Lampiran A

```
unit u_cafe;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, jpeg, DB, DBTables, Grids, DBGrids,
  AfDataDispatcher, AfComPort, AfPortControls;

type
  TForm1 = class(TForm)
    Image1: TImage;
    Tabledatabase: TTable;
    DataDatabase1: TDataSource;
    TableSlave1: TTable;
    DataSlave1: TDataSource;
    LabelMenu1: TLabel;
    LabelPorsi1: TLabel;
    DBGrid1: TDBGrid;
    ButtonResetSlave1: TButton;
    LabelStatusSlave1: TLabel;
    Label1: TLabel;
    Image2: TImage;
    DBGrid2: TDBGrid;
    LabelMenu2: TLabel;
    LabelStatusSlave2: TLabel;
    ButtonResetSlave2: TButton;
    LabelPorsi2: TLabel;
    DataSlave2: TDataSource;
    TableSlave2: TTable;
    Label2: TLabel;
    Label3: TLabel;
    AfComPort1: TAfComPort;
    AfDataDispatcher1: TAfDataDispatcher;
    Edit1: TEdit;
    AfPortRadioGroup1: TAfPortRadioGroup;
    Button1: TButton;
    Button2: TButton;
    procedure FormShow(Sender: TObject);
    procedure ButtonResetSlave1Click(Sender: TObject);
    procedure AfDataDispatcher1DataReceived(Sender: TObject);
    procedure ButtonResetSlave2Click(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation

{$R *.dfm}
var
recconx1 : integer = 0;
total1 : integer = 0;
subtotal1 : integer = 0;
harga1 : integer;

recconx2 : integer = 0;
total2 : integer = 0;
subtotal2 : integer = 0;
harga2 : integer;

comport : integer = 0 ;
SC : boolean = false;
data : variant;
lokasi1 : string;
indexno1 : integer = 0;
porsi1 : integer = 0 ;
reccon1 : integer;
reset1 : integer;
lokasi2 : string;
indexno2 : integer = 0;
porsi2 : integer = 0 ;
reccon2 : integer;
reset2 : integer;

procedure TForm1.FormShow(Sender: TObject);
begin
deletefile('.\database cafe\slave1.db');
with form1.TableSlave1 do
  begin
  Active := False;
  DatabaseName := '.\database cafe\';
  TableType := ttParadox;
  TableName := 'slave1';
  if not form1.TableSlave1.Exists
  then
```

```
begin
with FieldDefs do
  begin
    Clear;
    with AddFieldDef do begin
    Name := 'No';
    DataType := ftinteger;
    Required := True;
    end;
    with AddFieldDef do begin
    Name := 'Nama Menu';
    DataType := ftstring;
    size := 40;
    end;
    with AddFieldDef do begin
    Name := 'Porsi';
    DataType := ftInteger;
    end;
    with AddFieldDef do begin
    Name := 'jumlah';
    DataType := ftInteger;
    end;
  end;
with IndexDefs do
  begin
    Clear;
    with AddIndexDef do begin
    Name := '';
    Fields := 'No';
    Options := [ixPrimary];
    end;
    with AddIndexDef do begin
    Name := 'Fld2Indx';
    Fields := 'Nama Menu';
    Options := [ixCaseInsensitive];
    end;
    with AddIndexDef do begin
    Name := 'Fld3Indx';
    Fields := 'Porsi';
    Options := [ixCaseInsensitive];
    end;
    with AddIndexDef do begin
    Name := 'Fld4Indx';
    Fields := 'jumlah';
    Options := [ixCaseInsensitive];
    end;
  end;
  CreateTable;
```

```
   deletefile('.\database cafe\slave1.PX');
   deletefile('.\database cafe\slave1.VAL');
   deletefile('.\database cafe\slave1.XG0');
   deletefile('.\database cafe\slave1.YG0');
   deletefile('.\database cafe\slave1.XG1');
   deletefile('.\database cafe\slave1.YG1');
   deletefile('.\database cafe\slave1.XG2');
   deletefile('.\database cafe\slave1.YG2');
   deletefile('.\database cafe\slave1.XG3');
   deletefile('.\database cafe\slave1.YG3');

  end;
 end;
form1.TableSlave1.TableName := 'slave1.db';
form1.TableSlave1.Active := true;

deletefile('.\database cafe\slave2.db');
with form1.TableSlave2 do
 begin
 Active := False;
 DatabaseName := '.\database cafe\';
 TableType := ttParadox;
 TableName := 'slave2';
 if not form1.TableSlave2.Exists
 then
  begin
  with FieldDefs do
   begin
    Clear;
    with AddFieldDef do begin
    Name := 'No';
    DataType := ftinteger;
    Required := True;
    end;
    with AddFieldDef do begin
    Name := 'Nama Menu';
    DataType := ftstring;
    size := 40;
    end;
    with AddFieldDef do begin
    Name := 'Porsi';
    DataType := ftInteger;
    end;
    with AddFieldDef do begin
    Name := 'Jumlah';
    DataType := ftInteger;
    end;
   end;
  with IndexDefs do
   begin
    Clear;
    with AddIndexDef do begin
    Name := '';
    Fields := 'No';
    Options := [ixPrimary];
    end;
    with AddIndexDef do begin
    Name := 'Fld2Indx';
    Fields := 'Nama Menu';
    Options := [ixCaseInsensitive];
    end;
    with AddIndexDef do begin
    Name := 'Fld3Indx';
    Fields := 'Porsi';
    Options := [ixCaseInsensitive];
    end;
    with AddIndexDef do begin
    Name := 'Fld4Indx';
    Fields := 'Jumlah';
    Options := [ixCaseInsensitive];
    end;

   end;
   CreateTable;
   deletefile('.\database cafe\slave2.PX');
   deletefile('.\database cafe\slave2.VAL');
   deletefile('.\database cafe\slave2.XG0');
   deletefile('.\database cafe\slave2.YG0');
   deletefile('.\database cafe\slave2.XG1');
   deletefile('.\database cafe\slave2.YG1');
   deletefile('.\database cafe\slave2.XG2');
   deletefile('.\database cafe\slave2.YG2');
   deletefile('.\database cafe\slave2.XG3');
   deletefile('.\database cafe\slave2.YG3');
  end;
 end;
form1.TableSlave2.TableName := 'slave2.db';
form1.TableSlave2.Active := true;

end;

procedure TForm1.ButtonResetSlave1Click(Sender: TObject);
begin
reset1 := 255;
form1.AfDataDispatcher1.WriteData(reset1,1);
```

```
indexno1 := 0 ;
form1.Image1.Picture.LoadFromFile ('.\edit\back\welcome.jpg');
form1.LabelMenu1.Caption :=  'Nama Menu';
form1.LabelPorsi1.Caption :=  '0';
form1.LabelStatusSlave1.Caption := 'Status : idle';
form1.TableSlave1.Active := false;
deletefile('.\database cafe\slave1.db');
with form1.TableSlave1 do
  begin
  Active := False;
  DatabaseName := '.\database cafe\';
  TableType := ttParadox;
  TableName := 'slave1';
  if not form1.TableSlave1.Exists
  then
    begin
    with FieldDefs do
      begin
        Clear;
        with AddFieldDef do begin
        Name := 'No';
        DataType := ftinteger;
        Required := True;
        end;
        with AddFieldDef do begin
        Name := 'Nama Menu';
        DataType := ftstring;
        size := 40;
        end;
        with AddFieldDef do begin
        Name := 'Porsi';
        DataType := ftInteger;
        end;
        with AddFieldDef do begin
        Name := 'Jumlah';
        DataType := ftInteger;
        end;
      end;
    with IndexDefs do
      begin
        Clear;
        with AddIndexDef do begin
        Name := '';
        Fields := 'No';
        Options := [ixPrimary];
        end;
        with AddIndexDef do begin
        Name := 'Fld2Indx';
        Fields := 'Nama Menu';
        Options := [ixCaseInsensitive];
        end;
        with AddIndexDef do begin
        Name := 'Fld3Indx';
        Fields := 'Porsi';
        Options := [ixCaseInsensitive];
        end;
        with AddIndexDef do begin
        Name := 'Fld4Indx';
        Fields := 'Jumlah';
        Options := [ixCaseInsensitive];
        end;
      end;
      CreateTable;
      deletefile('.\database cafe\slave1.PX');
      deletefile('.\database cafe\slave1.VAL');
      deletefile('.\database cafe\slave1.XG0');
      deletefile('.\database cafe\slave1.YG0');
      deletefile('.\database cafe\slave1.XG1');
      deletefile('.\database cafe\slave1.YG1');
      deletefile('.\database cafe\slave1.XG2');
      deletefile('.\database cafe\slave1.YG2');
      deletefile('.\database cafe\slave1.XG3');
      deletefile('.\database cafe\slave1.YG3');
    end;
  end;
form1.TableSlave1.TableName := 'slave1.db';
form1.TableSlave1.Active := true;
form1.ButtonResetSlave1.Enabled := false;
recconx1 := 0;
total1 := 0;
subtotal1 := 0;
end;

procedure TForm1.AfDataDispatcher1DataReceived(Sender: TObject);
begin
data := form1.AfDataDispatcher1.ReadString;
form1.Edit1.Text := data;

if form1.Edit1.GetTextLen = 2
then
begin
data := strtoint(form1.Edit1.Text);

//user 1
if data = 11
then
```

```
  begin
  inc(indexno1);
  if indexno1 = 6
  then
    begin
    indexno1 := 1;
    end;
  lokasi1 := form1.TableDatabase.Lookup('no',indexno1,'location');
  form1.LabelMenu1.Caption := form1.Tabledatabase.Lookup('no',indexno1,'nama');
  form1.Image1.Picture.LoadFromFile(lokasi1);
  end;

if data = 12
then
  begin
  dec(indexno1);
  if ((indexno1 = 0) or (indexno1 = -1))
  then
    begin
    indexno1 := 5;
    end;
  lokasi1 := form1.TableDatabase.Lookup('no',indexno1,'location');
  form1.LabelMenu1.Caption := form1.Tabledatabase.Lookup('no',indexno1,'nama');
  form1.Image1.Picture.LoadFromFile(lokasi1);
  end;


if data = 13
then
  begin
  inc(porsi1);
  if indexno1 = 0
  then
    begin
    form1.LabelPorsi1.Caption := '';
    indexno1 := 0;
    end
  else
    begin
    form1.LabelPorsi1.Caption := inttostr(porsi1);
    end;
  end;

if data = 14
then
  begin
  dec(porsi1);
  if indexno1 = 0
  then
    begin
    form1.LabelPorsi1.Caption := '';
    indexno1 := 0;
    end
  else
    begin
    if porsi1 < 0 then
      begin
      porsi1 := 0 ;
      end;
    form1.LabelPorsi1.Caption := inttostr(porsi1);
    end;
  end;

if data = 15
then
  begin
  if indexno1 = 0
  then
    begin
    form1.LabelMenu1.Caption := 'Push "Next" 0>';
    end
  else
    begin
    if ((form1.LabelPorsi1.Caption = '') or (form1.LabelPorsi1.Caption = '0'))
    then
      begin
      form1.LabelMenu1.Caption := 'Porsi Masih 0';
      end
    else
      begin
      reccon1 := form1.TableSlave1.RecordCount;
      form1.TableSlave1.Append;
      form1.TableSlave1.FieldByName('no').AsInteger := reccon1 + 1;
      form1.TableSlave1.FieldByName('nama menu').AsString := form1.LabelMenu1.Caption;
      form1.TableSlave1.FieldByName('porsi').AsInteger :=
strtoint(form1.LabelPorsi1.Caption);
      harga1 := form1.Tabledatabase.Lookup('No',indexno1,'Harga');
      subtotal1 := harga1 * porsi1;
      form1.TableSlave1.FieldByName('Jumlah').AsInteger := subtotal1;

      form1.TableSlave1.Post;
      form1.LabelPorsi1.Caption := '0';
      indexno1 := 0;
      porsi1 := 0;
      end;
  end;
```

```
end;

if data = 16
then
  begin
  form1.ButtonResetSlave1.Enabled := true;
  form1.LabelStatusSlave1.Caption := 'Status : Make Menu';
  recconx1 := form1.TableSlave1.RecordCount;
  repeat
  subtotal1 := form1.TableSlave1.Lookup('no',recconx1,'Jumlah');
  total1 := total1 + subtotal1;
  dec( recconx1 );
  until (recconx1=0);
  form1.LabelMenu1.Caption := 'Total Pembelian';
  form1.LabelPorsi1.Caption := inttostr(total1);
  end;


//user 2
if (data=21)
then
  begin
  inc(indexno2);
  if indexno2 = 6
  then
    begin
    indexno2 := 1;
    end;
  lokasi2 := form1.TableDatabase.Lookup('no',indexno2,'location');
  form1.LabelMenu2.Caption := form1.Tabledatabase.Lookup('no',indexno2,'nama');
  form1.Image2.Picture.LoadFromFile(lokasi2);
  end;

if data = 22
then
  begin
  dec(indexno2);
  if ((indexno2 = 0) or (indexno2 = -1))
  then
    begin
    indexno2 := 5;
    end;
  lokasi2 := form1.TableDatabase.Lookup('no',indexno2,'location');
  form1.LabelMenu2.Caption := form1.Tabledatabase.Lookup('no',indexno2,'nama');
  form1.Image2.Picture.LoadFromFile(lokasi2);
  end;
```

```
if data = 23
then
  begin
  inc(porsi2);
  if indexno2 = 0
  then
    begin
    form1.LabelPorsi2.Caption := '';
    indexno2 := 0;
    end
  else
    begin
    form1.LabelPorsi2.Caption := inttostr(porsi2);
    end;
  end;

if data = 24
then
  begin
  dec(porsi2);
  if indexno2 = 0
  then
    begin
    form1.LabelPorsi2.Caption := '';
    indexno2 := 0;
    end
  else
    begin
    if porsi2 < 0 then
      begin
      porsi2 := 0 ;
      end;
    form1.LabelPorsi2.Caption := inttostr(porsi2);
    end;
  end;

if data = 25
then
  begin
  if indexno2 = 0
  then
    begin
    form1.LabelMenu2.Caption := 'Push ''Next'' 0>';
    end
  else
    begin
    if ((form1.LabelPorsi2.Caption = '') or (form1.LabelPorsi2.Caption = '0'))
    then
```

```
      begin
      form1.LabelMenu2.Caption := 'Porsi Masih 0';
      end
    else
      begin
      reccon2 := form1.TableSlave2.RecordCount;
      form1.TableSlave2.Append;
      form1.TableSlave2.FieldByName('no').AsInteger := reccon2 + 1;
      form1.TableSlave2.FieldByName('nama menu').AsString := form1.LabelMenu2.Caption;
      form1.TableSlave2.FieldByName('porsi').AsInteger :=
strtoint(form1.LabelPorsi2.Caption);
      harga2 := form1.Tabledatabase.Lookup('No',indexno2,'Harga');
      subtotal2 := harga2 * porsi2;
      form1.TableSlave2.FieldByName('Jumlah').AsInteger := subtotal2;

      form1.TableSlave2.Post;
      form1.LabelPorsi2.Caption := '0';
      indexno2 := 0;
      porsi2 := 0;
      end;
  end;
end;

if data = 26
then
  begin
  form1.ButtonResetSlave2.Enabled := true;
  form1.LabelStatusSlave2.Caption := 'Status : Make Menu';
  recconx2 := form1.TableSlave2.RecordCount;
  repeat
  subtotal2 := form1.TableSlave2.Lookup('no',recconx2,'Jumlah');
  total2 := total2 + subtotal2;
  dec( recconx2 );
  until (recconx2=0);
  form1.LabelMenu2.Caption := 'Total Pembelian';
  form1.LabelPorsi2.Caption := inttostr(total2);
  end;

end;
end;


procedure TForm1.ButtonResetSlave2Click(Sender: TObject);
begin
reset2 := 254;
form1.AfDataDispatcher1.WriteData(reset2,1);
indexno2 := 0 ;
form1.Image2.Picture.LoadFromFile ('.\edit\back\welcome.jpg');
```

```
form1.LabelMenu2.Caption :=  'Nama Menu';
form1.LabelPorsi2.Caption :=  '0';
form1.LabelStatusSlave2.Caption := 'Status : idle';
form1.TableSlave2.Active := false;
deletefile('.\database cafe\slave2.db');

with form1.TableSlave2 do
  begin
  Active := False;
  DatabaseName := '.\database cafe\';
  TableType := ttParadox;
  TableName := 'slave2';
  if not form1.TableSlave2.Exists
  then
    begin
    with FieldDefs do
      begin
        Clear;
        with AddFieldDef do begin
        Name := 'No';
        DataType := ftinteger;
        Required := True;
        end;
        with AddFieldDef do begin
        Name := 'Nama Menu';
        DataType := ftstring;
        size := 40;
        end;
        with AddFieldDef do begin
        Name := 'Porsi';
        DataType := ftInteger;
        end;
        with AddFieldDef do begin
        Name := 'Jumlah';
        DataType := ftInteger;
        end;

      end;
    with IndexDefs do
      begin
        Clear;
        with AddIndexDef do begin
        Name := '';
        Fields := 'No';
        Options := [ixPrimary];
        end;
        with AddIndexDef do begin
        Name := 'Fld2Indx';
        Fields := 'Nama Menu';
        Options := [ixCaseInsensitive];
        end;
        with AddIndexDef do begin
        Name := 'Fld3Indx';
        Fields := 'Porsi';
        Options := [ixCaseInsensitive];
        end;
        with AddIndexDef do begin
        Name := 'Fld4Indx';
        Fields := 'Jumlah';
        Options := [ixCaseInsensitive];
        end;
      end;
      CreateTable;
      //jgn lupa delete file na !!
      deletefile('.\database cafe\slave2.PX');
      deletefile('.\database cafe\slave2.VAL');
      deletefile('.\database cafe\slave2.XG0');
      deletefile('.\database cafe\slave2.YG0');
      deletefile('.\database cafe\slave2.XG1');
      deletefile('.\database cafe\slave2.YG1');
      deletefile('.\database cafe\slave2.XG2');
      deletefile('.\database cafe\slave2.YG2');
      deletefile('.\database cafe\slave2.XG3');
      deletefile('.\database cafe\slave2.YG3');

    end;
  end;
form1.TableSlave2.TableName := 'slave2.db';
form1.TableSlave2.Active := true;
form1.ButtonResetSlave2.Enabled := false;
recconx2 := 0;
total2 := 0;
subtotal2 := 0;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
form1.AfPortRadioGroup1.Visible := true;
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
form1.AfPortRadioGroup1.Visible := false;
end;

end.
```

