AN4834

Lưu ý áp dụng

Thực hiện các bộ phát và bộ thu cho các giao thức điều khiển từ xa hồng ngoại với STM32Cube

Giới thiệu

Bức xạ hồng ngoại là vùng của phổ điện từ nằm giữa vi sóng và ánh sáng nhìn thấy.

Bức xạ hồng ngoại có hai dãy. Ánh sáng hồng ngoại gần có bước sóng gần nhất với ánh sáng nhìn thấy, trong khi tia hồng ngoại xa gần vùng vi sóng của phổ điện từ hơn.

Các sóng ngắn hơn là những sóng được sử dụng bởi điều khiển từ xa. Thông tin được truyền và nhận bằng năng lượng điện từ, không sử dụng dây dẫn.

Công nghệ hồng ngoại cung cấp hình thức giao tiếp không dây đơn giản. Ngày nay, hầu hết tất cả các thiết bị âm thanh và video đều có thể được điều khiển bằng điều khiển từ xa hồng ngoại. Ở đầu nhận, một máy thu phát hiện các xung ánh sáng, các xung này được xử lý để lấy / giải mã thông tin mà chúng chứa.

Có nhiều tiêu chuẩn giao thức hồng ngoại phổ biến được sử dụng để truyền dữ liệu qua ánh sáng hồng ngoại, chẳng hạn như (trong số những tiêu chuẩn khác) RC5 và SIRC.

Mục đích của ghi chú ứng dụng này là cung cấp giải pháp chung để triển khai bộ phát IR (thiết bị điều khiển từ xa) và bộ thu sử dụng vi điều khiển thuộc Dòng STM32F0, STM32F3, STM32G0, STM32G4, STM32L4, STM32L4 +, STM32L5, STM32WB và STM32WL. Một ví dụ về triển khai phần mềm được cung cấp cho các giao thức RC5 và SIRC. Các giao thức khác cũng tương tự như vậy và bạn có thể dễ dàng điều chỉnh ví dụ để hỗ trợ giao tiếp cụ thể của nhà sản xuất khác.

Các giải pháp máy phát và máy thu hồng ngoại được mô tả trong tài liệu này được triển khai bằng cách sử dụng lớp trừu tượng phần cứng STM32Cube và gói phần mềm X-CUBE-IRREMOTE, có sẵn trên *www.st.com* .

Ghi chú: *Mặc dù hầu hết các MCU STM32 đều có IRTIM, bo mạch STM32 EVAL điển hình không được trang bị các thành phần IR cần thiết. Do đó, X-CUBE-IRREMOTE không hỗ trợ bảng EVAL này, chỉ cung cấp các ví dụ F0, F3 và G0.*

Nội dung

Danh sách các bảng

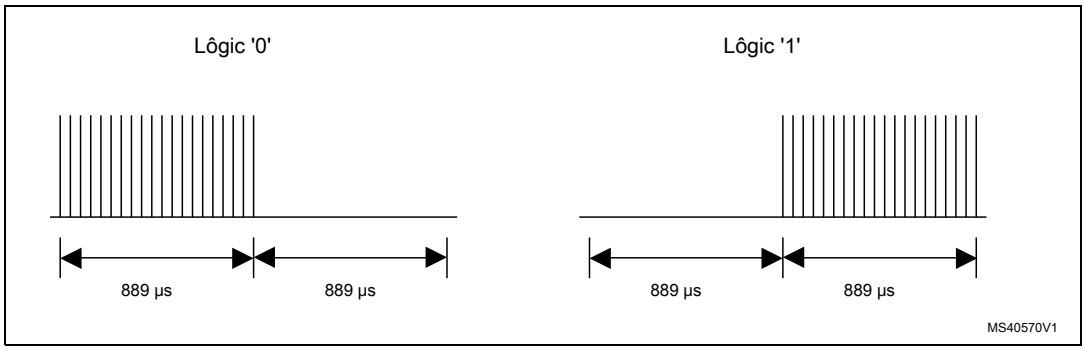
Danh sách các số liệu

1 Đặc tả giao thức hồng ngoại

1.1 Thông tin cơ bản về giao thức RC5

Mã RC5 là một từ 14 bit sử dụng điều chế hai pha (còn gọi là mã hóa Manchester) tần số sóng mang IR 36 kHz. Tất cả các bit đều có độ dài bằng nhau là 1,778 ms, với một nửa thời gian bit được lấp đầy bởi sự bùng nổ của sóng mang 36 kHz và nửa còn lại là không hoạt động. Một số 0 logic được biểu diễn bằng một cụm trong nửa đầu của thời gian bit. Số "1" hợp lý được biểu thị bằng một cụm trong nửa sau của thời gian bit. Chu kỳ hoạt động của tần số sóng mang 36 kHz là 33% hoặc 25%, giúp giảm tiêu thụ điện năng.

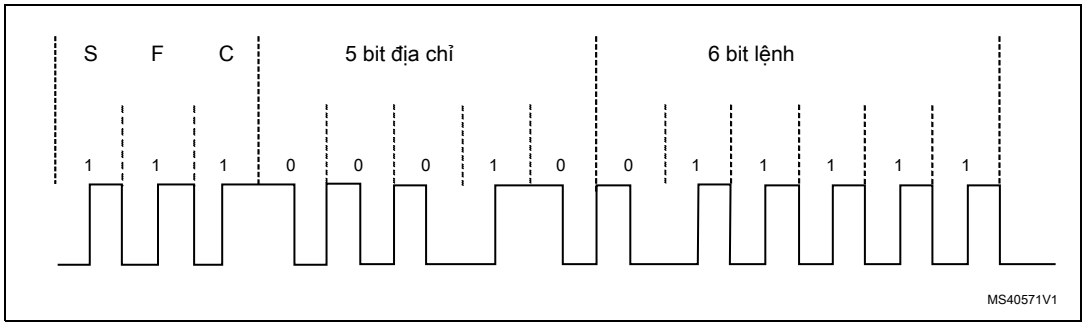
Hình 1. Biểu diễn bit RC5



Khung RC5 có thể tạo ra 2048 (32 x 64) lệnh khác nhau được tổ chức thành 32 nhóm. Mỗi nhóm có 64 lệnh khác nhau. Khung RC5 chứa các trường sau. Ví dụ về khung RC5 được hiển thị trong *Hình 2* .

- **Bắt đầu bit (S):** Độ dài 1 bit, luôn logic "1".
- **Trường bit (F):** Độ dài 1 bit, cho biết lệnh được gửi ở trường dưới (logic 1 = 0 đến 63 thập phân) hay trong trường trên (logic 0 = 64 đến 127 thập phân). Bit trường được thêm vào sau đó khi nhận ra rằng 64 lệnh trên mỗi thiết bị là không đủ. Trước đây, bit trường được kết hợp với bit bắt đầu. Nhiều thiết bị vẫn sử dụng hệ thống gốc này.
- **Bit điều khiển hoặc bit chuyển đổi (C):** Độ dài 1 bit, chuyển đổi mỗi khi nhấn nút. Điều này cho phép thiết bị nhận phân biệt giữa hai lần nhấn nút liên tiếp (chẳng hạn như “1”, “1” cho “11”).
- **Địa chỉ:** Độ dài 5 bit, chọn một trong 32 hệ thống có thể.
- **Chỉ huy:** Độ dài 6 bit, kết hợp với bit trường đại diện cho một trong 128 lệnh RC5 có thể.

Hình 2. Ví dụ về khung RC5

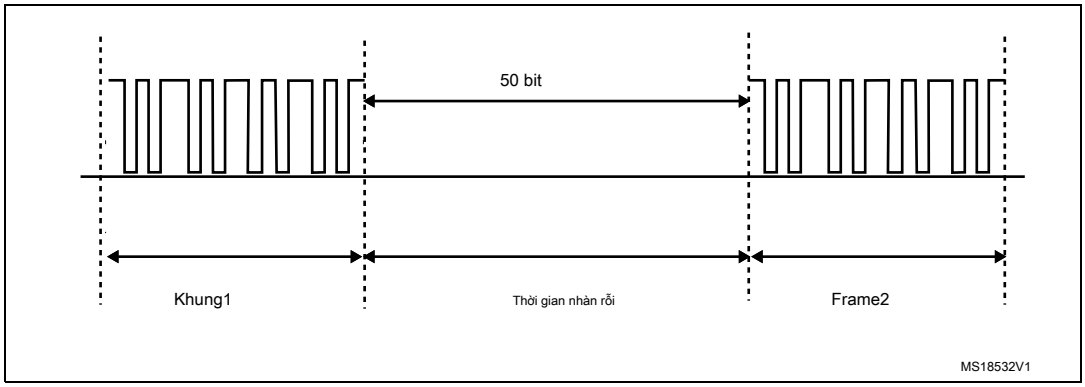


Để tránh xung đột khung, thời gian nhàn rỗi được chèn vào giữa hai khung liên tiếp có chiều rộng cụ thể (xem *Hình 3).*

Thời gian nhàn rỗi được định nghĩa là rộng 50 bit. Vì vậy, chu kỳ của khung có chiều rộng 64 x 1 bit:

64 x 1,778 = 113,792 ms.

Hình 3. Thời gian nhàn rỗi RC5



Bảng 1. Định thời RC5

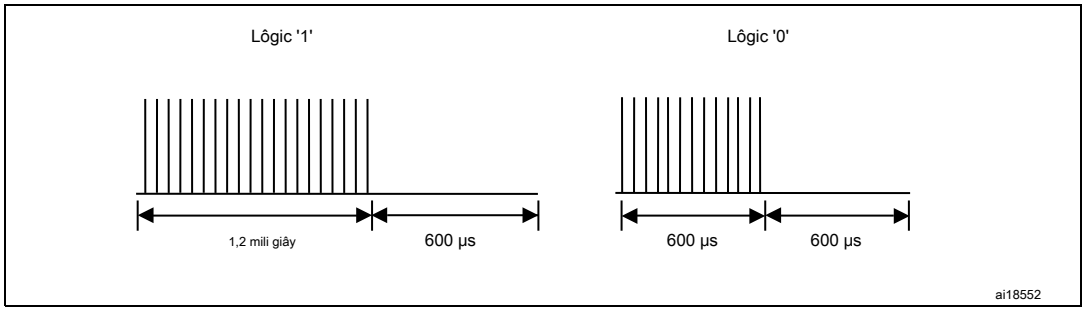
Sự miêu tả	Min	Điển hình	Max
RC5 nửa chu kỳ bit RC5 chu	640 µs	889 µs	1140 µs
kỳ bit đầy đủ RC5 thời gian	1340 µs	1778 µs	2220 µs
thông báo	23,644 mili giây	24,889 µs	26.133 mili giây
Thời gian lặp lại bản tin RC5 Thời gian bit	108,089 mili giây	113,778 mili giây	119,467 mili giây
xung sóng mang	27,233 µs	27,778 µs	28.349 µs

Ghi chú: *Việc triển khai giao thức hồng ngoại dựa trên các thông số kỹ thuật RC5 miễn phí được tải xuống từ http://www.sbprojects.com.*

1,2 Thông tin cơ bản về giao thức SIRC

Mã SIRC là một từ 12 bit. Nó sử dụng điều chế tần số sóng mang IR 40 kHz. Giao thức SIRC sử dụng mã hóa khoảng cách xung của các bit. Mỗi xung là một chùm sóng mang 40 kHz dài 600 µs. "1" lôgic cần 1,8 ms để truyền, trong khi "0" lôgic mất 1,2 ms để truyền (*hình 4).*

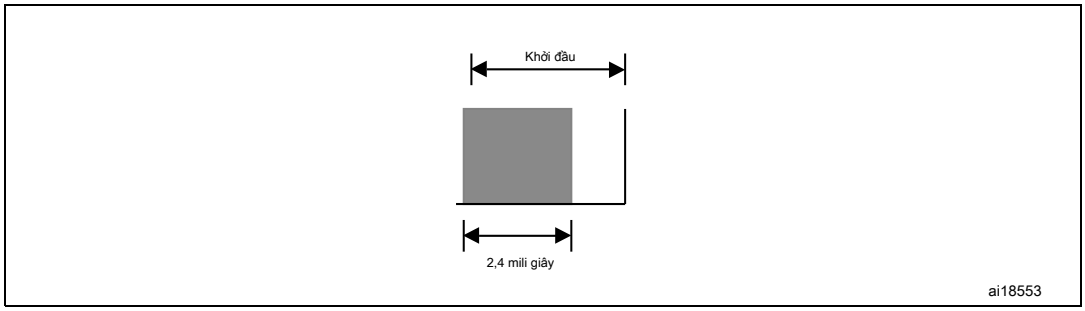
Hình 4. Độ dài của các bit logic



Khung SIRC chứa các trường sau.

