3823.112 -EN-
394

```
TITLE "Silicon Acquisition and Readout Module";

%-------------------------------------------------------

        D. Mendoza & Marcelo Mendes
        Fermilab
        10/11/93
-------------------------------------------------------%

DESIGN IS "pclevel"
BEGIN
 DEVICE IS "EPM7032LC44-10"
 BEGIN

  bdsel @ 16, vad1 @ 7, vad2 @ 8, vad3 @ 9,
  vad4 @ 11, vad5 @ 12, vad10 @ 13, vad9 @ 14,
  pc0 @ 6, pc2 @ 5, gl @ 4, hdi @ 41, vwrt @ 17,
  vdtck_oen_in @ 44, sysres @ 20, ds @ 27                    :INPUT;

  sel_delay_pc0 @ 40, sel_delay_pc1 @ 39, sel_delay_pc2 @ 38,
  sel_delay_pc3 @ 37, sel_delay_clk @21 ,wr_delay @ 34, rd_delay @ 36,
  en_fsm_pc0 @ 32 , en_fsm_pc1 @ 31,
  en_fsm_pc2 @ 29, en_fsm_pc3 @ 28 , vdtck @ 18,
  vdtck_aux @ 33                                             OUTPUT;

 END;

 CONNECTED_PINS
 BEGIN
        vdtck_oen_in     :INPUT;
        vdtck_aux        :OUTPUT;
 END;


END;


SUBDESIGN pclevel

(

        bdsel                    : INPUT;
        vad[5..1]                : INPUT;
        vad10                    : INPUT;
        vad9                     : INPUT;
        pc0                      : INPUT;
        pc2                      : INPUT;
        gl                       : INPUT;
        hdi                      : INPUT;
        vwrt                     : INPUT;
        vdtck_oen_in             : INPUT;
        sysres                   : INPUT;
        ds                       : INPUT;
        vdtck                    : OUTPUT;
```

```
        vdtck_aux                       : OUTPUT;
        sel_delay_pc[3..0]              : OUTPUT;
        sel_delay_clk                   : OUTPUT;
        wr_delay                        : OUTPUT;
        rd_delay                        : OUTPUT;
        en_fsm_pc[3..0]                 : OUTPUT;
)

VARIABLE

        vdtck_aux               :DFF;
        sel_delay_pc[3..0]      :DFF;
        sel_delay_clk           :DFF;
        en_fsm_pc[3..0]         :DFF;
        rd_delay_aux            :DFF;
        wr_delay                :DFF;
        vdtck_oen               :node;
        rd_delay                :node;

        aux                     :node;
        aux0                    :node;
        aux1                    :node;
        aux2                    :node;

BEGIN

        aux0 = LCELL ( NOT bdsel);
        aux1 = LCELL ( NOT aux0 );
        aux2 = LCELL ( NOT aux1);
        aux = LCELL ( aux2 # bdsel);

        sel_delay_pc[].clk = bdsel;
        sel_delay_pc[].prn = NOT sysres AND aux;
        sel_delay_clk.clk = bdsel;
        sel_delay_clk.prn = NOT sysres AND aux;

        en_fsm_pc[].clk = bdsel;
        en_fsm_pc[].clrn = aux;


        vdtck_aux.d = GND;
        vdtck_aux.clk = wr_delay # rd_delay_aux #
                        en_fsm_pc0 # en_fsm_pc1 # en_fsm_pc2 # en_fsm_pc3;
        vdtck_aux.prn = NOT sysres AND aux;


        vdtck = TRI ( vdtck_aux, vdtck_oen );
        vdtck_oen = NOT ( vdtck_oen_in );


        sel_delay_clk = NOT (!pc2 & !pc0 & !vad10 & vad9 & !gl &
                             !hdi & !vad5 & !vad4 & !vad3 & !vad2 & !vad1);
```

```
BEGIN

    DEFAULTS

        vdtck_aux = GND;

    END DEFAULTS;

    aux0 = LCELL ( NOT bdsel);
    aux1 = LCELL ( NOT aux0 );
    aux2 = LCELL ( NOT aux1);
    aux = LCELL ( aux2 # bdsel);


    vdtck_aux.d  = GND;
    vdtck_aux.clk =  (      clk_bitpatt_dat1_gl_0 #
                            clk_bitpatt_dat1_gl_1 #
                            clk_bitpatt_dat1_gl_2 #
                            clk_bitpatt_dat1_gl_3 #
                            clk_bitpatt_dat2_gl_0 #
                            clk_bitpatt_dat2_gl_1 #
                            clk_bitpatt_dat2_gl_2 #
                            clk_bitpatt_dat2_gl_3 );

    vdtck_aux.prn = NOT sysres AND aux;

    vdtck = TRI ( vdtck_aux, vdtck_oen );
    vdtck_oen = NOT ( vdtck_oen_in );

    % PORT CARD 0 %
    % G-LINK 0 %

    clk_bitpatt_dat1_gl_0.d = vwrt & vad10 & !vad9 & !pc2 & !pc0 & !gl &
                              !vad5 & !vad4 & !vad3 & !vad2 & !vad1;
    clk_bitpatt_dat1_gl_0.clk = ds;
    clk_bitpatt_dat1_gl_0.clrn = NOT sysres AND aux;

    clk_bitpatt_dat2_gl_0 = vwrt & vad10 & !vad9 & !pc2 & !pc0 & !gl &
                            !vad5 & !vad4 & !vad3 & !vad2 & vad1;
    clk_bitpatt_dat2_gl_0.clk = ds;
    clk_bitpatt_dat2_gl_0.clrn = NOT sysres AND aux;

    % PORT CARD 0 %
    % G-LINK 1 %


    clk_bitpatt_dat1_gl_1 = vwrt & vad10 & !vad9 & !pc2 & !pc0 & gl &
                            !vad5 & !vad4 & !vad3 & vad2 & !vad1;
    clk_bitpatt_dat1_gl_1.clk = ds;
    clk_bitpatt_dat1_gl_1.clrn = NOT sysres AND aux;
```

```
clk_bitpatt_dat2_gl_1 = vwrt & vad10 & !vad9 & !pc2 & pc0 & gl
& !vad5 & !vad4 & !vad3 & vad2 & vad1;
clk_bitpatt_dat2_gl_1.clk = ds;
clk_bitpatt_dat2_gl_1.clrn = NOT sysres AND aux;

% PORT CARD 1 %
% G-LINK 0 %


clk_bitpatt_dat1_gl_2 = vwrt & vad10 & !vad9 & !pc2 & pc0 & !gl &
                        !vad5 & !vad4 & vad3 & !vad2 & !vad1;
clk_bitpatt_dat1_gl_2.clk = ds;
clk_bitpatt_dat1_gl_2.clrn = NOT sysres AND aux;


clk_bitpatt_dat2_gl_2 = vwrt & vad10 & !vad9 & !pc2 & pc0 & !gl &
                        !vad5 & !vad4 & vad3 & !vad2 & vad1;
clk_bitpatt_dat2_gl_2.clk = ds;
clk_bitpatt_dat2_gl_2.clrn = NOT sysres AND aux;


% PORT CARD 1 %
% G-LINK 1 %


clk_bitpatt_dat1_gl_3 = vwrt & vad10 & !vad9 & !pc2 & pc0 & gl &
                        !vad5 & !vad4 & vad3 & vad2 &!vad1;
clk_bitpatt_dat1_gl_3.clk = ds;
clk_bitpatt_dat1_gl_3.clrn = NOT sysres AND aux;


clk_bitpatt_dat2_gl_3 = vwrt & vad10 & !vad9 & !pc2 & pc0 & gl &
                        !vad5 & !vad4 & vad3 & vad2 & vad1;
clk_bitpatt_dat2_gl_3.clk = ds;
clk_bitpatt_dat2_gl_3.clrn = NOT sysres AND aux;


END;
```

```
TITLE "Silicon Acquisition and Readout Module";

%-----------------------------------------------------

        D. Mendoza & Marcelo Mendes
        Fermilab
        10/11/93
-----------------------------------------------------%


%               * The SAR module will use the half lower section
                  of the 64K addresable space. It means that the
                  addressing spaces is now:

                  From 480000 to 487FFF

                  It can be acomplished by shifting one spot bits A15
                  through A1.  %

DESIGN IS "boardsel"
BEGIN
 DEVICE IS "EPM7032LC44-10"
 BEGIN
  vad11 @ 21, vad12 @ 20, vad13 @ 19, vad14 @ 18, vad15 @ 17,
  vad16 @ 16, vad17 @ 14, vad18 @ 13, vad19 @ 12,
  vad20 @ 11, vad21 @ 9,  vad22 @ 8,  vad23 @ 7,
  vam0 @ 29, vam1 @ 31 , vam2 @ 32, vam3 @ 33,
  vam4 @ 34, vam5 @ 36 , vas @ 6, vberr @ 5,
  gad0 @ 25, gad1 @ 26, gad2 @ 27, gad3 @ 28            :INPUT;
  bdsel @ 39                                             :OUTPUT;
 END;
END;

SUBDESIGN boardsel
(
        vad[23..11]         : INPUT;      % Address bus, bits 12 to 23          %
        vam[5..0]           : INPUT;      % Add. modifier bus                   %
        vas                 : INPUT;      % Address strobe after the inverter   %
        vberr               : INPUT;      % Bus error after inverter            %
        gad[3..0]           : INPUT;      % Geographical address of the board   %
        bdsel               : OUTPUT;     % Board select signal                 %


)

VARIABLE

        aux                         :DFF;        % Auxiliar variable to clear bdsel  %
        bdsel                       :DFF;
```

```
sel_delay_pc0 = NOT (!pc2 & !pc0 & !vad10 & vad9 & !gl &
                        !hdi & !vad5 & !vad4 & !vad3 & !vad2 & vad1);

sel_delay_pc1 = NOT (!pc2 & pc0 & !vad10  & vad9 & !gl &
                      !hdi & !vad5 & !vad4 & !vad3 & !vad2 & vad1);

sel_delay_pc2 = NOT (pc2 & !pc0 & !vad10 & vad9 & !gl &
                      !hdi & !vad5 & !vad4 & !vad3 & !vad2 & vad1);

sel_delay_pc3 = NOT (pc2 & pc0 & !vad10 & vad9 & !gl &
                      !hdi & !vad5 & !vad4 & !vad3 & !vad2 & vad1);


en_fsm_pc0 = !pc2 & !pc0 & !vad10 & vad9 & !gl &
             !hdi & !vad5 & !vad4 & !vad3 & vad2 & !vad1;

en_fsm_pc1 = !pc2 & pc0 & !vad10 & vad9 & !gl &
             !hdi & ! vad5 & !vad4 & !vad3 & vad2 & !vad1;

en_fsm_pc2 = pc2 & !pc0 & !vad10 & vad9 & !gl &
             !hdi & !vad5 & !vad4 & !vad3 & vad2 & !vad1;

en_fsm_pc3 = pc2 & pc0 & !vad10 & vad9 & !gl &
               !hdi & !vad5 & !vad4 & !vad3 & vad2 & !vad1;

wr_delay= vwrt AND NOT (sel_delay_pc0 AND sel_delay_pc1 AND
        sel_delay_pc2 AND sel_delay_pc3 AND sel_delay_clk);
wr_delay.clk = ds;
wr_delay.clrn = NOT sysres AND aux;

rd_delay_aux = !vwrt AND NOT (sel_delay_pc0 AND sel_delay_pc1 AND
               sel_delay_pc2 AND sel_delay_pc3 AND sel_delay_clk);
rd_delay_aux.clk = ds;
rd_delay_aux.clrn = NOT sysres AND aux;
rd_delay = NOT rd_delay_aux;


END;
```

```
TITLE "Silicon Acquisition and Readout Module";

%-----------------------------------------------------

        D. Mendoza
        Fermilab
        02/02/94
-----------------------------------------------------%

DESIGN IS "gllevel"
BEGIN
 DEVICE IS "EPM7032LC44-10"
 BEGIN

        vad1 @ 7,
        vad2 @ 8,
        vad3 @ 9,
        vad4 @ 11,
         vad5 @ 12,
        vad10 @ 13,
        vad9 @ 14,
        vwrt @ 17,
        pc0 @ 6,
        pc2 @ 5,
        gl @ 4,
        hdi @ 41,
        sysres @ 20,
        vdtck_oen_in @ 44,
        bdsel @ 16,
        ds @ 27                                                    :INPUT;

        vdtck @ 18, vdtck_aux @ 28,
        clk_bitpatt_dat1_gl_0 @ 39,
        clk_bitpatt_dat1_gl_1 @ 38,
        clk_bitpatt_dat1_gl_2 @ 37,
        clk_bitpatt_dat1_gl_3 @ 36,
        clk_bitpatt_dat2_gl_0 @ 34,
        clk_bitpatt_dat2_gl_1 @ 33,
        clk_bitpatt_dat2_gl_2 @ 32,
        clk_bitpatt_dat2_gl_3 @ 31                                 :OUTPUT;

 END;

 CONNECTED_PINS
 BEGIN
        vdtck_oen_in            :INPUT;
        vdtck_aux               :OUTPUT;
 END;

END;
```

```
SUBDESIGN gllevel

(

        bdsel                          : INPUT;
        vwrt                           : INPUT;
        vad[5..1]                      : INPUT;
        vad10                          : INPUT;
        vad9                           : INPUT;
        pc0                            : INPUT;
        pc2                            : INPUT;
        gl                             : INPUT;
        hdi                            : INPUT;
        vdtck_oen_in                   : INPUT;
        sysres                         : INPUT;
        ds                             : INPUT;
        vdtck                          : OUTPUT;
        vdtck_aux                      : OUTPUT;
        clk_bitpatt_dat1_gl_0          : OUTPUT;
        clk_bitpatt_dat1_gl_1          : OUTPUT;
        clk_bitpatt_dat1_gl_2          : OUTPUT;
        clk_bitpatt_dat1_gl_3          : OUTPUT;
        clk_bitpatt_dat2_gl_0          : OUTPUT;
        clk_bitpatt_dat2_gl_1          : OUTPUT;
        clk_bitpatt_dat2_gl_2          : OUTPUT;
        clk_bitpatt_dat2_gl_3          : OUTPUT;

)

VARIABLE

        vdtck_aux              : DFF;
        clk_bitpatt_dat1_gl_0  : DFF;
        clk_bitpatt_dat1_gl_1  : DFF;
        clk_bitpatt_dat1_gl_2  : DFF;
        clk_bitpatt_dat1_gl_3  : DFF;
        clk_bitpatt_dat2_gl_0  : DFF;
        clk_bitpatt_dat2_gl_1  : DFF;
        clk_bitpatt_dat2_gl_2  : DFF;
        clk_bitpatt_dat2_gl_3  : DFF;
        vdtck_oen              : node;
        aux                    : node;
        aux0                   : node;
        aux1                   : node;
        aux2                   : node;
```

```
BEGIN

        aux.clk = NOT vas;
        aux.d = vcc ;

        bdsel.d = vcc;
        bdsel.clrn = NOT aux;
        aux.clrn = bdsel;

        bdsel.clk = vas AND (!vberr & (vad[23..15] == H"090") &
                    (vad[14..11] == gad[3..0]) &
                    (vam[5..0] == H"39" #  vam[5..0] == H"3b"
                     # vam[5..0] == H"3d" # vam[5..0] == H"3f"));

        % * H"090" first 9 bits of the address bus, represents the
        base address for the SAR module within the Silicon Upgrade electronics.
         * Address modifier H 39 - Standard nonpriviledged data access
         * Address modifier H 3B - Standard nonpriviledged block transfer
         * Address modifier H 3D - Standard supervisory data access
         * Address modifier H 3F - Standard supervisory block transfer   %

END;
```

