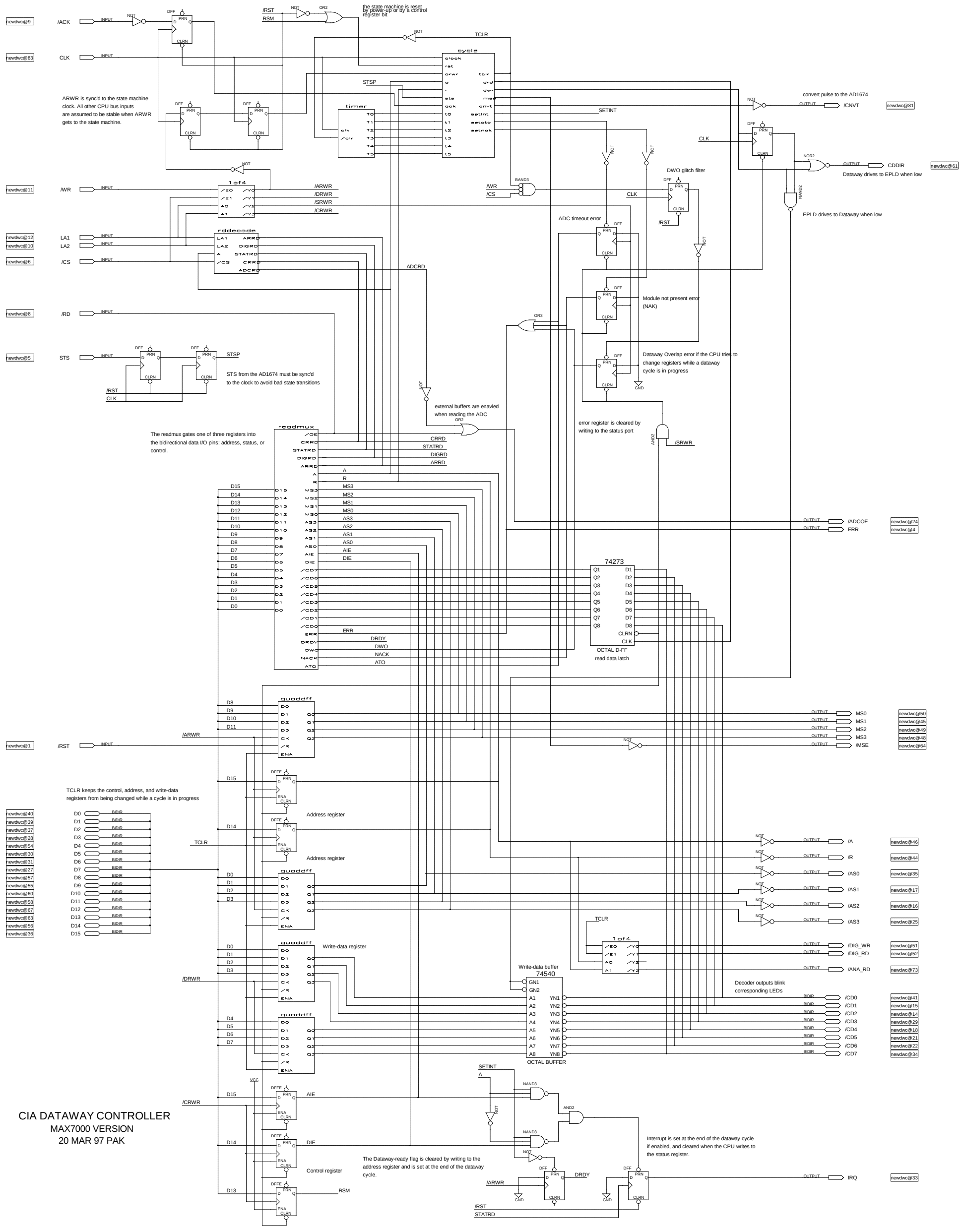

CIA DATAWAY CONTROLLER
MAX7000 VERSION
20 MAR 97 PAK
ARWR is sync'd to the state machine clock. All other CPU bus inputs are assumed to be stable when ARWR gets to the state machine.
