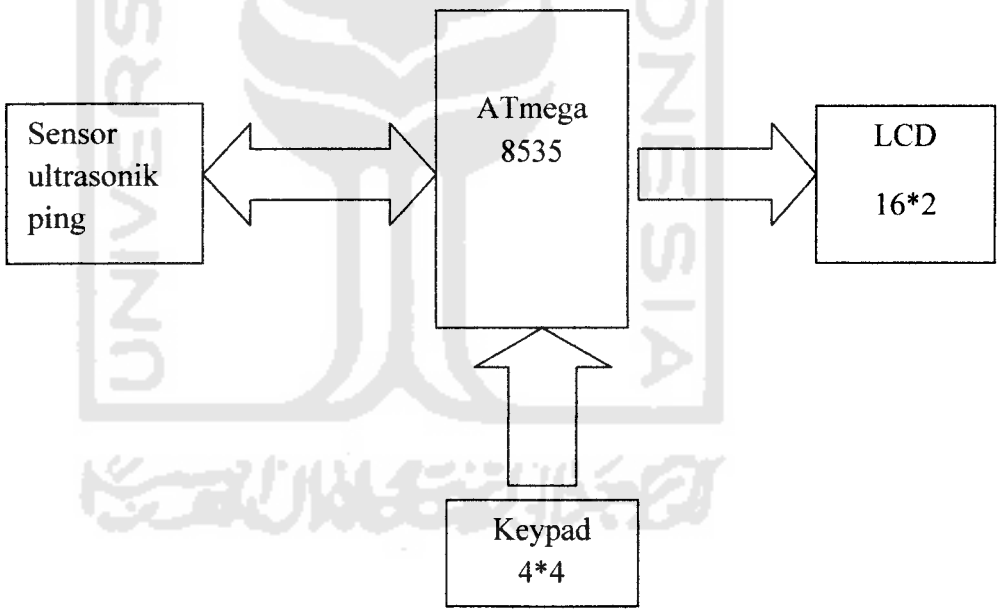
BAB III

PERANCANGAN SISTEM

3.1 Perancangan Alat

Dalam pembuatan sistem yang lengkap, harus mengetahui terlebih dahulu gambaran umum tentang rancangan yang akan dibuat. Hal ini perlu dilakukan agar pembuatan alat lebih terkonsep. Secara garis besar model rancangan alat pengukur jarak adalah sebagai berikut :



Gambar 3.1 Skema Model Perancangan Alat

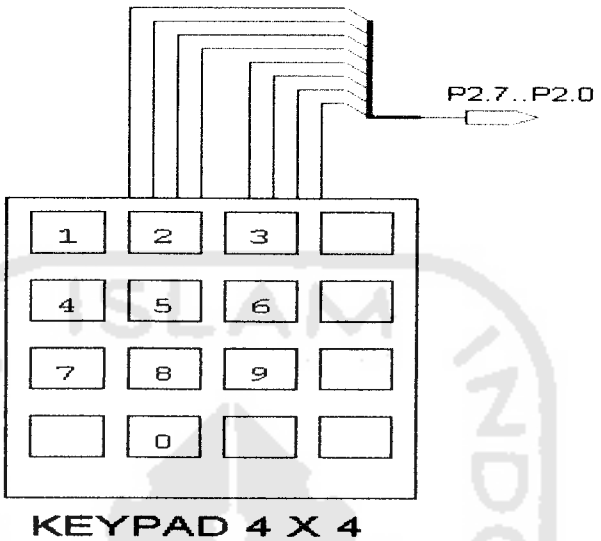
Pada dasarnya sistem ini terdiri dari 3 bagian penting, bagian yang pertama adalah sensor ultrasonik. Sensor ini bekerja dengan cara memancarkan gelombang ultrasonik ke obyek benda yang akan diukur dan kemudian gelombang yang dipantulkan oleh obyek diterima kembali oleh sensor. Dimana sensor akan

