Neighbor Discovery Using Galois Fields and its Hardware Implementation

Turhan Karadeniz, Ashok N Masilamani, J.J. Garcia-Luna-Aceves
Department of Computer Engineering
University of California, Santa Cruz
Santa Cruz, CA 95064
Email: tkaradeniz, ashok, jj@soe.ucsc.edu

***Abstract*—Neighbor discovery is essential in all channel access protocols based on transmission scheduling, whether such protocols are topology-dependent or topology-independent. We propose a novel approach to neighbor discovery based on relative GPS coordinates (RGPS), extending the work by Chlamtac and Farago on topology-transparent transmission scheduling. The proposed approach attains neighbor discovery with timing guarantees, which yields greater scalability and requires smaller transmission compared to neighbor discovery mechanisms based on probabilistic methods. We further present the design and implementation of a hardware accelerator for evaluating the Galois field polynomials used in the proposed neighbor-discovery scheme.**

I. Introduction

All medium access control (MAC) protocols aim to improve the transmission delays and throughput of nodes in the presence of multiple access interference (MAI). As Section II summarizes, almost all MAC protocols are based on contention, scheduling or reservation schemes, and all these approaches select transmission times for nodes in order to eliminate collisions. While contention-based MAC protocols avoid collisions by carrier sensing and the transmission of control packets to clear the channel, most contention-free MAC protocols avoid collisions by assigning slots to nodes by either scheduling or reserving time slots on a synchronized transmission frame. A common challenge to assigning time slots to nodes in all these schedule-based MAC protocols is neighbor discovery when new nodes join the network.

In this paper, we present a new scheme for neighbor discovery based on relative GPS coordinates (RGPS) that can be used in scheduling and reservation-based MAC protocols with delay guarantees. This approach extends the topology-independent transmission scheduling scheme introduced by Chlamtac and Farago [1]. Furthermore, we present the design and implementation of a hardware accelerator for evaluating the polynomials in the algorithm faster than a software-only solution.

Section II summarizes prior approaches to neighbor discovery. Section III describes our new scheme in detail, and Section IV presents the hardware accelerator. Section V concludes the paper.

II. Related Work

Many MAC protocols that have been proposed are based on transmission-scheduling or reservation schemes, and operate by dividing the channel into transmission frames. Traditionally, the number of slots in the frames is correlated with the number of nodes in the network or with the network density.

Topology-dependent transmission scheduling protocols attempt to establish transmission schedules taking into account the connectivity of the network and in some cases the traffic at each node. The assignment of time slots to nodes is based either on the election of entities (nodes or links) competing for the data time slots, or the selection of reservation requests for data time slots according to a set of predefined rules. Some schemes require an initial topology-independent schedule, followed by scheduling based on negotiation among network nodes (e.g.,[2], [3], [4], [5]). There are many examples of topology-dependent schemes based on the election of transmission schedules in a distributed manner. To elect transmission schedules, each node knows the identities of all other nodes one and two hops away from itself, and the present time in the network (e.g., [6], [7], [8]). Nodes use a contention-based approach during the control section of a frame to communicate with their neighbors, either based on the identifiers of their own neighbors and themselves, or based on the identifiers of the links to their own neighbors. Each node builds and maintains a list of contending entities and uses this list to determine which node should be given access to the channel during each time slot of the data section of the frame by applying a permutation function and selecting a winning node. All these MAC protocols use neighbor discovery and collision detection schemes that are either contention-based or probabilistic.

Neighbor discovery and schedule collision detection is a non-trivial and critical task, and algorithms exist that deal solely with these problems in wireless ad-hoc networks. Algorithms such as the “birthday protocol” [9] and slotted random transmission and reception [10] enable neighbor discovery by requiring each node to be randomly in a “transmitting” or “listening” state for a particular period of time so that every node gets a chance to hear every neighbor. However, such random transmissions do not guarantee any time threshold at which complete neighbor discovery is achieved.

Topology-transparent scheduling scheme based on polynomials over a Galois field (TSMA) has been proposed by Chlamtac and Farago [1]. Even though this protocol achieves collision-free topology-transparent scheduling, it does not offer scalability into larger networks. Kunz and Rentel [11] have shown that this approach has poor performance similar to that of slotted ALOHA and therefore it is unsuitable as a MAC protocol. However, we use the topology transparent approach for neighbor discovery with delay guarantees, accommodating greater scalability for very large networks by using RGPS as described in the following section.