- **Bắt đầu bit:** cụm bắt đầu luôn rộng 2,4 ms, theo sau là khoảng trống chuẩn là 0,6 ms.
- **Độ dài 7 bit của lệnh:** trường này chứa 7 bit được sử dụng làm trường lệnh.
- **Độ dài địa chỉ 5 bit:** trường này chứa 5 bit được sử dụng làm trường địa chỉ.

Hình 5. Độ dài của bit bắt đầu

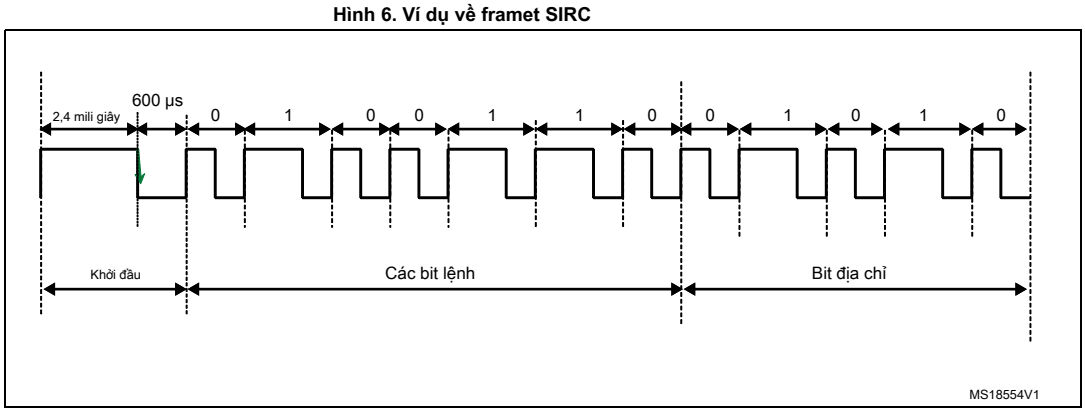


Với giao thức này, LSB được truyền đầu tiên (khung được lắp ráp LSB thành MSB). Vì nó được gửi dưới dạng 7 bit cho lệnh, tiếp theo là 5 bit cho địa chỉ vùng nhớ, mã phải chia 12 bit nhận được thành hai nhóm 7 và 5 bit.

Hình 6 hiển thị một ví dụ về khung SIRC.

Trong trường hợp này: Lệnh 26h (0100110b) và địa chỉ Ah (01010b).

Hình 6. Ví dụ về framet SIRC



Thời gian nhàn rỗi được chèn vào giữa hai khung liên tiếp để tránh va chạm. Cứ sau 45 ms, một mã lặp lại được truyền đi.

Bảng 2. Định thời SIRC

Sự miêu tả	Điển hình	Min	Max
Xung Syn mức cao Xung Syn	2,4 mili giây	2,3 mili giây	2,6 mili giây
mức thấp Khoảng thời gian Bit	0,6 mili giây	0,55 mili giây	0,7 mili giây
0	1,2 mili giây	1,1 mili giây	1,3 mili giây
Bit 1 khoảng thời gian	1,8 mili giây	1,7 mili giây	1,9 mili giây
Thời gian nhận bản tin SIRC Thời gian bit xung	45 mili giây	-	-
của nhà cung cấp dịch vụ	25 µs	-	-

Ghi chú: *1* *Việc triển khai giao thức hồng ngoại dựa trên các thông số kỹ thuật SIRC miễn phí được tải xuống từ http://www.sbprojects.com.*

2 *ban 2 hiển thị tổng quan về dung sai độ rộng xung dữ liệu được sử dụng trong ghi chú ứng dụng này. Người dùng có thể chỉ định thời gian SIRC tối thiểu.*

2 Máy phát hồng ngoại

2.1 Cân nhắc phần cứng

TX-IR LED là bộ phát hồng ngoại được thiết kế cho các liên kết dữ liệu nối tiếp hồng ngoại và các ứng dụng điều khiển từ xa. Dữ liệu hiện tại được điều chế ở tần số sóng mang đã chọn (36 kHz hoặc 40 kHz như trong ví dụ) cung cấp giải pháp đơn chip đơn giản cho các ứng dụng truyền thông dữ liệu hồng ngoại và điều khiển từ xa.

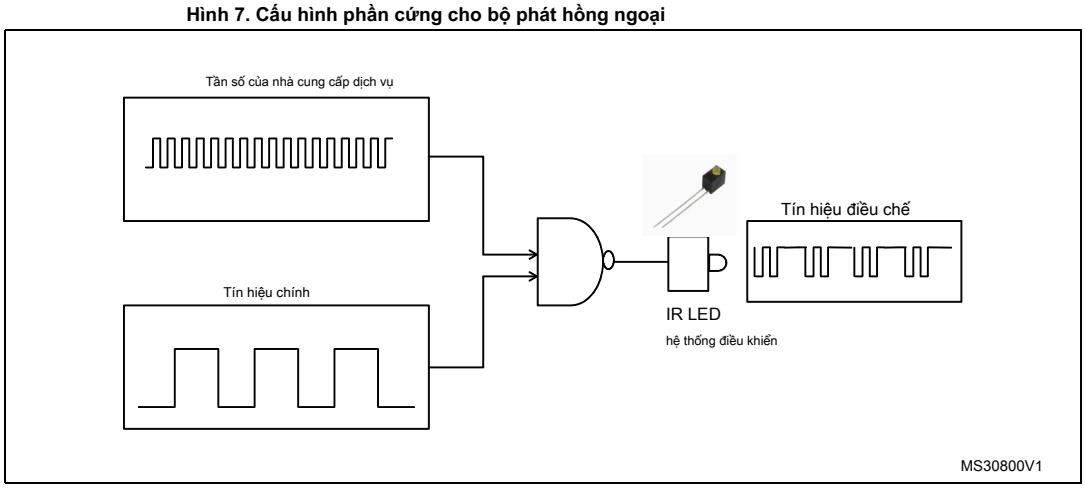
Giao diện hồng ngoại (IRTIM) để điều khiển từ xa khả dụng trên các thiết bị STM32F0xx, STM32F3xx và STM32L4xx. Nó có thể được sử dụng với đèn LED hồng ngoại để thực hiện chức năng điều khiển từ xa.

Giao diện kỹ thuật số IR được thiết kế để xuất tín hiệu kỹ thuật số tới mạch điều khiển diode hồng ngoại. Nó có thể xuất tín hiệu bằng cách sử dụng bất kỳ kiểu điều chế hiện có nào, kiểu điều chế phụ thuộc vào thuật toán phần mềm.

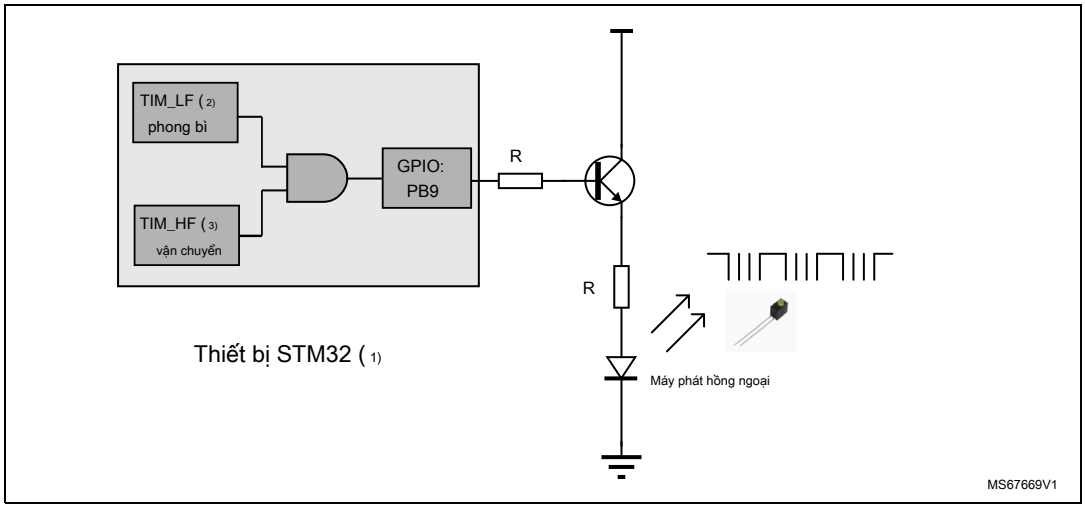
Giao diện IR rất dễ cấu hình và sử dụng hai tín hiệu được cung cấp bởi hai bộ định thời STM32 (TIM16 và TIM17 cho hầu hết các sản phẩm STM32, TIM15 và TIM16 cho các sản phẩm STM32L4x1 / 2/3).

TIM_HF (thường là TIM17) được sử dụng để cung cấp tần số sóng mang, trong khi TIM_LF (thường là TIM16) cung cấp tín hiệu thực tế được gửi đi.

Hình 7. Cấu hình phần cứng cho bộ phát hồng ngoại



Hình 8. Mô tả phần cứng



1. Trong các sản phẩm STM32L41x / L42x / L43x / L44x / L45x / L46x, chức năng IRTIM được liên kết với TIM15 và TIM16.

2. TIM16 trên hầu hết các sản phẩm.

3. TIM17 trên hầu hết các sản phẩm.

2,2 Máy phát hồng ngoại: giải pháp phổ quát

Giải pháp phát hồng ngoại dựa trên STM32 cho phép người dùng gửi IR đến các thiết bị thu khác nhau.

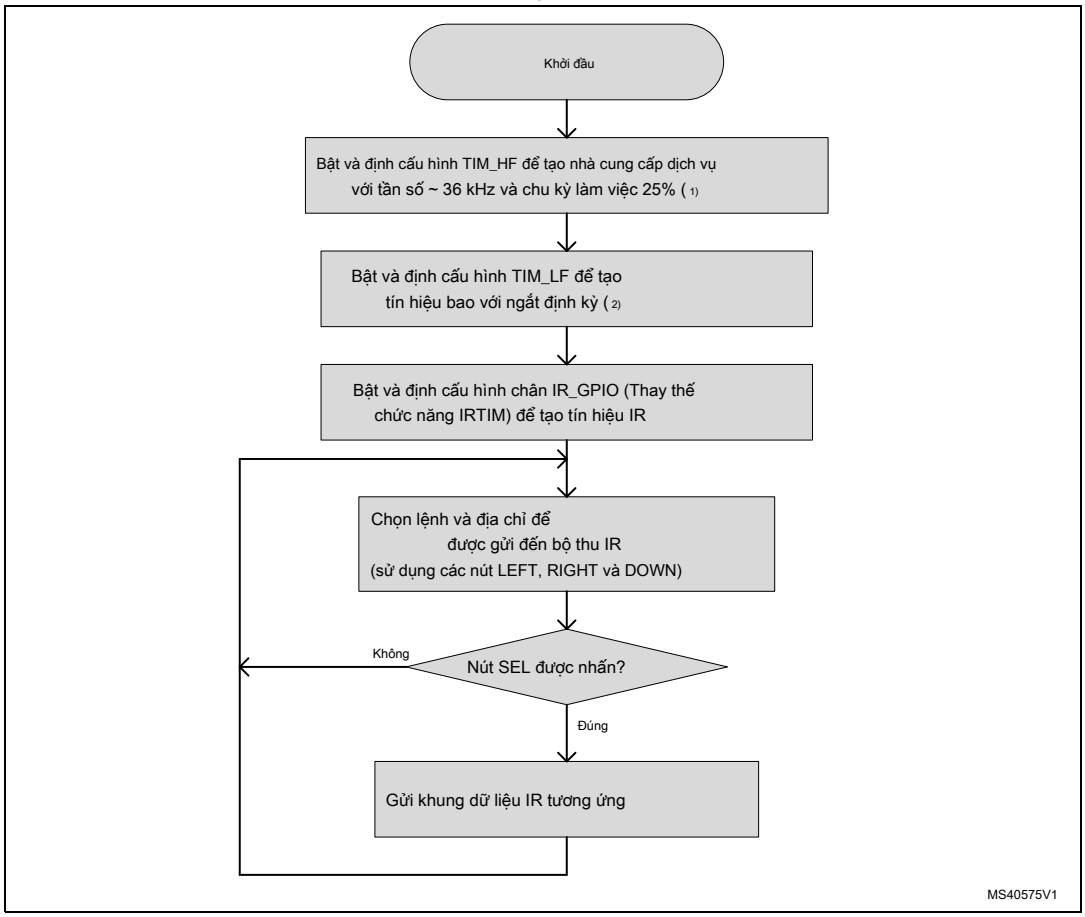
Giải pháp ứng dụng sử dụng bốn thiết bị ngoại vi:

