```
/*

================================================================================================================
  === Programma per sintonizzare Loop Magnetica con determinazione della posizione del condensatore variabile  ===
  === attraverso potenziometro dieci giri ed azionamento del motore per sintonia bande 40/80/160m.          ===
  === Franco Binda IZ2HFG - v1r1 (21 Agosto 2017)                                                            ===

================================================================================================================
  === Migliorire apportate nella versione "1r1":                                                             ===
  === Il valore indicato nella calibrazione aumenta o diminuisce il limite superiore ed inferiore impostato  ===
  === nelle variabili "bandHxxx" e "bandLxxx" per le tre bande, questo per compensare la variazione della    ===
  === capacità del condensatore variabile dovuta alla temperatura esterna visto che il metallo si dilata     ===

================================================================================================================
*/
#include <LiquidCrystal.h>
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
//
// Paragarafo per determinare il trascorrere del tempo e relativo timer.
//
unsigned long previousMillis = 0;
unsigned long currentMillis;
//
long timer1 = 500 ;     // Variabile usata quale delay nel DEBUG
int debug = 0 ;         // Variabile per DEBUG
int value = 1024 ;      // Varibile valore letto su pin analogico "A0" (Magic Number = 1024 è un paradosso! Serve solo per il Display nella sezione DEBUG)
int value2 = 0 ;        // Varibile valore letto su pin analogico "A2"
int valueS = 1024 ;     // Magic Number = 1024 usato nella sezione DEBUG per visualizzare su Display solo quando cambiano i valori.
int calS = 1024 ;       // Magic Number = 1024 usato nella sezione DEBUG per visualizzare su Display solo quando cambiano i valori.
int counter = 0 ;       // Usato quale contatore per la variabile "cal".
int cal = 0 ;           // Variabile per la "calibrazione" usata leggendo valore pin analogico "A2"
int calX = 0 ;          // Variabile per cambio "calibrazione" in corsa.
int calT = 1 ;          // Calibrazione totale in kHz.
int mx = 1 ;            // Moltiplicatore (*1,*2...*x) della variabile "cal" (1 originale)
int cla ;               // Variabile per mantenere/pulire righe Display quando cambiano valori
int clb ;               // Variabile per mantenere/pulire righe Display quando cambiano valori
int fH40 = 7200 ;       // Variabile che imposta il limite alto di frequenza per  40m
int fL40 = 7000 ;       // Variabile che imposta il limite basso di frequenza per 40m
int fH80 = 3800 ;       // Variabile che imposta il limite alto di frequenza per 80m
int fL80 = 3500 ;       // Variabile che imposta il limite basso di frequenza per 80m
int fH160 = 1850 ;      // Variabile che imposta il limite alto di frequenza per 160m
int fL160 = 1830 ;      // Variabile che imposta il limite basso di frequenza per 160m
int band = 0 ;          // Variabile per visualizzare bande 40/80m
int band40 = 0 ;        // Variabile banda 40m
int band80 = 0 ;        // Variabile banda 80m
int band160 = 0 ;       // Variabile banda 160m
//
////////////////////////////////////////////////////////////////////////////////
///                          !!!  PRESTARE ATTENZIONE !!!                    ///
////////////////////////////////////////////////////////////////////////////////
/// I valori sotto indicati sono veri se si collega il potenziometro nello stesso  ///
/// modo di quando sono sati rilevati attraverso VNA e la funzione di DEBUG.       ///
/// Infatti se i due fili del potenziometro, NEGATIVO e POSITIVO, sono invertiti i ///
/// dati rilevati da pin analogico "A0" non sono più validi!!!                     ///
////////////////////////////////////////////////////////////////////////////////
//
int limitH = 840 ;      // Definizione valore finecorsa HIGH
int limitL = 20 ;       // Definizione valore finecorsa LOW
int bandH40 = 753 ;     // Definizione limite superiore banda 40m (lettura A0)
int bandL40 = 746 ;     // Definizione limite inferiore banda 40m (lettura A0)
int bandH80 = 501 ;     // Definizione limite superiore banda 80m (lettura A0)
