BAB III

PELAKSANAAN TEKNIS KERJA PRAKTEK

UNIT PELAKSANA TEKNIS PERPUSTAKAAN
UNIKOM

3.1 JADWAL KERJA PRAKTEK

Berikut ini jadwal kerja praktek yang Penulis lakukan:

Minggu 1 : Mengumpulkan informasi mengenai hardware dan software yang dibutuhkan serta informasi mengenai mikrokontroller AT90S1200.

Minggu 2 : Ikut dalam pembuatan dan perbaikan rangkaian AT90S1200.

Minggu 3 : Mempelajari mikrokontroller AT90S1200.

Minggu 4 : Perumusan dan pembuatan program serta mempelajari mikrokontroller AT90S1200.

Minggu 5 : Pemrograman dan implementasi program pada mikrokontroller AT90S1200.

Minggu 6 : Penyelesaian akhir program.

3.2 DESKRIPSI PEKERJAAN

Adapun pekerjaan yang Penulis lakukan selama kerja praktek adalah sebagai berikut :

1. Mengumpulkan informasi mengenai hardware dan software yang dibutuhkan untuk pemrograman mikrokontroller AT90S1200 melalui internet.
2. Ikut langsung dalam pembuatan rangkaian AT90S1200 seperti melakukan penyolderan.

3. Melakukan presentasi setiap hari jumat mengenai kegiatan yang dilakukan selama seminggu.
4. Memprogram mikrokontroller AT90S1200.

3.3 HARDWARE DAN SOFTWARE YANG DIGUNAKAN

Adapun hardware dan software yang Penulis gunakan selama kerja praktek adalah sebagai berikut :

a) Hardware:
   1. Satu perangkat komputer online (terkoneksi ke internet)
   2. Multimeter, solder, board, serta komponen-komponen elektronik pembentuk rangkaian AT90S1200
   3. ATMEL AVR ISP dongle sebagai antarmuka ISP untuk menghubungkan AT90S1200 ke komputer.

b) Software:
   1. Sistem operasi Windows 98
   2. UMPS Revision 1.87 dan Ponyprog V117h
   3. Microsoft Powerpoint
   4. Microsoft Internet Explorer

3.4 LANDASAN TEORI

3.4.1 Mikrokontroller AT90S1200

Sebagai salah satu keluarga mikrokontroller AVR, AT90S1200 memiliki karakteristik sebagai berikut:

a) Memiliki arsitektur RISC

b) 89 buah perintah

c) 32 general purpose working register masing-masing 8 bit

d) 1K Byte in-system programmable Flash

e) 64 Byte in-system programmable EEPROM

f) Fungsi lock untuk keamanan program dan EEPROM

g) Satu 8-bit Timer/Counter

h) Watchdog Timer

i) Antarmuka serial SPI

j) Interrupt eksternal dan internal

k) 15 pin I/O

l) 20 pin PDIP dan SOIC

Informasi lebih detil mengenai mikrokontroller AT90S1200 dapat dilihat pada bagian lampiran.

3.4.2 Perangkat Keras Pemrograman Mikrokontroller AT90S1200

Untuk mempelajari pemrograman mikrokontroller AT90S1200 diperlukan sejumlah perangkat keras untuk menguji program yang sudah dibuat. Adapun perangkat keras yang diperlukan adalah antarmuka-ISP,

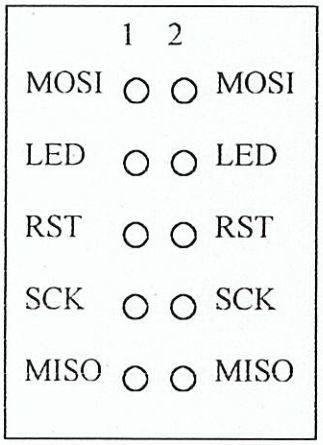
antarmuka yang menghubungkan rangkaian AT90S1200 ke komputer, dan rangkaian AT90S1200.

3.4.2.1 Antarmuka-ISP Keluarga Prosesor AVR

Antarmuka ISP keluarga prosesor AVR menggunakan mode pemrograman serial. Pada mikrokontroller keluarga AVR terdapat tiga antarmuka build-in yang bekerja secara serial:

1. SCK : sumber clock untuk mengirim/mengambil isi dari memori,
2. MOSI : untuk mengirim data ke mikrokontroller,
3. MISO : untuk menerima data dari mikrokontroller.

Antarmuka-ISP yang penulis gunakan adalah ISP10PIN. Antarmuka ISP ini berada pada rangkaian AT90S1200 dan dihubungkan dengan antarmuka rangkaian AT90S1200.



