MASTER INDEX BACK to TOC HELP

III. PERENCANAAN

1. PENDAHULUAN

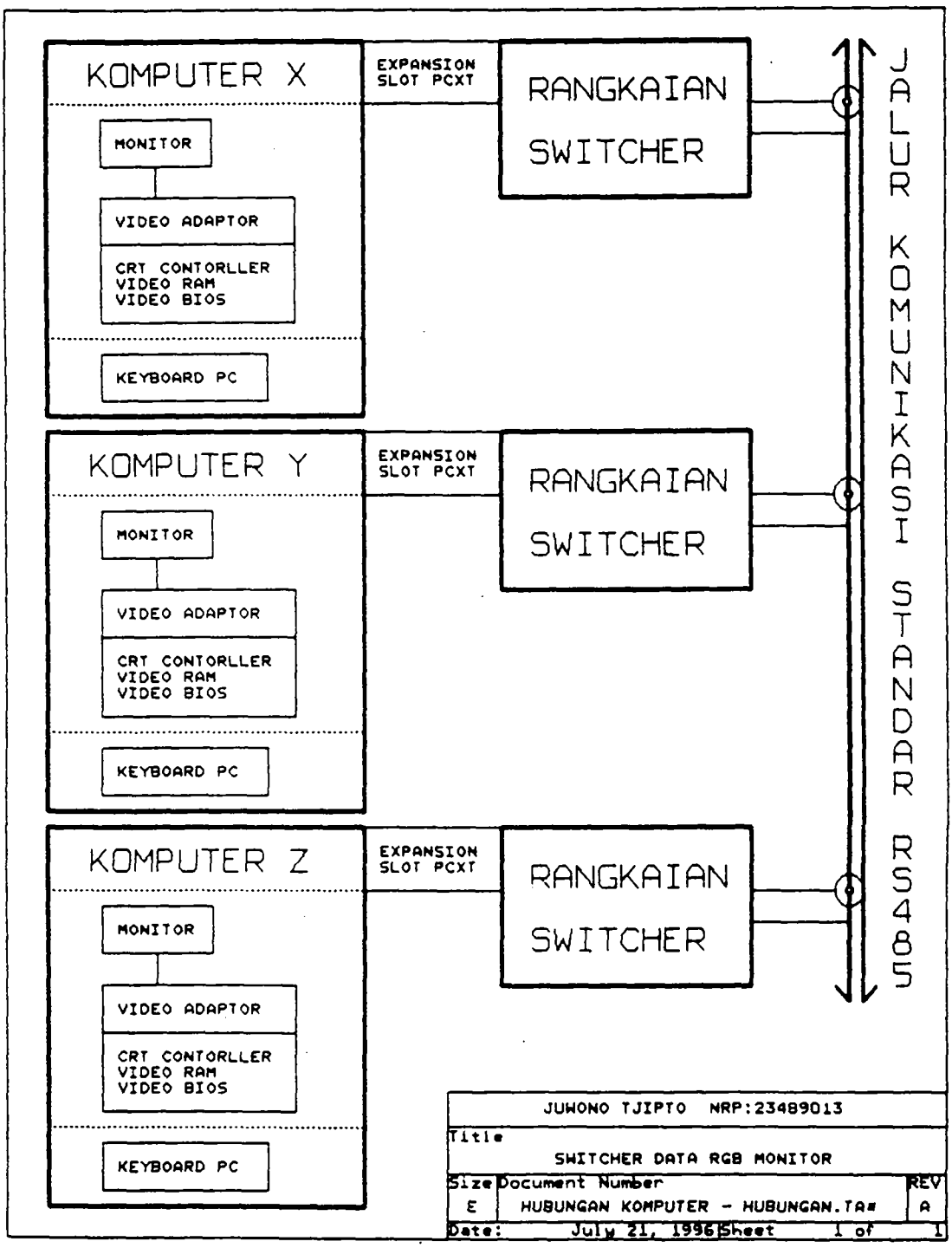
Perencanaan dan Pembuatan *SWITCHER Data RGB MONITOR* mempergunakan fasilitas komunikasi data serial yang mampu melakukan hubungan secara multipoint yaitu standar EIA-RS485. Fasilitas komunikasi data serial ini dimanfaatkan sebagai switcher berdasarkan kemudahan yang dapat diperoleh terutama dalam melakukan/menentukan konfigurasi dari hubungan tiap komputer yang dihubungkan melalui switcher dalam suatu network. Kemudahan ini didapat karena bila terjadi perubahan dalam network misalnya terjadi penambahan atau pengurangan komputer maka dengan melakukan perubahan pada konfigurasi program, network akan dengan mudah mengenali apakah terdapat penambahan atau pengurangan komputer dalam network. Dengan demikian switcher ini merupakan gabungan antara hardware yang berupa rangkaian switcher itu sendiri dan software berupa program pengendali switcher yang akan memudahkan pemakai bila terjadi penambahan/ pengurangan komputer yang dipakai.

Pemakaian software pada switcher sebagai program pengendali dalam pembuatan tugas akhir ini didasarkan pada kesamaan letak dan besar memory yang dipakai dalam video memory komputer PC dalam mode teks. Dimana dalam pembuatan software yang dibuat secara resident

untuk dapat menampilan sacara maksimal (satu layar penuh) mode teks berwarna 25 baris x 80 kolom yang dipergunakan memory maksimal sebesar 4Kbyte. Video memory ini yang dalam pelaksanaan proses switching akan dikirim antar komputer melalui jalur komunikasi RS485. Dalam proses pengiriman karakter hasil penekanan tombol keyboard juga diberlakukan proses yang sama hanya yang dikirim adalah karakter dan posisi kursor. Selama proses ini berlangsung, keyboard PC yang terhubung pada komputer yang menerima data karakter dan posisi kursor akan berada pada kondisi idle, karena penekanan tombol pada keyboard tidak akan memberikan reaksi pada komputer.

Dalam tugas akhir ini pembuatan switcher dibatasi sebanyak **tiga** buah dan memiliki fungsi yang sama untuk setiap station yaitu hardware dari standar komunikasi data serial EIA-RS485 dan pembedaan dilakukan secara software dimana terdapat station yang berfungsi sebagai supervisor dan station yang lain diperlakukan sebagai sebagai user. Supervisor memiliki kemampuan untuk dapat berhubungan dengan semua user sedangkan user dibatasi hanya dapat melakukan hubungan dengan supervisor dan tidak dengan user lainnya.

Hubungan antar komputer ini dapat dilihat pada Gambar 3-1, dimana pada gambar tersebut memperlihatkan bagan/diagram yang menyatakan adanya dua macam hubungan fisik yang terjadi dalam pembuatan rangkaian *Switcher Data RGB Monitor.*

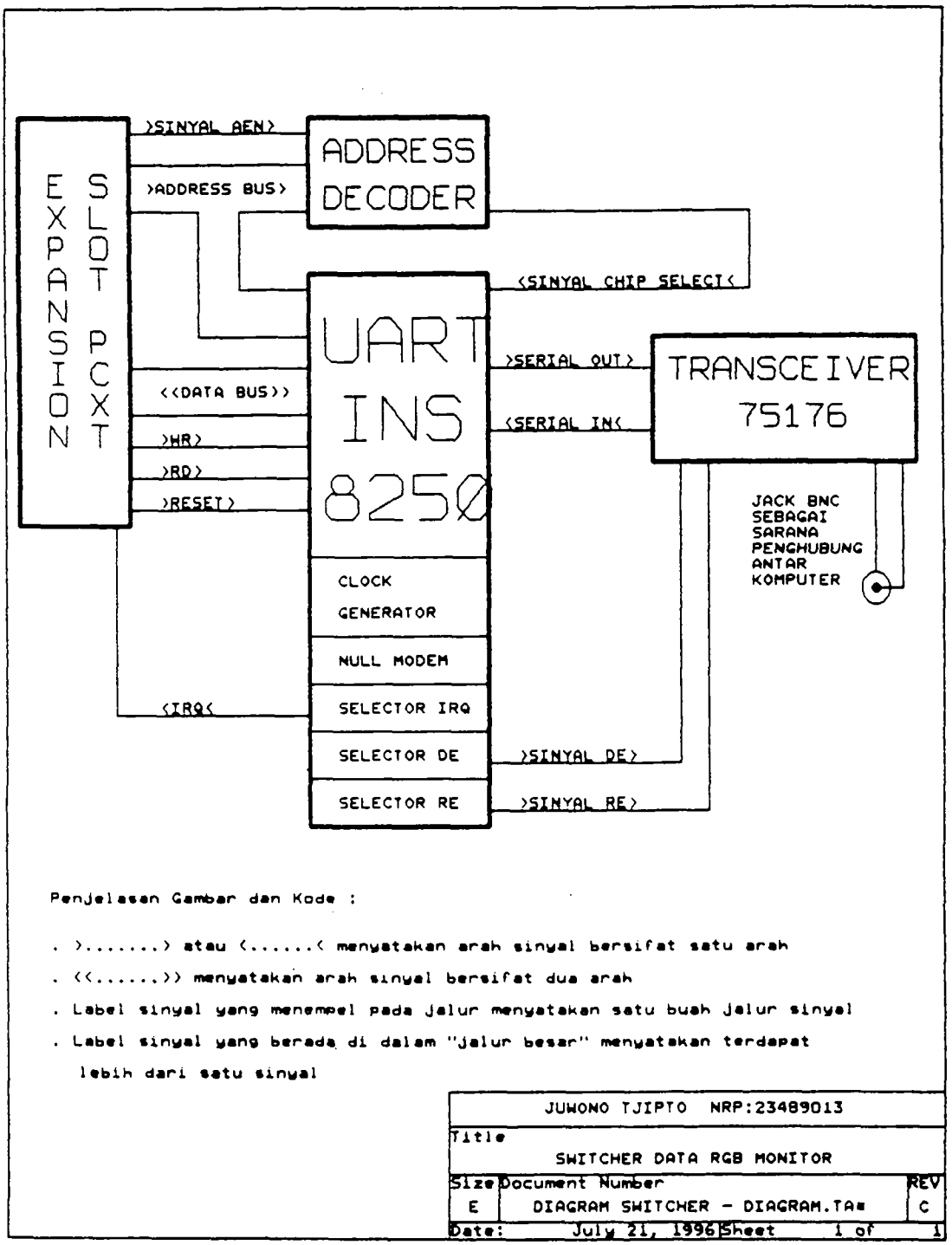


GAMBAR 3-1

DIAGRAM HUBUNGAN ANTAR STATION/KOMPUTER

Hubungan fisik yang pertama adalah hubungan yang terjadi antara tiap komputer dengan rangkaian switcher melalui expansion slot PC-XT, sedangkan hubungan fisik kedua merupakan hubungan yang terjadi antara rangkaian switcher yang terpasang pada sebuah komputer dengan rangkaian switcher lainnya yang juga terpasang pada sebuah komputer melalui jalur komunikasi dengan standar RS-485. Sehingga melalui jalur komunikasi ini antara tiap komputer dapat berhubungan satu dengan lainnya.

Pada gambar 3-2 terlihat diagram dari rangkaian switcher yang dibuat untuk Tugas Akhir ini, dalam diagram terlihat sebuah blok besar ditengah yang merupakan bagian penting dari switcher ini yaitu UART INS-8250 yang menjadi pengendali dari switcher. Dalam Blok UART INS-8250 terdapat beberapa pembagian yang merupakan rangkaian penunjang dan berhubungan langsung dengan UART. Blok Null Modem dan blok selektor dalam bentuk fisik berupa rangkaian jumper untuk menentukan konfigurasi dari INS-8250, sedangkan Blok Clock Generator merupakan rangkaian dengan basis kristal yang dipergunakan untuk menghasilkan clock bagi INS-8250. Blok Transceiver 75176 merupakan sebuah rangkaian konverter yang akan mengubah sinyal standar TTL yang dihasilkan oleh Blok UART INS-8250 menjadi sinyal dengan bentuk diferensial yang sesuai untuk dipergunakan pada komunikasi standar RS-485 dari rangkaian switcher ini.



Penjelasan Gambar dan Kode :

. >.....> atau <.....< menyatakan arah sinyal bersifat satu arah
. <<.....>> menyatakan arah sinyal bersifat dua arah
. Label sinyal yang menempel pada jalur menyatakan satu buah jalur sinyal
. Label sinyal yang berada di dalam "jalur besar" menyatakan terdapat lebih dari satu sinyal

JUWONO TJIPTO NRP:23489013		
Title: SWITCHER DATA RGB MONITOR		
Size: E	Document Number: DIAGRAM SWITCHER - DIAGRAM.TA#	REV: C
Date: July 21, 1996	Sheet 1 of 1	

GAMBAR 3-2

DIAGRAM BLOK RANGKAIAN SWITCHER

Blok Address Decoder merupakan rangkaian yang yang berfungsi melakukan pemilihan waktu mengaktifkan INS-8250 sehingga tidak terjadi bentrok bila komputer sedang melakukan suatu operasi atau menghubungi device lain yang berhubungan juga dengan komputer. Blok yang terakhir merupakan suatu expansion slot PC-XT yang memungkinkan rangkaian switcher ini dapat berhubungan dengan komputer melalui sebuah konektor/slot standar PC-XT sehingga sinyal-sinyal yang bersesuaian dari komputer ke rangkaian switcher maupun sebaliknya dapat terhubung dengan baik.

2. PERENCANAAN HARDWARE

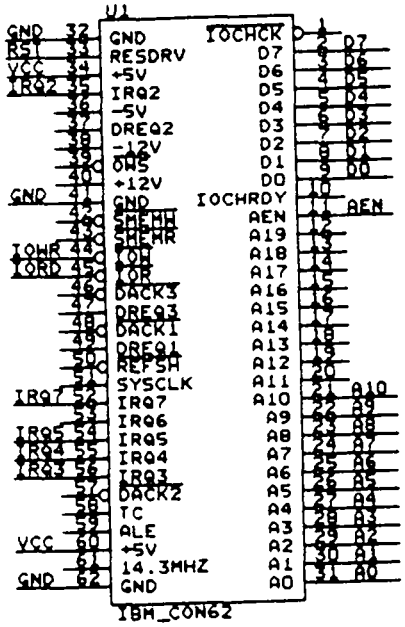
2.1 Rangkaian yang berhubungan dengan expansion slot

Rangkaian hanya berupa konektor yang dibentuk pada PCB dalam bentuk khusus sehingga dapat dimasukkan pada expansion slot jenis PC-XT (8 bit). Konfigurasi dan posisi pin yang dipergunakan dalam rangkaian switcher ini seperti pada gambar 3-3 terlihat bahwa pin yang dipergunakan adalah :

a. pin D0 sampai D7 yang akan dihubungkan dengan INS-8250 sebagai jalur data bus
b. pin A0 sampai A10 dimana pin A0, pin A1 dan pin A2 dihubungkan dengan INS-8250 untuk melakukan pemilihan alamat port sedangkan pin A3 sampai A10 dihubungkan dengan 74LS688 untuk mengalamati INS-8250 sendiri.

c. pin AEN dihubungkan dengan pin enable dari 74LS688 yang berguna untuk menghindari terjadi bentrokan pemakaian jalur data antara INS-8250 dengan DMA controller dimana kondisi AEN low berarti jalur data siap dipakai.

d. pin RESET dihubungkan dengan pin MR pada INS-8250 sehingga INS-8250 dapat melakukan operasi reset bersamaan dengan operasi reset/booting dari komputer.



