INTERNATIONAL ATOMIC ENERGY AGENCY
UNITED NATIONS EDUCATIONAL, SCIENTIFIC AND CULTURAL ORGANIZATION
INTERNATIONAL CENTRE FOR THEORETICAL PHYSICS
I.C.T.P., P.O. BOX 586, 34100 TRIESTE, ITALY, CABLE CENTRATOM TRIESTE

H4.SMR. 403/27

FIFTH COLLEGE ON MICROPROCESSORS: TECHNOLOGY AND APPLICATIONS IN PHYSICS

2 - 27 October 1989

Programmable Logic
Designing State Machines Using PALs: an Example

S. VENKATARAM
Microprocessors Laboratory, ICTP, Trieste, Italy

DESIGN EXAMPLE:

DESIGN OF A SIMPLE COMBINATION LOCK

INTRODUCTION:

- WE HAVE TO BUILD AN ELECTRONIC COMBINATION LOCK.
- COMBINATION LOCKS COME IN TWO FLAVOURS, SERIAL COMBINATION & PARALLEL COMBINATION
- A SERIAL LOCK IS A DIAL TYPE: THE DIAL IS ROTATED TO THREE OR MORE PRESET NUMBERS IN SEQUENCE.
- ANY WRONG NUMBER REQUIRES ONE TO START AGAIN
- A PARALLEL COMBINATION LOCK IS LIKE A BICYCLE CHAIN LOCK WITH THREE OR FOUR DISK THAT MUST BE ROATED TO THE CORRECT COMBINATION.
- EACH OF THE DISKS CAN BE ROTATED INDEPENDENTLY WITHOUT HAVING TO START ALL OVER IF THE COMBINATION IS WRONG.

STATING THE PROBLEM

- TO KEEP THE PROBLEM SIMPLE, WE DESIGN A SIMPLE SERIAL COMBINATION LOCK.
- WE ENTER THE COMBINATION IN BINARY FORM, ONE DIGIT AT A TIME.
- A WRONG DIGIT SHOULD ABORT THE WHOLE PROCEDURE, REQUIRING A SYSTEM RESET & A RESTART.

WE HAVE TO ASK SEVERAL QUESTIONS BEFORE DESIGNING THE ASM CHART, FOR INSTANCE WE CAN ASK.

(a) HOW MANY BIT COMBINATIONS SHOULD WE PROVIDE

(b) HOW IS THE DATA ENTERED?

(c) HOW DO WE RESET THE LOCK TO START A NEW SEQUENCE?

(d) HOW IS THE COMBINATION ENTERED, LEFT TO RIGHT OR RIGHT TO LEFT?

(e) HOW DO WE KNOW WE HAVE THE RIGHT COMBINATION OR AN ERROR STATE?

(f) WE MAY ANSWER THESE QUESTIONS AS FOLLOWS:

(a) WE CAN HAVE A MAXIMUM BIT COMBINATION OF 16 BITS. WE CAN VARY THE BIT COMBINATIONS FROM 0 - 16 TO KEEP THE DESIGN FLEXIBLE

(b) BINARY DATA IS SET BY AN OPERATOR ON A TOGGLE SWITCH. THE OPERATOR WILL SIGNAL THAT THE SWITCH DATA IS READY BY PRESSING A PUSH BUTTON CALLED READ.

(c) WE WILL USE A RESET SWITCH / RESET PUSH BUTTON

(d) WE ENTER DATA RIGHT TO LEFT

(c) WE CAN SIGNAL THE RIGHT COMBINATION BY LIGHTING A LIGHT EMITTING DIODE (LED). WE KEEP QUIET DURING AN ERROR STATE, SO AS TO NOT TO GIVE THE PROGRAMMED COMBINATION AWAY.

WE DO NOT TELL SOMEONE WHO TRIES THE LOCK HOW MANY BITS TO ENTER, THEREFORE LETTING HIM ENTER 5, 6 OR MORE DIGITS. WE ADD A PUSH BUTTON CALLED 'TRY' TO TEST THE ENTERED COMBINATION.

IF THE 'TRY' BUTTON IS PUSHED BEFORE OR AFTER PROCESSING M COMBINATION DATA WE SHOULD TERMINATE THE OPERATION IN AN ERROR STATE.

FROM THE ABOVE ANSWERS WE MIGHT HAVE A FEW MORE QUESTIONS LIKE:

(i) HOW IS A COMBINATION PROGRAMMED?

