Makalah Seminar Tugas Akhir

ALAT PENAMPIL TEKS INFORMASI MELALUI CHANNEL VIDEO PADA TELEVISI

Oleh : Andi Kristianto (L2F 097 609)
Jurusan Teknik Elektro Universitas Diponegoro Semarang

Abstrak

Masyarakat selalu membutuhkan informasi. Bentuk informasi bermacam-macam, salah satunya berupa teks. Untuk menampilkan teks informasi perlu suatu media. Televisi dapat menjadi media alternatif untuk menampilkan teks informasi. Dalam tugas akhir ini dirancang suatu alat penampil teks informasi pada televisi. Alat ini berbasis mikrokontroler P89C138. Informasi yang ditampilkan dapat disimpan, diubah, dan dihapus dengan menggunakan remote control.

1. Pendahuluan

1.1 Latar Belakang

Saat ini media penampil teks informasi umumnya menggunakan media konvensional seperti papan pengumuman. Pengisian informasinya dengan cara ditulis tangan atau dicetak. Media ini cukup efektif jika informasi yang disampaikan bersifat tetap atau paling tidak untuk jangka waktu yang relatif lama. Untuk informasi yang berubah sewaktu-waktu, penggantiannya akan menyulitkan karena biasanya media ini diletakkan pada tempat yang tinggi agar dapat dilihat oleh orang banyak. Kendala inilah yang menyebabkan perlunya media alternatif yang lebih efektif untuk menampilkan informasi dan dapat dengan mudah dilakukan penggantian jika diperlukan.

Dalam tugas akhir ini akan dibuat sebuah alat yang memanfaatkan televisi sebagai media penampil teks informasi melalui *channel* videonya. Dengan alat ini, informasi tetap maupun informasi yang sewaktu-waktu berubah dapat ditampilkan pada televisi. Untuk informasi tetap dapat disimpan dan ditampilkan terus-menerus sesuai kebutuhan. Sedangkan untuk informasi yang sewaktu-waktu berubah dapat disimpan dan dilakukan *edit* terlebih dahulu sesuai perubahan yang terjadi baru kemudian ditampilkan. Pengisian informasi dengan cara menekan tombol-tombol karakter pada *remote control*. Kemudian informasi tersebut akan dikirim dengan menggunakan sinar infra merah, dan ditampilkan pada televisi berupa teks informasi. Dengan menggunakan *remote control* maka pengisian, penggantian, dan penampilan informasi dapat dilakukan dari jarak jauh tanpa harus mendekati televisi.

1.2 Tujuan

Tujuan dari tugas akhir ini adalah merancang dan merealisasikan sebuah alat penampil teks informasi dengan memanfaatkan *channel* video pada televisi.

2. Landasan Teori

2.1 *Inter Integrated Circuit Bus* (I^2C *bus*)

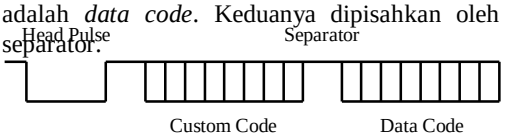
Sistem *bus* yang digunakan dalam pemrograman untuk skripsi ini adalah sistem I^2C *bus*. I^2C *bus* adalah sistem *bus* dua arah yang hanya membutuhkan 2 saluran *bus* yaitu *serial data line* (SDA) dan *serial clock line* (SCL) untuk pertukaran data antar IC. Berikut ini akan dijelaskan istilah-istilah yang digunakan dalam I^2C *bus* :

- *Transmitter* : komponen yang mengirim data ke *bus*.
- *Receiver* : komponen yang menerima data dari *bus*.
- *Master* : komponen yang memulai transfer, membangkitkan sinyal *clock*, dan mengakhiri transfer.
- *Slave* : komponen yang dialamati oleh *master*.
- *High* : logika '1'.
- *Low* : logika '0'.

Transfer atau pertukaran data berlangsung ketika *bus* tidak sibuk, yakni sewaktu SDA dan SCL dalam kondisi *high*. Selama pertukaran data, SDA harus stabil sewaktu SCL *high*, karena perubahan pada SDA ketika SCL *high* akan diartikan sebagai kondisi *start* atau *stop*.

2.2 Format Data *Remote Controller* Mitsubishi

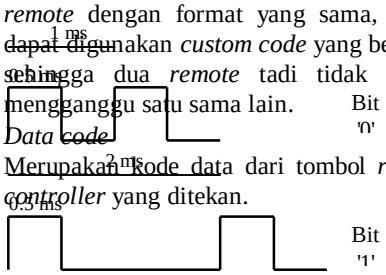
Format data yang dipancarkan oleh Mitsubishi mikrokontroler terdiri atas *head pulse* dan 16 bit kode. Kode 8 bit yang pertama adalah *custom code* dan kode 8 bit berikutnya adalah *data code*. Keduanya dipisahkan oleh separator.



Gambar 2.1 Format data *remote control* Mitsubishi

Bagian-bagian data *remote controller* Mitsubishi dapat dijelaskan sebagai berikut :

- *Head pulse*
  Head pulse terletak pada awal transmisi data dan berupa pulsa *low* 8 ms (milisekon) dan pulsa *high* 4 ms.
- Separator
  Digunakan untuk memisahkan data antara *custom code* dengan *data code*, yaitu berupa pulsa *high* selama 4 ms.
- *Custom code*
  Merupakan kode identifikasi dari suatu *remote controller*. Misalkan ada dua buah *remote* dengan format yang sama, maka dapat digunakan *custom code* yang berbeda sehingga dua *remote* tadi tidak saling mengganggu satu sama lain.
- *Data code*
  Merupakan kode data dari tombol *remote controller* yang ditekan.



Gambar 2.2 Pulsa bit ‘0’ dan ‘1’

2.3 *Infra Red Transmitter*

Infra red transmitter berupa *remote controller* dengan keluaran infra merah. *Remote controller* ini adalah alat untuk menuliskan teks informasi ke televisi. *Remote controller* ini terdiri dari beberapa tombol yang disebut *keypad*. Masing-masing tombol mewakili karakter-karakter yang berbeda.

