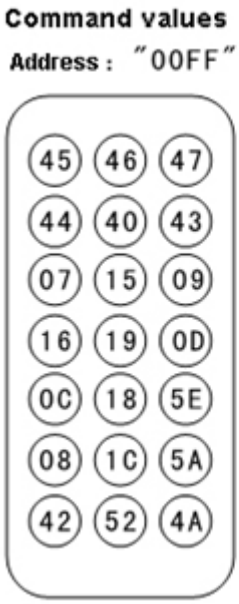
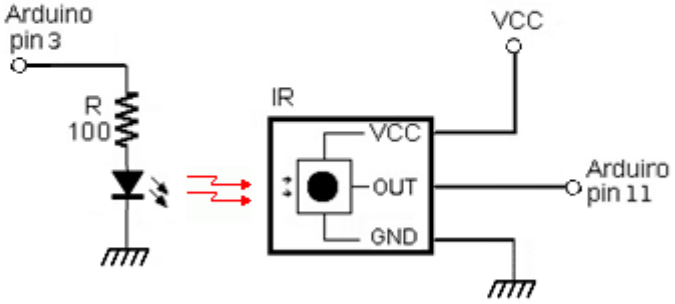
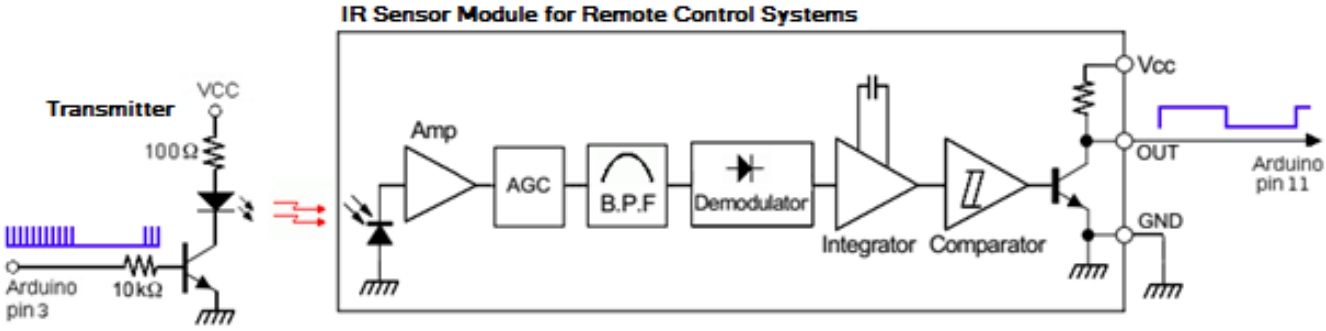
Infrared Remote Control Systems

การรับส่งข้อมูลด้วยแสงอินฟาเรด หรือ IR (Infra-red) เป็นการสื่อสารข้อมูลแบบไร้สายที่อาศัยย่านความถี่อยู่ในช่วงคลื่นอินฟาเรดที่ซึ่งสายตามนุษย์ไม่สามารถมองเห็นด้วยตาเปล่าได้ จึงนิยมนำมาทำเป็นรีโมทเพื่อใช้ในการเปิดปิดอุปกรณ์และควบคุมระบบการทำงานของเครื่องใช้ไฟฟ้าต่างๆ โดยทั่วไประบบรีโมท (Remote Control Systems) จะประประกอบด้วย รีโมทอินฟราเรดที่ใช้ตัว Infrared Emitting Diode ที่มีความยาวคลื่นประมาณ 940 nm ในการส่งข้อมูลแบบไร้สาย โดยจะมอดูเลตด้วยความถี่สัญญาณคลื่นพาห์ (Carrier frequencies) ที่มีช่วงความถี่ระหว่าง 30 KHz ถึง 60 KHz และมีโมดูลรับแสงอินฟราเรด (IR Receiver Module) หรืออาจจะเรียกว่า IR Sensor Module เป็นตัวรับสัญญาณที่ใช้ในงานควบคุมระบบไร้สาย เหมาะสำหรับการใช้งานในระยะไกลไม่เกิน 10 เมตร ตัวรีโมทและตัวรับสัญญาณในระบบแบบนี้จะมีขนาดเล็กใช้งานง่าย โดยจะต้องมีไฟเลี้ยงวงจรของโมดูลรับ-ส่งแสงอินฟราเรดนี้ในช่วงประมาณ 3V ถึง 5 VDC โดยมีตัวอย่างแสดงดังรูป

