LAPORAN PROJECT AKHIR
MICROPROCESSOR SYSTEM

Kelompok 7

- Kiagus Fathur Rahman NIM
- Putu Akatara Yoga Dhaniswara NIM
- Rizky Adrian Pradana NIM
- Vinsensius Christopher Nathaniel Arden NIM

Latar Belakang

Lampu lalu lintas sangat penting dalam mengatur arus kendaraan dan pejalan kaki di persimpangan dalam kehidupan sehari hari. Lampu lalu lintas adalah lampu yang memiliki 3 warna yaitu : Merah, Kuning, dan Hijau yang dikendalikan dengan microcontroller dengan cara mengatur timer dan perubahan warna ini, salah satu microcontroler yang digunakan untuk mengatur pergerakan/perubahan lampu lalu lintas adalah tipe 8051.

Microcontroler 8051 berfungsi sebagai unit pusat kendali untuk sistem lampu lalu lintas. Itu mengeksekusi program yang mengimplementasikan mesin negara yang terbatas untuk mengatur urutan lampu lalu lintas.

Tujuan

Praktikum ini bertujuan untuk mensimulasikan bagaimana cara kerja microcontroller pada sebuah lampu lalu lintas untuk mengatur timer maupun perpindahan lampu pada sistem lampu lalu lintas.

Data dan Analisis

Code Dengan Instruksi 8051 Lampu Lalu Lintas

```
org 0h

ljmp main


org 000Bh

ljmp isr


org 13h

ljmp cekbutton


main:

     LED_PORT EQU P1                ; LED Port (Port 1)
```

```
     mov tmod, #00100001B          ; Set Timer 0 in mode 1
(16-bit mode) and Timer 1 in mode 2 (8-bit auto-reload mode)

     mov tl0, #0fdh                ; Set initial value for Timer
0 low byte

     mov th0, #04bh                ; Set initial value for Timer
0 high byte

     mov IE, #10000110B            ; Enable Timer 0 and Timer 1
interrupts

     mov IP, #00000100B            ; Set Timer 0 interrupt
priority to high

     mov r0, #0

     mov r1, #5

     mov a, #1


here:

ceknilaiA1: cjne a, #1, ceknilaiA2

     mov r2, #0b6h

     mov r3, a

     mov a, #0

     setb tr0                     ; Start Timer 0

     sjmp here

ceknilaiA2: cjne a, #2, ceknilaiA3

     mov r2, #24h

     mov r3, a

     mov a, #0

     setb tr0                     ; Start Timer 0

     sjmp here

ceknilaiA3: cjne a, #3, ceknilaiA4
```

```
     mov r2, #6dh
     mov r3, a
     mov a, #0
     setb tr0                    ; Start Timer 0
     sjmp here
ceknilaiA4: cjne a, #4, ceknilaiA5
     mov r2, #0dbh
     mov r3, a
     mov a, #0
     setb tr0                    ; Start Timer 0
     sjmp here
ceknilaiA5: cjne a, #5, ceknilaiA1
     mov r2, #0b6h
     mov a, #1
     sjmp here


isr:
mov tl0, #0fdh                ; Reload Timer 0 low byte with
initial value
mov th0, #04bh                ; Reload Timer 0 high byte with
initial value
dec r1
cjne r1, #0, not_eq
MOV LED_PORT, r2              ; Output the value of r2 to the
LED port
Mov r1, #5
```

```
mov tl0, #0fdh                ; Reload Timer 0 low byte with
initial value

mov th0, #04bh                ; Reload Timer 0 high byte with
initial value

mov a, r3

inc a

reti


not_eq:

reti


cekbutton:

jb p2.7,  cekbutton

cekbutton7: jnb p2.7, cekbutton7

dec r3

acall munculinstatuslampu

setb tr0

ret


munculinstatuslampu:

mov th1, #-6                  ; Set Timer 1 to generate a 9600
baud rate at a 12MHz crystal frequency

mov scon, #50h                ; Set UART in mode 1 (8-bit UART
with variable baud rate)

setb tr1                      ; Start Timer 1

ceknilaiR1: cjne r3, #1, ceknilaiR2

acall outputmerah
```

```
ret
ceknilaiR2: cjne r3, #2, ceknilaiR3
acall outputmerahkuning
ret
ceknilaiR3: cjne r3, #3, ceknilaiR4
acall outputkuning
ret
ceknilaiR4: cjne r3, #0, ceknilaiR1
acall outputhijau
ret

outputmerah:
mov sbuf, #52h
acall trans
mov sbuf, #45h
acall trans
mov sbuf, #44h
acall trans
ret

outputmerahkuning:
mov sbuf, #52h
acall trans
mov sbuf, #45h
acall trans
```

```
mov sbuf, #44h
acall trans
mov sbuf, #26h
acall trans
mov sbuf, #59h
acall trans
mov sbuf, #45h
acall trans
mov sbuf, #4Ch
acall trans
mov sbuf, #4Ch
acall trans
mov sbuf, #4Fh
acall trans
mov sbuf, #57h
acall trans
ret

outputkuning:
mov sbuf, #59h
acall trans
mov sbuf, #45h
acall trans
mov sbuf, #4Ch
acall trans
```

