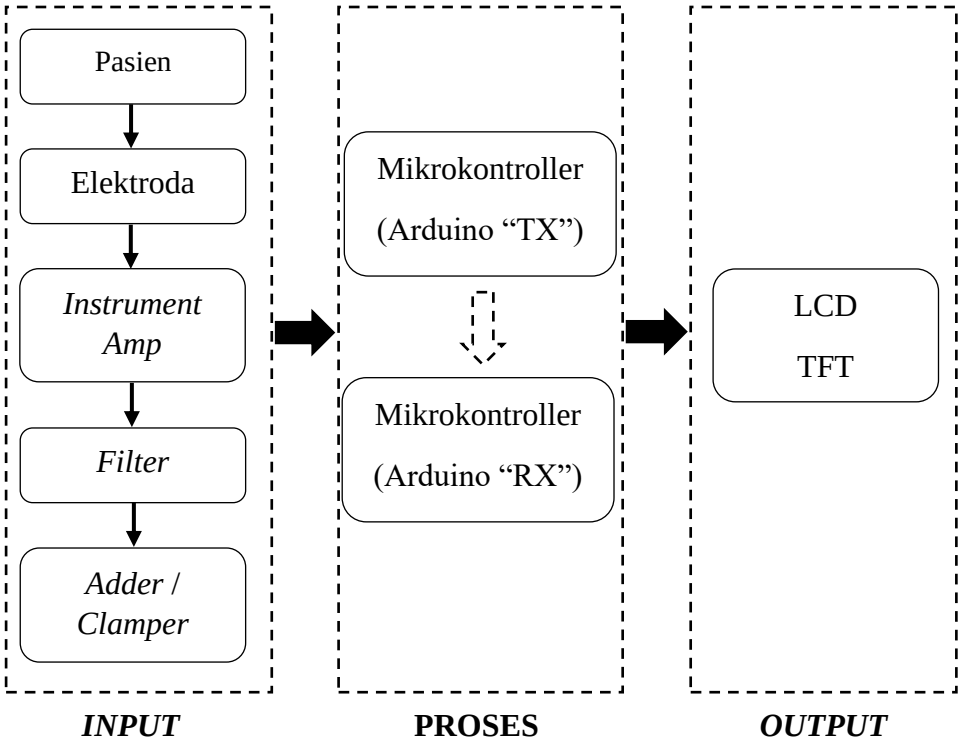
BAB III

METODOLOGI PENELITIAN

3.1 Diagram Blok Sistem

Penelitian dimulai dengan perancangan diagram blok sistem untuk dapat memahami alur sistem rancangan EKG yang akan dibuat.



Gambar 3.1 Diagram Blok Sistem

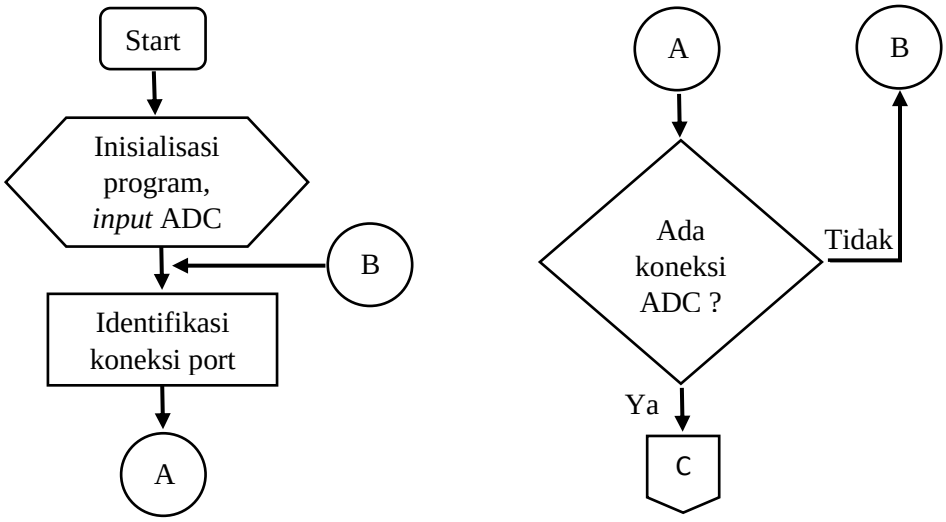
Berdasarkan gambar 3.1 yang menggambarkan sistem dimulai dengan *input* yang terdiri dari pasien sebagai sampel. Sampel diambil dengan menggunakan elektroda yang dipasang pada *lead* anggota badan (tangan kanan, tangan kiri, kaki kanan, kaki kiri). Elektroda tersebut berfungsi untuk menyalurkan sinyal yang ada pada tubuh ke modul EKG atau pengkondisi sinyal yang berisi rangkaian

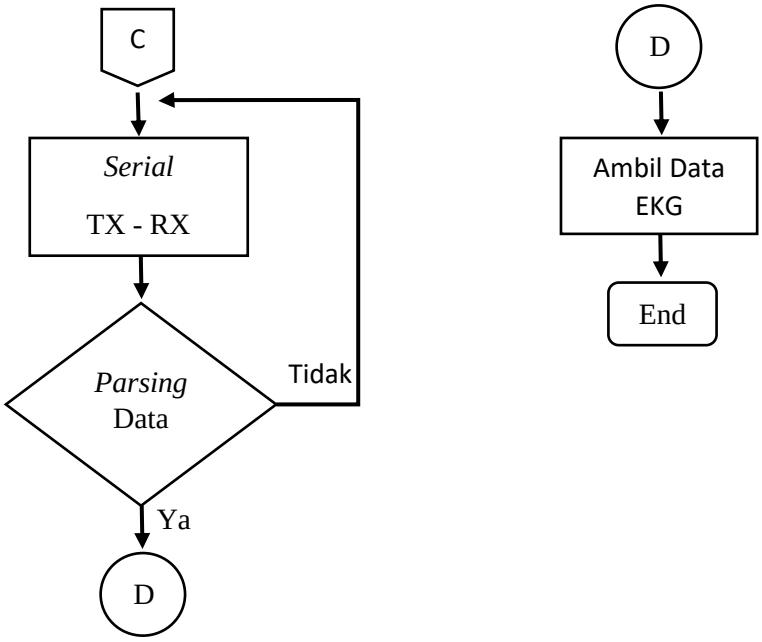
instrument amplifier yang berfungsi sebagai penguat awal untuk menyadap kelistrikan jantung sehingga sinyal jantung yang berorde *milivolt* dapat dideteksi, rangkaian *filter* untuk menghilangkan *noise* atau interferensi frekuensi yang berasal dari tubuh ataupun dari luar tubuh pasien, dan rangkaian *adder*/*clamper* sebagai penggeser tegangan yang dibutuhkan oleh mikrokontroller, yaitu 0V-5V.

Setelah sinyal jantung disalurkan oleh rangkaian pengkondisi sinyal, selanjutnya sinyal tersebut akan diteruskan dan diproses oleh mikrokontroller TX (Arduino Nano), sinyal masukan berupa sinyal jantung yang telah dikondisikan oleh modul EKG dengan amplitudo 0V-5V dan kemudian di konversi menjadi data digital oleh *internal* ADC (*analog to digital conversion*) yang terdapat pada mikrokontroller (TX), selanjutnya data ADC dikirim melalui *pin serial* antar arduino, sehingga data sinyal yang diperoleh dapat diolah menjadi grafik oleh mikrokontroller (RX) yang akan ditampilkan pada LCD TFT sebagai *output*nya.

3.2 Diagram Alir Sistem

Cara kerja diagram alir sistem ditunjukan pada gambar 3.2 berikut.



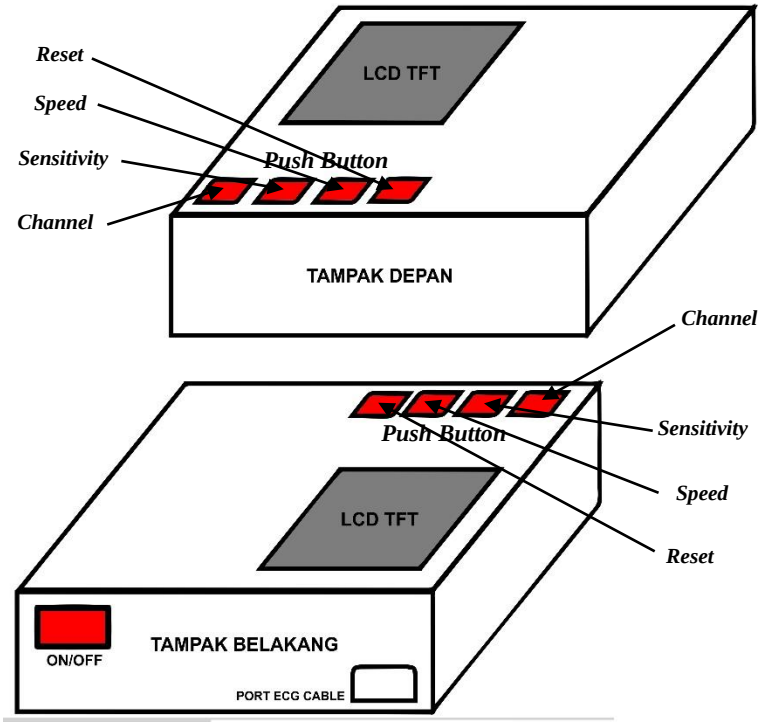


Gambar 3.2 Diagram Alir Sistem

Cara kerja diagram alir pertama kali adalah inisialisasi program dan inisialisasi ADC untuk perintah konversi data analog ke digital. Data analog diambil dari rangkaian sadapan yang sudah diolah oleh modul EKG kemudian dikonversi menjadi data digital oleh mikrokontroller (TX). Selanjutnya port ADC akan mendeteksi apakah ada koneksi kabel *lead* yang terhubung dengan perangkat EKG atau tidak, jika tidak maka alir sistem tidak berlanjut, jika ada koneksi kabel ke port, maka sistem berlanjut ke pengiriman data melalui komunikasi *serial* antar Arduino dari mikrokontroller (TX) ke mikrokontroller (RX), data yang diterima oleh mikrokontroller (RX) akan dipisahkan atau di *parsing* sehingga data yang diterima tidak terbaca acak, jika tidak ada data yang di*parsing* maka mikrokontroller (RX) akan kembali mengidentifikasi data yang akan dipisahkan pada *serial* komunikasi, jika ada data yang masuk maka dilanjutkan dengan pembacaan data sinyal EKG dari hasil sadapan yang telah diolah ke data digital.

