บรรณานุกรม

ภาษาไทย

สายัณห์ ธีรปัญญาวัฒน์, การพัฒนาเครื่องรับเทเลเท็กซ์แสดงผลตัวอักษรไทย/อังกฤษ, บัณฑิตวิทยาลัย จุฬาลงกรณ์มหาวิทยาลัย, พ.ศ. 2536

ภาษาอังกฤษ

Zainalabedin Navabi, VHDL Analysis and Modeling of Digital Systems, Mc Graw Hill,1993

David R. Coelho, The VHDL Handbook, Kluwer Academic Publishers, 1989

Mentor Graphics Introduction to VHDL, Mentor Graphics, 1993

A Guide to Design Process and Database Concepts, Mentor Graphics, 1993

Autologic User Interface Reference Manual, Mentor Graphics, 1993

Kinghorn,J.R. Enhanced Computer Controlled Teletext SAA5243 Series User Manual. English:Micham,1989. 143 pp.

ภาคผนวก

ภาคผนวก ก.
รายละเอียดการเขียนโปรแกรม
เพื่อการวิเคราะห์วงจร

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL,ieee.std_logic_1164_extensions.ALL;
USE std.textio.ALL;


ENTITY slicer IS
        PORT (
                        clk,vcs,ttc,ttd     : OUT std_ulogic:='0';
                        tcs,pl,cbb          : IN std_ulogic
                        );
END slicer;
ARCHITECTURE behavioral OF slicer IS
        SIGNAL          end_of_simulation : boolean:=false;
        SIGNAL          field_count                 : integer:=1;
        SIGNAL          input_line : string(1 to 44);
        SIGNAL ttc_i                        : std_ulogic;
BEGIN
        Clk1_Process: PROCESS
                VARIABLE clk_i : std_ulogic:='0';
                CONSTANT T1_2 : time:=83.4ns;
                CONSTANT T2_2 : time:=83.3ns;
        BEGIN
                clk_i := '0';
                clk <= clk_i;
                WHILE not end_of_simulation LOOP
                        FOR i in 1 to 6 LOOP
                                IF (i=5) OR (i=6) THEN
                                        WAIT FOR T1_2;
                                ELSE
                                        WAIT FOR T2_2;
                                END IF;
                                clk_i := not clk_i;
                                clk <= clk_i;
                        END LOOP;
                END LOOP;
                WAIT;
        END PROCESS Clk1_Process;
```

```
Clk2_Process: PROCESS
        CONSTANT T : time:=144ns;
BEGIN
        ttc_i <= '0';
        ttc <= ttc_i;
        WHILE not end_of_simulation LOOP
                WAIT FOR T/2;
                ttc_i <= not ttc_i;
                ttc <= ttc_i;
        END LOOP;
        WAIT;
END PROCESS Clk2_Process;


VCS_Process: PROCESS
        VARIABLE first : boolean := true;
        VARIABLE counter : integer := 1;
BEGIN
        IF first THEN
                WAIT ON pl UNTIL pl='1';
                WAIT FOR 30.5us;
        END IF;
        first := false;
        IF (counter = 1 ) OR (counter = 2) -- BP
        OR (counter = 314) OR (counter = 315) THEN
                vcs <= '1';
                WAIT FOR 27.33us;
                vcs <= '0';
                WAIT FOR 4.67us;
                vcs <= '1';
                WAIT FOR 27.33us;
                vcs <= '0';
                WAIT FOR 4.67us;
        ELSIF ( counter = 3 ) THEN
                vcs <= '1';
                WAIT FOR 27.33us;
                vcs <= '0';
```

```
            WAIT FOR 4.67us;
            vcs <= '1';
            WAIT FOR 2.33us;
            vcs <= '0';
            WAIT FOR 29.67us;
        ELSIF (counter = 313) THEN
            vcs <= '1';
            WAIT FOR 2.33us;
            vcs <= '0';
            WAIT FOR 29.67us;
            vcs <= '1';
            WAIT FOR 27.33us;
            vcs <= '0';
            WAIT FOR 4.67us;
        ELSIF (counter=4) OR (counter=5) OR (counter=311) OR
                    (counter=312) OR (counter=316) OR (counter=317) OR
                    (counter=623) OR (counter=624) OR (counter=625) THEN

            vcs <= '1';
            WAIT FOR 2.33us;
            vcs <= '0';
            WAIT FOR 29.67us;
            vcs <= '1';
            WAIT FOR 2.33us;
            vcs <= '0';
            WAIT FOR 29.67us;
        ELSE
            vcs <= '1';
            WAIT FOR 4.66us;
            vcs <= '0';
            WAIT FOR 59.34us;
        END IF;
        IF (counter /= 625) THEN
            counter := counter +1;
        ELSE
            counter := 1;
```

```
        END IF;
        field_count <= counter;
        IF (end_of_simulation) THEN
                vcs <= '0';
                WAIT;
        END IF;
        END PROCESS VCS_Process;

TTD_Process: PROCESS

        PROCEDURE send_data(stream: std_ulogic_vector) IS
                BEGIN
                        FOR i IN stream'LOW to stream'HIGH LOOP
                                WAIT ON ttc_i UNTIL ttc_i='1';
                                ttd <= stream(i);
                        END LOOP;
        END send_data;

        FUNCTION par_gen(data : std_ulogic_vector) RETURN std_ulogic_vector IS
                VARIABLE high_bit : std_ulogic:='1';
        BEGIN
                FOR i IN data'RANGE LOOP
                        IF (data(i) = '1') THEN
                                high_bit := not high_bit;
                        END IF;
                END LOOP;
                RETURN high_bit&data;
        END par_gen;

        FUNCTION ham_gen(data : std_ulogic_vector(3 downto 0)) RETURN std_ulogic_vector
IS
                VARIABLE par4           : std_ulogic_vector(3 downto 0);
                VARIABLE par8,X                         : std_ulogic_vector(7 downto 0);
        BEGIN
                X               := data(3)&'0'&data(2)&'0'&data(1)&'0'&data(0)&'0';
                par4    := par_gen(data(0)&data(2)&data(3));
```

```
                X(0)    := par4(3);
                par4    := par_gen(data(0)&data(1)&data(3));
                X(2)    := par4(3);
                par4    := par_gen(data(0)&data(1)&data(2));
                X(4)    := par4(3);
                par8    := par_gen(X(7)&X(5 downto 0));
                X(6)    := par8(7);
                RETURN X;
        END ham_gen;

        FILE textfile : TEXT IS IN "ain.txt";
        TYPE data_type IS (ascii,command);
        VARIABLE mode : data_type;
        VARIABLE ok                             : boolean;
        VARIABLE OneLine                : line;
        VARIABLE OneChar                : character;
        CONSTANT ClkRunIn               : std_ulogic_vector(15 downto 0)    :=
"0101010101010101";
        CONSTANT FramingCode : std_ulogic_vector( 7 downto 0)     := "00100111";
        VARIABLE data,mag,row   : std_ulogic_vector(7 downto 0);
        VARIABLE ascii_code,row1,row2,unit,ten,page
    : INTEGER;
        VARIABLE data_start
                                        : INTEGER:=4;
    BEGIN
        WHILE not endfile(textfile) LOOP
                WAIT ON field_count UNTIL ((field_count > 7 ) AND (field_count <    21 ))
                                                                        OR
((field_count >319) AND (field_count < 333 ));
                readline(textfile,OneLine);
                WAIT FOR 9us;
                data_start:=0;
                mode := ascii;
                send_data(ClkRunIn); send_data(FramingCode);
                read(OneLine,OneChar);
                ascii_code      := CHARACTER'POS(OneChar);
```