STS from the AD1674 must be sync'd to the clock to avoid bad state transitions
The readmux gates one of three registers into the bidirectional data I/O pins: address, status, or control.
TCLR keeps the control, address, and write-data registers from being changed while a cycle is in progress
the state machine is reset by power-up or by a control register bit
convert pulse to the AD1674
Dataway drives to EPLD when low
EPLD drives to Dataway when low
DWO glitch filter
ADC timeout error
Module not present error (NAK)
Dataway Overlap error if the CPU tries to change registers while a dataway cycle is in progress
error register is cleared by writing to the status port
external buffers are enavled when reading the ADC
OCTAL D-FF read data latch
Address register
Write-data register
Write-data buffer
OCTAL BUFFER
Decoder outputs blink corresponding LEDs
Control register
The Dataway-ready flag is cleared by writing to the address register and is set at the end of the dataway cycle.
Interrupt is set at the end of the dataway cycle if enabled, and cleared when the CPU writes to the status register.
cycle
timer
readmux
74273
quoddff
1of4
rddecode
74540

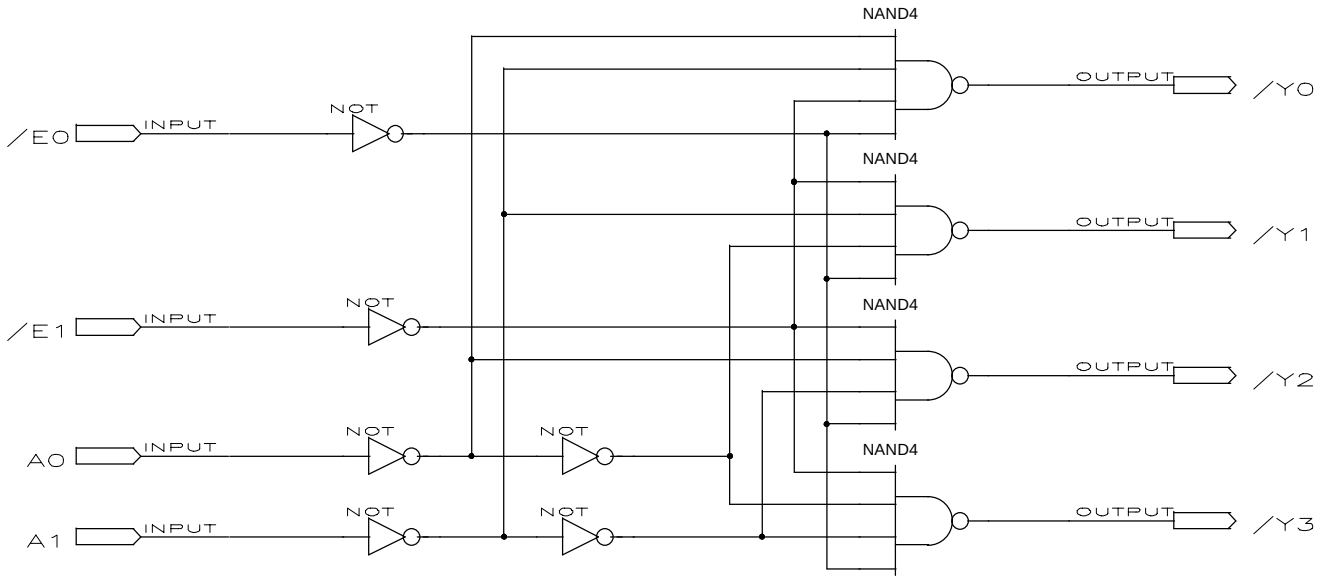
```
%****************************************
CIA Dataway Cycle Controller
20 March 1997 P. Kasley
Issue O
****************************************%

TITLE "CIA Dataway Cycle Controller";

% Cycle Timing                                          %
% Digital Read Cycle:                                   %
%                                                       %