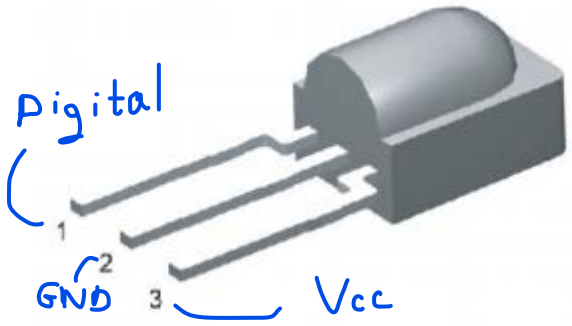




ภาครับสัญญาณอินฟราเรดจะใช้ตัว IR Sensor Module ที่เป็นไอซีแบบ 3 ขา มีวงจรภายในแสดงดังรูป ซึ่งต่อเข้ากับบอร์ด Arduino ทางขา Digital หมายเลข 11 การทำงานภายในของไอซีแบ่งเป็นส่วนๆดังนี้ เริ่มจากส่วนแรกรับข้อมูลอินพุทเข้ามาจาก infrared receiver diode เป็นแสง IR ที่มีระดับสัญญาณต่ำ จึงต้องผ่านเข้าวงจรขยายสัญญาณ (Amp) ก่อน แต่เนื่องจากระยะทางในการส่งสัญญาณโดยใช้ระบบไร้สายแบบนี้ตัวส่งสัญญาณ รีโมทอินฟราเรด อาจจะอยู่ใกล้หรือไกลไม่แน่นอน ทำให้ระดับสัญญาณที่รับเข้ามาไม่คงที่ ดังนั้นจึงต้องเอาสัญญาณที่ได้ผ่านเข้าวงจร AGC (Automatic Gain Control) ที่ทำหน้าที่ควบคุมปรับระดับสัญญาณพัลส์ที่รับเข้ามานี้ให้คงที่ โดยไม่ต้องคำนึงถึงระยะห่างระหว่างเครื่องรับกับเครื่องส่ง หลังจากนั้นแล้วจะต้องนำสัญญาณที่ได้นี้ไปเข้าวงจร Bandpass Filter เพื่อให้ความถี่ของสัญญาณคลื่นพาห์ผ่านออกไปได้เท่านั้นและตัดสัญญาณรบกวนต่างๆออกไป เสร็จแล้วนำสัญญาณที่ได้ไปเข้าวงจรถอดสัญญาณคลื่นพาห์ออก (Demodulator) ให้เหลือแต่สัญญาณข้อมูลดิจิตอลที่ต้องการกลับคืนมา ซึ่งสัญญาณที่ได้นี้ยังมีลักษณะเป็นอนาล็อกไม่มีความเสถียรจึงต้องเข้าวงจร Signal Shaping ประกอบด้วยวงจร Integrator และ Comparator เพื่อปรับรูปร่างของสัญญาณและสร้างให้เป็นสัญญาณดิจิตอลเพื่อส่งต่อไปเข้าขา 11 ของบอร์ด Arduino



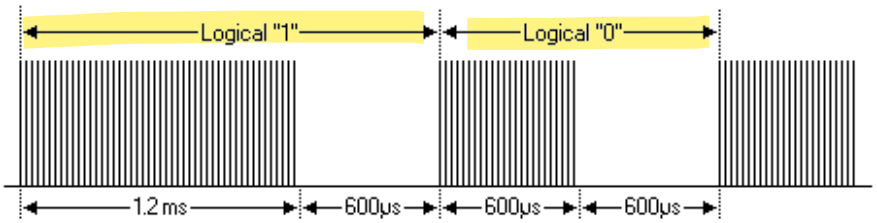
หลักการมอดูเลตสัญญาณ (Modulation) คือกระบวนการนำสัญญาณข้อมูลดิจิตอลที่เป็นความถี่ต่ำให้เกาะหรือผสมเข้ากับสัญญาณคลื่นพาห์ที่มีความถี่สูงและส่งสัญญาณที่มอดูเลตแล้วนี้ออกไป โดยความถี่ที่จะใช้เพื่อรับ-ส่งข้อมูลจากโมดูลสัญญาณ IR จะเป็นความถี่ 36 KHz , 38 KHz หรือ 40 KHz ซึ่งเป็นความถี่ที่นิยมใช้ในอุปกรณ์อิเล็กทรอนิกส์ทั่วไป โดยข้อมูลสัญญาณดิจิตอลที่ส่งออกไปนี้ช่วงที่เป็น 1 หรือ 0 จะขึ้นอยู่กับโปรโตคอลที่กำลังใช้อยู่ในแต่ละแบบ การทดลองจะให้ต่อวงจรฮาร์ดแวร์ของตัวรับ IR Sensor Module โดยมีขาที่ 1 เป็นขาส่งข้อมูลออกไปยังขาอินพุทแบบดิจิตอลของ Arduino ขา 2 จะเป็นขากราวด์ และขาที่ 3 ให้ต่อกับแหล่งจ่ายไฟ Vcc ที่ภาครับสัญญาณจะตรวจจับ IR demodulates ซึ่งสัญญาณ IR จะทำงานได้ดีที่สุดเมื่อความถี่ตัวรับตรงกับความถี่ของตัวส่งสัญญาณ แต่ในทางปฏิบัติแล้วตัวรับจะยังทำงานได้ถึงแม้ความถี่จะไม่ตรงกัน โดยในการทดลองนี้จะให้ใช้ความถี่ช่วงกลางของรีโมทที่ใช้กันทั่วไปคือ 38 KHz



