LAMPIRAN - LAMPIRAN

LAMPIRAN A : PROGRAM

LAMPIRAN B : DATA HASIL PERCOBAAN

LAMPIRAN C : LEMBAR DATA PERCOBAAN

LAMPIRAN A : PROGRAM

```
program kecepatan_motor_DC;
{ ************************************************************
  *                                                          *
  *          PROGRAM KONTROL KECEPATAN MOTOR DC              *
  *                                                          *
  *                NAMA      : ERIEKE Y ANTONY             *
  *                NIM       : J 401 90 0476               *
  *                JURUSAN   : F I S I K A                 *
  *                FAKULTAS  : M I P A                     *
  *                                                          *
  ************************************************************ }

Uses CRT,DOS,GRAPH;
const
   f10 = $4400;
   EscKey = $1B;          { konstanta nilai keyboard }
   EnterKey = $0D;
   UpKey = $4800;
   DownKey = $5000;
   LeftKey = $4B00;
   RightKey = $4D00;
   totalmem =4100;
   maxscr =10;
   BackSpaceKey = $08;
   jum_menu_utama = 5;
   maxorde      = 8;
   data_menu_utama : array[1..jum_menu_utama] of STRING
            = ('1. DATA MASUKAN  ',
               '2. SETUP         ',
               '3. RUN           ',
               '4. KARAKTERISTIK  ',
               '5. EXIT          ');
   data_menu    : array[1..jum_menu_utama-1] of string
          = (' DATA MASUKAN ',' SETUP ',' RUN ','KARAKTERISTIK');

VAR
  V_outs,i,pilih_lama,pilih_baru,pilihan : byte;
  ch,DAC : word;
  toprint,esc,okay,positif,MODE:boolean;
  Strin: STRING;
  masuk :word;
```

```
  tanda : array[1..2] of string;
  datpilsetup : array[1..2] of word;
  dat         : array[0..15] of word;
  RPM,RPM_In,beban :Integer;

{$I scr.pas}
{$I exec.pas}
{$I grafik.pas}

PROCEDURE In_Data(x:byte);
BEGIN
   case x of
   1   : In_RPM;
   2   : In_Beban;
   end;
   refresh;refresh;
END;

PROCEDURE setup(x:byte);
BEGIN
   case x of
   1   :begin
       Test_Alat;end;
   2   :begin
       alamat;refresh;end;
   end;
   refresh;
END;

PROCEDURE sub_menu2(pilihan : byte);
CONST
   jum_sub_menu  = 2;
   data_sub_menu2: array[1..jum_sub_menu] of string
            = ('1. TEST ALAT  ',
               '2. ALAMAT PPI ');
   data_sub_menu1: array[1..jum_sub_menu] of string
            = ('1. DATA RPM   ',
               '2. DATA BEBAN ');
```

```
VAR
  _i,_pilih_lama,_pilih_baru,_pilihan : byte;

BEGIN
   my_window(data_menu[pilihan],40,13,60,17,BLUE,WHITE,true);
   highlight1;
   if pilihan = 1 then begin
   gotoxy(2,1);write(data_sub_menu1[1]);
   normal1;
   gotoxy(2,2);write(data_sub_menu1[2]);end
   else begin
   gotoxy(2,1);write(data_sub_menu2[1]);
   normal1;
   gotoxy(2,2);write(data_sub_menu2[2]);end;

   repeat
   _pilih_lama:=1;
   _pilih_baru:=1;
   REPEAT
   ch:=getkey;

   CASE ch OF
      ord('1') :
            BEGIN
               _pilih_baru:=1;
               _pilihan :=1;
            END;
      ord('2') :
            BEGIN
               _pilih_baru:=2;
               _pilihan :=2;
            END;
      upkey :
          BEGIN
             IF _pilih_lama=1 THEN _pilih_baru:=jum_sub_menu
             ELSE _pilih_baru:= _pilih_lama-1;
          END;
      downkey :
           BEGIN
              IF _pilih_lama=jum_sub_menu THEN _pilih_baru:=1
```

```
                ELSE _pilih_baru:=_pilih_lama+1;
            END;
        enterkey :
            BEGIN
                _pilih_baru:=_pilih_lama;
                _pilihan:=_pilih_lama;
            END;
        esckey  : begin refresh; EXIT end;
        ELSE beep;
    END;

    normal1;gotoxy(2,_pilih_lama);
    if pilihan = 1 then write(data_sub_menu1[_pilih_lama])
    else write(data_sub_menu2[_pilih_lama]);
    highlight1;gotoxy(2,_pilih_baru);
    if pilihan = 1 then write(data_sub_menu1[_pilih_baru])
    else write(data_sub_menu2[_pilih_baru]);
    _pilih_lama:=_pilih_baru;
    UNTIL (ch=enterkey) OR (ch in [ord('1'),ord('2')]);
    my_window(data_menu[pilihan],40,13,70,17,lightgray,black,true);
    revert1;gotoxy(2,_pilih_baru);
    if pilihan = 1 then write(data_sub_menu1[_pilih_baru])
    else write(data_sub_menu2[_pilih_baru]);
    for i:=1 to 2 do begin
    if i<>_pilih_baru then begin
    gotoxy(2,i);revert;
    if pilihan = 1 then write(data_sub_menu1[i])
    else write(data_sub_menu2[i]); end;end;
    if pilihan = 1 then In_Data(_PILIHAN)
    else setup(_PILIHAN);
    UNTIL FALSE;
END;

BEGIN
    scrnum:=0;
    cursmode(hidden);
    clean;show;mode:=true;
    datpilsetup[1]:=0;datpilsetup[2]:=$300;
    initppi;
    beban:=0;
```

```
    repeat ch:=getkey until ch=enterkey;
    refresh;
    mess1(' FMIPA/FISIKA ');
    my_window(' KONTROL MOTOR DC ',25,8,47,16,lightgray,black,true);

REPEAT
    highlight;
    gotoxy(2,2);write(data_menu_utama[1]);
    normal;
    FOR i:=2 TO jum_menu_utama DO
    BEGIN
       gotoxy(2,i+1);write(data_menu_utama[i])
    END;
    pilih_lama:=1;
    pilih_baru:=1;
    REPEAT
    message(' Pilih dengan Nomer, '+chr(24)+', '+chr(25)+', atau Esc dan tekan
              Enter',white+red*16);
    ch:=getkey;
    CASE ch OF
       ord('1') :
             BEGIN
                pilih_baru:=1;
                pilihan :=1;
             END;
       ord('2') :
             BEGIN
                pilih_baru:=2;
                pilihan :=2;
             END;
       ord('3') :
             BEGIN
                pilih_baru:=3;
                pilihan :=3;
             END;
       ord('4') :
             BEGIN
                pilih_baru:=4;
                pilihan :=4;
             END;
```

```
    upkey :
        BEGIN
          IF pilih_lama=1 THEN pilih_baru:=jum_menu_utama
          ELSE pilih_baru:=pilih_lama-1;
        END;
    downkey :
         BEGIN
           IF pilih_lama=jum_menu_utama THEN pilih_baru:=1
           ELSE pilih_baru:=pilih_lama+1;
         END;
    enterkey :
          BEGIN
            pilih_baru:=pilih_lama;
            pilihan:=pilih_lama;
          END;
    esckey,ord('5') :

  BEGIN
       message('Anda yakin [Y/T] ?',white+red*16);
       repeat ch:= getkey;
       if not (ch IN [esckey,ord('Y'),ord('y'),ord('T'),ord('t')]) then beep;
       until ch IN [esckey,ord('Y'),ord('y'),ord('T'),ord('t')];
       IF ch in [ord('Y'),ord('y')] THEN
       BEGIN
            cursmode(big);
            window(1,1,80,25);
            textcolor(7);
            textbackground(BLACK);
            clrscr;
            halt;
       END;

        ELSE beep;
 END;
normal;gotoxy(2,pilih_lama+1);write(data_menu_utama[pilih_lama]);
highlight;gotoxy(2,pilih_baru+1);write(data_menu_utama[pilih_baru]);
pilih_lama:=pilih_baru;

UNTIL (ch=enterkey) OR (ch in [ord('1'),ord('2'),ord('3'),ord('4')]);
```

```
revert1;gotoxy(2,pilih_baru+1);write(data_menu_utama[pilih_baru]);

FOR i:=1 TO jum_menu_utama DO
BEGIN
   IF i <> pilih_baru THEN
   BEGIN
      gotoxy(2,i+1);revert;write(data_menu_utama[i])
   END
END;

IF pilihan=jum_menu_utama THEN
begin
   message('Anda yakin [Y/T] ?',white+red*16);
   repeat ch:= getkey;
   if not (ch IN [ord('Y'),ord('y'),ord('T'),ord('t')]) then beep;
   until ch IN [ord('Y'),ord('y'),ord('T'),ord('t')];
   IF ch in [ord('Y'),ord('y')] THEN begin
   cursmode(big);
   window(1,1,80,25);
   textcolor(7);
   textbackground(BLACK);
   clrscr;
   halt;
   end;
end;
if (pilihan = 1)OR(pilihan = 2) then sub_menu2(pilihan);
if pilihan = 3 then run;
if pilihan = 4 then motor_char;

UNTIL FALSE;
END.
```

```
TYPE
 string30 = STRING[30];
 pixel        = RECORD
                   kar : CHAR;
                   atr : BYTE;
                END;
 bufscreen          = ARRAY[1..27,1..80] OF pixel;
 buffer       = ARRAY[1..4100 div 2] OF WORD;
 Ptrbuffer    = ^buffer;
 Cursor       = (norm,big,hidden);
 curstype   = RECORD
                   xcurs,ycurs,firstcurs,lastscurs :BYTE;
                END;
 scrtype  = RECORD
                ptrscr  :ptrbuffer;
                cursrec  :curstype;
                actatr     :BYTE;
                ScrMin,
                ScrMax  :WORD;
                END;
VAR
 ScrNum   :BYTE;
 ScrDat   :ARRAY[0..9] OF scrtype;
 Screen   :bufscreen ABSOLUTE $B800 : 0;
 regs     :REGISTERS;

PROCEDURE message(mess:STRING;color:BYTE);
BEGIN
   FOR i:=1 TO LENGTH(mess) DO
   BEGIN
      MEM[$b800:3840+2*(i-1)]:=ORD(mess[i]);
      MEM[$b800:3840+2*(i-1)+1]:=color;
   END;

   FOR i:=LENGTH(mess) TO 79 DO
   BEGIN
      MEM[$b800:3840+2*i]:=ORD(' ');
      MEM[$b800:3840+2*i+1]:=color
   END;
END;
```

```
PROCEDURE mess1(mess:string);
BEGIN
    gotoxy(80-length(mess),1);textbackground(blue);
    textcolor(white+blink);write(mess);
END;

PROCEDURE savescr(x1,y1,x2,y2:BYTE;VAR ScrPtr:ptrbuffer);
VAR
i,j,index,segmen,ofset:WORD;

BEGIN
IF LASTMODE =MONO THEN segmen:=$B000 ELSE segmen :=$B800;
 ScrPtr^[1] :=SWAP (y1) + x1;
 ScrPtr^[2] :=SWAP (y2) + x2;
 index:=3;

 FOR i:= y1 TO y2+1 DO
   BEGIN
      ofset :=(i-1)*160+2*(x1-1);
      FOR j:=x1 TO x2+2 DO
      BEGIN
        ScrPtr^[index]:=MEMW[segmen:ofset];
         INC(index);INC(ofset,2);
      END;
   END;
END;

PROCEDURE retscr(VAR ScrPtr:ptrbuffer);
VAR
i,j,index,segmen,ofset,x1,y1,x2,y2 :WORD;

BEGIN
   IF LASTMODE = MONO THEN segmen :=$B000 ELSE segmen :=$B800;
   x1 := LO (ScrPtr^[1]);
   y1 := HI (ScrPtr^[1]);
   x2 := LO (ScrPtr^[2]);
   y2 := HI (ScrPtr^[2]);
   index:=3;
   FOR i:= y1 TO y2+1 DO
   BEGIN
       ofset:=(i-1) * 160 + 2 * (x1-1);
```

```
      FOR j :=x1 TO x2+2 DO
      BEGIN
         MEMW[segmen:ofset] := ScrPtr^[index];
          INC(index);INC(ofset,2);
      END;
   END;
END;

PROCEDURE Cursize(firstcurs,lastscurs :BYTE);
VAR regs:registers;

BEGIN
   regs.AH :=1;
   regs.CH :=firstcurs;
   regs.CL :=lastscurs;
   INTR ($10,regs);
END;

PROCEDURE Cursmode(mode :Cursor);
CONST
  CData :ARRAY[BOOLEAN,Cursor] OF
          RECORD
           first,lasts :BYTE
          END
          =(((first: 6; lasts: 7), {normal,color 80x25}
           (first: 1; lasts: 7), { besar,color 80x25}
           (first:17; lasts:16)), { sembunyi,color 80x25}
           ((first:12; lasts:13),  { normal,  MONO }
            (first: 0; lasts:13),  { besar ,  MONO }
            (first:32; lasts:32)));{ sembunyi,MONO }

VAR
  videomono  :BOOLEAN;
  firstcurs,lastscurs :BYTE;

BEGIN
  videomono:=(LASTMODE=MONO);
  firstcurs:=CData[videomono,mode].first;
  lastscurs:=CData[videomono,mode].lasts;
  Cursize(firstcurs,lastscurs);
END;
```

```
PROCEDURE Cursinf(VAR CData :curstype);
CONST halaman =0;
VAR
  regs:REGISTERS;

BEGIN
   regs.AH :=3;
   regs.BH := halaman;
   INTR($10,regs);
   WITH CData DO
    BEGIN
       firstcurs :=regs.CH;
       lastscurs:=regs.CL;
      xcurs :=WHEREx;
      ycurs :=WHEREy;
    END;
END;

PROCEDURE CLEAN;
VAR    i :integer;

BEGIN
    window(1,1,80,25);
    textattr:=7;
    clrscr;
    for i:=0 to ((2*80*25)-1) do
      if i mod 2 = 0 then
        mem[$b800:i]:=177
END;

PROCEDURE Beep;
{ Produces a low beep on the speaker }

BEGIN
 SOUND(500);
 DELAY(70);
 NOSOUND;
END; { Beep }
```

```
PROCEDURE revert;
BEGIN
    TEXTBACKGROUND(lightgray);
    TEXTCOLOR(darkgray);
END;

PROCEDURE revert1;
BEGIN
    textcolor(white);textbackground(darkgray);
END;

PROCEDURE normal;
BEGIN
    TEXTBACKGROUND(lightgray);
    TEXTCOLOR(black);
END;

PROCEDURE normal1;
BEGIN
    TEXTBACKGROUND(BLUE);
    TEXTCOLOR(WHITE);
END;


PROCEDURE highlight;
BEGIN
    TEXTBACKGROUND(green);TEXTCOLOR(BLACK);
END;

PROCEDURE highlight1;
BEGIN
    TEXTBACKGROUND(MAGENTA);TEXTCOLOR(WHITE);
END;

FUNCTION GetKey : WORD;
{ ambil nilai keyboard yang ditekan }
VAR
   Ch : CHAR;

BEGIN
  ch:=READKEY;
```

```
 IF Ch = #0 THEN                    { Extended character }
   GetKey := WORD(ord(READKEY)) SHL 8
 ELSE
   GetKey := ord(Ch);               { Normal character }
END; { GetKey }

PROCEDURE READSTR(VAR masuk1 : string; var EscPressed : Boolean; maxlength:byte);
{ menginput string dari layar }
VAR
 Pos,i,j,n: Byte;
 Key : Word;
 strhasil : string;
 Selesai : Boolean;
 salah,masuk:integer;
 sum :array[1..6] of integer;

BEGIN
    pos:=0;
    selesai:=false;strhasil:='$';
    EscPressed := False;
    masuk:=0;
    REPEAT
   Key := GetKey;
   case Key of
    Ord(' ')..Ord('') :
     if (LENGTH(strhasil)<=maxlength) AND
      (key in [ord('-'),ord('.'),ord('0')..ord('9'),ord('a')..ord('f'),ord('A')..ord('F')]) THEN
     begin
       Inc(Pos);
       strhasil:=strhasil+Chr(Key);
       write(chr(key));
     end else beep;
    EscKey : begin
               Selesai := True;
               EscPressed := True;
          end;
    EnterKey : begin
      Selesai := True;
      EscPressed := False;
    end;
    BackSpaceKey :
```

```
        IF Pos <> 0 THEN
        BEGIN
         DELETE(StrHasil, Pos, 1);
         GOTOXY(WHEREX - 1, WHEREY);
         WRITE(COPY(StrHasil, Pos, LENGTH(Strhasil) - Pos + 1) + ' ');
         GOTOXY(WHEREX - LENGTH(Strhasil) + Pos - 2, WHEREY);
         DEC(Pos);
        END
    END;
    UNTIL selesai or escpressed;

   masuk1:=strhasil;
END;

PROCEDURE READINT(VAR x : integer; var EscPressed : Boolean; maxlength:byte);
{ menginput string dari layar }
VAR
 Pos,i,j,n: Byte;
 Key : Word;
 strhasil : string;
 Selesai : Boolean;
 salah:integer;

BEGIN
   pos:=0;
   selesai:=false;strhasil:='';
   EscPressed := False;
   REPEAT
   Key := GetKey;
   case Key of
    Ord(' ')..Ord('') :
     if (LENGTH(strhasil)<=maxlength) AND (key in [ord('-'),ord('0')..ord('9')]) THEN
     BEGIN
        Inc(Pos);
        strhasil:=strhasil+Chr(Key);
        write(chr(key));
     END ELSE beep;
     EscKey : BEGIN
                Selesai := True;
                EscPressed := True;
              END;
```

```
    EnterKey : BEGIN
                    Selesai := True;
