LAMPIRAN A

Listing Program

1) Program *Hydrophone*

```
#include <ESP8266WiFi.h>
/* ================================== Connection Wifi */
 //const char* ssid = "TUGAS_AKHIR";
//const char* password = "00000000";
const char* ssid = "tio";
 const char* password = "00000000";
 int ModbusTCP_port = 502;
/* ================================== Mengatur IP Address */
 IPAddress IP(192,168,43,102);
 IPAddress NETMASK(255,255,255,0);
 IPAddress NETWORK(192,168,43,1);
 IPAddress DNS(8,8,8,8);
/* ================================== */
//////// Required for Modbus TCP / IP /// Requerido para Modbus TCP/IP /////////
#define maxInputRegister 20
#define maxHoldingRegister 20
#define MB_FC_NONE 0
#define MB_FC_READ_REGISTERS 3 //implemented
#define MB_FC_WRITE_REGISTER 6 //implemented
#define MB_FC_WRITE_MULTIPLE_REGISTERS 16
// MODBUS Error Codes
#define MB_EC_NONE 0
#define MB_EC_ILLEGAL_FUNCTION 1
#define MB_EC_ILLEGAL_DATA_ADDRESS 2
#define MB_EC_ILLEGAL_DATA_VALUE 3
#define MB_EC_SLAVE_DEVICE_FAILURE 4
#define MB_TCP_TID 0
#define MB_TCP_PID 2
#define MB_TCP_LEN 4
#define MB_TCP_UID 6
#define MB_TCP_FUNC 7
#define MB_TCP_REGISTER_START 8
```

```
#define MB_TCP_REGISTER_NUMBER 10
byte ByteArray[260];
unsigned int MBHoldingRegister[maxHoldingRegister];
WiFiServer MBServer(ModbusTCP_port);
 unsigned long baca;
/* ================================== define pin
Relay  */
#define Sound_Sensor D2
int data_ADC;
void setup() {
 pinMode (Sound_Sensor, INPUT);
delay(100) ;
 WiFi.mode(WIFI_STA);
 WiFi.begin(ssid, password);
 WiFi.config(IP, NETWORK, NETMASK, DNS);
while (WiFi.status() != WL_CONNECTED) {
 delay(500);
}
 MBServer.begin();
}
void loop() {
WiFiClient client = MBServer.available();
 if (!client) {
 return;
 }
 boolean flagClientConnected = 0;
 byte byteFN = MB_FC_NONE;
 int Start;
 int WordDataLength;
 int ByteDataLength;
 int MessageLength;
while (client.connected()) {
 if(client.available())
 {
 flagClientConnected = 1;
 int i = 0;
 while(client.available())
 {
 ByteArray[i] = client.read();
```

```
 i++;
 }
 client.flush();
 digitalWrite(2,!digitalRead(2));// BLINK
/*
 MBHoldingRegister[0] = random(0,12);
 MBHoldingRegister[1] = random(0,12);
 MBHoldingRegister[2] = random(0,12);
 MBHoldingRegister[3] = random(0,12);
 MBHoldingRegister[4] = random(0,12);
 MBHoldingRegister[5] = random(0,12);
 MBHoldingRegister[6] = random(0,12);
 MBHoldingRegister[7] = random(0,12);
 MBHoldingRegister[8] = random(0,12);
 MBHoldingRegister[9] = random(0,12);
*/
  data_ADC=digitalRead (Sound_Sensor);
 unsigned long duration_high = pulseIn(Sound_Sensor, HIGH,
10000000) ;
  Serial.println (duration_high);
  unsigned long duration_low = pulseIn(Sound_Sensor, LOW,
10000000) ;
  Serial.println (duration_low);
  float periode = duration_high+duration_low;
  float  frekuensi = 1.0*1000000.0/(duration_high+duration_low);
  Serial.println (frekuensi);
  Serial.println ("selesai");
  MBHoldingRegister[0] = frekuensi;
  MBHoldingRegister[1] = duration_low;
  MBHoldingRegister[2] = duration_high;
  MBHoldingRegister[3] = data_ADC;
  MBHoldingRegister[4] = periode;

byteFN = ByteArray[MB_TCP_FUNC];
 Start                                                    =
word(ByteArray[MB_TCP_REGISTER_START],ByteArray[MB_
TCP_REGISTER_START+1]);
```

```
 WordDataLength                                    =
word(ByteArray[MB_TCP_REGISTER_NUMBER],ByteArray[M
B_TCP_REGISTER_NUMBER+1]);
 }
switch(byteFN) {
 case MB_FC_NONE:
 break;
 case MB_FC_READ_REGISTERS: // 03 Read Holding Registers
 ByteDataLength = WordDataLength * 2;
 ByteArray[5] = ByteDataLength + 3; //Number of bytes after this
one.
 ByteArray[8] = ByteDataLength; //Number of bytes after this one
(or number of bytes of data).
 for(int i = 0; i < WordDataLength; i++)
 {
 ByteArray[ 9 + i * 2] = highByte(MBHoldingRegister[Start + i]);
 ByteArray[10 + i * 2] = lowByte(MBHoldingRegister[Start + i]);
 }
 MessageLength = ByteDataLength + 9;
 client.write((const uint8_t *)ByteArray,MessageLength);
 byteFN = MB_FC_NONE;
 break;
 case MB_FC_WRITE_REGISTER: // 06 Write Holding Register
 MBHoldingRegister[Start]                          =
word(ByteArray[MB_TCP_REGISTER_NUMBER],ByteArray[M
B_TCP_REGISTER_NUMBER+1]);
 ByteArray[5] = 6; //Number of bytes after this one.
 MessageLength = 12;
 client.write((const uint8_t *)ByteArray,MessageLength);
 byteFN = MB_FC_NONE;
 break;
 case MB_FC_WRITE_MULTIPLE_REGISTERS: //16 Write
Holding Registers
 ByteDataLength = WordDataLength * 2;
 ByteArray[5] = ByteDataLength + 3; //Number of bytes after this
one.
 for(int i = 0; i < WordDataLength; i++)
 {
```

```
 MBHoldingRegister[Start + i] = word(ByteArray[ 13 + i *
2],ByteArray[14 + i * 2]);
 }
 MessageLength = 12;
 client.write((const uint8_t *)ByteArray,MessageLength);
 byteFN = MB_FC_NONE;
 break;
 }
 }
}
```

