افزایش سرعت شناسایی در سیستم‌های RFID

عسگر مختاری[1] **مهرگان مهدوی**[2]

۱- دانش آموخته کارشناس ارشد- گروه مهندسی برق، دانشکده فنی، دانشگاه گیلان

a.mokhtari65@gmail.com

۲- استادیار- گروه مهندسی کامپیوتر، دانشکده فنی، دانشگاه گیلان

mahdavi@guilan.ac.ir

چکیده: **سیستم‌های RFID (Radio Frequency Identification) نسل جدیدی از سسیستم‌های شناسایی اتوماتیک مبتنی بر تکنولوژی مخابرات بی‌سیم هستند. در این سیستم‌ها همه تگ‌ها (Tags) از یک کانال برای برقراری ارتباط با دستگاه خواننده استفاده می‌کنند. اگر چندین تگ به طور همزمان اقدام به ارسال شماره شناسه خود کنند سیگنال ارسالی آن‌ها با هم برخورد خواهد کرد. نحوه بر طرف کردن این برخورد در سیستم‌های RFID مخصوصاً در سیستم‌های با تعداد تگ زیاد تاثیر مستقیم بر سرعت شناسایی این سیستم‌ها دارد. در این مقاله الگوریتم جدیدی با نام EBCDA (Enhanced Bit Collision Detection Algorithm) را بر پایه الگوریتم درختی و تشخیص بیت برخورد کرده در شماره شناسه ارسالی تگ‌ها، پیشنهاد خواهیم کرد. الگوریتم پیشنهادی موجب کاهش حجم اطلاعات تبادل شده بین تگ‌ها و دستگاه خواننده و در نتیجه افزایش سرعت شناسایی خواهد شد. با مقایسه الگوریتم پیشنهادی با سایر روش‌های موجود نشان خواهیم داد الگوریتم پیشنهادی دارای بهترین عملکرد است.**

کلمات کلیدی: **سیستم‌های RFID، مقابله با برخورد تگ، الگوریتم درختی، شناسایی تگ.**

تاریخ ارسال مقاله : ۱۳۹۲/۰۹/۱۰

تاریخ پذیرش مشروط : ۱۳۹۴/۱۰/۲۵

تاریخ پذیرش مقاله: ۱۳۹۴/۱۱/۱۴

نام نویسنده‌ی مسئول: دکتر مهرگان مهدوی

نشانی نویسنده‌ی مسئول: ایران- رشت - کیلومتر ۵ جاده تهران- دانشگاه گیلان - دانشکده فنی

[DOR: 20.1001.1.26765810.1395.13.4.3.3]

Journal of Iranian Association of Electrical and Electronics Engineers - Vol.13- No.4-Winter 2016

۱- مقدمه

سیستم‌های RFID برای شناسایی اتوماتیک مورد استفاده قرار می‌گیرند و هر روزه کاربردهای جدیدی در دنیای امروز برای این سیستم‌ها مطرح می‌شود. از متداول‌ترین کاربردهای این سیستم‌ها می‌توان به استفاده از آن‌ها در بلیط الکترونیکی، فروشگاه‌ها، انبارها و بندرگاه‌ها تجاری اشاره کرد. در سیستم‌های RFID از میکروچیپ‌هایی با نام تگ برای نگهداری و ارسال شماره شناسه به دستگاه خواننده استفاده می‌شود [۷]. شماره شناسه تگ‌های قابل شناسایی توسط هر سیستم در پایگاه داده سیستم ذخیره شده است. دستگاه خواننده بعد از دریافت هر شماره شناسه و تطبیق داشتن آن با یکی از شماره شناسه‌های موجود در پایگاه داده اقدام به شناسایی تگ مورد نظر خواهد کرد. تگ‌ها دارای قابلیت نصب بر روی کارت‌های الکترونیکی، کالاها، حیوانات و غیره هستند. اگر تگ در ساختار داخلی خود منبع تامین کننده توان نداشته باشد تگ پسیو خواننده می‌شود [۳-۱]. در این مقاله منظور از تگ، تگ پسیو می‌باشد. هر تگ با قرار گیری در محدوده تحت پوشش دستگاه خواننده توان مورد نیاز خود را از امواج رادیویی ارسال شده توسط دستگاه خواننده تامین کرده و اقدام به ارسال شماره شناسه خود خواهد کرد. دستگاه خواننده با دریافت شماره شناسه هر تگ، آن را شناسایی می‌کند [۸و۷].

سیستم‌های RFID جایگزین مناسبی برای سیستم‌های بارکد محسوب می‌شوند. برتری‌های سیستم‌های RFID نسبت به سیستم‌های بارکد عبارتند از [۸]:

- شناسایی همزمان چندین تگ.
- عدم نیاز به قرارگیری تگ دقیقاً در مقابل دستگاه خواننده.
- قابل استفاده بودن در مکان‌های امنیتی.

زمانی که تعداد تگ‌های قرار گرفته در ناحیه تحت پوشش سیستم بیشتر از یک تگ باشد شماره ارسالی تگ‌ها با هم برخورد کرده و دستگاه خواننده قادر به تشخیص شماره شناسه نخواهد بود. به این حالت اصطلاحاً برخورد تگ‌ها گفته می‌شود [۵و۴].

روش‌های ارائه شده برای بر طرف کردن برخورد تگ‌ها به دو گروه تقسیم می‌شوند که عبارتند از: روش‌های مبتنی بر الگوریتم ALOHA و روش‌های مبتنی بر الگوریتم درختی.اساس عملکرد روش‌های مبتنی بر الگوریتم ALOHA کاهش احتمال برخورد تگ‌ها به وسیله ارسال شماره شناسه توسط تگ در بازه‌های زمانی مختلف می‌باشد. به هر یک از این بازه‌های زمانی یک اسلات گفته می‌شود [۱۶]. به این صورت که دستگاه خواننده تعداد مشخصی بازه زمانی در اختیار تگ‌ها قرار می‌دهد هر تگ به صورت تصادفی یک بازه زمانی انتخاب کرده و شماره شناسه خود را ارسال خواهد کرد. اگر بیشتر از یک تگ یک بازه زمانی را برای ارسال شماره شناسه خود انتخاب کرده باشند برخورد رخ خواهد داد و تگ‌های شرکت کننده در این برخورد در سیکل بعد مجددا اقدام به ارسال شماره شناسه خود کرد [۸-۶].

در روش‌های مبتنی بر الگوریتم درختی تگ‌های موجود در ناحیه تحت پوشش دستگاه خواننده به گروه‌های مجزایی تقسیم‌بندی می‌شوند. این گروه‌بندی تا جای ادامه می‌یابد که در هر گروه تنها یک تگ باقی مانده باشد. بعد از آن الگوریتم حالت بازگشتی به خود گرفته و شروع به شناسایی تگ‌ها خواهد کرد. در این روش گروه‌بندی تگ‌ها با استفاده از شماره شناسه منحصر به فرد آنها صورت می‌گیرد[۹] [۱۶].

از میان الگوریتم‌های مختلف ارائه شده برای مقابله با برخورد تگ‌ها در سیستم‌های RFID اکثراً الگوریتم‌هایی که توانایی تشخیص بیت برخورد کرده در شماره شناسه ارسالی تگ‌ها را دارند از حجم اطلاعات تبادل شده کمتری برخوردار هستند [۷]. همه الگوریتم‌های مبتنی بر تشخیص بیت برخورد کرده بر مبنای الگوریتم درختی هستند. دلیل کاهش حجم اطلاعات تبادل شده در این الگوریتم‌ها تشخیص بیت برخورد و ذخیره بیت‌های تشخیص داده شده می‌باشد [۸].

