TEKNOKES, Vol. 12, No. 1, Maret 2019, pp:5-13
DOI: 10.35882/teknokes.v12i1.2 **ISSN:2407-8964**

Electrocardiograph Simulator Berbasis Mikrokontroler

I Dewa Gede Budi Whinangun[#], Andjar Pudji, M. Ridha Makruf, Bedjo Utomo, Sari Luthfiyah
Jurusan Teknik Elektromedik Poltekkes Kemenkes, Surabaya
Jl. Pucang Jajar Timur No. 10, Surabaya, 60245, Indonesia
[#]budiwhinangun@gmail.com, andjar@poltekkesdepkes-sby.ac.id, m.reedha@gmail.com,

Abstrak— *Electrocardiograph* (ECG) menjadi salah satu ilmu diagnostik yang sering dipelajari dalam mendiagnosis dan untuk terapi penyakit jantung. Mengingat pentingnya alat ECG *recorder*, maka diperlukan pengecekan fungsi alat ECG *recorder* yaitu dengan cara melakukan prosedur kalibrasi alat menggunakan Phantom ECG. Tujuan dari penelitian ini adalah membuat ECG *Simulator* untuk alat ECG 12 channel yang meliputi lead I, lead II, lead III, aVR, aVF, aVL, V1, V2, V3, V4, V5, dan V6 dan melengkapinya dengan selektor pemilihan sensitivitas serta menggunakan. Metode pembentukan sinyal jantung menggunakan DAC tipe MCP 4921 dengan mikrokontroler Atmega2560 dan untuk tampilan pengaturanya menggunakan LCD Karakter 2x16. Berdasarkan hasil pengukuran didapat nilai tingkat kesalahan sebesar 0.187% sensitivitas 0.5mV dan 0.327% sensitivitas 1.0mV pada setting BPM 30, didapat nilai tingkat kesalahan sebesar 1.173% sensitivitas 0.5mV dan 1.060% sensitivitas 1.0mV pada setting BPM 60, didapat nilai tingkat kesalahan sebesar 0.797% sensitivitas 0.5mV dan 0.739% sensitivitas 1.0mV pada setting BPM 120, didapat nilai tingkat kesalahan sebesar 0.269% sensitivitas 0.5mV dan 0.381% sensitivitas 1.0mV pada setting BPM 180 dan 0.010% sensitivitas 0.5mV dan 0.616% sensitivitas 1.0mV pada setting BPM 240. Modul ECG *Simulator* dilengkapi dengan fitur charge baterai dan biaya pembuatan yang lebih murah dibandingkan dengan alat pabrikan.

Keywords—BPM; Sensitivitas; ECG Simulator

I. PENDAHULUAN

Electrocardiograph (ECG) pertama kali ditemukan oleh Einthoven (1904). ECG menjadi salah satu ilmu diagnostik yang sering dipelajari dalam penyembuhan modern, salah satunya untuk mendiagnosis dan untuk terapi penyakit yang disebabkan oleh jantung dengan memanfaatkan visualisasi rekaman sinyal ECG. Pemahaman tentang bentuk sinyal dari alat ECG diperlukan dalam mempelajari alat. Berhubungan dengan hal tersebut, maka diperlukan suatu alat yang mampu mensimulasikan sinyal tubuh manusia yang dapat membantu dalam proses pembelajaran. ECG *Simulator* atau biasa disebut Phantom ECG pada prinsipnya merupakan suatu generator sinyal dengan bentuk sinyal "ECG like" atau sinyal ECG yang telah terekam. Perangkat ini berguna untuk pengetesan perangkat ECG pada saat pemeliharaan dan perbaikan. Perangkat ini bisa direalisasikan berbasis mikrokontroler, rangkaian analog biasa atau berbasis PC[1].

Pada penelitian sebelumnya ECG *Simulator* sudah pernah dibuat oleh Candan Caner dengan judul "The Programable ECG *Simulator*" dalam penelitian Candan Caner Phantom ECG menggunakan PIC mikrokontroler dan DAC MCP4922. Pada penelitian tersebut pemilihan Heart Rate hanya mencapai 120 BPM. Pada tahun 2014 juga dilakukan penelitian oleh Valais tentang "*Design and Construction of a Prototype ECG Simulator"*, dengan menggunakan Atmega 8515 dan diperoleh range Heart Rate 60-180 BPM [2]. Sedangkan di Kampus Teknik Elektromedik ECG *Simulator* pernah dibuat oleh Gregorius Mario Tani pada tahun 2017 dengan judul *Simulator* ECG (*Phantom Electrocardiograph*). Metode yang digunakan adalah menggunakan IC Digital To Analog (DAC) tipe MCP43921 untuk membentuk sinyal jantung yang diinginkan. Menurut data hasil penelitian didapatkan pengukuran BPM 90 memiliki nilai *error* sebesar 4,13%, pada BPM 60 nilai *error* sebesar 3,9 % dan pada BPM 110 memiliki *error* sebesar 2,63%. Alat tersebut hanya memiliki nilai *heart rate* dengan rentang 30-110 Beat Per Minute[3]. Selanjutnya tahun 2017 Ni Nyoman Sri Malini juga membuat alat ECG *Simulator* menggunakan pembentukan sinyal ECG dengan IC counter 4017 serta clock NE555. Berbeda dengan IC DAC yang dapat dibentuk sesuai plotting gambar asli dari sinyal jantung, sedangkan IC counter 4017 yang digunakan Ni Nyoman Sri Malini masih memiliki kelemahan pada penggunaan kapasitor, sehingga menyebabkan bentuk gelombang ECG yang kurang sempurna pada segmen S-T serta besar nilai kapasitor berpengaruh terhadap lebar segmen dan interval pada gelombang PQRST. ECG *Simulator* tersebut memiliki rentang BPM 30 – 240. Selain hal tersebut, terkait BPM juga tampak kelemahannya pada frekuensi yang dikeluarkan oleh mikrokontroler berubah-rubah sehingga BPM tidak stabil. Untuk nilai *error* interval dan segmen didapat pada interval P-R memiliki *error* sebesar 1,86%, pada interval QRS memiliki *error* sebesar 0,77% pada interval S-T *error* sebesar 196% serta pada interval Q-T *error* sebesar 7,11%[4]. Berikutnya pada tahun 2018 ECG *Simulator* juga pernah dibuat oleh M. Ziko Alamanda dengan judul Phantom ECG. Metode yang digunakan adalah menggunakan IC Digital To Analog (DAC) tipe MCP4921 untuk membentuk sinyal jantung yang

diinginkan. ECG *Simulator* tersebut memiliki rentang BPM 30 – 240 dan terdapat pemilihan sensitivitas 0,5 mV dan 1,0 mV. Karya ilmiah ini memiliki kelemahan pada output bidang frontal (RA, LA, LL dan RL), sehingga tidak dapat digunakan untuk mensimulasi peralatan ECG dengan lead lengkap (12 lead)[5].

