**ĐẠI HỌC QUỐC GIA HÀ NỘI**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ**

**Ngô Quang Anh**

**Tên đề tài**

# NGHIÊN CỨU CHUẨN KẾT NỐI KHÔNG DÂY ZIGBEE/IEEE 802.15.4

**KHÓA LUẬN TỐT NGHIỆP HỆ ĐẠI HỌC CHÍNH QUI**

**Ngành : Điện Tử Viễn Thông**

**Cán bộ hướng dẫn : GS.TSKH Phan Anh**

**Cán bộ đồng hướng dẫn: CN. Trần Anh Tuấn**

**HÀ NỘI – 2005**

Lời cảm ơn

*Đầu tiên, em xin phép được gửi lời cảm ơn sâu sắc đến tất cả các thầy cô giáo trong trường đã dìu dắt em trong suốt bốn năm học đại học. Đặc biệt em xin gửi lời cảm ơn chân thành tới GS.TSKH* Phan Anh*, thầy đã tạo cho em động lực rất lớn để hoàn thành bản luận văn này.*

*Em cũng xin cảm ơn anh* Trần Anh Tuấn, *các anh chị trên trung tâm, gia đình và bạn bè đã hết lòng hướng dẫn, chỉ bảo và luôn tạo mọi điều kiện tốt nhất cho em trong suốt thời gian qua.*

Sinh viên

Ngô Quang Anh

*Tóm tắt nội dung khóa luận*

*Hiện nay công nghệ ZigBee/ IEEE 802.15.4 đang được coi là hướng giải quyết hiệu quả cho vấn đề liên lạc trong dải băng tần eo hẹp và liệu pháp sử dụng chung kênh tần số giữa các thiết bị. Công nghệ ZigBee hoạt động ở băng tần 868/915 MHz ở Châu Âu và 2.4 GHz ở Mỹ và Nhật, được áp dụng cho những hệ thống điều khiển có tốc độ truyền tin thấp và chu kỳ hoạt động lâu dài. Công nghệ này tỏ ra ưu việt hơn Bluetooth ở mức độ tiêu hao năng lượng thấp, độ trễ truyền tin nhỏ, dễ dàng mở rộng, giá thành thấp. Trong khuôn khổ của đề tài này, em đã khảo cứu về công nghệ ZigBee và mô phỏng thành công quá trình định tuyến trong mạng mesh của ZigBee. Chương trình mô phỏng được viết bằng ngôn ngữ Visual C và chạy mô phỏng trên MatLab.*

# MỤC LỤC

# Lời nói đầu

Hàng ngày chúng ta đều thấy những ví dụ mới về cách thức mà công nghệ thông tin và viễn thông (ICT) tác động làm thay đổi cuộc sống của con người trên thế giới. Từ mức độ này hay mức độ khác, cuộc cách mạng kỹ thuật số đã lan rộng đến mọi ngõ ngách trên toàn cầu.

Trong mạng viễn thông ngày này, con người đang quản lý, trao đổi, giao tiếp tranh luận, "làm chính trị", mua bán và thử nghiệm – nghĩa là thực hiện tất cả các loại hình hoạt động bằng cách thức mà chỉ có ICT mới có thể làm được. Mạng viễn thông đã tạo ra một cầu nối liên kết loài người trên khắp hành tinh của chúng ta, và đang mở rộng không ngừng, đầy hứa hẹn, hy vọng và không một chút bí ẩn. Tuy vậy, trong một dải băng tần eo hẹp vẫn còn tồn đọng nhiều thách thức nếu muốn đạt được đầy đủ tiềm năng đó. Các nhà khoa học trên thế giới đã nghĩ đến việc sử dụng các băng tần cao hơn, nhưng việc này đang vấp phải nhiều trở ngại vì công nghệ điện tử và chế tạo chưa theo kịp. Vì vậy một giải pháp cấp bách được đưa ra là sử dụng chung kênh tần số, mặc dù vẫn còn nhiều vấn đề phát sinh, ví dụ như là can nhiễu lẫn nhau giữa các thiết bị cùng tần số, hay là vấn đề xung đột giữa các thiết bị... Một trong những công nghệ mới hiện đang được ứng dụng trong các mạng liên lạc đã đạt được hiệu quả là công nghệ ZigBee.

Công nghệ ZigBee là công nghệ được áp dụng cho các hệ thống điều khiển và cảm biến có tốc độ truyền tin thấp nhưng chu kỳ hoạt động dài. Công nghệ ZigBee hoạt động ở dải tần 868/915 MHz và 2,4 GHz, với các ưu điểm là độ trễ truyền tin thấp, tiêu hao ít năng lượng, giá thành thấp, ít lỗi, dễ mở rộng, khả năng tương thích cao. Trong luận văn này, em muốn trình bày các khảo cứu của em về công nghệ ZigBee và mô phỏng thuật toán định tuyến của ZigBee để có thể hiểu rõ hơn về công nghệ này.

Hy vọng thông qua các vấn đề được đề cập trong bản luận văn này, bạn đọc sẽ có được sự đánh giá và hiểu biết sâu sắc hơn về công nghệ ZigBee/IEEE 802.15.4 và vai trò cũng như tiềm năng của nó trong cuộc sống.

# CHƯƠNG 1 TỔNG QUAN VỀ MẠNG WPAN.

## 1.1 Khái niệm mạng WPAN (Wireless Personal Area Network).

Mạng cá nhân không dây được sử dụng để phục vụ truyền thông tin trong những khoảng cách tương đối ngắn. Không giống như mạng WLAN(mạng cục bộ không dây), mạng WPAN có thể liên lạc hiệu quả mà không đòi hỏi nhiều về cớ sở hạ tầng. Tính năng này cho phép có thêm các hướng giải quyết rẻ tiền, nhỏ gọn mà vẫn đem lại hiệu suất cao trong liên lạc nhất là trong một băng tần eo hẹp.

## 1.2 Sự phát triển của mạng WPAN

Trong suốt giữa thế kỷ 20 mạng điện thoại có dây đã được dử dụng rộng rãi và là một nhu cầu tất yếu cho cuộc sống. Tuy nhiên một thực tế đặt ra là khi xã hội ngày càng phát triển, các nhu cầu dịch vụ cũng vì thế mà tăng theo, trong thông tin liên lạc chi phí cho những phát sinh của mạng điện thoại có dây cũng tăng cộng thêm nhu cầu về tính cơ động trong thông tin liên lạc,…Và mạng điện thoại tế bào ra đời chính là xu phát triển, mở rộng tất yếu của mạng điện thoại có dây. Mạng điện thoại tế bào và biện pháp sử dụng lại tần số là phượng pháp duy nhất để giải quyết vấn đề nhiều người dùng độc lập trên một dải tần vô tuyến hạn chế (Ví dụ như các chuẩn GSM, IS-136, IS-95).

Trong thời gian giữa những năm 198x, chuẩn IEEE 802.11 ra đời phục vụ cho mạng WLAN (wireless local area network) nhằm thỏa mãn nhu cầu của các vùng tế bào nhỏ hơn nhưng lại có lưu lượng dữ liệu và mật độ người dùng cao. Trong khi mà IEEE 802.11 đề cập đến những thứ như là tốc độ truyền tin trong Ethernet, chuyển tiếp tin, lưu lượng dữ liệu trong khoảng cách tương đối xa (khoảng 100m), thì WPAN lại tập trung giải quyết vấn đề về điều khiển dữ liệu trong những khoảng không gian nhỏ hơn (bán kính 30m). Tính năng của chuẩn mạng WPAN là suy hao năng lượng nhỏ, tiêu tốn ít năng lượng, vận hành trong vùng không gian nhỏ, kích thước bé. Chính vì thế mà nó tận dụng được tốt nhất ưu điểm của kỹ thuật sử dụng lại kênh tần số, đó là

giải quyết được vấn đề hạn chế về băng tần như hiện nay. Nhóm chuẩn IEEE 802.15 ra đời để phục vụ cho chuẩn WPAN.

## 1.3 Phân loại các chuẩn mạng WPAN.

IEEE 802.15 có thể phân ra làm 3 loại mạng WPAN, chúng được phân biệt thông qua tốc độ truyền, mức độ tiêu hao năng lượng và chất lượng dịch vụ (QoS: quality of service).

- WPAN tốc độ cao (chuẩn IEEE 802.15.3) phù hợp với các ứng dụng đa phương tiện yêu cầu chất lượng dịch vụ cao.
- WPAN tốc độ trung bình (chuẩn IEEE 802.15.1 / Bluetooth) được ứng dụng trong các mạng điện thoại tế báo đến máy tính cá nhân bỏ túi PDA và có QoS phù hợp cho thông tin thoại.
- WPAN tốc độ thấp (IEEE 802.15.4 / LR-WPAN) dùng trong các sản phẩm công nghiệp dùng có thời hạn, các ứng dụng y học chỉ đòi hỏi mức tiêu hao năng lượng thấp, không yêu cầu cao về tốc độ truyền tin và QoS. Chính tốc độ truyền dữ liệu thấp cho phép LR-WPAN tiêu hao ít năng lượng. Trong chuẩn này thì công nghệ ZigBee/IEEE802.15.4 chính là một ví dụ điển hình.

## 1.4 Khái quát về ZigBee/ IEEE 802.15.4

### *1.4.1* Khái niệm

Cái tên ZigBee được xuất phát từ cách mà các con ong mật truyền những thông tin quan trọng với các thành viên khác trong tổ ong. Đó là kiểu liên lạc "**Zig**-Zag" của loài ong "honey**Bee**". Và nguyên lý ZigBee được hình thành từ việc ghép hai chữ cái đầu với nhau. Việc công nghệ này ra đời chính là sự giải quyết cho vấn đề các thiết bị tách rời có thể làm việc cùng nhau để giải quyết một vấn đề nào đó.

### *1.4.2* Đặc điểm

Đặc điểm của công nghệ ZigBee là tốc độ truyền tin thấp, tiêu hao ít năng lượng, chi phí thấp, và là giao thức mạng không dây hướng tới các ứng dụng điều khiển từ xa và tự động hóa.Tổ chức IEEE 802.15.4 bắt đầu làm việc với chuẩn tốc độ thấp được một thời gian ngắn thì tiểu ban về ZigBee và tổ chức IEEE quyết định sát nhập và lấy tên ZigBee đặt cho công nghệ mới này. Mục tiêu của công nghệ ZigBee là nhắm tới

việc truyền tin với mức tiêu hao năng lượng nhỏ và công suất thấp cho những thiết bị chỉ có thời gian sống từ vài tháng đến vài năm mà không yêu cầu cao về tốc độ truyền tin như Bluetooth. Một điều nổi bật là ZigBee có thể dùng được trong các mạng mắt lưới (mesh network) rộng hơn là sử dụng công nghệ Bluetooth. Các thiết bị không dây sử dụng công nghệ ZigBee có thể dễ dàng truyền tin trong khoảng cách 10-75m tùy thuộc và môi trường truyền và mức công suất phát được yêu cầu với mỗi ứng dụng, Tốc độ dữ liệu là 250kbps ở dải tần 2.4GHz (toàn cầu), 40kbps ở dải tần 915MHz (Mỹ+Nhật) và 20kbps ở dải tần 868MHz(Châu Âu).

Các nhóm nghiên cứu Zigbee và tổ chức IEEE đã làm việc cùng nhau để chỉ rõ toàn bộ các khối giao thức của công nghệ này. IEEE 802.15.4 tập trung nghiên cứu vào 2 tầng thấp của giao thức (tầng vật lý và tầng liên kết dữ liệu). Zigbee còn thiết lập cơ sở cho những tầng cao hơn trong giao thức (từ tầng mạng đến tầng ứng dụng) về bảo mật, dữ liệu, chuẩn phát triển để đảm bảo chắc chắn rằng các khách hang dù mua sản phẩm từ các hãng sản xuất khác nhau nhưng vẫn theo một chuẩn riêng để làm việc cùng nhau được mà không tương tác lẫn nhau.

Hiện nay thì IEEE 802.15.4 tập trung vào các chi tiết kỹ thuật của tầng vật lý PHY và tầng điều khiển truy cập MAC ứng với mỗi loại mạng khác nhau *(mạng hình sao, mạng hình cây, mạng mắt lưới)*. Các phương pháp định tuyến được thiết kế sao cho năng lượng được bảo toàn và độ trễ trong truyền tin là ở mức thấp nhất có thể bằng cách dùng các khe thời gian bảo đảm (**GTSs**_guaranteed time slots). Tính năng nổi bật chỉ có ở tầng mạng Zigbee là giảm thiểu được sự hỏng hóc dẫn đến gián đoạn kết nối tại một nút mạng trong mạng mesh. Nhiệm vụ đặc trưng của tầng PHY gồm có phát hiện chất lượng của đường truyền (LQI) và năng lượng truyền (ED), đánh giá kênh truyền (CCA), giúp nâng cao khả năng chung sống với các loại mạng không dây khác.

### *1.4.3* Ưu điểm của ZigBee/IEEE802.15.4 với Bluetooth/IEEE802.15.1

- Zigbee cũng tương tự như Bluetooth nhưng đơn giản hơn, Zigbee có tốc độ truyền dữ liệu thấp hơn, tiết kiệm năng lượng hơn. Một nốt mạng trong mạng Zigbee có khả năng hoạt động từ 6 tháng đến 2 năm chỉ với nguồn là hai ácqui AA.
- Phạm vi hoạt động của Zigbee là 10-75m trong khi của Bluetooth chỉ là 10m (trong trường hợp không có khuếch đại).

- Zigbee xếp sau Bluetooth về tốc độ truyền dữ liệu. Tốc độ truyền của Zigbee là 250kbps tại 2.4GHz, 40kbps tại 915MHz và 20kbps tại 868MHz trong khi tốc độ này của Bluetooth là 1Mbps.
- Zigbee sử dụng cấu hình chủ-tớ cơ bản phù hợp với mạng hình sao tĩnh trong đó các thiết bị giao tiếp với nhau thông qua các gói tin nhỏ. Loại mạng này cho phép tối đa tới 254 nút mạng. Giao thức Bluetooth phức tạp hơn bởi loại giao thức này hướng tới truyền file, hình ảnh, thoại trong các mạng ad hoc (ad hoc là một loại mạng đặc trưng cho việc tổ chức tự do, tính chất của nó là bị hạn chế về không gian và thời gian). Các thiết bị Bluetooth có thể hỗ trợ mạng scatternet là tập hợp của nhiều mạng piconet không đồng bộ. Nó chỉ cho phép tối đa là 8 nút slave trong một mạng chủ-tớ cơ bản.
- Nút mạng sử dụng Zigbee vận hành tốn ít năng lượng, nó có thể gửi và nhận các gói tin trong khoảng 15msec trong khi thiết bị Bluetooth chỉ có thể làm việc này trong 3sec.

## 1.4 Mạng ZigBee/ IEEE 802.15.4 LR-WPAN.

Đặc điểm chính của chuẩn này là tính mềm dẻo, tiêu hao ít năng lượng, chi phí nhỏ, và tốc độ truyền dữ liệu thấp trong khoảng không gian nhỏ, thuận tiện khi áp dụng trong các khu vực như nhà riêng, văn phòng....

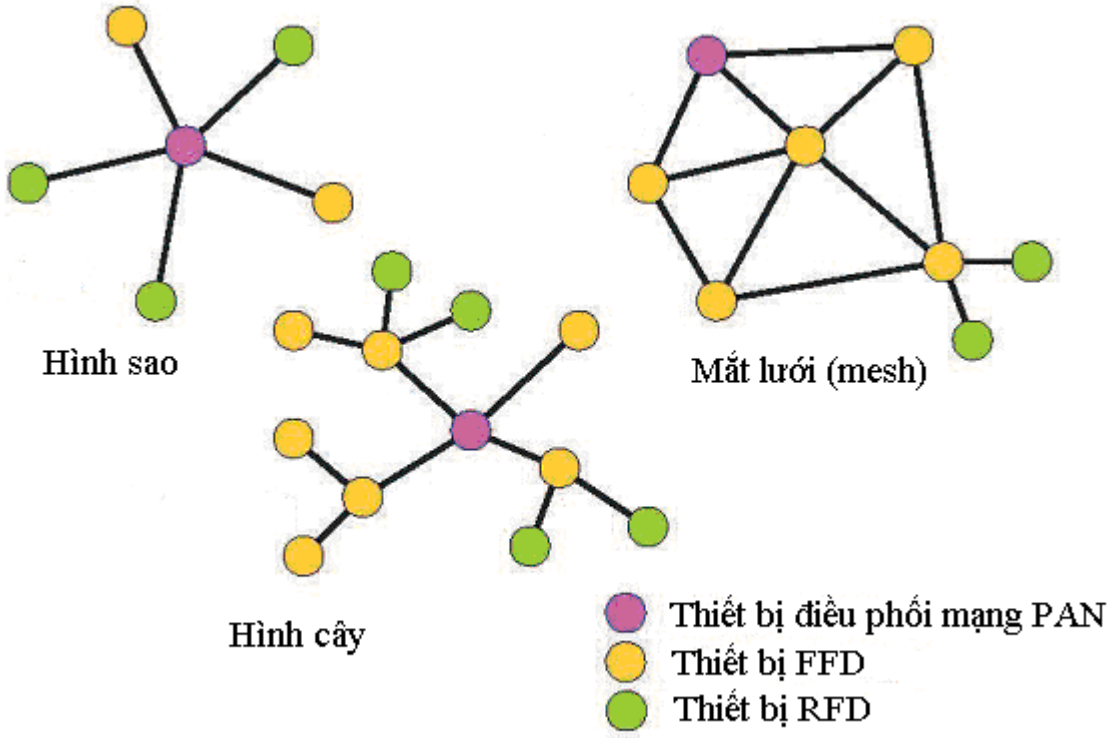
### 1.4.2 Thành phần của mạng LR-WPAN

Một hệ thống ZigBee/IEEE802.15.4 gồm nhiều phần tạo nên. Phần cơ bản nhất tạo nên một mạng là thiết bị có tên là FFD (full-function device), thiết bị này đảm nhận tất cả các chức năng trong mạng và hoạt động như một bộ điều phối mạng PAN, ngoài ra còn có một số thiết bị đảm nhận một số chức năng hạn chế có tên là RFD (reduced-function device). Một mạng tối thiểu phải có 1 thiết bị FFD, thiết bị này hoạt động như một bộ điều phối mạng PAN.

FFD có thể hoạt động trong ba trạng thái : là điều phối viên của toàn mạng PAN (personal area network), hay là điều phối viên của một mạng con, hoặc đơn giản chỉ là một thành viên trong mạng. RFD được dùng cho các ứng dụng đơn giản, không yêu cầu gửi lượng lớn dữ liệu. Một FFD có thể làm việc với nhiều RFD hay nhiều FFD, trong khi một RFD chỉ có thể làm việc với một FFD.

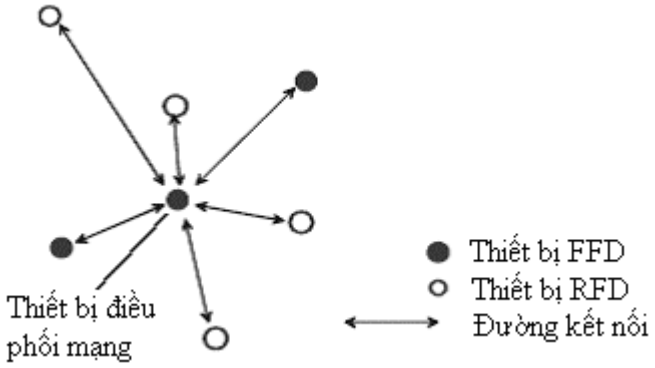
### 1.4.3 Kiến trúc liên kết mạng

Hiện nay Zigbee và tổ chức chuẩn IEEE đã đưa ra một số cấu trúc liên kết mạng cho công nghệ Zigbee. Các node mạng trong một mạng Zigbee có thể liên kết với nhau theo cấ u trúc mạng hình sao (star) cấu trúc mạng hình lưới( Mesh) cấu trúc bó cụm hình cây. Sự đa rạng về cấu trúc mạng này cho phép công nghệ Zigbee được ứng dụng một cách rộng rãi. Hình 1 cho ta thấy ba loại mạng mà ZigBee cung cấp: tôpô sao, tôpô mắt lưới, tôpô cây.