GAMBAR 3-3

RANGKAIAN EXPANSION SLOT

e. pin IRQ2, IRQ3, IRQ4, IRQ5, dan IRQ7 dihubungkan dengan JP_IRQ melalui selektor IRQ bila akan dipergunakan INS-8250 dan masih kosong/tidak dipergunakan pada komputer.

f. pin IOWR dihubungkan dengan pin WR dari INS-8250 yang akan menghasilkan logika low bila processor akan melakukan operasi tulis pada salah satu port dari INS-8250

g. pin IORD dihubungkan dengan pin RD dari INS-8250 yang akan menghasilkan logika low bila prosesor akan melakukan operasi baca pada salah satu port dari INS-8250.

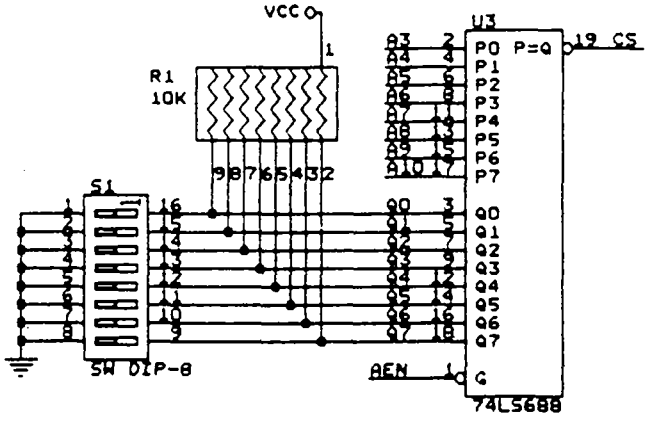
h. pin GND yang dihubungkan dengan jalur ground dari PCB rangkaian switcher dan pin VSS dari komponen untuk mendapat tegangan referensi yang sama antara komputer dengan PCB rangkaian switcher.

i. pin +5V dihubungkan dengan pin VCC dari komponen yang memerlukan supply +5V

2.2. Rangkaian decoding alamat

Rangkaian decoding alamat diperlukan untuk dapat mengalamati INS-8250 dimana hasil dari decoding alamat ini dihubungkan dengan pin CS pada INS-8250 dengan logika low yang menyatakan prosesor memilih INS-8250 untuk berhubungan dengannya untuk melakukan operasi I/O. Rangkaian decoding alamat ini dibentuk dengan tiga buah komponen yang dihubungkan seperti pada gambar 3-4 yaitu sebuah 74LS688 2x8bit komparator dimana 8bit yang pertama P0-P7 dihubungkan dengan pin address A3-A10 pada expansion slot dan 8bit yang lain Q0-Q7

dihubungkan dengan dip switch 8bit dan resistor array. Konfigurasi dip switch dan resistor array akan membentuk sebuah kombinasi logika high-low bergantung pada kondisi on-off dari dip switch. Bila dip switch pada posisi off maka kondisi bit yang berhubungan dengannya akan high karena terdapat resistor yang dihubungkan ke VCC, sedang bila dip switch pada kondisi on maka kondisi bit berlogika low karena berhubungan dengan ground. Pada tugas akhir ini untuk dapat menghubungi INS-8250 maka diberikan alamat kosong seperti COM3, Com4, 300H, 310H mulai A0 sampai A10, karena pada alamat ini dalam konfigurasi prosesor merupakan alamat yang belum terpakai.



GAMBAR 3-4

RANGKAIAN DECODER ALAMAT INS-8250

Bila alamat yang dipilih mulai 300H maka pada A2-

A10 yang terhubung pada 74LS688 akan membentuk konfigurasi biner dip switch yang terhubung pada Q0-Q7 sebagai 01000110 dan A0-A2 merupakan kombinasi 0/1 dan terhubung langsung dengan INS-8250.

2.3 Rangkaian INS-8250

INS-8250 memiliki hubungan dengan beberapa komponen dalam tugasnya melakukan operasi komunikasi data, komponen tersebut adalah expansion slot, rangkaian address decoder, rangkaian clock,rangkaian Transceiver 75176 dan rangkaian jumper tambahan.

Dalam pemakaiannya diperlukan fungsi dari register yang terdapat pada INS-8250. Pembuatan program untuk mengendalikan sistem komunikasi ini dapat menghubungi semua register UART yang dapat dilihat secara ringkas pada tabel 2-8. Register-register ini mengontrol semua operasi UART termasuk pengiriman dan penerimaan data. Tiap bit pada register memiliki fungsi tersendiri dan memiliki keadaan logika pada saat reset yang berbeda satu dengan lainnya.

A. Line Control Register

Pembuatan program untuk mengendalikan sistem komunikasi ini menentukan bentuk dari pertukaran komunikasi data asinkron dan menset

bit dari Divisor Latch Access melalui Line Control Register (LCR). Pembuatan program untuk sistem ini dapat juga melakukan pembacaan pada isi dari Line Control Register. Kemampuan pembaca an ini memudahkan dalam pembuatan program pengendali dan mengurangi diperlukannya tempat menyimpan data yang berbeda pada sistem memory khusus untuk menyimpan karakteristik dari jalur komunikasi yang terjadi. Isi dari register LCR terlihat pada tabel dan penjelasan fungsi dari tiap bit terlihat di bawah ini.

a. bit 0 dan 1

Kedua bit ini menyatakan jumlah bit dari tiap karakter serial yang dikirim maupun diterima. Kombinasi dari kedua bit ini adalah seperti terlihat pada tabel 3-1.

TABEL 3-1[23]

TABEL PILIHAN PANJANG KARAKTER DATA

Bit 0	Bit 1	Panjang Karakter
0	0	5 bit
0	1	6 bit
1	0	7 bit
1	1	8 bit

b. bit 2

Bit ini menyatakan jumlah stop bit yang

---

23) Microcommunication Elements Databook, (USA, National Semiconductor, 1987), hal. 2-106

dikirimkan dan diterima dari tiap karakter serial. Bila bit kedua ini berisi logika 0 maka akan dihasilkan atau diperiksa adanya sebuah stop bit pada data yang ditransmisikan. Bila bit kedua ini diberikan logika 1 maka bila data panjangnya 5 bit maka akan dihasilkan stopbit sebanyak setengah bit. bila bit 2 berlogika 1 dan panjang data 6, 7, atau 8 bit maka akan dihasilkan dua buah stop bit. Penerima hanya akan memeriksa stopbit yang pertama dan tidak bergantung pad banyaknya jumlah stop bit yang dipilih.

c. bit 3

Bit ini merupakan bit parity enable. Bila bit ketiga ini berisi logika 1 maka akan dihasilkan sebuah parity bit pada data yang dikirim atau diperiksa pada data yang diterima diantara bit terakhir dari data dan stop bit dari data serial. Parity bit dihasilkan dari penjumlahan banyaknya bit berlogika 1 pada sebuah data serial dengan kondisi genap atau ganjil.

d. bit 4

Bit ini merupakan pemilihan parity bit genap. Bila bit ketiga berlogika 1 dan bit keempat berlogika 0 maka jumlah ganjil dari data yang berlogika 1 akan dikirim atau

diperiksa pada data serial dan parity bit. Bila bit ketiga berlogika 1 dan bit keempat berlogika 1 maka jumlah genap dari bit yang berlogika 1 yang akan dikirim atau diperiksa.

e. bit 5

Bit kelima ini merupakan stick parity bit. Bila Bit ketiga, keempat, dan kelima semuanya berlogika 1 maka parity bit yang dikirim atau diperiksa memiliki logika 0. Bila bit ketiga dan kelima berlogika 1 dan bit keempat berlogika 0 maka parity bit yang dikirim atau diterima memiliki logika 0. Bila bit kelima ini berisi logika 0 maka tidak akan dihasilkan stick parity.

f. bit 6

Bit ini merupakan break control bit yang akan mengakibatkan kondisi break pada pengiriman data oleh UART. Bila bit ini diset pada logika 1 maka serial output (SOUT) dipaksa pada kondisi spacing yang berlogika 0. Kondisi break ini dapat ditiadakan dengan memberikan kondisi logika 0 pada bit keenam ini. Bit dari break control ini hanya berfungsi pada SOUT saja dan tidak memberikan gangguan apapun pada keadaan logika dari transmitter. Kemampuan ini memungkinkan CPU utuk memberitahu

terminal dari sistem komunikasi komputer. Bila keadaan diatas dipergunakan maka tidak ada kesalahan atau kelebihan karakter yang akan dikirm selama terjadinya kondisi break. Kondisi ini terjadi dengan urutan sebagai berikut :

- Memasukan semua logika 0, karakter sebagai respon dari THRE
- Menset kondisi brak setelah terjadi sinyal THRE
- Menunggu transmitter sampai kondisi idle, (TEMT + 1) dan mengosongkan kondisi break bila transmisi normal dapat dilakukan kembali.

Selama terjadinya kondisi break, transmitter dapat difungsikan sebagai timer karakter utuk mementukan secara tepat lama terjadinya kondisi break.

g. bit 7

Bit ini merupakn divisor latch access bit (DLAB) yang harus diset high (logika 1) untuk mengakses divisor latches dari baud generator selama terjadi operasi baca/tulis dan harus diset pada kondisi low (logika 0) untuk mengakses receiver buffer, transmitter holding register, dan interrupt enable register.

B. Line Status Register

Ke delapan bit register ini menyediakan informasi status kepada CPU yang berhubungan dengan transfer data. Penjelasan isi dari Line Status Register seperti terlihat di bawah ini.

a. bit 0

Bit ini merupakan indikator dari receiver Data Ready (DR). Bit 0 akan diset pada kondisi logika 1 bila sebuah karakter yang lengkap yang masuk pada receiver telah diterima dan ditransfer ke Receiver Buffer Register. Bit ini akan diset lagi pada kondisi logika 0 dengan membaca data pada Receiver Buffer Register.

b. bit 1

Bit ini merupakan indicator dari Overrun Error (OE) yang menandakan bahwa data pada Receiver Buffer Register tidak dibaca oleh CPU sebelum karakter berikutnya ditranfer ke Receiver Buffer Register, sehingga menghapus karakter sebelumnya. Indikator OE akan diset pada kondisi logika 1 pada saat terdeteksi adanya keadaan overrun diatas dan akan mengalami kondisi reset bila CPU melakukan pembacaan isi dari Line Status Register.

c. bit 2

Bit ini merupakan indikator Parity Error

(PE) yang menandakan bahwa karakter data yang diterima tidak memiliki parity even atau odd yang benar, seperti yang sudah dipilih pada bit pemilih parity even. Bit PE ini akan diset pada kondisi logika 1 pada saat terdeteksi adanya parity error dan akan direset ke kondisi logika 0 pada saat CPU membaca isi dari Line Status Register.

d. bit 3

Bit ini merupakan indikator dari Framing Error (FE) yang menandakan bahwa karakter yang diterima tidak memiliki stop bit yang tepat. Bit 3 ini akan diset pada kondisi logika 1 bila stopbit yang diterima setelah bit data terakhir atau prity bit memiliki logika 0 (level spacing). Indikator FE ini akan direset bila CPU melakukan pembacaan isi dari Line Status Register. UART akan berusaha melakukan sinkronisasi setelah terjadinya framing error. Untuk melaksanakan hal ini UART akan mengasumsi bahwa framing error terjadi sebelum start bit berikutnya sehingga akan mengambil start bit ini dua kali dan kemudian baru dilakukan pengambilan data.

e. bit 4

Bit ini menandakan terjadinya kondisi Break Interrupt (BI) dan akan diset pada kondisi

logika 1 bila data input yang diterima berada pada kondisi spacing (logika 0) untuk waktu selama terjadinya transmisi karakter data secara penuh (yaitu wktu yang dipelukan untuk mentransmisikan start bit + data bit + parity bit + stop bit). Indikator BI akan mengalami kondisi reset bila CPU melakukan pembacaan isi dari Line Status Register. Untuk memulai lagi kondisi transmisi setelah diterima kondisi break diperlukan keadaan logika pada pin SIN pada logika 1 selama paling sedikit ½ dari waktu transmisi sebuah bit. Bit 1 hingga bit 4 adalah keadaan error yang akan menghasilkan interrupt pada Receiver Line Status pada saat keadaan yang telah didefinisikan terjadi dan interrupt pada kondisi enable.

f. bit 5

Bit ini merupakan indikator dari Transmitter Holding Register Empty. Bit 5 menandakan bahwa UART siap menerima sebuah karakter untuk ditransmisikan. Seagai tambahan bit ini mengakibatkan UART Menghasilkan interrupt pada CPU bila Interrupt dari Transmitter Holding Register Empty berada pada kondisi enable (high). Bit THRE akan diset pada kondisi logika 1 bila sebuah karakter ditransfer dari Transmitter Holding

Register ke Transmitter Shift Register dan akan direset bial CPU melakukan penulisan pada Transmitter Holding Register.

g. Bit 6

Bit ini merupakan indikator dari Transmitter Empty (TEMT) yang akan diset pada kondisi logika 1 bila Transmitter Holding Register dan Transmitter Shift Register keduanya kosong (tidak berisi data. Dan akan berada pada kondisi reset yaitu pada logika 0 bila salah satu dari THR atau TSR berisi karakter data.

h. Bit 7

Bit ini diset secara tetap pada kondisi logika 0 dan tidak dipergunakakn dalam operasi UART. Line Status Register ini hanya dipergunakan pada operasi pembacaan saja pada operasi normal UART . Penulisan pada register ini tidak dianjurkan kecuali untuk pengetesan pabrik.