int bandL80 = 423 ;     // Definizione limite inferiore banda 80m (lettura A0)
int bandH160 = 305 ;    // Definizione limite superiore banda 160m (lettura A0)
int bandL160 = 261;     // Definizione limite inferiore banda 160m (lettura A0)
int bandh40 ;           // Usato con il valore di calibrazione per ridefinire in limite superiore banda 40m
int bandl40 ;           // Usato con il valore di calibrazione per ridefinire in limite inferiore
```

```
banda 40m
int bandh80 ;           // Usato con il valore di calibrazione per ridefinire in limite superiore
banda 80m
int bandl80 ;           // Usato con il valore di calibrazione per ridefinire in limite inferiore
banda 80m
int bandh160 ;          // Usato con il valore di calibrazione per ridefinire in limite superiore
banda 160m
int bandl160 ;          // Usato con il valore di calibrazione per ridefinire in limite inferiore
banda 160m
float frequency ;       // Varibile frequenza per 40/80m
float frequency160 ;    // Varibile frequenza per 160m
byte lcdoff = 1 ;       // Se la variabile è = 1 esegue "lcd.noDisplay - lcd.display" (usato su test
finecorsa)
int delayrele = 500 ;   // Ritardo in millisecondi disinserimento relè motore solo quando la
sintonia è in "Auto"
int rele7 = 0 ;         // Variabile "relè7" stato relè che comanda il motore in un senso che
AUMENTA la capacità del condensatore variabile
int rele9 = 0 ;         // Variabile "relè9" stato relè che comanda il motore nel senso opposto che
DIMINUISCE la capacità del condensatore variabile
int sw1 = 0 ;           // Variabile "switch1"  per relè condensatore aggiuntivo per 160m
int sw8 = 0 ;           // Variabile "switch8"  per 40m
int sw10 = 0 ;          // Variabile "switch10" per 80m
int sw6 = 0 ;           // Variabile "switch6"  per 160m
#define inp6 6          // pin 6  Digitale definito come input  (legge lo stato del commutatore se
posizionato su 160m)
#define inp7 7          // pin 7  Digitale definito come input  (legge lo stato del relè che comanda
il motore onde aumentare la capacità)
#define out7 7          // pin 7  Digitale definito come output (attiva il relè del motore in un
senso)
#define inp8 8          // pin 8  Digitale definito come input  (legge lo stato del commutatorte se
posizionato su 40m)
#define inp9 9          // pin 9  Digitale definito come input  (legge lo stato del relè che comanda
il motore onde diminuire la capacità)
#define out9 9          // pin 9  Digitale definito come output (attiva il relè del motore nel senso
opposto)
#define inp10 10        // pin 10 Digitale definito come input  (legge lo stato del commutatorte se
posizionato su 80m)
//
void setup() {
  //
  pinMode(inp6, INPUT);
  pinMode(inp7, INPUT);
  pinMode(out7, OUTPUT);
  pinMode(inp8, INPUT);
  pinMode(inp9, INPUT);
  pinMode(out9, OUTPUT);
  pinMode(inp10, INPUT);
  //
  // Serial.begin(9600);
  //
  digitalWrite(out7, LOW);
  digitalWrite(out9, LOW);
  //
  bandh40 = bandH40 ;
  bandl40 = bandL40 ;
  bandh80 = bandH80 ;
  bandl80 = bandL80 ;
  bandh160 = bandH160 ;
  bandl160 = bandL160 ;
  //
  lcd.begin(20, 4);
  //
  // ********************************
  // ***    Messaggio iniziale     ***
  // ********************************
  //
  lcd.clear();
  delay (3000);
  lcd.setCursor(0, 0);
  lcd.print("\x7c\x20\x20\x20\x20\x20\x20\x20\x48\x61\x6d\x20\x20\x20\x20\x20\x20\x20\x20\x7c");
  lcd.setCursor(0, 1);
  lcd.print("\x7c\x20\x20\x20\x45\x6c\x65\x63\x74\x72\x6f\x6e\x69\x63\x73\x20\x20\x20\x20\x7c");
  lcd.setCursor(0, 2);
  lcd.print("\x7c\x20\x20\x49\x5a\x32\x48\x46\x47\x20\x28\x32\x30\x31\x37\x29\x20\x20\x20\x7c");
  lcd.setCursor(0, 3);
  lcd.print("\x7c\x20\x20\x20\x56\x65\x72\x73\x69\x6f\x6e\x20\x31\x72\x31\x20\x20\x20\x20\x7c");
  delay (3000);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("--------------------");
```

```
  lcd.setCursor(0, 1);
  lcd.print("- Inizio  Verifica -");
  lcd.setCursor(0, 2);
  lcd.print("- <-> SINTONIA <-> -");
  lcd.setCursor(0, 3);
  lcd.print("--------------------");
  delay (3000);
  lcd.clear();
}
void loop() {
  //
  unsigned long currentMillis = millis();
  //
  value = analogRead(A0);
  value2 = analogRead(A2);
  //
  ////////////////////////////////////////////////////////////////////////////////