III. Neighbor Discovery Scheme

In this section, we first present the Galois field (GF) based algorithm as described in [1], followed by the description of our modifications based on RGPS, yielding greater scalability and smaller frames. The original proposal achieves a transmission frame size smaller than the network size N, using the properties of GF polynomials. Even though the transmission frame size offered by the original proposal is smaller than N, it is still computed as a function of N, and as a result, it is unscalable for larger networks. Our proposal based on RGPS allows frame sizes to be independent of N. For simplicity, in this paper we assume that the location information is always available for each node, either through the GPS or by the use of an accelerometer when GPS connection is lost.

A. The Algorithm

The network is viewed as an undirected graph G(V,E) with N nodes, where V is the set of nodes (vertices) and E is the set of links (edges). D_{max} denotes maximum degree of the network. $GF(q)$ is defined to be a Galois field of order q, where $q = p^m$, p is a prime number and $m \geq 1 \in Z^+$ is an arbitrary positive integer. Every element in $GF(q)$ is indexed with the integers $0, 1, ...q-1$. The values of q and k are determined such that the inequalities $q \geq kD_{max} + 1$ and $q^{k+1} \geq N$ hold. Additionally, $q^2 < N$ is required to hold such that the frame size is smaller than N. q value is chosen from a list of prime powers as the smallest value that satisfies the inequalities (Algorithm 1). The transmission frame is composed of q transmission subframes, each with q slots, thus q^2 slots in total. Each node in the network is assigned a distinct set A_n of $k+1$ coefficients, where n denotes the node id. $GF(q)$ denotes q polynomials, one for each subframe, in the form of $S_i = An_k i^k + An_{k-1} i^{k-1} + ... + An_0 i^0$, where i indexes $0, 1, ..., (q-1)$th subframes (Algorithm 2). Evaluation of the polynomial i yields the exact slot value a node should transmit, in the ith subframe. This algorithm assumes D value is not violated at any point of the execution time, and it dictates that the set A_n of $k+1$ coefficients is distinct for each node. [1]

Algorithm 1 Algorithm to set the values of q and k

```
1: Let $q\_list$ denote the sequence of increasing powers of primes (i.e. $2, 3, 4, 5, 7, 8...$).
2: i=0; k=0;
3: repeat
4:     $i = i + 1$
5:     $q = q\_list_i$;
6:     $k = \lfloor \frac{(q-1)}{D_{max}} \rfloor$
7: until $k \geq 1$ and $q^{k+1} \geq N$
```

Even though the transmission frame size is not a direct function of N, q and k are assigned values that satisfy the inequalities involving N. Our proposal extends the original by removing N from the aforementioned inequalities, replacing it with N', the maximum two-hop degree of the network. In Figure 1, we present a network of randomly and uniformly distributed nodes. We divide the network into groups where each group is of size $2d_t \times 2d_t$, where d_t denotes transmission distance, such that for any two nodes located in the same relative position in different groups, their transmissions do not collide at a receiver node. For example, the concurrent transmissions of nodes in A, B, C and D do not collide with each other. A finite number of relative positions is assumed within the group. It is also assumed that two nodes do not share the same relative position within the same group. For a given node n, let Xn and Yn be the latitude and longitude coordinates obtained via GPS. Each node in the network calculates its relative GPS coordinates using modulo operation, such that $Xn_{relative} = Xn \mod 2d_t$ and $Yn_{relative} = Yn \mod 2d_t$. In this way, we are able to remove N from the inequalities, such that the new requirements are $q \geq kD_{max} + 1$, where k needs to satisfy only the inequality $q^{k+1} > q^2$, or simply $k > 1$. Assigning $k = 2$, the inequality becomes $q \geq 2 \times D_{max} + 1$. This approach yields smaller q. We assign the actual GPS coordinates to the coefficients in A_n set, in order to assert the distinctiveness of these values.

