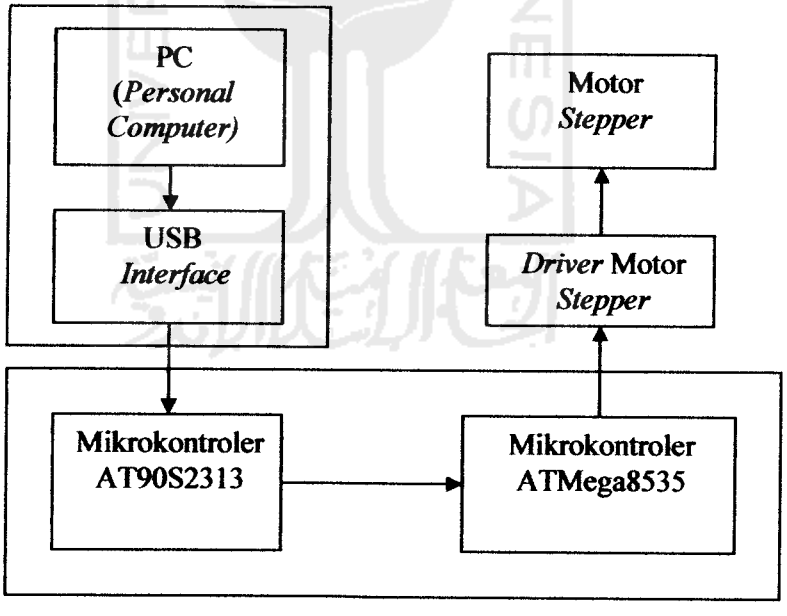
BAB III

PERANCANGAN SISTEM

Dalam bab III ini akan dibahas mengenai perancangan sistem yang di dalamnya terdapat perancangan rangkaian elektronika dan sistem pengendalian motor *stepper* dengan komunikasi USB berbasis Delphi berdasar pada teori-teori yang telah di bahas sebelumnya. Perancangan sistem tersebut dapat digambarkan melalui diagram blok seperti terlihat pada gambar 3.1

3.1 Diagram Blok Sistem



Gambar 3.1 Diagram blok sistem

Adapun penjelasan dari diagram blok tersebut adalah sebagai berikut:

PC *(Personal Computer)*

Blok ini berfungsi sebagai *interface* (antarmuka) antara user dengan komputer/PC sebagai pengendali. Pengendali yang dimaksud adalah dengan menggunakan bahasa Delphi, data-data yang diperlukan diproses, kemudian komputer/PC akan mengirimkan data yang diperoleh ke AT90S2313 yang berfungsi sebagai *pull-up* data pada mikrokontroler ATMega8535. Pengiriman data ini ditransfer melalui komunikasi USB, yang sintaksnya telah disediakan pada Delphi.

USB *(Universal Serial Bus)*

Sebuah bus I/O (*input/output*) yang dapat mentransfer data hingga 12 megabit per detik. Digunakan sebagai penghubung antara PC dengan Mikrokontroler AT90S2313 yang akan meneruskan data-data yang telah diproses ke mikrokontroler ATMega8535.

Mikrokontroler AT90S2313

Mikrokontroler AT90S2313 memilik instruksi yang dihimpun dalam kode 16-bit dan sebagian besar instruksinya dieksekusi dalam 1 (satu) siklus *clock.* berfungsi sebagai *pull-up* data-data instruksi pada ATMega8535.

Mikrokontroler ATMega8535

Pada blok ini mengendalikan arah gerakan motor *stepper* ke kiri dan ke kanan dan menentukan besarnya derajat arah yang diinginkan. ATMega8535 sendiri berfungsi untuk mengubah data-data dari komputer (PC) agar dapat dibaca oleh motor *stepper.*

Perancangan Perangkat Lunak

Perangkat lunak sisi PC yaitu program Delphi 7. *Source* program dimodifikasi untuk 3 perintah dasar yaitu memutar ke kiri, memutar ke kanan, dan motor berhenti. Serta 5 perintah derajat sudut yaitu 10, 30, 60, 90, 180 derajat.