2) Program Transmiter

```
#include <ESP8266WiFi.h>
/* ===================================== Connection
Wifi */
// const char* ssid = "TUGAS_AKHIR";
 //const char* password = "00000000";
const char* ssid = "tio";
 const char* password = "00000000";
 int ModbusTCP_port = 502;
/* ====================================== Mengatur IP
Address */
 IPAddress IP(192,168,43,104);
 IPAddress NETMASK(255,255,255,0);
 IPAddress NETWORK(192,168,43,1);
 IPAddress DNS(8,8,8,8);
/* ====================================== */
//////// Required for Modbus TCP / IP /// Requerido para Modbus
TCP/IP /////////
#define maxInputRegister 20
#define maxHoldingRegister 20
#define MB_FC_NONE 0
#define MB_FC_READ_REGISTERS 3 //implemented
#define MB_FC_WRITE_REGISTER 6 //implemented
#defineMB_FC_WRITE_MULTIPLE_REGISTERS 16
#define MB_EC_NONE 0
#define MB_EC_ILLEGAL_FUNCTION 1
#define MB_EC_ILLEGAL_DATA_ADDRESS 2
#define MB_EC_ILLEGAL_DATA_VALUE 3
```

```
#define MB_EC_SLAVE_DEVICE_FAILURE 4
#define MB_TCP_TID 0
#define MB_TCP_PID 2
#define MB_TCP_LEN 4
#define MB_TCP_UID 6
#define MB_TCP_FUNC 7
#define MB_TCP_REGISTER_START 8
#define MB_TCP_REGISTER_NUMBER 10
byte ByteArray[260];
unsigned int MBHoldingRegister[maxHoldingRegister];
WiFiServer MBServer(ModbusTCP_port);
 unsigned long baca;
 int play;
#include <DFPlayer_Mini_Mp3.h> //memanggil library DFPlayer
mini
#include <SoftwareSerial.h>    //memanggil library SoftwareSerial
SoftwareSerial mSerial(0,4); // Declare pin RX & TX
//SoftwareSerial nSerial(14,2); // Declare pin RX & TX
void setup() {
 WiFi.mode(WIFI_STA);
 WiFi.begin(ssid, password);
 WiFi.config(IP, NETWORK, NETMASK, DNS);
 while (WiFi.status() != WL_CONNECTED) {
 delay(500);
 }
 MBServer.begin();
 mSerial.begin (9600);
 mp3_set_serial (mSerial); //set softwareSerial for DFPlayer
delay(10);   //wait 1ms for respon command
 mp3_set_volume (100); //set Volume module DFPlayer
 delay(1000);
}
void loop() {
  WiFiClient client = MBServer.available();
 if (!client) {
 return;
 }
 boolean flagClientConnected = 0;
 byte byteFN = MB_FC_NONE;
```

```
int Start;
int WordDataLength;
int ByteDataLength;
int MessageLength;
// Modbus TCP/IP
while (client.connected()) {
if(client.available())
{
flagClientConnected = 1;
int i = 0;
while(client.available())
{
ByteArray[i] = client.read();
i++;
}
client.flush();
digitalWrite(2,!digitalRead(2));// BLINK
play = MBHoldingRegister[0];
if(play == 100) { mp3_play(1);}
if(play == 200) { mp3_play(2);}
if(play == 300) { mp3_play(3);}
if(play == 400) { mp3_play(4);}
if(play == 500) { mp3_play(5);}
if(play == 600) { mp3_play(6);}
if(play == 700) { mp3_play(7);}
if(play == 800) { mp3_play(8);}
if(play == 900) { mp3_play(9);}
if(play == 1000) { mp3_play(10);}
if(play == 2000) { mp3_play(20);}
if(play == 8000) { mp3_play(80);}
if(play == 10000) { mp3_play(100);}
if(play == 14000) { mp3_play(140);}
byteFN = ByteArray[MB_TCP_FUNC];

word(ByteArray[MB_TCP_REGISTER_START],ByteArray[MB_
TCP_REGISTER_START+1]);
WordDataLength                                    =
word(ByteArray[MB_TCP_REGISTER_NUMBER],ByteArray[M
B_TCP_REGISTER_NUMBER+1]);
```

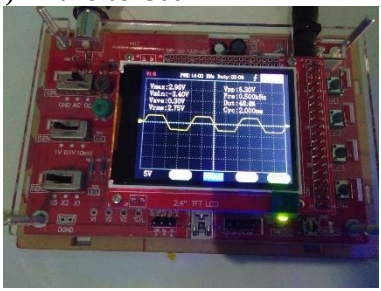
```
 }
 switch(byteFN) {
 case MB_FC_NONE:
 break;
 case MB_FC_READ_REGISTERS: // 03 Read Holding Registers
 ByteDataLength = WordDataLength * 2;
 ByteArray[5] = ByteDataLength + 3; //Number of bytes after this
one.
 ByteArray[8] = ByteDataLength; //Number of bytes after this one
(or number of bytes of data).
 for(int i = 0; i < WordDataLength; i++)
 {
 ByteArray[ 9 + i * 2] = highByte(MBHoldingRegister[Start + i]);
 ByteArray[10 + i * 2] = lowByte(MBHoldingRegister[Start + i]);
 }
 MessageLength = ByteDataLength + 9;
 client.write((const uint8_t *)ByteArray,MessageLength);
 byteFN = MB_FC_NONE;
 break;
 case MB_FC_WRITE_REGISTER: // 06 Write Holding Register
 MBHoldingRegister[Start]                                    =
word(ByteArray[MB_TCP_REGISTER_NUMBER],ByteArray[M
B_TCP_REGISTER_NUMBER+1]);
 ByteArray[5] = 6; //Number of bytes after this one.
 MessageLength = 12;
 client.write((const uint8_t *)ByteArray,MessageLength);
 byteFN = MB_FC_NONE;
 break;
 case MB_FC_WRITE_MULTIPLE_REGISTERS: //16 Write
Holding Registers
 ByteDataLength = WordDataLength * 2;
 ByteArray[5] = ByteDataLength + 3; //Number of bytes after this
one.
 for(int i = 0; i < WordDataLength; i++)
 {
 MBHoldingRegister[Start + i] = word(ByteArray[ 13 + i *
2],ByteArray[14 + i * 2]);
 }
 MessageLength = 12;
```

```
 client.write((const uint8_t *)ByteArray,MessageLength);
 byteFN = MB_FC_NONE;
 break;
 }
 }
}
```