(ii) HOW DO WE SET M (≤16) COMBINATIONS BEFORE THE SYSTEM?

WE HAVE THE FOLLOWING ANSWERS:

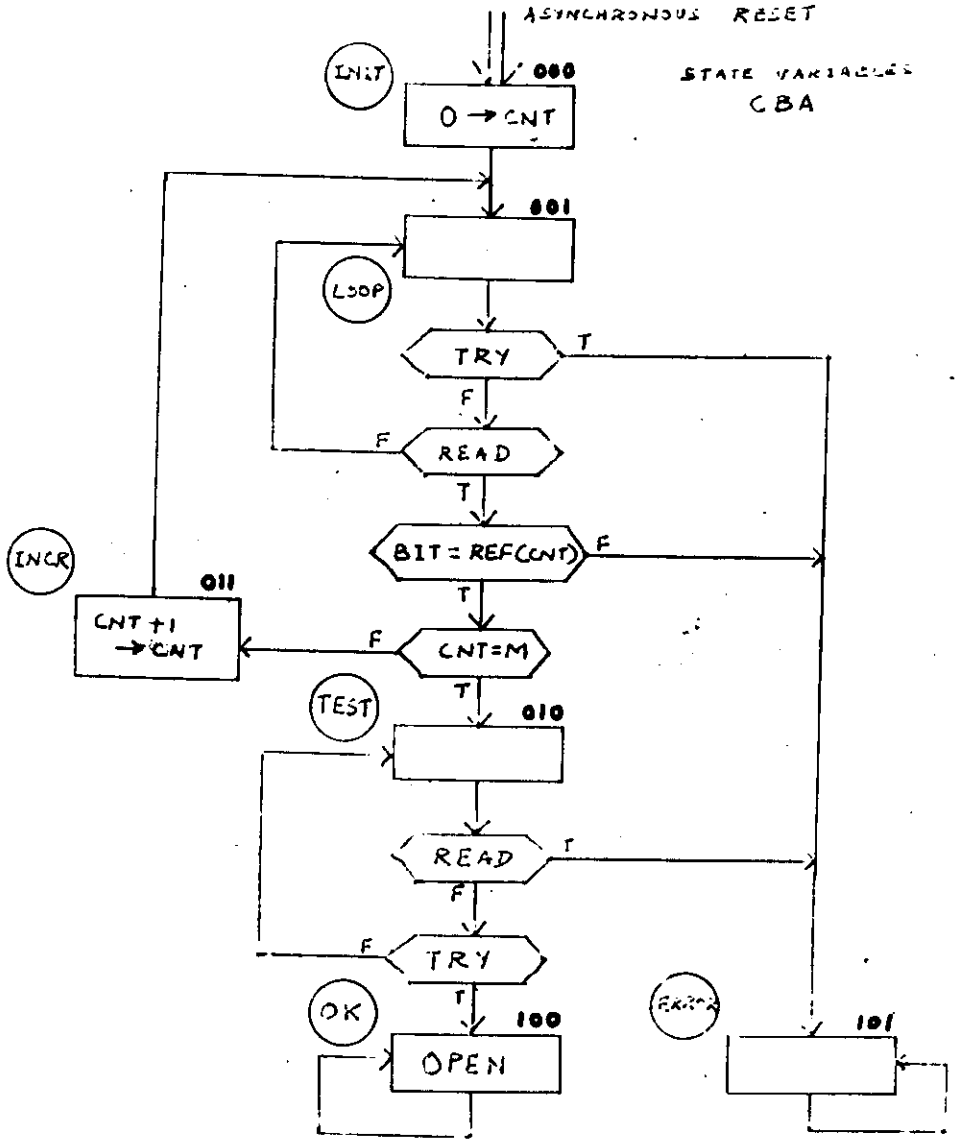
(i) WE SET THE COMBINATION THRU' 16 - TOGGLE SWITCHES AND USE A TABLE LOOK UP METHOD TO COMPARE THE COMBINATION BIT BY BIT AS THE DATA IS ENTERED.

(ii) AS THE DATA IS BEING ENTERED, A COUNTER CAN BE INCREMENTED. THE COUNTER'S OUTPUTS CAN BE TESTED FOR THE TERMINAL COUNT OF 'M' (≤16). AGAIN 'M' CAN BE PROGRAMMED THRU' TOGGLE SWITCHES.