لو[۱] و همکارانش [۱۰] الگوریتم QT (Query Tree) را برای مقابله با برخورد تگ‌ها معرفی کردند. در این الگوریتم دستگاه خواننده در هر اسلات رشته بیت q را به تگ‌ها ارسال می‌کند. هر تگ که دارای پیشوندی برابر با رشته بیت q است شماره شناسه خود را به دستگاه خواننده ارسال خواهد کرد. در صورت رخ دادن برخورد در هر اسلات دستگاه خواننده یک بیت به طول رشته بیت ارسالی خود خواهد افزود. بعد از شناسایی اولین تگ طی یک عملیات شناسایی مابقی تگ‌ها نیز شناخته خواهد شد. هر چند این الگوریتم عملکرد مطلوبی برای شناسایی تگ‌ها دارد، اما زمان زیادی برای شناسایی همه تگ‌ها قرار گرفته در ناحیه تحت پوشش نیاز دارد.

فینگ بو[۲] و همکارانش [۱۱] الگوریتم IDS (ID-Binary Tree Stack Anti-Collision) را پیشنهاد کردند. در این الگوریتم ارسال شماره شناسه تگ‌ها به صورت بیت به بیت انجام می‌شود. در صورت رخ دادن برخورد در بیت ارسالی تگ‌ها، تگ‌های که دارای بیت "1" در یک گروه و تگ‌های که دارای بیت "0" هستند در گروه دیگر قرار می‌گیرند. این عمل تا دریافت کامل شماره شناسه تگ ادامه می‌یابد. استفاده از حافظه stack برای ذخیره رشته بیت‌های دریافتی مزیت این الگوریتم محسوب می‌شود. زمانی که طول شماره شناسه زیاد باشد حجم اطلاعات تبادل شده در این الگوریتم به شدت افزایش می‌یابد.

هوسنگ گو[۳] و یانگ یو[۴] [۱۲] الگوریتم BHQT (Bit Collision Detection based Hybrid Query Tree Protocol) را برای افزایش سرعت شناسایی سیستم‌های RFID مطرح کردند. در این الگوریتم ابتدا تگ‌ها همه شماره شناسه خود را ارسال می‌کنند. دستگاه خواننده بیت‌های برخورد کرده را تشخیص داده و فرمان ارسال مجدد این بیت برخورد کرده را به تگ‌ها ارسال می‌کند. اگر تنها یک بیت برخورد کرده باشد دستگاه خواننده همزمان دو تگ را شناسایی خواهد کرد. در این الگوریتم بعد از هر اسلات برخورد کرده رشته بیتی به طول شماره شناسه به تگ‌ها ارسال می‌شود. هرچند در این الگوریتم حجم اطلاعات ارسال شده از سوی تگ‌ها کم است. ولی طول رشته

بیت ارسالی دستگاه خواننده زیاد است.

مینجیایکسنگ هی[5] و همکارانش [۹] روش جدیدی با نام EAA (Enhanced Anti-collision Algorithm) را ارائه دادند. در این الگوریتم در صورتی که تعداد بیت‌های برخورد کرده دو بیت باشد دستگاه خواننده، تگ‌های ارسال کننده را به دو گروه تقسیم می‌کند. در اسلات بعد تگ‌ها، بیت‌های موجود در شماره شناسه خود، بعد از محل برخورد اول را ارسال خواهند کرد. در این الگوریتم هر چه قدر فاصله بین دو بیت برخورد کرده زیاد باشد حجم اطلاعات ارسالی بیشتر می‌شود.

در این مقاله الگوریتم جدیدی با نام EBCDA بر پایه الگوریتم درختی و تشخیص بیت برخورد کرده، پیشنهاد خواهیم کرد. این الگوریتم با تشخیص بیت برخورد کرده و جلوگیری از تبادل اطلاعات غیر ضروری بین تگ و دستگاه خواننده از حجم اطلاعات تبادل شده کاسته و به سرعت شناسایی تگ‌ها خواهد افزود.

در ادامه این مقاله و در بخش دوم الگوریتم پیشنهادی خود را به صورت کامل توضیح خواهیم داد. در بخش سوم روش پیشنهادی خود را تحلیل خواهیم کرد. ارزیابی الگوریتم پیشنهادی با استفاده از نتایج شبیه‌سازی را در بخش چهارم انجام خواهیم داد. در بخش آخر نیز نتیجه‌گیری خود را بیان خواهیم کرد.

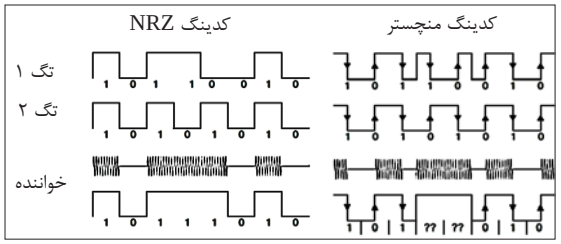
۲- روش پیشنهادی

در الگوریتم‌های درختی مقابله با برخورد، تشخیص بیت برخورد کرده از اهمیت خاصی برخوردار است از اینرو در این الگوریتم‌ها از کدینگ منچستر[6] استفاده می شود.

۲-۱- کدینگ منچستر

شکل (۱) مثالی از سیگنال ارسالی دو تگ و سیگنال دریافتی دستگاه خواننده با استفاده از کدینگ منچستر و NRZ (Non Return to Zero، عدم بازگشت به صفر) را نشان می‌دهد.

لبه پایین رونده سیگنال نشان دهنده "1" و لبه بالا رونده سیگنال نشان دهنده "0" می‌باشد [۱۳] [۱۷]. در کدینگ NRZ سطح ولتاژ نشان دهنده"1" یا "0" بودن بیت می‌باشد. رشته بیت دریافتی دستگاه خواننده در کدینگ منچستر و NRZ به ترتیب عبارتند از: $1_0 0_1 1_2 \times_3 \times_4 0_5 1_6 0_7$ و $1_0 0_1 1_2 1_3 1_4 0_5 1_6 0_7$ است. مشاهده می‌شود با وجود برخورد بیت‌های چهارم و پنجم با هم کدینگ NRZ قادر به تشخیص این برخورد نمی‌باشد. هرچند استفاده از کدینگ منچستر موجب می‌شود پهنای باند بیشتری مورد نیاز باشد ولی با استفاده از این کدینگ از ارسال مجدد بیت‌هایی که برخورد نکرده‌اند جلوگیری می‌شود[۱] [۱۷]. به همین جهت در اکثر مقالات و کتب معتبر در زمینه مقابله با برخورد در سیستم‌های RFID از این نوع کدینگ استفاده می‌شود [۹] [۱۲].



شکل (۱): کدینگ منچستر و NRZ

۲-۲- الگوریتم EBCDA

در الگوریتم پیشنهادی دستگاه خواننده تا حد ممکن اقدام به بر طرف کردن برخورد کرده و از تبادل اطلاعات با تگ جلوگیری می‌کند؛ در این راستا نحوه عملکرد دستگاه خواننده در دریافت انواع بیت‌های مختلف متفاوت خواهد بود.

به منظور ذخیره اطلاعات تشخیص داده در اسلات‌های شکست خورده و جلوگیری از تبادل اطلاعات اضافی دستگاه خواننده نیاز دارد بخش‌های از حافظه خود را به این منظور اختصاص دهد. حافظه‌های مورد نیاز دستگاه خواننده برای اجرای الگوریتم پیشنهادی عبارتند از:

- Stack: برای ذخیره اطلاعات تشخیص داده شده در اسلات‌های شکست خورده.
- Tstring: برای نگهداری بخش ازشماره شناسه که قبلاً توسط تگ‌ها ارسال شده است.
- Pointer: برای ذخیره کردن محل‌های برخورد.
- Pstring: برای نشان دادن محل بیت از شماره شناسه که بیت‌های قبل از آن توسط دستگاه خواننده تشخیص داده شده است.