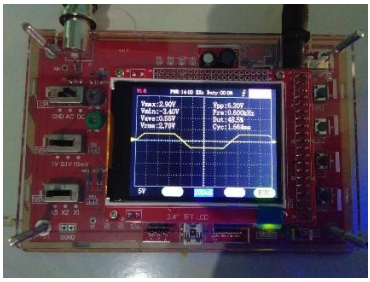
LAMPIRAN B

Proses Kalibrasi Transmiter

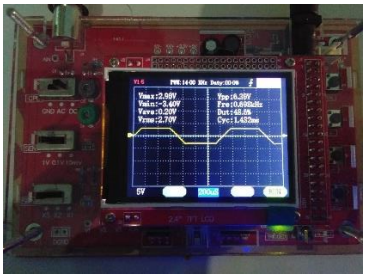
1) Kalibrasi 500Hz



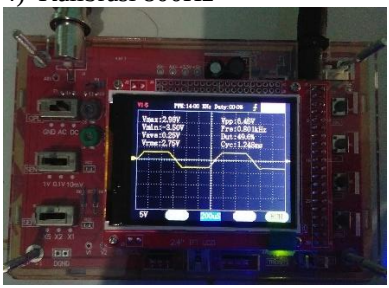
2) Kalibrasi 600Hz



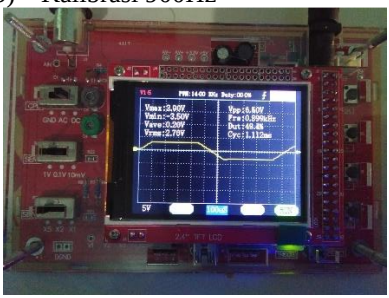
3) Kalibrasi 700Hz



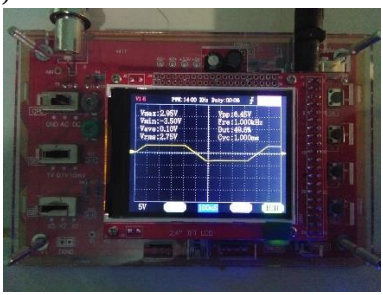
4) Kalibrasi 800Hz



5) Kalibrasi 900Hz



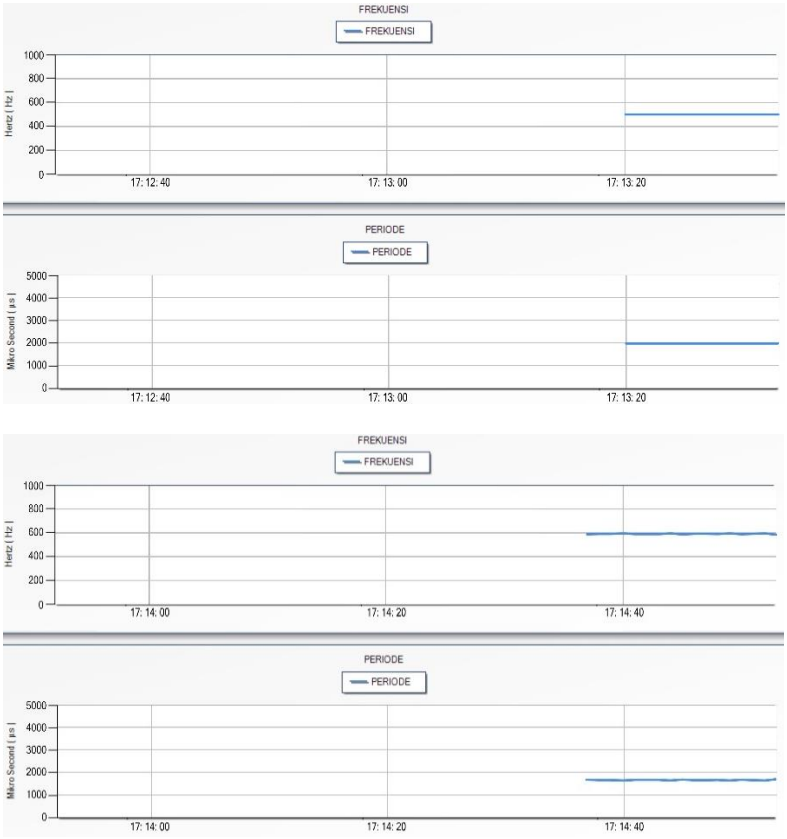
6) Kalibrasi 1000Hz



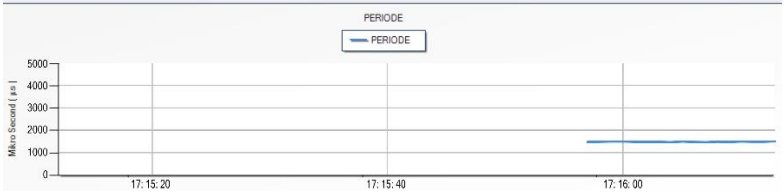
LAMPIRAN C

Tampilan Scada Wintr

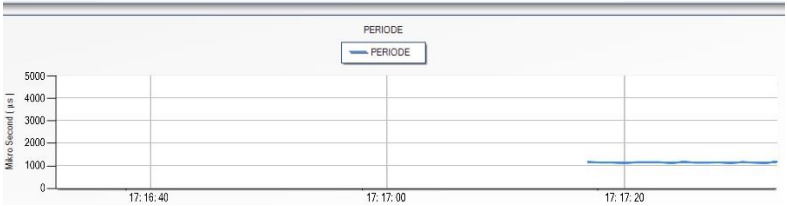
Untuk Frekuensi dan Periode Pada Kedalaman 1 Cm



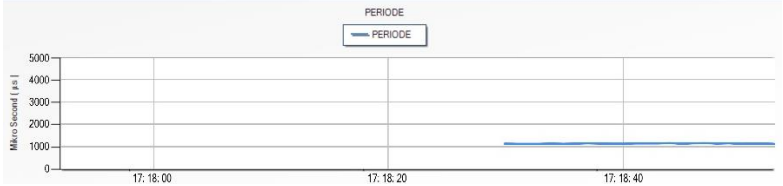

FREKUENSI
FREKUENSI
1000
800
600
400
200
0
Hertz [Hz]
17: 15: 20
17: 15: 40
17: 16: 00


PERIODE
PERIODE
5000
4000
3000
2000
1000
0
Mikro Second [µs]
17: 15: 20
17: 15: 40
17: 16: 00

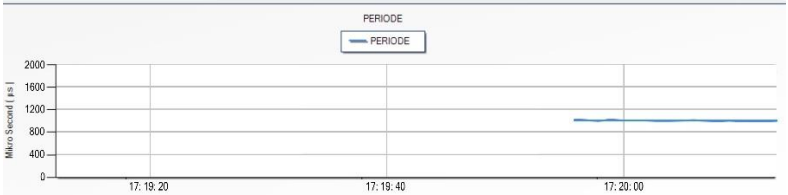

FREKUENSI
FREKUENSI
1000
800
600
400
200
0
Hertz [Hz]
17: 16: 40
17: 17: 00
17: 17: 20