C. Interrupt Identification Register

Supaya dapat menyediakan pemakaian software kontrol yang sederhana (minimal) selama terjadinya transfer karakter, UART menyediakan prioritas interrupt sebanyak 4 level dan menyimpan keadaan dari keempat level tersebut pada Interrupt Identification Register. Keempat

level dari prioritas interrupt dalam urutan mulai dari prioritas yang tertinggi adalah Receiver Line Status, Received Data Ready, Transmitter Holding Register Empty, dan Modem Status. Bila CPU mengakses IIR maka UART akan menghentikan semua interrupt yang terjadi dan menandai prioritas interrupt yang tertinggi yang terjadi selama masa penghentian itu untuk kemudian dapat diakses oleh CPU. Pada saat akses dari CPU sedang berlangsung maka UART akan menyimpan interrupt yang baru tersebut tetapi tidak mengubah keadaan indikator semula dari IIR sampai akses dari CPU selesai. Penjelasan Isi dari IIR untuk tiap bit seperti terlihat di bawah ini.

a. bit 0 dan 1

Bit ini dapat dipergunakan pada keadaan interrupt untuk memberikan tanda apakah terjadi pembatalan sementara kondisi interrupt. Bila bit 0 ini memiliki keadaan logika 0 maka interrupt berada pada kondisi pembatalan sementara dan isi dari IIR dapat dipergunakan sebagai pointer untuk routine interrupt service yang diinginkan. Sebaliknya bila bit 0 ini berada pada kondisi logika 1 maka tidak terjadi pembatalan sementara dari interrupt.

b. bit 1 dan 2

Kedua bit dari IIR ini dipergunakan untuk menandai prioritas tertinggi intterupt bila terjadi pembatalan sementara, berdasar kombinasi bit dengan prioritasseperti di atas .

c. bit 3, 4, 5, 6, 7

Kelima bit dari IIR ini senantiasa diset pada keadaan logika 0 dan tidak dipergunakan pada operasi UART.

D. Interrupt Enable Register.

Register ini akan memberikan kondisi enable pada keempat tipe dari interrupt UART dimana tiap interrupt secara sendiri-sendiri sesuai dengan kondisinya akan mengaktifkan sinyal output interrupt (INTR). Sistem Interrupt ini dapat di-nonaktifkan semua dengan memberikan kondisi reset pada bit 0 sampai bit 3 dari Interrupt Enable Register (IER). Demikian pula dengan menset bit 0 hingga bit 3 dari IER pada kondisi logika 1 akan memberikan kondisi enable pada interrupt yang bersangkutan. Dengan meniadakan sebuah kondisi interrupt akan mencegah interrupt yang bersangkutan terdeteksi oleh IIR dan mengakibatkan sinyal output INTR aktif. Semua sistem lainnya akan berfungsi secara normal termasuk Line Status Register dan

Modem Status Register tanpa dipengaruhi oleh keadaan setting dari interrupt. Penjelasan isi dari register IER untuk tiap bit terlihat dibawah ini.

a. bit 0

Bit ini akan memberikan kondisi enable pada interrupt Received Data Available bila diset pada kondisi logika 1.

b. bit 1

Bit ini akan memberikan kondisi enable pada interrupt Transmitter Holding Register bila diset pada kondisi logika 1.

c. bit 2

Bit ini akan memberikan kondisi enable pada interrupt Receiver Line Status bila diset pada kondisi logika 1.

d. bit 3

Bit ini akan memberikan kondisi enable pada interrupt Modem Status bila diset pada kondisi logika 1.

e. bit 4, 5, 6, 7

Keempat bit ini diset selalu pada kondisi logika 0 dan tidak dipergunakan dalam operasi UART.

E. Modem Control Register

Register ini akan melakukan operasi pengontrolan pada interface yang dihubungkan

dengan MODEM atau data set (atau peralatan periperal yang berfungsi mengemulasi MODEM). Isi dari MODEM Control Register (MCR) dan penjelasan tiap bit secara rinci terdapat dibawah ini.

a. bit 0

Bit ini mengontrol output dari Data Terminal Ready (DTR), bila bit 0 ini diset pada keadaan logika 1 maka output DTR akan berada pada kondisi logika 0, sedang bila bit 0 ini direset pada kondisi logika 0 maka output dari DTR akan diubah pada kondisi logika 1. Output dari DTR dapat dipergunakan pada peralatan EIA untuk membalik kondisi logika pada line driver (seperti DS1488) untuk menghasilkan polaritas input yang benar sesuai yang diperlukan oleh MODEM atau data set.

b. bit 1

Bit ini akan mengontrol kondisi sinyal output dari Request To Send (RTS). Bit 1 ini memiliki perubahan fungsi logika yang sama dengan penjelasan perubahan kondisi logika seperti yang dijelaskan diatas untuk bit 0.

c. bit 2

Bit ini akan mengontrol sinyal output Output 1 (OUT1) yang merupakan output tambahan yang dapat dipergunakan sesuai keinginan pemakai.

Bit 2 ini memiliki perubahan fungsi logika yang sama dengan penjelasan perubahan kondisi logika seperti yang dijelaskan diatas untuk bit 0.

d. bit 3

Bit ini akan mengontrol sinyal output Output 2 (OUT2) yang merupakan output tambahan yang dapat dipergunakan sesuai keinginan pemakai. Bit 3 ini memiliki perubahan fungsi logika yang sama dengan penjelasan perubahan kondisi logika seperti yang dijelaskan diatas untuk bit 0.

e. bit 4

Bit ini menyediakan sebuah fasilitas yang dinamakan local loopback yang berguna untuk melakukan pengetesan diagnosa dari UART. Bila bit 4 ini diset pada keadaan logika 1 maka kejadian sebagai berikut akan terjadi: Transmitter Serial Output (SOUT) akan diset pada keadaan marking (keadaan logika 1); Receiver Serial Input (SIN) dalam keadaan terputus; Output dari Transmitter Shift Register berada pada kondisi looped back pada input Receiver Shift Register; keempat inpur dari MODEM Control (CTS, DSR, RI, DCD) berada dalam keadaan terputus; dan Keempat output dari MODEM Control (DTR, RTS, OUT1,

OUT2) akan terhubung secara internal dengan keempat input dari MODEM Control. Output dari MODEM Control akan berada pada kondisi logika high (tidak aktif). Pada mode diagnosa UART ini, data yang ditransmisikan akan secara cepat diterimakembali, sehingga memungkinkan prosesor untuk mencocokkan jalur data pengiriman dan penerimaan dari UART. Dalam mode diagnosa ini, interrupt untuk transmitter dan receiver berfungsi penuh, dimana interrupt dari MODEM Control juga bekerja tetapi berasal bukan dari keempat sinyal input MODEM Control tetapi dari keempat bit akhir dari MODEM Control Register. Interrupt yang terjadi tetap dikendalikan oleh Interrupt Enable Register.

f. bit 5, 6, 7

Ketiga bit ini secara tetap dikondisikan pada keadaan logika 0 dan tidak dipergunakan pada operasi UART.

F. MODEM Status Register

Register ini menyediakan kondisi dari jalur kontrol dari MODEM (atau peralatan periperal) yang terhubung pada CPU. Sebagai tambahan pada informasi kondisi ini, terdapat empat bit pada MSR yang akan menandai bila terjadi perubahan kondisi dari kontrol MODEM tertentu. Bit ini

diset pada keadaan logika 1 bila input kontrol dari MODEM mengalami perubahan keadaan logika dan akan mengalami kondisi reset pada logika 0 bila CPU memlakukan pembacaan pada MODEM Status Register. Penjelasan terperinci dari register MSR untuk tiap bit terlihat dibawah ini.

a. bit 0

Bit ini merupakan indikator dari Delta Clear To Send (DTCS) dimana bit 0 ini akan menandai terjadinya perubahan logika dari input CTS setelah dilakukan pembacaan oleh CPU yang terakhir.

b. bit 1

Bit ini merupakan indikator dari Delta Data Set Ready (DDSR) dimana bit 1 ini akan menandai terjadinya perubahan logika dari input CTS setelah dilakukan pembacaan oleh CPU yang terakhir.

c. bit 2

Bit ini merupakan detektor untuk Trailing Edge of Ring Indicator (TERI), dimana bit 2 ini menandakan bahwa input RI telah mengalami perubahan kondisi dari logika low ke logika high.

d. bit 3

Bit ini merupakan indikator dari Delta Data Carrier Detect (DDCD) dimana bit 3 ini akan menandai terjadinya perubahan logika dari

input CTS setelah dilakukan pembacaan oleh CPU yang terakhir. Bilamana bit 0, 1, 2, atau 3 berada pada kondisi logika 1 maka interrupt dari MODEM Status akan dihasilkan.

e. bit 4

Bit ini merupakan komplemen dari input Clear To Send (CTS), bila bit 4 (loop) dari MCR diset pada logika 1 maka bit ini akan berada pada kondisi yang sama dengan kondisi RTS pada MCR.

f. bit 5

Bit ini merupakan komplemen dari input Data Set Ready (DSR), bila bit 4 (loop) dari MCR diset pada logika 1 maka bit ini akan berada pada kondisi yang sama dengan kondisi DTR pada MCR.

g. bit 6

Bit ini merupakan komplemen dari input Ring Indicator (RI), bila bit 4 (loop) dari MCR diset pada logika 1 maka bit ini akan berada pada kondisi yang sama dengan kondisi OUT1 pada MCR.

h. bit 7

Bit ini merupakan komplemen dari input Data Carrier Detect (DCD), bila bit 4 (loop) dari MCR diset pada logika 1 maka bit ini akan berada pada kondisi yang sama dengan kondisi OUT2 pada MCR.

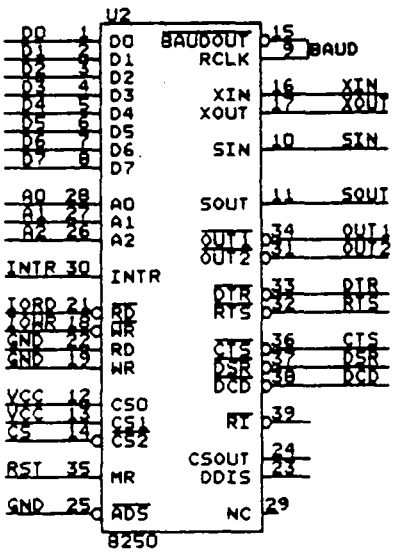
G. Scratchpad Register

Register baca/tulis berisi 8 bit ini tidak memiliki fungsi sama sekali untuk meengontrol UART. Register ini disediakan untuk dipergunakan oleh pemrogram UART sebagai penyimpan data sementara.

Sedangkan konfigurasi dan hubungan pin dari rangkaian INS-8250 secara fisik terlihat pada gambar 3-5 yang terbagi atas beberapa kelompok hubungan yang dapat dijelaskan sebagai berikut :

a. hubungan data bus yaitu D0-D7 yang terhubung dengan data bus pada expansion slot.
b. Hubungan kontrol dari INS-8250 yang dapat diuraikan sebagai berikut:
   - pin RD yang terhubung dengan jalur IORD dari expansion slot yang akan berlogika low pada saat terjadi operasi baca dari porsesor, sedang pin RD terhubung ke GND.
   - pin WR yang terhubung dengan jalur IOWR dari expansion slot yang akan berlogika low pada saat terjadi operasi tulis dari porsesor, sedang pin WR terhubung ke GND.
   - pin A0, A1 dan A2 yang terhubung dengan pin A0. pin A1 dan pin A2 pada expansion slot yang merupakan akan membentuk kombinasi 0-1

unutk melakukan decoding alamat port pada INS-8250.



GAMBAR 3-5

RANGKAIAN HUBUNGAN INS-8250

- pin MR yang terhubung pada jalur RST dari expansion slot yang memungkinkan INS-8250 melakukan operasi reset bersamaan dengan komputer melakukan reset/booting.
- pin CS2 yang terhubung pada jalur CS dari address decoder sehingga kombinasi antara CS dengan IOWR atau IORD akan mengakibatkan prosesor dapat melakukan operasi tulis atau baca pada INS-8250, sedang pin CS0 dan pin CS1 terhubung ke VCC dan pin ADS terhubung ke

GND.

- pin INTR terhubung ke JP_IRQ yang merupakan selektor dari pin IRQ yang terdapat pada expansion bus yang tidak dipakai dalam operasi normal komputer.

c. Hubungan serial input/outpput dari INS-8250 dapat diuraikan :

- pin SIN terhubung ke pin RO dari Tranceiver 75176 sebagai jalur serial in yaitu pada saat INS-8250 berfungsi sebagai receiver data serial.
- pin SOUT terhubung ke pin DI dari Tranceiver 75176 sebagai jalur serial out dimana INS-8250 berfungsi sebagai transmitter.

d. Hubungan external clock input/outpput yang memungkinkan INS8250 membangkitkan clock unutk pengiriman data serial yaitu pin XIN dan pin XOUT yang terhubung ke rangkaian clock

e. Hubungan Modem kontrol INS-8250 yaitu pin OUT1, pin OUT2, pin DTR, pin RTS, pin CTS, pin DSR, pin DCD akan terhubung ke JP_8250 untuk membentuk konfigurasi dari null-modem dan JP_75176 untuk mengontrol Tranceiver 75176.

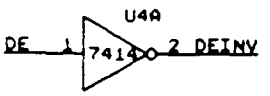
f. Hubungan pin BAUDOT dan pin RCLK akan memberikan clock rate yang sama antara transmitter dan receiver untuk membentuk komunikasi asinkron.

g. pin lain seperti CSOUT, DDIS, dan RI dibiarkan

dalam kondisi terbuka karena tidak diperlukan.

2.4 Rangkaian Inverter untuk Pengontrol Tranceiver 75176

Rangkaian inverter ini mempergunakan sebuah 74LS14 dimana inverter yang dipergunakan hanya sebuah untuk melakukan tugas membalik kondisi logika dari pin OUT1 atau pin DTR pada selektor DE yang berhubungan dengan pin DE pada tranceiver 75176, hal ini karena pada kondisi reset diperlukan keadaan jalur komunikasi pada kondisi high impedance.



GAMBAR 3-6

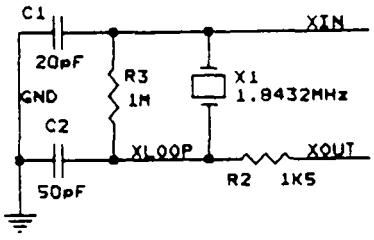
RANGKAIAN INVERTER 74LS14

Pada pin OUT1 dan pin DTR kondisi reset akan memberikan output berlogika high yang bila terhubung ke pin DE pada 75176 akan mengakibatkan kondisi tranceiver 75176 pada keadaan sebagai transmitter, sehingga diperlukan sebuah inverter antara output OUT1 dan DTR dengan pin DE pada Transceiver 75176 sehingga pada kondisi reset

Transceiver berada pada keadaan high impedance.