```
                mag                     := to_StdUlogicVector(ascii_code-48,8);
                ASSERT (mag >= "00000001") AND (mag <= "00000111") REPORT "
MAGAZINE OUT OF RANGE" SEVERITY ERROR;
                read(OneLine,OneChar);
                row1                    := CHARACTER'POS(OneChar)-48;
                ASSERT (row1 >= 0) AND (row1 <= 9) REPORT "ROW OUT OF RANGE"
SEVERITY ERROR;
                read(OneLine,OneChar);
                row2                    := CHARACTER'POS(OneChar)-48;
                ASSERT (row2 >= 0) AND (row1 <= 9) REPORT "ROW OUT OF RANGE"
SEVERITY ERROR;
                row1                    := row1*10+row2;
                row                     := to_StdUlogicVector(row1,8);
                ASSERT (row >= "00000000") AND (row <= "00011001") REPORT "ROW
OUT OF RANGE" SEVERITY ERROR;
                data := ham_gen(row(0)&mag(2 downto 0)); send_data(data);
                data := ham_gen(row(4 downto 1)); send_data(data);
                FOR i IN 4 TO 5 LOOP
                        IF (row1 = 0) THEN
                                read(OneLine,OneChar);
                                page := CHARACTER'POS(OneChar)-48;
                                ASSERT (page >= 0) AND (page <= 9) REPORT "PAGE
OUT OF RANGE" SEVERITY ERROR;
                                data            := to_StdUlogicVector(page,8);
                                data            := ham_gen(data(3 downto 0));
                                send_data(data);
                                data_start := 2;
                        END IF;
                END LOOP;
                FOR i IN data_start to 39 LOOP
                        read(OneLine,OneChar);
                        IF (OneChar='^') THEN
                                mode := command;
                                read(OneLine,OneChar);
                        ELSIF (OneChar='@') THEN
                                mode := ascii;
```

```
                    read(OneLine,OneChar);
                END IF;
                IF (mode=ascii) THEN
                    ascii_code      := CHARACTER'POS(OneChar);
                ELSE
                    ten     := CHARACTER'POS(OneChar)-48;
                    IF (ten > 9) THEN ten:=ten-7; END IF;
                    ASSERT (ten >= 0) AND (ten <= 15) REPORT
"HEXADECIMAL OUT OF RANGE" SEVERITY ERROR;

                    read(OneLine,OneChar);
                    unit    := CHARACTER'POS(OneChar)-48;
                    IF (unit > 9) THEN unit:=unit-7; END IF;
                    ASSERT (unit >= 0) AND (unit <= 15) REPORT
"HEXADECIMAL OUT OF RANGE" SEVERITY ERROR;
                    ascii_code      := ten*16 + unit;
                END IF;
                    data    := to_StdUlogicVector(ascii_code,8);
                    data    := par_gen(data(6 downto 0));
                    send_data(data);
                END LOOP;
            END LOOP;
            end_of_simulation <= true;
            WAIT;
        END PROCESS TTD_Process;
END behavioral;

LIBRARY ieee;
USE ieee.std_logic_1164.ALL,ieee.std_logic_1164_extensions.ALL;
USE std.textio.ALL;

ENTITY rgbmonitor IS
        PORT (
            tcs,R,G,B  : IN std_ulogic );
END rgbmonitor;
```

```
ARCHITECTURE behavioral OF rgbmonitor IS
        SIGNAL NewField : boolean:=false;
        SIGNAL FieldCount: integer:=0;
BEGIN
        Seek_New_Field: PROCESS
                VARIABLE rising_time, falling_time,pulse_width :time:=0ns;
        BEGIN
                WAIT ON tcs;
                IF tcs='1' and tcs'LAST_VALUE='0' THEN
                        rising_time := now;
                ELSIF tcs='0' AND tcs'LAST_VALUE='1' THEN
                        falling_time := now;
                        pulse_width := falling_time - rising_time;
                        IF (pulse_width < 5us) AND (pulse_width > 0ns) THEN
                                NewField <= true;
                        ELSE
                                NewField <= false;
                        END IF;
                        IF (NewField = true) THEN
                                NewField <= false;
                                WAIT FOR 17ms;
                        END IF;
                END IF;
        END PROCESS Seek_New_Field;

        Field_Count: PROCESS
        BEGIN
                        IF (NewField) THEN
                                FieldCount <= 0;
                        ELSIF (falling_edge(tcs)) THEN
                                FieldCount <= FieldCount + 1;
                                WAIT FOR 40us;
                        END IF;
                        WAIT ON NewField,tcs;
        END PROCESS Field_Count;
```

```
Display_Process: PROCESS
        FILE textfile           : TEXT IS OUT "rgb.dot";
        VARIABLE    OneLine : line;
        VARIABLE no                 : integer:=0;
        VARIABLE rgb            : string(1 to 3);
        CONSTANT NEXTLINE : string(1 to 1):= "n";
        CONSTANT NEXTFIELD : string(1 to 1):= "e"       ;
BEGIN
        WAIT ON tcs UNTIL ((FieldCount >= 40) AND (FieldCount <250))
                                OR      ((FieldCount >= 352) AND (FieldCount
<562));
        WAIT FOR 14us;
        FOR J IN 1 TO 50 LOOP
                FOR I IN 1 TO 12 LOOP
                        rgb := "111";
                        IF R='0' THEN rgb(1):= '0'; END IF;
                        IF G='0' THEN rgb(2):= '0'; END IF;
                        IF B='0' THEN rgb(3):= '0'; END IF;
                        WRITE(OneLine,rgb);
                        WAIT FOR 83.33ns;
                END LOOP;
        END LOOP;
        WRITE(OneLine,NextLine);
        WRITELINE(TextFile,OneLine);
        IF (FieldCount = 249 ) THEN
                WRITE(OneLine,NextField);
                WRITELINE(TextFile,OneLine);
        END IF;
END PROCESS Display_Process;
END behavioral;


LIBRARY ieee;
USE ieee.std_logic_1164.ALL,ieee.std_logic_1164_extensions.ALL;
USE std.textio.ALL;
ENTITY rom4kbyte IS
```

```
        GENERIC (ACCESSTIME: TIME:=120ns);
        PORT (
                                A               : IN std_ulogic_vector (11 downto 0);
                                D               : INOUT std_logic_vector (7 downto 0) BUS ;
                                OEB     : IN std_ulogic ;
                                CEB     : IN std_ulogic );
END rom4kbyte;

ARCHITECTURE behavioral OF rom4kbyte IS
        CONSTANT TOP : integer:=4097;
BEGIN
        Read_Rom: PROCESS
                TYPE ROM_ARRAY IS ARRAY (0 TO TOP) OF bit_vector(7 downto 0);
                FILE RomFile : TEXT IS IN "font.rom";
                VARIABLE OneLine        : line;
                VARIABLE ok                     : boolean;
                VARIABLE data_i         : bit_vector(7 downto 0);
                VARIABLE        index   : integer RANGE 0 TO TOP;
                VARIABLE initial        : boolean := false;
                VARIABLE tce.toe        : time:=0ns;
                VARIABLE register_file : ROM_ARRAY;
        BEGIN
        IF not initial THEN
                FOR i IN 0 TO TOP LOOP
                        EXIT WHEN  endfile(RomFile);
                        READLINE (RomFile,OneLine);
                        READ (OneLine,data_i,Ok);
                        register_file(i) := data_i;
                END LOOP;
        END IF;
        initial := true;
        index := to_integer('0'&a(a'HIGH downto 0));
        data_i:= register_file(index);
        IF ( ceb = '0' AND ceb'EVENT) THEN
                IF (not oeb'EVENT) THEN WAIT UNTIL oeb = '0' FOR 60ns; END IF;
                d <= (OTHERS => 'X');
```

```
                WAIT FOR 1ns;
                WAIT ON a,ceb,oeb FOR ACCESSTIME;
                IF (a'EVENT OR ceb'EVENT OR oeb'EVENT) THEN
                        ASSERT false REPORT "UNCOMPLETE READ CYCLE" SEVERITY
WARNING;
                END IF;
                        d <= To_stdlogicvector(data_i);
                        WAIT UNTIL oeb = '1' OR ceb = '1';
                        d <= NULL;
        END IF;
        WAIT ON A,ceb,oeb;
        END PROCESS Read_Rom;
END behavioral;


LIBRARY ieee;
USE ieee.std_logic_1164.ALL,ieee.std_logic_1164_extensions.ALL;
USE std.textio.ALL;
ENTITY ram4kbyte IS
        GENERIC (ACCESSTIME: TIME:=120ns);
        PORT (
                                A               : IN std_ulogic_vector (11 downto 0);
                                D               : INOUT std_logic_vector (7 downto 0) BUS ;
                                OEB     : IN std_ulogic ;
                                WEB     : IN std_ulogic ;
                                CEB     : IN std_ulogic );
END ram4kbyte;


