# Process-Stacking Multiplexing Access for 60 GHz Millimeter-Wave WPANs

Claudio Estevez[†], David Fuentealba[†], Aravind Kailas[‡]

[†] Department of Electrical Engineering, Universidad de Chile, Santiago, Chile | {cestevez, dfuentea}@ing.uchile.cl

[‡] Electrical and Computer Engineering Department, UNC Charlotte, Charlotte, NC, USA | aravind.kailas@uncc.edu

***Abstract*—Millimeter-waves technology shows high potential for future wireless personal area networks, reaching over 1 Gb/s transmissions using simple modulation techniques. This technology is on the innovation stages; therefore current specifications consider dividing the spectrum into effortlessly separable spectrum ranges. In this work, a process-stacking multiplexing access algorithm is designed for single channel operation. The concept is intuitive and simple, but its implementation is not trivial. The key to stacking single channel events is to operate while simultaneously obtaining and handling a-posteriori time-frame information of scheduled events. This information is used to shift a global time pointer that the wireless access point manages and uses to synchronize all serviced nodes. The performance of the proposed multiplexing access technique is lower bounded by the performance of legacy TDMA and can improve the effective throughput significantly. Detailed implementation is presented and hypothesis is validated by simulation results.**

***Keywords-60 GHz; access; MAC; millimeter; mm-wave; multiplexing; process-stacking; WPAN.***

## I. INTRODUCTION

Throughout history, technologies have demonstrated a logistic function growth pattern, commonly referred as the technology s-curve. The s-curve is composed of an innovation, improvement, maturity, and aging stage. In the wireless local area network (WLAN) field, the current dominating technology is WiFi, which has a 2.4/5 GHz modulating frequency with typical channel widths of 20/40 MHz, and can reach throughputs of 600 Mb/s [1]. To achieve these speeds WiFi requires multiple-input multiple-output (MIMO) antenna arrays, orthogonal frequency division multiplexing (OFDM), and relatively dense quadrature amplitude modulation (QAM) constellations, which indicate that this technology is reaching the maturity stages of the technology s-curve and in the future it will become increasingly difficult to make significant throughput improvements. For this reason, millimeter-wave technology is rapidly becoming the new alternative for wireless personal area network (WPAN).

mm-Wave systems have already proven to transmit at over 2.5 Gb/s [2], using single-input single-out (SISO) antenna setup and on/off keying (OOK) modulation. Considering the simple methods employed to achieve this throughput, this can be categorized to be in the innovation stages of the technology s-curve. This is a good indication that when this technology reaches maturity it could potentially transmit at over 40 Gb/s by using spectrally-efficient modulation methods, dense frequency multiplexing, wide bandwidths due to frequency reuse, efficient use of time allocation, MIMO, beamforming algorithms for spatial filtering, and other potential techniques.

The 60 GHz frequency range is attractive to very-high-throughput applications found in research fields such as: WPANs [3][4], e-Health [5], and home entertainment. Some examples are: cloud computing services, web-based file hosting, health related traffic (a single high-quality multi-focal microscope picture format can occupy 1 GB), distribution link to a body area network, low-latency high-capacity gaming capabilities, and high-definition video-on-demand. Because this technology is still callow many of these applications are still not implemented but are feasible.

Initial attempts to standardize the 60 GHz frequency range already exists. Among these standards the ECMA-387 [6] is one the best known. The ECMA-387 discusses issues related to the physical and data-link layers. One particular interest to our work is the band allocation. This initial attempt by ECMA-387 to standardize the 60 GHz region divides the operating frequency range (57 – 66 GHz) in four bands: 57.240 ↔ 59.400 ↔ 61.560 ↔ 63.720 ↔ 65.880, as seen in Figure 1. One motive for having a low number of bands is that filter design becomes less trivial and more expensive as the Q-factor increases and since the design is for WPAN it is not expected to have large number of users per wireless access point (WAP). For this reason, the focus of this work is to design a time-domain multiplexing technique with efficient allocating capabilities.