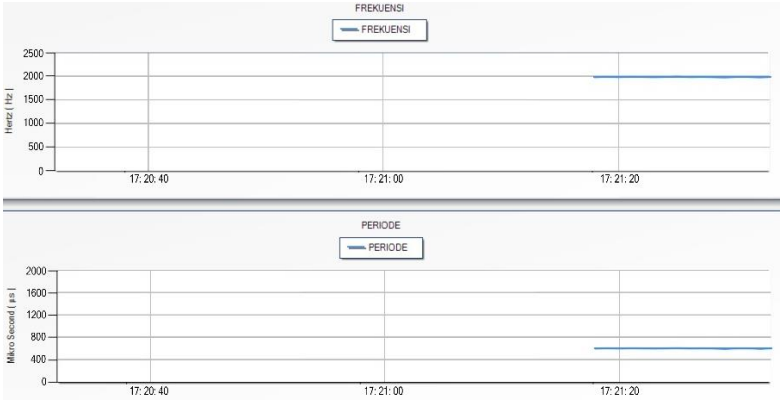

PERIODE
PERIODE
5000
4000
3000
2000
1000
0
Mikro Second [µs]
17: 16: 40
17: 17: 00
17: 17: 20


FREKUENSI
FREKUENSI
1000
800
600
400
200
0
Hertz [Hz]
17: 18: 00
17: 18: 20
17: 18: 40

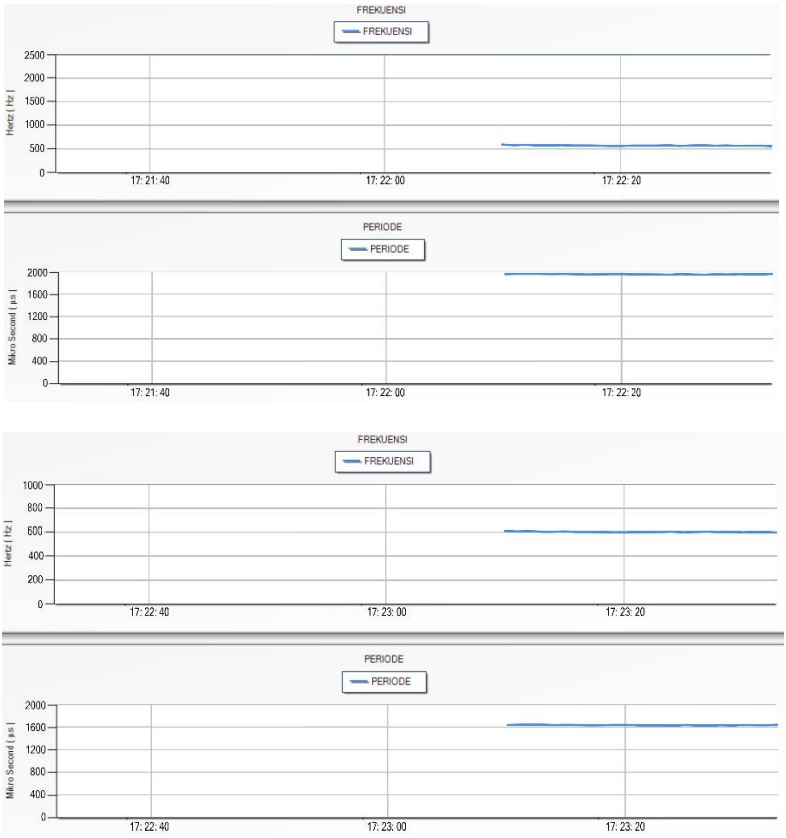

PERIODE
PERIODE
5000
4000
3000
2000
1000
0
Mikro Second [µs]
17: 18: 00
17: 18: 20
17: 18: 40

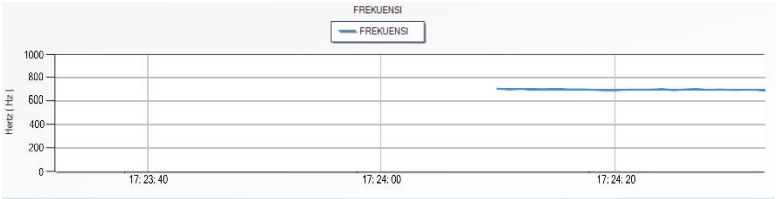

FREKUENSI
FREKUENSI
2500
2000
1500
1000
500
0
Hertz [Hz]
17: 19: 20
17: 19: 40
17: 20: 00

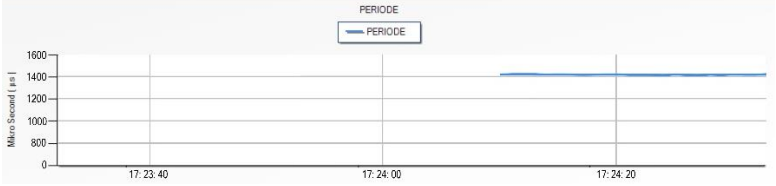

PERIODE
PERIODE
2000
1600
1200
800
400
0
Mikro Second [µs]
17: 19: 20
17: 19: 40
17: 20: 00


FREKUENSI
FREKUENSI
2500
2000
1500
1000
500
0
Hertz (Hz)
17: 20: 40
17: 21: 00
17: 21: 20
PERIODE
PERIODE
2000
1600
1200
800
400
0
Mikro Second (µs)
17: 20: 40
17: 21: 00
17: 21: 20

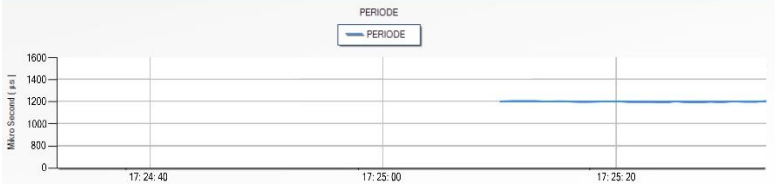
Tampilan Scada Wintr
Untuk Frekuensi dan Periode Pada kedalaman 2 Cm

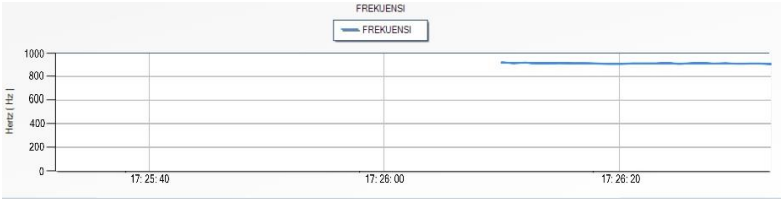


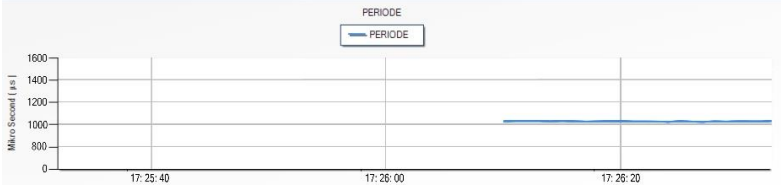

FREKUENSI
FREKUENSI
1000
800
600
400
200
0
Hertz [Hz]
17: 23: 40
17: 24: 00
17: 24: 20