  //                                SEZIONE CALIBRAZIONE                        //
  ////////////////////////////////////////////////////////////////////////////////
  // Valori dipendenti dal moltiplicatore "mx"; se "mx=1" "cal" da "-10" a "+10"    //
  // Attraverso il potenziometro connesso al pin "A2" si rileva da 0 a 1023.        //
  // Successivamente tramite le condizion sotto si forza la variabile "call"        //
  // che aggiunge o sottrae tale valore alle variabili "bandXX" onde correggere     //
  // l'errore di risonanza dovuto alla temperatura delbox esterno.                  //
  ////////////////////////////////////////////////////////////////////////////////
  //
  if (value2 > 980 && value2 <= 1023) {
    cal = 10;
  }
  if (value2 > 940 && value2 < 980) {
    cal = 9;
  }
  if (value2 > 900 && value2 < 940) {
    cal = 8;
  }
  if (value2 > 850 && value2 < 900) {
    cal = 7;
  }
  if (value2 > 800 && value2 < 850) {
    cal = 6;
  }
  if (value2 > 750 && value2 < 800) {
    cal = 5;
  }
  if (value2 > 700 && value2 < 750) {
    cal = 4;
  }
  if (value2 > 650 && value2 < 700) {
    cal = 3;
  }
  if (value2 > 600 && value2 < 650) {
    cal = 2;
  }
  if (value2 > 560 && value2 < 600) {
    cal = 1;
  }
  if (value2 > 490 && value2 < 560) {
    cal = 0;
  }
  if (value2 > 450 && value2 < 490) {
    cal = -1;
  }
  if (value2 > 400 && value2 < 450) {
    cal = -2;
  }
  if (value2 > 350 && value2 < 400) {
    cal = -3;
  }
  if (value2 > 300 && value2 < 350) {
    cal = -4;
  }
  if (value2 > 250 && value2 < 300) {
    cal = -5;
  }
  if (value2 > 200 && value2 < 250) {
    cal = -6;
  }
  if (value2 > 150 && value2 < 200) {
    cal = -7;
  }
```

```
  if (value2 > 100 && value2 < 150) {
    cal = -8;
  }
  if (value2 > 50 && value2 < 100) {
    cal = -9;
  }
  if (value2 >= 0 && value2 < 50) {
    cal = -10;
  }
  //
  // Contatore di cicli "loop" con memorizzazione valore variabile "cal".
  // Inoltre si ridefinisce il limite superiore ed inferione di ogni banda aggiungendo o sottraendo
  // il valore della variabile "cal" questo per compensare manualmente la variazione
  // di capacità del variabile in base alla temperatura esterna.
  //
  if (counter == 0) {
    (calX = cal);
    (bandH40 = (bandh40 + (cal * (mx))));
    (bandL40 = (bandl40 + (cal * (mx))));
    (bandH80 = (bandh80 + (cal * (mx))));
    (bandL80 = (bandl80 + (cal * (mx))));
    (bandH160 = (bandh160 + (cal * (mx))));
    (bandL160 = (bandl160 + (cal * (mx))));
}
  counter = (counter + 1);
  if (cal != calX) {                  // Confronto se il valore di calibrazione "cal" è cambiato
    (counter = 0);                    // Azzero il contatore per aggiornare limiti banda con nuovo
valore "cal"
    (bandH40 = (bandh40 + (cal * (mx))));
    (bandL40 = (bandl40 + (cal * (mx))));
    (bandH80 = (bandh80 + (cal * (mx))));
    (bandL80 = (bandl80 + (cal * (mx))));
    (bandH160 = (bandh160 + (cal * (mx))));
    (bandL160 = (bandl160 + (cal * (mx))));
}
  //
  // ================================================================================
  // === SEZIONE DEBUG:                                                           ===
  // === Se pin "A3" è "HIGH" visualizzo solo quello che leggo                    ===
  // === dai due pins analogici "A0" ed "A2" per analizzare i realtivi valori.    ===
  // ================================================================================
  //
  if (digitalRead(A3) == HIGH) {
    if ((digitalRead(A3) == HIGH) && (debug == 0)) {
      lcd.setCursor(0, 0);
      lcd.print("********************");
      lcd.setCursor(0, 1);
      lcd.print("*   Attivazione    *");
      lcd.setCursor(0, 2);
      lcd.print("*  Sezione DEBUG   *");
      lcd.setCursor(0, 3);
      lcd.print("********************");
      delay(2000);
      // Pulizia Display
      lcd.setCursor(0, 0);
      lcd.print("                    ");
      lcd.setCursor(0, 1);
      lcd.print("                    ");
      lcd.setCursor(0, 2);
      lcd.print("                    ");
      lcd.setCursor(0, 3);
      lcd.print("                    ");
      (debug = 1);
    }
    else if (debug == 1) {
      if ((currentMillis - previousMillis) > (timer1)) {    // Sezione "delay" attraverso uso
"millis" e relativo timer.
