de K I T S

PC-LINK

SERIAL PPI

Trademarks & Copyright

PC-LINK					
Serial (COM) Port	*Parallel (LPT) Port*	*USB*	*Firewire*	*ISA slot*	*PCI slot*
✓					

Daftar Isi

1. PENDAHULUAN

PC-Link SERIAL PPI merupakan pengendali 40 bit jalur input/output melalui antarmuka UART RS-232 yang dapat dihubungkan ke komputer secara langsung. Contoh aplikasi dari SERIAL PPI adalah sebagai pengendali tampilan LED, sebagai pembaca kondisi saklar, penghitung pulsa counter, dan lain-lain.

1.1. SPESIFIKASI EKSTERNAL SERIAL PPI

Spesifikasi Eksternal SERIAL PPI adalah sebagai berikut:

- Menggunakan antarmuka UART RS-232.
- 4 pilihan Baud Rate.
- 16 bit jalur Input/Output (Port 1 dan Port 2) dengan level CMOS.
- 24 bit jalur Programmable Peripheral Interface 82C55 (Port A, Port B, dan Port C) dengan level CMOS.
- 2 Counter 16 bit (Counter 0 dan Counter 1) dengan level CMOS.
- Sumber tegangan input 12 VDC.
- Tersedia Voltage Regulator dengan tegangan output 5 VDC.

1.2. SPESIFIKASI INTERNAL SERIAL PPI

Dalam penggunaan dari UART SERIAL PPI dikenal adanya Protocol Layer.

UART Protocol Layer adalah lapisan yang dipergunakan untuk mengatur semua lalu lintas data dan sudah tersusun sesuai kegunaan menjadi paket Sub-rutin.
Adapun daftar API Command terdapat pada **bagian 3.2**.

1.3. SISTEM YANG DIANJURKAN

Perangkat Keras:

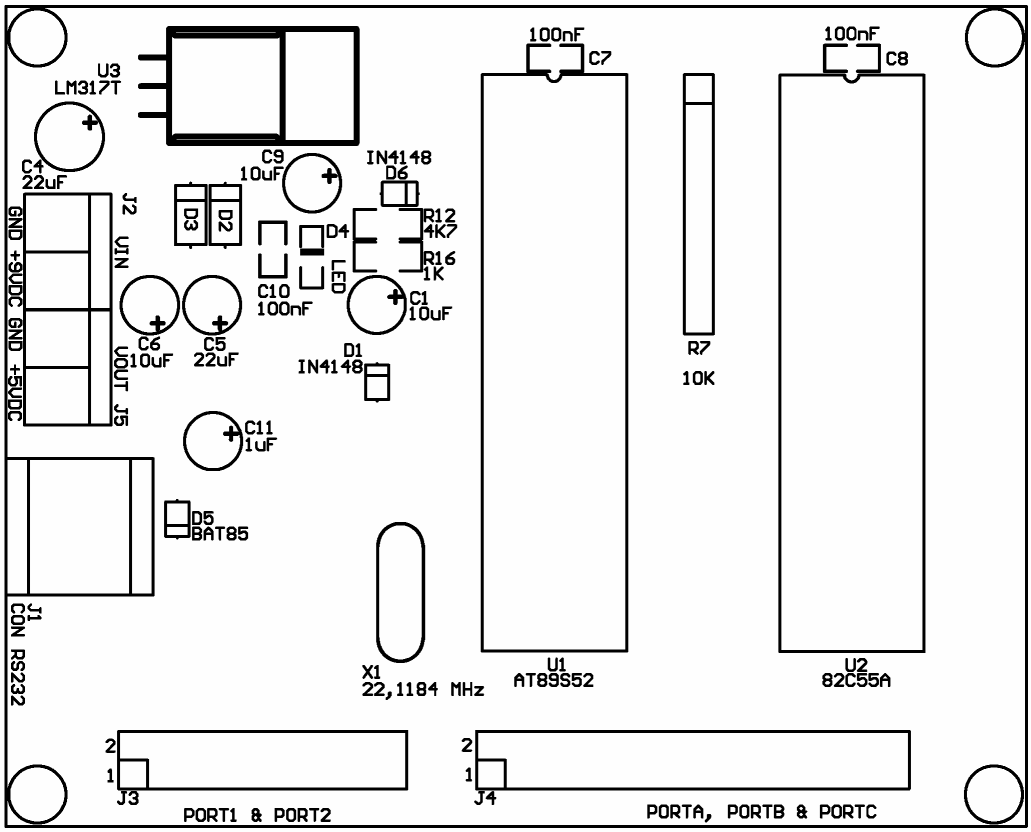
- PC AT Pentium® IBM Compatible dengan Serial Port (COM1 / COM2).
- CD-ROM Drive.
- Ruang hard disk minimum 2 Mbytes.

Perangkat Lunak:

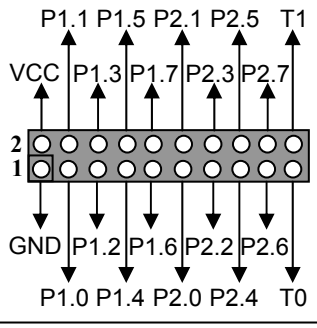
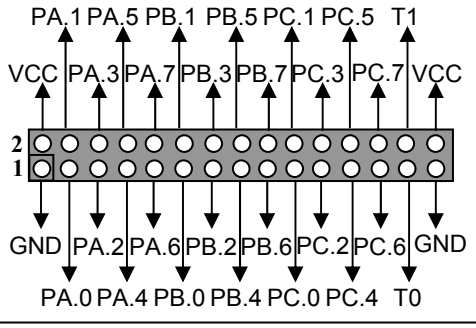
- Sistem Operasi MS-DOS®, PC-DOS™, atau Windows® 9x ke atas.
- Borland© Delphi 5.0 atau Turbo© Pascal 7.0.
- File-file dalam CD:
  SERPPI.EXE, PASPPI.EXE, SERLIB.DLL, SERPAS.TPU, MANUAL SERIAL PPI.PDF, QUICK START SERIAL PPI.PDF, 89S52.PDF, 82C55.PDF

