## ANEXO A – Algoritmo de Ghassaei

```
//generalized wave freq detection with 38.5kHz sampling rate and interrupts
//by Amanda Ghassaei
//https://www.instructables.com/id/Arduino-Frequency-Detection/
//Sept 2012

/*
 * This program is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 3 of the License, or
 * (at your option) any later version.
 *
*/


//clipping indicator variables
boolean clipping = 0;

//data storage variables
byte newData = 0;
byte prevData = 0;
unsigned int time = 0;//keeps time and sends vales to store in timer[]
occasionally
int timer[10];//sstorage for timing of events
int slope[10];//storage for slope of events
unsigned int totalTimer;//used to calculate period
unsigned int period;//storage for period of wave
byte index = 0;//current storage index
float frequency;//storage for frequency calculations
int maxSlope = 0;//used to calculate max slope as trigger point
int newSlope;//storage for incoming slope data

//variables for decided whether you have a match
byte noMatch = 0;//counts how many non-matches you've received to reset
variables if it's been too long
byte slopeTol = 3;//slope tolerance- adjust this if you need
int timerTol = 10;//timer tolerance- adjust this if you need

//variables for amp detection
unsigned int ampTimer = 0;
byte maxAmp = 0;
byte checkMaxAmp;
byte ampThreshold = 30;//raise if you have a very noisy signal

void setup(){

  Serial.begin(9600);

  pinMode(13,OUTPUT);//led indicator pin
  pinMode(12,OUTPUT);//output pin

  cli();//diable interrupts
```

```
  //set up continuous sampling of analog pin 0 at 38.5kHz

  //clear ADCSRA and ADCSRB registers
  ADCSRA = 0;
  ADCSRB = 0;

  ADMUX |= (1 << REFS0); //set reference voltage
  ADMUX |= (1 << ADLAR); //left align the ADC value- so we can read highest
8 bits from ADCH register only

  ADCSRA |= (1 << ADPS2) | (1 << ADPS0); //set ADC clock with 32 prescaler-
16mHz/32=500kHz
  ADCSRA |= (1 << ADATE); //enabble auto trigger
  ADCSRA |= (1 << ADIE); //enable interrupts when measurement complete
  ADCSRA |= (1 << ADEN); //enable ADC
  ADCSRA |= (1 << ADSC); //start ADC measurements

  sei();//enable interrupts
}

ISR(ADC_vect) {//when new ADC value ready

  PORTB &= B11101111;//set pin 12 low
  prevData = newData;//store previous value
  newData = ADCH;//get value from A0
  if (prevData < 127 && newData >=127){//if increasing and crossing
midpoint
    newSlope = newData - prevData;//calculate slope
    if (abs(newSlope-maxSlope)<slopeTol){//if slopes are ==
      //record new data and reset time
      slope[index] = newSlope;
      timer[index] = time;
      time = 0;
      if (index == 0){//new max slope just reset
        PORTB |= B00010000;//set pin 12 high
        noMatch = 0;
        index++;//increment index
      }
      else if (abs(timer[0]-timer[index])<timerTol && abs(slope[0]-
newSlope)<slopeTol){//if timer duration and slopes match
        //sum timer values
        totalTimer = 0;
        for (byte i=0;i<index;i++){
          totalTimer+=timer[i];
        }
        period = totalTimer;//set period
        //reset new zero index values to compare with
        timer[0] = timer[index];
        slope[0] = slope[index];
        index = 1;//set index to 1
        PORTB |= B00010000;//set pin 12 high
        noMatch = 0;
      }
```

```
      else{//crossing midpoint but not match
        index++;//increment index
        if (index > 9){
          reset();
        }
      }
    }
    else if (newSlope>maxSlope){//if new slope is much larger than max
slope
      maxSlope = newSlope;
      time = 0;//reset clock
      noMatch = 0;
      index = 0;//reset index
    }
    else{//slope not steep enough
      noMatch++;//increment no match counter
      if (noMatch>9){
        reset();
      }
    }
  }

  if (newData == 0 || newData == 1023){//if clipping
    PORTB |= B00100000;//set pin 13 high- turn on clipping indicator led
    clipping = 1;//currently clipping
  }

  time++;//increment timer at rate of 38.5kHz

  ampTimer++;//increment amplitude timer
  if (abs(127-ADCH)>maxAmp){
    maxAmp = abs(127-ADCH);
  }
  if (ampTimer==1000){
    ampTimer = 0;
    checkMaxAmp = maxAmp;
    maxAmp = 0;
  }

}

void reset(){//clea out some variables
  index = 0;//reset index
  noMatch = 0;//reset match couner
  maxSlope = 0;//reset slope
}


void checkClipping(){//manage clipping indicator LED
  if (clipping){//if currently clipping
    PORTB &= B11011111;//turn off clipping indicator led
    clipping = 0;
  }
}
```

```
void loop(){

  checkClipping();

  if (checkMaxAmp>ampThreshold){
    frequency = 38462/float(period);//calculate frequency timer rate/period

    //print results
    Serial.print(frequency);
    Serial.println(" hz");
  }

  delay(100);//delete this if you want

  //do other stuff here
}
```