        //
        ////////////////////////////////////////////////////////////////////////////
        //            Gestione visualizzazione barra in base alla lettura "A0"          //
        ////////////////////////////////////////////////////////////////////////////
        //
        lcd.setCursor(0, 0);
        if (value > 1020) {
          for (int i = 19; i < 20; i++) {
            lcd.write(1023);  // 19-19
          }
        }
        if (value > 1000) {
          for (int i = 1; i < 20; i++) {
```

```
    lcd.write(1023);  // 0-19
  }
  for (int z = 20; z < 21; z++) {
    lcd.write(1022);
  }
}
else if (value > 950) {
  for (int i = 1; i < 19; i++) {
    lcd.write(1023);  // 0-18
  }
  for (int z = 19; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 900) {
  for (int i = 1; i < 18; i++) {
    lcd.write(1023);
  }
  for (int z = 18; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 850) {
  for (int i = 1; i < 17; i++) {
    lcd.write(1023);
  }
  for (int z = 17; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 800) {
  for (int i = 1; i < 16; i++) {
    lcd.write(1023);
  }
  for (int z = 16; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 750) {
  for (int i = 1; i < 15; i++) {
    lcd.write(1023);
  }
  for (int z = 15; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 700) {
  for (int i = 1; i < 14; i++) {
    lcd.write(1023);
  }
  for (int z = 14; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 650) {
  for (int i = 1; i < 13; i++) {
    lcd.write(1023);
  }
  for (int z = 13; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 600) {
  for (int i = 1; i < 12; i++) {
    lcd.write(1023);
  }
  for (int z = 12; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 550) {
  for (int i = 1; i < 11; i++) {
    lcd.write(1023);
  }
  for (int z = 11; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 500) {
  for (int i = 1; i < 10; i++) {
```

```
    lcd.write(1023);
  }
  for (int z = 10; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 450) {
  for (int i = 1; i < 9; i++) {
    lcd.write(1023);
  }
  for (int z = 9; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 400) {
  for (int i = 1; i < 8; i++) {
    lcd.write(1023);
  }
  for (int z = 8; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 350) {
  for (int i = 1; i < 7; i++) {
    lcd.write(1023);
  }
  for (int z = 7; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 300) {
  for (int i = 1; i < 6; i++) {
    lcd.write(1023);
  }
  for (int z = 6; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 250) {
  for (int i = 1; i < 5; i++) {
    lcd.write(1023);
  }
  for (int z = 5; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 200) {
  for (int i = 1; i < 4; i++) {
    lcd.write(1023);
  }
  for (int z = 4; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 150) {
  for (int i = 1; i < 3; i++) {
    lcd.write(1023);  // 1,2
  }
  for (int z = 3; z < 20; z++) {
    lcd.write(1022);
  }
}
else if (value > 75) {
  for (int i = 1; i < 2; i++) {
    lcd.write(1023);  // 1
  }
  for (int z = 2; z < 20; z++) {
    lcd.write(1022);
  }
}
else if ((value < 75) && (value > 0)) {
  for (int i = 0; i < 1; i++) {
    // lcd.write(1026);  // 0
    lcd.write(1023);  // 0
  }
}
else if (value == 0) {
  lcd.setCursor(0, 0);
  lcd.write(1022);
}
```

```
        //
        // Refresh Display solo in caso di variazione dei dati "value" e "cal"
        //
        if ((value) != (valueS)) {
          lcd.setCursor(16, 1);
          lcd.print("    ");
          lcd.setCursor(0, 1);
          lcd.print("Valore A0 (val):");
          lcd.print(value, DEC);
          valueS = value;
        }
        if ((cal) != (calS)) {
          lcd.setCursor(16, 2);
          lcd.print("    ");
          lcd.setCursor(0, 2);
          lcd.print("Valore A2 (cal):");
          lcd.print(cal, DEC);
          lcd.setCursor(0, 3);
          lcd.print("--------------------");
          calS = cal;
        }
        previousMillis = currentMillis;
      }
    }
    goto fordebug ;
  }
  if ((debug == 1) && (digitalRead(A3)) == LOW) { // Condizione presente ogni volta che si commuta
da "DEBUG" a normal.
    debug = 0;                                    // Setto la variabile = 0 evitando così
l'esecuzione ciclica della if.
    cla = 0;                                      // Azzeramento variabili per evitare permanenza
caratteri precedenti su display.
    clb = 0;                                      // Azzeramento variabili per evitare permanenza
caratteri precedenti su display.
    lcd.clear();                                  // Pulizia display.
  }
  //
  // ================================================================================
  // ===                          FINE SEZIONE DI DEBUG                           ===
  // ================================================================================
  //
  //
  // ********************************************************************************
  // *** Sezione Bande 40/80/160m in base al valore letto su pin  "A0"              ***
  // ********************************************************************************
  //
  ////////////////////////////////////////////////////////////////////////////////
  // Verifica solo per test, se "A1" è LOW oppure HIGH.                          //
  ////////////////////////////////////////////////////////////////////////////////
  // if ((digitalRead(A1) == LOW)) {
  // (statusA1 = 0);
  // }
  // if ((digitalRead(A1) == HIGH)) {
  // (statusA1 = 1);
  // }
  ////////////////////////////////////////////////////////////////////////////////
  //
  if ((digitalRead(inp10) == HIGH) && ((value >= bandL80) && (value <= bandH80))) { // Commutatore
su 80m e siamo in banda 80m.