*Hình1.1 Cấu trúc liên kết mạng*

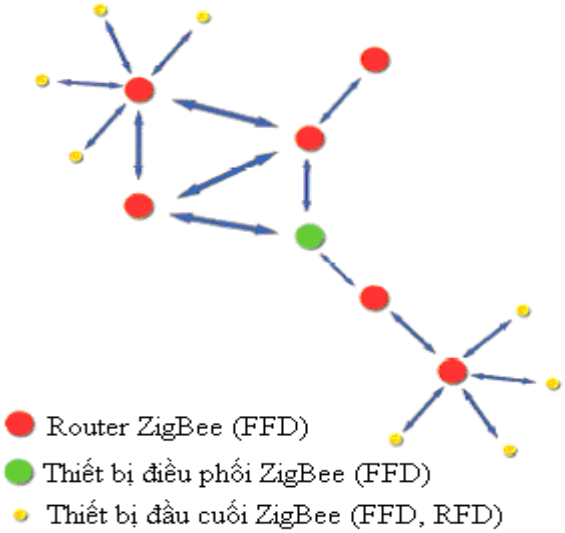
### 1.5.2.1 Cấu trúc liên kết mạng hình sao (Star)




*Hình1.2 Cấu trúc mạng hình sao*

Đối với loại mạng này, một kết nối được thành lập bởi các thiết bị với một thiết bị điều khiển trung tâm điều khiển được gọi là bộ điều phối mạng PAN. Sau khi FFD được kích hoạt lần đầu tiên nó có thể tạo nên một mạng độc lập và trở thành một bộ điều phối mạng PAN. Mỗi mạng hình sao đều phải có một chỉ số nhận dạng cá nhân của riêng mình được gọi là PAN ID(PAN identifier), nó cho phép mạng này có thể hoạt động một cách độc lập. Khi đó cả FFD và RFD đều có thể kết nối tới bộ điều phối mạng PAN. Tất cả mạng nằm trong tầm phủ sóng đều phải có một PAN duy nhất,các nốt trong mạng PAN phải kết nối với (PAN coordinator) bộ điều phối mạng PAN.

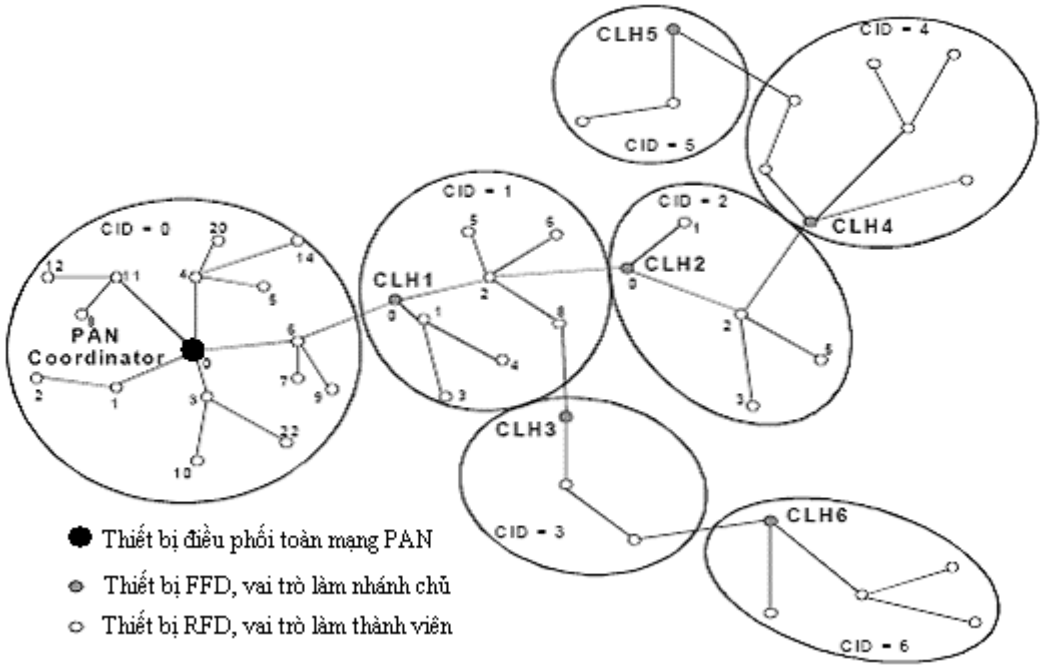
### 1.5.2.2 Cấu trúc liên kết mạng mắt lưới (mesh)




*Hình1.3 Cấu trúc mạng mesh*

Kiểu cấu trúc mạng này cũng có một bộ điều phối mạng PAN (PAN coordinator). Thực chất đây là kết hợp của 2 kiểu cấu trúc mạng hình sao và mạng ngang hàng, ở cấu trúc mạng này thì một thiết bị A có thể tạo kết nối với bất kỳ thiết nào khác miễn là thiết bị đó nằm trong phạm vi phủ sóng của thiết bị A. Các ứng dụng của cấu trúc này có thể áp dụng trong đo lường và điều khiển, mạng cảm biến không dây, theo dõi cảnh báo và kiểm kê (cảnh báo cháy rừng….).

### 1.5.2.3Cấu trúc liên kết mạng hình cây (cluster-tree)




*Hình1.4 Cấu trúc mạng hình cây*

Cấu trúc này là một dạng đặc biệt của cấu trúc mắt lưới, trong đó đa số thiết bị là FFDvà một RFD có thể kết nối vào mạng hình cây như một nốt rời rạc ở điểm cuối của nhánh cây. Bất kỳ một FFD nào cũng có thể hoạt động như là một coordinator và cung cấp tín hiệu đồng bộ cho các thiết bị và các coordinator khác vì thế mà cấu trúc mạng kiểu này có qui mô phủ sóng và khả năng mở rộng cao.Trong loại cấu hình này mặc dù có thể có nhiều coordinator nhưng chỉ có duy nhất một bộ điều phối mạng PAN (PAN coordinator).

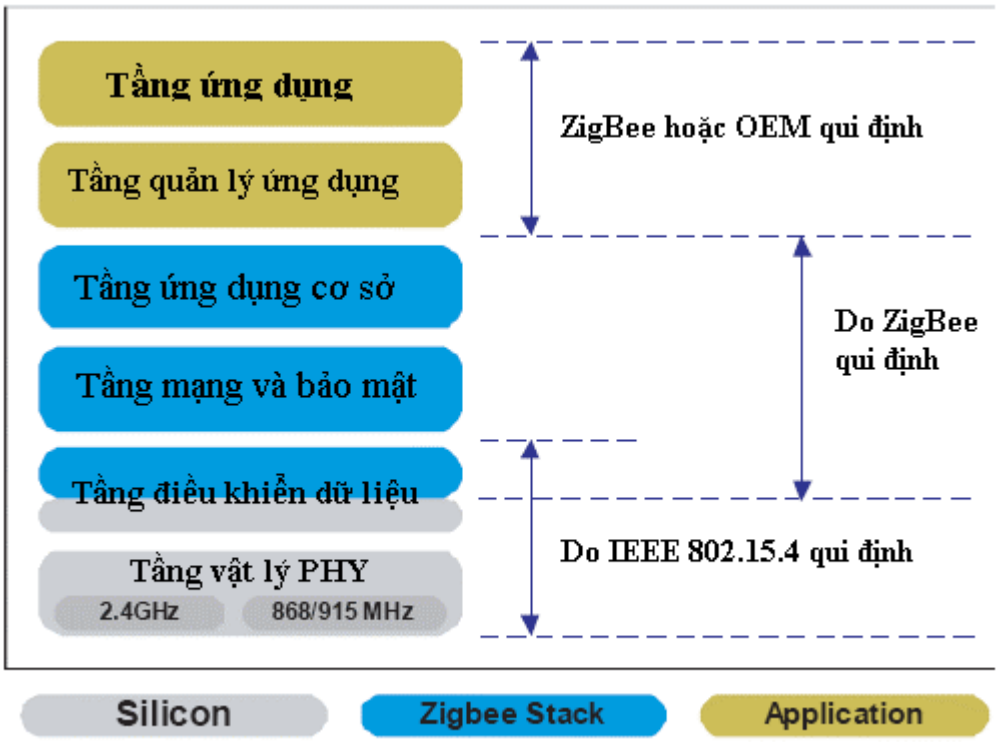
Bộ điều phối mạng PAN coordinator này tạo ra nhóm đầu tiên cách tự bầu ra người lãnh đạo cho mạng của mình, và gán cho người lãnh đạo đó một chỉ số nhận

dạng cá nhân đặc biệt  gọi là là CID-0  bằng cách tự thành lập CLH (cluster head) bằng CID-0 (cluster identifier), nó chọn một PAN identifier rỗi và phát khung tin quảng bá nhận dạng tới các thiết bị lân cận. Thiết bị nào nhận được khung tin này có thể yêu cầu kết nối vào mạng với CLH. Nếu  bộ điều phối mạng PAN (PAN coordinator) đồng ý cho thiết bị đó kết nối thì nó sẽ ghi tên thiết bị đó vào danh sách. Cứ thế thiết bị mới kết nối này lại trở thành CLH của nhánh cây mới và bắt đầu phát quảng bá định kỳ để các thiết bị khác có thể kết nối vào mạng. Từ đó có thể hình thành được các CLH1,CLH2,...(như hình1.4 ).

# CHƯƠNG 2 CHUẨN ZigBee/IEEE 802.15.4.

## 2.1 Mô hình giao thức của ZigBee/IEEE802.15.4

ZigBee/IEEE802.15.4 là công nghệ mới phát triển được khoảng gần một năm trở lại đây. Công nghệ này xây dựng và phát triển các tầng ứng dụng và tầng mạng trên nền tảng là hai tầng PHY và MAC theo chuẩn IEEE 802.15.4, chính vì thế nên nó thừa hưởng được ưu điểm của chuẩn IEEE802.15.4. Đó là tính tin cậy, đơn giản, tiêu hao ít năng lượng và khả năng thích ứng cao với các môi trường mạng. Dựa vào mô hình như hình2.1, các nhà sản xuất khác nhau có thể chế tạo ra các sản phẩm khác nhau mà vẫn có thể làm việc tương thích cùng với nhau.




*Hình2.1 Mô hình giao thức của ZigBee*

## *2.2* Tầng vật lý ZigBee/IEEE 802.15.4

Tầng vật lý (PHY) cung cấp hai dịch vụ là dịch vụ dữ liệu PHY và dịch vụ quản lý PHY, hai dịch vụ này có giao diện với dịch vụ quản lý tầng vật lý PLME (physical layer management). Dịch vụ dữ liệu PHY điều khiển việc thu và phát của khối dữ liệu PPDU (PHY protocol data unit) thông qua kênh sóng vô tuyến vật lý.

Các tính năng của tầng PHY là sự kích hoạt hoặc giảm kích hoạt của bộ phận nhận sóng, phát hiện năng lượng, chọn kênh, chỉ số đường truyền, giải phóng kênh truyền, thu và phát các gói dữ liệu qua môi trường truyền.

Chuẩn IEEE 802.15.4 định nghĩa ba dải tần số khác nhau theo khuyến nghị của Châu Âu, Nhật Bản, Mỹ.

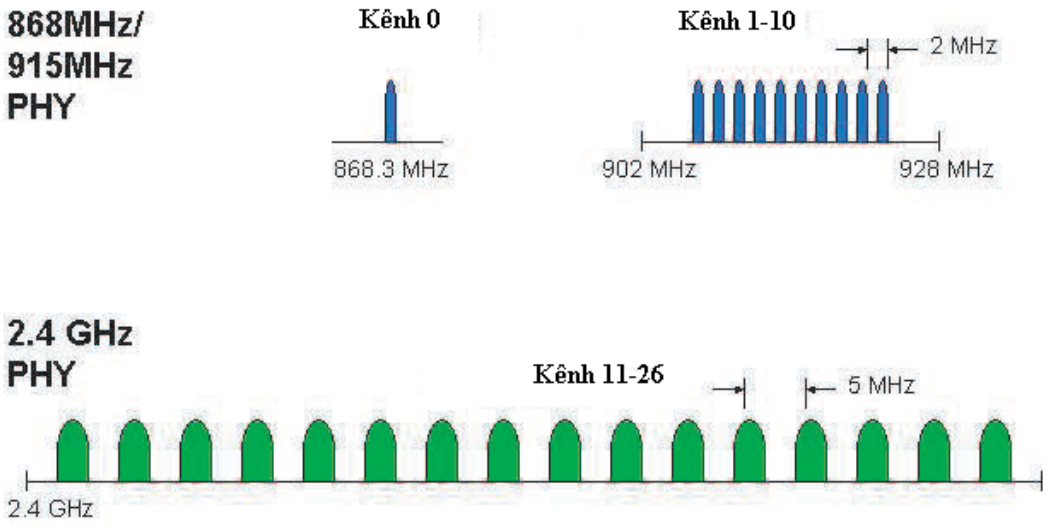
| PHY (MHz) | Băng tần (MHz) | Tốc độ chip (kchips/s) | Điều chế | Tốc độ bit (kb/s) | Tốc độ ký tự (ksymbol/s) | Ký tự |
|---|---|---|---|---|---|---|
| 868 | 868-868.6 | 300 | BPSK | 20 | 20 | Nhị phân |
| 915 | 902-928 | 600 | BPSK | 40 | 40 | Nhị phân |
| 2450 | 2400-2486.5 | 2000 | O-QPSK | 250 | 62.5 | Hệ 16 |

*Bảng 2.1 Băng tần và tốc độ dữ liệu.*

Có tất cả 27 kênh truyền trên các dải tần số khác nhau được mô tả như bảng dưới đây

| Tần số trung tâm (MHz) | Số lượng kênh (N) | Kênh | Tần số kênh trung tâm (MHz) |
|---|---|---|---|
| 868 | 1 | 0 | 868.3 |
| 915 | 10 | 1 – 10 | 906+2(k-1) |
| 2450 | 16 | 11 – 26 | 2405+5(k-11) |

*Bảng2.2 Kênh truyền và tần số*




*Hình2.2 Băng tần hệ thống của ZigBee*

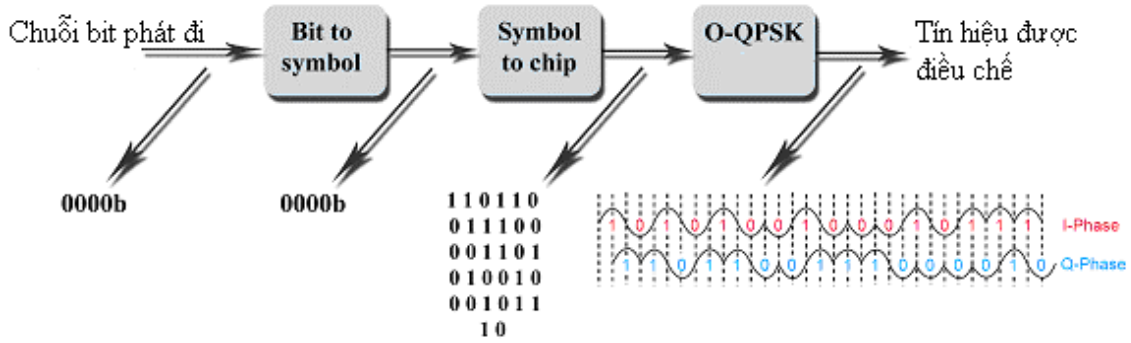
### *2.2.1* Mô hình điều chế tín hiệu của tầng vật lý.

#### *2.2.1.1*Điều chế tín hiệu của tầng PHY tại dải số 2.4 GHz

Tốc độ truyền dữ liệu của PHY 2405MHz có thể đạt tới 250 kb/s

##### *2.2.1.1.1 Sơ đồ điều chế*

Việc điều chế từ bít dữ liệu nhị phân sang dạng tín hiệu trong dải tần 2,4GHz được mô tả theo sơ đồ dưới đây. Một chuỗi số nhị phân "0000b" được biến đổi sang chuỗi dải tần cơ sở với định dạng xung.




*Hình2.3 Sơ đồ điều chế*

##### *2.2.1.1.2* Bộ chuyển bit thành ký tự :

Theo như sơ đồ trên thì đây là bước đầu tiên để mã hóa tất cả dữ liệu trong PPDU từ mã nhị phân sang dạng ký tự. Mỗi byte được chia thành ký tự và ký tự có nghĩa nhỏ nhất được phát đầu tiên. Đối với trường đa byte thì byte có nghĩa nhỏ nhất được phát đầu tiên ngoại trừ trường hợp trường byte đó liên quan đến bảo mật thì trong trường đó byte có nghĩa lớn nhất sẽ được phát trước.

##### *2.2.1.1.3* Bộ chuyển ký tự thành chip:

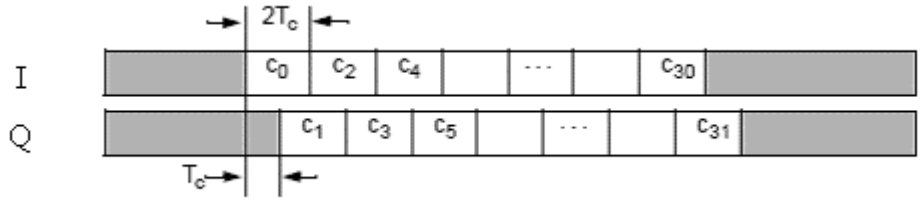
Theo như sơ đồ thì đây là bước thứ hai trong quá trình mã hóa. Mỗi ký tự dữ liệu được sắp xếp trong một chuỗi giả ngẫu nhiên (*Pseudo-random)* 32-chip. Chuỗi chip này được truyền đi với tốc độ 2Mchip/s với chip có nghĩa nhỏ nhất (c0) được truyền trước mọi ký tự.

| Ký tự dữ liệu (hệ thập phân) | Giá trị Chip ($c_0\ c_1 \ldots c_{30}\ c_{31}$) |
|---|---|
| 0 | 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 |
| 1 | 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 |
| 2 | 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 |
| 3 | 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 |
| 4 | 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 0 0 1 1 |
| 5 | 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 1 1 0 0 |
| 6 | 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 1 0 0 1 |
| 7 | 1 0 0 1 1 1 0 0 0 0 1 1 0 1 0 1 0 0 1 0 0 0 1 0 1 1 1 0 1 1 0 1 |
| 8 | 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 |
| 9 | 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 |
| 10 | 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 |
| 11 | 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 |
| 12 | 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 0 1 1 0 |
| 13 | 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 1 0 0 1 |
| 14 | 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 1 1 0 0 |
| 15 | 1 1 0 0 1 0 0 1 0 1 1 0 0 0 0 0 0 1 1 1 0 1 1 1 1 0 1 1 1 0 0 0 |

Bảng2.3 Sơ đồ biến đổi *symbol to chip*

### *2.2.1.1.4* Bộ điều chế O-QPSK :

Phương pháp điều chế được dùng ở đây là phương pháp điều chế khóa dịch pha góc ¼ có chọn gốc dịch pha ban đầu O-QPSK (Offset-Quadrature Phase Shift Keying) tương đương với phương pháp điều chế khóa dịch pha tối thiểu MSK (Minimum Shift Keying). QPSK là phương pháp hiệu quả đối với dải tần hạn chế. Mỗi phần tử tín hiệu biểu diễn cho 2 bit. Bằng việc sử dụng độ dịch offset trong O-QPSK, thay đổi pha trong tín hiệu tổng hợp tối đa là $90^0$, cũng trong trường hợp này mà dùng QPSK thì độ lệch pha tối đa là $180^0$.




*Hình 2.4 Pha của sóng mang*

Như vậy O-QPSK cung cấp một phương pháp tốt hơn QPSK khi kênh truyền có các thành phần không tuyến tính.