ARCHITECTURE behavioral OF ram4kbyte IS
        CONSTANT TOP : integer:=4097;
BEGIN
        Read_Write: PROCESS
                TYPE MEMORY_ARRAY IS ARRAY (0 TO TOP) OF std_ulogic_vector(7 downto 0);
                VARIABLE register_file : MEMORY_ARRAY;
                FILE RamFile : TEXT IS IN "ramfile.ini";
                VARIABLE OneLine        : line;
```

```
        VARIABLE data_i         : bit_vector(7 downto 0);
        VARIABLE      index     : integer RANGE 0 TO TOP;
        VARIABLE initial        : boolean:= false;
    BEGIN

    IF not initial THEN
        FOR i IN register_file'RANGE LOOP
            register_file(i) := (OTHERS =>'0');
        END LOOP;
        FOR i IN 0 TO TOP LOOP
            EXIT WHEN  endfile(RamFile);
            READLINE (RamFile,OneLine);
            READ (OneLine,data_i);
            register_file(i) := To_StdUlogicVector(data_i);
        END LOOP;
    END IF;
    initial := true;

    index := to_integer('0'&a(a'HIGH downto 0));
    IF ( ceb ='0' AND ceb'EVENT) THEN
        IF (web = '0') THEN
            WAIT FOR 1ns;
            WAIT ON a,ceb,oeb,web FOR ACCESSTIME;
            IF (a'EVENT OR ceb'EVENT OR oeb'EVENT OR web'EVENT) THEN
                ASSERT false REPORT "UNCOMPLETE WRITE CYCLE"
SEVERITY WARNING;
            ELSE
                register_file(index) := std_ulogic_vector(d);
            END IF;
        ELSIF (not web'EVENT AND not oeb'EVENT) THEN
            WAIT UNTIL web = '0' OR oeb = '0' FOR 60ns;
            IF (web = '0') THEN
                WAIT FOR 1ns;
                WAIT ON a,ceb,oeb,web FOR ACCESSTIME;
                IF (a'EVENT OR ceb'EVENT OR oeb'EVENT OR web'EVENT)
THEN
```