3.3 Diagram Mekanik Alat

Diagram mekanik alat *Electrocardiograph* 6 *Lead* Berbasis Arduino akan dirancang sedemikian rupa seperti pada Gambar 3.3 berikut ini.

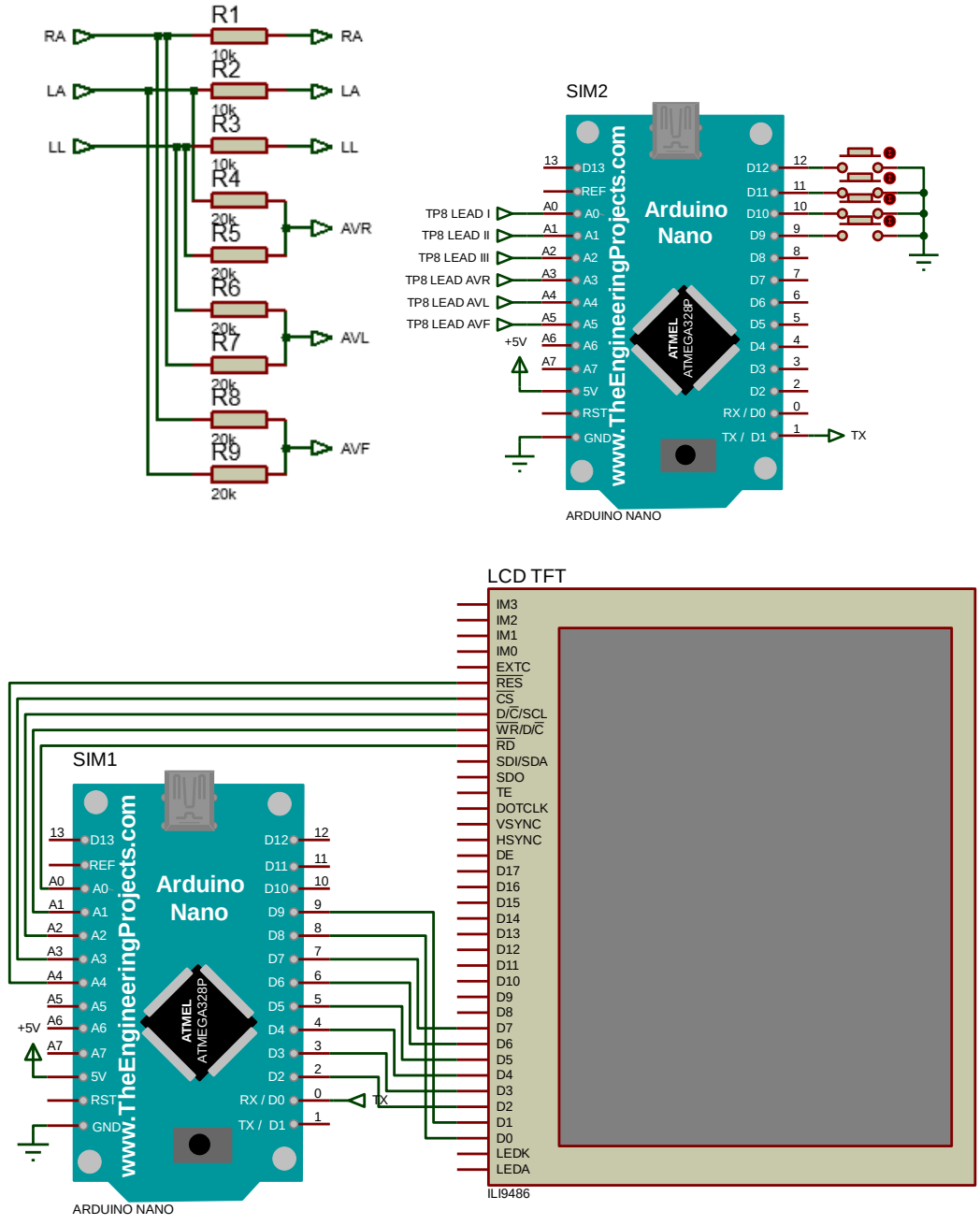


Gambar 3.3 Diagram Mekanik Alat

3.4 Rangkaian Sistem Keseluruhan

3.4.1 Rangkaian *Input Lead Unipolar* dan TFT

Rangkaian pada gambar 3.4 berikut ini merupakan rangkaian *input* sadapan *lead* dan *output* LCD TFT.



Gambar 3.4 *Input lead Unipolar* dan LCD TFT.

Input lead unipolar didapatkan dari sadapan elektroda RA (tangan kanan), LA (tangan kiri) dan RL (kaki kanan), masing-masing elektroda dihubungkan menjadi satu terminal dengan resistansi yang sama. *Output* RA, LA dan LL masing-masing menggunakan resistansi sebesar 10KΩ, sedangkan *output* pada AVR, AVL dan AVF menggunakan resistansi 20KΩ. AVR didapatkan dari elektroda LA dan LL yang diparalel, AVL didapatkan dari RA dan LL yang diparalel sedangkan AVF didapatkan dari RA dan LA yang diparalel, sehingga setiap sadapan mendapatkan resistansi yang sama yaitu sebesar 10KΩ. *Output* pada setiap rangkaian terminal tersebut nantinya akan diproses oleh rangkaian *filter* pada masing-masing *lead* yang dibutuhkan.

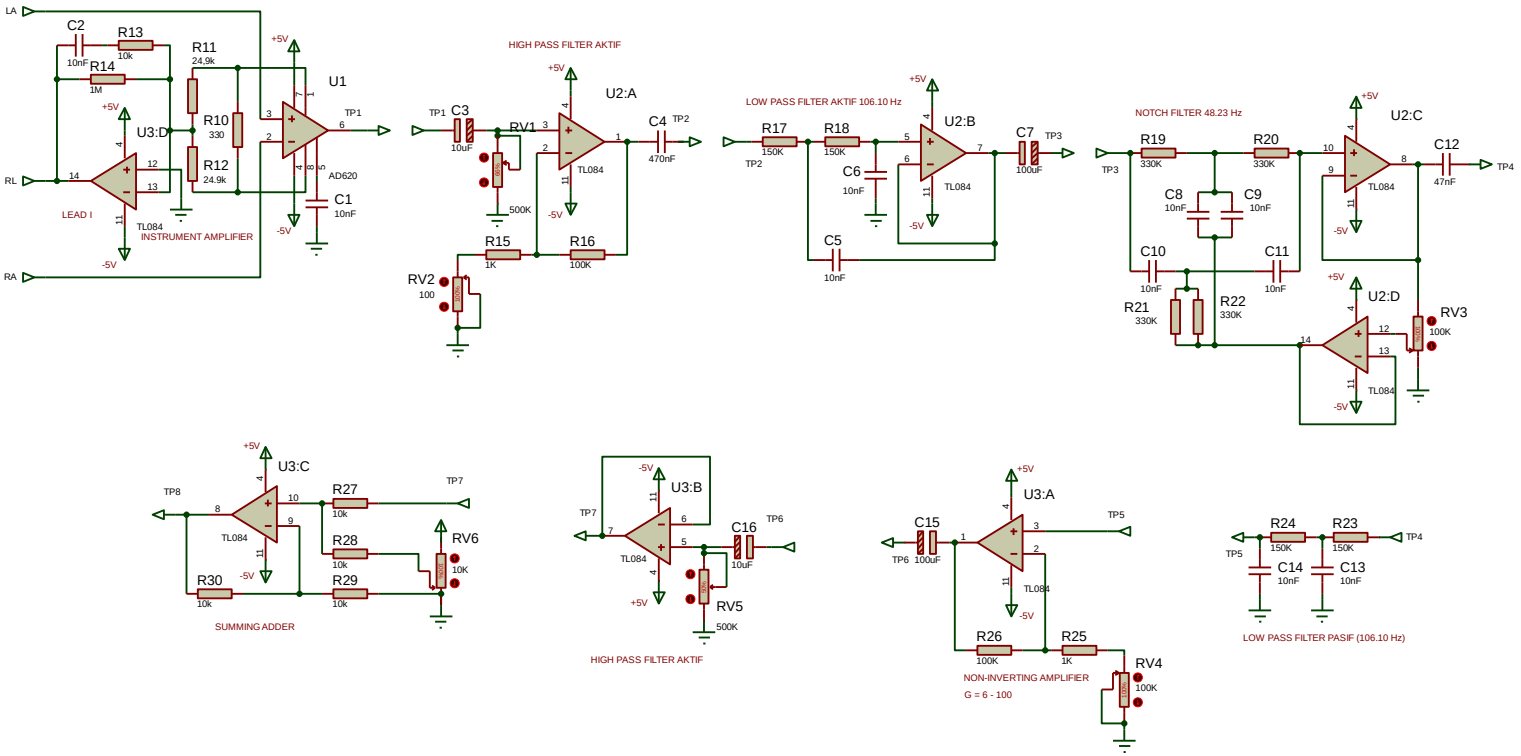
Pengolahan grafis menggunakan LCD TFT diproses oleh arduino nano, dimana hasil dari setiap *filter* yang dihasilkan oleh masing-masing *lead* akan dijadikan data digital dan ditampilkan dalam bentuk grafik pada LCD TFT. LCD TFT membutuhkan 16 *pin* arduino untuk proses pembacaan, yaitu pin A0,A1,A2,A3,A4, 2, 3, 4, 5, 6, 7, 8, 9,5V, 3,3V dan *ground*. *Pin* A0 dihubungkan ke LCD_RD, *pin* A1 dihubungkan ke LCD_WR, *pin* A2 dihubungkan ke LCD_RS, *pin* A3 dihubungkan ke LCD_CS, *pin* A4 dihubungkan ke LCD_RST, pin 2 dihubungkan ke D2, pin 3 dihubungkan ke D3, pin 4 dihubungkan ke D4, pin 5 dihubungkan ke D5, pin 6 dihubungkan ke D6, pin 7 dihubungkan ke d7, pin 8 dihubungkan ke D0, pin 9 dihubungkan ke D1, pin 5V dihubungkan pada 5V, pin 3,3V dihubungkan ke 3,3V dan pin ground dihubungkan ke ground. Karena arduino nano hanya memiliki 8 pin analog, sedangkan 5 pin sudah digunakan oleh LCD TFT, maka untuk proses pembacaan sinyal yang membutuhkan pin analog tidak

mencukupi, sehingga digunakan dua arduino nano, dimana arduino 2 akan digunakan untuk pemrosesan sinyal, dan arduino 1 digunakan untuk proses penampilan sinyal.