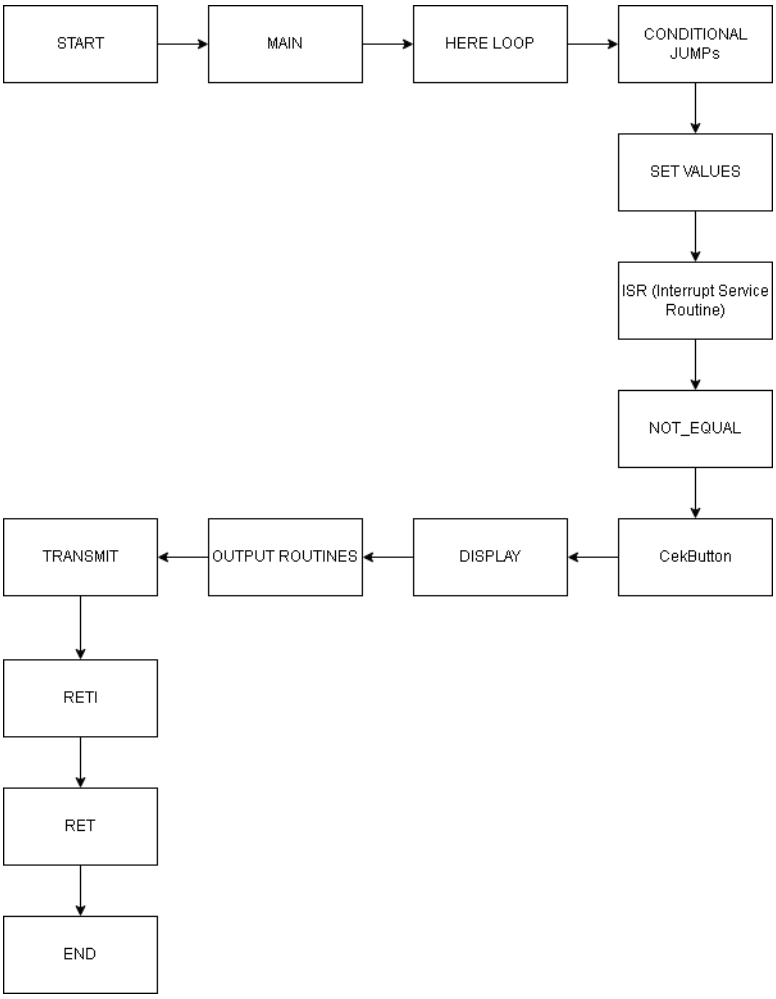
```
mov sbuf, #4Ch
acall trans
mov sbuf, #4Fh
acall trans
mov sbuf, #57h
acall trans
ret

outputhijau:
mov sbuf, #47h
acall trans
mov sbuf, #52h
acall trans
mov sbuf, #45h
acall trans
mov sbuf, #45h
acall trans
mov sbuf, #4Eh
acall trans
ret

trans:
cekti: jnb ti,cekti
clr ti
Ret
```

```
End
```

