BAB II
TINJAUAN PUSTAKA

2.1 TELEPON

Telepon berasal dari kata dasar *Tele* dan *Phone*. *Tele* Artinya jauh dan *Phone* artinya pembicaraan/bicara. Maka kata *telepon* dapat diartikan suatu pembicaraan jarak jauh. Pada masa sekarang ini jaringan telepon yang sudah menyebar sedemikian luasnya di seluruh dunia menjadikan telepon sebagai sarana komunikasi jarak jauh yang paling utama.

2.1.1 Prinsip-prinsip Dasar Telepon

Pesawat telepon dapat dikelompokkan menjadi 3 bagian besar yang tentunya saling mendukung, bagian tersebut antara lain:

a. Dialer

   Dialer merupakan bagian telepon yang menghubungkan telepon dengan sentral telepon, sehingga sentral telepon dapat berhubungan dengan nomor telepon yang dituju.

b. Tone ringing

   Tone ringing merupakan bagian telepon yang berfungsi untuk mendeteksi sinyal dering dari sentral, sehingga sentral dapat menghubungkan telepon dengan pemanggilnya. Di mana tegangannya adalah 80 - 120 Vac. Dering telepon dideteksi oleh rangkaian ring detector yang kemudian dihitung oleh sistem prosesor.

c. Speech Network

   Speech Network merupakan bagian telepon yang berfungsi untuk mengubah sinyal suara menjadi sinyal listrik pada *microphone* dan mengubah sinyal listrik menjadi sinyal suara pada speaker.

2.1.2 Cara Kerja Pesawat Telepon

Cara kerja pesawat telepon erat kaitannya dengan status saluran telepon. Dimana keadaan telepon dibedakan menjadi 2 keadaan, telepon dalam keadaan tertutup, dan keadaan terbuka.

a. Keadaan Tertutup (On Hook)

Dalam keadaan tertutup pesawat telepon, kondisi ini dinamakan on-hook. Tegangan pada keadaan on hook adalah 48 VDC.

b. Keadaan terbuka (Off Hook)

Pesawat telepon dalam kondisi terbuka disebut dengan kondisi off-hook dan pada kondisi ini terjadi perubahan tegangan dari 48 VDC menjadi 10 VDC, dikarenakan pada kondisi ini terjadi komunikasi antar komponen. Kemudian pada saat gagang telepon ditutup kembali secara tidak langsung terjadinya penekanan pada push button/tombol reset.

2.1.3 **DTMF** (*Dual Tone Multiple Frequency*)

Telepon PSTN maupun handphone saat ini menggunakan sistem yang dikenal secara umum disebut DTMF yaitu *Dual Tone Multiple Frequencys.* Telepon PSTN pada umumnya memiliki 10 buah tombol ditambah * dan # jadi jumlahnya adalah 12. Di dalam komunikasi ke 12 tombol tersebut dikirimkan dengan 2 frekuensi yang berbeda. Satu frekuensi tinggi (*High Frequency)* dan satu frekuensi lagi frekuensi rendah (*Low Frequency).* Masing-masing memiliki 4 macam variasi (nilai frekuensi) sinyal sehingga dengan gabungan 2 frekuensi tadi dapat di kodekan 16 (4 pangkat 2) macam simbol. Untuk lebih jelas dapat dilihat pada tabel dibawah ini:

Tabel 2.1 Frekuensi Pada Keypad Telepon

Frekuensi		**High**		
		1209 Hz	*1336 Hz*	*1477 Hz*
Low	*697 Hz*	1	2	3
	770 Hz	4	5	6
	852 Hz	7	8	9
	941 Hz	*	0	#

Tabel 2.2 Frekuensi dari 3 kondisi

Event	**Low frequency**	**High frequency**
Busy signal	480 Hz	620 Hz
Dial tone	350 Hz	440 Hz
Ringback tone (US)	440 Hz	480 Hz

Dari table 2.1 dapat ketahui bahwa setiap penekanan tombol di pesawat telepon, telepon akan membangkitkan dua nada (*tone*) yaitu nada berfrekuensi tinggi dan satu nada berfrekuensi rendah. Kedua sinyal tersebut dikirimkan ke penerima (*operator*). Dengan cara melakukan penguraian (*decoding*) terhadap kedua sinyal tadi maka penerima dapat mendeteksi tombol-tombol apa saja yang ditekan oleh pemakai (*User).* Sifat inilah yang akan digunakan untuk membangun aplikasi perespon telepon menggunakan peranti elektronik, yaitu interface dan komputer.