2.5 Rangkaian Clock Generator

Rangkaian Clock Generator dibangun dengan mempergunakan kristal (X-TAL) berfrekuensi 1,8432MHz dengan susunan rangkaian seperti terlihat pada gambar 3-7.



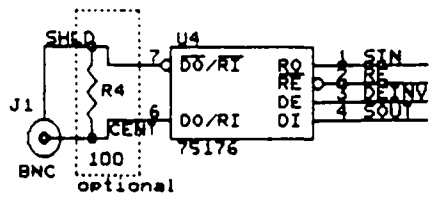
GAMBAR 3-7

RANGKAIAN CLOCK GENERATOR

Resistor yang dipergunakan dipilih di antara range yang disarankan berdasarkan databook yaitu untuk C1 berada pada range 10-30pF , C2 pada range 40-60pF, R3 ditentukan sebesar 1MΩ, dan R2 ditentukan sebesar 1,5KΩ. Pin X-TAL tidak terhubung dengan R2 dihubungkan langsung pada pin XIN pada INS-8250, sedang kaki R2 yang tidak berhubungan dengan X-TAl dihubungkan dengan pin XOUT dari INS-8250.

2.6 Rangkaian Transceiver 75176

Rangkaian Transceiver 75176 dibentuk untuk dapat melakukan proses komunikasi data berdasarkan standar EIA-RS485. Pin RO terhubung pada pin SIN dari INS-8250 sebagai pin output bagi receiver, sedangkan pin DI dihubungkan dengan pin SOUT sebagai pin input bagi transmitter. Pengontrolan kondisi dari 75176 diwakili oleh pin RE dan DE dimana pin RE terhubung ke OUT2 atau GND dari INS-8250 sesuai selektor JP_75176 sedangkan pin DE melalui sebuah inverter 74LS14 untuk menyesuaikan kondisi logika terhubung ke pin DTR atau pin OUT1 dari INS-8250 sesuai selektor JP_75176.



GAMBAR 3-8

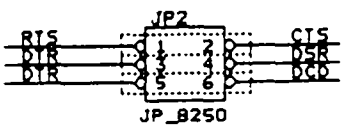
RANGKAIAN TRANSCEIVER 75176

Pin 7 dan pin 8 yaitu pin DO/RI adalah pin yang akan terhubung pada jalur komunikasi dalam hal ini kabel twisted-pair dalam topologi bus. Sedangkan R4 merupakan resistor terminator yang dipergunakan pada kedua ujung jalur/bus bila diperlukan untuk

menjaga kondisi dari jalur komunikasi, dari databook diperoleh besar resistor ini yaitu 100Ω atau 120Ω.

2.7. Rangkaian Selektor Null Modem

Rangkaian selektor dalam tugas akhir ini terbentuk dari beberapa jumper diantaranya adalah JP_8250 yang merupakan selektor bila dipergunakan konfigurasi null-modem oleh UART INS-8250, dimana pada JP_8250 pin DTR harus dihubungkan dengan pin DSR dan pin DCD, sednagkan pin RTS harus dihubungkan dengan pin CTS.



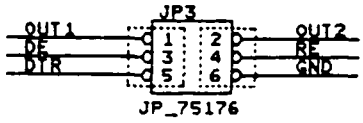
GAMBAR 3-9

RANGKAIAN SELEKTOR NULL MODEM

2.8. Rangkaian Selektor DE dan Selektor RE

Rangkaian selektor kedua dalam tugas akhir ini adalah JP_75176 merupakan selektor untuk menentukan kemampuan hubungan kontrol yang dilakukan INS-8250 pada transceiver 75176 sehingga dapat meghasilkan beberapa kondisi yaitu kondisi transmit, kondisi receive, atau kondisi high impedance, yang akan dapat diperoleh bila pin DE

dihubungkan dengan pin OUT1 dan pin RE dihubungkan dengan pin OUT2.

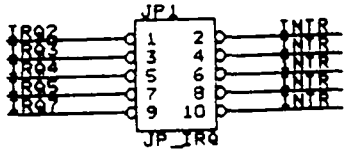


GAMBAR 3-10

RANGKAIAN SELEKTOR DE DAN SELEKTOR RE

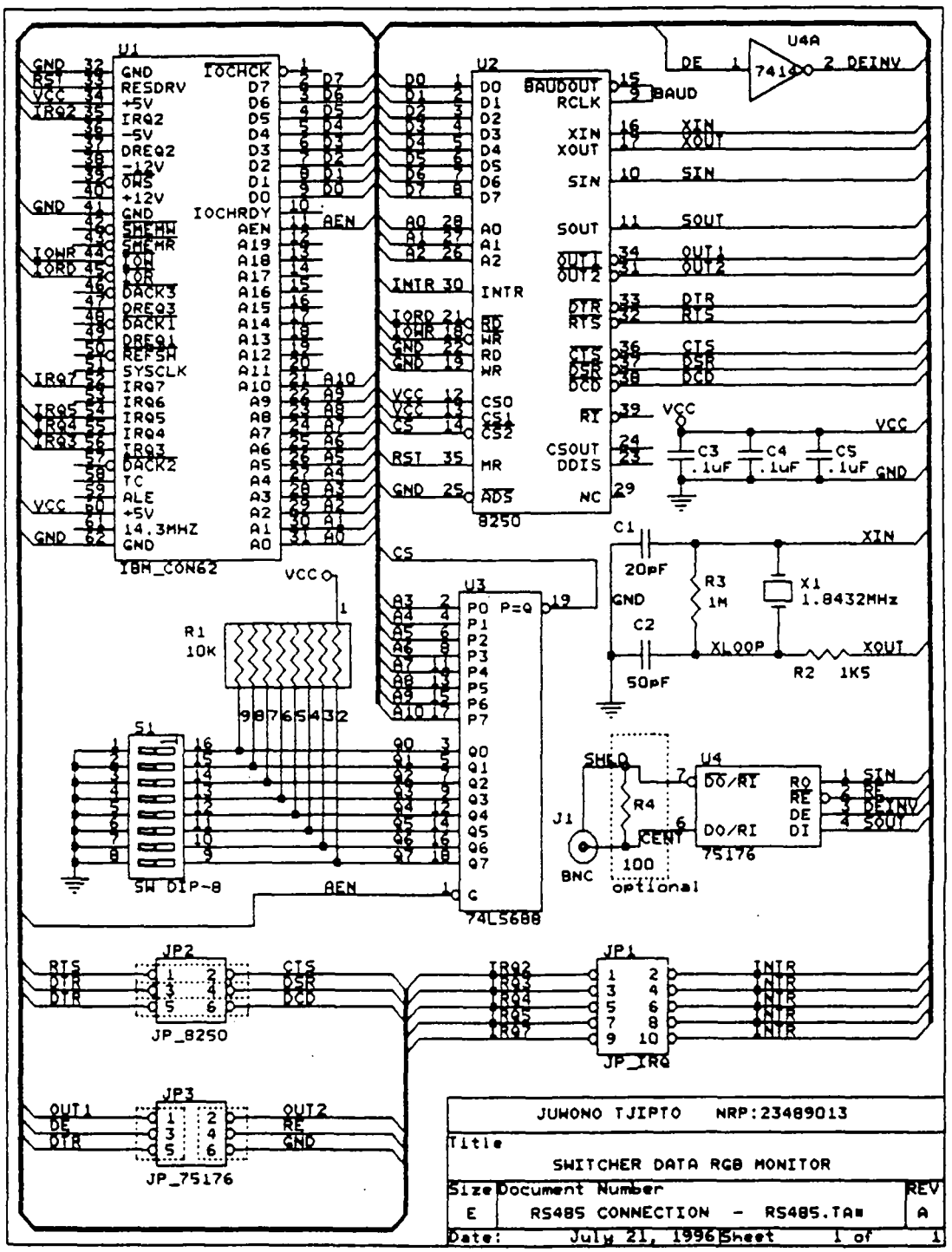
2.9. Rangkaian Selektor IRQ

Rangkaian selektor terakhir dalam Tugas Akhir ini adalah JP_IRQ yang merupakan selektor pemilihan IRQ yang akan dipergunakan oleh INS-8250 yang merupakan IRQ bebas dan tidak dipergunakan oleh peralatan lain dari komputer.



GAMBAR 3-11

RANGKAIAN SELEKTOR IRQ



GAMBAR 3-12

RANGKAIAN LENGKAP

3. PERENCANAAN SOFTWARE

Software yang dipergunakan untuk mengendalikan *Switcher Data RGB Monitor* dibuat dengan mempergunakan bahasa pemrograman **PASCAL.** Pemilihan bahasa pemrograman ini disebabkan kemudahan yang disediakan dalam bentuk routine maupun procedure tertentu yang telah didefinisikan terlebih dahulu.

Program yang dibuat dibentuk sebagai program resident yaitu program yang akan tetap tinggal dalam memory setelah dieksekusi dan baru akan aktif lagi setelah dilakukan pemanggilan ulang dengan interrupt tertentu. Dalam tugas akhir ini interrupt yang dipergunakan untuk mengaktifkan program resident ini terdiri dari dua bagian yaitu interrupt timer ($1C), interrupt keyboard dengan hot-keys SHIFT-PRTSC ($5).

Perencanaan software diawali dengan penentuan data format dari pengiriman data serial yang dilakukan oleh RS-485. Format dari protokol data serial yang dipergunakan adalah sebagai berikut:

Untuk pengiriman layar monitor :

Supervisor

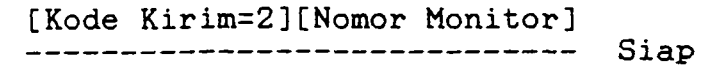
[Kode Kirim=1][Nomor Monitor]	[Delay]	[Data Karakter]	[Data Attribut]
8 bit	..as	2000 bytes	2000 bytes

User

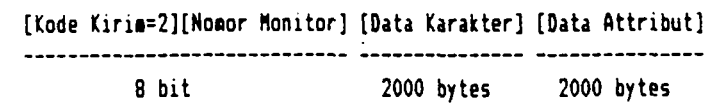
[Kode Kirim=1][Nomor Monitor] ---------- Siap Terima Data...
8 bit

Untuk penerimaan layar monitor :

Supervisor

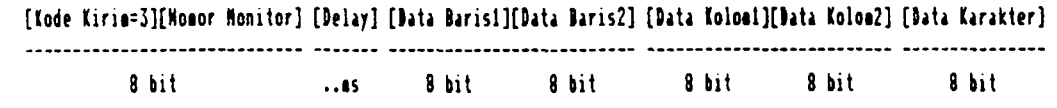


User

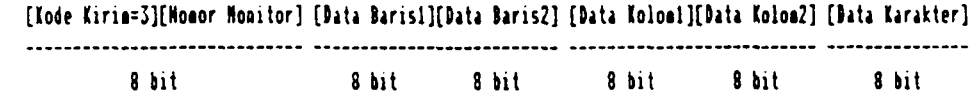


Untuk pengiriman karakter hasil pengetikan keyboard :

Supervisor

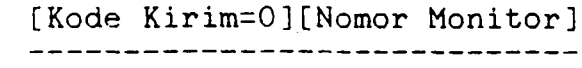


User

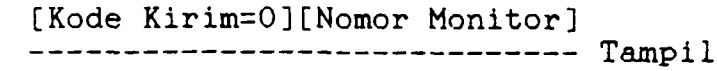


Untuk permintaan hubungan yang dilakukan oleh user :

User



Supervisor



Protokol di atas memiliki kemampuan untuk dapat mengalamati sebanyak 16 buah komputer (sesuai banyaknya bilangan dasar pada sistem hexadesimal). Bila diperlukan untuk dapat mengalamati lebih banyak

komputer maka antara pengiriman kode nomor monitor dan kode perintah masing2 sebanyak 8 bit, sehingga dalam format protokol data akan memerlukan lebih banyak penyesuaian penerimaan data awal.

Perencanaan software ini dibagi menjadi dua buah bagian dimana bagian pertama merupakan penjelasan struktur program dalam bentuk diagram alir dan bagian yang kedua merupakan penjelasan program itu sendiri yang dibuat dalam bahasa pemrograman **PASCAL**.

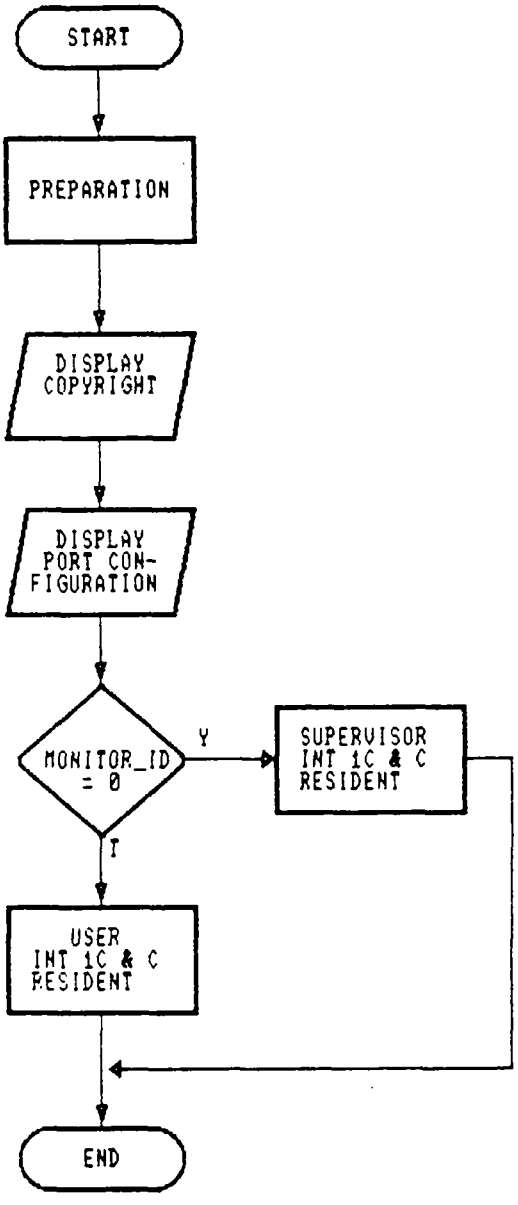
3.1 Struktur Program Dalam Bentuk Diagram Alir

Perencanaan struktur program terdiri dari beberapa bagian dimana setiap bagian memiliki tugas tersendiri dan dapat merupakan bagian dari fungsi-fungsi lainnya.