2.4 *Infra Red Receiver*

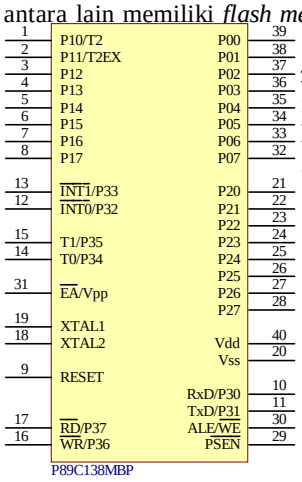
Receiver ini menerima sinyal infra merah yang dipancarkan oleh *remote controller*. Data serial hasil pengkodean dari sinyal infra merah yang diterima digunakan sebagai masukan mikrokontroler.

2.5 Mikrokontroler P89C138

Mikrokontroler P89C138 adalah sebuah chip mikrokomputer 8 bit yang mempunyai 16 *Kbyte* memori program jenis *flash* atau disebut *Flash Erasable & Programmable Read Only Memory* (FEPROM). Piranti mikrokontroler ini dibuat dengan menggunakan teknologi memori *non-volatile* dari *Philips* dengan instruksi-instruksi program yang kompatibel dengan mikrokontroler standar MCS’51.

Oleh karena chip mikrokontroler ini terdiri dari kombinasi CPU 8 bit dengan *flash memory*, menjadikan P89C138 sebagai mikrokomputer yang sangat populer, berdaya guna dan dapat memberikan solusi paling efektif dan sangat fleksibel untuk aplikasi-aplikasi pengontrolan.

P89C138 mempunyai beberapa kelebihan antara lain memiliki *flash memory* 16 *Kbyte*, I I/O, tiga *timer* 1(*ort* serial dua arah nternal. kontroler P89C13(.



Gambar 2.3 Konfigurasi pin mikrokontroler P89C138

Masing – masing pin memiliki fungsi tersendiri sebagai berikut :

- ***VCC***

Pin VCC adalah terminal untuk *power supply*.

- ***GND***

Pin GND adalah pin terminal sebagai *ground*.

- ***Port 0* (P0.0-P0.7)**

Port 0 pada pin 32-39 mempunyai fungsi ganda. Pada perancangan dengan sistem minimum, port 0 difungsikan sebagai I/O (*Input/Output*). Sedangkan pada perancangan dengan penambahan memori eksternal (baik RAM maupun ROM/EPROM), port 0 digunakan untuk *multiplexing* alamat (*address*) dan data. Jadi pada setengah siklus waktu kerja mikrokontroler, port 0 berfungsi sebagai keluaran alamat A0-A7 dan pada setengah siklus berikutnya port 0 berfungsi sebagai keluaran data D0-D7. Pergantian *address* dan data pada port 0 tersebut diketahui EPROM/RAM eksternal pada keluaran pin ALE.

- ***Port 1*** **(P1.0-P1.7)**

Port 1 pada pin 1-8 murni berfungsi sebagai I/O port. Terutama digunakan untuk menghubungkan mikrokontroler dengan piranti luar (misalnya DAC, ADC, *keyboard*, *display*, dan lain-lain).

- ***Port 2*** **(P2.0-P2.7)**

Port 2 pada pin 21-28 juga mempunyai fungsi ganda. Pada perancangan dengan sistem minimum, port 2 difungsikan sebagai I/O (*Input/Output*). Sedangkan pada perancangan dengan penambahan memori eksternal, port 2 digunakan sebagai *address bus* (A8-A15).

- ***Port 3*** **(P3.0-P3.7)**

Port 3 pada pin 10-17 adalah port multifungsi, setiap pin dari port 3 mempunyai fungsi-fungsi tersendiri seperti terlihat pada Tabel 2.1.

Tabel 2.1 Fungsi alternatif port 3

Port 3	Nama Pin	Fungsi Alternatif
P3.0	RXD	Penerima data pada komunikasi serial
P3.1	TXD	Pengirim data pada komunikasi serial
P3.2	INT0	*Interrupt* eksternal 0
P3.3	INT1	*Interrupt* eksternal 1
P3.4	T0	*Input* eksternal untuk *Timer* 0
P3.5	T1	*Input* eksternal untuk *Timer* 1
P3.6	WR	Indikator penulisan (*write*) memori eksternal
P3.7	RD	Indikator pembacaan (*read*) memori eksternal

- ***RST***

Sebagai pin masukan *reset*. Tegangan logika 1 pada pinnya selama dua kali siklus mesin ketika osilator bekerja menyebabkan *reset* pada piranti.

- ***PSEN***

PSEN (*Program Store Enable*) pada pin 29 berguna untuk output sinyal pengontrol pada pengambilan program (kode) dari ROM/EPROM eksternal. Pin PSEN akan mengeluarkan sinyal *low* setiap dua kali siklus mesin saat P89C138 mengeksekusi kode dari memori program eksternal. Pin PSEN biasa dihubungkan dengan pin OE (*Output Enable*) pada IC EPROM.

- ***ALE***

ALE (*Address Latch Enable*) pada pin 30 berguna untuk output sinyal pengontrol pada pengaturan *multiplexing* alamat dan data pada port 0. Pin ALE akan mengeluarkan sinyal *high* pada saat keluaran dari port 0 adalah *address*, dan akan mengeluarkan sinyal *low* pada saat keluaran dari port 0 adalah data.

- ***EA***

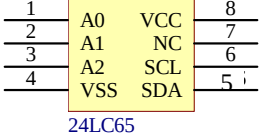
EA (*External Access*) pada pin 31 berguna untuk memilih apakah program (kode) mikrokontroler akan diletakkan di ROM/EPROM internal atau ROM/EPROM eksternal. Jika program (kode) diletakkan di ROM/EPROM eksternal, pin EA harus bernilai *low* (pin EA dihubungkan ke *ground*). Sebaliknya jika program (kode) diletakkan di ROM/EPROM internal, pin EA harus bernilai *high* (pin EA dihubungkan ke VCC). Pin PSEN otomatis tidak akan berfungsi jika EA bernilai *high*. Pin EA juga menerima tegangan 12 volt selama pemrograman *flash*.