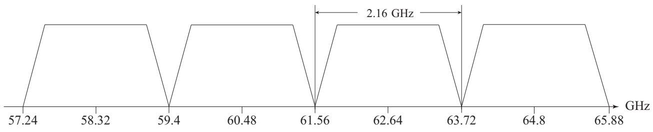


Figure 1. ECMA-387 operating frequency range and band allocation.

To multiplex various transmitting nodes into a single channel, a process-stacking multiplexing access (PSMA) algorithm is proposed. If frequency reuse is incorporated by taking advantage of the topology, as in [7], a two-dimensional space/time domain algorithm can be achieved, but this work focuses on improving the time dimension efficiency. The reason for naming this access technique process-stacking is because its versatility expands the packet-transmission-time reservation to a diverse-process-time reservation ideology. This allows for easy incorporation of any process that wishes to reserve access to the antenna, even if it does not require to transmit data.

Various useful processes could be implemented without modifying the algorithm, a few examples are: idle time, QoS and beacon signal. In many environments, particularly in those associated with e-Health, energetically self-sustainable nodes are desirable. If the node were equipped with an energy-harvesting device and it drained most of its energy, an idle-time process can be inserted into the packet transmission cycle to allow the device to recharge, by trading-off throughput, as suggested in [8][9]. The advantage of PSMA over TDMA (used in [8][9]) is that the idle-time process can be inserted at any time that the system determines is necessary while TDMA divides the cycle in slots and if it needs to enter the energy-savings mode it needs to undo the slot assignments. Another advantage of process-stacking is that QoS can be implemented in several ways. For example, if the information is labeled as time sensitive, the algorithm can switch to smaller and more frequent process reservations such that the effective throughput remains the same but the stream is more continuous rather than bursty. Also, different priority level traffic can have different reservation frequency privileges and different reservation time frames. The final example is the use of a broadcast synchronization signal, which some literature refer to beacon signal. If a collision is detected by the WAP, usually due to a new node attempting link, the WAP can insert a beacon signal and idle time (to wait for a response) process allowing the unlinked node to connect. A graphical comparison between TDMA and PSMA is portrayed in Figure 2.

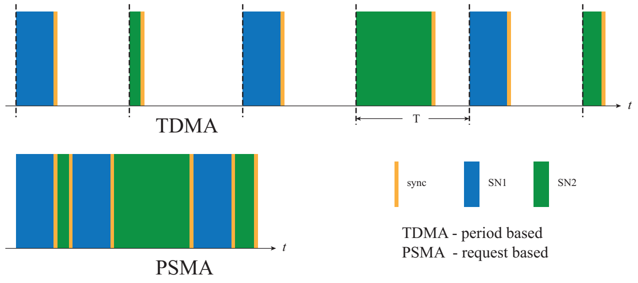


Figure 2. Comparison between time allocation schemes PSMA vs TDMA

There are many advantages to having a process-stacking multiplexing access (PSMA) algorithm: 1) Time is used more efficiently: In a TDMA regime each node has a fixed time-slot and the width of the slot is independent of the amount of data that is queued, in worse-case scenario there is no data queued and the time slot goes unused. In PSMA each node occupies the channel for the amount of time required, hence does not occupy time if it does not need to transmit. 2) No limit for amount of nodes per channel: In TDMA, the period is divided into N time slots and each node has a dedicated slot, if an (N+1)th node requests transmission there will be no slots available. In PSMA, there is no limit, but having an excessive amount of nodes will result in a low throughput per node. 3) It is versatile; In TDMA, if a process or control signal needs to be inserted into the channel, the recurrent period needs to be interrupted and reconfigured. In PSMA, if distinct processes (other than scheduling nodes) need to reserve the antenna in a PSMA scheme, the WAP is designed to perform this task without disrupting the transmission of the already linked nodes. 4) Quality of service (QoS) integration: In TDMA, QoS is limited to assigning more time-slots to higher priority nodes, but the configurable granularity is limited to units of time-slots. In PSMA, as discussed previously, it can be implemented using various techniques.

This paper is organized in the following order: Section II describes the proposed PSMA algorithm and explains how it was implemented. In Section III, the simulation scenario is described and results are presented. Finally, in Section IV the conclusions are summarized.