Driver Motor *Stepper*

Pengarah motor *Stepper* menerima isyarat *low-level* dari pengindeks atau sistem pengendalinya dan mengkonversi ke dalam energi listrik untuk menjalankan motor *stepper* tersebut.

Motor *Stepper*

Motor *stepper* ini adalah sebagai objek, yang mana arah putaran motor atau besarnya derajat arah yang diinginkan diatur. Kecepatan motor *stepper* dapat diatur dengan menentukan *step* yang diinginkan.

3.2 Perancangan Perangkat Keras

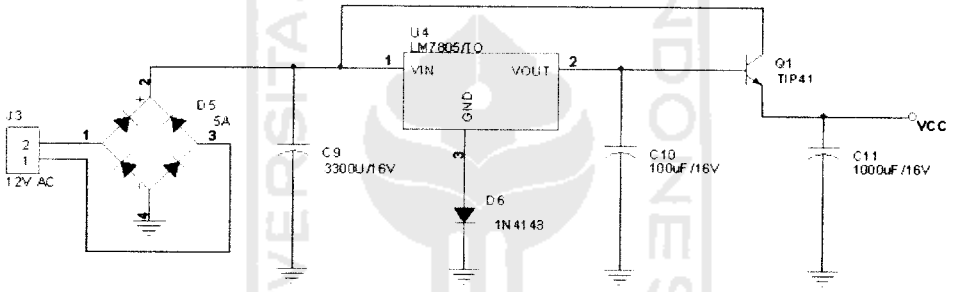
Pada penelitian ini perangkat keras yang digunakan berupa motor *stepper*, yang didukung oleh perlengkapan-perlengkapan yang lain seperti : transformator atau catu daya, sistem mikrokontroler AT90S2313, mikrokontroler ATMega8535, dan rangkaian regulator yang berfungsi sebagai pembangkit tegangan pada motor *stepper*.

3.2.1 Catu Daya

Catu daya merupakan bagian yang sangat penting pada rangkaian karena tanpa catu daya alat ini tidak dapat bekerja. Motor *stepper* memerlukan catu daya yang dapat memberikan tegangan sebesar 5 V.

Mempertahankan suatu level tegangan yang konstan sangat diperlukan dalam rangkaian catu daya ini, dengan demikian rangkaian catu daya pada tugas akhir ini menggunakan rangkaian regulator tegangan (*voltage regulator)* yang mengandung sejumlah rangkaian untuk tegangan referensi, alat pengontrol, penguat komparator, dan pelindung tegangan berlebih (*overload protection*).

Pada alat ini pelindung tegangan berlebih digunakan regulator jenis positif regulator dengan tipe LM7805 untuk penstabil tegangan 5VDC