BUILDING THE ALGORITHM

- THE RESET PUTS THE SYSTEM IN A KNOWN STATE REGARDLESS OF THE PRESENT STATE AND THE ASM CLOCK.
- IN OUR CASE IT IS AN ASYNCHRONOUS EVENT THAT PUTS OUR ASM IN AN INITIAL STATE: NOTHING NEEDS TO BE SAVED IN THE ASM.
- WE CHOOSE THE STATE VARIABLES TO BE ALL ZEROS IN THE INITAL STATE AS THE DESTINATION OF THE ASYNCHRONOUS RESET.
- OUR MACHINE TESTS THE DATA ENTERED, BIT BY BIT. WE HAVE A COUNTER CALLED 'CNT' THAT
  - PRESENTS THE ADDRESS TO OUR LOOK-UP TABLE,
  - COUNTS THE NUMBER OF BITS OF DATA THAT HAS BEEN ENTERED.
- OUR MACHINE NEED NOT STORE ALL THE BITS THAT ARE ENTERED EXCEPT THE PRESENT BIT FOR COMPARING WITH THE TABLE. PREVIOUS VALUE CAN BE DISCARDED.

THE ASM CHART FOR THE SERIAL COMBINATION LOCK IS SHOWN BELOW:



WE TAKE THE MULTIPLEXER METHOD OF SYNTHESIS AND ARRANGE THE STATE TRANSITION DATA SUITABLY:

	PRESENT STATE	NEXT STATE C B A	CONDITION FOR ASM
0	INIT	LOOP 0 0 1	T
1	LOOP	LOOP 0 0 1	$\overline{TRY} \cdot \overline{READ}$
		TEST 0 1 0	$\overline{TRY} \cdot READ \cdot (BIT = REF(CNT)) \cdot (CNT = M)$
		INCR 0 1 1	$\overline{TRY} \cdot READ \cdot (BIT = REF(CNT)) \cdot \overline{(CNT = M)}$
		ERROR 1 0 1	$TRY + \overline{TRY} \cdot READ \cdot \overline{(BIT = REF(CNT))}$
			$= TRY + READ \cdot \overline{(BIT = REF(CNT))}$
2	TEST	TEST 0 1 0	$\overline{READ} \cdot \overline{TRY}$
		OK 1 0 0	$\overline{READ} \cdot TRY$
		ERROR 1 0 1	READ
3	INCR	LOOP 0 0 1	T
4	OK	OK 1 0 0	T
5	ERROR	ERROR 1 0 1	T
	(any)	INIT 0 0 0	RESET (ASYNCHRONOUS)

- WE WISH THAT THE LOCK YIELD NO INFORMATION EXCEPT TURNING AN 'OPEN' LIGHT FOLLOWING A SUCCESSFUL TRY.
- IF THE MACHINE'S CLOCK IS FASTER THAN HUMAN REACTION, WE CAN AVOID 'HAND SHAKES' BETWEEN THE OPERATOR & THE MACHINE.
- THE 'TRY' AND 'READ' SIGNALS ARE DEBOUNCED PUSH BUTTON SIGNALS. ALSO THEY ARE SYNCHRONISED

THE DATA-ENTRY SWITCH FOR THE LOCK MAY BE A PLAIN TOGGLE SWITCH.

THE 'OPEN' OUTPUT DRIVES AN LED WHEN PROPER CONDITIONS ARE MET.

THERE IS A BINARY COUNTER WITH OUTPUT 'CNT' TO CONTROL THE LOOP COUNT. A 74LS163 FOUR BIT PROGRAMMABLE BINARY COUNTER CAN BE USED FOR OUR PURPOSE. 'CNT' WILL THEN CONSIST OF 4-BITS CNT0-CNT3

AN EIGHT BIT OR 16 BIT MUX WITH OUTPUT 'REF(CNT)' WILL BE USED TO PROVIDE INDEXED TABLE LOOKUP FOR BIT COMBINATIONS. WE CAN USE 74LS151 OR 74LS154

THE 'CNT=M' BLOCK HAS INPUTS CNT AND M AND OUTPUT ('CNT=M') THAT DETERMINES THE TERMINAL COUNT. SINCE THE LOOP STARTS WITH A COUNT OF ZERO, 'M' MUST BE ONE LESS THAN THE NUMBER OF BITS IN THE COMBINATION. WE USE TOGGLE SWITCHES TO SET 'M' AND A 74L381 4-BIT MAGNITUDE COMPARATOR WITH 4-BIT INPUTS 'CNT' & 'M' & AN OUTPUT (CNT=M).