2. PERANGKAT KERAS SERIAL PPI

2.1. TATA LETAK KOMPONEN SERIAL PPI



2.2. ALOKASI DAN SPESIFIKASI PORT

Konfigurasi J3	Konfigurasi J4
	

Spesifikasi untuk Port 1, 2, Counter 0, dan Counter 1 adalah sebagai berikut:

Simbol	Parameter	Nilai	Satuan
I_{OL}	Arus saat output berlogika '0'	1,6	mA
I_{OH}	Arus saat output berlogika '1'	-10	µA

I_{OL} maksimum per pin adalah 10 mA.
I_{OL} maksimum per port adalah 15 mA.
I_{OL} maksimum untuk semua port adalah 71 mA.

Spesifikasi untuk Port A, B, dan C adalah sebagai berikut:

Simbol	Parameter	Nilai	Satuan
I_{OL}	Arus saat output berlogika '0'	2,5	mA
I_{OH}	Arus saat output berlogika '1'	-100	μA

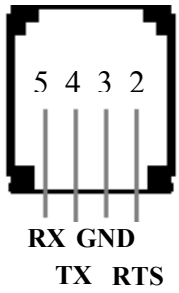
Keterangan lebih lanjut dapat dilihat pada datasheet IC yang bersangkutan.

2.3. HUBUNGAN SERIAL PPI DENGAN KOMPUTER

SERIAL PPI dapat dihubungkan dengan COM port komputer atau dengan kontroler lain yang juga memiliki interface UART RS-232. Perhatikan hubungan jalur komunikasinya.

COM Port Komputer DB9 Male	SERIAL PPI RJ11 Female
RTS (Pin 7)	RTS (Pin 2)
GND (Pin 5)	GND (Pin 3)
TX (Pin 3)	TX (Pin 4)
RX (Pin 2)	RX (Pin 5)

J1 Tampak Depan



2.4. MENCOBA SERIAL PPI DENGAN SERPPI.EXE

- Hubungkan kabel serial antara SERIAL PPI dan komputer.
- Hubungkan perangkat output (misalnya rangkaian LED) ke port SERIAL PPI.
- Hubungkan sumber tegangan. Hubungkan juga referensi Ground antara rangkaian tambahan dengan Ground pada SERIAL PPI.
- Jalankan program SERPPI.EXE (under Windows®), pilih COM yang digunakan dan tekan tombol **Start**. LED yang dihubungkan pada port SERIAL PPI akan menyala bergantian.

2.5. MENCOBA SERIAL PPI DENGAN PASPPI.EXE

- Hubungkan kabel serial antara SERIAL PPI dan komputer.
- Hubungkan perangkat output (misalnya rangkaian LED) ke port SERIAL PPI.
- Hubungkan sumber tegangan. Hubungkan juga referensi Ground antara rangkaian tambahan dengan Ground pada SERIAL PPI.
- Jalankan program PASPPI.EXE (under DOS). Ketik COM port yang digunakan dan tekan Enter. LED yang dihubungkan pada port SERIAL PPI akan menyala bergantian.

3. PERANGKAT LUNAK SERIAL PPI

Waktu yang dibutuhkan SERIAL PPI mulai menyala hingga siap dioperasikan (Start-up Time) = 25 ms.

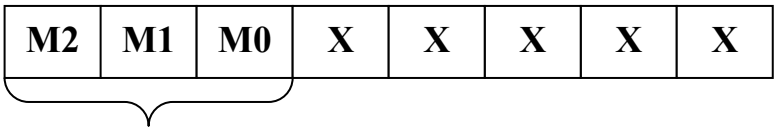
3.1. SPESIFIKASI UART RS-232

Secara default, komunikasi UART RS-232 bekerja pada **Baud Rate 9600 bps, 8 Data Bit, No Parity Bit, 1 Stop Bit, No Flow Control**. Pilihan baud rate yang lain terdapat pada **bagian 3.2.4.**

3.2. COMMAND

Semua transaksi data selalu dimulai dengan mengirimkan Command ke SERIAL PPI.

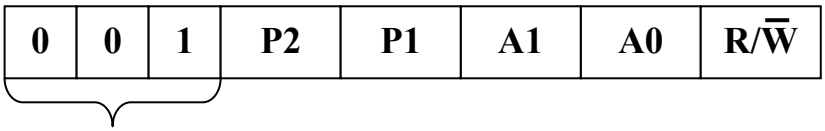
Command



M2	M1	M0	MODE
0	0	0	*Tidak Terpakai*
0	0	1	Byte Transfer
0	1	0	Bit Set/Reset
0	1	1	Counter
1	0	0	Baud Rate
1	1	X	*Tidak Terpakai*

3.2.1. BYTE TRANSFER

Command



Command untuk Byte Transfer akan mengakses salah satu dari Port 1, Port 2, Port A, Port B, Port C, atau Control Word. Mengakses lebih dari satu port pada saat yang sama tidak diperkenankan.

Proses tulis/output dan proses baca/input dibedakan oleh R/$\overline{W}$. R/$\overline{W}$ diberi nilai 1 untuk proses pembacaan dari port dan diberi nilai 0 untuk proses penulisan ke port.

Proses baca akan diikuti oleh satu byte hasil pembacaan port yang dikirim oleh modul SERIAL PPI. Proses tulis harus diikuti oleh satu byte yang akan dikirimkan ke port.