Gambar 3.2 Rangkaian regulator tegangan

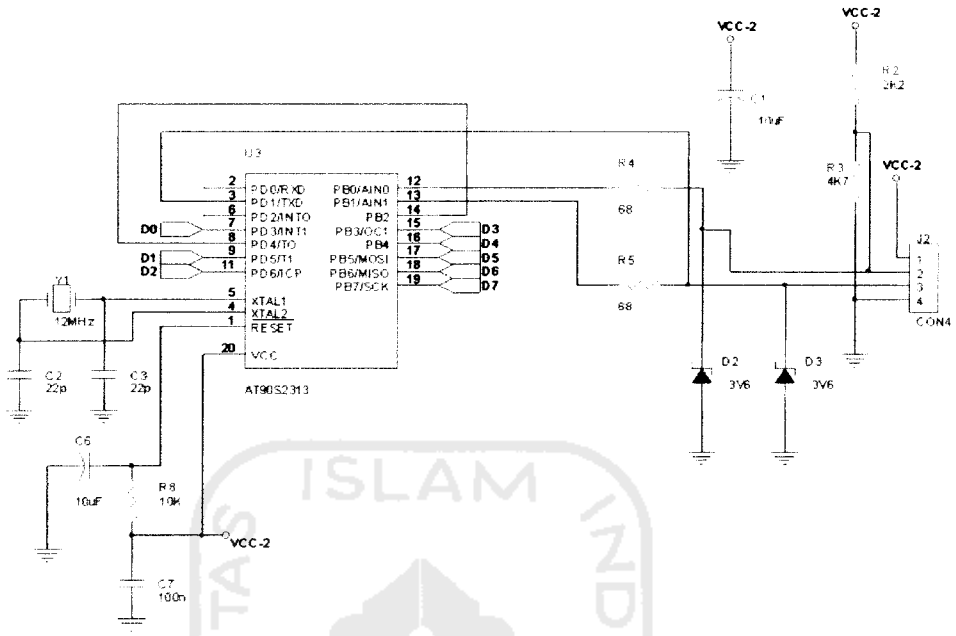
Dalam perancangan ini dipergunakan beberapa komponen seperti pada Gambar 3.2 dengan penjelasan sebagai berikut :

1. Dioda *Bridge*, merupakan penyearah yang mengubah arus AC dari transformator menjadi arus DC, namun sinyal keluarannya masih mengandung riak atau *ripple*.
2. Kapasitor elektrolit C9 dengan nilai 3300uF berfungsi untuk memperhalus dan menghilangkan riak keluaran dari penyearah dioda *bridge*.

3. IC LM7805 merupakan regulator tegangan yang berfungsi untuk menurunkan tegangan dari 12V menjadi 5V, serta menyetabilkan keluarannya.
4. Dioda 1N4148 berfungsi untuk menjaga keluaran LM7805 agar berada di 5V, karena tanpa dioda ini keluaran LM7805 hanyalah sebesar 4,8V.
5. Kapasitor elektrolit 100uF berfungsi untuk melakukan filtrasi keluaran dari LM7805 agar lebih halus sebelum diumpankan ke penguat arus TIP 41.
6. TIP 41 merupakan transistor penguat arus untuk memperbesar arus keluaran dari catu daya, hal ini diperlukan untuk menyuplai mikrokontroler serta motor *stepper* yang membutuhkan arus yang cukup besar.

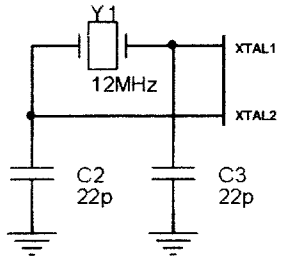
3.2.2 Mikrokontroler AT90S2313

Mikrokontroler AT90S2313 digunakan sebagai *interface* antara PC/komputer dengan mikrokontroler ATMega8535 sebelum diproses ke motor *stepper*. Perangkaian dan konfigurasi pin-pin dari mikrokontroler ini telah dijelaskan pada bab sebelumnya. Sistem minimum mikrokontroler AT90S2313 dapat dilihat pada Gambar 3.3 berikut.



Gambar 3.3 Sistem minimum mikrokontroler AT90S2313

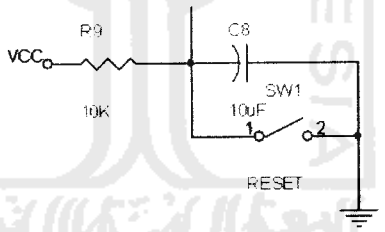
Rangkaian sistem minimum dari mikrokontroler AT90S2313 terdiri dari rangkaian osilator dan *power on reset*. Rangkaian osilator ini digunakan untuk membangkitkan *clock*/detak. Kristal yang dipakai adalah 12 MHz, untuk menghasilkan kecepatan transmisi USB 1.0 sebesar 12 MHz / 8 = 1,5 MHz.