## II. Process-Stacking Multiplexing Access (PSMA)

PSMA functions using a different technique than TDMA; the only similarity is that both perform multiplexing access in the time domain. PSMA schedules processes, which means that time is reserved only when it is requested and for the amount of time requested. To organize transmissions a global time pointer is used. In the event that a process or node requests the use of the antenna, this process is scheduled at the time stored in the global pointer and the pointer is shifted by the amount of time requested, in practice a buffer time is also inserted. If the current time reaches the global pointer time, then the node switches to listen mode. This is the basic operation, but the details are found in the WAP section below. The serviced node also plays an important role, as it must report to the WAP the reserved time window.

The detailed operation of the nodes follows. For simplicity only the upstream process is described. The half-duplex mode can be obtained by extending the process reservation procedure to the downstream direction. If both stream directions are using the same modulation frequency, this implies that the throughput in both directions will decrease due to the sharing of the channel.

The network topology consists of a wireless access point (WAP) and four serviced nodes (SNs) as shown in 0 All nodes operate at the same frequency in both upstream and downstream directions, for reasons mentioned in the introduction. As the technology matures the spectrum can be divided into narrower bands (than ECMA-387) and models can have control packets in a different (smaller bandwidth) channel, or separate upstream and downstream channels for full-duplex transmission.

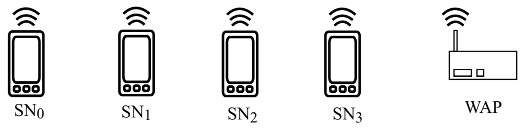


Figure 3. Modeled network topology

### A. Serviced node (SN) operation

Packets flow through the following elements: incoming port, queue, packet encapsulator, main processor, and antenna, as shown in Figure 4. It should be pointed out that this model uses the same modulating frequency for upstream and downstream traffic, therefore when a frame is sent through the transmitter its own receiver captures this frame returning it to the main processor. These frames are referred to as echo frames and they are discarded.

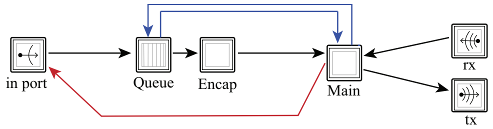


Figure 4. Internal arquitecture of the serviced node

### 1) Queue

If a packet enters the queue, the queue processor informs the main processor it has at least one packet that needs to be serviced. The main processor connects the SN to the WAP; once connected the main processor sends a request to the queue. The queue forwards all stored packets (up to defined limit) to the packet encapsulator followed by a completed-signal.

### 2) Packet Encapsulator

The packet encapsulator has a crucial role in the PSMA scheme. It gathers the network layer packets (typically IP) coming from the queue and encapsulates these into a frame. Once the frame is built, its size is translated into a synchronization time using the transmission bit rate. The time, which is used to synchronize this frame, is inserted in a previously stored frame. The previous frame is sent to the main processor for transmission, while the recently encapsulated frame is temporarily stored. Since the recently built frame triggers the transmission of the previously stored frame, this process is referred to as a push (i.e. the arriving frame pushes the stored one). This continues until the queue is empty. The frame format is composed of a header, data, and ECC field, as shown in Figure 5. The header is composed by the source, destination, flags, and sync fields. The flags are: hello and close. Additional flags can be reserved for QoS and/or other implementations. The methodology, including the use of the frame field, is explained in detail ahead.

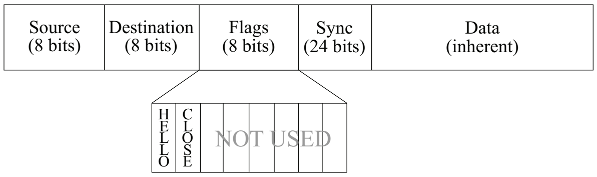


Figure 5. Frame format for PSMA

### 3) Main Processor

The process-stacking multiplexing access algorithm is located in this processor. The main processor is composed of various states, as shown in Figure 6. Overall, the processor collects the frame and sends it at the synchronization time specified by the WAP. The details of each state follow:

**init**: Initializes state variables, assigns a unique MAC address to the node, and schedules a starting time based on a uniformly distributed random variable (within a specified range).