Diagram alir dalam struktur progarm ini dapat dijelaskan dengan pembagian sebagai berikut :

A. Diagram Alir Program Utama.

Diagram alir seperti terlihat pada gambar 3-13 meliputi urutan routine preparation yang berisi pembacaan parameter yang dimasukkan bersamaan dengan eksekusi program, inisialisasi awal INS 8250, dan kondisi awal dari beberapa variabel yang dipakai. Urutan selanjutnya merupakan tam pilan pembuat program diikuti tampilan kondisi dari status monitor dan port komunikasi yang dipakai. Untuk mencapai kondisi resident,



GAMBAR 3-13

DIAGRAM ALIR PROGRAM UTAMA

program harus melalui sebuah pilihan yaitu bila status monitor milik komputer pengawas maka program yang akan resident adalah routine khusus pengawas, bila monitor milik komputer user maka yang akan resident adalah routine khusus user yang berbeda pada routine untuk interrupt keyboard print-screen.

B. Diagram Alir Routine Program Dengan Interrupt Timer.

Seperti terlihat pada gambar 3-14, Diagram alir ini memiliki dua buah bagian besar yaitu bagian untuk menampilkan status monitor dan bagian untuk menangani bila komputer yang bersangkutan memerlukan komunikasi serial.

Pada bagian untuk menampilkan Status Monitor akan dilakukan penampilan status pada bagian kanan atas monitor dan akan diulangi setiap kurang lebih 1 detik.

Pada bagian untuk pelaksanaan komunikasi serial akan melalui beberapa kondisi yaitu kondisi pertama adalah kondisi dari isi dari port Line Status Register yang berisi data 1 yang menandakan adanya data pada buffer receiver dari INS8250. Data akan diambil pada bagian preparation dengan beberapa perintah lain kemudian data yang diambil tersebut akan dibandingkan dengan status monitor yang

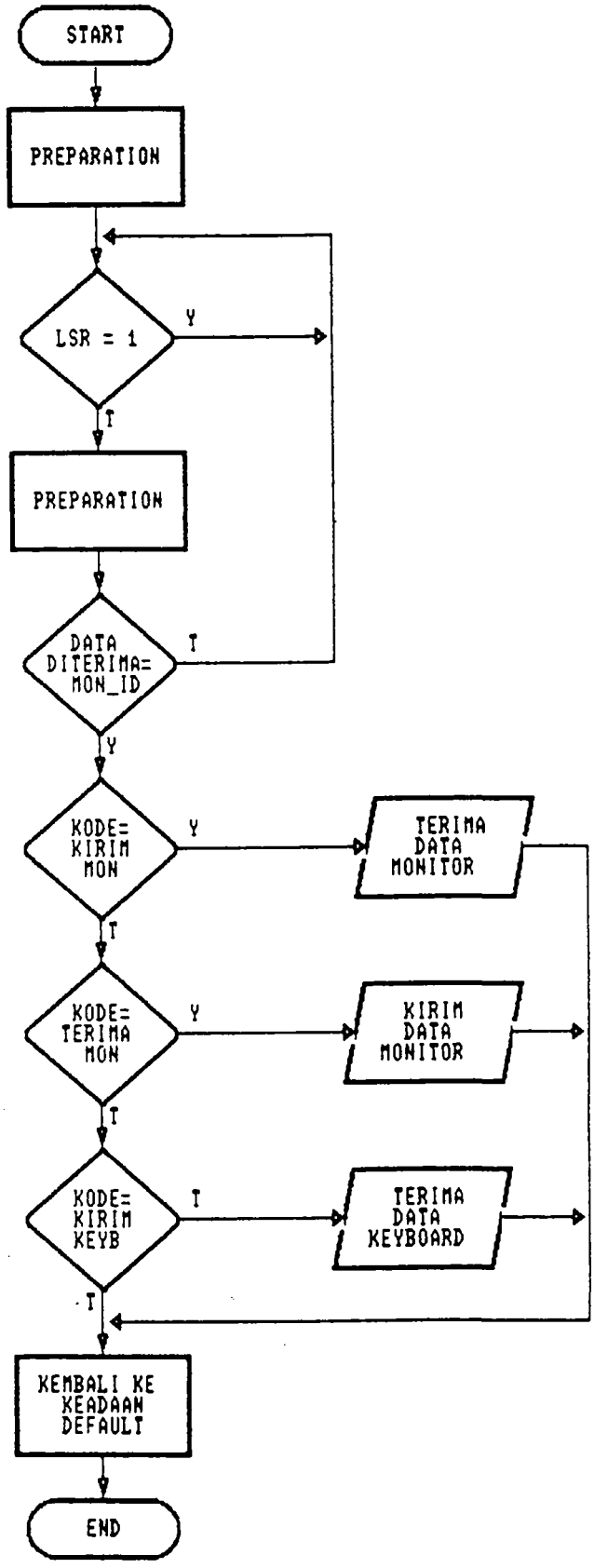
bersangkutan, bila sama maka komputer yang bersangkutan akan mengirim data yang menyatakan siap untuk menerima, sedang bila tidak maka akan dilanjutkan untuk menampilkan tampilan status monitor tiap 1 detik.

Komputer sekarang berada pada keadaan siap untuk melakukan komunikasi serial dan siap menerima perintah yang terdiri dari tiga buah perintah yaitu perintah dengan kode kirim monitor maka komputer akan melakukan penerimaan byte selanjutnya yang merupakan data monitor dan kemudian menampilkannya pada monitornya.

Bila kode yang diterima merupakan kode terima monitor maka komputer akan melakukan pengiriman data monitor yang saat itu sedang ditampilkan.

Sedangkan bila diterima kode kirim keyboard maka akan dilakukan penerimaan data keyboard yang menyatakan posisi kursor dan karakter yang ada pada posisi tersebut.

Setelah selesai melakukan penerimaan data serial sesuai kode yang diterima maka akan kembali ke kondisi default yaitu melakukan penampilan status monitor sambil melakukan pengecekan terhadap Line Status Register.



GAMBAR 3-14

DIAGRAM ALIR ROUTINE DENGAN INTERRUPT TIMER

C. Diagram Alir Routine Program Dengan Interrupt Keyboard PrintScreen.

Diagram alir seperti dapat dilihat pada gambar 3-15a akan dikerjakan pada komputer pengawas bila terjadi penekanan tombol PrintScreen pada keyboard. Diagram alir yang merupakan program awal ini diawali dengan preparation yang berisi memberikan kondisi default pada variabel yang dipakai, kemudian akan dilanjutkan dengan menampilkan menu awal yang berisi pilihan untuk melakukan operasi monitor atau operasi keyboard atau akan keluar. Bila dilakukan penekanan tombol pada keybaord diluar tombol yang diperkenankan akan ditampilkan nama pembuat program.

Bila dilakukan penekanan tombol m/M maka program akan menuju pada routine monitor, sedang untuk menuju ke routine keyboard dapat dilakukan dengan melakukan penekanan tombol k/K dimana kondisi nomor monitor bukan 0 dan sudah dilakukan operasi penerimaan monitor. Bila dilakukan penekanan tombol x/X maka program akan resident kembali. Penekanan tombol diluar tombol diatas akan menampilkan nama pembuat program.

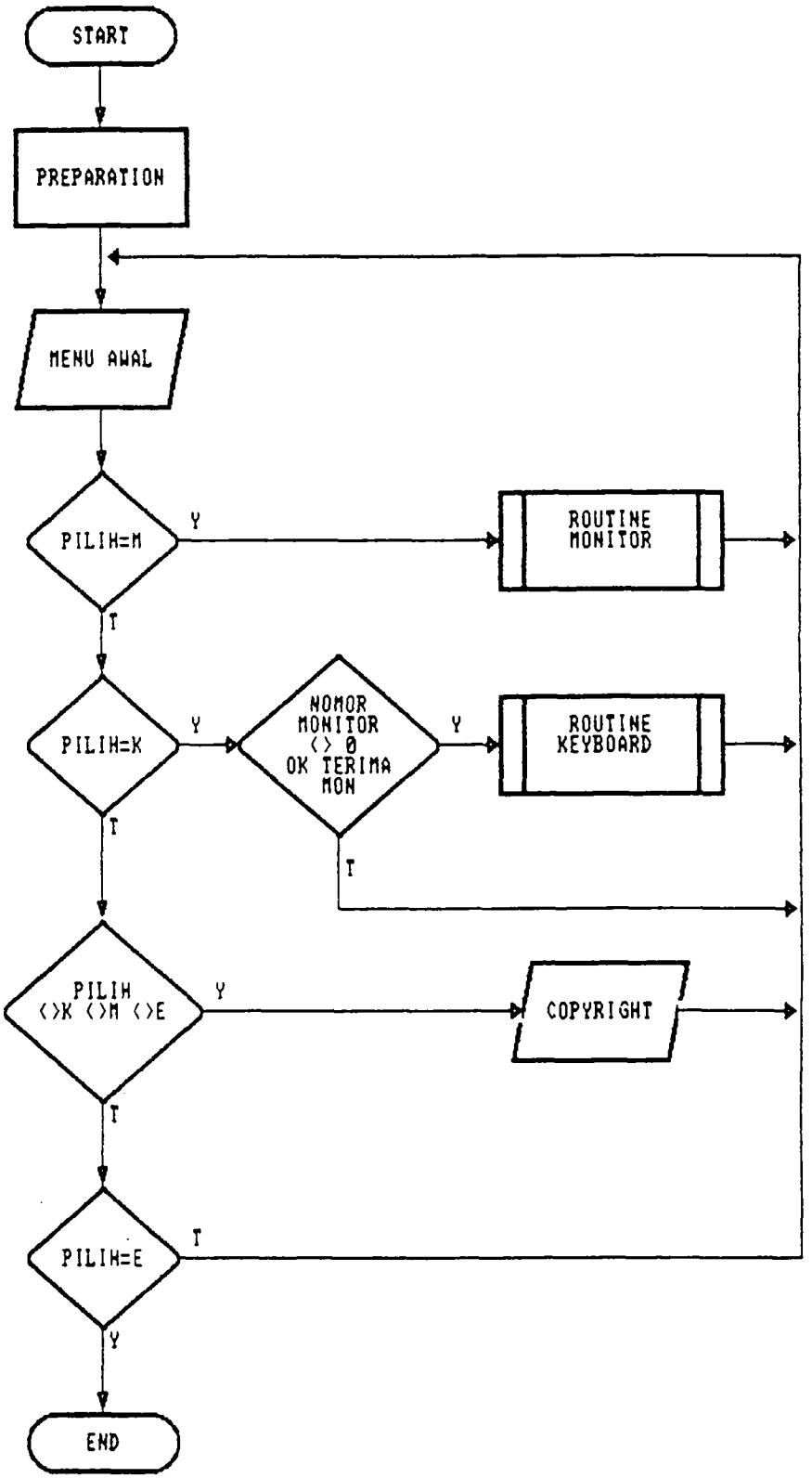
Sedangkan Diagram Alir seperti tampak pada gambar 3.15b akan dikerjakan pada komputer user bila dilakukan penekanan tombol printscreen.

Ruotine ini diperlukan bila komputer user ingin meminta hubungan dengan komputer pengawas. Routine ini diawali dengan preparation yang berisi pengsian nilai awal untuk variabel yang dipergunakan, kemudian dilanjutkan dengan mengirim nomor komputer pengawas, bila telah diterima kode siap terima maka akan dikirimkan kode tanya pada supervisor. Setelah kode siap terima diterima untuk kedua kalinya maka komputer user akan mengirim nomor komputernya sendiri untuk memberitahu komputer pengawas siapa yang meminta hubungan.

D. Diagram Alir Routine Program Monitor.

Diagram alir pada gambar 3-16 merupakan bagian dari routine program awal yang khusus menangani operasi monitor.

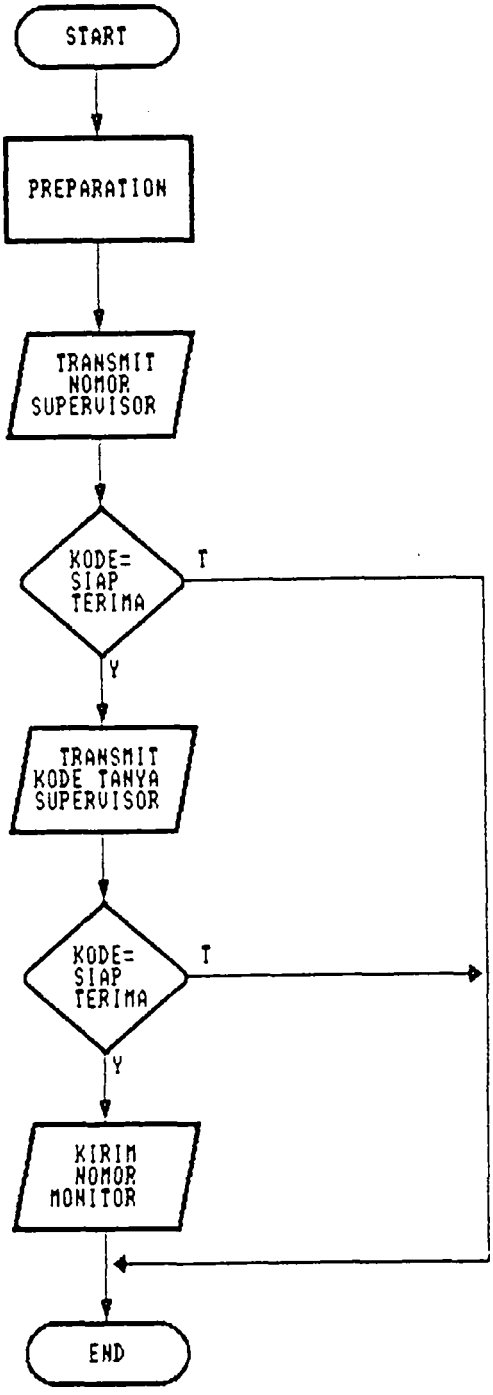
Pada diagram alir ini terdapat dua buah pilihan yaitu tombol t/T untuk melakukan routine terima monitor dengan syarat nomor monitor yaang dihubungi sudah dipilih dan tombol k/K untuk melakukan routine kirim monitor dengan syarat nomor monitor uang dihubungi sudah dipilih dan belum melakukan operasi keyboard.