Gambar 3.4 Rangkaian osilator eksternal

Pada rangkaian osilator ini digunakan dua buah kapasitor 22pF. Sedangkan rangkaian *power on reset* berfungsi untuk menjaga agar pin RST mikrokontroler selalu berlogika rendah saat mikrokontroler mengeksekusi program. Mikrokontroler direset pada transisi tegangan rendah ke tegangan tinggi, oleh karena itu pada pin RST dipasang kapasitor yang terhubung ke VCC dan resistor ke *ground* yang akan menjaga RST bernilai 1 (satu) saat pengisian kapasitor dan bernilai 0 (nol) saat kapasitor penuh.

Pada saat sumber tegangan diaktifkan kapasitor terhubung singkat sehingga arus mengalir dari VCC langsung ke kaki RST sehingga reset berlogika 1, kemudian kapasitor terisi hingga tegangan pada kapasitor sama dengan VCC. Pada saat itu kapasitor terisi penuh. Dengan demikian tegangan reset akan turun menjadi 0 sehingga kaki RST berlogika 0.



Gambar 3.5 Rangkaian Power On Reset

Port B Mikrokontroler AT90S2313

Port B merupakan *port* I/O 8-bit *bi-directional*. Pin-pin pada *port* ini dapat diberi resistor *pull-up* internal secara individu. PB0 dan PB1 juga dapat digunakan untuk melayani *input* sebagai komparator analog. *Buffer port* B dapat mencatu

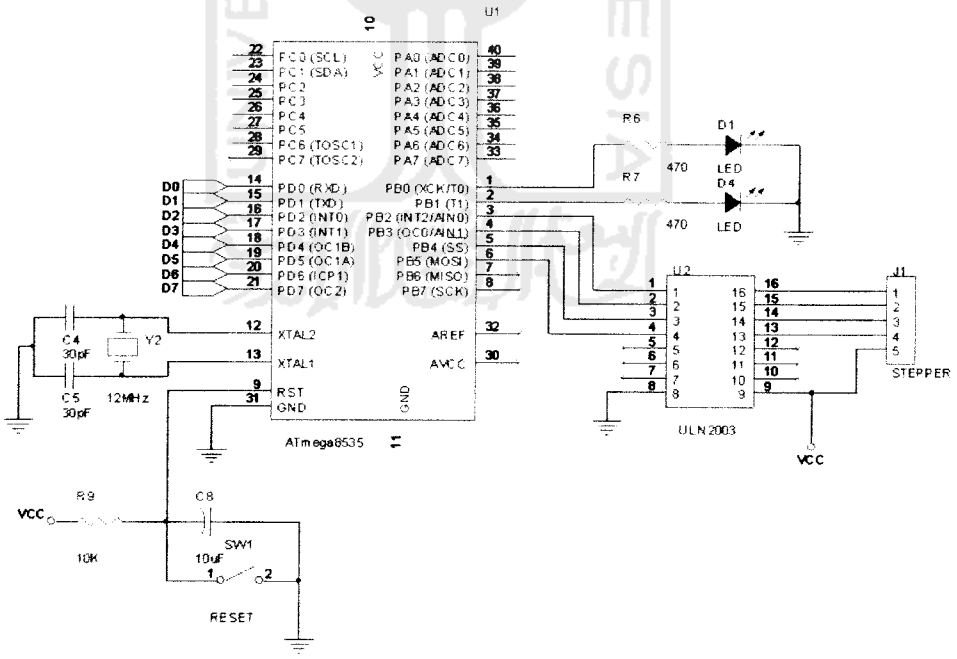
arus hingga 20 mA.dan men-*drive* LED secara langsung. *Port* PB5 dan PB6 sebagai jalur *input* dan *output* data untuk *download* memori.

***Port* D Mikrokontroler AT90S2313**

Pada *Port* D0 berfungsi untuk menerima data *input* dari USB komputer, *port* D1 berfungsi mengirim data *output* kembali ke komputer melalui USB, *port* PD4 dan PD5 berfungsi sebagai sumber *clock* yang digunakan.

