```
//
//  Working Version V2
//



/* Written by Michael Welsh September 7, 2017 for implementation using the Arduino UNO
 * To be used with complementary program transmitter_sample_JMW_V2
 *
 * This program is designed to integrate the UNO and nRF905.  It operates in a
 * RECEIVE mode to obtain data from the transmitter.  The transmitter is also integrated to
 * a UNO and NRF905.  This program receives up to 16 integer data types (2 bytes each) for a
 * maximum 32 byte packet.  Range for integer is -32768 through 32767.
 *
 *
 * Program and Library adopted from the Rethink Tech Inc. - Tinkbox (nRF905) and the
 * Jeff Rowberg I2C device class
 *
 *
 * UNO to nRF905 BOARD PIN/Control Feature
 *
 * 7 -> CE     Standby - High = TX/RX mode, Low = Standby
 * 8 -> PWR    Power Up - High = On, Low = Off
 * 9 -> TXE    TX or RX mode - High = TX, Low = RX
 * 2 -> CD     Carrier Detect - High when RF signal detected, for collision avoidance
 * 3 -> DR     Data Ready - High when finished transmitting/data received
 * 10 -> CSN   SPI SS
 * 12 -> SO    SPI MISO
 * 11 -> SI    SPI MOSI
 * 13 -> SCK   SPI SCK
 * GND -> GND  Ground

 Note on use of the CD Pin.  I have completed a test in which the trasmitter
 was able to send data to the Receiver with this pin disconnected.  This may be usefull
 because this pin may be required when using the MPU6050 which uses Pin 2 of the UNO
 for interrupt processing.  Both the transmitter and receiver CD connection is
 optional for this program.

*/

#include <nRF905.h>  //Library Author Zak Kemble, Web: http://blog.zakkemble.co.uk/nrf905-
avrarduino-librarydriver/
#include <SPI.h>  //SPI Master Library for Arduino

/*
  Note in selection of the RXADDR and TXADDR.  Nordic recommends that the address length be 24 bits
or higher
```

```
  in length.   Smaller lengths can lead to statistical failures due to the address bein repeated as part of
the
  data packet.  Each byte should be unique.  The address should have several level shifts (101010101).
*/

#define RXADDR {0x58, 0x6F, 0x2E, 0x10} // Address of this device (4 bytes)
#define TXADDR {0xFE, 0x4C, 0xA6, 0xE5} // Address of device to send to (4 bytes)

#define PACKETPAUSE 250 // Short Break after receiving each data packet

void setup()
{
  // Power up nRF905 and initialize
  nRF905_init();

  // Send this device address to nRF905
  byte incoming[] = RXADDR;
  nRF905_setRXAddress(incoming);

  // Send remote device address to nRF905
  byte outgoing[] = TXADDR;
  nRF905_setTXAddress(outgoing);

  // Put into receive mode
  nRF905_receive();

  // Start serial coomunication with monitor.  Send start message.
  Serial.begin(9600);
  Serial.print(F("Transmitter started....."));
}

void loop()
{
          // Make data array buffer
          int data[NRF905_MAX_PAYLOAD];   //array size is 32 bytes defined by NRF905_MAX_PAYLOAD
in library

          // Wait for data packet

  Serial.print(F("Waiting for data....."));

  while(!nRF905_getData(data, sizeof(data)));

  Serial.println(F("Data Received....."));
  Serial.println();

  // Display communications parameters
  Serial.print("NRF905 Max Payload (bytes)= ");
```

```
  Serial.println(NRF905_MAX_PAYLOAD);
  Serial.print("Received Data Buffer Size= ");
  Serial.println(sizeof(data));
  Serial.println();
  Serial.println();

  // Display data packet received in buffer array
  for (int i=0; i<=15; i++)
       {
        Serial.print("   ");
        Serial.print(data[i], DEC);
       }

  // Create horizontal spacing and pause between data packets.
  Serial.println();
  Serial.println();
  delay(PACKETPAUSE);

}  // Continue to receive further data if available.
```