Mode	P2	P1	A1	A0	R/$\overline{W}$	Proses
001	0	0	0	0	0	Tulis ke Port A
	0	0	0	0	1	Baca dari Port A
	0	0	0	1	0	Tulis ke Port B
	0	0	0	1	1	Baca dari Port B
	0	0	1	0	0	Tulis ke Port C
	0	0	1	0	1	Baca dari Port C
	0	0	1	1	0	Tulis ke Control Word
	0	0	1	1	1	*Tidak Terpakai*
	0	1	0	0	0	Tulis ke Port 1

Mode	P2	P1	A1	A0	R/$\overline{W}$	Proses
001	0	1	0	0	1	Baca dari Port 1
	0	1	0	1	X	*Tidak Terpakai*
	0	1	1	X	X	*Tidak Terpakai*
	1	0	0	0	0	Tulis ke Port 2
	1	0	0	0	1	Baca dari Port 2
	1	0	0	1	X	*Tidak Terpakai*
	1	0	1	X	X	*Tidak Terpakai*
	1	1	X	X	X	*Tidak Terpakai*

Secara default, kondisi semua port pada saat awal adalah:
- Port 1 dan Port 2 : sebagai input, bernilai FFH
- Port A, Port B, Port C : Tidak terprogram sebagai input maupun output

Untuk memprogram Port A, B, dan C, proses inisialisasi harus diakukan dengan mengirim 1 byte ke Control Word sebelum ketiga port digunakan.

MSB (Most Significant Bit) LSB (Least Significant Bit)

D7	D6	D5	D4	D3	D2	D1	D0

Simbol	Fungsi
D7	Set Flag, diberi nilai 1 untuk mengaktifkan PPI Port.
D6 & D5	Mode Select untuk Port A dan C Upper (bit 7 – bit 4), bernilai 00 untuk mode 0, 01 untuk mode 1, dan 10 atau 11 untuk mode 2. Umumnya diberi nilai 00.
D4	Port A, diberi nilai 1 untuk input dan diberi nilai 0 untuk output.
D3	Port C Upper, diberi nilai 1 untuk input dan diberi nilai 0 untuk output.
D2	Mode Select untuk Port B dan C Lower (bit 3 – bit 0), diberi nilai 0 untuk mode 0 dan diberi nilai 1 untuk mode 1. Umumnya diberi nilai 0.
D1	Port B, diberi nilai 1 untuk input dan diberi nilai 0 untuk output.
D0	Port C Lower, diberi nilai 1 untuk input dan diberi nilai 0 untuk output.

Untuk penggunaan mode lain, lihat keterangan lebih lengkap pada datasheet 82C55.
Selain diakses per byte, PPI Port dapat juga diakses per bit (Bit Set/Reset). Akses per bit hanya terbatas pada Port C saja dan hanya sebagai output saja. Meski menggunakan Port C, tetapi akses per bit bukan berarti mengakses langsung ke alamat Port C. Akses per bit dapat dilakukan dengan mengirim 1 byte ke Control Word.

MSB LSB

D7	D6	D5	D4	D3	D2	D1	D0

Simbol	Fungsi
D7	Set Flag, diberi nilai 0 untuk mengaktifkan akses per bit.
D6, D5, D4	Bit bersifat don't care (dapat diberi nilai 0 atau 1). Disarankan untuk mengisi dengan nilai 0.
D3, D2, D1	Bit Select untuk menentukan bit pada Port C yang akan diakses. Keterangan lebih jelas ditunjukkan oleh tabel 2.
D0	Bit Set/Reset untuk Port C. Diberi nilai 1 untuk Set (memberi logika 1 pada bit) dan 0 untuk Reset (memberi logika 0 pada bit).

	Bit dari Port C							
	7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
D3	1	1	1	1	0	0	0	0
D2	1	1	0	0	1	1	0	0
D1	1	0	1	0	1	0	1	0

3.2.2. BIT SET/RESET

Command

0	1	0	P2/$\overline{P1}$	D2	D1	D0	V

Mode

Command untuk Bit Set/Reset akan menulis/output salah satu bit dari Port 1 atau Port 2.
Pemilihan bit ditentukan oleh nilai D0, D1, dan D2. Sedangkan nilai yang dituliskan ke bit ditentukan oleh V.

D2	D1	D0	Bit yang Diakses
0	0	0	Bit 0
0	0	1	Bit 1
0	1	0	Bit 2
0	1	1	Bit 3
1	0	0	Bit 4
1	0	1	Bit 5
1	1	0	Bit 6
1	1	1	Bit 7

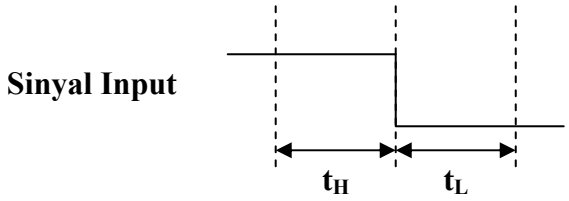
Mode	P2/$\overline{P1}$	D2	D1	D0	V	Proses
010	0	X	X	X	0	Clear/Reset (memberi logika 0) pada Bit Port 1
	0	X	X	X	1	Set (memberi logika 1) pada Bit Port 1
	1	X	X	X	0	Clear/Reset (memberi logika 0) pada Bit Port 2
	1	X	X	X	1	Set (memberi logika 1) pada Bit Port 2

3.2.3. COUNTER

Command

0	1	1	0	0	C1/$\overline{C0}$	GET/ $\overline{INIT}$	START/ $\overline{STOP}$
Mode							

Command untuk Counter akan mengakses salah satu dari Counter 0 atau Counter 1.
Start dan **Stop** berfungsi untuk mengaktifkan atau menghentikan counter dan hanya bekerja apabila **Init** diberi nilai 0.
Register Counter 0 dan Counter 1 akan bertambah jika pin yang bersangkutan (T0 atau T1) mendapatkan sinyal sesuai timing diagram berikut.