Lampiran B

```
#include <mega16.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>
#include <delay.h>

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE<256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif

// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index]=data;
   if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      };
   };
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index];
if (++rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART Transmitter buffer
#define TX_BUFFER_SIZE 8
char tx_buffer[TX_BUFFER_SIZE];

#if TX_BUFFER_SIZE<256
unsigned char tx_wr_index,tx_rd_index,tx_counter;
#else
unsigned int tx_wr_index,tx_rd_index,tx_counter;
#endif

// USART Transmitter interrupt service routine
interrupt [USART_TXC] void usart_tx_isr(void)
{
if (tx_counter)
   {
   --tx_counter;
   UDR=tx_buffer[tx_rd_index];
   if (++tx_rd_index == TX_BUFFER_SIZE) tx_rd_index=0;
   };
}
```

```
#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter == TX_BUFFER_SIZE);
#asm("cli")
if (tx_counter || ((UCSRA & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer[tx_wr_index]=c;
   if (++tx_wr_index == TX_BUFFER_SIZE) tx_wr_index=0;
   ++tx_counter;
   }
else
   UDR=c;
#asm("sei")
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here

void main(void)
{
int index_menu = 0;
int porsi = 0;
char tampil[4];
char data;
unsigned long int total = 0;
unsigned long int subtotal = 0;
unsigned long int harga = 0;
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=P State6=P State5=P State4=P State3=P State2=P State1=P State0=P
PORTA=0xFF;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=Out
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=0
PORTB=0x00;
```

```
DDRB=0x01;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud rate: 57600
UCSRA=0x00;
UCSRB=0xD8;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x0B;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// LCD module initialization
lcd_init(16);

// Global enable interrupts
#asm("sei")


lcd_gotoxy(0,0);
lcd_putsf("Welcome to");
lcd_gotoxy(0,1);
lcd_putsf("David's Cafe");
```

```
delay_ms(2000);



while (1)
      {
lcd_clear();
lcd_gotoxy(0,0);
lcd_putsf("Push Next >>");
index_menu = 0;
porsi = 0;

mulai: {
        PORTB = 0X00;
        if ((PINA.0 == 0) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 1))
           { delay_ms(200);
             goto next; };
        if ((PINA.0 == 1) && (PINA.1 == 0) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 1))
           { delay_ms(200);
             goto prev; };
        if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 0) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 1))
           { delay_ms(200);
             goto plus; };
        if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 0) &&(PINA.4
== 1) && (PINA.5 == 1))
           { delay_ms(200);
             goto minus; };
        if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 0) && (PINA.5 == 1))
           { delay_ms(200);
             goto enter; };
        if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 0))
           { delay_ms(200);
             goto endx; };
       goto mulai ;
       }

next:  { PORTB = 0XFF;
         printf("11");
         index_menu = index_menu + 1;
         if (index_menu == 6) { index_menu = 1;}
         goto tulis_lcd;}
prev:  { PORTB = 0XFF;
         printf("12");
         index_menu = index_menu - 1;
         if (index_menu == 0) { index_menu = 5;}
         goto tulis_lcd;}
plus:  { PORTB = 0XFF;
         printf("13");
         porsi = porsi + 1;
         goto tulis_lcd;}
minus: { PORTB = 0XFF;
         printf("14");
         porsi = porsi - 1;
         if (porsi < 0) { porsi = 0;}
         goto tulis_lcd;}
enter: { if ( porsi == 0 )
           { lcd_gotoxy(0,0);
             lcd_clear();
                  lcd_putsf("Porsi masih 0");
                  delay_ms(2000);
                  goto tulis_lcd;
           }
         if ( porsi != 0 )
           { PORTB = 0XFF;
             printf("15");
             subtotal = harga * porsi;
             total = total + subtotal;
             porsi = 0;
             index_menu = 0;
             goto tulis_lcd;
           }
       }
endx:  { PORTB = 0XFF;
         printf("16");
         goto tulis_lcdx;}

tulis_lcd: { if (index_menu == 0)
                { lcd_clear();
                  lcd_gotoxy(0,0);
                  lcd_putsf("Next Menu Order.");
                  lcd_gotoxy(0,1);
                  lcd_putsf("End To Make Menu");
                  goto jump;
                }
             if (index_menu == 1)
                { lcd_gotoxy(0,0);
                  lcd_clear();
                  lcd_putsf("Aromatic Duck");
                  lcd_gotoxy(0,1);
                  lcd_putsf("20000");
                  harga = 20000;
```

```
               }
            if (index_menu == 2)
               { lcd_gotoxy(0,0);
                 lcd_clear();
                 lcd_putsf("Aromatic Pork");
                 lcd_gotoxy(0,1);
                 lcd_putsf("30000");
                 harga = 30000;
               }
            if (index_menu == 3)
               { lcd_gotoxy(0,0);
                 lcd_clear();
                 lcd_putsf("Bento Noodle");
                 lcd_gotoxy(0,1);
                 lcd_putsf("12000");
                 harga = 12000;
               }
            if (index_menu == 4)
               { lcd_gotoxy(0,0);
                 lcd_clear();
                 lcd_putsf("Beef Satay");
                 lcd_gotoxy(0,1);
                 lcd_putsf("17500");
                 harga = 17500;
               }
            if (index_menu == 5)
               { lcd_gotoxy(0,0);
                 lcd_clear();
                 lcd_putsf("Chicken Satay");
                 lcd_gotoxy(0,1);
                 lcd_putsf("15000");
                 harga = 15000;
               }

            if ( porsi <10 )
               { lcd_gotoxy(14,1);
                 sprintf(tampil," %d",porsi);
                 lcd_puts(tampil);
               }
            if ( porsi >=10 )
               { lcd_gotoxy(14,1);
                 sprintf(tampil,"%d",porsi);
                 lcd_puts(tampil);
               }
           goto jump;
          }

jump:     { PORTB = 0X00;
```

```
        goto mulai;
       }

tulis_lcdx: {   lcd_clear();
          lcd_gotoxy(0,0);
          lcd_putsf("Please Wait...");

/* ga tau nama persen untuk variabel unsigned long int

          lcd_gotoxy(0,1);
          lcd_putsf("Total");

          lcd_gotoxy(6,1);
          sprintf(tampil,"%X",total);
          lcd_puts(tampil);
*/
          PORTB = 0X00;
          goto waitreset;
       }

waitreset: { data = getchar();
        if( ((int)data == 254))
        { lcd_clear();
         lcd_gotoxy(0,0);
         lcd_putsf("Push Next >>");
         goto mulai;
        }
        if( ((int)data != 254))
        { lcd_gotoxy(0,0);
         lcd_putsf("Please Wait...");
         goto waitreset;
        }
       }


};
}
```

Lampiran C

```
#include <mega16.h>

// Alphanumeric LCD Module functions
#asm
   .equ __lcd_port=0x15 ;PORTC
#endasm
#include <lcd.h>
#include <delay.h>

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY (1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE<256
unsigned char rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int rx_wr_index,rx_rd_index,rx_counter;
#endif

// This flag is set on USART Receiver buffer overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSRA;
data=UDR;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer[rx_wr_index]=data;
   if (++rx_wr_index == RX_BUFFER_SIZE) rx_wr_index=0;
   if (++rx_counter == RX_BUFFER_SIZE)
      {
      rx_counter=0;
      rx_buffer_overflow=1;
      };
   };
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index];
if (++rx_rd_index == RX_BUFFER_SIZE) rx_rd_index=0;
#asm("cli")
--rx_counter;
#asm("sei")
return data;
}
#pragma used-
#endif

// USART Transmitter buffer
#define TX_BUFFER_SIZE 8
char tx_buffer[TX_BUFFER_SIZE];

#if TX_BUFFER_SIZE<256
unsigned char tx_wr_index,tx_rd_index,tx_counter;
#else
unsigned int tx_wr_index,tx_rd_index,tx_counter;
#endif

// USART Transmitter interrupt service routine
interrupt [USART_TXC] void usart_tx_isr(void)
{
if (tx_counter)
   {
   --tx_counter;
   UDR=tx_buffer[tx_rd_index];
   if (++tx_rd_index == TX_BUFFER_SIZE) tx_rd_index=0;
   };
}
```

```
#ifndef _DEBUG_TERMINAL_IO_
// Write a character to the USART Transmitter buffer
#define _ALTERNATE_PUTCHAR_
#pragma used+
void putchar(char c)
{
while (tx_counter == TX_BUFFER_SIZE);
#asm("cli")
if (tx_counter || ((UCSRA & DATA_REGISTER_EMPTY)==0))
   {
   tx_buffer[tx_wr_index]=c;
   if (++tx_wr_index == TX_BUFFER_SIZE) tx_wr_index=0;
   ++tx_counter;
   }
else
   UDR=c;
#asm("sei")
}
#pragma used-
#endif

// Standard Input/Output functions
#include <stdio.h>

// Declare your global variables here

void main(void)
{
int index_menu = 0;
int porsi = 0;
char tampil[4];
char data;
unsigned long int total = 0;
unsigned long int subtotal = 0;
unsigned long int harga = 0;
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=P State6=P State5=P State4=P State3=P State2=P State1=P State0=P
PORTA=0xFF;
DDRA=0x00;

// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=Out
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=0
PORTB=0x00;
```