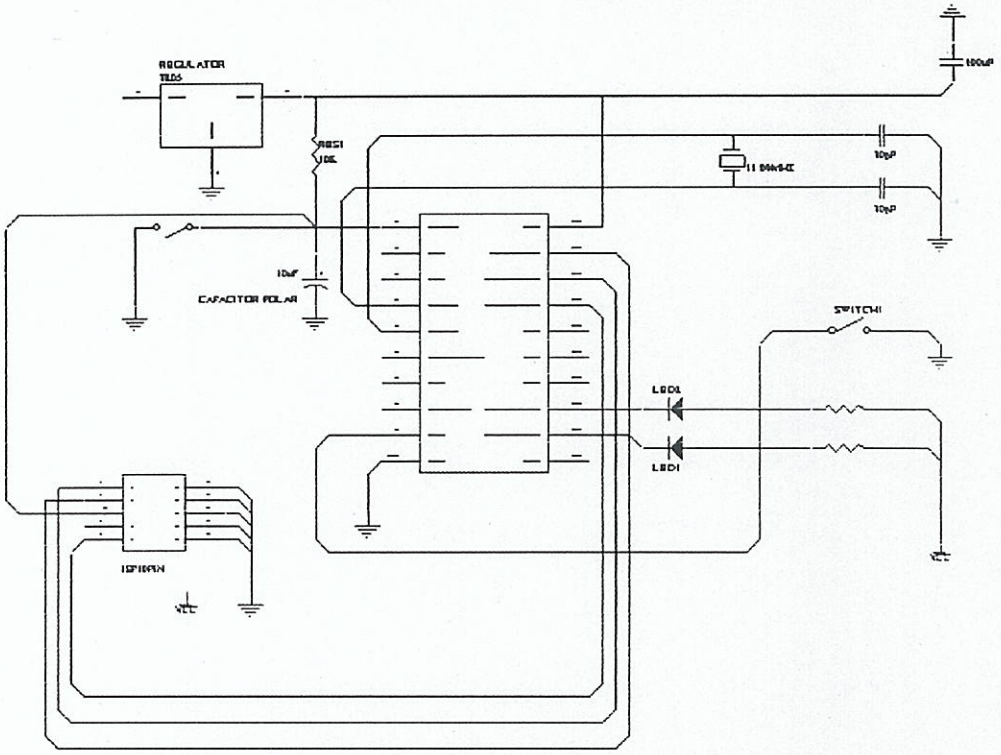
Gambar 1. ISP10PIN

3.4.2.2 Antarmuka Rangkaian AT90S1200 Ke PC

Antarmuka rangkaian AT90S1200 ke PC yang Penulis gunakan adalah rangkaian dongle yang dihubungkan ke komputer melalui port printer atau port parallel. Antarmuka ini dihubungkan ke rangkaian AT90S1200 melalui antarmuka ISP ISP10PIN. Informasi mengenai rangkaian dongle dapat dilihat di bagian lampiran.

3.4.2.3 Rangkaian AT90S1200

Rangkaian AT90S1200 merupakan rangkaian utama untuk pemrograman mikrokontroller AT90S1200. Pada rangkaian ini terdapat dua buah LED dan satu buah tombol. Masing-masing LED dihubungkan ke port B yaitu PB0 dan PB7, sedangkan tombol dihubungkan ke port D yaitu PD4. Pada keadaan awal, kedua LED dalam keadaan tidak menyala dan PB0 dan PB7 difungsikan sebagai output port dan memiliki nilai berlogika "1". Sedangkan tombol dalam keadaan terbuka atau tidak ditekan dan memiliki nilai berlogika "1". Ilustrasi rangkaian AT90S1200 dapat dilihat pada gambar 2 berikut ini.



Gambar 2. Rangkaian AT90S1200

3.4.3 Tool Untuk Pemrograman Assembly AT90S1200

Berikut ini empat program dasar yang dibutuhkan dalam pemrograman mikrokontroller AT90S1200:

1. Editor,
2. Program asembler,
3. Chip programming interface, dan
4. Simulator.

Untuk melakukan keempat fungsi di atas, Penulis menggunakan perangkat lunak UMPS dan Ponyprog. UMPS digunakan sebagai editor, asembler dan simulator. Sedangkan Ponyprog digunakan sebagai antarmuka

untuk memprogram mikrokontroller AT90S1200. UMPS bisa berfungsi sebagai antarmuka untuk memprogram mikrokontroller AT90S1200, tetapi dalam kerja praktek tidak dapat difungsikan, karena UMPS yang penulis gunakan merupakan versi demo.

3.4.3.1 UMPS Revision 1.87

UMPS yang penulis gunakan adalah versi demo 1.87 sehingga memiliki keterbatasan sebagai berikut:

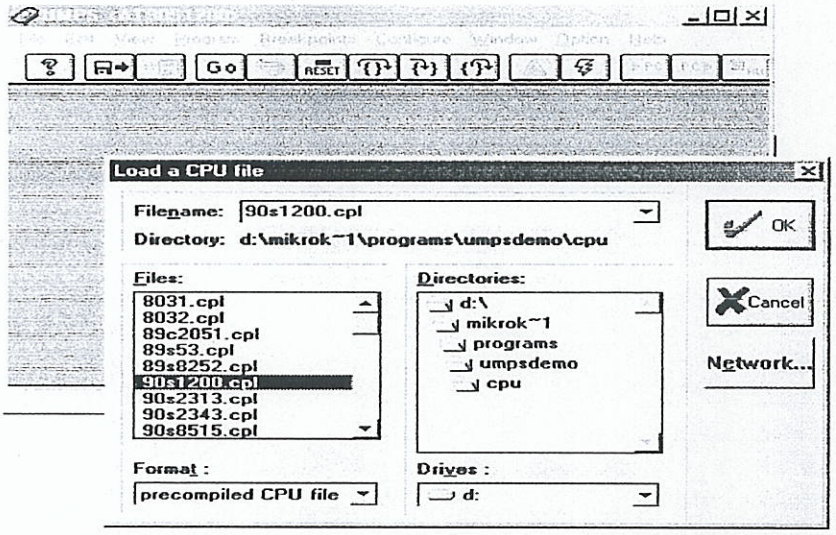
a) Waktu simulasi terbatas sampai 800000 cycle
b) Kapasitas ROM/RAM yang dapat digunakan sebesar 256 byte
c) Tidak dapat digunakan sebagai chip programming interface
d) Dokumentasi CPU tidak lengkap
e) File library CPU tidak lengkap
f) Batas waktu pemakaian selama tiga bulan

UMPS dapat menterjemahkan dan mengeksekusi instruksi-instruksi sejumlah mikrokontroller. Sehingga untuk memulai pemrograman harus dipilih terlebih dahulu jenis CPU yang akan digunakan yang terdapat pada file CPL (.CPU). Selain itu UMPS juga dapat mensimulasikan sejumlah mikroprosesor dan resource dengan menghubungkan environment virtual atau resource tersebut ke pin-pin CPU atau mikrokontroller. Tiap resource ditampilkan pada window yang berbeda dengan editor teks program. UMPS juga dapat

digunakan untuk menampilkan dan mengubah isi register, ROM, RAM.

