```
    SUBDESIGN frmtimng      %REV 2.1 1-6-94%
                % DELETED GOFF AND ETONFF TO REFLECT CHANGES IN THE VME CHIP%
                %REV 2.01 11/15/93%
                %ADDED INIT DONE BIT%
                %REV 2.0 10/12/93%
                %ADDED RAM CHIP ARBITRATION AND%
                %RAM CHIP AND ASSOCIATED BUFFER FUNCTIONS%
                %REV 1.0 6/30/93%
                % ADDED PARTRUE AND FRMTRUE INPUTS TO BE ABLE%
                % TO GENERATE A PARITY ERROR AND A FRAME ERROR%
                % REV 1 4-9-93%
                % UPDATED FOR 24 BIT DATA%
                % REV 0 10-12-92%
                % THIS  STATE MACHINE WILL ENABLE THE PROM OUTPUT%
                % FOR THE CORRECT TYPE CODE SO THAT THE SHIFT REG%
                % CAN BE LOADED FROM THE PROPER INPUT BOARD.%

    (CLK10M                 :INPUT; %10MHz CLOCK FROM ENCODE STATE MACHINE%
     ETONCMD                :INPUT; %EXTERNAL TRIGGER ENABLED COMMAND INPUT%
     EVNTEXT/               :INPUT; %EXTERNAL TRIGGER EVENT ACTIVE LOW FROM OPTICAL ISOLATOR%
     EVNTDLY                :INPUT; %DELAYED 720 Hz CLOCK EVENT%
     PARAMID[7..0]          :INPUT; % 8 BIT PARAMETER ID FROM RAM OUTPUT%
     DATACK                 :INPUT;
     CLEAR/                 :INPUT; %ASYNCHRONOUS CLEAR FROM USER (PUSHBUTTON?)%
     DATIN                  :INPUT; %DATA FROM SHIFT REGISTER OUTPUT%
     PARTRUE                :INPUT; %SEND TRUE PARITY%
     FRMTRUE                :INPUT; %SEND TRUE PARITY%
     GOCMD                  :INPUT; %COMMAND THAT SIGNIFIES THE RAM HAS BEEN LOADED%
    %CMDCLK1                :INPUT;% %COMMAND INPUT STROBE LO BYTE%

     TCLTCH                 :OUTPUT;% TYPE CODE LATCH%
     PRMTRAD[7..0]          :OUTPUT;% ADDRESS OF PARAMETER STORED IN RAM%
     TCVALID                :OUTPUT;% PARAMETER VALID TO INPUT BOARD%

     SHFTEN                 :OUTPUT;%ENABLE TO SHIFT REGISTERS%
     ENCDEN                 :OUTPUT;%ENABLE ENCODE STATE MACHINE%
     DATOUT                 :OUTPUT;%DATA TO ENCODE STATE MACHINE%
     NOREPLY                :OUTPUT;
     LOADDATA               :OUTPUT;%LOAD DATA INTO OUTPUT REGISTER OF INPUT BOARD%
     SHFTLD                 :OUTPUT;%LOADS PARAMTER DATA INTO THE SHIFT REGISTERS%
     VMEHOLD                :OUTPUT;%STOP VMEbus CYCLE TO RAM%
     PRMTROE                :OUTPUT;%STATE MACHINE RAM OUTPUT ENABLE FOR PARAMETER%
     LSTFRM                 :OUTPUT;%SIGNIFIES LAST FRAME SENT%
    %ETONSTA                :OUTPUT;%%EXTERNAL TRIGGER ENABLED STATUS OUTPUT%
    %GOSTA                  :OUTPUT;%%RAM LOADED STATUS OUTPUT%

     Q5, Q4, Q3,
     Q2, Q1, Q0             :OUTPUT;%STATE MACHINE VARIABLES%
    )
    VARIABLE
        COUNT:          MACHINE
                OF BITS (Q5, Q4, Q3, Q2, Q1, Q0)
                WITH STATES (S0 = B"000000",
                         S1 = B"000001",
                         S2 = B"000010",
                         S3 = B"000011",
                         S4 = B"000100",
                         S5 = B"000101",
                         S6 = B"000110",
                         S7 = B"000111",
                         S8 = B"001000",
                         S9 = B"001001",
                        S10 = B"001010",
                        S11 = B"001011",
                        S12 = B"001100",
                        S13 = B"001101",
                        S14 = B"001110",
                        S15 = B"001111",
                        S16 = B"010000",
                        S17 = B"010001",
```

```
                    S18 = B"010010",
                    S19 = B"010011",
                    S20 = B"010100",
                    S21 = B"010101",
                    S22 = B"010110",
                    S23 = B"010111",
                    S24 = B"011000",
                    S25 = B"011001",
                    S26 = B"011010",
                    S27 = B"011011",
                    S28 = B"011100",
                    S29 = B"011101",
                    S30 = B"011110",
                    S31 = B"011111",
                    S32 = B"100000",
                    S33 = B"100001",