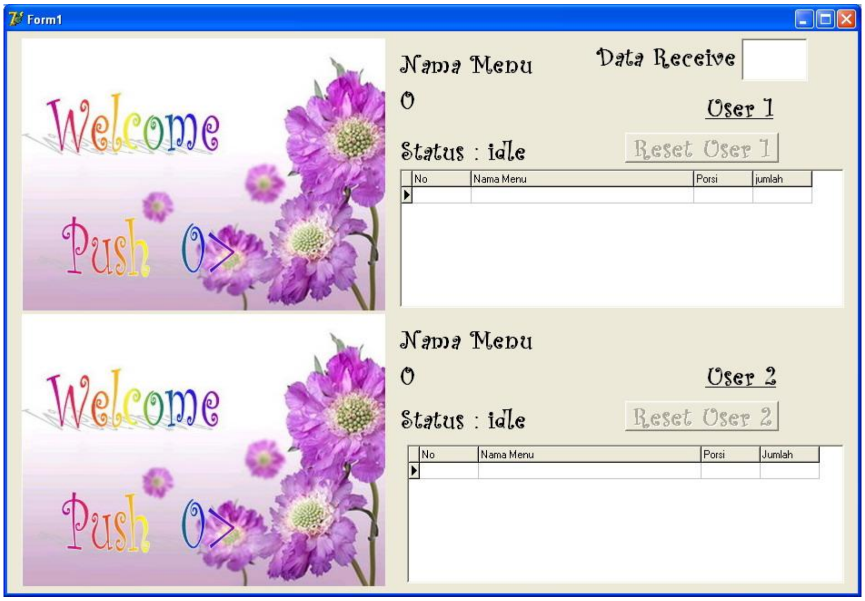
```
DDRB=0x01;

// Port C initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTC=0x00;
DDRC=0x00;

// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// USART initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART Receiver: On
// USART Transmitter: On
// USART Mode: Asynchronous
// USART Baud rate: 57600
UCSRA=0x00;
UCSRB=0xD8;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x0B;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

// LCD module initialization
lcd_init(16);

// Global enable interrupts
#asm("sei")


lcd_gotoxy(0,0);
lcd_putsf("Welcome to");
lcd_gotoxy(0,1);
lcd_putsf("David's Cafe");
```

```
delay_ms(2000);



while (1)
    {
lcd_clear();
lcd_gotoxy(0,0);
lcd_putsf("Push Next >>");
index_menu = 0;
porsi = 0;

mulai: {
     PORTB = 0X00;
     if ((PINA.0 == 0) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 1))
       { delay_ms(200);
        goto next; };
     if ((PINA.0 == 1) && (PINA.1 == 0) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 1))
       { delay_ms(200);
        goto prev; };
     if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 0) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 1))
       { delay_ms(200);
        goto plus; };
     if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 0) &&(PINA.4
== 1) && (PINA.5 == 1))
       { delay_ms(200);
        goto minus; };
     if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 0) && (PINA.5 == 1))
       { delay_ms(200);
        goto enter; };
     if ((PINA.0 == 1) && (PINA.1 == 1) &&(PINA.2 == 1) &&(PINA.3 == 1) &&(PINA.4
== 1) && (PINA.5 == 0))
       { delay_ms(200);
        goto endx; };
    goto mulai ;
    }

next:  { PORTB = 0XFF;
      printf("21");
      index_menu = index_menu + 1;
      if (index_menu == 6) { index_menu = 1;}
      goto tulis_lcd;}
prev:  { PORTB = 0XFF;
      printf("22");
```

```
        index_menu = index_menu - 1;
        if (index_menu == 0) { index_menu = 5;}
        goto tulis_lcd;}
plus:  { PORTB = 0XFF;
        printf("23");
        porsi = porsi + 1;
        goto tulis_lcd;}
minus: { PORTB = 0XFF;
        printf("24");
        porsi = porsi - 1;
        if (porsi < 0) { porsi = 0;}
        goto tulis_lcd;}
enter: { if ( porsi == 0 )
          { lcd_gotoxy(0,0);
            lcd_clear();
                lcd_putsf("Porsi masih 0");
                delay_ms(2000);
                goto tulis_lcd;
          }
        if ( porsi != 0 )
          { PORTB = 0XFF;
            printf("25");
            subtotal = harga * porsi;
            total = total + subtotal;
            porsi = 0;
            index_menu = 0;
            goto tulis_lcd;
          }
      }
endx:  { PORTB = 0XFF;
        printf("26");
        goto tulis_lcdx;}

tulis_lcd: { if (index_menu == 0)
              { lcd_clear();
                lcd_gotoxy(0,0);
                lcd_putsf("Next Menu Order.");
                lcd_gotoxy(0,1);
                lcd_putsf("End To Make Menu");
                goto jump;
              }
            if (index_menu == 1)
              { lcd_gotoxy(0,0);
                lcd_clear();
                lcd_putsf("Aromatic Duck");
                lcd_gotoxy(0,1);
                lcd_putsf("20000");
                harga = 20000;
          }
        if (index_menu == 2)
          { lcd_gotoxy(0,0);
           lcd_clear();
           lcd_putsf("Aromatic Pork");
           lcd_gotoxy(0,1);
           lcd_putsf("30000");
           harga = 30000;
          }
        if (index_menu == 3)
          { lcd_gotoxy(0,0);
           lcd_clear();
           lcd_putsf("Bento Noodle");
           lcd_gotoxy(0,1);
           lcd_putsf("12000");
           harga = 12000;
          }
        if (index_menu == 4)
          { lcd_gotoxy(0,0);
           lcd_clear();
           lcd_putsf("Beef Satay");
           lcd_gotoxy(0,1);
           lcd_putsf("17500");
           harga = 17500;
          }
        if (index_menu == 5)
          { lcd_gotoxy(0,0);
           lcd_clear();
           lcd_putsf("Chicken Satay");
           lcd_gotoxy(0,1);
           lcd_putsf("15000");
           harga = 15000;
          }

        if ( porsi <10 )
          { lcd_gotoxy(14,1);
           sprintf(tampil," %d",porsi);
           lcd_puts(tampil);
          }
        if ( porsi >=10 )
          { lcd_gotoxy(14,1);
           sprintf(tampil,"%d",porsi);
           lcd_puts(tampil);
          }
       goto jump;
      }

jump:    { PORTB = 0X00;
```

```
            goto mulai;
           }

tulis_lcdx: {   lcd_clear();
               lcd_gotoxy(0,0);
               lcd_putsf("Please Wait...");

/* ga tau nama persen untuk variabel unsigned long int

               lcd_gotoxy(0,1);
               lcd_putsf("Total");

               lcd_gotoxy(6,1);
               sprintf(tampil,"%X",total);
               lcd_puts(tampil);
*/
               PORTB = 0X00;
               goto waitreset;
            }

waitreset: { data = getchar();
            if( ((int)data == 255))
            { lcd_clear();
              lcd_gotoxy(0,0);
              lcd_putsf("Push Next >>");
              goto mulai;
            }
            if( ((int)data != 255))
            { lcd_gotoxy(0,0);
              lcd_putsf("Please Wait...");
              goto waitreset;
            }
           }


};
}
```

Lampiran D



Nama Menu

0

Data Receive

Status : idle

No	Nama Menu	Porsi	jumlah

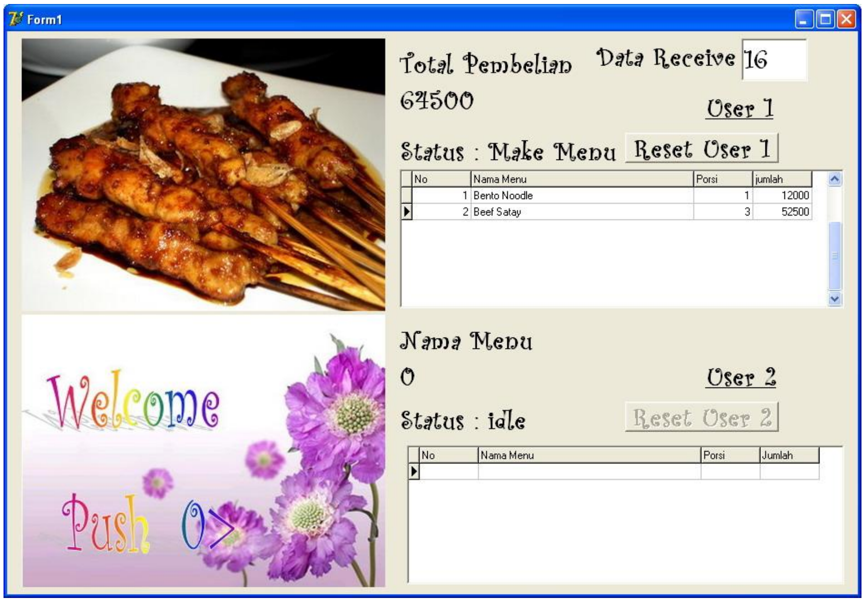
Nama Menu

0

Status : idle

No	Nama Menu	Porsi	Jumlah

Gambar Sistem menu



Total Pembelian

64500

Data Receive 16

Status : Make Menu

No	Nama Menu	Porsi	jumlah
1	Bento Noodle	1	12000
2	Beef Satay	3	52500

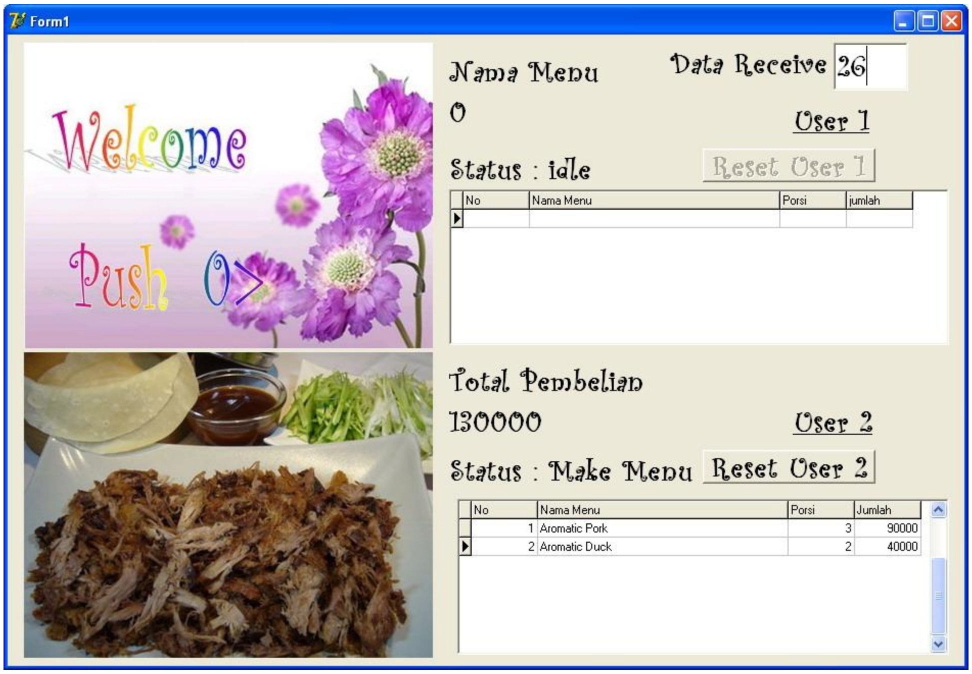
Nama Menu

0

Status : idle

No	Nama Menu	Porsi	Jumlah

Gambar user 1 memesan menu



Nama Menu
0

Data Receive 26

Status : idle

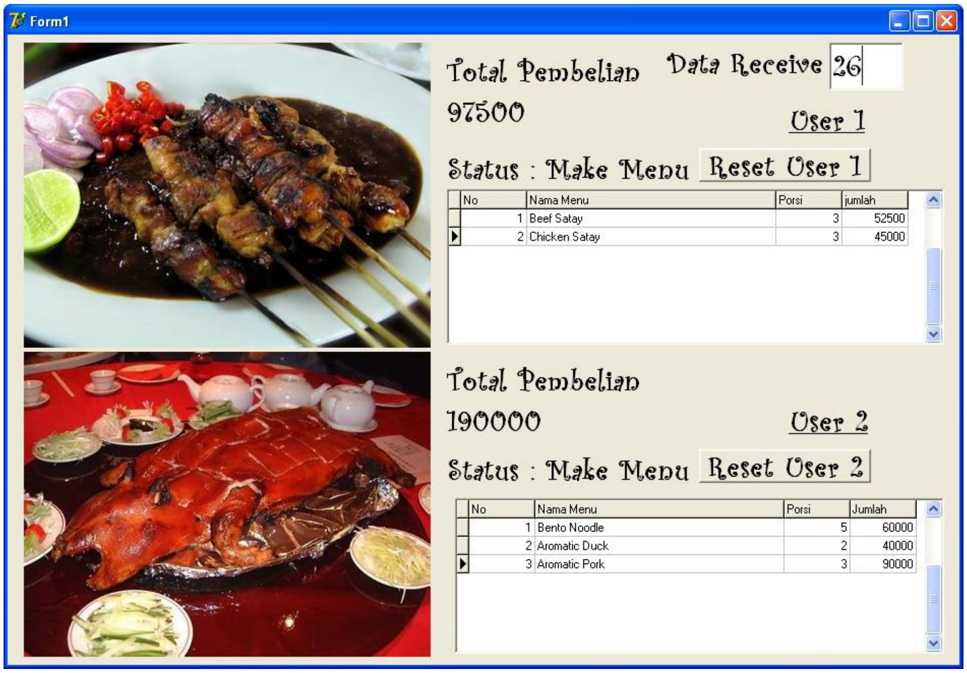
No	Nama Menu	Porsi	jumlah

Total Pembelian
130000

Status : Make Menu

No	Nama Menu	Porsi	Jumlah
1	Aromatic Pork	3	90000
2	Aromatic Duck	2	40000

Gambar User 2 memesan menu



Total Pembelian
97500

Data Receive 26

Status : Make Menu

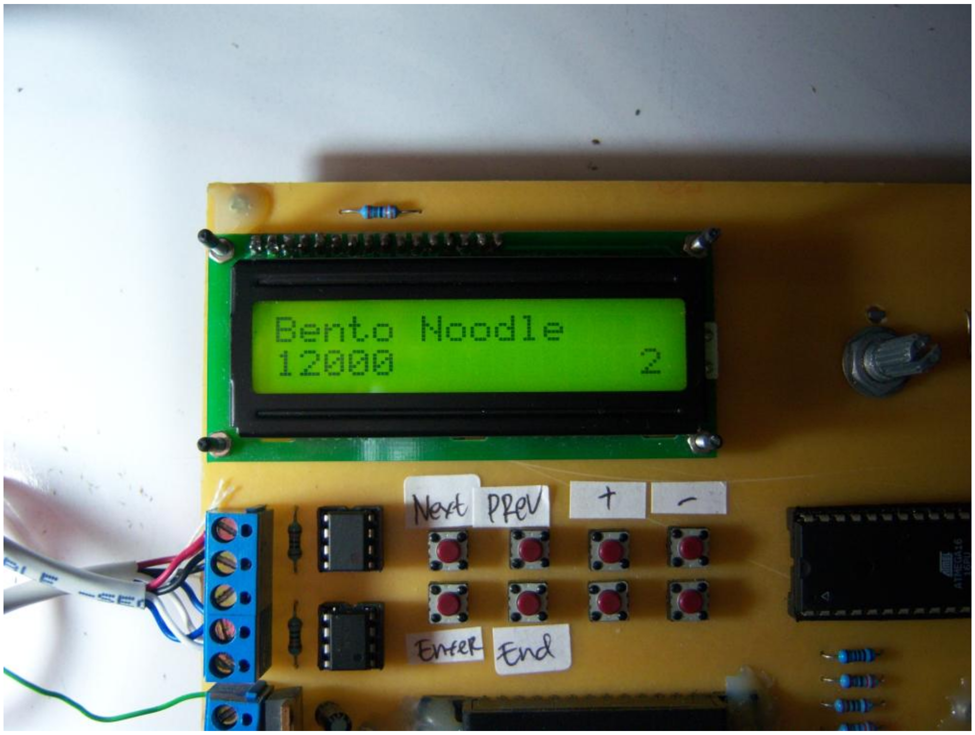
No	Nama Menu	Porsi	jumlah
1	Beef Satay	3	52500
2	Chicken Satay	3	45000