menghasilkan sebuah pulsa yang menpresentsikan jarak benda. Untuk selanjutnya mikrokontroller ATMega8535 akan memproses sinyal masukan yang di dapat dari sensor ultrasonik dengan bantuan *software* yang telah diinstruksikan untuk memabaca, memproses dan mengeluarkan hasilnya yang akan ditampilkan pada bagian terakhir yaitu pada layar LCD. Tampilan pada LCD berupa jarak obyek dengan alat dalam satuan millimeter (mm). Sedangkan untuk *keypad* merupakan fitur tambahan, yang berfungsi untuk mengaktifkan, penyimpanan data hasil pengukuran, menghapus data, dan mengeluarkan data yang telah disimpan untuk dapat ditampilkan pada layar LCD.

3.2 Perancangan Hardware

Pada perancangan *hardware* penelitian ini, seluruh rangkaian elektronik baik yang merupakan rangkaian kontroler maupun utilitas, semuanya terhubung dengan *wiring* secara fisik. Perancangan *hardware* ini didukung oleh rangkaian-rangkaian elektris yang membantu kerja mikrokontroller sebagai pengendali utama, seperti: sistem minimum, *power supply* serta rangkaian elektris lainnya yang menjalankan sistem secara keseluruhan. Berikut merupakan komponen-komponen yang digunakan.

3.2.1 Rangkaian *Keypad* 4x4



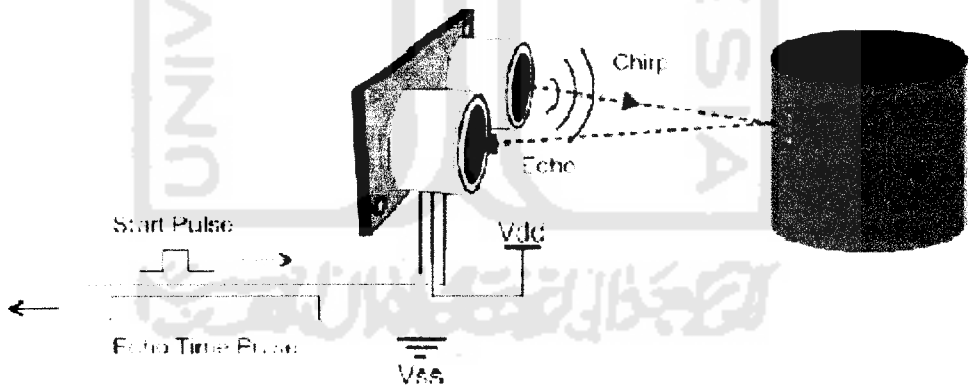
Gambar 3.2 Rangkaian *Keypad*

Keypad digunakan sebagai media *input*, dalam satu rangkaian *keypad* mempunyai 16 tombol *push button,* masing-masing tombol mempunyai fungsi yang berbeda, yaitu :

1. Tombol A : Untuk memulai pengukuran
2. Tombol B : Untuk mngakses slot eeprom berikutnya sebagai tempat penyimpanan data (*forward*)
3. Tombol C : Untuk membaca data yang telah disimpan (*backward*)
4. Tombol D : untuk menyimpan data
5. Tombol * : Untuk mengakses data yang ingin kita tampilkan
6. Tombol # : Untuk menampilkan data yang dipilih
7. Tombol 0 – 9 : Sebagai media pemilih data (maksimal 2 *digit*, misal 01-99)
8. Pushbutton : sebagai penghapus data