%  time                        0us         2us          %
%  MODE SEL    ---------------____________---------     %
%  ACK sampled                            X             %
%                                                       %
%  Digital Write Cycle:                                 %
%                                                       %
%  time               0us 1us           3us   4us      %
%  WR DATA OE  -------________________________-----     %
%  MOD SEL     -----------_______________-----------     %
%  ACK sampled                           X              %
%                                                       %
%  Analog Read Cycle:                                   %
%                                                       %
%  time           0us 2us               100us     110us %
%  MOD SEL    ----________________________________----  %
%  ACK sampled        X                                 %
%  R/C*       --------------------------_-------------  %
%  STS        __________________________---------____  %
% (analog time-out occurs at 120us)                     %

SUBDESIGN cycle
(clock      : INPUT;    % 10 Mhz clock %
 rst        : INPUT;    % power-on reset %
 arwr       : INPUT;    % address register write %
 a          : INPUT;    % analog cycle type bit %
 r          : INPUT;    % read cycle type bit %
 sts        : INPUT;    % analog-to-digital conversion status %
 ack        : INPUT;    % module acknowledge %
 t0         : INPUT;    % delay 0 %
 t1         : INPUT;    % delay 1 %
 t2         : INPUT;    % delay 2 %
 t3         : INPUT;    % delay 3 %
 t4         : INPUT;    % delay 4 %
 t5         : INPUT;    % delay 5 %
 tclr       : OUTPUT;   % timer reset %
 drd        : OUTPUT;   % digital read latch strobe %
 dwr        : OUTPUT;   % digital write buffer enable %
 mse        : OUTPUT;   % module select enable %
 cnvt       : OUTPUT;   % start ADC conversion %
 setint     : OUTPUT;   % set interrupt %
 setato     : OUTPUT;   % set ADC timeout error %
 setnak     : OUTPUT;   % set module not present error %
 )

VARIABLE
cycle: MACHINE OF BITS (d[3..0])
    WITH STATES (
                          idle = B"0000",
                         init0 = B"1000",
                         init1 = B"0100",
                        digrd0 = B"0110",
                        digrd1 = B"0010",
```

```
                        digwr0 = B"1101",
                        digwr1 = B"1110",
                        digwr2 = B"1100",
                        anard0 = B"0111",
                        anard1 = B"0101",
                        anard2 = B"0011",
                        anard3 = B"0001",
                          erra = B"1010",
                          errb = B"1011",
                          err9 = B"1001",
                          errf = B"1111");

BEGIN
%  Set up the clock and set idle state at power-up%
cycle.clk=clock;
cycle.reset=rst;

% Set up current state decoding %

% state transitions %
TABLE
cycle,t0,t1,t2,t3,t4,t5,arwr,a,r,sts,ack =>
    drd,dwr,mse,cnvt,setint,setato,setnak,tclr,cycle;

% All illegal states lead to Idle %