Total Pembelian
190000

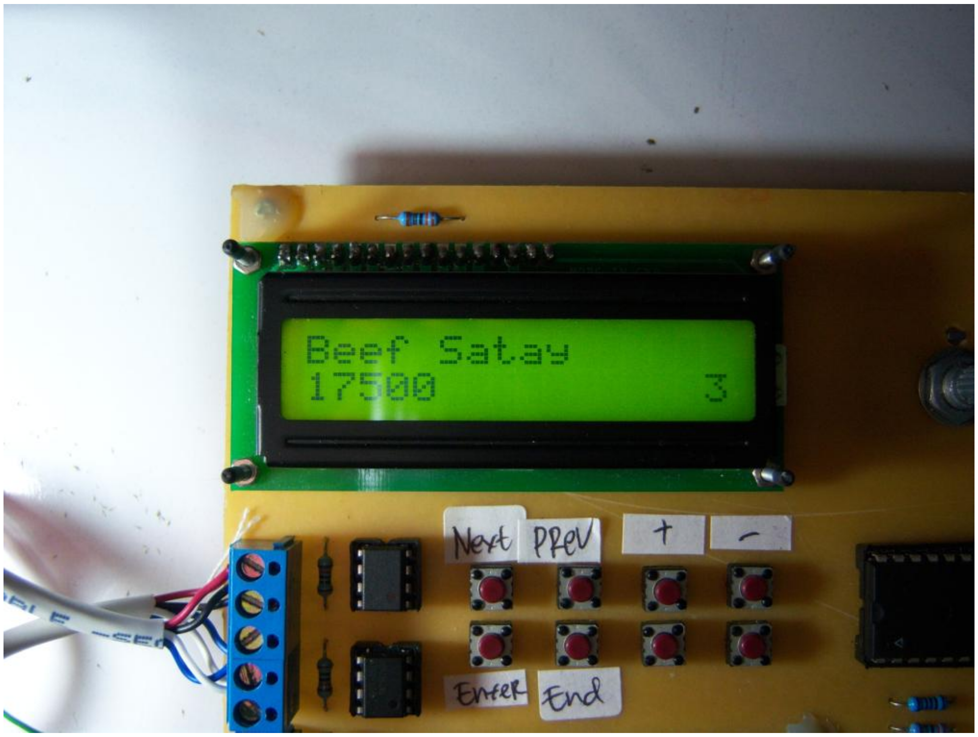
Status : Make Menu

No	Nama Menu	Porsi	Jumlah
1	Bento Noodle	5	60000
2	Aromatic Duck	2	40000
3	Aromatic Pork	3	90000

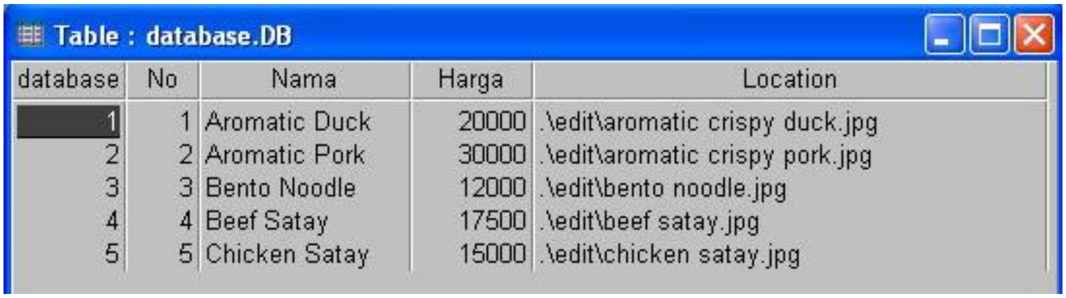
Gambar User 1 dan user 2 memesan menu



Gambar LCD menampilkan bento noodle



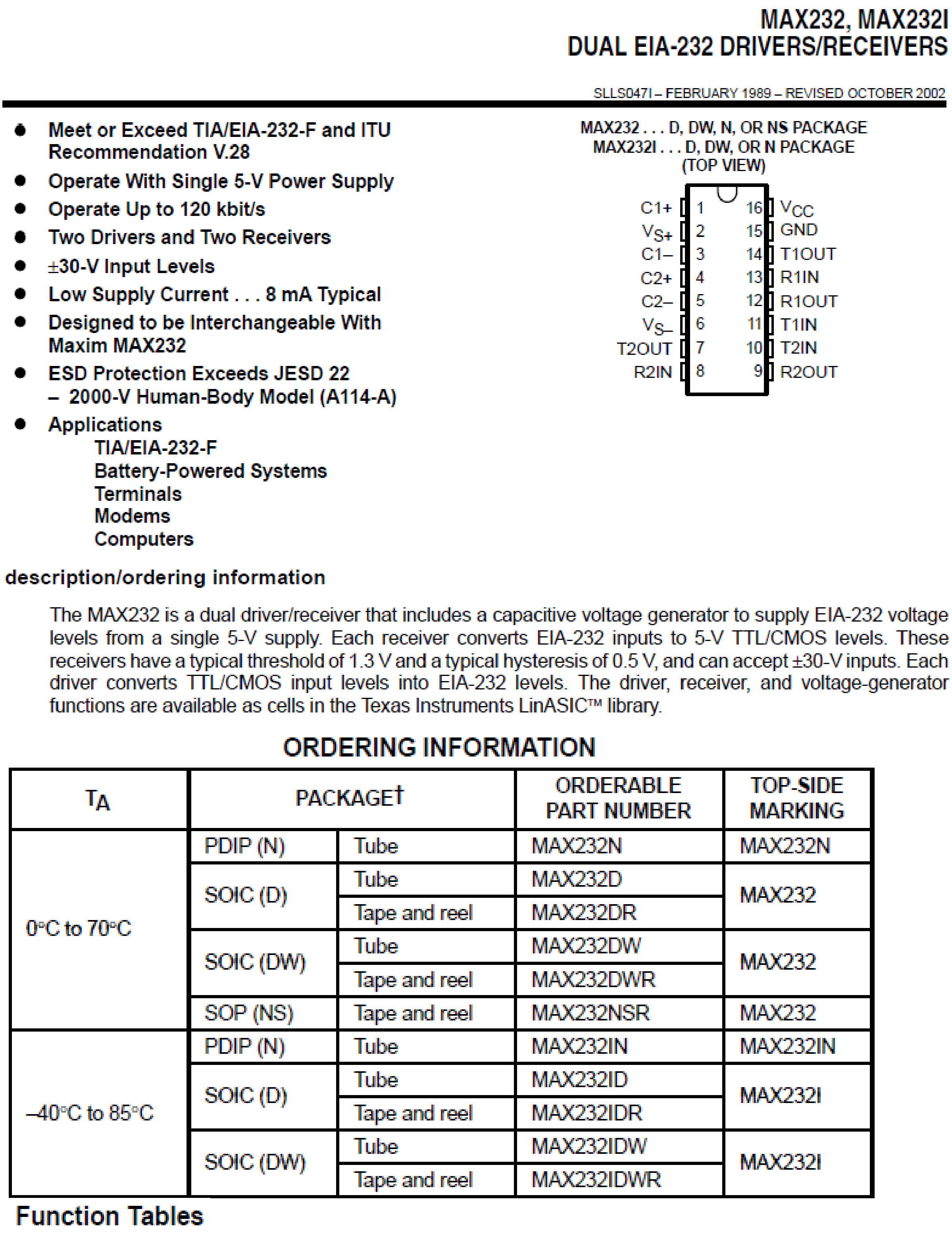
Gambar LCD menampilkan beef satay



database	No	Nama	Harga	Location
1	1	Aromatic Duck	20000	.\edit\aromatic crispy duck.jpg
2	2	Aromatic Pork	30000	.\edit\aromatic crispy pork.jpg
3	3	Bento Noodle	12000	.\edit\bento noodle.jpg
4	4	Beef Satay	17500	.\edit\beef satay.jpg
5	5	Chicken Satay	15000	.\edit\chicken satay.jpg

Gambar Database

Lampiran E



MAX232, MAX232I
DUAL EIA-232 DRIVERS/RECEIVERS

SLLS047I – FEBRUARY 1989 – REVISED OCTOBER 2002

- Meet or Exceed TIA/EIA-232-F and ITU Recommendation V.28
- Operate With Single 5-V Power Supply
- Operate Up to 120 kbit/s
- Two Drivers and Two Receivers
- ±30-V Input Levels
- Low Supply Current . . . 8 mA Typical
- Designed to be Interchangeable With Maxim MAX232
- ESD Protection Exceeds JESD 22
  – 2000-V Human-Body Model (A114-A)
- Applications
  TIA/EIA-232-F
  Battery-Powered Systems
  Terminals
  Modems
  Computers

MAX232 . . . D, DW, N, OR NS PACKAGE
MAX232I . . . D, DW, OR N PACKAGE
(TOP VIEW)

C1+	1	16	V_{CC}
V_{S+}	2	15	GND
C1–	3	14	T1OUT
C2+	4	13	R1IN
C2–	5	12	R1OUT
V_{S-}	6	11	T1IN
T2OUT	7	10	T2IN
R2IN	8	9	R2OUT

description/ordering information

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply EIA-232 voltage levels from a single 5-V supply. Each receiver converts EIA-232 inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V and a typical hysteresis of 0.5 V, and can accept ±30-V inputs. Each driver converts TTL/CMOS input levels into EIA-232 levels. The driver, receiver, and voltage-generator functions are available as cells in the Texas Instruments LinASIC™ library.

ORDERING INFORMATION

T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	PDIP (N)	Tube	MAX232N	MAX232N
	SOIC (D)	Tube	MAX232D	MAX232
		Tape and reel	MAX232DR	
	SOIC (DW)	Tube	MAX232DW	MAX232
		Tape and reel	MAX232DWR	
	SOP (NS)	Tape and reel	MAX232NSR	MAX232
–40°C to 85°C	PDIP (N)	Tube	MAX232IN	MAX232IN
	SOIC (D)	Tube	MAX232ID	MAX232I
		Tape and reel	MAX232IDR	
	SOIC (DW)	Tube	MAX232IDW	MAX232I
		Tape and reel	MAX232IDWR	

Function Tables

EACH DRIVER

INPUT TIN	OUTPUT TOUT
L	H
H	L

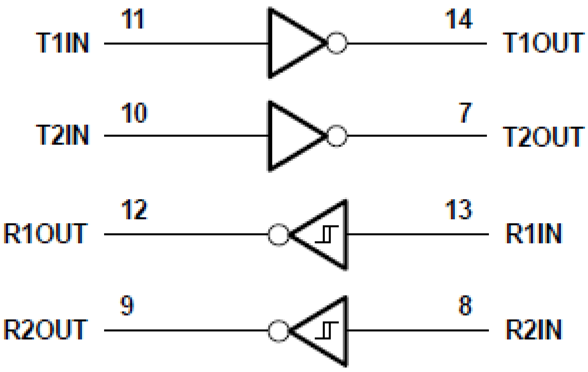
H = high level, L = low level

EACH RECEIVER

INPUT RIN	OUTPUT ROUT
L	H
H	L

H = high level, L = low level

logic diagram (positive logic)



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Input supply voltage range, V_{CC} (see Note 1)		–0.3 V to 6 V
Positive output supply voltage range, V_{S+}		V_{CC} – 0.3 V to 15 V
Negative output supply voltage range, V_{S-}		–0.3 V to –15 V
Input voltage range, V_I:	Driver	–0.3 V to V_{CC} + 0.3 V
	Receiver	±30 V
Output voltage range, V_O:	T1OUT, T2OUT	V_{S-} – 0.3 V to V_{S+} + 0.3 V
	R1OUT, R2OUT	–0.3 V to V_{CC} + 0.3 V
Short-circuit duration:	T1OUT, T2OUT	Unlimited
Package thermal impedance, θ_{JA} (see Note 2):	D package	73°C/W
	DW package	57°C/W
	N package	67°C/W
	NS package	64°C/W
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds		260°C
Storage temperature range, T_{stg}		–65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 1: All voltage values are with respect to network ground terminal.

2. The package thermal impedance is calculated in accordance with JESD 51-7.

recommended operating conditions

			MIN	NOM	MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	V
V_{IH}	High-level input voltage (T1IN,T2IN)		2			V
V_{IL}	Low-level input voltage (T1IN, T2IN)				0.8	V
R1IN, R2IN	Receiver input voltage				±30	V
T_A	Operating free-air temperature	MAX232	0		70	°C
		MAX232I	–40		85	

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted) (see Note 3 and Figure 4)

PARAMETER		TEST CONDITIONS	MIN	TYP‡	MAX	UNIT
I_{CC}	Supply current	V_{CC} = 5.5 V, All outputs open, T_A = 25°C		8	10	mA

‡ All typical values are at V_{CC} = 5 V and T_A = 25°C.