                     EscPressed := False;
                END;
    BackSpaceKey :
        IF Pos <> 0 THEN
        BEGIN
          DELETE(StrHasil, Pos, 1);
          GOTOXY(WHEREX - 1, WHEREY);
          WRITE(COPY(StrHasil, Pos, LENGTH(Strhasil) - Pos + 1) + ' ');
          GOTOXY(WHEREX - LENGTH(Strhasil) + Pos - 2, WHEREY);
          DEC(Pos);
        END
    END;
    IF strhasil='' then selesai:=false

   UNTIL Selesai or escpressed;
   VAL(strhasil,x,salah);
END;

PROCEDURE frame(UpTitle:STRING;x1,y1,x2,y2:BYTE;back,fore:byte);
CONST
  data :ARRAY[0..7] OF CHAR
        :=(#218,#191,#192,#217,#179,#179,#196,#196);
VAR
 i,segmen :WORD;

BEGIN
      IF (x1<1) OR (x2>80) OR (y1<1) OR (y2>25) OR ((x2-x1)<1) OR ((y2-y1)<1) THEN
      EXIT;
      TEXTBACKGROUND(back);
      window(1,1,80,25);textcolor(fore);
      gotoxy(x1,y1);write(data[0]);
      FOR i:= x1+1 TO x2-1 DO write (data[6]);
      gotoxy(x2,y1);write(data[1]);
      FOR i:=y1+1 TO y2-1 DO
       BEGIN
            gotoxy(x1,i);write(data[4]);
            gotoxy(x2,i);write(data[5]);
        END;
       gotoxy(x1,y2);write(data[2]);
```

```
FOR i:=x1+1 TO x2-1 DO write(data[7]);
IF (x2=80) AND(y2=25) THEN
  BEGIN
    IF LASTMODE=MONO THEN segmen :=$B000 ELSE segmen :=+$B800;
    MEM[segmen:3998] := ord(data[3]);
  END

ELSE
  BEGIN
   gotoxy(x2,y2);
   write(data[3]);
  END;

 {membuat UpTitle}

 IF LENGTH(UpTitle) >(x2-x1+1) THEN
 UpTitle:= COPY(UpTitle,1,x2-x1+1);
 gotoxy(x1+(x2-x1+1-LENGTH(UpTitle)) div 2,y1);
 write(UpTitle);
 window(X1+1,Y1+1,x2,Y2);
END;

PROCEDURE my_window(UpTitle:STRING;x1,y1,x2,y2:BYTE;
                back,fore:BYTE;shade:BOOLEAN);
BEGIN
 IF ScrNum>(maxscr-1) THEN EXIT;
 IF (x1<1) OR (x2>80) OR ((x2-x1)<3) OR ((y2-y1)<3) THEN
 BEGIN
    WINDOW(x1,y1,x2,y2);
    EXIT
 END;
 WITH ScrDat[ScrNum] DO
  BEGIN
  IF maxavail >=totalmem THEN NEW(ptrscr)
  ELSE
    BEGIN
     WRITELN('memori tak cukup.');halt(1);
    END;

  savescr(x1,y1,x2,y2,ptrscr);
  Cursinf(cursrec);
```

```
  actatr:=TEXTATTR;
  ScrMin:=WINDMIN;
  ScrMax:=WINDMAX;
  WINDOW(1,1,80,25);
  frame(UpTitle,x1,y1,x2,y2,back,fore);
  IF shade THEN
  BEGIN
     FOR i:=x1+1 TO x2+2 DO
        Screen[y2+1,i].atr:=Screen[y2+1,i].atr shr 4;
     FOR i:=y1+1 TO y2 DO
     BEGIN
          Screen[i,x2+1].atr:=Screen[i,x2+1].atr shr 4;
          Screen[i,x2+2].atr:=Screen[i,x2+2].atr shr 4;
     END
  END;
  WINDOW(x1+1,y1+1,x2-1,y2-1);
  TEXTBACKGROUND(back);
  CLRSCR;
  END;
INC(ScrNum);
END;

PROCEDURE refresh;
BEGIN
IF ScrNum=0 THEN EXIT;
DEC(ScrNum);
WITH ScrDat[ScrNum],cursrec DO
   BEGIN
    retscr(ptrscr);
    WINDOW(LO(ScrMin)+1,HI(ScrMin)+1,
        LO(ScrMax)+1,HI(ScrMax)+1);
    TEXTATTR:=actatr;
    Cursize(firstcurs,lastscurs);
    GOTOXY(xcurs,ycurs+1);
    DISPOSE(ptrscr);
   END;
END;
```

```
FUNCTION reserve(V:byte):byte;
const
   real_data : array [1..256] of byte
         =($00,$01,$02,$03,$04,$05,$06,$07,$08,$09,$0A,$0B,$0C,$0D,$0E,$1F,
          $10,$11,$12,$13,$14,$15,$16,$17,$18,$19,$1A,$1B,$1C,$1D,$1E,$0F,
          $20,$21,$22,$23,$24,$25,$26,$27,$28,$29,$2A,$2B,$2C,$2D,$2E,$2F,
          $30,$31,$32,$33,$34,$35,$36,$37,$38,$39,$3A,$3B,$3C,$3D,$3E,$3F,
          $40,$41,$42,$43,$44,$45,$46,$47,$48,$49,$4A,$4B,$4C,$4D,$4E,$4F,
          $50,$51,$52,$53,$54,$55,$56,$57,$58,$59,$5A,$5B,$5C,$5D,$5E,$5F,
          $60,$61,$62,$63,$64,$65,$66,$67,$68,$69,$6A,$6B,$6C,$6D,$6E,$6F,
          $70,$71,$72,$73,$74,$75,$76,$77,$78,$79,$7A,$7B,$7C,$7D,$7E,$7F,
          $80,$81,$82,$83,$84,$85,$86,$87,$88,$89,$8A,$8B,$8C,$8D,$8E,$8F,
          $90,$91,$92,$93,$94,$95,$96,$97,$98,$99,$9A,$9B,$9C,$9D,$9E,$9F,
          $A0,$A1,$A2,$A3,$A4,$A5,$A6,$A7,$A8,$A9,$AA,$AB,$AC,$AD,$AE,$AF,
          $B0,$B1,$B2,$B3,$B4,$B5,$B6,$B7,$B8,$B9,$BA,$BB,$BC,$BD,$BE,$BF,
          $C0,$C1,$C2,$C3,$C4,$C5,$C6,$C7,$C8,$C9,$CA,$CB,$CC,$CD,$CE,$CF,
          $D0,$D1,$D2,$D3,$D4,$D5,$D6,$D7,$D8,$D9,$DA,$DB,$DC,$DD,$DE,$DF,
          $E0,$E1,$E2,$E3,$E4,$E5,$E6,$E7,$E8,$E9,$EA,$EB,$EC,$ED,$EE,$EF,
          $F0,$F1,$F2,$F3,$F4,$F5,$F6,$F7,$F8,$F9,$FA,$FB,$FC,$FD,$FE,$FF);

   Res_data  : array [1..256] of byte
         =($00,$80,$40,$C0,$20,$A0,$60,$E0,$10,$90,$50,$D0,$30,$B0,$70,$F0,
          $08,$88,$48,$C8,$28,$A8,$68,$E8,$18,$98,$58,$D8,$38,$B8,$78,$F8,
          $04,$84,$44,$C4,$24,$A4,$64,$E4,$14,$94,$54,$D4,$34,$B4,$74,$F4,
          $0C,$8C,$4C,$CC,$2C,$AC,$6C,$EC,$1C,$9C,$5C,$DC,$3C,$BC,$7C,$FC,
          $02,$82,$42,$C2,$22,$A2,$62,$E2,$12,$92,$52,$D2,$32,$B2,$72,$F2,
          $0A,$8A,$4A,$CA,$2A,$AA,$6A,$EA,$1A,$9A,$5A,$DA,$3A,$BA,$7A,$FA,
          $06,$86,$46,$C6,$26,$A6,$66,$E6,$16,$96,$56,$D6,$36,$B6,$76,$F6,
          $0E,$8E,$4E,$CE,$2E,$AE,$6E,$EE,$1E,$9E,$5E,$DE,$3E,$BE,$7E,$FE,
          $01,$81,$41,$C1,$21,$A1,$61,$E1,$11,$91,$51,$D1,$31,$B1,$71,$F1,
          $09,$89,$49,$C9,$29,$A9,$69,$E9,$19,$99,$59,$D9,$39,$B9,$79,$F9,
          $05,$85,$45,$C5,$25,$A5,$65,$E5,$15,$95,$55,$D5,$35,$B5,$75,$F5,
          $0D,$8D,$4D,$CD,$2D,$AD,$6D,$ED,$1D,$9D,$5D,$DD,$3D,$BD,$7D,$FD,
          $03,$83,$43,$C3,$23,$A3,$63,$E3,$13,$93,$53,$D3,$33,$B3,$73,$F3,
          $0B,$8B,$4B,$CB,$2B,$AB,$6B,$EB,$1B,$9B,$5B,$DB,$3B,$BB,$7B,$FB,
          $07,$87,$47,$C7,$27,$A7,$67,$E7,$17,$97,$57,$D7,$37,$B7,$77,$F7,
          $0F,$8F,$4F,$CF,$2F,$AF,$6F,$EF,$1F,$9F,$5F,$DF,$3F,$BF,$7F,$FF);

 var l:integer;
```

```
BEGIN
    for l:=1 to 256 do
    BEGIN
        if real_data[l] = V then
        begin reserve:=res_data[l];exit;end
    END
END;

PROCEDURE show;

BEGIN
    my_window('',20,8,65,16,BLUE,white,TRUE);
    gotoxy(1,2);WRITE('      PENGONTROL KECEPATAN MOTOR DC      ');
    gotoxy(1,4);WRITE('      NAMA    : ERIEKE Y ANTONY      ');
    gotoxy(1,5);WRITE('      NIM     : J 401 90 0476        ');
    gotoxy(1,6);WRITE('      JURUSAN : F I S I K A          ');
    gotoxy(1,7);WRITE('      FAKULTAS : M I P A             ');
END;
```

```
VAR
   pA,pB,pC,pR  : word;
   karakter,
   sign         : char;
   Hasil_GraphicP,Hasil_Asli                  : real;
   Saklar,Choice,Naik_OG, Naik_G, DelayO      : Integer;
   Hasil_ZCD                                  : byte;

FUNCTION des2bin(des:word) : string;
VAR temp,i : integer;
   hasil,hasil0 : string;
BEGIN
    hasil := '';
    for i := 1 to 8 do
    BEGIN
    temp := des mod 2;
    des := des div 2;
    str(temp,hasil0);
    hasil := hasil0+hasil;
    END;
    des2bin := hasil;
END;

FUNCTION des2hex(des:word) : string;
CONST
    dataHex  = '0123456789ABCDEF';
VAR temp,i : integer;
   hasil,hasil0 : string;
BEGIN
   hasil := '';
   REPEAT
   temp := des mod 16;
   des := des div 16;
   hasil0 := copy(dataHex,temp+1,1);
   hasil := hasil0+hasil;
   until des=0;
   des2hex := hasil;
END;

PROCEDURE initPPI;
BEGIN
    pA := $300;
    pB := pA+1;
    pC := pA+2;
    pR := pA+3;
```

```
    port[pR] := $81;          {initial PPI}
END;

PROCEDURE In_RPM;
BEGIN
  esc:=false;
  my_window('',33,6,67,9,BLUE,WHITE,FALSE);
  gotoxy(2,7);write(' Kecepatan Yang Diinginkan :');
  cursmode(big);
  gotoxy(30,7);readint(RPM,esc,3);
  if esc then exit;
END;

PROCEDURE In_Beban;
BEGIN
  esc:=false;
  my_window('',33,6,58,9,BLUE,WHITE,FALSE);
  gotoxy(2,7);write(' Beban Pada Motor :');
  cursmode(big);
  gotoxy(21,7);readint(beban,esc,3);
  if esc then exit;
END;

PROCEDURE test_alat;

VAR Pulsa              : integer;
   Jam,Mnt,Dtk,Sdtk1   : word;
   PC_new,PC_old       : byte;
   Time                : real;
   title               : string;
BEGIN
    message('  Check Modul ... !!!',white+red*16+blink);
    Port[pA] := reserve($80);        {kirim tegangan ke DAC
                                     setengah tegangan maksimum}
    My_window('',15,6,60,10,blue,white,true);
    title:=' Alat Sedang di Coba, Tunggu sebentar !.... ';
    gotoxy(1,1);write(title);
    Pulsa := 0;PC_old:=$00;
    Settime(0,0,0,0);
    REPEAT
        PC_new := Port[pC];
        If (PC_new Xor PC_old)= 1 then
        BEGIN
           Inc(Pulsa);PC_old:=PC_new;
        END;
```

```
        Gettime(Jam,Mnt,Dtk,Sdtk1);
    Until DTK >= 2;
    if pulsa<>0 then
     BEGIN
       gotoxy(15,3);textcolor(white+blink);write('Alat OK !!');
     END;
    else
    BEGIN
       gotoxy(10,3);textcolor(white+blink);write('Alat Tidak bekerja !!');
     END;
    message('  Check Modul Finished !!!',white+red*16);
    REPEAT until keypressed;refresh;
END;

PROCEDURE alamat;
VAR i : integer;
    bacarata : word;
    masuk,kode : integer;
    baca,masuk1  :string;
    ch : char;
BEGIN
   esc:=false;
   my_window('',33,6,53,9,BLUE,WHITE,FALSE);
   gotoxy(2,7);write(' Alamat PPI : $');
   gotoxy(16,7);writeln(des2hex(datpilsetup[2]));
   gotoxy(2,8);write(' Alamat baru: $');
    cursmode(big);
    gotoxy(16,8);readstr(masuk1,esc,3);
    if esc then exit;
   val(masuk1,masuk,kode);
   datpilsetup[2]:=masuk;
   initppi;
   cursmode(hidden);
END;

PROCEDURE Get_RPM;
VAR Pulsa,j            : integer;
    Jam,Mnt,Dtk,Sdtk1   : word;
    PC_old,PC_new,V_out : byte;
    Time               : real;
BEGIN
     Pulsa := 0;
     Settime(0,0,0,0);
     REPEAT
         PC_new := Port[pC];
```

```
        PC_new:=PC_new and $01;
        PC_old:=PC_old and $01;
        If (PC_old Xor PC_new) = 1 then
        BEGIN
            Inc(Pulsa);
            PC_old:=PC_new;
         END; {menghitung banyaknya pinggiran turun dan   naik}
        gettime(Jam,mnt,dtk,sdtk1);
    Until sdtk1>50;
    RPM_In  := round(15*Pulsa);
END;

PROCEDURE run; {start konversi}
VAR Pulsa, X,j,k        : integer;
   Jam,Mnt,Dtk,Sdtk1    : word;
   PC_old,PC_new,V_out,
   PC_Out               : byte;
   Time,tegangan        : real;
   title                : string;
   Control              : boolean;
   RPM_bef :array[2..4000] of integer;
BEGIN
    If RPM = 0 then V_out := $00;
    If (RPM >0) and (RPM<= 300) then V_out := $3C;
    If (RPM > 300) and (RPM <= 600) then V_out := $48;
    If (RPM >600) and (RPM <= 900) then V_out := $51;
    If (RPM >900) and (RPM <= 1200) then V_out := $56;
    If (RPM >1200) and (RPM <= 1500) then V_out := $5A;
    If (RPM >1500) and (RPM <= 1800) then V_out := $62;
    If (RPM >1800) and (RPM <= 2100) then V_out := $68;
    If (RPM >2100) and (RPM <= 2400) then V_out := $6B;
    If (RPM >2400) and (RPM <= 2700) then V_out := $72;
    if RPM > 2700 then
       BEGIN
         message('Kecepatan Melebihi Maximum',red+white+blink);
         exit;
       END;
    X := 30;j:=2;k:=0;PC_old:=$00;
    My_window('',15,1,65,24,blue,white,true);
    title:=' No.   KECEPATAN      TEGANGAN MOTOR      BEBAN    ';
    gotoxy(1,1);write(title);
    REPEAT
        Control := true;
        V_outs:=reserve(V_out);
        Port[pA] := V_outs;         {kirim tegangan ke DAC}
```

```
        Tegangan := (V_out/$100)*10;
        inc(j);
        get_rpm;  {ambil kecepatan motor}
        RPM_bef[j]:=RPM_in;
        if j>2 then
           BEGIN
              while (RPM_bef[j-1]-RPM_bef[j]) > 40 do
                 BEGIN
                    port[pC]:=reserve(V_out);
                    get_rpm;RPM_bef[j]:=RPM_in;
                 END;
           END;
        If RPM < (RPM_In-X) then
        BEGIN
           V_out := V_out-$01;
           Control := false;
        END;
        If RPM > (RPM_in+X) then
        BEGIN
           V_out := V_out+$01;
           Control := false;
        END;
        gotoxy(1,j);Write(' ',j-2+(k*20):3,RPM_In:10,tegangan:16:1,Beban:14);
        if (j mod 23) = 0 THEN
        BEGIN
           inc(k);j:=2;clrscr;gotoxy(1,1);write(title);
        END;
        delay(500);
        until Control or keypressed;
      readln;refresh;
END;
```

```
PROCEDURE graph_init;
VAR
  grDriver : integer;
  grMode   : integer;
  grErrCod : integer;

BEGIN
    clrscr;
    grDriver := detect;
    {grMode :=CGAC1;}
    InitGraph(grDriver,grMode,'');
    grErrCod := GraphResult;
    if grErrCod <> grOk then
        WriteLn('Graphics error:', GraphErrorMsg(grErrCod))
    else
        restoreCRTMode;
END;

PROCEDURE layar;
VAR x,y,x1,y1,x2,y2,x3,y3:integer;
BEGIN
    x:=getmaxx;y:=getmaxy;
    setcolor(MAGENTA);
    settextstyle(defaultfont,horizdir,1);
    outtextxy(x*3 div 8,y div 16,'Motor Characteristic');
    settextstyle(smallfont,horizdir,6);
    outtextxy(x*9 div 64,y*3 div 16,' RPM  ');
    settextstyle(smallfont,horizdir,6);          {tampilan ac}
    outtextxy((x*13 div 16)+2,y*25 div 32,'V');
    rectangle(x div 8,y div 8,x*7 div 8,y*7 div 8);setcolor(white);
    x1:=x*2 div 8;x2:=x*3 div 16;x3:=x*13 div 16;
    y1:=y*3 div 16;y2:=y*13 div 16;y3:=y*6 div 8;
    line(x1,y1,x1,y2);
    line(x2,y3,x3,y3);
    settextstyle(smallfont,horizdir,4);
    outtextxy(x*23 div 128,y*2 div 8,'     ');
    outtextxy(x*23 div 128,y*3 div 8,'     ');
    outtextxy(x*15 div 64,y*33 div 64,' ');
    outtextxy(x*23 div 128,y*5 div 8,'     ');
    outtextxy(x*23 div 128,y*6 div 8,'     ');
END;

PROCEDURE motor_char; {start konversi}
VAR hasilbaca,ukuran : word;
   sam_dat : array [0..450] of word;
   status : array [0..450] of boolean;
```

```
  tandha : array [0..450] of char;
  y,x,yl,xl,i,x1,x2,x3,y1,y2,y3 : integer;
  k,k1:byte;
  free : pointer;
  esc:boolean;
BEGIN
   graph_init;
   SetGraphMode(GetGraphMode);
   layar;k:=$00;i:=0;
   yl:=getmaxy;xl:=getmaxx;
   x1:=xl*2 div 8;x2:=xl*3 div 16;x3:=xl*13 div 16;
   y1:=yl*3 div 16;y2:=yl*13 div 16;y3:=yl*6 div 8;
   moveto(x1,y3);
   setcolor(yellow);
   REPEAT
       inc(i);
       k:=k+$02;
       port[pA]:=reserve(k);
       get_rpm;
       y:=y3-round((RPM_in / 2700)*(y3-y1));
       lineto((i*3)+x1,y);
   until rpm_in > 2700;
   REPEAT until keypressed;
   restoreCrtMode;
   clean;
   mess1(' FMIPA/FISIKA ');
   my_window(' KONTROL MOTOR DC ',25,8,47,16,lightgray,black,true);
   cursmode(hidden);exit;
END;
```