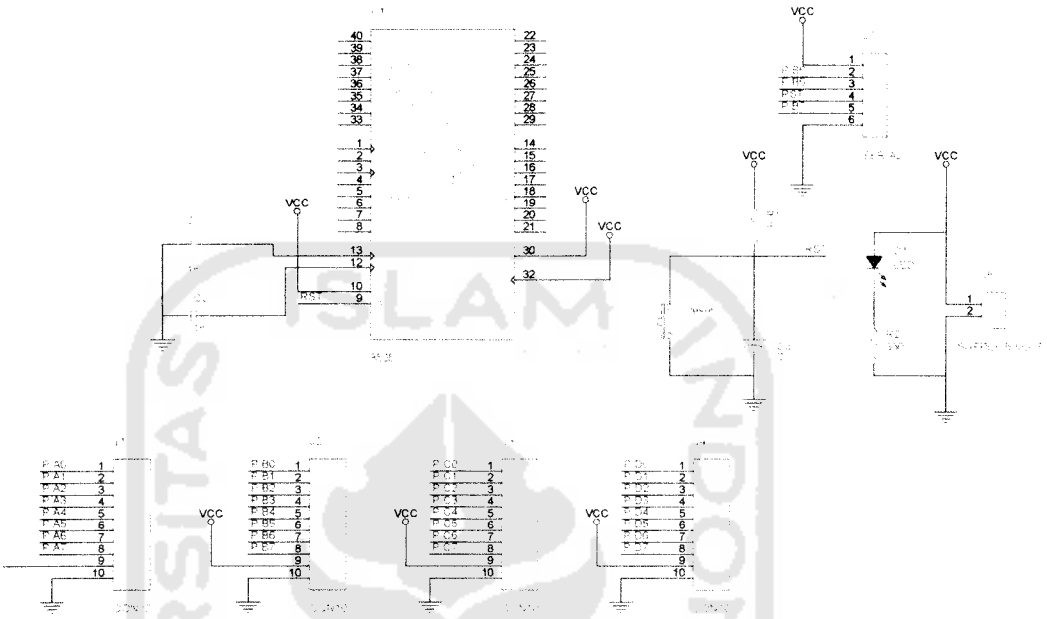
3.2.3 Sensor PING

Sensor PING mendeteksi jarak obyek dengan cara memancarkan gelombang ultrasonik (40 KHz) selama t (200µs) kemudian mendeteksi pantulannya. Sensor PING memancarkan gelombang sesuai dengan kontrol dari mikrokontroller pengendali. Gelombang ultrasonik ini melalui udara dengan kecepatan 344 meter per detik, mengenai obyek dan memantul kembali ke sensor. PING mengeluarkan pulsa *output high* pada pin SIG setelah memancarkan gelombang ultrasonik dan setelah gelombang pantulan terdeteksi PING akan membuat *output low* pada pin SIG. lebar pulsa *High* akan sesuai dengan lama waktu tempuh gelombang ultrasonik untuk 2x jarak ukur dengan obyek.



Gambar 3. 4 Cara Kerja Sensor PING

3.2.4 Sistem Minimum Mikrokontroller ATMega8535



Gambar 3.5 Sistem Minimum Mikrokontroller ATMega 8535

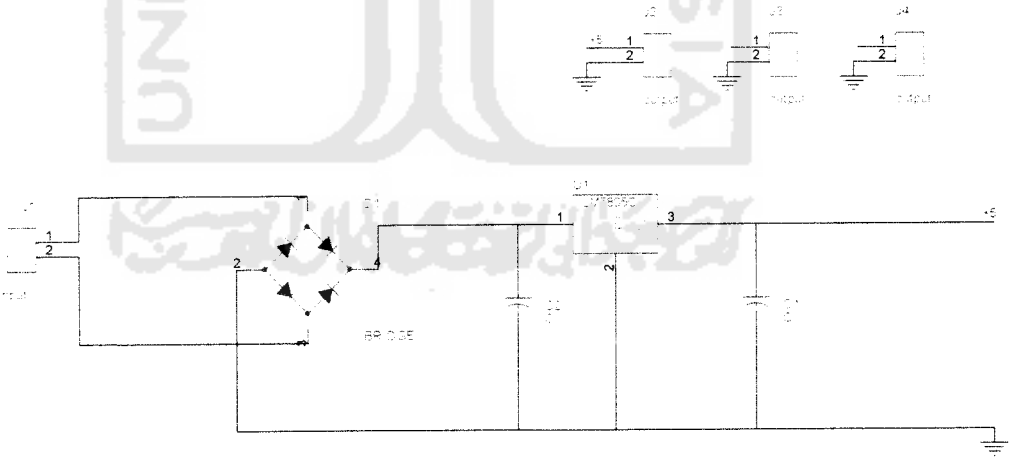
Mikrokontroler AVR memiliki arsitektur RISC 8 bit, dimana semua instruksi dikemas dalam kode 16-bit (16-*bits word*) dan sebagian besar instruksi dieksekusi dalam 1 (satu) siklus *clock*. Rangkaian osilator pada sistem ini digunakan oleh mikrokontroller sebagai sinyal denyut (*clock*). Dengan memanfaatkan kristal internal sebesar 4MHz sebagai pembangkit sinyal denyut (*clock*).