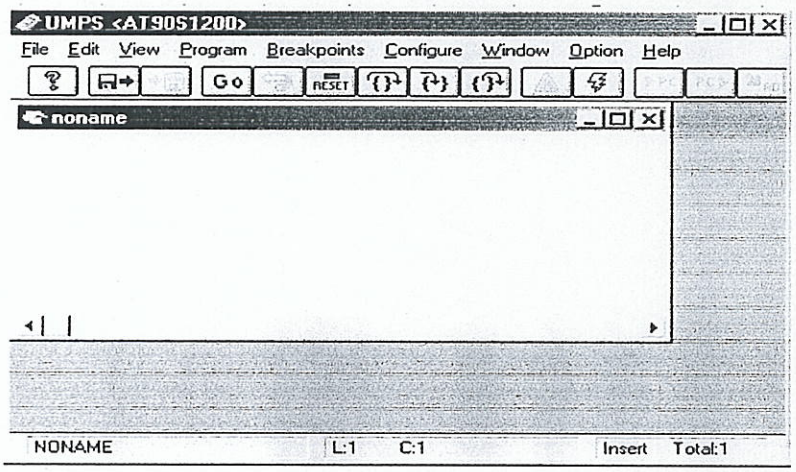
Untuk melakukan pemrograman, dilakukan langkah-langkah sebagai berikut setelah UMPS dijalankan:

1. Pilih tipe CPU dengan ekstensi (.CPL): **Configure Load CPU**



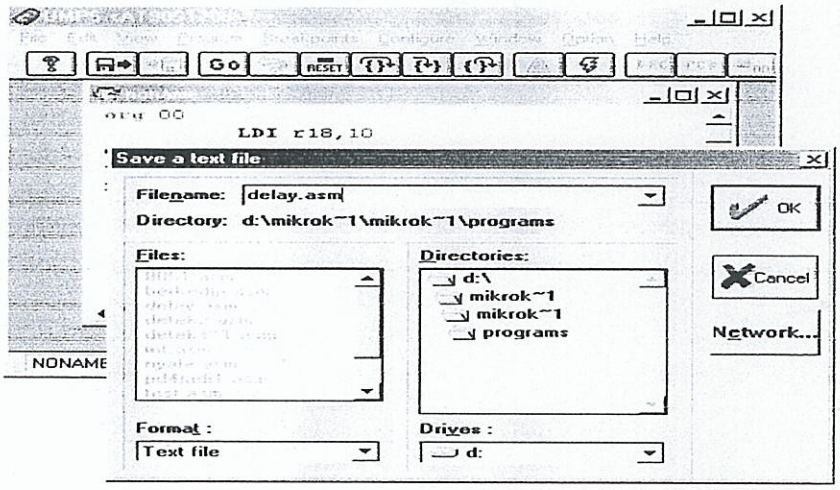
pilih **90s1200.cpl** untuk mikrokontroller AT90S1200, kemudian tekan tombol OK.

2. Buka teks editor baru: **File New**

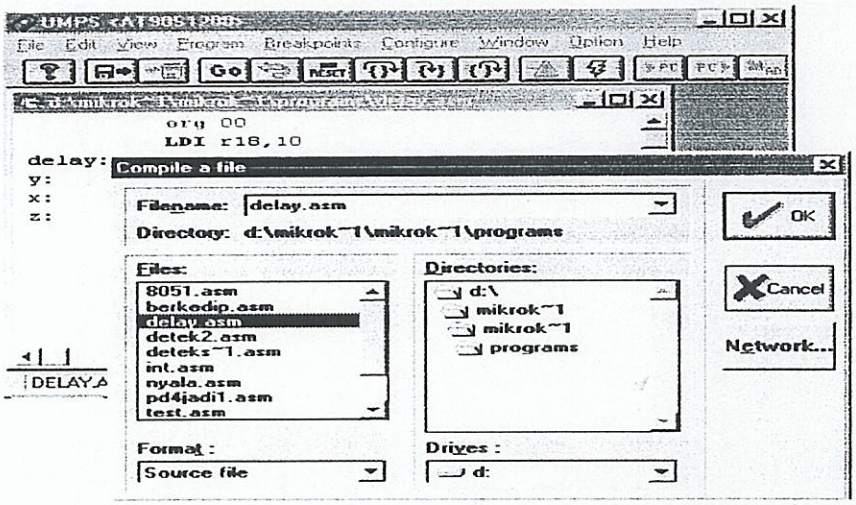


Setelah kedua langkah di atas dilakukan, pemrograman sudah dapat dilakukan. Tetapi untuk pemrograman yang memerlukan operasi timer maupun operasi transfer data perlu dilakukan konfigurasi CPU yang digunakan. Disini penulis tidak menjelaskan konfigurasi, karena program yang penulis gunakan tidak memerlukan operasi timer dan operasi serial. Kemudian, ketikkan program pada Editor teks. UMPS akan membedakan warna instruksi dari assembler dengan yang bukan instruksi.

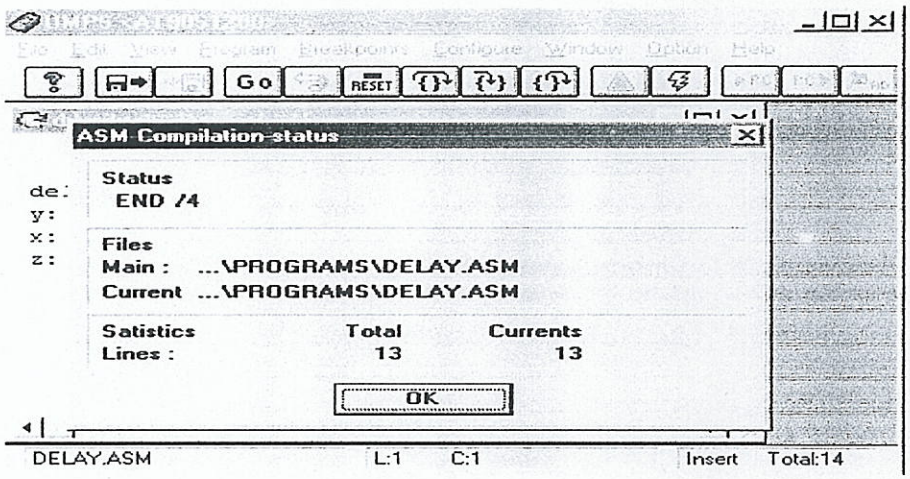
3. Simpan program yang baru dibuat dengan ekstensi (.asm): **File Save**



4. Kompile program yang baru disimpan: **Program Compile**



Pada pilihan files, pilih nama file program kemudian tekan tombol OK. UMPS akan menampilkan hasil kompile program.



Hasil terjemahan program di atas menunjukkan tidak ada kesalahan program. Jika terjadi kesalahan, UMPS akan memberitahu letak kesalahan program.