                    S34 = B"100010",
                    S35 = B"100011",
                    S36 = B"100100",
                    S37 = B"100101",
                    S38 = B"100110",
                    S39 = B"100111",
                    S40 = B"101000",
                    S41 = B"101001",
                    S42 = B"101010",
                    S43 = B"101011",
                    S44 = B"101100",
                    S45 = B"101101",
                    S46 = B"101110",
                    S47 = B"101111",
                    S48 = B"110000",
                    S49 = B"110001",
                    S50 = B"110010",
                    S51 = B"110011",
                    S52 = B"110100",
                    S53 = B"110101",
                    S54 = B"110110",
                    S55 = B"110111",
                    S56 = B"111000",
                    S57 = B"111001",
                    S58 = B"111010",
                    S59 = B"111011",
                    S60 = B"111100",
                    S61 = B"111101",
                    S62 = B"111110",
                    S63 = B"111111");

    PRAD[7..0]            :DFF;
    PRTY                  :TFFE;
    PRTYENFF              :DFF;
    TRIGEVNT              :DFF;
    CLEAR                 :NODE;
    ENCDENFF              :DFF;
    SHFTENFF              :DFF;
    EVNTEXT               :NODE;
    VMEHLDFF              :DFF;
   %GOFF                  :DFF;%
    NOREPFF               :DFF;%CATCHES THE FIRST INSTANCE OF A NON-RESPONSE%
      % ETONFF                :DFF;%%LATCHED COMMAND%
   %CMDCLK1/              :NODE;%

BEGIN

DEFAULTS
   PRAD[ ].d = GND;
   PRMTROE = GND;
   TCLTCH = GND;
   TCVALID = VCC;
   TRIGEVNT.prn = VCC;
   DATOUT = GND;
```

```
        SHFTLD = VCC;
        VMEHLDFF.clrn = VCC;
        VMEHLDFF.prn = VCC;
        %GOFF.prn = VCC;%
        LSTFRM = GND;


     END DEFAULTS;

    CLEAR = !CLEAR/;
    EVNTEXT = !EVNTEXT/;
    %CMDCLK1/ = !CMDCLK1;%

    %GOFF.clk = CMDCLK1/;%
    %GOFF.clrn = CLEAR/;%
    %GOFF.d = GOCMD;%
    %GOSTA = GOFF.q;%

    %ETONFF.d = ETONCMD;%
    %ETONFF.clk = CMDCLK1/;%
    %ETONFF.clrn = CLEAR/;%
    %ETONFF.prn = VCC;%
    %ETONSTA = ETONFF.q;%

    TRIGEVNT.d = VCC;
    TRIGEVNT.clk = (EVNTEXT & ETONCMD) # (!ETONCMD & EVNTDLY);
    TRIGEVNT.clrn = !S63 & CLEAR/;
    LOADDATA = TRIGEVNT.q & GOCMD;

    ENCDENFF.clk = S1;
    ENCDENFF.d = VCC;
    ENCDENFF.prn = VCC;
    ENCDENFF.clrn = !(!CLK10M & S5);

    SHFTENFF.clk = GND;
    SHFTENFF.d = GND;
    SHFTENFF.clrn = !S39;
    SHFTENFF.prn = !((!CLK10M & S7) # S0);

    SHFTEN = SHFTENFF.q;

    PRTY.clk = CLK10M;
    PRTY.t = DATIN;
    PRTY.ena = PRTYENFF.q;
    PRTY.clrn = !(S0 # CLEAR # S63);

    PRTYENFF.d = GND;
    PRTYENFF.clk = GND;
    PRTYENFF.prn = !S39;
    PRTYENFF.clrn = !(!CLK10M & S6);

    VMEHLDFF.d = GND;
    VMEHLDFF.clk = GND;

    NOREPFF.d = VCC;
    NOREPFF.prn = VCC;
    NOREPLY = NOREPFF.q;
    NOREPFF.clrn = !S0 & CLEAR/;

    COUNT.RESET = CLEAR;
    COUNT.CLK = CLK10M;
    PRAD[ ].CLK = S5 & !CLK10M;
    PRAD[ ].CLRN = !TRIGEVNT.q;

    CASE (COUNT) IS
            WHEN S0 =>
                IF (!TRIGEVNT.q # !GOCMD) THEN
                    VMEHLDFF.clrn = GND;
                    COUNT = S0;                                %WAIT FOR CLOCK EVENT%
                ELSIF ( TRIGEVNT.q & GOCMD) THEN
```

```
                COUNT = S63;                            % START LOADING DATA TO TRANSMITTER%
            END IF;
        WHEN S63 =>
            VMEHLDFF.prn = GND;
            COUNT = S62;
        WHEN S62 =>
            PRMTROE = VCC;
            COUNT = S61;
        WHEN S61 =>
            PRMTROE = VCC;
            IF (PARAMID[ ] == H"00") THEN
            LSTFRM = VCC;
            COUNT = S0;                             %START AT FIRST PROM ADDRESS%
            ELSE TCLTCH = VCC;
            COUNT = S60;
            END IF;
        WHEN S60 =>
            TCVALID = GND;
            %VMEHLDFF.clrn = GND;%
            IF  (!DATACK) THEN              % THIS IS LD-XMIT FOR MDATFRME%
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;                             % START TRANSMITTER%