Berdasarkan hasil identifikasi dari latar belakang masalah di atas, maka penulis ingin menyempurnakan alat yang sudah dibuat sebelumnya dengan membuat ECG *Simulator* untuk alat ECG 12 channel yang meliputi lead I, lead II, lead III, aVR, aVF, aVL, V1, V2, V3, V4, V5, dan V6, dan melengkapinya dengan selektor pemilihan sensitivitas serta menggunakan metode pembentukan sinyal jantung melalui DAC tipe MCP 4921 dengan mikrokontroler ATmega 2560. Selain itu alat yang akan dibuat juga dilengkapi dengan fitur charge baterai sehingga diharapkan menghemat biaya penggantian baterai dan biaya pembuatan yang lebih murah dibandingkan dengan alat pabrikan.

II. BAHAN DAN METODE

A. Batasan Masalah

Penelitian ini memiliki output 10 sadapan ECG (RA, LA, LL, RL, V1, V2, V3, V4, V5, dan V6). Sehingga dapat mengeluarkan output lead I, II, III, aVR, aVL, aVF, V1, V2, V3, V4, V5, dan V6 apabila direkam dengan ECG recorder. Selain itu terdapat pemilihan sensitivitas dengan nilai 0.5, 1.0, dan 2.0 mV, serta memiliki rentang BPM 30, 60, 120, 180, dan 240 dengan kontrol saklar rotary.

1) Alat dan Bahan

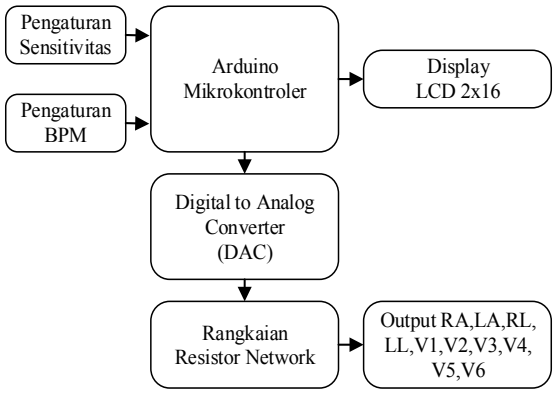
Penelitian ini menggunakan IC DAC MCP4921 untuk pembetukan sinyalnya. Hasil ouput dari IC DAC selanjutnya di olah dengan rangkaian *resistror network* yang menghasilkan output 10 sadapan ECG (RA, LA, LL, RL, V1, V2, V3, V4, V5, dan V6). Sebagai display pengaturan alat ini menggukan LCD karakter 2 x 16. Hasil output dari alat ini diuji menggunakan ECG Recorder (Digital *Electrocardiograph*, Model : ECG-9012A, SN : ECG-9012A120435). Selanjutnya ECG *Simulator* (Bio Tek, Model: ECGPlus, SN: 91219) digunakan sebagai alat pembanding hasil output dari alat.

2) Experiment

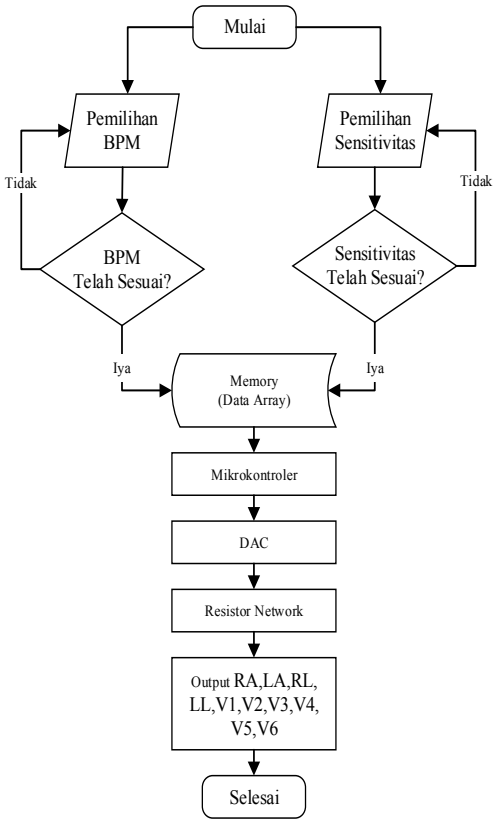
Dalam penelitian ini, setelah alat selesai untuk dikerjakan, hasil sinyal output alat direkam menggukan ECG Recorder. Kemudian hasil perekaman sinyal alat yang telah dibuat dibandingkan dengan ECG *Simulator* (Bio Tek, Model: ECGPlus, SN: 91219).

B. Blok Diagram

Saat alat dinyalakan MCU atau mikrokontroler akan menunggu perintah dari kontrol untuk pemilihan jenis gelombang dan kontrol lainnya selain itu display akan menampilkan jenis sinyal yang langsung dikeluarkan saat pertama alat dinyalakan. Sedangkan saklar rotary (BPM kontrol) akan langsung bekerja sesuai putarannya dan menyebabkan dikeluarkan bentuk sinyal pertama oleh MCU meskipun tanpa menyentuh kontrol untuk dikeluarkan menuju rangkaian Digital to Analog Converter atau DAC dalam bentuk Digital. Pemilihan bentuk sinyal dengan memutar saklar rotary akan memiliki range BPM berbeda dan kemudian juga akan ditampilkan pada display LCD 2x16. Sedangkan blok LA, RA, LL, RL, V1, V2, V3, V4, V5, dan V6 akan menerima bentuk sinyal yang dikeluarkan oleh DAC dalam bentuk Analog yang sebelumnya telah diolah dalam blok Resistor Network Circuit. Blok Resistor Network Circuit ini berfungsi untuk memberikan perbedaan impedansi pada setiap Lead.