Biểu thức sau đây chỉ ra cách mà O-QPSK có thể diễn đạt:

$$s(t) = \frac{1}{\sqrt{2}} I(t)\cos 2\pi f_c t - \frac{1}{\sqrt{2}} Q(t-T_c)\sin 2\pi f_c t \qquad (1)$$

$f_c$ : là tần số trung tâm.

$T_c$ : là thời gian mà Q trễ đạt tới thay đổi pha $90^o$

Q : sóng mang vuông pha.

I : sóng mang cùng pha.

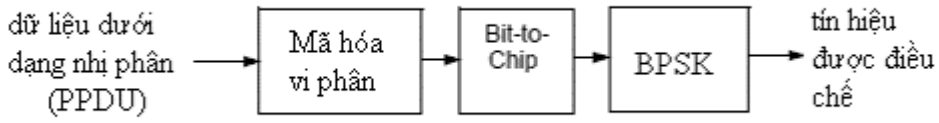
Việc sử dụng dạng xung nửa sin để khử đi những biến thiên biên độ. Công thức sau mô tả dạng xung nửa sin.

$$p(t) = \begin{cases} \sin(\pi \frac{t}{2T_e}) & , 0 \leq t \leq 2T_c \\ 0 & , \text{ngoài ra} \end{cases} \qquad (2)$$

### *2.2.1.2* Điều chế tín hiệu của tầng PHY tại dải tần 868/915MHz

Tốc độ truyền dữ liệu của ZigBee/IEEE802.15.4 PHY tại băng tần 868 MHz có thể đạt tới 20kb/s, và có thể đạt tới 40 kb/s ở băng tần 915MHz.

#### *2.2.1.2.1* Sơ đồ điều chế




*Hinh2.5 Sơ đồ điều chế*

#### *2.2.1.2.2* Bộ mã hóa vi phân

Mã hóa vi phân hay còn gọi là mã hóa trước. Khi cho tín hiệu nhị phân vào bộ mã hóa này thì bit có giá trị 0 sẽ được chuyển tiếp, có nghĩa là số được tách là số 1 nếu số liền trước nó là số 0 và ngược lại. Nếu một số được tách xung sai, lỗi này sẽ có xu hướng lan truyền đi , và để loại trừ việc này thì Lender đã đề nghị việc mã hóa trước số các dữ liệu. Có nghĩ là nếu chuỗi số dữ liệu thô là $R_n$ thì ta sẽ phát đi chuỗi số $E_n$ theo qui tắc:

$$E_n = R_n \oplus E_{n-1} \qquad (3)$$

Trong đó:

$1 \oplus 1 = 0 \oplus 0 = 0$

$0 \oplus 1 = 1 \oplus 0 = 1$

$E_n$ là chuỗi bit sau khi mã hóa

$R_n$ là chuỗi bít thô

$E_{n-1}$ là chuỗi bit mã hóa liền trước

#### *2.2.1.2.3* Bộ ánh xạ bit thành chip.

Mỗi bít đầu vào có thể ánh xạ sang chuỗi giả ngẫu nhiên (PN) 15-chip theo như bảng dưới đây. Trong khoảng thời gian mỗi symbol thì ký tự $c_0$ được truyền đầu tiên, ký tự $c_{14}$ được truyền sau cùng.

| Bit đầu vào | Giá trị chip<br>$(c_0\ c_1 \ldots c_{14})$ |
|---|---|
| 0 | 1 1 1 1 0 1 0 1 1 0 0 1 0 0 0 |
| 1 | 0 0 0 0 1 0 1 0 0 1 1 0 1 1 1 |

*Bảng 2.4 Biến đổi bit to chip*

#### *2.2.1.2.4* Bộ điều chế khóa dịch pha nhị phân BPSK

Chuỗi chip được điều chế trên sóng mang sử dụng phương pháp điều chế BPSK có dạng xung là xung cosin nâng (raised cosine). Tốc độ chip là 300kchip/s trong dải tần 868 MHz và đạt được 600 kchip/s trong dải tần 915MHz. Công thức sau mô tả dạng xụng này:

$$p(t) = \frac{\sin(\frac{\pi t}{T_c}).\cos(\frac{\pi t}{T_c})}{\pi t/T_c.(1-(4t^2/T_c^2))} \qquad (4)$$

### *2.2.2* Các thông số kỹ thuật trọng tầng vật lý của IEEE 802.15.4

#### *2.2.2.1*Chỉ số ED (energy detection)

Chỉ số ED đo đạc được bởi bộ thu ED. Chỉ số này sẽ được tầng mạng sử dụng như là một bước trong thuật toán chọn kênh. ED là kết quả của sự ước lượng công suất năng lượng của tín hiệu nhận được trong băng thông của kênh trong IEEE 802.15.4. Nó không có vai trò trong việc giải mã hay nhận dạng tín hiệu truyền trong kênh này. Thời gian phát hiện và xửlý ED tương đương khoảng thời gian 8 symbol. Kết quả phát hiện năng lượng sẽ được thông báo bằng 8 bit số nguyên trong khoảng từ 0x00 tới 0xff. Giá trị nhỏ nhất của ED (=0) khi mà công suất nhận được ít hơn mức +10dB so với lý

thuyết. Độ lớn của khoảng công suất nhận được để hiển thị chỉ số ED tối thiểu là 40dB và sai số là $\pm 6dB$.

#### *2.2.2.2*Chỉ số chất lượng đường truyền (LQI)

Chỉ số chất lượng đừong truyền LQI là đặc trưng chất lượng gói tin nhận được. Số đo này có thể bổ sung vào ED thu được, đánh giá tỷ số tín trên tạp SNR, hoặc một sự kết hợp của những phương pháp này. Giá trị kết quả LQI được giao cho tầng mạng và tâng ứng dụng xử lý.

#### *2.2.2.3*Chỉ số đánh giá kênh truyền (CCA)

CCA được sử dụng để xem xem khi nào một kênh truyền được coi là rỗi hay bận. Có ba phương pháp để thực hiện việc kiểm tra này:

- CCA 1 : "Năng lượng vượt ngưỡng". CCA sẽ thông báo kênh truyền bận trong khi dò ra bất kỳ năng lượng nào vượt ngưỡng ED.
- CCA 2 : "Cảm biến sóng mang". CCA thông báo kênh truyền bận chỉ khi nhận ra tín hiệu có đặc tính trải phổ và điều chế của IEEE802.15.4. Tín hiệu này có thể thấp hoặc cao hơn ngưỡng ED.
- CCA 3 : "Cảm biến sóng mang kết hợp với năng lượng vượt ngưỡng". CCA sẽ báo kênh truyền bận chỉ khi dò ra tín hiệu có đặc tính trải phổ và điều chế của IEEE 802.15.4 với năng lượng vượt ngưỡng ED.

### *2.2.3* Định dạng khung tin PPDU.

Mỗi khung tin PPDU bao gồm các trường thông tin.

- SHR (*synchronization header*) : đồng bộ thiết bị thu và chốt chuỗi bit
- PHR (PHY header): chứa thông tin độ dài khung
- PHY payload: chứa khung tin của tầng MAC

| Octets: 4 | 1 | 1 | | variable |
|---|---|---|---|---|
| Đầu khung | SFD (bắt đầu phân định khung) | Độ dài khung (7 bits) | Phần giành riêng (1 bit) | PSDU |
| SHR | | PHR | | PHY payload |

*Bảng 2.5 Định dạng khung PPDU*

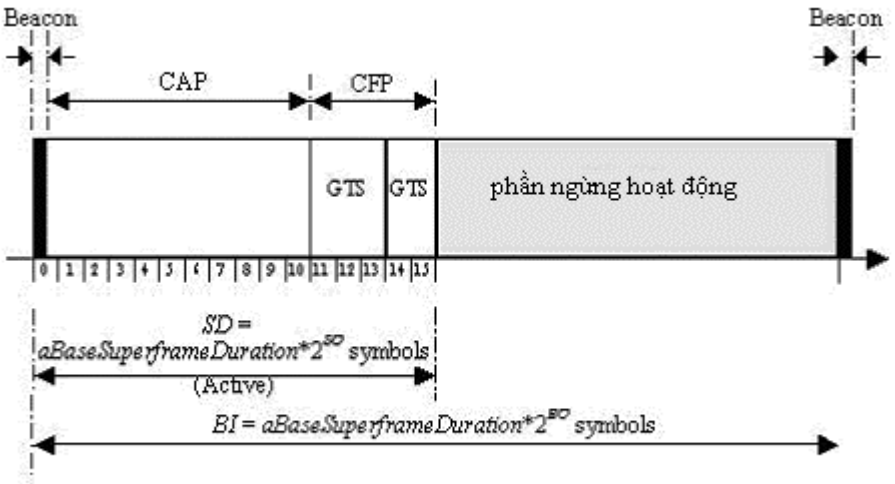
## *2.3* Tầng điều khiển dữ liệu ZigBee/IEEE 802.15.4 MAC

Tầng điều khiển môi trường truy cập MAC (media access control) cung cấp 2 dịch vụ là dịch vụ dữ liệu MAC và quản lý MAC, nó có giao diện với điểm truy cập dịch vụ của thực thể quản lý tầng MAC (MLMESAP). Dịch vụ dữ liệu MAC có nhiệm vụ quản lý việc thu phát của khối MPDU (giao thức dữ liệu MAC) thông qua dịch vụ dữ liệu PHY.

Nhiệm vụ của tầng MAC là quản lý việc phát thông tin báo hiệu *beacon*, định dạng khung tin để truyền đi trong mạng, điều khiển truy nhập kênh, quản lý khe thời gian GTS, điều khiển kết nối và giải phóng kết nối, phát khung Ack**.**

### *2.3.1* Cấu trúc siêu khung.

LR-WPAN cho phép sử dụng theo nhu cầu cấu trúc siêu khung. Định dạng của siêu khung được định rõ bởi PAN coordinator. Mỗi siêu khung được giới hạn bởi từng mạng và được chia thành 16 khe như nhau. Cột mốc báo hiệu dò đường *beacon* được gửi đi trong khe đầu tiên của mỗi siêu khung. Nếu một PAN coordinator không muốn sử dụng siêu khung thì nó phải dừng việc phát mốc *beacon*. Mốc này có nhiệm đồng bộ các thiết bị đính kèm, nhận dạng PAN và chứa nội dung mô tả cấu trúc của siêu khung.




*Hinh2.6 Cấu trúc siêu khung*

Siêu khung có 2 phần "hoạt động" và "nghỉ". Trong trạng thái "nghỉ" thì PAN coordinator không giao tiếp với các thiết bị trong mạng PAN, và làm việc ở mode công suất thấp. Phần "hoạt động" gồm 2 giai đoạn: giai đoạn tranh chấp truy cập (CAP) và giai đoạn tranh chấp tự do(CFP), giai đoạn tranh chấp trong mạng chính là khoảng thời gian tranh chấp giữa các trạm để có cơ hội dùng một kênh truyền hoặc tài nguyên trên mạng). Bất kỳ thiết bị nào muốn liên lạc trong thời gian CAP đều phải cạnh tranh với các thiết bị khác bằng cách sử dụng kỹ thuật CSMA-CA. Ngược lại CFD gồm có các GTSs, các khe thời gian GTS này thường xuất hiện ở cuối của siêu khung tích cực mà siêu khung này được bắt đầu ở khe sát ngay sau CAP. PAN cooridinator có thể định vị được bảy trong số các GTSs, và mỗi một GTS chiếm nhiều hơn một khe thời gian.

Khoảng thời gian tồn tại của các phần khác nhau của siêu khung được định nghĩa bởi giá trị của *macBeaconOrder và macSuperFrameOrder. macBeaconOrder* mô tả khoảng thời gian mà bộ điều phối coordinator truyền khung báo hiệu tìm đường. Khoảng thời gian giữa hai mốc *beacon* BI(beacon interval) có quan hệ tới *macBeaconOrder (BO)* theo biểu thức sau: BI = aBaseSuperFrameDuration* $2^{BO}$ symbol, với $0 \le BO \le 14$. Lưu ý rằng siêu khung được bỏ qua nếu BO=15.

Giá trị của *macSuperFrameOrder* cho biết độ dài của phần tích cực của siêu khung. Khoảng thời gian siêu khung_SD (superframe duration) có quan hệ *macSuperFrameOrder*_ SO theo biểu thức sau: SD = aBaseSuperFrameDuration* $2^{SO}$ symbol. Nếu SO=15 thì siêu khung vẫn có thể ở phần "nghỉ" sau mốc *beacon* của khung.

Phần tích cực của mỗi siêu khung được chia thành 3 phần CAP,CFP và *beacon*. Mốc *beacon* được phát vào đầu ở khe số 0 mà không cần sử dụng CSMA.

#### *2.3.1.1*Khung CAP

CAP được phát ngay sau mốc *beacon* và kết thúc trước khi phát CFP. Nếu độ dài của phần CFP = 0 thì CAP sẽ kết thúc tại cuối của siêu khung. CAP sẽ có tối thiểu *aMinCAPLength symbols* trừ trường hợp phần không gian thêm vào được dùng để điều chỉnh việc tăng độ dài của khung *beacon* để vẫn có thể duy trì được GTS và điều chỉnh linh động tăng hay giảm kích thước của CFP.

Tất cả các khung tin ngoại trừ khung Ack và các khung dữ liệu phát ngay sau khung Ack trong lệnh yêu cầu, mà chúng được phát trong CAP sẽ sử dụng thuật toán CSMA-CA để truy nhập kênh. Một thiết bị phát trong khoảng thời gian phần CAP kết thúc sẽ khoảng thời gian IFS trước khi hết phần CAP. Nếu không thể kết thúc được thì thiết bị này sẽ trì hoãn việc phát cho đến khi CAP của khung tiếp theo được phát. Khung chứa lệnh điều khiển MAC sẽ được phát trong phần CAP.

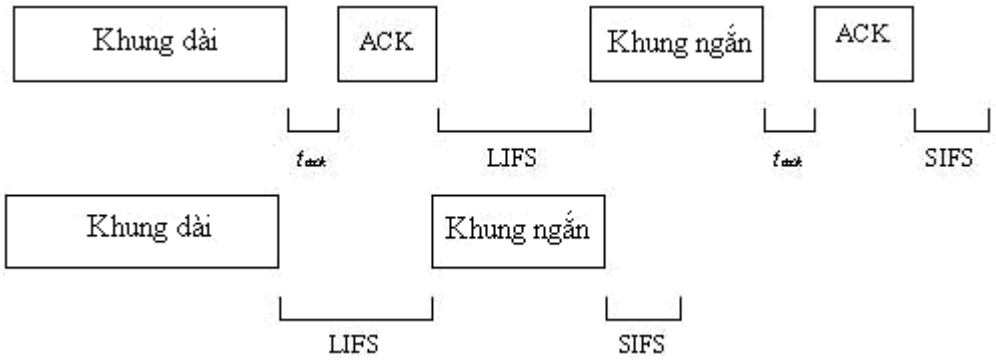
#### *2.3.1.2*Khung CFP

Phần CFP sẽ được phát ngay sau phần CAP và kết thúc trước khi phát *beacon* của khung kế tiếp. Nếu bất kỳ một GTSs nào được cấp phát bởi bộ điều phối mạng PAN , chúng sẽ được đặt bên trong phần CFP và lấp đầy một loạt các khe liền nhau. Bởi vậy nên kích thước của phần CFP sẽ do tổng độ dài các khe GTSs này quyết định. CFP không sử dụng thuật toán CSMA-CA để truy nhập kênh. Một thiết bị phát trong CFP sẽ kết thúc trong khoảng một IFS trước khi kết thúc GTS.

#### *2.3.1.3*Khoảng cách giữa hai khung (IFS)

Khoảng thời gian IFS là thời gian cần thiết để tầng PHY xử lý một gói tin nhận được. Khung tin được truyền theo chù kỳ IFS, trong đó độ dài của chu kỳ IFS phụ thuộc vào kích thước của khung vừa được truyền đi. Khung có độ dài phụ thuộc vào

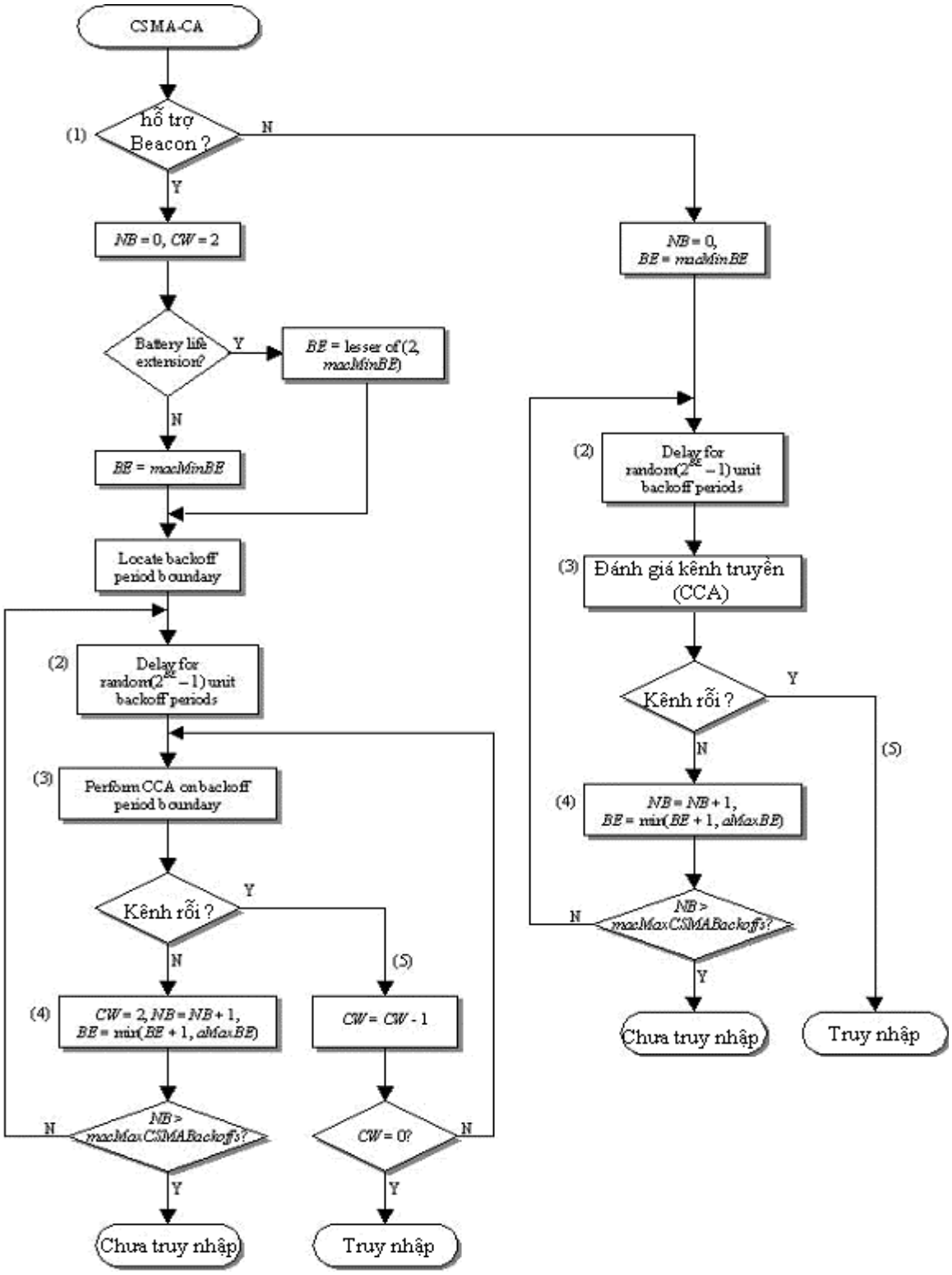
*aMaxSIFSFrameSize* sẽ tuân theo chu kỳ SIFS (*là khoảng thời gian tối thiểu aMinSIFSPeriod symbols*), và các khung có độ dài lớn hơn *aMaxSIFSFrameSize* sẽ tuân theo chu kỳ LIFS(*là khoảng thời gian tối thiểu aMinLIFSPeriod* symbols).