LAMPIRAN B : DATA HASIL PERCOBAAN

LAMPIRAN DATA TEGANGAN AWAL MOTOR DC

- 0 < kecepatan <= 600 rpm tegangan awal motor DC adalah 1,89 volt
- 300 < kecepatan <= 900 rpm tegangan awal motor DC adalah 2,75 volt
- 900 < kecepatan <= 1200 rpm tegangan awal motor DC adalah 2,93 volt
- 1200 < kecepatan <= 1500 rpm tegangan awal motor DC adalah 3,02 volt
- 1500 < kecepatan <= 1800 rpm tegangan awal motor DC adalah 3,20 volt
- 1800 < kecepatan <= 2100 rpm tegangan awal motor DC adalah 3,41 volt
- 2100 < kecepatan <= 2400 rpm tegangan awal motor DC adalah 3,55 volt
- 2400 < kecepatan <= 2700 rpm tegangan awal motor DC adalah 3,77 volt

TABEL TEGANGAN KELUARAN D/AC DAN TEGANGAN MOTOR DC

TABEL TEGANGAN KELUARAN D/AC DAN TEGANGAN MOTOR DC

DATA (H)	TEGANGAN KELUARAN D/AC (V)	TEGANGAN MOTOR DC (V)
OO	0,00	0,00
O1	0,04	0,00
O2	0,08	0,00
O3	0,12	0,00
O4	0,16	0,00
O5	1,20	0,00
O6	0,23	0,00
O7	0,27	0,00
O8	0,31	0,00
O9	0,35	0,00
0A	0,39	0,00
0B	0,43	0,00
0C	0,47	0,00
0D	0,51	0,00
0F	0,55	0,01
0F	0,59	0,02
10	0,63	0,03
11	0,66	0,04
12	0,70	0,05
13	0,74	0,08
14	0,78	0,12
15	0,82	0,15
16	0,86	0,19
17	0,90	0,24
18	0,94	0,28
19	0,98	0,32
1A	1,02	0,36
1B	1,05	0,39
1C	1,09	0,44
1D	1,13	0,48
1E	1,17	0,52
1F	1,21	0,57

DATA (H)	TEGANGAN KELUARAN D/AC (V)	TEGANGAN MOTOR DC (V)
20	1,25	0,61
21	1,29	0,66
22	1,33	0,70
23	1,37	0,73
24	1,41	0,76
25	1,45	0,80
26	1,48	0,84
27	1,52	0,83
28	1,56	0,87
29	1,60	0,91
2A	1,64	0,94
2B	1,68	0,98
2C	1,72	1,03
2D	1,76	1,07
2E	1,80	1,11
2F	1,84	1,16
30	1,88	1,20
31	1,91	1,24
32	1,95	1,28
33	1,99	1,32
34	2,03	1,36
35	2,07	1,41
36	2,11	1,45
37	2,15	1,49
38	2,19	1,53
39	2,23	1,57
3A	2,27	1,61
3B	2,30	1,65
3C	2,34	1,68
3D	2,38	1,72
3E	2,42	1,76
3F	2,46	1,80

DATA (H)	TEGANGAN KELURAN D/AC (V)	TEGANGAN MOTOR DC (V)
40	2,50	1,84
41	2,54	1,89
42	2,58	1,93
43	2,62	1,96
44	2,66	2,00
45	2,70	2,03
46	2,73	2,07
47	2,77	2,11
48	2,81	2,15
49	2,85	2,18
4A	2,89	2,24
4B	2,93	2,28
4C	2,97	2,33
4D	3,01	2,37
4E	3,05	2,41
4F	3,09	2,44
50	3,13	2,49
51	3,16	2,54
52	3,20	2,58
53	3,24	2,61
54	3,28	2,65
55	3,32	2,70
56	3,36	2,75
57	3,40	2,80
58	3,44	2,84
59	3,48	2,89
5A	3,52	2,93
5B	3,55	2,96
5C	3,59	2,99
5D	3,63	3,02
5E	3,67	3,05
5F	3,71	3,08

DATA (H)	TEGANGAN KELUARAN D/AC (V)	TEGANGAN MOTOR DC (V)
60	3,75	3,11
61	3,79	3,14
62	3,83	3,17
63	3,87	3,20
64	3,91	3,23
65	3,95	3,26
66	3,98	3,30
67	4,02	3,33
68	4,06	3,37
69	4,10	3,41
6A	4,14	3,45
6B	4,18	3,48
6C	4,22	3,52
6D	4,26	3,55
6E	4,30	3,60
6F	4,34	3,64
70	4,38	3,68
71	4,41	3,73
72	4,45	3,77
73	4,49	3,80
74	4,53	3,84
75	4,57	3,87
76	4,61	3,91
77	4,65	3,95
78	4,69	3,99
79	4,73	4,03
7A	4,77	4,07
7B	4,80	4,11
7C	4,84	4,15
7D	4,88	4,19
7E	4,92	4,22
7F	4,96	4,26

DATA (H)	TEGANGAN KELUARAN D/AC (V)	TEGANGAN MOTOR DC (V)
80	5,00	4,29
81	5,04	4,32
82	5,08	4,36
83	5,12	4,40
84	5,16	4,44
85	5,20	4,48
86	5,23	4,52
87	5,27	4,56
88	5,31	4,60
89	5,35	4,64
8A	5,39	4,68
8B	5,43	4,72
8C	5,47	4,76
8D	5,51	4,80
8E	5,55	4,83
8F	5,59	4,87
90	5,63	4,91
91	5,66	4,95
92	5,70	4,99
93	5,74	5,13
94	5,78	5,16
95	5,82	5,19
96	5,86	5,23
97	5,90	5,27
98	5,94	5,31
99	5,98	5,35
9A	6,02	5,38
9B	6,05	5,41
9C	6,09	5,44
9D	6,13	5,47
9E	6,17	5,50
9F	6,21	5,53

DATA (V)	TEGANGAN KELUARAN D/AC (V)	TEGANGAN MOTOR DC (V)
A0	6,25	5,57
A1	6,29	5,60
A2	6,33	5,64
A3	6,37	5,68
A4	6,41	5,72
A5	6,45	5,76
A6	6,48	5,79
A7	6,52	5,83
A8	6,56	5,87
A9	6,60	5,90
AA	6,64	5,94
AB	6,68	5,98
AC	6,72	6,01
AD	6,76	6,05
AE	6,80	6,09
AF	6,84	6,14
B0	6,88	6,18
B1	6,91	6,21
B2	6,95	6,25
B3	6,99	6,28
B4	7,03	6,32
B5	7,07	6,36
B6	7,11	6,39
B7	7,15	6,43
B8	7,19	6,47
B9	7,23	6,51
BA	7,27	6,54
BB	7,30	6,57
BC	7,34	6,61
BD	7,38	6,64
BE	7,42	6,68
BF	7,46	6,72

DATA (V)	TEGANGAN KELUARAN D/AC (V)	TEGANGAN MOTOR DC (V)
C0	7,50	6,76
C1	7,54	6,78
C2	7,58	6,83
C3	7,62	6,88
C4	7,66	6,91
C5	7,70	6,94
C6	7,73	6,98
C7	7,77	7,03
C8	7,81	7,08
C9	7,85	7,12
CA	7,89	7,16
CB	7,93	7,20
CC	7,97	7,24
CD	8,01	7,28
CE	8,05	7,31
CF	8,09	7,35
D0	8,13	7,38
D1	8,16	7,42
D2	8,20	7,45
D3	8,24	7,49
D4	8,28	7,53
D5	8,32	7,57
D6	8,36	7,60
D7	8,40	7,64
D8	8,44	7,68
D9	8,48	7,71
DA	8,52	7,75
DB	8,55	7,79
DC	8,59	7,83
DD	8,63	7,87
DE	8,67	7,91
DF	8,71	7,95

DATA (V)	TEGANGAN KELUARAN D/AC (V)	TEGANGAN MOTOR DC (V)
E0	8,75	7,98
E1	8,79	8,02
E2	8,83	8,06
E3	8,87	8,10
E4	8,91	8,13
E5	8,95	8,17
E6	8,98	8,21
E7	9,02	8,25
E8	9,06	8,29
E9	9,10	8,33
EA	9,14	8,37
EB	9,18	8,40
EC	9,22	8,44
ED	9,26	8,48
EE	9,30	8,52
EF	9,34	8,56
F0	9,38	8,61
F1	9,41	8,65
F2	9,45	8,71
F3	9,49	8,76
F4	9,53	8,79
F5	9,57	8,82
F6	9,61	8,85
F7	9,65	8,88
F8	9,69	8,92
F9	9,73	8,95
FA	9,77	8,99
FB	9,80	9,03
FC	9,84	9,06
FD	9,88	9,10
FE	9,92	9,13
FF	9,96	9,16

TABEL KECEPATAN MOTOR DC

TABEL KECEPATAN MOTOR DC

Motor DC yang digunakan adalah motor DC 9 V dengan kecepatan optimal 2400 rpm

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
0,00	0
0,01	0
0,03	0
0,03	0
0,04	0
0,05	0
0,08	0
0,12	0
0,15	0
0,19	0
0,24	0
0,28	0
0,32	0
0,36	0
0,39	0
0,44	0
0,48	0
0,52	0
0,57	0
0,61	0
0,66	0
0,70	0
0,73	0
0,76	0
0,80	0
0,84	0
0,87	0
0,91	0
0,94	0
0,98	0

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
1,03	0
1,07	0
1,11	0
1,16	0
1,20	0
1,24	0
1,28	0
1,32	0
1,36	0
1,41	0
1,45	0
1,49	0
1,53	0
1,57	0
1,61	0
1,65	0
1,68	0
1,72	0
1,76	0
1,84	0
1,89	0
1,93	0
1,96	0
2,00	0
2,03	0
2,07	0
2,11	0
2,15	0
2,24	0
2,28	0

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
2,33	0
2,37	0
2,41	0
2,44	0
2,49	0
2,54	60
2,58	120
2,61	195
2,65	225
2,70	345
2,75	420
2,80	465
2,84	540
2,89	615
2,93	675
2,96	735
2,99	810
3,02	855
3,05	915
3,08	1005
3,11	1095
3,14	1155
3,17	1215
3,20	1250
3,23	1290
3,26	1340
3,30	1390
3,33	1440
3,37	1500
3,41	1575

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
3,45	1620
3,48	1680
3,52	1740
3,55	1800
3,60	1860
3,64	1915
3,68	1975
3,73	2040
3,77	2100
3,80	2145
3,84	2200
3,87	2250
3,91	2310
3,95	2355
3,99	2400
4,03	2460
4,07	2490
4,11	2520
4,15	2550
4,19	2580
4,22	2625
4,26	2640
4,29	2655
4,32	2670
4,36	2685
4,40	2700
4,44	2700
4,48	2700
4,52	2700
4,56	2700
4,60	2700
4,64	2700
4,68	2700
4,72	2700
4,76	2700
4,80	2700
4,83	2700
4,87	2700
4,91	2700

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
4,95	2715
4,99	2715
5,13	2715
5,16	2715
5,19	2715
5,23	2715
5,27	2715
5,31	2715
5,35	2715
5,38	2715
5,41	2715
5,47	2715
5,50	2715
5,53	2715
5,60	2715
5,64	2715
5,68	2715
5,72	2715
5,76	2715
5,79	2715
5,83	2715
5,87	2715
5,90	2715
5,94	2715
5,98	2715
6,01	2715
6,05	2715
6,09	2715
6,14	2715
6,18	2715
6,21	2715
6,25	2715
6,28	2715
6,36	2715
6,39	2715
6,43	2715
6,47	2715
6,51	2715
6,54	2715

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
6,57	2715
6,61	2715
6,64	2715
6,68	2715
6,72	2715
6,76	2715
6,78	2715
6,83	2715
6,88	2715
6,91	2715
6,94	2715
6,98	2730
7,03	2730
7,09	2730
7,12	2730
7,16	2730
7,20	2730
7,24	2730
7,28	2730
7,31	2730
7,35	2730
7,38	2730
7,42	2730
7,45	2730
7,49	2730
7,53	2730
7,57	2730
7,60	2730
7,64	2730
7,68	2730
7,71	2730
7,75	2730
7,79	2715
7,83	2715
7,87	2715
7,91	2715
7,95	2715
7,98	2715
8,02	2715

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
8,06	2715
8,10	2715
8,13	2715
8,17	2715
8,21	2715
8,25	2715
8,29	2715
8,33	2715
8,37	2715
8,40	2715

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
8,44	2715
8,48	2715
8,52	2715
8,56	2715
8,61	2715
8,65	2700
8,71	2700
8,75	2700
8,79	2700
8,82	2700

TEGANGAN MOTOR DC (V)	KECEPATAN (RPM)
8,85	2700
8,88	2700
8,92	2700
8,95	2700
8,99	2700
9,03	2700
9,06	2700
9,10	2700
9,13	2700
9,16	2700

LAMPIRAN C : LEMBAR DATA KOMPONEN

13. Data Transistor Daya

tipe	P_{maks} dalam W	U_{maks} dalam V	I_{maks} dalam A	penguatan arus dengan I_C = ... mA		U_{pengk} U_{CE} dengan I_C = V	mA	frekuensi transisi MHz	T dalam ndet	kemasan	jenis

tipe	P_{maks} dalam W	U_{maks} dalam V	I_{maks} dalam A	penguatan arus dengan I_C = ... mA		U_{pengk} U_{CE} dengan I_C = V	A	frekuensi transisi MHz	T dalam ndet	kemasan	jenis

I

II

III

IV

V

Permukaan logam pendingin dihubungkan pada kolektor

LF 351
Penguat Operasi Masukan JFET dengan Lebarjalur Lebar
(Wide Bandwidth JFET Input Operational Amplifiers)

Penjelasan Umum

LF 351 adalah sebuah penguat operasi masukan JFET yang murah dan berkecepatan tinggi dan dilengkapi tegangan gelincir masukan *(input-offset voltage)* yang distel di dalam. Teknologi TM (BI-FET II). Peranti ini memerlukan arus catu kecil, namun dapat mempertahankan hasilkali penguatan-lebarjalur yang besar disertai laju lantingan *(slew rate)* cepat. Tambahan pula, adanya peranti-peranti masukan JFET memberikan panjaran masukan dan juga arus-arus gelincir yang sangat rendah. LF 351 dilengkapi pena-pena seperti halnya LM 741 standar dan menerapkan rangkaian penapat tegangan gelincir yang sama pula. Sifat-sifat ini memungkinkan para perancang untuk sekaligus meng-*upgrade* kualitas kerja menyeluruh yang ada pada rancangan-rancangan LM 741 yang ada.

LF 351 dapat dipakai dalam penerapan-penerapan seperti misalnya: integrator kecepatan tinggi, konvertor D/A cepat, rangkaian cuplik-genggam, dan banyak rangkaian lain yang memerlukan tegangan gelincir masukan kecil, arus panjar masukan kecil, impedansi masukan tinggi, laju pelantingan yang tinggi, dan lebarjalur lebar.

Tarif Maksimum Mutlak

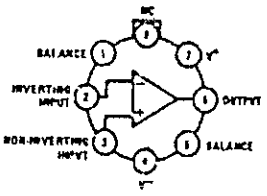
Tegangan Catu	±18 V
Borosan (disipasi) Daya (Catatan 1)	500 mW
Jangkah suhu operasi	0° C – +70° C
$T_{j(Maks)}$	115° C
Tegangan masukan diferensial	±30 V
Jangkah tegangan masukan (Catatan 2)	±15 V
Lama hubungsingkat keluaran	Terus-menerus
Jangkah suhu simpan	−65° C – +150° C
Suhu timah (Penyolderan 10 detik)	300° C

Sifat-sifat:

- Tegangan gelincir distel di dalam 2 mW
- Arus panjar masukan rendah 50 µA
- Tegangan desah masukan rendah 16 nV√Hz
- Arus desah masukan rendah 0,01 pA√Hz
- Lebarjalur penguatan lebar 4 MHz
- Laju lantingan tinggi 13 V/180det
- Arus catu rendah 1,8 mA
- Impedansi masukan tinggi $10^{12}\Omega$
- Cacat larasan total rendah $A_V=10$ $R_L=10$ k, $V_o=20\ V_{pp}$ BW=20Hz–20 kHz 0,02% (BW=*Bandwidth*)
- Sudut desah 1/f rendah 50 Hz
- Waktu endap *(settling time)* cepat hingga 0,01% 2 µdet

Diagram konekal

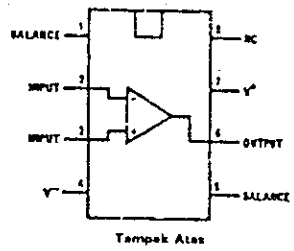
Kemasan Kaleng Logam



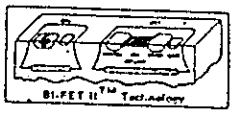
Tampak Atas

Cat. Pena 4 dihubungkan pd kaleng

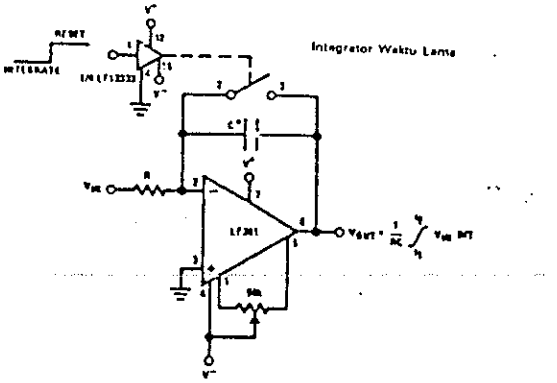
Kemasan Berbaris-berdua (DIL)



Tampak Atas

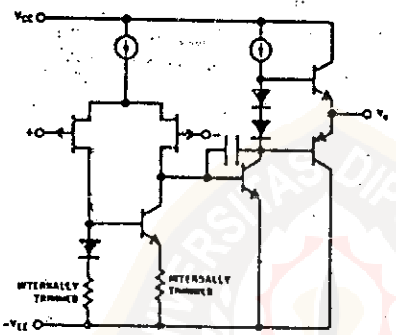


Peranti pun memiliki desah rendah dan hanyutan tegangan gelincir kecil, namun dalam peranti yang besaran-besaran ini kritis dianjurkan menggunakan LF 356. Kalau arus catu maksimum yang dipentingkan, maka LF 351 adalah pilihan yang lebih kena.