Masing-masing arduino akan saling berkomunikasi untuk proses pembacaan dan pengolahan data melalui *serial port* / *serial communication* menggunakan pin 0 (RX) dan pin 1 (TX). Arduino 2 akan berperan sebagai pengirim sinyal sedangkan arduino 1 akan berperan sebagai penerima data serta melakukan proses pengolahan grafik. Selain untuk proses pengolahan ADC dan pengiriman sinyal, arduino 1 juga berperan untuk membaca perintah dan mengolah perintah dari *push button*, sehingga perintah yang diterima dapat diolah dan diteruskan / dikirim dari arduino 2 ke arduino 1.

3.4.2 Rangkaian *Lead* I

Rangkaian pada gambar 3.5 merupakan rangkaian pemrosesan *filter* pada *lead* I.



Gambar 3.5 Rangkaian *Lead* I.

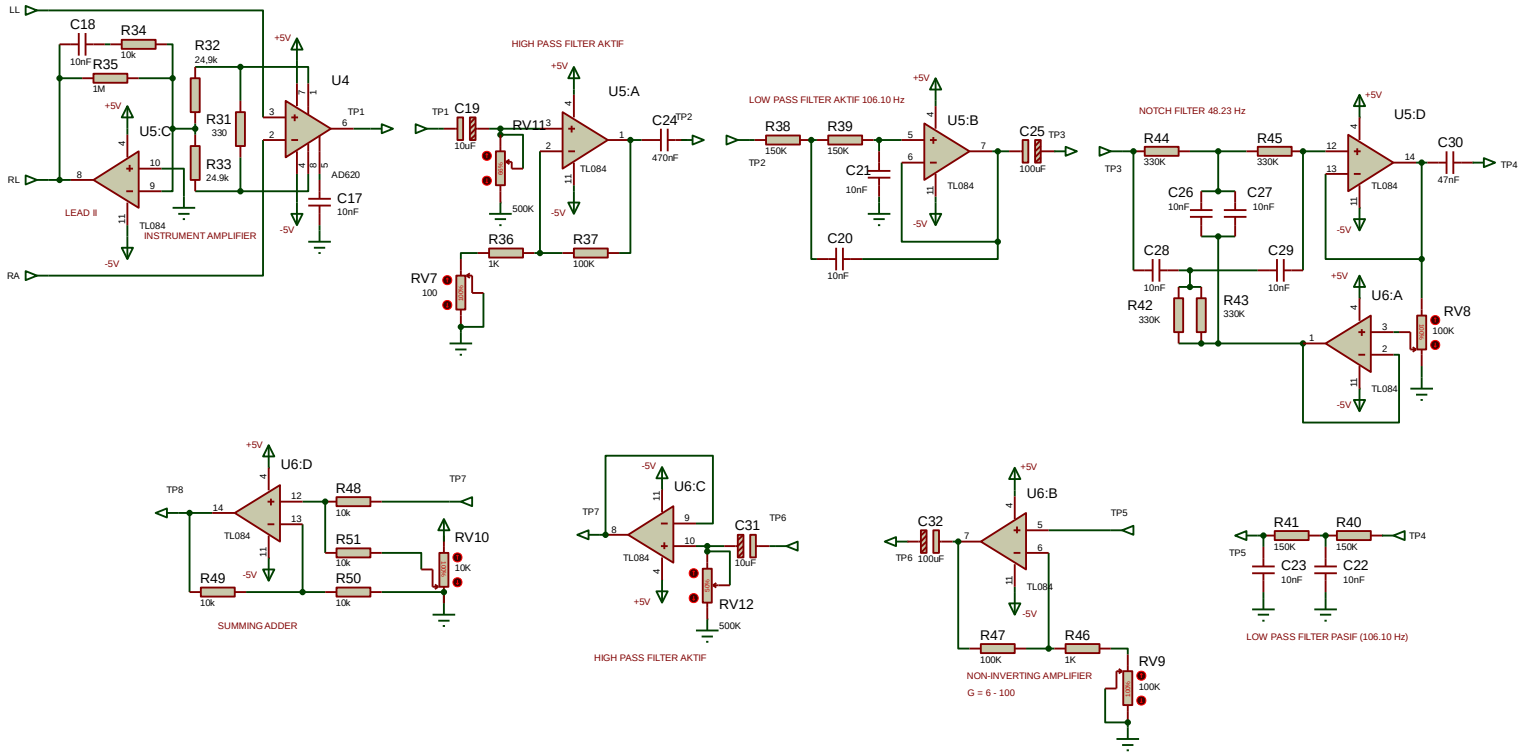
Rangkaian *lead* I merupakan rangkaian yang digunakan untuk pemrosesan *filter* sinyal yang telah disadap atau didapat dari tubuh, sehingga sinyal tersebut dapat dipisahkan antara sinyal otot tubuh dengan sinyal otot jantung. *Input* pada *lead* I diperoleh melalui sadapan bipolar atau langsung dari elektroda RA,LA dan RL. Sinyal yang didapat dari elektroda akan diproses oleh *instrument amplifier* yang menggunakan IC AD620, selanjutnya sinyal tersebut akan diolah lagi untuk mengurangi *noise* atau interferensi dari sinyal tubuh. Setelah melewati *instrument amplifier*, sinyal diteruskan ke rangkaian HPF aktif -20dB 0,03Hz dengan penguatan sebesar 2 – 101 kali, setelah melewati rangkaian HPF sinyal diteruskan ke rangkaian *filter* selanjutnya, yaitu rangkaian LPF aktif -40dB 106,15Hz. Untuk mencegah adanya gangguan sinyal yang disebabkan oleh PLN (kurang lebih 50Hz) maka sinyal di*filter* dengan menggunakan rangkaian *notch filter* dengan frekuensi *cut-off* sebesar 48,3Hz. Setelah melalui *notch filter*, maka sinyal diteruskan ke rangkaian LPF pasif, -20dB, LPF pasif -20dB pada rangkaian berguna agar sinyal dapat lebih dikondisikan dan mendapatkan proses *filter* yang lebih baik. Selanjutnya sinyal diteruskan ke rangkaian penguat *non-inverting* agar sinyal yang masih berorde kecil dapat diatur amplitudonya sehingga mudah terbaca, karena pada rangkaian *filter* cukup banyak menggunakan komponen resistor, sehingga tidak menutup kemungkinan amplitudo sinyal akan dilemahkan. Setelah sinyal dapat terbaca, sinyal akan diteruskan ke rangkaian HPF pasif -20dB yang disusul oleh rangkaian *buffer*, sehingga menjadikannya rangkaian HPF aktif -20dB tanpa penguatan, rangkaian ini berfungsi untuk menanggulangi sinyal *noise* tubuh yang sekiranya masih lolos, sekaligus berfungsi sebagai *blocking* DC dan penyangga

agar sinyal yang telah diolah pada *filter-filter* sebelumnya tidak mengalami penurunan tegangan.

Rangkaian pengkondisi sinyal yang telah dijelaskan diatas berfungsi untuk menyadap sinyal jantung yang berkisar antara 0,03Hz sampai dengan kurang lebih 100Hz, dimana untuk melakukan *monitoring* jantung frekuensi yang dibutuhkan berkisar antara 0,03Hz sampai dengan 300Hz. Rangkaian *filter* hanya berfungsi untuk melakukan pengolahan penyaringan frekuensi jantung, sedangkan amplitudo sinyal yang dibutuhkan untuk pemrosesan pada mikrokontroller berkisar antara 0V sampai 5V, sehingga sinyal dibawah 0V atau diatas 5V tidak akan terbaca oleh mikrokontroller. Agar sinyal dapat terbaca oleh mikrokontroller maka diperlukan rangkaian tambahan untuk menggeser fasa, yaitu berupa rangkaian *summing adder*. Maka setelah semua rangkaian pengkondisi *filter* ditambahkan rangkaian *summing adder*, agar amplitudo sinyal dapat digeser antara 0V sampai dengan 5V.

3.4.3 Rangkaian *Lead* II

Rangkaian pada gambar 3.6 merupakan rangkaian pemrosesan *filter* pada *lead* II.