- IRTIM: (Giao diện IR với bộ hẹn giờ) tạo ra tín hiệu IR bằng TIM_HF và TIM_LF
  - TIM_HF (TIM17 trên các sản phẩm STM32Fxxx): cung cấp tín hiệu sóng mang với tần số tùy thuộc vào đặc điểm giao thức
  - TIM-LF (TIM16 trên các sản phẩm STM32Fxxx): cung cấp tín hiệu chính được gửi (Khung RC5, Khung SIRC hoặc Khung giao thức khác)
- GPIO: (I / O mục đích chung) cung cấp I / O được kết nối với các nút của điều khiển từ xa và kết nối với IR-LED
- CLK: (bộ điều khiển đồng hồ) bật đồng hồ và cung cấp tần số đồng hồ chính xác cho bộ đếm thời gian

Để tạo tín hiệu điều khiển từ xa hồng ngoại, TIM_LF kênh 1 (TIMX_OC1) và TIM_HF kênh 1 (TIMX_OC1) phải được định cấu hình đúng cách để tạo ra dạng sóng chính xác. Tất cả các chế độ điều chế xung IR tiêu chuẩn có thể thu được bằng cách lập trình hai kênh so sánh đầu ra của bộ định thời. Chức năng hồng ngoại được xuất trên chân TIM_IR. Việc kích hoạt chức năng này được thực hiện thông qua thanh ghi GPIOx_AFRx bằng cách cho phép bit chức năng thay thế liên quan. Tài liệu hướng dẫn tham khảo cũng đề cập đến bit I2C_PB9_FMP trong thanh ghi SYSCFG_CFGR1 để kích hoạt khả năng tản dòng cao để điều khiển trực tiếp đèn LED hồng ngoại. Với mạch được sử dụng trong bảng EVAL, bit này sẽ vẫn được đặt lại.

Luồng chương trình chính được hiển thị trong *Hình 9* .

Hình 9. Lưu đồ vòng lặp chính



1. Tần số sóng mang cụ thể cho từng giao thức IR (trong ví dụ là 36 kHz cho RC5 và 40 kHz cho SIRC).

2. Tín hiệu bao phủ là cụ thể cho từng giao thức IR (trong ví dụ là 889 µs cho RC5 và 600 µs cho SIRC).

Mục tiêu của TIM_HF là tạo ra tín hiệu sóng mang.

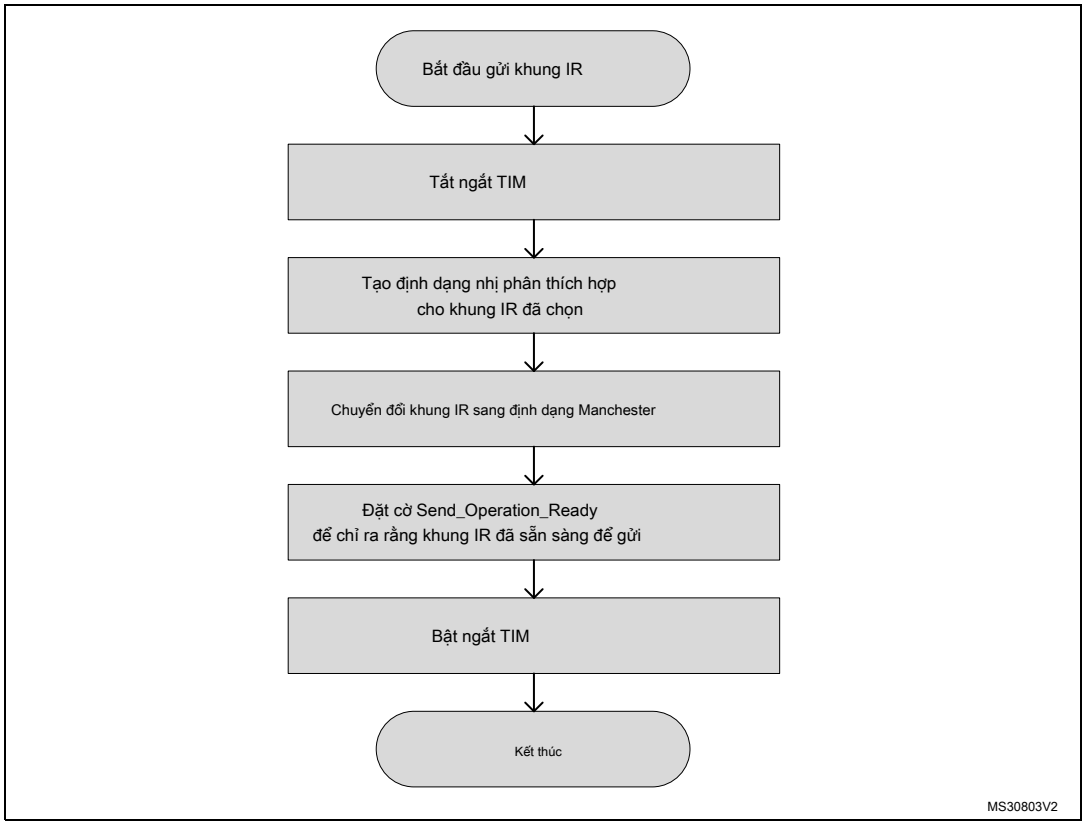
TIM_HF_Period = (SystemCoreClock / FrequencyCarrier) - 1

TIM_LF được sử dụng để tạo ra tín hiệu bao.

TIM_LF_Period = (SystemCoreClock / FrequencyEnvelop) - 1

Khi các mô-đun được khởi tạo (IRTIM, trường khung), ứng dụng sẽ đợi nút SEL được nhấn để gửi dữ liệu IR. *Hình 10* cho thấy *gửi khung* sơ đồ.

Hình 10. Gửi lưu đồ khung IR

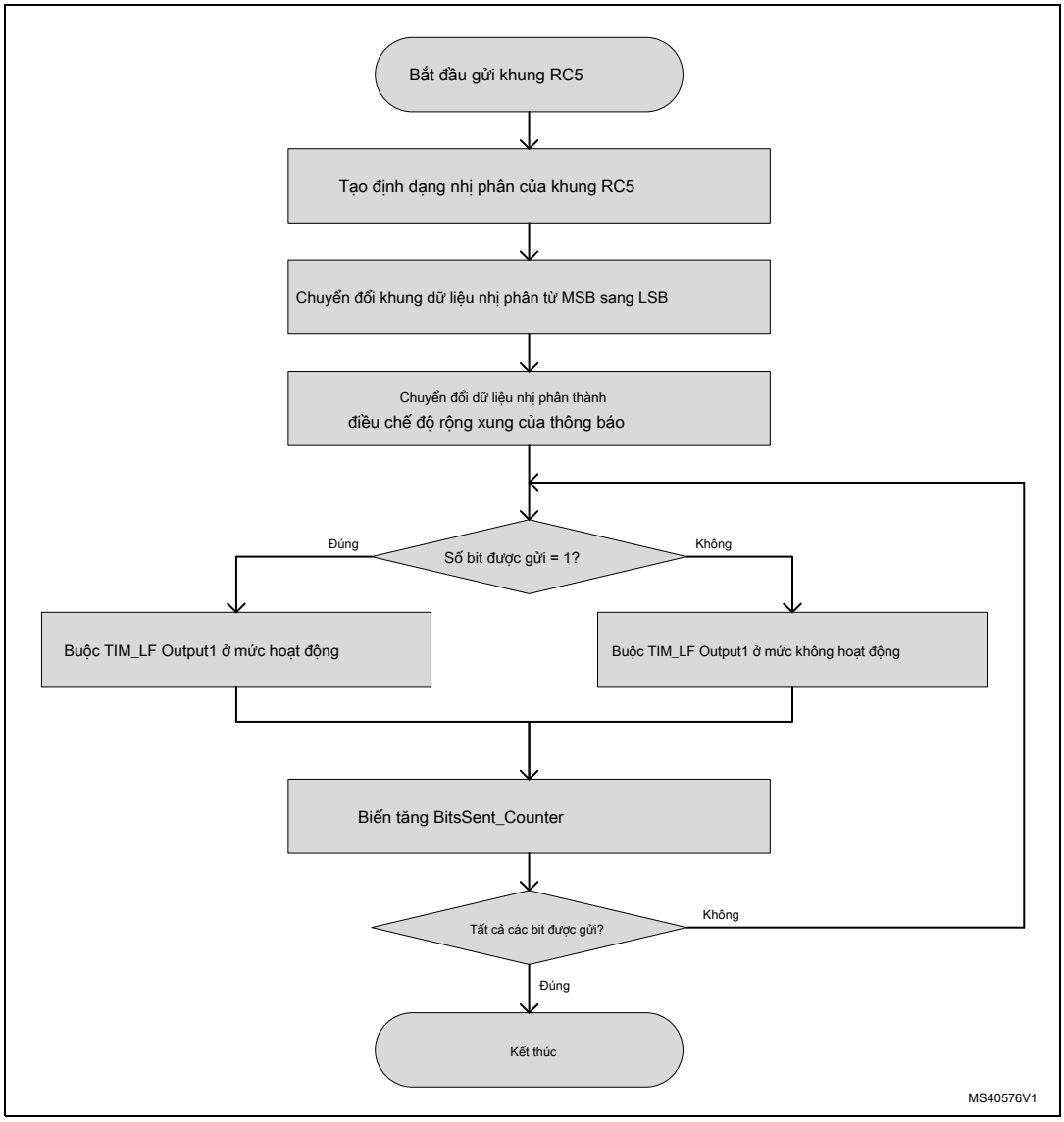


2.2.1 Giải pháp mã hóa RC5

Cơ chế mã hóa RC5

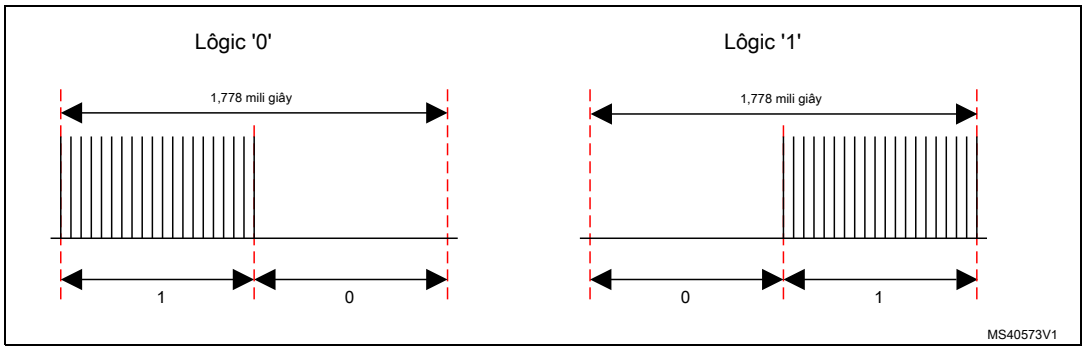
Hình 11 cho biết cách tạo khung RC5. Lưu đồ được mô tả được gọi trong quy trình ngắt cập nhật TIM16.

Hình 11. Lưu đồ khung gửi RC5



Trong mã hóa Manchester, logic "0" và logic "1" được biểu thị tương ứng bằng chuyển tiếp 0 sang 1 và chuyển tiếp 1 sang 0 ở trung tâm của chuỗi, như được tóm tắt trực quan trong *Hình 12*, trong đó các đường đứt nét màu đỏ biểu thị các khoảng thời gian.