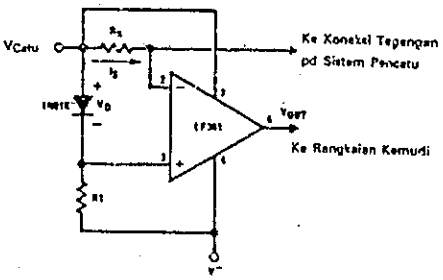
* Kondensator bocoran kecil
● Pot. 50 kΩ dipakai untuk pengatur kepekaan V_{os}

Skema disederhanakan



Penerapan lumrah

Indikator/Pemangkas Arus Catu



● V_{OUT} berguling ke tinggi, bila $R_{SIS} > V_D$

Karakteristik Elektrik DC (Catatan 3)

SYMBOL	PARAMETER	CONDITIONS	LF351A			LF351B			LF351			UNITS
			MIN	TYP	MAX	MIN	TYP	MAX	MIN	TYP	MAX	
V_{OS}	Input Offset Voltage	R_S = 10 kΩ, T_A = 25°C		1	2		3	5		5	10	mV
		Over Temperature			4			7			13	mV
$\Delta V_{OS}/\Delta T$	Average TC of Input Offset Voltage	R_S = 10 kΩ		10			10			10		pV/°C
I_{OS}	Input Offset Current	T_j = 25°C, (Notes 3, 4)		25	50		25	100		25	100	pA
		$T_j \leq$ 70°C			2			4			4	nA
I_B	Input Bias Current	T_j = 25°C, (Notes 3, 4)		50	100		50	200		50	200	pA
		$T_j \leq$ 70°C			4			8			8	nA
R_{IN}	Input Resistance	T_j = 25°C		10^{12}			10^{12}			10^{12}		Ω
AVOL	Large Signal Voltage Gain	V_S = ±15V, T_A = 25°C	50	100		50	100		25	100		V/mV
		V_O = ±10V, R_L = 2 kΩ										
		Over Temperature	25			25			15			V/mV
V_O	Output Voltage Swing	V_S = ±15V, R_L = 10 kΩ	±12	±13.5		±12	±13.5		±12	±13.5		V
V_{CM}	Input Common-Mode Voltage Range	V_S = ±15V	±11	+15 −12		±11	+15 −12		±11	+15 −12		V V
CMRR	Common-Mode Rejection Ratio	$R_S \leq$ 10 kΩ	80	100		80	100		70	100		dB
PSRR	Supply Voltage Rejection Ratio	(Note 5)	80	100		80	100		70	100		dB
I_D	Supply Current			1.8	2.8		1.8	2.8		1.8	3.4	mA

National Semiconductor Corporation

DAC0808, DAC0807, DAC0806 8-Bit D/A Converters

General Description

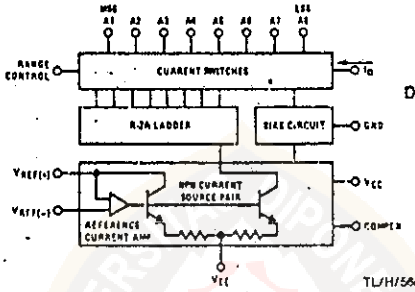
The DAC0808 series is an 8-bit monolithic digital-to-analog converter (DAC) featuring a full scale output current settling time of 150 ns while dissipating only 33 mW with ±5V supplies. No reference current (I_{REF}) trimming is required for most applications since the full scale output current is typically ±1 LSB of 255 I_{REF}/ 256. Relative accuracies of better than ±0.19% assure 8-bit monotonicity and linearity while zero level output current of less than 4 μA provides 8-bit zero accuracy for $I_{REF} \geq 2$ mA. The power supply currents of the DAC0808 series are independent of bit codes, and exhibits essentially constant device charactoristics over the entire supply voltage range.

The DAC0808 will interface directly with popular TTL, DTL or CMOS logic levels, and is a direct replacement for the MC1508/MC1408. For higher speed applications, see DAC0800 data sheet.

Features

- Relative accuracy: ±0.19% error maximum (DAC0808)
- Full scale current match: ±1 LSB typ
- 7 and 6-bit accuracy available (DAC0807, DAC0806)
- Fast settling time: 150 ns typ
- Noninverting digital inputs are TTL and CMOS compatible
- High speed multiplying input slew rate: 8 mA/μs
- Power supply voltage range: ±4.5V to ±18V
- Low power consumption: 33 mW @ ±5V

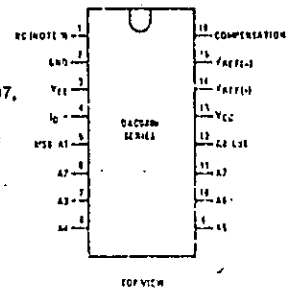
Block and Connection Diagrams



TL/H/5687–1

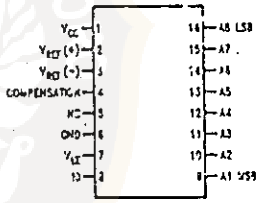
Order Number DAC0808, DAC0807, or DAC0806
See NS Package Number J16A, M16A or N16A

Dual-In-Line Package



TL/H/5687–2

Small-Outline Package



Top View

TL/H/5687–13

Ordering Information

ACCURACY	OPERATING TEMPERATURE RANGE	ORDER NUMBERS				
		J PACKAGE (J16A)*		N PACKAGE (N16A)*		SO PACKAGE (M16A)
8-bit	$-55°C \leq T_A \leq +125°C$	DAC0808LJ	MC1508L8			
8-bit	$0°C \leq T_A \leq +75°C$	DAC0808LCJ	MC1408L8	DAC0808LCN	MC1408P8	DAC0808LCM
7-bit	$0°C \leq T_A \leq +75°C$	DAC0807LCJ	MC1408L7	DAC0807LCN	MC1408P7	DAC0807LCM
6-bit	$0°C \leq T_A \leq +75°C$	DAC0806LCJ	MC1408L6	DAC0806LCN	MC1408P6	DAC0806LCM

*Note. Devices may be ordered by using either order number.

Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, contact the National Semiconductor Sales Office/ Distributors for availability and specifications.

Power Supply Voltage	
V_{CC}	+18 V_{DC}
V_{EE}	−18 V_{DC}
Digital Input Voltage, V5–V12	−10 V_{DC} to +18 V_{DC}
Applied Output Voltage, V_O	−11 V_{DC} to +18 V_{DC}
Reference Current, I_{14}	5 mA
Reference Amplifier Inputs, V14, V15	V_{CC}, V_{EE}
Power Dissipation (Note 3)	1000 mW
ESD Susceptibility (Note 4)	TBD
Storage Temperature Range	−65°C to +150°C
Lead Temp. (Soldering, 10 seconds)	
Dual-In-Line Package (Plastic)	260°C
Dual-In-Line Package (Ceramic)	300°C
Surface Mount Package	
Vapor Phase (60 seconds)	215°C
Infrared (15 seconds)	220°C

Operating Ratings

Temperature Range	$T_{MIN} \le T_A \le T_{MAX}$
DAC0808L	−55°C ≤ T_A ≤ +125°C
DAC0808LC Series	0 ≤ T_A ≤ +75°C

Electrical Characteristics

(V_{CC} = 5V, V_{EE} = −15 V_{DC}, $V_{REF}/R14$ = 2 mA, DAC0808: T_A = −55°C to +125°C, DAC0808C, DAC0807C, DAC0806C, T_A = 0°C to +75°C, and all digital inputs at high logic level unless otherwise noted.)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
E_r	Relative Accuracy (Error Relative to Full Scale I_O)	*(Figure 4)*				%
	DAC0808L (LM1508-8),				±0.19	%
	DAC0808LC (LM1408-8)					
	DAC0807LC (LM1408-7), (Note 5)				±0.39	%
	DAC0806LC (LM1408-6), (Note 5)				±0.78	%
	Settling Time to Within ½ LSB (Includes t_{PLH})	T_A = 25°C (Note 6), *(Figure 5)*		150		ns
t_{PLH}, t_{PHL}	Propagation Delay Time	T_A = 25°C, *(Figure 5)*		30	100	ns
TCI_O	Output Full Scale Current Drift			±20		ppm/°C
MSB	Digital Input Logic Levels	*(Figure 3)*				
V_{IH}	High Level, Logic "1"		2			V_{DC}
V_{IL}	Low Level, Logic "0"				0.8	V_{DC}
MSB	Digital Input Current	*(Figure 3)*				
	High Level	V_{IH} = 5V		0	0.040	mA
	Low Level	V_{IL} = 0.8V		−0.003	−0.8	mA
I_{15}	Reference Input Bias Current	*(Figure 3)*		−1	−3	µA
	Output Current Range	*(Figure 3)*				
		V_{EE} = −5V	0	2.0	2.1	mA
		V_{EE} = −15V, T_A = 25°C	0	2.0	4.2	mA
I_O	Output Current	V_{REF} = 2.000V, R14 = 1000Ω, *(Figure 3)*	1.9	1.99	2.1	mA
	Output Current, All Bits Low	*(Figure 3)*		0	4	µA
	Output Voltage Compliance (Note 2)	$E_r \le 0.19\%$, T_A = 25°C				
	V_{EE} = −5V, I_{REF} = 1 mA				−0.55, +0.4	V_{DC}
	V_{EE} Below −10V				−5.0, +0.4	V_{DC}

Electrical Characteristics (Continued)

(V_{CC} = 5V, V_{EE} = −15 V_{DC}, $V_{REF}/R14$ = 2 mA, DAC0808: T_A = −55°C to +125°C, DAC0808C, DAC0807C, DAC0806C, T_A = 0°C to +75°C, and all digital inputs at high logic level unless otherwise noted.)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
SRI_{REF}	Reference Current Slew Rate	*(Figure 6)*	4	8		mA/μs
	Output Current Power Supply Sensitivity	$-5V \le V_{EE} \le -16.5V$		0.05	2.7	μA/V
	Power Supply Current (All Bits Low)	*(Figure 3)*				
I_{CC}				2.3	22	mA
I_{EE}				−4.3	−13	mA
	Power Supply Voltage Range	T_A = 25°C, *(Figure 3)*				
V_{CC}			4.5	5.0	5.5	V_{DC}
V_{EE}			−4.5	−15	−16.5	V_{DC}
	Power Dissipation					
	All Bits Low	V_{CC} = 5V, V_{EE} = −5V		33	170	mW
		V_{CC} = 5V, V_{EE} = −15V		106	305	mW
	All Bits High	V_{CC} = 15V, V_{EE} = −5V		90		mW
		V_{CC} = 15V, V_{EE} = −15V		160		mW

Note 1: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications do not apply when operating the device beyond its specified operating conditions.

Note 2: Range control is not required.

Note 3: The maximum power dissipation must be derated at elevated temperatures and is dictated by T_{JMAX}, θ_{JA}, and the ambient temperature, T_A. The maximum allowable power dissipation at any temperature is $P_D = (T_{JMAX} - T_A)/\theta_{JA}$ or the number given in the Absolute Maximum Ratings, whichever is lower. For this device, T_{JMAX} = 125°C, and the typical junction-to-ambient thermal resistance of the dual-in-line J package when the board mounted is 100°C/W. For the dual-in-line N package, this number increases to 175°C/W and for the small outline M package this number is 100°C/W.

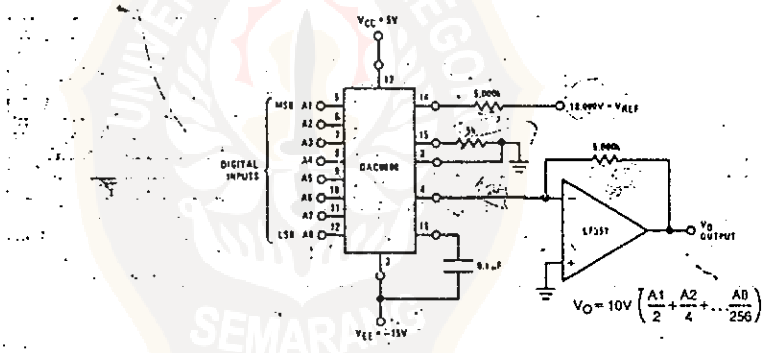
Note 4: Human body model, 100 pF discharged through a 1.5 kΩ resistor.

Note 5: All current switches are tested to guarantee at least 50% of rated current.

Note 6: All bits switched.

Note 7: Pin-out numbers for the DAL080X represent the dual-in-line package. The small outline package pinout differs from the dual-in-line package.

Typical Application



TL/H/5687–3

FIGURE 1. +10V Output Digital to Analog Converter (Note 7)

- MCS-85™ Compatible 8255A-5
- 24 Programmable I/O Pins
- Completely TTL Compatible
- Fully Compatible with Intel® Microprocessor Families
- Improved Timing Characteristics
- Direct Bit Set/Reset Capability Easing Control Application Interface
- 40-Pin Dual In-Line Package
- Reduces System Package Count
- Improved DC Driving Capability

The Intel® 8255A is a general purpose programmable I/O device designed for use with Intel® microprocessors. It has 24 I/O pins which may be individually programmed in 2 groups of 12 and used in 3 major modes of operation. In the first mode (MODE 0), each group of 12 I/O pins may be programmed in sets of 4 to be input or output. In MODE 1, the second mode, each group may be programmed to have 8 lines of input or output. Of the remaining 4 pins, 3 are used for handshaking and interrupt control signals. The third mode of operation (MODE 2) is a bidirectional bus mode which uses 8 lines for a bidirectional bus, and 5 lines, borrowing one from the other group, for handshaking.

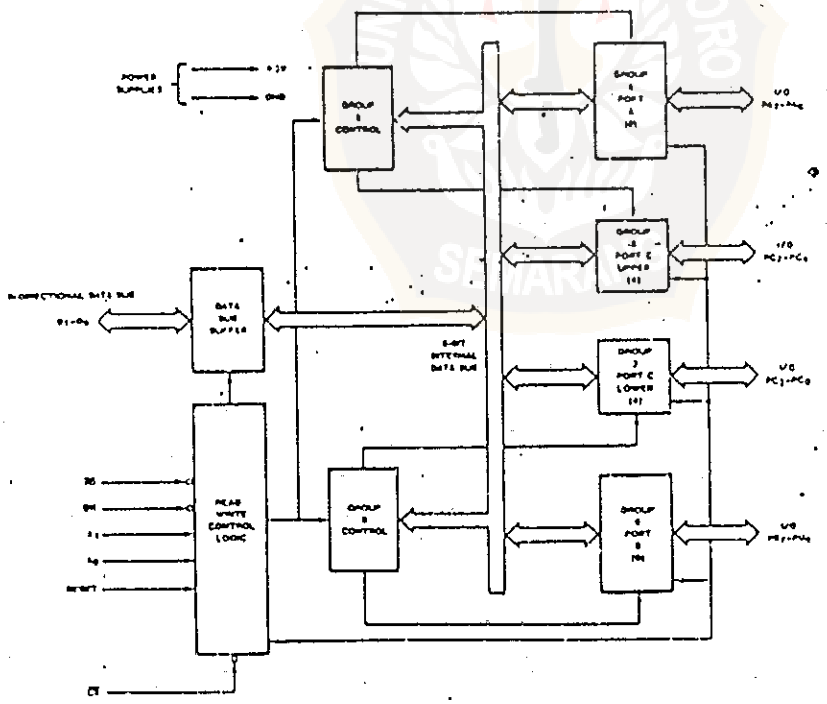


Figure 1. 8255A Block Diagram

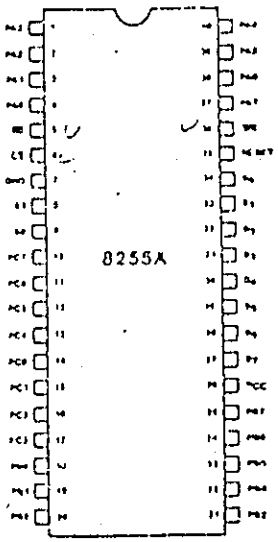


Figure 2. Pin Configuration

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

8255A FUNCTIONAL DESCRIPTION

General

The 8255A is a programmable peripheral interface (PPI) device designed for use in Intel® microcomputer systems. Its function is that of a general purpose I/O component to interface peripheral equipment to the microcomputer system bus. The functional configuration of the 8255A is programmed by the system software so that normally no external logic is necessary to interface peripheral devices or structures.

Data Bus Buffer

This 3-state bidirectional 8-bit buffer is used to interface the 8255A to the system data bus. Data is transmitted or received by the buffer upon execution of input or output instructions by the CPU. Control words and status information are also transferred through the data bus buffer.

Read/Write and Control Logic

The function of this block is to manage all of the internal and external transfers of both Data and Control or Status words. It accepts inputs from the CPU Address and Control busses and in turn, issues commands to both of the Control Groups.

($\overline{CS}$)

Chip Select. A "low" on this input pin enables the communiction between the 8255A and the CPU.

($\overline{RD}$)

Read. A "low" on this input pin enables the 8255A to send the data or status information to the CPU on the data bus. In essence, it allows the CPU to "read from" the 8255A.

($\overline{WR}$)

Write. A "low" on this input pin enables the CPU to write data or control words into the 8255A.

(A_0 and A_1)

Port Select 0 and Port Select 1. These input signals, in conjunction with the RD and WR inputs, control the selection of one of the three ports or the control word registers. They are normally connected to the least significant bits of the address bus (A_0 and A_1).