Gambar 3.6 Rangkaian *Lead* II

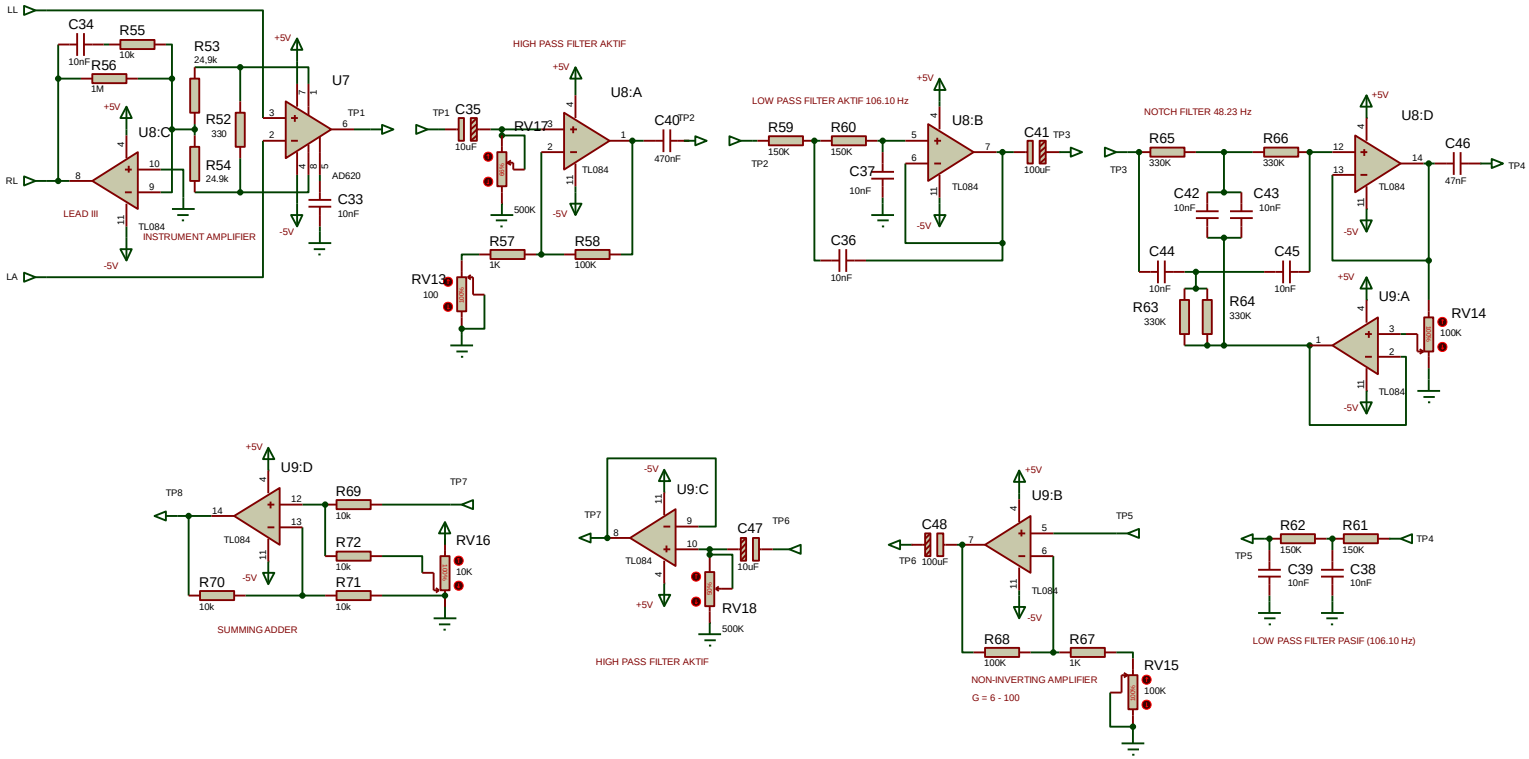
Rangkaian *Lead* II merupakan rangkaian yang digunakan untuk pemrosesan *filter* sinyal yang telah disadap atau didapat dari tubuh, sehingga sinyal tersebut dapat dipisahkan antara sinyal otot tubuh dengan sinyal otot jantung. *Input* pada *lead* II diperoleh melalui sadapan bipolar atau langsung dari elektroda RA,LL dan RL. Sinyal yang didapat dari elektroda akan diproses oleh *instrument amplifier* yang menggunakan IC AD620, selanjutnya sinyal tersebut akan diolah lagi untuk mengurangi *noise* atau interferensi dari sinyal tubuh. Setelah melewati *instrument amplifier*, sinyal diteruskan ke rangkaian HPF aktif -20dB 0,03Hz dengan penguatan sebesar 2 – 101 kali, setelah melewati rangkaian HPF sinyal diteruskan ke rangkaian *filter* selanjutnya, yaitu rangkaian LPF aktif -40dB 106,15Hz. Untuk mencegah adanya gangguan sinyal yang disebabkan oleh PLN (kurang lebih 50Hz) maka sinyal di*filter* dengan menggunakan rangkaian *notch filter* dengan frekuensi *cut-off* sebesar 48,3Hz. Setelah melaui *notch filter*, maka sinyal diteruskan ke rangkaian LPF pasif, -20dB, LPF pasif -20dB pada rangkaian berguna agar sinyal dapat lebih dikondisikan dan mendapatkan proses *filter* yang lebih baik. Selanjutnya sinyal diteruskan ke rangkaian penguat *non-inverting* agar sinyal yang masih berorde kecil dapat diatur amplitudonya sehingga mudah terbaca, karena pada rangkaian *filter* cukup banyak menggunakan komponen resistor, sehingga tidak menutup kemungkinan amplitudo sinyal akan dilemahkan. Setelah sinyal dapat terbaca, sinyal akan diteruskan ke rangkaian HPF pasif -20dB yang disusul oleh rangkaian *buffer*, sehingga menjadikannya rangkaian HPF aktif -20dB tanpa penguatan, rangkaian ini berfungsi untuk menanggulangi sinyal *noise* tubuh yang sekiranya masih lolos, sekaligus berfungsi sebagai *blocking* DC dan penyangga

agar sinyal yang telah diolah pada *filter-filter* sebelumnya tidak mengalami penurunan tegangan.

Rangkaian pengkondisi sinyal yang telah dijelaskan diatas berfungsi untuk menyadap sinyal jantung yang berkisar antara 0,03Hz sampai dengan kurang lebih 100Hz, dimana untuk melakukan *monitoring* jantung frekuensi yang dibutuhkan berkisar antara 0,03Hz sampai dengan 300Hz. Rangkaian *filter* hanya berfungsi untuk melakukan pengolahan penyaringan frekuensi jantung, sedangkan amplitudo sinyal yang dibutuhkan untuk pemrosesan pada mikrokontroller berkisar antara 0V sampai 5V, sehingga sinyal dibawah 0V atau diatas 5V tidak akan terbaca oleh mikrokontroller. Agar sinyal dapat terbaca oleh mikrokontroller maka diperlukan rangkaian tambahan untuk menggeser fasa, yaitu berupa rangkaian *summing adder*. Maka setelah semua rangkaian pengkondisi *filter* ditambahkan rangkaian *summing adder*, agar amplitudo sinyal dapat digeser antara 0V sampai dengan 5V.

3.4.4 Rangkaian *Lead* III

Rangkaian pada gambar 3.7 merupakan rangkaian pemrosesan *filter* pada *lead* III.



Gambar 3.7 Rangkaian *Lead* III

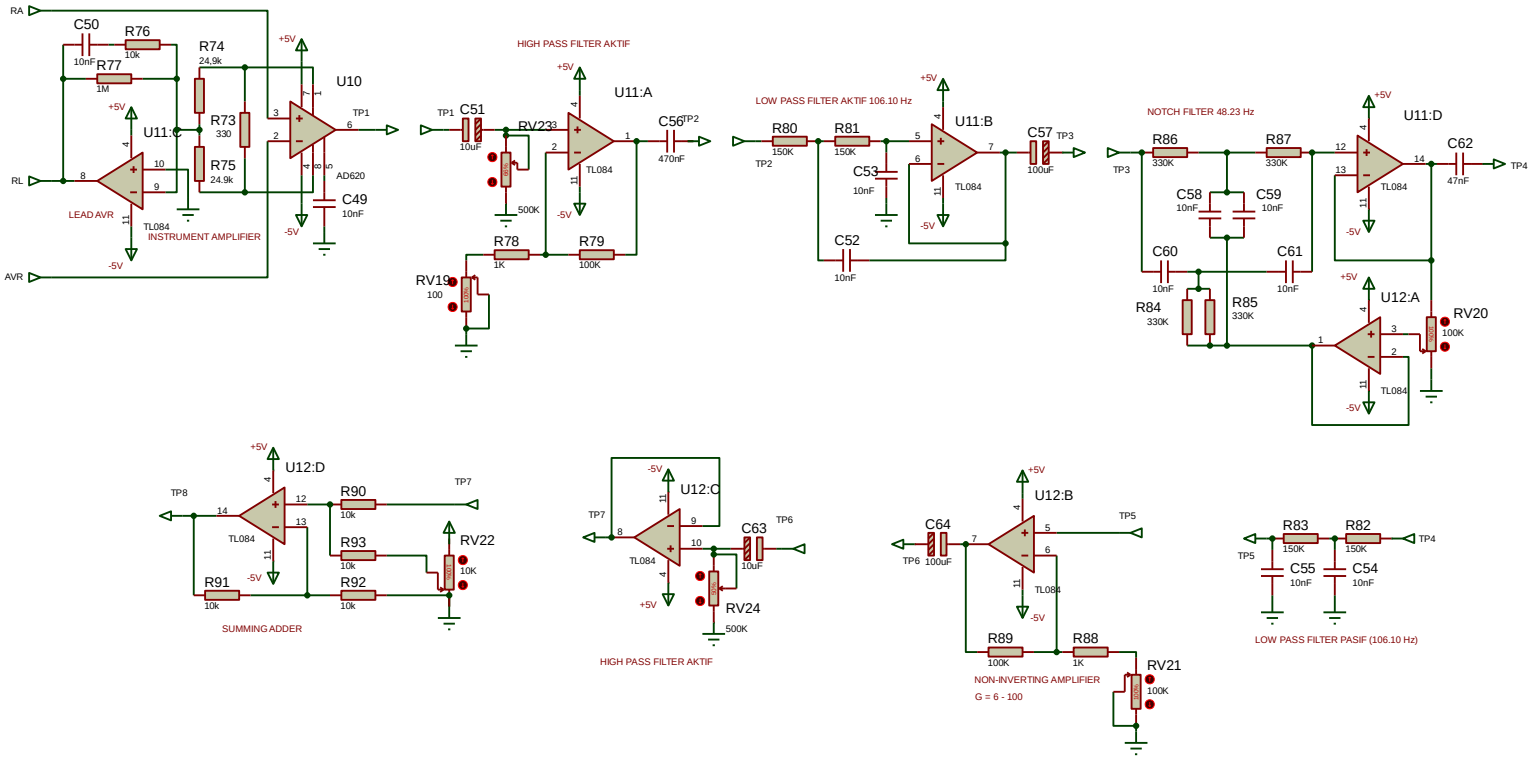
Rangkaian *Lead* III merupakan rangkaian yang digunakan untuk pemrosesan *filter* sinyal yang telah disadap atau didapat dari tubuh, sehingga sinyal tersebut dapat dipisahkan antara sinyal otot tubuh dengan sinyal otot jantung. *Input* pada *lead* III diperoleh melalui sadapan bipolar atau langsung dari elektroda LL,LA dan RL. Sinyal yang didapat dari elektroda akan diproses oleh *instrument amplifier* yang menggunakan IC AD620, selanjutnya sinyal tersebut akan diolah lagi untuk mengurangi *noise* atau interferensi dari sinyal tubuh. Setelah melewati *instrument amplifier*, sinyal diteruskan ke rangkaian HPF aktif -20dB 0,03Hz dengan penguatan sebesar 2 – 101 kali, setelah melewati rangkaian HPF sinyal diteruskan ke rangkaian *filter* selanjutnya, yaitu rangkaian LPF aktif -40dB 106,15Hz. Untuk mencegah adanya gangguan sinyal yang disebabkan oleh PLN (kurang lebih 50Hz) maka sinyal di*filter* dengan menggunakan rangkaian *notch filter* dengan frekuensi *cut-off* sebesar 48,3Hz. Setelah melaui *notch filter*, maka sinyal diteruskan ke rangkaian LPF pasif, -20dB, LPF pasif -20dB pada rangkaian berguna agar sinyal dapat lebih dikondisikan dan mendapatkan proses *filter* yang lebih baik. Selanjutnya sinyal diteruskan ke rangkaian penguat *non-inverting* agar sinyal yang masih berorde kecil dapat diatur amplitudonya sehingga mudah terbaca, karena pada rangkaian *filter* cukup banyak menggunakan komponen resistor, sehingga tidak menutup kemungkinan amplitudo sinyal akan dilemahkan. Setelah sinyal dapat terbaca, sinyal akan diteruskan ke rangkaian HPF pasif -20dB yang disusul oleh rangkaian *buffer*, sehingga menjadikannya rangkaian HPF aktif -20dB tanpa penguatan, rangkaian ini berfungsi untuk menanggulangi sinyal *noise* tubuh yang sekiranya masih lolos, sekaligus berfungsi sebagai *blocking* DC dan penyangga