2.2 Mikrokontroller ATMEL AT89S51

Mikrokontroler AT89S51 merupakan IC yang memiliki 4Kb flash PEROM *(Programmatable dan Erasable Read Only Memory*) yaitu ROM yang dapat ditulis dan dihapus menggunakan perangkat programmer, memiliki 128 x 8 bit internal RAM 32. Programmable I/O lines dan keperluan port parallel. Satu port parallel terdiri dari 8 kaki dengan demikian 32 kaki tersebut membentuk 4 buah port parallel, yang masing-masing di kenal sebagai *port-0*, *port-1*, *port-2*, dan *port-3*. nomor dari masing-masing jalur/kaki dari port parallel mulai dari 0 sampai 7, jalur/kaki pertama port-0 disebut sebagai P0.0 dan jalur/kaki terakhir untuk port-3 adalah P3.7.

a. Konfigurasi pin dari Mikrokontroller AT89S51

a. Pin 1 sampai 8 (port 1) merupakan port pararel 8 bit dua arah (bidirectional) yang dapat digunakan untuk berbagai keperluan (general purpose)
b. Pin 9 (reset) adalah masukan reset (aktif high). Pulsa transisi dari rendah ke tinggi akan me-reset. Pin ini dihubungkan dengan rangkaian power on reset pada gambar 2.2.
c. Pin 10 sampai 17 (port 3) adalah port pararel 8 bit dua arah yang memiliki fungsi pengganti. Fungsi pengganti meliputi TxD (Transmit Data), RxD (Receive Data), Int0 (Interupt 0), Int1 (Interupt 1), T0 (timer 0), T1 (timer 1), WR(Write), RD(Read). Bila pin-pin ini tidak dipakai, pena-pena ini dapat digunakan sebagai port pararel serbaguna.
d. Pin 18 (XTAL 1) adalah pin masukan ke rangkaian osilator internal.

e. Pin 19 (XTAL 2) adalah pin keluaran ke rangkaian osilator internal. Pin ini di pakai bila mengunakan osilator kristal.

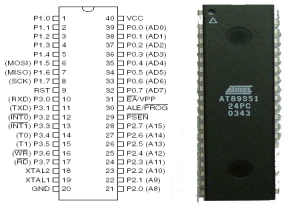
f. Pin 20 (GND) dihubungkan ke ground.

g. Pin 21 sampai 28 (port 2) adalah port pararel 2 selebar 8 bit dua arah (bidirectional).

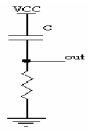
h. Pin 29 adalah pin PSEN (Program store enable).

i. Pin 30 adalah pin ALE (Address Latch Enable) yang digunakan untuk menahan alamat memori eksternal selama pelaksanaan instruksi.

j. Pin 31 (EA). Bila pin ini diberi logika tinggi mikrokontroller akan melaksanakan instruksi dari ROM/EPROM. Bila diberi logika rendah mikrokontroller akan melaksanakan seluruh instruksi dari memori luar.



Gambar 2.1 Konfigurasi pin dari mikrokontroller AT89S51



Gambar 2.2 Rangkaian power on reset

k. Pin 32 sampai 39 (port 0) merupakan port pararel 8 bit open drain dua arah.

l. Pin 40 (Vcc) dihubungkan ke Vcc (+5 volt).

Tabel 2.3 Fungsi-fungsi Alternatif pada Port-3

Kaki Port	**Fungsi Alternatif**
P3.0 (1)	*T2 (masukan eksternal pewaktu/pencacah)*
P3.1 (1)	*T2 EX (pemicu capture/ reload pewaktu/pencacah)*
P3.0	*RXD (port masukan serial)*
P3.1	*TXD (port keluaran serial)*
P3.2	*INTO (interupsi Eksternal 0)*
P3.3	*ITOTI (interupsi Eksternal 1)*
P3.4	*T0 (masukan Eksternal pencacah 0)*
P3.5	*T1 (masukan Eksternal pencacah 1)*
P3.6	*WR (sinyal tanda baca memori data eksternal)*
P3.7	*RD (sinyal tanda tulis memori data eksternal)*

b. Konfigurasi I/O

Mikrokotroller AT89S51 mempunyai 32 bit jalur I/O yang terbagi dalam 4 port yaitu port 0 sampai dengan port 3.