8255A BASIC OPERATION

A_1	A_0	$\overline{RD}$	$\overline{WR}$	$\overline{CS}$	INPUT OPERATION (READ)
0	0	0	1	0	PORT A → DATA BUS
0	1	0	1	0	PORT B → DATA BUS
1	0	0	1	0	PORT C → DATA BUS
					OUTPUT OPERATION (WRITE)
0	0	1	0	0	DATA BUS → PORT A
0	1	1	0	0	DATA BUS → PORT B
1	0	1	0	0	DATA BUS → PORT C
1	1	1	0	0	DATA BUS → CONTROL
					DISABLE FUNCTION
X	X	X	X	1	DATA BUS → 3-STATE
1	1	0	1	0	ILLEGAL CONDITION
X	X	1	1	0	DATA BUS → 3-STATE

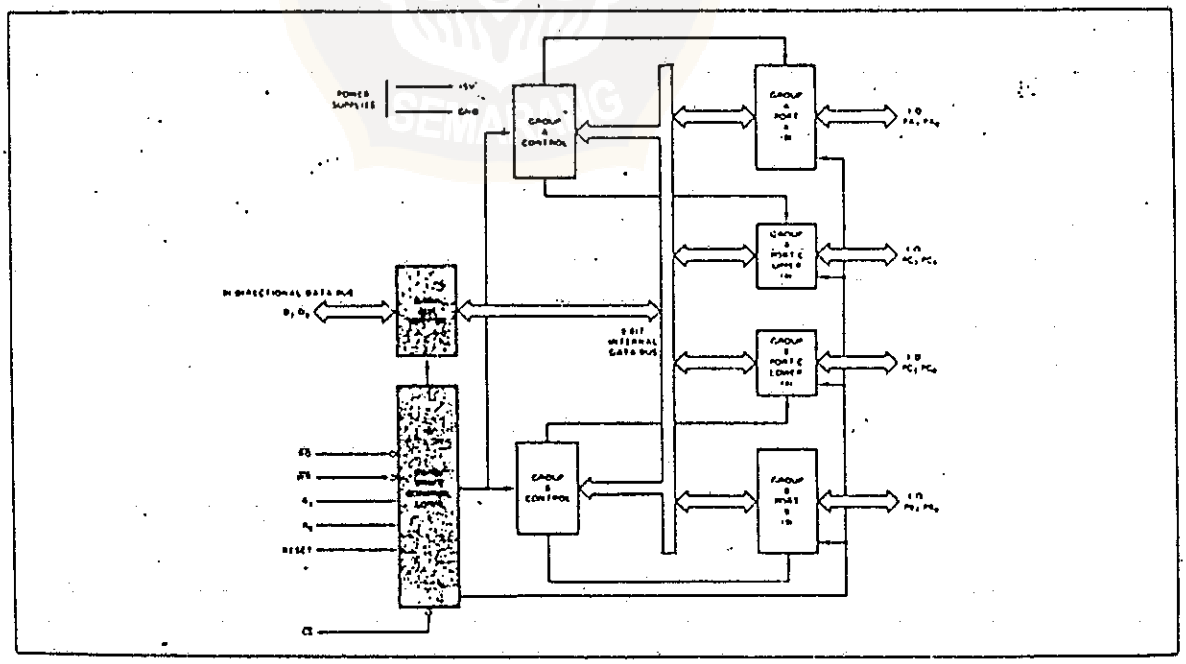


Figure 3. 8255A Block Diagram Showing Data Bus Buffer and Read/Write Control Logic Functions

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

(RESET)

Reset. A "high on this input clears the control register and all ports (A, C, C) are set to the input mode.

Group A and Group B Controls

The functional configuration of each port is programmed by the systems software. In essence, the CPU "outputs" a control word to the 8255A. The control word contains information such as "mode", "bit set", "bit reset", etc., that initializes the functional configuration of the 8255A.

Each of the Control blocks (Group A and Group B) accepts "commands" from the Read/Write Control Logic, receives "control words" from the internal data bus and issues the proper commands to its associated ports.

Control Group A – Port A and Port C upper (C7-C4)
Control Group B – Port B and Port C lower (C3-C0)

The Control Word Register can Only be written into. No Read operation of the Control Word Register is allowed.

Ports A, B, and C

The 8255A contains three 8-bit ports (A, B, and C). All can be configured in a wide variety of functional characteristics by the system software but each has its own special features or "personality" to further enhance the power and flexibility of the 8255A.

Port A. One 8-bit data output latch/buffer and one 8-bit data input latch.

Port B. One 8-bit data input/output latch/buffer and one 8-bit data input buffer.

Port C. One 8-bit data output latch/buffer and one 8-bit data input buffer (no latch for input). This port can be divided into two 4-bit ports under the mode control. Each 4-bit port contains a 4-bit latch and it can be used for the control signal outputs and status signal inputs in conjunction with ports A and B.

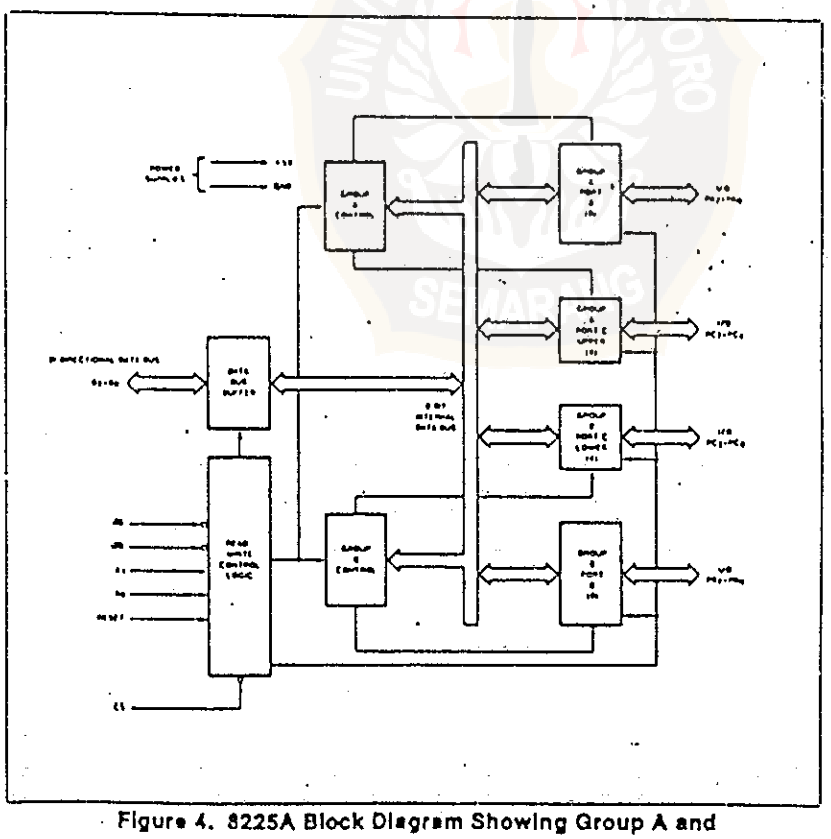
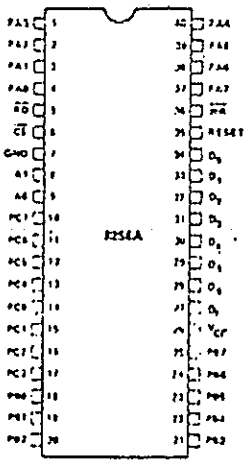


Figure 4. 8225A Block Diagram Showing Group A and Group B Control Functions

PIN CONFIGURATION



PIN NAMES

D_7-D_0	DATA BUS (BI-DIRECTIONAL)
RESET	RESET INPUT
$\overline{CS}$	CHIP SELECT
$\overline{RD}$	READ INPUT
$\overline{WR}$	WRITE INPUT
A0, A1	PORT ADDRESS
PA7-PA0	PORT A (BIT)
PB7-PB0	PORT B (BIT)
PC7-PC0	PORT C (BIT)
V_{CC}	+5 VOLTS
GND	0 VOLTS

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

8255A OPERATIONAL DESCRIPTION

Mode Selection

There are three basic modes of operation that can be selected by the system software:

Mode 0 – Basic Input/Output
Mode 1 – Strobed Input/Output
Mode 2 – Bi-Directional Bus

When the reset input goes "high" all ports will be set to the input mode (i.e., all 24 lines will be in the high impedance state). After the reset is removed the 8255A can remain in the input mode with no additional initialization required. During the execution of the system program any of the other modes may be selected using a single output instruction. This allows a single 8255A to service a variety of peripheral devices with a simple software maintenance routine.

The modes for Port A and Port B can be separately defined, while Port C is divided into two portions as required by the Port A and Port B definitions. All of the output registers, including the status flip-flops, will be reset whenever the mode is changed. Modes may be combined so that their functional definition can be "tailored" to almost any I/O structure. For instance; Group B can be programmed in Mode 0 to monitor simple switch closings or display computational results, Group A could be programmed in Mode 1 to monitor a keyboard or tape reader on an interrupt-driven basis.

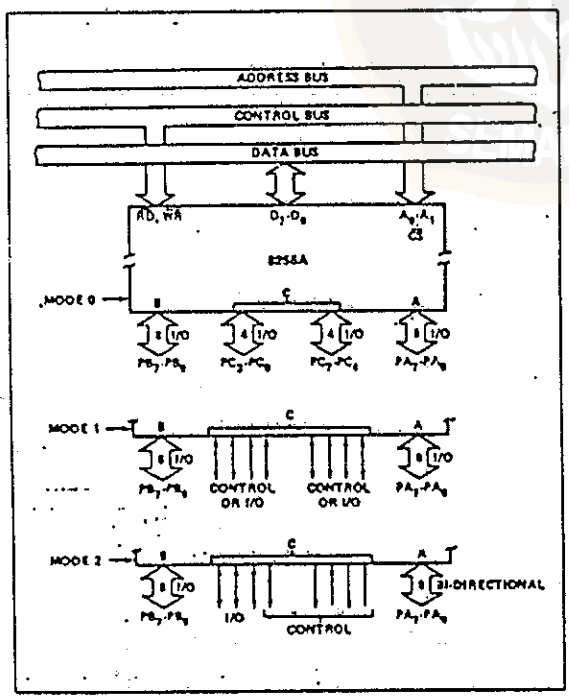


Figure 5. Basic Mode Definitions and Bus Interface

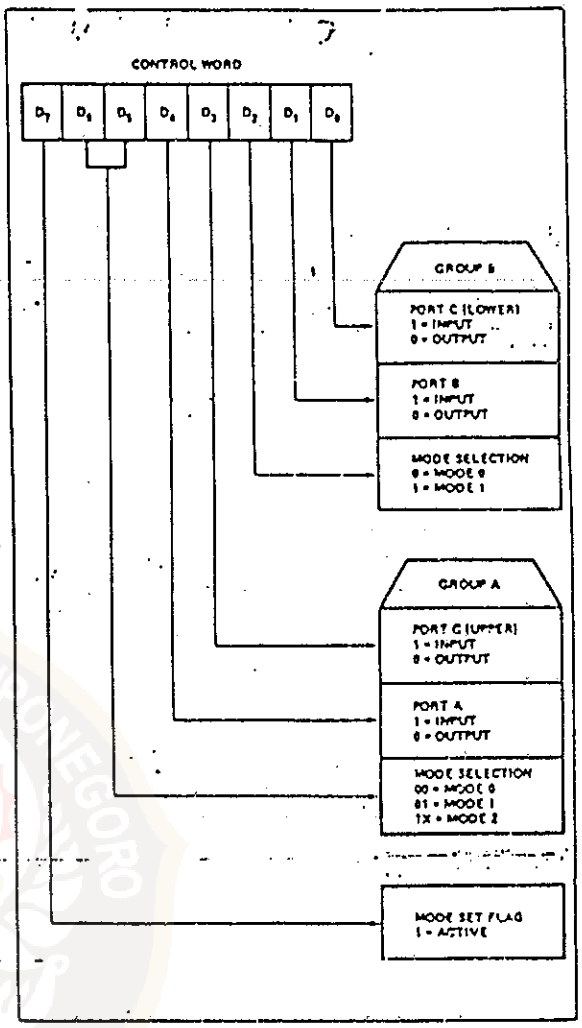


Figure 6. Mode Definition Format

The mode definitions and possible mode combinations may seem confusing at first but after a cursory review of the complete device operation a simple, logical I/O approach will surface. The design of the 8255A has taken into account things such as efficient PC board layout, control signal definition vs PC layout and complete functional flexibility to support almost any peripheral device with no external logic. Such design represents the maximum use of the available pins.

Single Bit Set/Reset Feature

Any of the eight bits of Port C can be Set or Reset using a single OUTput instruction. This feature reduces software requirements in Control-based applications.

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

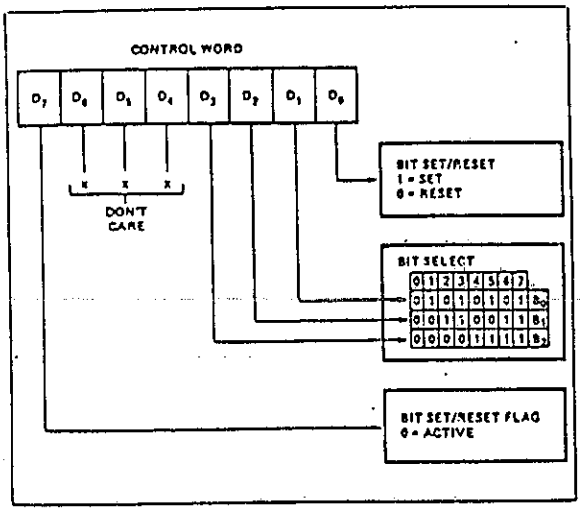


Figure 7. Bit Set/Reset Format

When Port C is being used as status/control for Port A or B, these bits can be set or reset by using the Bit Set/Reset operation just as if they were data output ports.

Interrupt Control Functions

When the 8255A is programmed to operate in mode 1 or mode 2, control signals are provided that can be used as interrupt request inputs to the CPU. The interrupt request signals, generated from port C, can be inhibited or enabled by setting or resetting the associated INTE flip-flop, using the bit set/reset function of port C.

This function allows the Programmer to disallow or allow a specific I/O device to interrupt the CPU without affecting any other device in the interrupt structure.

INTE flip-flop definition:

(BIT-SET) — INTE is SET — Interrupt enable
(BIT-RESET) — INTE is RESET — Interrupt disable

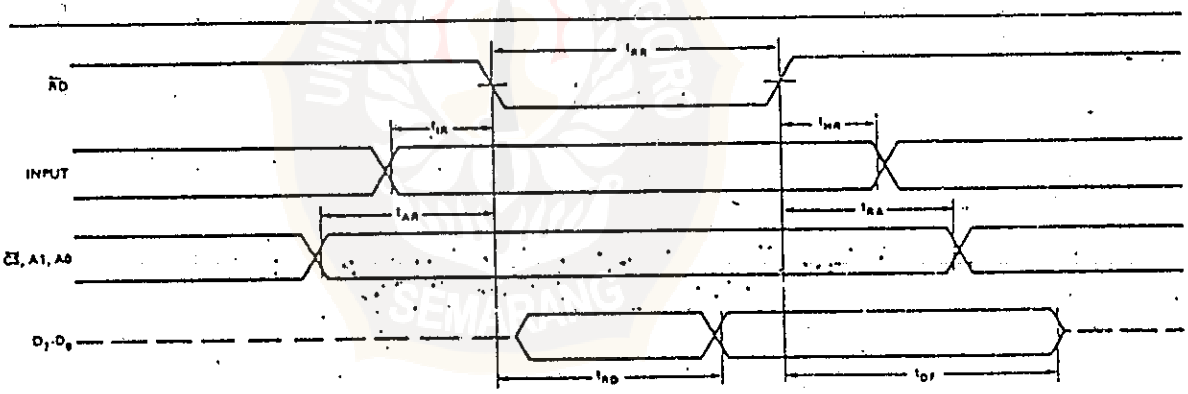
Note: All Mask flip-flops are automatically reset during mode selection and device Reset.

Operating Modes

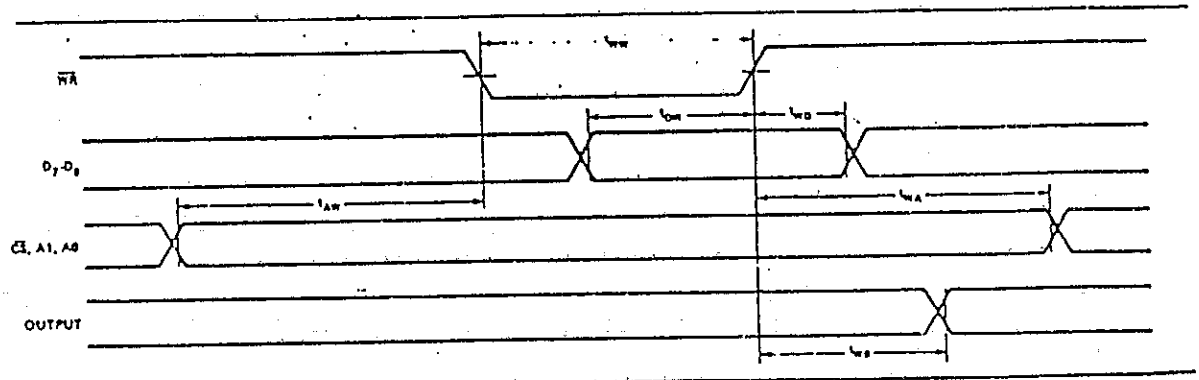
MODE 0 (Basic Input/Output). This functional configuration provides simple input and output operations for each of the three ports. No "handshaking" is required, data is simply written to or read from a specified port.

Mode 0 Basic Functional Definitions:

- Two 8-bit ports and two 4-bit ports.
- Any port can be input or output.
- Outputs are latched.
- Inputs are not latched.
- 16 different Input/Output configurations are possible in this Mode.



MODE 0 (Basic Input)



MODE 0 (Basic Output)

Input Control Signal Definition

STB (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F)

A "high" on this output indicates that the data has been loaded into the input latch; in essence, an acknowledgement. IBF is set by STB input being low and is reset by the rising edge of the RD input.

INTR (Interrupt Request)

A "high" on this output can be used to interrupt the CPU when an input device is requesting service. INTR is set by the $\overline{STB}$ is a "one", IBF is a "one" and INTE is a "one". It is reset by the falling edge of $\overline{RD}$. This procedure allows an input device to request service from the CPU by simply strobing its data into the port.

- INTE A
  Controlled by bit set/reset of PC_4.
- INTE B
  Controlled by bit set/reset of PC_2.