Dengan mengkoneksikan LCD M1632 dengan port A pada mikrokontroller, untuk pin yang digunakan adalah pin 2 sampai dengan pin 8 pada port A. Port D di gunakan sebagai pengkoneksi antara sensor PING dengan mikrokontroller, sensor PING hanya membutuhkan 1 pin sebagai media

koneksinya yaitu melalui pin SIG yang berfungsi sebagai input sekaligus output data, sedangkan 2 pin lainnya adalah sebagai sumber catu daya 5 volt dan *ground.*

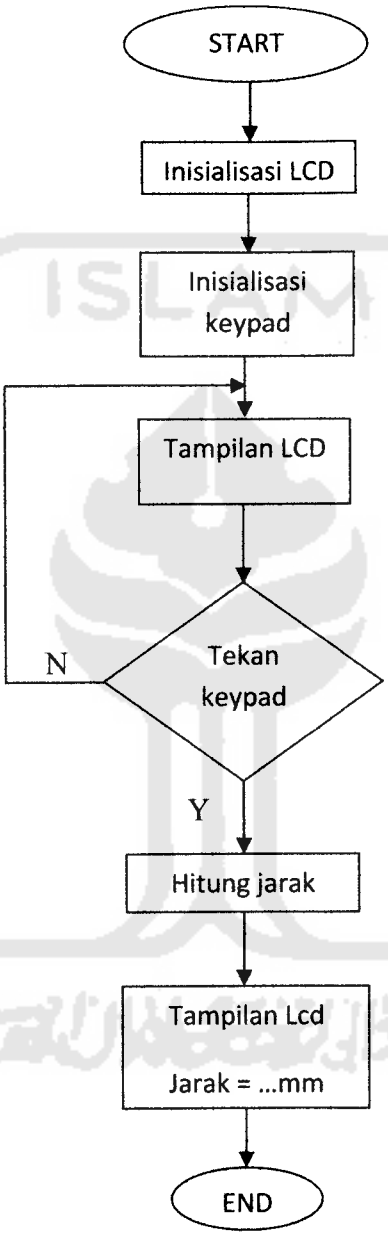
3.2.5 Rangkaian Catu Daya

Setiap rangkaian elektronik tentunya membutuhkan catu daya, sehingga perancangan catu daya menjadi hal yang sangat penting, agar rangkaian ini dapat memberikan kebutuhan arus dan tegangan yang sesuai. Selain arus dan tegangan yang sesuai, hal lain yang perlu diperhatikan adalah kestabilan dari tegangan dan arus tersebut. Pada alat catu daya yang digunakan terdiri dari +5V, dan 220 AC. Untuk rancangan catu daya +5V berikut menggunakan IC LM7805 sebagai regulator tegangannya, sehingga tegangan outputnya lebih stabil. Selain itu dilengkapi dengan kapasitor C1 dan C2 sebagai filter tegangan.