- ***XTAL***

Seperti layaknya mikroprosesor pada umumnya, mikrokontroler juga memerlukan *clock* untuk operasinya. Meskipun sudah terdapat fasilitas *clock generator* di dalam P89C138, tetapi tetaplah dibutuhkan osilator kristal sebagai penggerak *clock generator* di dalam P89C138 tersebut. Pin XTAL1 adalah *input* ke *inverting amplifier oscilator* dan *input* ke rangkaian *clock* internal. Pin XTAL2 adalah *output* dari *inverting amplifier oscilator*. Frekuensi dari osilator kristal akan mempengaruhi siklus kerja dari mikrokontroler. Pada penggunaan umum (sesuai *datasheet*) biasa digunakan osilator kristal 12 MHz. Jadi kira-kira satu siklus kerja dari P89C138 adalah 1μs.

2.6 *Serial Electrically Erasable Programmable Read Only Memory* (SEEPROM)

Pada skripsi ini digunakan memori berupa EEPROM untuk menyimpan data dan EEPROM yang dipakai adalah serial EEPROM. EEPROM adalah ROM yang dapat dibaca dan ditulisi, dan bersifat *non volatile*, dimana data yang tersimpan tidak hilang walaupun tidak ada tegangan sumber. Serial EEPROM adalah EEPROM yang proses pembacaan dan penulisan datanya secara serial. IC SEEPROM yang



digunakan adalah 24LC65 yang terdiri dari 8 pin seperti terlihat pada Gambar 2.4.

Gambar 2.4 Konfigurasi pin Serial EEPROM 24LC65

Fungsi tiap-tiap pin dari 24LC65 SEEPROM adalah sebagai berikut :

- A0, A1, A2

Ketiga pin ini digunakan oleh *master* untuk memilih memori yang akan diakses. Jadi delapan 24LC65 dapat dihubungkan ke *bus* sesuai dengan jumlah kombinasi yang dapat dibentuk oleh A0, A1, dan A2.

- V_{CC} dan V_{SS}

V_{CC} adalah tegangan catu daya 5 Volt dan V_{SS} adalah *ground*. Nilai V_{CC} yang diijinkan agar memori masih mampu beroperasi adalah 2,5 sampai 6 Volt.

- SDA (*Serial Address*, Data *Input* atau *Output*)

Pin ini dapat dioperasikan secara *bidirectional* untuk mengirim alamat dan data dari dan ke memori.

- SCL (*Serial Clock*)

Digunakan untuk sinkronisasi pertukaran data dari dan ke memori.

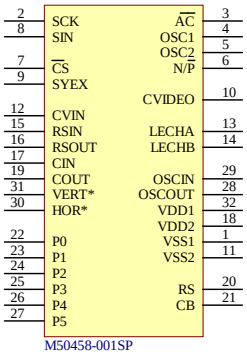
- NC (*No Connection*)

Pengalamatan IC 24LC65 adalah sebagai berikut:

- *Control byte* adalah byte pertama setelah kondisi *start*. *Control byte* terdiri dari 4 bit *control code*, 3 bit *device select* (A2, A1, A0), dan bit R/W. *Control code* 24LC65 untuk operasi baca dan tulis adalah 1010.
- Byte kedua setelah kondisi *start* adalah byte MSB yang menunjukkan alamat lokasi memori, byte selanjutnya adalah byte LSB dari alamat lokasi memori.
- Byte selanjutnya adalah byte data. Setelah satu byte data terkirim, alamat lokasi memori akan dinaikkan secara otomatis.

2.7 *On Screen Display Controller* (OSD)

Integrated Circuit (IC) *On Screen Display Controller* (OSD) adalah sebuah IC *television screen display control*, yakni untuk menampilkan karakter pada layar televisi. *Output* OSD berupa sinyal yang kompatibel video, karena itu keluaran dari OSD ini dapat dijadikan *input* untuk masukan video pada televisi.



Pa ıkan OSD *Controller* gurasi pin M50458-0(mbar 2.5.

Gambar 2.5 Konfigurasi pin OSD M50458-001SP

Tidak semua pin pada M50458-001SP digunakan dalam skripsi ini, karena itu hanya pin yang digunakan saja yang akan dijelaskan fungsinya.

- VSS dan VDD

Baik VSS1 maupun VSS2 merupakan *ground*, VDD1 dan VDD2 adalah tegangan catu daya 5 Volt.

- SCK (*Serial Clock input*)

Saat SCK berlogika '1' (*high*) maka data serial (SIN) diambil.

- AC (*Auto Clear input*)

Logika '0' (*low*) pada pin ini akan membuat *reset* IC.

- OSC1 dan OSC2

Pin ini berhubungan dengan rangkaian osilator eksternal. OSC1 untuk *input* sedangkan OSC2 untuk *output*.

- N/P (NTSC/PAL)

Pin ini menentukan sinkronisasi secara NTSC atau PAL.

- CS (*Chip Select input*)

Ketika pengiriman data serial sedang berlangsung maka pin ini *low*.

- SIN (*Serial Data input*)

Pin ini adalah masukan data serial atau alamat untuk *display control register* dan *display memory data*.

- SYEX (*Synchronization signal switching input*)

Saat *high*, pin ini menentukan sinkronisasi eksternal dan saat *low*, pin ini menentukan sinkronisasi internal.

- CVIDEO (*Composite Video signal output*)

Merupakan pin keluaran untuk *composite video signal*.

- LECHA (*Character level input*)

Ini adalah pin masukan yang menentukan level '*white*' karakter pada *composite video signal*.

- LECHB (*Blanking level input*)

Ini adalah pin masukan yang menentukan level '*black*' karakter dan level *blanking* pada *composite video signal*.