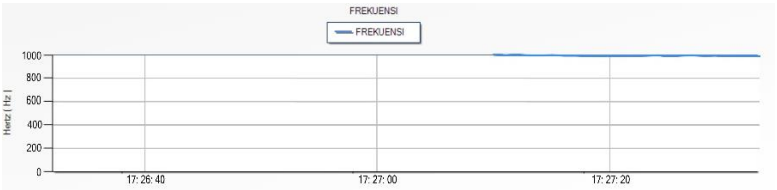

PERIODE
PERIODE
1600
1400
1200
1000
800
0
Mikro Second [µs]
17: 23: 40
17: 24: 00
17: 24: 20

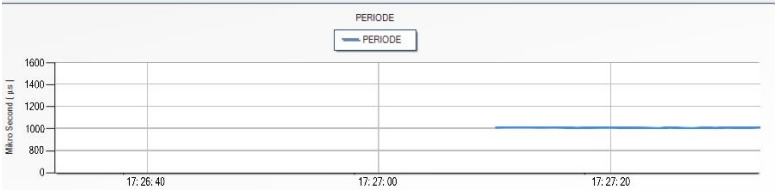

FREKUENSI
FREKUENSI
1000
800
600
400
200
0
Hertz [Hz]
17: 24: 40
17: 25: 00
17: 25: 20

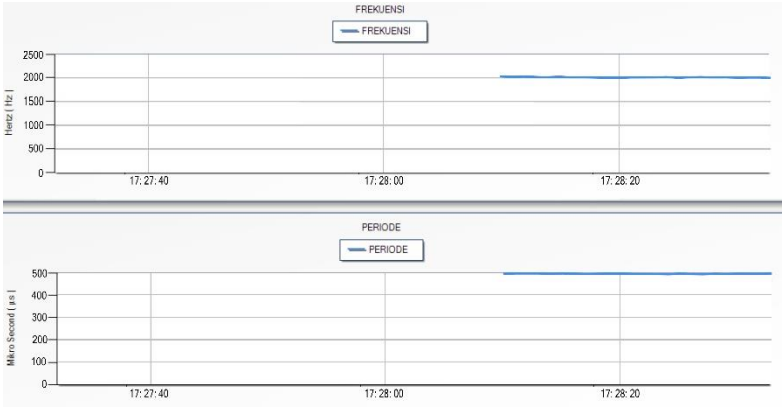

PERIODE
PERIODE
1600
1400
1200
1000
800
0
Mikro Second [µs]
17: 24: 40
17: 25: 00
17: 25: 20


FREKUENSI
FREKUENSI
1000
800
600
400
200
0
Hertz [Hz]
17: 25: 40
17: 26: 00
17: 26: 20

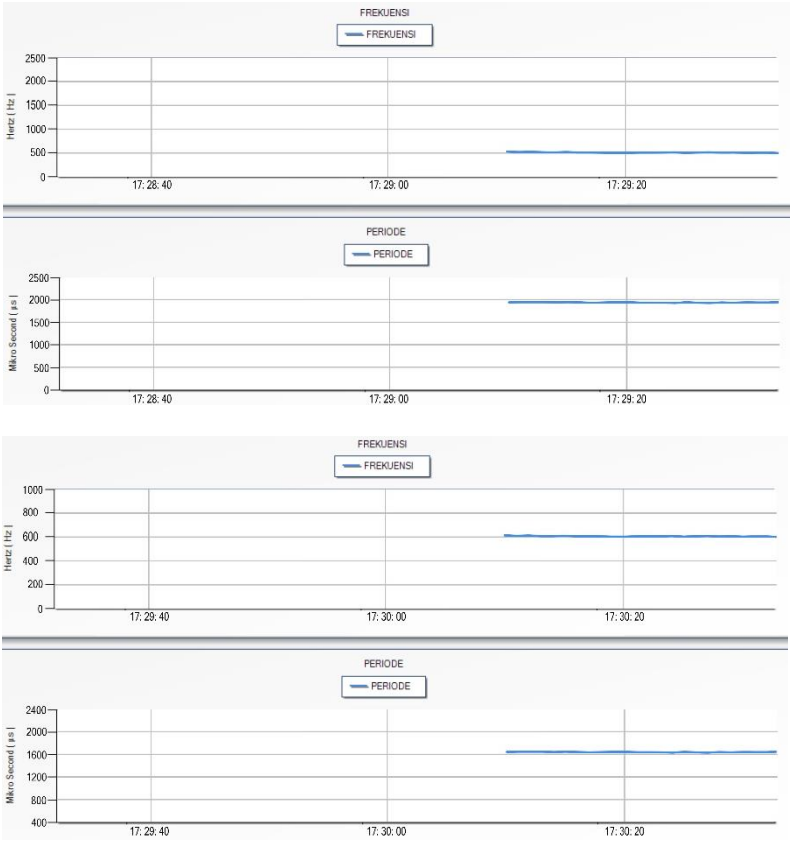

PERIODE
PERIODE
1600
1400
1200
1000
800
0
Mikro Second [µs]
17: 25: 40
17: 26: 00
17: 26: 20


FREKUENSI
FREKUENSI
1000
800
600
400
200
0
Hertz [Hz]
17: 26: 40
17: 27: 00
17: 27: 20

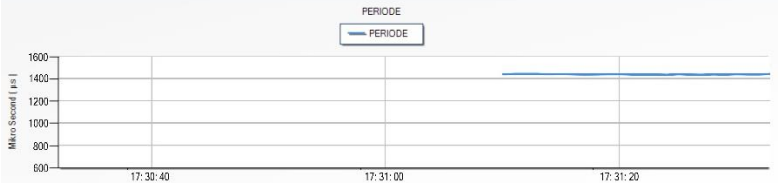

PERIODE
PERIODE
1600
1400
1200
1000
800
0
Mikro Second [µs]
17: 26: 40
17: 27: 00
17: 27: 20

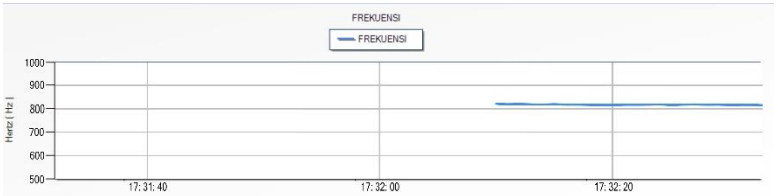

FREKUENSI
FREKUENSI
2500
2000
1500
1000
500
0
Hertz (Hz)
17: 27: 40
17: 28: 00
17: 28: 20
PERIODE
PERIODE
500
400
300
200
100
0
Mikro Second (µs)
17: 27: 40
17: 28: 00
17: 28: 20