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

DRIVER SECTION

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature range (see Note 3)

PARAMETER			TEST CONDITIONS	MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	T1OUT, T2OUT	R_L = 3 kΩ to GND	5	7		V
V_{OL}	Low-level output voltage‡	T1OUT, T2OUT	R_L = 3 kΩ to GND		–7	–5	V
r_o	Output resistance	T1OUT, T2OUT	V_{S+} = $V_{S–}$ = 0, V_O = ±2 V	300			Ω
I_{OS}§	Short-circuit output current	T1OUT, T2OUT	V_{CC} = 5.5 V, V_O = 0		±10		mA
I_{IS}	Short-circuit input current	T1IN, T2IN	V_I = 0			200	µA

† All typical values are at V_{CC} = 5 V, T_A = 25°C.

‡ The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.

§ Not more than one output should be shorted at a time.

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 3)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
SR	Driver slew rate	R_L = 3 kΩ to 7 kΩ, See Figure 2			30	V/µs
SR(t)	Driver transition region slew rate	See Figure 3		3		V/µs
	Data rate	One TOUT switching		120		kbit/s

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

RECEIVER SECTION

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature range (see Note 3)

PARAMETER			TEST CONDITIONS	MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	R1OUT, R2OUT	I_{OH} = –1 mA	3.5			V
V_{OL}	Low-level output voltage‡	R1OUT, R2OUT	I_{OL} = 3.2 mA			0.4	V
V_{IT+}	Receiver positive-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C		1.7	2.4	V
$V_{IT–}$	Receiver negative-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C	0.8	1.2		V
V_{hys}	Input hysteresis voltage	R1IN, R2IN	V_{CC} = 5 V	0.2	0.5	1	V
r_i	Receiver input resistance	R1IN, R2IN	V_{CC} = 5, T_A = 25°C	3	5	7	kΩ

† All typical values are at VCC = 5 V, TA = 25°C.

‡ The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.

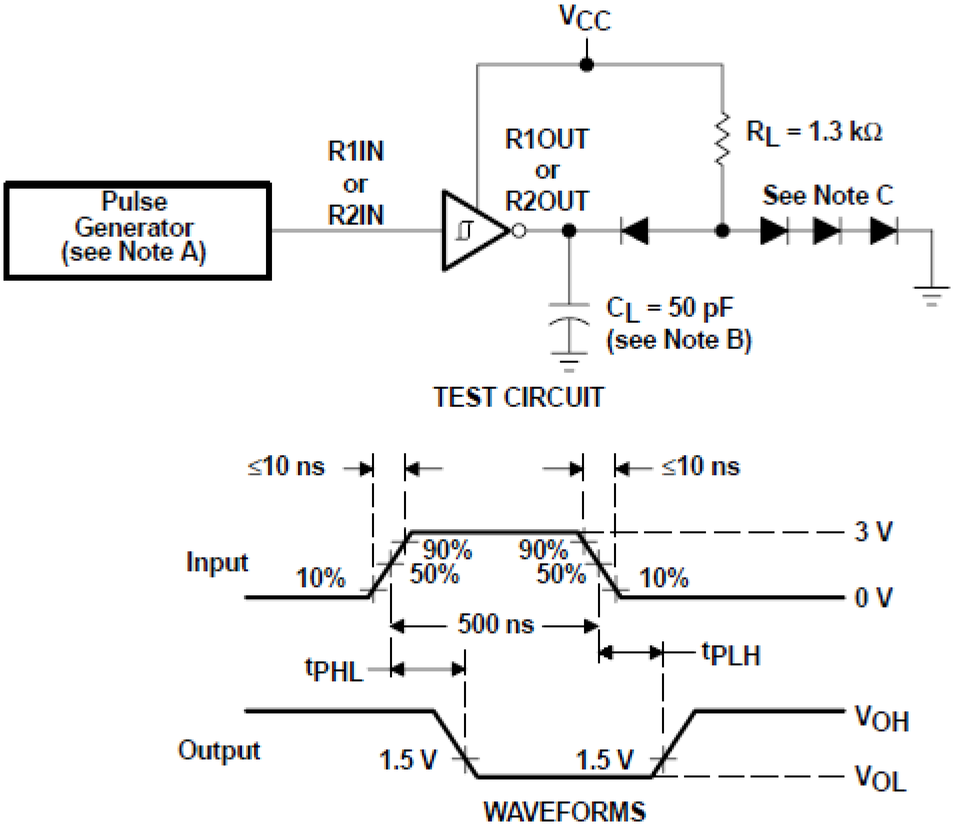
NOTE 3: Test conditions are C1–C4 = 1 ⌠F at VCC = 5 V ± 0.5 V.

switching characteristics, VCC = 5 V, TA = 25°C (see Note 3 and Figure 1)

PARAMETER		TYP	UNIT
$t_{PLH(R)}$	Receiver propagation delay time, low- to high-level output	500	ns
$t_{PHL(R)}$	Receiver propagation delay time, high- to low-level output	500	ns

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

PARAMETER MEASUREMENT INFORMATION



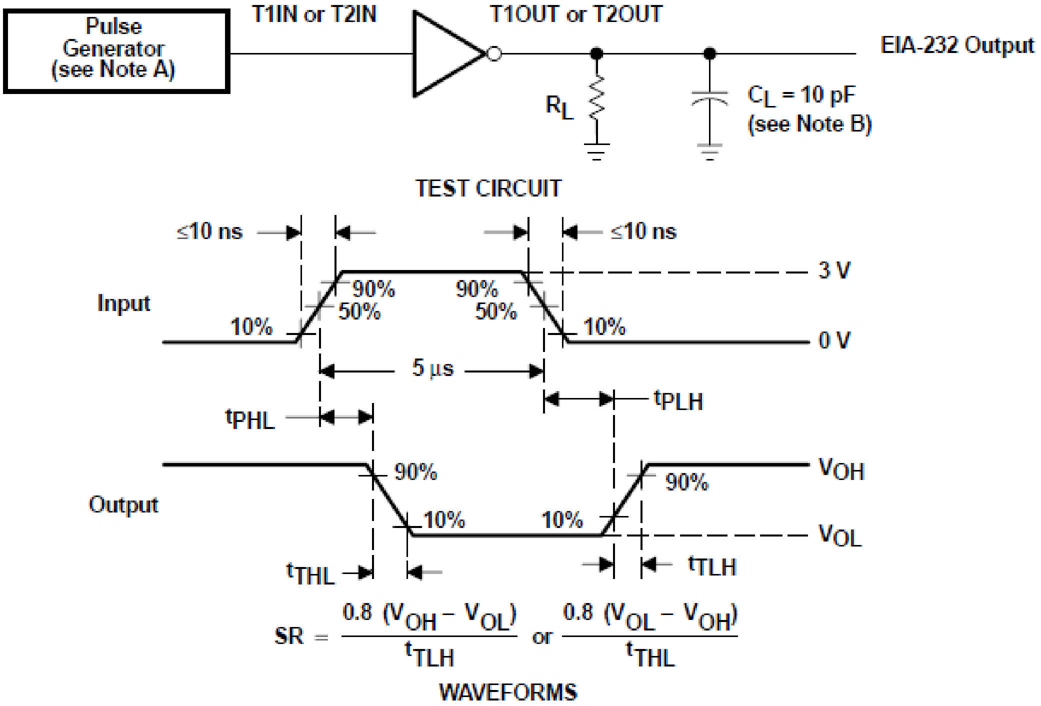
TEST CIRCUIT

WAVEFORMS

NOTES: A. The pulse generator has the following characteristics: ZO = 50 ∧, duty cycle δ 50%.
B. CL includes probe and jig capacitance.
C. All diodes are 1N3064 or equivalent.

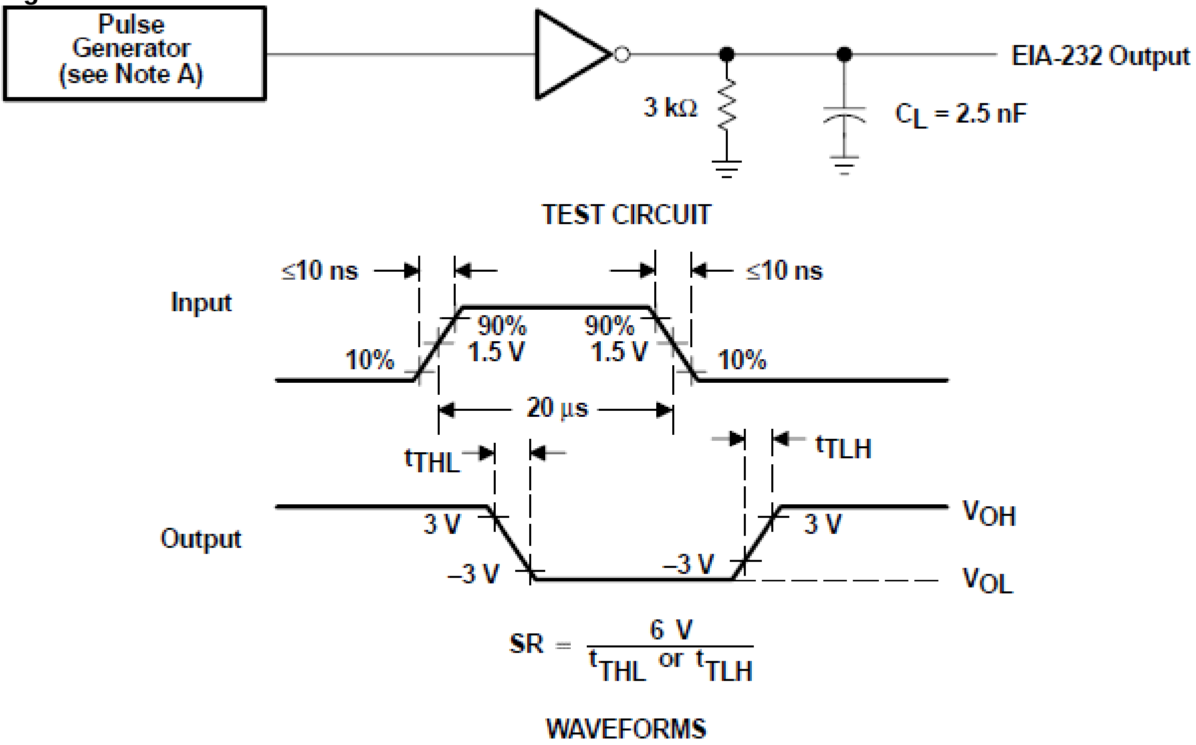
Figure 1. Receiver Test Circuit and Waveforms for tPHL and tPLH Measurements

PARAMETER MEASUREMENT INFORMATION



NOTES: A. The pulse generator has the following characteristics: ZO = 50 ∧, duty cycle δ 50%.
B. CL includes probe and jig capacitance.

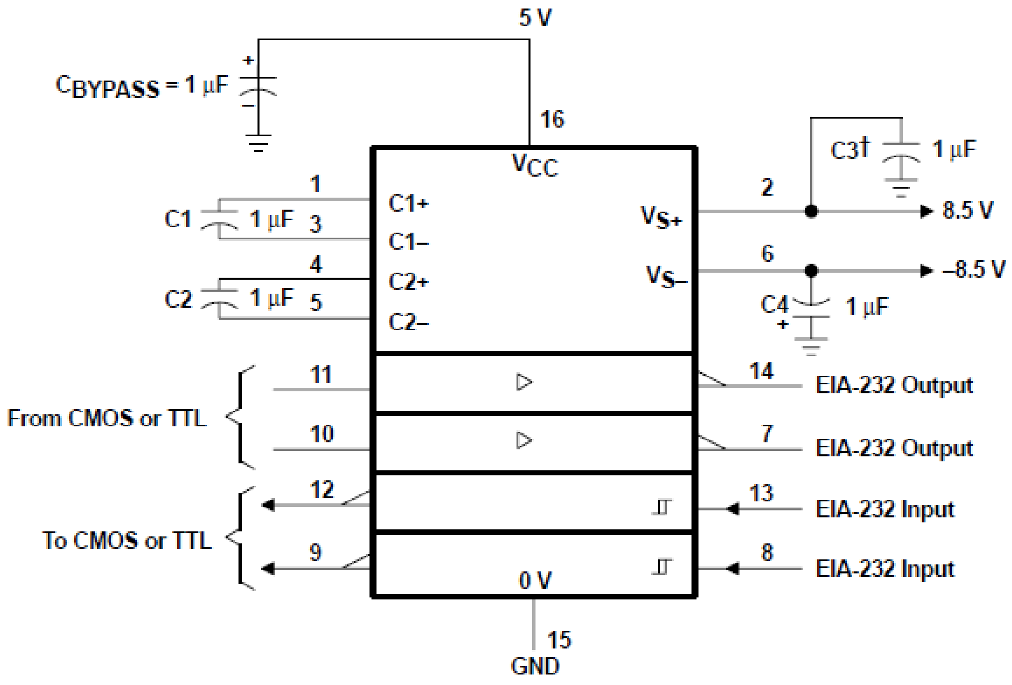
Figure 2. Driver Test Circuit and Waveforms for tPHL and tPLH Measurements (5-µs Input)



NOTE A: The pulse generator has the following characteristics: ZO = 50 Ω, duty cycle ≤ 50%.