a. Port 0

Port 0 merupakan 8 bit biderectional I/O dengan rangkaian open drain sebagai penggerak di dalamnya, sehingga saat diisi logika satu maka port ini akan bersifat mengambang (*high impedance*). Jadi supaya dapat dijadikan rangkaian I/O diperlukan resistor pull-up eksternal. Port 0 juga dapat dijadikan sebagai *multiplexed low order address bus s*aat dikonfigurasikan dengan rangkaian memori eksternal. Saat flash programing port ini menerima byte kode program dan mengeluarkan byte kode saat proses *verifikasi*.

b. Port 1

Port 1 terdiri dari 8 bit I/O biderectional dengan internal resistor pull-up

c. Port 2

Port 2 Merupakan 8 bit biderectional I/O dengan internal resistor pullup. Saat dikonfigurasikan dengan rangkaian memori eksternal port ini akan berfungsi sebagai addres bus byte yang tinggi (A8-A15). Port 2 berfungsi sebagai *multiplexed high order address bus* saat flash programing dan sebagai bit kontrol saat proses verifikasi.

d. Port 3

Port 3 Merupakan 8 bit biderectional I/O dengan internal resistor pullup. Port 3 juga dapat difungsikan untuk beberapa fungsi khusus seperti yang ditunjukkan pada tabel 2.3

c. Timer AT89S51

Sebuah timer AT89S51 bekerja dengan mencacah. Tidak tergantung pada fungsi sebagai *timer*, *counter*, atau *generator baud rate*, sebuah timer akan selalu ditambah satu oleh mikrokontroler.

d. Penggunaan Timer Untuk Mengukur Waktu

Fungsi timer yang utama adalah untuk mengukur waktu. Saat sebuah timer digunakan untuk mengukur waktu, dia akan bertambah satu setiap satu siklus

mesin. Setiap siklus mesin membutuhkan 12 pulsa kristal. Maka apabila sebuah AT89S51 dengan kristal 11,059 MHz, maka timer setiap detiknya akan berharga:

$$11.059.000/12 = 921.583$$

Dengan kata lain, terdapat 921.583 kali pencacahan dalam setiap detiknya. Tidak seperti instruksi-instruksi yang bisa memakan satu hingga empat siklus mesin, sebuah timer selalu konsisten bertambah satu setiap satu kali siklus mesin. Sehingga, jika diinginkan sebuah timer yang berharga 50.000, berarti memakan waktu sebesar:

$$50.000/921.583 = 0{,}0542 \text{ s}$$

Dengan kata lain, diperlukan waktu 0,0542 detik untuk mendapatkan timer yang telah berharga 50.000. Melalui cara serupa, jika ingin mendapatkan waktu 0,05 detik, maka dibutuhkan timer yang mencacah hingga:

$$0{,}05 \times 921.583 = 46.079{,}15$$

Ini berarti kita perlu memonitor cacahan dari timer hingga mencapai harga 46.079. Walaupun tidak benar-benar presisi karena menghilangkan hitungan 0,15 namun cukup mendekati.

e. Cara Membaca Status Timer

Cara membaca status timer ada dua cara. Yang pertama dengan membaca harga aktual (*actual)* 16 bit dari timer, dan yang kedua adalah mendeteksi apakah timer menemui kondisi *overflow*. Jika timer yang digunakan adalah timer mode 8-bit, pembacaan harga aktual cukup mudah. Caranya hanya membaca harga 1 byte tersebut. Namun jika timer yang digunakan adalah mode 13 bit atau 16 bit, permasalah menjadi lebih rumit. Bagaimana jika harga aktual *low byte* adalah 255 dan pembacaan *high byte* adalah 15. Seharusnya harga sebenarnya adalah high byte 14 dan low byte adalah 255, karena saat membaca low byte sebesar 255, beberapa saat kemudian high byte akan bertambah satu saat pembacaan, sehingga pembacaan menjadi meleset sebesar 256 hitungan karena terletak pada high byte. Pemecahannya adalah dengan membaca high byte terlebih dahulu dan kemudian membaca low byte. Setelah itu high byte dibaca lagi dan kemudian dibandingkan dengan pembacaan semula, bila berbeda, maka yang dipakai adalah pembacaan high byte yang pertama.