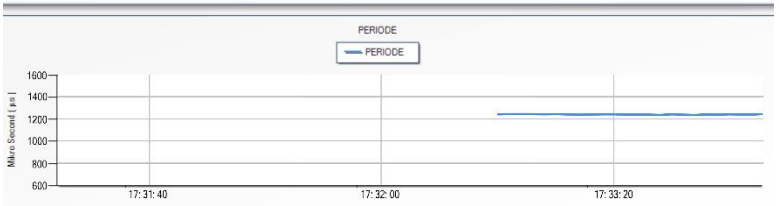
Tampilan Scada Wintr
Untuk Frekuensi dan Periode Pada kedalaman 3 Cm

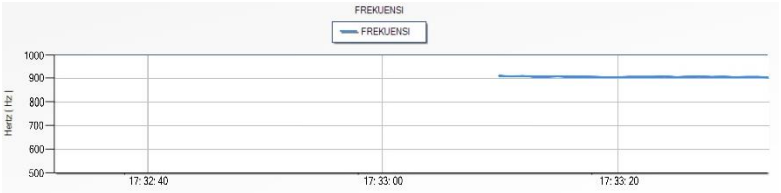


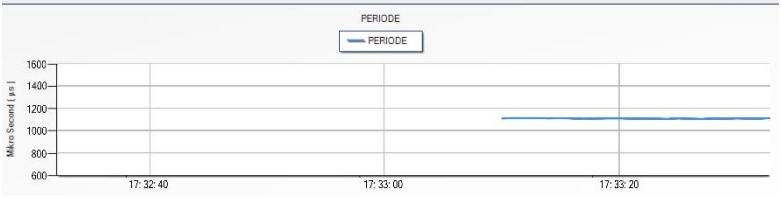

FREKUENSI
FREKUENSI
Hertz [Hz]
1000
900
800
700
600
500
17: 30: 40
17: 31: 00
17: 31: 20

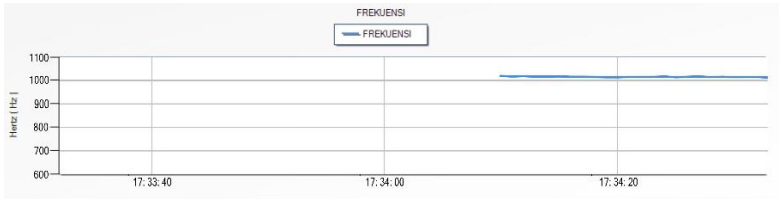

PERIODE
PERIODE
Mikro Second [µs]
1600
1400
1200
1000
800
600
17: 30: 40
17: 31: 00
17: 31: 20

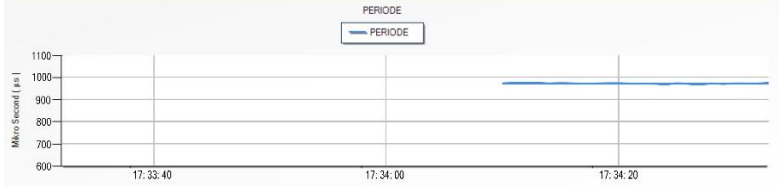

FREKUENSI
FREKUENSI
Hertz [Hz]
1000
900
800
700
600
500
17: 31: 40
17: 32: 00
17: 32: 20

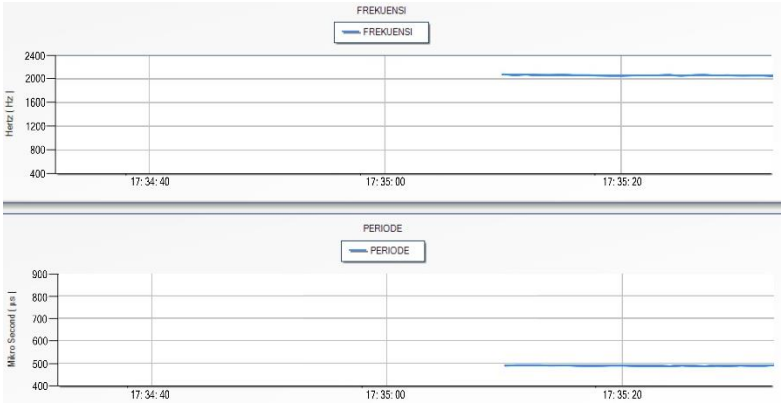

PERIODE
PERIODE
Mikro Second [µs]
1600
1400
1200
1000
800
600
17: 31: 40
17: 32: 00
17: 33: 20


FREKUENSI
FREKUENSI
1000
900
800
700
600
500
Hertz (Hz)
17: 32: 40
17: 33: 00
17: 33: 20

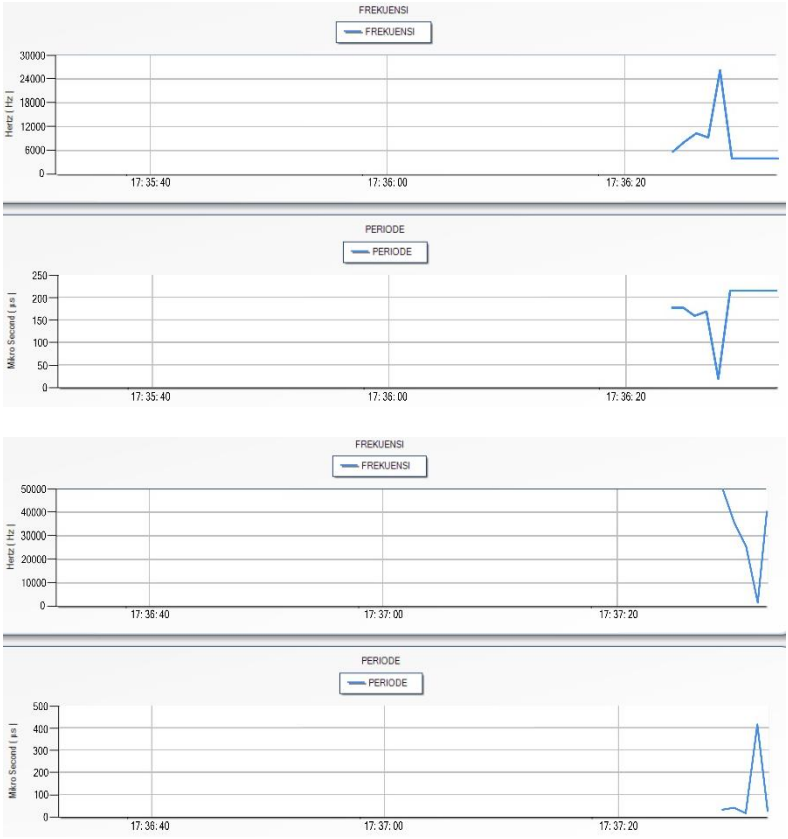

PERIODE
PERIODE
1600
1400
1200
1000
800
600
Mikro Second (µs)
17: 32: 40
17: 33: 00
17: 33: 20