علاوه بر موارد فوق دستگاه خواننده به حافظه C برای نگهداری موقت محل بیت برخورد کرده و شمارنده‌های Pc و Rc برای تعیین وضعیت الگوریتم نیاز دارد (اگر اعداد موجود در Rc و Pc برابر باشند به معنای پایان عملیات شناسایی است). Rc تعداد اسلات‌های شکست خورده و Pc تعداد اسلات‌ها موفق را نشان می‌دهند. تگ‌های مورد استفاده در الگوریتم پیشنهادی نیز به یک شمارنده Tc برای تعیین موقعیت خود در ساختار درختی نیاز دارند. زمانی که عدد موجود در Tc برابر صفر باشد تگ شماره شناسه خود را ارسال خواهد کرد. مقادیر اولیه شمارنده‌های مورد استفاده عبارتند از: Rc=1، Pc=0 و Tc=0.

اگر دو تگ که شماره‌شناسه آن‌ها تنها در یک بیت اختلاف دارند اقدام به ارسال شماره شناسه خود کنند الگوریتم پیشنهادی همانند الگوریتم‌های BHQT و EAA قادر است همزمان این دو تگ را شناسایی کند.

در الگوریتم پیشنهادی دستگاه خواننده بعد از دریافت بیت موفق (برخورد نکرده) به دریافت اطلاعات ادامه می‌دهد. در مواجه با بیت‌های برخورد کرده با توجه به نوع بیت برخورد کرده یکی از دو عملیات زیر را اجرا می کند.

عملیات شماره۱: جایگذاری بیت برخورد کرده با "0"، ذخیره محل

[DOR: 20.1001.1.26765810.1395.13.4.3.3]
Journal of Iranian Association of Electrical and Electronics Engineers - Vol.13- No.4-Winter 2016

بیت برخورد کرده در حافظه C و ادامه دریافت اطلاعات.

عملیات شماره۲: موقف کردن دریافت اطلاعات، الحاق رشته بیت دریافتی تا قبل از بیت برخورد قبلی با "1" و ذخیره آن در حافظه Stack، ذخیره عدد موجود در C در حافظه Pointer و ارسال فیدبک به تگ‌ها (عملگر الحاق را با || نشان خواهیم داد).

انواع بیت‌های که دستگاه خواننده در مواجه با آنها عملیات شماره۱ را اجرا می‌کند عبارتند از:

- FBC (First Bit Collision): اولین بیت برخورد کرده در ابتدای عملیات شناسایی و یا بعد از شناسایی هر تگ.
- SBC (Second Bit Collision): بیتی که برای بار دو برخورد کرده باشد.
- BCP (a Bit Collision that previous bit collision of that did not replace with "0"): بیتی برخوردی که بیت برخورد کرده قبل از آن با "0" جایگذاری نشده باشد (وجود این بیت به معنای وجود حداقل دو تگ متناسب با بیت‌های تشخیص داده شده می‌باشد).

اگر بیت برخورد کرده جزوء موارد فوق نباشد دستگاه خواننده در مواجه با آن عملیات شماره۲ را اجرا خواهد کرد. این بیت را به اختصار BC (Bit Collision) می‌نامیم.

در حالت کلی دو نوع اسلات در الگوریتم پیشنهادی وجود خواهد داشت اسلات موفق و اسلات شکست خورده. در اسلات موفق یک یا دو تگ شناسایی می‌شود. اما در اسلات شکست خورده هیچ تگی شناسایی نمی‌شود. دستگاه خواننده بعد از هر اسلات موفق رشته بیت "1" || Feedback= Pstring و بعد از هر اسلات شکست خورده رشته بیت "0" || Feedback= C به تگ‌ها ارسال می‌کند (Pstring و C هر دو رشته بیتی با طول یکسان هستند). هر تگ با دریافت Feedback وضعیت خود را تعیین می کند. شکل (۲) فلوچارت عملیات اجرایی دستگاه خواننده در الگوریتم پیشنهادی را نشان می‌دهد.

اگر دستگاه خواننده در طی عملیات شناسایی با یک بیت برخورد کرده از نوع BC مواجه شود به این معناست که اسلات مورد بررسی یک اسلات شکست خورده می‌باشد. عملیات دستگاه خواننده در مواجه با این اسلات به صورت زیر است.

- الحاق رشته بیت تشخیص داده شده تا قبل از محل نشان داده شده توسط C با "1" و ذخیره آن در Stack (در این حالت C نشان دهنده محل برخورد قبلی است).

"1" || رشته بیت تشخیص داده شده تا قبل از Push= C Into Stack

- ارسال فیدبک

Feedback= C || "0"

- ذخیره محل برخورد قبلی در حافظه Pointer.

Push= C into pointer

- ذخیره محل بیت برخورد کرده در C

محل بیت برخورد کرده = C

- افزایش شمارنده تعداد اسلات‌های شکست خورده.

Rc=Rc+1

عملکرد دستگاه خواننده در مواجه با هر کدام از انواع بیت‌های FBC، SBC و BCP به صورت زیر می‌باشد.

- ذخیره محل بیت برخوردکرده در حافظه C.

محل بیت برخورد کرده = C

- الحاق Tstring با "0"

Tstring= Tstqing || "0"

هر بیت تشخیص داده شده توسط دستگاه خواننده بعد از دریافت با رشته بیت موجود در Tstring الحاق می‌شود اگر رشته بیت حاصل نشان دهنده یک شماره شناسه باشد؛ اسلات مورد بررسی به عنوان یک اسلات موفق در نظر گرفته می‌شود. عملکرد دستگاه خواننده در این حالت به صورت زیر خواهد بود.

- شناسایی یک تگ متناسب با شماره شناسه تشخیص داده شده.
- افزایش شمارنده تعداد اسلات‌های موفق

Pc=Pc+1

- شناسایی یک تگ دیگر در صورت امکان. اگر دستگاه خواننده به جای آخرین برخورد "0" گذاشته باشد به معناست که تگ دیگری تنها با یک بیت اختلاف نسبت به شماره شناسه تشخیص داده شده در محدوده تحت پوشش سیستم وجود دارد. در این حالت دستگاه خواننده با قرار داده "1" در محل آخرین برخورد تگ دیگری را بدون ارسال و یا دریافت اطلاعات دیگری را شناسایی می‌کند.
- قرار دادن آخرین رشته بیت موجود در Stack در Tstring. با این عمل رشته بیت تشخیص داده شده تا قبل از آخرین بیت BC در Tstring قرار می‌گیرد و تگ‌های که قرار است در اسلات بعدی شناسایی شوند اقدام به ارسال مجدد این رشته بیت نخواهند کرد.

Tsring= pop (Stack)

- قرار دادن آخرین رشته بیت موجود در Pointer در Pstring. با این کار محل آخرین بیت BC در Pstring قرار می‌گیرد.

Pstring= pop (Pointer)

- ارسال سیگنال فیدبک به تگ‌ها.