**hello timer**: Awaits for an interrupt. If the interrupt is a self-timer, a signal is sent to queue to release the stored packets to the encapsulation process where (besides encapsulating) the packets are counted and temporarily stored (as described in Section II-A-2) and pushes an empty packet that contains the synchronization information of the next packet. If the interrupt is a packet arrival from the encapsulation processor, it extracts the sync information and discards the packet. Afterwards, it moves to the hello state.

**hello**: Creates first packet with frame format, also called the hello frame in this work. It inserts its own MAC address in the source field, 0 in the destination field (since it does not know the WAP's MAC address at this stage), the hello flag is set to 1, and the first sync information is appended. The hello frame is sent to the antenna port. It moves to the id state after completing these steps.

**id**: Waits for frame [hello response]. If a frame arrives, the header information is extracted: If the hello flag is set to 1 and destination field is its own MAC address then sender field is stored as the WAP MAC address, the first synchronization information is extracted, and the node is synchronized accordingly. If sender field has stored its own MAC address this is an echo frame, therefore discarded. If LOST condition is met it moves to the hello state. It moves to idle state if SYNC condition is met.

**idle**: Waits for an interrupt. If the interrupt is due to a frame arrival then the header information is extracted before moving to the SYNC state. If the interrupt is a self-timer then it moves to the tx state.

**sync**: Upon the frame arrival, the destination field is examined to check if it is equal to its own MAC address. If this is the case, the sync field is extracted and the next transmission time is scheduled. If the destination field does not match the node's MAC address, the frame is ignored. Afterwards it returns to idle state.

**req_pkt**: Upon entering this state the main processor signals the queue processor to release the packets and it waits for the frame. Once a packet arrives, it checks if it is the frame from the encapsulation processor, if so it extracts the close flag field and moves to the tx state, otherwise it returns to the state (to wait for the frame).

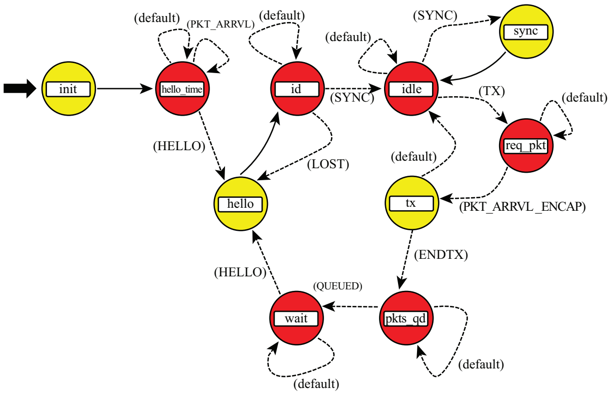


Figure 6. State diagram for the serviced node main processor.

**tx**: The header information is appended to the frame, which already contains the synchronization and data information. Subsequently, it is sent to the antenna port. If ENDTX condition is set it moves to the pkts_qd state, otherwise it moves to the idle state.

**pkts_qd**: Once it enters this state it monitors the queue, if a packet enters the QUEUE condition is met and it assigns a random (but small) time to reestablish the link with the WAP. It then moves to the wait state.

**wait**: Waits for hello timer before moving to hello state [and reestablish another session].

### *B. Wireless Access Point (WAP) operation*

The WAP contains a main processor (WAP label) and a decapsulation processor, as shown in Figure 7.

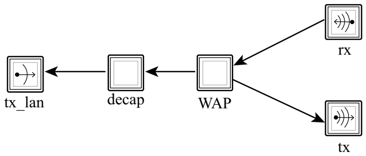


Figure 7. Internal arquitecture of the wireless access point.

#### *1) Main Processor*

The WAP's main processor links all nodes requesting service to the distribution network and organizes the transmission order. It uses a global time pointer that shifts each time a new event is scheduled. If current time is greater than the pointer time it means that no events are scheduled and the WAP goes into the wait state. It is designed to recover from collisions and, since all nodes are being multiplexed into the same channel, collisions are expected.