agar sinyal yang telah diolah pada *filter-filter* sebelumnya tidak mengalami penurunan tegangan.

Rangkaian pengkondisi sinyal yang telah dijelaskan diatas berfungsi untuk menyadap sinyal jantung yang berkisar antara 0,03Hz sampai dengan kurang lebih 100Hz, dimana untuk melakukan *monitoring* jantung frekuensi yang dibutuhkan berkisar antara 0,03Hz sampai dengan 300Hz. Rangkaian *filter* hanya berfungsi untuk melakukan pengolahan penyaringan frekuensi jantung, sedangkan amplitudo sinyal yang dibutuhkan untuk pemrosesan pada mikrokontroller berkisar antara 0V sampai 5V, sehingga sinyal dibawah 0V atau diatas 5V tidak akan terbaca oleh mikrokontroller. Agar sinyal dapat terbaca oleh mikrokontroller maka diperlukan rangkaian tambahan untuk menggeser fasa, yaitu berupa rangkaian *summing adder*. Maka setelah semua rangkaian pengkondisi *filter* ditambabahkan rangkaian *summing adder*, agar amplitudo sinyal dapat digeser antara 0V sampai dengan 5V.

3.4.5 Rangkaian *Lead* AVR

Rangkaian pada gambar 3.8 merupakan rangkaian pemrosesan *filter* pada *lead* AVR.



Gambar 3.8 Rangkaian *Lead* AVR

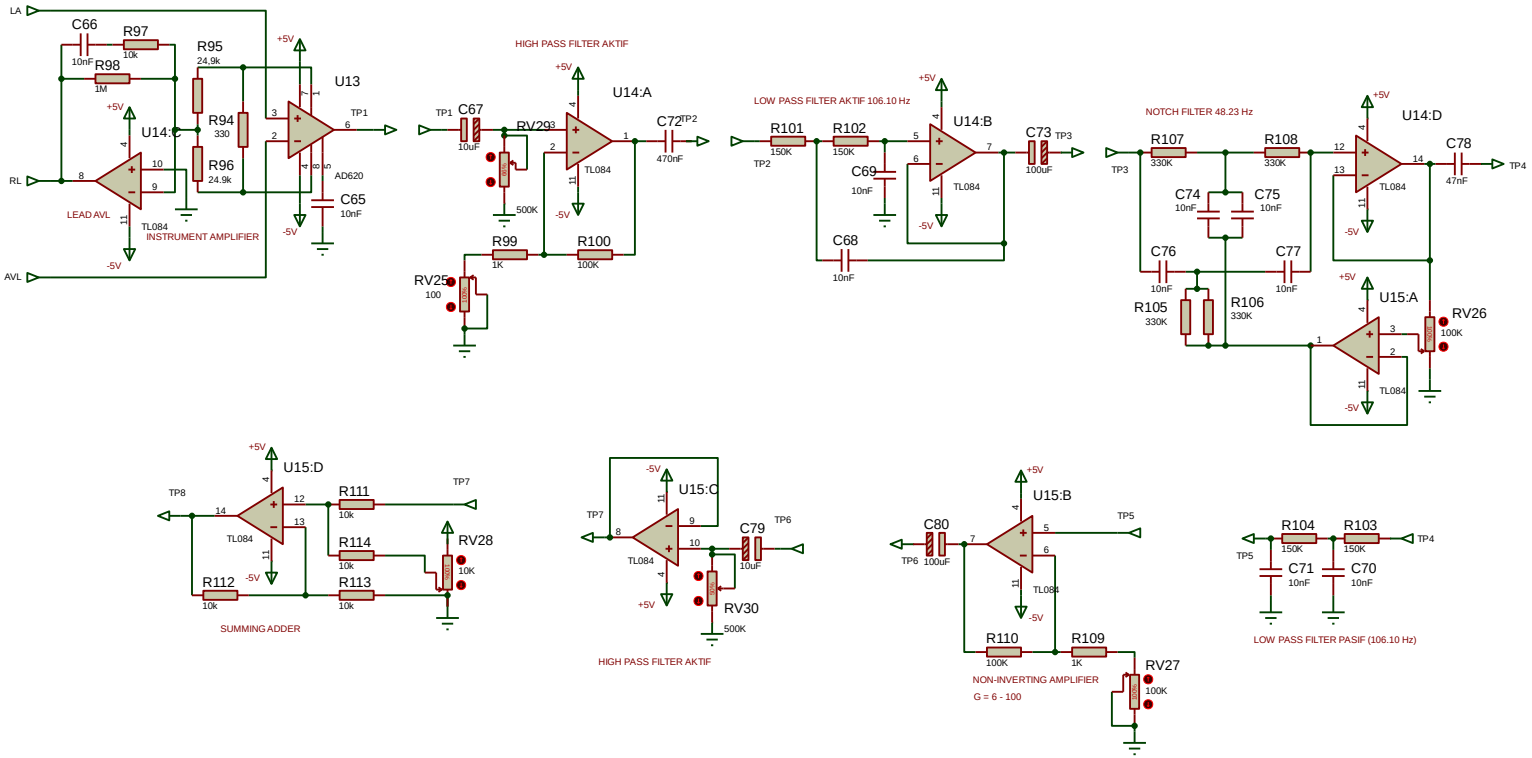
Rangkaian *lead* AVR merupakan rangkaian yang digunakan untuk pemrosesan *filter* sinyal yang telah disadap atau didapat dari tubuh, sehingga sinyal tersebut dapat dipisahkan antara sinyal otot tubuh dengan sinyal otot jantung. *Input* pada *lead* AVR diperoleh melalui sadapan unipolar RA, AVR dan RL. Sinyal yang didapat dari elektroda akan diproses oleh *instrument amplifier* yang menggunakan IC AD620, selanjutnya sinyal tersebut akan diolah lagi untuk mengurangi *noise* atau interferensi dari sinyal tubuh. Setelah melewati *instrument amplifier*, sinyal diteruskan ke rangkaian HPF aktif -20dB 0,03Hz dengan penguatan sebesar 2 – 101 kali, setelah melewati rangkaian HPF sinyal diteruskan ke rangkaian *filter* selanjutnya, yaitu rangkaian LPF aktif -40dB 106,15Hz. Untuk mencegah adanya gangguan sinyal yang disebabkan oleh PLN (kurang lebih 50Hz) maka sinyal di*filter* dengan menggunakan rangkaian *notch filter* dengan frekuensi *cut-off* sebesar 48,3Hz. Setelah melaui *notch filter*, maka sinyal diteruskan ke rangkaian LPF pasif, -20dB, LPF pasif -20dB pada rangkaian berguna agar sinyal dapat lebih dikondisikan dan mendapatkan proses *filter* yang lebih baik. Selanjutnya sinyal diteruskan ke rangkaian penguat *non-inverting* agar sinyal yang masih berorde kecil dapat diatur amplitudonya sehingga mudah terbaca, karena pada rangkaian *filter* cukup banyak menggunakan komponen resistor, sehingga tidak menutup kemungkinan amplitudo sinyal akan dilemahkan. Setelah sinyal dapat terbaca, sinyal akan diteruskan ke rangkaian HPF pasif -20dB yang disusul oleh rangkaian *buffer*, sehingga menjadikannya rangkaian HPF aktif -20dB tanpa penguatan, rangkaian ini berfungsi untuk menanggulangi sinyal *noise* tubuh yang sekiranya masih lolos, sekaligus berfungsi sebagai *blocking* DC dan penyangga agar sinyal

yang telah diolah pada *filter-filter* sebelumnya tidak mengalami penurunan tegangan.