CONTROL / CONTROL SIGNALS

CNT(CLR) = INIT

CNT(COUNT) = INCR

OPEN = OK.

INPUTS TO MULTIPLEXERS

SINCE WE HAVE '6' STATES WE NEED '3' FLIP FLOPS TO ENCODE THE STATES AND '3' MULTIPLEXERS AS INPUTS TO EACH OF THE FLIP-FLOPS. THE MULTIPLEXERS WILL HAVE '8' INPUTS.

THE EQUATION FOR THE INPUTS WILL BE:

MUXA(0) = T

MUXA(1) = $\overline{TRY} \cdot \overline{READ} + \overline{TRY} \cdot READ \cdot (BIT = REF(CNT)) \cdot (\overline{CNT = M}) +$
$TRY + READ \cdot (\overline{BIT = REF(CNT)})$

MUXA(2) = READ

MUXA(3) = T

MUXA(4) = F

MUXA(5) = T

MUXA(6) = F

MUXA(7) = F

MUXB(0) = F

MUXB(1) = $\overline{TRY} \cdot READ \cdot (BIT = REF(CNT)) \cdot (CNT = M) +$
$\overline{TRY} \cdot READ \cdot (BIT = REF(CNT)) \cdot \overline{(CNT = M)}$

MUXB(2) = $\overline{READ} \cdot \overline{TRY}$

MUXB(3) = F

MUXB(4) = F

MUXB(5) = F

MUXB(6) = F

MUXB(7) = F

MUXC(0) = F

MUXC(1) = TRY + $\overline{TRY}$. READ . (BIT = REF(CNT))