Gambar 1. Blok Diagram ECG *Simulator*



Gambar 2. Diagram Alir ECG *Simulator*

C. Diagram Alir

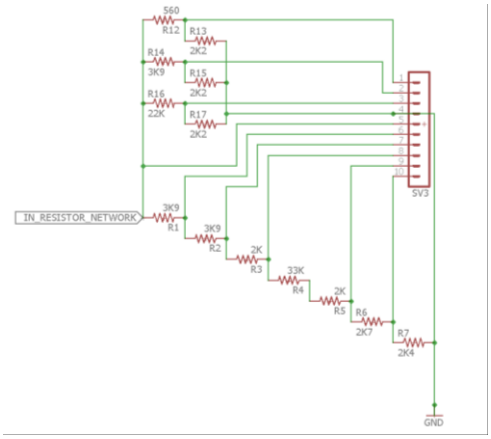
Program Arduino dibangun berdasarkan diagram alir yang ditunjukkan pada Gambar 2. Setelah dimulai, program akan membaca pemilihan BPM dan sensitivitas yang dilakukan. Hasil pembacaan akan mengatur komposisi data array yang gunakan. Kemudian data array akan dikeluarkan oleh mikrokontroler dan diterima oleh DAC. Selanjutnya hasil pengolahan pada DAC akan dikeluarkan ke rangkaian resistor network yang kemudian akan dibagi menjadi 10 outputan.

D. Rangkaian Analog

Bagian penting dari pengembangan modul ini adalah rangkaian analog yang menggambarkan pada Gambar. 3 rangkaian digital to analog converter) dan Gambar. 4 (rangkaian resistor network). Kedua rangkaian tersebut digunakan dalam pemrosesan sinyal yang di keluarkan oleh mikrokontroler.

1) Rangkaian Digital to Analog Converter (DAC)

Rangkaian DAC merupakan rangkaian yang digunakan untuk mikrokontroler dapat berkomunikasi dengan DAC MCP4921 yaitu melalui *PIN* CS, SCK dan SDI. Mikrokontroler akan memberikan bentuk sinyal *digital* melalui program dan DAC MCP4921 akan menterjemahkan menjadi data *analog* dengan resolusi 12-bit. Untuk rangkaianya dapat dilihat Gambar



Gambar 3. Digital to Analog Converter

Nilai tegangan output dari DAC dapat dihitung dengan rumus berikut:

$$Vout = Vref\ \frac{D}{12bit} \quad (1)$$

Selain digunakan untuk mengatur tegangan *output* yang berupa tegangan *analog*, DAC ini digunakan untuk mengatur Frekuensi atau Periode dari sinyal ECG. Perhitungannya sebagai berikut:

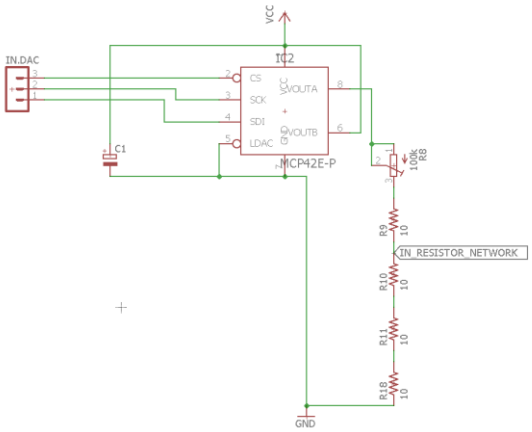
$$Frekuensi\ (f) = \frac{Heart\ rate}{60\ \text{detik}} \quad (2)$$

Sedangkan periode didapatkan setelah kita mengetahui nilai frekuensi dari setiap BPM yang akan diukur atau sebaliknya sebagai berikut :

$$Periode(T) = \frac{1}{\text{f}} \quad (3)$$

2) Rangkaian Resistor Network

Rangkaian ini digunakan untuk membagi nilai sinyal ECG sesuai impedansi tubuh. Berikut merupakan rincian dari pin kontektornya: pin1 = LA, pin2 = LL, pin3 = RA, pin4 = RL, pin5 = V4, pin 6 = V3, pin 7 = V5, pin8 = V6, pin9 = V1, dan pin10=V2. Selanjutnya nilai sinyal di setiap pinya dapat dihitung dengan rumus ebagai berikut jika diumpamakan IN_RESISTOR_NETWORK = INPUT.



Gambar 4. Rangkaian Resistor Network

Rangkaian diatas merupakan rangkaian pembagian tegangan, dimana berikut merupakan hasil perhitungan nilai masing-masing outputnya:

$$V(pin1) = \frac{R13}{R12 + R13} \times \text{INPUT}$$
$$= \frac{2K2}{560 + 2K2} \times \text{INPUT}$$
$$= \frac{2K2}{2760} \times \text{INPUT} = 0{,}7971014492 \times \text{INPUT}$$

$$V\ (pin2) = \frac{R15}{R14 + R15} \times \text{INPUT}$$
$$= \frac{2K2}{3K9 + 2K2} \times \text{INPUT}$$
$$= \frac{2K2}{6K1} \times \text{INPUT} = 0{,}3606557377\ \times \text{INPUT}$$

$$V(pin3) = \frac{R17}{R16 + R17} \times \text{INPUT}$$
$$= \frac{2K2}{22K + 2K2} \times \text{INPUT}$$
$$= \frac{2K2}{24K2} \times \text{INPUT} = 0{,}0909090909 \times \text{INPUT}$$