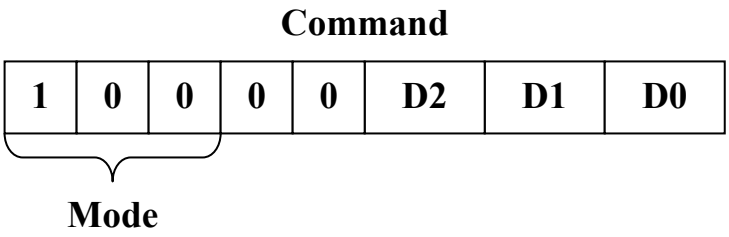
Simbol	Keterangan	Nilai	Satuan
t_H	Durasi sinyal berlogika '1' sebelum transisi	0,55	µs
t_L	Durasi sinyal berlogika '0' setelah transisi	0,55	µs

Pembacaan dengan Get akan diikuti oleh dua byte hasil pembacaan nilai counter yang dikirim oleh modul SERIAL PPI. Data pertama adalah nilai Least Significant Byte, sedangkan data kedua adalah nilai Most Significant Byte.

Mode	00	C1/$\overline{C0}$	GET/$\overline{INIT}$	START/$\overline{STOP}$	Proses
011	00	0	0	0	Menghentikan Counter 0
	00	0	0	1	Mengaktifkan Counter 0
	00	0	1	0	Get Counter 0 Data
	00	0	1	1	*Tidak Terpakai*
	00	1	0	0	Menghentikan Counter 1
	00	1	0	1	Mengaktifkan Counter 1
	00	1	1	0	Get Counter 1 Data
	00	1	1	1	*Tidak Terpakai*
	01	X	X	X	*Tidak Terpakai*
	1X	X	X	X	*Tidak Terpakai*

Secara default, kondisi awal Counter 0 dan Counter 1 adalah tidak aktif.

3.2.4. BAUD RATE



Command untuk Baud Rate akan mengaktifkan salah satu dari beberapa setting Baud Rate.
Pemilihan setting baud rate ditentukan oleh nilai D0, D1, dan D2.

Mode	00	D2	D1	D0	Baud Rate
100	00	0	0	0	9600 bps
	00	0	0	1	19200 bps
	00	0	1	0	38400 bps
	00	0	1	1	57600 bps
	00	1	0	0	115200 bps
	01	1	X	1	*Tidak Terpakai*
	1X	X	X	X	*Tidak Terpakai*

Secara default, kondisi baud rate pada saat awal adalah:
- Baud Rate : 9600 bps

3.3. RUTIN DLL DAN TPU

SERIAL PPI memiliki library berupa file SERLIB.DLL yang dapat digunakan oleh bahasa pemrograman yang dapat mengakses file tersebut (misalnya Borland© Delphi, Borland© C++, dll). Selain itu SERIAL PPI juga memiliki unit berupa file SERPAS.TPU yang dapat digunakan oleh bahasa pemrograman Turbo Pascal. Kedua file ini dapat digunakan untuk mempermudah user dalam pemrograman.

Berikut ini adalah rutin-rutin yang digunakan dalam SERLIB.DLL dan SERPAS.TPU:

IOFlag

Fungsi : Memeriksa status komunikasi terakhir.
Tipe : Function
Input : -
Output : IOFlag Tipe : Boolean
Keterangan : -
Metode : Panggil rutin ini untuk memeriksa apakah SERIAL PPI membalas dengan kode Acknowledge (FAH) pada komunikasi terakhir.
Jika command terakhir dibalas dengan FAH oleh SERIAL PPI, maka IOFlag bernilai = True.
Jika command terakhir tidak dibalas dengan FAH oleh SERIAL PPI, IOFlag bernilai = False.

COMBaud(Rate)

Fungsi : Mengubah baud rate pada COM port komputer.
Tipe : Function
Input : Rate Tipe : Longint
Output : COMBaud Tipe : Boolean
Keterangan :

- ❖ Nilai Rate yang valid = 9600, 19200, 38400, 57600, dan 115200.
- ❖ Pemanggilan rutin ini hanya mengubah baud rate komputer, baud rate SERIAL PPI tidak berubah.

Metode : Isikan nilai Rate lalu panggil rutin ini seperti contoh.
Jika nilai Rate valid, maka baud rate akan berubah dan COMBaud bernilai = True.
Jika nilai Rate tidak valid, maka baud rate tidak akan berubah dan COMBaud bernilai = False.
Contoh : COMBaud(19200) akan mengubah baud rate komputer menjadi 19200 bps.

DeviceBaud(Rate)

Fungsi : Mengubah baud rate SERIAL PPI dan COM port komputer.
Tipe : Function
Input : Rate Tipe : Longint
Output : DeviceBaud Tipe : Boolean
Keterangan :

- ❖ Nilai Rate yang valid = 9600, 19200, 38400, 57600, dan 115200.
- ❖ Pemanggilan rutin ini masih menggunakan baud rate sebelumnya.
- ❖ Pemanggilan rutin ini akan mengubah baud rate komputer dan SERIAL PPI.

Metode : Isikan nilai Rate lalu panggil rutin ini seperti contoh.
Jika nilai Rate valid dan SERIAL PPI dapat menjawab komunikasi, maka baud rate akan berubah dan DeviceBaud bernilai = True.
Jika nilai Rate tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka baud rate tidak akan berubah dan DeviceBaud bernilai = False.
Contoh : DeviceBaud(19200) akan mengubah baud rate komputer dan SERIAL PPI menjadi 19200 bps.