MUXC(2) = TRY . $\overline{READ}$ + READ

MUXC(3) = F

MUXC(4) = T

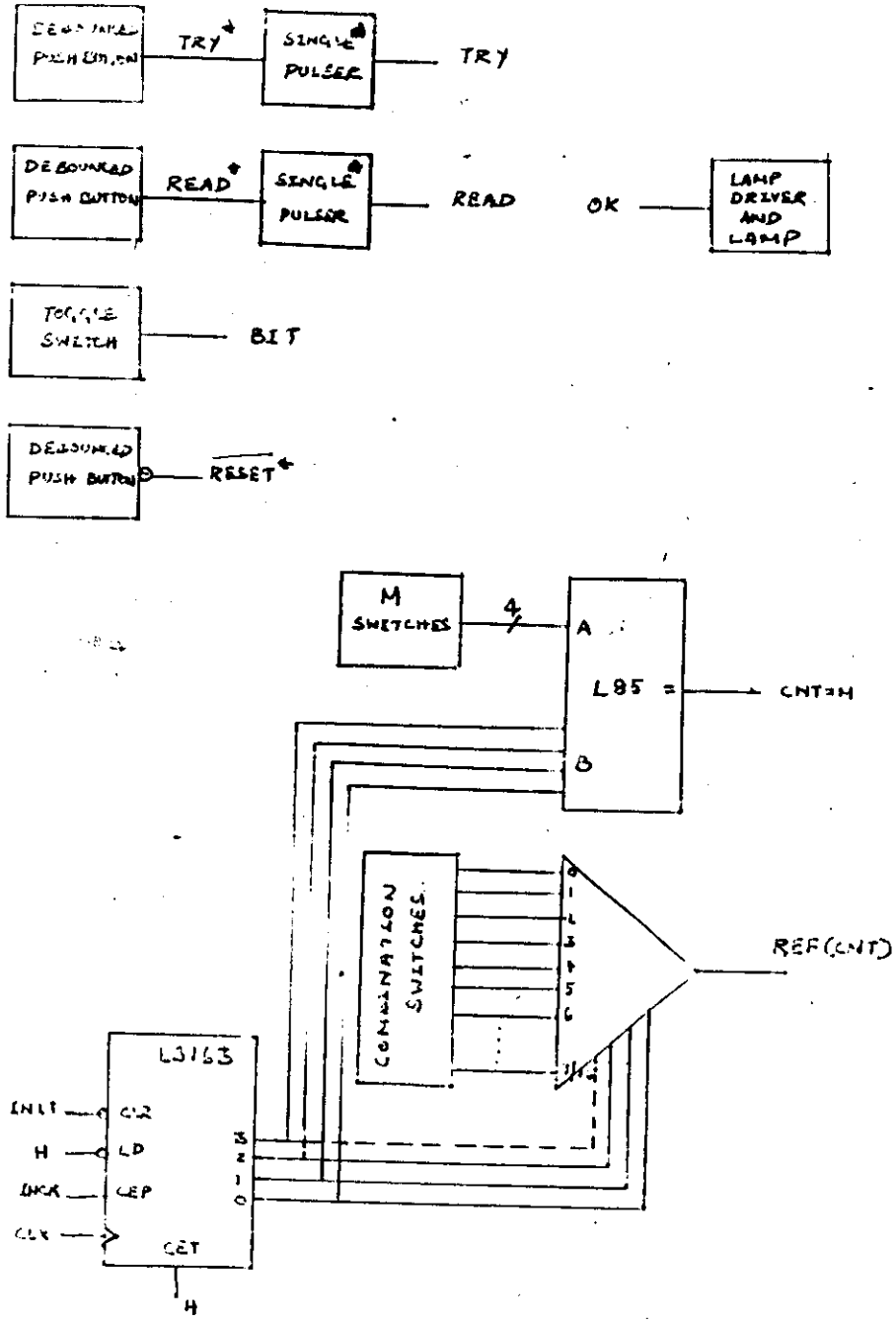
MUXC(5) = T

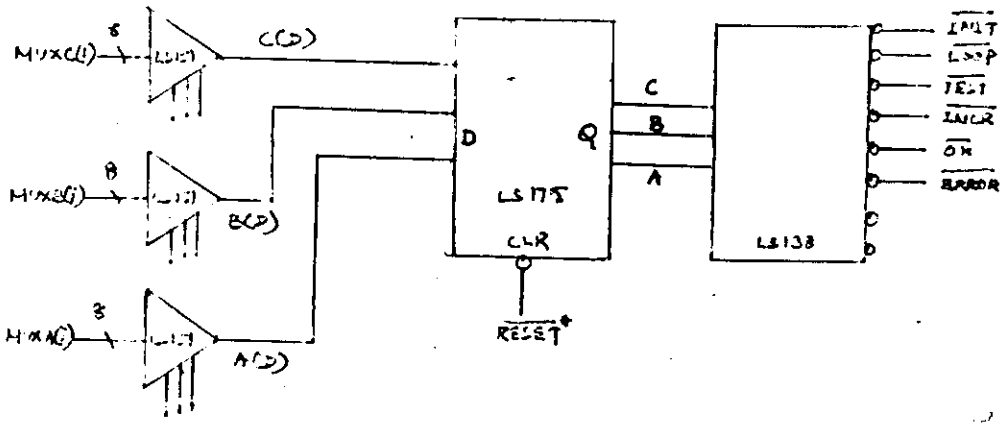
MUXC(6) = F

MUXC(7) = F

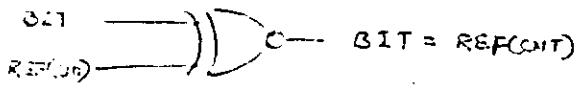
* ASYNCHRONOUS

+ GIVES ONLY A SINGLE PULSE TO THE STATE MACHIN





THE STATE GENERATOR FOR THE LOCK



CONTROL INPUT

NEXT STATE	PRESENT STATE	CONDITION
LOOP	INIT	T
	LOOP	$\overline{TRY} \cdot \overline{READ}$
	INCR	T
TEST	LOOP	$\overline{TRY} \cdot READ \cdot (BIT = REF(CNT)) \cdot (CNT=1$
	TEST	$\overline{READ} \cdot \overline{TRY}$
OK	TEST	$\overline{READ} \cdot TRY$
	OK	T
ERROR	LOOP	TRY
	LOOP	$\overline{TRY} \cdot READ \cdot \overline{(BIT = REF(CNT))}$
	TEST	READ
	ERROR	T
INCR	LOOP	$\overline{TRY} \cdot READ \cdot (BIT = REF(CNT)) \cdot \overline{(CNT=M}$
INIT	ANY	RESET.