Rangkaian pengkondisi sinyal yang telah dijelaskan diatas berfungsi untuk menyadap sinyal jantung yang berkisar antara 0,03Hz sampai dengan kurang lebih 100Hz, dimana untuk melakukan *monitoring* jantung frekuensi yang dibutuhkan berkisar antara 0,03Hz sampai dengan 300Hz. Rangkaian *filter* hanya berfungsi untuk melakukan pengolahan penyaringan frekuensi jantung, sedangkan amplitudo sinyal yang dibutuhkan untuk pemrosesan pada mikrokontroller berkisar antara 0V sampai 5V, sehingga sinyal dibawah 0V atau diatas 5V tidak akan terbaca oleh mikrokontroller. Agar sinyal dapat terbaca oleh mikrokontroller maka diperlukan rangkaian tambahan untuk menggeser fasa, yaitu berupa rangkaian *summing adder*. Maka setelah semua rangkaian pengkondisi *filter* ditambahkan rangkaian *summing adder*, agar amplitudo sinyal dapat digeser antara 0V sampai dengan 5V.

3.4.6 Rangkaian *Lead* AVL

Rangkaian pada gambar 3.9 merupakan rangkaian pemrosesan *filter* pada *lead* AVL.



Gambar 3.9 Rangkaian *Lead* AVL

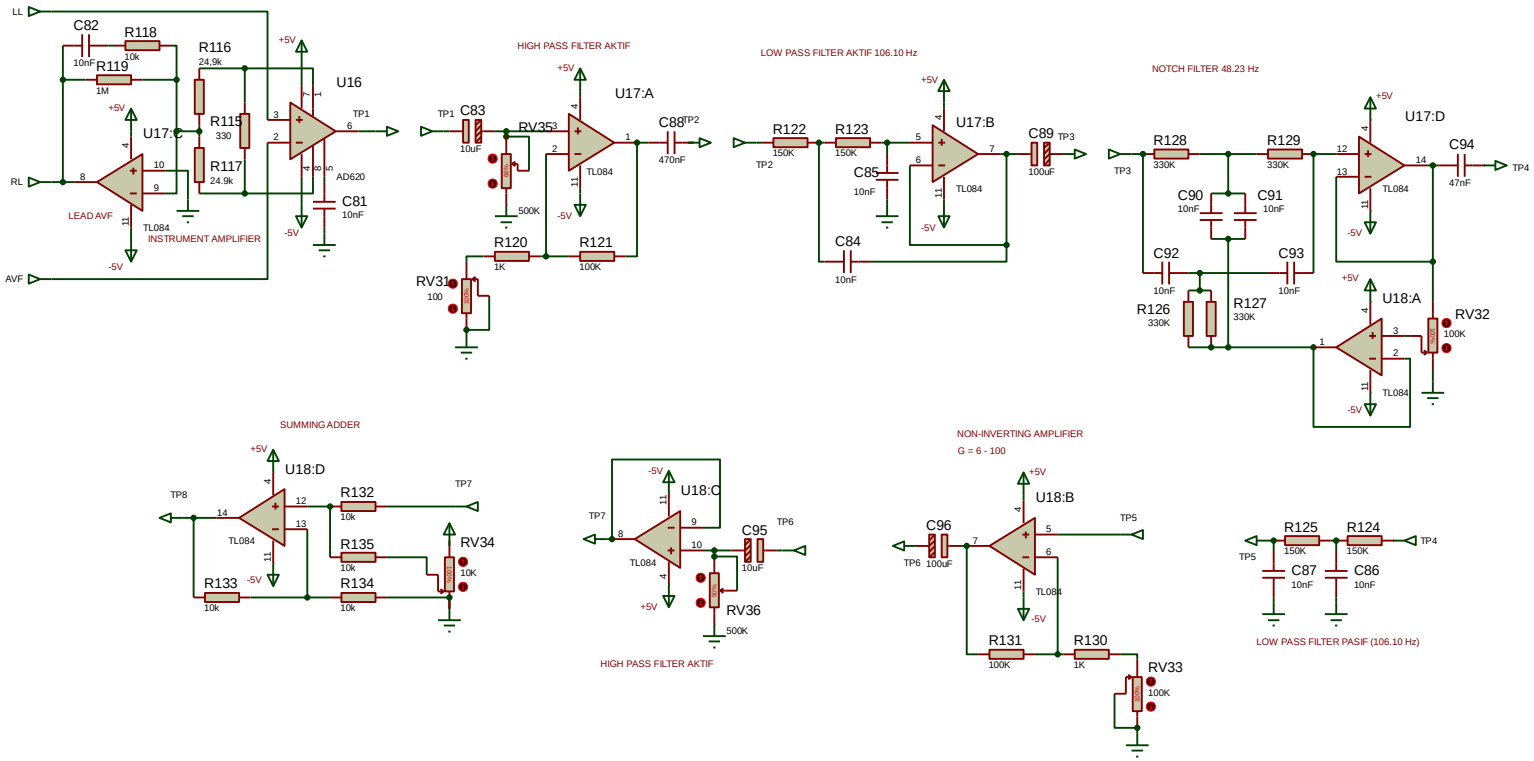
Rangkaian *lead* AVL merupakan rangkaian yang digunakan untuk pemrosesan *filter* sinyal yang telah disadap atau didapat dari tubuh, sehingga sinyal tersebut dapat dipisahkan antara sinyal otot tubuh dengan sinyal otot jantung. *Input* pada *lead* AVL diperoleh melalui sadapan unipolar LA, AVL dan RL. Sinyal yang didapat dari elektroda akan diproses oleh *instrument amplifier* yang menggunakan IC AD620, selanjutnya sinyal tersebut akan diolah lagi untuk mengurangi *noise* atau interferensi dari sinyal tubuh. Setelah melewati *instrument amplifier*, sinyal diteruskan ke rangkaian HPF aktif -20dB 0,03Hz dengan penguatan sebesar 2 – 101 kali, setelah melewati rangkaian HPF sinyal diteruskan ke rangkaian *filter* selanjutnya, yaitu rangkaian LPF aktif -40dB 106,15Hz. Untuk mencegah adanya gangguan sinyal yang disebabkan oleh PLN (kurang lebih 50Hz) maka sinyal di*filter* dengan menggunakan rangkaian *notch filter* dengan frekuensi *cut-off* sebesar 48,3Hz. Setelah melaui *notch filter*, maka sinyal diteruskan ke rangkaian LPF pasif, -20dB, LPF pasif -20dB pada rangkaian berguna agar sinyal dapat lebih dikondisikan dan mendapatkan proses *filter* yang lebih baik. Selanjutnya sinyal diteruskan ke rangkaian penguat *non-inverting* agar sinyal yang masih berorde kecil dapat diatur amplitudonya sehingga mudah terbaca, karena pada rangkaian *filter* cukup banyak menggunakan komponen resistor, sehingga tidak menutup kemungkinan amplitudo sinyal akan dilemahkan. Setelah sinyal dapat terbaca, sinyal akan diteruskan ke rangkaian HPF pasif -20dB yang disusul oleh rangkaian *buffer*, sehingga menjadikannya rangkaian HPF aktif -20dB tanpa penguatan, rangkaian ini berfungsi untuk menanggulangi sinyal *noise* tubuh yang sekiranya masih lolos, sekaligus berfungsi sebagai *blocking* DC dan penyangga agar sinyal

yang telah diolah pada *filter-filter* sebelumnya tidak mengalami penurunan tegangan.

Rangkaian pengkondisi sinyal yang telah dijelaskan diatas berfungsi untuk menyadap sinyal jantung yang berkisar antara 0,03Hz sampai dengan kurang lebih 100Hz, dimana untuk melakukan *monitoring* jantung frekuensi yang dibutuhkan berkisar antara 0,03Hz sampai dengan 300Hz. Rangkaian *filter* hanya berfungsi untuk melakukan pengolahan penyaringan frekuensi jantung, sedangkan amplitudo sinyal yang dibutuhkan untuk pemrosesan pada mikrokontroller berkisar antara 0V sampai 5V, sehingga sinyal dibawah 0V atau diatas 5V tidak akan terbaca oleh mikrokontroller. Agar sinyal dapat terbaca oleh mikrokontroller maka diperlukan rangkaian tambahan untuk menggeser fasa, yaitu berupa rangkaian *summing adder*. Maka setelah semua rangkaian pengkondisi *filter* ditambahkan rangkaian *summing adder*, agar amplitudo sinyal dapat digeser antara 0V sampai dengan 5V.

3.4.7 Rangkaian *Lead* AVF

Rangkaian pada gambar 3.10 merupakan rangkaian pemrosesan *filter* pada *lead* AVF.



Gambar 3.10 Rangkaian *Lead* AVF

Rangkaian *lead* AVF merupakan rangkaian yang digunakan untuk pemrosesan *filter* sinyal yang telah disadap / didapat dari tubuh, sehingga sinyal tersebut dapat dipisahkan antara sinyal otot tubuh dengan sinyal otot jantung. *Input* pada *lead* AVF diperoleh melalui sadapan unipolar LL, AVF dan RL. Sinyal yang didapat dari elektroda akan diproses oleh *instrument amplifier* yang menggunakan IC AD620, selanjutnya sinyal tersebut akan diolah lagi untuk mengurangi *noise* atau interferensi dari sinyal tubuh. Setelah melewati *instrument amplifier*, sinyal diteruskan ke rangkaian HPF aktif -20dB 0,03Hz dengan penguatan sebesar 2 – 101 kali, setelah melewati rangkaian HPF sinyal diteruskan ke rangkaian *filter* selanjutnya, yaitu rangkaian LPF aktif -40dB 106,15Hz. Untuk mencegah adanya gangguan sinyal yang disebabkan oleh PLN (kurang lebih 50Hz) maka sinyal di*filter* dengan menggunakan rangkaian *notch filter* dengan frekuensi *cut-off* sebesar 48,3Hz. Setelah melaui *notch filter*, maka sinyal diteruskan ke rangkaian LPF pasif, -20dB, LPF pasif -20dB pada rangkaian berguna agar sinyal dapat lebih dikondisikan dan mendapatkan proses *filter* yang lebih baik. Selanjutnya sinyal diteruskan ke rangkaian penguat *non-inverting* agar sinyal yang masih berorde kecil dapat diatur amplitudonya sehingga mudah terbaca, karena pada rangkaian *filter* cukup banyak menggunakan komponen resistor, sehingga tidak menutup kemungkinan amplitudo sinyal akan dilemahkan. Setelah sinyal dapat terbaca, sinyal akan diteruskan ke rangkaian HPF pasif -20dB yang disusul oleh rangkaian *buffer*, sehingga menjadikannya rangkaian HPF aktif -20dB tanpa penguatan, rangkaian ini berfungsi untuk menanggulangi sinyal *noise* tubuh yang sekiranya masih lolos, sekaligus berfungsi sebagai *blocking* DC dan penyangga agar sinyal

yang telah diolah pada *filter-filter* sebelumnya tidak mengalami penurunan tegangan.