$$V(pin4) = \text{GND}$$

$$V(pin5) = \text{INPUT}$$

$$V(pin6) = \frac{R2 + R3 + R4 + R5 + R6 + R7}{R1 + R2 + R3 + R4 + R5 + R6 + R7} \times \text{INPUT}$$

$$= \frac{3K9 + 2K + 33K + 2K + 2K7 + 2K4}{3K9 + 3K9 + 2K + 33K + 2K + 2K7 + 2K4} \times \text{INPUT}$$
$$= \frac{46K}{49K9} \times \text{INPUT} = 0{,}9218436873747495 \times \text{INPUT}$$

$$V(pin7) = \frac{R3 + R4 + R5 + R6 + R7}{R1 + R2 + R3 + R4 + R5 + R6 + R7} \times \text{INPUT}$$
$$= \frac{2K + 33K + 2K + 2K7 + 2K4}{3K9 + 3K9 + 2K + 33K + 2K + 2K7 + 2K4} \times \text{INPUT}$$
$$= \frac{42K1}{49K9} \times \text{INPUT} = 0{,}843687374749499 \times \text{INPUT}$$

$$V(pin8) = \frac{R4 + R5 + R6 + R7}{R1 + R2 + R3 + R4 + R5 + R6 + R7}$$
$$= \frac{33K + 2K + 2K7 + 2K4}{3K9 + 3K9 + 2K + 33K + 2K + 2K7 + 2K4} \times \text{INPUT}$$
$$= \frac{40K1}{49K9} \times \text{INPUT} = 0{,}8036072144288577 \times \text{INPUT}$$

$$V(pin9) = \frac{R6 + R7}{R1 + R2 + R3 + R4 + R5 + R6 + R7} \times \text{INPUT}$$
$$= \frac{2K7 + 2K4}{3K9 + 3K9 + 2K + 33K + 2K + 2K7 + 2K4} \times \text{INPUT}$$
$$= \frac{5K1}{49K9} \times \text{INPUT} = 0{,}1022044088176353 \times \text{INPUT}$$

$$V(pin10) = \frac{R7}{R1 + R2 + R3 + R4 + R5 + R6 + R7} \times \text{INPUT}$$
$$= \frac{2K4}{3K9 + 3K9 + 2K + 33K + 2K + 2K7 + 2K4} \times \text{INPUT}$$
$$= \frac{2K4}{49K9} \times \text{INPUT} = 0{,}0480961923847695 \times INPUT$$

III. HASIL

Dalam penelitian ini, ECG *Simulator* telah diuji menggunakan phantom ECG (BioTek, Model : ECGPlus, SN : 91219) dengan cara membandingkan hasil perekeman sinyal ECG dari ECG Recorder (*Digital Electrocardiograph*, Model : ECG-9012A, SN : ECG-9012A120435). Hasil perekaman menunjukan bahwa otputan ECG *Simulator* dan phantom ECG tidak terlalu berbeda atau masih dalam batas toleransi.

Gambar 5. The Holter ECG design

1) Desain ECG *Simulator*

Dari gambar diatas dapat dilihat komponen-komponen yang digunakan pada alat. Rangkaian yang pertama adalah mikrokontroller dengan menggunakan IC ATmega 2560, yang kedua rangkaian DAC (Digital To Analog Converter) menggunakan MCP4921 yang berfungsi untuk mengubah data digital dari mikrokontroller menjadi data analog.

2) Listing Program ECG *Simulator*

Dalam tulisan ini, pemograman diatur menggunakan arduino untuk mengatur sinyal output alat. Program terdiri dari Set Up SPI Interface (untuk mengaktifkan mode serial komunikasi SPI Interface), timer2 interrupt, pengaturan sentitivitas, pengaturan BPM, data sampel sinyal ECG, pengaturan jumlah sampel data, program perhitungan periode sampel data dengan periode diam program fungsi interrupt, program pengiriman DTOA, program tampilan LCD.

Listing program 1. Program Set Up SPI Interface

```
#include "SPI.h" // supports the SPI interface to the D/A converter and 7-segment display
#include <Wire.h> // need the Wire library
pinMode(53, OUTPUT); // D/A converter chip select (low to select chip)
pinMode(51, OUTPUT); // SDI data
pinMode(52, OUTPUT); // SCK clock
// initial state of SPI interface
SPI.begin(); // wake up the SPI bus.
SPI.setDataMode(0); // mode: CPHA=0, data captured on clock's rising edge (low→high)
SPI.setClockDivider(SPI_CLOCK_DIV64); // system clock / 64
SPI.setBitOrder(MSBFIRST); // bit 7 clocks out first
```

Listing Program 2. Program Timer2 Interrupt

```
// First disable the timer overflow interrupt while we're configuring
TIMSK2 &= ~(1<<TOIE2);
// Configure timer2 in normal mode (pure counting, no PWM etc.)
TCCR2A &= ~((1<<WGM21) | (1<<WGM20));
TCCR2B &= ~(1<<WGM22);
// Select clock source: internal I/O clock
ASSR &= ~(1<<AS2);
// Disable Compare Match A interrupt enable (only want overflow)
TIMSK2 &= ~(1<<OCIE2A);
// Now configure the prescaler to CPU clock divided by 128
TCCR2B |= (1<<CS22) | (1<<CS20); // Set bits
TCCR2B &= ~(1<<CS21); // Clear bit
// We need to calculate a proper value to load the timer counter.
// The following loads the value 131 into the Timer 2 counter register
```

```
// The math behind this is:
// (CPU frequency) / (prescaler value) = 125000 Hz = 8us.
// (desired period) / 8us = 125.
// MAX(uint8) + 1 - 125 = 131;
//
// Save value globally for later reload in ISR /
tcnt2 = 131;
// Finally load end enable the timer
TCNT2 = tcnt2;
TIMSK2 |= (1<<TOIE2);
```