Hình 12. Các bit mã hóa Manchester



Thư viện mã hóa RC5

Trình điều khiển bộ mã hóa RC5 dựa trên các chức năng sau.

RC5_Encode_Init ()

Chức năng này khởi tạo các thiết bị ngoại vi khác nhau (GPIO, TIMER, ...).

RC5_Encode_SendFrame ()

Hàm này gửi khung RC5 định dạng Manchester.

RC5_Encode_SignalGenerate ()

Chức năng này tạo ra tín hiệu khung bằng cách giám sát mức đầu ra của TIM_LF. Nó được gọi trong quá trình ngắt cập nhật TIM_LF để xử lý tín hiệu đầu ra.

2.2.2 Cách sử dụng trình điều khiển bộ mã hóa RC5

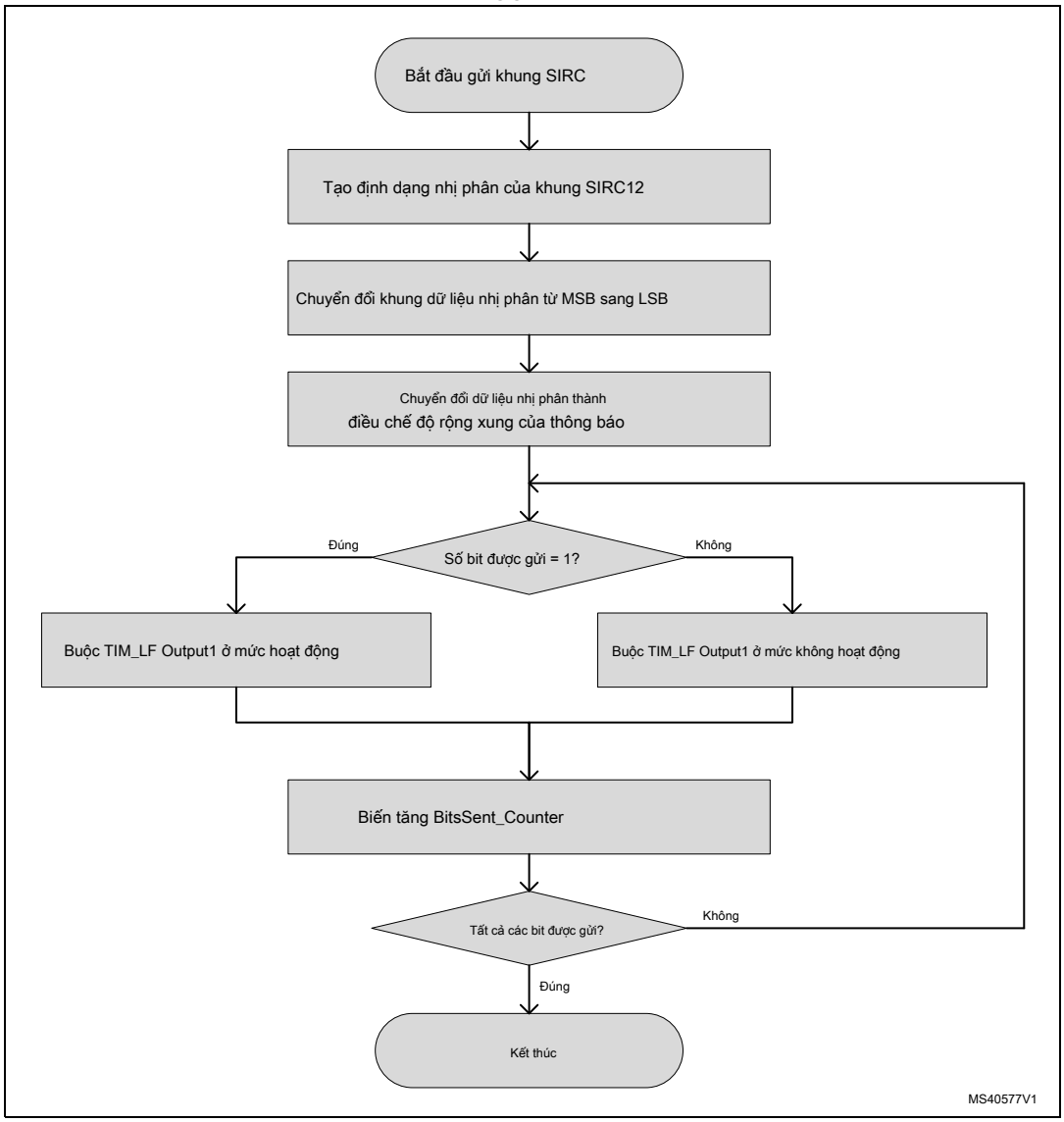
Để sử dụng trình điều khiển bộ mã hóa RC5, hãy tiến hành như sau.

- Gọi hàm RC5_Encode_Init () để định cấu hình bộ đếm thời gian và tài nguyên phần cứng GPIO cần thiết cho mã hóa RC5.
- Gọi hàm RC5_Encode_SendFrame () để gửi khung RC5.
- TIM_LF Các ngắt cập nhật được sử dụng để mã hóa khung RC5 trong điều chế độ rộng xung.

2.2.3 Giải pháp mã hóa SIRC

Cơ chế mã hóa SIRC

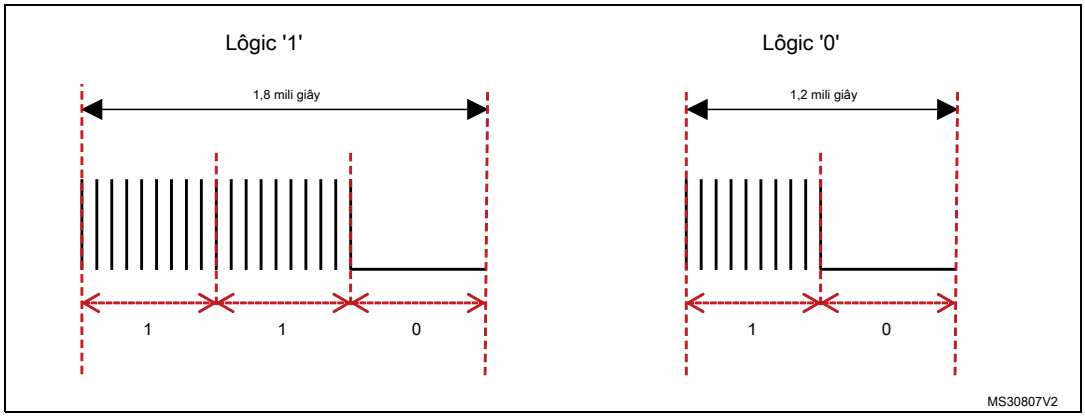
Hình 13. Lưu đồ khung gửi SIRC



Sau khi tạo khung định dạng nhị phân, mỗi bit logic được chuyển đổi thành tổ hợp của "0" s và "1", đại diện cho định dạng điều chế độ rộng xung.

"1" logic mất 1,8 ms để truyền, với 1,2 ms ở mức cao và 600 µs ở mức thấp. Đối với "0" logic, phải mất 1,2 ms, với 600 µs ở mức cao và 600 µs ở mức thấp (tham khảo
Hình 14 , nơi các đường đứt nét màu đỏ đã được thêm vào để biểu thị các khoảng thời gian). Thời gian gốc được chọn là 600 µs, vì vậy logic "1" được chuyển đổi thành 110 và logic "0" thành 10.

Hình 14. Chuyển đổi bit logic SIRC



Thư viện mã hóa SIRC

Trình điều khiển bộ mã hóa SIRC dựa trên các chức năng sau.

SIRC_Encode_Init ()

Chức năng này khởi tạo các thiết bị ngoại vi khác nhau (GPIO, TIMER, NVIC, ...).

SIRC_Encode_SendFrame ()

Hàm này gửi điều chế độ rộng xung của định dạng Khung SIRC12.

SIRC_Encode_SignalGenerate ()

Chức năng này tạo ra tín hiệu khung bằng cách giám sát mức đầu ra của TIM_LF. Nó được gọi trong ngắt cập nhật TIM_LF để xử lý tín hiệu đầu ra.

2.2.4 Cách sử dụng trình điều khiển bộ mã hóa SIRC

Để sử dụng trình điều khiển bộ mã hóa SIRC, hãy tiến hành như sau.

- Gọi hàm SIRC_Encode_Init () để định cấu hình bộ hẹn giờ và tài nguyên phần cứng GPIO cần thiết cho mã hóa SIRC.
- Gọi hàm SIRC_Encode_SendFrame () để gửi khung SIRC.
- TIM_LF Các ngắt cập nhật được sử dụng để mã hóa khung SIRC trong điều chế độ rộng xung.

3 Máy thu hồng ngoại

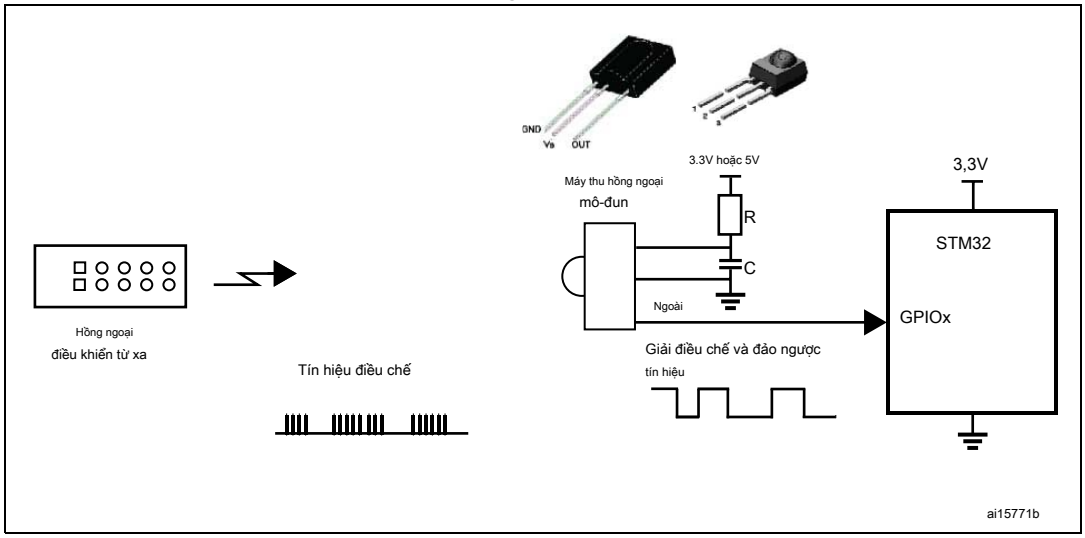
3.1 Cân nhắc phần cứng

Để cải thiện khả năng loại bỏ tiếng ồn, các xung IR được điều chế ở khoảng 36 kHz, 38 kHz hoặc 40 kHz. Cách dễ nhất để nhận các xung này là sử dụng mô-đun bộ thu / giải điều chế IR tích hợp như TSOP1736 (phiên bản nguồn 5 V), TSOP34836 (phiên bản nguồn cung cấp 3,3 V) hoặc các số bộ phận tương đương khác (xem *Hình 15).*

Đây là các thiết bị 3 chân nhận chùm tia hồng ngoại và xuất ra dòng bit đã được giải điều chế trên chân đầu ra được kết nối trực tiếp với một trong các chân GPIO của vi điều khiển STM32 hoặc các kênh GP-Timers Input Capture. Nếu TSOP1736 được sử dụng, GPIO được chọn phải là Dung lượng năm volt (FT). Đầu ra của mô-đun IR được đảo ngược so với dữ liệu được truyền (dữ liệu ở mức cao nhàn rỗi và logic "0" trở thành logic "1" và ngược lại).

Ghi chú: *Mô-đun IR cần hai thành phần bên ngoài: tụ điện và điện trở (tham khảo biểu dữ liệu mô-đun IR liên quan để biết các giá trị của chúng).*

Hình 15. Cấu hình phần cứng



3.2 Giải pháp phổ quát: triển khai phần mềm bằng cách sử dụng GP-Timer được định cấu hình ở chế độ đầu vào PWM

Mỗi giao thức hồng ngoại có thể được giải mã bằng cách sử dụng một trong các thiết bị ngoại vi hẹn giờ được nhúng trong vi điều khiển STM32. Bộ đếm thời gian này có thể được cấu hình ở chế độ đầu vào PWM và được sử dụng để lấy mẫu các bit khung hồng ngoại. Chức năng chụp đầu vào bộ hẹn giờ đang hoạt động trên các cạnh có cực tính đối diện.