*Hình2.7*

### *2.3.2* Thuật toán tránh xung đột đa truy cập sử dụng cảm biến sóng mang CSMA-CA.

CSMA/CA (Carrier Sense Multiple Access-Collision Avoidance). Phương pháp tránh xung đột đa truy cập nhờ vào cảm biến sóng. Thực chất đây là phương pháp truy cập mạng dùng cho chuẩn mạng không dây IEEE 802.15.4. Các thiết bị trong mạng (các nốt mạng) sẽ liên tục lắng nghe tín hiệu thông báo trước khi truyền. Đa truy cập (multiple access) chỉ ra rằng nhiều thiết bị có thể cùng kết nối và chia sẻ tài nguyên của một mạng (ở đây là mạng không dây). Tất cả các thiết bi đều có quyền truy cập như nhau khi đường truyền rỗi. Ngay cả khi thiết bị tìm cách nhận biết mạng đang sử dụng hay không, vẫn có khả năng là có hai trạm tìm cách truy cập mạng đồng thời. Trên các mạng lớn, thời gian truyền từ đầu cáp nầy đến đầu kia là đủ để một trạm có thể truy cập đến cáp đó ngay cả khi có một trạm khác vừa truy cập đến. Nó tránh xung đột bằng cách là mỗi nốt sẽ phát tín hiệu về yêu cầu truyền trước rồi mới truyền thật sự.


CSMA-CA
(1) hỗ trợ Beacon ?
N
Y
NB = 0, CW = 2
Battery life extension?
Y
BE = lesser of (2, macMinBE)
N
BE = macMinBE
Locate backoff period boundary
(2) Delay for random(2^BE – 1) unit backoff periods
(3) Perform CCA on backoff period boundary
Kênh rỗi ?
Y
N
(5)
(4) CW = 2, NB = NB + 1, BE = min(BE + 1, aMaxBE)
CW = CW - 1
NB > macMaxCSMABackoffs?
N
Y
CW = 0?
N
Y
Chưa truy nhập
Truy nhập
NB = 0, BE = macMinBE
(2) Delay for random(2^BE – 1) unit backoff periods
(3) Đánh giá kênh truyền (CCA)
Kênh rỗi ?
Y
N
(5)
(4) NB = NB + 1, BE = min(BE + 1, aMaxBE)
NB > macMaxCSMABackoffs?
N
Y
Chưa truy nhập
Truy nhập

*Hình2.8 Lưu đồ thuật toán*

Thuật toán truy nhập kênh CSMA-CA được sử dụng trước khi phát dữ liệu hoặc trước khi phát khung tin MAC trong phần CAP. Thuật toán này sẽ không sử dụng để phát khung tin thông báo beacon, khung tin Ack, hoặc là khung tin dữ liệu trong phần CFP. Nếu bản tin báo hiệu được sử dụng trong mạng PAN thì thuật toán CSMA-CA gán khe thời gian được dùng, ngược lại thuật toán CSMA-CA không gán khe thời gian sẽ được sử dụng. Tuy nhiên trong cả hai trường hợp thuật toán đều được bổ xung bằng cách sử dụng khối thời gian *backoff* bằng với thời gian của tham số *aUnitBackoffPeriod.* Trong thuật toán truy nhập kênh CSMA-CA gán khe thời gian, biên của khoảng thời gian backoff của mỗi thiết bị trong mạng PAN được sắp thẳng hàng với biên của khe siêu khung của thiết bị điều phối mạng PAN. Trong thuật tóan này, mỗi lần thiết bị muốn truyền dữ liệu trong CAP thì nó phải xác định biên thời gian backoff kế tiếp. Trong thuật toán CSMA-CA không gán khe thời gian thì khoảng thời gian backoff của một thiết bị trong mạng không cần phải đồng bộ với khoảng thời gian backoff của thiết bị khác.

Mỗi thiết bị chứa 3 biến số:NB, BW, BE. Trong đó NB là số lần mà thuật toán này bị yêu cầu rút lại trong khi đang cố gắng truyền. Giá trị ban đầu của nó là 0 trước khi truyền. Biến CW là độ dài cửa sổ tranh chấp, nó cho biết khoảng thời gian cần thiết để làm sạch kênh truyền trước khi phát, giá trị ban đầu của nó là 2 trước khi cố gắng phát và quay trở lại 2 khi kênh truy nhập bị bận. Biến số CW chỉ sử dụng cho thuật toán gán khe thời gian CSMA-CA. Biến số BE (backoff_exponent) cho biết một thiết bị phải chờ bao lâu để có thể truy nhập vào một kênh. Cho dù bộ thu của thiết bị làm việc trong suốt khoảng thời gian CAP của thuật tóan nhưng nó vẫn bỏ qua bất kỳ khung tin nào nhận đươc trong khoảng thời gian này.

Trong thuật toán CSMA-CA gán khe thời gian, NB, CW, BE được thiết lập trước, biên của khoảng thời gian backoff kế tiếp cũng được xác định trước. Trong thuật toán CSMA-CA không gán khe thời gian thì NB và BE được thiết lập trước (*bước 1*). Tầng MAC sẽ trễ ngẫu nhiên trong phạm vi 0 đến 2*BE -1(bước2) sau đó yêu cầu tầng PHY thực hiện đánh giá truy kênh truy nhập xem là rỗi hay bận.(bước3). Nếu kênh truyền bận(bước4), tầng MAC sẽ tăng NB và BE lên 1, nhưng cũng luôn đảm bảo rằng giá trị này nhỏ hơn *aMaxBE*. Trong CSMA-CA gán khe thời gian thì việc truyền khung tin, Ack phải được thực hiện trước khi kết thúc phần CAP trong siêu khung, nếu không

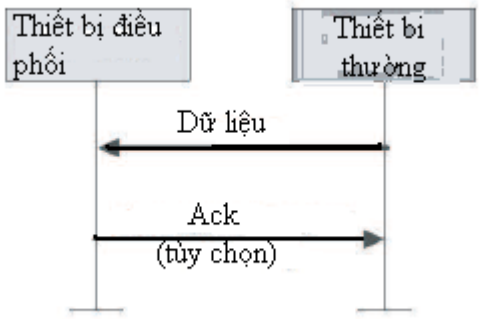
sẽ phải chờ đến CAP của siêu khung kế tiếp, trong thuật toán này thì CW có thể cũng reset lại thành giá trị 2. Nếu giá trị của NB nhỏ hơn hoặc bằng giá trị tham số *macMaxCSMABackoffs*, thì sẽ quay lại bước2 đồng thời thông báo lỗi truy nhập kênh.

Nếu kênh truyền là rỗi (bước5) , trong CSMA-CA gán khe thời gian, tầng MAC phải giảm CW đi 1. nếu CW ≠ 0 quay trở lại bước 3. Nếu CW=0 thì thôgn báo truy nhập kênh thành công. Còn trong CSMA-CA không gán khe thời gian thì tầng MAC bắt đầu phát ngay nếu kênh truyền rỗi.

### *2.3.3* Các mô hình truyền dữ liệu.

Dựa trên cấu trúc mạng WPAN thì ta có thể phân ra làm ba kiểu, ba mô hình truyền dữ liệu: từ thiết bị điều phối mạng PAN coordinator tới thiết bị thường, từ thiết bị thường tới thiết bị điều phối mạng PAN coordinator, và giữa các thiết bị cùng loại. Nhưng nhìn chung thì mỗi cơ chế truyền đều phụ thuộc vào việc là kiểu mạng đó có hỗ trợ việc phát thông tin thông báo beacon hay không.

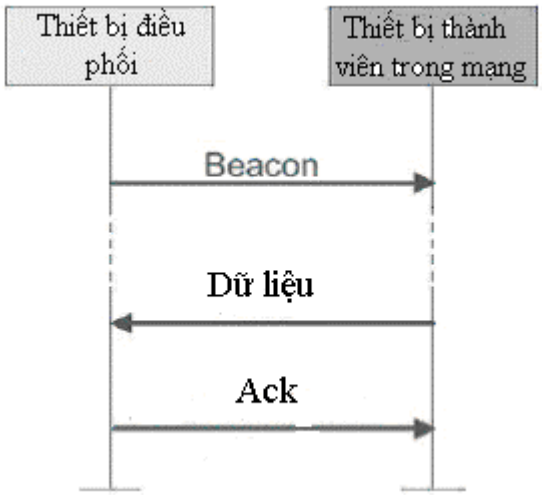
Khi một thiết bị muốn truyền dữ liệu trong một mạng không hỗ trợ việc phát beacon, khi đó thì nó chỉ đơn giản là truyền khung dữ liệu tới thiết bị điều phối bằng cách sử dụng thuật toán không gán khe thời gian. Thiết bị điều phối Coordinator trả lời bằng khung Ack như *hình2.9*




*Hình 2.9Liên lạc trong mạng không hỗ trợ beacon*

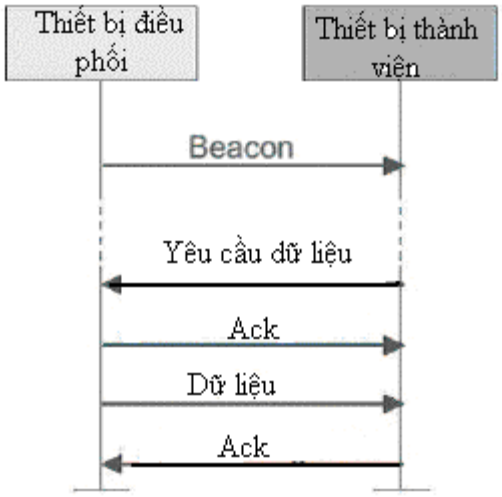
Khi một thiết bị muốn truyền dữ liệu tới thiết bị điều phối trong mạng có hỗ trợ beacon. Lúc đầu nó sẽ chờ báo hiệu beacon của mạng. Khi thiết bị nhận được báo hiệu beacon, nó sẽ sử dụng tín hiệu này để đồng bộ các siêu khung. Đồng thời, nó cũng phát

dữ liệu sử dụng phương pháp CSMA-CA gán khe thời gian và kết thúc quá trình truyền tin bằng khung tin xác nhận Ack.



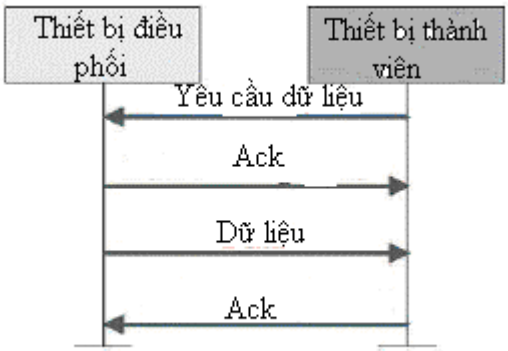

*Hình 2.10 liên lạc trong mạng có hỗ trợ beacon.*

Các ứng dụng truyền dữ liệu được điều khiển hoàn toàn bởi các thiết bị trong mạng PAN hơn là được điều khiển bởi thiết bị điều phối mạng. Chính khả năng này cung cấp tính năng bảo toàn năng lượng trong mạng ZigBee. Khi thiết bị điều phối muốn truyền dữ liệu đến một thiết bị khác trong loại mạng có hỗ trợ phát beacon, khi đó nó sẽ chỉ thị trong thông tin báo hiệu beacon là đang truyền dữ liệu. Các thiết bị trong mạng luôn luôn lắng nghe các thông tin báo hiệu beacon một cách định kỳ, khi phát hiện ra có dữ liệu liên quan tới nó đang được truyền, nó sẽ phát lệnh yêu cầu dữ liệu này, công việc này sử dụng slotted CSMA-CA. Công việc này được mô tả bằng *hình2.11* , trong hình này thì khung tin Ack của thiết bị điều phối cho biết rằng gói tin đã được truyền thành công, việc truyền gói tin sử dụng kỹ thuật gán khe thời gian CSMA-CA, khung Ack thiết bị thường trả lời là nhận gói tin thành công. Vào lúc nhận khung tin Ack từ thiết bị thường thì bản tin sẽ được xóa khỏi danh sách bản tin trong thông tin báo hiệu beacon.




*Hình2.11Kết nối trong mạng hỗ trợ beacon*

Trong trường hợp mạng không hỗ trợ phát beacon (*hình2.8*)thiết bị điều phối muốn truyền dữ liệu tới các thiết bị khác, nó sẽ phải lưu trữ dữ liệu để cho thiết bị liên quan có thể yêu cầu và tiếp xúc với dữ liệu đó. Thiết bị có thể tiếp xúc được với dữ liệu liên quan đến nó bằng cách phát đi lệnh yêu cầu dữ liệu tới thiết bị điều phối, sử dụng thuật toán không gán khe thời CSMA-CA. Nếu dữ liệu đang được truyền, thì thiết bị điều phối sẽ phát khung tin bằng cách sử dụng thuật toán không gán khe thời gian CSMA-CA, nếu dữ liệu không được truyền thì thiết bị điều phối sẽ phát đi khung tin không có nội dung để chỉ ra rằng dữ liệu không được phát.




*Hình 2.12 kết nối trong mạng không hỗ trợ phát beacon*

Nói chung trong mạng mắt lưới, tất cả các thiết bị đều bình đẳng và có khả năng kết nối đến bất kỳ thiết bị nào trong mạng miễn là thiết bị đó nằm trong bán kính phủ

sóng của nó. Có hai cách để thực hiện việc kết nối. Cách thứ nhất là nốt trong mạng liên tục lắng nghe và phát dữ liệu của nó đi bằng cách sử dụng thuật tóan không gán khe thời gian CSMA-CA. Cách thứ hai là các nốt tự đồng bộ với các nốt khác để có thể tiết kiệm đươc năng lượng.

### *2.3.4* Phát thông tin báo hiệu beacon

Một thiết bị FFD hoạt động trong chế độ không phát thông tin báo hiệu hoặc có thể phát thông tin báo hiệu giống như là thiết bị điều phối mạng. Một thiết bị FFD không phải là thiết bị điều phối mạng PAN có thể bắt đầu phát thông tin báo hiệu beacon chỉ khi nó kết nối với thiết bị điều phối PAN. Các tham số *macBeaconOrder* và *macSuperFrameOrder* cho biết khoảng thời gian giữa hai thông tin báo hiệu và khoảng thời gian của phần hoạt động và phần nghỉ. Thời gian phát bào hiệu liền trước được ghi lại trong tham số *macBeaconTxTime* và được tính toán để giá trị của tham số này giống như giá trị trong khung thông tin báo hiệu beacon.

### *2.3.5* Quản lý và phân phối khe thời gian đảm bảo GTS.

Khe thời gian đảm bảo GTS cho phép một thiết bị có thể hoạt động trong một kênh truyền bên trong một phần của siêu khung dành riêng cho thiết bị đó. Một thiết bị chỉ có thể chiếm và sử dụng một khe thời gian khi mà thiết bị đó liên quan đến thông tin báo hiệu beacon hiện thời lúc đó. Thiết bị điều phối mạng PAN có thể chiếm hữu khe thời gian GTS và sử dụng khe thời gian này để liên lạc với các thiết bị khác trong mạng. Một khe thời gian đơn có thể kéo dài hơn thời gian của siêu khung. Thiết bị điều phối mạng PAN có thể chiếm hữu tới bảy khe thời gian GTS cùng một lúc miễn là nó có đủ thẩm quyền trong siêu khung.

Một khe thời gian có thể được chiếm hữu trước khi sử dụng nếu có sự yêu cầu của thiết bị điều phối mạng PAN. Tất cả các khe thời gian GTS đều được đặt liền nhau ở cuối của siêu khung sau phần CAP, và hoạt động theo cơ chế FCFS (*first-come-first-serve*) đến trước dùng trước. Mỗi khe thời gian GTS có thể được giải phóng nếu không có yêu cầu nào, và một khe thời gian GTS có thể được giải phóng vào bất kỳ lúc nào khi thiết bị chiếm hữu nó không dùng nữa.

Chỉ duy nhất thiết bị điều phối PAN mới có quyền quản ly khe thời gian. Để quản ly mỗi khe thời gian đảm bảo, thiết bị điều phối có thể lưu trữ khe bắt đầu, độ dài, phương hướng (thu hay phát) và địa chỉ thiết bị kết nối.

Mỗi thiết bị trong mạng có thể yêu cầu một khe thời gian phát hay một khe thời gian thu. Để chiếm hữu được một khe thời gian thì thiết bị đó phải lưu trữ thông tin khe bắt đầu, độ dài và phương hướng. Nếu một thiết bị đươc cấp phát một khe thời gian GTS thu, nó sẽ có toàn quyền sử dụng trọn vẹn khe thời gian đó để nhận dữ liệu. Tương tự như vậy thiết bị điều phối mạng PAN cũng có thể có toàn quyền sử dụng trọn vẹn khe thời gian đó để nhận đữ liệu khi có một thiết bị khác chiếm khe thời gian phát.

Một thiết bị yêu cầu chiếm hữu khe thời gian mới thông qua lệnh yêu cầu GTS với các tính chất (độ dài, thu hay phát?,…) thiết lập theo yêu cầu ứng dụng. Để xác nhận lệnh này thì thiết bị điều phối sẽ gửi một khung tin Ack. Sau khi phát khung tin Ack thì thiết bị điều phối sẽ kiểm tra khả năng hiện thời của siêu khung dựa trên độ dài của phần CAP và độ dài khe thời gian GTS được yêu cầu. Siêu khung sẽ sẵn sàng nếu độ dài khe thời gian GTS không làm giảm độ dài của phần CAP đi quá độ dài nhỏ nhất của CAP được qui đinh trong tham số *aMinCAPLength*. Thiết bị điều phối mạng PAN thực hiện quyết định của nó bên trong siêu khung *aGTSDescPersistenceTime.* Trong khi xác nhận gói tin Ack từ thiết bị điều phối thì thiết bị này vẫn tiếp tục theo dõi thông tin báo hiệu và chờ siêu khung *aGTSDescPersistenceTime.* Khi thiết bị điều phối quyết định xem xem nó có sẵn sàng cho yêu cầu GTS không, nó sẽ phát đi mô tả về GTS với chi tiết yêu cầu và đoạn ngắn địa chỉ của thiết bị yêu cầu. Nó sẽ chỉ ra độ dài và khe GTS đầu tiên trong siêu khung rồi thông báo cho tầng trên về việc cấp phát khe GTS mới này. Nếu sau khi kiểm tra mà thấy khả năng của siêu khung là không đủ để cấp phát theo yêu cầu về GTS, thì khe đầu tiên sẽ được đánh số 0 tới độ dài khe GTS lớn nhất có thể cung cấp được hiện thời. Những mô tả về GTS sẽ đươc giữ trong khung tin báo hiệu beacon cho *aGTSPersistenceTime.* Trong khi xác nhận khung tin báo hiệu beacon, thiết bị sẽ xử ly và thông báo lên tầng trên.

Tượng tự như khi yêu cầu cấp phát GTS, một thiết bị cho biết nó yêu cầu được giải phóng sự chiếm hữu GTS thông qua lệnh yêu cầu giải phóng với các thông số của GTS đang tồn tại. Sau đó thì khe thời gian này sẽ được tự do. Thiết bị điều phối PAN phải đảm bảo rằng không có khoảng trống náo xuất hiện trong CFP khi giải phóng khe

thời gian GTS, độ dài maximum CAP nhờ thế mà được tăng lên (độ tăng đúng bằng độ dài của khe thời gian đươc giải phóng).

Thực thể quản lý tầng MAC (MLME) của thiết bị điều phối mạng PAN có nhiệm vụ phát hiện khi một thiết bị dừng sử dụng khe thời gian GTS. Công việc đó thực hiện bằng nguyên tắc sau. Đối với khe GTS phát, MLME sẽ công nhận một khe thời gian GTS được giải phóng nếu khung dữ liệu không được nhận trong tối thiểu 2*n siêu khung. Đối với khe GTS thu, MLME sẽ công nhận thiết bị không còn sử dụng GTS nữa nếu khung tin xác nhận Ack không được nhận trong tối thiểu 2*n siêu khung.