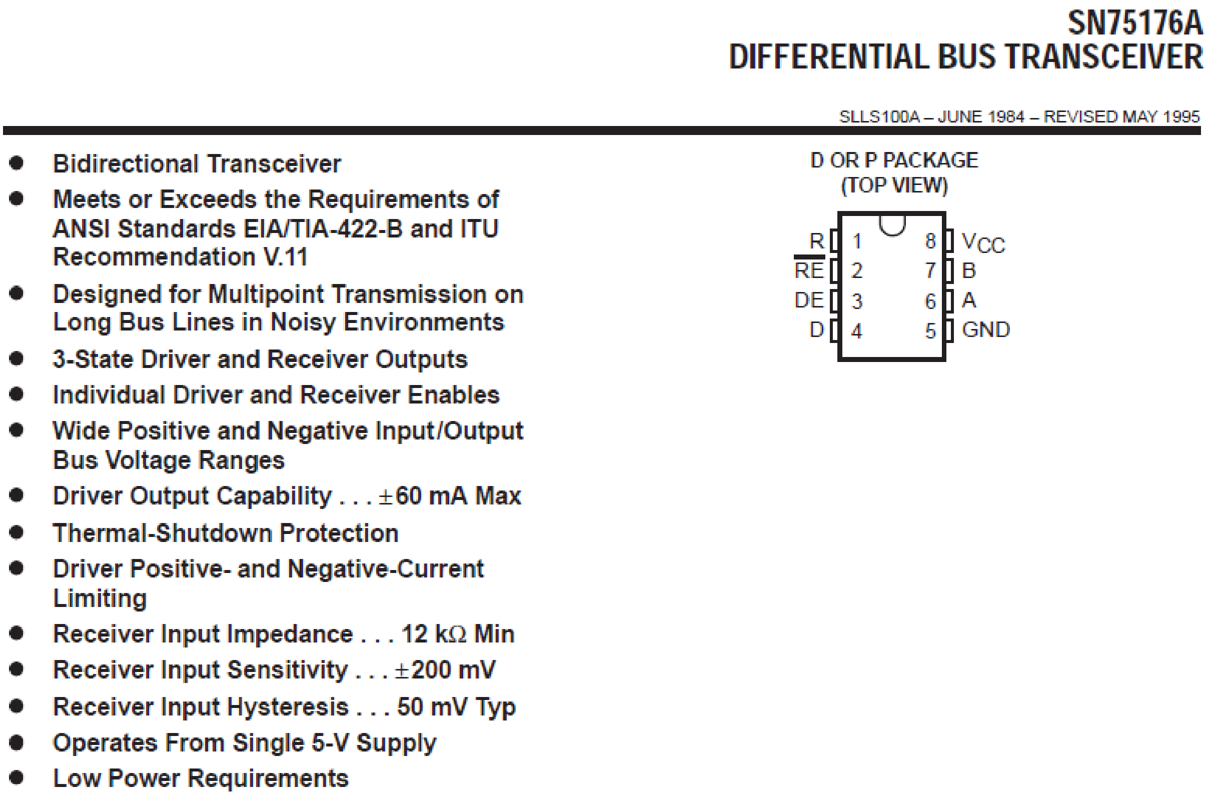
Figure 3. Test Circuit and Waveforms for tTHL and tTLH Measurements (20-µs Input)

APPLICATION INFORMATION



† C3 can be connected to V_{CC} or GND.

Lampiran F



SN75176A
DIFFERENTIAL BUS TRANSCEIVER

SLLS100A – JUNE 1984 – REVISED MAY 1995

- Bidirectional Transceiver
- Meets or Exceeds the Requirements of ANSI Standards EIA/TIA-422-B and ITU Recommendation V.11
- Designed for Multipoint Transmission on Long Bus Lines in Noisy Environments
- 3-State Driver and Receiver Outputs
- Individual Driver and Receiver Enables
- Wide Positive and Negative Input/Output Bus Voltage Ranges
- Driver Output Capability . . . ±60 mA Max
- Thermal-Shutdown Protection
- Driver Positive- and Negative-Current Limiting
- Receiver Input Impedance . . . 12 kΩ Min
- Receiver Input Sensitivity . . . ±200 mV
- Receiver Input Hysteresis . . . 50 mV Typ
- Operates From Single 5-V Supply
- Low Power Requirements

D OR P PACKAGE
(TOP VIEW)

description

The SN75176A differential bus transceiver is a monolithic integrated circuit designed for bidirectional data communication on multipoint bus-transmission lines. It is designed for balanced transmission lines and meets ANSI Standard EIA/TIA-422-B and ITU Recommendation V.11.

The SN75176A combines a 3-state differential line driver and a differential input line receiver, both of which operate from a single 5-V power supply. The driver and receiver have active-high and active-low enables, respectively, that can be externally connected together to function as a direction control. The driver differential outputs and the receiver differential inputs are connected internally to form differential input/output (I/O) bus ports that are designed to offer minimum loading to the bus whenever the driver is disabled or VCC = 0. These ports feature wide positive and negative common-mode voltage ranges making the device suitable for party-line applications.

The driver is designed to handle loads up to 60 mA of sink or source current. The driver features positive- and negative-current limiting and thermal shutdown for protection from line fault conditions. Thermal shutdown is designed to occur at a junction temperature of approximately 150°C. The receiver features a minimum input impedance of 12 kW, an input sensitivity of ±200 mV, and a typical input hysteresis of 50 mV.

The SN75176A can be used in transmission-line applications employing the SN75172 and SN75174 quadruple differential line drivers and SN75173 and SN75175 quadruple differential line receivers. The SN75176A is characterized for operation from 0°C to 70°C.

Function Tables

DRIVER

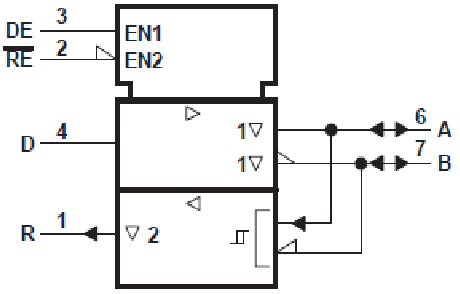
INPUT D	ENABLE DE	OUTPUTS	
		A	B
H	H	H	L
L	H	L	H
X	L	Z	Z

RECEIVER

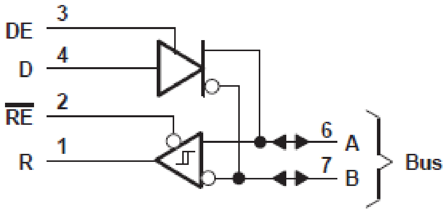
DIFFERENTIAL INPUTS A – B	ENABLE $\overline{RE}$	OUTPUT R
$V_{ID} \geq 0.2$ V	L	H
-0.2 V $< V_{ID} < 0.2$ V	L	?
$V_{ID} \leq -0.2$ V	L	L
X	H	Z
Open	L	?

H = high level, L = low level, ? = indeterminate,
X = irrelevant, Z = high impedance (off)

logic symbol†

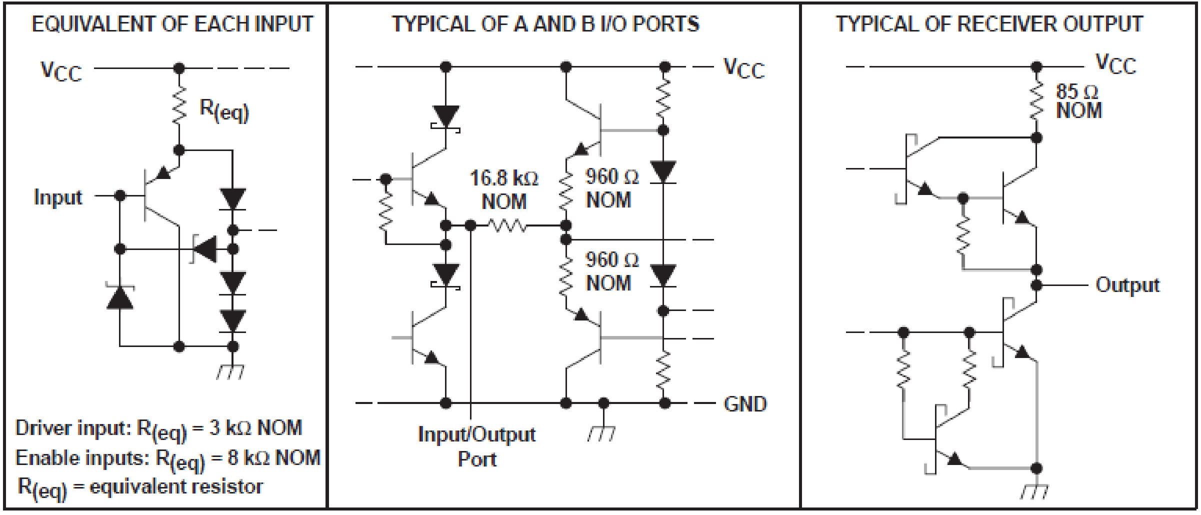


logic diagram (positive logic)

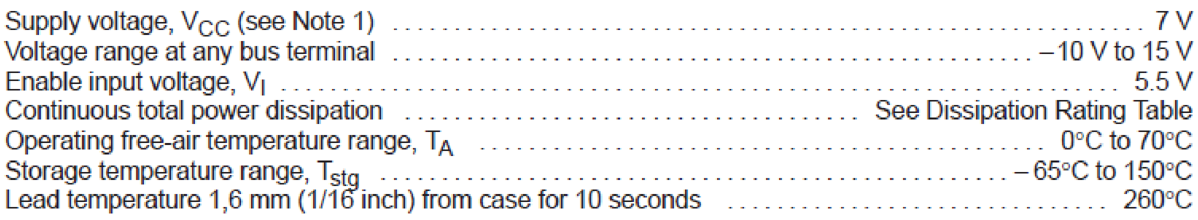


† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

schematics of inputs and outputs

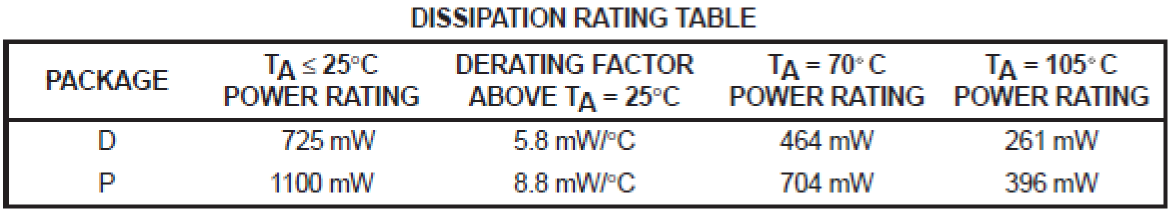

EQUIVALENT OF EACH INPUT
V_{CC}
$R_{(eq)}$
Input
Driver input: $R_{(eq)}$ = 3 kΩ NOM
Enable inputs: $R_{(eq)}$ = 8 kΩ NOM
$R_{(eq)}$ = equivalent resistor
TYPICAL OF A AND B I/O PORTS
V_{CC}
16.8 kΩ NOM
960 Ω NOM
960 Ω NOM
GND
Input/Output Port
TYPICAL OF RECEIVER OUTPUT
V_{CC}
85 Ω NOM
Output

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†



Supply voltage, V_{CC} (see Note 1)	7 V
Voltage range at any bus terminal	−10 V to 15 V
Enable input voltage, V_I	5.5 V
Continuous total power dissipation	See Dissipation Rating Table
Operating free-air temperature range, T_A	0°C to 70°C
Storage temperature range, T_{stg}	−65°C to 150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds	260°C

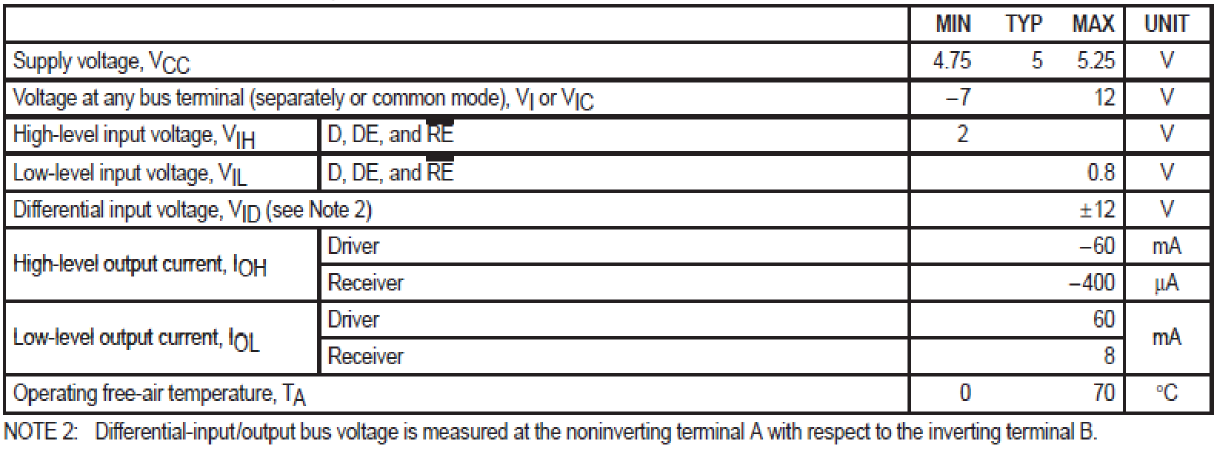
† Stresses beyond those listed under “absolute maximum ratings” may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under “recommended operating conditions” is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.
NOTE 1: All voltage values, except differential input/output bus voltage, are with respect to network ground terminal.