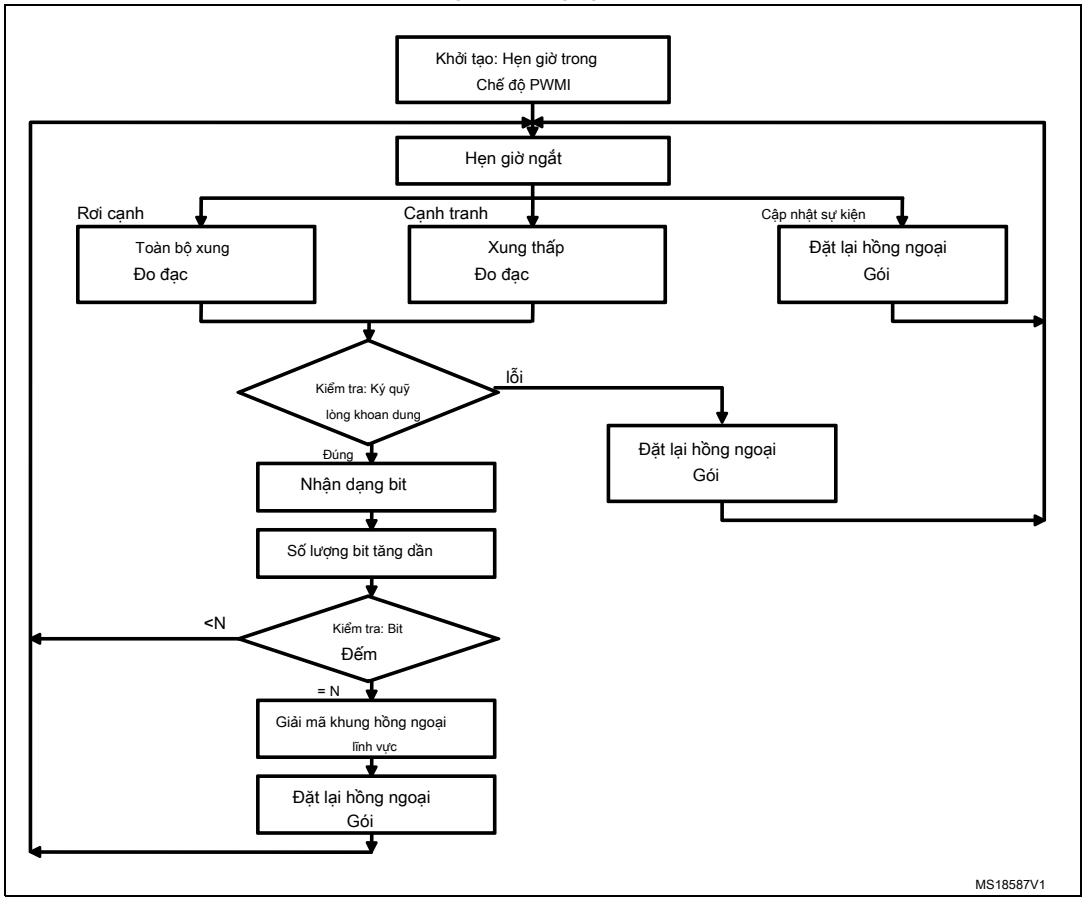
Bộ định thời tạo ra ba kiểu ngắt

- Ngắt tại mỗi cạnh rơi: điều này có thể được sử dụng để đo toàn bộ xung (khoảng thời gian giữa hai cạnh rơi liên tiếp)
- Ngắt ở mỗi cạnh tăng: điều này có thể được sử dụng để đo xung thấp (khoảng thời gian giữa cạnh giảm và cạnh tăng)
- Sự kiện cập nhật: điều này được sử dụng để đặt gói hồng ngoại vào trạng thái mặc định (đếm bit, dữ liệu và trạng thái) khi bộ đếm thời gian bị tràn

Xung thấp và toàn bộ thời gian xung được sử dụng để xác định giá trị bit. Nếu khoảng thời gian nằm trong phạm vi dung sai của thời gian bit, người ta xác định giá trị bit (Logic0, Logic1 hoặc Header).

Lưu đồ dưới đây cho ta một cái nhìn tổng quan về quy trình giải mã hồng ngoại.

Hình 16. Lưu đồ giải mã hồng ngoại



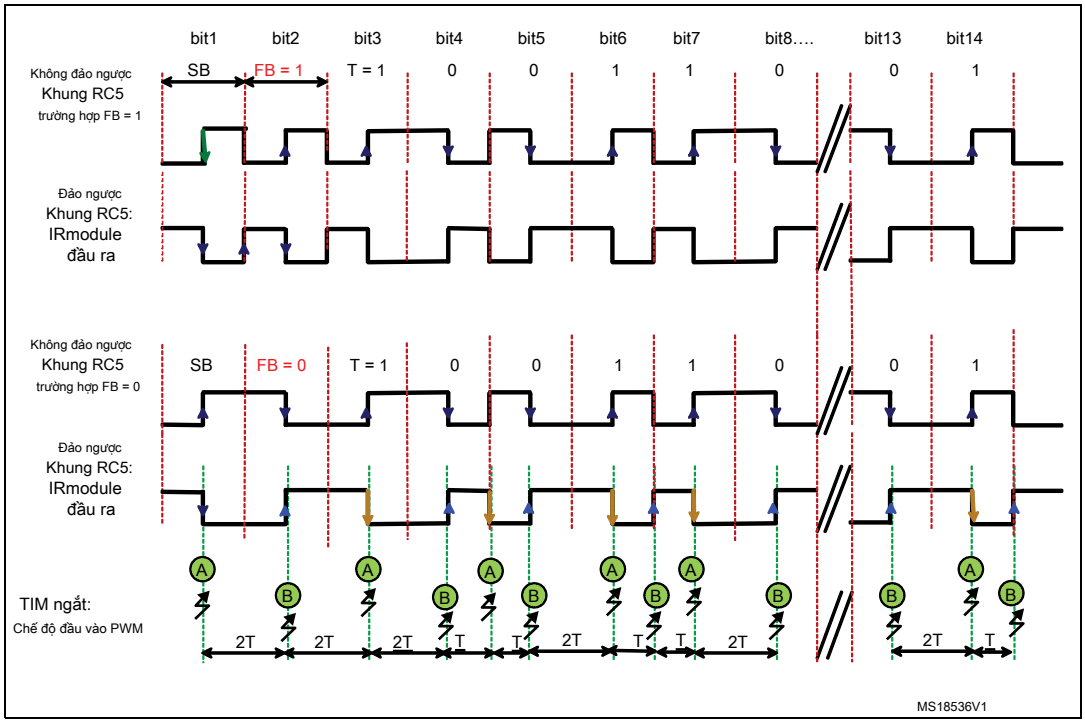
3,3 Giải pháp giao thức RC5

3.3.1 Cơ chế giải mã khung RC5

Hình 17 hiển thị cách nhận khung RC5. Một trong những bộ đếm thời gian mục đích chung của vi điều khiển STM32 được cấu hình ở chế độ đầu vào PWM.

Đầu vào này có thể nắm bắt giá trị bộ đếm thời gian hiện tại ở cả cạnh giảm và cạnh lên cũng như tạo ra một ngắt trên cả hai cạnh. Tính năng này giúp bạn dễ dàng đo thời gian cao và thấp của xung RC5.

Hình 17. Cơ chế giải mã khung RC5



- **Sự kiện ngắt TIMER:** rơi mép

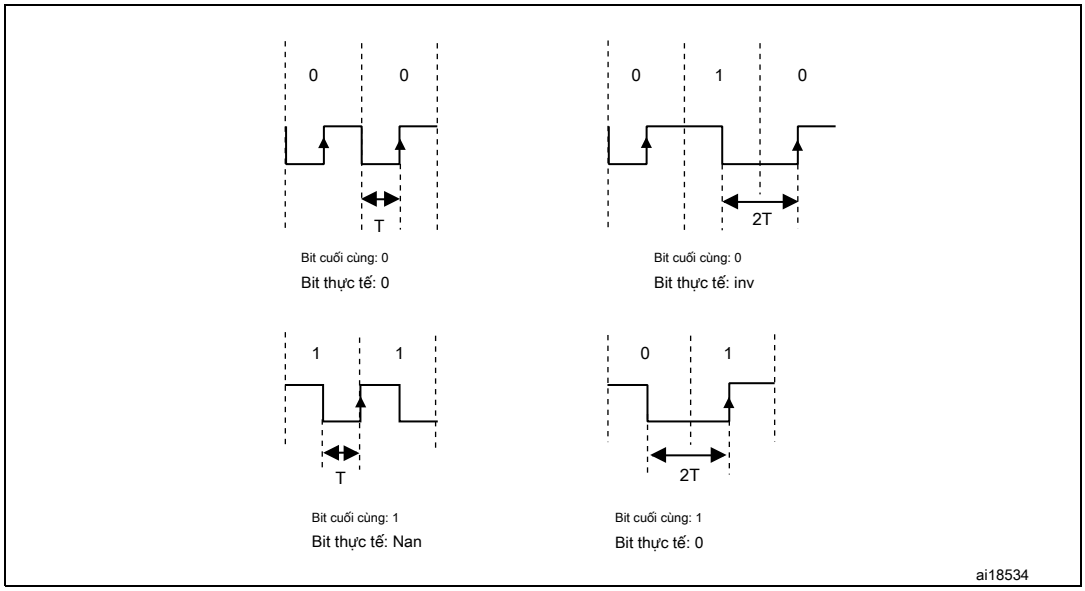
  A: ngắt TIMER được sử dụng để đo khoảng thời gian giữa hai cạnh rơi liên tiếp (khoảng thời gian một hoặc một nửa xung).

- **Sự kiện ngắt TIMER:** cạnh tăng

  B: TIMER được sử dụng để đo khoảng thời gian giữa các cạnh giảm và tăng (thời gian xung thấp).

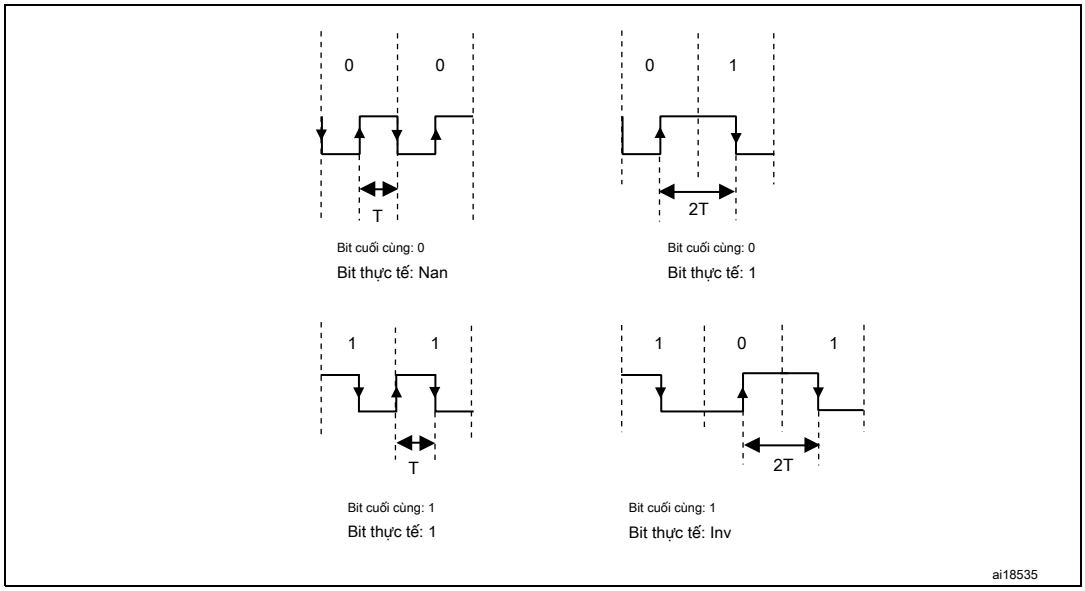
Hai khoảng thời gian được sử dụng để xác định giá trị bit. Mỗi giá trị bit được xác định liên quan đến bit cuối cùng.