Algorithm 2 Algorithm to compute the transmission slot

```
1: Let $An$ denote the array of $k+1$ coefficients, distinct for every node.
2: Assume that $q$ and $k$ values have been computed using Algorithm 1.
3: Let $S$ denote a $q$ sized array containing the slot values computed in this function.
4: for $(i = 0 : q-1)$ do
5:     $S_i = An_k i^k + An_{k-1} i^{k-1} + ... + An_0 i^0$
6: end for
```

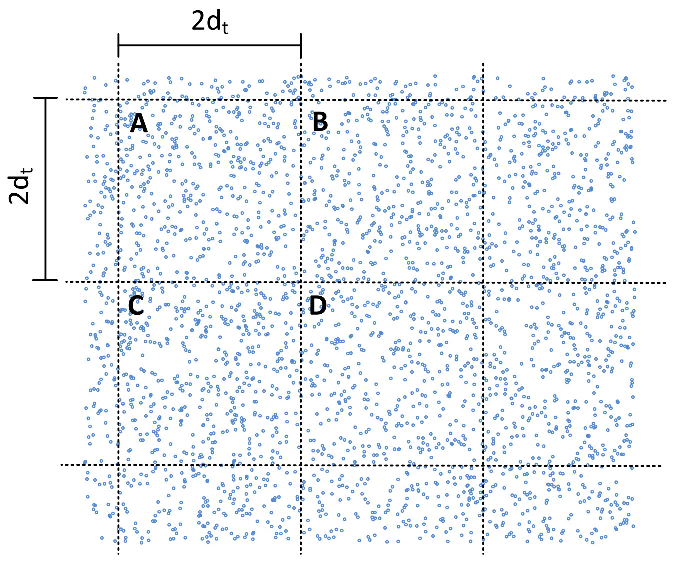

Fig. 1: Modulo $2d_t$ node groups

B. Neighbor Discovery Time

As explained in the previous subsection, the transmission frame, and thus the neighbor discovery time, consists of q^2