- OSCOUT dan OSCIN

Pin ini sebagai penghubung dengan rangkaian osilator eksternal untuk pembangkitan sinyal sinkronisasi. Osilasi 14,32 MHz diperlukan untuk NTSC dan 17,73 MHz diperlukan untuk PAL dengan menggunakan osilator kristal.

2.8 Televisi

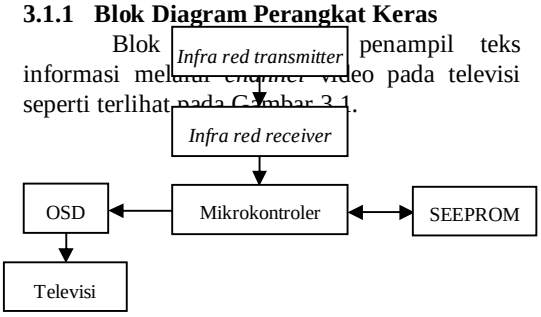
Televisi yang dapat digunakan dalam skripsi ini adalah televisi yang mempunyai masukan video (AV *in*). Masukan video ini berbeda dengan masukan RF, dimana pada masukan video, sinyal langsung diolah oleh video prosesor di dalam televisi sedangkan pada masukan RF, sinyal harus melalui bagian RF terlebih dahulu untuk didemodulasikan. Kualitas gambar yang melalui masukan video lebih baik dibanding yang lewat masukan RF, karena sinyal yang lewat masukan RF akan mengalami distorsi yang disebabkan oleh adanya proses modulasi dan demodulasi sinyal RF.

3. Perancangan Alat

3.1 Perancangan Perangkat Keras

3.1.1 Blok Diagram Perangkat Keras

Blok penampil teks informasi melalui *channel* video pada televisi seperti terlihat pada Gambar 3.1.



Gambar 3.1 Blok diagram alat

Cara kerja alat penampil teks informasi melalui *channel* video pada televisi adalah sebagai berikut :

Pengiriman teks informasi dilakukan dengan menggunakan infra merah, yakni tiap kali terjadi penekanan tombol pada *keypad remote controller*. Infra merah yang dipancarkan diterima oleh penerima infra merah. Data dari penerima infra merah ini didekodekan oleh mikrokontroler. Hasil pengolahan mikrokontroler yang berupa data teks informasi dijadikan masukan untuk IC *On Screen Display Controller* (OSD), kemudian mikrokontroler memerintahkan OSD menampilkan teks informasi tersebut melalui *channel* video pada televisi. Data teks informasi disimpan pada SEEPROM agar dapat ditampilkan kembali.

3.1.2 *Infra Red Transmitter* dan *Infra Red Receiver*

Sebagai *infra red transmitter* digunakan *remote controller*. *Remote controller* yang dipakai dikendalikan oleh mikrokontroler 4 bit buatan Mitsubishi yakni M50560-003P. Sedangkan *infra red receiver* yang dipakai pada skripsi ini adalah EVERLIGHT yang mempunyai nomor tipe IRM 8110. Kemudian dilakukan pengkodean data keluaran dari mikrokontroler *remote controller*, yang dipancarkan oleh *infra red transmitter* dalam bentuk frekuensi. Data yang dipancarkan oleh *infra red transmitter* ini sama dengan data yang diterima oleh *infra red receiver*. Hal ini dimungkinkan karena *infra red receiver* tersebut merupakan tranduser frekuensi ke amplitudo tegangan, sehingga tegangan keluaran dari *infra red receiver* ini sama dengan tegangan keluaran dari mikrokontroler *remote controller* yang diumpankan ke *infra red transmitter*. Data di sini berupa serangkaian tegangan 5V atau logika '1' (*high*) dan 0V atau logika '0' (*low*).

3.1.3 Rangkaian Mikrokontroler P89C138

Untuk pembangkit *clock* internal digunakan osilator kristal senilai 12 MHz (tipikal) sebagai *input* (masukan) untuk pin XTAL1 dan XTAL2. Osilator kristal ini juga terhubung dengan kapasitor C1 dan C2 yang bernilai 22 pF sesuai dengan karakteristik XTAL1.

Pin EA/VPP dibuat selalu *high* dengan diberi masukan VCC, agar saat terjadi *reset*, *Central Processing Unit* (CPU) dari mikrokontroler mengeksekusi perintah yang ada pada *internal memory* ROM. Rangkaian *reset* mikrokontroler berupa *power-on reset* yang dapat melakukan *reset* secara otomatis ketika catu daya dihidupkan (*on*). Pin RESET dihubungkan dengan VCC melalui sebuah kapasitor dan diberi R *pull down*. Saat catu daya *on*, tegangan pada pin RESET sama dengan VCC dikurangi tegangan kapasitor, dan semakin berkurang dari VCC sebagaimana kapasitor termuati melalui resistor internal ke *ground*. Dimana semakin besar kapasitor maka semakin lambat tegangan *reset* berkurang. Jadi setiap kali catu daya dihidupkan (sebelumnya

mati), mula-mula terjadi *reset* akibat transisi dari *low* ke *high* pada pin RESET, tetapi beberapa milidetik kemudian tidak terjadi *reset* karena pin RESET mendapat logika *low* kembali.

Pin P1.0 dioperasikan sebagai masukan dari rangkaian *infra red receiver*. Pin ini yang digunakan oleh mikrokontroler untuk mendeteksi adanya penekanan tombol *remote control* atau tidak. Selanjutnya masukan pada pin P1.0 ini diolah dengan *software* (perangkat lunak) untuk mengetahui kode dari tombol yang ditekan.

Pin P0.0 dan P0.1 dioperasikan sebagai *output* (keluaran) untuk rangkaian SEEPROM, di mana pin P0.0 sebagai masukan untuk pin SDA (*Serial Data*) dan pin P0.1 sebagai masukan untuk pin SCL (*Serial Clock*).