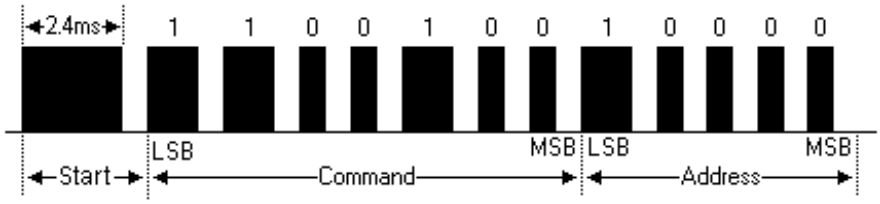
สำหรับภาคส่งให้เชื่อมต่อเอาท์พุท PWM ขา 3 กับ IR LED และตัวต้านทาน 100 โอห์ม ซึ่งจะได้ช่วงการรับ-ส่งอยู่ที่ประมาณ 5 เมตร ถ้าต้องการให้ได้ช่วงระยะทางเพิ่มมากขึ้นสามารถขยายเอาต์พุตได้ด้วยการต่อทรานซิสเตอร์เพิ่มเติม รูปแบบการทำงานของรีโมท IR จะเป็นการเปิดและปิดไฟ LED และเพื่อป้องกันไม่ให้แหล่งกำเนิดแสงอินฟราเรด เช่น แสงแดดหรือแสงไฟจากภายนอกมารบกวนการทำงาน จึงต้องเปิดและปิดที่ความถี่สูงโดยการมอดูเลตด้วยความถี่ 38 KHz ตามรูปแบบของโปรโตคอลที่ใช้ ช่วงเวลาที่สัญญาณมอดูเลตถูกส่งออกไปจะเรียกว่า mark และเมื่อไฟ LED ดับลงเป็นช่วงว่างที่ไม่มีสัญญาณเรียกว่า space แต่ละคีย์บนรีโมทจะมีรหัสเฉพาะขึ้นอยู่กับบรษัทผู้ผลิตสินค้า โดยทั่วไปจะมีจำนวนบิทตั้งแต่ 12 ถึง 32 บิท โดยรีโมทจะส่งรหัสออกทุกครั้งเมื่อมีการกดปุ่ม และถ้าหากกดปุ่มคีย์ค้างไว้รีโมทจะส่งรหัสบอกว่าคีย์ถูกกดค้าง โดยในรีโมท NEC จะมีการส่งรหัสเมื่อกดคีย์ค้างไว้เป็นรหัส repeate สำหรับรีโมท Philips RC5 หรือ RC6 ค่าของในแต่ละบิทจะถูก toggle ทุกครั้งที่มีการกดปุ่มนั้นอีกครั้ง ข้อมูลที่ใช้สำหรับการรับ-ส่งผ่าน IR ใช้วัดที่ช่วงเวลาที่มีสัญญาณคือ mark และเทียบกับช่วงเวลาที่ไม่มีสัญญาณคือ space ค่าคาบเวลาที่ใช้ มีหน่วยเป็นไมโครวินาที ผลที่ได้จะถูกแปลงเป็นค่าข้อมูลลอจิก 1 และ 0 โดยมีรูปแบบของสัญญาณที่ใช้ในการส่งข้อมูลของรีโมทแต่ละบริษัทผู้ผลิตเป็นดังนี้

Sony Protocol

Modulation



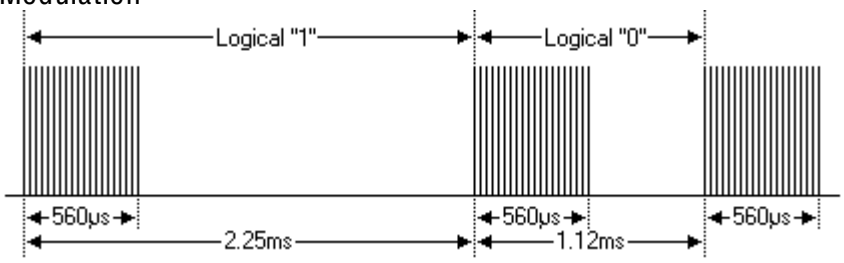
Protocol



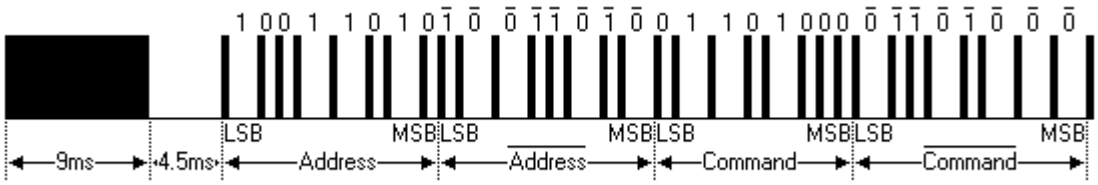
Address = Device
Command = Function
12-bit version = 7 command bits, 5 address bits