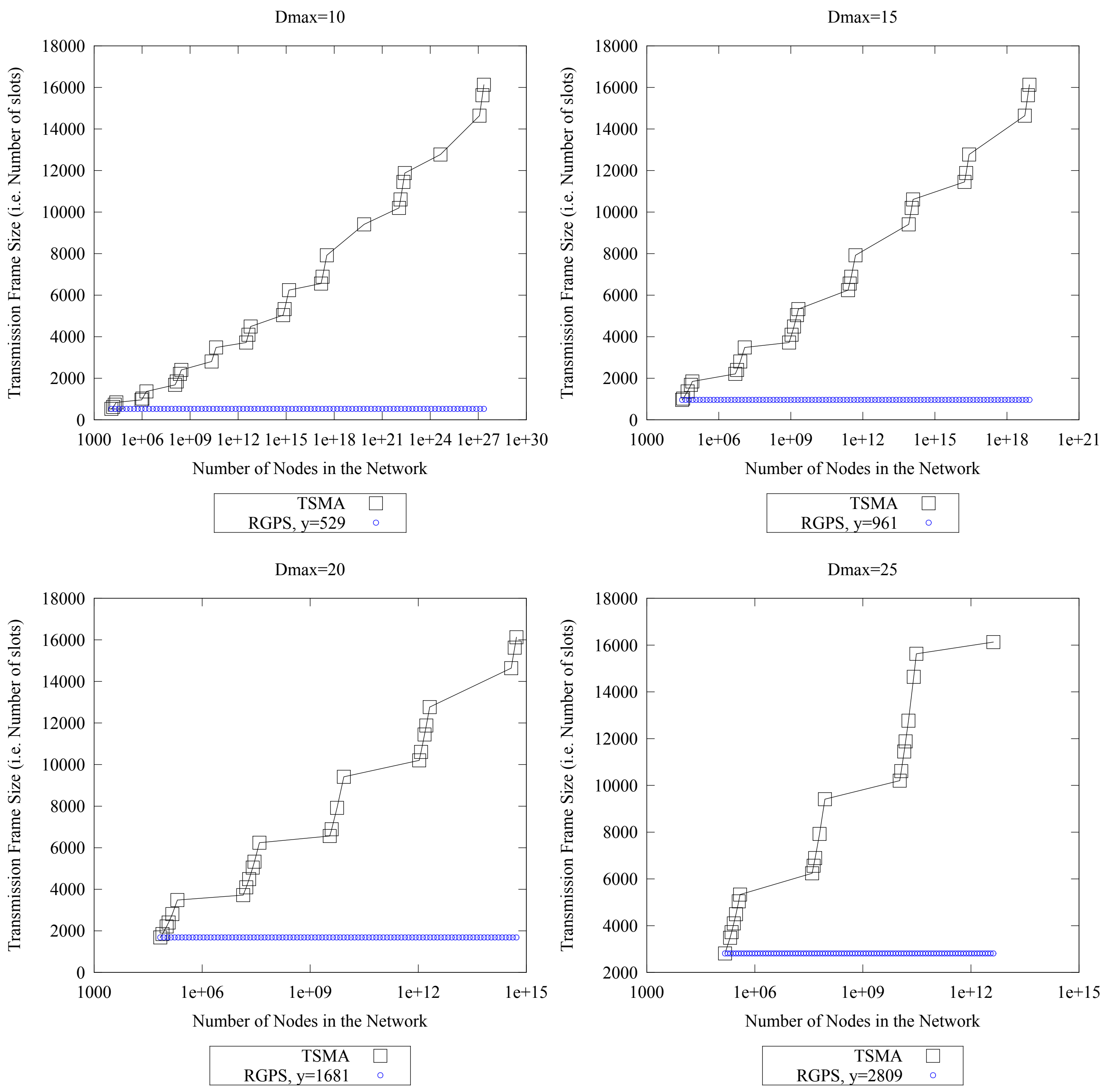

Fig. 2: Neighbor Discovery Time for D_{max} = 10, 15, 20, & 25

slots, where q is a function of D_{max} and N in the original proposal, whereas it is a function of D_{max} only in ours. Figure 2 presents transmission frame size vs. network size N, for various values of D_{max}. These plots show how q exponentially increases as network size is increased in the original our proposal (please note that x axes are logarithmic), whereas it does not increase w.r.t. N in our proposal. q, in our case, is the smallest prime power that satisfies the inequality $q \geq 2 \times D_{max} + 1$.

Table I presents various frame sizes and the corresponding time durations in milliseconds, for various 802.11 standard phy rates, taking into account the packet size, MAC and Phy headers, and the training sequences. It can be seen that the neighbor discovery time is in the order of milliseconds allowing scalability into very large network sizes.

IV. The Hardware Accelerator

The neighbor discovery scheme we describe relies on the Galois field arithmetic, involving evaluation of polynomials, which requires $2 \times q \times (k+1)$ multiplications (including exponentiations), and $q \times k$ summation operations, where q and k are the parameters in Algorithm 1. As these values increase,

TABLE I: Neighbor Discovery Time (in ms)

Frame Size (in slots)	802.11b	802.11a	802.11n	802.11ac
2k	156	62	42	40
4k	312	124	84	80
6k	468	186	126	120
8k	624	248	168	160
10k	780	310	210	200
12k	936	372	252	240
14k	1092	434	294	280
16k	1248	496	336	320

the computational time becomes longer. Moreover, our RGPS based approach requires frequent evaluation of GF polynomials with new coefficients. As a result, we propose the following system and hardware accelerator, instead of a software only solution.

A. General System Architecture

Figure 3 presents the block diagram for a general system including a radio, the physical (PHY) and link-layer components as well as a general purpose CPU, running an operating system and applications. The analog components of the system are the antenna and the modem. The baseband component filters the bit stream over baseband channel, as well as carrying out the clock recovery. MAC controller receives a control signal from the baseband after a packet has been written to the queueing memory. If the packet is a control packet directed to the MAC controller, it is read by the controller; in case it is a data packet, the controller issues an interrupt signal to the CPU, which then reads the packet from the queueing memory. The software layers including the operating system kernel, drivers and user space applications are then traversed by the data packet.

For this paper, the component of interest is the MAC controller, which can be implemented as a software-only, hardware-only, or hardware-software co-design solution. We choose to design the MAC component as a microcontroller running the algorithm in software, in conjunction with a hardware accelerator, dedicated to evaluate the polynomial expressions. Similar approaches based on hardware acceleration for communication algorithms have been proposed in [12], [13] and [14] among others.

B. Profiling of the Algorithm

Software profiling is a methodology used for analysing a program's various behaviours, including the percentage of computational time shared by different portions of the algorithm. We profile the algorithm implemented in software by using the gprof tool and observe that nearly 95% of the computational time is occupied by the evaluation of the polynomial expressions for computing the S values. This dictates the necessity to propose a hardware accelerator for our algorithm.