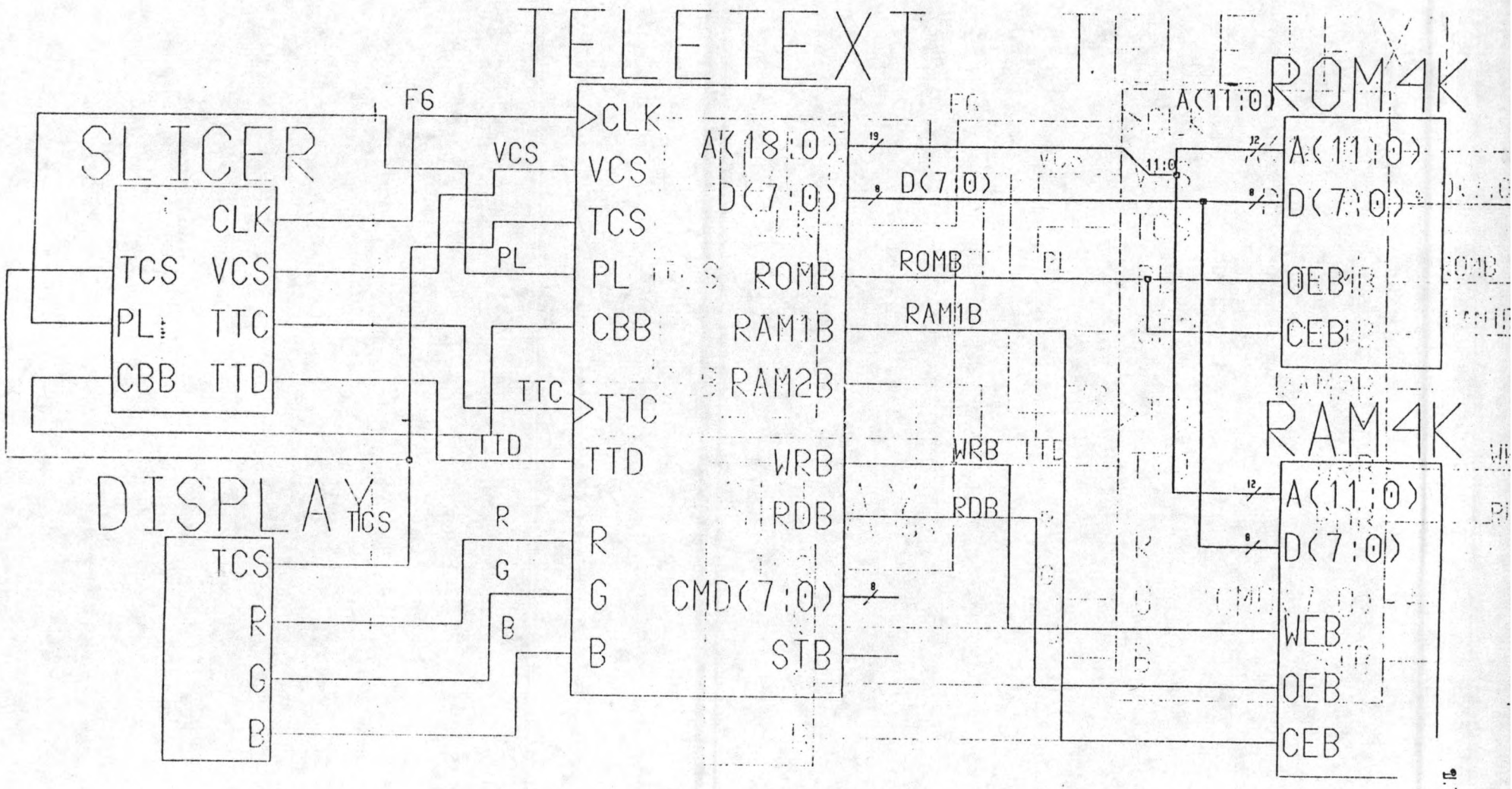
```
                                        ASSERT false REPORT "UNCOMPLETE WRITE CYCLE"
SEVERITY WARNING;
                                ELSE
                                        register_file(index) := std_ulogic_vector(d);
                                END IF;
                        ELSIF (oeb = '0') THEN
                                d <= (OTHERS =>'X');
                                WAIT ON a,ceb,oeb,web FOR ACCESSTIME;
                                IF (a'EVENT OR ceb'EVENT OR oeb'EVENT OR web'EVENT)
THEN
                                        ASSERT false REPORT "UNCOMPLETE READ CYCLE"
SEVERITY WARNING;
                                END IF;
                                d <= std_logic_vector(register_file(index));
                                WAIT UNTIL ceb='1' OR oeb='1';