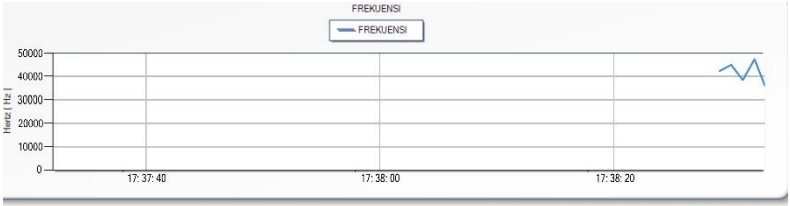

FREKUENSI
FREKUENSI
1100
1000
900
800
700
600
Hertz (Hz)
17: 33: 40
17: 34: 00
17: 34: 20

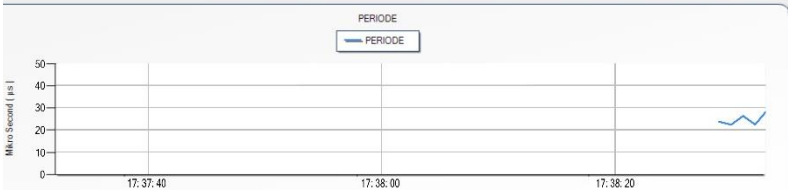

PERIODE
PERIODE
1100
1000
900
800
700
600
Mikro Second (µs)
17: 33: 40
17: 34: 00
17: 34: 20


FREKUENSI
FREKUENSI
2400
2000
1600
1200
800
400
Hertz (Hz)
17: 34: 40
17: 35: 00
17: 35: 20
PERIODE
PERIODE
900
800
700
600
500
400
Mikro Second (µs)
17: 34: 40
17: 35: 00
17: 35: 20

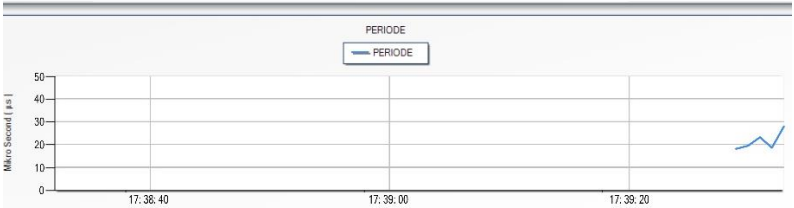
Tampilan Scada Wintr
Untuk Frekuensi dan Periode Pada kedalaman 4 Cm

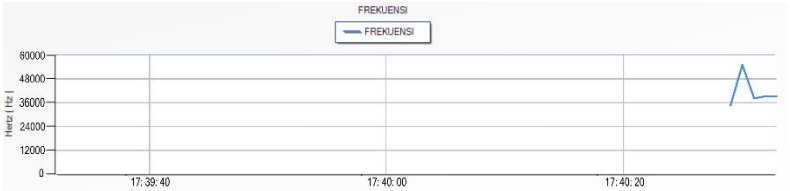


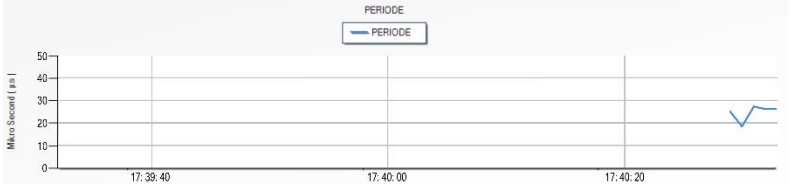

FREKUENSI
FREKUENSI
50000
40000
30000
20000
10000
0
Hertz [Hz]
17: 37: 40
17: 38: 00
17: 38: 20


PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second [µs]
17: 37: 40
17: 38: 00
17: 38: 20

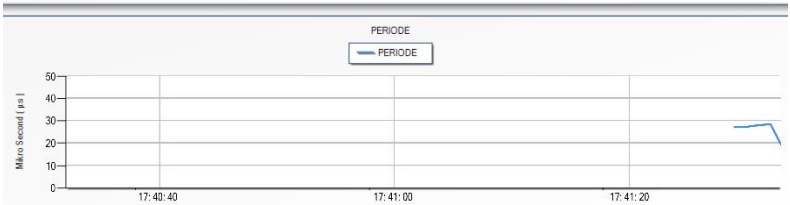

FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz [Hz]
17: 38: 40
17: 39: 00
17: 39: 20

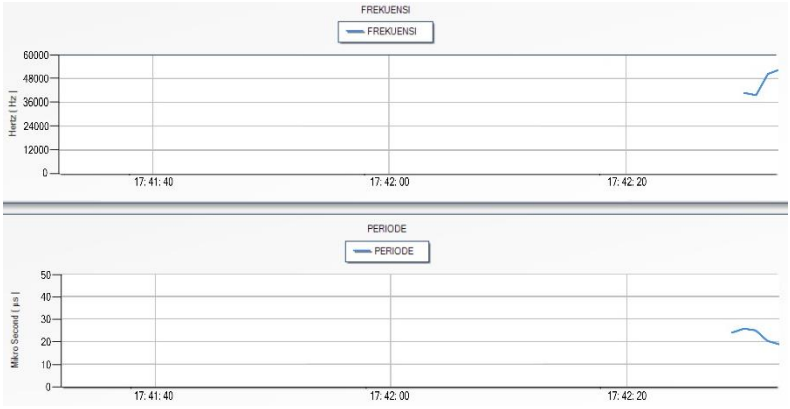

PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second [µs]
17: 38: 40
17: 39: 00
17: 39: 20


FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz [Hz]
17: 39: 40
17: 40: 00
17: 40: 20