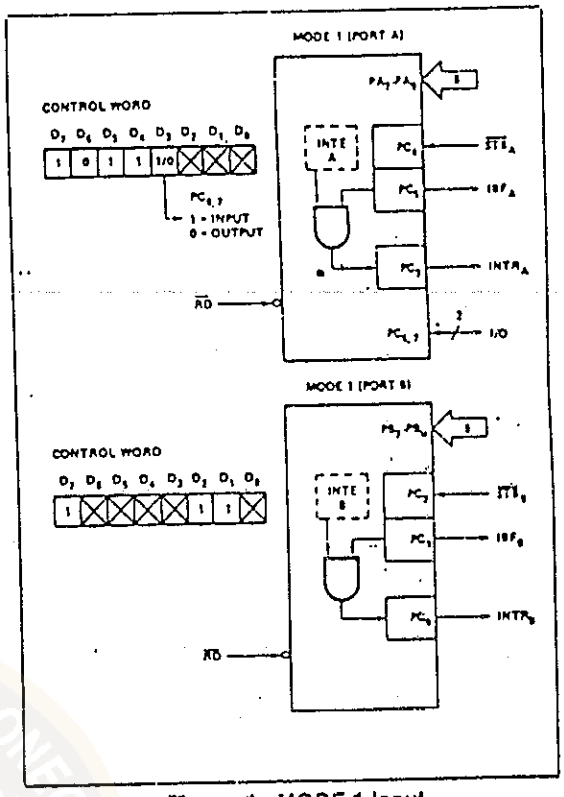


Figure 8. MODE 1 Input

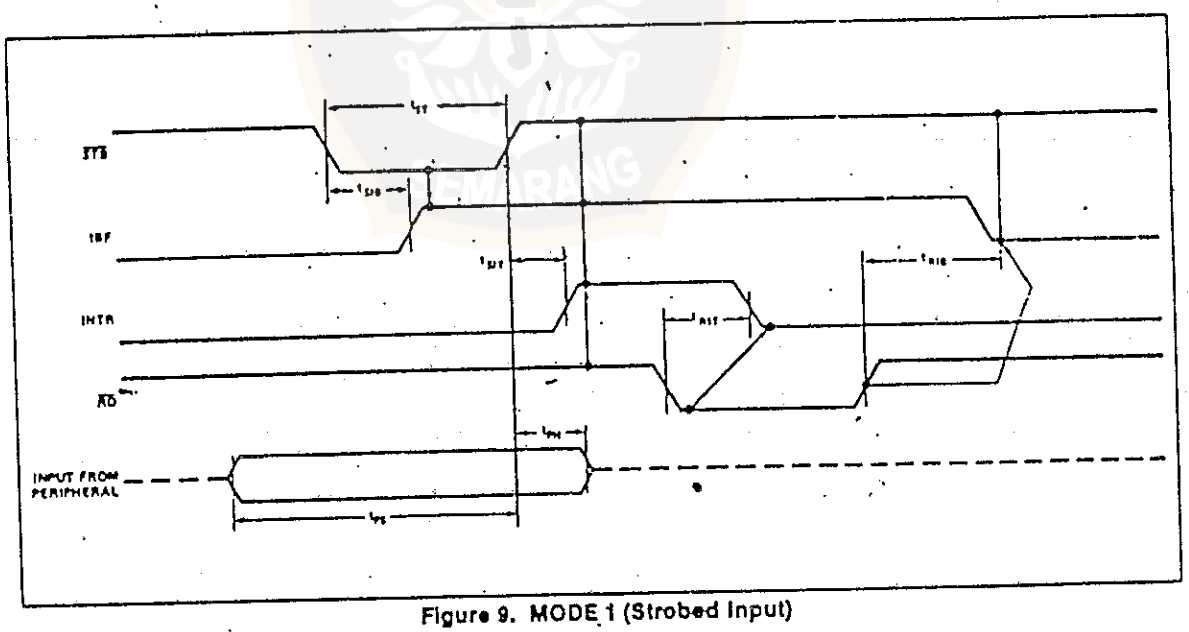


Figure 9. MODE 1 (Strobed Input)

Output Control Signal Definition

$\overline{OBF}$ (Output Buffer Full F/F). The OBF output will go "low" to indicate that the CPU has written data out to the specified port. The OBF F/F will be set by the rising edge of the WR input and reset by ACK input being low.

$\overline{ACK}$ (Acknowledge Input). A "low" on this input informs the 8255A that the data from port A or port B has been accepted. In essence, a response from the peripheral device indicating that it has received the data output by the CPU.

INTR (Interrupt Request). A "high" on this output can be used to interrupt the CPU when an output device has accepted data transmitted by the CPU. INTR is set when ACK is a "one", OBF is a "one" and INTE is a "one". It is reset by the falling edge of WR.

INTE A.

Controlled by bit set/reset of PC_6.

INTE B

Controlled by bit set/reset of PC_2.

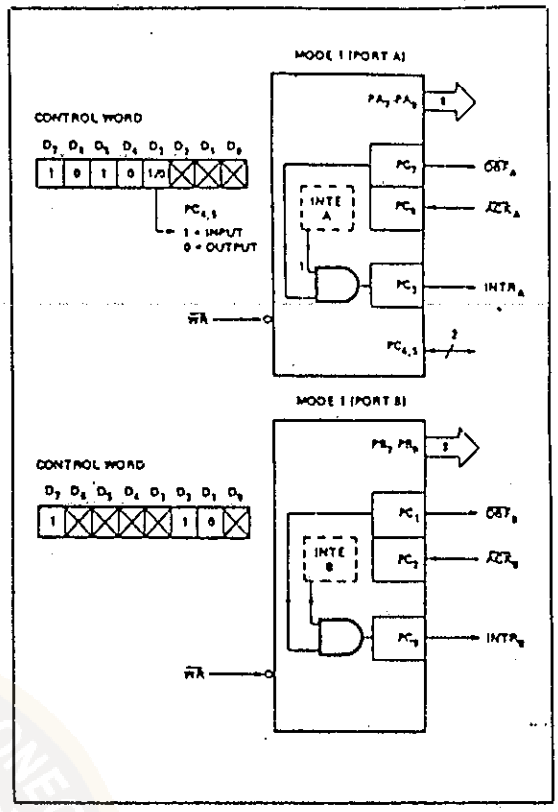


Figure 10. MODE 1 Output

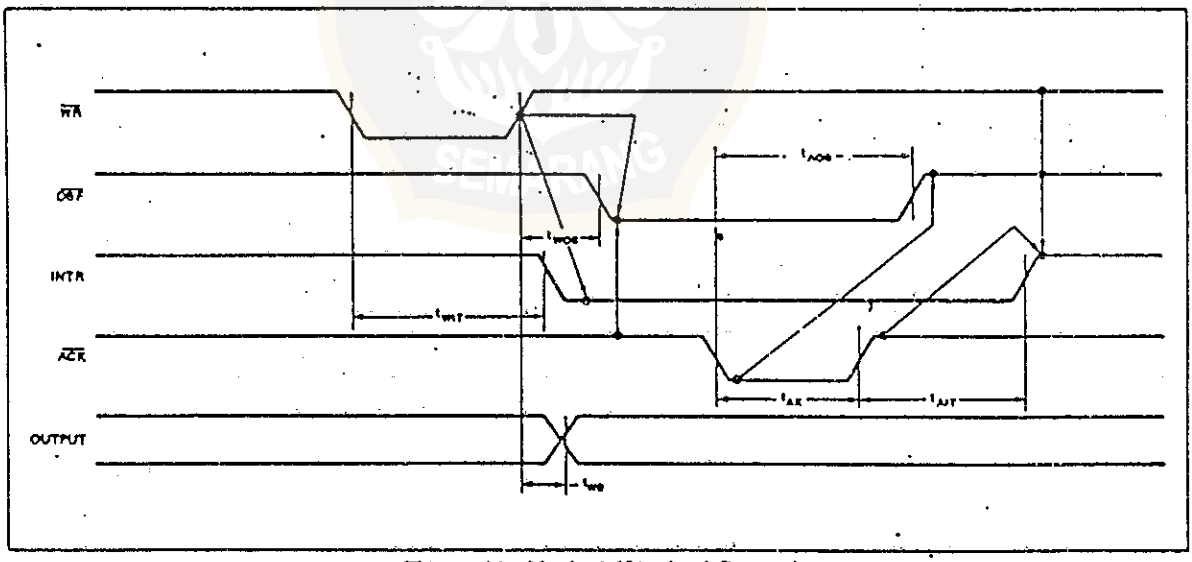
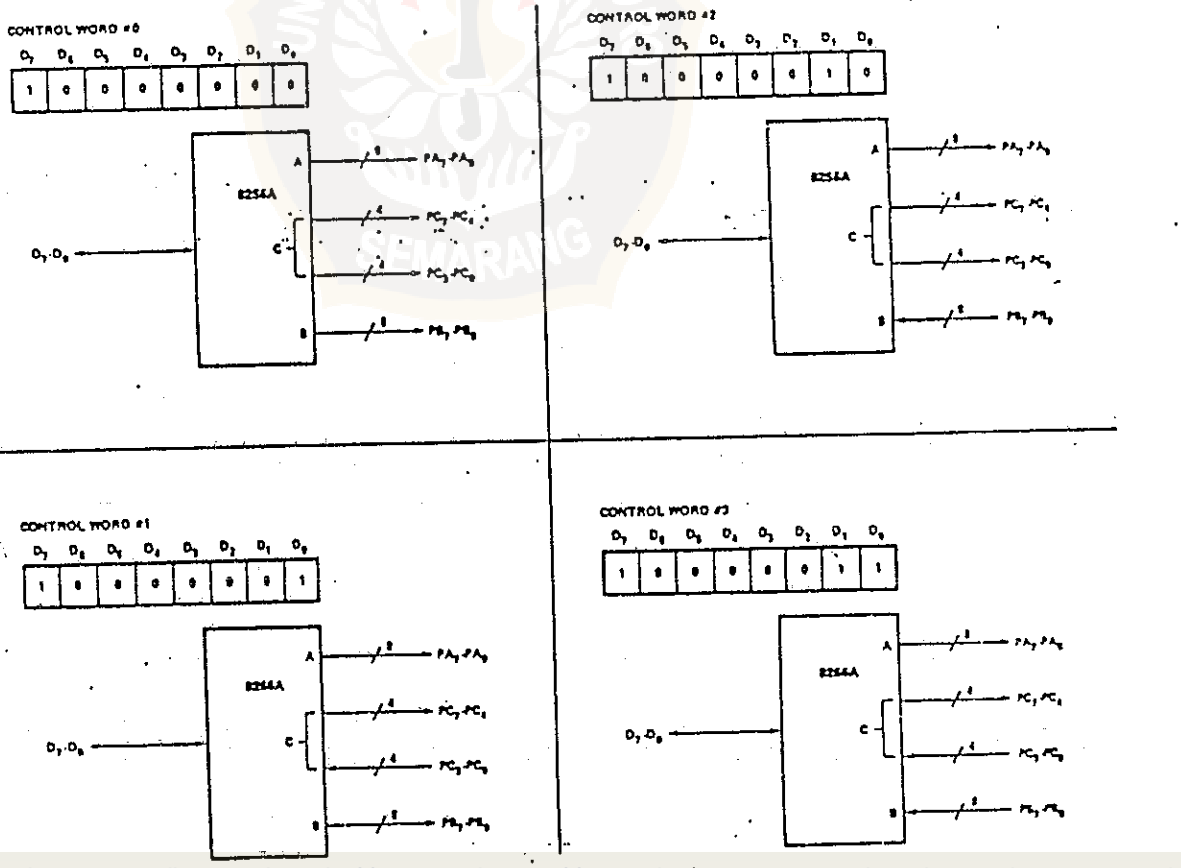


Figure 11. Mode 1 (Strobed Output)

MODE 0 Port Definition

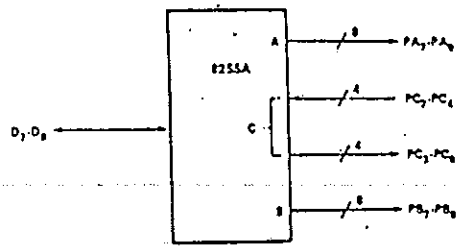
A		B		GROUP A			GROUP B	
D_4	D_3	D_1	D_0	PORT A	PORT C (UPPER)	#	PORT B	PORT C (LOWER)
0	0	0	0	OUTPUT	OUTPUT	0	OUTPUT	OUTPUT
0	0	0	1	OUTPUT	OUTPUT	1	OUTPUT	INPUT
0	0	1	0	OUTPUT	OUTPUT	2	INPUT	OUTPUT
0	0	1	1	OUTPUT	OUTPUT	3	INPUT	INPUT
0	1	0	0	OUTPUT	INPUT	4	OUTPUT	OUTPUT
0	1	0	1	OUTPUT	INPUT	5	OUTPUT	INPUT
0	1	1	0	OUTPUT	INPUT	6	INPUT	OUTPUT
0	1	1	1	OUTPUT	INPUT	7	INPUT	INPUT
1	0	0	0	INPUT	OUTPUT	8	OUTPUT	OUTPUT
1	0	0	1	INPUT	OUTPUT	9	OUTPUT	INPUT
1	0	1	0	INPUT	OUTPUT	10	INPUT	OUTPUT
1	0	1	1	INPUT	OUTPUT	11	INPUT	INPUT
1	1	0	0	INPUT	INPUT	12	OUTPUT	OUTPUT
1	1	0	1	INPUT	INPUT	13	OUTPUT	INPUT
1	1	1	0	INPUT	INPUT	14	INPUT	OUTPUT
1	1	1	1	INPUT	INPUT	15	INPUT	INPUT