NEC Protocol

Modulation

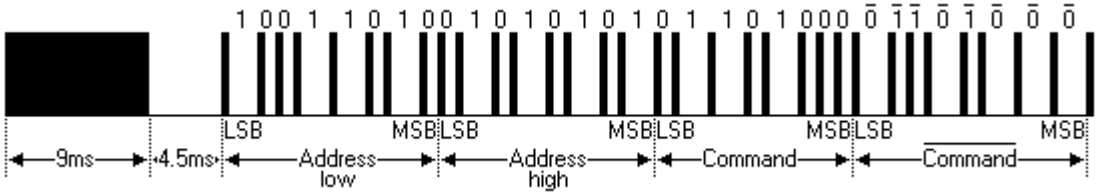


Protocol

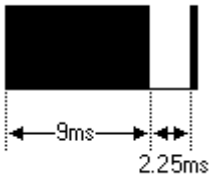


8 bit address and 8 bit command length

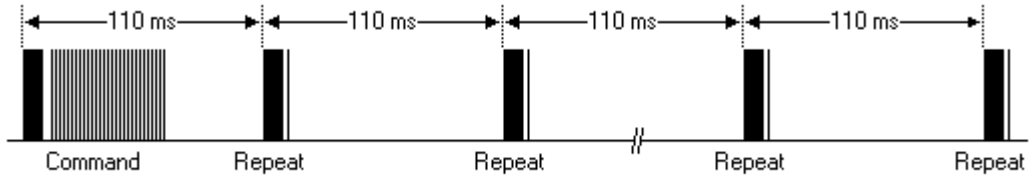
Extended NEC Protocol



repeat pluse

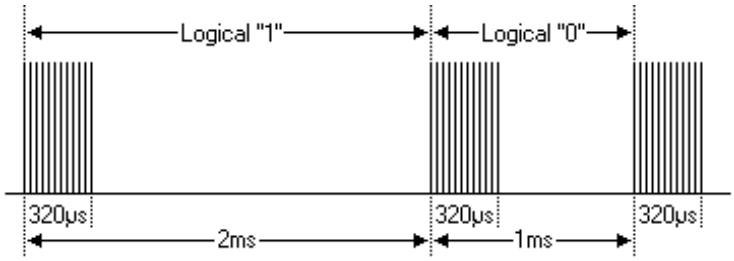


repeat code

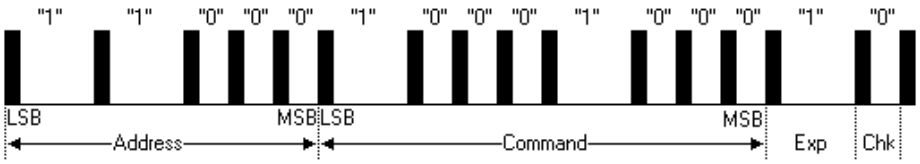


Sharp Protocol

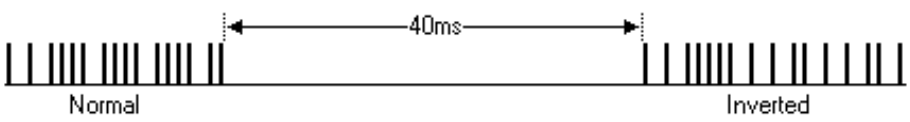
Modulation



Protocol

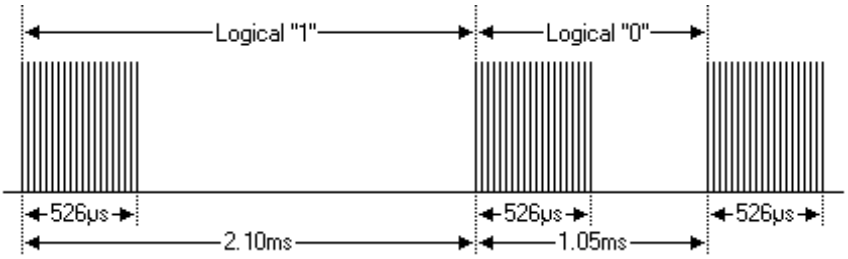


8 bit command , 5 bit address length



JVC Protocol

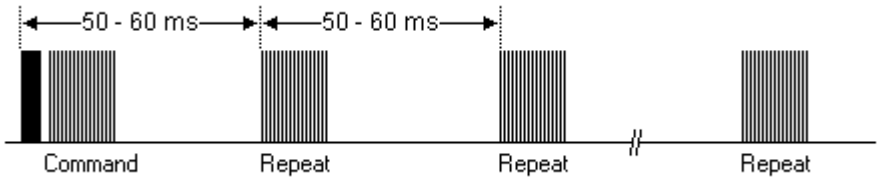
Modulation



Protocol

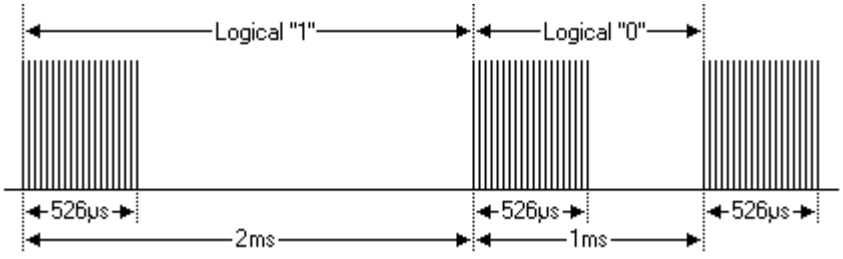
8 bit address and 8 bit command length

repeat code



Mitsubishi Protocol

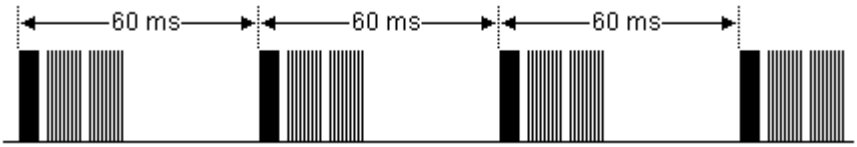
Modulation



Protocol

8 bit address and 8 bit command length

repeat code



การเขียนโปรแกรมเพื่อใช้งานรับ-ส่งข้อมูลร่วมกับอุปกรณ์รีโมท จะต้องสร้างและตรวจจับข้อมูลจากสัญญาณที่รับ-ส่งนั้น เพื่อให้ง่ายต่อการเขียนโปรแกรมจะเรียกใช้ไลบรารี เพื่อถอดรหัสของข้อมูลและพิมพ์รหัสจากรีโมทที่ใช้อยู่รูปแบบของรีโมทสำหรับอุปกรณ์ต่างๆ ในผู้ผลิตเดียวกันมักจะใช้รหัสเดียวกันสำหรับผลิตภัณฑ์ในกลุ่มเดียวกัน วิธีจัดการกับโปรโตคอลของข้อมูลที่รับ-ส่งนั้นจะต้องเขียนโปรแกรมเพื่อเรียกใช้ไลบรารี IRremote.h ที่ถูกสร้างขึ้นสำหรับใช้รหัสในแต่ละแบบ รายละเอียดของไลบรารีที่ได้เป็นดังนี้