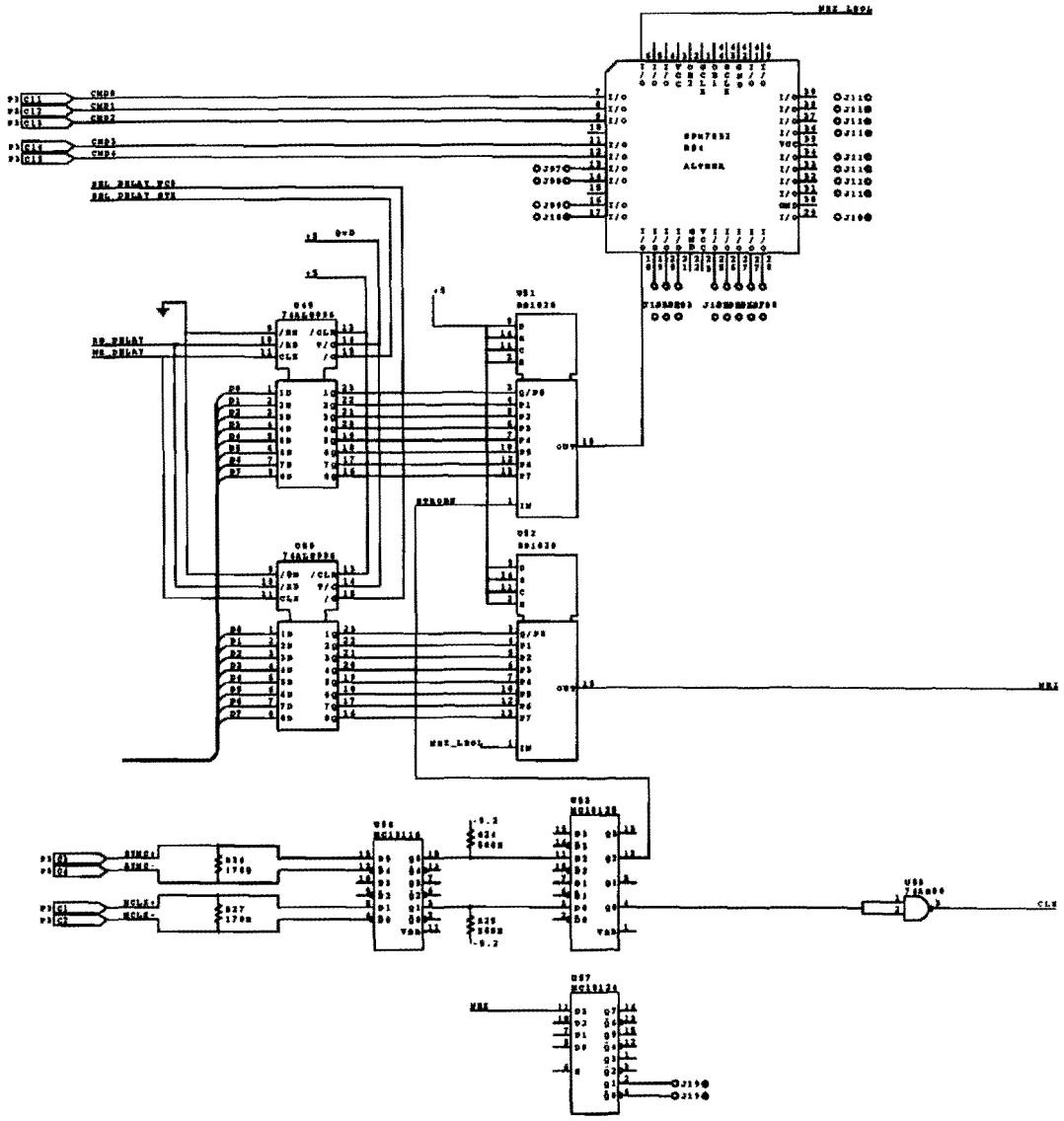
```
TITLE "Silicon Acquisition and Readout Module";

%-----------------------------------------------------

        D. Mendoza & Marcelo Mendes
        Fermilab
        10/11/93
-----------------------------------------------------%

DESIGN IS "bdlevel"
BEGIN

 DEVICE IS "EPM7032LC44-10"
 BEGIN

        vad1 @ 25,
        vad2 @ 24,
        vad3 @ 21,
         vad4 @ 17,
        vad5 @ 16,
        vad6 @ 14,
        vad7 @ 13,
        vad8 @ 12,
        vad9 @ 11,
        vad10 @ 9,
        bdsel @ 7,
        vwrt @ 5 ,
        vlword @ 6,
        vds0 @ 41,
        vds1  @ 40,
        sysres @ 36,
        vdtck_oen_in @ 44                                :INPUT;

        bufen0 @ 18,
        bufen1 @ 19,
        bufen2 @ 20,
        en_glwct_inv @ 8,
         str_cntl_reg @ 32,
        read_cntl_reg_inv @ 33 ,
        vdtck_aux @ 34,
        pc2 @ 28,
        pc0 @ 27,
        gl @ 29,
        hdi @ 31,
        vdtck @ 39,
         ds_out @ 38,
        ds_1_2 @ 37,
         read_buffer @ 4                          :OUTPUT;

 END;

 CONNECTED_PINS
 BEGIN
        vdtck_oen_in     :INPUT;
```

```
        vdtck_aux                   :OUTPUT;
 END;

END;


SUBDESIGN bdlevel
(
        vad[10..1]                  : INPUT;
        vwrt                        : INPUT;
        vlword                      : INPUT;
        vds0                        : INPUT;
        vds1                        : INPUT;
        bdsel                       : INPUT;
        vdtck_oen_in                : INPUT;
        sysres                      : INPUT;
        vdtck_aux                   : OUTPUT;
        bufen[2..0]                 : OUTPUT;
        en_glwct_inv                : OUTPUT;
        read_cntl_reg_inv:           OUTPUT;
        str_cntl_reg                : OUTPUT;
        ds_out                      : OUTPUT;
        ds_1_2                      : OUTPUT;
        pc2                         : OUTPUT;
        pc0                         : OUTPUT;
        gl                          : OUTPUT;
        hdi                         : OUTPUT;
        vdtck                       : OUTPUT;
        read_buffer                 : OUTPUT;
)

VARIABLE

        vdtck_aux                   :DFF;
        bufen[2..0]                 :node;
        en_glwct                    :DFF;
        pc2                         :DFF;
        str_cntl_reg_aux            :DFF;
        read_cntl_reg               :DFF;
        pc0                         :DFF;
        gl                          :DFF;
        hdi                         :DFF;
        ds                          :DFF;
        read_buffer                 :DFF;
        str_cntl_reg                :node;
        ds_out                      :node;
        vdtck_oen                   :node;
        ds_1_2                      :node;
        aux                         :node;
        aux0                        :node;
        aux1                        :node;
        aux2                        :node;
        bufen0_inv                  :DFF;
        bufen1_inv                  :DFF;
```

```
        bufen2_inv              :DFF;

BEGIN

        DEFAULTS

        read_cntl_reg           = GND;
        bufen0_inv              = VCC;
        bufen1_inv              = VCC;
        bufen2_inv              = VCC;


        END DEFAULTS;

        aux0 = LCELL ( NOT ( vds0 AND vds1));
        aux1 = LCELL ( NOT aux0 );
        aux2 = LCELL ( NOT aux1);
        aux = LCELL ( aux2 # (vds0 AND vds1 ));


        vdtck_aux.d   = GND;
        vdtck_aux.clk = en_glwct # str_cntl_reg # read_cntl_reg ;
        vdtck_aux.prn = NOT sysres AND aux;

        vdtck = TRI ( vdtck_aux, vdtck_oen );
        vdtck_oen = NOT vdtck_oen_in ;

        bufen0_inv.d = VCC;
        bufen0_inv.clk=((!vds0 & vds1 & vad1 & !vlword) #          % byte 0 %
                        (vds0 &  vds1 & !vad1 & !vlword) #         % bytes 0-1 %
                        (vds0 &  vds1 & !vad1 & vlword));          % bytes 0-3 %
        bufen0_inv.clrn = NOT sysres AND aux;
        bufen0 = NOT bufen0_inv;

        bufen1_inv.d  = VCC;
        bufen1_inv.clk=  ((vds0 & vds1 & !vad1 & !vlword) #        % bytes 0-1 %
                        (vds0 & vds1 & !vad1 & vlword));           % bytes 0-3 %
        bufen1_inv.clrn = NOT sysres AND aux;
        bufen1 = NOT bufen1_inv;

        bufen2_inv.d = VCC;
        bufen2_inv.clk=(vds0 & vds1 & !vad1 & vlword);             % bytes 0-3       %
        bufen2_inv.clrn = NOT sysres AND aux;
        bufen2 = NOT bufen2_inv;

        ds.d = vds0 AND vds1;
        ds.clk = bufen0_inv # bufen1_inv # bufen2_inv;
        ds.clrn = NOT sysres AND aux;

        ds_out   = LCELL(ds);
        ds_1_2 = LCELL(vds0 AND vds1);

        pc0.d = vad8;
```

```
pc0.clk = bdsel;
pc0.clrn = NOT sysres AND aux;

pc2.d = vad9;
pc2.clk = bdsel;
pc2.clrn = NOT sysres AND aux;

gl.d = vad7;
gl.clk = bdsel;
gl.clrn = NOT sysres AND aux;

hdi.d = vad6;
hdi.clk = bdsel;
hdi.clrn = NOT sysres AND aux;

en_glwct.d = vcc ;
en_glwct.clk =  vds0 AND vds1 AND bdsel & !vwrt & (vad[10..1] == H"002") ;
en_glwct.clrn = NOT sysres AND aux;
en_glwct_inv = NOT ( en_glwct );

read_buffer.d = vcc ;
read_buffer.clk =  bdsel & !vwrt & (vad[10..1] == H"004");
read_buffer.clrn = NOT sysres AND LCELL(LCELL(vdtck_aux));


str_cntl_reg_aux.d = bdsel & vwrt & (vad[10..1] == H"001");
str_cntl_reg_aux.clk =      ds;
str_cntl_reg_aux.clrn = NOT sysres AND aux;
str_cntl_reg = LCELL(LCELL ( str_cntl_reg_aux ));

read_cntl_reg.d = bdsel & !vwrt &  (vad[10..1] == H"001");
read_cntl_reg.clk = ds;
read_cntl_reg.clrn = NOT sysres AND aux;
read_cntl_reg_inv = LCELL (NOT ( read_cntl_reg )) ;


END;
```