Rangkaian pengkondisi sinyal yang telah dijelaskan diatas berfungsi untuk menyadap sinyal jantung yang berkisar antara 0,03Hz sampai dengan kurang lebih 100Hz, dimana untuk melakukan *monitoring* jantung frekuensi yang dibutuhkan berkisar antara 0,03Hz sampai dengan 300Hz. Rangkaian *filter* hanya berfungsi untuk melakukan pengolahan penyaringan frekuensi jantung, sedangkan amplitudo sinyal yang dibutuhkan untuk pemrosesan pada mikrokontroller berkisar antara 0V sampai 5V, sehingga sinyal dibawah 0V atau diatas 5V tidak akan terbaca oleh mikrokontroller. Agar sinyal dapat terbaca oleh mikrokontroller maka diperlukan rangkaian tambahan untuk menggeser fasa, yaitu berupa rangkaian *summing adder*. Maka setelah semua rangkaian pengkondisi *filter* ditambahkan rangkaian *summing adder*, agar amplitudo sinyal dapat digeser antara 0V sampai dengan 5V.

3.5 *Listing* Program

Pengolahan program dibedakan menjadi 4 tahap, yaitu tahap pengiriman pada arduino *transmitter* (Arduino TX), penerimaan serta *parsing* data pada arduino *receiver* (Arduino RX) dan pengolahan grafik yang dilakukan pada arduino RX.

3.5.1 *Listing* Program Pengolahan ADC (Arduino TX)

Berikut ini merupakan pengolahan program pada arduino TX untuk pembacaan sinyal analog sekaligus memproses ADC dan pengiriman data melalui *serial* komunikasi (TX/RX).

```
    a = analogRead(A0);
    b = analogRead(A1);
    c = analogRead(A2);
    d = analogRead(A3);
    e = analogRead(A4);
    f = analogRead(A5);
    Serial.print ("A ");
    Serial. println (a);
    Serial.print ("B ");
    Serial. println (b);
    Serial.print ("C ");
    Serial. println (c);
    Serial.print ("D ");
    Serial. println (d);
    Serial.print ("E ");
    Serial. println (e);
    Serial.print ("F ");
    Serial. println (f);
```

Listing Program 3.1 Program ADC dan Pengiriman Data

Data analog yang telah diolah oleh rangkaian pengolah sinyal ECG diteruskan ke *pin* ADC pada kaki A0 (*Lead* I), A1 (*Lead* II), A2 (*Lead* III), A3 (*Lead* AVR), A4 (*Lead* AVL) dan A5 (*Lead* AVF) Arduino. Setelah proses pengambilan sinyal dan pengolahan ADC, selanjutnya nilai ADC dari masing-masing *lead* dikirimkan oleh arduino melalui *serial* komunikasi *pin* TX (*pin* 1). Proses pengiriman data pada setiap *lead* dipisahkan dengan simbol berupa karakter huruf.

3.5.2 *Listing* Program Penerimaan Data *Serial* dan *Parsing* (Pemisahan) Data (Arduino RX)

Data ADC yang dikirim oleh arduino TX lalu diterima oleh arduino RX melalui pin RX (pin 0), data tersebut nantinya akan diolah menjadi grafik yang tertampil.

```
#include <Messenger.h>
  if ( message.checkString("A") ) {
    I = message.readInt();
  }
  if ( message.checkString("B") ) {
    II = message.readInt();
  }
  if ( message.checkString("C") ) {
    III = message.readInt();
  }
if ( message.checkString("D") ) {
    IV = message.readInt();
  }
  if ( message.checkString("E") ) {
    V = message.readInt();
  }
  if ( message.checkString("F") ) {
    VI = message.readInt();
  }
```

Listing Program 3.2 Penerimaan dan *Parsing* Data

Data yang diterima oleh arduino RX merupakan data acak dari semua sinyal *lead* yang dipisahkan oleh karakter huruf, sehingga diperlukan pemisahan data per-*lead* agar dapat diolah menjadi grafik, proses pemisahan menggunakan library *messenger* yang berfungsi untuk pemilahan nilai berupa *integer* dibelakang karakter *string*, sehingga data yang diterima dapat diambil nilai *integer*nya dan dapat diolah menjadi grafik.

3.5.3 *Listing* Program Pengolahan Grafik (Arduino RX)

Data yang telah diterima selanjutnya diolah lagi per-bagian agar dapat menjadi bentuk grafik. Proses menampilkan grafik menggunakan *input library* yang diperlukan untuk menampilkan grafik pada LCD TFT. Berikut ini merupakan program yang digunakan dalam proses pengolahan grafik.

```
#include <Adafruit_GFX.h>
#include <MCUFRIEND_kbv.h>
      up1 = 26; down1 = 75;
      up2 = 126; down2 = 175;
      up3 = 226; down3 = 275;
      up4 = 26; down4 = 75;
      up5 = 126; down5 = 175;
      up6 = 226; down6 = 275;
    Lead1 = map(I, 0, 1023, down1, up1);
    Lead2 = map(II, 0, 1023, down2, up2);
    Lead3 = map(III, 0, 1023, down3, up3);
    Lead4 = map(IV, 0, 1023, down4, up4);
    Lead5 = map(V, 0, 1023, down5, up5);
    Lead6 = map(VI, 0, 1023, down6, up6);
```

Listing Program 3.3 *Mapping* Data

Sebelum melakukan pengolahan grafik, dilakukan *mapping* data atau pemetaan data terlebih dahulu. *Mapping* ini berfungsi untuk memetakan data dari ADC yang meiliki nilai 0 sampai 1023 menjadi nilai dalam rentang tertentu. Proses ini untuk memudahkan pembagian letak grafik yang akan ditampilkan pada LCD.

```
void AllLead() {
  if (x_pos <= 201) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, Lead1_old,
      x_pos, Lead1, GREEN);
      Lead1_old = Lead1;
      TFT.drawLine(x_pos - x_scale, Lead2_old,
      x_pos, Lead2, RED);
      Lead2_old = Lead2;
      TFT.drawLine(x_pos - x_scale, Lead3_old,
      x_pos, Lead3, BLUE);
      Lead3_old = Lead3;
      TFT.fillRect(x_pos + 1, 0, x_scale, 306,
      BLACK);
      }
  if (x_pos2 <= 403) {
    x_pos2 += x_scale;
      TFT.drawLine(x_pos2 - x_scale, Lead4_old,
      x_pos2, Lead4, YELLOW);
      Lead4_old = Lead4;
      TFT.drawLine(x_pos2 - x_scale, Lead5_old,
      x_pos2, Lead5, MAGENTA);
      Lead5_old = Lead5;
      TFT.drawLine(x_pos2 - x_scale, Lead6_old,
      x_pos2, Lead6, CYAN);
      Lead6_old = Lead6;
```

```
      TFT.fillRect(x_pos2 + 1, 0, x_scale, 306,
      BLACK);

      }

  if (x_pos >= 200 || x_pos2 >= 402) {

    x_pos = (1 + x_scale);

    x_pos2 = (203 + x_scale);

      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      306 , BLACK);

      TFT.fillRect(202, 0, x_scale + (x_scale + 2),
      306, BLACK);

      }

}
```

Listing Program 3.4 Pengolahan Grafik Semua *Lead*

Penampilan semua *lead* dilakukan dengan pembagian 6 tempat dengan 2 kolom dan 3 baris. Kolom pertama untuk *lead* I, *lead* II dan *lead* III. Kolom kedua untuk *lead* AVR, *lead* AVL dan *lead* AVF. Kolom pertama dimulai pada posisi x = 1 sampai dengan 201, sedangkan kolom kedua dimulai pada posisi x = 203 sampai dengan 403, sehingga masing-masing kolom memperoleh panjang x sepanjang 200 titik, sedangkan untuk tinggi y telah dilakukan pemetaan pada *listing* program sebelumnya. *Lead* I ditandai dengan grafik warna hijau, *lead* II ditandai dengan grafik warna merah, *lead* III ditandai dengan grafik warna biru, *lead* AVR ditandai dengan grafik warna kuning, *lead* AVL ditandai dengan grafik warna *magenta* dan *lead* AVF ditandai dengan grafik warna *cyan*.

```
void Lead123() {
  if (x_pos <= 403) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, Lead1_old,
      x_pos, Lead1, GREEN);
    Lead1_old = Lead1;
      TFT.drawLine(x_pos - x_scale, Lead2_old,
      x_pos, Lead2, RED);
    Lead2_old = Lead2;
      TFT.drawLine(x_pos - x_scale, Lead3_old,
      x_pos, Lead3, BLUE);
    Lead3_old = Lead3;
      TFT.fillRect(x_pos + 1, 0, x_scale, 306,
      BLACK);
  }
  if (x_pos >= 403) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      306 , BLACK);
  }
}
```