ไลบรารี IRrecv ใช้เป็นเครื่องตรวจจับอินฟราเรดที่เชื่อมต่อกับขาอินพุทแบบดิจิตอล ใช้ในการรับและถอดรหัสของข้อมูลที่รับเข้ามา ประกอบด้วยสองส่วน คือ ส่วนแรกมอดูเลต (Modulation) จะเป็นขั้นตอนการเรียกใช้ interrupt routine ทุกๆ เวลา 50 microsecond วัดคาบเวลาของ mark และspace และบันทึกคาบเวลาลงในบัฟเฟอร์ และในส่วนที่สองโปรโตคอล (Protocol) ผู้ใช้จะเรียกรูทีนการถอดรหัสเพื่อถอดรหัสการวัดบัฟเฟอร์แปลงเป็นค่าโค้ดที่ถูกส่งมาตามแต่ละโปรโตคอลของบริษัทผู้ผลิตนั้นซึ่งค่าที่ได้นี้ปกติจะมีค่าตั้งแต่ 11 ถึง 32 บิท

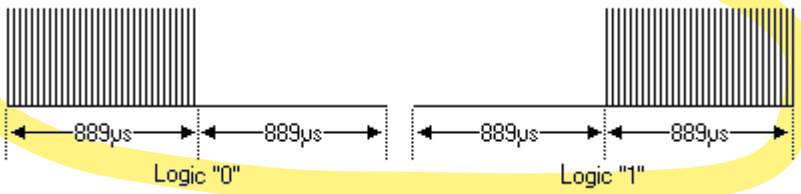
ไลบรารีของการขัดจังหวะ (interrupt) ในการรับสัญญาณจะถูกเรียกทุกครั้งโดยการตั้งค่า TIMER ให้เกิดขึ้นทุก 50 microseconds ในแต่ละครั้งของการ interrupt จะมีการตรวจสอบสถานะอินพุทและเพิ่มค่า counter เป็นการตรวจจับสัญญาณ modulate ซึ่งประกอบด้วยส่วนที่มีสัญญาณ (mark) และส่วนที่ไม่มีสัญญาณ (space) และบันทึกคาบเวลาลงในบัฟเฟอร์ ส่วนไลบรารีถอดรหัสจะสามารถถอดรหัสโปรโตคอลที่แตกต่างกันอย่างต่อเนื่องและจะหยุดหากทำได้สำเร็จโดยจะส่งกลับ (return) ในรูปแบบโครงสร้าง (structure) ที่ประกอบด้วยข้อมูลดิบ (raw data) , ข้อมูลที่ถอดรหัส (decode data) , จำนวนบิทของข้อมูลที่ถอดรหัส และโปรโตคอล (protocol) ที่ใช้ในการถอดรหัสข้อมูล

การเรียกใช้เมธอด จะเริ่มจากการ initial ด้วยคำสั่ง enableIRIn () แล้วใช้เมธอด decode() เพื่อเรียกดูว่าได้รับรหัสหรือไม่ ถ้าได้รับข้อมูลรหัสเข้ามาจะ return ค่าที่ไม่ใช่ศูนย์ และให้ผลลัพธ์เก็บไว้ใน structure ที่ชื่อ decode_results เมื่อข้อมูลถูกถอดรหัสแล้วต้องใช้เมธอด resume () เพื่อเรียกรับข้อมูลรหัสต่อไป

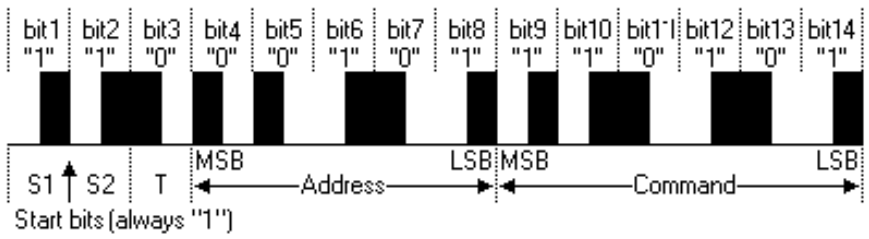
การถอดรหัส RC5 / 6 จะแตกต่างจากโค้ดอื่น ๆ เนื่องจากบิทการเข้ารหัส RC5 / 6 จะมี mark ตามด้วย space ได้เป็นลอจิก 0 และ space ตามด้วย mark ได้เป็นลอจิก 1 ดังรูป

Philips RC-5 Protocol

Modulation



Protocol



5 bit address and 6 bit command lengt4h (7 command bits for RC5X)

สำหรับการปุ่มค้างไว้จะเป็นการส่งค่าซ้ำ (repeat) การถอดรหัสจะ return ค่าที่ถอดรหัสเดียวกันนั้นซ้ำไปอีก ยกเว้นแต่ของ NEC ซึ่งจะส่งรหัสซ้ำพิเศษแทนในกรณีนี้รูทีนการถอดรหัสจะ return ค่า REPEAT