# Transmitter Section




SHEET No 4 OF 5

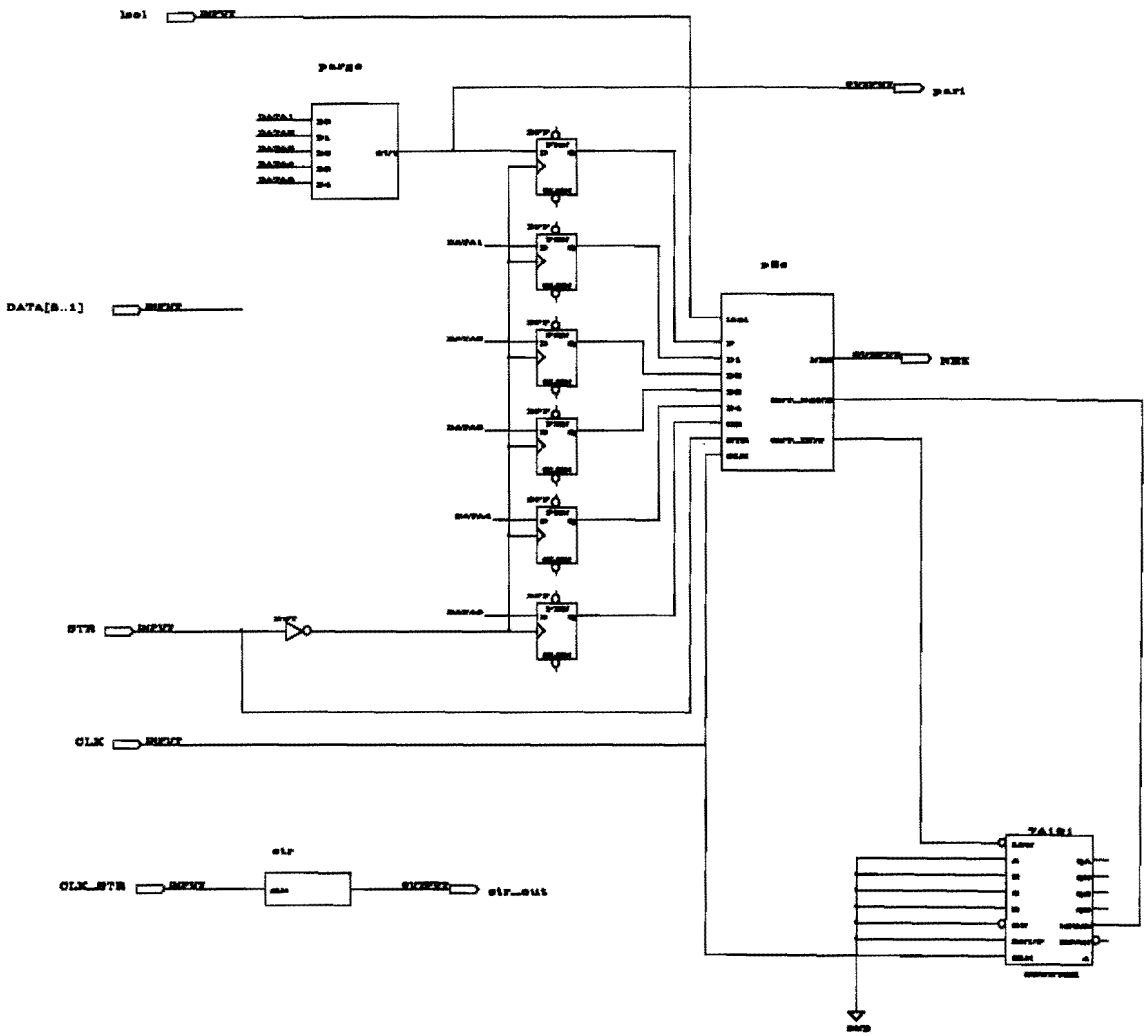
| ITEM | PART | DESCRIPTION OR SIZE | QTY |
|---|---|---|---|
| PART LIST | | | |

| | | | |
|---|---|---|---|
| DRAWN BY | MENDOZA | 01.31.94 |
| CHECKED BY | MENDOZA | 01.31.94 |
| APPROVED BY | | |
| DATE | | |
| MATERIAL | | |

FERMI NATIONAL ACCELERATOR LABORATORY

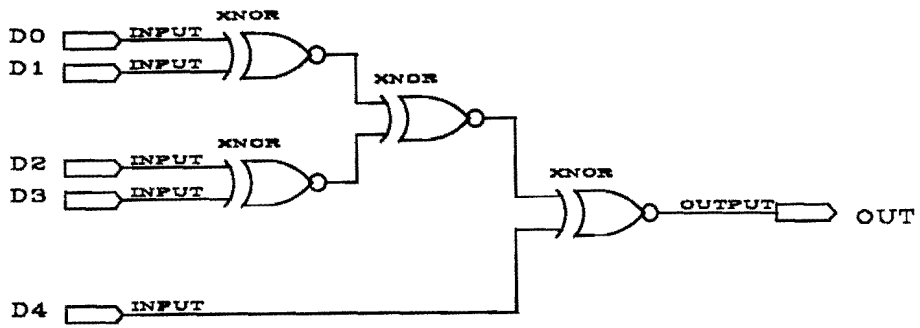
DO NI OPERATOR - SAR MODULE
TRANSMITTER SECTION

SCALE | STAMPS | DESIGN NUMBER | REV




| | |
|---|---|
| TITLE | TRANSMITTER |
| COMPANY | FERMILAB |
| DESIGNER | D. MENDOZA |
| SIZE D | NUMBER 1.00 REV A |
| DATE 8:50p 5-03-1994 | SHEET 1 OF 1 |