Gambar 3.6 Skema Rangkaian Catu Daya DC 5 volt

3.3 Perancangan Software

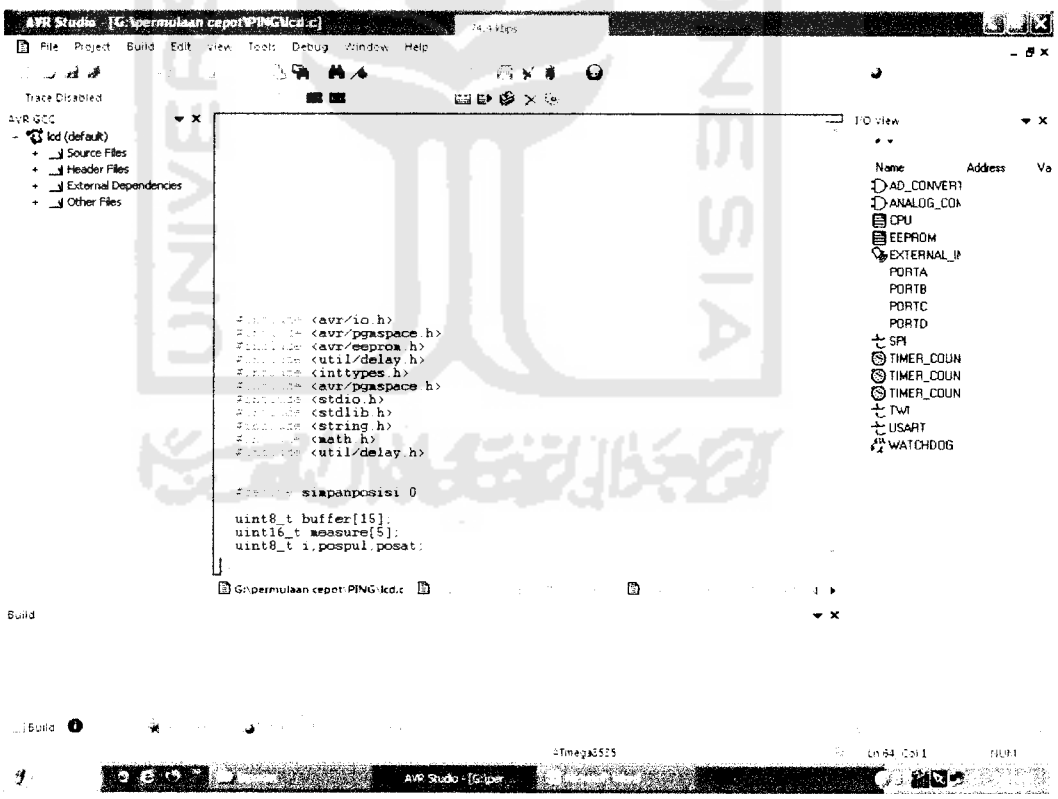


Gambar 3.7 Flowcart Program

Untuk mempermudah dan memfokuskan pembuatan *software* maka perlu adanya suatu perencanaan awal, salah satunya dengan membuat *flowcart* program.