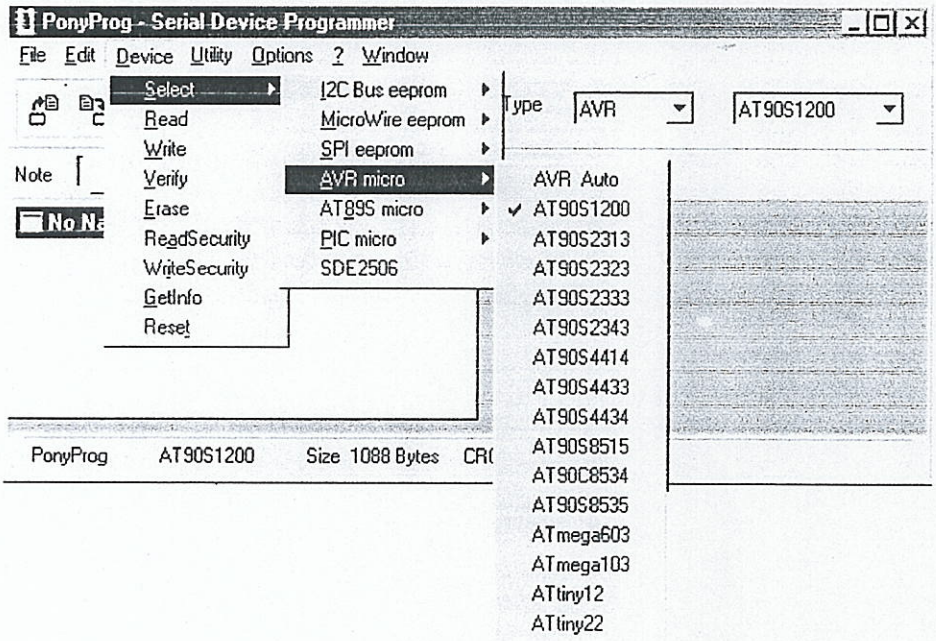
Setelah program berjalan dengan benar, selanjutnya program diisikan ke mikrokontroller AT90S1200 menggunakan Ponyprog. Karena UMPS yang penulis gunakan versi demo, perintah *nop* (no

instruction) harus ditambahkan pada akhir program kurang lebih sebanyak jumlah baris instruksi program yang dibuat.

3.4.3.2 Ponyprog V117H

Untuk mengisi program ke memori mikrokontroller menggunakan Ponyprog dilakukan langkah sebagai berikut:

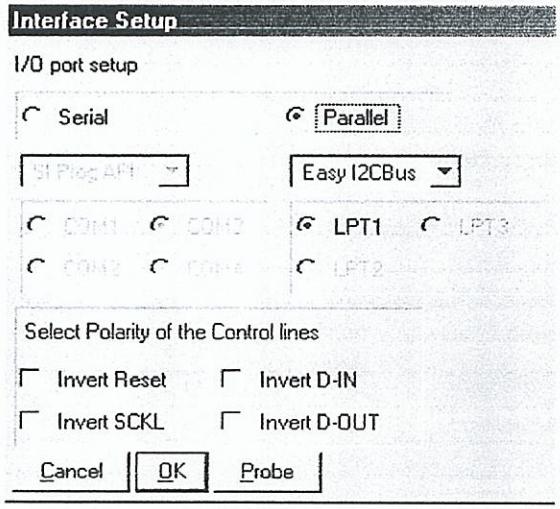
1. Pilih tipe mikrokontroller : **Device Select AVR micro AT90S1200**



Pemilihan tipe mikrokontroller ini harus dilakukan terlebih dahulu sebelum melakukan operasi baca dan tulis mikrokontroller. Pilihan "Auto XXX" berarti, tipe mikrokontroller ditentukan oleh program selama operasi baca dan tulis berdasarkan hasil deteksi Ponyprog. Ada beberapa keluarga mikrokontroller yang berbeda yang dapat dipilih: I²C Bus eeproms dengan alamat 8 bit, I²C Bus eeproms

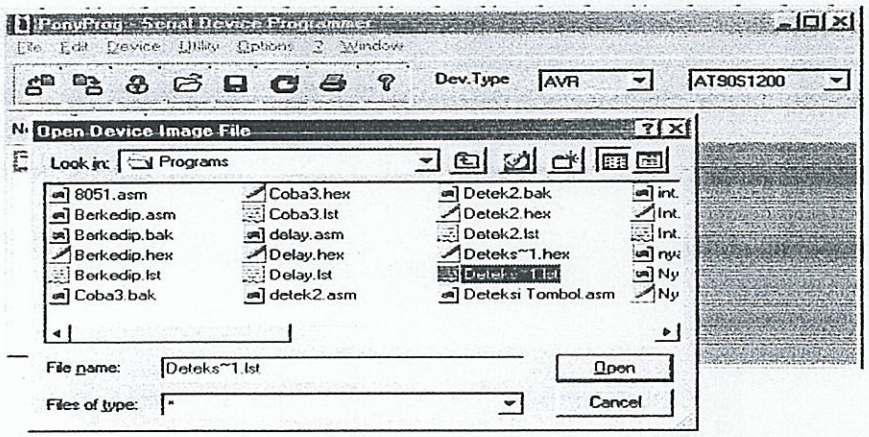
dengan alamat 16 bit, Microwire eeproms, SPI eeproms, mikrokontroller AVR dan mikrokontroller PIC. Keluarga mikrokontroller ini dapat dipilih pada tool bar ataupun langsung pada device type pada menu. Pada pilihan ini penulis menggunakan Device Type AVR dan AT90S1200.