$n = 2^{8-macBeaconOrder}$ , nếu $0 \le macBeaconOrder \le 8$;

$n = 1$ , nếu $9 \le macBeaconOrder \le 14$;

### *2.3.6* Định dạng khung tin MAC.

Mỗi khung bao gồm các thành phần sau:

- Đầu khung MHR(MAC header): gồm các trường thông tin về điều khiển khung tin, số chuỗi, và trường địa chỉ
- Tải trọng khung (MAC payload) : chứa các thông tin chi tiết về kiểu khung. Khung tin của bản tin xác nhận Ack không có phần này.
- Cuối khung MFR(MAC footer) chứa chuỗi kiểm tra khung FCS (*frame check sequence)*

<table>
<tr><td>Octets: 2</td><td>1</td><td>0/2</td><td>0/2/8</td><td>0/2</td><td>0/2/8</td><td>biến thiên</td><td>2</td></tr>
<tr><td rowspan="2">Điều khiển khung</td><td rowspan="2">Số chuỗi</td><td>ID mạng PAN đích</td><td>Địa chỉ đích</td><td>ID PAN nguồn</td><td>Địa chỉ nguồn</td><td rowspan="2">Tải trọng khung</td><td rowspan="2">Chuỗi kiểm tra khung (FCS)</td></tr>
<tr><td colspan="4">Trường địa chỉ</td></tr>
<tr><td colspan="6">Phần đầu khung MHR</td><td>Tải trọng</td><td>Cuối khungMFR</td></tr>
</table>

*Bảng2.6 Định dạng khung MAC*

## *2.4* Tầng mạng của ZigBee/IEEE802.15.4

### *2.4.1* Dịch vụ mạng

Tầng vật lý trong mô hình của giao thức ZigBee được xây dựng trên nền của tầng điều khiển dữ liệu, nhờ những đặc điểm của tầng MAC mà tầng vật lý có thể kéo dài việc đưa tin, có thể mở rộng được qui mô mạng dễ dàng, một mạng có thể hoạt động cùng các mạng khác hoặc riêng biệt. Tầng vật lý phải đảm nhận các chức năng như là:

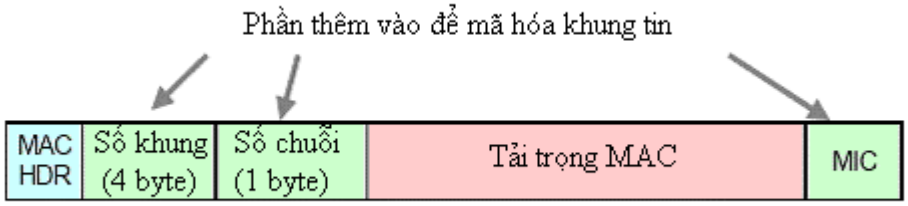
- Thiết lập một mạng mới.
- Tham gia làm thành viên của một mạng đang hoạt hoặc là tách ra khỏi mạng khi đang là thành viên của một mạng nào đó.
- Cấu hình thiết bị mới như hệ thống yêu cầu, gán địa chỉ cho thiết bị mới tham gia vào mạng.
- Đồng bộ hóa các thiết bị trong mạng để có thể truyền tin mà không bị tranh chấp, nó thực hiện đồng bộ hóa này bằng gói tin thông báo beacon.
- Bảo mật: gán các thông tin bảo mật vào gói tin và gửi xuống tầng dưới
- Định tuyến, giúp gói tin có thể đến được đúng đích mong muốn. Có thể nói rằng thuật toán của ZigBee là thuật toán định tuyến phân cấp sử dụng bảng định tuyến phân cấp tối ưu được áp dụng từng trường hợp thích hợp.

### *2.4.2* Dịch vụ bảo mật

Khi khung tin tầng MAC cần được bảo mật, thì ZigBee sử dụng dịch vụ bảo mật của tầng MAC để bảo vệ các khung lệnh MAC, các thông tin báo hiệu beacon, và các khung tin xác nhận Ack. Đối với các bản tin chỉ phải chuyển qua một bước nhảy đơn, tức là truyền trực tiếp từ nốt mạng này đến nốt mạng lân cận của nó, thì ZigBee chỉ cần sử dụng khung tin bảo mật MAC để mã hóa bảo vệ thông tin. Nhưng đối với các bản tin phải chuyển gián tiếp qua nhiều nốt mạng mới tới được đích thì nó cần phải nhờ vào tầng mạng để làm công việc bảo mật này. Tầng điều khiển dữ liệu MAC sử dụng thuật tóan AES (chuẩn mã hóa cao cấp). Nói chung thì tầng MAC là một quá trình mã hóa, nhưng công việc thiết lập các khóa key, chỉ ra mức độ bảo mật, và điều khiển quá trình mã hóa thì lại thuộc về các tầng trên. Khi tầng MAC phát hoặc nhận một khung tin nào đó được bảo mật, đầu tiên nó sẽ kiểm tra địa chỉ đích hoặc nguồn của khung tin đó, tìm

ra cái khóa kết hợp với địa chỉ đích hoặc địa chỉ nguồn, sau đó sử dụng cái khóa này để xử lý khung tin theo qui trình bảo mật mà cái khóa đó qui định. Mỗi khóa key được kết hợp với một qui trình bảo mật đơn lẻ. Ở đầu mỗi khung tin của MAC luôn có 1 bit để chỉ rõ khung tin này có được bảo mật hay không.

Khi phát một khung tin, mà khung tin này yêu cầu cần được bảo toàn nguyên vẹn. Khi đó phần đầu khung và phần tải trọng khung MAC sẽ tính tóan cân nhắc để tạo ra một trường mã hóa tin nguyên vẹn (MIC- Message Integrity) phù hợp, MIC gồm khoảng 4,8 hoặc 16 octets. MIC sẽ được gán thêm vào bên phải phần tải trọng của MAC.

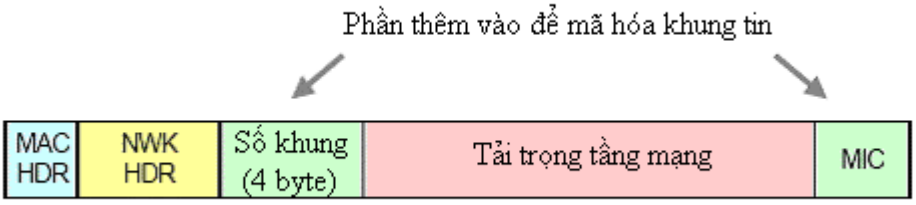



*Hình2.13Khung tin mã hóa tầng MAC*

Khi khung tin phát đi đòi hỏi phải có độ tin cậy cao, thì biện pháp được sử dụng để mã hóa thông tin là số chuỗi và số khung sẽ được gán thêm vào bên trái phần tải trọng khung tin MAC. Trong khi nhận gói tin, nếu phát hiện thấy MIC thì lập tức nó sẽ kiểm tra xem khung tin nào bị mã hóa để giải mã. Cứ mỗi khi có một bản tin gửi đi thì thiết bị phát sẽ tăng số đếm khung lên và thiết bị nhận sẽ theo dõi căn cứ vào số này. Nhờ vậy nếu như có một bản tin nào có số đếm khung tin đã bị nhận dạng một lần thì thiết bị nhận sẽ bật cờ báo lỗi bảo mật. Bộ mã hóa của tầng MAC dựa trên ba trạng thái của hệ thống.

- Để bảo đảm tính nguyên vẹn: Mã hóa sử dụng AES với bộ đếm CTR
- Để bảo đảm tính tinh cậy : Mã hóa sử dụng AES với chuỗi khối mã CBC-MAC
- Để đảm bảo tính tin cậy cũng như nguyên vẹn của bản tin thì kết hợp cả hai trạng thái CTR và CBC-MAC trên thành trạng thái CCM.

Tầng mạng cũng sử dụng chuẩn mã hóa AES. Tuy nhiên khác với tầng điều khiển dữ liệu MAC, bộ mã hóa của tầng mạng làm việc dựa trên trạng thái CCM* của hệ thống. Trạng thái này thực chất là sự cải biên từ CCM của tầng MAC, nó thêm vào chuẩn mã hóa này các chức năng là *chỉ mã hóa tính tin cậy* và *chỉ mã hóa tính nguyên vẹn*. Sử dụng CCM* giúp làm đơn giản hóa quá trình mã hóa dữ liệu của tầng mạng, các chuỗi mã hóa này có thể dùng lại khóa key của chuỗi mã hóa khác. Như vậy thì khóa key này không hoàn toàn còn là ranh giới của các chuỗi mã hóa nữa. Khi tầng mạng phát hoặc nhận một gói tin được mã hóa theo qui ước bởi nhà cung cấp dịch vụ, nó sẽ kiểm tra địa chỉ nguồn hoặc đích của khung tin để tìm ra khóa key liên quan tới địa chỉ đó, sau đó sẽ áp dụng bộ mã hóa này giải mã hoặc mã hóa cho khung tin. Tương tự như quá trình mã hóa tầng MAC, việc điều khiển quá trình mã hóa này được thực hiện bởi các tầng cao hơn, các số đếm  khung và MIC cũng được thêm vào để mã hóa khung tin.




*Hình 2.14 Khung tin mã hóa tầng mạng*

## *2.5* Tầng ứng dụng của ZigBee/IEEE 802.15.4

Lớp ứng dụng của ZigBee/IEEE802.15.4 thực chất gồm các ba tầng như hình vẽ trên, các tầng này tương ứng với các tầng phiên, trình diễn và ứng dụng trong mô *hình2.1* OSI  7 tầng.

Trong ZigBee/IEEE 802.15.4 thì chức năng của tầng A*pplication Framework*  là:

- Dò tìm ra xem có nốt hoặc thiết bị nào khác đang hoạt động trong vùng phủ sóng của thiết bị đang hoạt động hay không.
- Duy trì kết nối, chuyển tiếp thông tin giữa các nốt mạng.

Chức năng của tầng Application Profiles là:

- Xác định vai trò của các thiết bị trong mạng. (thiết bị điều phối mạng, hay thiết bị đầu cuối, FFD hay RFD….)
- Thiết lập hoặc trả lời yêu cầu kết nối.
- Thành lập các mối quan hệ giữa các thiết bị mạng.

Chức năng của tầng Application là thực hiện các chức năng do nhà sản xuất qui định (giao diện…) để bổ sung thêm vào các chức năng do ZigBee qui định

# CHƯƠNG 3 CÁC THUẬT TOÁN ĐỊNH TUYẾN CỦA ZigBee/IEEE 802.15.4.

Trong ZigBee/ IEEE802.15.4 sử dụng thuật toán chọn đường có phân cấp nhờ xét các phương án tối ưu. Khởi điểm của thuật tóan định tuyến này chính là thuật toán miền công cộng đã được nghiên cứu rất kỹ có tên là AODV (Ad hoc On Demand Distance Vector) dùng cho những mạng có tính chất tự tổ chức và thuật toán hình cây của Motorola.

## *3.1* Thuật toán định tuyến theo yêu cầu AODV (Ad hoc On Demand Distance Vector)

AODV (Ad hoc On Demand Distance Vector) đơn thuần chỉ là thuật toán tìm đường theo yêu cầu trong mạng ad hoc (một mạng tự tổ chức). Có thể hiểu như sau, những nốt trong mạng khi mà không nằm trong tuyến đường truyền tin thì không duy trì thông tin nào về tuyến đường truyền và cũng không tham gia vào quá trình định tuyến theo chu kỳ. Nói kỹ hơn nữa, một nốt mạng không có chức năng tự định tuyến và lưu trữ tuyến đường tới một nốt mạng khác cho đến khi cả hai nốt mạng trên liên lạc với nhau, trừ trường hợp những nốt mạng cũ đề nghị dich vụ như là một trạm chuyển tiếp để giữ liên lạc giữa hai nốt mạng khác.

Mục đích đầu tiên của thuật toán là chỉ phát quảng bá các gói tin dò đường khi cần thiết hoặc khi có yêu cầu, việc làm này để phân biệt giữa việc quản lý liên lạc cục bộ với việc bảo quản giao thức liên lạc chung và để phát quảng bá thông tin về sự thay đổi trong liên kết cục bộ tới những nốt di động lân cận (là những nốt cần thông tin để cập nhật). Khi một nốt nguồn cần để kết nối tới nốt khác, mà nốt nguồn không chứa thông tin về thông tin tuyến đường tới nốt đó, như vậy một quá trình tìm đường được thiết lập.

Để thiết lập quá trình tìm đường này thì mỗi nốt mạng đều lưu hai bộ đếm độc lập: *sequence number* và *broadcast id*. Để bắt đầu quá trình tìm đường, nốt nguồn sẽ khởi tạo một gói tin tìm đường (RREQ) và phát quảng bá gói tin này tới tất cả các nốt mạng lân cận, gói tin RREQ này chứa các thông tin về địa chỉ nguồn *(source addr)*, số chuỗi nguồn(*source sequence number*), số id quảng bá *(broadcast id)*, địa chỉ đích (*dest addr)*, số chuỗi đích *(dest sequence number)*, số đếm bước truyền (*hop cnt)*.

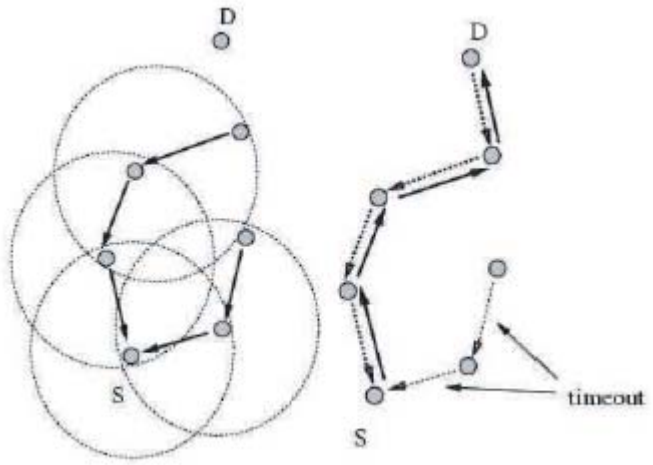
Bởi mỗi khi nốt mạng nguồn phát ra một gói tìn RREQ mới thì số id quảng bá sẽ tăng lên, nên trong mỗi gói tin RREQ thì cặp địa chỉ nguồn và số id quảng bá luôn luôn là duy nhất. Khi nốt mạng trung gian nhận được một gói tin RREQ mới, nó sẽ đem so sánh địa chỉ nguồn và số id quảng bá với gói tin RREQ trước đó, nếu giống nhau nốt mạng trung gian này sẽ tự động xóa RREQ dư thừa này và dừng việc phát gói tin này lại. Nhưng nếu so sánh thấy khác nhau thí nốt mạng này sẽ tự động tăng số đếm bước truyền (*hop cnt)* lện và tiếp tục phát quảng bá gói tin RREQ này tới các nốt lân cận để tiếp tục quá trình tìm đường. Trong mỗi một nốt mạng đều lưu trữ các thông tin về địa chỉ IP đích, địa chỉ IP nguồn, số id quảng bá, số chuỗi nốt nguồn, và thời gian thời gian hạn định cho phép gói tin mang thông tin xác nhận được gửi trả lại nơi phát

Khi gói tin RREQ được truyền trên mạng từ nguồn tới đích, nó sẽ tự động thiết lập con đường ngược lại từ các nốt mạng này quay trở lại nốt nguồn. Để thiết lập tuyến đường ngược chiều, mỗi nốt phải lưu giữ bảng địa chỉ của các nốt bên cạnh mà nó sao chép được trong gói tin RREQ đầu tiên. Tuyến đường ngược chiều được lưu giữ trong thời gian tối thiểu để gói tin RREQ này vượt qua mạng và trở về nơi xuất phát ban đầu.

Khi RREQ tới một nốt nào đấy mà có thể nốt mạng này là đích đến của nó, hoặc nốt này nằm trên tuyến đường truyền từ nguồn tới đích, nốt nhận tin này đầu tiên sẽ kiểm tra xem gói tin RREQ vừa nhận qua kết nối hai chiều. Nếu nốt mạng này chưa phải là nốt mạng đích nhưng có lưu giữ tuyến  đường tới nốt đích, khi đó nó sẽ quyết định xem xem tuyến đường này có chính xác không bằng cách so sánh số chuỗi nguồn chứa bên trong gói tin RREQ này với số chuỗi nguồn trong bảng định tuyến của nốt mạng đó. Nếu số chuỗi đích của RREQ lớn hơn số chuỗi đích trong các nốt trung gian, thì nốt trung gian đó không không nằm trên tuyến đương truyến ứng với gói tin RREQ này.

Nếu tuyến đường này có số chuỗi đích lớn hơn hoặc bằng với số chuỗi đích trong RREQ nhưng có số bước truyền nhỏ hơn, thì nó có thể phát một gói tin RREP *(route reply packet)* trở lại nốt mạng đã phát RREQ cho nó. Một gói tin RREP gồm có các trường thông tin sau: trường địa chỉ nguồn, trường địa chỉ đích, số chuỗi đích, số đếm bước truyền và thời gian sống. Khi mà gói tin RREP quay trở lại đực nốt nguồn, các nốt mạng dọc theo tuyến đường của RREP sẽ thiết lập con chỏ hướng tới nốt mạng RREP vừa đến, cập nhật thông tin timeout (timeout là khoảng thời gian mà một nốt không còn hoạt động nữa và nằm trong trạng thái chờ) của nó cho bảng định tuyến

đường tới nguồn và đích, đồng thời sao lưu lại số chuỗi đích cuối của nốt đích cần tới. Những nốt mạng nằm dọc theo tuyến đường xác định bởi RREP sẽ "chết" sau khi hết thời gian yêu cầu đinh tuyến và con chỏ đảo bị xóa khi chúng không còn nằm trên tuyến đường truyền từ nguồn tới đích. Thời gian "chết" này phụ thuộc vào kích cỡ của mạng**.**




*Hình 3.1 Định dạng tuyến đường trong giao thức AODV*

Nốt nguồn có thể phát dữ liệu ngay khi nó nhận được gói tin RREP đầu tiên, đồng thời cũng luôn cập nhật thông tin về tuyến đừong nếu phát hiện ra tuyến đường tối ưu hơn.

Mỗi bảng định tuyến gồm các trường thông tin sau: trường thông tin về đích đến, bước truyền kế tiếp, số bước truyền, số chuỗi đích, nút lân cận tích cực thuộc tu yến đường, thời gian chết cho nhập liệu vào bảng định tuyến.

Để duy trì đường truyền, mỗi nốt mạng luôn phải có địa chỉ của các nốt mạng tích cực lân cận (*một nốt mạng được coi là tích cực nếu nó có chức năng khởi phát hoặc chuyển tiếp tối thiểu một gói tin đến đích trong thời gian cho phép)*. Khi mà bước truyền kế tiếp nằm trong tuyến đường từ nguồn tới đích này không thực hiện đươc (tức là thông tin yêu cầu không được nhận trong một khoảng thời gian nào đó, thông tin yêu cầu này đảm bảo rằng chỉ có những nốt mạng nào liên lạc hai chiều mới được coi là nốt mạng lân cận). Quá trình này cứ tiếp diễn đến khi tất cả các nốt nguồn tích cực được thông báo. Nhờ vào việc nhận những thông báo về gián đoạn đường truyền, mà các nốt

nguồn có thể khởi động lại quá trình tìm đường nếu chúng vẫn cần một tuyến đường tới đích cũ. Nếu nốt nguồn lựa chọn việc xây dựng lại tuyến đường mới từ nguồn tới đích, nó cần phải phân phát một gói tin RREQ mới với sô chuỗi đích mới lớn hơn số chuỗi đích cũ.