ไลบรารีการส่งข้อมูลด้วยรีโมท เพื่อให้เอาท์พุทได้ความถี่และรอบการทำงานที่ถูกต้องจะใช้ PWM timer โดยใช้คำสั่ง enableIROut เพื่อตั้งค่า timer ของเอาต์พุทขา 3 ที่เป็น PWM ให้ได้ความถี่ที่เหมาะสม ส่วนของเมธอด mark() จะส่ง mark โดยการเปิดใช้งานเอาท์พุท PWM ส่วนเมธอด space () จะส่ง space โดยการปิดใช้งานเอาท์พุท PWM

ไลบรารี IRremote ใช้โปรโตคอลที่แตกต่างกันดังนี้:

NEC: ส่งบิทข้อมูลทั้งหมด 32 บิท ตัวอย่างโปรโตคอลของบิทข้อมูลที่ส่งให้ดูจากรูป

Sony: ส่งบิทข้อมูล 12 บิทหรือมากกว่า โดยปกติจะใช้ 12 หรือ 20 บิท

RC5: ส่งบิทข้อมูล 12 บิทหรือมากกว่า ข้อมูลที่ส่งเริ่มต้นด้วย start bit จำนวน 2 บิทที่มีค่าเป็น 1 ซึ่งไม่ใช่เป็นส่วนหนึ่งของค่าข้อมูลที่ต้องการส่ง

สำหรับ Sony และ RC5 / 6 การส่งแต่ละครั้งต้องทำซ้ำ 3 ครั้ง ตามที่ระบุในโปรโตคอล

1. ให้ต่อวงจรใช้ Infrared Receiver Modules for PCM Remote Control Systems TSOP1838 เข้าที่ขา D11 และป้อนโปรแกรมดังตัวอย่าง โดยใช้ไลบรารี IRremote.h แล้วทดลองการทำงานโดยใช้ Remote แบบต่างๆ และให้อธิบายผลลัพธ์ที่ได้

```
#include <IRremote.h>                                   // Include the library
#define RECV 11                                         // IR receiver pin 11

IRrecv irrecv(RECV);
decode_results results;

void setup()
{
  Serial.begin(9600);                                   // Message will be sent to PC
  irrecv.enableIRIn();                                  // Start the receiver
  irrecv.blink13(true);                                 // pin 13 blink
}

void loop()
{
  if (irrecv.decode(&results))                          // Received IR signal
  {
    switch (results.decode_type)
    {
       case NEC:   Serial.print("NEC: "); break;
       case SONY:  Serial.print("SONY: "); break;
       case SHARP: Serial.print("SHARP: "); break;
       case PANASONIC: Serial.print("PANASONIC: "); break;
       case JVC: Serial.print("JVC: ");  break;
       case SANYO: Serial.print("SANYO: "); break;
       case MITSUBISHI: Serial.print("MITSUBISHI: "); break;
       case RC5: Serial.print("Philips RC5: "); break;
       case RC6: Serial.print("Philips RC6: "); break;
       case DISH:  Serial.print("DISH: ");   break;
       case DENON: Serial.print("DENON: ");  break;
       case SAMSUNG: Serial.print("SAMSUNG: "); break;
       case LG: Serial.print("LG: ");  break;
       case UNKNOWN: Serial.print("UNKNOWN: "); break;
    }

    Serial.println(results.value, HEX);                 // Print raw data
    irrecv.resume();                                    // Enable receiving of the next value
  }
}
```

เมื่อกดปุ่มใดๆ บนรีโมตจะมีข้อความปรากฏบน serial monitor ว่า NEC : FFFFFFFF ซึ่งแสดงว่ารีโมตตัวนี้ใช้ protocol ของ NEC นั่นเอง

2. กรณีถ้าใช้ไลบรารี IRremote.hpp จะมีรูปแบบของคำสั่งที่แตกต่างกัน ให้ทดลองการทำงานโดยใช้ Remote แบบต่างๆ และอธิบายผลลัพธ์ว่าแตกต่างกับข้อ 1 อย่างไร

```
#include <IRremote.hpp>                                   // Include the library
#define IR_RECEIVE_PIN 11                                 // IR receiver pin 11

void setup()
{
  Serial.begin(9600);                                     // Message will be sent to PC
  IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK);  // Start the receiver
}

void loop()
{
  if (IrReceiver.decode())                                // Received IR signal
  {
    switch (IrReceiver.decodedIRData.protocol)
    {
      case NEC: Serial.print("NEC: "); break;
      case SONY: Serial.print("SONY: "); break;
      case SHARP: Serial.print("SHARP: "); break;
      case PANASONIC: Serial.print("PANASONIC: "); break;
      case JVC: Serial.print("JVC: "); break;
      case RC5: Serial.print("Philips RC5: "); break;
      case RC6: Serial.print("Philips RC6: "); break;
      case DENON: Serial.print("DENON: "); break;
      case SAMSUNG: Serial.print("SAMSUNG: "); break;
      case LG: Serial.print("LG: "); break;
      case UNKNOWN: Serial.print("UNKNOWN: "); break;
    }

    Serial.println(IrReceiver.decodedIRData.decodedRawData, HEX);  // Print raw data
    IrReceiver.resume();                                           // Enable receiving of the next value
  }
}
```

ในข้อนี้หากกดปุ่มที่แตกต่างกันจะขึ้นข้อความ NEC : ___ จากนั้นจะแสดงผลข้อมูลดิบที่ถูกถอดรหัสออกมา แตกต่างกันไปตามปุ่มที่ถูกกด