3.2.3 Mikrokontroler ATMega8535

Sistem minimum mikrokontroler ATMega8535 digunakan sebagai *interface* antara PC dengan motor *stepper* yang data-data sebelumnya telah di *pull-up* oleh mikrokontroler AT90S2313. Perangkaian dan konfigurasi pin-pin dari mikrokontroler ini telah dijelaskan pada bab sebelumnya.



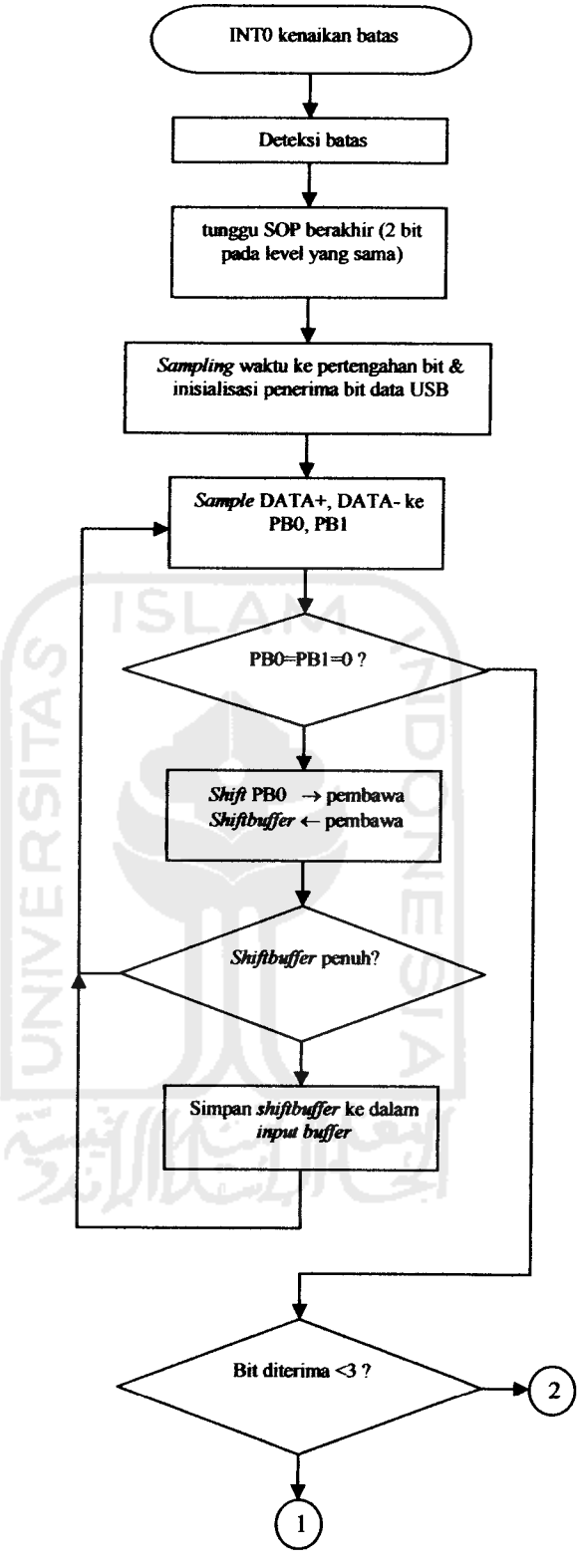
Gambar 3.6 Sistem minimum mikrokontroler ATMega8535