Kadang yang perlu diketahui hanyalah saat timer reset menjadi nol. Dengan kata lain, tidak penting berapa harga aktual dari timer, namun kapan timer overflow dan kembali menjadi nol. Saat overflow, mikrokontroler secara otomatis mengeset bit *TFx* dalam register *TCON*. Ini berarti pengecekan overflow cukup dengan mengecek apakah bit TFx set atau tidak. Dengan cara ini bisa dibuat program untuk menentukan selang yang pasti. Dari pembahasan sebelumnya, diketahui bahwa untuk mendapatkan selang 0,05 detik diperlukan pencacahan hingga 46.079 kali. Penggunaan mode ini akan menginisialisasi timer dengan harga selisih antara 65.535 dan 46.079, yaitu 19.457. Sehingga 46.079 cacahan berikutnya setelah 19.457 akan menyebabkan timer overflow. Contoh program timer:

```
MOV TH0,#76        ; (76X256=19.456)
MOV TL0,#01        ; (19.456+1=19.457)
MOV TMOD,#01       ; Timer 0 Mode 16-bit
SETB TR0           ; Start Timer 0
JNB TF0,$          ; Loop sampai overflow
```

Tabel 2.4 Register Timer

	Register Timer				Register Interupsi			
88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
	TCON.7	TCON.6	TCON.5	TCON.4	TCON.3	TCON.2	TCON.1	TCON.0

2.3 Bahasa Assembly

Bahasa pemograman adalah program komputer yang menstranslitasi program dari bahasa assembly ke bahasa mesin. *Mnemonics*, *alpanumerik* digambarkan sebagai alat bantu bagi programmer antara memprogram mesin komputer dari pada menggunakan serangkaian 0 dan 1 (*bahasa mesin*) yang panjang rumit.

Bahasa mesin adalah satu-satunya bahasa yang dimengerti oleh mikrokontroler. Bahasa ini tidak mudah untuk dimengerti oleh manusia. Sedangkan bahasa assembly adalah suatu bentuk bahasa mesin yang bisa dimengerti oleh manusia. Setiap pernyataan dari bahasa assembly menggambarkan satu pernyataan bahasa mesin. Sebagai contoh instruksi JMP (asal kata *JUMP*) akan lebih mudah dimengerti dibandingkan instruksi B3H.

Pemrograman dengan menggunakan bahasa assembly/bahasa mesin menghasilkan program yang kecil dan cepat. Hal ini dikarenakan kita sepenuhnya mengontrol kerja dari program, tetapi tentu saja jika kita membuat program yang bertele-tele dan berbelit akan menyebabkan program berjalan lambat.

2.4 Relay

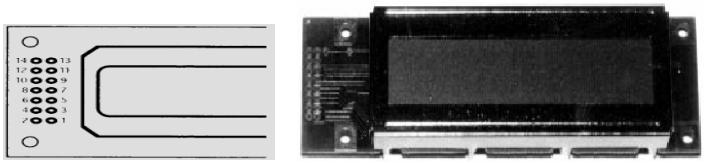
Relay adalah sebuah saklar elektromagnetik yang mempunyai prinsif kerja dimana kumparan yang berintikan sebuah lempengan besi lunak dan apabila dialiri aliran listrik, maka lempengan besi lunak tersebut akan menjadi magnit. Magnit tersebut menarik/menolak pegas kontak sebuah alat penghubung dan akibatnya akan terjadi kontak dan lepas kontak dari alat penghubung tersebut. Prinsip seperti ini dapat dimanfaatkan sebagai dasar pembuatan saklar otomatis yang banyak dipergunakan dalam bidang elektronika.