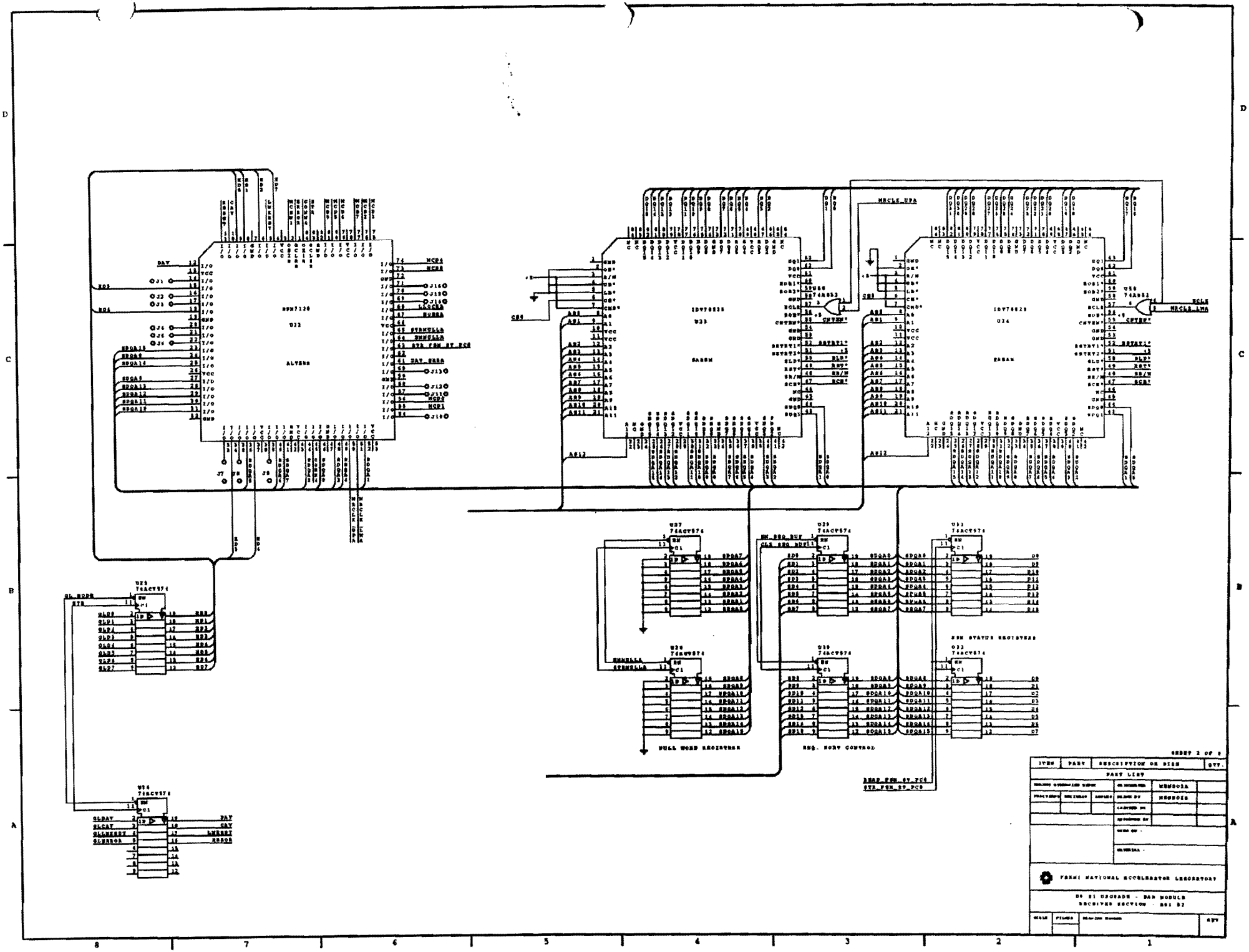
PARGE



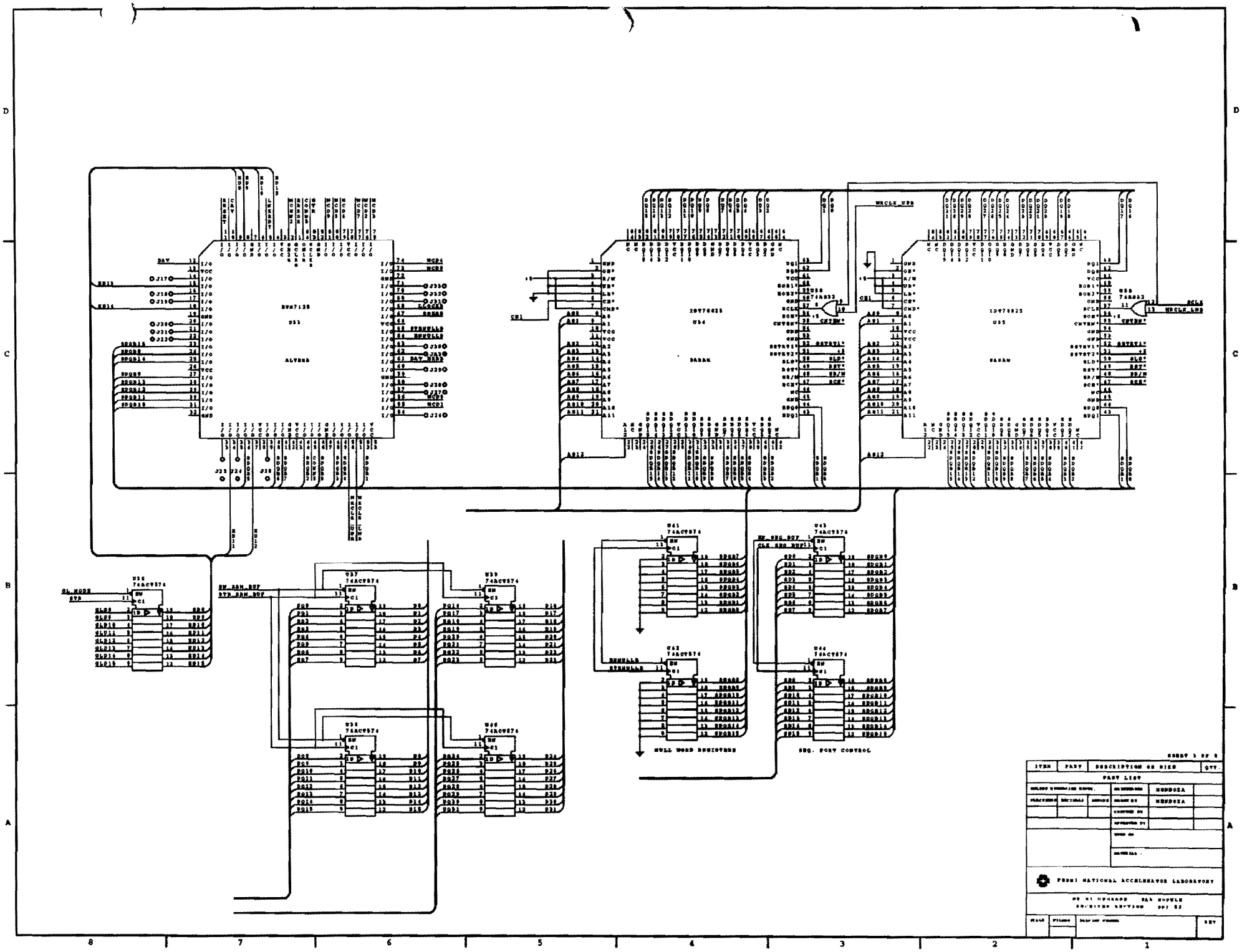

| | | | |
|---|---|---|---|
| TITLE | ODD PARITY GENERATOR | | |
| COMPANY | FERMILAB | | |
| DESIGNER | D. MENDOZA | | |
| SIZE B | NUMBER 1.00 | | REV A |
| DATE 3:51p 5-09-1994 | | SHEET 1 OF 1 | |

```
SUBDESIGN 'p2s'
(
        str,d[4..1],cr,p,clk,lsol,cnt_done        : INPUT;
        nrz, cnt_init                             : OUTPUT;
)

VARIABLE

        ss : MACHINE  WITH STATES ( s0,s1,s2,s3,s4,s5,s6,s7);


BEGIN

DEFAULTS

        cnt_init = VCC;

END DEFAULTS;

        ss.clk = clk;

        CASE ss IS

                WHEN s0 =>

                        IF lsol THEN
                                cnt_init = GND;
                                ss = s7;
                                ELSE

                                        IF str THEN
                                                nrz = GND;
                                                ss = s0;
                                        ELSIF !str THEN
                                                nrz = VCC;
                                                ss = s1;
                                        END IF;
                        END IF;


                WHEN s1 =>

                        IF cr THEN
                                nrz = VCC;
                                ss = s2;
                        END IF;

                        IF !cr THEN
                                nrz = GND;
                                ss = s2;
                        END IF;
```