Pin P2.0, P2.1 dan P2.2 dioperasikan sebagai *output* (keluaran) untuk rangkaian OSD, di mana pin P2.0 sebagai masukan untuk pin SCK (*Serial Clock*), pin P2.1 sebagai masukan untuk pin SIN (*Serial Input*) dan pin P2.2 sebagai masukan untuk pin CS (*Chip Select*). Pin P2.1 ini memberikan data serial pada OSD, data inilah yang kemudian ditampilkan ke televisi sebagai teks informasi. Pin P2.0 memberikan serangkaian pulsa *clock* untuk sinkronisasi penerimaan data dari pin P2.1 pada OSD, sedangkan pin P2.2 untuk mengaktifkan OSD atau menonaktifkan OSD.

3.1.4 SEEPROM 24LC65

Pada skripsi ini hanya digunakan sebuah SEEPROM sebagai tempat penyimpanan teks informasi. Data teks informasi disimpan dalam SEEPROM agar dapat ditampilkan kembali. Tiga bit *device select* (A0,A1,A2) dibuat *low*, karena hanya satu SEEPROM yang dihubungkan dengan Mikrokontroler.

3.1.5 *On Screen Display Controller* M50458-001SP

Pada skripsi ini digunakan OSD *Controller* M50458-001SP untuk menampilkan seluruh karakter dari teks informasi yang diinginkan pada layar televisi. Pin SCK merupakan masukan dari P2.0 dan digunakan sebagai bus untuk *clock*. Pin SDA merupakan masukan dari P2.1 dan digunakan sebagai bus untuk data. Pin CS merupakan masukan dari P2.2, *low* pada pin ini akan mengaktifkan IC OSD.

Pin AC diset *low* supaya bila terjadi reset, semua alamat pada *display control register* diset 0. Pin OSC1 dan OSC2 dihubungkan dengan rangkaian osilator LC dengan nilai L = 22μH dan C = 15pF, sehingga frekuensi osilasi yang dihasilkan sekitar 7 MHz.

Pin N/P diset *low* karena menggunakan mode PAL. Pin CVIDEO adalah pin keluaran untuk *composite video signal*. Keluaran dari pin ini dihubungkan dengan rangkaian penguat transistor *common collector* dengan tujuan agar diperoleh impedansi output yang rendah, sehingga semua sinyal video komposit dapat diteruskan ke masukan video dari televisi.

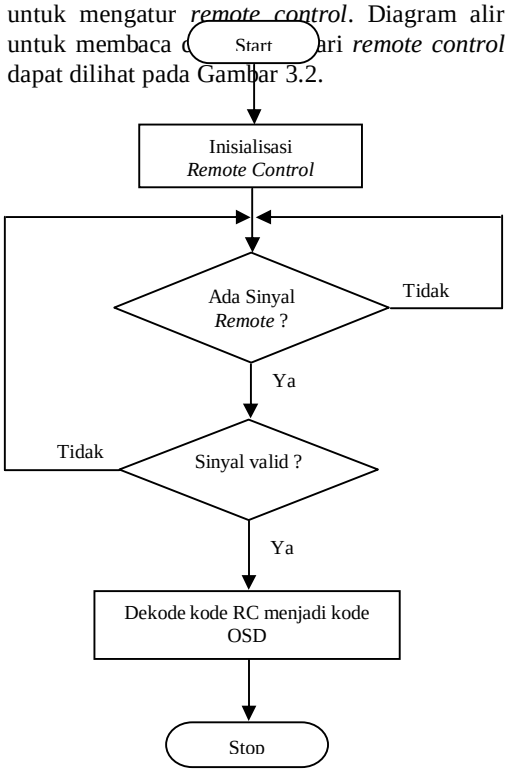
LECHA digunakan untuk menentukan tingkat warna putih dari karakter yang ditampilkan, sedangkan LECHB digunakan untuk menentukan tingkat warna hitam dari karakter yang ditampilkan. Pin OSCIN dan OSCOUT merupakan pin masukan dan keluaran untuk osilator eksternal yang digunakan untuk sinkronisasi sinyal. Osilator ini dibentuk dari keramik resonator dengan frekuensi osilasi sebesar 17,73 MHz (untuk mode PAL) dan kapasitor 12 pF.

3.2 Perancangan Perangkat Lunak

Perancangan perangkat lunak pada skripsi ini menggunakan bahasa pemrograman untuk mikrokontroler keluarga MCS 51. Perangkat lunak yang dirancang terdiri dari dua program, yaitu program pengaturan *remote control* dan program utama yang berisi pilihan menu.

3.2.1 Program *Remote Control*

Sebelum perancangan program utama, perlu terlebih dahulu dirancang perangkat lunak untuk mengatur *remote control*. Diagram alir untuk membaca data dari *remote control* dapat dilihat pada Gambar 3.2.



Gambar 3.2 Diagram alir program pengaturan *remote control*

Pada program *remote control*, ketika start rutin program membaca inisialisasi sinyal *remote control* yang ditentukan.

```
;*************************************
******;
; RemoteInit untuk inisialisasi sinyal
remote  ;
;*************************************
******;
RemoteInit:
              MOV     TMOD,#02H
              MOV     TL0,#55
              SETB    TR0
           CLR     EverInput
              CLR     IsRemoteData
           CLR     RemoteNotReleasedYet
           CLR     RepeatPermission
              SETB    RemoteInput
              RET
```

Bit TR0 diberi logika ‘1’ sehingga *timer* yang digunakan adalah *Timer0*. Kemudian register *timer* mode (TMOD) sebagai pengontrol pemilihan mode operasi *timer/counter* diberi nilai 02H, yang berarti *Timer0* dioperasikan sebagai *timer* 8 bit *auto-reload* (mode 2). *Timer* ini dijalankan dengan periode waktu sebesar 200μs karena TL0 diberi nilai 55.

Bit EverInput, IsRemoteData, RemoteNotReleasedYet, RepeatPermission diberi nilai ‘0’ (*clear*). Bit-bit ini berperan dalam pengolahan sinyal *remote control*. Bit RemoteInput (P1.0) diberi nilai ‘1’ (*set*), sehingga sistem siap menerima data sinyal dari rangkaian *infra red receiver*.