Block Diagram



Tahapan block diagram di awal dari awal hingga akhir:

1. START: Menginisialisasi kan program untuk mulai
2. MAIN: Instruksi ini diadakan untuk mensetting timer microcontroller, interrupt settings dan menginisialisasi beberapa register.
3. HERE LOOP: Program mengecek jika nilai di Accumulator (a) sama dengan 1. Jika iya, maka accumulator akan di clear, dan Timer 0 dipakai, dan ketika program memasuki infinite loop. Namun, jika nilai tidak sama dengan 1, maka program akan lompat ke label selanjutnya.

4. CONDITIONAL JUMPS: Melakukan operasi berbeda berdasarkan nilai register a. Ini menetapkan nilai spesifik dalam register r2, r3, dan a, dan kemudian menetapkan bendera TR0. Program kemudian melanjutkan eksekusi dari label ‘HERE’.
5. SET VALUES: Untuk meng-inisialisasi nilai-nilai dalam register atau memori tertentu kemudian dapat digunakan dalam operasi selanjutnya dalam program.
6. ISR: ISR (Interupt Service Routine) yaitu routine-routine yang khusus dijalankan sebagai layanan dari sebuah interupt
7. NOT_EQUAL: Melakukan instruksi return dari interrupt service routine dan mengizinkan agar interrupt terjadi.
8. CekButton: Untuk memeriksa keadan switch tertentu pada mikrokontroler dimana itu merupakan tempat logika pengontrolan tombol.
9. DISPLAY: untuk menampilkan text pada UART display
10. OUTPUT ROUTINE: Serangkaian instruksi yang digunakan untuk menghasikan output dari microcontroller ke perangkat external seperti LED, LCD, motor, dll
11. TRANSMIT: Transmit atau ‘trans’ digunakan untuk melakukan pengiriman data yang berbentuk string dari mikroprosesor ke display serial UART.
12. RETI: Berfungsi untuk kembali ke alamat interrupt yang tertunda setelah suatu interupt selesai diproses
13. RET: Untuk mengakhiri subrutin lalu kembali pada eksekusi utama dari titik dimana subrutin dipanggil.
14. END: Merupakan langkah akhir dari program.

Pada blok diagram di atas, terdapat mikrokontroler 8051 sebagai pusatnya. Mikrokontroler ini terhubung dengan beberapa periferal, yaitu:

1. Timer 0: Timer ini digunakan untuk mengatur waktu dan generate interrupt yang digunakan dalam kode untuk mengendalikan tindakan berdasarkan waktu.
2. LED (P1): LED dihubungkan ke Port 1 (P1) pada mikrokontroler. Port ini digunakan untuk mengontrol LED berdasarkan nilai yang dihasilkan dalam kode.
3. Timer 1: Timer ini digunakan untuk mengatur baud rate UART.
4. UART: UART (Universal Asynchronous Receiver Transmitter) digunakan untuk mentransmisikan data melalui komunikasi serial.

Mikrokontroler 8051 berkomunikasi dengan periferal-periferal ini melalui pin-pin yang ditentukan pada chip mikrokontroler. Pin-pin tersebut terhubung dengan pin-pin input/output (I/O) pada periferal-periferal yang sesuai. Dalam kode tersebut, pengaturan port dan konfigurasi periferal-periferal dilakukan melalui instruksi Assembly.

Deskripsi Sistem

Sistem ini merupakan sebuah program yang mengambil dasar dari bagaimana sistem lampu lalu lintas bekerja. Sistem pada program tersebut bekerja sebagai berikut:

- ➔ Kode dimulai dengan org 0h yang menunjukkan bahwa program dimulai dari alamat memori 0x0000 dan ljmp main untuk melompat ke subrutin main.
- ➔ Subrutin main digunakan untuk menginisialisasi beberapa variabel dan periferal yang akan digunakan. Port LED (LED_PORT) ditentukan sebagai P1. Timer 0 dan Timer 1 diatur dalam mode yang sesuai. Nilai awal untuk Timer 0 diatur dan interrupt Timer 0 dan Timer 1 diaktifkan. Variabel r0, r1, dan a diinisialisasi dengan nilai tertentu.
- ➔ Program melanjutkan ke label here dan memeriksa nilai variabel a menggunakan instruksi cjne. Jika nilai a memenuhi kondisi tertentu, nilai r2 dan r3 diatur sesuai dengan kondisi tersebut. Setelah itu, Timer 0 diaktifkan dengan instruksi setb tr0 dan program akan melompat kembali ke label here.
- ➔ Jika nilai a tidak memenuhi kondisi di ceknilaiA1, ceknilaiA2, ceknilaiA3, atau ceknilaiA4, maka program akan melompat ke ceknilaiA5. Di sini, r2 diatur kembali dan nilai a juga diatur kembali menjadi 1. Kemudian, program akan melompat kembali ke label here.
- ➔ Ketika Timer 0 mencapai interrupt, program akan melompat ke subrutin isr. Timer 0 diatur ulang dengan nilai awal, dan variabel r1 dikurangi 1. Jika r1 masih lebih besar dari 0, program melompat ke label not_eq untuk menghindari output LED. Jika r1 telah mencapai 0, nilai r2 dikeluarkan ke port LED (LED_PORT) dan Timer 0 diatur ulang. Nilai variabel r3 juga ditingkatkan dan subrutin isr selesai dengan instruksi reti.
- ➔ Jika r1 tidak sama dengan 0, program akan melompat ke label not_eq dan langsung mengakhiri subrutin isr dengan instruksi reti.
- ➔ Jika tombol di p2.7 ditekan, program akan melompat ke subrutin cekbutton. Jika tombol tidak ditekan, program akan tetap berada di subrutin cekbutton hingga tombol ditekan. Setelah tombol ditekan, nilai r3 dikurangi 1 dan program memanggil subrutin munculinstatuslampu. Setelah itu, Timer 0 diaktifkan dan program kembali ke tempat pemanggilan subrutin cekbutton.
- ➔ Subrutin munculinstatuslampu mengatur Timer 1 untuk menghasilkan baud rate 9600. Setelah itu, kondisi variabel r3 diperiksa menggunakan instruksi cjne. Berdasarkan kondisi tersebut, subrutin munculinstatuslampu memanggil subrutin outputmerah, outputmerahkuning, outputkuning, atau outputhijau. Setelah setiap pemanggilan subrutin, program kembali ke subrutin munculinstatuslampu dan melanjutkan pemrosesan kondisi berikutnya.
- ➔ Subrutin outputmerah, outputmerahkuning, outputkuning, dan outputhijau digunakan untuk mengirim data karakter ke UART. Setelah mengirim karakter, subrutin trans digunakan untuk menunggu hingga karakter dikirim sebelum mengirim karakter berikutnya.
- ➔ Setelah selesai mengirim karakter, subrutin munculinstatuslampu kembali ke tempat pemanggilan subrutin dan program melanjutkan pemrosesan berikutnya.

Pada dasarnya, program ini menggunakan mikrokontroler 8051 untuk mengendalikan LED dan berkomunikasi melalui UART. Timer digunakan untuk mengatur waktu dan interrupt digunakan untuk mengendalikan tindakan berdasarkan waktu. Pemeriksaan tombol juga dilakukan untuk memicu tindakan tertentu.

Penjelasan Implementasi Pada Program

3.1 Setiap “SUBROUTINES” Pada Program

Setiap subroutine melakukan tugas tertentu dan dipanggil dari bagian-bagian yang berbeda dalam program sesuai kebutuhan.

'munculinstatuslampu' adalah subrutin yang dipanggil dari rutin 'cekbutton'. Subroutine ini memeriksa nilai 'r3' dan memanggil rutin output yang berbeda berdasarkan nilai tersebut.

'outputmerah' adalah subroutine yang dipanggil dari 'munculinstatuslampu' ketika 'r3' sama dengan 1. Subrutin ini mengatur nilai yang sesuai di 'sbuf' untuk menampilkan output berwarna merah.