Feedback= Pstring || "1"

شکل (۳) فلوچارت عملیات اجرایی تگ در الگوریتم پیشنهادی را نشان می‌دهد. وضعیت هر تگ وابسته به سیگنال فیدبک ارسالی دستگاه خواننده است. حالات ممکن برای سیگنال فیدبک و وضعیت تگ عبارتند از:

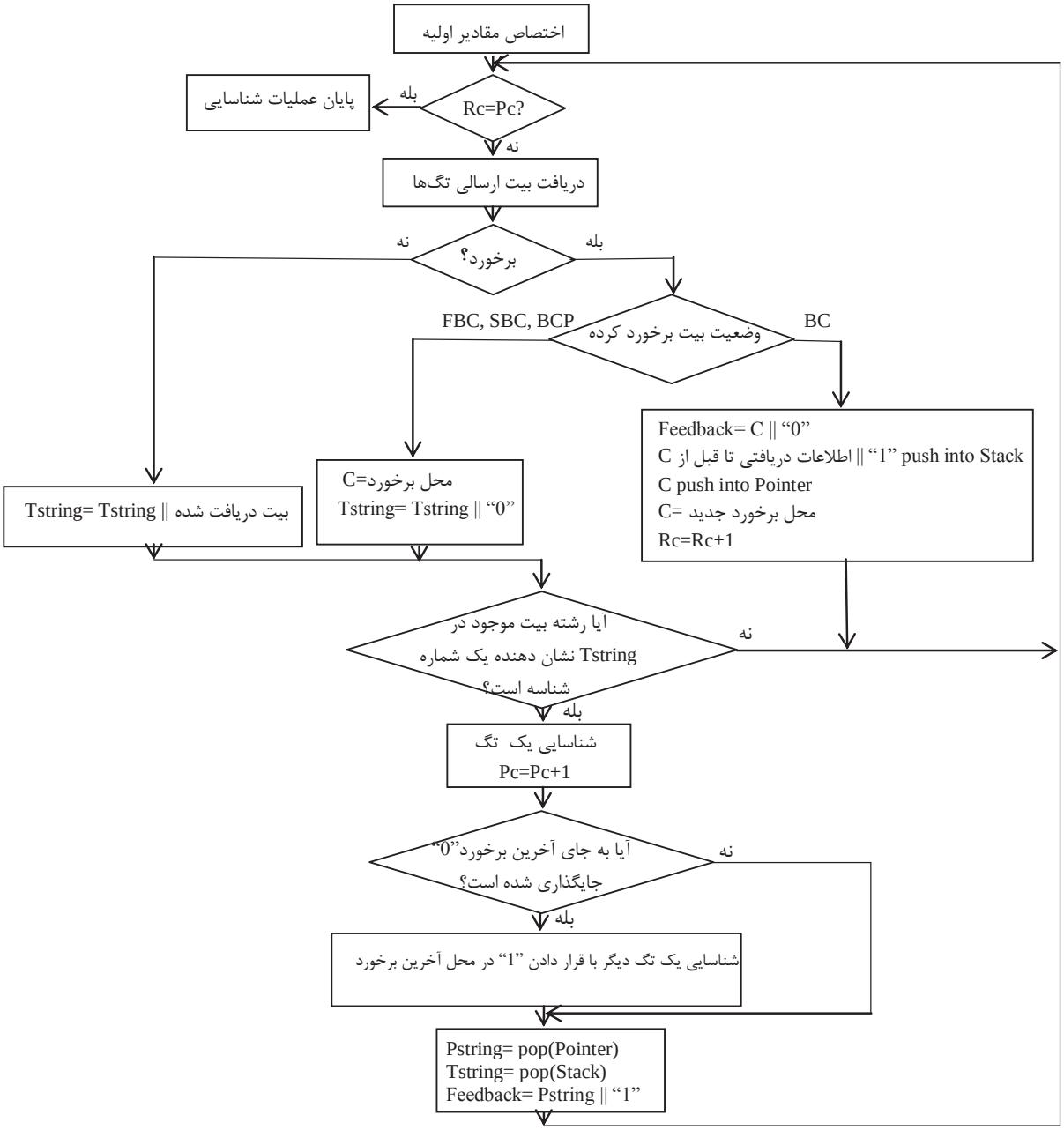
- "1" || Feedback= Pstring : این فیدبک بیان کننده اسلات موفق است که درآن یک یا دو تگ شناسایی شده است. در این حالت عدد موجود در شمارنده (Tc) تگ‌های که مورد شناسایی قرار گرفتند صفر است؛ این تگ‌ها با

[DOR: 20.1001.1.26765810.1395.13.4.3.3]

دریافت این فیدبک به حالت خاموش می‌روند. سایر تگ‌ها یک واحد از شمارنده خود می‌کاهند (Tc=Tc-1). تگ‌های که شمارنده آن‌ها صفر می‌شود شماره شناسه خود را از محل نشان داده توسط Pstring به بعد را ارسال خواهند کرد.

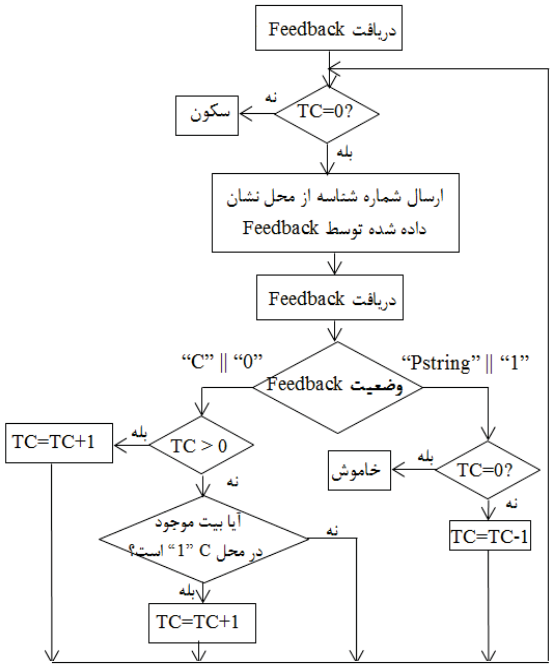
- "0" || Feedback= C : این فیدبک بیان کننده اسلات شکست خورده است. در این حالت تگ‌های که در شماره شناسه خود و در محل نشان توسط C دارای "1" می‌باشند و یا عدد موجود در شمارنده آن‌ها بزرگ‌تر از صفر است یک واحد به شمارنده خود می‌افزایند (Tc=Tc+1). با این کار این تگ‌ها به حالت سکون می‌روند و موقتاً از ارسال شماره شناسه خودداری می‌کنند.

Pstring و C هر دو نشان دهنده محلی ازشماره شناسه تگ‌ها بوده و دارای طول ثابتی هستند. بنابراین تگ در تشخیص نوع فیدبک مشکلی نخواهد داشت.



شکل (۲): فلوچارت عملیات اجرایی دستگاه خواننده در الگوریتم پیشنهادی EBCDA

[DOR: 20.1001.1.26765810.1395.13.4.3.3]
Journal of Iranian Association of Electrical and Electronics Engineers - Vol.13- No.4-Winter 2016



