacy EDCA competing with three other nodes running the MB scheme. All the nodes, when active, generate greedy traffic flows by running an *iperf* UDP client towards a common Access Point. Since the IPT values used by MB for tuning the contention windows do not depend on the frame size, we run our tests with small frames of 200 bytes transmitted at 24Mbps for min-

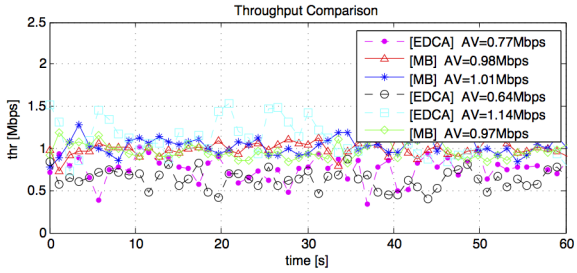


(a) W_{MB} tuning - Simulation

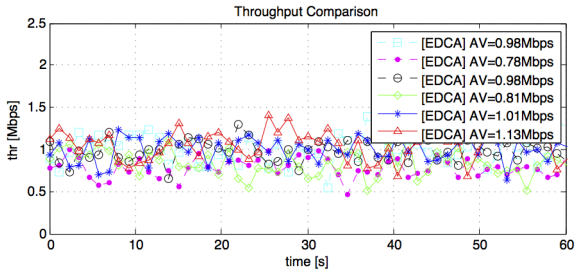


(b) W_{MB} tuning - Experiment

Fig. 5. Comparison on the tuning of the contention window values between simulation and experimental results.



(a) 3 MB and 3 EDCA stations



(b) 6 EDCA stations

Fig. 6. Per-station throughput results when 3 MB stations coexist with 3 EDCA stations and when all stations employ legacy EDCA.

imizing the duration of the backoff freezes[1]. Although all the nodes are in reciprocal visibility, our experiment environment is not interference-free (because of the coexisting University WiFi networks). Therefore, another goal of our experiments is understanding if the compatibility with legacy EDCA is robust in presence of position-dependent interference conditions.

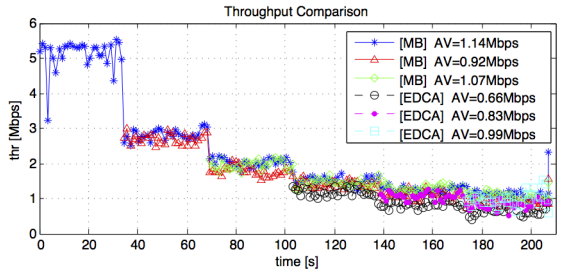
A. Protocol Operation

As a first validation, we compare the contention window values tuned by our implementation of MB with the ones found in simulation. The simulator has been configured for performing consecutive tunings of the cw values (according to equation 5) at the end of each transmission, while the real implementation performs consecutive tunings at regular intervals of 100ms, as described in the previous section. Therefore, the β value of the filter used in equation 5 has been configured to different values (namely, 0.1 in simulation and to 0.7 in the experiment).

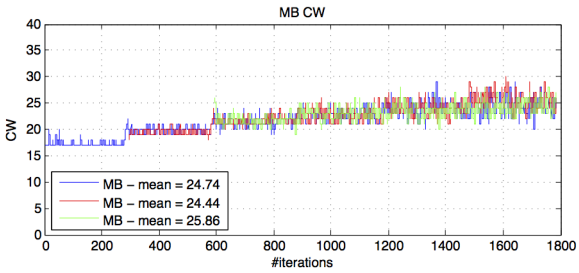
Figures 5(a) and 5(b) shows the time-varying cw results of the stations employing moderated backoff (labeled as Moderated Backoff, MB). Because of the different filter operations, the experimental results are plotted as a function of the time, while the simulation results are plotted as a function of a progressive number of tunings. The figure also quantifies the average value obtained in simulation and in the experiment, which proves that the real implementation works in good agreement with the ideal operations.

For proving the compatibility with the EDCA legacy scheme, figure 6(a) plots the throughput results in a reference scenario with 3 EDCA stations contending with 3 MB stations. The legend also quantifies the average throughput. From the

[1]This setting makes more critical the collection of IPT measurements, since the time between two consecutive transmissions performed by the same station is very short.



(a) Throughput

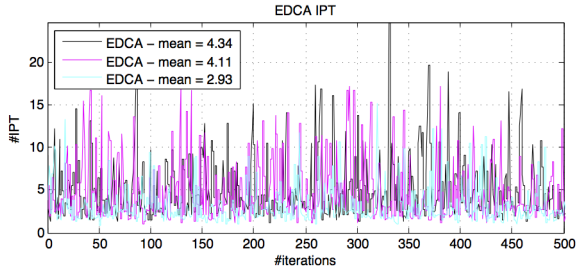


(b) Contention Window

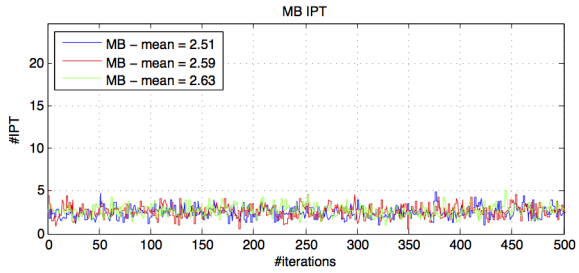
Fig. 7. Per-station throughput results and contention window tunings in the real experiment with sequential activation of the stations.

figure we can observe that the throughput perceived by different stations are not exactly the same; however, this variability is not due to the MB scheme, but rather to the specific link conditions perceived by each station in a real environment. This conclusion is proved in figure 6(b), where we plot the throughput perceived by the same stations, in an experiment carried out a few minutes after the previous one, when all the stations employ legacy EDCA. We can see that also in this case the throughput results are not the same, despite the fact that EDCA is in principle throughput-fair.