GAMBAR 3-15a

DIAGRAM ALIR ROUTINE DENGAN INTERRUPT PRT-SC

UNTUK KOMPUTER PENGAWAS



GAMBAR 3-15b

DIAGRAM ALIR ROUTINE DENGAN INTERRUPT PRT-SC

UNTUK KOMPUTER USER

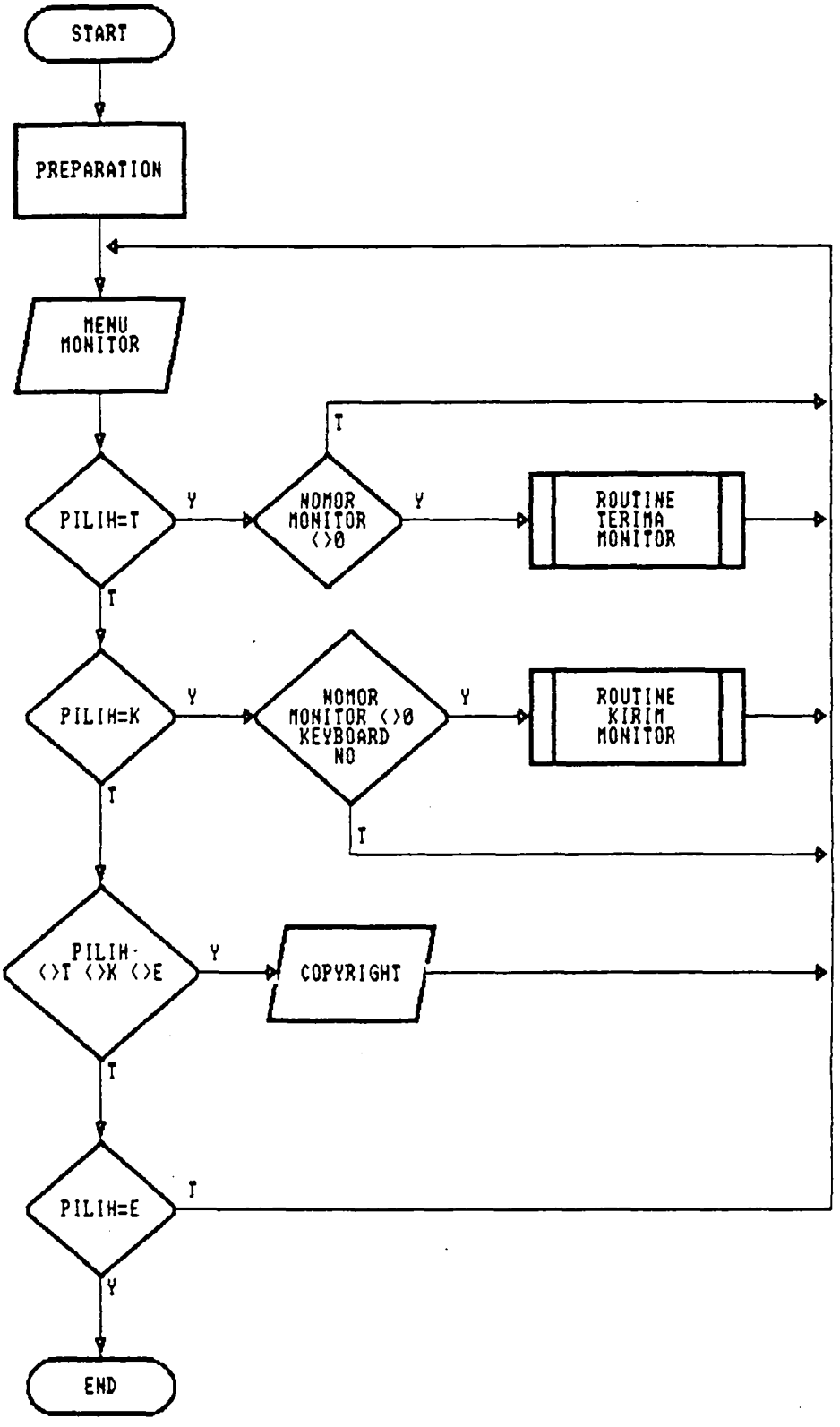
Penekanan tombol x/X akan mengakibatkan routine kembali ke program awal dan penekanan tombol selain yang diperkenankan akan menampilkan nama pembuat program.

E. Diagram Alir Routine Program Keyboard.

Diagram alir pada gambar 3-17 merupakan routine program khusus operasi keyboard yang juga merupakan bagian dari routine program awal.

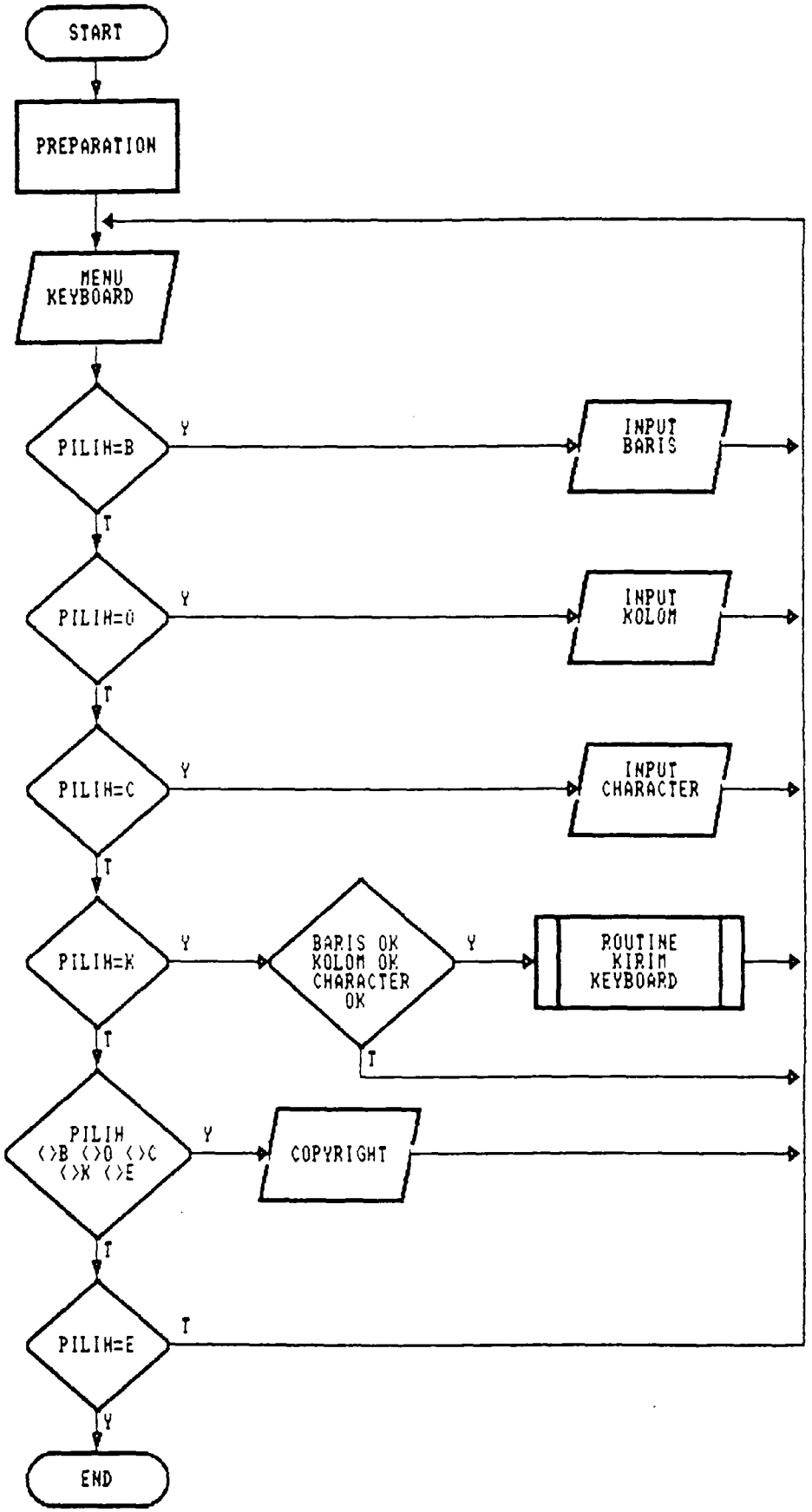
Terdapat beberapa pilihan tombol yang harus dilakukan untuk melakukan operasi keyboard yaitu tombol b/B, o/O, c/C untuk mengisi lokasi baris dan kolom dan karakter yang akan diubah. Sedangkan untuk mengirim diperlukan penekanan tombol k/K yang dapat dilakukan bila ketiga syarat diatas sudah dipenuhi.

Penekanan tombol x/X akan mengakibatkan routine kembali ke program awal dan penekan tombol selain yang diperkenankan akan menampilkan nama pembuat program.



GAMBAR 3-16

DIAGRAM ALIR ROUTINE PROGRAM MONITOR



GAMBAR 3-17

DIAGRAM ALIR ROUTINE PROGRAM KEYBOARD

F. Diagram Alir Routine Procedure Terima Monitor

Diagram Alir seperti dapat dilihat pada gambar 3-18 merupakan procedure yang harus dilalui untuk dapat melakukan penerimaan isi dari layar monitor. Routine ini diawali dengan preparation yang berisi kondisi default dari variabel yang dipergunakan dan diikuti pengiriman nomor monitor yang akan dihubungi. Bila kode siap terima telah diterima maka akan dikirim kode yang menyatakan akan dilakukan pengiriman kode terima monitor. Bila kode siap terima telah diterima maka komputer akan menyiapkan diri untuk menerima data monitor dari komputer yang dihubungi.

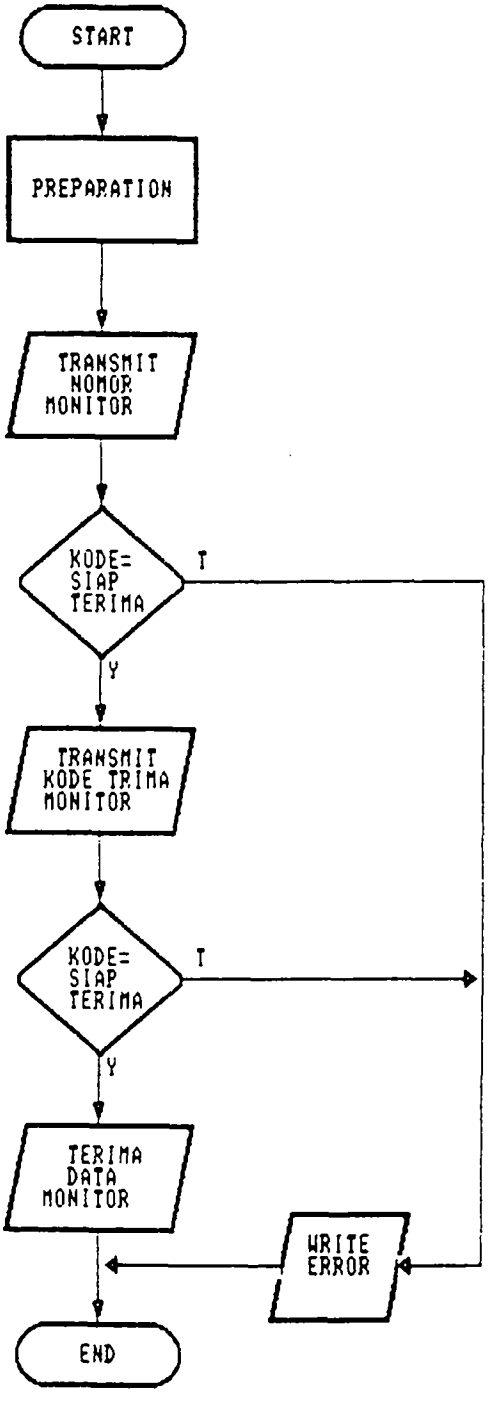
G. Diagram Alir Routine Procedure Kirim Monitor

Diagram alir pada gambar 3-19 memperlihatkan Routine procedure untuk melakukan pengiriman isi dari layar monitor diawali dengan preparation yang berisi kondisi awal dari variabel yang diperlukan, yang diikuti dengan pengiriman nomor monitor yang akan dihubungi. Bila telah diterima kode siap terima maka dikirim kode dair terima monitor. Setelah kode siap terima diterima maka komputer akan melakukan pengiriman data monitor ke komputer yang dihubungi.

H. Diagram Alir Routine Procedure Kirim Keyboard

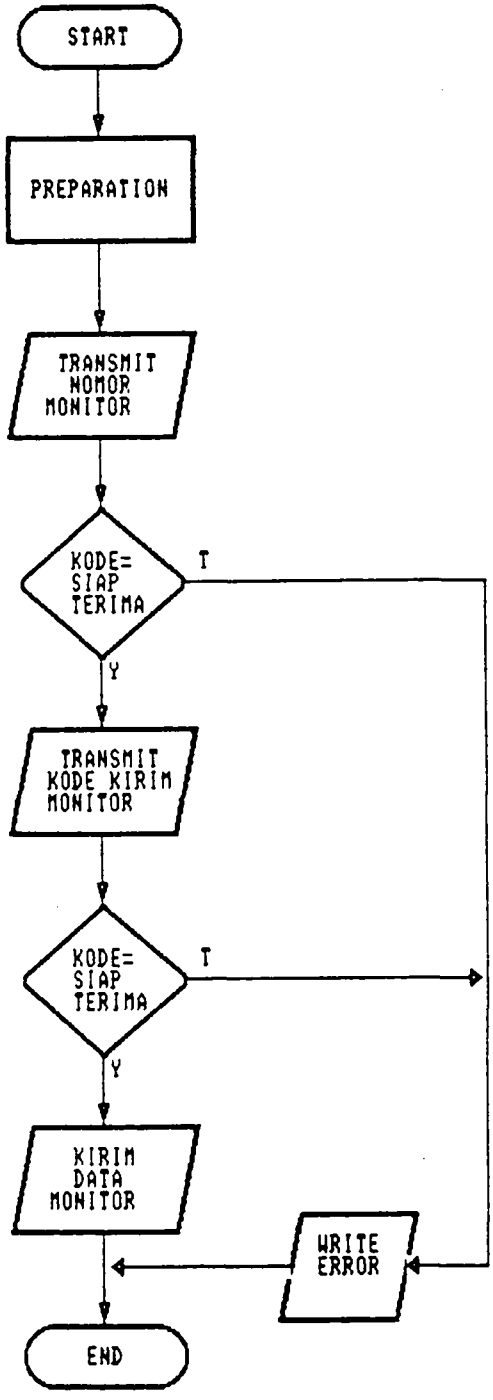
Diagram Alir yang dapat dilihat pada gambar 3-20 ini merupakan procedure dari bagian routine operasi keyboard yang mengatur pengiriman data keyboard bila dilakukan penekanan tombol k/K dengan syarat-syarat yang telah dipenuhi.

Routine procedure ini diawali dengan prepara tion untuk memberikan nilai awal pada variabel yang dipergunakan pada routine ini, kemudian dilanjutkan dengan melakukan pengiriman nomor monitor yang isi layarnya telah diterima. Bila telah diterima kode siap terima maka dikirim kode kirim keyboard. Setelah kode siap terima kembali diterima maka dilakukan pengiriman data keyboard yang berisi posisi kursor dan karakter yang terdapat pada posisi kursor tersebut.