Gambar 2.3 Kontak Relay

2.5 Liquid Crystal Dysplay (LCD)

Liquid Crystal Display (LCD) merupakan salah satu media tampilan yang sering digunakan seperti pada telepon seluler, kalkulator, jam, dan lain-lain. Berdasarkan tampilannya. Jenis – jenis size LCD adalah (1×16, 2×16, 2×20 dsb). Namun ada standarisasi yang cukup populer digunakan banyak *vendor* LCD, yaitu HD44780U, yang memiliki *chip* kontroler Hitachi 44780. LCD bertipe ini memungkinkan pemrogram untuk mengoperasikan komunikasi data secara 8 bit atau 4 bit. Jika menggunakan jalur data 4 bit akan ada 7 jalur data (3 untuk jalur kontrol & 4 untuk jalur data). Jika menggunakan jalur data 8 bit maka akan ada 11 jalur data (3 untuk jalur kontrol & 8 untuk jalur data). 3 jalur kontrol ke LCD ini adalah EN (*Enable*), RS (*Register Select*) dan R/W (*Read*/*Write*).



Gambar 2.4 Konfigurasi Pin LCD

Tabel 2.5 Deskripsi Pin LCD

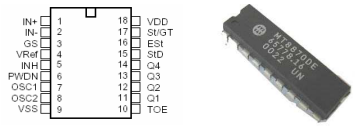
Pin	Deskripsi
1	Ground (Vss)
2	Vcc (Vdd)
3	Pengatur kontras (Vee)
4	Instruction/Register Select (RS)
5	Read/Write LCD Registers (R/W)
6	Enable clock (E)
7-14	Data I/O Pins (D0,D1….D7)

Sebagaimana terlihat pada kolom deskripsi, interface LCD merupakan sebuah parallel bus, dimana hal ini sangat memudahkan dan sangat cepat dalam pembacaan dan penulisan data dari atau ke LCD. Kode ASCII yang ditampilkan sepanjang 8 bit dikirim ke LCD secara 4 atau 8 bit pada satu waktu. Jika mode 4 bit yang digunakan, maka 2 *nibble* data dikirim untuk membuat sepenuhnya 8 bit (pertama dikirim 4 bit MSB lalu 4 bit LSB dengan pulsa clock EN setiap nibblenya). Jalur kontrol EN digunakan untuk memberitahu LCD bahwa mikrokontroller mengirimkan data ke LCD. Untuk mengirim data ke LCD program harus menset EN ke kondisi high (1) dan kemudian menset dua jalur kontrol lainnya (RS dan R/W) atau juga mengirimkan data ke jalur data bus. Saat jalur lainnya sudah siap, EN harus diset ke 0 dan tunggu beberapa saat (tergantung pada *datasheet* LCD), dan set EN kembali ke high (1). Ketika jalur RS berada dalam kondisi low (0), data yang dikirimkan ke LCD dianggap sebagai sebuah perintah atau instruksi khusus (seperti bersihkan layar, posisi kursor dll). Ketika RS dalam kondisi high atau 1, data yang dikirimkan adalah data ASCII yang akan ditampilkan dilayar. Misal, untuk menampilkan huruf “A” pada layar maka RS harus diset ke 1. Jalur kontrol R/W harus berada dalam kondisi low (0) saat informasi pada data bus akan dituliskan ke LCD. Apabila R/W berada dalam kondisi high (1), maka program akan melakukan *query* (pembacaan) data dari LCD. Instruksi pembacaan hanya satu, yaitu Get LCD status (membaca status LCD), lainnya merupakan instruksi penulisan. Jadi hampir setiap aplikasi yang menggunakan LCD, R/W selalu diset ke 0. Jalur data dapat terdiri 4 atau 8 jalur (tergantung mode yang dipilih pengguna), mereka dinamakan DB0, DB1, DB2, DB3, DB4, DB5, DB6 dan DB7. Mengirim data secara parallel baik 4 atau 8 bit

merupakan 2 mode operasi primer. Untuk membuat sebuah aplikasi interface LCD, menentukan mode operasi merupakan hal yang paling penting. Mode 8 bit sangat baik digunakan ketika kecepatan menjadi keutamaan dalam sebuah aplikasi dan setidaknya minimal tersedia 11 pin I/O (3 pin untuk kontrol, 8 pin untuk data). Sedangkan mode 4 bit minimal hanya membutuhkan 7 bit (3 pin untuk kontrol, 4 pin untuk data). Bit “RS” digunakan untuk memilih apakah data atau instruksi yang akan ditransfer antara mikrokontroller dan LCD. Jika bit ini di set (RS = 1), maka byte pada posisi kursor LCD saat itu dapat dibaca atau ditulis. Jika bit ini di reset (RS = 0), bisa merupakan instruksi yang dikirim ke LCD atau status eksekusi dari instruksi terakhir yang dibaca.