'outputmerahkuning' adalah subroutine yang dipanggil dari 'munculinstatuslampu' ketika 'r3' sama dengan 2. Subroutine ini mengatur nilai yang sesuai di 'sbuf' untuk menampilkan output berwarna merah-kuning.

'outputkuning' adalah subroutine yang dipanggil dari 'munculinstatuslampu' ketika 'r3' sama dengan 3. Subroutine ini mengatur nilai yang sesuai di 'sbuf' untuk menampilkan output berwarna kuning.

'outputhijau' adalah subroutine yang dipanggil dari 'munculinstatuslampu' ketika 'r3' tidak sama dengan 0, 1, 2, atau 3. Subroutine ini mengatur nilai yang sesuai di 'sbuf' untuk menampilkan output berwarna hijau.

'trans' atau 'transmit' adalah subroutine yang digunakan oleh routine output ('outputmerah', 'outputmerahkuning', 'outputkuning', dan 'outputhijau') untuk mengirimkan nilai-nilai di 'sbuf'. Subrutin ini menunggu flag 'ti' diatur dan kemudian menghapusnya.

3.2 Port Setup

Terdapat beberapa port pada program yang berfungsi agar program berjalan sesuai tujuan yaitu:

1. **P1**

   Dalam potongan kode yang diberikan, terdapat pengaturan port **P1** sebagai pengaturan untuk panel LED. Berikut adalah penjelasan mengenai cara pengaturan tersebut:

   Pada bagian awal program, terdapat definisi konstanta **LED_PORT EQU P1**, yang memberikan alias atau label **LED_PORT** untuk port **P1**. Dengan demikian, port **P1** diidentifikasi sebagai **LED_PORT** dalam program.

   Selanjutnya, pada bagian main, terdapat pengaturan untuk port **P1** yang digunakan sebagai panel LED. Langkah-langkah pengaturan tersebut adalah:

   - **LED_PORT EQU P1**: Perintah ini memberikan label atau alias **LED_PORT** untuk port **P1**. Hal ini memudahkan dalam mengacu pada port **P1** sebagai **LED_PORT** dalam program.
   - **MOV LED_PORT, r2**: Perintah ini mengambil nilai yang ada di register **r2** dan memindahkannya ke port **P1 (LED_PORT)**. Nilai yang ditempatkan di register **r2** akan mengontrol pencahayaan atau keadaan LED yang terhubung ke port **P1**.

   Dengan cara ini, melalui pengaturan tersebut, port **P1** dapat digunakan sebagai pengaturan untuk panel LED. Nilai yang ditetapkan pada register **r2** akan mengendalikan pencahayaan LED yang terhubung ke port **P1** sesuai dengan kebutuhan program.

2. **P2.0**

   Port **P2.0** digunakan sebagai *setup* 8051 dengan hardware interrupt, berikut merupakan langkah dalam membuat port **P2.0** melakukan *setup:*

   - Port dapat diatur dalam mode input atau output. Untuk mengonfigurasikan **P2.0** sebagai input, maka program harus mengatur bit 0 pada register **P2** (**P2.0**) menjadi 1, contohnya, setb **P2.0** akan mengatur **P2.0** sebagai input.
   - Aktifkan fitur interrupt pada port **P2.0** dengan cara mengatur bit 0 pada register IE1 (P2.0IE) menjadi 1.
   - Atur prioritas interrupt: Lakukan dengan cara mengatur bit 0 pada register IP1 (P2.0IP) sesuai dengan prioritas yang ditujukan.