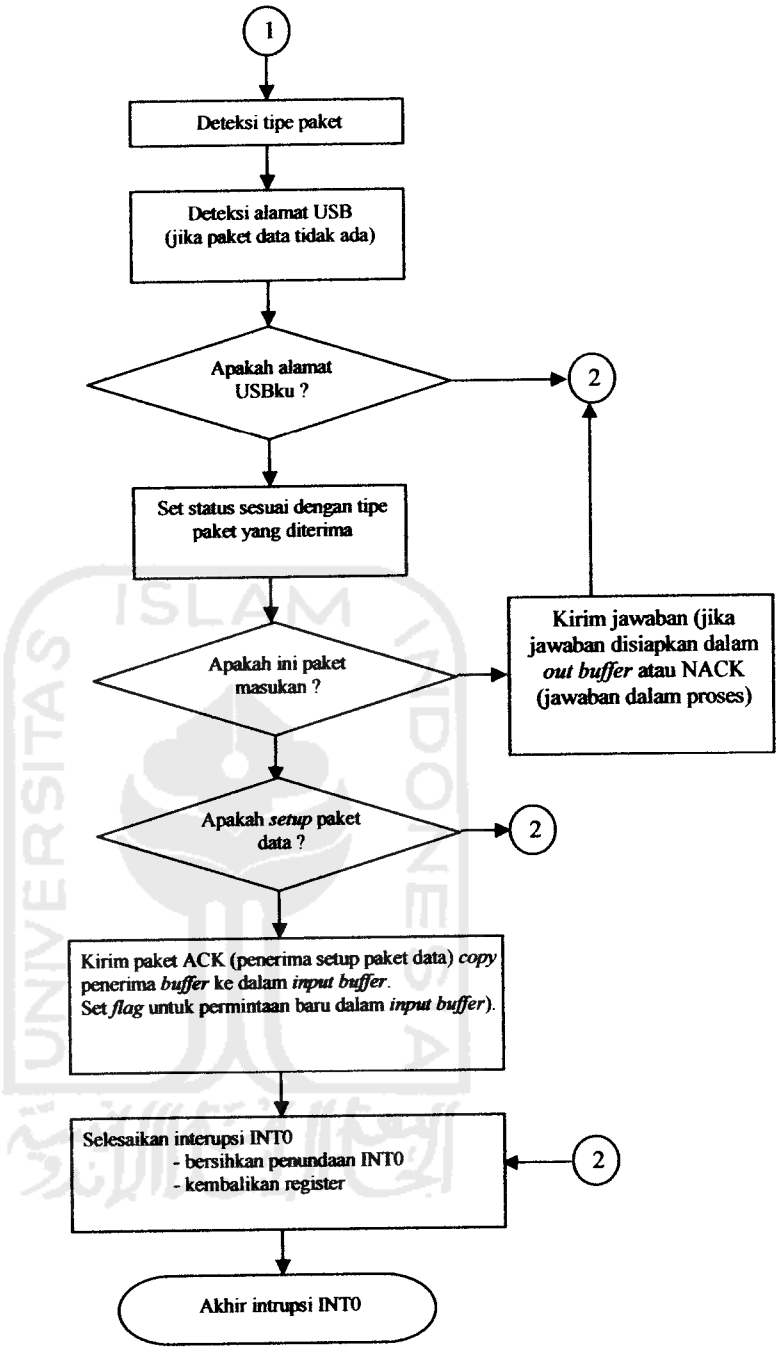
3.3 Perancangan Perangkat Lunak

Dalam perancangan perangkat lunak besarnya derajat serta arah putaran motor *stepper* ditentukan dalam program yang dapat dipilih oleh *user*. Untuk alat ini telah diberi ketentuan kecepatan motor *stepper* 10 derajat per step.

Penulisan program yang digunakan dalam rangkaian ini adalah dengan bahasa Delphi menggunakan bahasa *assembler* yang diperuntukkan bagi mikrokontroler ATMega8535.

Untuk dapat mengetahui lebih jelas dari proses pemrograman pada mikrokontroler, dari gambar diagram alir, jalannya program pada mikrokontroler akan akan terimplementasi pada motor *stepper*. maka berikut akan ditampilkan *flowchart* atau diagram alirnya.