                                d <= NULL;
                        END IF;
                END IF;
        END IF;
        WAIT ON A,ceb,oeb,web ;
        END PROCESS Read_Write;
END behavioral;
```

ภาคผนวก ข.

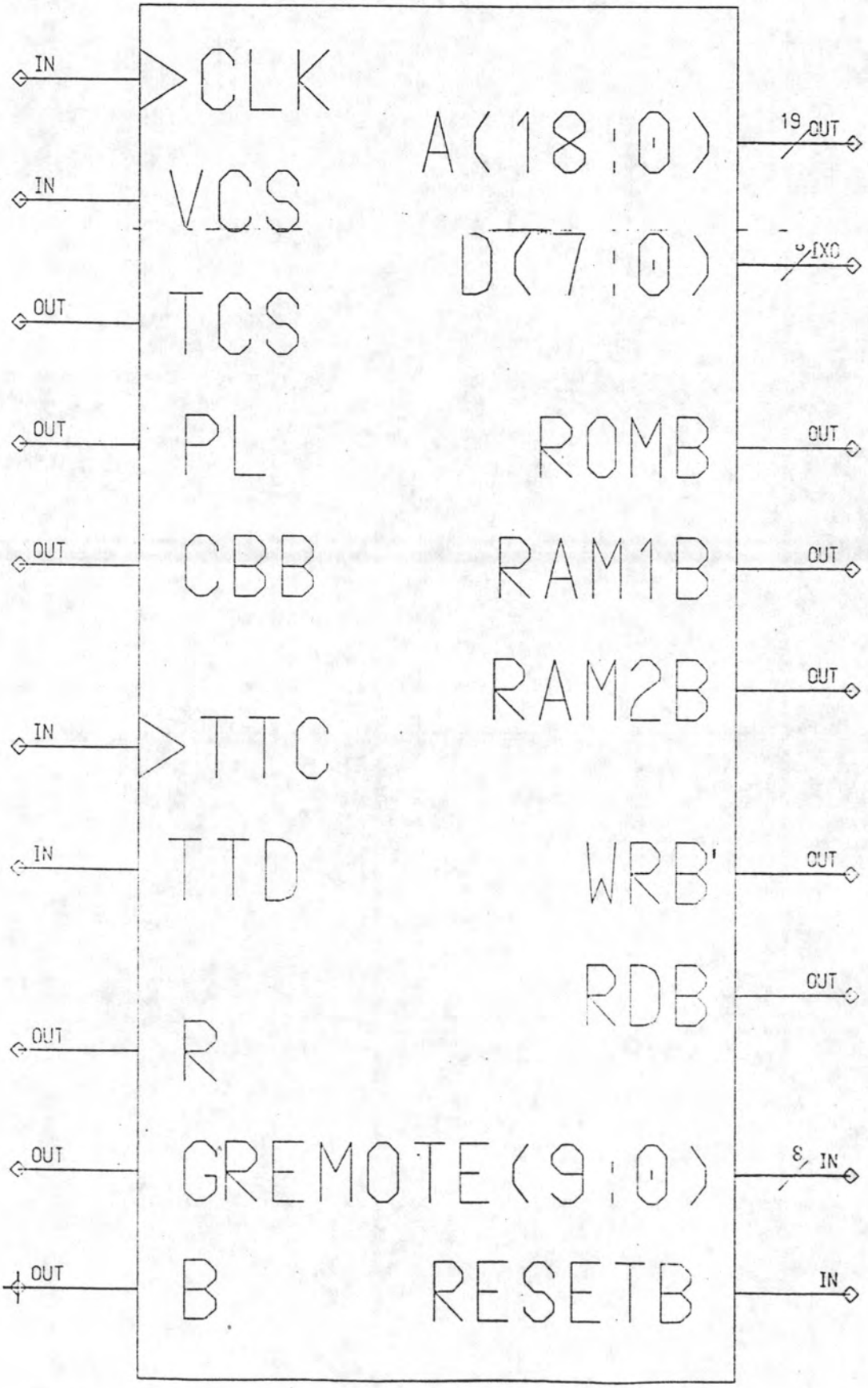
Block Diagram

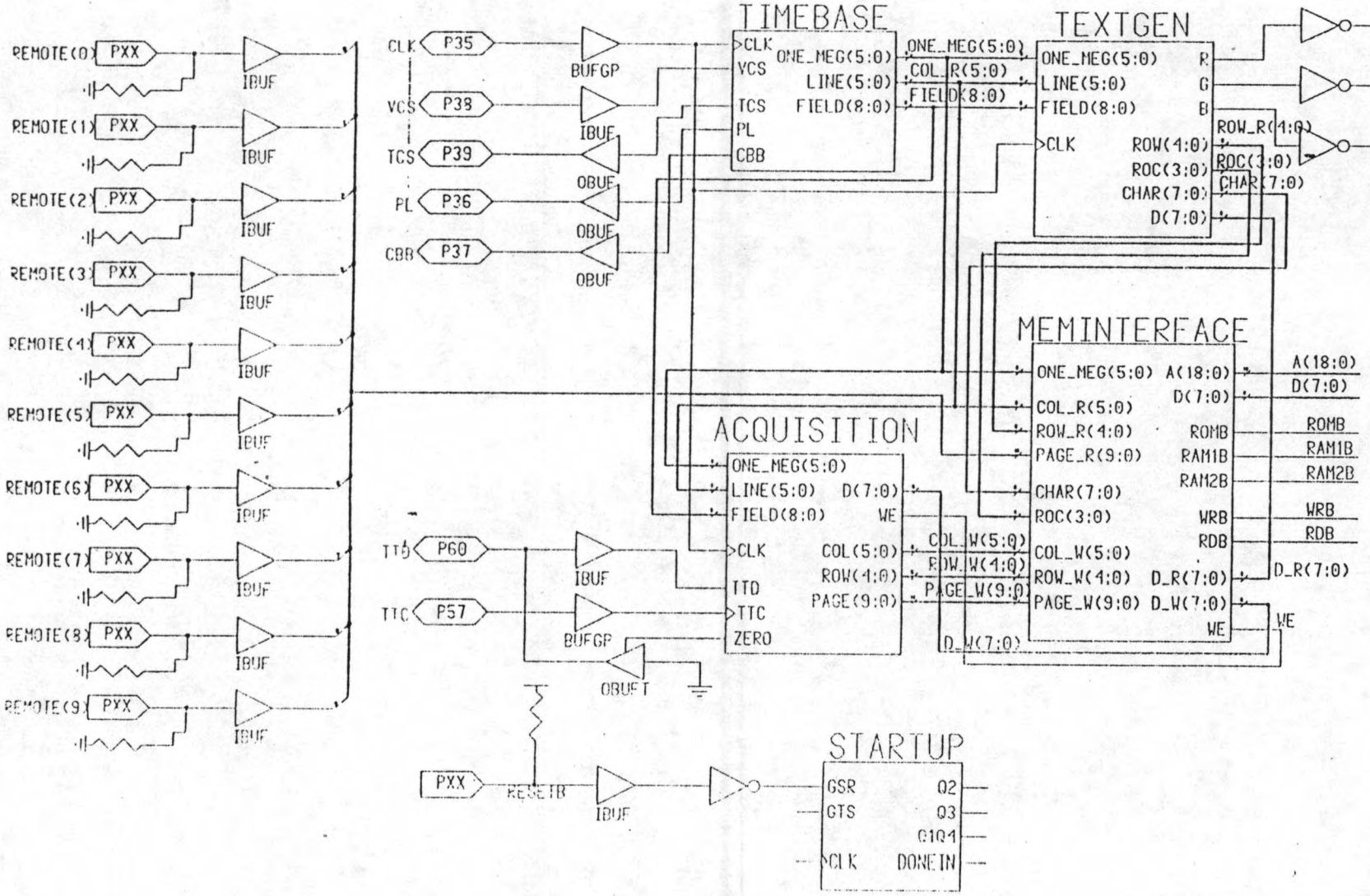
ของ

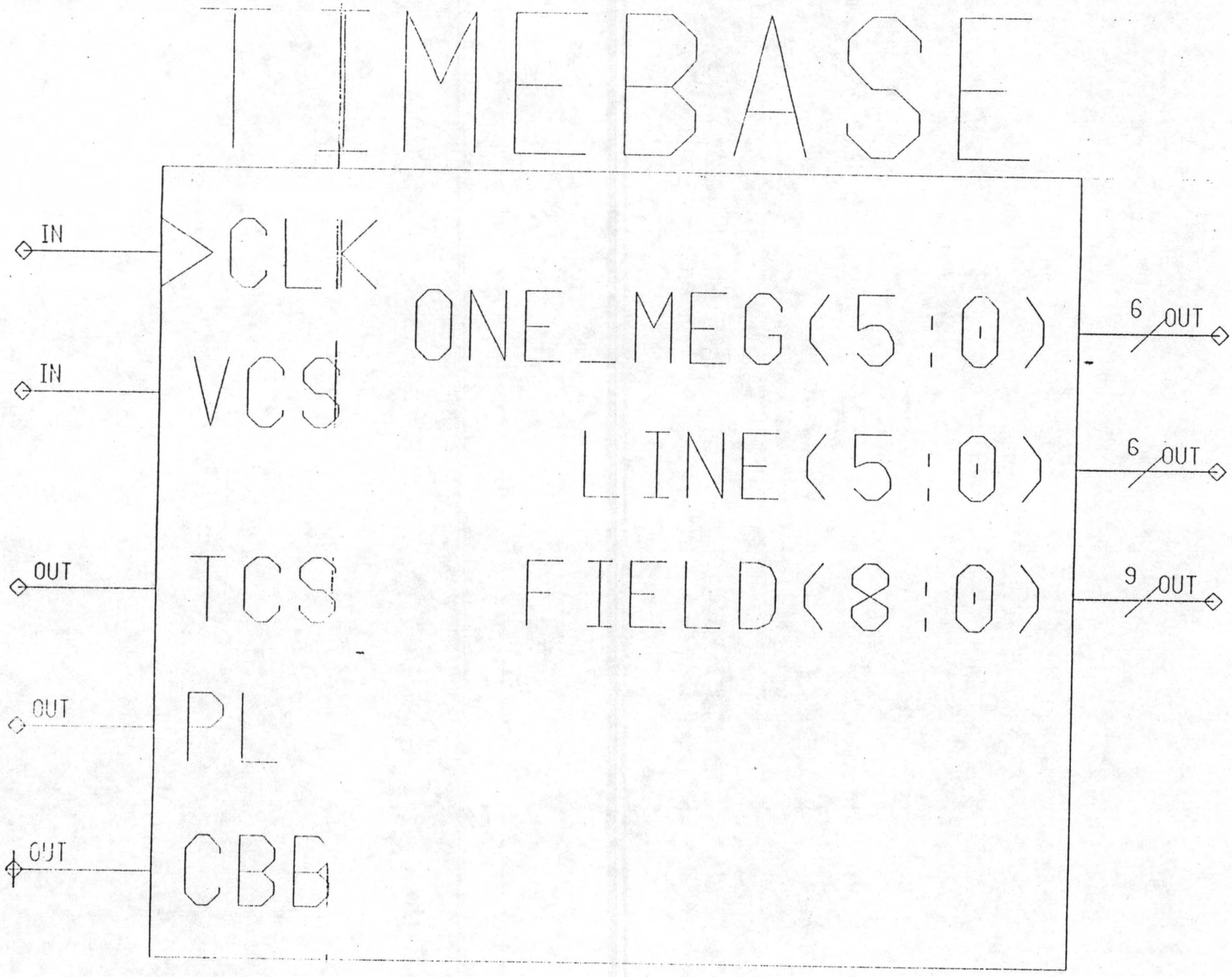
เครื่องถอดรหัสเทเลเท็กซต์

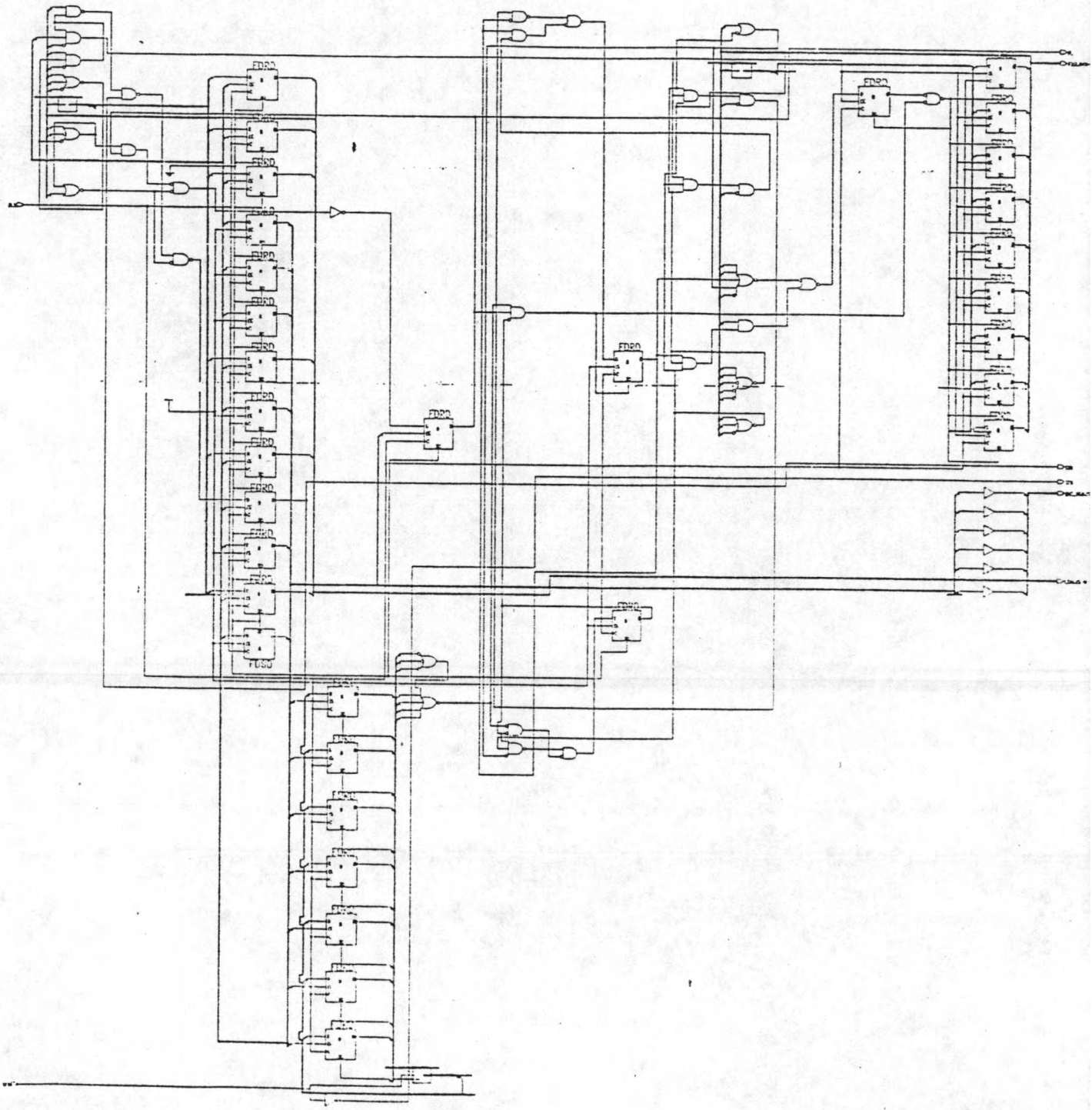

TELETEXT
SLICER
CLK
TCS
VCS
PL
TTC
CBB
TTD
F6
VCS
PL
TTC
TTD
DISPLAY
TCS
R
G
B
CLK
VCS
TCS
PL
CBB
TTC
TTD
R
G
B
A(18:0)
D(7:0)
ROMB
RAM1B
RAM2B
WRB
RDB
CMD(7:0)
STB
ROM4K
A(11:0)
D(7:0)
OEB
CEB
RAM4K
A(11:0)
D(7:0)
WEB
OEB
CEB

TELETEXT

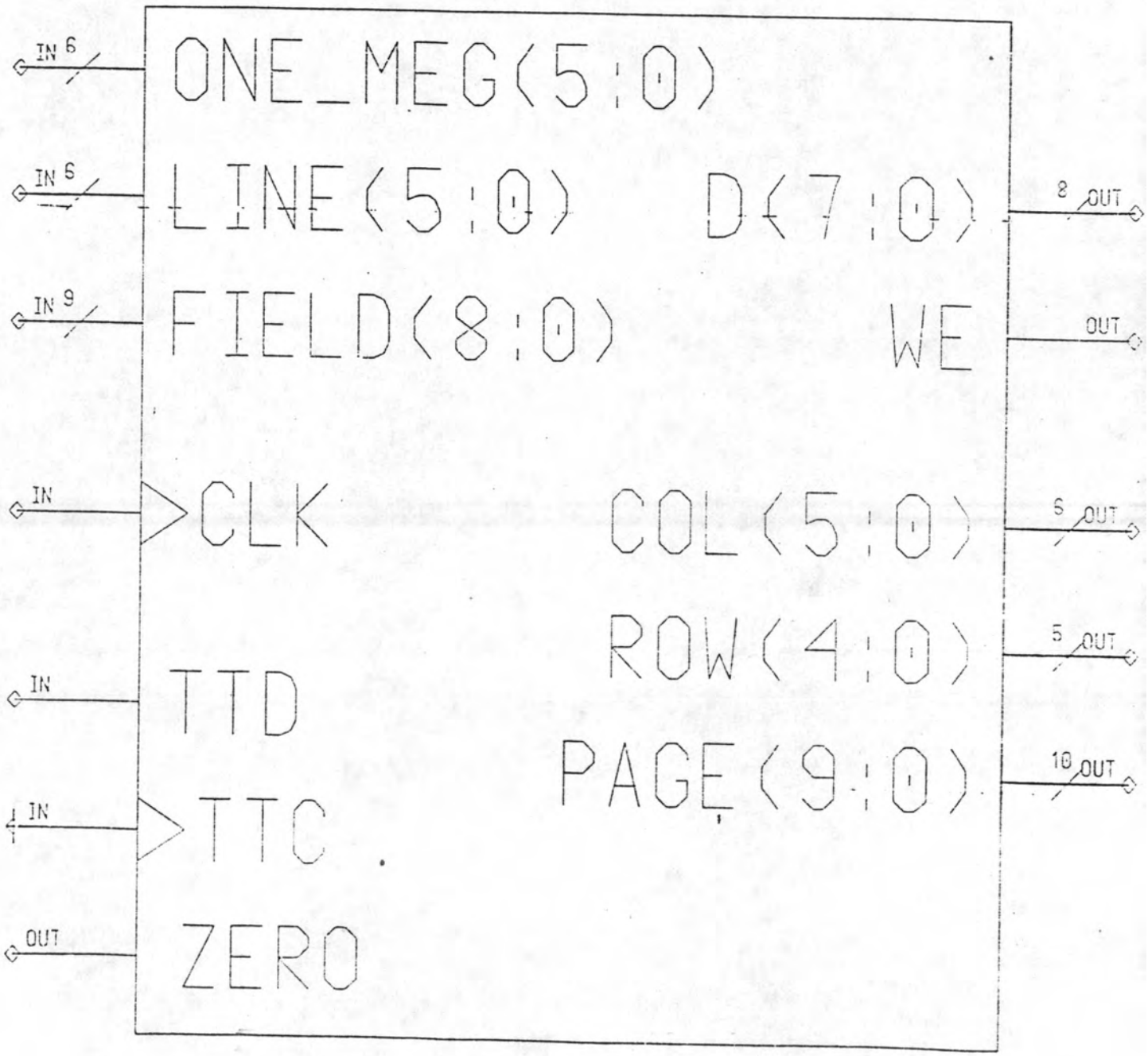

IN
CLK
A(18:0)
19
OUT
IN
VCS
D(7:0)
8
IXC
OUT
TCS
OUT
PL
ROMB
OUT
OUT
CBB
RAM1B
OUT
RAM2B
OUT
IN
TTC
IN
TTD
WRB'
OUT
RDB
OUT
OUT
R
OUT
GREMOTE(9:0)