2. Buka dialog box untuk memilih tipe antarmuka dan nomor port tempat perangkat keras dihubungkan: **Options Setup**.



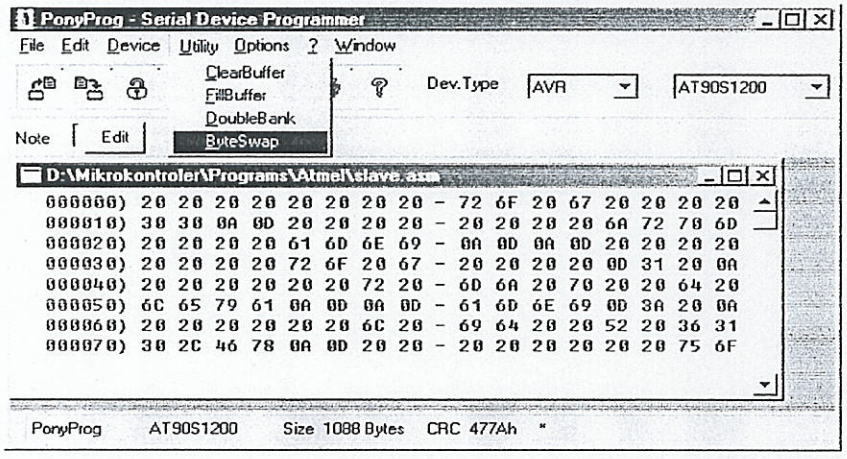
Pada Interface Setup dipilih Parallel, Easy 12Cbus, LPT1. Kemudian klik tombol OK.

3. Buka file yang telah dibuat dengan UMPS: **File Open**



Ponyprog dapat membuka empat tipe format file yang berbeda: e2p, intel hex, motorola S-record dan raw biner. Bila file yang terpilih bukan tipe e2p, Ponyprog akan membuka file tersebut sebagai intel hex; dan bila operasi buka ini gagal, maka file akan dibuka sebagai biner. Pada pilihan File of type pilih tipe *.*hex.* maka akan ditampilkan semua file bertipe *.hex* kemudian pilih nama file program yang telah dibuat dengan UMPS.

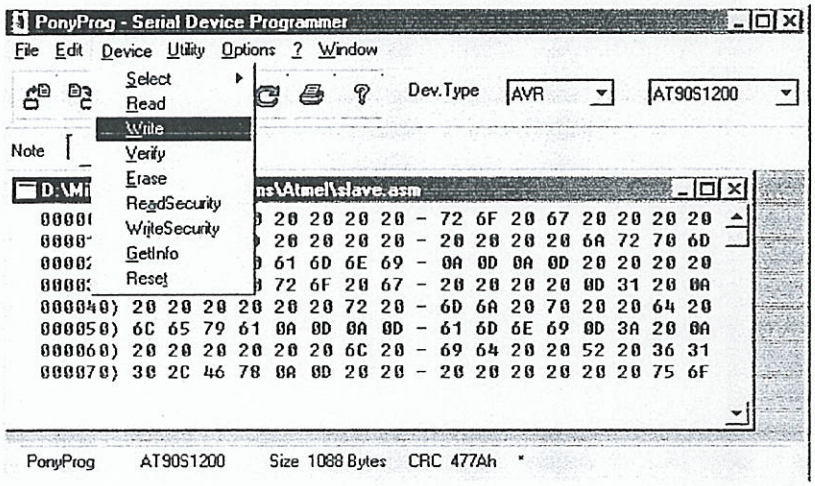
4. Lakukan penukaran byte: **Utility ByteSwap**



Operasi ini menukarkan tiap satu word isi buffer. Perintah ini berfungsi untuk mengubah representasi little endiand menjadi big

endiand dan sebaliknya. Beberapa mikrokontroller menggunakan 16 bit word, sehingga dapat direpresentasikan dengan dua cara tergantung kebutuhan. Sebagai contoh angka 1234 hex (4660 desimal), little endiand adalah 34 – 12, sedangkan big endiand adalah 12 –34.

5. Tulis program ke mikrokontroller: **Device Write**



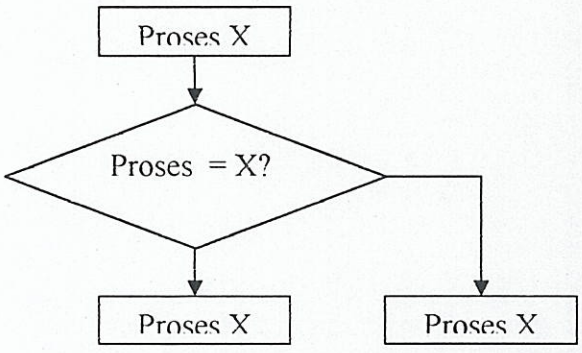
operasi ini membutuhkan sedikit waktu untuk diselesaikan, dan selama proses, dialog box akan menunjukkan perkembangan operasi. Untuk menghentikan operasi tulis tekan tombol "Abort". Setelah operasi tulis selesai, dialog box akan menunjukkan hasil operasi tulis.

3.4.4 Teknik Merancang Program

Dalam merancang sebuah program, seringkali para pembuat program menganggap sebuah program rancangannya sudah selesai jika program tersebut sudah berjalan dengan baik. Namun dalam aplikasinya, seringkali

sebuah program yang dirancang, perlu ditelusuri lagi untuk keperluan pengembangan lebih lanjut maupun digabungkan dengan program yang lain dalam suatu tim. Pihak lain dalam tim seringkali membutuhkan cara kerja dari program rancangan tersebut untuk mengembangkan atau menggabungkan lebih lanjut program rancangan yang dianggap sudah selesai. Untuk itu, sebuah program yang baik tidak hanya dapat berjalan dengan baik saja, namun program tersebut harus dapat ditelusuri kembali dengan mudah. Bahkan pembuat program sendiri seringkali menemui kesulitan dalam menelusuri program tersebut jika program tersebut tidak dirancang dengan struktur yang teratur.