CountStatus(CounterSelect, Run)

Fungsi : Mengaktifkan atau menghentikan Counter.
Tipe : Function
Input : CounterSelect Tipe : Byte
Run Tipe : Boolean
Output : CountStatus Tipe : Boolean

Keterangan :

- ❖ Nilai CounterSelect yang valid = 0 dan 1.

Metode : Pilih Counter yang akan diaktifkan dengan mengisi CounterSelect.

Jika Run diberi nilai True, maka Counter yang ditentukan akan aktif dan input Counter tersebut akan terbaca oleh SERIAL PPI.
Jika Run diberi nilai False, maka Counter yang ditentukan akan dihentikan dan input Counter tersebut tidak akan terbaca.
Jika nilai CounterSelect valid dan SERIAL PPI dapat menjawab komunikasi, maka kondisi Counter akan berubah dan CountStatus akan bernilai = True.
Jika nilai CounterSelect tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka kondisi Counter tidak akan berubah dan CountStatus akan bernilai = False.

Contoh : CountStatus(0,True) akan menjalankan Counter 0.

CountRead(CounterSelect)

Fungsi : Membaca nilai Counter.
Tipe : Function
Input : CounterSelect Tipe : Byte
Output : CountRead Tipe : Word
Keterangan :

- Nilai CounterSelect yang valid adalah 0 dan 1.

Metode : Pilih Counter yang akan diaktifkan dengan mengisi CounterSelect.
Jika nilai CounterSelect valid dan SERIAL PPI dapat menjawab komunikasi, maka nilai Counter yang dipilih akan berada pada CountRead dan register internal SERIAL PPI akan kembali ke nilai 0.
Jika nilai CounterSelect tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka CountRead akan bernilai = 0 dan register internal SERIAL PPI tidak akan kembali ke nilai 0.

Contoh : CountRead(1) akan membaca nilai Counter 1.

PortWrite(IOport, DataOut)

Fungsi : Menuliskan data ke Port.
Tipe : Function
Input : IOport Tipe : String
DataOut Tipe : Byte
Output : PortWrite Tipe : Boolean
Keterangan :

- Nilai IOport yang valid = ‘1’, ‘2’, ‘CW’ atau ‘cw’, ‘A’ atau ’a’, ‘B’ atau ‘b’, dan ‘C’ atau ‘c’.
- Penulisan ke CW(Control Word) dapat berfungsi untuk menentukan mode Port A, B, dan C atau untuk menjalankan Bit Set/Reset untuk Port C.
- Penulisan ke Port A, B, atau C dapat dilakukan jika port dikonfigurasi sebagai output.

Metode : Pilih Port yang akan diakses dengan mengisi IOport.
Nilai yang akan dikirimkan ke Port diisikan ke DataOut.
Jika nilai IOport valid dan SERIAL PPI dapat menjawab komunikasi, maka nilai DataOut akan dikirimkan ke Port dan PortWrite bernilai = True.

Jika nilai IOport tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka nilai DataOut tidak akan dikirimkan ke Port dan PortWrite bernilai = False.

Contoh : PortWrite('1',20) akan menuliskan nilai 20 ke Port 1.

PortRead(IOport)

Fungsi : Membaca data dari Port.
Tipe : Function
Input : IOport Tipe : String
Output : PortRead Tipe : Byte
Keterangan :

- ❖ Nilai IOport yang valid = '1', '2', 'A' atau 'a', 'B' atau 'b', dan 'C' atau 'c'.
- ❖ Pada pembacaan Port 1 dan 2, semua bit Port akan diberi logika = 1 sebelum Port dibaca.
- ❖ Pembacaan dari Port A, B, atau C dapat dilakukan jika port dikonfigurasi sebagai input.

Metode : Pilih Port yang akan diakses dengan mengisi IOport.
Jika nilai IOport valid dan SERIAL PPI dapat menjawab komunikasi, maka hasil pembacaan Port akan berada pada PortRead.
Jika nilai IOport tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka PortRead akan bernilai = 0.

Contoh : PortRead('2') akan membaca nilai Port 2.

BitWrite(IOport, BitSelect, SetReset)

Fungsi : Memberi logika 0 atau 1 ke bit Port 1 atau Port 2.
Tipe : Function
Input : IOport Tipe : Byte
BitSelect Tipe : Byte
SetReset Tipe : Boolean
Output : BitWrite Tipe : Boolean
Keterangan :

- ❖ Nilai IOport yang valid = 1 dan 2.
- ❖ Nilai BitSelect yang valid = 0, 1, 2, 3, 4, 5, 6, dan 7.

Metode : Pilih Port yang akan diakses dengan mengisi IOport.
Pilih Bit yang diakses dengan mengisi BitSelect.
Jika SetReset bernilai = True, maka Bit akan diberi logika = 1.
Jika SetReset bernilai = False, maka Bit akan diberi logika = 0.
Jika nilai IOport dan BitSelect valid dan SERIAL PPI dapat menjawab komunikasi, maka BitWrite akan bernilai = True.
Jika nilai IOport atau BitSelect tidak valid atau SERIAL PPI tidak dapat menjawab komunikasi, maka BitWrite akan bernilai = False.

Contoh : BitWrite(1,5,True) akan memberi logika 1 pada P1.5.

COMSetPort(PortSer)

Fungsi : Memilih COM port.
Tipe : Function
Input : PortSer Tipe : String
Output : COMSetPort Tipe : Boolean
Keterangan :

- ❖ Nilai PortSer yang valid = ‘COMn’ dimana n adalah nomor COM port yang digunakan.
- ❖ Nilai PortSer umumnya hanya berkisar antara ‘COM1’ dan ‘COM2’, namun ada komputer yang memiliki ‘COM3’ atau lebih.
- ❖ Rutin ini hanya terdapat pada SERLIB.DLL.