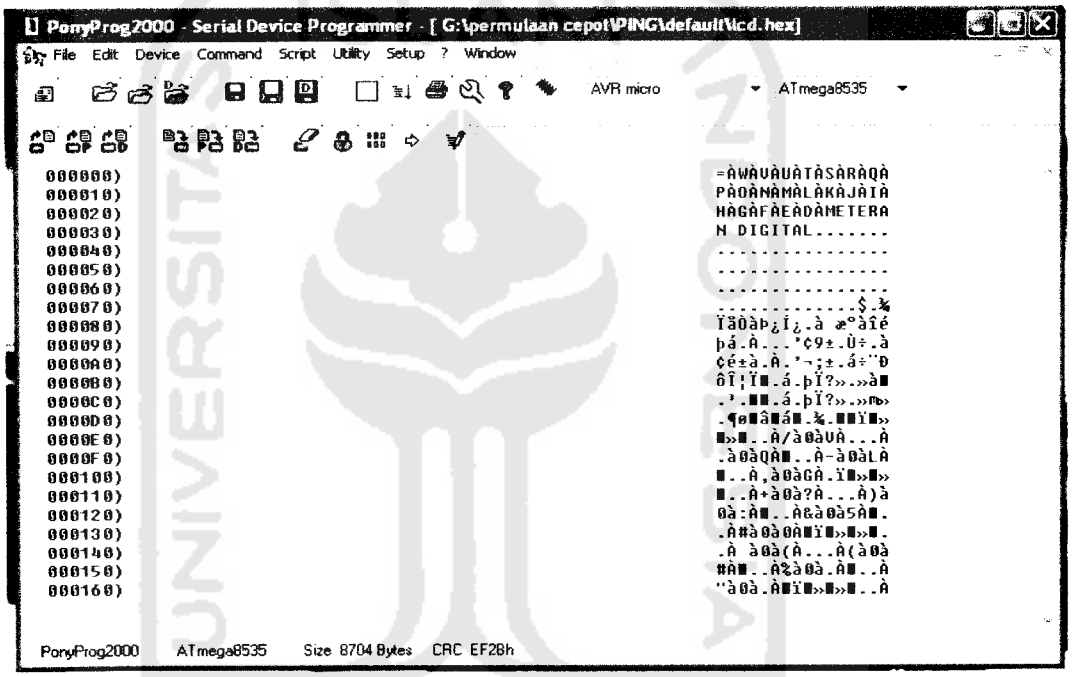
Langkah pertama adalah penginisialisasian komponen-komponen yang nantinya akan digunakan dalam *hardware*. Setelah proses inisialisasi selesai, masuk ke tahap selanjutnya yaitu proses penampilan hasil pada layar LCD berupa tulisan "METERAN DIGITAL P = 01 JRK = mm". kemudian menuju keproses selanjutnya yaitu pada tombol *keypad* yang berfungsi sebagai media untuk mengaktifkan fungsi-fungsi dari alat, dan yang terakhir merupakan tampilan hasil jarak pengukuran alat.

Sebagai media penulisan dan *compiller software* digunakan aplikasi AVR Studio. Berikut adalah tampilan dari aplikasi AVR Studio :



Gambar 3.8 Tampilan AVR Studio

Setelah *software* selesai di *compille* dan tidak terdapat *error* maka proses selanjutnya adalah men-*download software* ke mikrokontroller melalui port serial dengan menggunakan aplikasi Ponyprog. Aplikasi ini dipilih karena penggunaannya relatif lebih mudah dan praktis. Berikut tampilan dari aplikasi Ponyprog:



Gambar 3.9 Tampilan Ponyprog

3.4 Proses Pemrograman

3.4.1 Inisialisasi Program

```
#include <avr/io.h>
#include <avr/pgmspace.h>
#include <avr/eeprom.h>
#include <inttypes.h>
#include <avr/pgmspace.h>
#include <stdio.h>
#include <string.h>
#include <math.h>
#include <util/delay.h>
```

Instruksi – instruksi diatas merupakan intruksi penginisialisasian program. Diantaranya inisialisai mikrokontroller jenis AVR, *eeprom* dari mikrokontroller, inisialisasi *delay* atau waktu tunda, dan operasi matematik untuk melakukan perhitungan yang dilakukan oleh mikrokontroler. *File include* berfungsi memberitahu *compiler* agar membaca *file* yang di *include*-kan lebih daulu agar mengenali definisi–definisi yang digunakan dalam program sehingga tidak dianggap *error*.