Selanjutnya masuk ke rutin program Timer0Routine yang digunakan untuk mengecek ada tidaknya sinyal *remote control* dan juga untuk memeriksa kevalidan sinyal, yakni panjang *header* dan *separator*.

```
;*************************************
******;
; Timer0Routine untuk cek sinyal remote
;
;*************************************
******;
Timer0Routine:
              PUSH    ACC
              PUSH    PSW
              PUSH    B
           MOV     B,R1
              PUSH    B
              PUSH    DPH
              PUSH    DPL
           JB      EverInput,T0_Next
           JB      RemoteInput,T0_Finish
           SETB    EverInput
              SETB    HeaderFlag
              SETB    FirstHalfFlag
              CLR     SecondHalfFlag
              CLR     SeparatorFlag
         SETB    FirstDataFlag
         MOV     RemoteCount,#0

T0_Next:
          INC    RemoteCount
          JB
HeaderFlag,T0_Check_Header_
          JB
SeparatorFlag,T0_Check_Separator_
          CALL   GetRemoteData
          SJMP   T0_Finish

T0_Check_Header_:
               CALL    CheckRemoteHeader
               SJMP    T0_Finish

T0_Check_Separator_:
               CALL
CheckRemoteSeparator

T0_Finish:
               POP     DPL
               POP     DPH
               POP     B
               MOV     R1,B
               POP     B
               POP     PSW
               POP     ACC
               RETI
```

Jika ada sinyal *remote control* maka nilai dari bit EverInput akan *set*, sehingga program masuk ke subrutin T0_Next. Jika tidak ada sinyal *remote control* maka program kembali ke awal rutin Timer0Routine. Selanjutnya pada subrutin T0_Next, program memeriksa kevalidan sinyal termasuk panjang *header* dan *separator*. Jika sinyal tidak valid maka program kembali ke awal rutin Timer0Routine. Sedangkan jika sinyal valid maka program memanggil rutin GetRemoteData untuk membaca data *remote* kemudian melakukan pengkodean data dari kode-kode yang dipancarkan oleh *remote control* menjadi kode-kode yang akan ditampilkan oleh OSD melalui rutin Remote2GeneralCode.

3.2.2 Program Utama

Program utama terdiri dari 4 pilihan menu, yaitu pesan baru, ubah pesan, hapus pesan, dan lihat pesan. Pesan disini maksudnya adalah informasi-informasi yang akan ditampilkan pada televisi berupa teks informasi.

Program utama berfungsi untuk inisialisasi awal mikrokontroler dan OSD, mengatur tampilan judul dan menu utama, meminta masukan dan mengarahkan masukan ke salah satu menu.

```
;*************************************
******;
; Inisialisasi Mikrokontroler
;
;*************************************
******;
Start:
```

```
MOV   IE,#82H
   MOV     P1,#FFH
   MOV     P2,#FFH
   MOV     P3,#00H
   MOV     P0,#FFH
MOV   SP,#07H
```

Pada saat Start, register *interrupt enable* (IE) diberi nilai 82H, yang artinya *Timer0* dapat menerima interupsi. Port mikrokontroler yang digunakan dalam perancangan (P0, P1, dan P2) diberi nilai FFH, sedangkan port yang tidak digunakan (P3) diberi nilai 00H. Lalu yang terakhir, register *stack pointer* (SP) diberi nilai 07H.

Setelah inisialisasi mikrokontroler, program membaca inisialisasi OSD yang berupa inisialisasi *display control register*, meliputi posisi awal dan ukuran karakter secara horisontal maupun vertikal.

```
;*************************************
************;
; OSDInit untuk inisialisasi display
control register ;
;*************************************
************;
OSDInit:
               SETB    OSD_CS
               SETB    OSD_SIN
               SETB    OSD_SCK
                MOV     A,#20H
                MOV     B,#01H
               CALL    OSDSendAddress
                MOV     A,#0
                MOV     B,#00001010B
               CALL    OSDSendData
                MOV     A,#01011001B
                MOV     B,#00010000B
               CALL    OSDSendData
                MOV     A,#01001100B
                MOV     B,#00000000B
               CALL    OSDSendData
                MOV     A,#10100000B
                MOV     B,#00000000B
               CALL    OSDSendData
                MOV     A,#00000001B
                MOV     B,#00000000B
               CALL    OSDSendData
                MOV     A,#00010010B
                MOV     PatternDisplay,A
                MOV     B,#00000001B
               CALL    OSDSendData
                MOV
A,#RASTER_MASK_LSB
                ORL     A,#OSD_BLACK
                MOV
B,#RASTER_MASK_MSB
               CALL    OSDSendData
                MOV     A,#01110101B
                MOV     B,#0H
               CALL    OSDSendData

               OSD_FINISH_WRITE
                  RET
```

Rutin program OSDInit dimulai dengan memberi nilai ‘1’ (*set*) pada bit OSD_CS, OSD_SIN, dan OSD_SCK, yang bertujuan membuat OSD dalam keadaan siap untuk menerima masukan data serial dari mikrokontroler. Kemudian program memanggil rutin OSDSendAddress untuk mengirim *address* 16 bit ke OSD. *Address* 16 bit awal yang dikirim ke OSD adalah 0120H yang diperoleh dari nilai pada akumulator (20H) dan register B (01H). Jadi akumulator berperan sebagai *first byte* (*Least Significant Byte*) dan register B berperan sebagai *second byte* (*Most Significant Byte*). Setelah alamat terkirim ke OSD maka program mulai mengirimkan data-data 16 bit yang ada pada akumulator (LSB) dan register B (MSB) dengan memanggil rutin OSDSendData. Data-data inilah yang memberikan inisialisasi pada *display control register* pada OSD.

```
;***********************************;
; MenuManager untuk mengatur menu    ;
;***********************************;
MenuManager:
                JNB
SwitchState,MM_Begin
                CLR     SwitchState
                MOV     ActiveColorSet,#0

               CALL    OSDSetRasterColor
               CALL    OSDSetReverseMode
               CALL    OSDClearScreen
               CALL    DisplayTitle
               CALL    DisplayMainMenu
```