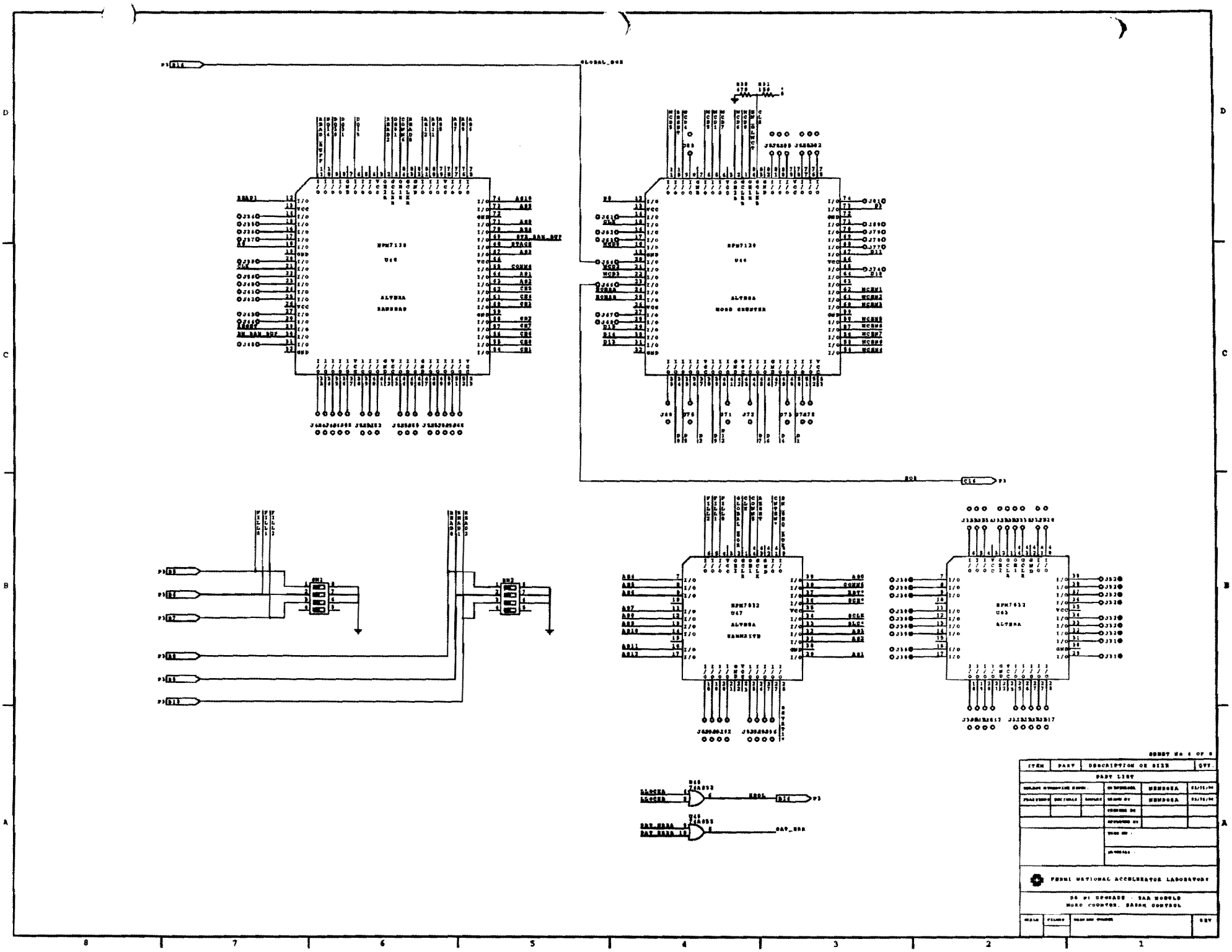
# Receiver Section



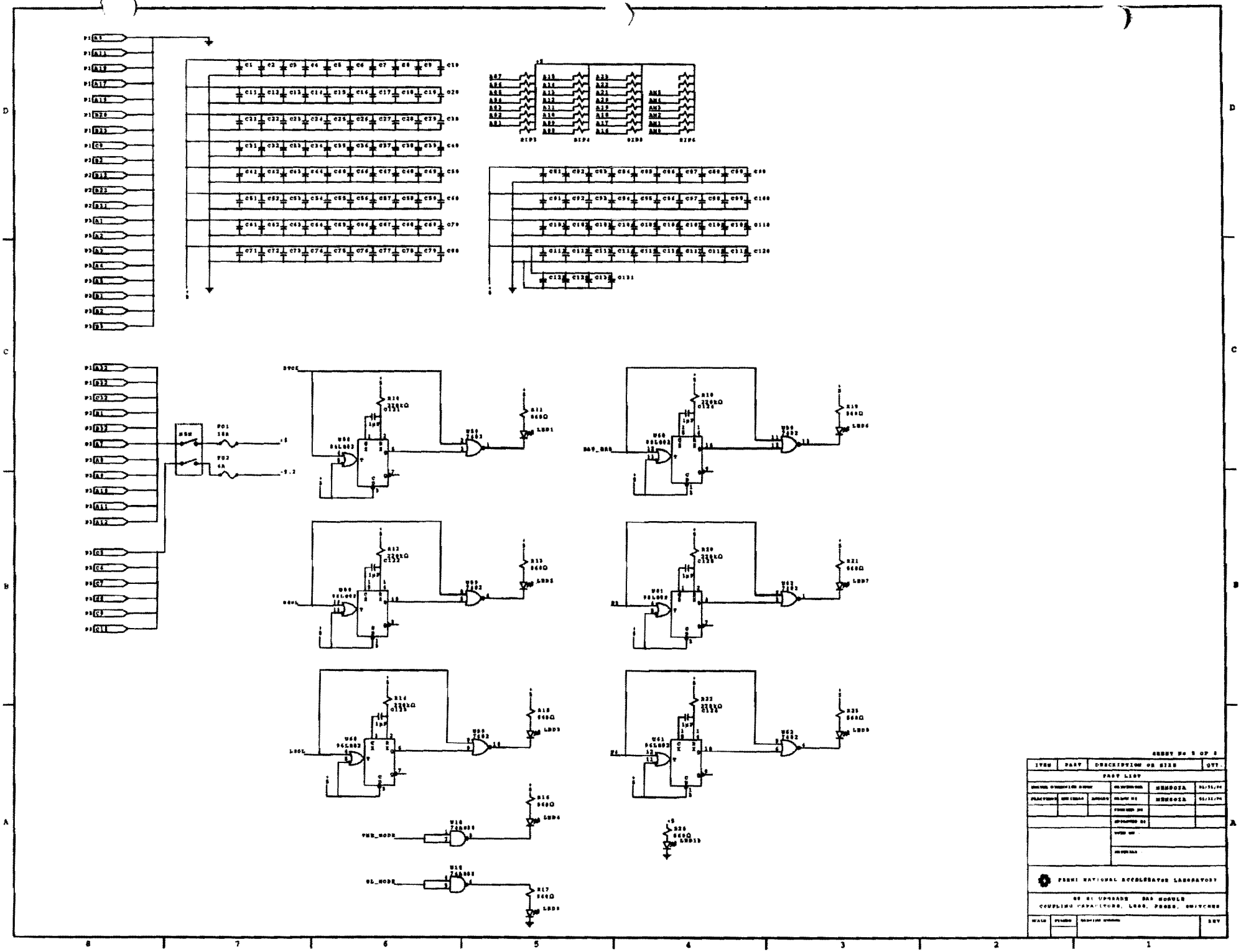
SHEET 2 OF 8
ITEM
PART
SUBSCRIPTION OR SIZE
QTY.
PART LIST
MENDOZA
MENDOZA
FERMI NATIONAL ACCELERATOR LABORATORY
D0 S1 UPGRADE - BAR MODULE
RECEIVER SECTION - RS1 S2
SCALE
FILMED
REV

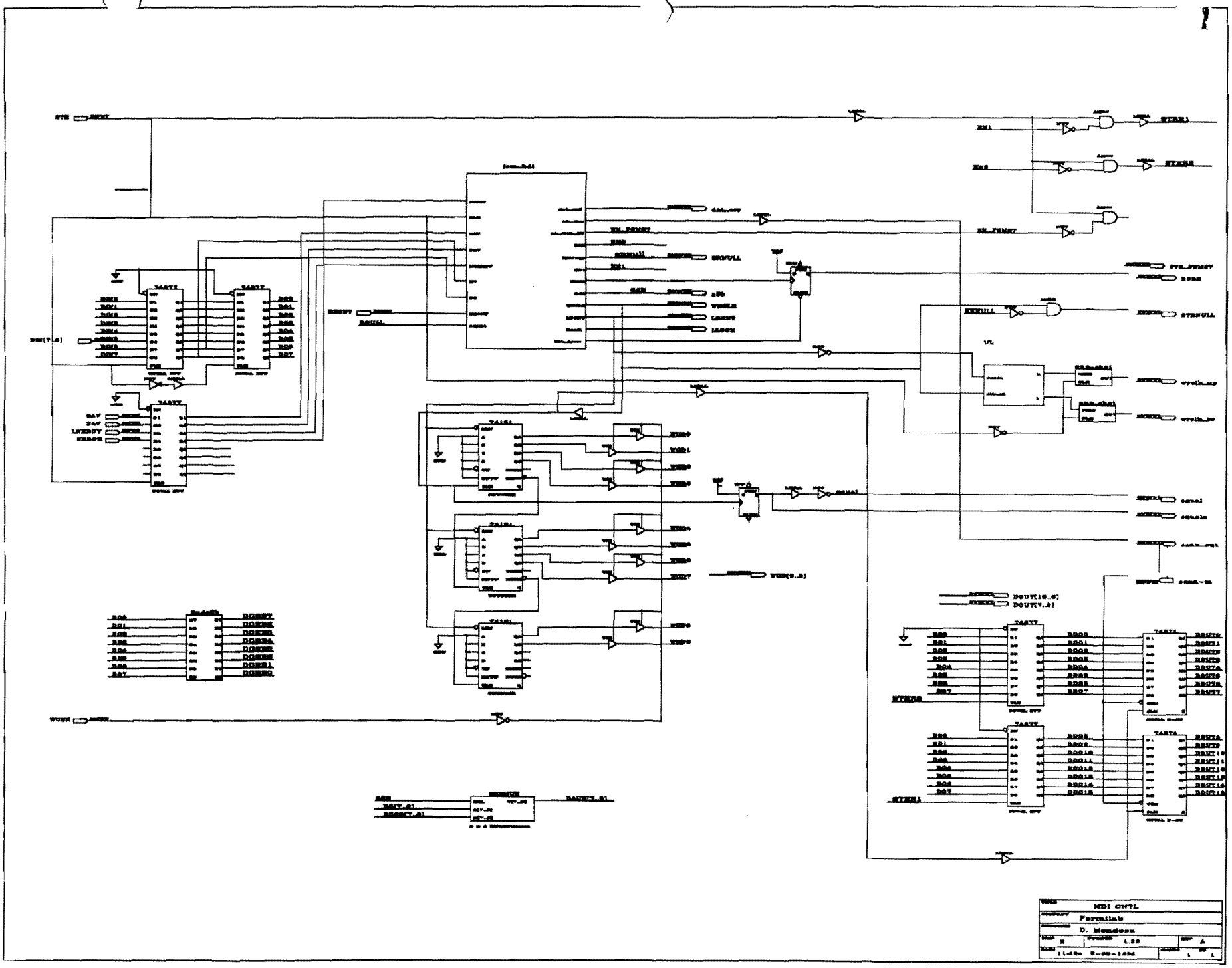


ALTERA
SARAM
SARAM
74ACT574
MULL WORD REGISTERS
SEQ. PORT CONTROL
ITEM
PART
DESCRIPTION OR SIZE
QTY.
PART LIST
MENDOZA
MENDOZA
FERMI NATIONAL ACCELERATOR LABORATORY