2.6 MT8870D

MT8870D adalah jenis DTMF penerima monolitis memiliki ukuran kecil, dan butuhkan tegangan yang rendah namun memiliki perporm tinggi. Arsitekturnya terdiri dari suatu bandsplit bagian saringan, yang dapat memisahkan nada frekuensi pada kelompok tinggi dan rendah, dan terdapat suatu digital penghitung yang bagian memverifikasi setetelah menerima frekuensi nada dan frekuensi nada yang diterima akan diterjemahkan (output) menjadi bahasa mesin, kristal yang dibutuhkan oleh MT8870D adalah 3.579545 MHz.



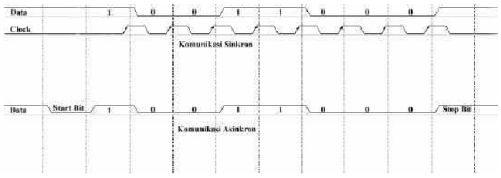
Gambar 2.5. Konfigurasi pin MT8870D

Tabel 2.6 Deskripsi Pin MT8870D

Pin #		Name	Description
18	20		
1	1	IN+	Non-Inverting Op-Amp (Input).
2	2	IN-	Inverting Op-Amp (Input).
3	3	GS	Gain Select. Gives access to output of front end differential amplifier for connection of feedback resistor.
4	4	V_{Ref}	Reference Voltage (Output). Nominally $V_{DD}/2$ is used to bias inputs at mid-rail (see Fig. 6 and Fig. 10).
5	5	INH	Inhibit (Input). Logic high inhibits the detection of tones representing characters A, B, C and D. This pin input is internally pulled down.
6	6	PWDN	Power Down (Input). Active high. Powers down the device and inhibits the oscillator. This pin input is internally pulled down.
7	8	OSC1	Clock (Input).
8	9	OSC2	Clock (Output). A 3.579545 MHz crystal connected between pins OSC1 and OSC2 completes the internal oscillator circuit.
9	10	V_{SS}	Ground (Input). 0V typical.
10	11	TOE	Three State Output Enable (Input). Logic high enables the outputs Q1-Q4. This pin is pulled up internally.
11-14	12-15	Q1-Q4	Three State Data (Output). When enabled by TOE, provide the code corresponding to the last valid tone-pair received (see Table 1). When TOE is logic low, the data outputs are high impedance.
15	17	StD	Delayed Steering (Output).Presents a logic high when a received tone-pair has been registered and the output latch updated; returns to logic low when the voltage on St/GT falls below V_{TSt}.
16	18	ESt	Early Steering (Output). Presents a logic high once the digital algorithm has detected a valid tone pair (signal condition). Any momentary loss of signal condition will cause ESt to return to a logic low.
17	19	St/GT	Steering Input/Guard time (Output) Bidirectional. A voltage greater than V_{TSt} detected at St causes the device to register the detected tone pair and update the output latch. A voltage less than V_{TSt} frees the device to accept a new tone pair. The GT output acts to reset the external steering time-constant; its state is a function of ESt and the voltage on St.
18	20	V_{DD}	Positive power supply (Input). +5V typical.
	7, 16	NC	No Connection.

2.7 Komunikasi Serial

Port Serial 89C51 dapat digunakan untuk komunikasi data secara sinkron maupun asinkron Komunikasi data serial secara sinkron adalah merupakan bentuk komunikasi data serial yang memerlukan sinyal clock untuk sinkronisasi di mana sinyal clock tersebut akan tersulut pada setiap bit pengiriman data sedangkan komunikasi asinkron tidak memerlukan sinyal clock sebagai sinkronisasi. Pengiriman data pada komunikasi serial 89C51 dilakukan mulai dari bit yang paling rendah (LSB) hingga bit yang paling tinggi (MSB).