Metode : Gunakan rutin ini untuk memilih COM port sebelum membukanya.
Jika rutin ini tidak digunakan sebelum membuka COM port, maka COM port yang digunakan adalah COM1.
Jika nilai PortSer valid, maka COMSetPort bernilai = True.
Jika nilai PortSer tidak valid, maka COMSetPort bernilai = False.

Contoh : COMSetPort(‘COM2’) akan membuka COM2.

Penting!
Jika COM port sedang terbuka lalu diubah menggunakan rutin COMSetPort, maka COM port sebelumnya akan ditutup secara otomatis dan port yang baru akan dibuka secara otomatis.
Setting Baud Rate komputer tidak akan berubah meskipun COM port diubah dengan COMSetPort.

COMOpen(Stat)

Fungsi : Membuka atau menutup COM port.
Tipe : Function
Input : Stat Tipe : Boolean
Output : COMOpen Tipe : Boolean
Keterangan :

- ❖ Rutin ini hanya terdapat pada SERLIB.DLL.

Metode : Jika COMSetPort tidak digunakan sebelum rutin ini, maka COM port yang diakses adalah COM1.
Jika Stat bernilai = True, maka COM port akan terbuka.
Jika Stat bernilai = False, maka COM port akan ditutup.
Jika proses membuka COM port dapat dilakukan, COMOpen akan bernilai = True.
Jika proses membuka/menutup COM port tidak dapat dilakukan atau COM port sudah tertutup, COMOpen akan bernilai = False.

Contoh : COMOpen(True) akan membuka COM port.

Penting!
Sebelum COM port terbuka atau setelah COM port ditutup, komunikasi dengan SERIAL PPI tidak dimungkinkan.
Jika COM port sedang terbuka lalu dipindah menggunakan rutin COMSetPort, maka COM port sebelumnya akan ditutup secara otomatis dan port yang baru akan dibuka secara otomatis.

COMSet(PortSer)

Fungsi : Memilih COM port.
Tipe : Procedure
Input : PortSer Tipe : String
Output : -

Keterangan :
- ❖ Nilai PortSer yang valid = ‘COM1’ dan ‘COM2’.
- ❖ Rutin ini hanya terdapat pada SERPAS.TPU.

Metode : Pilih COM port dengan mengisi PortSer.
Rutin ini hanya akan melakukan pemindahan COM port yang digunakan.

Contoh : COMInit(‘COM1’) akan memindah jalur komunikasi ke COM 1.

COMInit(PortSer)

Fungsi : Memilih COM port dan melakukan inisialisasi.
Tipe : Procedure
Input : PortSer Tipe : String
Output : -
Keterangan :
- ❖ Nilai PortSer yang valid = ‘COM1’ dan ‘COM2’.
- ❖ Rutin ini hanya terdapat pada SERPAS.TPU.

Metode : Pilih COM port dengan mengisi PortSer.
Rutin ini akan melakukan inisialisasi terhadap COM port yang dipilih dan mengatur baud rate komputer menjadi 9600 bps.

Contoh : COMInit(‘COM1’) akan memilih COM 1 dengan baud rate 9600 bps.

Penting!
Setelah COMInit digunakan, baud rate komputer menjadi 9600 bps, namun baud rate SERIAL PPI tidak akan berubah.

DeviceReset

Fungsi : Melakukan reset pada SERIAL PPI.
Tipe : Procedure
Input : -
Output : -
Keterangan :-
Metode : Rutin ini akan mereset SERIAL PPI agar sistem kembali seperti saat baru pertama kali dinyalakan (default).
Contoh : Pemanggilan DeviceReset akan mengembalikan baud rate komputer dan SERIAL PPI menjadi 9600 bps, Port 1 dan Port 2 sebagai input, Port A, Port B, dan Port C tidak terprogram, serta kedua Counter tidak aktif.

3.4. CONTOH APLIKASI DAN PROGRAM

Berikut ini adalah cuplikan program dalam Borland© Delphi 5.0 untuk mengakses Port 1 dan Counter 0.

```
//Tulis ke Port 1
procedure TForm1.SendP1Click(Sender: TObject);
begin
      if not PortWrite('1',dataout) then showmessage('Error');
      if IOFlag then edit1.text:='Success' else
      edit1.text:='Error';
end;

//Baca Port 1
procedure TForm1.ReceiveP1Click(Sender: TObject);
```

```
begin
      r1.text:=inttohex(PortRead('1'),2);
      if IOFlag then edit1.text:='Success' else
      edit1.text:='Error';
end;

//Aktifkan Counter 0
procedure TForm1.StartC0Click(Sender: TObject);
begin
      if not CountStatus(0,True) then showmessage('Error');
      if IOFlag then edit1.text:='Success' else
      edit1.text:='Error';
end;

//Baca register Counter 0
procedure TForm1.GetC0Click(Sender: TObject);
begin
      g0.text:=inttohex(CountRead(0),4);
      if IOFlag then edit1.text:='Success' else
      edit1.text:='Error';
end;

//Hentikan Counter 0
procedure TForm1.StopC0Click(Sender: TObject);
begin
      if not CountStatus(0,False) then showmessage('Error');
      if IOFlag then edit1.text:='Success' else
      edit1.text:='Error';
end;
```