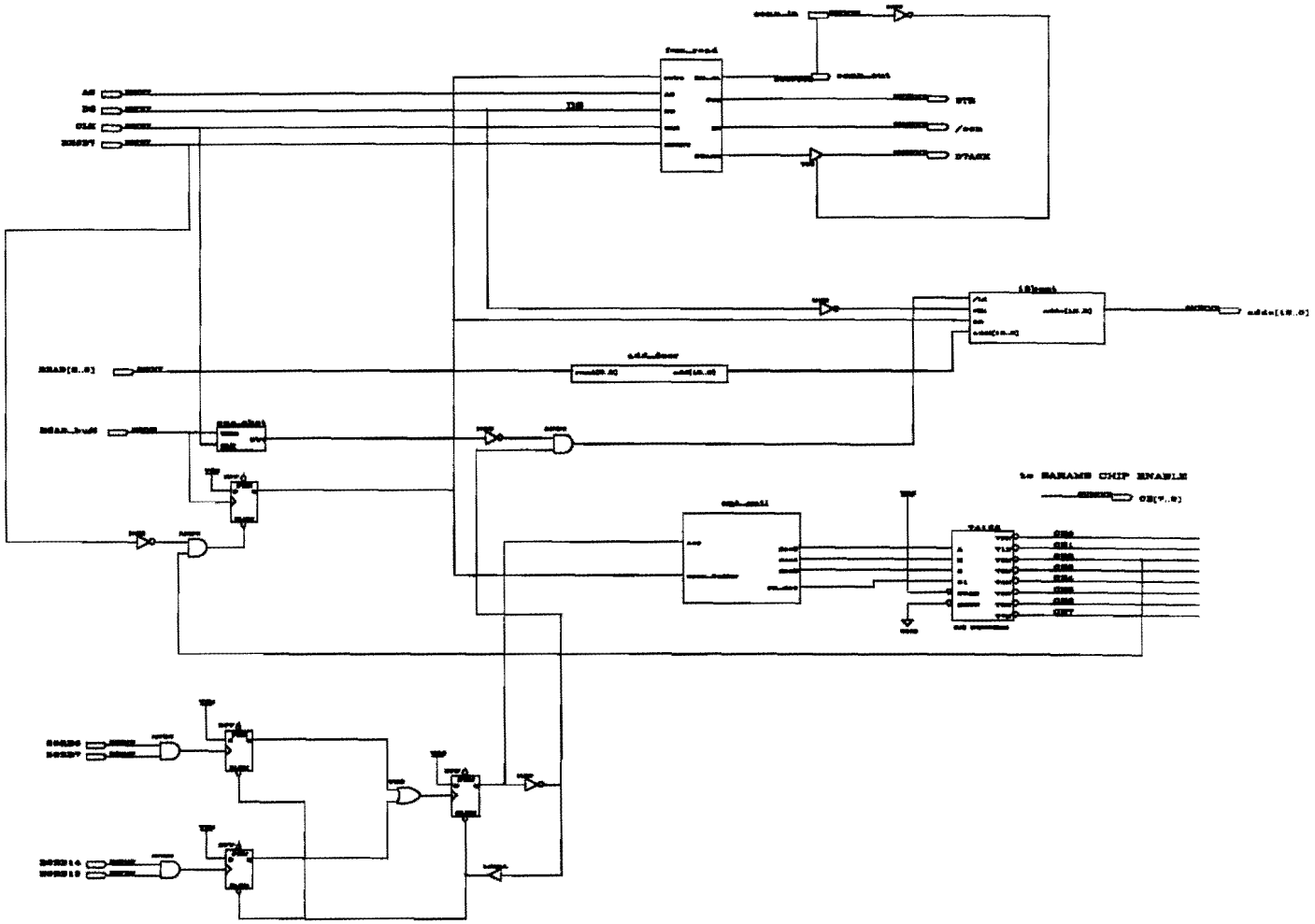


SHEET No 4 OF 8
ITEM
PART
DESCRIPTION OR SIZE
QTY.
PART LIST
FERMI NATIONAL ACCELERATOR LABORATORY
DO PI UPGRADE - SAM MODULE
MOD COUNTER, BASEM CONTROL


SHEET No 5 OF 5
ITEM
PART
DESCRIPTION OR SIZE
QTY.
PART LIST
FERMI NATIONAL ACCELERATOR LABORATORY
COUPLING CAPACITORS, LEDS, PROBE, SWITCHES


MDI CNTL
Fermilab
D. Meadows

```
SUBDESIGN 'ul'
(
        clk_ul,reset            : INPUT;
        u,l                     : OUTPUT;
)

VARIABLE

        ss : MACHINE  WITH STATES ( s0,s1,s2);

BEGIN

DEFAULTS

        l = GND;
        u = GND;

END DEFAULTS;

        ss.clk = clk_ul;
        ss.reset = reset;

        CASE ss IS


                WHEN s0 =>


                                ss= s1;

                WHEN s1 =>

                        l = VCC;

                                ss= s2;

                WHEN s2 =>

                        u = VCC;
                        ss= s1;


        END CASE;
END;
```



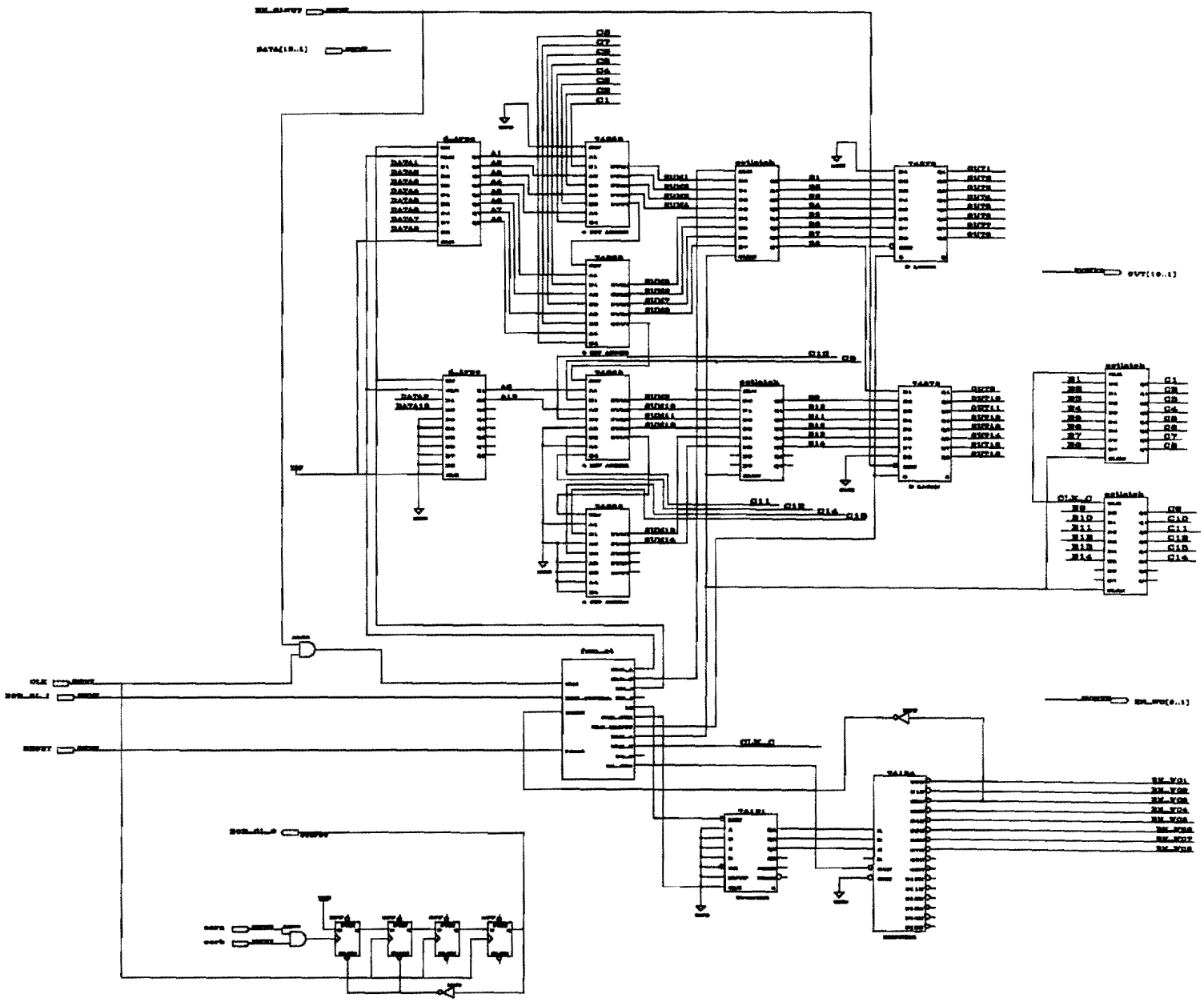

| | |
|---|---|
| TITLE | SARAM READ CONTROL |
| COMPANY | FERMILAB |
| DESIGNER | D. MENDOZA |
| SIZE | B |
| NUMBER | 1.00 |
| REV | A |
| DATE | 10:05p 8-30-1994 |
| SHEET | 1 OF 4 |

```
        WHEN s2 =>

                IF d4 THEN
                        nrz = VCC;
                        ss = s3;
                END IF;

                IF !d4 THEN
                        nrz = GND;
                        ss = s3;
                END IF;

        WHEN s3 =>

                IF d3 THEN
                        nrz = VCC;
                        ss = s4;
                END IF;

                IF !d3 THEN
                        nrz = GND;
                        ss = s4;
                END IF;

        WHEN s4 =>

                IF d2 THEN
                        nrz = VCC;
                        ss = s5;
                END IF;

                IF !d2 THEN
                        nrz = GND;
                        ss = s5;
                END IF;


        WHEN s5 =>

                IF d1 THEN
                        nrz = VCC;
                        ss = s6;
                END IF;

                IF !d1 THEN
                        nrz = GND;
                        ss = s6;
                END IF;

        WHEN s6 =>
```

```
                    IF p THEN
                           nrz = VCC;
                           ss = s0;
                    END IF;

                    IF !p THEN
                           nrz = GND;
                           ss = s0;
                    END IF;

             WHEN s7 =>

                    IF cnt_done THEN
                           ss= s0;
                    ELSE
                           ss= s7;
                    END IF;

       END CASE;

END;
```