3. **P2.7**

port **P2.7** sebagai tombol switch pada panel modul LED. Berikut adalah penjelasan mengenai cara pengaturan tersebut:

Pada bagian '**cekbutton**', terdapat perintah-perintah yang mengatur penggunaan port **P2.7** sebagai tombol switch. Berikut ini adalah penjelasan langkah-langkahnya:

- '**jb p2.7, cekbutton**': Perintah ini melakukan pengecekan apakah bit 7 dari port **P2** (**P2.7**) telah diatur menjadi logika 1. Jika iya, program akan melanjutkan ke langkah selanjutnya. Jika tidak, program akan tetap berada di titik ini dan terus memeriksa hingga bit tersebut berubah menjadi logika 1.
- '**cekbutton7: jnb p2.7, cekbutton7**': Setelah langkah pertama, program akan melakukan pengecekan terus-menerus terhadap bit 7 dari port **P2 (P2.7)** dengan menggunakan perintah jnb (jump if not bit set). Jika bit tersebut tetap tidak diatur menjadi logika 1, program akan terus berada di titik ini dan melakukan pengecekan secara berulang hingga bit tersebut berubah menjadi logika 1.
- Setelah bit 7 dari port **P2** (**P2.7**) berubah menjadi logika 1, program akan melanjutkan ke perintah-perintah selanjutnya yang berada di dalam subrutin '**cekbutton**'. Subrutin ini berfungsi untuk menangani aksi yang terjadi saat tombol ditekan.

4. **SBUF**

Port **SBUF** digunakan sebagai *setup* untuk 8051 agar dapat melakukan *setup* kepada serial port. Terdapat pengaturan untuk port SBUF yang digunakan sebagai pengaturan pada serial port. Berikut adalah penjelasan mengenai cara pengaturan tersebut:

MOV SBUF, #nilai: Perintah ini digunakan untuk memindahkan nilai ke port **SBUF**. Port **SBUF** adalah bagian dari serial port yang digunakan untuk mengirim atau menerima data secara serial.

Dalam kode yang diberikan, terdapat beberapa subrutin yang menggunakan perintah **MOV SBUF, #nilai** untuk mengatur port **SBUF** sebagai berikut:

outputmerah: Subrutin ini mengatur port **SBUF** dengan nilai yang diperlukan untuk menampilkan output berwarna merah.

outputmerahkuning: Subrutin ini mengatur port **SBUF** dengan nilai yang diperlukan untuk menampilkan output berwarna merah-kuning.

outputkuning: Subrutin ini mengatur port **SBUF** dengan nilai yang diperlukan untuk menampilkan output berwarna kuning.

outputhijau: Subrutin ini mengatur port **SBUF** dengan nilai yang diperlukan untuk menampilkan output berwarna hijau.

Pada setiap subrutin tersebut, perintah **MOV SBUF, #nilai** digunakan untuk mengatur port **SBUF** agar mengirimkan data yang sesuai dengan output yang diinginkan.

Dengan melakukan pengaturan pada port **SBUF**, program dapat mengirimkan data melalui serial port untuk berkomunikasi dengan perangkat eksternal atau menampilkan output yang diinginkan melalui komunikasi serial.

3.3 Timer

Timer yang digunakan dapat dilihat dalam potongan kode:

TMOD "**#00100001B**"

Pada kode dapat dilihat ada 2 yaitu:

- **Timer 0**:

  Mode: Timer 0 diatur dalam mode 1 (16-bit mode).

  Nilai Awal: Timer 0 diatur dengan nilai awal tertentu menggunakan instruksi **MOV TL0, #nilai_low** dan **MOV TH0, #nilai_high**.

  Fungsi: Timer 0 digunakan sebagai penghitung waktu atau pengatur waktu tertentu. Dalam program ini, Timer 0 digunakan untuk mengatur interval waktu sebelum beberapa tindakan dijalankan, seperti dalam subrutin **ceknilaiA1**, **ceknilaiA2**, **ceknilaiA3**, **ceknilaiA4**, dan **ceknilaiA5**.

- **Timer 1**:

  Mode: Timer 1 diatur dalam mode 2 (8-bit auto-reload mode).

  Nilai Awal: Timer 1 tidak diberikan nilai awal secara langsung dalam potongan kode yang diberikan.

  Fungsi: Timer 1 digunakan sebagai penghitung waktu atau pengatur waktu tertentu. Dalam program ini, Timer 1 digunakan dalam subrutin **munculinstatuslampu** untuk mengatur baud rate serial pada 4800 baud dengan frekuensi kristal 12MHz. Timer 1 juga digunakan dalam subrutin trans untuk mengatur kecepatan transmisi data melalui serial port.