Listing Program 3. Program Pengaturan Sensitivitas

```
void button()
{
 if (SW1==LOW)
 {  S=1; digitalWrite(SW2,HIGH); digitalWrite(SW3,HIGH);
                    }
 if (S==1){change = 1;}
 if (SW2==LOW)
 {S=2; digitalWrite(SW1,HIGH); digitalWrite(SW3,HIGH);}
 if (S==2){change = 2;}
 if (SW3==LOW)
 {S=3; digitalWrite(SW1,HIGH); digitalWrite(SW2,HIGH);}
 if (S==3){change = 3;}
SW1= digitalRead(S1);
SW2= digitalRead(S2);
SW3= digitalRead(S3);
}
```

Listing Program 4. Program Pengaturan BPM

```
Value = analogRead(0);
  if (Bpmmap<=400)
 { Bpm = map(Value, 0, 1023, 301, 2400);
  Bpmmap = map(Bpm, 301, 2400, 300, 2400);
  BeatsPerMinute = (float)Bpm/10.0;
  pembagi= 0; gelombang=0;}
  if (Bpmmap>400 && Bpmmap<=1190)
 { Bpm = map(Value, 0, 1023, 305, 2400);
  Bpmmap = map(Bpm, 305, 2400, 300, 2400);
  BeatsPerMinute = (float)Bpm/10.0;
  pembagi= 0;  gelombang=1;}
  if (Bpmmap>1190 && Bpmmap<=1500)
 {Bpm = map(Value, 0, 1023, 305, 2400);
  Bpmmap = map(Bpm, 305, 2400, 300, 2400);
  BeatsPerMinute = (float)Bpm/10.0;
  pembagi= map(Bpmmap,1190,2400,0,203);
 gelombang=1;}
 if (Bpmmap>1500 && Bpmmap<=2300)
 { Bpm = map(Value, 0, 1023, 305, 2400);
  Bpmmap = map(Bpm, 305, 2400, 300, 2400);
  BeatsPerMinute = (float)Bpm/10.0;
  pembagi= map(Bpmmap,1190,2400,0,203);
  gelombang=2;}
```

```
 if (Bpmmap>2300)
 {Bpm = map(Value, 0, 1023, 305, 2400);
  Bpmmap = map(Bpm, 305, 2400, 300, 2400);
  BeatsPerMinute = (float)Bpm/10.0;
  pembagi= map(Bpmmap,1190,2400,0,203);
  gelombang=3;}
```

Listing Program 5. Program Data Sampel Sinyal ECG

```
if (change == 1 && gelombang==0)
 {
  sense=0.23;
const short  y_data[] = {
448*sense,  464*sense,  480*sense,  496*sense,  512*sense,
528*sense, 544*sense, 560*sense, 576*sense, 592*sense,
608*sense,  623*sense,  634*sense,  646*sense,  658*sense,
669*sense, 681*sense, 692*sense, 704*sense, 715*sense,
727*sense,  739*sense,  750*sense,  762*sense,  773*sense,
785*sense, 796*sense, 808*sense, 820*sense, 831*sense,
843*sense,  854*sense,  866*sense,  875*sense,  883*sense,
891*sense, 898*sense, 906*sense, 914*sense, 922*sense,
929*sense,  937*sense,  945*sense,  953*sense,  960*sense,
968*sense, 973*sense, 971*sense, 970*sense, 968*sense,
966*sense,  965*sense,  963*sense,  962*sense,  960*sense,
959*sense, 957*sense, 955*sense, 954*sense, 952*sense,
951*sense,  949*sense,  948*sense,  946*sense,  944*sense,
940*sense, 931*sense, 922*sense, 913*sense, 904*sense,
895*sense,  886*sense,  877*sense,  868*sense,  859*sense,
850*sense, 841*sense, 832*sense, 823*sense, 814*sense,
805*sense,  796*sense,  787*sense,  778*sense,  768*sense,
755*sense, 742*sense, 729*sense, 717*sense, 704*sense,
691*sense,  678*sense,  666*sense,  653*sense,  639*sense,
624*sense, 609*sense, 594*sense, 579*sense, 564*sense,
549*sense,  536*sense,  528*sense,  521*sense,  514*sense,
506*sense, 499*sense, 492*sense, 484*sense, 477*sense,
471*sense,  468*sense,  466*sense,  463*sense,  461*sense,
458*sense, 456*sense, 453*sense, 451*sense, 449*sense,
446*sense,  444*sense,  441*sense,  439*sense,  436*sense,
434*sense, 431*sense, 429*sense, 426*sense, 422*sense,
419*sense,  415*sense,  411*sense,  407*sense,  403*sense,
400*sense, 396*sense, 392*sense, 388*sense, 384*sense,
381*sense,  367*sense,  353*sense,  338*sense,  324*sense,
310*sense, 296*sense, 278*sense, 258*sense, 237*sense,
217*sense,  196*sense,  179*sense,  162*sense,  156*sense,
158*sense, 159*sense, 161*sense, 189*sense, 229*sense,
274*sense,  312*sense,  348*sense,  493*sense,  686*sense,
880*sense,  1073*sense,  1267*sense,  1460*sense,
1654*sense,
1847*sense,  2041*sense,  2232*sense,  2415*sense,
2597*sense,  2728*sense,  2803*sense,  2878*sense,
2953*sense, 3024*sense,
3095*sense,  3166*sense,  3236*sense,  3275*sense,
3270*sense,  3266*sense,  3260*sense,  3252*sense,
3230*sense, 3186*sense,
```