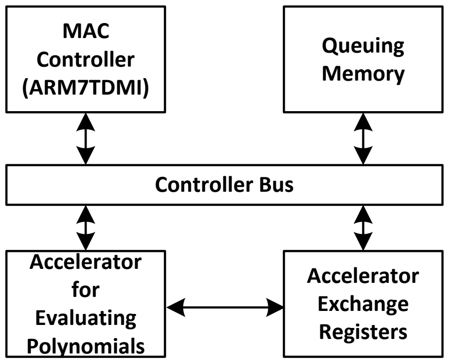
MAC Controller (ARM7TDMI)
Queuing Memory
Controller Bus
Accelerator for Evaluating Polynomials
Accelerator Exchange Registers

Fig. 4: MAC Controller & Accelerator Interface

C. MAC Controller & Accelerator Interface

In Figure 4, we present how the MAC microcontroller (µc) is interfaced to the accelerator. The µc runs the algorithm implemented in C++, except the computationally intensive segment, a double nested for loop to compute $S[i]$ values, is replaced with a memory write operation into the accelerator exchange registers (AER). AER memory is mapped to the global memory space, thus the MAC µc can write to the registers using a regular memory write operation.

According to the algorithm, $S[i] = A_k i^k + A_{k-1} i^{k-1} + ... + A_0 i^0$, where $i = q-1, q-2, ..., 0$ indexing the subframes, and the set $A = A_k, A_{k-1}, ..., A_0$ denoting the distinctive set of coefficients assigned for each and every node in the network. Our algorithm based on RGPS assigns the coefficients based on the GPS coordinate values, global node IDs, timestamps, among other unique values. The software segment to be replaced by the accelerator's operation corresponds to the code segment in Listing 1.

Listing 1: Double-Nested Loop to Compute S_i

```
int qtemp;
for (int i=Q-1;i>=0;i--)
{
        qtemp = 1;
        for (int j=0;j<K+1;j++)
        {
                S[i]+= qtemp * A[j];
                qtemp *= i;
        }
}
```

As a result the accelerator would require the input argument values Q, K and the coefficient set A. The µc would write these values to the AER and then set the $start$ register to commence the hardware acceleration. Once the accelerator completes the operation it will write the $S[i]$ values into AER, interrupt the µc, and the software would proceed to the next step of the algorithm after retrieving the values in AER into array S.

D. Accelerator Design

As we explained in the previous subsections, we chose the most computationally-intensive parts of the algorithm,

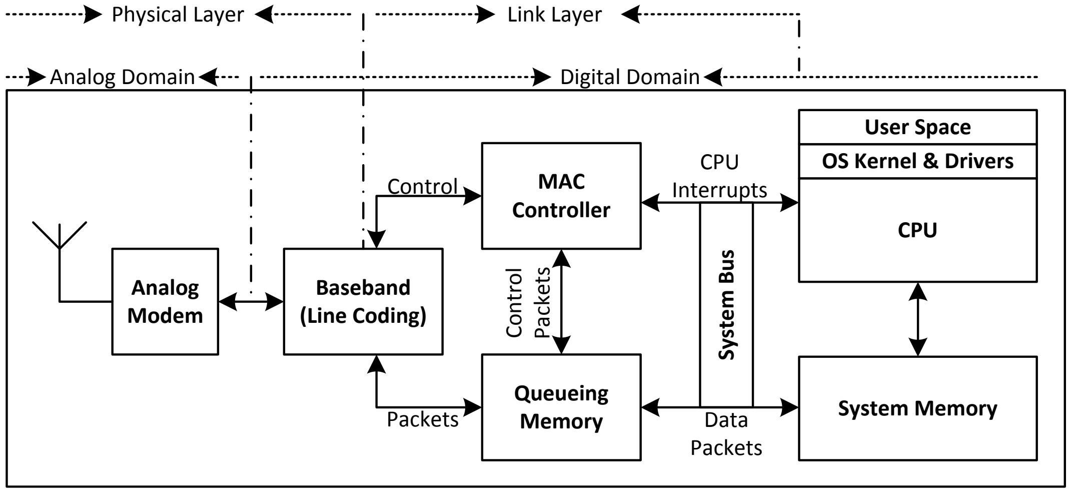

Fig. 3: General System Architecture, Block Diagram

and proposed to design it in the form of an accelerator and implement it in register-transfer level (RTL) using Verilog.

The accelerator is a very basic, 3 pipe-stage design, with 2 multipliers and 1 adder, as shown in Figure 5. Its HW description in Verilog is provided in Listing 2.