شکل (۳): فلوچارت عملیات اجرایی تگ در الگوریتم پیشنهادی EBCDA

جدول (۱) مثالی از مراحل شناسایی چهار تگ با استفاده از روش پیشنهادی را نشان می‌دهد. شماره شناسه تگ‌ها در این مثال عبارتند از: ID1=0000، ID2=0101، ID3=1101 و ID4=1111. از آنجایی که مقدار پیش فرض Tc همه تگ‌ها صفر می‌باشد در ابتدا هر چهار تگ اقدام به ارسال شماره شناسه خود خواهند کرد. در نتیجه در بیت دریافتی اول توسط دستگاه خواننده برخورد رخ خواهد داد. با توجه به اینکه این برخورد اولین برخورد است دستگاه خواننده به جای بیت برخورد کرده "0" گذاشته و به دریافت اطلاعات ادامه می‌دهد. دستگاه خواننده در بیت دوم دریافتی نیز برخورد را تشخیص می‌دهد. با تشخیص برخورد دوم تگ‌هایی را که در برخورد قبلی "1" داشتند را به حالت سکون می‌برد. به عبارتی عدد موجود در Tc آنها را یک واحد افزایش می‌دهد. در مرحله دوم تنها تگ‌های ۱و ۲ در حالت فعال باقی ماندن و اقدام به ارسال باقیمانده شماره شناسه (بیت‌های دوم و سوم) خود خواهند کرد. دستگاه خواننده در دریافت بیت دوم مشکلی نخواهد داشت ولی در دریافت بیت سوم با برخورد مواجه خواهد شد. در این حالت تگی که در برخورد قبلی (بیت اول) "1" دارد به حالت سکون خواهد رفت. در مرحله سوم تنها تگ ۱ بیت سوم خود را ارسال خواهد و دستگاه خواننده با موفقیت این تگ را شناسایی خواهد کرد. در مرحله چهارم تنها تگ ۲ فعال است (Tc_2=0) و باقیمانده شماره شناسه خود از شماره بیتی که توسط Pointer نشان داده می‌شود به بعد را ارسال خواهد کرد. به این ترتیب تگ شماره ۲ نیز شناسایی خواد شد. در مرحله پنجم تگ‌های ۳ و ۴ فعال هستند و شماره شناسه‌های خود را ارسال خواهند کرد. دستگاه خواننده بیت دوم برخورد را به عنوان بیت برخورد کرده تشخیص خواهد داد و چون اولین بیت برخورد بعد از شناسایی یک تگ است این بیت را با "0" را جایگذاری کرده و تگ ۳ را شناسایی خواهد کرد. بعد از پایان شناسایی تگ ۳ چون دستگاه خواننده به جای آخرین بیت برخورد "0" گذاشته است با قرار دادن "1" به جای این بیت، تگ ۴ را هم شناسایی خواهد کرد. به عبارتی دستگاه خواننده دو تگ را به صورت همزمان شناسایی خواهد کرد.

۳- تحلیل روش پیشنهادی

بر خلاف اکثر الگوریتم‌های مقابله با برخورد مبتنی بر ساختار درختی، الگوریتم EBCDA اسلات خالی ندارد. اسلات خالی زمانی ایجاد می‌شود که هیچ تگی در محدوده تحت پوشش سیستم متناسب با اسلات در حالی بررسی توسط دستگاه خواننده وجود نداشته باشد. در الگوریتم EBCDA هر اسلات متناسب با تگ‌های موجود در ناحیه تحت پوشش سیستم ایجاد می‌شود. به عبارتی یک اسلات زمانی به وجود خواهد که حتماً تگ یا تگ‌های متناسب با آن اسلات در ناحیه تحت پوشش سیستم وجود داشته باشد. اسلات خالی همواره موجب اتلاف زمان می‌شود و جلوگیری از به وجود آمدن آن باعث افزایش سرعت شناسایی خواهد شد.

جدول (۱): مثالی از الگوریتم EBCDA

وضعیت اطلاعات تشخیص داده شده	اطلاعات تشخیص داده شده	بیت‌های ارسالی تگ‌ها	Tstring	Pointer	Stack	Rc	Tc_4	Tc_3	Tc_2	Tc_1	Pc
(۱) برخورد	0×	00,01,11	-	0	1	1=>2	0=>1	0=>1	0	0	0
(۲) برخورد	000×	00,01	0	0,1	1,01	2=>3	1=>2	1 =>2	0=>1	0	0
(۳) شناسایی تگ	0000	0	000	0,1	1,01	3	2=>1	2 =>1	1=>0	0	0=>1
(۴) شناسایی تگ	0101	01	01	0	1	3	1=>0	1=>0	0	-	1=>2
(۵) شناسایی دو تگ	1101	101,111	1	-	-	3	0	0	-	-	2=>3

در الگوریتم EBCDA دستگاه خواننده در پایان هر اسلات به جز اسلاتی که در آن عدد موجود در شمارنده‌های Rc و Pc برابر می‌شوند سیگنال فیدبکی به تگ‌های موجود در محدوده تحت پوشش خود ارسال می‌کند. طول (تعداد بیت) این سیگنال از رابطه زیر به دست در می‌آید.

$$LF = \log_2^{LID} + 1 \qquad (۱)$$

در رابطه فوق LID طول شماره شناسه تگ‌ها می‌باشد. Log_2^{LID} تعداد بیت‌های لازم برای نشان دادن محل یک بیت در رشته بیتی به طول LID می‌باشد. علاوه بر این همواره یک بیت نیز که نشان دهنده نوع اسلات می‌باشد همراه سیگنال فیدبک ارسال می‌شود (“0” || Log_2^{LID} بعد از اسلات موفق، “1” || Log_2^{LID} بعد از اسلات برخورد کرده). تعداد دفعاتی که دستگاه خواننده اقدام به ارسال فیدبک می‌کند برابر است با تعداد اسلات‌ها به جز آخرین اسلات. ازاین‌رو تعداد بیت‌های که از سوی دستگاه خواننده در قالب سیگنال فیدبک به تگ‌ها ارسال می‌شود از رابطه زیر به دست خواهد آمد.

$$RT = (N-1) \times LF \qquad (۲)$$

در رابطه فوق N بیانگر تعداد اسلات مورد نیاز دستگاه خواننده برای شناسایی همه تگ‌های قرار گرفته در محدوده تحت پوشش سیستم می‌باشد. اگر تعداد برخورد تگ‌ها را با C نشان دهیم، با قرار گیری تگ-ها در محدوده تحت پوشش تعداد اسلات همواره از رابطه زیر به دست خواهد آمد.

$$N = 2C + 1 \qquad (۳)$$

بهترین حالت در الگوریتم EBCDA زمانی خواهد بود که دستگاه خواننده قادر باشد در هر اسلات به صورت همزمان دو تگ را شناسایی کند. به عبارتی همه اسلات‌ها از نوع OBCT خواهند شد. (البته زمانی که تعداد تگ‌ها فرد باشد دستگاه خواننده الزاماً حداقل یک اسلات ایجاد خواهد کرد که در آن تنها یک تگ را شناسایی کند). در این حالت تعداد برخوردها از رابطه زیر به دست خواهد آمد.

$$C_{\min} = round(\frac{n}{2} - 1) \qquad (۴)$$

در رابطه فوق n تعداد تگ‌های قرار گرفته در ناحیه تحت پوشش می‌باشد. بدترین حالت در الگوریتم EBCDA زمانی خواهد بود که در طی عملیات شناسایی هیچ اسلاتی وجود نداشته باشد که دستگاه خواننده موفق شود در آن همزمان دو تگ را شناسایی کند. در این حالت تعداد برخوردها از رابطه زیر به دست خواهد آمد.

$$C_{\max} = n - 1 \qquad (۵)$$

با جایگذاری روابط (۴) و (۵) در رابطه (۳) حداقل و حداکثر تعداد اسلات مصرفی دستگاه خواننده از روابط زیر به دست خواهد آمد.

$$\begin{aligned} N_{\min} &= 2C_{\min} + 1 \\ N_{\max} &= 2C_{\max} + 1 \end{aligned} \qquad (۶)$$

با توجه به مطالب گفته شده تعداد بیت‌های ارسالی دستگاه خواننده به تگ‌ها در بازه نشان داده شده مطابق رابطه (۷) خواهد بود

$$N_{\min} \times LF \le RT \le N_{\max} \times LF \qquad (۷)$$

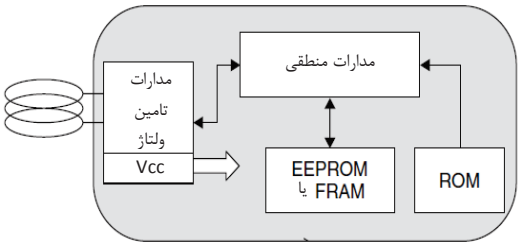
تعداد بیت‌های ارسالی هر تگ در هر اسلات وابسته به شماره شناسه تگ‌های قرار گرفته در ناحیه تحت پوشش سیستم و موقعیت بیت برخورد کرده می‌باشد.