Gambar 2.6 Komunikasi Sinkron dan Komunikasi Asinkron

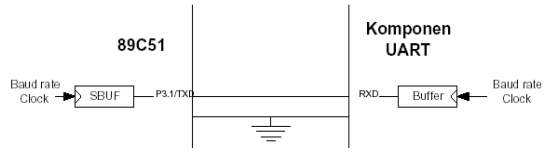
a. Komunikasi Sinkron

Sinyal clock pada komunikasi sinkron diperlukan oleh peralatan penerima data untuk mengetahui adanya pengiriman setiap bit data. Tampak pada gambar 2.6 bahwa sinyal clock tersulut *(positive edge)* pada saat pengiriman bit yang pertama dan setiap perubahan bit data. Peralatan atau komponen penerima akan mengetahui adanya pengiriman bit yang pertama ataupun perubahan bit data dengan mendeteksi sinyal clock.

b. Komunikasi Asinkron

Seperti telah disebutkan sebelumnya, komunikasi asinkron tidak memerlukan sinyal clock sebagai sinkronisasi, namun pengiriman data ini harus diawali dengan start bit dan diakhiri dengan stop bit seperti yang tampak pada gambar 2.6. Sinyal clock yang merupakan *baud rate* dari komunikasi data ini dibangkitkan oleh masing-masing baik penerima maupun pengirim data dengan frekwensi yang sama.

Penerima hanya perlu mendeteksi adanya start bit sebagai awal pengiriman data, selanjutnya komunikasi data terjadi antar dua buah shift register yang ada pada pengirim maupun penerima. Setelah 8 bit data diterima, maka penerima akan menunggu adanya stop bit sebagai tanda bahwa 1 byte data telah terkirim dan penerima dapat siap untuk menunggu pengiriman data berikutnya.



Gambar 2.7 Komunikasi UART

Pada aplikasinya proses komunikasi asinkron ini selalu digunakan untuk mengakses komponen-komponen yang mempunyai fasilitas *UART (Universal Asynchronous Receiver/Transmitter)* seperti Port Serial PC atau Port Serial mikrokontroler yang lain.

c. Baudrate

Baud rate dari Port Serial 89C51 dapat diatur pada Mode 1 dan Mode 3, namun pada Mode 0 dan Mode 2, baud rate tersebut mempunyai kecepatan yang permanen yaitu untuk Mode 0 adalah 1/12 frekwensi osilator dan Mode 2 adalah 1/64 frekwensi osilator.

Dengan mengubah bit SMOD yang terletak pada Register PCON menjadi set (kondisi awal pada saat sistem reset adalah clear) maka baud rate pada Mode 1, 2 dan 3 akan berubah menjadi dua kali lipat. Pada Mode 1 dan 3 baud rate dapat diatur dengan menggunakan Timer1. Cara yang biasa digunakan adalah Timer Mode 2 (8 bit auto reload) yang hanya menggunakan register TH1 saja. Pengiriman setiap bit data terjadi setiap Timer 1 overflow sebanyak 32 kali sehingga dapat disimpulkan bahwa:

Lama pengiriman setiap bit data = Timer 1 Overflow X 32

Baud rate (jumlah bit data yang terkirim tiap detik) = $\frac{1}{\text{Timer 1 Overflow X 32}}$

Apabila diinginkan baud rate 9600 bps maka timer 1 harus diatur agar overflow setiap $\frac{1}{9600 \text{ X } 32}$ detik

Timer 1 overflow setiap kali TH1 mencapai nilai limpahan (*overflow*) dengan frekwensi sebesar fosc/12 atau periode 12/fosc. Dari sini akan ditemukan formula $\frac{12 \text{ X } (256\text{-TH1})}{\text{fosc}} = \frac{1}{9600 \text{ X } 32}$ sebagai berikut: $9600 = \frac{\text{fosc}}{12 \text{ X } (256\text{-TH1}) \text{ X } 32}$

Dengan frekwensi osilator sebesar 11,0592 MHz maka TH1 adalah 253 atau 0FDH. Selain variabel-variabel di atas, masih terdapat sebuah variabel lagi yang menjadi pengatur baud rate serial yaitu Bit SMOD pada Register PCON.