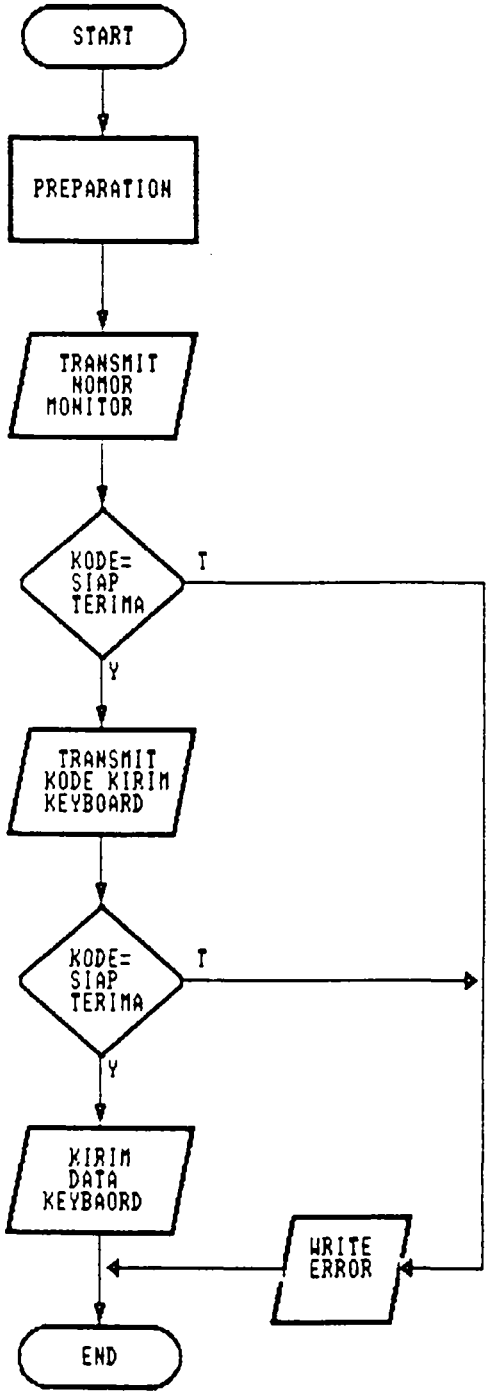
GAMBAR 3-18

DIAGRAM ALIR ROUTINE TERIMA MONITOR



GAMBAR 3-19

DIAGRAM ALIR ROUTINE KIRIM MONITOR



GAMBAR 3-20

DIAGRAM ALIR ROUTINE PROCEDURE KIRIM KEYBOARD

3.2 Listing Program Dalam Bahasa Pemrograman Pascal

Listing program yang akan mengendalikan fungsi dari peralatan *Switcher Data RGB Monitor* seperti tercetak dibawah ini, penjelasan dari tiap bagian atau perintah tertentu terletak di dalam listing program.

```
{TUGAS AKHIR/SKRIPSI}
{PERENCANAAN DAN PEMBUATAN SWITCHER DATA RGB MONITOR}
{Oleh : Juwono Tjipto}
{NRP  : 23489013}
{NIRM : 89.7.002.31073.44029}

{Untuk batas maksimal station sebanyak 16, dibatasi dengan banyaknya parameter yang dapat dibaca
 dan routine yang disediakan}

{$M 13000,0,2048}

Uses Dos,Crt,JTRight;

Type Pixel       = Record                    {tipe variable memory}
                       karakter : Char;
                       Atribut  : Byte;
                   End;
     BufferLayar = Array[1..25,1..80] Of Pixel; {variable memory layar asli}
     Data        = Array[1..25,1..80] Of Byte;  {variable memory data layar}

{Address awal :$300 - Port Data Serial}

var i,j,x,y,q,w,z,a,xk,yk,xl,yl : byte;
    reg     : registers;
    tombol,B  : word;
    CharASCII,NomorMonitor,PopUp,Par,Tanya     : char;
    DataTemp,k,t,p      : word;
    BarisLayar,KolomLayar : array [1..2] of char;
```

```
    LayarBaris,LayarKolom : string[2];
    PosisiBaris,PosisiKolom,PosBaris,PosKolom : string[2];
    Baris,Kolom,Error,N,NoBaris,NoKolom   : Integer;
    TerimaDariMonitor : Boolean;
    Keyboard : Boolean;
    Layar        : BufferLayar Absolute $B800 : $0000;
    LayarSimpan  : BufferLayar;
    LayarTampil  : BufferLayar;
    Character    : Data;
    Attribute    : Data;
    Parameter    : array[1..3] of string[5];
    Datakey      : array[1..6] of byte;
    Monitor_Id   : string[10];
    ComPort      : string[4];
    AlamatAwal,
    BaudRate     : Word;
    Monitor,No_Monitor,DataMonitor,NoUser    : Byte;
    SiapTerima,Counter           : Byte;

Procedure SimpanIsiLayar; {Penyimpanan Tampilan Layar Awal}
Begin
     Move(Layar,LayarSimpan,4000);
End;

Procedure CharKeByte; {Pengubahan data dari tipe Char ke tipe Byte}
Begin
     I:=1;
     For I := 1 To 25 Do
         Begin
         J:=1;
         For J := 1 To 80 Do
         With LayarSimpan[I,J] Do
         Begin
              Character[I,J] := Ord(karakter);
              Attribute[I,J] := atribut;
         End;
         End;
```

```
End;

Procedure ByteKeChar; {Pengubahan data dari tipe Byte ke tipe Char}
Begin
    I:=1;
    For I := 1 To 25 Do
        Begin
        J:=1;
        For J := 1 To 80 Do
        With LayarTampil[I,J] Do
        Begin
            karakter := Chr(character[I,J]);
            atribut := attribute[I,J];
        End;
        End;
End;

Procedure TampilIsiLayar; {Menampilkan layar hasil pengiriman}
Begin
    Move(LayarTampil,Layar,4000);
End;

Procedure DefaultIsiLayar; {Menampilkan kembali layar awal}
Begin
    Move(LayarSimpan,Layar,4000);
End;

procedure TransmitSerial(srlout:byte); {Prosedur pengiriman data serial}
var
   k:byte;
begin
    port[$304]:=4;        {Aktifkan 75176 ke kondisi transmit - Modem Control Register (Out1&Out2)
    k:=port[$300];        {kosongkan port data serial}
    port[$300]:=srlout;   {kirim data serial}
    repeat
    begin
        k:=port[$305];   {check apakah data serial sudah terkirim - Line Status Register}
```

```
        k:=k and $60;    {THRE=1 AND TEMT=1}
    end;
    until k=$60;
    port[$304]:=$8;      {Aktifkan 75176 ke kondisi receive - Modem Control Register (Out1&Out2)
end;

procedure Receive(var srlin:byte); {Prosedur penerimaan data serial}
var
  l:byte;
begin
    repeat
    begin
    l:=port[$305];       {check apakah ada data serial yang diterima - Line Status Register}
    l:=l and 1;          {DR=1 (Data Ready)}
    end;
    until l=1;
    srlin:=port[$300];   {Ambil Data Serial Tersebut}
end;

Procedure BacaParameter;  {Pembacaan parameter untuk menentukan status & nomor station}
Begin
    If ParamCount > 0 Then
    Begin
        Parameter[1] := ParamStr(1);
        If (Parameter[1] = '0') Or (Parameter[1] = '1') Or (Parameter[1] = '2') Then
        Par := 'Y';
    End;
End;

Procedure Parameter1; {Penentuan status dan nomor station untuk menentukan routine yang akan
resident}
Begin
If Parameter[1] = '0'  Then
Monitor_Id := 'SuperVisor'
Else
If Parameter[1] = '1' Then
Monitor_Id := 'User no. '+Parameter[1]
```

```
Else
If Parameter[1] = '2' Then
Monitor_Id := 'User no. '+Parameter[1]
End;

Procedure InitSerial; {Inisialisasi awal INS-8250}
Begin
Port[$303] := $8b;              {Line Control Register - 8 bit, stop bit, parity bit, DLAB=on}
Port[$301] := 0;                {Divisor Latch - most significant byte} {Baurate=38400}
Port[$300] := 3;                {Divisor Latch - least significant byte}
port[$303] := port[$303] and $7b; {Line Control Register - 01111011 - 8 bit, stop bit, enable parity
bit}
port[$304] := $8;               {Modem Control Register - 74176 dalam keadaan receive}
end;

Procedure MainMenu; {Tampilan menu awal}
begin
gotoxy(1,25);
textbackground(3);
textcolor(14);
write('  Monitor    Keyboard                                          Exit  ');
textcolor(4);
gotoxy(4,25);write('M');
gotoxy(16,25);write('K');
gotoxy(75,25);write('X');
End;

Procedure WriteBlank; {Kosongkan Daerah Pencetakan Status}
Begin
TextBackGround(3);
TextColor(4);
GotoXY(55,25);
Write('            ');
End;

Procedure WriteReady; {Cetak Status = Ready}
Begin
```

```
WriteBlank;
TextBackGround(3);
TextColor(4);
GotoXY(55,25);
Write('Ready');
End;

Procedure WriteOK;   {Cetak Status = OK}
Begin
WriteBlank;
TextBackGround(3);
TextColor(4);
GotoXY(55,25);
Write('OK');
End;

Procedure WriteFree; {Cetak Status = Line is Free}
Begin
WriteBlank;
TextBackGround(3);
TextColor(4);
GotoXY(55,25);
Write('Line is free');
End;

Procedure WriteError; {Cetak Status = Error}
Begin
WriteBlank;
TextBackGround(3);
TextColor(4);
GotoXY(55,25);
Write('Error');
End;

Procedure ExitMenu; {Kembali ke kondisi semula, program resident kembali}
Begin
    PopUp := 'N';
```

```
    DefaultIsiLayar;
    CursorOn;
    DefaultPosisiKursor;
End;

Procedure MenuMonitor; {Tampilan untuk menu monitor}
Begin
gotoxy(1,25);
textbackground(3);
textcolor(14);
write('  Terima     Kirim      No. Monitor:                                Exit  ');
textcolor(4);
gotoxy(4,25);write('T');
gotoxy(16,25);write('K');
gotoxy(28,25);write('M');
gotoxy(75,25);write('X');
End;

Procedure KirimMonitor; {Pengiriman data layar monitor supervisor ke layar user}
Begin
If NomorMonitor = '1' Then  {Untuk user nomor 1}
Begin
TransmitSerial($11);
WriteBlank;
Delay(100);
CharKeByte;
I:=1;
For I := 1 To 25 Do
   Begin
    J:=1;
    For J := 1 To 80 Do TransmitSerial(Character[I,J]);
   End;
WriteReady;
I:=1;
For I := 1 To 25 Do
   Begin
    J:=1;
```

```
    For J := 1 To 80 Do TransmitSerial(Attribute[I,J]);
   End;
WriteBlank;
WriteOK;
End
Else
If NomorMonitor = '2' Then  {Untuk User nomor 2}
Begin
TransmitSerial($12);
WriteBlank;
Delay(100);
CharKeByte;
I:=1;
For I := 1 To 25 Do
   Begin
    J:=1;
    For J := 1 To 80 Do TransmitSerial(Character[I,J]);
   End;
WriteReady;
I:=1;
For I := 1 To 25 Do
   Begin
    J:=1;
    For J := 1 To 80 Do TransmitSerial(Attribute[I,J]);
   End;
WriteBlank;
WriteOK;
End
Else
If (NomorMonitor = 'b') Or (NomorMonitor = 'B') Then {Untuk pengiriman ke semua user}
Begin
TransmitSerial($41);
WriteBlank;
Delay(100);
CharKeByte;
I:=1;
For I := 1 To 25 Do
```

```
    Begin
     J:=1;
     For J := 1 To 80 Do TransmitSerial(Character[I,J]);
    End;
WriteReady;
I:=1;
For I := 1 To 25 Do
    Begin
     J:=1;
     For J := 1 To 80 Do TransmitSerial(Attribute[I,J]);
    End;
WriteBlank;
WriteOK;
End
Else WriteError;
End;

Procedure TerimaMonitor; {Penerimaan layar monitor user oleh supervisor}
Begin
If NomorMonitor = '1' Then {Untuk user nomor 1}
Begin
TerimaDariMonitor:=True;
WriteBlank;
TransmitSerial($21);
I:=1;
For I := 1 To 25 Do
    Begin
     J:=1;
     For J := 1 To 80 Do Receive(character[I,J]);
    End;
I:=1;
For I := 1 To 25 Do
    Begin
     J:=1;
     For J := 1 To 80 Do Receive(attribute[I,J]);
    End;
ByteKeChar;
```

```
TampilIsiLayar;
MenuMonitor;
WriteOk;
End
Else
If NomorMonitor = '2' Then {Untuk user nomor 2}
Begin
TerimaDariMonitor:=True;
WriteBlank;
TransmitSerial($22);
I:=1;
For I := 1 To 25 Do
    Begin
     J:=1;
     For J := 1 To 80 Do Receive(character[I,J]);
    End;
I:=1;
For I := 1 To 25 Do
    Begin
     J:=1;
     For J := 1 To 80 Do Receive(attribute[I,J]);
    End;
ByteKeChar;
TampilIsiLayar;
MenuMonitor;
WriteOk;
End
Else WriteError;
End;

Procedure KirimKeyboard; {Pengiriman data hasil pengetikan keyboard oleh supervisor ke layar user}
Begin
If  (Baris > 2) And (Baris < 25) And (Kolom > 2) And {Pembatasan besar layar yang bisa  di-edit  dan
karakter}
   (Kolom < 81) And (Ord(CharASCII) > $1F) And
   (Ord(CharASCII) < $7F) Then
   Begin
```

```
    DataKey[1] := Ord(BarisLayar[1]); {Posisi baris - puluhan}
    DataKey[2] := Ord(Barislayar[2]); {Posisi baris - satuan}
    DataKey[3] := Ord(KolomLayar[1]); {Posisi kolom - puluhan}
    DataKey[4] := Ord(KolomLayar[2]); {Posisi kolom - satuan}
    DataKey[5] := Ord(CharASCII);     {Karakter yang diketikkan}
    If NomorMonitor = '1' Then        {Untuk user nomor 1}
    Begin
        TransmitSerial($31);
        WriteBlank;
        Delay(100);
        G:=1;
        For G := 1 to 5 Do
        TransmitSerial(DataKey[G]);
        WriteOk;
        Gotoxy(Kolom,Baris);
        write(CharAscii);
   End
   Else
    If NomorMonitor = '2' Then        {Untuk user nomor 2}
    Begin
        TransmitSerial($32);
        WriteBlank;
        Delay(100);
        G:=1;
        For G := 1 to 5 Do
        TransmitSerial(DataKey[G]);
        WriteOk;
        Gotoxy(Kolom,Baris);
        write(CharAscii);
   End;
End;
End;

Procedure MenuKeyboard; {Tampilan Menu Keyboard}
Begin
TextBackGround(5);
TextColor(10);
```

```
Gotoxy(1,1);
For M:=1 to 9 do Write(' ');
For G:=1 to 7 do for M:=0 to 9 do Write(G);  {Penulisan koordinat kolom}
Write(8);
Gotoxy(1,2);
For G:=1 to 8 do
begin
    For M:=1 to 9 do
    Write(M);                              {Penulisan koordinat baris}
    Write(0);
End;
Gotoxy(1,1);
For M:=1 to 9 do Writeln(' ',M);
For M:=0 to 9 do Writeln(1,M);
For M:=0 to 4 do Writeln(2,M);
Gotoxy(1,1); Write('11');
Gotoxy(1,2); Write('11');
textbackground(3);
textcolor(14);
Gotoxy(1,25);
write('  Baris:     Kolom:     Character:      Kiria                    Exit  ');
textcolor(4);
gotoxy(4,25);write('B');
gotoxy(17,25);write('O');
gotoxy(28,25);write('C');
gotoxy(45,25);write('K');
gotoxy(74,25);write('X');
end;

Procedure ShowUser1; {Routine user 1 yang akan dijalankan dengan interrupt $1c}
Interrupt;
Begin
If Port[$305] And 1 = 1 Then {Line Status Register - Data Ready=1}
Begin
    If (Port[$300] = $11) Or (Port[$300] = $41) Then {Penerimaan data layar monitor}
    Begin
        x:=WhereX;
```