۴- نتایج شبیه‌سازی

تگ‌هایی که فقط یک بار توسط کارخانه سازنده برنامه ریزی می‌شوند و شماره شناسه آنها توسط دستگاه خواننده، خوانده می‌شود دارای ساختاری مشابه شکل (۴) می‌باشند [1]. الگوریتم‌های مربوط به مقابله با برخورد تگ‌ها در ماژول مدارات منطقی توسط کارخانه سازنده طراحی و پیاده‌سازی می‌شوند. در این مقاله تنها پروسه شناسایی تگ‌ها توسط دستگاه خواننده مورد بحث می‌باشد، از اینرو جهت ارزیابی الگوریتم پیشنهادی این پروسه توسط نرم افزار Matlab شبیه‌سازی و مورد بررسی قرار گرفته است. در شبیه‌سازی‌های انجام گرفته در این مقاله الگوریتم پیشنهادی با الگوریتم‌های QT، IDS، BHQT و EAA مورد مقایسه قرار گرفته شده است. طول شماره شناسه تگ‌ها در همه شبیه سازی‌های این مقاله ۶۴ بیت می‌باشد.

با توجه اینکه تعداد بیت‌های تبادل شده بین تگ و دستگاه خواننده در هر اسلات متفاوت می‌باشد تعداد اسلات مصرفی معیار مناسبی برای ارزیابی عملکرد الگوریتم‌های مقابله با برخورد مبتنی بر الگوریتم درختی نیست. از این‌رو در این مقاله از تعداد بیت‌های تبادل شده بین تگ‌ها و دستگاه خواننده برای ارزیابی الگوریتم پیشنهادی استفاده خواهیم کرد.

شکل (۵.الف) تعداد بیت‌های ارسال شده از سوی دستگاه خواننده به تگ‌ها را نشان می‌دهد. از آنجایی بعد از هر اسلات دستگاه خواننده سیگنال فیدبکی برای تعیین وضعیت تگ‌ها ارسال می‌کند؛ طول این سیگنال (تعداد بیت) و تعداد ارسال این سیگنال تاثیر مستقیم بر تعداد بیت‌های ارسالی دستگاه خواننده دارد. در الگوریتم پیشنهادی این سیگنال تشکیل شده از یک رشته بیت که نشان دهنده محل یک بیت از شماره شناسه و همچنین یک بیت “0” یا “1” که نشان دهنده نوع اسلات می‌باشد. دراین شبیه‌سازی مشاهده می‌شود حجم اطلاعات ارسال شدده از سوی دستگاه خواننده در الگوریتم پیشنهادی کمتر از سایر الگوریتم‌ها است.



شکل (۴): بلوک دیاگرام (ساختار) تگ غیر قابل نوشتن

[DOR: 20.1001.1.26765810.1395.13.4.3.3]
Journal of Iranian Association of Electrical and Electronics Engineers - Vol.13- No.4-Winter 2016

در شکل (۵.الف) مشخص است تعداد بیت‌های ارسالی از سوی دستگاه خواننده در الگوریتم IDS بیشتر از سایر الگوریتم‌ها است دلیل این امر این است که در هر اسلات این الگوریتم تنها یک بیت از سوی تگ‌ها ارسال می‌شود.

شکل (۵.ب) نشان می‌دهد تعداد بیت‌های ارسالی تگ‌ها به دستگاه خواننده در الگوریتم EAA، الگوریتم IDS و الگوریتم پیشنهادی کمتر ازسایر الگوریتم‌ها و بسیار نزدیک به هم هستند. علت این امر تلاش دستگاه خواننده در این الگوریتم‌ها در هنگام مواجه با بیت برخورد کرده در راستای بر طرف کردن برخورد بدونه نیاز به تبادل اطلاعات اضافی است.

میانگین زمان مورد نیاز برای شناسایی یک تگ عبارت از: زمان مصرفی برای شناسایی همه تگ‌های قرار گرفته در محدوده تحت پوشش تقسیم بر تعداد تگ‌ها. شکل (۵.ج) میانگین شناسایی یک تگ در الگوریتم‌های مختلف را نشان می‌دهد. مطابق این شبیه سازی الگوریتم پیشنهادی به زمان کمتری نیاز دارد (زمان مورد نیاز برای ارسال یک بیت ۵ میکروثانیه فرض شده است).

در اکثر الگوریتم‌های مبتنی بر تشخیص بیت برخورد، ابتدا همه تگ اقدام به ارسال پیشوند خود کرده و در مراحل بعد از ارسالی مجدد این پیشوند خودداری خواهند کرد. به عنوان مثال اگر تگ‌های با شماره شناسه‌های ID1=01010000، ID2=01010001 و ID3=01010010 را در نظر بگیریم. هر تگ پیشوند برابر خود با سایر تگ‌ها (“010100”) را تنها یک بار ارسال خواهد کرد.

در سیستم‌های واقعی طول‌های متداول شماره شناسه ۶۴ و یا ۹۶ بیت می‌باشد و اکثر تگ‌ها دارای پیشوندهای برابری هستند [14,15]. عدم ارسال این پیشوندها موجب کاهش چشم‌گیری در حجم اطلاعات تبادل شده خواهد شد. شکل‌های (۶.الف)، (۶.ب) و (۶.ج) به ترتیب تعداد بیت‌های ارسالی از سوی دستگاه خواننده به تگ‌ها، تعداد بیت‌های ارسالی از سوی تگ‌ها به دستگاه خواننده و میانگین زمان شناسایی یک تگ در الگوریتم‌ها مختلف را با هم مقایسه می‌کند. تعداد تگ‌های موجود در ناحیه تحت پوشش در این شبیه سازی ۲۵۶ تگ در نظر گرفته شده است. محور افقی در این شکل تعداد طول پیشوندهای برابر در شماره شناسه تگ‌ها را نشان می‌دهد. واضح است که الگوریتم پیشنهادی دارای عملکرد بهتری است.

در اولین ارتباط بین تگ و دستگاه خواننده سیگنالی از سوی دستگاه خواننده برای فعال کردن تگ‌ها ارسال می‌کند. اگر قبل از اینکه تگ‌های قرار گرفته در ناحیه تحت پوشش شناسایی شوند تگ یا تگ‌های جدیدی وارد ناحیه تحت پوشش شوند. دستگاه خواننده اقدام به شناسایی تگ‌های تازه وارد شده نخواهد کرد. این تگ‌ها تا پایان شناسایی تگ‌های موجود در ناحیه در حالت سکون باقی خواهند ماند. شکل (۷) عملکرد الگوریتم‌های مختلف را در این حالت نشان می‌دهد. در این شبیه‌سازی تعداد تگ‌های موجود در ناحیه تحت پوشش ۲۵۶ تگ و پیشوند برابر شماره شناسه‌ها ۳۲ بیت در نظر گرفته شده است.

پارامتر Ra نسبتی از تگ‌های جدید الورود بوده و از رابطه زیر پیروی می‌کند.