    (band = 80);
    (band80 = 80);
    (frequency = (fH80 - ((bandH80 - value) * (fH80 - fL80) / (bandH80 - bandL80))));
  }
  else
  {
    (band80 = 0);
  }
  if ((digitalRead(inp8) == HIGH) && (value >= bandL40 && value <= bandH40)) {       // Commutatore
su 40m, siamo in banda 40m.
    (band = 40);
    (band40 = 40);
    (frequency = (fH40 - ((bandH40 - value) * (fH40 - fL40) / (bandH40 - bandL40))));
  }
  else
  {
    (band40 = 0);
  }
  if ((digitalRead(inp6) == HIGH) && ((value >= bandL160) && (value <= bandH160))) { // Commutatore
su 160m, siamo in banda 160m.
    (band160 = 160);
```

```
    (frequency160 = (fH160 - ((bandH160 - value) * (fH160 - fL160) / (bandH160 - bandL160))));
  }
  else
  {
    (band160 = 0);
  }
  if (digitalRead(inp6) == LOW) {  // COMMUTATORE NON SU 160m, SIAMO PER DEFINIZIONE FUORI BANDA
160m.
    (band160 = 0);
  }
  if ((((digitalRead(A1) == HIGH)) && (digitalRead(inp6) == HIGH) && ((value >= bandL160) && (value
<= bandH160)))) { // Sintonia Automatica, Commutatore su 160m, siamo in banda 160m, motore
arrestato.
    digitalWrite(out7, LOW);
    digitalWrite(out9, LOW);
    (sw1 = 1);
    (band160 = 160);
    (band40 = 0);
    (band80 = 0);
    (frequency160 = (fH160 - ((bandH160 - value) * (fH160 - fL160) / (bandH160 - bandL160))));
  }
  if (((digitalRead(A1) == HIGH) && (digitalRead(inp6) == HIGH) && ((value < bandL160) || (value >
bandH160)))) { // Sintonia Automatica, siamo fuori banda 160m.
    (band160 = 0);
  }
  if ((digitalRead(inp10) == HIGH) && (digitalRead(A1) == HIGH) && (value <= bandH80 && value >=
bandL80)) {     // Sintonia Automatica, Commutatore su 80m, siamo in banda 80m, motore arrestato.
    digitalWrite(out7, LOW);
    digitalWrite(out9, LOW);
  }
  if ((digitalRead(inp8) == HIGH) && (digitalRead(A1) == HIGH) && (value <= bandH40 && value >=
bandL40)) {      // Sintonia Automatica, Commutatore su 40m, siamo in banda 80m, motore arrestato.
    digitalWrite(out7, LOW);
    digitalWrite(out9, LOW);
  }
  //
  // ********************************************************************************
  // *** Questa sezione verifica ed effettua:                                     ***
  // *** 1) Quale banda in base allo stato del commutatore risulta commutata.     ***
  // *** 2) Quale risulta il valore letto su pin analogico "A0" e se il valore è  ***
  // *** 3) diverso da quello prefissato, viene eccitato il relè relativo         ***
  // *** 4) Raggiunto il valore prefissato, viene diseccitato il relè relativo    ***
  // *** 5) Inoltre anche in base allo stato del pin analogico "A1" (Auto/Man)    ***
  // ********************************************************************************
  //
  //
  // Verifica Banda 160m (sotto 1.830 MHz)
  //
  if ((digitalRead(inp6) == HIGH)  && ((digitalRead(A1) == HIGH) && (value < bandL160))) { //
Sintonia Automatica, commutatore su 160m, il valore letto è < al voluto.
    digitalWrite(out9, LOW);
    digitalWrite(out7, HIGH);
    (sw6 = 1);
    (rele7 = 1);
  }
  if ((digitalRead(inp6) == HIGH) && (value == bandL160)) {                                  //
Sintonia Automatica o Manuale, commutatore su 160m, siamo in banda 160m.