Teknik merancang sebuah program yang mempunyai struktur baik, biasanya diawali dengan pembuatan *flowchart* (diagram alir), pseudo code dan program assembly yang dirancang. Diagram alir digunakan untuk menggambarkan terlebih dahulu mengenai "apa yang harus dikerjakan" sebelum mulai merancang program seperti contoh pada gambar berikut:



Gambar 3. Diagram Alir

Pseudo Code biasanya digambarkan dalam bahasa C atau Pascal yang lebih manusiawi daripada bahasa assembly seperti pada contoh berikut dimana pseudo code digambarkan dalam bahasa Pascal:

Contoh:

```
Proses;
  If Hasil = X then
  Begin
        ProsesX
  End
  Else
  Begin
        ProsesY
  End;
```

3.4.4.1 Struktur Rancangan Program

Struktur dari sebuah program biasanya terdiri atas:

a) *Statement* (pernyataan)

b) *Loops* (putaran)

c) *Choice* (pilihan)

1. Pernyataan

Pernyataan menggambarkan dasar mekanisme "melakukan sesuatu", seperti pemberian nilai suatu variabel atau pemanggilan subrutin berikut:

Contoh:

```
Ldi     R16,0xF       ; memberi nilai pada register R16
Rcall   SerialOut     ; memanggil subrutin serial out
```

2. Putaran

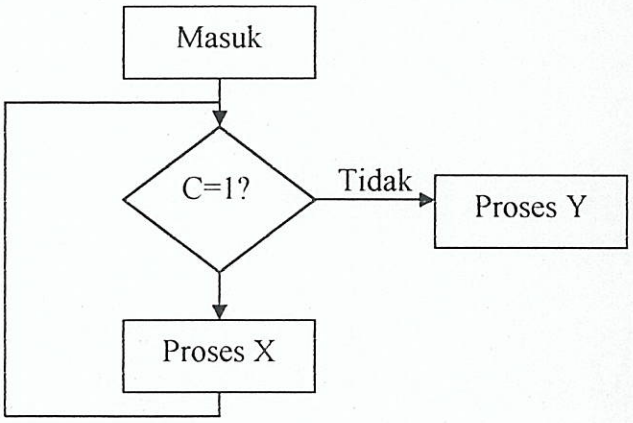
Struktur putaran digunakan untuk melakukan sebuah operasi yang dilakukan berulang-ulang. Berbagai macam struktur putaran biasanya dibentuk oleh *While/Do* atau *Repeat/Until*.

a. While/Do

Pernyataan while/do atau jika kondisi tertentu melakukan sesuatu hingga kondisi tersebut berubah merupakan pernyataan yang melakukan suatu proses tertentu selama kondisi yang diperiksa benar. Proses dilakukan berulang-ulang hingga kondisi yang diperiksa salah. Bentuk pernyataan ini adalah:

While (kondisi) do
(pernyataan)

Contoh:



Gambar 4. Diagram Alir Pernyataan While/Do

Pseudo code dari diagram alir di atas adalah:

While (c = =1) do

ProsesX

Program assembly:

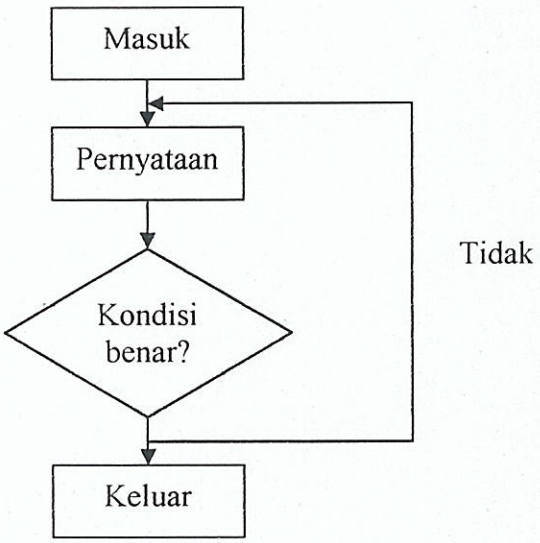
```
Masuk: Brcc   Keluar  ; jika flag carry bernilai 0 lompat ke  label
                      ;Keluar
Rcall  ProsesX ; jika flag carry bernilai 1 lakukan proses X
Rjmp   Masuk   ; kembali ke label masuk
```

b. Repeat/Until

Hampir sama dengan while/do, hanya pada pernyataan while/do kondisi diperiksa terlebih dahulu. Pada pernyataan ini, kondisi diperiksa setelah proses berjalan satu kali. Bentuk pseudo code dari repeat/until adalah sebagai berikut:

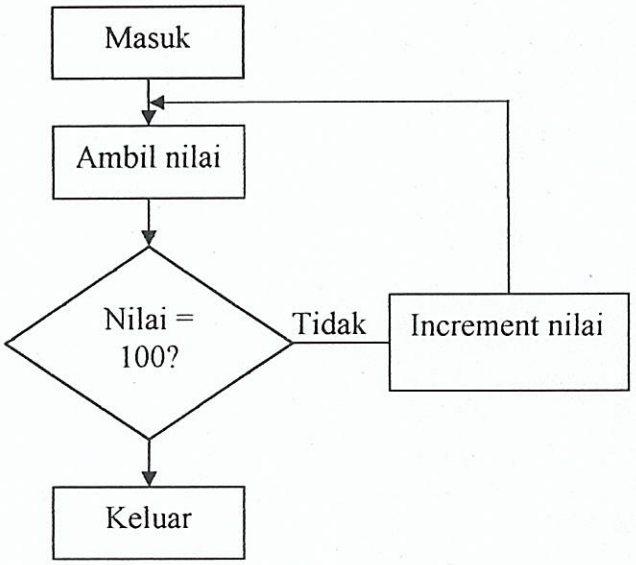
```
Repeat    [pernyataan]
  Until   [kondisi]
```

Diagram alir:



Gambar 5. Diagram Alir Pernyataan Repeat/Until

Contoh:



Gambar 6. Diagram Alir Contoh Repeat/Until

Pseudo Code:

Repeat
    [ambil nilai]
    [increment nilai]
Until [nilai = = 100]

Program assembly:

```
    Ldi     R17,0x00
Pernyataan:
    Ldi     R16,R17
    Inc     R17
    Cp      R16,R17
    Breq    Pernyataan
Keluar:
```