DOI: 10.35882/teknokes.v12i1.2 ISSN:2407-8964

```
3108*sense, 2996*sense, 2745*sense, 2589*sense, 2433*sense, 2277*sense, 2121*sense, 1965*sense, 1809*sense, 1653*sense,
1497*sense, 1341*sense, 1185*sense, 1030*sense, 874*sense, 718*sense, 562*sense, 417*sense, 278*sense, 182*sense,
130*sense, 77*sense, 25*sense, 3*sense, 9*sense, 15*sense, 21*sense, 27*sense, 33*sense, 39*sense,
45*sense, 51*sense, 91*sense, 135*sense, 193*sense, 230*sense, 254*sense, 278*sense, 301*sense, 311*sense,
322*sense, 332*sense, 342*sense, 352*sense, 359*sense, 367*sense, 374*sense, 381*sense, 388*sense, 394*sense,
397*sense, 399*sense, 401*sense, 404*sense, 406*sense, 409*sense, 411*sense, 413*sense, 415*sense, 415*sense,
415*sense, 415*sense, 416*sense, 416*sense, 416*sense, 416*sense, 416*sense, 417*sense, 417*sense, 417*sense,
417*sense, 418*sense, 418*sense, 418*sense, 418*sense, 419*sense, 419*sense, 419*sense, 419*sense, 419*sense,
419*sense, 419*sense, 419*sense, 419*sense, 419*sense, 420*sense, 420*sense, 420*sense, 420*sense, 420*sense,
420*sense, 420*sense, 420*sense, 420*sense, 420*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense,
421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense,
421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense,
421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense,
421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense,
421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense,
421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense, 421*sense,
423*sense, 425*sense, 427*sense, 429*sense, 432*sense, 434*sense, 436*sense, 438*sense, 440*sense, 442*sense,
444*sense, 446*sense, 451*sense, 457*sense, 462*sense, 467*sense, 472*sense, 477*sense, 482*sense, 488*sense,
493*sense, 498*sense, 503*sense, 508*sense, 514*sense, 519*sense, 524*sense, 529*sense, 534*sense, 539*sense,
545*sense, 550*sense, 555*sense, 560*sense, 565*sense, 570*sense, 576*sense, 581*sense, 586*sense, 595*sense,
605*sense, 614*sense, 623*sense, 633*sense, 642*sense, 651*sense, 660*sense, 670*sense, 679*sense, 688*sense,
698*sense, 707*sense, 716*sense, 725*sense, 735*sense, 744*sense, 753*sense, 762*sense, 772*sense, 781*sense,
790*sense, 800*sense, 809*sense, 818*sense, 827*sense, 837*sense, 846*sense, 855*sense, 865*sense, 874*sense,
883*sense, 892*sense, 902*sense, 911*sense, 920*sense, 930*sense, 939*sense, 948*sense, 957*sense, 967*sense,
976*sense, 984*sense, 993*sense, 1001*sense, 1009*sense, 1017*sense, 1025*sense, 1033*sense, 1041*sense, 1050*sense,
1058*sense, 1066*sense, 1074*sense, 1082*sense, 1090*sense, 1098*sense, 1107*sense, 1115*sense, 1123*sense, 1131*sense,
1139*sense, 1147*sense, 1155*sense, 1164*sense, 1172*sense, 1180*sense, 1188*sense, 1194*sense, 1201*sense, 1207*sense,
1213*sense, 1220*sense, 1226*sense, 1232*sense, 1238*sense, 1245*sense, 1251*sense, 1257*sense, 1264*sense, 1270*sense,
1276*sense, 1283*sense, 1289*sense, 1295*sense, 1302*sense, 1306*sense, 1309*sense, 1312*sense, 1314*sense, 1317*sense,
1320*sense, 1322*sense, 1325*sense, 1328*sense, 1330*sense, 1333*sense, 1336*sense, 1338*sense, 1341*sense, 1344*sense,
1346*sense, 1349*sense, 1352*sense, 1354*sense, 1357*sense, 1360*sense, 1362*sense, 1365*sense, 1366*sense, 1360*sense,
1354*sense, 1349*sense, 1343*sense, 1338*sense, 1332*sense, 1327*sense, 1321*sense, 1316*sense, 1310*sense, 1305*sense,
1299*sense, 1294*sense, 1288*sense, 1282*sense, 1277*sense, 1271*sense, 1266*sense, 1260*sense, 1255*sense, 1249*sense,
1244*sense, 1238*sense, 1233*sense, 1223*sense, 1209*sense, 1196*sense, 1182*sense, 1169*sense, 1155*sense, 1142*sense,
1129*sense, 1115*sense, 1102*sense, 1088*sense, 1075*sense, 1062*sense, 1048*sense, 1035*sense, 1021*sense, 1008*sense,
995*sense, 981*sense, 968*sense, 954*sense, 941*sense, 929*sense, 916*sense, 904*sense, 891*sense, 879*sense,
866*sense, 854*sense, 841*sense, 829*sense, 816*sense, 803*sense, 791*sense, 778*sense, 766*sense, 753*sense,
741*sense, 728*sense, 716*sense, 703*sense, 690*sense, 678*sense, 665*sense, 653*sense, 640*sense, 628*sense,
615*sense, 603*sense, 590*sense, 578*sense, 565*sense, 552*sense, 540*sense, 530*sense, 522*sense, 514*sense,
506*sense, 498*sense, 490*sense, 482*sense, 474*sense, 467*sense, 459*sense, 451*sense, 443*sense, 435*sense,
427*sense, 419*sense, 411*sense, 404*sense, 396*sense, 388*sense, 380*sense, 372*sense, 364*sense};
```

Listing Program 6. Program Pengaturan Jumlah Sampel Data

```
NumSamples = sizeof(y_data) / (2.00+(pembagi/100));
```

Listing Program 7. Program Perhitungan Periode Sampel Data dengan Periode Diam

```
BeatsPerMinute = (float)Bpm / 10.0;
noInterrupts();
```

```
IdlePeriod = (unsigned int)((float)60000.0 / BeatsPerMinute) – (float)NumSamples;
interrupts();
```

Listing Program 8. Program Fungsi Interrupt

```
  switch (State)
  {
   case INIT:
          // zero the QRS and IDLE counters
          QRSCount = 0;
          IdleCount = 0;
          // set next state to QRS
          State = QRS;
   break;
   case QRS:
         // output the next sample in the QRS waveform to the D/A converter
          DTOA_Send(y_data[QRSCount]);
          // advance sample counter and check for end
          QRSCount++;
          if (QRSCount >= NumSamples)
          {
           // start IDLE period and output first sample to DTOA
           QRSCount = 0;
           DTOA_Send(y_data[0]);
           State = IDLE;
          }
  break;

  case IDLE:
          // since D/A converter will hold the previous value written, all we have
          // to do is determine how long the IDLE period should be.
          // advance idle counter and check for end
          IdleCount++;
          // the IdlePeriod is calculated in the main loop (from a pot)
          if (IdleCount >= IdlePeriod)
          {
           IdleCount = 0;
           State = QRS;
          }
  break;

  default:
  break;
 }
```