INT0 kenaikan batas
Deteksi batas
tunggu SOP berakhir (2 bit pada level yang sama)
Sampling waktu ke pertengahan bit & inisialisasi penerima bit data USB
Sample DATA+, DATA- ke PB0, PB1
PB0=PB1=0 ?
Shift PB0 → pembawa
Shiftbuffer ← pembawa
Shiftbuffer penuh?
Simpan *shiftbuffer* ke dalam *input buffer*
Bit diterima <3 ?
2
1



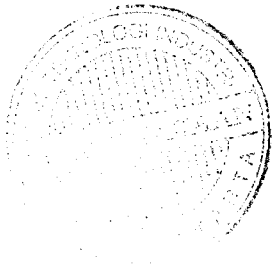
Gambar 3.7 Diagram alir mikrokontroler

Adapun penjelasan diagram *flowchart* tersebut adalah sebagai berikut: Interupsi eksternal 0 aktif sepanjang waktu, sementara *firmware* bekerja. Rutin ini

memulai penerimaan serial data USB (bisa juga disebut "Resepsi USB"). Interupsi eksternal terjadi pada batas peningkatan di pin INT0 (batas peningkatan menunjukkan awal dari pola *sync* dari sebuah paket USB). Hal ini mengaktifkan rutin penerimaan USB.

Pertama-tama *sampling* data harus disinkronkan ke dalam pertengahan lebar bit, hal ini dikerjakan sesuai pola *sync* (berupa gelombang kotak). Karena durasi bit hanya 8 siklus dari *clock* XTAL dan interupsi yang terjadi dapat ditunda (+/- 4 siklus), batas sinkronisasi dalam pola *sync* harus dijalankan dengan hati-hati. Akhir dari pola *sync* dan awal dari bit data dideteksi menurut dua bit terakhir dengan level rendah dalam paket *sync*.

Setelah ini, *sampling* data dimulai. *Sampling* dijalankan di pertengahan bit. Karena kecepatan data 1.5Mbit/detik (1.5MHz) dan kecepatan mikrokontroler 12MHz, maka kita hanya mempunyai 8 siklus untuk sampling bit data, menyimpannya ke dalam *buffer bit*, menggilir *buffer* bit , memeriksa jika seluruh *byte* telah diterima, menyimpan bit ke dalam SRAM, dan memeriksa untuk EOP. Proses ini mungkin merupakan bagian paling penting dari *firmware*, segala hal harus dikerjakan secara sinkron pada saat yang tepat. Ketika seluruh paket USB telah diterima, penguraian kode paket harus dijalankan. Pertama-tama, alat harus menentukan dengan cepat tipe paket (*SETUP, IN, OUT, DATA*) dan menerima alamat USB. Penguraian kode yang cepat ini harus dijalankan di dalam rutin interupsi karena sebuah jawaban dibutuhkan sangat cepat setelah penerimaan paket USB (alat harus menjawab dengan ACK ketika paket dengan alamat alat



telah diterima, dan dengan NAK ketika paket adalah untuk alat tetapi tidak ada jawaban yang sudah siap).

Pada akhir rutin penerimaan (setelah paket ACK/NAK dikirimkan) sampel data dari *buffer* harus dikopikan ke dalam *buffer* yang lain dimana penguraian kode akan dijalankan. Ini dilakukan agar *buffer* penerima bisa menerima paket baru.

Selama proses penerimaan tipe paket diuraikan kodenya dan *Flag* koresponden di set. *Flag* ini di uji dalam *loop* program utama dan menurut nilainya, tindakan yang sesuai akan diambil dan jawaban koresponden akan disiapkan dengan tanpa memperhatikan ketentuan kecepatan mikrokontroler.

INT0 harus diaktifkan untuk menjaga waktu invokasi yang sangat cepat pada semua rutin *firmware*, sehingga tidak ada interupsi yang menyebabkan eksekusi interupsi yang lain tidak berjalan (sebagai contoh serial *line* interupsi yang diterima). INT0 harus bisa melakukan proses ini, penerimaan yang cepat di dalam INT0 dalam rutin interupsi sangat penting, dan penting juga untuk mengoptimalkan *firmware* untuk kecepatan dan waktu yang tepat. Yang terpenting adalah mengoptimalkan register *backup* dalam rutin interupsi.