DISSIPATION RATING TABLE

PACKAGE	$T_A \leq 25°C$ POWER RATING	DERATING FACTOR ABOVE $T_A = 25°C$	$T_A = 70°C$ POWER RATING	$T_A = 105°C$ POWER RATING
D	725 mW	5.8 mW/°C	464 mW	261 mW
P	1100 mW	8.8 mW/°C	704 mW	396 mW

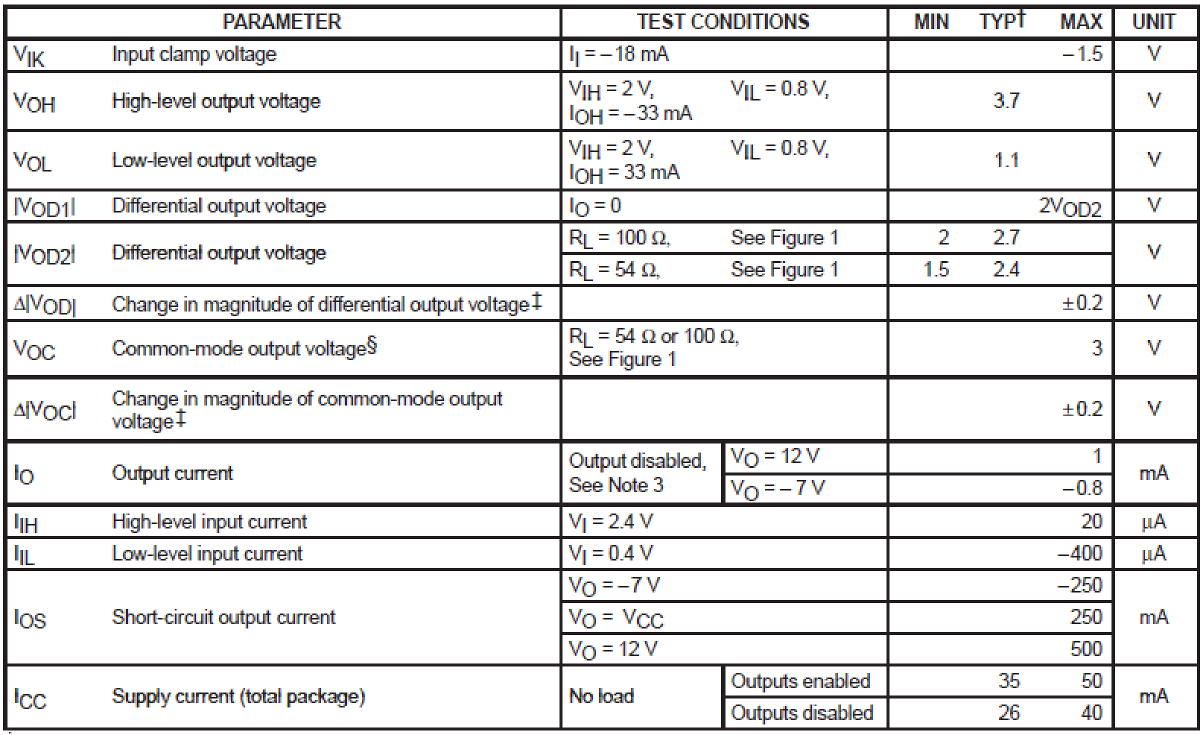
recommended operating conditions



		MIN	TYP	MAX	UNIT
Supply voltage, V_{CC}		4.75	5	5.25	V
Voltage at any bus terminal (separately or common mode), V_I or V_{IC}		−7		12	V
High-level input voltage, V_{IH}	D, DE, and $\overline{\text{RE}}$	2			V
Low-level input voltage, V_{IL}	D, DE, and $\overline{\text{RE}}$			0.8	V
Differential input voltage, V_{ID} (see Note 2)				±12	V
High-level output current, I_{OH}	Driver			−60	mA
	Receiver			−400	µA
Low-level output current, I_{OL}	Driver			60	mA
	Receiver			8	
Operating free-air temperature, T_A		0		70	°C

NOTE 2: Differential-input/output bus voltage is measured at the noninverting terminal A with respect to the inverting terminal B.

DRIVER SECTION
electrical characteristics over recommended ranges of supply voltage and operating free-air
temperature (unless otherwise noted)



PARAMETER		TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{IK}	Input clamp voltage	$I_I = -18$ mA				–1.5	V
V_{OH}	High-level output voltage	$V_{IH} = 2$ V, $I_{OH} = -33$ mA	$V_{IL} = 0.8$ V,		3.7		V
V_{OL}	Low-level output voltage	$V_{IH} = 2$ V, $I_{OH} = 33$ mA	$V_{IL} = 0.8$ V,		1.1		V
$\|V_{OD1}\|$	Differential output voltage	$I_O = 0$				$2V_{OD2}$	V
$\|V_{OD2}\|$	Differential output voltage	$R_L = 100\ \Omega$,	See Figure 1	2	2.7		V
		$R_L = 54\ \Omega$,	See Figure 1	1.5	2.4		
$\Delta\|V_{OD}\|$	Change in magnitude of differential output voltage‡					±0.2	V
V_{OC}	Common-mode output voltage§	$R_L = 54\ \Omega$ or $100\ \Omega$, See Figure 1				3	V
$\Delta\|V_{OC}\|$	Change in magnitude of common-mode output voltage‡					±0.2	V
I_O	Output current	Output disabled, See Note 3	$V_O = 12$ V			1	mA
			$V_O = -7$ V			–0.8	
I_{IH}	High-level input current	$V_I = 2.4$ V				20	μA
I_{IL}	Low-level input current	$V_I = 0.4$ V				–400	μA
I_{OS}	Short-circuit output current	$V_O = -7$ V				–250	mA
		$V_O = V_{CC}$				250	
		$V_O = 12$ V				500	
I_{CC}	Supply current (total package)	No load	Outputs enabled		35	50	mA
			Outputs disabled		26	40	

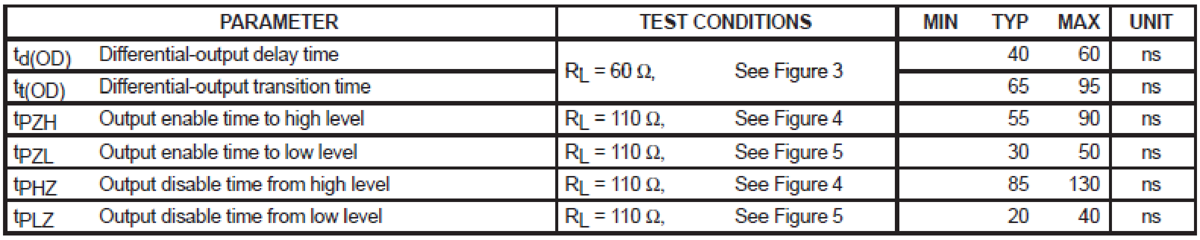
† All typical values are at VCC = 5 V and TA = 25°C.

‡ Δ|VOD| and Δ|VOC| are the changes in magnitude of VOD and VOC respectively, that occur when the input is changed from a high level to a low
level.

§ In ANSI Standard EIA/TIA-422-B, VOC, which is the average of the two output voltages with respect to GND, is called output offset voltage, VOS.

NOTE 3: This applies for both power on and off; refer to ANSI Standard EIA/TIA-422-B for exact conditions.

switching characteristics, VCC = 5 V, TA = 25°C



PARAMETER		TEST CONDITIONS		MIN	TYP	MAX	UNIT
$t_{d(OD)}$	Differential-output delay time	$R_L = 60\ \Omega$,	See Figure 3		40	60	ns
$t_{t(OD)}$	Differential-output transition time				65	95	ns
t_{PZH}	Output enable time to high level	$R_L = 110\ \Omega$,	See Figure 4		55	90	ns
t_{PZL}	Output enable time to low level	$R_L = 110\ \Omega$,	See Figure 5		30	50	ns
t_{PHZ}	Output disable time from high level	$R_L = 110\ \Omega$,	See Figure 4		85	130	ns
t_{PLZ}	Output disable time from low level	$R_L = 110\ \Omega$,	See Figure 5		20	40	ns

RECEIVER SECTION

electrical characteristics over recommended ranges of common-mode input voltage, supply voltage, and operating free-air temperature (unless otherwise noted)

PARAMETER		TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{IT+}	Positive-going input threshold voltage	V_O = 2.7 V,	I_O = −0.4 mA			0.2	V
V_{IT-}	Negative-going input threshold voltage	V_O = 0.5 V,	I_O = 8 mA	−0.2‡			V
V_{hys}	Input hysteresis voltage ($V_{IT+} - V_{IT-}$)				50		mV
V_{IK}	Enable clamp voltage	I_I = −18 mA				−1.5	V
V_{OH}	High-level output voltage	V_{ID} = 200 mV, See Figure 2	I_{OH} = −400 µA,	2.7			V
V_{OL}	Low-level output voltage	V_{ID} = −200 mV, See Figure 2	I_{OL} = 8 mA,			0.45	V
I_{OZ}	High-impedance-state output current	V_O = 0.4 V to 2.4 V				±20	µA
I_I	Line input current	Other input = 0 V, See Note 3	V_I = 12 V			1	mA
			V_I = −7 V			−0.8	
I_{IH}	High-level enable input current	V_{IH} = 2.7 V				20	µA
I_{IL}	Low-level enable input current	V_{IL} = 0.4 V				−100	µA
r_i	Input resistance			12			kΩ
I_{OS}	Short-circuit output current			−15		−85	mA
I_{CC}	Supply current (total package)	No load	Outputs enabled		35	50	mA
			Outputs disabled		26	40	

† All typical values are at VCC = 5 V, TA = 25°C.

‡ The algebraic convention, in which the less-positive (more-negative) limit is designated minimum, is used in this data sheet for common-mode
input voltage and threshold voltage levels only.

NOTE 3: This applies for both power on and power off. Refer to ANSI Standard EIA/TIA-422-B for exact conditions.

switching characteristics, VCC = 5 V, CL = 15 pF, TA = 25°C

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
t_{PLH}	Propagation delay time, low-to-high-level output	V_{ID} = −1.5 V to 1.5 V, See Figure 6		21	35	ns
t_{PHL}	Propagation delay time, high-to-low-level output			23	35	ns
t_{PZH}	Output enable time to high level	See Figure 7		10	30	ns
t_{PZL}	Output enable time to low level			12	30	ns
t_{PHZ}	Output disable time from high level	See Figure 7		20	35	ns
t_{PLZ}	Output disable time from low level			17	25	ns

PARAMETER MEASUREMENT INFORMATION

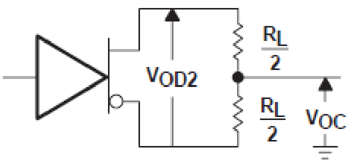


Figure 1. Driver V_{OD} and V_{OC}

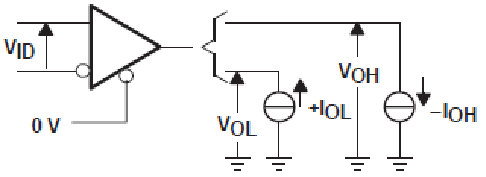
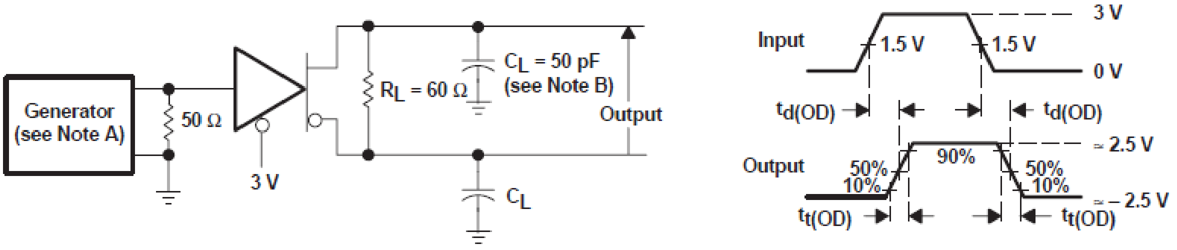


Figure 2. Receiver V_{OH} and V_{OL}

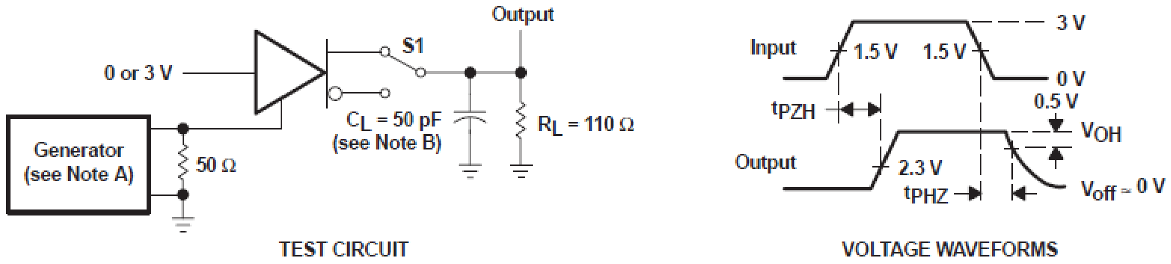


TEST CIRCUIT

VOLTAGE WAVEFORMS

NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, 50% duty cycle, tr 3 6 ns, tf 3 6 ns,
ZO = 50 Ω.
B. CL includes probe and jig capacitance.

Figure 3. Driver Test Circuit and Voltage Waveforms

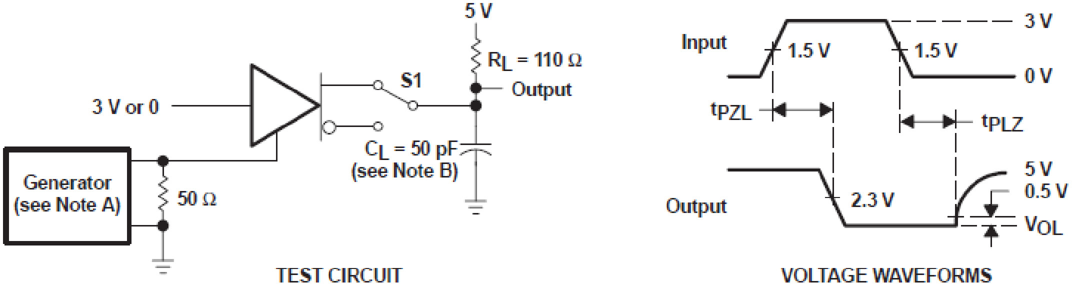


NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, 50% duty cycle, tr 3 6 ns, tf 3 6 ns,

ZO = 50 Ω.

B. CL includes probe and jig capacitance.

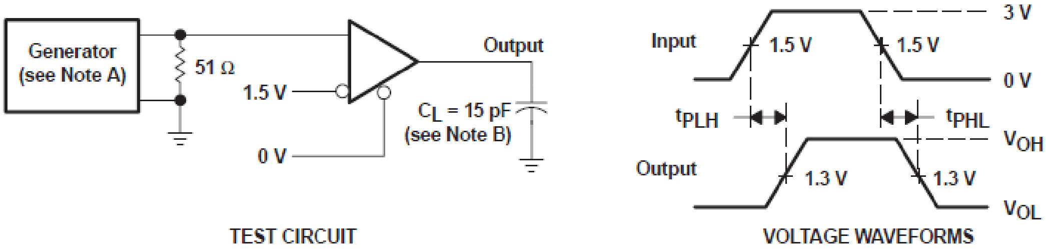
Figure 4. Driver Test Circuit and Voltage Waveforms



NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, 50% duty cycle, $t_r \leq 6$ ns, $t_f \leq 6$ ns, $Z_O = 50\ \Omega$.