err9    ,x,x,x,x,x,x,x,x,x,x,x => 0,0,0,0,0,0,0,1, idle     ;% Wait for address reg. write %
erra    ,x,x,x,x,x,x,x,x,x,x,x => 0,0,0,0,0,0,0,1, idle     ;% Wait for address reg. write %
errb    ,x,x,x,x,x,x,x,x,x,x,x => 0,0,0,0,0,0,0,1, idle     ;% Wait for address reg. write %
errf    ,x,x,x,x,x,x,x,x,x,x,x => 0,0,0,0,0,0,0,1, idle     ;% Wait for address reg. write %

%STATE   T T T T T T A A R S A => D D M C S S S T NEXT STATE  Comment  %
%        0 1 2 3 4 5 R     T C    R W S N E E E C                      %
%                    W     S K    D R E V T T T L                      %
%                    R                  T I A N R                      %

idle    ,x,x,x,x,x,x,0,x,x,x,x => 0,0,0,0,0,0,0,1, idle     ;% Wait for address reg. write %
idle    ,x,x,x,x,x,x,1,x,x,x,x => 0,0,0,0,0,0,0,1, init0;

% Reset one-shot timer and dispatch to cycle type %
init0   ,x,x,x,x,x,x,1,x,x,x,x => 0,0,0,0,0,0,0,1, init0    ;% Loop until arwr goes away %
init0   ,x,x,x,x,x,x,0,x,x,x,x => 0,0,0,0,0,0,0,1, init1    ;% 100ns settling delay %
init1   ,x,x,x,x,x,x,x,0,1,x,x => 0,0,0,0,0,0,0,0, digrd0   ;% Go to digital read %
init1   ,x,x,x,x,x,x,x,0,0,x,x => 0,0,0,0,0,0,0,0, digwr0   ;% Go to digital write %
init1   ,x,x,x,x,x,x,x,1,x,x,x => 0,0,0,0,0,0,0,0, anard0   ;% Go to analog read %

digrd0  ,0,x,x,x,x,x,x,x,x,x,x => 0,0,1,0,0,0,0,0, digrd0   ;% Assert MSSTB and wait for ack
and data (T0 delay) %
digrd0  ,1,x,x,x,x,x,x,x,x,x,x => 0,0,1,0,0,0,0,0, digrd1   ;% Assert MSSTB and wait for
data %