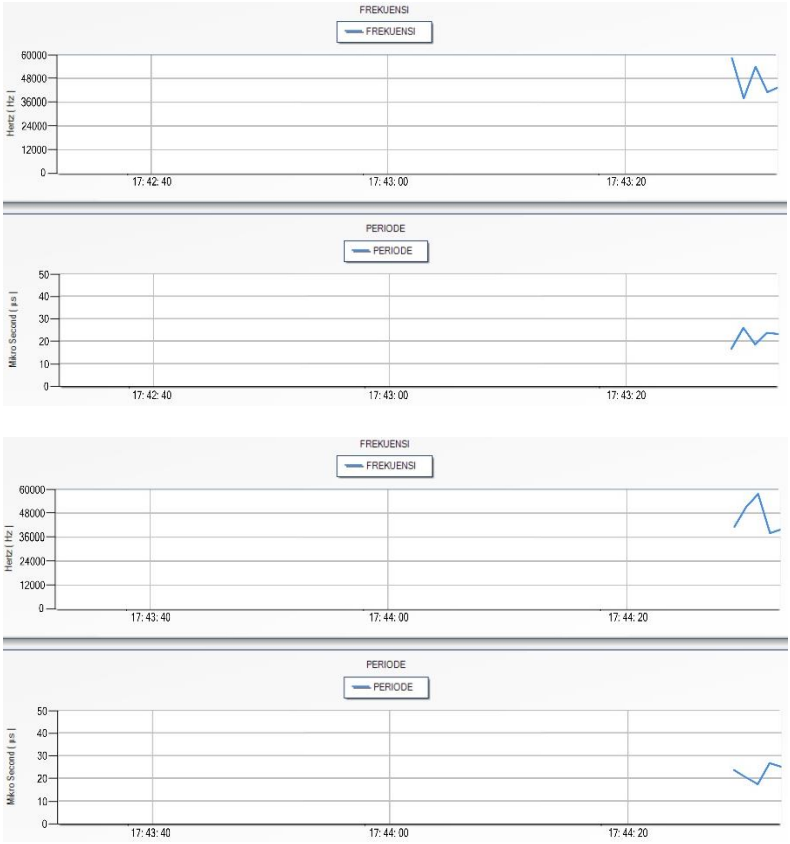

PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second [µs]
17: 39: 40
17: 40: 00
17: 40: 20


FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz [Hz]
17: 40: 40
17: 41: 00
17: 41: 20

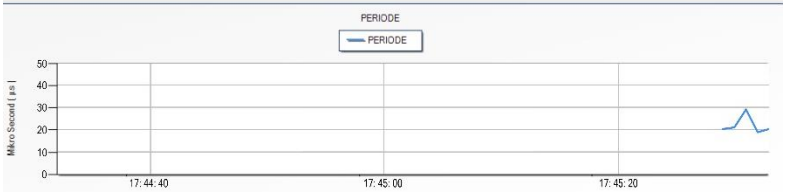

PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second [µs]
17: 40: 40
17: 41: 00
17: 41: 20

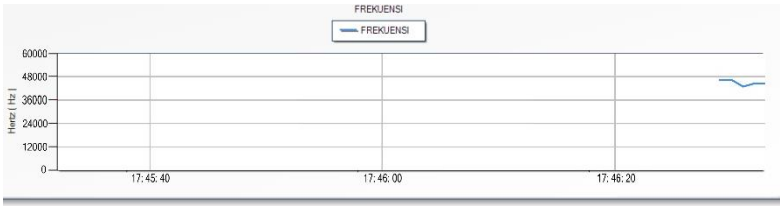

FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz [Hz]
17: 41: 40
17: 42: 00
17: 42: 20
PERIODE
PERIODE
50
40
30
20
10
0
Micro Second [µs]
17: 41: 40
17: 42: 00
17: 42: 20

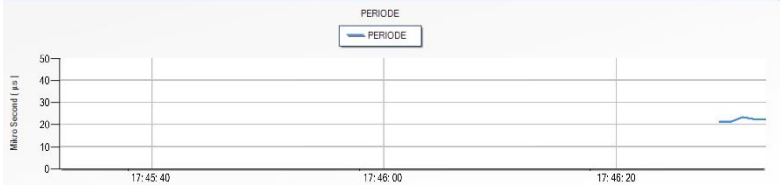
Tampilan Scada Wintr
Untuk Frekuensi dan Periode Pada kedalaman 5 Cm

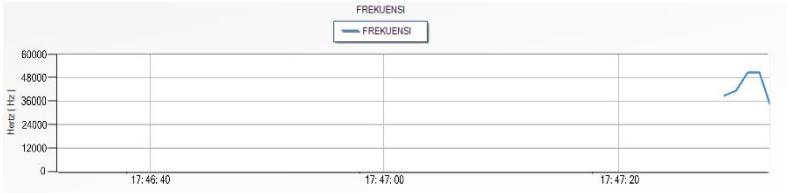



FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz (Hz)
17: 44: 40
17: 45: 00
17: 45: 20

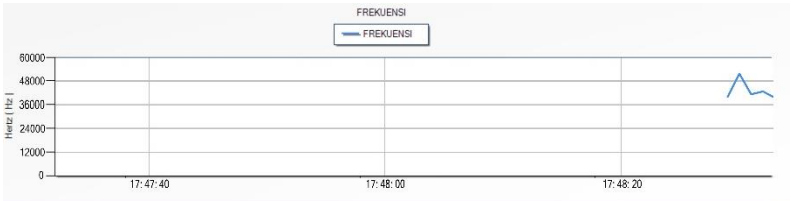

PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second (µs)
17: 44: 40
17: 45: 00
17: 45: 20

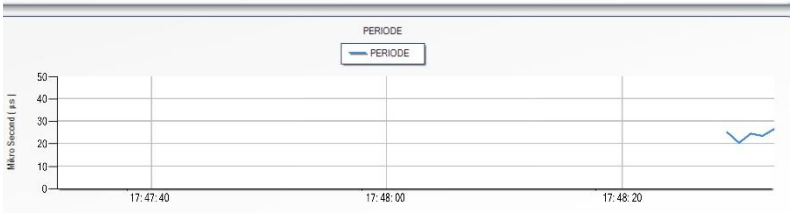

FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz (Hz)
17: 45: 40
17: 46: 00
17: 46: 20


PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second (µs)
17: 45: 40
17: 46: 00
17: 46: 20


FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz (Hz)
17: 46: 40
17: 47: 00
17: 47: 20


PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second (µs)
17: 46: 40
17: 47: 00
17: 48: 20


FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz (Hz)
17: 47: 40
17: 48: 00
17: 48: 20


PERIODE
PERIODE
50
40
30
20
10
0
Mikro Second (µs)
17: 47: 40
17: 48: 00
17: 48: 20


FREKUENSI
FREKUENSI
60000
48000
36000
24000
12000
0
Hertz (Hz)
17: 48: 40
17: 49: 00
17: 49: 20
PERIODE
PERIODE
50
40
30
20
10
0
Micro Second (µs)
17: 48: 40
17: 49: 00
17: 49: 20