Listing Program 9. Program Pengiriman DTOA

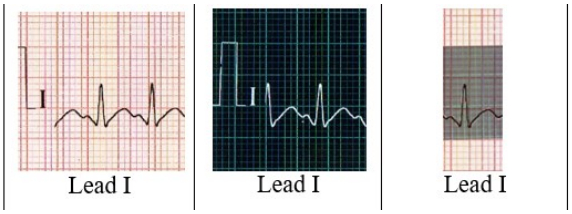
```
  void  DTOA_Send(unsigned short DtoAValue)
{
  byte Data = 0;
  digitalWrite(53, 0);
  // send the high byte first 0011xxxx
  Data = highByte(DtoAValue);
  Data = 0b00001111 & Data;
  Data = 0b00110000 | Data;
  SPI.transfer(Data);
  // send the low byte next xxxxxxxx
  Data = lowByte(DtoAValue);
  SPI.transfer(Data);
  digitalWrite(53, 1);
}
```

Listing Program 10. Program Tampilan LCD

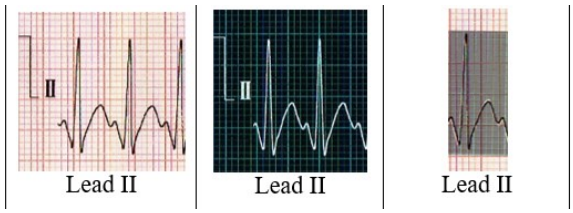
```
void displaylcd()
{
  lcd.setCursor(0, 0);
  lcd.print("PHANTOM ECG");
  lcd.setCursor(0, 1);
  lcd.print("BPM:");
  lcd.setCursor(4, 1);
  lcd.write(byte(0));
  lcd.setCursor(5, 1);
  lcd.print(int(Bpmmap/10));
  button();
  delay((10000/(Bpm/60))/2);
  analogWrite(ledPin,0);
  lcd.setCursor(4, 1);
  lcd.print(" ");
  lcd.print(int(Bpmmap/10));
  button();
  delay((10000/(Bpm/60))/2);
}
```

3) Sinyal ECG Simulator Dibandingkan dengan Phantom ECG

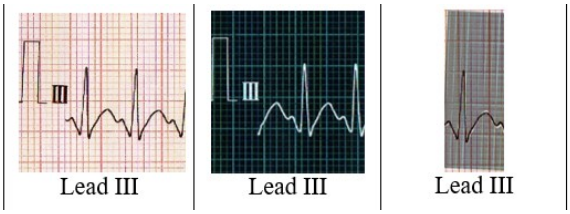
Berikut perbandingan hasil *printout* Modul ECG *Simulator* dan alat pembanding *ECG Simulator* (BioTek, Model : ECGPlus, SN : 91219). Pada kali ini hasil *printout* yang ditampilkan hanya hasil *printout* dari pengaturan BPM 180 dengan sensitivitas 2.0mV. Pada gambar berikut sinyal *ECG Simulator* dengan warna line hitam dan background merah dan Modul ECG *Simulator* warna line putih background hitam.



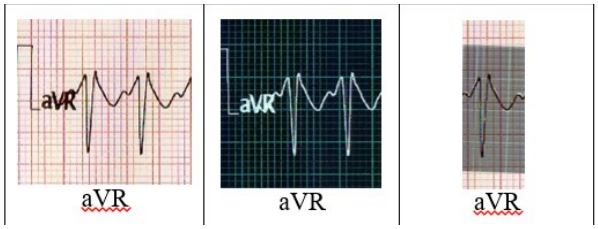
Gambar 6. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead I)



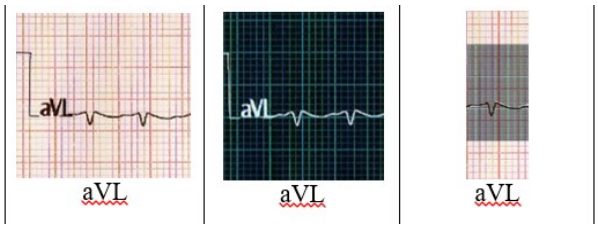
Gambar 7. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead II)



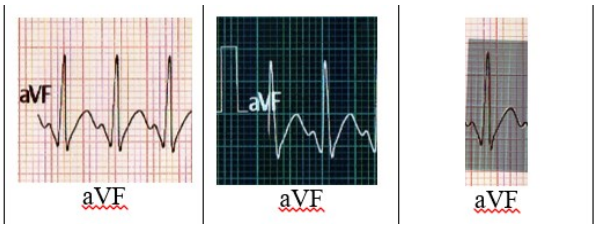
Gambar 8. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead III)



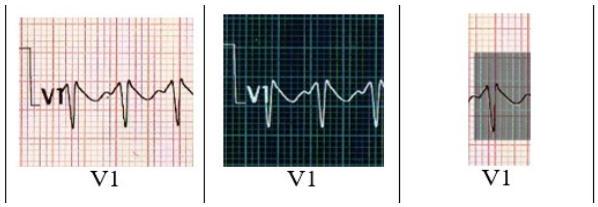
Gambar 9. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead aVR)



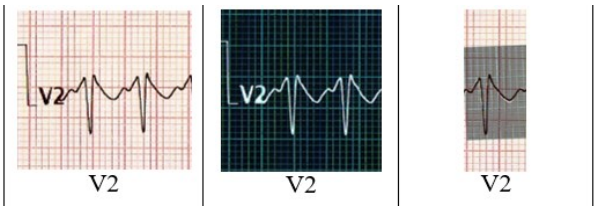
Gambar 10. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead aVL)



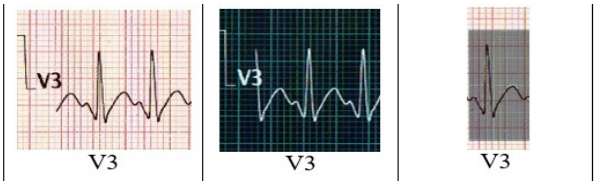
Gambar 11. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead aVF)