GLOBAL WORD COUNTER

DEC 09/95

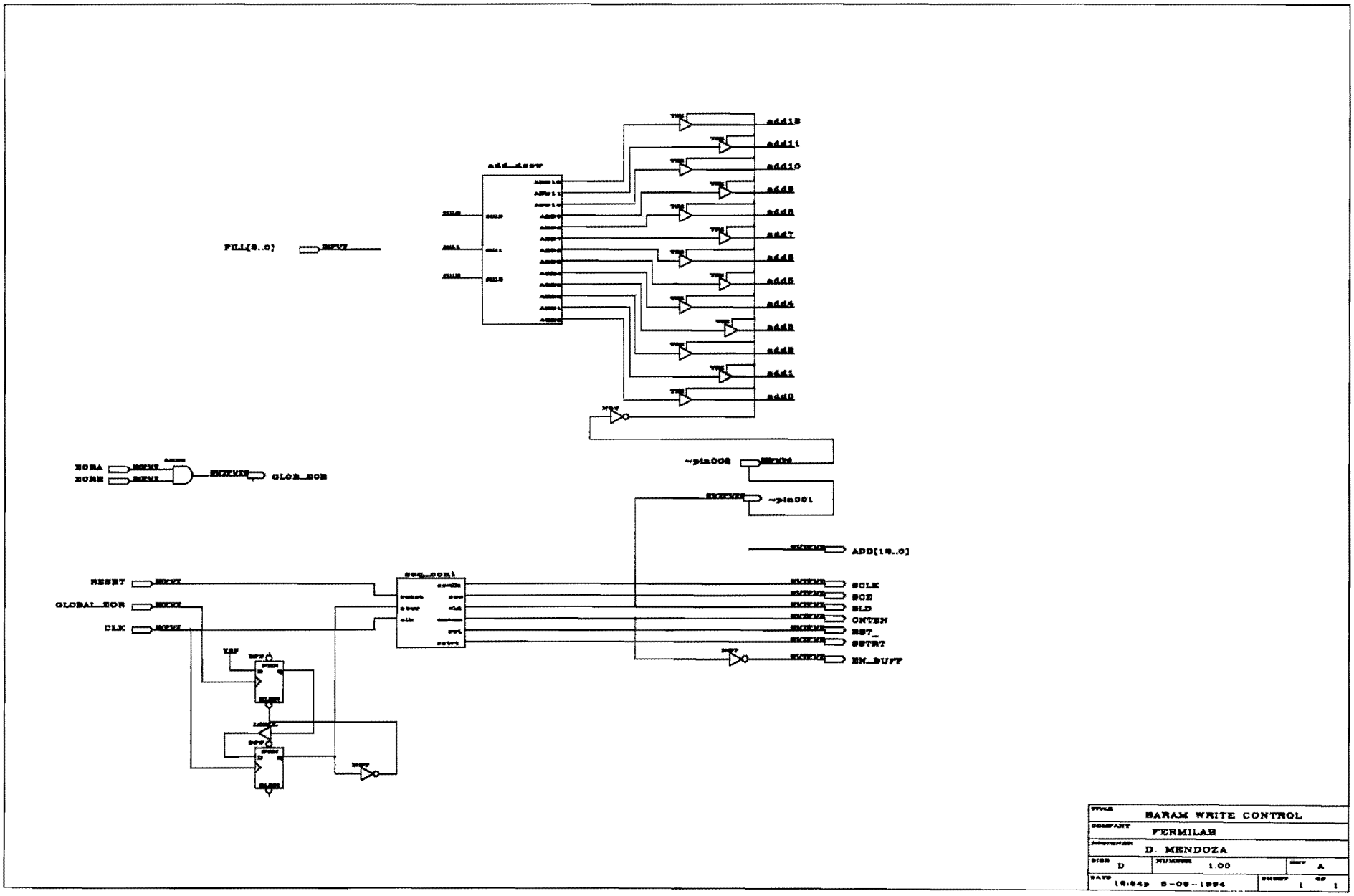


| | |
|---|---|
| GLOBAL WORD COUNTER | |
| FERMILAB | |
| D. MENDOZA | |
| 1.00 | A |
| 8:11p 8-09-1996 | 1 / 1 |

```
SUBDESIGN 'fsm_ct'
(
        EOR_GLOBAL,CLK,DONE,RESET                                          : INPUT;
        clk_a,CLK_B,CLK_C,EN_A,EN_B,EN_C,LD,CLK_CTR,CLK_GLWCT,
        CLR_A,en_dec                                                       : OUTPUT;
)


VARIABLE

        ss : MACHINE WITH STATES (
                                                idle,
                                                clear,
                                                clear1,
                                                enaa,
                                                getword,
                                                enab,
                                                regwordb,
                                                enac,
                                                regwordc,
                                                finish);


BEGIN

DEFAULTS

clk_a = GND;
clk_b = GND;
clk_c = GND;
en_a = VCC;
en_b = VCC;
en_c = VCC;
ld = VCC;
clk_ctr = GND;
clk_glwct = GND;
clr_a = VCC;
en_dec = VCC;

END DEFAULTS;

        ss.clk = clk;
        ss.reset = reset;

        CASE ss IS

        WHEN idle =>


                IF EOR_GLOBAL THEN
                        ss = clear;
```

```
            ELSE
                    ss = idle;
            END IF;

        WHEN clear =>

            en_dec = GND;
            LD              = GND;
            CLR_A   = GND;

            ss = clear1;

        WHEN clear1 =>

            en_dec = GND;
            LD              = GND;
            CLK_CTR         = VCC;

            ss = enaa;

        WHEN enaa =>

            EN_A    = GND;
            en_dec = GND;

            ss = getword;

        WHEN getword =>


            clk_a           = VCC;
            EN_A    = GND;
            CLK_CTR         = VCC;
            en_dec = GND;


            ss = enab;

        WHEN enab =>

            EN_B    = GND;
            en_dec = GND;

            ss = regwordb;


        WHEN regwordb =>


            CLK_B   = VCC;
            EN_B    = GND;
            en_dec = GND;
```

```
            ss= enac;

        WHEN enac =>

            EN_C    = GND;
            LD              = VCC;
            en_dec = GND;

            ss = regwordc;

        WHEN regwordc =>

            CLK_C   = VCC;
            EN_C    = GND;
            LD              = VCC;
            en_dec = GND;

        IF done THEN
            ss = finish;
        ELSE
            ss = enaa;
        END IF;

WHEN finish =>

            CLK_GLWCT= VCC;
            ld = gnd;
            ss = idle;

END CASE;

END;
```


add_decr
PILI[8..0]
add12
add11
add10
add9
add8
add7
add6
add5
add4
add3
add2
add1
add0
~pin008
~pin001
ADD[12..0]
EORA
EORB
GLOB_EOR
RESET
GLOBAL_EOR
CLK
SCLK
SOE
SLD
CNTEN
RST_
SSTRT
EN_BUFF
TITLE
SARAM WRITE CONTROL
COMPANY
FERMILAB
DRAWN BY
D. MENDOZA
SIZE
D
REVISION
1.00
SHEET
A
DATE
12:34p 5-09-1994
SHEET
1 OF 1

```
SUBDESIGN 'seq_cont'
(
        reset,eoer,clk                                          : INPUT;
        seclk,sce,sld,cnten,rst,sstrt                   : OUTPUT;
)

VARIABLE

        ss: MACHINE WITH STATES ( idle,
                                                        reset1,
                                                        reset2,
                                                        aplc,
                                                        datain,
                                                        naplc,
                                                        wait,
                                                        endsam,
                                                        ldadd,
                                                        wait2,
                                                        rstclk,
                                                        stadd);

BEGIN

DEFAULTS

                sstrt   = VCC;
                cnten   = GND;
                seclk   = GND;
                rst             = VCC;
                sld             = VCC;
                sce     = VCC;

END DEFAULTS;

ss.clk = clk;

        CASE ss IS

                WHEN idle =>

                        IF reset THEN
                                ss = reset1;
                        ELSIF eoer THEN
                                ss = aplc;
                        ELSE
                                ss = idle;
                        END IF;

                WHEN reset1 =>

                        cnten = VCC;
                        rst = GND;
```

```
                sld = VCC;
                sce = GND;
                ss = reset2;

        WHEN reset2 =>

                cnten = VCC;
                rst = VCC;
                sld = VCC;
                seclk = GND;
                ss = aplc;

        WHEN aplc =>

                cnten = VCC;
                sld = GND;
                ss = datain;

        WHEN datain =>

                cnten = VCC;
                rst = VCC;
                seclk = VCC;
                sld = GND;
                ss = naplc;

        WHEN naplc =>

                cnten = VCC;
                seclk = VCC;
                ss = wait;

        WHEN wait =>

                cnten = VCC;
                seclk = VCC;
                ss= endsam;

        WHEN endsam =>

                cnten = VCC;
                ss = ldadd;

        WHEN ldadd=>

                cnten = VCC;
                seclk = VCC;
                ss = wait2;

        WHEN wait2 =>

                cnten = VCC;
```