    delay (delayrele);
    digitalWrite(out9, LOW);
    (rele9 = 0);
  }
  //
  // Verifica Banda 160m (sopra 1.850 MHz)
  //
  if ((digitalRead(inp6) == HIGH) && ((digitalRead(A1) == HIGH) && (value > bandH160))) {
    digitalWrite(out7, LOW);
    digitalWrite(out9, HIGH);
    (sw6 = 1);
    (rele9 = 1);
  }
  if ((digitalRead(inp6) == HIGH) && (value == bandL160)) {
    delay (delayrele);
    digitalWrite(out7, LOW);
    (rele7 = 0);
  }
  //
  // Verifica Banda 80m (sopra 3.800 MHz)
  //
  if ((digitalRead(inp10) == HIGH) && ((digitalRead(A1) == HIGH) && (value > bandH80))) {
    digitalWrite(out7, LOW);
```

```
  digitalWrite(out9, HIGH);
  (sw10 = 1);
  (rele9 = 1);
}
if ((digitalRead(inp10) == HIGH) && (value == bandH80)) {
  delay (delayrele);
  digitalWrite(out9, LOW);
  (rele9 = 0);
}
//
// Verifica Banda 80m (sotto 3.500 MHz)
//
if ((digitalRead(inp10) == HIGH) && ((digitalRead(A1) == HIGH) && (value < bandL80))) {
  digitalWrite(out9, LOW);
  digitalWrite(out7, HIGH);
  (sw10 = 1);
  (rele7 = 1);
}
if ((digitalRead(inp10) == HIGH) && (value == bandL80)) {
  delay (delayrele);
  digitalWrite(out7, LOW);
  (rele7 = 0);
}
//
// Verifica Banda 40m (sopra 7.200 MHz)
//
if ((digitalRead(inp8) == HIGH) && ((digitalRead(A1) == HIGH) && (value > bandH40))) {
  digitalWrite(out7, LOW);
  digitalWrite(out9, HIGH);
  (sw8 = 1);
  (rele9 = 1);
}
if ((digitalRead(inp8) == HIGH) && (value == bandH40)) {
  delay (delayrele);
  digitalWrite(out9, LOW);
  (rele9 = 0);
}
//
// Verifica Banda 40m (sotto 7.000 MHz)
//
if ((digitalRead(inp8) == HIGH) && ((digitalRead(A1) == HIGH) && (value < bandL40))) {
  digitalWrite(out9, LOW);
  digitalWrite(out7, HIGH);
  (sw8 = 1);
  (rele7 = 1);
}
if ((digitalRead(inp8) == HIGH) && (value == bandL40)) {
  delay (delayrele);
  digitalWrite(out7, LOW);
  (rele7 = 0);
}
//
// ********************************************************************************
// ***                              Sezione Avvisi                              ***
// ********************************************************************************
//
////////////////////////////////////////////////////////////////////////////////////
//                 Sezione visualizzazione valore di "calibrazione"                //
////////////////////////////////////////////////////////////////////////////////////
//
calT = cal * (mx);
if (counter == 1) {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("--------------------");
  lcd.setCursor(0, 1);
  lcd.print(" Calibrazione Val.  ");
  if (calT <= -10) {
    lcd.setCursor(4, 2);
    lcd.print("=-> ");
  }
  else if ((calT >= -9) && (calT <= -1) || (calT >= 10)) {
    lcd.setCursor(5, 2);
    lcd.print("=-> ");
  }
  else
  {
    lcd.setCursor(6, 2);
    lcd.print("=-> ");
  }
```

```
  lcd.print(calT, DEC);
  lcd.setCursor(11, 2);
  lcd.print(" <-=");
  lcd.setCursor(0, 3);
  lcd.print("--------------------");
  delay (1500);
  lcd.clear();
  //
  cla = 0; // Azzeramento variabili per evitare permanenza caratteri precedenti su display
  clb = 0; // Azzeramento variabili per evitare permanenza caratteri precedenti su display
}
//
////////////////////////////////////////////////////////////////////////////////
// Verifica se sono stati raggiunti i "finecorsa" logici (vedi variabili)          //
////////////////////////////////////////////////////////////////////////////////
//
if ((value <= limitL) || (value >= limitH)) {
  digitalWrite(out7, LOW);
  digitalWrite(out9, LOW);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("!!!!!!!!!!!!!!!!!!!!");
  lcd.setCursor(0, 1);
  lcd.print(" ->  FINECORSA  <-  ");
  lcd.setCursor(0, 2);
  lcd.print("INVERTI DIR. MOTORE ");
  lcd.setCursor(0, 3);
  lcd.print("!!!!!!!!!!!!!!!!!!!!");
  delay (5000);
  //
  if (lcdoff == 1) {
    lcd.noDisplay();
    delay (200);
    lcd.display();
  }
  //
  // Pulizia Display
  lcd.setCursor(0, 0);
  lcd.print("                    ");
  lcd.setCursor(0, 1);
  lcd.print("                    ");
  lcd.setCursor(0, 2);
  lcd.print("                    ");
  lcd.setCursor(0, 3);
  lcd.print("                    ");
}
//
////////////////////////////////////////////////////////////////////////////////