(۸) تعداد تگ‌ها×Ra =تعداد تگ‌های وارد شده

در شبیه‌سازی شکل (۷) تعداد تگ‌ها ۲۵۶ در نظر گرفته شده است. به عنوان مثال اگر Ra=0.5 باشد دستگاه خواننده بعد از شناسایی ۲۵۶ تگ موجود در ناحیه تحت پوشش، ۱۲۸ (128=256×0.5) تگ را نیز باید شناسایی کند. شکل‌های (۷.الف)، (۷.ب) و (۷.ج) به ترتیب تعداد بیت‌های ارسالی دستگاه خواننده، تعداد بیت‌های ارسالی تگ‌ها و میانگین زمان شناسایی یک تگ در الگوریتم‌های مختلف را نشان می‌دهد. الگوریتم پیشنهادی دارای زمان کمتر و در نتیجه سرعت شناسایی بیشتری است.

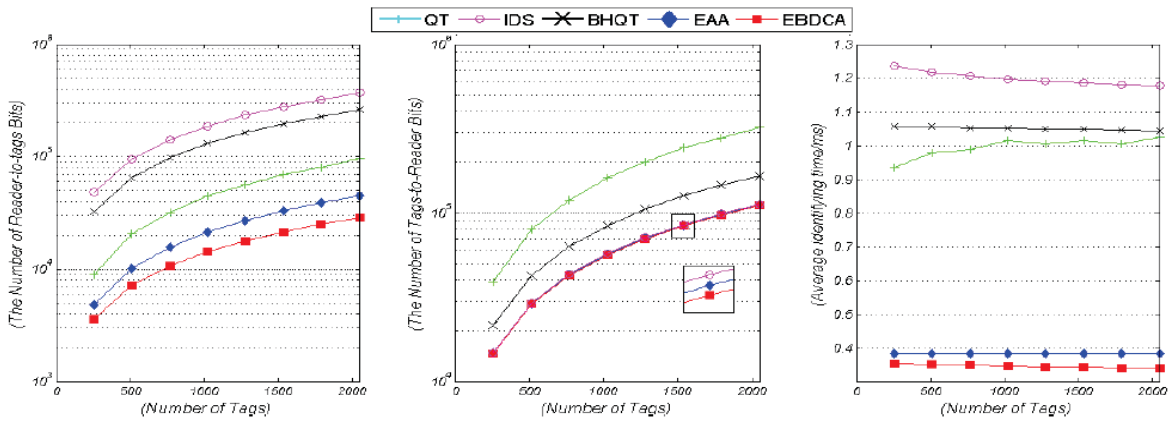
علاوه بر سرعت شناسایی میزان پیچیدگی تگ‌های مورد استفاده در هر الگوریتم معیار ارزیابی الگوریتم‌های مقابله با برخورد است. جدول (۲) میزان پیچیدگی تگ در الگوریتم‌های مختلف را با هم مقایسه می کند. با توجه به این جدول تگ در الگوریتم QT دارای ساده‌ترین ساختار است با این حال در شبیه‌سازی‌های انجام شده مشاهده شد این الگوریتم در بین سایر الگوریتم‌های انتخابی برای مقایسه، دارای سرعت عملکرد کمتری می‌باشد.

با وجود پیچیدگی ساختار تگ در الگوریتم BHQT، این الگوریتم از سرعت شناسایی مناسبی برخوردار نیست (شکل‌های (۵)، (۶) و (۷)). هزینه اجرایی کردن این الگوریتم نسبت به چهار الگوریتم دیگر مورد ارزیابی قرار گرفته، بیشتر می‌باشد.

الگوریتم IDS در هر اسلات یک بیت ارسال می‌کند. از این‌رو نیاز دارد که تگ‌های مورد استفاده در این الگوریتم از توانایی ارسال بیت به بیت شماره شناسه برخوردار باشند. بعد از هر اسلات شکست خورده هر تگ با استفاده از سیگنال ارسالی دستگاه خواننده وضعیت خود را برای ارسال یا عدم رسال تعیین کند. همچنین هر تگ در این الگوریتم به یک شمارنده برای تعیین موقعیت خود در ساختار درختی نیاز دارد.

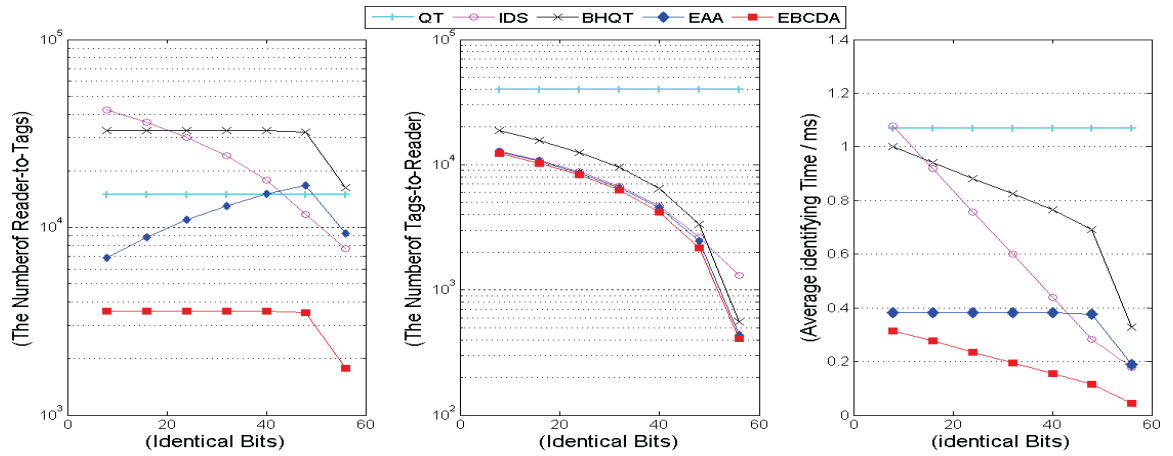
در الگوریتم‌های EAA و EBCDA هر تگ بعد از اسلات شکست خورده، بیت‌های موجود از این محل برخورد به بعد را مجدداً ارسال می‌کنند (بخش‌های ۲ و ۳-۲). در الگوریتم EBCDA هر تگ تنها به یک شمارنده (Tc) برای تعیین موقعیت خود در ساختار درختی نیاز دارد. ولی تگ‌های مورد استفاده در الگوریتم EAA به دو شمارنده نیاز دارند.

با توجه به جدول (۲) و مطالب توضیح داده شده مشاهده می‌شود. در الگوریتم پیشنهادی (EBCDA) پیچیدگی خاصی (نسبت به الگوریتم‌های مانند BHQT و EAA) به ساختار تگ اضافه نشده است. با این وجود و با توجه به نتایج شبیه‌سازی نشان داده شده در شکل‌های (۵)، (۶) و (۷) مشاهده می‌شود الگوریتم پیشنهادی باعث کاهش چشم‌گیری در حجم اطلاعات تبادل شده و در نتیجه سرعت شناسایی تگ‌ها شده است.