3. ให้แก้ไขคำสั่งภายใน switch case ดังตัวอย่างข้างล่าง โดยเพิ่มเข้าไปในโปรแกรมข้อที่ 1 หรือข้อที่ 2 ก็ได้ ตาม Remote Control ที่นำมาทดลองใช้งานได้ เพื่อให้แสดงปุ่มกดต่างๆ ของคำสั่งใน Remote ที่นำมาใช้งาน

```
switch(results.value)                                  // Remote IR codes
{
  case 0xFF02FD: Serial.print("OK "); break;
  case 0xFF22DD: Serial.print("LEFT "); break;
  case 0xFFC23D: Serial.print("RIGHT "); break;
  case 0xFF629D: Serial.print("FORWARD "); break;
  case 0xFFA857: Serial.print("REVERSE "); break;
  case 0xFF6897: Serial.print("1 "); break;
  case 0xFF9867: Serial.print("2 "); break;
  case 0xFFB04F: Serial.print("3 "); break;
  case 0xFF30CF: Serial.print("4 "); break;
  case 0xFF18E7: Serial.print("5 "); break;
  case 0xFF7A85: Serial.print("6 "); break;
  case 0xFF10EF: Serial.print("7 "); break;
  case 0xFF38C7: Serial.print("8 "); break;
  case 0xFF5AA5: Serial.print("9 "); break;
  case 0xFF4AB5: Serial.print("0 "); break;
  case 0xFF42BD: Serial.print("* "); break;
  case 0xFF52AD: Serial.print("# "); break;
  case 0xFFFFFFFF: Serial.print("REPEAT ");break;
  default:  Serial.print("other button ");
}
```

4. ให้แก้ไขโปรแกรมเพื่อให้แสดงผลของ Remote ที่ได้ โดยให้แสดงรหัสคำสั่งบน 7-segment และแสดงผลลัพธ์ของปุ่มกดที่ LED Matrix 12x8

การส่งสัญญาณรีโมท ปกติตัวเครื่องจะต้องมีขนาดเล็กใช้แบตเตอรี่และกินกำลังไฟให้น้อยที่สุดเท่าที่จะเป็นไปได้ โดยมีตัวส่งสัญญาณอินฟราเรดที่มีความแรงมากที่สุด เพื่อให้ได้ระยะการควบคุมที่ไกล ปกติแล้วเมื่อไม่มีการกดปุ่มเครื่องรีโมทควรอยู่ในโหมดสลีป ซึ่งจะใช้พลังงานไฟฟ้าต่ำมากแทบจะไม่มีการกินกระแสไฟเลย โดยตัวประมวลผลจะถูกปลุกให้ตื่นเพื่อส่งสัญญาณ IR เมื่อมีการกดที่ปุ่มบนรีโมท ผู้ผลิตในแต่ละรายได้ออกแบบผลิตชิพไอซีจำนวนมากเพื่อใช้เป็นเครื่องส่งสัญญาณ IR และมีโปรโตคอลของตนเองตามแต่ละผู้ผลิต ซึ่งถูกคิดค้นขึ้นมาใช้ในอุปกรณ์รุ่นต่างๆที่ตนเองเป็นผู้ผลิตเท่านั้น ในปัจจุบันไมโครโปรเซสเซอร์ที่ใช้งานกินกำลังไฟฟ้าที่ต่ำจึงมีการนำมาใช้เป็นเครื่องส่งสัญญาณ IR ด้วย ซึ่งจะมีความยืดหยุ่นในการใช้งานได้หลายแบบ โดยการสั่งงานให้ไมโครโปรเซสเซอร์จ่ายกระแสไฟผ่านไปยัง LED ที่เป็นชนิด Infrared Emitting Diode ที่มีช่วงความยาวคลื่น (wavelength) อยู่ในช่วง 940 nm และใช้กระแสไฟฟ้าประมาณ 100mA ซึ่งจะได้ระยะการควบคุมที่เหมาะสม เนื่องจากสัญญาณที่ส่งไปขับ LED นี้จะเป็นพัลส์ที่มีช่วงเวลาในการจ่ายกระแสไฟฟ้าที่สั้นมากๆ ดังนั้นถ้าต้องการให้ได้ระยะการควบคุมที่ไกลขึ้นต้องคำนวณกระแสไฟที่จะจ่ายได้มากที่สุดให้กับ LED โดยดูได้จากค่า Peak forward current ซึ่งจะเป็นค่ากระแสสูงสุดเท่าที่จะเป็นไปได้และจะได้ระยะการควบคุมมากที่สุด แต่ก็จะต้องดูค่าเฉลี่ยของพลังงานไฟฟ้าที่ LED ใช้ด้วยว่าไม่ควรเกินค่าสูงสุดที่ LED นั้นจะรับได้ ซึ่งค่าพารามิเตอร์เหล่านี้สามารถดูข้อมูลได้จาก datasheet ของ LED นั้น และในระหว่างที่ไม่ส่งกระแสไปที่ LED ควรจะสั่งให้ปิดเครื่องเพื่อยืดอายุการใช้งานของแบตเตอรี่ เนื่องจากขาเอาท์พุทของไมโครโปรเซสเซอร์แต่ละตัวจะจ่ายกระแสไฟฟ้าได้จำกัด ดังนั้นถ้าต้องการให้จ่ายกระแสไฟมากขึ้นจะต้องใช้ทรานซิสเตอร์ทำเป็นวงจรขับกระแสให้กับ LED ควรเลือกทรานซิสเตอร์ที่มีความเร็วในการสวิตช์และมีค่า HFE ที่เหมาะสมโดยใช้ค่าตัวความต้านทานที่คำนวณได้จากกฎของโอห์ม