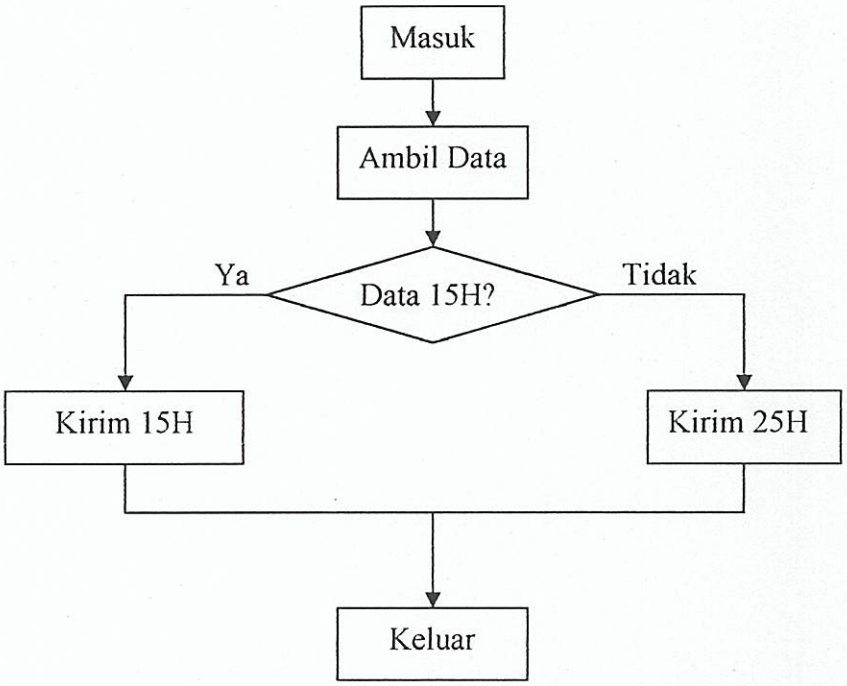
3. Pilihan

Dua buah pernyataan yang seringkali digunakan dalam struktur ini adalah IF/THEN/ELSE dan CASE.

1. IF/THEN/ELSE

Pernyataan ini digunakan jika dua buah kondisi harus dipilih. Bagian ELSE bersifat pilihan, dapat digunakan dan dapat juga tidak digunakan.

Contoh:



Gambar 7. Diagram Alir Contoh If/Then/Else

Pseudo Code:

```
If  [Data = 15H]
    Then [Kirim 15H]
    Else  [Kirim 25H]
```

Program assembly:

```
Ldi     R16,0xF
Rcall   AmbilData
Cp      R16,R17      ;R17 berisi data yang baru saja diambil
```

```
    Breq   Pernyataan1
    Rjmp   Pernyataan2
Peryataan2:
           Rcall   Kirim15H
           Rjmp    Keluar
Pernyataan1:
           Rcall   Kirim25H
           Rjmp    Keluar
Keluar:
```

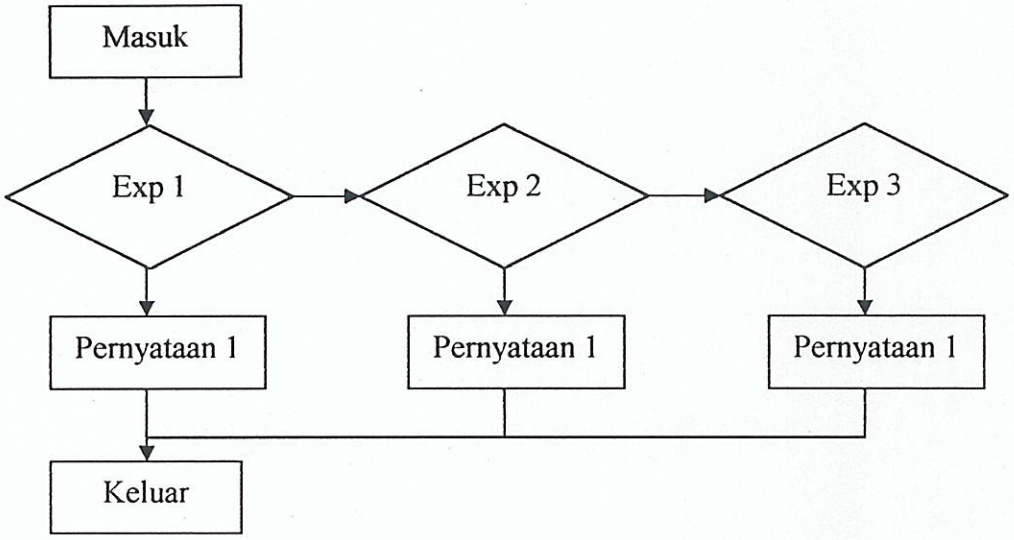
2. CASE

Jika terdapat berbagai macam kondisi yang harus dipilih, pernyataan CASE lebih efektif digunakan. Bentuk dari pernyataan case adalah sebagai berikut:

Pseudo Code:

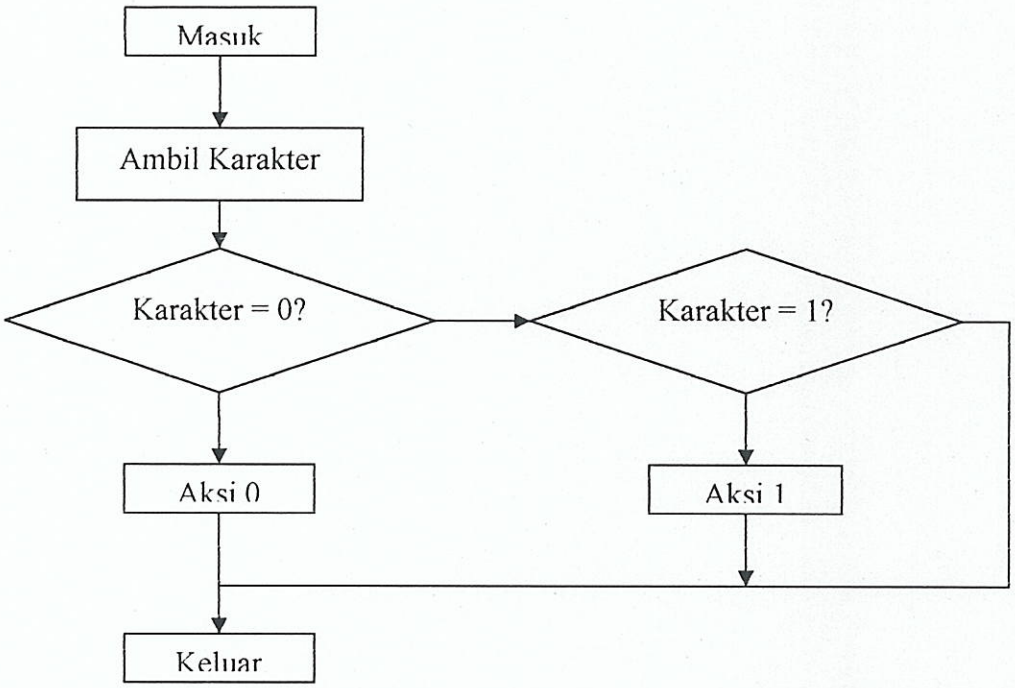
```
Case [ekspresi] of
      0:  [pernyataan 0]
      1:  [pernyataan 1]
          ……..
          ……..
      n:  [pernyataan n]
End_Case
```