Pada rutin MenuManager, program memanggil rutin OSDClearScreen, DisplayTitle, dan DisplayMainMenu untuk membersihkan layar, menampilkan judul, dan menampilkan menu utama. Selanjutnya jika bit SwitchState bernilai ‘0’ maka program masuk ke subrutin MM_Begin.

```
MM_Begin:
                JB
AnyGeneralKey,MM_CheckKey
                JMP     MM_Exit
```

Jika bit AnyGeneralKey bernilai ‘1’ atau ada sinyal *remote control* maka program masuk ke subrutin MM_CheckKey untuk mengecek kevalidan sinyal dan data sinyal. Setelah data sinyal valid diperoleh maka untuk mengatur dan mengarahkan input ke menu yang sesuai diperlukan rutin Manager.

```
;*************************************
**************;
; Manager untuk mengatur dan mengarahkan
input ke menu  ;
;*************************************
**************;
Manager:
                MOV     A,State
                RL      A
                MOV
DPTR,#MENU_JUMP_TABLE
                JMP     @A+DPTR

MENU_JUMP_TABLE:
                   AJMP    MenuManager
               AJMP    PesanBaruManager
               AJMP    UbahPesanManager
               AJMP    HapusPesanManager
               AJMP    LihatPesanManager
```

Nilai pada State (1, 2, 3, atau 4) mengarahkan program untuk masuk ke salah

satu menu yang ada pada menu utama, yaitu pesan baru (1), ubah pesan (2), hapus pesan (3), atau lihat pesan (4).

3.2.2.1 Pesan Baru

Menu ini berfungsi untuk menulis informasi baru. Ketika State bernilai 1, program masuk ke rutin PesanBaruManager.

```
;*************************************
******;
; PesanBaruManager untuk membuat halaman
baru  ;
;*************************************
******;
PesanBaruManager:
       JNB     SwitchState,PBM_BEGIN
       CLR     SwitchState
        CALL    OSDSetCursorMode
        CALL    OSDClearScreen
       MOV     A,#200
        CALL    Delay
       CALL    DisplayTitle
        CALL    FindBlankPage
       JNB   ErrorFlag,PBM_PrepareNewPage
        CALL    DisplayError
       SJMP    PBM_BEGIN

   PBM_PrepareNewPage:
         MOV   ShiftState,#SHIFT_INACTIVE
         MOV   CurrentLine,#Message_pos
             MOV     CurrentColumn,#0
             MOV     NamePage,#0
            CALL    DisplayMessage
            CALL
DisplayCurrentNamePage
            SETB    CapsLock
             MOV
CurrentLine,#Message_pos
             MOV     CurrentColumn,#0
            SETB    C
            CALL    DisplaySetCursor
            CALL    DisplayAllStatus
```

Pada saat masuk ke rutin PesanBaruManager, bit SwitchState sudah bernilai ‘1’ (*set*) sehingga program tidak masuk ke subrutin PBM_BEGIN. Kemudian bit SwitchState diberi nilai ‘0’ (*clear*) agar dapat dibuat *set* kembali saat ada sinyal input ke menu. Setelah layar dibersihkan dan judul ditampilkan maka program mulai mencari halaman kosong untuk penulisan informasi dengan memanggil rutin FindBlankPage. Halaman kosong disini maksudnya adalah adakah tempat tersisa pada SEEPROM untuk penyimpanan informasi. Jika ada halaman kosong maka bit ErrorFlag akan *clear* sehingga program masuk ke subrutin PBM_PrepareNewPage untuk menyiapkan halaman baru yang akan diisi data teks informasi sesuai dengan data masukan dari *remote control*. Informasi akan otomatis tersimpan di memori saat pengisian telah selesai. Jika tidak ada halaman kosong maka bit ErrorFlag akan *set* sehingga program masuk ke rutin DisplayError untuk menampilkan pemberitahuan kesalahan di layar televisi. Untuk kembali ke menu utama maka dilakukan penekanan tombol menu.

3.2.2.2 Ubah Pesan

Menu ini berfungsi untuk melakukan pengubahan informasi. Program masuk ke menu ubah pesan (rutin UbahPesanManager) ketika State bernilai 2.

```
;*************************************
*;
; UbahPesanManager untuk mengedit page
;
;*************************************
*;
UbahPesanManager:
            JNB     SwitchState,UPM_BEGIN
            SETB    Browsing
             CLR     ErrorFlag
             CLR
AlreadyGotTheChoosenPage
           CALL    OSDClearScreen
           CALL    DisplayTitle
           CALL    BrowsePage
            CLR     SwitchState
           CJNE    A,#BP_NOPAGE,UPM_BEGIN
           SETB    ErrorFlag
            MOV     ErrorCode,#ERR_NOPAGE
           CALL    DisplayError

   UPM_BEGIN:
         JNB   ErrorFlag,UPM_BeginAsUsual
         JNB   AnyGeneralKey,UPM_Exit
          CLR   AnyGeneralKey
         SJMP  UPM_BackToMenu
```

Pada saat masuk ke rutin UbahPesanManager, bit SwitchState sudah bernilai ‘1’ (set) sehingga program tidak masuk ke subrutin UPM_BEGIN. Bit Browsing dibuat set sedangkan bit ErrorFlag dan AlreadyGotTheChoosenPage dibuat clear. Setelah layar dibersihkan dan judul ditampilkan maka program mulai melihat halaman yang ada dengan memanggil rutin BrowsePage. Bit SwitchState diberi nilai ‘0’ (clear) agar dapat dibuat set kembali saat ada sinyal input ke menu. Kemudian program mengecek isi akumulator dengan membandingkan nilai pada akumulator dengan nilai 0. Jika akumulator bernilai 0 atau semua halaman kosong (tidak ada informasi) maka bit ErrorFlag dibuat set dan program memanggil rutin DisplayError untuk menampilkan pemberitahuan kesalahan di layar televisi. Sedangkan jika akumulator tidak bernilai 0 atau ada halaman yang berisi informasi maka program memanggil subrutin UPM_BEGIN untuk mengubah informasi. Pengubahan informasi dilakukan dengan terlebih dahulu memilih informasi yang akan diubah dengan tombol kiri atau kanan. Jika tidak ada informasi yang akan diubah, program akan kembali ke menu utama. Jika pengubahan informasi telah selesai maka informasi tersebut

akan disimpan di memori lalu kembali ke menu utama.