IN THE ONE-HOT CONTROLLER METHOD OF IMPLEMENTIN THE LOCK, THERE WILL BE '6' D-FLIP-FLOPS. THE 'D' INPUT OF THE FLIP-FLOPS WILL BE: (USING THE TRANSITION DATA).

NEXT_STATE_LOOP(D) = INIT + LOOP. $\overline{TRY}$. $\overline{READ}$ + INCR

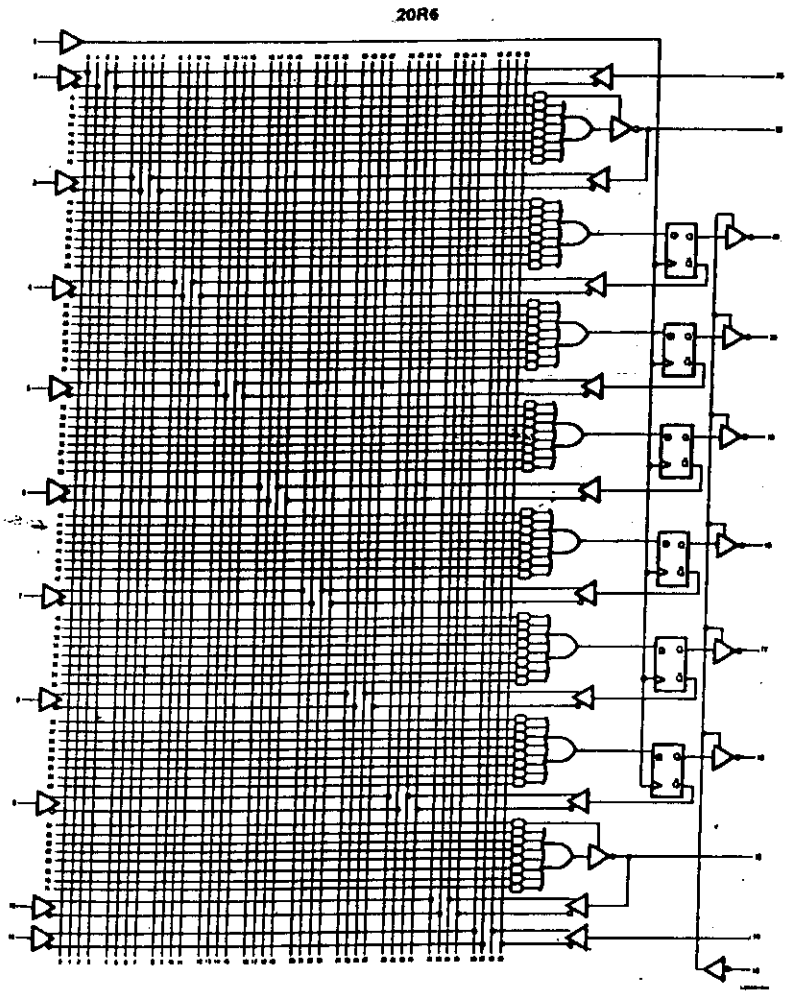
NEXT_STATE_TEST(D) = LOOP. $\overline{TRY}$. READ. (BIT = REF(CNT)). (CNT=M
+ TEST. $\overline{READ}$. $\overline{TRY}$

NEXT_STATE_OK(D) = TEST. $\overline{READ}$. TRY + OK

NEXT_STATE_ERROR(D) = LOOP. TRY + LOOP. $\overline{TRY}$. READ. $\overline{(BIT=REF(CNT}$
+ TEST. READ + ERROR

NEXT_STATE_INCR(D) = LOOP. $\overline{TRY}$. READ. (BIT=REF(CNT)). $\overline{(CNT=M)}$