Hình 18. Xác định bit bằng cạnh lên: xung thấp



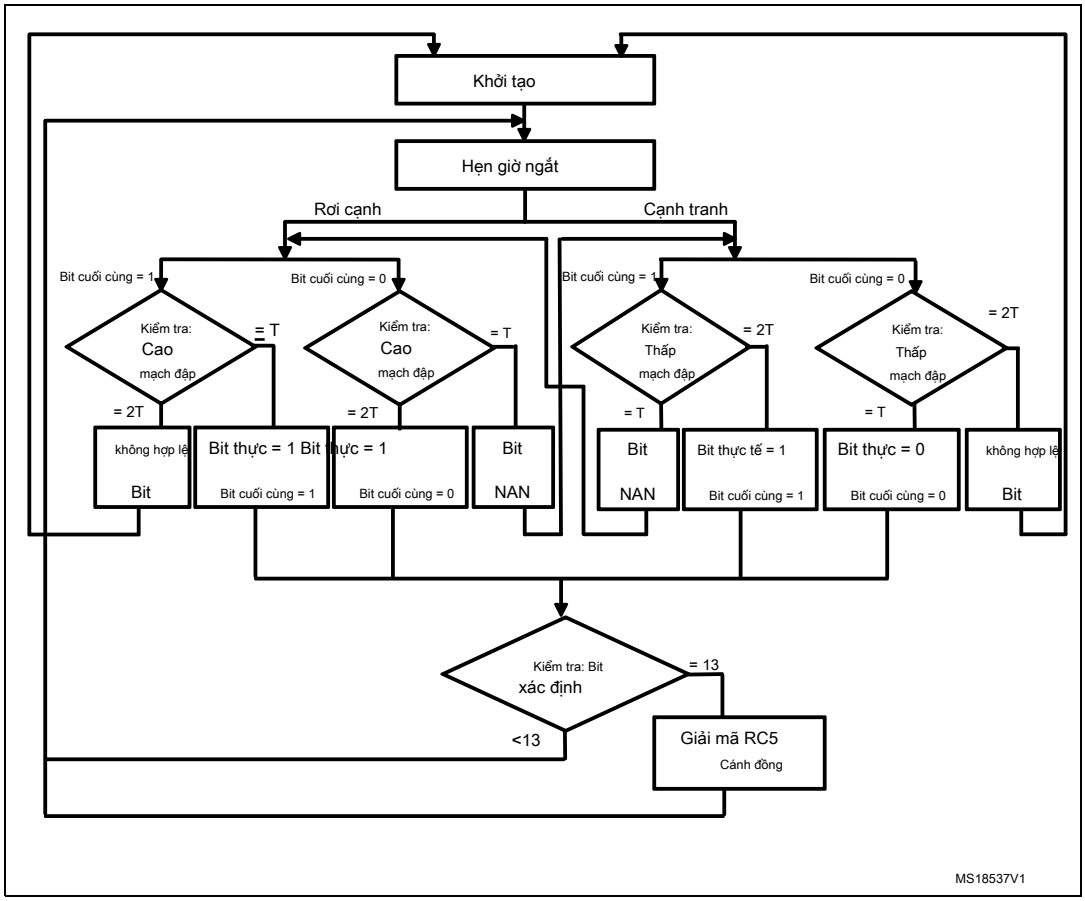
- Nếu thời lượng xung thấp bằng T và bit cuối cùng được xác định là "0", thì thực tế bit là ***lôgic0.***
- Nếu thời lượng xung thấp bằng 2T và bit cuối cùng được xác định là "0", thì bit thực là ***Inv (*** **trường hợp không hợp lệ:** trường hợp này không thể được phát hành).
- Nếu thời lượng xung thấp bằng T và bit cuối cùng được xác định là "1", thì bit thực là ***Nan (*** **không bit:** bit này được xác định tại cạnh rơi tiếp theo).
- Nếu thời lượng xung thấp bằng 2T và bit cuối cùng được xác định là "1", thì thực tế bit là ***lôgic0.***

Hình 19. Xác định bit bằng cạnh rơi: xung cao



- Nếu thời lượng xung cao bằng T và bit cuối cùng được xác định là "0", thì bit thực là ***Nan (*** **không bit:** bit này được xác định tại cạnh lên tiếp theo).
- Nếu thời lượng xung cao bằng 2T và bit cuối cùng được xác định là "0", thì bit thực là ***logic1.***
- Nếu thời lượng xung cao bằng T và bit cuối cùng được xác định là "1", thì thực tế bit là ***logic1.***
- Nếu thời lượng xung cao bằng 2T và bit cuối cùng được xác định là "1", thì bit thực là ***Inv (*** **trường hợp không hợp lệ:** trường hợp này không thể được phát hành).

Hình 20. Lưu đồ giải pháp RC5



3.3.2 Thư viện giải mã RC5

Trình điều khiển RC5 rất đơn giản để sử dụng.

RC5_Init ()

Chức năng này khởi tạo các thiết bị ngoại vi khác nhau (GPIO, TIMER, ...).

RC5_ResetPacket ()

Chức năng này đặt cấu trúc gói ở trạng thái mặc định. Hàm này chủ yếu được gọi trong hàm HAL_TIM_PeriodElapsedCallback. Nó xảy ra ở mỗi lần tràn TIMER, để thiết lập lại gói RC5.

RC5_Decode (RC5_Frame_TypeDef * rc5_frame)

Chức năng này nhằm mục đích được gọi trong ứng dụng người dùng. Nó giải mã các tin nhắn nhận được RC5. Cấu trúc sau đây chứa các giá trị khác nhau của khung RC5.

```
typedef struct
{
    __IO uint8_t FieldBit; / * Trường bit trường * / __IO uint8_t ToggleBit; / *
    Chuyển đổi trường bit * /
```

```
    __IO uint8_t Địa chỉ;          / * Trường địa chỉ          * /
    __IO uint8_t Lệnh;             / * Trường lệnh             * /
} RC5_Frame_TypeDef;
```

RC5_Decode () được thực thi khi cờ RC5FrameReceive bằng YES.

RC5_DeInit ()

Chức năng này khử khởi tạo các thiết bị ngoại vi khác nhau (GPIO, TIMER ...).

HAL_TIM_CaptureCallback ()

Chức năng này xử lý ngắt TIM Capture Compare.

- **Sự kiện cạnh rơi bộ hẹn giờ:** giá trị này được sử dụng để đo khoảng thời gian giữa hai cạnh rơi liên tiếp (toàn bộ khoảng thời gian xung).
- **Sự kiện Hẹn giờ Rising Edge:** điều này được sử dụng để đo khoảng thời gian giữa các cạnh giảm và tăng (thời lượng xung thấp).

Khoảng thời gian xung thấp và toàn bộ thời lượng xung được sử dụng để xác định giá trị bit. Mỗi giá trị bit được xác định liên quan đến bit cuối cùng.

HAL_TIM_PeriodElapsedCallback ()

Chức năng này xử lý ngắt Cập nhật TIM.

- **Cập nhật sự kiện (sự kiện hết giờ):** điều này sẽ đặt lại gói RC5. Tràn bộ hẹn giờ được đặt thành 3,7 ms.

3.3.3 Cách sử dụng trình điều khiển bộ giải mã RC5

Để sử dụng trình điều khiển bộ giải mã RC5, hãy tiến hành như sau.

- Gọi hàm RC5_Init () để định cấu hình bộ đếm thời gian và tài nguyên phần cứng GPIO cần thiết cho việc giải mã RC5.
- Ngắt TIM2 Capture So sánh và Cập nhật được sử dụng để giải mã khung RC5, nếu một khung được nhận đúng, một biến toàn cục "RC5FrameReceive" được đặt để thông báo cho ứng dụng.
- Sau đó, ứng dụng sẽ gọi hàm RC5_Decode () để truy xuất khung RC5 đã nhận.

Mã ví dụ

```
# bao gồm "rc5_decode.h"

/ * IR_FRAME sẽ giữ khung RC5 (Địa chỉ, Lệnh, ...) * / RC5_Frame_TypeDef IR_FRAME;

/ * Khởi tạo trình điều khiển RC5 * / RC5_Init ();

trong khi (1)
{
```

```
    / * Giải mã khung RC5 nhận được và lưu trữ nó trong biến IR_FRAME
    * /
    RC5_Decode (& IR_FRAME);

    / * Tại đây thêm mã sẽ xử lý khung vừa nhận được, tức là
        Biến IR_FRAME, nếu không, nó sẽ bị ghi đè bởi khung tiếp theo
        * /
    . . .
}
```

Ghi chú: *1* *TIMx_IRQHandler ISR được mã hóa trong stm32f0xx_it.c, stm32g0xx_it.c hoặc stm32f3xx_it.c*

- Nếu một hoặc cả hai ngắt được sử dụng trong ứng dụng, phải đặc biệt chú ý:
  - thêm mã ứng dụng trong các ISR này hoặc
  - sao chép nội dung của các ISR này trong mã ứng dụng.

2 *Người dùng có thể dễ dàng điều chỉnh ứng dụng này cho phù hợp với phần cứng bằng cách sử dụng các khai báo xác định khác nhau trong tệp "ir_common.h". Tham khảo bản số 3 .*

Bảng 3. Ví dụ về cách thực hiện

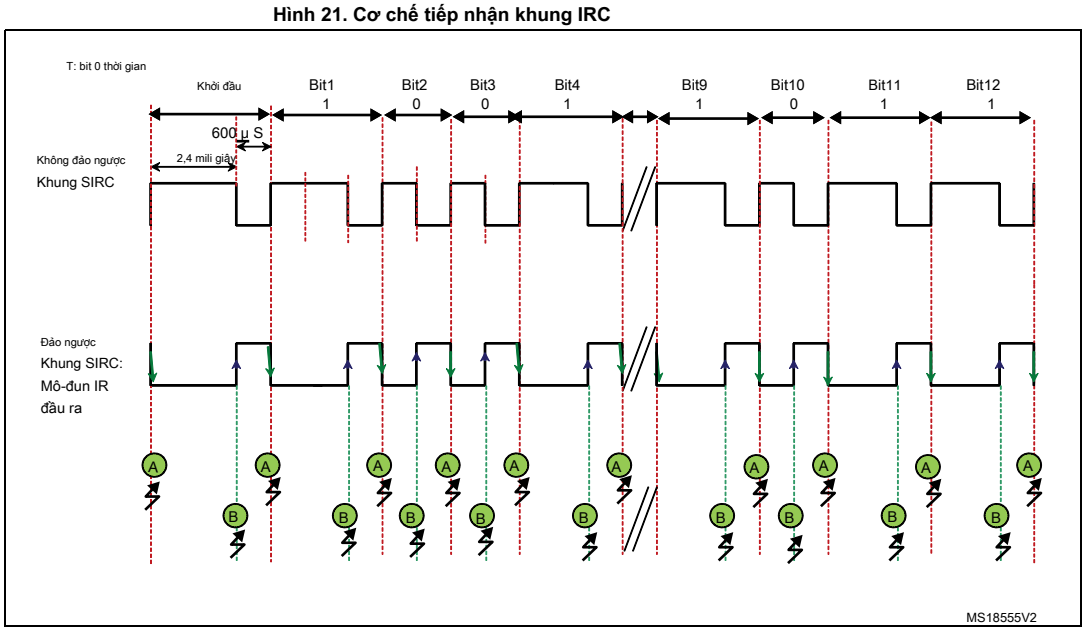
Xác định tên	Sự miêu tả	STM32F07x Dòng STM32G0	STM32F30x	STM32F37x
# xác định IR_TIM	Bộ hẹn giờ được sử dụng để giải mã IR (1)	TIM3	TIM1	TIM3
# định nghĩa TIM_PRESCALER	Bộ định mức TIM Tham số này là tính toán để có 1 µs làm cơ sở thời gian. Tần số TIM (trong MHz) / (bộ đếm trước + 1)	47	71	71
# xác định IR_TIM_CLK	Đồng hồ APB của bộ đếm thời gian đã sử dụng	__HAL_RCC_TIM3_ CLK_ENABLE	__HAL_RCC_TIM1_ CLK_ENABLE	__HAL_RCC_TIM3_ CLK_ENABLE
# xác định IR_TIM_IRQn	IR TIM IRQ	TIM3_IRQn	TIM1_CC_IRQn	TIM3_IRQn
# định nghĩa IR_TIM_Channel	Kênh IR TIM	TIM_CHANNEL_1	TIM_CHANNEL_2	TIM_CHANNEL_2
# định nghĩa IR_GPIO_PORT	Cổng mà đầu ra IR là kết nối (1)	GPIOC	GPIOA	GPIOB
# định nghĩa IR_GPIO_PORT_CLK	Cổng đồng hồ GPIO chân hồng ngoại	__HAL_RCC_GPIOC _CLK_ENABLE	__HAL_RCC_GPIOA _CLK_ENABLE	__HAL_RCC_GPIOB _CLK_ENABLE
# xác định IR_GPIO_PIN	Ghim cho ai IR được kết nối (1)	GPIO_PIN_6	GPIO_PIN_9	GPIO_PIN_5

1. Để biết thêm chi tiết về các tài nguyên STM32 hiện có, hãy tham khảo bảng dữ liệu sản phẩm.