3.4.2. Pengkalibrasian Sensor

Dengan melakukan pengkalibrasian sensor yang terpusat pada mikrokontroller, agar pembacaan sensor tidak meleset terlalu jauh dari jarak sebenarnya atau agar dapat mendapatkan hasil se akurat mungkin. Berikut cara pengkalibrasian sensor ping melalui AVR studio:

```
uint16_t baca_ping(void){
        DDRD |= (1<<INP);
        PORTD |=  (1<<INP);
        _delay_us(7);
        PORTD &=  ~(1<<INP);
        DDRD &=  ~(1<<INP);
        PORTD |=  (1<<INP);
        while (bit_is_clear(PIND,INP)) {};
    while (!bit_is_clear(PIND,INP))
     {
      count++;
     }
  jarak= count * 0.34442 / 2 * 0.734 – 0.7742;      //rumus mencari jarak
count=0;
return jarak;
}
```

DDRD | = (1<<INP) menunjukan input diberikan pada setiap *bit* di port D, kemudaian menggeser tiap *bit* ke kiri dan tiap pasangan bit di *or*-kan sehingga

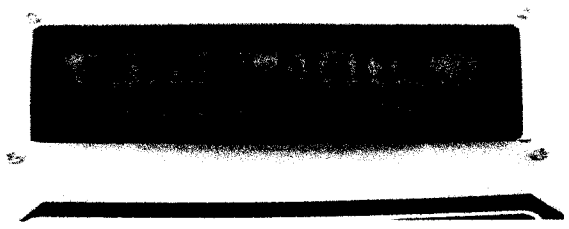
menghasilkan 0 jika keduanya berisi 0, dan 1 untuk yang lainnya. Kemudian di tunda selama 7 mikro detik, selanjutnya bit pada port D di *and*-kan dan di balik isi tiap *bit*-nya agar bernilai sama seperti awalnya. Setelah itu ditunggu sampai pin D set, langkah berikutnya adalah melakukan perhitungan jarak yang diproses di dalam mikro.

3.4.3 Tampilan Pada LCD

Untuk mengetahui hasil dari pengukuran maka hasil dari pngukuran ditampilkan pada layar LCD. Berikut program untuk menampilkan hasil pengukuran pada LCD:

```
sprintf(buffer,"P=%02i;JRK:%imm      \0",ukurke,average););
LCDGotoXY(0,1);
```

Perintah *sprint* berfungsi untuk menyimpan data *string* ke dalam memori SRAM. Dalam hal ini memori yang di tuju adalah memori yang ditunjukan variable *array buffer*. Sedangkan untuk perintah *LCDGoto* digunakan untuk melompat ke baris program tertentu yang telah diberi label.



Gambar 3.10 Tampilan Pada Layar LCD

3.4.4 Penyimpanan Data

Untuk penyimpanan data pengukuran digunakan memori *eeprom* pada ATMega8535. Memori *eeprom* dapat digunakan untuk menyimpan data pada saat *chip running* dan tidak dapat terhapus meskipun catu daya mati. Untuk menuliskan data pada *eeprom* menggunakan perintah *eeprom_write type data* dan untuk membaca data pada memori *eeprom* menggunakan perintah e*eprom_read type data.*

Contoh penulisan program pada eeprom:

```
if (keypad()==11){
                while(keypad()==11){}
                ukurke=pospul*10+posat;
                eeprom_write_byte((uint8_t*)simpanposisi,ukurke);
                pos=ukurke*2;
                average = eeprom_read_word((uint16_t*)pos);
                sprintf(buffer,"P=%02i;JRK:%imm
\0",ukurke,average);//RPM/T);
                LCDGotoXY(0,1);
        LCDstring(buffer,15);
                }
```

Untuk langkah pertama tentukan jenis data yang akan disimpan, kemudian membuat nama variabel penyimpanan agar mudah untuk mengingat. Dan untuk pengalamatannya sudah menjadi tugas dari *compiller*. Sedangkan untuk proses pembacaan memory ada yang harus diperhatikan yaitu tipe data variabel penampung harus sama dengan tipe data yang yang di baca. Setelah proses pembacaan dilakukan maka selanjutnya data akan ditampilkan pad layar LCD.