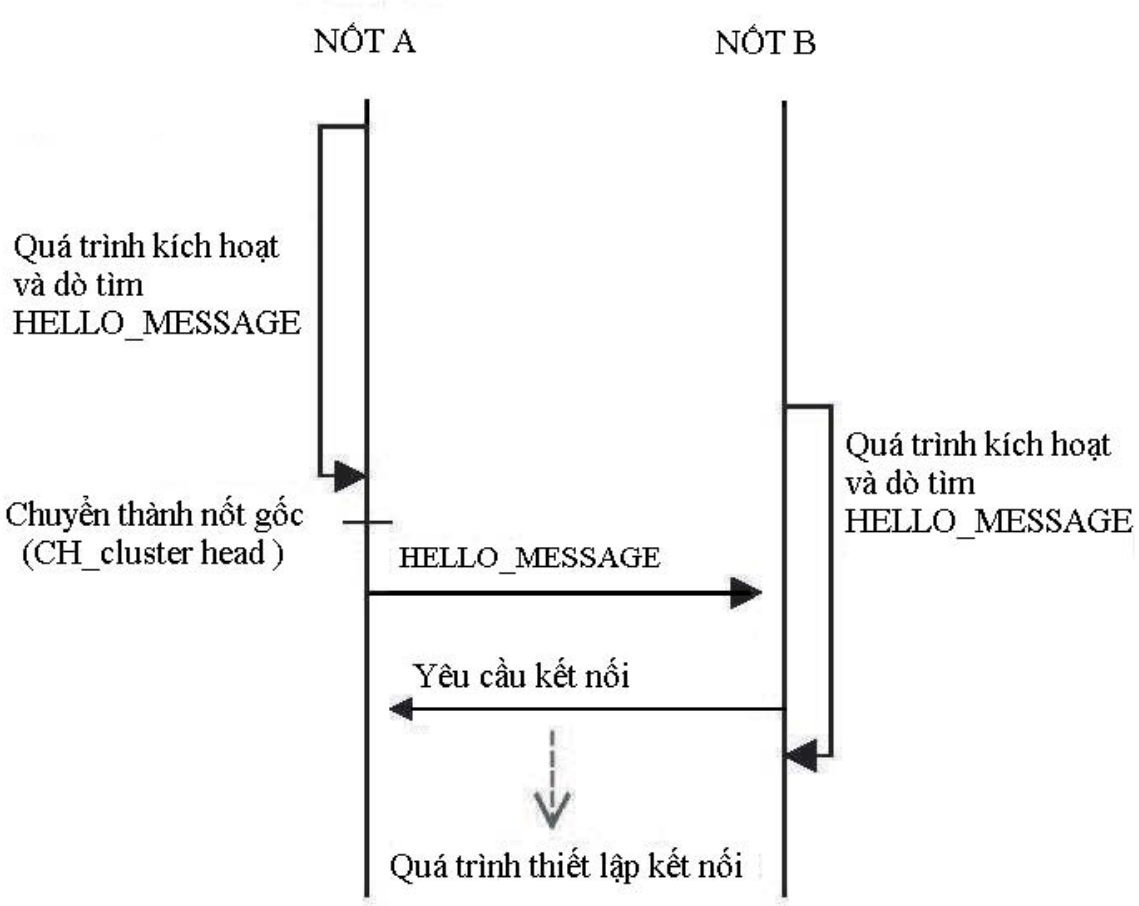
## *3.2* Thuật toán hình cây

Giao thức hình cây là giao thức của tầng mạng và tầng datalink, giao thức này sử dụng gói tin "trạng thái kết nối" để định dạng một mạng hình cây đơn, cũng như một mạng hình cây mở rộng. Loại mạng này cơ bản là một loại mạng có tính chất tự tổ chức và tự hỗ trợ để hạn chế lỗi mạng  một mức độ lỗi cho phép, đặc biệt hơn do đây là một loại mạng có tính chất tự tổ chức nên nó cũng có thể tự sửa chữa khi gặp sự cố ở một nốt mạng nào đó**.** Các nốt mạng chọn một nốt làm gốc cây và tạo các nhánh cây một cách tự do. Sau đó cách nhánh cây tự phát triển kết nối tới những nhánh cây khác nhờ vào thiết bị gốc (DD- Designated Deviece).

### *3.2.1* Thuật tóan hình cây đơn nhánh

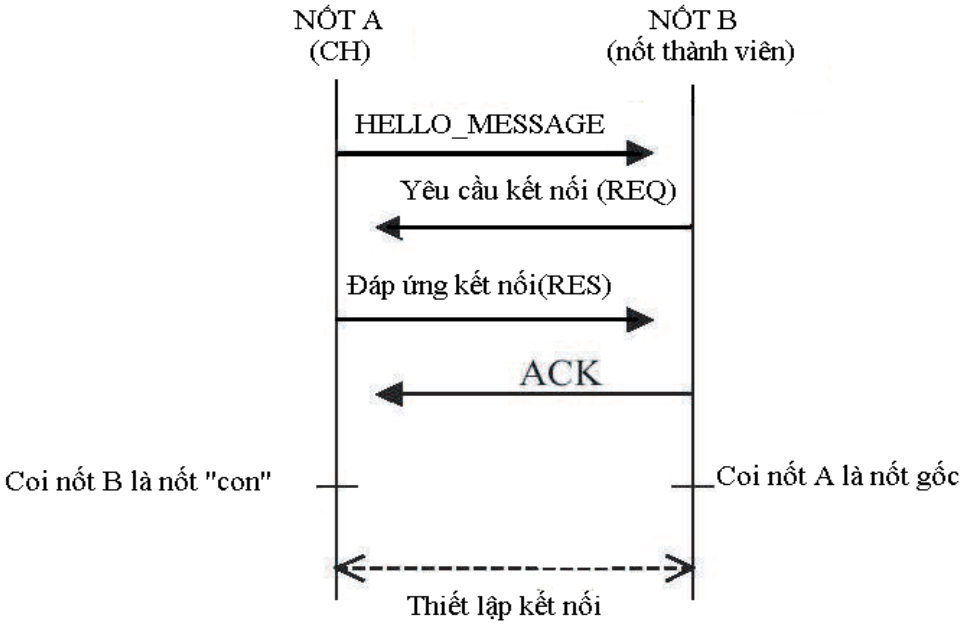
Quá trình hình thành nhánh cây bắt đầu bằng việc chọn gốc cây. Sau khi một nốt gốc được chọn, nó sẽ mở rộng kết nối với các nốt khác để tạo thành một nhóm.

Sau khi một nốt được kích hoạt nó sẽ dò tìm HELLO message từ các nốt khác (HELLO message tương tự như *beacon* trong tầng MAC theo chuẩn IEEE 802.15.4). Nếu trong một thời gian nhất định nào đó nó không nhận được bất kỳ một HELLO message nào, thì nốt này sẽ tự trở thành nốt gốc và lại gửi HELLO message tới các nốt lân cận. Nốt gốc mới này sẽ chờ gói tin yêu cầu kết nối từ các nốt lân cận trong một khoảng thời gian nào đó, nếu nó vẫn không nhận được bất kỳ yêu cầu kết nối nào từ các nốt lân cận thì nó sẽ trở lại thành một nốt bình thường và lại tiếp tục dò tìm HELLO_MESSAGE. Nốt gốc cũng có thể được chọn lựa dựa trên tham số của mỗi nốt mạng (ví dụ như phạm vi truyền, công suốt, vị  trí, khả năng tính toán ).



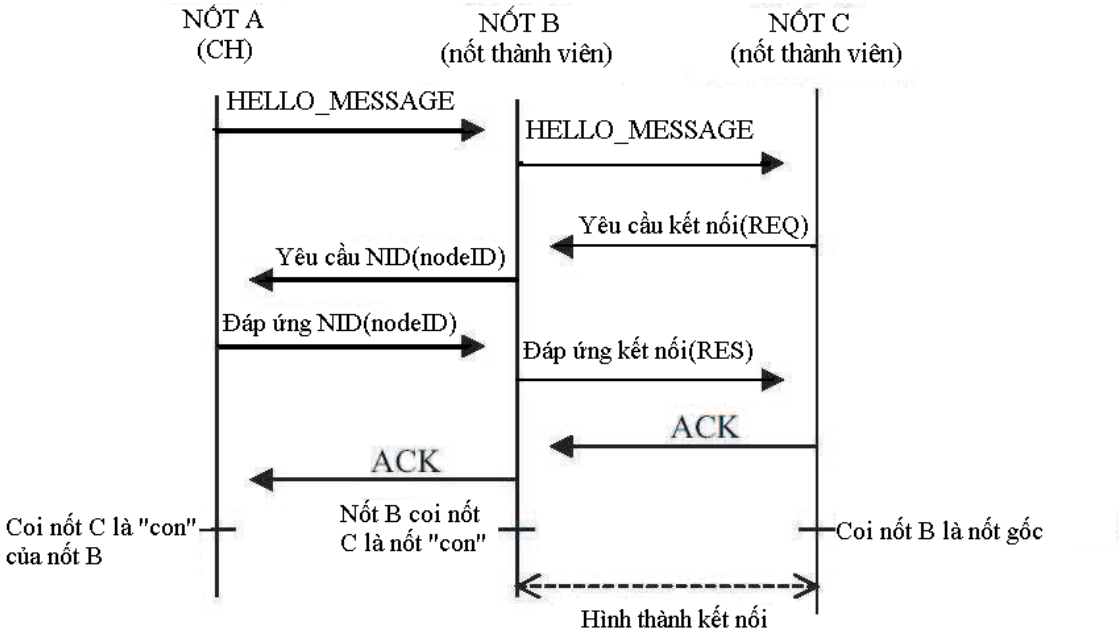

*Hình 3.2 Quá trình chọn nốt gốc (CH)*

Sau khi trở thành nốt gốc, nó sẽ phát quảng bá gói tin HELLO_MESSAGE theo chu kỳ, gói tin HELLO_MESSAGE này gồm một phần địa chỉ MAC và địa chỉ ID của nốt gốc. Những nốt mạng nhận được gói tin này sẽ gửi trả lời lại bằng gói tin yêu cầu kết nối (REQ) tới nốt gốc (nơi vừa phát đi). Khi nốt gốc nhận được gói tin yêu cầu kết nối, nó sẽ ngay lập tức gửi trả lại gói tin vừa đưa ra yêu cầu bằng một gói tin khác CONNECTION_RESPONSE., gói tin này chứa địa chỉ ID cho nốt thành viên (nốt B), địa chỉ ID này do nốt gốc qui định. Để xác nhận thông tin thì nốt thành viên B này sẽ gửi lại nốt gốc gói tin Ack. Quá trình trao đổi tin này đươc mô tả qua *hình3.3*




*Hình3.3 Thiết lập kết nối giữa CH và nốt thành viên*

Nếu tất cả các nốt đều ở trong phạm vi phủ song của nốt gốc thì kiến trúc mạng là kiến trúc hình sao, tất cả các nốt thành viên sẽ liên lạc trực tiếp với nốt gốc qua một bước truyền *(onehop)*. Một nhánh có thể phát triển thành cấu trúc mạng liên lạc qua nhiều bước truyền *(multihop)*.




*Hình 3.4 Quá trình hình thành nhánh nhiều bậc*

Tất nhiên nốt gốc chỉ có thể quản lý được một số hữu hạn các nốt, và các nhánh của mạng cũng chỉ có thể vươn tới những khoảng cách hạn chế… chính vị thế mà có lúc nốt mạng cũng cần phải từ chối kết nối của những nốt mới. Việc từ chối này được thực hiện nhờ vào việc chỉ định một ID đặc biệt cho nốt này. Bảng danh sách các nốt lân cận và tuyến đường luôn luôn được cập nhật mới thông qua gói tin HELLO_MESSAGE. Trong một thời gian nhất định, nếu vì một lý do nào đó mà một nốt không được cập nhật các thông tin trên thì nó sẽ bị loại bỏ.

Tất nhiên trong một mạng có tính chất tự do, tự tổ chức như loại mạng này thì không thể tránh khỏi việc một nốt mạng thuộc nhánh này lại nhận được gói tin HELLO_MESSAGE của nhánh khác. Vậy trong trường hợp này nốt mạng này sẽ tự động thêm địa chỉ ID của nhánh mới này (*CID*) vào danh sách các nốt lân cận và gửi nó tới nốt gốc *(CH)* thông qua gói tin báo cáo tình trạng đường truyền, để từ đó nốt gốc (CH) có thể biết được nhánh mạng nào tranh chấp để xử lý.

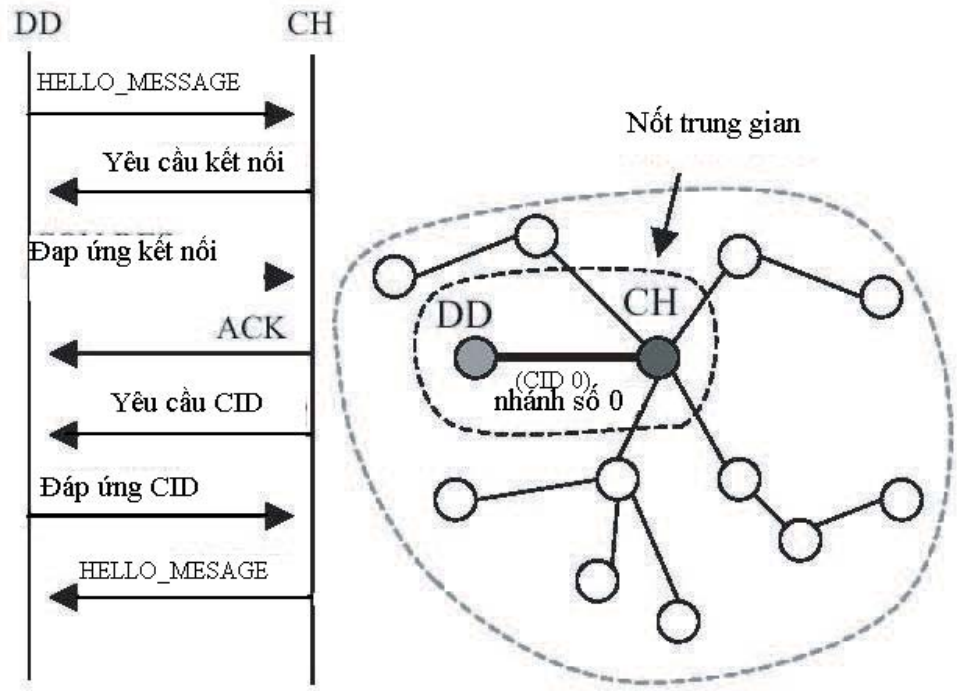
Bản tin báo cáo tình trạng kết nối cũng chứa danh sách ID nốt lân cận của nốt đó, điều này giúp cho nốt gốc biết được trọn vẹn cấu trúc mạng để có thể đưa ra cấu trúc tối ưu. Khi cấu trúc mạng cần thay đổi, nốt gốc (CH) sẽ phát đi bản tin cập nhật tới các nốt thành viên. Nốt thành viện nào nhận được bản tin cập nhật này lập tức thay đổi các thông tin về nốt gốc như trong bản tin này, đồng thời cũng tiếp tục gửi đến các nốt ở cấp thấp hơn trong nhánh cây tại thời điểm đó.

Khi một nốt thành viên có vấn đề, không thể kết nối được thì nốt gốc phải định dạng lại tuyến đường. Thông qua bản tin báo cáo tình trạng đường truyền được gửi theo chu kỳ thì nốt gốc có thể biết được vấn đề của nốt mạng đó. Nhưng khi nốt gốc gặp phải vấn đề trong liên lạc thì việc phát bản tin HELLO_MESSAGE theo chu kỳ sẽ bị gián đoạn, khi đó các nốt thành viên sẽ mất đi nốt gốc, và nhánh đó sẽ phải tự định dạng lại từ đầu theo cách tương tự như quá trình định dạng nhánh cây

### *3.2.2* Thuật toán hình cây đa nhánh.

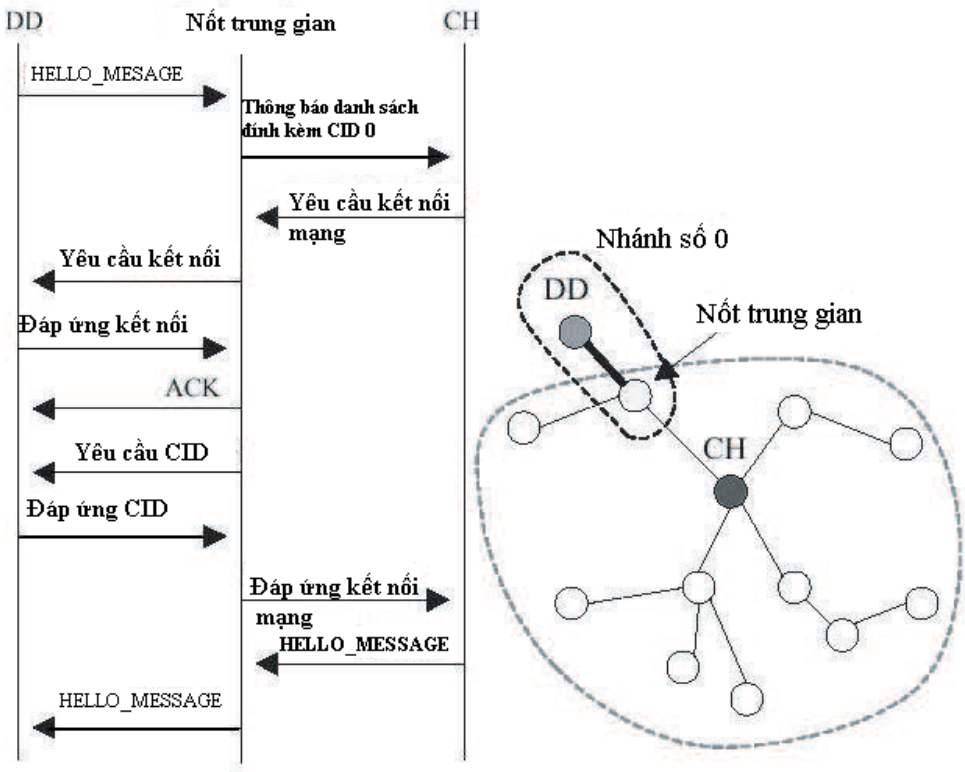
Để tạo định dạng lên loại mạng này thì cần phải sử dụng thiết bị gốc (DD). Thiết bị này có trách nhiệm gán địa chỉ ID nhóm (địa chỉ này là duy nhất) cho các nốt gốc(CH). Địa chỉ ID nhóm này kết hợp với địa chỉ ID nốt (là địa chỉ NID mà nốt gốc gán cho các nốt thành viên trong nhánh của mình) tạo ra địa chỉ logic và được sử dụng

trong các gói tin tìm đường. Một vai trò quan trọng nữa của thiết bị gốc DD là tính toán quãng đường ngắn nhất từ nhánh mạng tới DD và thông báo nó tới tất cả các nốt mạng.