MODE 0 Configurations



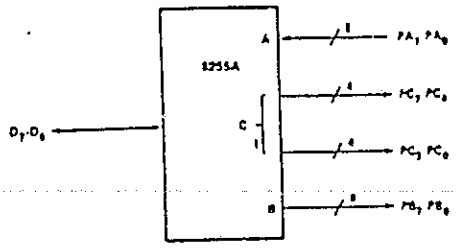
CONTROL WORD #4

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	0



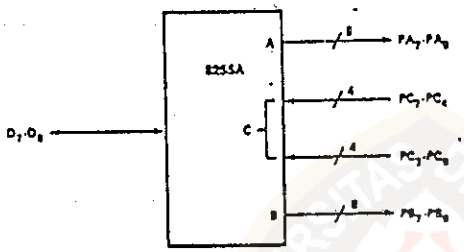
CONTROL WORD #8

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	0



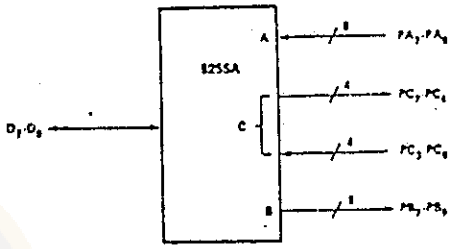
CONTROL WORD #5

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	0	1



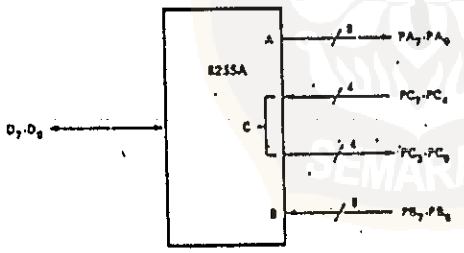
CONTROL WORD #9

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	0	1



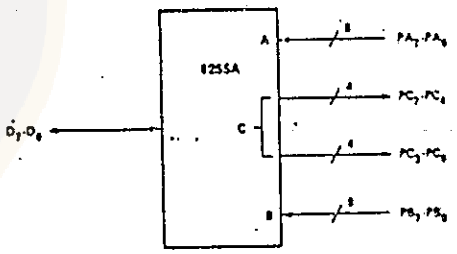
CONTROL WORD #6

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	0



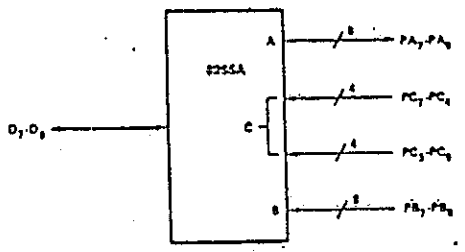
CONTROL WORD #10

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	0



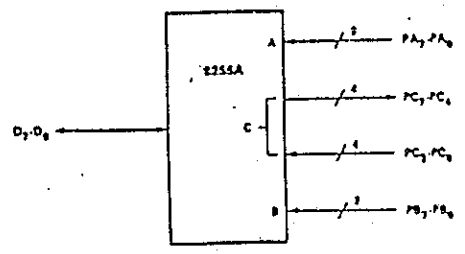
CONTROL WORD #7

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	0	1	0	1	1

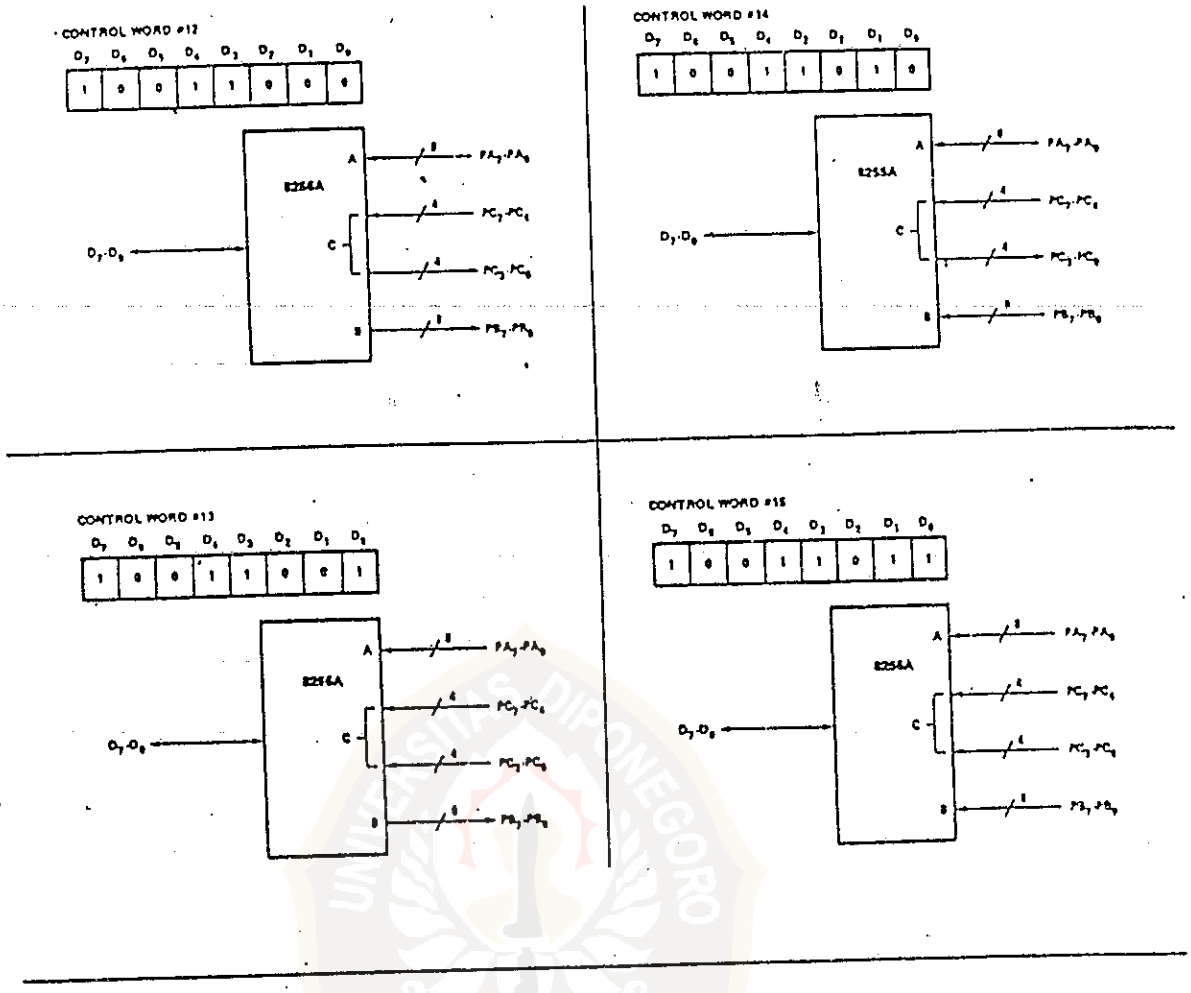


CONTROL WORD #11

D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
1	0	0	1	0	0	1	1



8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE



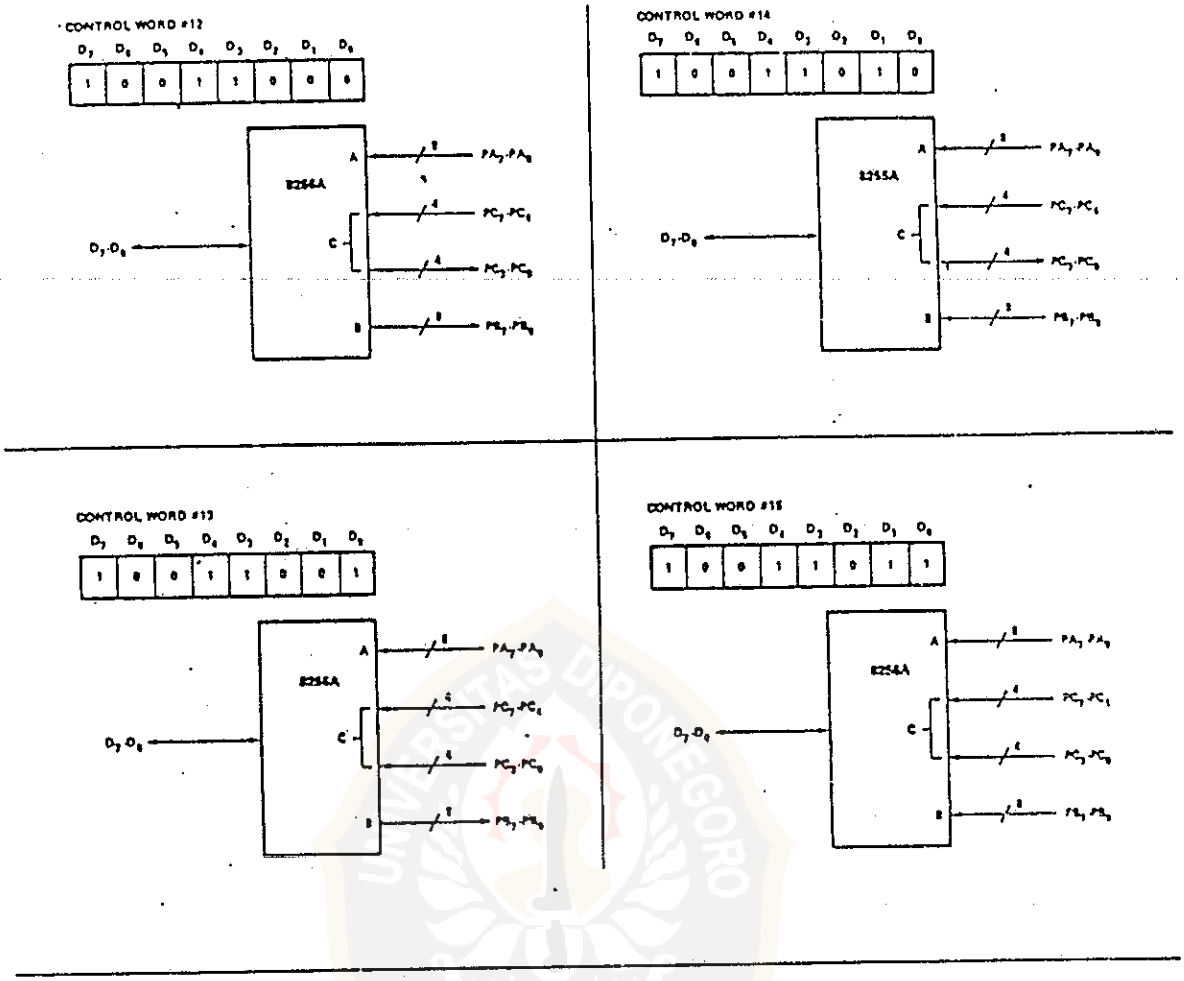
Operating Modes

MODE 1 (Strobed Input/Output). This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In mode 1, port A and Port B use the lines on port C to generate or accept these "handshaking" signals.

Mode 1 Basic Functional Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE



Operating Modes

MODE 1 (Strobed Input/Output). This functional configuration provides a means for transferring I/O data to or from a specified port in conjunction with strobes or "handshaking" signals. In mode 1, port A and Port B use the lines on port C to generate or accept these "handshaking" signals.

Mode 1 Basic Functional Definitions:

- Two Groups (Group A and Group B)
- Each group contains one 8-bit data port and one 4-bit control/data port.
- The 8-bit data port can be either input or output. Both inputs and outputs are latched.
- The 4-bit port is used for control and status of the 8-bit data port.

8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

Combinations of MODE 1

Port A and Port B can be individually defined as input or output in Mode 1 to support a wide variety of strobed I/O applications.

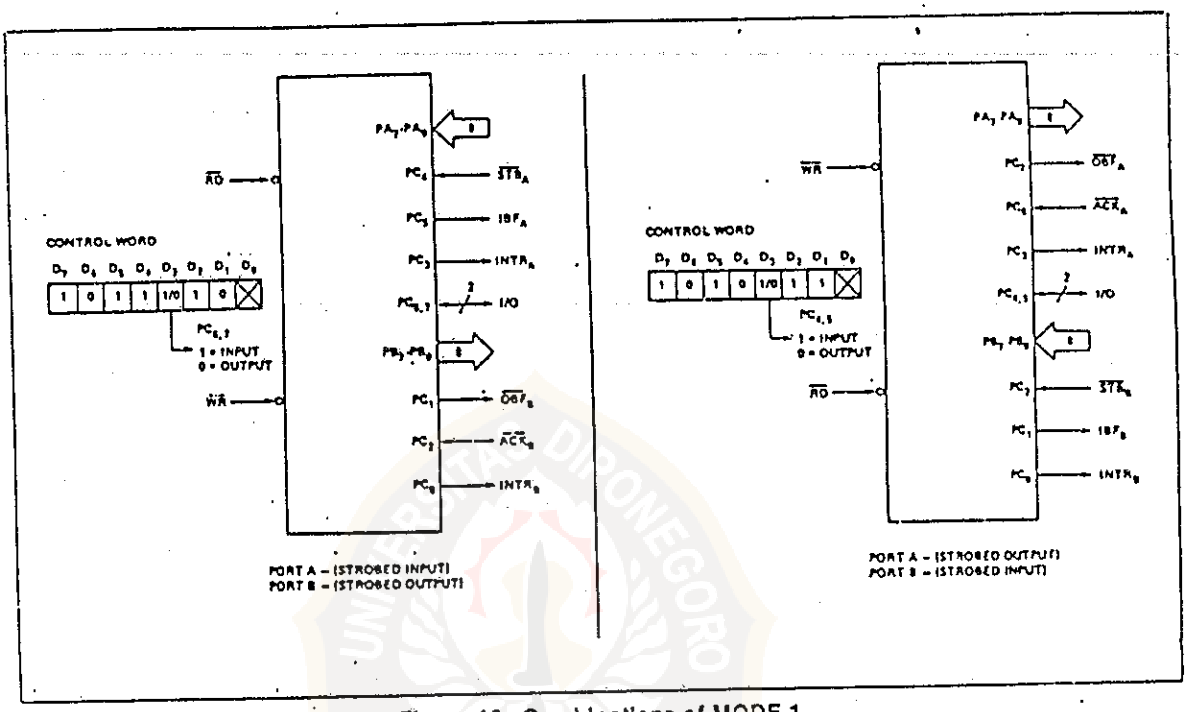


Figure 12. Combinations of MODE 1

Operating Modes

MODE 2 (Strobed Bidirectional Bus I/O). This functional configuration provides a means for communicating with a peripheral device or structure on a single 8-bit bus for both transmitting and receiving data (bidirectional bus I/O). "Handshaking" signals are provided to maintain proper bus flow discipline in a similar manner to MODE 1. Interrupt generation and enable/disable functions are also available.

MODE 2 Basic Functional Definitions:

- Used in Group A only.
- One 8-bit, bi-directional bus Port (Port A) and a 5-bit control Port (Port C).
- Both inputs and outputs are latched.
- The 5-bit control port (Port C) is used for control and status for the 8-bit, bi-directional bus port (Port A).

Bidirectional Bus I/O Control Signal Definition

INTR (Interrupt Request). A high on this output can be used to interrupt the CPU for both input or output operations.

Output Operations

$\overline{OBF}$ (Output Buffer Full). The OBF output will go "low" to indicate that the CPU has written data out to port A.

$\overline{ACK}$ (Acknowledge). A "low" on this input enables the tri-state output buffer of port A to send out the data. Otherwise, the output buffer will be in the high impedance state.

INTE 1 (The INTE Flip-Flop Associated with $\overline{OBF}$). Controlled by bit set/reset of PC_6.

Input Operations

$\overline{STB}$ (Strobe Input).

STB (Strobe Input). A "low" on this input loads data into the input latch.

IBF (Input Buffer Full F/F). A "high" on this output indicates that data has been loaded into the input latch.

INTE 2 (The INTE Flip-Flop Associated with IBF). Controlled by bit set/reset of PC_4.

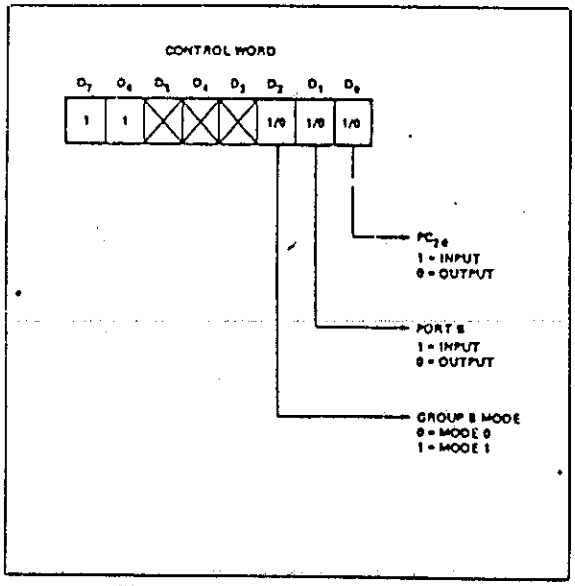


Figure 13. MODE Control Word

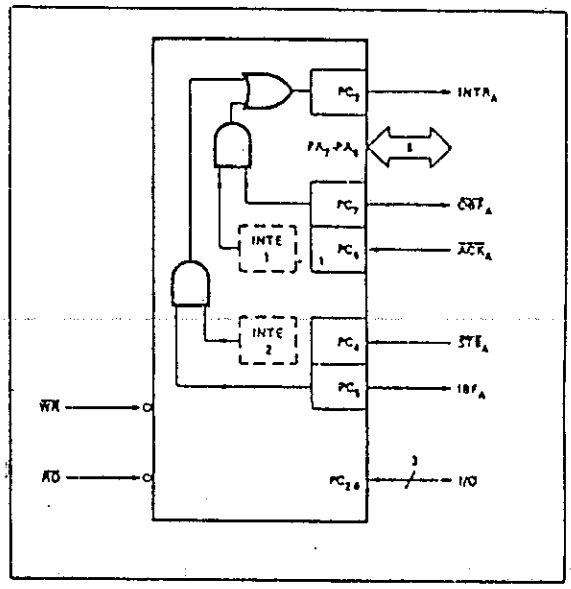


Figure 14. MODE 2

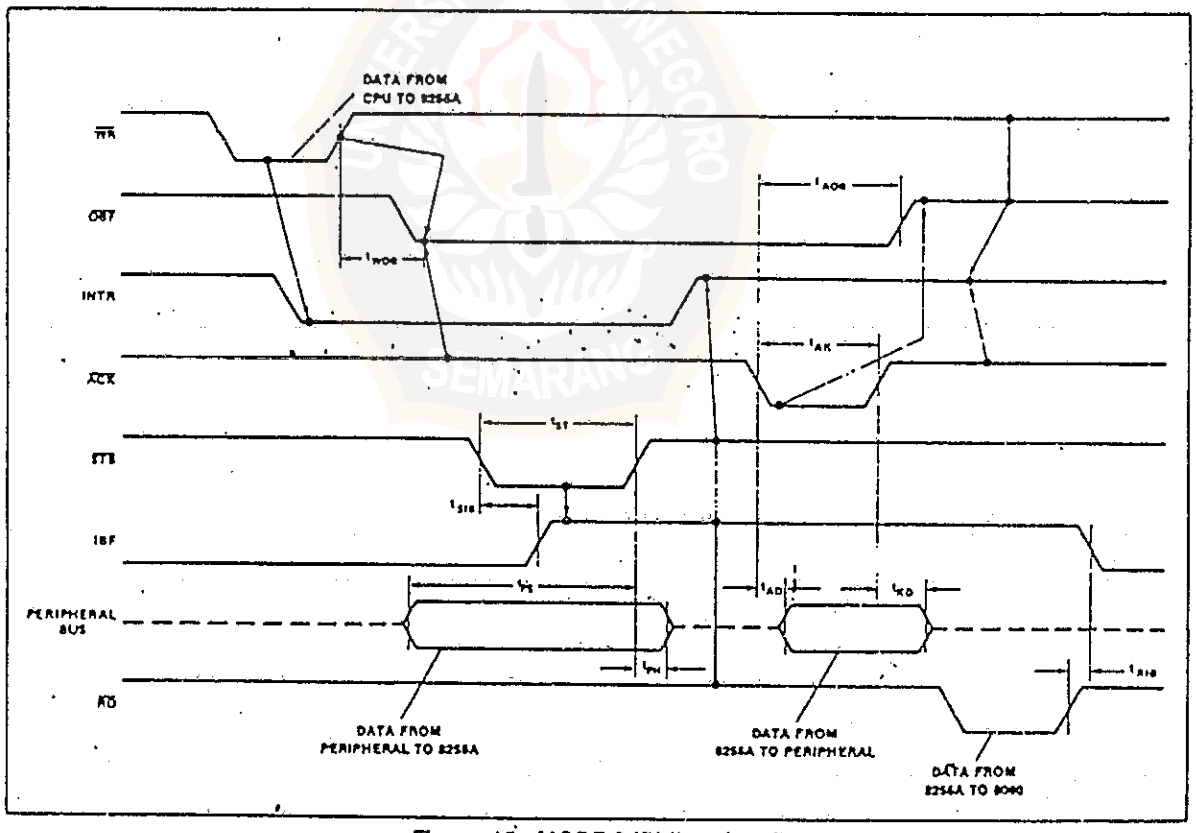
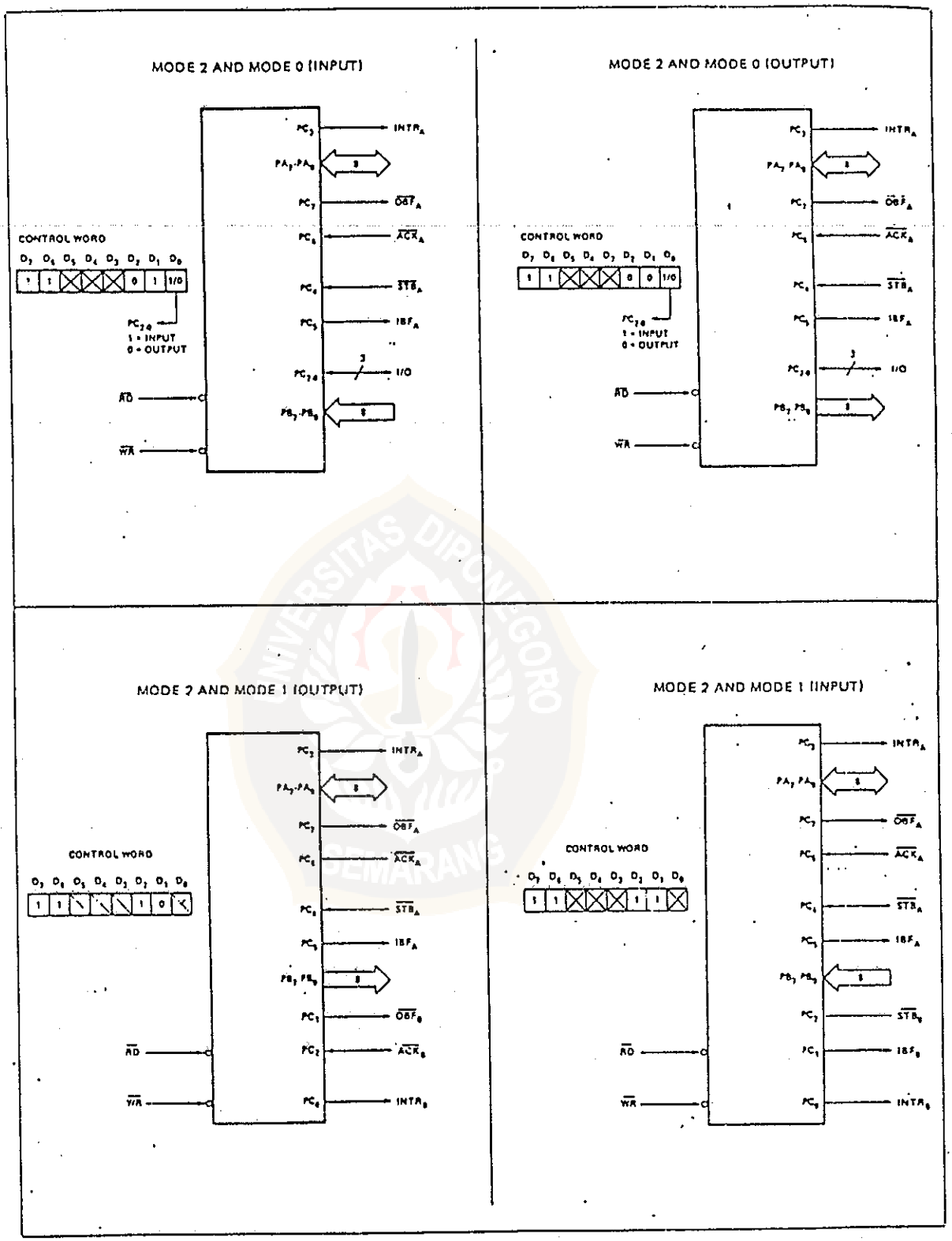