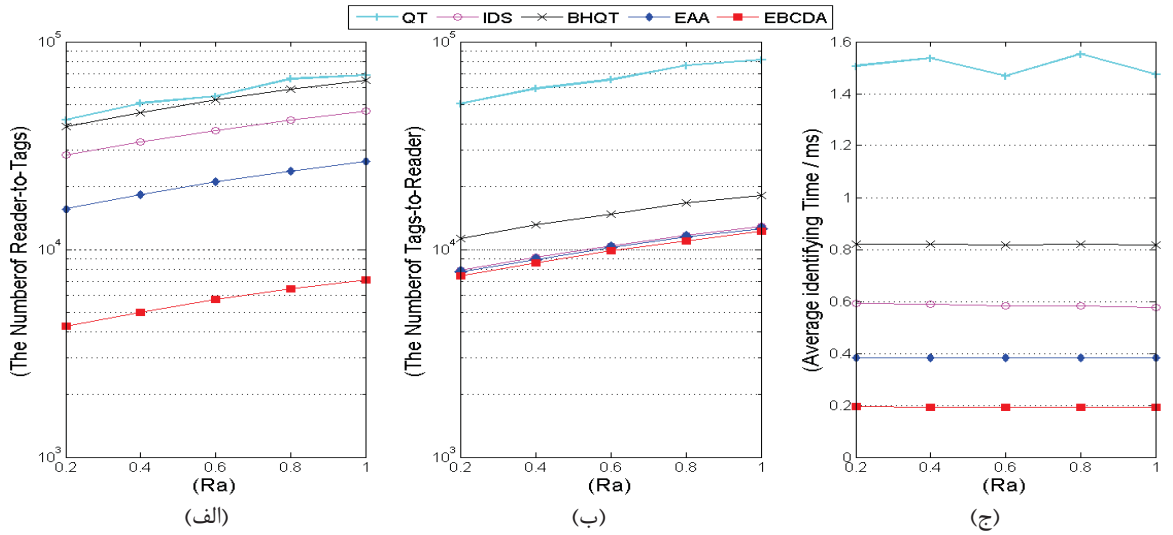
(الف) (ب) (ج)

شکل (۵): مقایسه عملکرد الگوریتم با استفاده از تعداد بیت تبادل شده (طول شماره شناسه=۶۴)



(الف) (ب) (ج)

شکل (۶): مقایسه عملکرد الگوریتم به ازایی پیشوندهای برابر در شماره شناسه(طول شماره شناسه=۶۴)



(الف) (ب) (ج)

شکل (۷): مقایسه عملکرد الگوریتم در مواجه با تگ‌های جدیدالورود به ناحیه تحت پوشش (طول شماره شناسه=۶۴)

[DOR: 20.1001.1.26765810.1395.13.4.3.3]

جدول (۲): میزان پیچیدگی تگ در الگوریتم‌های مختلف

الگوریتم	QT	EBCDA	IDS	EAA	BHQT
ارسال بیت به بیت شماره شناسه	–	–	✔	–	–
تشخیص بیت برخورد کرده با استفاده از سیگنال دریافتی ازدستگاه خواننده	–	✔	✔	✔	✔
توانایی ارسال بیت‌های خاصی از شماره شناسه	–	✔	–	✔	✔
تشخیص محل بیت برخورد	–	–	–	–	✔
توانایی مقایسه سیگنال دریافتی با شماره شناسه	✔	–	–	–	✔
تعداد شمارنده مورد نیاز تگ	۱	۱	۱	۲	–

۵- نتیجه‌گیری:

در این مقاله الگوریتم جدیدی با نام EBCDA برای مقابله با برخورد تگ‌ها در سیستم‌های RFID معرفی شد. این الگوریتم مبتنی بر الگوریتم درختی و تشخیص بیت برخورد کرده می‌باشد. برای کاهش حجم اطلاعات تبادل شده در الگوریتم EBCDA دستگاه خواننده در مواجه با بیت برخورد تا حد ممکن اقدام به بر طرف کردن بیت برخورد کرده و از تبادل اطلاعات اضافی به شدت می‌کاهد. نتـایج شبیه‌سازی-های انجام شده نیز حاکی از همین امر است. این امر در حـالی است که تگ‌ها مورد استفاده در الگوریتم پیشنهادی نه تنها دارای ساختار خیلی پیچیده‌ای نمی‌باشـد، بلکه ساخـتار تگ در این الگوریتـم از تـگ‌های مورد استفاده در الگوریتم‌های مانند EAA و BHQT ساده‌تر است.

مراجع

[1] Finkenzeller, K., RFID handbook fundamentals and application in contactless smart cards and identification, 3ed edition, United Kingdom: Wiley, 2010.

[2] Dobkin, D., The RF In RFID Passive UHF RFID In Practice, Elsevier, 2008.

[3] Teoh Q. J., Kramkar N. C., Handbook of Smart Antennas for RFID systems, John Wiley & Sons, 2010.

[4] Alchazidis, N., Data Integrity in RFID Systems, Master Thesis, Naval Postgraduate School, Monterey, California, 2006.

[5] Kavindya, S. D., Performance of Slotted ALOHA Anti-collision Protocol for RFID Systems Under Interfering Environments, Master Thesis, University of Wichita State, 2010.

[6] Deng, D. J., Taso, H. W., "Optimal Dynamic Framed Slotted ALOHA Based Anti-collision Algorithm for RFID Systems", Wireless Personal Communication. Vol. 59, No. 1, pp. 109-122, 2011.

[7] Wu, H., Zeng, Y., Fneg, J., Gu, Y., "Binary Tree Slotted ALOHA for passive RFID Tag Anti-Collision", IEEE Transactions on Parallel and Distributed Systems, Vol, pp, Issue 99, pp. 1-14, 2012

[8] Klair, K. D., Chin K. W., Raad, R., "A Survey and Tutorial of RFID Anti-Collision Protocols", IEEE Communications Surveys & Tutorials, Vol. 12, No. 3, pp. 400-420, 2010.

[9] He, M., Horng, S. J., Fan, P., Khan, M. K., Run, R. S., Lai, J. L., Chen, R. J., "A Fast RFID Identification Algorithm Based on Counter and Stack", Expert Systems with Applications, Vol. 38, Issue 6, pp. 6829-6838, June 2011.

[10] Law, C., Lee, K., Siu, K. Y., "Efficient Memory less Protocol for Tag Identification (extended abstract)", in Proceedings of the 4th International Workshop on Discrete Algorithm and Methods for Mobile Computing and Communications, Toronto, 2000.

[11] Bo, F., Tao, J. L., Bo, j. G., Hua Z. D., "ID-Binary Tree Stack Anti-collision Algorithm for RFID", Proc. of IEEE Symposium on Computer and communication, Poland, 2006.

[12] Gou, H. and Yoo, Y., "A Bit Collision Detection based Hybrid Query Tree Protocol for Anti-Collision in RFID System," Proc. of IEEE Int. Conf. on Computer and Information Technology, Pafos, 2011.

[13] Liu, L., Xie, Z., Xi, J., Lai, S., "An Improved Anti-Collision Algorithm in RFID System", Proc. of IEEE Int. Conf. on Mobile Technology Application and Systems, Guangzhou, 2005.

[14] EPCglobal, EPCTM Radio Frequency Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860MHz-969MHz, Version 1.0.9, 2005.

[15] ISO/IEC FDIS 18000-6C, Information Technology Automatic and Data Capture Techniques Radio Frequency Identification for Item Management Air interface- part 6: Parameters for Air Interface Communication at 860-960MHz, 2003.

[۱۶] رضایی، فرخی، "بهینه سازی چند منظوره برای کنترل توان در سیستم‌های سلولار CDMA" نشریه انجمن مهندسی برق و الکترونیک ایران، سال یازدهم، شماره دوم،صفحات ۳۳-۴۲، دانشگاه امیرکبیر تهران، پاییز و زمستان ۱۳۹۳.

[۱۷] سلماسی، گلستانی، "ظرفیت شبکه‌های حذفی تحت کدینگ فضایی شبکه" نشریه انجمن مهندسی برق و الکترونیک ایران، سال هشتم شماره دوم، صفحات ۱-۱۲، دانشگاه امیرکبیر تهران، پاییز و زمستان ۱۳۹۰.

زیر نویس‌ها

[1] Law
[2] Feng Bo
[3] Haosong Gou
[4] Younghwan Yoo
[5] Mingxing He
[6] Manchester code

[DOR: 20.1001.1.26765810.1395.13.4.3.3]