digrd1  ,x,x,x,x,x,x,x,x,x,x,1 => 1,0,1,0,1,0,0,0, idle     ;% Latch data and set interrupt
%
digrd1  ,x,x,x,x,x,x,x,x,x,x,0 => 0,0,0,0,1,0,1,0, idle     ;% Set NACK error and set
interrupt %

digwr0  ,x,0,x,x,x,x,x,x,x,x,x => 0,1,0,0,0,0,0,0, digwr0   ;% Set up data (T1 delay) %
digwr0  ,x,1,x,x,x,x,x,x,x,x,x => 0,1,0,0,0,0,0,0, digwr1   ;
digwr1  ,x,x,0,x,x,x,x,x,x,x,x => 0,1,1,0,0,0,0,0, digwr1   ;% Assert MSSTB and wait (T2
delay) %
digwr1  ,x,x,1,x,x,x,x,x,x,x,1 => 0,1,1,0,0,0,0,0, digwr2   ;% Release MSSTB and hold %
digwr1  ,x,x,1,x,x,x,x,x,x,x,0 => 0,1,1,0,1,0,1,0, idle     ;% Set NACK error and set
interrupt %
digwr2  ,x,x,x,0,x,x,x,x,x,x,x => 0,1,0,0,0,0,0,0, digwr2   ;% Hold data (T3 delay) %
digwr2  ,x,x,x,1,x,x,x,x,x,x,x => 0,0,0,0,1,0,0,0, idle     ;% Set interrupt %
```

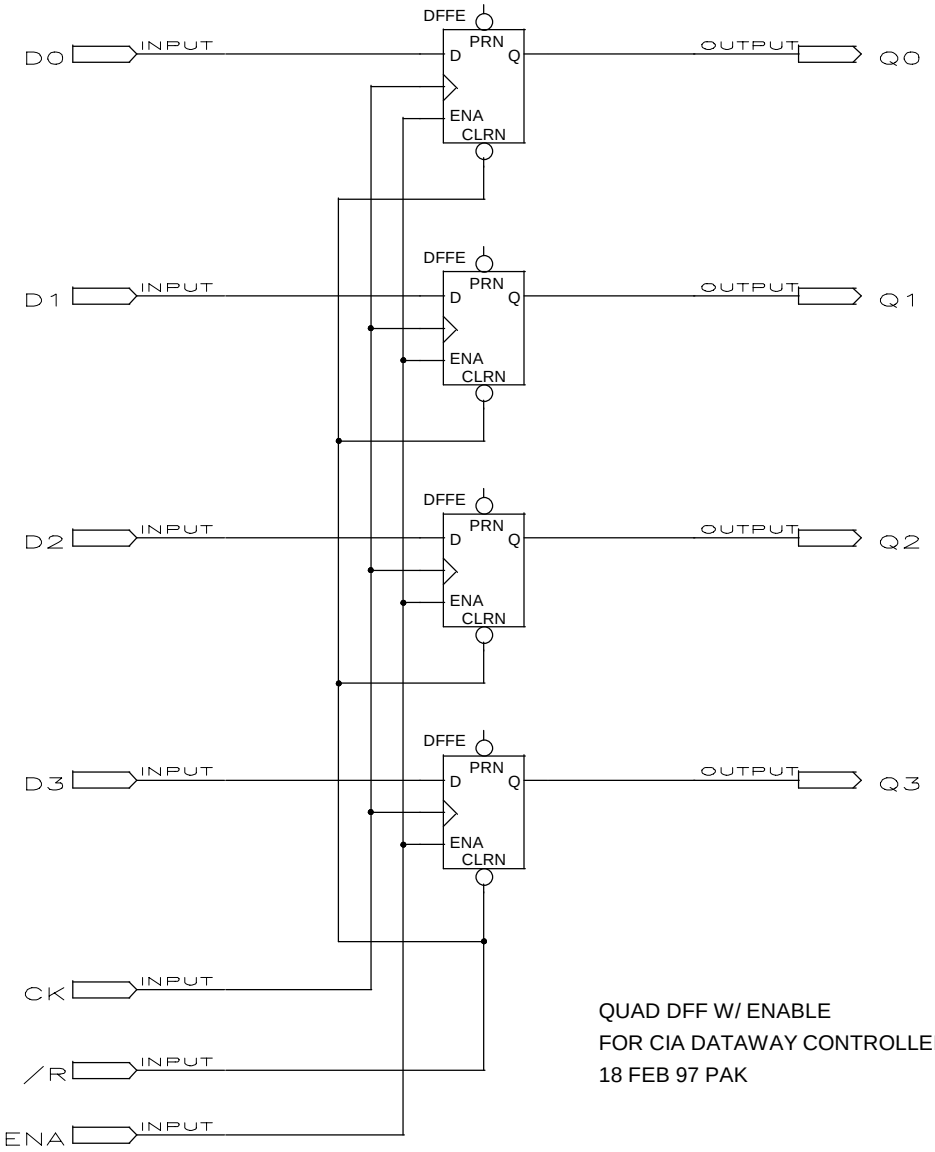
```
anard0  ,0,x,x,x,x,x,x,x,x,x,x => 0,0,1,0,0,0,0,0, anard0   ;% Wait for module ack (T0
delay) %
anard0  ,1,x,x,x,x,x,x,x,x,x,1 => 0,0,1,0,0,0,0,0, anard1   ;% Wait for module ack (T0
delay) %
anard0  ,1,x,x,x,x,x,x,x,x,x,0 => 0,0,1,0,1,0,1,0, idle     ;% Set NACK error and interrupt
%
anard1  ,x,x,x,x,0,x,x,x,x,x,x => 0,0,1,0,0,0,0,0, anard1   ;% Wait for stable analog bus
(T4 delay) %
anard1  ,x,x,x,x,1,x,x,x,x,x,x => 0,0,1,1,0,0,0,0, anard2   ;% Start conversion %

anard2  ,x,x,x,x,x,0,x,x,x,0,x => 0,0,1,0,0,0,0,0, anard2   ;% Wait for STS hi or timeout
(T5 delay) %
anard2  ,x,x,x,x,x,0,x,x,x,1,x => 0,0,1,0,0,0,0,0, anard3   ;% Start conversion %
anard2  ,x,x,x,x,x,1,x,x,x,x,x => 0,0,1,0,1,1,0,0, idle     ;% Set timeout error and set
interrupt %