3,4 Giải pháp điều khiển hồng ngoại SIRC

3.4.1 Triển khai phần mềm

Hình 21. Cơ chế tiếp nhận khung IRC

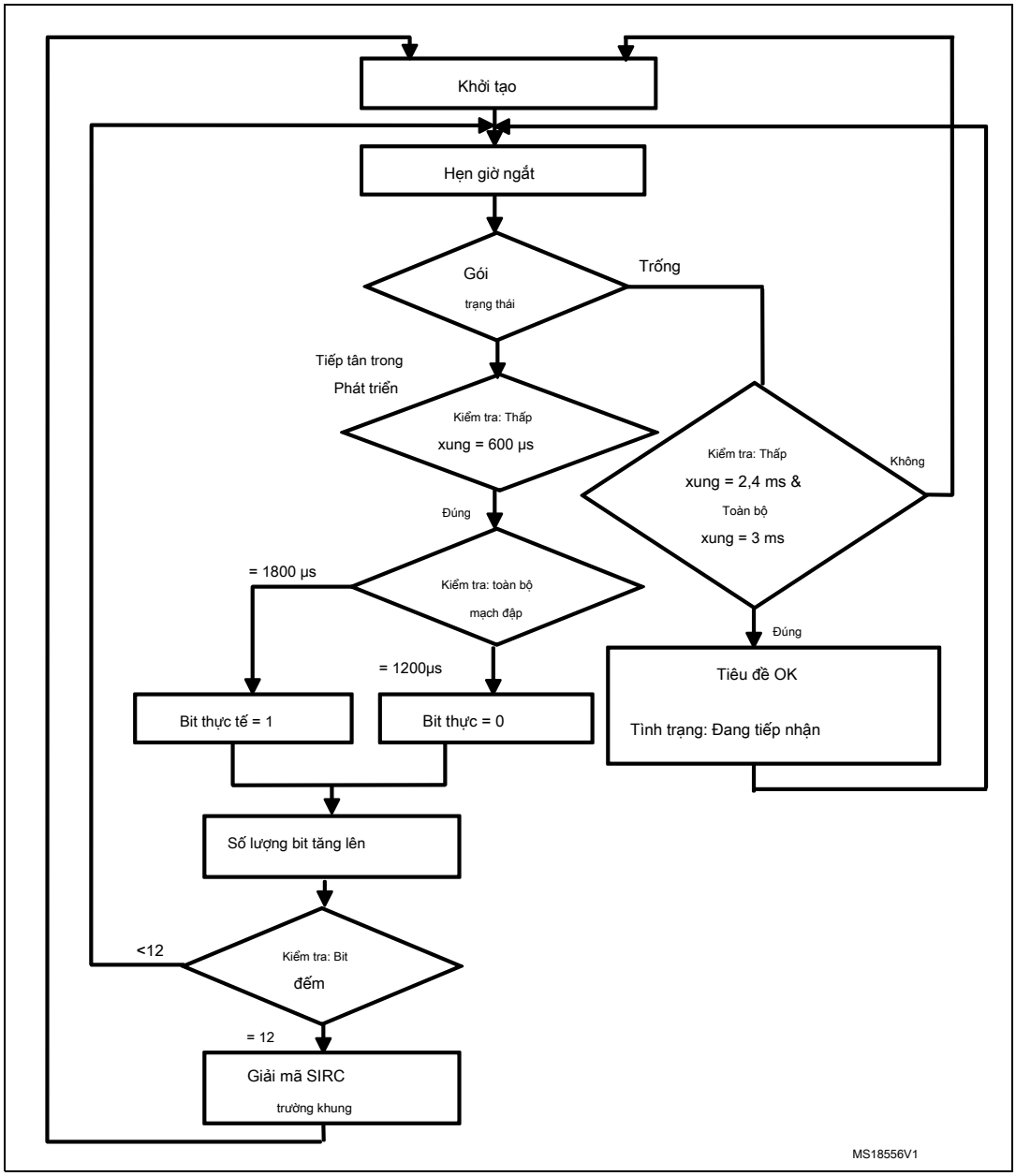


TIMER ngắt: ở chế độ đầu vào PWM

TIMER được sử dụng để lấy mẫu các bit khác nhau của khung SIRC. Giá trị bộ đếm thời gian hiện tại được ghi lại cả ở cạnh giảm và cạnh lên, và một ngắt được tạo ra trên cả hai cạnh. Tính năng này giúp bạn dễ dàng đo toàn bộ xung SIRC và thời gian thấp.

- Nếu chu kỳ đo được bằng T = 1200 µs và khoảng thời gian xung thấp bằng T / 2 = 600 µs thì bit là ***logic "0".***
- Nếu chu kỳ đo được bằng 3T / 2 = 1800 µs và thời gian xung thấp bằng T = 1200 µs thì bit là ***logic "1".***
- Nếu toàn bộ chu kỳ đo được bằng 3000 µs và khoảng thời gian xung thấp bằng 2400 µs, thì bit là " ***bắt đầu bit ".***

Hình 22. Lưu đồ giải pháp SIRC



3.4.2 Thư viện SIRC

SIRC_Init ()

Chức năng này khởi tạo các thiết bị ngoại vi khác nhau được sử dụng cho giao thức SIRC.

SIRC_Decode (SIRC_Frame_TypeDef * sirc_frame)

Chức năng này nhằm mục đích được gọi trong ứng dụng người dùng. Nó giải mã các tin nhắn SIRC nhận được. Nó có cấu trúc như một tham số chứa các giá trị khác nhau của khung IR.

```
typedef struct
{
    __IO uint8_t Lệnh;                  / * Trường lệnh          * /
    __IO uint8_t Địa chỉ;               / * Trường địa chỉ       * /
} SIRC_Frame_TypeDef;
```

SIRC_decode () phải được thực thi khi cờ IRFrameReceive bằng YES.

SIRC_ResetPacket ()

Chức năng này đặt gói IR về trạng thái mặc định. Hàm này được gọi trong quy trình TIM2_IRQHandler. Nó xảy ra mỗi lần tràn bộ đếm thời gian để đặt lại gói IR.

SIRC_DeInit ()

Chức năng này khử khởi tạo các thiết bị ngoại vi khác nhau được sử dụng cho giao thức SIRC.

HAL_TIM_IC_CaptureCallback

Chức năng này xử lý ngắt TIM Capture Compare.

- **Sự kiện cạnh rơi bộ hẹn giờ:** điều này được sử dụng để đo các chu kỳ khác nhau giữa hai cạnh rơi liên tiếp để xác định các bit khung.
- **Sự kiện Hẹn giờ Rising Edge:** điều này được sử dụng để đo khoảng thời gian giữa các cạnh giảm và tăng (thời lượng xung thấp).

Giá trị bit được xác định từ hai khoảng thời gian này.

HAL_TIM_PeriodElapsedCallback ()

Chức năng này xử lý ngắt Cập nhật TIM.

- **Cập nhật sự kiện (sự kiện hết giờ):** điều này sẽ đặt lại gói RC5. Tràn bộ hẹn giờ được đặt thành 4 ms.

3.4.3 Cách sử dụng trình điều khiển bộ giải mã SIRC

Để sử dụng trình điều khiển bộ giải mã SIRC, hãy tiến hành như sau.

- Ngắt TIM2 Capture So sánh và Cập nhật được sử dụng để giải mã khung IR. Nếu một khung được nhận đúng, một biến toàn cục "IRFrameReceive" sẽ được thiết lập để thông báo cho ứng dụng.
- Sau đó, ứng dụng sẽ gọi hàm SIRC_Decode () để truy xuất khung IR đã nhận.
- Người dùng có thể dễ dàng điều chỉnh trình điều khiển này cho bất kỳ giao thức hồng ngoại nào khác bằng cách điều chỉnh đơn giản các định nghĩa từ sirc_decode.h với đặc tả giao thức hồng ngoại (Bit Duration, Header Duration, Marge Tolerance, Number of bits ...) và các bảng lệnh và thiết bị.

Mã ví dụ

```
# bao gồm "sirc_decode.h"

/ * SIRC_FRAME sẽ giữ khung SIRC (Địa chỉ, Lệnh, ...) * / SIRC_Frame_TypeDef SIRC_FRAME;

/ * Khởi tạo trình điều khiển SIRC * / SIRC_Init ();

trong khi (1)
    {
    / * Giải mã khung SIRC đã nhận và lưu trữ trong biến SIRC_FRAME * / SIRC_Decode (& SIRC_FRAME);

    / * Tại đây thêm mã sẽ xử lý khung vừa nhận được, tức là
         Biến SIRC_FRAME, nếu không, nó sẽ bị ghi đè bởi khung tiếp theo * /

    . . .
    }
```

Ghi chú: *1* *TIMx_IRQHandler ISR được mã hóa trong trình điều khiển stm32f0xx_it.c, stm32g0xx_it.c hoặc stm32f3xx_it.c.*

- Nếu một hoặc cả hai ngắt được sử dụng trong ứng dụng, phải đặc biệt chú ý:
  - thêm mã ứng dụng của bạn vào các ISR này hoặc
  - sao chép nội dung của các ISR này trong mã ứng dụng của bạn.

2 *Người dùng có thể dễ dàng điều chỉnh ứng dụng này cho phù hợp với phần cứng bằng cách sử dụng các khai báo xác định khác nhau trong tệp "ir_common.h".*