            ELSE COUNT = S59;               % ERROR CONDITION%
            END IF;
        WHEN S59 =>
            IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
            ELSE COUNT = S58;
            END IF;
        WHEN S58 =>
            IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
            ELSE COUNT = S57;
            END IF;
        WHEN S57 =>
            IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
            ELSE COUNT = S56;
            END IF;
        WHEN S56 =>
        IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
            ELSE COUNT = S55;
            END IF;
        WHEN S55 =>
            IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
            ELSE COUNT = S54;
            END IF;
        WHEN S54 =>
            IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
            ELSE COUNT = S53;
            END IF;
        WHEN S53 =>
            IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
            ELSE COUNT = S52;
            END IF;
        WHEN S52 =>
            IF (!DATACK) THEN
            SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
            COUNT = S1;
```

```
                ELSE COUNT = S51;
                END IF;
            WHEN S51 =>
                IF (!DATACK) THEN
                SHFTLD = GND;
                COUNT = S1;
                ELSE COUNT = S50;
                END IF;
            WHEN S50 =>
                IF (!DATACK) THEN
                SHFTLD = GND;                   % LOAD PARAMETER DATA INTO SHIFT REGISTERS%
                COUNT = S1;
                ELSE NOREPFF.clk = VCC;             %THIS WILL LATCH THE TYPE CODE%
                COUNT = S5;
                END IF;
            WHEN S1 =>
                DATOUT = GND;
                COUNT = S39; %START SHIFTING DATA%
            WHEN S39 => COUNT = S38;
            WHEN S38 => COUNT = S37;
            WHEN S37 => COUNT = S36;
            WHEN S36 => COUNT = S35;
            WHEN S35 => COUNT = S34;
            WHEN S34 => COUNT = S33;
            WHEN S33 => COUNT = S32;
            WHEN S32 => COUNT = S31;
            WHEN S31 => COUNT = S30;
            WHEN S30 => COUNT = S29;
            WHEN S29 => COUNT = S28;
            WHEN S28 => COUNT = S27;
            WHEN S27 => COUNT = S26;
            WHEN S26 => COUNT = S25;
            WHEN S25 => COUNT = S24;
            WHEN S24 => COUNT = S23;
            WHEN S23 => COUNT = S22;
            WHEN S22 => COUNT = S21;
            WHEN S21 => COUNT = S20;
            WHEN S20 => COUNT = S19;
            WHEN S19 => COUNT = S18;
            WHEN S18 => COUNT = S17;
            WHEN S17 => COUNT = S16;
            WHEN S16 => COUNT = S15;
            WHEN S15 => COUNT = S14;
            WHEN S14 => COUNT = S13;
            WHEN S13 => COUNT = S12;
            WHEN S12 => COUNT = S11;
            WHEN S11 => COUNT = S10;
            WHEN S10 => COUNT = S8;
            WHEN S8 =>
                %VMEHLDFF.prn = GND;%
                COUNT = S7;
            WHEN S7 => COUNT = S6;  %PARITY BIT%
            WHEN S6 => COUNT = S5;  %STOP BIT%
            WHEN S5 =>
                PRAD[ ].d = PRAD[ ].q + 1;% NEXT TYPE CODE ADDRESS%
                COUNT = S4;
            WHEN S4 =>
                IF (PRAD[ ].q == H"00") THEN
                COUNT = S0;
                ELSE COUNT = S63;
                END IF;

            WHEN OTHERS => COUNT = S0;
        END CASE;

    %PRAD[ ].d = PRAD[ ].q;%
    PRMTRAD[ ] = PRAD[ ];

    %ENCDEN = S1 # S5 # S6 # S7 # (Q5 & !Q4 & !Q3) # (!Q5 & (Q4 # Q3));%
```

```
    ENCDEN = ENCDENFF.q;

        IF  S1 THEN
            DATOUT = GND;
        ELSIF  (S6 & PARTRUE) THEN
            DATOUT = PRTY.q;
        ELSIF (S6 & !PARTRUE) THEN
            DATOUT = !PRTY.q;
            ELSIF  (S5 & FRMTRUE) THEN
            DATOUT = VCC;
        ELSIF  (S5 & !FRMTRUE) THEN
            DATOUT = GND;
        ELSE
            DATOUT = DATIN;
        END IF;

        IF CLEAR THEN
            ENCDEN = GND;
        END IF;

    VMEHOLD = VMEHLDFF.q;

    END;
```