NEXT_STATE_INIT(D) = (INIT+LOOP+TEST + OK +ERROR+ INCR)
RESET



(40) SOURCE FILE FOR THE CUPL SOFTWARE

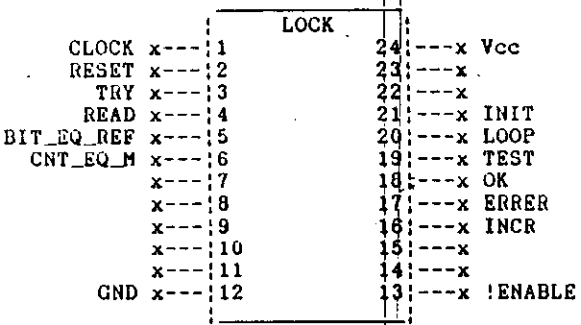
```
Name     LOCK;
Partno   12345;
Date     12/10/89;
Revision 00;
Designer VENKA;
Company  MICROLAB;
Assembly NIL;
Location ICTP;

/*****************************************************************/
/*                                                               */
/*                                                               */
/*                                                               */
/*****************************************************************/
/*  Allowable Target Device Types: PAL20R6                       */
/*****************************************************************/

/**  Inputs  **/

Pin  1      = CLOCK;          /*                                 */
Pin  2      = RESET;          /*                                 */
Pin  13     = !ENABLE;        /*                                 */
Pin  3      = TRY;            /*                                 */
Pin  4      = READ;           /*                                 */
Pin  5      = BIT_EQ_REF;     /*                                 */
Pin  6      = CNT_EQ_M;       /*                                 */

/**  Outputs  **/

Pin 21      = INIT;           /*                                 */
Pin 20      = LOOP;           /*                                 */
Pin 19      = TEST;           /*                                 */
Pin 18      = OK;             /*                                 */
Pin 17      = ERRER;          /*                                 */
Pin 16      = INCR ;          /*                                 */


/** Declarations and Intermediate Variable Definitions **/

INIT.D  =   RESET & (INIT # LOOP # TEST # OK # ERRER # INCR);

LOOP.D  =   !RESET & (INIT
          # LOOP & !TRY & !READ
          # INCR);

TEST.D  =   !RESET & (LOOP & !TRY & READ & BIT_EQ_REF & CNT_EQ_M
          # TEST & !READ & !TRY);

OK.D    =   !RESET & (TEST & !READ &TRY
          # OK);

ERRER.D =   !RESET & (LOOP & TRY
          # LOOP & !TRY & READ & !BIT_EQ_REF
          # TEST & READ
          # ERRER);

INCR.D  =   !RESET & (LOOP & !TRY & READ & BIT_EQ_REF & !CNT_EQ_M);


/**  Logic Equations  **/
```

(41) PART OF LOCK.DOC OUTPUT OF CUPL SOFTWARE

```
                                   LOCK
*************************************************************************

CUPL            2.15b Serial# 6-00002-198
Device          p20r6  Library DLID-h-24-10
Created         Thu Oct 12 18:02:07 1989
Name            LOCK
Partno          12345
Revision        00
Date            12/10/89
Designer        VENKA
Company         MICROLAB
Assembly        NIL
Location        ICTP

=========================================================================
                                 Fuse Plot
=========================================================================

Pin #22
 0000 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0040 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0080 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0120 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0160 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0200 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0240 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0280 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #21
 0320 ----------x---x---x---x---x---x--------
 0360 -x-----------------------------------------
 0400 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0440 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0480 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0520 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0560 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0600 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #20
 0640 ----x-------x---------------x--------
 0680 --------x--x---------------x--------
 0720 ----------x---x-------------x--------
 0760 x-----------------------------------------
 0800 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0840 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0880 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 0920 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #19
 0960 ---------x---------x-------------------
 1000 --------x-------x----------------------
 1040 --------x--------x---------------------
 1080 --------x----x-----------------------
 1120 ----x-------------------------------------
 1160 x-----------------------------------------
 1200 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 1240 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #18
 1280 -----x---------------x----------------
 1320 -----------------x---x----------------
 1360 --------x-------------x----------------
 1400 x-----------------------------------------
 1440 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 1480 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 1520 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 1560 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #17
 1600 -----x-------x------x--------x------------
 1640 -----x---x-----------------x------------
 1680 ---------------x---x-------x------------
 1720 ---------x-----x---------x------------
 1760 x-----------------------------------------
 1800 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 1840 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 1880 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #16
 1920 ----x-------------------------------------
 1960 x-----------------------------------------
 2000 ---------x-------------------------------
 2040 ------------x-----------------------------
 2080 ----------------x-------------------------
 2120 ----- -------x-------------------------
 2160 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2200 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
Pin #15
 2240 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2280 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2320 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2360 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2400 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2440 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2480 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
 2520 xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
```

(+2)