B. C_L includes probe and jig capacitance.

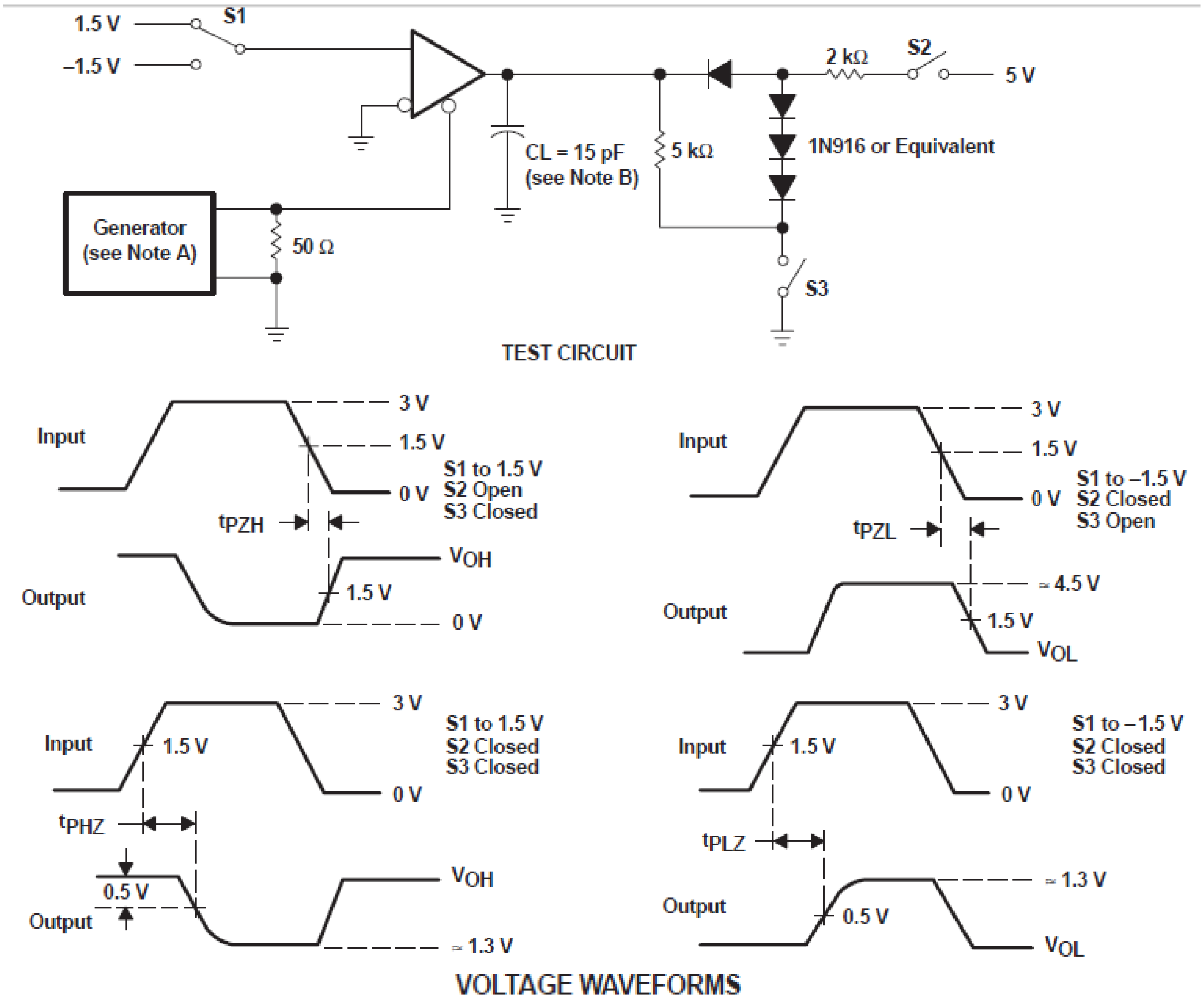
Figure 5. Driver Test Circuit and Voltage Waveforms



NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, 50% duty cycle, $t_r \leq 6$ ns, $t_f \leq 6$ ns, $Z_O = 50\ \Omega$.

B. C_L includes probe and jig capacitance.

Figure 6. Receiver Test Circuit and Voltage Waveforms



NOTES: A. The input pulse is supplied by a generator having the following characteristics: PRR = 1 MHz, 50% duty cycle, tr 3 6 ns, tf 3 6 ns,
ZO = 50 Ω.
B. CL includes probe and jig capacitance.

Figure 7. Receiver Test Circuit and Voltage Waveforms

TYPICAL CHARACTERISTICS

DRIVER
HIGH-LEVEL OUTPUT VOLTAGE
vs
HIGH-LEVEL OUTPUT CURRENT

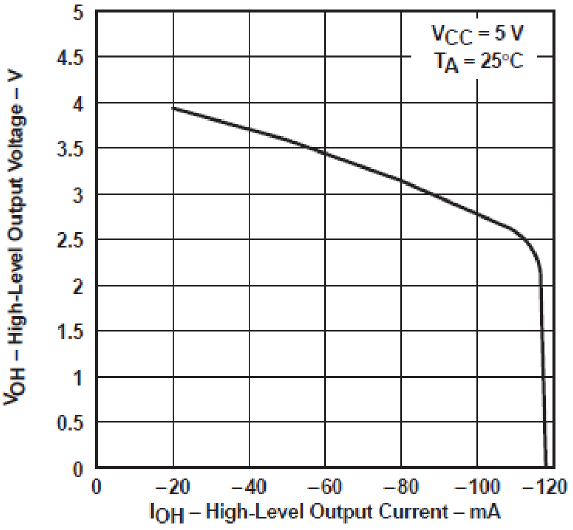


Figure 8

DRIVER
LOW-LEVEL OUTPUT VOLTAGE
vs
LOW-LEVEL OUTPUT CURRENT

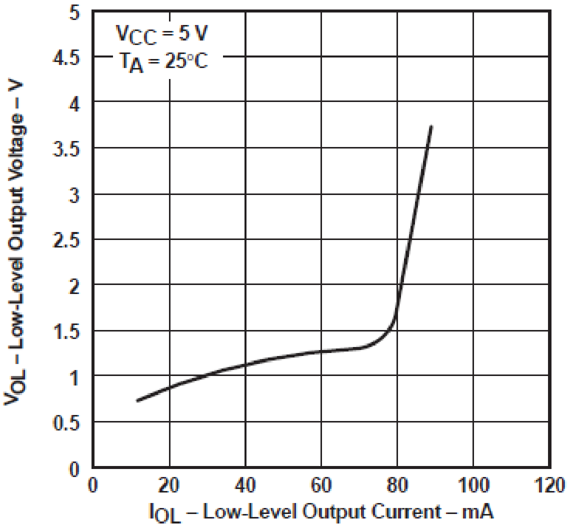


Figure 9

DRIVER
DIFFERENTIAL OUTPUT VOLTAGE
vs
OUTPUT CURRENT

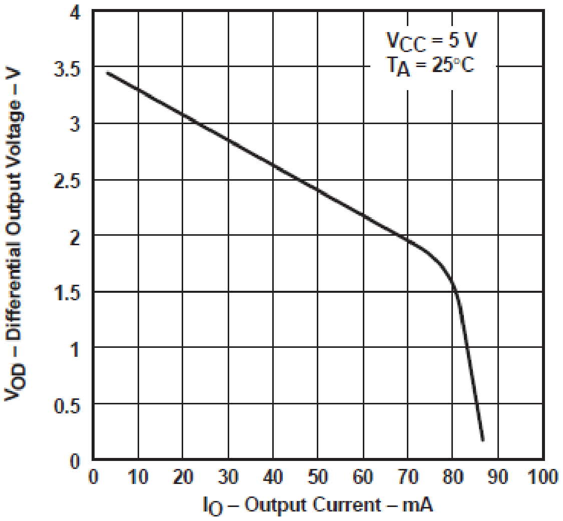


Figure 10

RECEIVER
LOW-LEVEL OUTPUT VOLTAGE
vs
LOW-LEVEL OUTPUT CURRENT

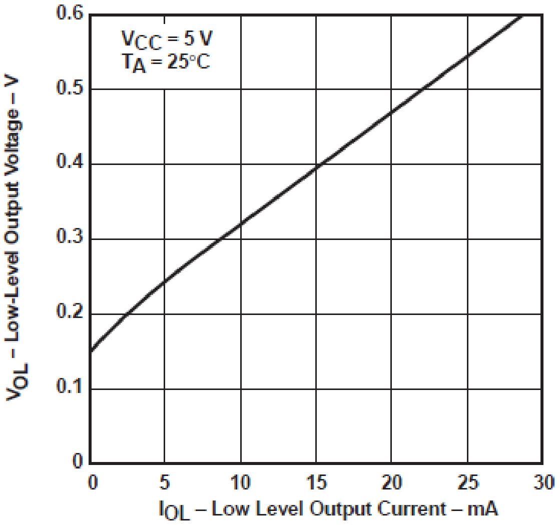


Figure 11

TYPICAL CHARACTERISTICS

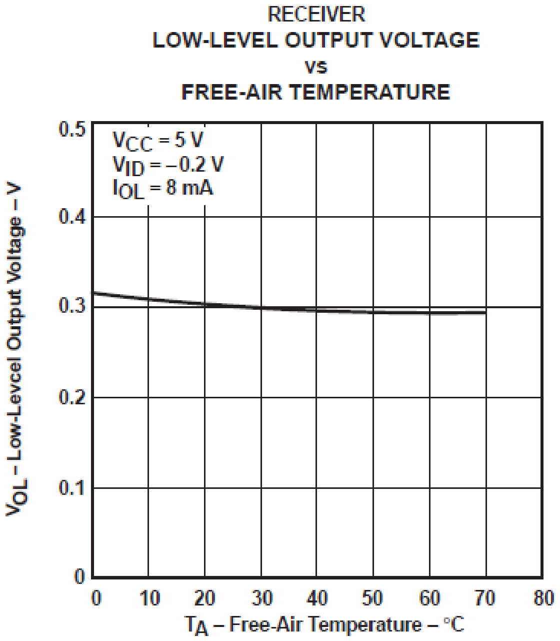


Figure 12

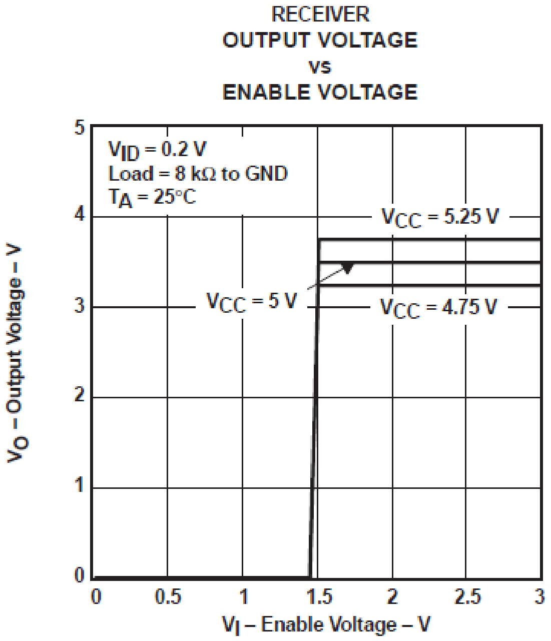


Figure 13

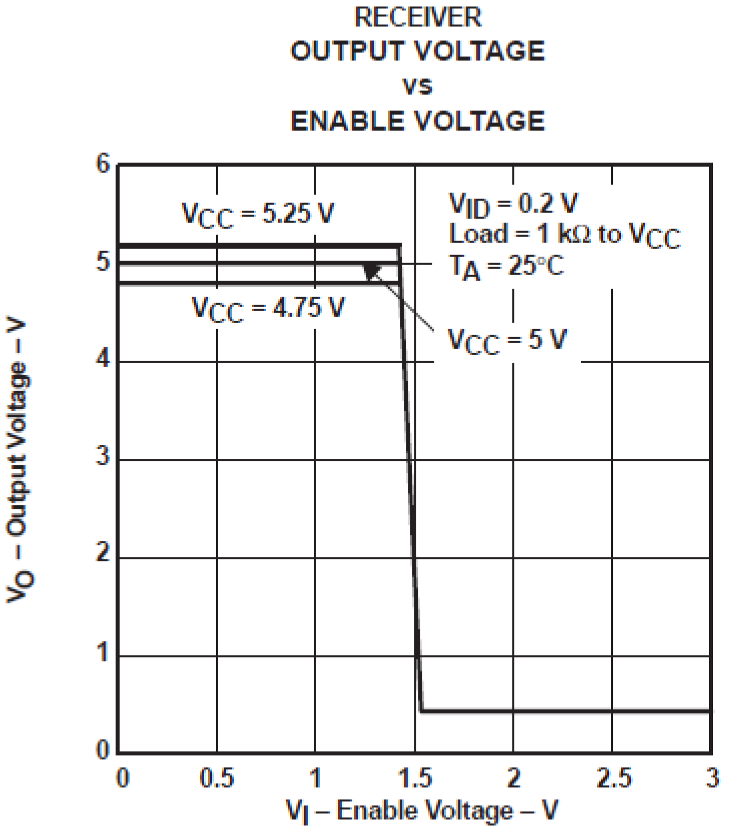
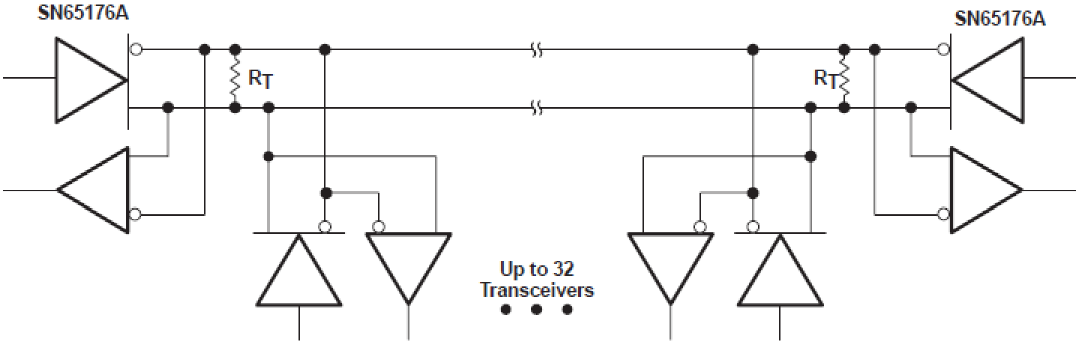


Figure 14

APPLICATION INFORMATION



NOTE A: The line should be terminated at both ends in its characteristic impedance ($R_T = Z_O$). Stub lengths off the main line should be kept as short as possible.

Figure 15. Typical Application Circuit