Figure 7(a) shows the throughput results in the same mixed



(a) IPT filtered measurements - EDCA



(b) IPT filtered measurements - MB

Fig. 8. Comparison between IPT experimental measurements gathered by EDCA and MB stations.

Rate	#MB	#EDCA	THR [Mbps]	E[CW]
24Mbps	0	14	0.3922	41, 1922
24Mbps	7	7	0.3623	43, 2854
24Mbps	14	0	0.3969	40, 7475

TABLE II
PER-STATION THROUGHPUT IN A NETWORK SCENARIO WITH 14 NODES.

scenario (3 EDCA and 3 MB stations), when each network station is activated sequentially. We can observe that the per-station throughput in different time-intervals depend on the number of active stations, regardless of the specific moderated or exponential backoff employed by the stations. The corresponding dynamics of the contention window values tuned by the MB stations in the real experiment are shown in figure 7(b). The results prove that MB implemented in our system is reliable and does not introduce latency or transient performance unfairness.

Figures 8(a) and 8(b) show the filtered values of the IPT measurements carried out by EDCA and MB stations. The measurements are used for tuning the contention window value of MB stations, but can also be considered as a performance metric of the protocol short-term unfairness. As evident from the figure, MB stations suffer of a number of backoff interruptions that is much more stable than EDCA stations.

Finally, in table II we consider a larger scale network scenario, with 14 contending nodes. We run three experiments in which we vary the number of stations running legacy EDCA and MB. In all the cases, we observe no significant differences on the per-station throughput and average values of the contention window.

VI. CONCLUSIONS

In this paper we demonstrated that non-standard MAC protocols for 802.11 networks can be experimentally validated by exploiting the emerging architectures for programmable wireless NICs, such as the Wireless MAC Processor architecture, and an opportunistic split of time-critical and non-time-critical functionalities. Indeed, the WMP platform easily allows to prototype a MAC program performing customized access operations and statistics, by using an high-level programming language which completely hides the complexity of the card internals. Moreover, the MAC program can expose some variables in specific registers that can be accessed by the host, for implementing an effective interface with high-level operations requiring complex processing. As a specific case study, we implemented the Moderated Backoff scheme and experimentally confirmed its compatibility with legacy EDCA.

ACKNOWLEDGEMENT

The research has been partially funded by the European Horizon 2020 Programme under grant agreement n645274 (WiSHFUL project) and n671563 (Flex5Gware project).

REFERENCES

[1] T. Imboden, K. Akkaya, and Z. Moore, "Performance evaluation of wireless mesh networks using ieee 802.11s and ieee 802.11n," in *Communications (ICC), 2012 IEEE International Conference on*, June 2012, pp. 5675–5679.

[2] K. Kosek-Szott, M. Natkaniec, S. Szott, A. Krasilov, A. Lyakhov, A. Safonov, and I. Tinnirello, "What's new for qos in ieee 802.11?" *Network, IEEE*, vol. 27, no. 6, pp. 95–104, November 2013.

[3] A. Khattab, J. Camp, C. Hunter, P. Murphy, A. Sabharwal, and E. W. Knightly, "Warp: A flexible platform for clean-slate wireless medium access protocol design," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 12, no. 1, pp. 56–58, Jan. 2008. [Online]. Available: http://doi.acm.org/10.1145/1374512.1374532

[4] USRP: Universal Software Radio Peripheral, accessed February 2012. [Online]. Available: http://www.ettus.com/

[5] K. Tan, J. Zhang, J. Fang, H. Liu, Y. Ye, S. Wang, Y. Zhang, H. Wu, W. Wang, and G. M. Voelker, "Sora: High performance software radio using general purpose multi-core processors." in *NSDI*. USENIX Association, 2009, pp. 75–90.

[6] X. Zhang, J. Ansari, G. Yang, and P. Mhnen, "Trump: Supporting efficient realization of protocols for cognitive radio networks," in *New Frontiers in Dynamic Spectrum Access Networks (DySPAN), 2011 IEEE Symposium on*, May 2011, pp. 476–487.

[7] P. D. Sutton, J. Lotze, H. Lahlou, S. A. Fahmy, K. E. Nolan, B. Ozgul, T. W. Rondeau, J. Noguera, and L. E. Doyle, "Iris: an architecture for cognitive radio networking testbeds," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 114–122, Sept 2010.

[8] Open firmware for WiFi networks. [Online]. Available: http://www.ing.unibs.it/openfwwf/

[9] I. Tinnirello, G. Bianchi, P. Gallo, D. Garlisi, F. Giuliano, and F. Gringoli, "Wireless MAC processors: programming MAC protocols on commodity hardware," in *INFOCOM, 2012 Proceedings IEEE*. IEEE, 2012, pp. 1269–1277.

[10] Moderated Backoff - Menzo Wentink, Setember 2015. [Online]. Available: https://mentor.ieee.org/802.11/dcn/15/11-15-1152-01-000m-moderated-backoff.ppt

[11] G. Bianchi and I. Tinnirello, "Remarks on IEEE 802.11 DCF performance analysis," *IEEE Communications Letters*, vol. 9, no. 8, pp. 765–767, August 2005.

[12] I. Tinnirello and G. Bianchi, "Rethinking the ieee 802.11e edca performance modeling methodology," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 540–553, Apr. 2010.

[13] G. Bianchi and I. Tinnirello, "Kalman filter estimation of the number of competing terminals in an ieee 802.11 network," in *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, vol. 2, March 2003, pp. 844–852 vol.2.