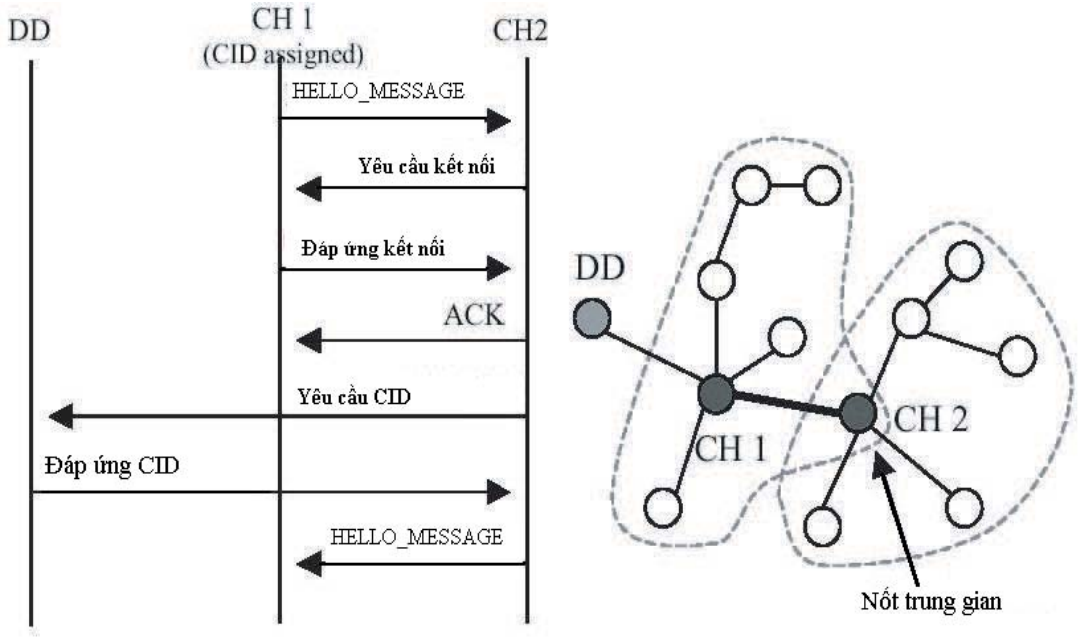

*Hình3.5 Gán địa chỉ nhóm trực tiếp*

Khi thiết bị gốc DD tham gia vào mạng, nó sẽ hoạt động như một nốt gốc của nhánh số 0 (*CID 0*) và bắt đầu phát quảng bá HELLO_MESSAGE tới các nốt lân cận. Nếu một nốt gốc (CH) nhận được bản tin này, nó sẽ gửi bản tin yêu cầu kết nối tới DD để tham gia vào CID 0, sau đó nốt gốc này sẽ yêu cầu DD gán cho nó một ID nhánh (CID). Như vậy thì nốt gốc này có hai địa chỉ logic, một là thành viên của CID 0, thứ hai là địa chỉ của nốt gốc. Khi nốt gốc tạo ra một nhánh mới, (một CID mới), nó sẽ thông báo đến các nốt thành viên của nó bằng bản tin HELLO_MESSAGE.

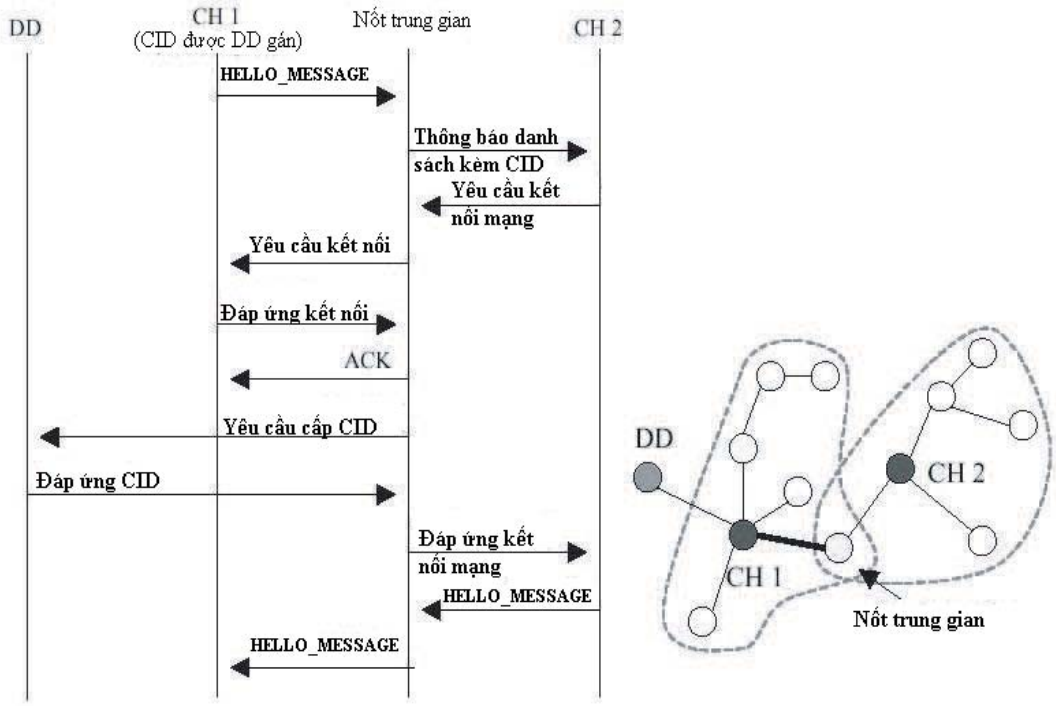



*Hình3.6 Gán địa chỉ nhóm qua nốt trung gian*

Khi một thành viên nhận được bản tin HELLO_MESSAGE từ thiết bị DD, nó sẽ thêm địa chỉ ID của CID 0 vào danh sách thành viên rồi thông báo cho nốt gốc. Nốt gốc được thông báo này sẽ chọn nốt thành viên này như là một nốt trung gian giữa nó với nốt gốc của nó, rồi gửi bản tin yêu cầu kết nối mạng tới các nốt thành viên để thiết lập kết nối với thiết bị DD. Nốt trung gian này yêu cầu một kết nối và tham gia vào thành viên của nhóm số 0. Sau đó nó sẽ gửi bản tin yêu cầu CID tới thiết bị DD. Đến khi nhận được đáp ứng CID, nốt trung gian này gửi bản tin đáp ứng liên kết mạng này tới nốt CH, bản tin này chứa các thông tin về địa chỉ ID nhánh mới cho nốt gốc CH. Sau khi nốt gốc có được CID mới, thì cách thành viên trong nhánh của nốt gốc cũng sẽ nhận được thông qua HELLO_MESSAGE




*Hình 3.7 Gán địa chỉ nhóm qua nốt gốc*




*Hình3.8 Gán địa chỉ nhóm qua nốt gốc và nốt trung gian*

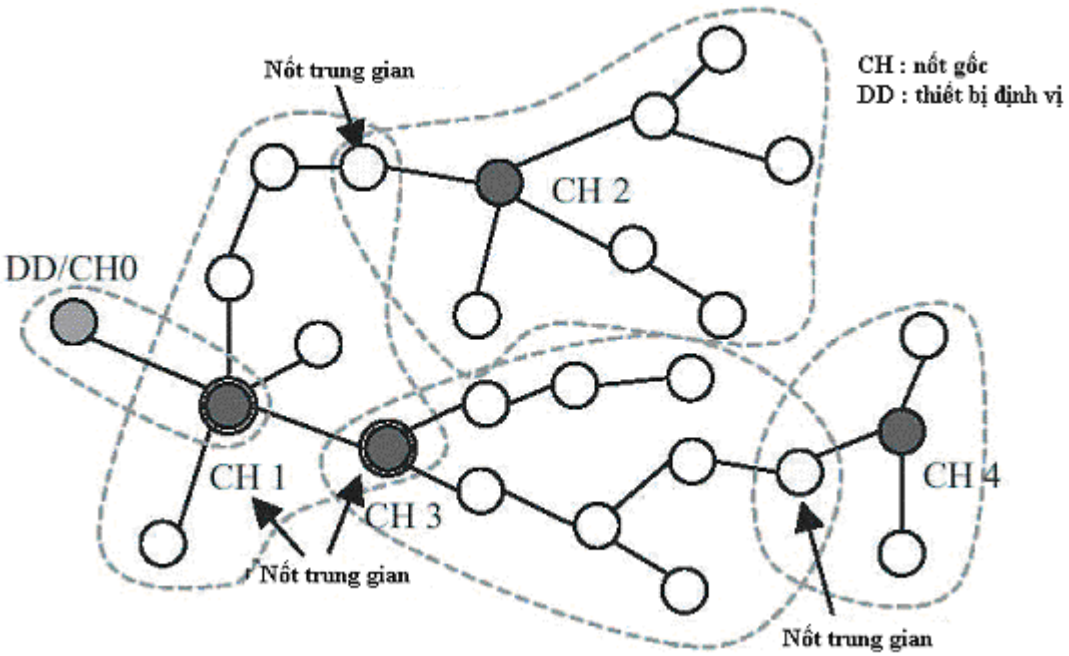
Trong mạng này thì việc tự tổ chức mạng là một tính chất khá mạnh mẽ, và mềm dẻo. Cứ nhánh mạng liền trước sẽ có nhiệm vụ gán CID cho nhánh mạng sau. Quá trình này được mô tả rõ nét hơn ở *hình 3.5,3.6,3.7,3.8.*

Mỗi một nốt thành viên của nhánh phải ghi lại thông tin về nhánh gốc và các nhánh con của nó, hoặc cả ID của nốt trung gian nếu có. Thiết bị gốc phải có trách nhiệm lưu giữ toàn bộ thông tin về cấu trúc cây mạng của các nhánh.

Cũng giống như các nốt thành viên của nhánh thì các nốt gốc CH cũng là thành viên của thiết bị gốc và như vậy chúng cũng phải có trách nhiệm thông báo tình trạng đường truyền đến DD. Để thực hiện thì nốt gốc phải gửi định kỳ bản tin thông báo tình trạng đường truyền trong mạng tới DD, bản tin này chứa danh sách CID lân cận. DD sau khi xử lý thông tin sẽ tính toán, chọn lựa ra đường truyền tối ưu nhất rồi thông báo định kỳ tới các nhánh của nó thông qua bản tin cập nhật.

Như trên ta có thể thấy vai trò của thiết bị gốc này là rất quan trọng, chính vì thế luôn cần có những thiết bị gốc dự phòng (BDD) sẵn sang thay thế thiết bị chính khi gặp sự cố.

*Hình 3.9* mô tả việc liên lạc trong nhánh. Các nốt trung gian vừa liên kết các nhánh mạng, vừa chuyển tiếp các gói tin giữa các nhánh mạng. Khi nốt trung gian nhận được một gói tin, nó sẽ kiểm tra địa chỉ đích của gói tin đó, sau đó sẽ chuyển tới địa chỉ đích của nó nếu địa chỉ đích nằm trong nhánh này hoặc là chuyển tiếp tới nốt trung gian tiếp theo của nhánh liền kề nếu địa chỉ đích không nằm trong nhánh của nó.



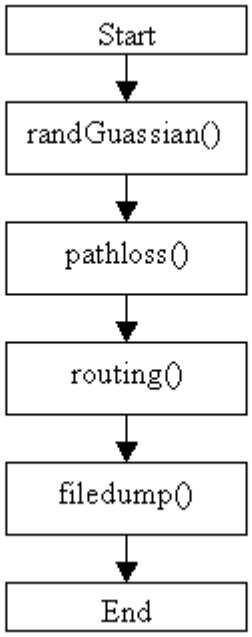

*Hình3.9Mạng cây đa nhánh và các nốt trung gian*

Chỉ duy nhất thiết bị gốc mới có thể gửi bản tin tới tất cả các nốt trong mạng, bản tin này được chuyển dọc theo tuyến đường của các nhánh. Các nốt trung gian thì chuyển tiếp các gói tin quảng bá từ nhánh gốc đến các nhánh con.

# CHƯƠNG 4 Mô phỏng thuật toán định tuyến trong mạng mesh của ZigBee/IEEE802.15.4 bằng phần mềm MatLab và Visual C.

Để có thể hiểu được rõ thuật toán định tuyến Zigbee, em đã xây dưng một chương trinh phầm mềm mô phỏng quá trình định tyến để đưa ra một bản đồ định tuyến của một node trong mạng đến tất cả các nốt khác trong mạng của nó. Chương trình này dựa trên thuật toán tìm đường ADOV và thuật toán bó cụm hình cây của Motorola. Để thuận tiện cho việc xây dựng được chương trình, đầu tiên em tiến hành xây dựng một lưu đồ thuật toán cho chương trình như hình 4.1. Lưu đồ này được cụ thể hoá bằng một chương trình viết bằng ngôn ngữ Visual C để xử lý quá trình tính toán, kết quả của chương trình được xuất ra file có định dạng MatLab để chạy mô phỏng.

## *4.1* Sơ đồ thuật toán.




*Hình4.1 Sơ đồ chức năng của chương trình.*

Chương trình chính main() gồm có các chương trình con như hình4.1 để thực hiện quá trình tính toán và xử lý.

Chương trình con *randGaussian()* có tác dụng tạo ra các biến ngẫu nhiên Gauss với tham số đầu vào từ hàm *rand()*. Kết quả của hàm này cho ra một số nốt (ở đây là 30 và 100) được phân bố ngẫu nhiên trên một vùng hình quạt từ một nốt cho trước.. Từ đó

ta có thể sử dụng hàm *distant()* để tính khoảng cách giữa các nốt bất kỳ với nhau trong vùng hình quạt này.

Chương trình con *pathloss()* có nhận tham số đầu vào là kết quả của hàm *distant()*, và năng lượng giữa các nốt lân cận tích cực trong mạng. Hàm này sẽ dựa vào kết quả của hàm tính khoảng cách để tính ra năng lượng nhiễu có ích giữa các nốt lân cận với nhau, giá trị năng lượng này chỉ được tính khi giá tri khoảng cách với nốt lân cận nằm trong một khoảng xác định [dmin, dmax]. Sau đó sẽ đưa ra giá trị năng lượng thật mà một nốt nhận được bằng việc cộng các giá trị năng lượng này với nhau.

Chương trình con *routing()* là hàm nhận tham số đầu vào là khoảng cách và năng lượng (năng lượng của một nốt nhận được tính thông qua năng lượng truyền từ nốt nguồn tới và năng lượng nhiễu có ích nhận đực từ các nốt lân cận) sắp xếp theo thuật toán bọt xà phòng và thuật toán sắp xếp Dijkstra để chọn ra tuyến đường tối ưu nhất (ngắn nhất và năng lượng truyền tin nhỏ nhất) từ một nốt tới tất cả các nốt còn lại. Cứ sau mỗi lần lặp, kết quả sẽ được lưu và bộ nhớ cho đến khi đến được đích cần đến.

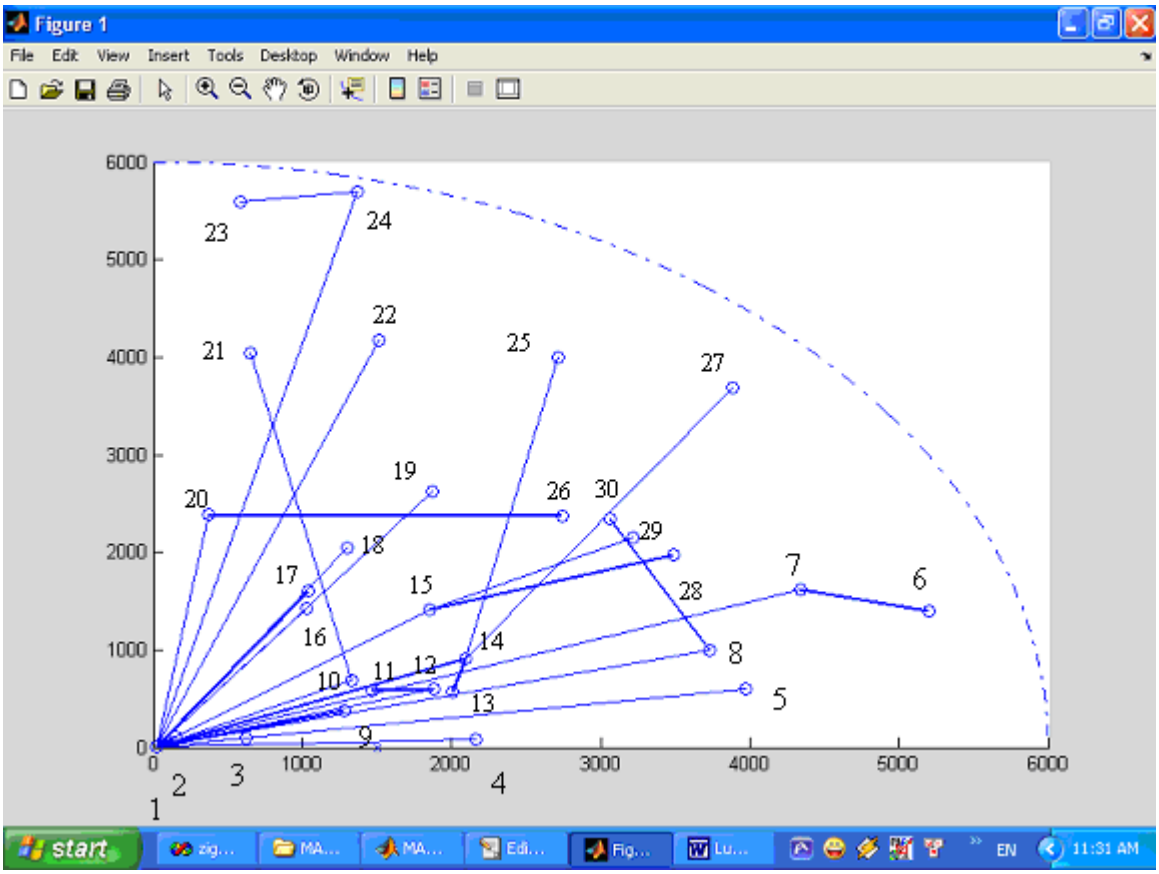
Chương trình con *filedump()* hàm này nhận tham số đầu vào là các kết quả của hàm *routing()* và bộ nhớ để từ đó vẽ đường từ nốt nguồn tới nốt đích vào file có định dạng MatLab.

### *4.2* Kết quả và đánh giá

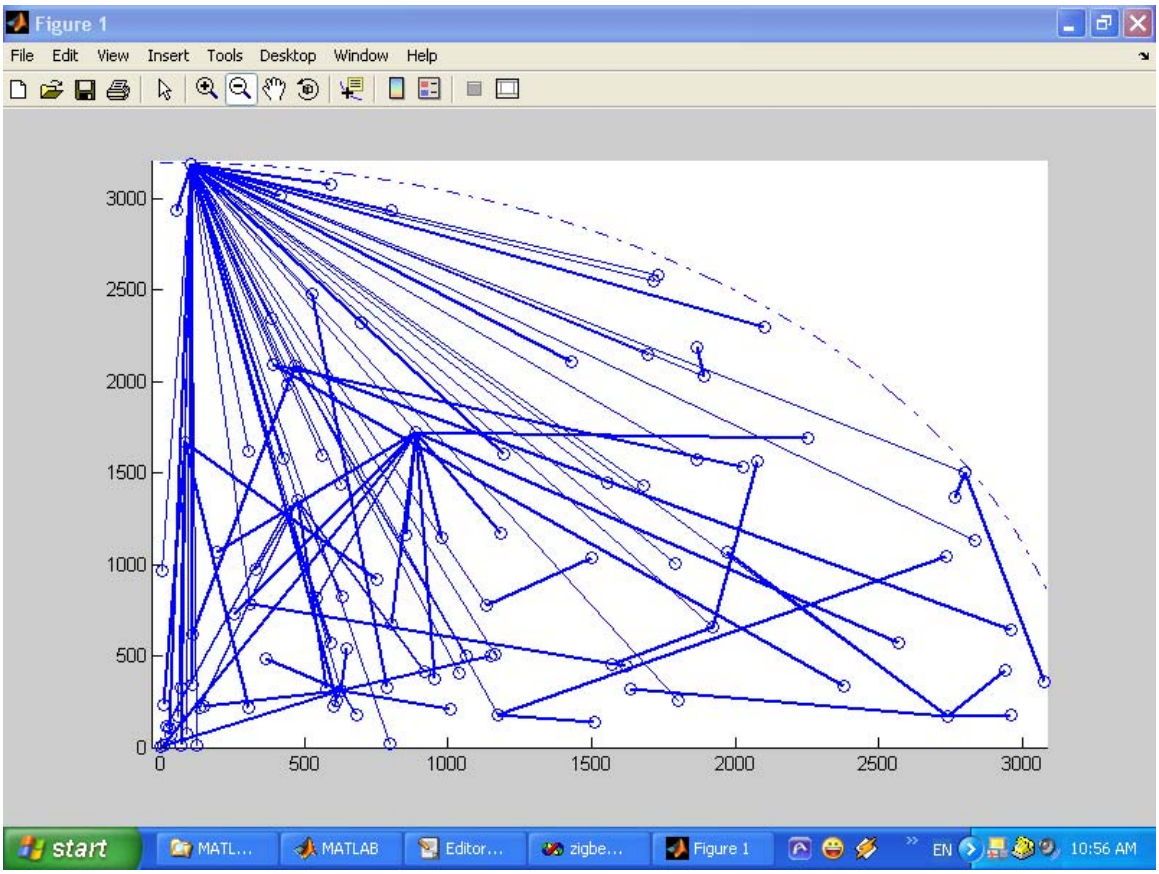
Sau khi xây dựng chương trình từ các chương trình con em đã thu được kết quả như hình 4.2. và 4.3. Chương trình này có thể mở rộng để xử lý quá trình định tuyến cho trên 100 nốt mạng mà không ảnh hưởng đến kết quả. Theo kết quả như hình 4.2 ta có thể lập ra được một bảng định tuyến từ nốt nguồn đến nốt đích như bảng 4.1.

| Nốt gốc | Bước truyền thứ 2 | Bước truyền thứ 3 |
|---|---|---|
| 1 | 2, 3, 4, 9, 10, 8, 16, 17, 5, 18, 20, 11 , 14, 15, 22, 24 | 23, 21, 13, 28, 29, 30, 6, 27, 25, 7, 19, 12, 26 |

*Bảng 4.1 Bảng định tuyến*




*Hinh4.2 Kết quả với 30 nốt mạng.*




*Hình4.3Kết quả với 100 nốt mạng.*

Từ kết quả mô phỏng của chương trình như được thấy ở hình 4.2 và 4.3, em thấy rằng kết quả này là đã đạt được mục tiêu của thuật toán định tuyến của tầng mạng Zigbee đã đề ra đó là đã xây dựng được một bảng định tuyến cho một thành viên bất kỳ trong mạng. Chương trình mô phỏng chỉ xin phép đưa ra một ví dụ tìm đường tối ưu cho một nốt cụ thể trong mạng đến tất cả các thành viên khác trong mạng, và tính toán được nhiễu xuyên kênh giữa các thành viên lân cận. Việc tìm đường tối ưu là một yêu cầu rất quan trọng trong công nghệ Zigbee vì nó giải quyết được vấn đề hiệu quả năng lượng trong truyền tin và vấn đề chống xung đột của mạng. Thời gian thực của chương trình khá nhanh và kích thước chương trình nhỏ (208 kb) nên có thể nhúng được vào trong chip của thiết bị Zigbee.