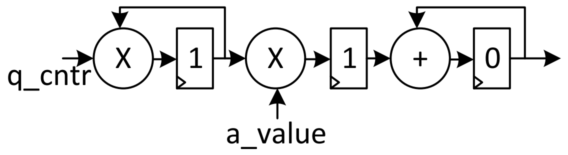

Fig. 5: Accelerator, Block Diagram

Listing 2: Verilog Description for Computing S_i

```
always @(posedge clock or posedge reset)
if (reset) begin
        exp_v <= 1;
        mult_v <= 1;
        sum_r <= 0;
end
else if (start) begin
        exp_v <= 1;
        mult_v <= a_value * exp_v;
        sum_r <= sum_w;
end
else if (start_d) begin
        exp_v <= exp_v * q_cntr;
        mult_v <= a_value;
        sum_r <= sum_w;
end
else if (start_2d) begin
        exp_v <= exp_v * q_cntr;
        mult_v <= a_value * exp_v;
        sum_r <= mult_v;
end
else begin
        exp_v <= exp_v * q_cntr;
        mult_v <= a_value * exp_v;
        sum_r <= sum_w;
end
```

a_value is a pointer to the first element of an array of registers, holding $A_k, A_{k-1}, ..., A_0$ values, which are shifted every cycle in a circular fashion, thus pointing to the same value every $k+1$ cycles. q_cntr register is loaded with $q-1$ in the first cycle of the execution, and it is decremented after every $k+1$ cycles. The execution of the algorithm for given q, k, and A set is initialized with start signal. $start\_d$ and $start\_2d$ are 1 and 2 cycles delayed copies of $start$ signal, and three of these signals are used to differentiate the pipe-stage states. After start signal is set, it takes $k+1$ cycles to complete computing $S[i]$, for a single value of i, and it takes $q \times (k+1)$ for all $i = q-1, q-2, ..., 0$.

E. RTL Synthesis & Simulation

We carry out synthesis in Xilinx ISE XST tool for Virtex2-XC2V500-FG256, which yields a 160.051MHz minimum clock period, where the critical path is through $a\_value \rightarrow Multiplier \rightarrow mult\_v$.

For the RTL simulations, we use a functional model of ARM7TDMI for the µc, which is then interfaced with the accelerator as described in Subsection IV-C and Figure 4. The wrapper module for the µc, the bus, AER, and our accelerator are all synthesizable components.

The overhead of the communication in between the µc and accelerator is $k+4$ write operations and q read operations. The write operations include $k+1$ elements of the set A, k, q, as well as the write operation to set the start flag initializing the accelerator execution. The read operations are for retrieving the set S of q elements.

Despite the communication overhead we are able to show speedup in our simulations. The results are presented in

Table II and Figure 6. Both the CPU-only solution and its accelerated version grow exponentially as $(q+k)$ increases, however the latter yield much smaller execution time.

TABLE II: Comparison of Number of Cycles for Software-Only vs. with Accelerator

q	k	accelerator communication time	accelerator computational time	accelerator total time	CPU only total time
10	5	19	60	112	459
20	10	34	220	285	1555
30	15	49	480	561	3371
40	20	64	840	962	6047
50	25	79	1300	1421	9464
60	30	94	1860	1985	13832
70	35	109	2520	2679	18961
80	40	124	3280	3444	24823
90	45	139	4140	4299	31528
100	50	154	5100	5284	39255

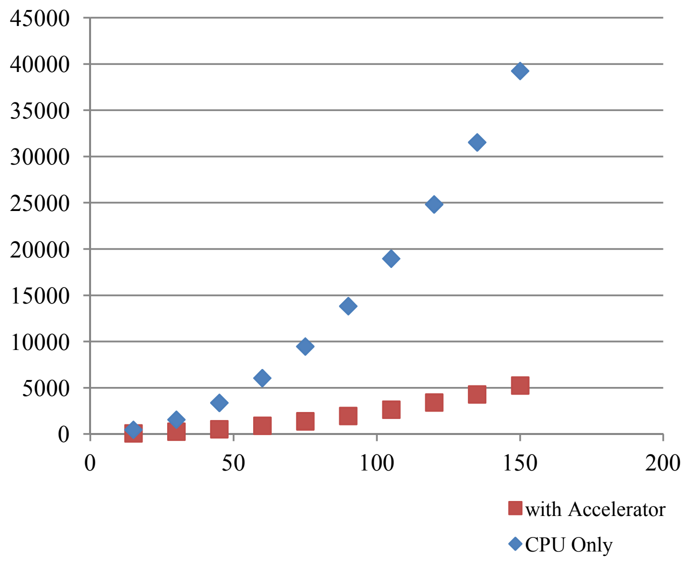

Fig. 6: Number of Cycles vs. (q+k)