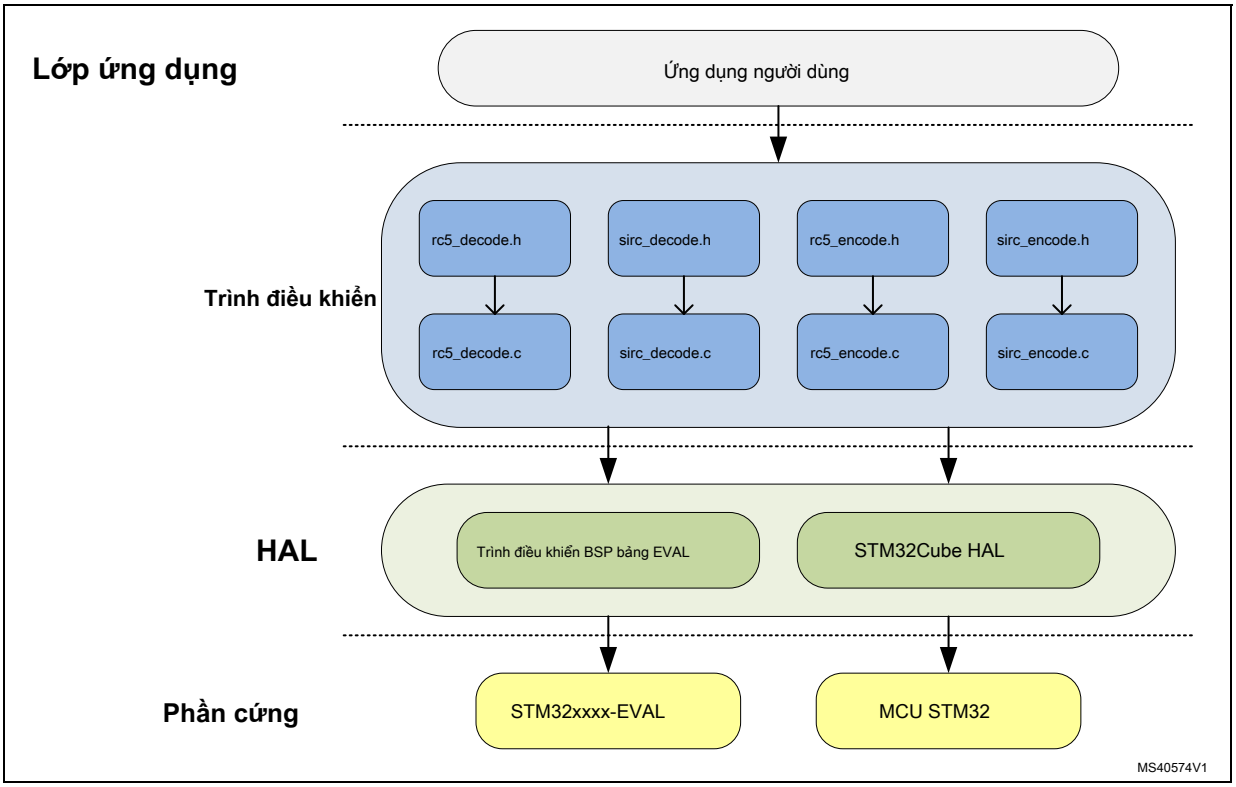
Bảng 4. Ví dụ về cách thực hiện

Xác định tên	Sự miêu tả	STM32F07x Dòng STM32G0	STM32F30x	STM32F37x
# xác định IR_TIM	Bộ hẹn giờ được sử dụng để giải mã IR (1)	TIM3	TIM1	TIM3
# định nghĩa TIM_PRESCALER	Bộ định mức TIM Tham số này là tính toán để có 1 µs làm cơ sở thời gian. Tần số TIM (trong MHz) / (bộ đếm trước + 1)	47	71	71
# xác định IR_TIM_CLK	Đồng hồ APB của bộ đếm thời gian đã sử dụng	__HAL_RCC_TIM3_ CLK_ENABLE	__HAL_RCC_TIM1_ CLK_ENABLE	__HAL_RCC_TIM3_ CLK_ENABLE
# xác định IR_TIM_IRQn	IR TIM IRQ	TIM3_IRQn	TIM1_CC_IRQn	TIM3_IRQn
# định nghĩa IR_TIM_Channel	Kênh IR TIM	TIM_CHANNEL_2	TIM_CHANNEL_2	TIM_CHANNEL_2
# định nghĩa IR_GPIO_PORT	Cổng mà đầu ra IR là kết nối (1)	GPIOC	GPIOA	GPIOB
# định nghĩa IR_GPIO_PORT_CLK	Cổng đồng hồ GPIO chân hồng ngoại	__HAL_RCC_GPIOC _CLK_ENABLE	__HAL_RCC_GPIOA _CLK_ENABLE	__HAL_RCC_GPIOB _CLK_ENABLE
# xác định IR_GPIO_PIN	Ghim IR là kết nối (1)	GPIO_PIN_6	GPIO_PIN_9	GPIO_PIN_5

1. Để biết thêm chi tiết về các tài nguyên STM32 hiện có, hãy tham khảo bảng dữ liệu sản phẩm.

4 Lớp giao diện

Hình 23. Kiến trúc lớp ứng dụng



Có nhiều giao thức hồng ngoại tương tự, được phân biệt với giao thức SIRC bởi các tham số thời gian. Các giao thức này được xử lý bởi các hàm sirc_decode.c / sirc_encode.c. Người dùng chỉ cần cập nhật các giá trị thời gian.

Có những cái khác khá khác biệt và được quản lý bởi các chức năng cụ thể như RC5 và trình điều khiển liên quan của nó rc5_decode.c / rc5_encode.c.

Mỗi giao thức có một khung cấu trúc cụ thể. IR_FRAME là một con trỏ đến cấu trúc giao thức hồng ngoại đã chọn và nó chứa thông tin chính cần thiết cho giao tiếp (địa chỉ thiết bị và lệnh).

4.1 Các chương trình trình diễn

Để đảm bảo khởi động nhanh chóng, bộ phát và bộ thu hồng ngoại được trình bày trong tài liệu này được triển khai bằng ngôn ngữ C và có sẵn để tải xuống miễn phí dưới dạng gói X-CUBE-IRREMOTE.

4.1.1 Trình diễn máy phát bằng IRTIM

Trình diễn này bao gồm việc truyền các thông điệp IR được hiển thị trên màn hình LCD.

Mỗi thông báo IR được hiển thị thành hai phần

- Bộ thu thiết bị IR
- Lệnh được thực thi

4.1.2 Trình diễn máy thu sử dụng GP-Timer được định cấu hình ở chế độ PWM

Trình diễn này bao gồm việc nhận các thông điệp IR và gửi chúng đến màn hình LCD.

Mỗi thông báo IR được hiển thị thành hai phần

- Thiết bị truyền khung IR
- Lệnh được thực thi

4.2 Cách tùy chỉnh trình điều khiển IR Trình điều

4.2.1 khiển bộ thu IR

Để bao gồm trình điều khiển bộ giải mã hồng ngoại dựa trên giải pháp đầu vào PWM trong ứng dụng người dùng, hãy làm theo các bước sau:

1. Thêm tệp tiêu đề của giao thức IR thích hợp vào dự án Ví dụ: rc5_decode.h.

2. Thêm tệp.c tương ứng với giao thức IR vào dự án Ví dụ: rc5_decode.c.

3. Gọi hàm khởi tạo giao thức trong hàm main ()
   Thí dụ: RC5_Init ();
4. Thêm các chức năng ngắt TIMx vào *stm32f0xx_it.c, stm32g0xx_it.c hoặc stm32f3xx_it.c*
   Thí dụ:

```
void TIM2_IRQHandler (void)
{
    HAL_TIM_IRQHandler (& TimHandleDEC);
}
```

5. Xác định cấu trúc cho giao thức IR trong tệp main.c Ví dụ:

```
RC5_Frame_TypeDef IR_FRAME;
```

6. Gọi hàm giải mã trong hàm main () Ví dụ:

```
void main (void)
{
    . . .
    RC5_Init ();
    trong khi (1)
    {
        RC5_Decode (& IR_Frame);
    }
}
```

Các thay đổi cần thiết để hỗ trợ bất kỳ giao thức IR nào

Giải pháp này có thể được sử dụng để hỗ trợ bất kỳ giao thức hồng ngoại nào bằng cách chỉ thực hiện một số thay đổi trong tệp tiêu đề và cập nhật lệnh và bảng thiết bị.

- Tạo một tệp tiêu đề (exp: ir_protocol_name.h) tương tự như tệp sirc_decode.h. Thay đổi các định nghĩa để điều chỉnh nó cho phù hợp với các thông số kỹ thuật của giao thức IR đã chọn (thời lượng bit tối thiểu / tối đa, thời lượng tiêu đề tối thiểu / tối đa, tổng số bit, thời gian chờ ...)

Bảng 5. Danh sách các định nghĩa trong tệp tiêu đề cho các tham số giao thức IR

Xác định	Ý nghĩa	Cài đặt mẫu cho giao thức SIRC
SIRC_TIME_OUT_US	Thời gian chờ tính bằng µs	4050
SIRC_BITS_COUNT	Số lượng bit	11
SIRC_TOTAL_BITS _COUNT	Tổng số bit Xung thấp tối thiểu	11
SIRC_ONTIME_MIN_US	tính bằng µs Xung thấp tối đa	(600 - 60)
SIRC_ONTIME_MAX_US	tính bằng µs	(1200 + 60)
SIRC_HEADER_LOW_MIN_US	Xung thấp tiêu đề tối thiểu tính bằng µs Xung	(2400 - 150)
SIRC_HEADER_LOW_MAX_US	thấp tiêu đề tối đa tính bằng µs	(2400 + 150)
SIRC_HEADER_WHOLE_MIN_US	Toàn bộ thời lượng tiêu đề tối thiểu tính bằng µs	(2400 + 600 - 60)
SIRC_HEADER_WHOLE_MAX_US Toàn bộ thời lượng tiêu đề tối đa tính bằng µs		(2400 + 600 + 60)
SIRC_VALUE_STEP_US	Giá trị bước giữa bit0 và bit1 tính bằng µs 600	
SIRC_VALUE_MARGIN_US	Lợi nhuận tính bằng µs	100
SIRC_VALUE_00_US	Thời lượng bit0 tính bằng µs	1200

- Thay đổi trường khung giao thức IR trong cấu trúc IR_Frame_TypeDef

```
typedef struct
{

/ * Cấu trúc của khung IR (Địa chỉ, Lệnh, ....) * /

} IR_Frame_TypeDef;
```

- trong tệp sirc_decode.c, thêm các bảng IR_Commands và IR_devices thích hợp cho giao thức IR

4.2.2 Trình điều khiển máy phát hồng ngoại

Để bao gồm trình điều khiển bộ mã hóa hồng ngoại dựa trên giải pháp IRTIM trong ứng dụng người dùng, hãy làm theo các bước sau:

1. Thêm tệp tiêu đề của giao thức IR thích hợp vào dự án Ví dụ: rc5_encode.h

2. Thêm tệp.c tương ứng với giao thức IR vào dự án Ví dụ: rc5_encode.c

3. Gọi hàm khởi tạo giao thức trong main ()
   Thí dụ: RC5_Encode_Init ();
4. Thêm các chức năng ngắt TIMx vào *stm32f0xx_it.c, stm32g0xx_it.c hoặc stm32f3xx_it.c*
   Thí dụ:

```
void TIM16_IRQHandler (void)
{
    HAL_TIM_IRQHandler (& TimHandleLF);

}
```

5. Gọi hàm mã hóa trong main () Ví dụ:

```
void main (void)
{
    . . .
    RC5_Encode_Init ();
    trong khi (1)
    {
        RC5_Encode_SendFrame (Địa chỉ, Chỉ dẫn, Điều khiển);
    }
}
```

5 Phần kết luận

Ghi chú ứng dụng này cung cấp giải pháp triển khai phần mềm bộ phát / thu IR sử dụng bộ định thời có sẵn trên vi điều khiển STM32.

Ứng dụng mã hóa IR sử dụng các bộ vi điều khiển thuộc Sê-ri STM32F0, STM32F3 STM32Gx, STM32Lx và STM32Wx và tận dụng bộ điều biến phần cứng gọi là IRTIM kết hợp tín hiệu từ hai bộ hẹn giờ bên trong để điều khiển giao diện IR. Tính năng này làm cho bộ vi điều khiển đặc biệt phù hợp cho các ứng dụng yêu cầu khả năng tạo tín hiệu IR.

Ứng dụng giải mã IR cho phép tích hợp giải pháp IR trong mô-đun HDMI-CEC để hỗ trợ các chức năng điều khiển cấp cao cho tất cả các sản phẩm nghe nhìn khác nhau trong một môi trường nhất định.

Việc triển khai bộ giải mã IR được mô tả trong ghi chú ứng dụng này hoạt động với bộ hẹn giờ mục đích chung và có thể được chuyển sang bất kỳ bộ vi điều khiển STM32 nào.

6 Lịch sử sửa đổi

Bảng 6. Lịch sử sửa đổi tài liệu

Ngày	Bản sửa đổi	Những thay đổi
14-03-2016	1	Phát hành lần đầu.
24-11-2020	2	Đã cập nhật: - *Giới thiệu* - *Phần 2.1: Cân nhắc phần cứng* - *Hình 8: Mô tả phần cứng* - *Mục 2.2: Máy phát hồng ngoại: giải pháp phổ quát* - *Phần 3.3.3: Cách sử dụng trình điều khiển bộ giải mã RC5* - *Bảng 3: Ví dụ về cách thực hiện* - *Phần 3.4.3: Cách sử dụng trình điều khiển bộ giải mã SIRC* - *Bảng 4: Ví dụ về cách thực hiện* - *Phần 4.2.1: Trình điều khiển bộ thu IR* - *Phần 4.2.2: Trình điều khiển bộ phát hồng ngoại* - *Phần 5: Kết luận*