Diagram alir:



Gambar 8. Diagram Alir Pernyataan CASE

Contoh:



Gambar 9. Diagram Alir Contoh CASE

Pseudo Code:

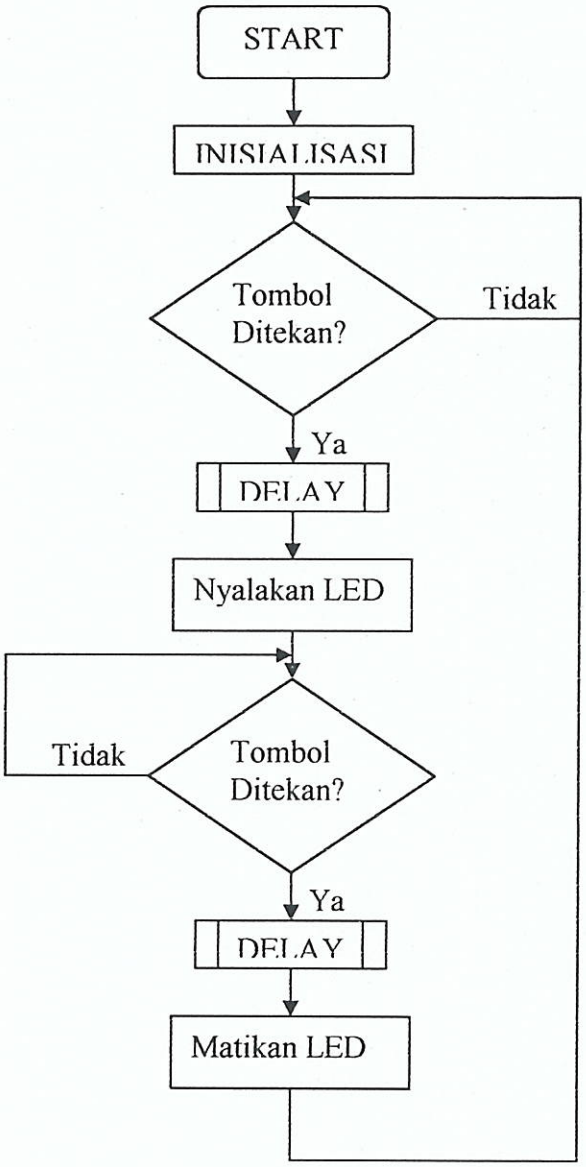
```
    [ambil karakter]
CASE [karakter] of
    '0':    [aksi 0]
    '1':    [aksi 1]
End_Case
```

Program assembly:

```
    Rcall   AmbilKarakter
    Cpi     R16,0x00
    Breq    Aksi0
    Cpi     R16,0x01
    Breq    Aksi1
    Aksi0: ... ... ... ... ...
                    Rjmp   Keluar
    Aksi1: ... ... ... ... ...
                    Rjmp   Keluar
    Keluar:
```

3.5 APLIKASI PROGRAM PADA LED DAN TOMBOL

3.5.1 Flowchart Program



Gambar 10. Flowchart Program

Penjelasan flowchart:

1. INISIALISASI

Inisialisasi merupakan bagian program untuk melakukan:

a) Mendefinisikan PORT D menjadi port input

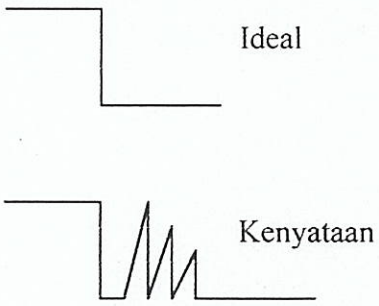
b) Membuat keadaan awal tombol tidak ditekan

c) Mendefinisikan PORT B menjadi port output

d) Membuat keadaan awal LED tidak menyala

2. DELAY

DELAY berfungsi untuk menyebabkan waktu tunda pada saat perpindahan LED dari keadaan tidak menyala menjadi menyala dan sebaliknya. DELAY juga berfungsi untuk mengatasi masalah debounching.



Gambar 11. Masalah Debounching

Subprogram DELAY berisi delay untuk memastikan bahwa tombol benar-benar telah mencapai keadaan *steady state*.

3 5.2 Listing Program

```
        Org   00H
        Rjmp  Inisial

        Org   30H
Inisial:
        ; PORT D menjadi input port
        Ldi   R16,0x00
        Out   DDRD,R16
        ; Membuat keadaan awal switch tidak terdeteksi
        Ldi   R16,0xFF
        Out   PORTD,R16
        ; PORT B menjadi output port
        Ldi   R16,0xFF
        Out   DDRB,R16
        ; MATIKAN LED
        Out   PORTB,R16

Main:
        Sbis   PIND,5   ; Branch jika switch pada pin 8 ditekan
        Rjmp   Lampu
        Rjmp   Main
Lampu:
        Rcall  delay
        Ldi    R16,0xFC
        Out    PORTB,R16  ; Nyalakan LED
loop:
        Sbis   PIND,5
        Rjmp   LED_OFF
        Rjmp   loop
LED_OFF:
        Rcall  delay
        Ldi    R16,0xFF
        Out    PORTB,R16  ; Matikan LED
        Rjmp   Main
        Ret

delay:
        Ldi    R20,50
y:      Ldi    R21,50
x:      Ldi    R22,10
z:      Dec    R22
        Brne   z
        Dec    R21
        Brne   x
```

```
Dec   R20
Brne  y
Ret
nop
nop
nop
```


UNIVERSITAS KOMPUTER INDONESIA
UNIKOM