// Disecccita i due relè nel caso in cui entrambi fossero contemporaneamnte attivi //
////////////////////////////////////////////////////////////////////////////////
//
if ((digitalRead(inp7) == HIGH) && (digitalRead(inp9) == HIGH)) {
  digitalWrite(out7, LOW);
  digitalWrite(out9, LOW);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("!!!!!!!!!!!!!!!!!!!!");
  lcd.setCursor(0, 1);
  lcd.print("<- NON SINTONIZZO ->");
  lcd.setCursor(0, 2);
  lcd.print(" RIDUCI VEL. MOTORE ");
  lcd.setCursor(0, 3);
  lcd.print("!!!!!!!!!!!!!!!!!!!!");
  delay (5000);
  // Pulizia Display
  lcd.setCursor(0, 0);
  lcd.print("                    ");
  lcd.setCursor(0, 1);
  lcd.print("                    ");
  lcd.setCursor(0, 2);
  lcd.print("                    ");
  lcd.setCursor(0, 3);
  lcd.print("                    ");
}
//
////////////////////////////////////////////////////////////////////////////////
// Disecccita i due relè che azionano il motore quando si cambia banda             //
////////////////////////////////////////////////////////////////////////////////
//
if (((digitalRead(inp6) == HIGH) && (digitalRead(inp10) == HIGH)) ||
```

```
    ((digitalRead(inp10) == HIGH) && (digitalRead(inp8) == HIGH))) {
  digitalWrite(out7, LOW);
  digitalWrite(out9, LOW);
  lcd.setCursor(0, 0);
  lcd.print("====================");
  lcd.setCursor(0, 1);
  lcd.print("= Commutata  Banda =");
  lcd.setCursor(0, 2);
  lcd.print("=    Attederere    =");
  lcd.setCursor(0, 3);
  lcd.print("====================");
  delay(2000);
  // Pulizia Display
  lcd.setCursor(0, 0);
  lcd.print("                    ");
  lcd.setCursor(0, 1);
  lcd.print("                    ");
  lcd.setCursor(0, 2);
  lcd.print("                    ");
  lcd.setCursor(0, 3);
  lcd.print("                    ");
  delay(500);
}
//
////////////////////////////////////////////////////////////////////////////////////
// Verifica se uno dei due relè è attivo e contemporaneamente il deviatore da       //
// "Sintonia Automatica" è passato a "Sintonia Manuale", diseccita i relè del motore //
////////////////////////////////////////////////////////////////////////////////////
//
if ((digitalRead(A1) == LOW) && ((digitalRead(out7) == HIGH) || (digitalRead(out9) == HIGH))) {
  digitalWrite(out7, LOW);
  digitalWrite(out9, LOW);
  lcd.setCursor(0, 0);
  lcd.print("====================");
  lcd.setCursor(0, 1);
  lcd.print("= Sintonia Manuale =");
  lcd.setCursor(0, 2);
  lcd.print("= Motore Arrestato =");
  lcd.setCursor(0, 3);
  lcd.print("====================");
  delay (2000);
  // Pulizia Display
  lcd.setCursor(0, 0);
  lcd.print("                    ");
  lcd.setCursor(0, 1);
  lcd.print("                    ");
  lcd.setCursor(0, 2);
  lcd.print("                    ");
  lcd.setCursor(0, 3);
  lcd.print("                    ");
}
//
// ********************************************************************************
// ********************************************************************************
// ***   Sezione visualizzazione Display in base alle varie condizioni presenti   ***
// ********************************************************************************
// ********************************************************************************
//
if ((band40 != 40) && (band80 != 80) && (band160 != 160)) {
  if (currentMillis - previousMillis > timer1 + 5000) {
    lcd.setCursor(0, 0);
    lcd.print("<<<<<<<<<<>>>>>>>>>>");
    lcd.setCursor(0, 1);
    lcd.print("< RISONANZA  FUORI >");
    lcd.setCursor(0, 2);
    lcd.print("<      BANDA       >");
    lcd.setCursor(0, 3);
    lcd.print("<<<<<<<<<<>>>>>>>>>>");
    delay (1000);
    // Pulizia Display
    lcd.setCursor(0, 0);
    lcd.print("                     ");
    lcd.setCursor(0, 1);
    lcd.print("                     ");
    lcd.setCursor(0, 2);
    lcd.print("                     ");
    lcd.setCursor(0, 3);
    lcd.print("                     ");
    previousMillis =  currentMillis;
  }
```

```
  }
  if ((band40 != 40) && (band80 != 80) && (band160 != 160)) { // Bande diverse da 40/80/160
(Sintonia Automatica/Manuale)
    lcd.setCursor(0, 0);
    lcd.print("                    ");
    lcd.setCursor(0, 0);
    lcd.print("Verifica - Risonanza");
    clb = 0;
  }
  //
  cla = (cla + 1);