The main processor is composed of several states, which are shown in Figure 8. The operations performed at each state are described below:

**init**: Initializes state variables and assigns a unique MAC address to the WAP.

**wait**: Awaits for first serviced node to connect. Upon a packet arrival the header is extracted and the time pointer is set to the current time.

**sync**: This state is entered only if a frame is received. Upon reception, it checks if the frame is an echo frame, in this case it is discarded, otherwise it performs these operations: extracts the hello flag to determine if the node is new, in which case it stores in the order log and adds one to the number of currently connected nodes. Then it extracts the close flag to determine if node is closing the session, if its closing it checks if there are any remaining nodes connected, if no other nodes are connected it moves to the wait state, otherwise it moves to the idle state. Also, if it is closing it fixes the order log to remove this node and updates the number of currently connected nodes. If the close flag indicates it is not closing, then the lost expiration time is scheduled (self-timer), a synchronization frame is created, the fields are written including the sync field using the global time pointer, the frame is sent to the antenna port, and the global time pointer is shifted.

**idle**: Awaits for interrupt. If a frame arrives, the header information is extracted. The synchronization order is checked and if the expected packet arrived the log is updated and it moves to the sync state. If no frames arrive before a predetermined deadline, a self-timer triggers and the PKT_LOST condition turns true, forcing it to move to the lost state. If the interrupt is a broadcast timer (set after entering the lost state), it will move to the bcast state.

**lost**: It enters this state only if a frame was lost. Once entered it stores the expected MAC address in the destination field of a newly created frame, it sends the frame to the antenna port, and updates the synchronization order. Since the sync field of the lost packet could not be retrieved the time pointer is shifted to the maximum allowable time window (known by the serviced nodes) and a self-timer is set with the new deadline. Also, a beacon message is scheduled. The reason for this is that if the packet was lost due to collision a new node is trying to enter the cycle and the beacon process will allow it to sync. The global time pointer is shifted to include this beacon message and its response.

**bcast**: Creates and sends a beacon frame to the antenna port.

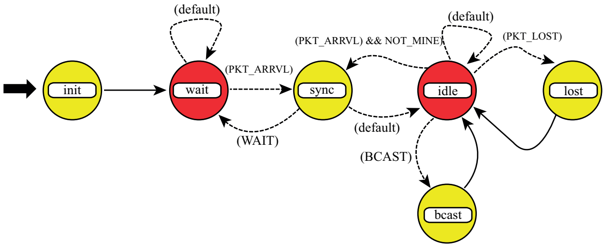


Figure 8. State diagram for the serviced node main processor.

### *C. Communication Dynamics of PSMA*

This section explains the inter-process communication of the SN processors and a broad view of the frame exchange between SN and WAP. These two mechanisms are essential in the operation of PSMA.

#### *1) Inter-process Communication in the SN*

To support PSMA, the SN must include the synchronization information of the subsequent frame into each frame. To accomplish this task a well synchronized and systematic routine is implemented, described ahead and portrayed in Figure 9. After resetting the system the queue is empty, the encapsulation processor has an empty frame and the main processor has nothing to transmit. Upon a packet arrival at the queue, the queue signals the main processor. The main processor initiates the linking process between the SN and WAP by creating an empty hello frame and signals back to the queue that it is ready to receive frames. Once the queue receives the signal, it releases all queued packet (up to certain limit) and follows it with an empty packet with a close instruction. Once the packets start arriving at the encap, it creates a new empty frame where it encapsulates these packets. When the close instruction arrives, the frame size is multiplied by the bit rate to compute the time it will take to transmit the frame. This time is stored in the sync field of the empty frame that was initially in the encapsulation process (at reset). The empty frame is sent to the main processor where the sync time is extracted and included in the hello frame. The hello frame is sent to the WAP. The WAP assigns the next available time to transmit indicated by the global time pointer. It sends this information through the sync frame. The SN, upon receiving the sync frame, schedules the next transmission time. Once this time is reached the main processor signals the queue to release the

ISBN: 978-1-61208-205-9

queued packets and the cycle is repeated, with the only difference that in the encapsulation processor there is a queued frame ready to receive the frame size of the subsequent frame. This routine iterates until the queue is emptied. When this happens the queue only sends the close instruction. This informs the encap that there are no more packets to transmit and the encap sets the close flag of the last frame. The close flag is eventually read by the WAP, which does not schedule further occurrences for this SN until a new request arrives.