8
IN
OUT
B
RESETB
IN

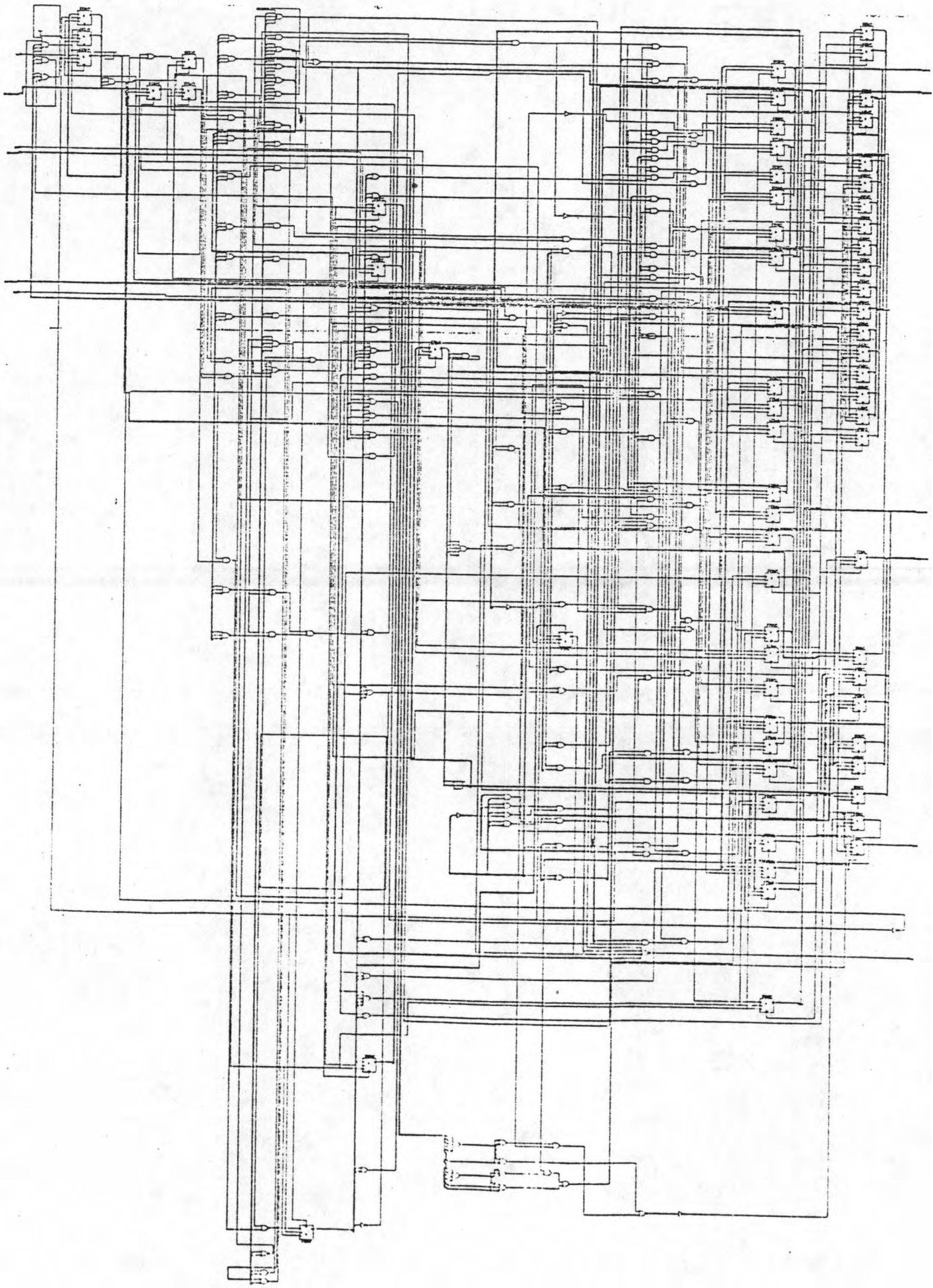

REMOTE(0)
REMOTE(1)
REMOTE(2)
REMOTE(3)
REMOTE(4)
REMOTE(5)
REMOTE(6)
REMOTE(7)
REMOTE(8)
REMOTE(9)
PXX
IBUF
CLK
VCS
TCS
PL
CBB
P35
P38
P39
P36
P37
BUFGP
OBUF
TTD
TTC
P60
P57
OBUFT
PXX
RESETB
TIMEBASE
ONE_MEG(5:0)
LINE(5:0)
FIELD(8:0)
TCS
PL
CBB
TEXTGEN
COL_R(5:0)
ROW_R(4:0)
ROC(3:0)
CHAR(7:0)
D(7:0)
R
G
B
P62
P41
P40
MEMINTERFACE
A(18:0)
D(7:0)
COL_R(5:0)
ROW_R(4:0)
PAGE_R(9:0)
CHAR(7:0)
ROC(3:0)
ROMB
RAM1B
RAM2B
WRB
RDB
COL_W(5:0)
ROW_W(4:0)
PAGE_W(9:0)
D_R(7:0)
D_W(7:0)
WE
ACQUISITION
ONE_MEG(5:0)
LINE(5:0)
FIELD(8:0)
D(7:0)
WE
CLK
COL(5:0)
ROW(4:0)
PAGE(9:0)
TTD
TTC
ZERO
STARTUP
GSR
GTS
CLK
Q2
Q3
Q1Q4
DONEIN


TIMEBASE
IN
CLK
IN
VCS
OUT
TCS
OUT
PL
OUT
CBG
ONE_MEG(5:0)
LINE(5:0)
FIELD(8:0)
6
OUT
6
OUT
9
OUT

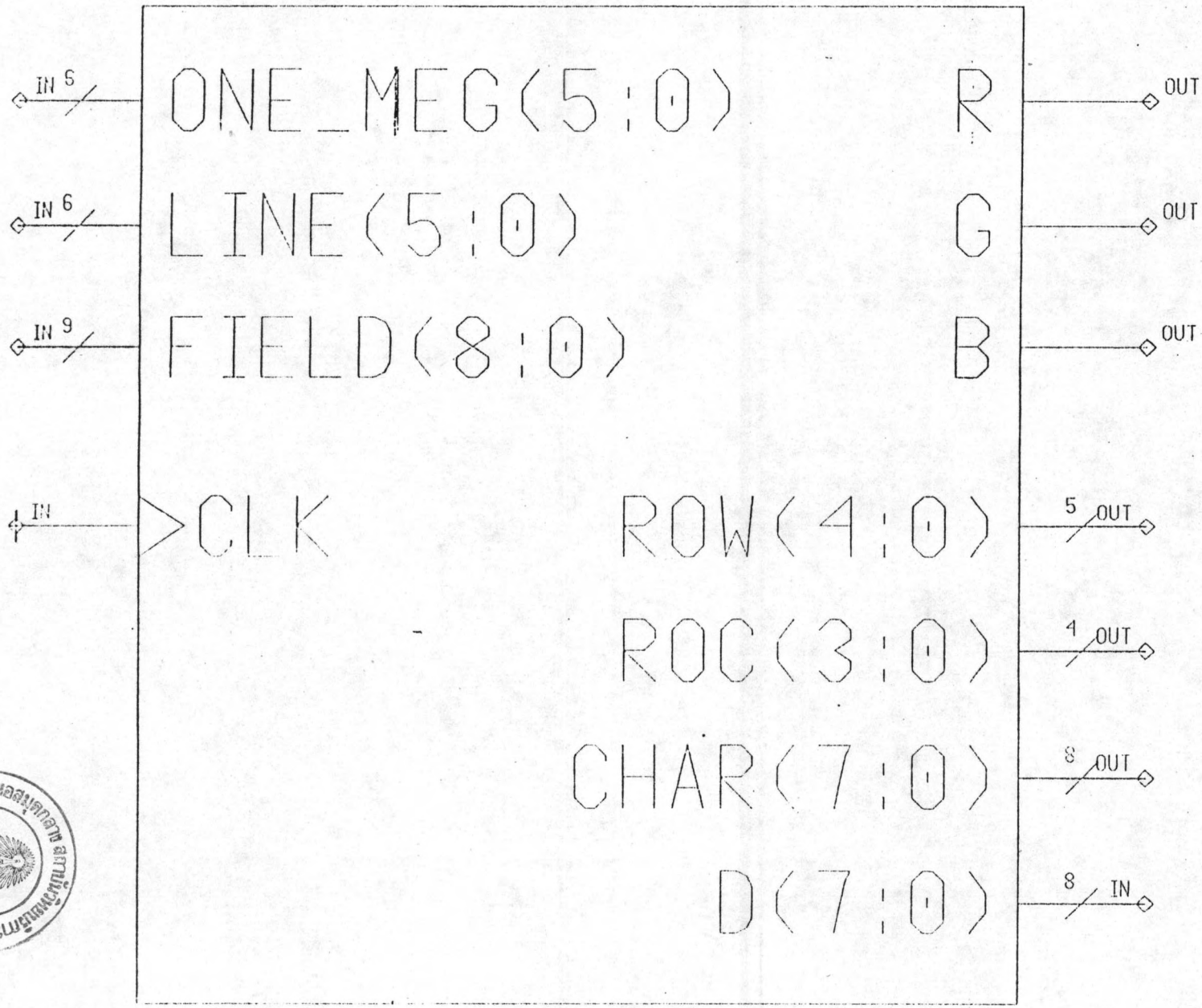

FDRD
FDSD