Figure 15. MODE 2 (Bidirectional)

NOTE: Any sequence where $\overline{WR}$ occurs before $\overline{ACK}$ and $\overline{STB}$ occurs before $\overline{RD}$ is permissible.
(INTR = IBF • $\overline{MASK}$ • $\overline{STB}$ • $\overline{RD}$ + $\overline{OBF}$ • $\overline{MASK}$ • $\overline{ACK}$ • $\overline{WR}$)



8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

Mode Definition Summary

	MODE 0		MODE 1		MODE 2	
	IN	OUT	IN	OUT	GROUP A ONLY	
PA_0	IN	OUT	IN	OUT	←→	
PA_1	IN	OUT	IN	OUT	←→	
PA_2	IN	OUT	IN	OUT	←→	
PA_3	IN	OUT	IN	OUT	←→	
PA_4	IN	OUT	IN	OUT	←→	
PA_5	IN	OUT	IN	OUT	←→	
PA_6	IN	OUT	IN	OUT	←→	
PA_7	IN	OUT	IN	OUT	←→	
PB_0	IN	OUT	IN	OUT	—	MODE 0 OR MODE 1 ONLY
PB_1	IN	OUT	IN	OUT	—	
PB_2	IN	OUT	IN	OUT	—	
PB_3	IN	OUT	IN	OUT	—	
PB_4	IN	OUT	IN	OUT	—	
PB_5	IN	OUT	IN	OUT	—	
PB_6	IN	OUT	IN	OUT	—	
PB_7	IN	OUT	IN	OUT	—	
PC_0	IN	OUT	$INTR_B$	$INTR_B$	I/O	
PC_1	IN	OUT	IBF_B	$\overline{OBF}_B$	I/O	
PC_2	IN	OUT	$\overline{STB}_B$	$\overline{ACK}_B$	I/O	
PC_3	IN	OUT	$INTR_A$	$INTR_A$	$INTR_A$	
PC_4	IN	OUT	$\overline{STB}_A$	I/O	$\overline{STB}_A$	
PC_5	IN	OUT	IBF_A	I/O	IBF_A	
PC_6	IN	OUT	I/O	$\overline{ACK}_A$	$\overline{ACK}_A$	
PC_7	IN	OUT	I/O	$\overline{OBF}_A$	$\overline{OBF}_A$	

Special Mode Combination Considerations

There are several combinations of modes when not all of the bits in Port C are used for control or status. The remaining bits can be used as follows:

If Programmed as Inputs –
All input lines can be accessed during a normal Port C read.

If Programmed as Outputs –
Bits in C upper (PC_7-PC_4) must be individually accessed using the bit set/reset function.

Bits in C lower (PC_3-PC_0) can be accessed using the bit set/reset function or accessed as a threesome by writing into Port C.

Source Current Capability on Port B and Port C

Any set of eight output buffers, selected randomly from Ports B and C can source 1mA at 1.5 volts. This feature allows the 8255 to directly drive Darlington type drivers and high-voltage displays that require such source current.

Reading Port C Status

In Mode 0, Port C transfers data to or from the peripheral device. When the 8255 is programmed to function in Modes 1 or 2, Port C generates or accepts "hand-shaking" signals with the peripheral device. Reading the contents of Port C allows the programmer to test or verify the "status" of each peripheral device and change the program flow accordingly.

There is no special instruction to read the status information from Port C. A normal read operation of Port C is executed to perform this function.

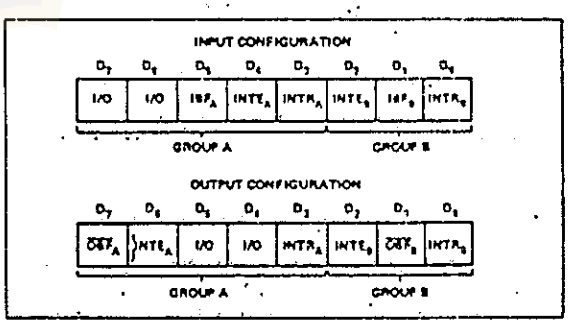


Figure 17. MODE 1 Status Word Format

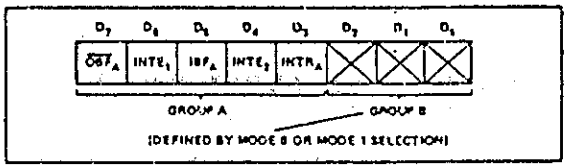


Figure 18. MODE 2 Status Word Format

APPLICATIONS OF THE 8255A

The 8255A is a very powerful tool for interfacing peripheral equipment to the microcomputer system. It represents the optimum use of available pins and is flexible enough to interface almost any I/O device without the need for additional external logic.

Each peripheral device in a microcomputer system usually has a "service routine" associated with it. The routine manages the software interface between the device and the CPU. The functional definition of the 8255A is programmed by the I/O service routine and becomes an extension of the system software. By examining the I/O devices interface characteristics for both data transfer and timing, and matching this information to the examples and tables in the detailed operational description, a control word can easily be developed to initialize the 8255A to exactly "fit" the application. Figures 19 through 25 present a few examples of typical applications of the 8255A.

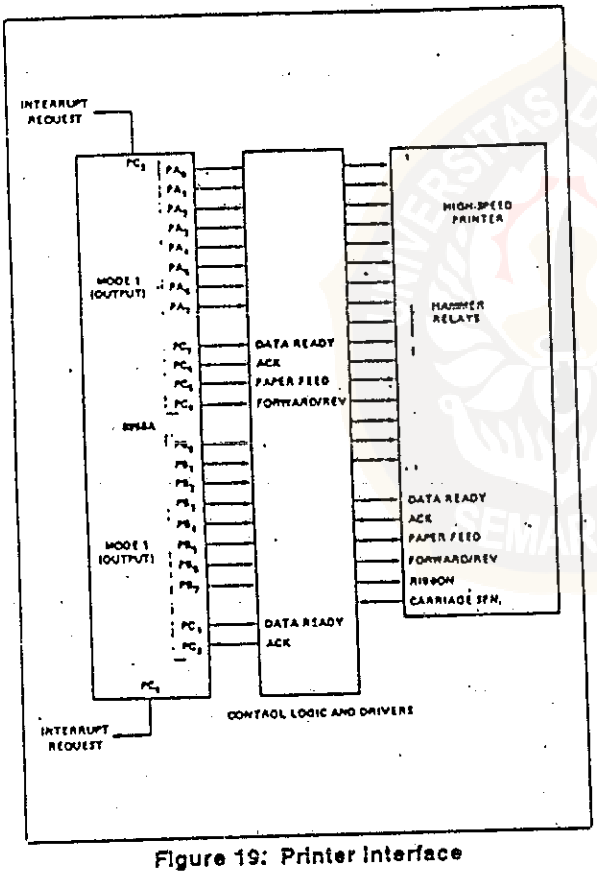


Figure 19: Printer Interface

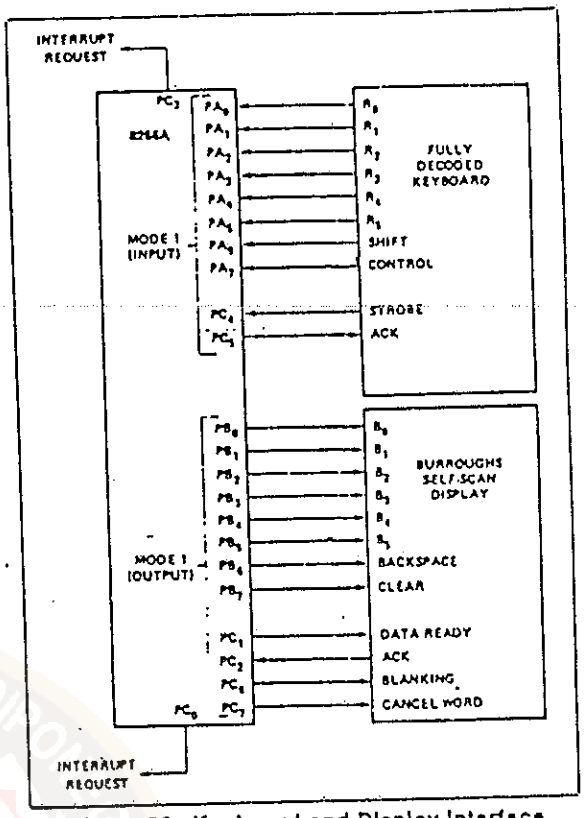


Figure 20. Keyboard and Display Interface

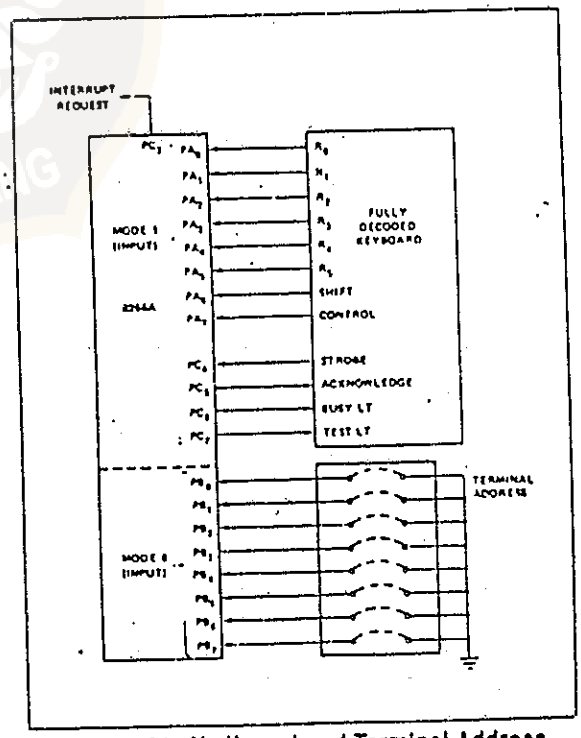


Figure 21. Keyboard and Terminal Address Interface

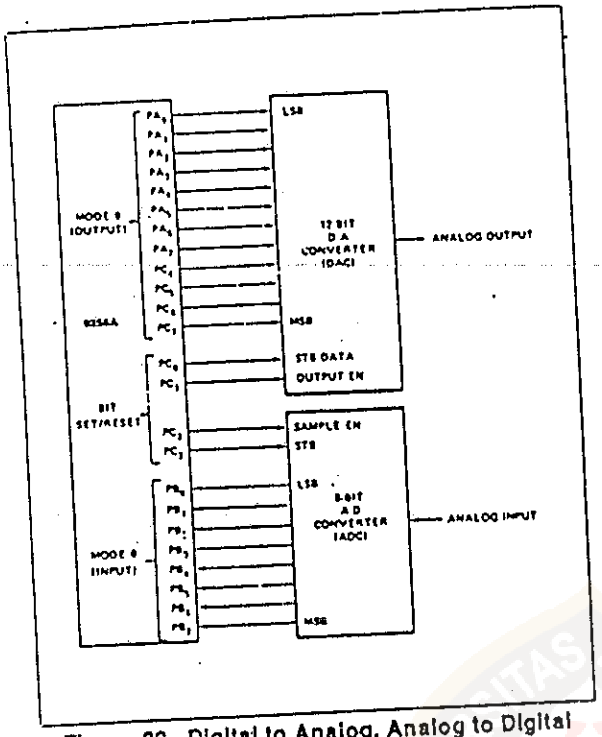


Figure 22. Digital to Analog, Analog to Digital

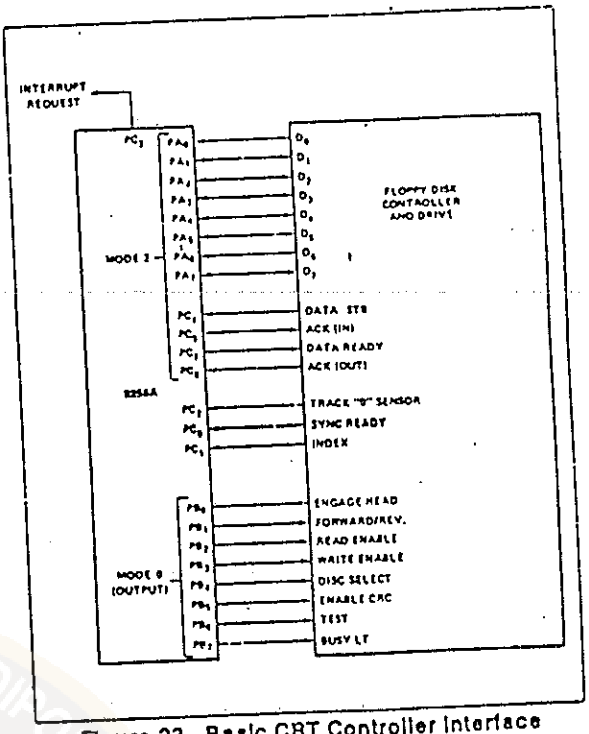


Figure 23. Basic CRT Controller Interface

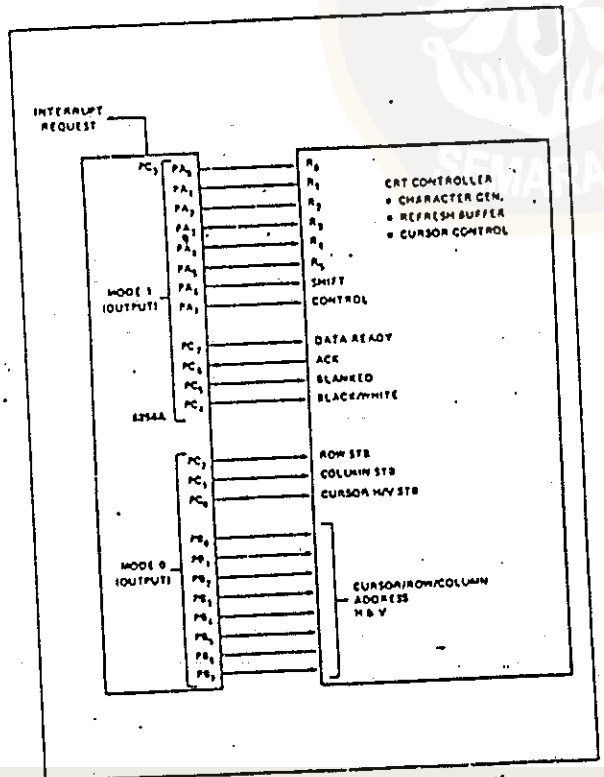


Figure 24. Basic Floppy Disc Interface

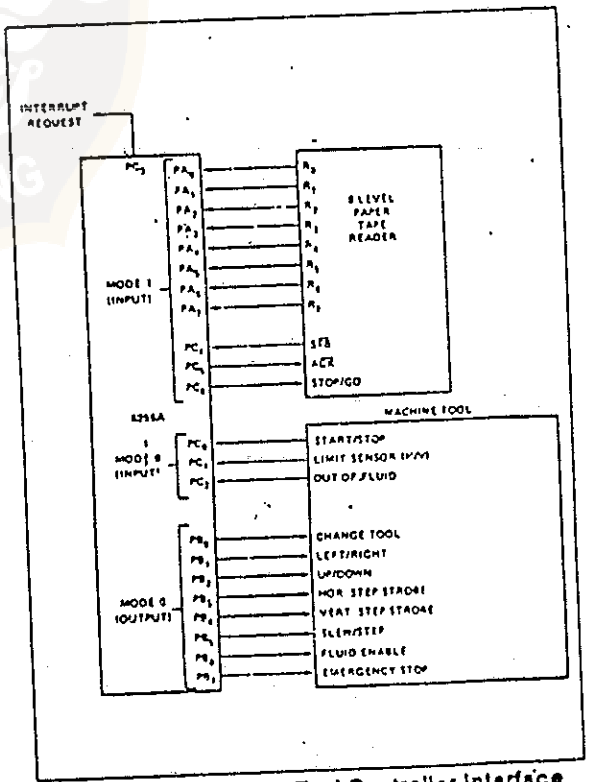
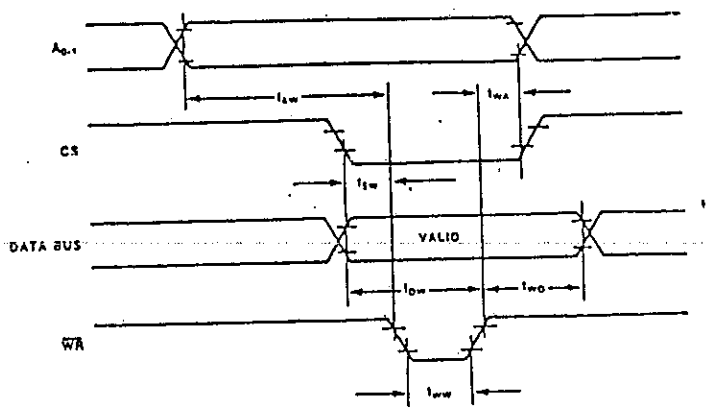


Figure 25. Machine Tool Controller Interface

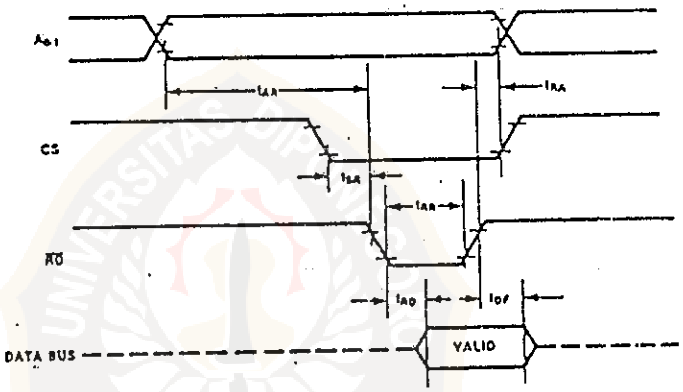
8255A/8255A-5
PROGRAMMABLE PERIPHERAL INTERFACE

WAVEFORMS

WRITE



READ



RECOVERY

CLOCK AND GATE