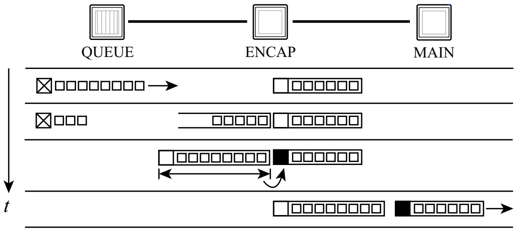


Figure 9. Inter-process Communication in the SN

### 2) *Frame Exchange between the SN and the WAP*

To stack the frames the following rules are applied, see Figure 10. The task of inserting the first SN is trivial, since there are no active transmissions. If various nodes are sharing the channel and an inactive node turns active, such that is must enter the sharing cycle, it will send a hello frame. The processes are stacked such that there is insufficient time to transmit in between frames; this will cause collisions[†]. A collision-free MAC protocol can be designed by incorporating cyclic beacon signals, but since this is an infrequent event and this measure consumes time its usefulness depends on the specific application. When collisions occurs, the WAP sends a new schedule time to the expected SN with the maximum allocation time (assuming the sync field could not be recovered). Since this is the last event a beacon signal is scheduled after this allowing enough time for a hello frame to be retransmitted.

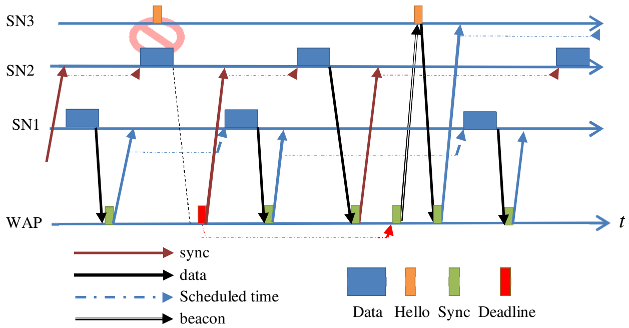


Figure 10. Frame Exchange between multiple SNs and the WAP.

Because there is a scheduled SN the unlinked SN has to wait until the beacon signal to establish a link. Once the WAP has serviced and rescheduled the linked SNs, it sends the beacon signal. The unlinked SN detects that this is a broadcast message (not a hello frame response) and retransmits the hello frame and holds on to the data until the next cycle. The WAP links the SN and from there on frame exchange returns to its routine, but with an additional SN.

## III. Results

The simulation scenario is portrayed in Figure 3. It consists of one WAP servicing 4 SNs. All nodes have a bit rate of 1 Gb/s. The scenario is built in OPNET modeler 16.0. The first test is to demonstrate that the WAP is capable of synchronizing the SNs in the WPAN network.

### A. *Scenario 1 - Systematic Synchronization Capability*

The starting times are pseudorandom with a uniform distribution within a 0.2 second interval. The starting times of the four SN nodes are: 0.0108, 0.0705, 0.1129, and 0.1493 seconds, but do not transmit data until 0.0108, 0.0901, 0.1488, and 0.2075 seconds, respectively, as observed in Figure 11. This simulation run has several interesting aspects: SN1 enters at the same time it requests to enter since it is the only active node at that instant. At 0.0705 s SN2 requests the use of the channel and causes a collision. When the deadline for SN1 expires it is rescheduled; since it is the only node this occurs immediately. It is assumed that upon collisions the sync field is unrecoverable, so the maximum allocation time is granted (in this case 8.3 ms) for the next transmission. It can be seen that SN1 does not require the full allocated time and a gap is produced. Near the end of the gap a beacon frame is sent and the SN2 is linked, but does not transmit data until the following iteration (shown in red). SN3 is requests to use the channel at time 0.1129 s. It can be seen that it collides with SN2. SN2 is rescheduled and following this the beacon frame (near 0.13 s), at which time SN3 is linked. In the next cycle SN3 transmits data. SN4 achieves a link in the same way SN2 and SN3 accomplish it. This shows the implementation is success at establishing links, handling collisions and organizing transmissions in a systematic manner.