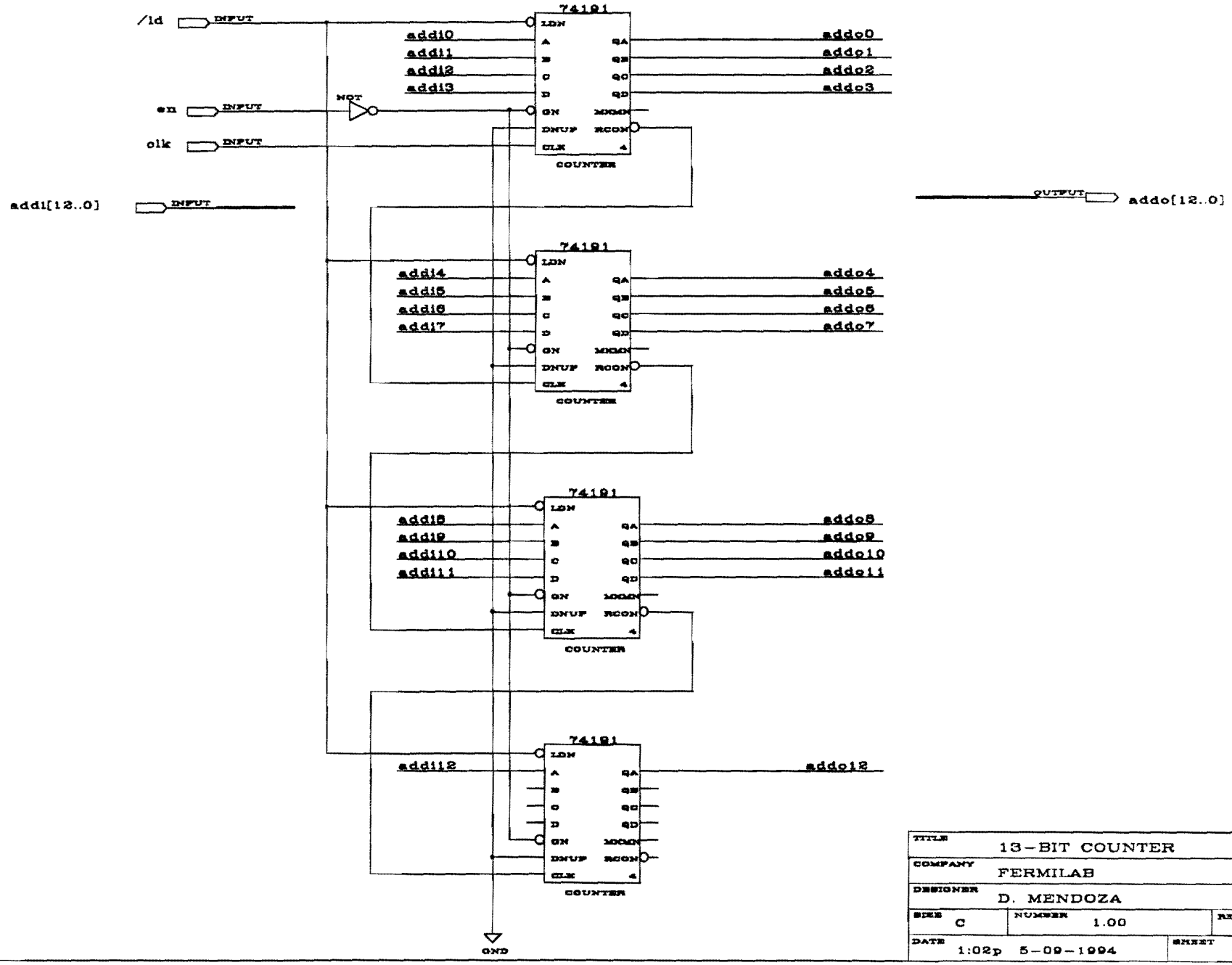
```
                    seclk = VCC;
                    ss = rstclk;

            WHEN rstclk =>

          sstrt = GND;
                    cnten = VCC;
                    ss = stadd;

            WHEN stadd =>

                    sstrt = GND;
                    cnten = VCC;
                    seclk = VCC;
                    ss = idle;


      END CASE;

END;
```

```
SUBDESIGN 'add_decw'
(
        fill[2..0]          : INPUT;
        add[12..0]          : OUTPUT;
)


BEGIN

        TABLE

                fill[2..0]      => add[12..0];

                        B"000"    =>  H"0001";
                        B"001"    =>  H"0401";
                        B"010"    =>  H"0801";
                        B"011"    =>  H"0C01";
                        B"100"    =>  H"1001";
                        B"101"    =>  H"1401";
                        B"110"   =>  H"1801";
                        B"111"    =>  H"1C01";

        END TABLE;
END;
```

```
SUBDESIGN 'fsm_read'
(
        reset,ds,as,rebu,clk                    : INPUT;
        dtack,str,en, en_dt                             : OUTPUT;
)

VARIABLE

        ss: MACHINE WITH STATES ( idle,
                                                wait_ds,
                                                wait_25n,
                                                strobe,
                                                dtck);

BEGIN
DEFAULTS

        dtack = VCC;
        en = VCC;
        en_dt = VCC;

END DEFAULTS;


ss.clk                  = clk;
ss.reset        = reset;

        CASE ss IS

                WHEN idle =>

                        IF rebu AND !ds THEN
                                ss = wait_ds;
                        ELSIF ds AND as AND rebu THEN
                                ss = wait_25n;
                        ELSE
                                ss = idle;
                        END IF;

                WHEN wait_ds =>

                        IF ds  THEN
                                ss = wait_25n;
                        ELSIF !ds THEN
                                ss = wait_ds;
                        END IF;

                WHEN wait_25n =>

                                ss= strobe;
                                en = GND;
```

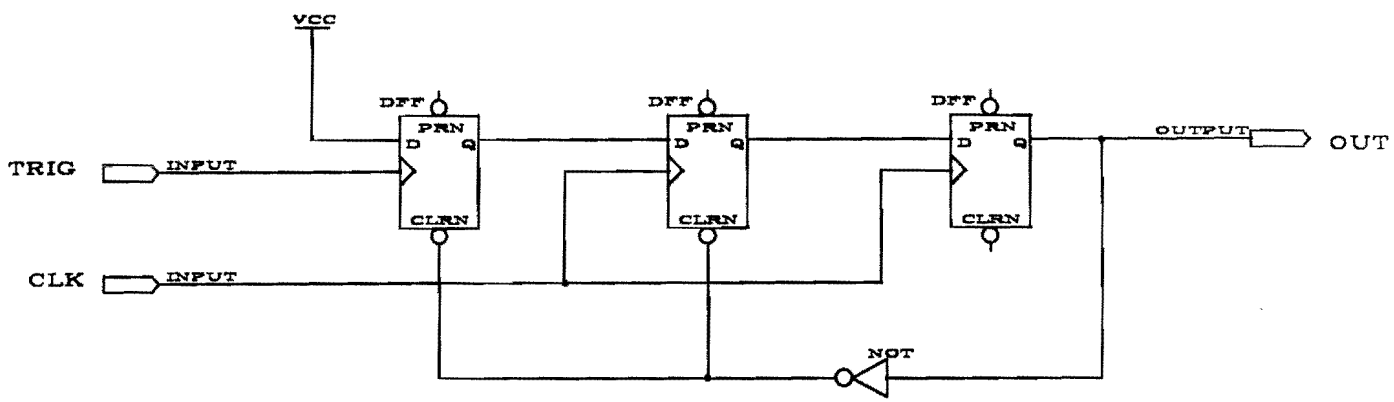
```
            WHEN strobe =>

                    str = VCC;
                    en = GND;
                    en_dt = GND;

                    IF ds  THEN
                           ss = dtck;
                    ELSE
                           ss = idle;
                    END IF;

            WHEN dtck =>

                    dtack = GND;
                    en = GND ;
                    en_dt = GND;

                    IF ds  THEN
                           dtack = GND;
                           en = GND;
                           en_dt = GND;
                           ss = dtck;

                    ELSIF !ds THEN
                           dtack = VCC;
                           ss = idle;
                    END IF;

            END CASE;

END;
```

```
SUBDESIGN cnt_cntl

(

        read_buffer                     : INPUT;
        eor                                     : INPUT;
        en_dec                          : OUTPUT;
        dec[2..0]                       : OUTPUT;


)

VARIABLE

        dec[2..0]                       : dff;

BEGIN

DEFAULTS

        dec[] = GND;
        en_dec = VCC;

END DEFAULTS;

        dec[2..0].clk = eor;

        IF read_buffer THEN
                en_dec= GND;

                IF !en_dec THEN
                        dec[]=dec[]+1;
                ELSE
                        dec[]=dec[];
                END IF;
        ELSE
                en_dec = VCC;
        END IF;

END;
```




| | | | |
|---|---|---|---|
| TITLE | 13-BIT COUNTER | | |
| COMPANY | FERMILAB | | |
| DESIGNER | D. MENDOZA | | |
| SIZE C | NUMBER 1.00 | | REV A |
| DATE 1:02p 5-09-1994 | | SHEET 1 | OF 1 |

```
SUBDESIGN 'add_decr'
(
        read[2..0]          : INPUT;
        add[12..0]          : OUTPUT;
)


BEGIN

        TABLE

                read[2..0]      => add[12..0];

                        B"000"    =>  H"0000";
                        B"001"    =>  H"0400";
                        B"010"    =>  H"0800";
                        B"011"    =>  H"0C00";
                        B"100"    =>  H"1000";
                        B"101"    =>  H"1400";
                        B"110"   =>  H"1800";
                        B"111"    =>  H"1C00";

        END TABLE;
END;
```




| | | | |
|---|---|---|---|
| TITLE | ONE SHOT | | |
| COMPANY | FERMILAB | | |
| DESIGNER | D. MENDOZA | | |
| SIZE B | NUMBER 1.00 | | REV A |
| DATE 2:07p 5-09-1994 | | SHEET 1 | OF 1 |

```
SUBDESIGN 'fsm_read'
(
        reset,ds,as,rebu,clk                    : INPUT;
        dtack,str,en, en_dt                             : OUTPUT;
)

VARIABLE

        ss: MACHINE WITH STATES ( idle,
                                                wait_ds,
                                                wait_25n,
                                                strobe,
                                                dtck);

BEGIN
DEFAULTS

        dtack = VCC;
        en = VCC;
        en_dt = VCC;

END DEFAULTS;


ss.clk                  = clk;
ss.reset        = reset;

        CASE ss IS

                WHEN idle =>

                        IF rebu AND !ds THEN
                                ss = wait_ds;
                        ELSIF ds AND as AND rebu THEN
                                ss = wait_25n;
                        ELSE
                                ss = idle;
                        END IF;

                WHEN wait_ds =>

                        IF ds  THEN
                                ss = wait_25n;
                        ELSIF !ds THEN
                                ss = wait_ds;
                        END IF;

                WHEN wait_25n =>

                                ss= strobe;
                                en = GND;
```

```
        WHEN strobe =>

            str = VCC;
            en = GND;
            en_dt = GND;

            IF ds  THEN
                   ss = dtck;
            ELSE
                   ss = idle;
            END IF;

        WHEN dtck =>

            dtack = GND;
            en = GND ;
            en_dt = GND;

            IF ds  THEN
                   dtack = GND;
                   en = GND;
                   en_dt = GND;
                   ss = dtck;

            ELSIF !ds THEN
                   dtack = VCC;
                   ss = idle;
            END IF;

        END CASE;

END;
```

# APPENDIX C
# Parts List

## Parts List*

| PART | | QTY |
|---|---|---|
| | | |
| TI 74LS240 | | 1 |
| TI 74F38 | | 1 |
| TI 74ACT573 | | 4 |
| TI 74ACT245 | | 4 |
| ALTERA EPM7032 | | 6 |
| ALTERA EPM7128 | | 10 |
| IDT 70825 - SARAM | | 16 |
| 74F14 | | 1 |
| 74ALS996 | | 4 |
| MOTOROLA MC10116 | | 1 |
| MOTOROLA MC10125 | | 1 |
| MOTOROLA MC10124 | | 1 |
| DALLAS DS1020 | | 3 |
| NATIONAL 100H600 | | 4 |
| TI 74ACT574 | | 75 |
| 96LS02 | | 3 |
| FINISAR FMT-8500 | | 4 |
| OPTICAL RECEIVER MODULE | | |
| HEWLETT PACKARD RX CHIPSET | | 4 |
| HDMP-1000 RX | | |
| 74AS00 | | 3 |
| 0.1 uF CAPACITORS | | 119 |
| 0.2 uF CAPACITORS | | 104 |
| 100 uF CAPACITORS | | 1 |
| TRANSORB | | 1 |
| 1 K RESISTORS | | 5 |
| 2.2K RESISTORS | | 2 |
| 96-PIN EUROCARD CONNECTORS | | 2 |
| 48-PIN CONNECTOR | | 1 |
| MAC II CONNECTOR | | 1 |

* The Parts List is yet incomplete. It does not include the components of the VSB interface and the Trigger Processor interface.