anard3  ,x,x,x,x,x,0,x,x,x,1,x => 0,0,1,0,0,0,0,0, anard3   ;% Wait for STS low or timeout
(T5 delay) %
anard3  ,x,x,x,x,x,0,x,x,x,0,x => 0,0,1,0,1,0,0,0, idle     ;% Set interrupt %
anard3  ,x,x,x,x,x,1,x,x,x,x,x => 0,0,1,0,1,1,0,0, idle     ;% Set timeout error and set
interrupt %
END TABLE;
END;
```

```
TITLE "TIMER.TDF";
%-----------------------------------------------------------------------%
% AUTHOR: John Smolucha                                                 %
% COMPANY: Fermilab                                                     %
% DESCRIPTION: 11 Bit timer with clear and multiple time markers.       %
% Used to drive timing for a dataway cycle on new CIA Crate Driver.     %
% REVISION HISTORY                                                      %
% DATE     COMMENTS                                              AUTHOR  %
% -----    ----------------------------------------------     --------- %
% 3/6/96   Original Version                                       JMS   %
% 3/20/97  First revision                                         pak   %
%-----------------------------------------------------------------------%
CONSTANT 1us = 10;
CONSTANT 2us = 20;
CONSTANT 3us = 30;
CONSTANT 4us = 40;
CONSTANT 100us = 1000;
CONSTANT 120us = 1200;

% Timer provides markers to the CIA Dataway Cycle Controller %

SUBDESIGN timer
( clk       :INPUT;
  /clr      :INPUT;
  T0        :OUTPUT; % Marks 2 us %
  T1        :OUTPUT; %       1 us %
  T2        :OUTPUT; %       3 us %
  T3        :OUTPUT; %       4 us %
  T4        :OUTPUT; %       100 us %