UNIVERSITAS KRISTEN PETRA PERPUSTAKAAN

```
    y:=WhereY;
    SimpanIsiLayar;
    CursorOff;
    I:=1;
    For I:=1 to 25 do
        Begin
        J:=1;
        For J:=1 to 80 do receive(Character[I,J]);
        End;
    I:=1;
    For I:=1 to 25 do
        Begin
        J:=1;
        For J:=1 to 80 do receive(Attribute[I,J]);
        End;
    BytekeChar;
    TampilIsiLayar;
    Delay(9999);
    DefaultIsiLayar;
    GotoXY(x,y);
    CursorOn;
    Transmitserial($10);
End
Else
If Port[$300] = $21 Then {Pengiriman data layar monitor}
Begin
    Delay(100);
    Simpanisilayar;
    charkebyte;
    I:=1;
    For I:=1 to 25 do
    Begin
        J:=1;
        For J:=1 to 80 do transmitserial(character[i,j]);
        end;
    I:=1;
    For I:=1 to 25 do
```

```
            Begin
                J:=1;
                For J:=1 to 80 do transmitserial(attribute[i,j]);
            End;
        End
        Else
        If Port[$300] = $31 Then {Penerimaan data keyboard : posisi dan karakter}
        Begin
            G:=1;
            For G:=1 to 5 do receive(Datakey[G]);
            PosBaris:=Chr(DataKey[1])+Chr(DataKey[2]);
            PosKolom:=Chr(DataKey[3])+Chr(DataKey[4]);
            Val(PosBaris,NoBaris,Error);
            Val(PosKolom,NoKolom,Error);
            xk:=wherex;
            yk:=wherey;
            Gotoxy(NoKolom,NoBaris);
            Write(Chr(DataKey[5]));
            gotoxy(xk,yk);
        End;
    End;
End;

Procedure ShowUser2; {Routine User 2 yang akan dijalankan dengan interrupt $1c}
Interrupt;
Begin
If Port[$305] And 1 = 1 Then
Begin
    If (Port[$300] = $12) Or (Port[$300] = $41) Then {Penerimaan data layar monitor}
    Begin
        x:=WhereX;
        y:=WhereY;
        SimpanIsiLayar;
        CursorOff;
        I:=1;
        For I:=1 to 25 do
            Begin
```

```
            J:=1;
            For J:=1 to 80 do receive(Character[I,J]);
            End;
        I:=1;
        For I:=1 to 25 do
            Begin
            J:=1;
            For J:=1 to 80 do receive(Attribute[I,J]);
            End;
        BytekeChar;
        TampilIsiLayar;
        Delay(9999);
        DefaultIsiLayar;
        GotoXY(x,y);
        CursorOn;
        Transmitserial($10);
    End
    Else
    If Port[$300] = $22 Then {Pengiriman data layar monitor}
    Begin
        Delay(100);
        Simpanisilayar;
        charkebyte;
        I:=1;
        For I:=1 to 25 do
        Begin
            J:=1;
            For J:=1 to 80 do transmitserial(character[i,j]);
            end;
        I:=1;
        For I:=1 to 25 do
        Begin
            J:=1;
            For J:=1 to 80 do transmitserial(attribute[i,j]);
        End;
    End
    Else
```

```
    If Port[$300] = $32 Then {Penerimaan data keyboard : posisi dan karakter}
    Begin
        G:=1;
        For G:=1 to 5 do receive(Datakey[G]);
        PosBaris:=Chr(DataKey[1])+Chr(DataKey[2]);
        PosKolom:=Chr(DataKey[3])+Chr(DataKey[4]);
        Val(PosBaris,NoBaris,Error);
        Val(PosKolom,NoKolom,Error);
        xk:=Wherex;
        yk:=wherey;
        Gotoxy(NoKolom,NoBaris);
        Write(Chr(DataKey[5]));
        gotoxy(xk,yk);
    End;
End;
End;

Procedure User1; {Routine user 1 yang akan dijalankan dengan interrupt $5}
Interrupt;
Begin
If Tanya = 'N' Then
Begin
Tanya := 'Y';
InitSerial;
TransmitSerial($01); {Permintaan hubungan pada supervisor}
Tanya := 'N';
End;
End;

Procedure User2; {Routine yang akan dijalankan dengan interrupt $5}
Interrupt;
Begin
If Tanya = 'N' Then
Begin
Tanya := 'Y';
InitSerial;
TransmitSerial($02); {Permintaan hubungan pada supervisor}
```

```
Tanya := 'N';
End;
End;

Procedure ShowSuperVisor; {Routine supervisor yang akan dijalankan dengan interrupt $1c}
Interrupt;                {Bila ada permintaan hubungan oleh user}
Begin
If Port[$305] and 1 = 1 Then
Begin
    If Port[$300] = $01 Then {Bila oleh user 1}
    Begin
        x1:=wherex;
        y1:=wherey;
        Gotoxy(1,1);
        textcolor(4+blink);
        textbackground(7);
        Write('User 1');
        gotoxy(x1,y1);
    End
    Else
    If Port[$300] = $02 Then {Bila oleh user 2}
    Begin
        x1:=wherex;
        y1:=wherey;
        Gotoxy(1,1);
        textcolor(4+blink);
        textbackground(7);
        Write('User 2');
        gotoxy(x1,y1);
    End
    Else
    If Port[$300] = $10 Then WriteFree
End;
End;

Procedure OnlySupervisor; {Routine Supervisor yang akan dijalankan dengan interrupt $5}
Interrupt;
```

```
Label GetOut;
Begin
If PopUp = 'M' Then {Untuk menghindari pengaktifan program resident sebanyak lebih dari 1 kali}
Begin
PopUp := 'Y';
TerimaDariMonitor := False;
Keyboard := False;
SimpanPosisiKursor;
SimpanIsiLayar;
InitSerial;
textbackground(0);
GetOut:
MainMenu;
CursorOff;
Repeat tombol:=Ord(upcase(readkey));
case tombol of                                          {Pemilihan penekanan tombol
keyboard untuk menu awal}
    $4D : Begin              {tombol m/M untuk menu monitor}
          MenuMonitor;
          NomorMonitor := '0';
          repeat tombol:=Ord(upcase(readkey));
          case tombol of
          $54 : TerimaMonitor;   {tombol t/T untuk perintah terima}
          $4B  : If Keyboard = False Then KirimMonitor Else WriteError;
                                 {tombol k/K untuk perintah kirim}
          $4E : Begin            {tombol n/N untuk pengisian nomor monitor}
                WriteBlank;
                TextColor(9);
                Repeat
                Gotoxy(41,25);Write('         ');
                CursorOn;
                Gotoxy(41,25);
                NomorMonitor := Readkey;
                DataTemp := Ord(NomorMonitor)
                Until (DataTemp > $30) And (DataTemp < $3A) Or (DataTemp = $42) Or (DataTemp =$62);
                If (DataTemp = $42) Or (DataTemp = $62) Then
                Write('Broadcast') Else
```

```
      Write(NomorMonitor);
      CursorOff;
      End;
$58 : Goto GetOut;     {tombol x/X untuk kembali ke menu awal}
else
Copyright;
end;
until tombol = $58;
end;
$4B : If (TerimaDariMonitor = True) And (NomorMonitor <> '0') Then
Begin                  {tombol k/K untuk menu keyboard}
Keyboard := True;
MenuKeyboard;
repeat tombol:=Ord(upcase(readkey));
case tombol of
$42 : Begin            {tombol b/B untuk pengisian baris}
      WriteBlank;
      TextColor(9);
      Repeat
      Gotoxy(11,25);Write(' ');
      Gotoxy(12,25);Write(' ');
      CursorOn;
      Gotoxy(11,25);
      Repeat
      BarisLayar[1] := Readkey;
      DataTemp := Ord(BarisLayar[1]);
      Until (DataTemp > $2F) And (DataTemp < $33);
      Write(BarisLayar[1]);
      Gotoxy(12,25);
      Repeat
      BarisLayar[2] := Readkey;
      DataTemp := Ord(BarisLayar[2]);
      Until (DataTemp > $2F) And (DataTemp < $3A);
      Write(BarisLayar[2]);
      Datatemp := Ord(BarisLayar[1]);
      If DataTemp = $30 Then Val(BarisLayar[2],Baris,Error)
      Else
```

```
        Begin
        PosisiBaris := BarisLayar[1]+BarisLayar[2];
        Val(PosisiBaris,Baris,Error)
        End;
        Until (Baris > 2) And (Baris < 25);
        Gotoxy(11,25);Write(' ');
        Gotoxy(12,25);Write(' ');
        GotoXY(11,25);Write(Baris);
        CursorOff;
        End;
$4F : Begin             {tombol o/O untuk pengisian kolom}
        WriteBlank;
        TextColor(9);
        Repeat
        Gotoxy(23,25);Write(' ');
        Gotoxy(24,25);Write(' ');
        CursorOn;
        Gotoxy(23,25);
        Repeat
        KolomLayar[1] := Readkey;
        Datatemp := Ord(KolomLayar[1]);
        Until (DataTemp > $29) And (DataTemp < $39);
        Write(KolomLayar[1]);
        Gotoxy(24,25);
        Repeat
        KolomLayar[2] := Readkey;
        DataTemp := Ord(KolomLayar[2]);
        Until (DataTemp > $29) And (DataTemp < $3A);
        Write(KolomLayar[2]);
        DataTemp := Ord(KolomLayar[1]);
        If DataTemp = $30 Then Val(KolomLayar[2],Kolom,Error)
        Else
        Begin
        PosisiKolom := KolomLayar[1]+KolomLayar[2];
        Val(PosisiKolom,Kolom,Error);
        End;
        Until (Kolom > 2) And (Kolom < 81);
```

```
                GotoXY(23,25);Write(' ');
                GotoXY(24,25);Write(' ');
                GotoXY(23,25);Write(Kolom);
                CursorOff;
                End;
        $43 : Begin             {tombol c/C untuk pengisian karakter}
                WriteBlank;
                TextColor(9);
                Repeat
                GotoXY(39,25);Write(' ');
                CursorOn;
                GotoXY(39,25);
                CharASCII := Readkey;
                DataTemp := Ord(CharASCII)
                Until (DataTemp > $1F) And (DataTemp < $7F);
                Write(CharASCII);
                CursorOff;
                End;
        $4B : KirimKeyboard;    {tombol k/K untuk perintah kirim}
        $58 : Goto GetOut;      {tombol x/X untuk kembali ke menu awal}
        else
        Copyright;
        end;
        until tombol = $58
        end;
    $58 : ExitMenu;             {tombol x/X untuk membuat program kembali resident}
    else
    Copyright;
    end;
until tombol = $58
End;
End;

{Main Program} {Routine ini tidak resident, untuk inisialisasi awal program}
Begin
Par := 'N';
BacaParameter;
```

```
If Par = 'N' Then
Writeln('Syntax : Switcher [0 atau 1 atau 2]')
Else
Begin
PopUp := 'N';
Tanya := 'N';
Clrscr;
CursorOff;
Parameter1;
Box(6,6,75,20,14,3);
TextColor(0);
Gotoxy(15,8);
Write('U N I V E R S I T A S   K R I S T E N   P E T R A');
Gotoxy(20,10);
Write('FAKULTAS TEKNIK JURUSAN TEKNIK ELEKTRO');
TextColor(5);
Gotoxy(24,12);
Write(' T U G A S   A K H I R');
TextColor(4);
GotoXY(25,15);
Write('SWITCHER DATA RGB MONITOR');
TextColor(1);
GotoXY(32,17);
For i:=1 to 13 do Write(Chr(Ord(Juwono[i])-i));
Gotoxy(17,18);
Write('NRP : 234891013 ; NIRM : 89.7.002.31073.44029');
TextColor(7);
TextBackGround(0);
Gotoxy(1,24);
Write('Tekan sembarang Tombol');
Gotoxy(59,24);
JT_Write;
AnyKey;
Clrscr;
Box(20,10,61,16,11,5);
TextColor(13);
Gotoxy(25,12);
```

```
Write('Monitor ID.          : ',Monitor_Id);
Gotoxy(25,14);
Write('Communication Port at : $300');
Gotoxy(25,15);
Write('Baudrate at          : 38400 Bps');
Gotoxy(1,20);
Counter := 0;
CursorOn;
InitSerial;
If Parameter[1] = '0' Then          {Pilihan untuk membuat program untuk supervisor -> resident}
Begin
setintvec($1C,@ShowSuperVisor);
setintvec($5,@OnlySupervisor);
keep(0);
End
Else
If Parameter[1] = '1' Then          {Pilihan untuk membuat program untuk user 1 -> resident}
Begin
setintvec($1C,@ShowUser1);
setintvec($5,@User1);
keep(0);
end
Else
If Parameter[1] = '2' Then          {Pilihan untuk membuat program untuk user 2 -> resident}
Begin
setintvec($1C,@ShowUser2);
setintvec($5,@User2);
keep(0);
End;
End;
End.
```