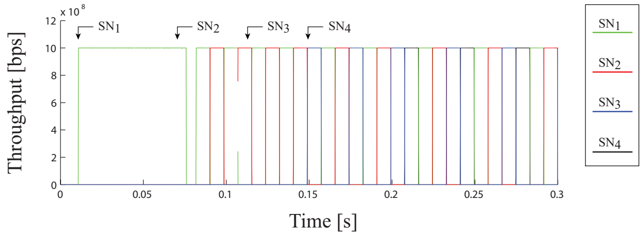
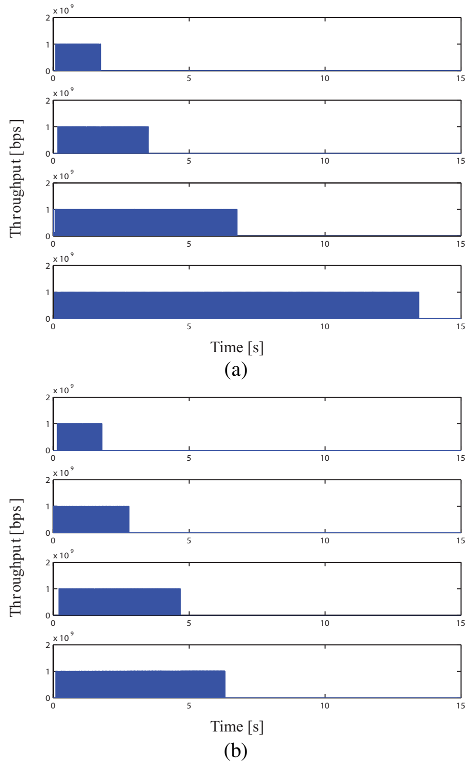


Figure 11. Interlacing of serviced nodes data multiplexed with PSMA.

### B. *Scenario 2 – Diverse SN Load Transmission*

To demonstrate the performance capabilities of PSMA it is compared with the legacy TDMA. In this scenario SN1 transmits 34950 packets, SN2 69900 packets (SN1x2), SN3 139800 packets (SN1x4), and SN4 279600 packets (SN1x8), and the results are shown in Figure 12. The implementation of TDMA used reserves a time slot for each node and does not have knowledge about the amount of packets that need to be serviced. In this scenario, TDMA takes approximately 13.3 seconds to complete all transmissions. Because PSMA stacks all pending events, it achieves to transmit the same load in approximately 6.25 seconds, nearly half the time consumed by TDMA.



(a)

(b)

Figure 12. Diverse load transmission using (a) TDMA and (b) PSMA

### *C. Scenario 3 – Relative Percentage Loads*

In this scenario, SN4 transmits a fixed number of packets (699000), while all remaining nodes transmit a percentage of this load. Results are shown in Figure 13.

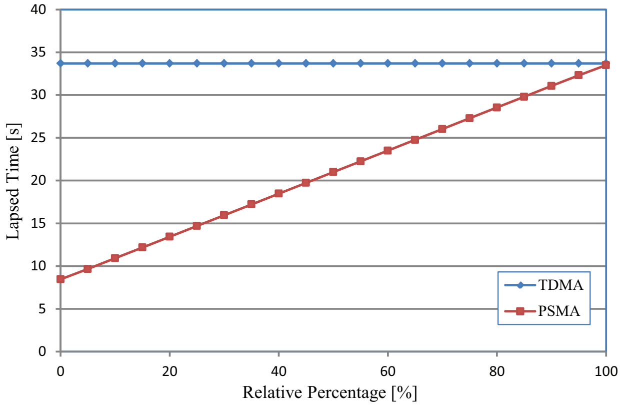


Figure 13. Transmission duration time for varied load percentages.