3.2.2.3 Hapus Pesan

Menu ini berfungsi untuk menghapus informasi. Program masuk ke menu hapus pesan (rutin HapusPesanManager) ketika State bernilai 3.

```
;*************************************
******;
; HapusPesanManager untuk menghapus
halaman    ;
;*************************************
******;
HapusPesanManager:
        JNB     SwitchState,HPM_Begin
        SETB    Browsing
         CLR     ErrorFlag
        CALL    OSDClearScreen
        CALL    DisplayTitle
        CALL    BrowsePage
         CLR     SwitchState
        CJNE    A,#BP_NOPAGE,HPM_Begin
        SETB    ErrorFlag
         MOV     ErrorCode,#ERR_NOPAGE
        CALL    DisplayError

   HPM_Begin:
       JNB     ErrorFlag,HPM_BeginAsUsual
       JNB     AnyGeneralKey,HPM_Exit
        CLR     AnyGeneralKey
       SJMP    HPM_BackToMenu
```

Pada saat masuk ke rutin HapusPesanManager, bit SwitchState sudah bernilai ‘1’ (set) sehingga program tidak masuk ke subrutin HPM_BEGIN. Bit Browsing dibuat set sedangkan bit ErrorFlag dibuat clear. Setelah layar dibersihkan dan judul ditampilkan maka program mulai melihat halaman yang ada dengan memanggil rutin BrowsePage. Bit SwitchState diberi nilai ‘0’ (clear) agar dapat dibuat set kembali saat ada sinyal input ke menu. Kemudian program mengecek isi akumulator dengan membandingkan nilai pada akumulator dengan nilai 0. Jika akumulator bernilai 0 atau semua halaman kosong (tidak ada informasi) maka bit ErrorFlag dibuat set dan program memanggil rutin DisplayError untuk menampilkan pemberitahuan kesalahan di layar televisi. Sedangkan jika akumulator tidak bernilai 0 atau ada halaman yang berisi informasi maka program memanggil subrutin HPM_BEGIN untuk menghapus informasi. Penghapusan informasi dilakukan dengan terlebih dahulu memilih informasi yang akan dihapus. Jika tidak ada informasi yang akan dihapus, program akan kembali ke pilihan menu. Jika ada informasi yang akan dihapus, informasi tersebut akan dihapus dari memori saat penekanan tombol Enter.

3.2.2.4 Lihat Pesan

Menu ini berfungsi untuk melihat informasi-informasi yang ada. Program masuk ke menu lihat pesan (rutin LihatPesanManager) ketika State bernilai 4.

```
;*************************************
********;
; LihatPesanManager untuk melihat halaman
yg ada ;
;*************************************
********;
LihatPesanManager:
       JNB     SwitchState,LPM_Begin
       SETB    Browsing
        CLR     ErrorFlag
       CALL    OSDClearScreen
       CALL    DisplayTitle
       CALL    BrowsePage
        CLR     SwitchState
       CJNE    A,#BP_NOPAGE,LPM_Begin
       SETB    ErrorFlag
        MOV     ErrorCode,#ERR_NOPAGE
       CALL    DisplayError
```

Pada saat masuk ke rutin LihatPesanManager, bit SwitchState sudah bernilai ‘1’ (*set*) sehingga program tidak masuk ke subrutin LPM_BEGIN. Bit Browsing dibuat *set* sedangkan bit ErrorFlag dibuat *clear*. Setelah layar dibersihkan dan judul ditampilkan maka program memanggil rutin BrowsePage. Bit SwitchState diberi nilai ‘0’ (*clear*). Kemudian program mengecek isi akumulator. Jika akumulator bernilai 0 atau semua halaman kosong maka bit ErrorFlag dibuat *set* dan program memanggil rutin DisplayError. Sedangkan jika akumulator tidak bernilai 0 atau ada halaman yang berisi informasi maka program memanggil subrutin LPM_BEGIN untuk melihat informasi. Jika sudah selesai, program akan kembali ke pilihan menu dengan menekan tombol enter.

4. Kesimpulan

Setelah menyelesaikan perancangan alat penampil pesan berupa teks informasi melalui *channel* video pada televisi, maka beberapa hal dapat disimpulkan sebagai berikut :

1. Penggunaan *remote control* lain tidak dapat digunakan bila format data *remote control* tidak sama. Format data di sini meliputi *header*, *custom code*, *separator* dan *data code*.
2. Tampilan pesan yang meliputi ukuran karakter, warna karakter, dan warna latar dapat diatur dengan perangkat lunak.
3. Jumlah karakter yang dapat ditampilkan dalam satu baris maksimal 24 karakter.
4. Jenis televisi yang dapat digunakan untuk menampilkan pesan adalah

televisi yang mempunyai masukan berupa video (*Video In*).

5. Jarak *infra red transmitter* ke *infra red receiver* dimana sinyal *remote control* masih dapat diterima dengan baik maksimal 5 meter.

Daftar Pustaka

1. Moh. Ibnu Malik dan Anwardi, *...nen dengan Mikrokontroler 8031*, ...edia Komputindo, Jakarta, 1997
2. *24LC65 Data Sheet*, Microchip ... Inc., Chandler-AZ, 1996
3. *...8xCx38 Data Sheet*, Philips ...ctors, 1998
4. Mitsubishi, *M50458-XXXSP/FP Data Sheet*, Mitsubishi Microcomputers

Andi Kristianto (L2F 097 609)
Mahasiswa Jurusan Teknik Elektro Telkom Universitas Diponegoro Semarang.

Menyetujui,
Dosen Pembimbing II

Sumardi, ST, MT
NIP. 132 125 670