  T5        :OUTPUT; %       120 us %
)

VARIABLE
        count[10..0]      :DFF;

BEGIN

        count[].clk       = clk;          % use input clock %
        count[].clrn      = /clr;         % When %

        count[] = count[] + 1;              % count up always %


        IF ( count[] == 2us ) THEN
            T0 = VCC;
        ELSE
            T0 = GND;
        END IF;


        IF ( count[] == 1us ) THEN
            T1= VCC;
        ELSE
            T1= GND;
        END IF;

        IF ( count[] == 3us ) THEN
            T2 = VCC;
        ELSE
            T2 = GND;
        END IF;

        IF ( count[] == 4us ) THEN
            T3 = VCC;
```

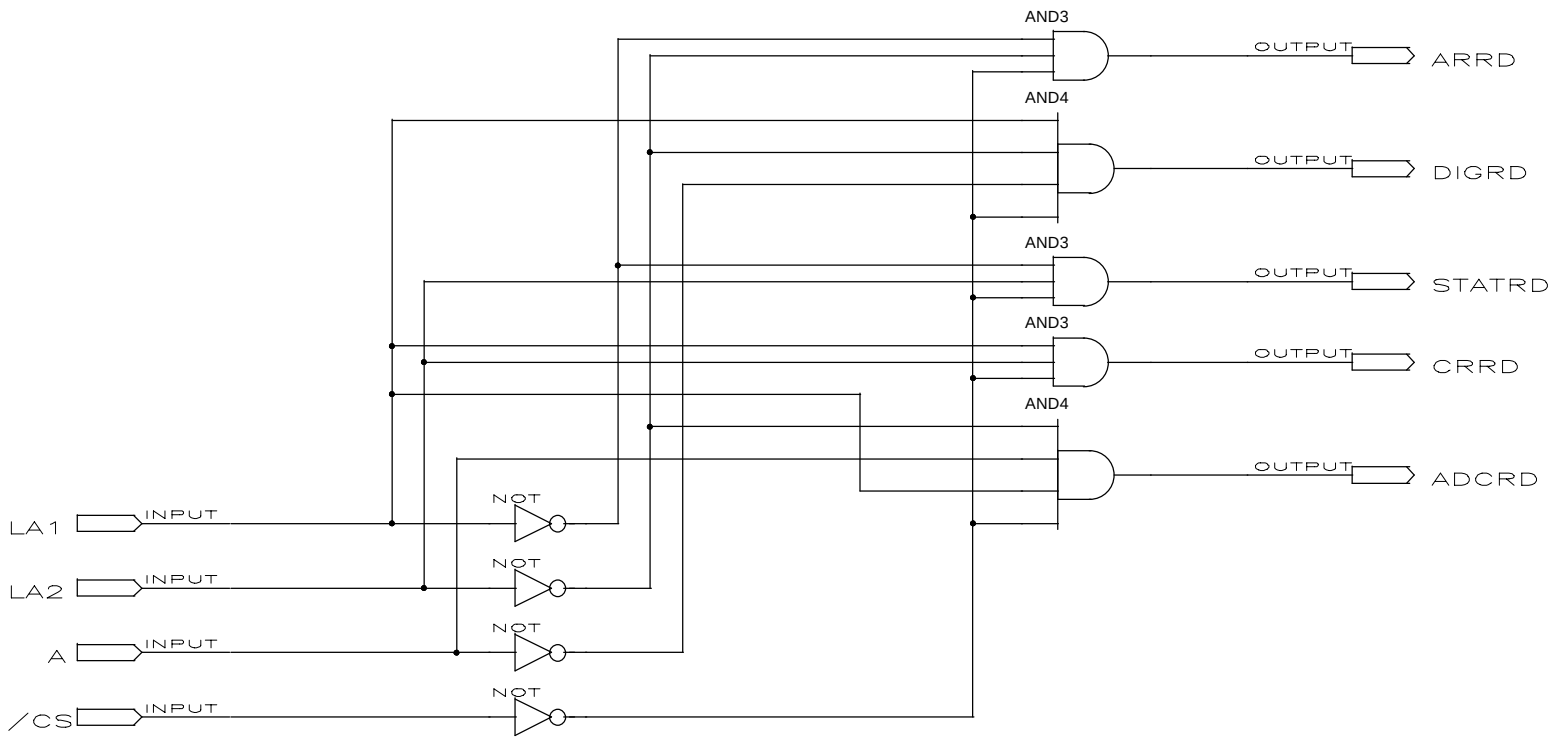
```
        ELSE
            T3 = GND;
        END IF;

        IF ( count[] == 100us ) THEN
            T4 = VCC;
        ELSE
            T4 = GND;
        END IF;

        IF ( count[] == 120us ) THEN
            T5 = VCC;
        ELSE
            T5 = GND;
        END IF;

END;
```



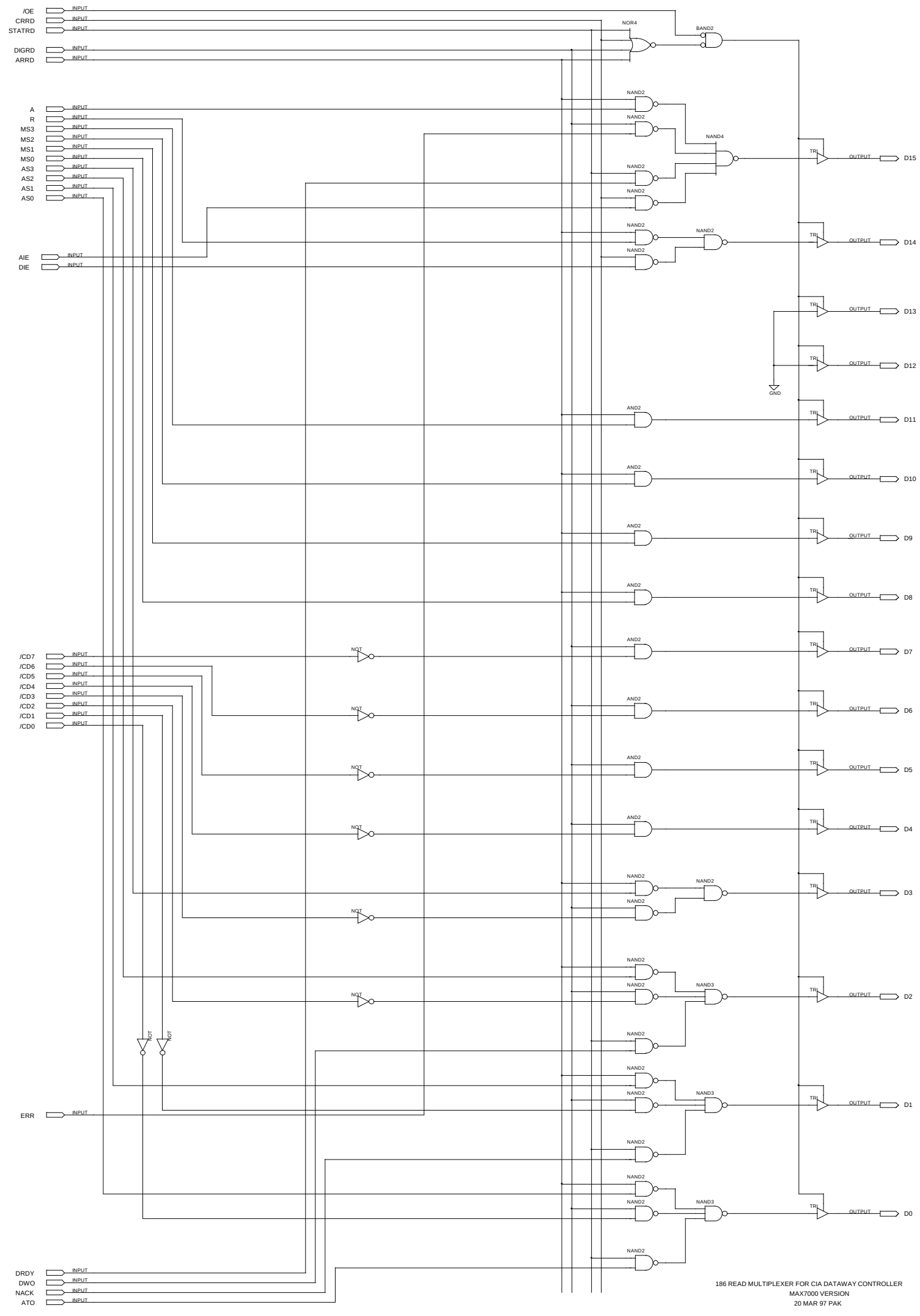
1 OF 4 DECODER
FOR CIA DATAWAY CONTROLLER
18 FEB 97 PAK



QUAD DFF W/ ENABLE
FOR CIA DATAWAY CONTROLLER
18 FEB 97 PAK



186 READ DECODER
FOR CIA DATAWAY CONTROLLER
18 FEB 97 PAK


/OE
CRRD
STATRD
DIGRD
ARRD
A
R
MS3
MS2
MS1
MS0
AS3
AS2
AS1
AS0
AIE
DIE
/CD7
/CD6
/CD5
/CD4
/CD3
/CD2
/CD1
/CD0
ERR
DRDY
DWO
NACK
ATO
INPUT
NOR4
BAND2
NAND2
NAND4
NAND3
AND2
NOT
TRI
OUTPUT
GND
D15
D14
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
186 READ MULTIPLEXER FOR CIA DATAWAY CONTROLLER
MAX7000 VERSION
20 MAR 97 PAK