Since TDMA reserves the slot times independently of there usage, SN4 is unable to take advantage of the unused portions. Regardless of the amount of traffic delivered by the other nodes (as long as it is less than SN4), SN4 will occupy the same amount of time, in this case 33.48 seconds. For the case of PSMA, the processes are stacked, so there is no unused time. If no packets are transmitting, SN4 will use the full channel bandwidth. Compared with TDMA, it will improve the effective throughput by a factor proportional to the number of nodes present in the network.

## IV. CONCLUSIONS

PSMA is an efficient time-domain multiplexing access technique. It has many advantages over legacy TDMA: PSMA has a very efficient use of time as it not only stacks processes that actively require the use of the channel but it allocates only the necessary amount of time for this process to complete. The proposed access algorithm does not have a theoretical node limit as there is no predefined number of time slots, but individual throughput is sacrificed. PSMA is versatile in the sense that in can easily schedule processes not related to data exchange without affecting significantly the transmission of the linked SNs. PSMA supports QoS, as different priority levels can have different benefits, e.g., time duration of allocations and scheduling frequency.

## ACKNOWLEGMENTS

Partially funded by U-INICIA VID 2011 program, grant U-INICIA 11/14; Vicerrectoría de Investigación y Desarrollo, Universidad de Chile.

## REFERENCES

[1] H. Kano, S. Yoshizawa, T. Gunji, T. Saito, and Y. Miyanaga, “Development of 600 Mbps 2×2 MIMO-OFDM baseband and RF transceiver at 5 GHz band,” International Symposium on Communications and Information Technologies (ISCIT), Tokyo, Japan, Oct 2010, Page(s): 891-894.

[2] H.-C. Chien, Y.-T. Hsueh, A. Chowdhury, J. Yu, and G.-K. Chang, “Optical Millimeter-Wave Generation and Transmission Without Carrier Suppression for Single- and Multi-Band Wireless Over Fiber Applications,” Journal of Lightwave Technology, vol. 28, no. 16, pp. 2230-2237, 2010.

[3] S. Singh, F. Ziliotto, U. Madhow, E. M. Belding, and M. J. W. Rodwell, “Millimeter Wave WPAN: Cross-Layer Modeling and Multihop Architecture,” IEEE International Conference on Communications, Dresden, Germany, June 2009.

[4] C.-S. Sum, R. Funada, J. Wang, T. Baykas, M. A. Rahman, and H. Harada, “Error Performance and Throughput Evaluation of a Multi-Gbps Millimeter-Wave WPAN System in the Presence of Adjacent and Co-Channel Interference,” Vol. 27, No. 8, pp. 1433-1442, 2009.

[5] C. Estevez, W. Jian, A. Kailas, D. Fuentealba, G.-K. Chang. “Very-High-Throughput Millimeter-Wave System Oriented for Health Monitoring Applications,” IEEE Healthcom, Columbia, MO, June 2011.

[6] ECMA International, “Standard ECMA-387: High Rate 60 GHz PHY, MAC and HDMI PALs,” 2nd Ed., Dec. 2010.

[7] C.-S. Sum, Z. Lan, R. Funada, J. Wang, T. Baykas, M. A. Rahman, H. Harada, and S. Kato, “Virtual Time-Slot Allocation Scheme for Throughput Enhancement in a Millimeter-wave Gbps WPAN Cross Layer Design,” IEEE International Conference on Communications, Dresden, Germany, June 2009.

[8] W. Jian, A. Chowdhury, Z. Jia, C. Estevez, G.-K. Chang, “Energy-Efficient Multi-Access Technologies for Very-High-Throughput Avionic Millimeter Wave, Wireless Sensor Communication Networks,” Journal of Lightwave Technology, Vol. 28, No. 16, pp. 2398-2405, Aug. 15, 2010.

[9] W. Jian, C. Estevez, A. Chowdhury, Z. Jia, G.-K. Chang, “A Hybrid MAC Protocol Design for Energy-Efficient Very-High-Throughput Millimeter Wave Wireless Sensor Communication Networks,” Asia Communications and Photonics Conference and Exhibition, Shanghai, China, December 2010.