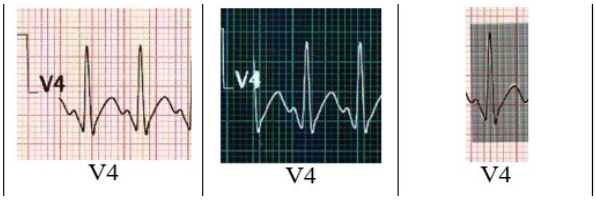
Gambar 12. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (V1)



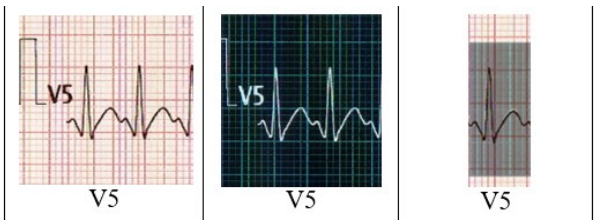
Gambar 13. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead V2)



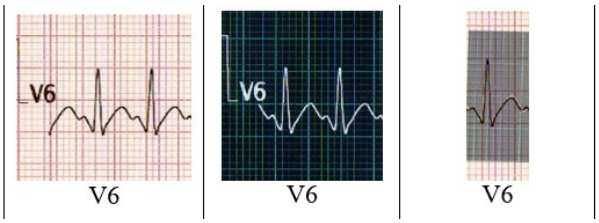
Gambar 14. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead V3)



Gambar 15. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead V4)



Gambar 16. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead V5)



Gambar 17. Perbandingan Hasil Sinyal ECG *Simulator* dengan Phantom ECG (Lead V6)

4) Nilai Eror BPM ECG Simulator

Dari pengukuran hasil *printout* nilai BPM akan dihitung nilai *error*nya pada setiap setingan BPM 30, 60, 120, 180 dan 240 baik pada sensitivitas 0.5 mV, 1.0 mV dan 2.0 mV.

TABEL I. NILAI *ERROR* PADA PENGATURAN SESNITIVITAS 0.5MV.

No	BPM	*Error* (%)
1	30	0.683%
2	60	0.562%
3	120	0.321%
4	180	0.890%
5	240	0.486%

TABEL II. NILAI *ERROR* PADA PENGATURAN SESNITIVITAS 1.0MV.

No	BPM	*Error* (%)
1	30	0.32%
2	60	0.562%
3	120	0.16%
4	180	0.646%
5	240	0.486%

TABEL III. NILAI *ERROR* PADA PENGATURAN SESNITIVITAS 2.0MV.

No	BPM	*Error* (%)
1	30	0.32%
2	60	0.643%
3	120	0.16%
4	180	0.401%
5	240	0.486%

IV. DISKUSI

Desain ECG *Simulator* telah diperiksa dan diuji sepenuhnya dalam penelitian ini. Berdasarkan pengukuran output ECG *Simulator* dan phantom ECG, sinyal EKG yang dihasilkan ketika menggunakan input dari *Simulator* ECG menunjukkan pola yang benar dari sinyal EKG yang terdiri dari gelombang P, Q, R, S, dan T dengan BPM mulai dari 30-240,dan sensitivitas 0.5 Mv, 1 mV, dan 2 mV. Dengan membandingkan hasil perekaman ECG *Simulator* dengan phantom ECG dengan pembacaan perekaman menggunakan EKG Recorder. Kesalahan BPM antara desain dan phantom ECG (dengan pembacaan perekaman dari ECG Recorder didapat nilai tingkat kesalahan sebesar 0.683% sensitivitas 0.5mV, 0.32% sensitivitas 1.0mV, dan 0.32% sensitivitas 2.0mV pada setting BPM 30, didapat nilai tingkat kesalahan sebesar 0.562% sensitivitas 0.5mV, 0.562% sensitivitas 1.0mV, dan 0.643% sensitivitas 2.0mV pada setting BPM 60, didapat nilai tingkat kesalahan sebesar 0.321% sensitivitas 0.5mV, 0.16% sensitivitas 1.0mV, dan 0.16% sensitivitas 2.0mV pada setting BPM 120, didapat nilai tingkat kesalahan sebesar 0.646% sensitivitas 0.5mV, 0.401% sensitivitas 1.0mV, dan 0.890% sensitivitas 2.0mV pada setting BPM 180 dan 0.486% sensitivitas 0.5mV, 0.486% sensitivitas 1.0mV dan 0.486% sensitivitas 2.0mV pada setting BPM 240.

V. KESIMPULAN

Studi ini telah menunjukkan perkembangan ECG *Simulator* dari penelitian sebelumnya dengan menambahkan outputan menjadi 10 sadapan yang apabila dibaca menggunakan ECG Recorder akan terbaca menjadi 12 chanel. Studi ini dibangun berdasarkan mikrokontroler Arduino dan beberapa rangkaian analog. Studi ini telah membuktikan bahwa keakuratannya layak digunakan untuk mengkalibrasi EKG Recorder. Di masa depan, penelitian ini dapat dibuat dan digunakan di klinik kecil di desa-desa dengan biaya rendah.

DAFTAR PUSTAKA

[1] R. Achmad, *Instrumentasi Biomedis*, no 1. July. Yogyakarta: Graha Ilmu, 2014.

[2] C. Caner and M. Engin, "The programmable ECG *Simulator*," *J. Med. Syst.*, vol. 32, no. 4, pp. 355–359, 2008.

[3] G. M. Tani, P. C. Nugraha, and Syaifudin, "Simulasi ECG (Phantom electrocardiograph) Berbasis Mikrokontroler (Gregorius Mario Tani, Priyambada Cahya Nugraha, Syaifudin)," 2017.

[4] N. N. S. Malini, I. D. G. H. Wisana, and M. R. Makruf, "ECG *Simulator*," pp. 8–9, 2017.

[5] M. Z. Alamanda, A. Pudji, and M. R. Makruf, "Phantom ECG," pp. 5–11, 2018.