Berikut ini adalah cuplikan program dalam Turbo© Pascal 7.0 untuk mengakses Port 2.7 per bit, mengubah baud rate, serta melakukan reset.

```
{mengubah bit P2.7 secara toggle}
if p27 then
      begin
      if BitWrite(2,nl,false) then
            begin
            p27:=false;
            textbackground(4);
            clrscr;
            end;
      end
else
      begin
      if BitWrite(2,nl,true) then
            begin
            p27:=true;
            textbackground(2);
            clrscr;
            end;
      end;

{mengubah baud rate SERIAL PPI}
if ym=2 then DeviceBaud(19200)
{mengubah baud rate komputer}
else if ym=3 then ComBaud(19200)
{melakukan reset}
else if ym=4 then DeviceReset;
```

3.5. KERANGKA PROGRAM

Bagi user yang ingin membuat program aplikasi SERIAL PPI dengan menggunakan rutin yang sudah ada, maka file SERLIB.DLL atau SERPAS.TPU harus digunakan/dipanggil.
SERLIB.DLL merupakan library yang akan selalu digunakan untuk setiap aplikasi SERIAL PPI yang menggunakan pemrograman under Windows®.
SERPAS.TPU merupakan unit yang akan selalu digunakan untuk setiap aplikasi SERIAL PPI yang menggunakan pemrograman Pascal (under DOS).

Kerangka pemrograman SERIAL PPI menggunakan Borland© Delphi 5.0 adalah sebagai berikut :

```
unit SERIALPPI;

interface

uses
  Windows, Forms;                          { Isi sesuai kebutuhan }

type
  TForm1 = class(TForm)
      .              { Deklarasi komponen/prosedur/fungsi user }
      .
      .
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
      .                       { Deklarasi variabel program user }
      .
      .
implementation

{$R *.DFM}

      .                                    { Nama rutin SerLib.dll }
stdcall; external 'SerLib.dll';

      .            { Panggil rutin SerLib.dll sesuai kebutuhan }
      .
      .

      .                            { Prosedur/fungsi program user }
      .
      .
end.
```

Kerangka pemrograman SERIAL PPI menggunakan Turbo© Pascal 7.0 adalah sebagai berikut:

```
program SERIALPPI;

uses
  Dos, Crt, Serpas;                        { Isi sesuai kebutuhan }
```

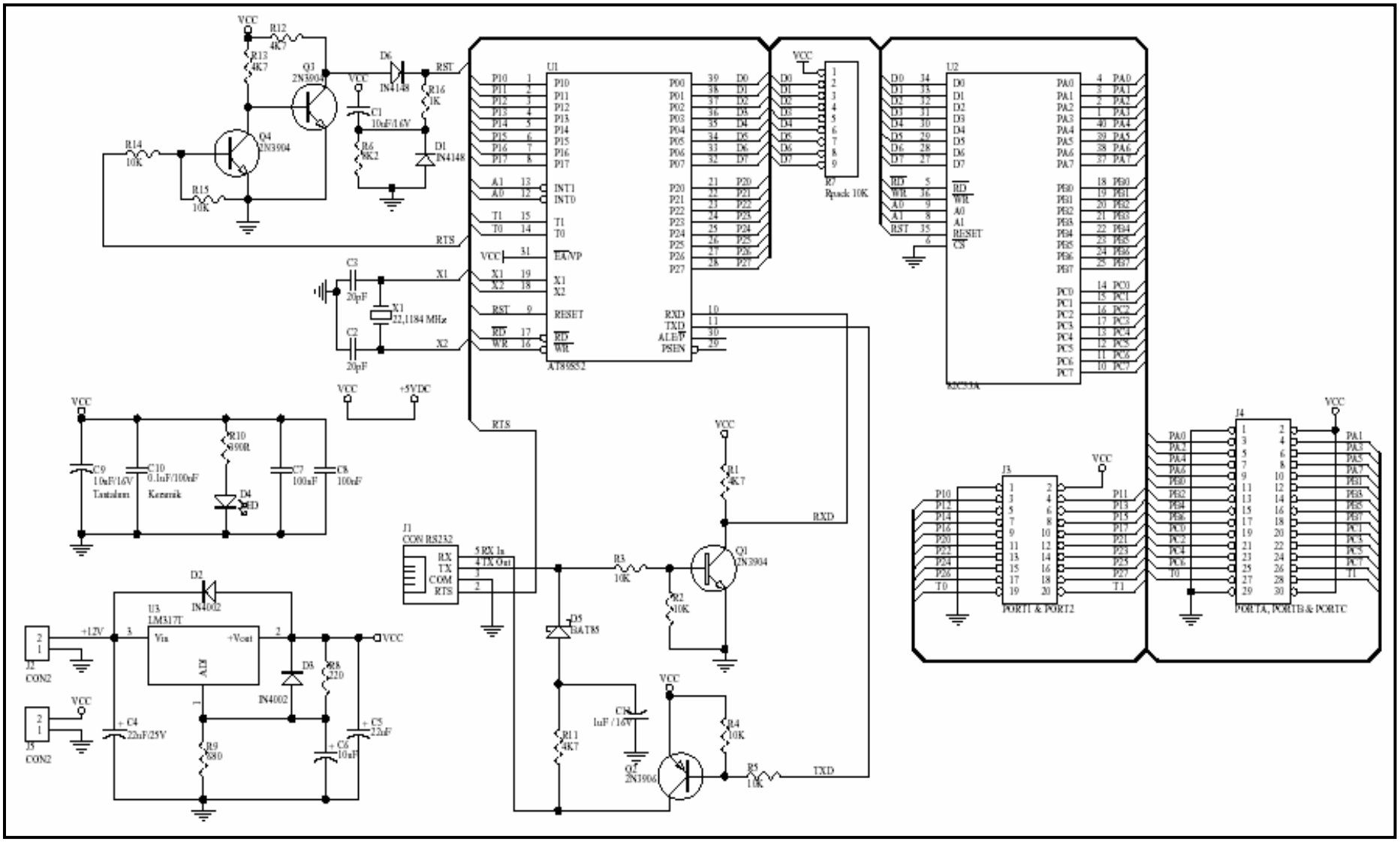
```
const
      .                    { Deklarasi konstanta program user }
      .
      .

type
      .              { Deklarasi komponen/prosedur/fungsi user }
      .
      .

var
      .                     { Deklarasi variabel program user }
      .
      .

      .                        { Prosedur/fungsi program user }
      .
      .
      .            { Panggil rutin SerPas.tpu sesuai kebutuhan }
      .
      .

begin
      .                                    { Program utama user }
      .
      .
end.
```