V. Conclusion

In this paper, we introduced a new neighbor discovery scheme based on relative GPS coordinates (RGPS) extending the topology transparent scheduling scheme proposed in the past by Chlamtac and Farago [1]. An advantage of our approach over traditional neighbor discovery protocols is the deterministic delay guarantees for new nodes joining the network. Moreover, our approach brings in smaller transmission frame sizes and greater scalability compared to the original proposal, by detaching the network size from the inequalities the original algorithm is based on.

Additionally, we presented a hardware-software co-design implementation of our algorithm, a µc interfaced to a simple accelerator to carry out the functionality of a MAC controller. HW/SW co-design greatly improves the execution time of our algorithm, as opposed to a software-only solution. Through our RTL simulation/performance results, we are able to show that our design is suitable for quick network convergence and deployment in real ad-hoc networks.

References

[1] I. Chlamtac and A. Faragó, "Making transmission schedules immune to topology changes in multi-hop packet radio networks," *IEEE/ACM Trans. Netw.*, vol. 2, p. 23–29, Feb. 1994.

[2] I. Chlamtac and A. Lerner, "Fair algorithms for maximal link activation in multihop radio networks," *IEEE Transactions on Communications*, vol. 35, no. 7, pp. 739–746, 1987.

[3] A. Ephremides and T. Truong, "Scheduling broadcasts in multihop radio networks," *IEEE Transactions on Communications*, vol. 38, no. 4, pp. 456–460, 1990.

[4] C. Young, "USAP: a unifying dynamic distributed multichannel TDMA slot assignment protocol," in *MILCOM '96, Conference Proceedings, IEEE Military Communications Conference, 1996*, vol. 1, pp. 235–239 vol.1, 1996.

[5] J. Garcia-Luna-Aceves and A. Masilamani, "NOMAD: deterministic collision-free channel access with channel reuse in wireless networks," in *2011 8th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*, pp. 1–9, 2011.

[6] L. Bao and J. J. Garcia-Luna-Aceves, "A new approach to channel access scheduling for ad hoc networks," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, MobiCom '01, (New York, NY, USA), p. 210221, ACM, 2001.

[7] L. Bao and J. J. Garcia-Luna-Aceves, "Hybrid channel access scheduling in ad hoc networks," in *10th IEEE International Conference on Network Protocols, 2002. Proceedings*, pp. 46–57, 2002.

[8] A. Masilamani and J. Garcia-Luna-Aceves, "VCMA: efficient channel access in wireless mesh networks using virtual coordinates," in *MILITARY COMMUNICATIONS CONFERENCE, 2012 - MILCOM 2012*, pp. 1–6, 2012.

[9] M. J. McGlynn and S. A. Borbash, "Birthday protocols for low energy deployment and flexible neighbor discovery in ad hoc wireless networks," in *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking &amp; computing*, MobiHoc '01, (New York, NY, USA), p. 137145, ACM, 2001.

[10] S. A. Borbash, A. Ephremides, and M. J. McGlynn, "An asynchronous neighbor discovery algorithm for wireless sensor networks," *Ad Hoc Netw.*, vol. 5, p. 9981016, Sept. 2007.

[11] C. H. Rentel and T. Kunz, "On the average throughput performance of code-based scheduling protocols for wireless ad hoc networks," in *6th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, (Urbana-Champaign, IL), May 2005.

[12] J. Hill and D. Culler, "Mica: a wireless platform for deeply embedded networks," *IEEE Micro*, vol. 22, no. 6, pp. 12–24, 2002.

[13] Y. Cai, Y. Wu, H. Li, F. Liang, and Z. Zhou, "Flexible platform design of IEEE 802.15.3a MAC over UWB with optimized protocol accelerator," in *ASIC, 2005. ASICON 2005. 6th International Conference On*, vol. 1, pp. 181–184, 2005.

[14] G. Panic, D. Dietterle, Z. Stamenkovic, and K. Tittelbach-Helmrich, "A system-on-chip implementation of the IEEE 802.11a MAC layer," in *Euromicro Symposium on Digital System Design, 2003. Proceedings*, pp. 319–324, 2003.