Listing Program 3.5 Pengolahan Grafik *Lead* I, II, III

Penampilan 3 *lead* pada l *lead* I, *lead* II dan *lead* III menggunakan 1 kolom dengan 3 baris. Dimana pada tiap baris mendapat 402 panjang titik x, dimulai dari x = 1 sampai x = 403, sedangkan tinggi titik y sebagai amplitudo memiliki tinggi yang sama dengan penampilan sebelumnya ketika menampilkan semua *lead*.

```
void Lead456() {
  if (x_pos <= 403) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, Lead4_old,
      x_pos, Lead4, YELLOW);
    Lead4_old = Lead4;
      TFT.drawLine(x_pos - x_scale, Lead5_old,
      x_pos, Lead5, MAGENTA);
    Lead5_old = Lead5;
      TFT.drawLine(x_pos - x_scale, Lead6_old,
      x_pos, Lead6, CYAN);
    Lead6_old = Lead6;
      TFT.fillRect(x_pos + 1, 0, x_scale, 306,
      BLACK);
  }
  if (x_pos >= 403) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      305 , BLACK);
  }
}
```

Listing Program 3.6 Pengolahan Grafik *Lead* AVR, AVL, AVF

Tampilan 3 *lead* selanjutnya adalah *lead* AVR, *lead* AVL dan *lead* AVF, tampilan ini memiliki kodingan yang sama dengan tampilan 3 *lead* sebelumnya yang membedakan hanya pada warna grafik yang digunakan.

```
 void LeadI() {
  if (x_pos <= 478) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, SLead1_old,
      x_pos, SLead1, GREEN);
      TFT.fillRect(x_pos + 1, 0, x_scale, 250,
      BLACK);
    SLead1_old = SLead1;
  }
  if (x_pos >= 477) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      250, BLACK);
  }
}
void LeadII() {
  if (x_pos <= 478) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, SLead2_old,
      x_pos, SLead2, RED);
      TFT.fillRect(x_pos + 1, 0, x_scale, 249,
      BLACK);
    SLead2_old = SLead2;
  }
  if (x_pos >= 477) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      250, BLACK);
  }
}
```

```
void LeadIII() {
  if (x_pos <= 478) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, SLead3_old,
      x_pos, SLead3, BLUE);
      TFT.fillRect(x_pos + 1, 0, x_scale, 249,
      BLACK);
    SLead3_old = SLead3;
  }
  if (x_pos >= 477) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      250, BLACK);
  }
}
void LeadAVR() {
  if (x_pos <= 478) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, SLead4_old,
      x_pos, SLead4, YELLOW);
      TFT.fillRect(x_pos + 1, 0, x_scale, 249,
      BLACK);
    SLead4_old = SLead4;
  }
  if (x_pos >= 477) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      250, BLACK);
  }
}
```

```
void LeadAVL() {
  if (x_pos <= 478) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, SLead5_old,
      x_pos, SLead5, MAGENTA);
      TFT.fillRect(x_pos + 1, 0, x_scale, 249,
      BLACK);
    SLead5_old = SLead5;
  }
  if (x_pos >= 477) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      250, BLACK);
  }
}
void LeadAVF() {
  if (x_pos <= 478) {
    x_pos += x_scale;
      TFT.drawLine(x_pos - x_scale, SLead6_old,
      x_pos, SLead6, CYAN);
      TFT.fillRect(x_pos + 1, 0, x_scale, 249,
      BLACK);
    SLead6_old = SLead6;
  }
  if (x_pos >= 477) {
    x_pos = (1 + x_scale);
      TFT.fillRect(0, 0, x_scale + (x_scale + 2),
      250, BLACK);
  }
}
```

Listing Program 3.7 Pengolahan Grafik Setiap *Lead*

Penampilan grafik per*lead* kurang lebih sama dengan program sebelumnya, warna grafik yang digunakan pada setiap *lead* juga menggunakan warna yang sama seperti koding sebelumnya, yang membedakan adalah pemetaan menggunakan 1 baris serta1 kolom penuh dengan panjang titik x sebesar 478 titik. Jika panjang titik x kurang dari 478 maka grafik akan tergambar hingga titik x memiliki panjang 477, jika melebihi titik 477 maka grafik akan kembali lagi ke posisi awal. Sedangkan pemetaan yang digunakan sebagai amplitudo telah dijelaskan pada *listing* Program 3.3 *Mapping* data.

3.5.4 *Listing* Program BPM (Arduino TX)

Berikut ini merupakan pengolahan program pada Arduino TX yang berfungsi untuk melakukan perhitungan BPM.

```
int lastn=0;
int LastTime=0;
int ThisTime;
bool BPMTiming=false;
bool BeatComplete=false;
int BPM=0;
#define UpperThreshold 700
#define LowerThreshold 600


void HeartRate(){
ThisTime=millis();
int value=analogRead(A1);
int n=60-(value/16);
lastn=n;
```

```
if(value>UpperThreshold){
if(BeatComplete){
BPM=ThisTime-LastTime;
BPM=int(60/(float(BPM)/1000));
BPMTiming=false; BeatComplete=false; }
if(BPMTiming==false){
LastTime=millis();
BPMTiming=true; } }
if((value<LowerThreshold)&(BPMTiming))
BeatComplete=true; }
```

Listing Program 3.8 Pengolahan BPM

Pengolahan program BPM dilakukan untuk mendeteksi gelombang *peak to peak* atau gelombang puncak tertinggi dari sinyal jantung, yaitu sinyal R. Pendeteksian sinyal tertinggi dilakukan dengan cara menentukan nilai *threshold* atau ambang batas nilai gelombang yang dideteksi oleh mikrokontroller, dimana ketika nilai *threshold* telah tercapai maka terhitung 1 gelombang puncak. Gelombang puncak yang telah terdeteksi *threshold* selanjutnya diukur waktu tempuhnya dari puncak gelombang 1 ke puncak gelombang selanjutnya. Waktu tempuh yang telah didapat dari puncak ke puncak pada setiap sinyal selanjutnya dikalkulasikan menjadi hitungan tiap menit, sehingga didapat perhitungan waktu tempuh puncak sinyal dalam satuan *Beat Per Minute* (BPM).

3.6 Alat dan Bahan

Untuk menunjang pelaksanaan penelitian ini, maka dibutuhkan beberapa alat dan bahan dalam proses kelancaran pembuatan *Portable Electrocardiograph* 6 *Lead* Berbasis Arduino Nano.

3.6.1 Alat

Berikut daftar alat penunjang yang digunakan selama proses perancangan.

Tabel 3.1 Alat

No	Nama Alat	Jumlah
1	Solder	1 buah
2	*Atractor*	1 buah
3	*Osciloscope*	1 buah
4	*Phantom* ECG	1 buah
5	Multimeter	1 buah
6	Obeng	2 buah
7	Bor	1 buah
8	Laptop	1 unit
9	Tang Potong	1 buah
10	Tang Jepit	1 buah
No	**Nama Alat**	**Jumlah**
11	*Cutter*	1 buah
12	Elektroda jepit	1 set
13	*Function Generator*	1 buah

3.6.2 Bahan

Berikut daftar bahan yang dibutuhkan selama proses perancangan.

Tabel 3.2 Bahan

No	Nama Bahan / Komponen	Jumlah
1	Rangkaian ECG	
	IC AD620 + *socket*	6 buah
	IC TL084 + *socket*	12 buah
	Resistor	135 buah
	Kapasitor	95 buah
	Multiturn	34 buah
	Header	6 sisir

No	Nama Bahan / Komponen	Jumlah
	Kabel *jumper*	2 sisir
	Saklar *on*/*off*	1 buah
	Kabel	5 meter
	Jepit Buaya	
	PCB	68cmx36cm
2	Mikrokontroller	
	Modul Arduino Nano	2 buah
3.	Display	
	LCD TFT	1 buah
4	Bahan Lain	
	Timah	1 *roll*
	Solder *wick*	1 *roll*
	Elektroda *disposible*	1 set

3.7 Teknik Analisis Data

Teknis analisis data yang digunakan dengan cara membandingkan rancangan EKG yang dibuat dengan *phantom* EKG dan dilakukan perhitungan nilai rata-rata serta simpangan *error.*

3.7.1 Rata-rata

Rata-rata (*mean)* adalah nilai suatu bilangan yang mewakili sekumpulan data, nilai ini didapat dari hasil pembagian banyaknya jumlah data yang diambil. Rumus untuk mencari nilai rata-rata adalah sebagai berikut.

$$\text{Rata-rata } (\bar{X}) = \frac{\sum xi}{n} \quad \text{..... (3-1)}$$

Keterangan :

$\bar{X}$ = rata-rata

$\sum xi$ = jumlah nilai data

n = banyak data (1,2,3,...,n)

3.7.2 *Error*

Error atau penyimpangan adalah selisih atau perbedaan nilai antara nilai rata-rata dengan masing-masing data. Rumus untuk mencari *error* adalah sebagai berikut.

Error (%) = $\frac{Y-X}{Y} \times 100\%$ ..... (3-2)

Keterangan :

Y = Data *setting*

X = Rata-rata (*mean*)

3.8 Teknik Pengujian

Teknik pengujian dilakukan untuk mengukur dan mengetahui hasil dari modul alat rancangan bekerja dengan baik atau tidak. *Oscilloscope* dan *phantom* ECG digunakan sebagai alat pembanding untuk pengujian rangkaian pengkondisi sinyal dan *output visual*. Berikut teknik pengujian yang dilakukan.

1. Mengukur dan menganalisis stabilitas tegangan *output* di setiap rangkaian *instrument amplifier* pada *lead* I, *lead* II, *lead* III, *lead* AVR, *lead* AVL dan *lead* AVF.
2. Mengukur dan menganalisis stabilitas frekuensi *cut-off* pada setiap rangkaian *filter*.
3. Menguji hasil gelombang *output* rangkaian dengan menggunakan *phantom* ECG.
4. Menguji dan menganalisis program mikrokontroller Arduino yang digunakan untuk *output visual* LCD TFT.

5. Menguji dan membandingkan hasil *output visual* dari keseluruhan rangkaian dengan *phantom* ECG dengan hasil *visual* yang ditampilkan oleh *oscilloscope*.

3.9 Waktu dan Tempat Pembuatan Rancangan

Waktu : September 2018 – Juni 2019

Tempat : Rejodani, Sariharjo, Ngaglik, Sleman, Daerah Istimewa Yogyakarta.

Laboratorium Teknologi Elektro-medis, Kampus Elektromedik, Program Vokasi, Universitas Muhammadiyah Yogyakarta.