♦ *Terima Kasih atas kepercayaan Anda menggunakan produk kami, bila ada kesulitan, pertanyaan atau saran mengenai produk ini silahkan menghubungi technical support kami :*

support@innovativeelectronics.com

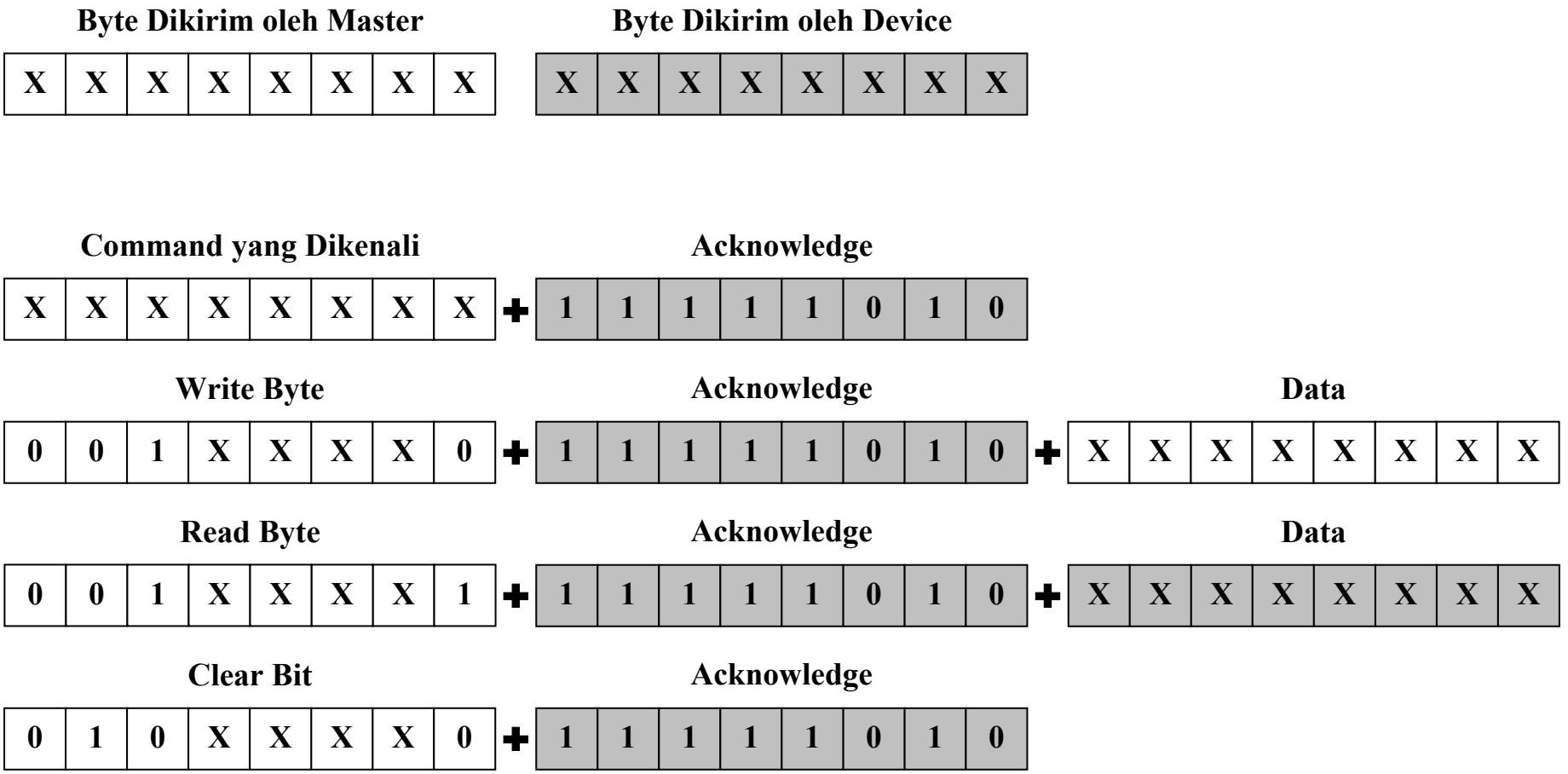
LAMPIRAN A

SKEMA SERIAL PPI



LAMPIRAN B

PROTOKOL SERIAL PPI



Set Bit

0	1	0	X	X	X	X	1

Acknowledge

1	1	1	1	1	0	1	0

Start/Stop Counter

0	1	1	0	0	X	0	X

Acknowledge

1	1	1	1	1	0	1	0

Get Counter Data

0	1	1	0	0	X	1	X

Acknowledge

1	1	1	1	1	0	1	0

Lower Data

X	X	X	X	X	X	X	X

Upper Data

X	X	X	X	X	X	X	X

Baud Rate

1	0	0	0	0	X	X	X

Acknowledge

1	1	1	1	1	0	1	0

Baud rate berubah setelah Acknowledge

Command yang Tidak Dikenali

X	X	X	X	X	X	X	X

Not Acknowledge

1	1	1	1	1	1	0	0