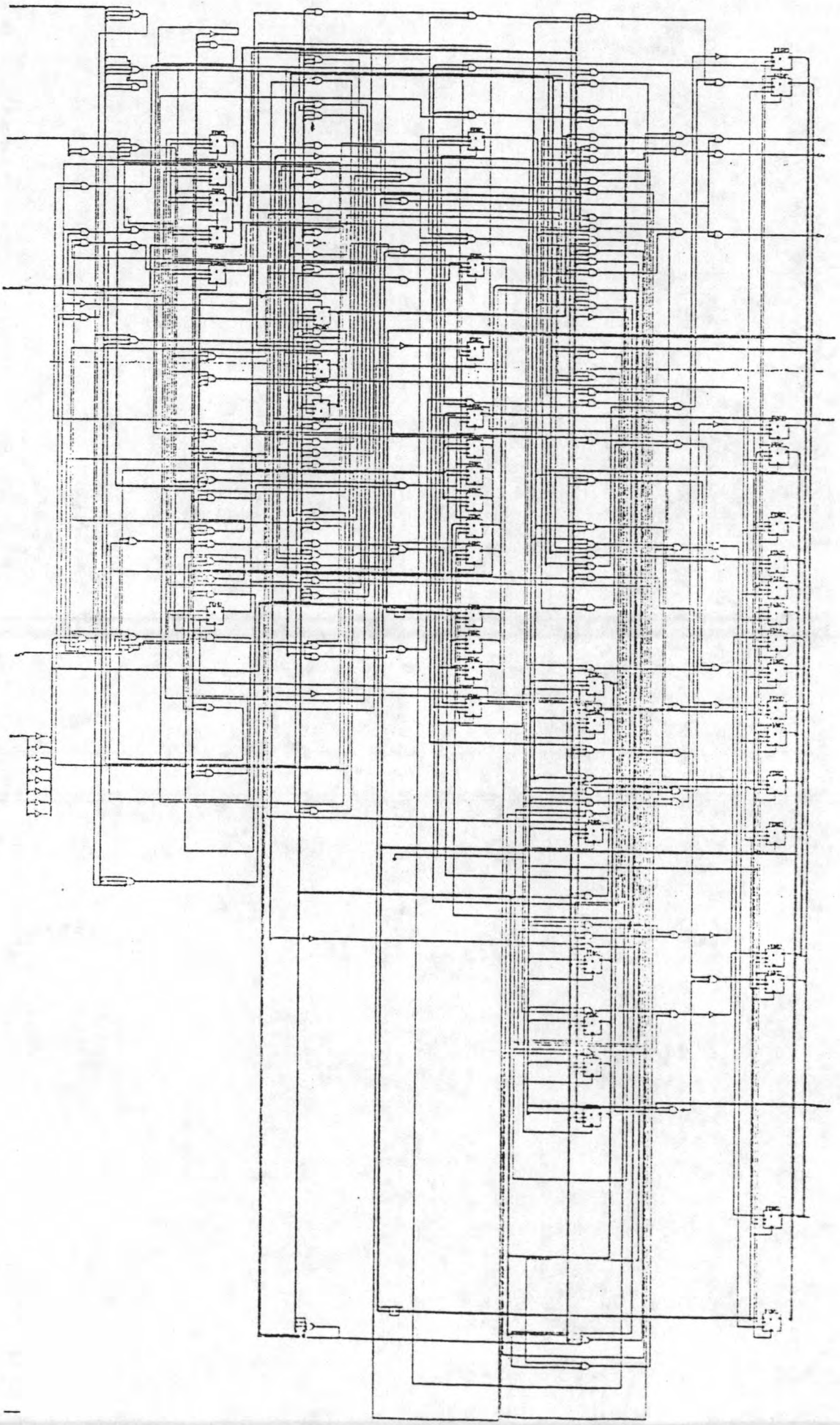
ACQUISITION



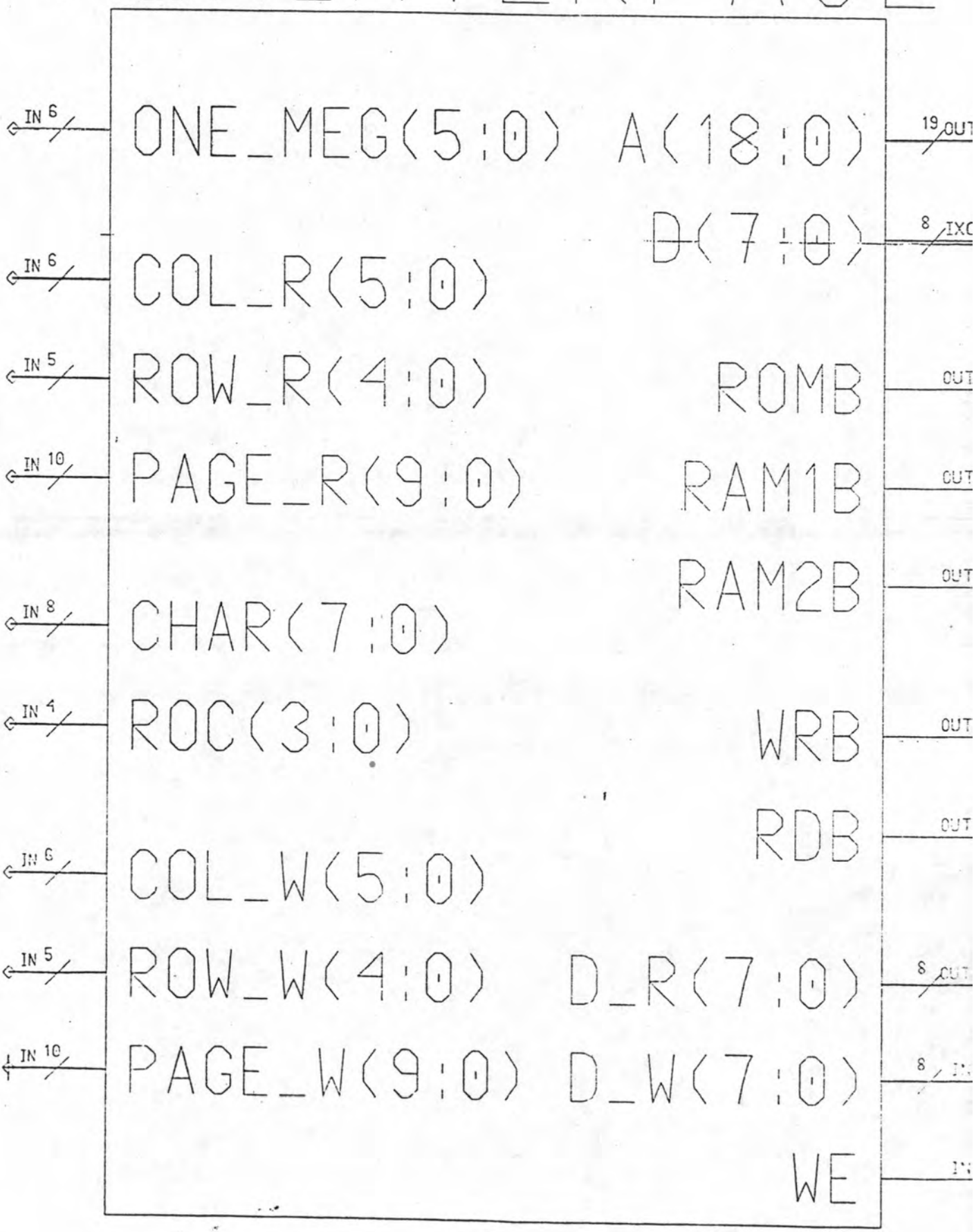


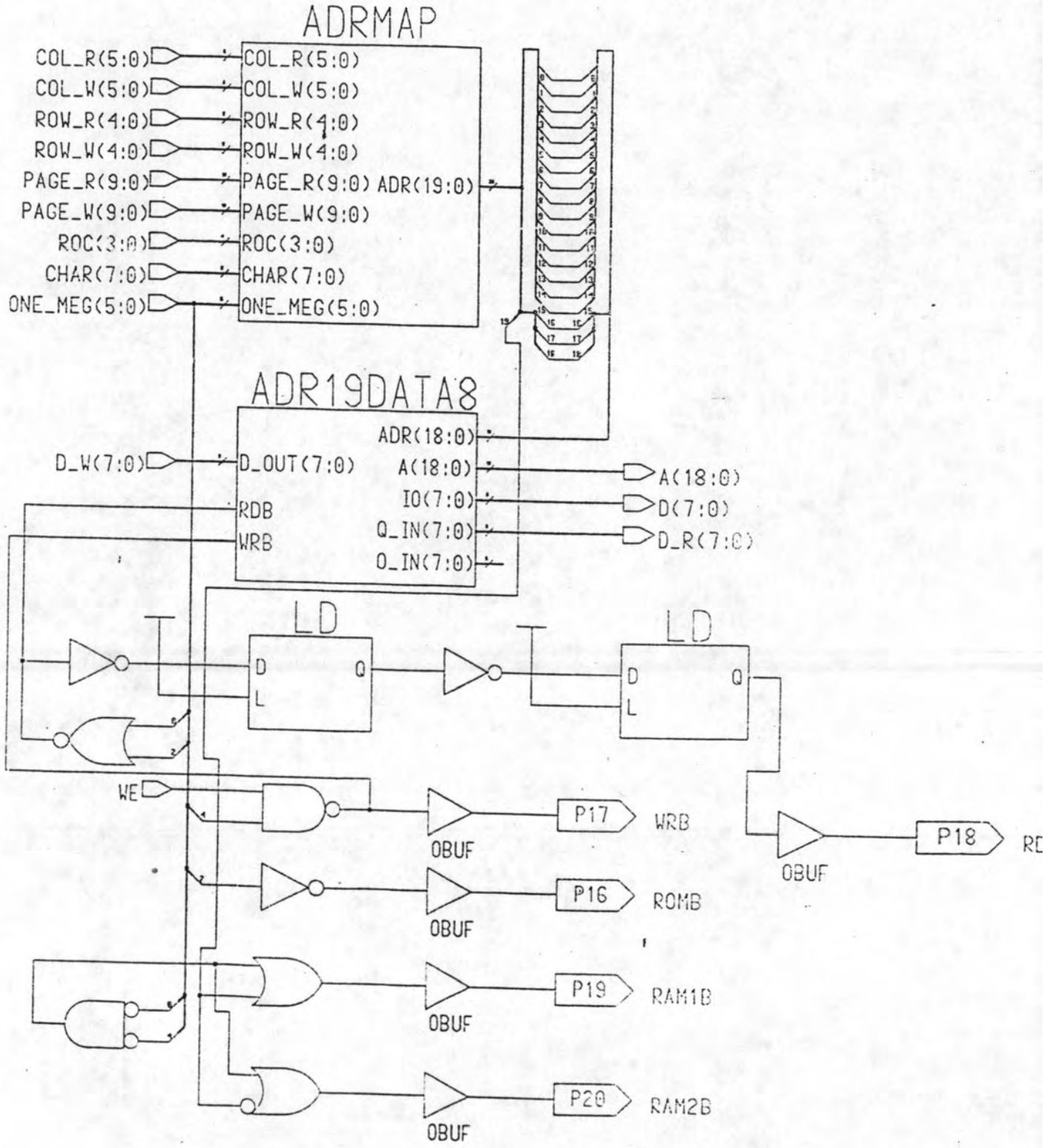
TEXTGEN





MEMINTERFACE




ADRMAP
COL_R(5:0)
COL_W(5:0)
ROW_R(4:0)
ROW_W(4:0)
PAGE_R(9:0)
PAGE_W(9:0)
ROC(3:0)
CHAR(7:0)
ONE_MEG(5:0)
ADR(19:0)
ADR19DATA8
D_W(7:0)
D_OUT(7:0)
RDB
WRB
ADR(18:0)
A(18:0)
IO(7:0)
Q_IN(7:0)
O_IN(7:0)
D(7:0)
D_R(7:0)
LD
D
Q
L
WE
OBUF
P17
WRB
P18
P16
ROMB
P19
RAM1B
P20
RAM2B

ประวัติผู้เขียน

นายวันเฉลิม โปรา เกิดวันที่ 5 ธันวาคม 2513 ณ.กรุงเทพมหานคร สำเร็จการศึกษาปริญญาตรีวิศวกรรมศาสตร์ สาขาวิศวกรรมไฟฟ้า จากจุฬาลงกรณ์มหาวิทยาลัยในปีการศึกษา 2535 และเข้าศึกษาต่อในหลักสูตรวิศวกรรมศาสตรมหาบัณฑิต ที่จุฬาลงกรณ์มหาวิทยาลัย เคยผ่านการดูงานและอบรมสัมมนาในการออกแบบ วงจรรวมหลายครั้งทั้งในและนอกประเทศ ปัจจุบันเป็นอาจารย์ประจำของจุฬาลงกรณ์มหาวิทยาลัย สังกัดห้องปฏิบัติการวิจัยระบบเชิงเลข