### *4.3* Kết luận

Thông qua đề tài này em đã có thể hiểu một cách rõ hơn về công nghệ truyền dẫn không dây ZigBee/IEEE 802.15.4, từ mô hình giao thức đến thuật toán truyền tin. Và từ đó có thể thấy được tính ưu việt nổi trội của công nghệ ZigBee với các công nghệ hiện nay.

Tuy nhiên do thời gian hạn chế nên chương trình vẫn còn nhiều thiếu xót. Đó là chưa mô phỏng được tính tự cấu hình mạng mạng của thuật toán định tuyến ZigBee trong mạng mesh. Hy vọng trong thời gian tới nếu có điều kiện em có thể tiếp tục hoàn thiện, phát triển và mở rộng chương trình để có thể tiếp cận sâu hơn với các ứng dụng của công nghệ hiện hẵng còn rất mới mẻ này.

# PHỤ LỤC

## Mã nguồn của chương trình:

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <time.h>
#include <limits.h>
#include <memory.h>
#include <string.h>
//#include "mesh.h"
#define RCV_TN -108.0 +10*log10(BW/1000000.0)
#define C 29800000
#define DOUBLE_MAX (double)INT_MAX
#define DOUBLE_MIN (double)INT_MIN
#define EPS 0.0000001
#define FALSE 0
#define TRUE 1
#define DESIRED 0
#define INTERFERENCE 1
#define MAXHOPS 10
#ifndef M_PI
#define M_PI  3.14159265358979311 60E0
# endif
#ifndef M_PI_2
#define M_PI_2   1.57079632679489655 80E0
# endif
typedef struct location
{
double x,y;
} Location;
```

```
#define MINIMIZE_HOPS 1
#define MINIMIZE_EPB 2
static double RCVSENS[] = RECEIVER_SENSITIVITY;
static double PL_EXP_D[] = PATHLOSS_EXPONENT_DESIRED;
static double PL_EXP_I[] = PATHLOSS_EXPONENT_INTERFERENCE;
static double PL_DIST_D[] = PATHLOSS_DISTANCE_DESIRED;
static double PL_DIST_I[] = PATHLOSS_DISTANCE_INTERFERENCE;
static double INR = INTERFERENCE_TO_NOISE_RATIO;
static int BPS[] = BYTES_PER_SYMBOL;
double RefPathLoss;
double dmax(double x, double y) { return (x>y)?x:y;}
double dmin(double x, double y) { return (x>y)?y:x;}
static int newGap(int gap) {
/* Chuong trinh sap xep theo thuat toan combsorting */
gap = (gap * 10) / 13;
if (gap == 9 || gap == 10)
gap = 11;
if (gap < 1)
gap = 1;
return gap;
}
static void combsort(double a[], int aSize) {
int i,j;
double tmp;
int gap, swapped;
gap = aSize;
for (;;) {
gap = newGap(gap);
swapped = FALSE;
for (i = 0; i < aSize - gap; i++) {
```

```
j = i + gap;
if (a[i] > a[j]) {
tmp = a[i];
a[i] = a[j];
a[j] = tmp;
swapped = TRUE;
}
}
if (gap == 1 && !swapped)
break;
}
}
double randGaussian() {
double x1,x2,w;
do {
x1= 2.0*((double)rand()/RAND_MAX) -1.0;
x2= 2.0*((double)rand()/RAND_MAX) -1.0;
w = x1*x1+x2*x2;
} while (w >= 1.0);
w = sqrt( (-2.0*log(w) ) / w);
return x1*w;
}
Location randLocInCircle() {
Location newLoc;
double a,r;
a = SECTOR*((double)rand()/RAND_MAX);
r = CELLRADIUS*((double)rand()/RAND_MAX);
newLoc.x = r*sin(a);
newLoc.y = r*cos(a);
return newLoc;
```

```
}
double distance(Location x, Location y) {
/* Khoang cach giu hai not  mang */
return sqrt( (x.x-y.x)*(x.x-y.x)+(x.y-y.y)*(x.y-y.y) );
}
double pathloss_dB(double Distance, int dORi) {
/* Multi-component pathloss calculation */
int i;
double pl;
pl = RefPathLoss;
if (dORi== DESIRED) {
for ( i=0;i<sizeof(PL_EXP_D)/8; i++ ) {
if( Distance >= PL_DIST_D[i] )
pl += 10*PL_EXP_D[i]*log10( dmin(PL_DIST_D[i+1],Distance)/PL_DIST_D[i] );
}
} else {
for ( i=0;i<sizeof(PL_EXP_I)/8; i++ ) {
if( Distance >= PL_DIST_I[i] )
pl += 10*PL_EXP_I[i]*log10( dmin(PL_DIST_I[i+1],Distance)/PL_DIST_I[i] );
}
}
return pl + PL_STD_DEV*randGaussian();
}
int dijkstraIteration(int Picked[], int Route[], int Hops[], int Links[][NODES]) {
int i,j;
int bestNode;
int bestHops;
bestHops = INT_MAX;
for ( i=0; i<NODES; i++ ) {
if (( !Picked[i] ) && (Hops[i] < bestHops )) {
```

```
bestNode = i;
bestHops = Hops[i];
}
}
if ( bestHops == INT_MAX) {
return 0;
}
i = bestNode;
Picked[i] = 1;
/* Cap nhat cac not lan can hoat dong cua mot not */
for ( j=0; j<NODES; j++ ) {
if ( i!=j && Links[i][j] && Hops[j]>Hops[i]+1 ) {
Hops[j] = Hops[i]+1;
Route[j] = i;
}
}
return 1;
}
void powercontrol(double pathLoss[][NODES], int links[][NODES], double
xmtPwr[][NODES] ) {
/* Minimizes the power on each link given the selected modulation for that link*/
int i,j;
for ( i=0;i<NODES; ++i ) {
for ( j=0;j<NODES; ++j ) {
xmtPwr[i][j] = DOUBLE_MIN;
if(i!=j && links[i][j])
xmtPwr[i][j] = pathLoss[i][j] - 2*ANTGAIN + RCVSENS[links[i][j]-1] + FM;
}
}
}
```

```
double modulation_avg( int route[],int links[][NODES] ) {
int i,j,k,l;
int modulation[sizeof(RCVSENS)/8];
memset(modulation,0,sizeof(RCVSENS)/8*sizeof(int));
for( l=0,j=0,i=0;i<NODES;++i ) {
k = i;
do {
j = route[k];
if(j!=k) {
modulation[links[k][j]-1]++;
l++;
}
} while(k=j);
}
for ( j=0,i=0;i<sizeof(RCVSENS)/8;++i)
j += modulation[i]*BPS[i];
return ((double)j)/((double)l);
}
double avgNumberOfHops(int hops[]) {
/* tinh toan buoc nhay trung binh tren 1 tuyen duong */
int hopCnt[MAXHOPS];
int i;
double avgNrHops;
memset(hopCnt,0,MAXHOPS*sizeof(int));
for ( i=1; i<NODES; i++ ) {
if ( hops[i] < MAXHOPS )
hopCnt[hops[i]]++;
else
hopCnt[0]++;
}
```

```
for ( avgNrHops=0.0,i=1;i<MAXHOPS; i++ )
avgNrHops += hopCnt[i]*i;
avgNrHops /= (double) NODES;
return avgNrHops;
}
double txActivity(int hops[],double avgMod,double avgNrHops) {
double avgBurstSize,OFDMsymbols;
avgBurstSize = (0.15*1500+0.22*560+0.5*48+0.13*400)/avgMod;
OFDMsymbols =
USER_DATA*1024.0*1024.0/avgBurstSize*(avgBurstSize+PREAMBLE_SIZE);
OFDMsymbols *= avgNrHops*USER_ACTIVE_TIME/avgMod;
OFDMsymbols +=
MSH_CTRL_SLOTS*MSH_CTRL_SLOT_SIZE/NODES*3600000/
FRAME_DURATION;
return OFDMsymbols/(BW*OVERSAMPLING/FFT_SIZE*(1+CP))/3600;
}
long intercell_interference( int route[], double xmtPwr[][NODES], Location pos[],
Location posI, double intLevel[],long intLevel_index ) {
int i,j,k;
double pl;
for( j=0,i=0;i<NODES;++i ) {
k = i;
do {
j = route[k];
if(j!=k) {
pl = pathloss_dB(distance(pos[i],posI),INTERFERENCE);
intLevel[intLevel_index++]= xmtPwr[k][j] + 2*ANTGAIN - pl -
CHANNEL_REJECTION;
}
} while(k=j);
}
```

```
return intLevel_index;
}
void routing(int route[],int hops[],int links[][NODES],double xmtPwr[][NODES],int
type) {
/* thuat toan nay dua ra chon lua giua buoc nhay nho nhat va nang luong truyen*/
int picked[NODES];
int i,j;
int changes;
double xmtEpB[NODES];
double t;
if ( type == MINIMIZE_HOPS ) {
/* tim duong dua tren quang duong ngan nhat.*/
memset(picked,0,NODES*sizeof(int));
i =1;
while (i)
i = dijkstraIteration(picked,route,hops,links);
for (i=0; i<NODES; i++ ) {
for ( j=0; j< NODES; j++ ) {
if (hops[j] == hops[route[i]] && links[i][j] > links[i][route[i]])
route[i] = j;
}
}
}
if (type == MINIMIZE_EPB) {
/* thuat toan tim duong dua tren muc nang luong la nho nhat.*/
memset(xmtEpB,0,NODES*sizeof(double));
for (i=0; i<NODES; ++i)
xmtEpB[i] = DOUBLE_MAX;
xmtEpB[0] = 0; /* Mesh Gateway */
for (;;) {
```

```
changes = FALSE;
for( i=0;i<NODES;i++ ) {
for( j=1;j<NODES;j++ ) {
if( i!=j && links[i][j]) {
t= pow(10.0,xmtPwr[j][i]/10.0)/BPS[links[j][i]-1];
if(xmtEpB[i] + t + EPS< xmtEpB[j] ) {
hops[j] = hops[i]+1;
route[j] = i;
xmtEpB[j] = xmtEpB[i] + t;
changes = TRUE;
}
}
}
}
if (changes == FALSE)
break;
}
}
}
void filedump(double intLevel[],int intLevel_index,double TxActivity,double
avgNrHops) {
FILE *fp;
int i;
char fn[20];
double t;
#if (ROUTING_TYPE == MINIMIZE_HOPS)
sprintf(fn,"%s\0","meshHops.m");
#elif (ROUTING_TYPE == MINIMIZE_EPB)
sprintf(fn,"%s\0","meshEpB.m");
#endif
```

```
if((fp = fopen(fn,"wr"))== NULL) {
printf("Failed to open file: %s.\n",FILENAME);
exit(-1);
} else {
#if (ROUTING_TYPE == MINIMIZE_HOPS)
sprintf(fn,"%s\0","Hops");
#elif (ROUTING_TYPE == MINIMIZE_EPB)
sprintf(fn,"%s\0","EpB");
#endif
fprintf(fp,"Iv%s = [\n",fn);
for (i=0;i<intLevel_index;i++)
fprintf(fp,"% 4.3f\n",intLevel[i]);
fprintf(fp,"];\nTxActivity%s = % 2.5f;\n",fn,TxActivity);
fprintf(fp,"Realizations%s = %d;\n",fn,REALIZE_INTERFERENCE);
fprintf(fp,"Cells%s = %d;\n",fn,REALIZE_CELL);
fprintf(fp,"AvgNrHops%s = % 2.5f;\n",fn,avgNrHops);
t = TxActivity/REALIZE_CELL/REALIZE_INTERFERENCE/avgNrHops;
fprintf(fp,"semilogy(Iv%s,fliplr([1:size(Iv%s,1)])*% 2.8e)\n",fn,fn,t);
fprintf(fp,"axis([-140 -90 0.0001 1]);\n");
fprintf(fp,"hold on;\n");
fprintf(fp,"line([% 2.3f % 2.3f],[0.0001 1]);\n",RCV_TN+INR,RCV_TN+INR);
fclose(fp);
}
}
void filedumpcell(Location pos[],Location posI,int links[][NODES],int route[], int cell)
{
FILE *fp;
int i;
char fn[20];
#if (ROUTING_TYPE == MINIMIZE_HOPS)
```

```
sprintf(fn,"mang_meshzigbee_Hops%d%s\0",cell,FILENAME);
#elif (ROUTING_TYPE == MINIMIZE_EPB)
sprintf(fn,"mang_meshzigbee_EbP%d%s\0",cell,FILENAME);
#endif
if((fp = fopen(fn,"wr")) == NULL) {
printf("Failed to open file: %s.\n",fn);
exit(-1);
} else {
fprintf(fp,"figure;\n");
fprintf(fp,"axis([0 % 2.3f 0
%d]);\n",dmax(CELLRADIUS,INT_DIST),CELLRADIUS);
fprintf(fp,"hold on;\n");
for(i=0;i<NODES;i++) {
fprintf(fp,"plot(% 2.3f, % 2.3f,'o');\n",pos[i].x,pos[i].y);
if(i!=route[i]) {
fprintf(fp,"line([% 2.3f % 2.3f],[% 2.3f % 2.3f],'LineWidth',% 1.2f);\n",
pos[i].x,pos[route[i]].x,pos[i].y,pos[route[i]].y,(double)links[i][route[i]]/4.0);
}
}
fprintf(fp,"plot(% 2.3f, % 2.3f,'x');\n",posI.x,posI.y);
fprintf(fp,"Coverage = [");
for(i=0;i<CELLRADIUS;i++)
fprintf(fp,"% 2.3f ",sqrt((double)(CELLRADIUS*CELLRADIUS-i*i)));
fprintf(fp,"];\nplot([0:%d],Coverage,'-.');\n",CELLRADIUS-1);
fclose(fp);
}
}
 main()
{
int hops[NODES];
```

```
int links[NODES][NODES];
double pathLoss[NODES][NODES];
int route[NODES];
Location pos[NODES];
double xmtPwr[NODES][NODES];
double
intLevel[MAXHOPS*NODES*REALIZE_INTERFERENCE*REALIZE_CELL];
double avgMod,avgNrHops,TxActivity;
Location posI;
int i,j,k,cell;
long intLevel_index,changes;
srand( (unsigned) time(NULL));
memset(intLevel,0,MAXHOPS*NODES*REALIZE_INTERFERENCE*REALIZE_C
ELL*sizeof(double));
posI.x = INT_DIST;
posI.y = 0.0;
intLevel_index = 0;
RefPathLoss = 20*log10(4*M_PI*FC/C);
for ( cell=0;cell<REALIZE_CELL;cell++ ) {
memset(pos,0,NODES*sizeof(Location));
memset(hops,0,NODES*sizeof(int));
memset(route,0,NODES*sizeof(int));
memset(xmtPwr,0,NODES*NODES*sizeof(double));
memset(links,0,NODES*NODES*sizeof(int));
memset(pathLoss,0,NODES*NODES*sizeof(double));
for (i=0; i<NODES; ++i) {
pos[i] = randLocInCircle();
hops[i] = INT_MAX;
}
hops[0] = 0; /* Mesh Gateway */
for (i=0; i<NODES; i++) {
```

```
pathLoss[i][i] = DOUBLE_MAX;
for ( j=0; j<i; j++ ) {
pathLoss[i][j] = pathloss_dB( distance(pos[i],pos[j]),DESIRED );
for ( k=0;k<sizeof(RCVSENS)/8;k++ ) {
if ( XMTPWR + 2*ANTGAIN - RCVSENS[k] - FM > pathLoss[i][j] )
links[i][j] = k+1;
}
pathLoss[j][i] = pathLoss[i][j];
links[j][i] = links[i][j];
}
}
powercontrol(pathLoss,links,xmtPwr);
routing(route,hops,links,xmtPwr,ROUTING_TYPE);
avgNrHops = avgNumberOfHops( hops );
avgMod = modulation_avg( route,links );
TxActivity = txActivity( hops,avgMod,avgNrHops );
for (i=0; i< REALIZE_INTERFERENCE; i++) {
intLevel_index = intercell_interference(route,xmtPwr,pos,posI,
intLevel,intLevel_index);
}
filedumpcell(pos,posI,links,route,cell);
}
combsort(intLevel,intLevel_index);
filedump(intLevel,intLevel_index,TxActivity,avgNrHops);
return 1;
 }
```

## Tài liệu tham khảo

[1] Le Tuan Khanh, "Implementation of Zigbee Ready IEEE 802.15.4-RFIC" http://www.chipcon.com , 2005-04

[2] Nilesh Rajbharti, AN965, http://www.microchip.com

[3] Samir R. Das, Charles E. Perkins, Elizabeth M. Royer, "Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks"- 2005

[4] Heikki N. Koivo, "Basics using MATLABNeural Network Toolbox", 2005

[5] Wendi Rabiner Heinzelman, Anantha Chandrakasan, and Hari Balakrishnan "Energy-Efficient Communication Protocol forWireless Microsensor Networks"

[6] Chipcon, http://www.chipcon.com , 2005-03

[7] "IEEE Standards 802.15.4, IEEE 2003, ISBN 0-7381-3677-5 SS95127",2004

[8] "IEEE 802.15.4 Standard Specification", http://www.standards.ieee.org

[9] "Figure 8 Wireless", http://www._gure8wireless.com , 2005-03

[10] ZigBee Alliance, http://www.zigbee.org , 2005-03

[11] Zigbee technical documents, www.zigbee.org

[12] MSDN Library Visual Studio 6 , 2003-04

[13] Mã nguồn mở tại trang web http://www.koders.com/

[14] http://www.freescale.com

[15] http://www.metrowerks.com