  if ((band40 == 40) || (band80 == 80) || (band160 == 160)) { // Siamo in una delle tre bande
40/80/160 (Sintonia Automatica/Manuale)
    clb = (clb + 1);
    if ((clb != cla) || (cla == 1)) {
      lcd.setCursor(0, 0);
      lcd.print("                    ");
      lcd.setCursor(0, 0);
      lcd.print("ANTENNA IN RISONANZA");
      (cla = clb);
    }
  }
  if ((clb != cla) || (cla == 1)) {
    lcd.setCursor(0, 1);
    lcd.print("                    ");
    lcd.setCursor(0, 1);
    lcd.print("Rilevamento...: ");                          // Sempre visualizzato (Sintonia
Automatica/Manuale)
    lcd.print(value, DEC);
  }
  else if ((clb == cla) || (cla != 1)) {
    lcd.setCursor(16, 1);
    lcd.print("    ");
    lcd.setCursor(16, 1);
    lcd.print(value, DEC);
  }
  if (((rele7 == 1) || (rele9 == 1)) && ((band40 != 40) || (band80 != 80) || (band160 != 160))) { //
Uno dei relè è attivo, ma siamo fuori bande 40/80/160 (Sintonia Automatica)
    lcd.setCursor(0, 2) ;
    lcd.print("                    ");
    lcd.setCursor(0, 2) ;
    lcd.print("BANDE NoN 160/80/40m");
  }
  else if ((clb != cla) || (cla == 1)) {
    lcd.setCursor(0, 2) ;
    lcd.print("                    ");
    lcd.setCursor(0, 2) ;
    lcd.print("> Sintonia Manuale <");
  }
  if (((value >= bandL160) && (value <= bandH160)) && (band160 == 160)) {                          //
Siamo in banda 160
    if ((clb != cla) || (cla == 1)) {
      lcd.setCursor(0, 2);
      lcd.print("                    ");
      lcd.setCursor(0, 2);
      lcd.print("banda metri...: ");
      lcd.print(band160, 1);
    }
  }
  else if (((band40 == 40) || (band80 == 80)) && (band160 == 0)) {                                 //
Siamo in bande 40/80
    if ((clb != cla) || (cla == 1)) {
      lcd.setCursor(0, 2);
      lcd.print("                    ");
      lcd.setCursor(0, 2);
      lcd.print("Banda metri...: ");
      lcd.print(band, 1);
    }
  }
  if ((rele7 == 1) || (rele9 == 1)) {                                                              //
Quando uno dei relè è attivo (Sintonia Automatica)
    lcd.setCursor(0, 3);
    lcd.print("Sintonia in corso...");
    delay(500);
  }
  if ((rele7 == 0) && (rele9 == 0) && (band160 == 160)) {                                          //
Il motore non è attivo, siamo all'interno della banda dei 160m
    if ((clb != cla) || (cla == 1)) {
      lcd.setCursor(0, 3);
      lcd.print("                    ");
```

```
      lcd.setCursor(0, 3);
      lcd.print("frequenza kHz.: ");
      lcd.print(frequency160, 0);
    }
    else if ((clb == cla) || (cla != 1)) {
      lcd.setCursor(16, 3);
      lcd.print(frequency160, 0);
    }
  }
  else if ((rele7 == 0) && (rele9 == 0) && (band40 == 40 || band80 == 80) && (band160 == 0)) {
// Il motore non è attivo, siamo all'interno bande 40/80m
    if ((clb != cla) || (cla == 1)) {
      lcd.setCursor(0, 3);
      lcd.print("                    ");
      lcd.setCursor(0, 3);
      lcd.print("Frequenza kHz.: ");
      lcd.print(frequency, 0);
    }
    else if ((clb == cla) || (cla != 1)) {
      lcd.setCursor(16, 3);
      lcd.print(frequency, 0);
    }
  }
  if (((sw6 == 0 && sw8 == 0 && sw10 == 0)) && (band40 != 40) && (band80 != 80) && (band160 != 160))
{ // Commutatori LOW e bande non 40/80/160
    lcd.setCursor(0, 3);
    lcd.print("                    ");
    lcd.setCursor(0, 3);
    lcd.print("BANDE NON 40/80/160m");
    delay(1000);
  }
  //
  // Azzeramento di alcune variabili prima di inizare nuovo ciclo
  //
  (rele9 = 0);
  (rele7 = 0);
  (sw6 = 0);
  (sw8 = 0);
  (sw10 = 0);
  //
fordebug :
  //
  ////////////////////////////////////////////////////////////////////////////////
  // Cambia la visualizzazzione messaggi DEBUG se relativo PIN "A3" è "HIGH" ma il   //
  // commutatore è stato azionato dopo avvio del programma.                         //
  ////////////////////////////////////////////////////////////////////////////////
  //
  if ((digitalRead(A3) == HIGH) && (debug == 1)) {
    delay(500);
  }
  else
  {
    delay(750);
  }
  //
  // lcd.clear();    // Occhio che sfarfalla!
  //
  /*
    Serial.println("------------");
    Serial.println(Valore "A0");
    Serial.println(value);
    Serial.println("------------");
  */
}
```