ไลบรารีของ Infrared Remote Control ประกอบด้วยสองส่วนคือ IRsend จะส่งแพ็กเก็ตของข้อมูลรีโมทอินฟาเรด ในขณะที่ IRrecv จะรับและถอดรหัสข้อมูลนั้น คำสั่ง IR IRsend ใช้อินฟราเรด LED ที่เชื่อมต่อกับเอาท์พุทดิจิตอลขา 3 ในการส่งข้อมูลให้เรียกใช้เมธอดส่ง โดยมีโปรโตคอลที่ใช้พร้อมกับข้อมูลที่จะส่งและจำนวนบิทที่จะส่ง ตัวอย่าง sketch ในการส่งรหัสข้อมูลไปเปิด- ปิดเครื่องทีวี Sony ซึ่งจะสั่งให้ข้อมูลถูกส่งออกไปยังพอร์ตแบบอนุกรม เพื่อให้ Arduino เปิดหรือปิดทีวี

5. ให้ต่อวงจรโดยใช้ Infrared Emitting Diode TSAL6200 และความต้านทาน 100 ohm เข้าที่ PWM ขา D3 แล้วต่อ Switch ระหว่างขา D12 และขากราวด์ เสร็จแล้วให้ป้อนโปรแกรมกำหนดให้ขา D12 เป็น Input ที่มีการต่อ Internal Resistor แบบ pull-up และให้ทำการตรวจสอบสถานะของขา D12 กำหนดเงื่อนไขไว้ว่า เมื่อมีการกด Switch ให้ทำงานเป็น Remote จะส่งค่ารหัสออกมาทาง Infrared Emitting Diode TSAL6200 โดยใช้ไลบรารี IRremote.h ตัวอย่างจะส่งเป็น Protocol ของ Sony ขนาด 12 บิต และของ NEC ขนาด 32 บิต โดยมี LED ที่ขา 13 แสดงสถานะว่ากำลังส่งข้อมูล แล้วให้ทดลองการทำงานและอธิบายผลลัพธ์ที่ได้

```
#include <IRremote.h>                    // Include the library

IRsend irsend;
int key = 12;                            // KEY pin
int led =  13;                           // LED pin
```

จากวงจรจะมีการต่อ led สีแดงขนานกับหลอด infrared เพื่อสังเกตการทำงานของหลอด infrared โดยเมื่อกดสวิตช์แล้วจะนับเวลา 1 วินาที led สีแดงจะดับ ∴ infrared ดับ จึงไม่มีการส่งข้อมูล

```
unsigned long codeSony = 0xa90;                    // Sony TV power code
int codeLenSony = 12;                              // the length of the code
unsigned long codeNEC = 0x1234ABCD;                // NEC code
int codeLenNEC = 32;                               // the length of the code

void setup()
{
  pinMode(led, OUTPUT);
  pinMode(key, INPUT_PULLUP);
}

void loop()
{
  if (digitalRead(key) == HIGH)
  {
    digitalWrite(led, HIGH);                       // turn LED on

    irsend.sendSony(codeSony, codeLenSony);        // Sony code
    delay(1000);

   irsend.sendNEC(codeNEC, codeLenNEC);            // NEC code
   delay(1000);

    digitalWrite(led, LOW);                        // turn LED off
  }
}
```

6. กรณีใช้ไลบรารี IRremote.hpp ในการส่ง โดยใช้ Protocol ของ NEC ขนาด 32 บิต จะมีรูปแบบของคำสั่งดังโปรแกรมด้านล่าง ให้ทดลองการทำงานและอธิบายผลลัพธ์ว่าแตกต่างกับข้อ 5 อย่างไร

```
#include "PinDefinitionsAndMore.h"
#include <IRremote.hpp>                    // Include the library

uint16_t sAddress = 0x01A2;
uint16_t sCommand = 0x0CB34;
uint8_t sRepeats = 0;

void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);            // LED pin
  Serial.begin(9600);
  IrSender.begin();                        // Start with IR_SEND_PIN
  disableLEDFeedback();                    // Disable feedback LED at default feedback LED pin
}

void loop() {
  if (Serial.read() != -1)
  {
    Serial.println();
    Serial.print("Send IR signals at pin ");
    Serial.println(IR_SEND_PIN);

    Serial.print("Send Protocol=NEC : address=");
    Serial.print(sAddress, HEX);
    Serial.print(", command=");
    Serial.print(sCommand, HEX);
    Serial.print(", repeats=");
    Serial.println(sRepeats);

    IrSender.sendNEC(sAddress, sCommand, sRepeats);
    delay(100);
  }
}
```

ตอบข้อ 6 เมื่อกด upload โปรแกรมจะแสดงผลข้อความใน Serial monitor เป็นชื่อ protocol พร้อมกับ Address และ Command ของแต่ละ protocol

7. ให้แก้ไขโปรแกรมในข้อที่ 6 เป็นการส่งรหัสข้อมูลเมื่อมีการกดสวิทช์แทนการสั่งจาก Serial Monitor กำหนดให้ส่งรหัสข้อมูลโดยใช้ Address แทนด้วยรหัสประจำตัวนักศึกษา 4 หลักหลัง และ Command 4 ค่าแทนด้วยสวิทช์ 4 ตัว