Apabila bit ini set maka faktor pengali 32 pada formula 3.1 akan berubah menjadi 16. Oleh karena itu dapat disimpulkan formula untuk baud rate serial untuk Mode 1 dan Mode 3 adalah: $\text{Baud rate} = \frac{fosc}{12 \times (256-TH1) \times K}$

Tabel 2.7 Tabel Mode Serial vs baud rate

Mode	Baud rate	
0	$1/12\ f_{osc}$	
1	SMOD = 0	SMOD = 1
	$\text{Baud rate} = \frac{fosc}{12 \times (256 - TH1) \times 32}$	$\text{Baud rate} = \frac{fosc}{12 \times (256 - TH1) \times 16}$
2	$1/32\ f_{osc}$	$1/32\ f_{osc}$
3	$\text{Baud rate} = \frac{fosc}{12 \times (256 - TH1) \times 32}$	$\text{Baud rate} = \frac{fosc}{12 \times (256 - TH1) \times 16}$

2.8 Bahasa Pemograman Visual Basic

Visual Basic adalah salah satu *development tool*s untuk membangun aplikasi dalam lingkungan Windows. Dalam pengembangan aplikasi, Visual Basic menggunakan pendekatan Visual untuk merancang user interface dalam bentuk form, sedangkan untuk kodingnya menggunakan dialek bahasa Basic yang cenderung mudah dipelajari. Visual Basic telah menjadi *tools* yang terkenal bagi para pemula maupun para *developer* dalam pengembangan aplikasi skala kecil sampai skala besar.

Dalam lingkungan Windows *User-interface* sangat memegang peranan penting, karena dalam pemakaian aplikasi yang kita buat, pemakai senantiasa berinteraksi dengan *User-Interface* tanpa menyadari bahwa dibelakangnya berjalan intruksi-instruksi program yang mendukung tampilan dan proses yang dilakukan.

Pada pemograman Visual, pengembangan aplikasi dimulai dengan pembentukan *User-interface*, kemudian mengatur properti dari objek-objek yang digunakan dalam *user-interface*, dan baru dilakukan penulisan kode program untuk menangani kejadian-kejadian (*event*). Tahap pengembangan aplikasi demikian dikenal dengan istilah pengembangan aplikasi dengan pendekatan *bottom up.*

2.9 Penyearah (*Rectifier*)

Penyearahan adalah proses menyearahkan arus bolak-balik menjadi arus searah. Arus bolak balik-balik ini berasal dari tegangan jala-jala. Komponen

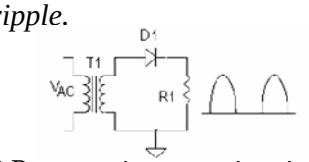
utama yang diperlukan dalam penyearahan adalah *transformator*, dioda dan kapasitor elektrolit.

Secara umum penyearah dibagi menjadi empat kategori yaitu :

a) Penyearah setengah gelombang
b) Penyearah gelombang penuh sistem CT
c) Penyearah gelombang penuh sistem jembatan

2.9.1 Penyearah Setengah Gelombang

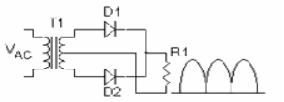
Penyearah ini bekerja dengan menggunakan satu dioda, sehingga hanya pulsa positif yang dapat terambil. Penyerah ini praktis sederhana, tetapi kekurangannya adalah bahwa gelombang keluaran bukan gelombang penuh sehingga rentan sekali akan *ripple.*



Gambar 2.8 Penyearah setengah gelombang

2.9.2 Penyearah Gelombang Penuh Sistem CT

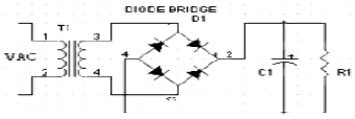
Penyearah ini menggunakan transformator jenis CT dengan dua buah dioda sebagai penyearah. Dioda bekerja secara bergantian untuk mengambil pulsa positif dan negatif, sehingga keluaran berupa gelombang penuh.



Gambar 2.9 Penyearah sistem CT

2.9.3 Penyearah Gelombang Penuh Sistem Jembatan

Penyearah ini menggunakan 4 buah dioda sebagai penyearah. Pada siklus pertama dua dioda bekerja untuk menyearahkan atau mengambil pulsa positif. Siklus selanjutnya dua dioda berikutnya yang bekerja untuk mengambil pulsa negatif. Keuntungan penyearah ini adalah bahwa keluaran berupa gelombang penuh dan jika salah satu dioda rusak, maka dioda yang satunya lagi akan tetap bekerja.



Gambar 2.10 Penyearah dengan sistem jembatan