Kegunaan dari penggunaan timer dalam potongan kode ini adalah untuk mengontrol waktu dan interval tertentu dalam program. Timer memungkinkan program untuk melakukan tindakan atau perhitungan berdasarkan waktu yang telah ditentukan, seperti

mengendalikan lampu LED dengan interval waktu tertentu atau mengatur kecepatan transmisi data melalui serial port.

3.4 Serial

Pada program kita menggunakan serial yang deklarasinya dapat dilihat dari potongan kode:

MOV TH1, #-6

MOV SCON, #50H

Yang dimana nilai "TH1" yang dimuat ke dalam register adalah "#-6" dan nilai "SCON" yang dimuat ke dalam register adalah "#50H",

Ini berarti kecepatan baud yang akan digunakan adalah 4800 baud untuk panel data RX.

Output panel serial UART diperoleh dari subroutine berikut :

1. **outputmerah**: Mengirim karakter 'R', 'E', 'D' ke UART.
2. **outputmerahkuning**: Mengirim karakter 'R', 'E', 'D', '&', 'Y', 'E', 'L', 'L', 'O', 'W' ke UART.
3. **outputkuning**: Mengirim karakter 'Y', 'E', 'L', 'L', 'O', 'W' ke UART.
4. **outputhijau**: Mengirim karakter 'G', 'R', 'E', 'E', 'N' ke UART.

Tahapan terakhir sebelum dikirim ada pada subroutine '**Trans**' yang bertanggung jawab untuk mentransmisikan karakter melalui UART (Universal Asynchronous Receiver Transmitter). Berikut adalah penjelasan langkah-langkah yang terjadi dalam subrutin tersebut:

1. Label 'cekti' digunakan untuk mengecek apakah bit 'ti' (transmit interrupt flag) telah diatur.
2. Jika bit 'ti' belum diatur, program akan tetap berada dalam loop 'cekti' untuk menunggu hingga bit 'ti' diatur.
3. Setelah bit 'ti' diatur, program akan menghapus (clear) bit 'ti'.
4. Kemudian, program akan kembali ke subrutin pemanggil.

Dengan demikian, subrutin 'trans' memastikan bahwa karakter dikirimkan melalui UART hanya setelah bit 'ti' siap untuk ditransmisikan.

3.5 Komunikasi antar 8051 Dengan Periferal Lain

Port LED:

Port LED didefinisikan sebagai LED_PORT dan dipetakan ke Port 1 (P1).

Di dalam subrutin isr, nilai yang disimpan di r2 dipindahkan ke port LED menggunakan instruksi: MOV LED_PORT, r2.

Komunikasi ini memperbarui status LED yang terhubung ke Port 1 berdasarkan nilai dalam r2.

Button:

Input button dideteksi menggunakan Port 2, Pin 7 (P2.7).

Pada subrutin cekbutton, program menunggu hingga button ditekan (jb p2.7, cekbutton).

Setelah button ditekan, program berlanjut ke label cekbutton7 dan menunggu hingga tombol dilepas (jnb p2.7, cekbutton7).

Komunikasi ini memungkinkan program untuk merespons penekanan dan pelepasan tombol.

Komunikasi UART:

Komunikasi UART diatur menggunakan Timer 1 (T1) dan serial control register (SCON).

Di dalam subrutin munculinstatuslampu, Timer 1 dikonfigurasi dengan baud rate tertentu (mov th1,#-6 dan mov scon,#50h).

Transmisi UART dimulai dengan menyetel Timer 1 run flag (setb tr1) dan dibersihkan dengan memeriksa flag interupsi transmisi (jnb ti,cekti dan clr ti).

Trans subrutin digunakan untuk memeriksa bendera interupsi transmisi (ti) dan menghapusnya jika perlu.

Berbagai subrutin keluaran mengirim karakter ASCII (sbuf) tertentu ke UART dengan memanggil trans setelah setiap karakter.

Komunikasi ini memungkinkan komunikasi serial antara mikrokontroler 8051 dan perangkat eksternal melalui UART.