Chip Diagram



PART OF LOCK.DOC

AN OUTPUT OF CUPL SOFTWARE

(+2) SOURCE FILE FOR THE CSIM SIMULATOR LOCK.SI

```
Name        LOCK;
Partno      12345;
Date        12/10/89;
Revision    00;
Designer    VENKA;
Company     MICROLAB;
Assembly    NIL;
Location    ICTP;

ORDER: CLOCK, %2, RESET, %2, TRY, %2, READ, %2, BIT_EQ_REF, %2,
       CNT_EQ_M, %2, !ENABLE, %4, INIT, %2, LOOP, %2, TEST, %2,
       OK, %2, ERRER, %2, INCR;

VECTORS:
$MSG  " C  R  T  R  B  C  !     I  L  T  O  E  I  ";
$MSG  " L  E  R  E  I  N  E     N  O  E  K  R  N  ";
$MSG  " O  S  Y  A  T  T  N     I  O  S     R  C  ";
$MSG  " C  E     D  :  :  A     T  P  T     E  R  ";
$MSG  " K  T        E  E  B                 R     ";
$MSG  "             Q  Q  L                       ";
$MSG  "             :  :  E                       ";
$MSG  "             R  M                          ";
$MSG  "             E                             ";
$MSG  "             F                             ";
$MSG  "-------------------     -------------------";
$MSG  "POWER ON RESET                             ";
       P  1  X  X  X  X  0     1  0  0  0  0  0
       C  1  X  X  X  X  0     H  L  L  L  L  L
       C  1  X  X  X  X  0     H  L  L  L  L  L
$MSG  "CHANGE OF STATE FROM INIT TO LOOP          ";
       C  0  0  0  0  0  0     L  H  L  L  L  L
       C  0  0  0  0  0  0     L  H  L  L  L  L
$MSG  "LOOP TO ERROR                              ";
       C  0  1  0  0  0  0     L  L  L  L  H  L
$MSG  "STAYS IN ERROR                             ";
       C  0  0  0  0  0  0     L  L  L  L  H  L
$MSG  "TAKE EVASIVE ACTION CALLED RESET TO INIT   ";
       C  1  0  0  0  0  0     H  L  L  L  L  L
$MSG  "CHANGE STATE FROM INIT TO LOOP             ";
       C  0  0  0  0  0  0     L  H  L  L  L  L
$MSG  "CHANGE STATE TO INCR                       ";
       C  0  0  1  1  0  0     L  L  L  L  L  H
$MSG  "CHANGE STATE TO LOOP                       ";
       C  0  0  0  0  0  0     L  H  L  L  L  L
$MSG  "STATE CNG FROM LOOP TO ERROR ON BIT<>REF   ";
       C  0  0  1  0  0  0     L  L  L  L  H  L
$MSG  "RESET TO GO TO INIT FROM ERROR             ";
       C  1  0  0  0  0  0     H  L  L  L  L  L
$MSG  "INIT TO LOOP                               ";
       C  0  0  0  0  0  0     L  H  L  L  L  L
$MSG  "LOOP TO TEST                               ";
       C  0  0  1  1  1  0     L  L  H  L  L  L
$MSG  "REMAIN IN STATE TEST                       ";
       C  0  0  0  0  0  0     L  L  H  L  L  L
$MSG  "TEST TO ERROR STATE                        ";
       C  0  0  1  0  0  0     L  L  L  L  H  L
$MSG  "COME OUT OF ERROR STATE TO INIT            ";
       C  1  0  0  0  0  0     H  L  L  L  L  L
$MSG  "PROCEDURE TO GO TO STATE TEST              ";
       C  0  0  0  0  0  0     L  H  L  L  L  L
       C  0  0  1  1  1  0     L  L  H  L  L  L
$MSG  "TEST TO STATE OK!                          ";
       C  0  1  0  0  0  0     L  L  L  H  L  L
$MSG  "REMAIN IN STATE OK                         ";
       C  0  0  0  0  0  0     L  L  L  H  L  L
$MSG  "RESET TO STATE INIT                        ";
       C  1  0  0  0  0  0     H  L  L  L  L  L
```

(44) OUTPUT OF THE SIMULATOR ON THE SCREEN

```
CSIM: CUPL Simulation Program
Version 2.15b Serial# 6-00002-198
Copyright (C)  1983,1987 Personal CAD Systems, Inc.

csima
 C  R  T  R  B  C  !     I  L  T  O  E  I
 L  E  R  E  I  N  E     N  O  E  K  R  N
 O  S  Y  A  T  T  N     I  O  S     R  C
 C  E     D  |  |  A     T  P  T     E  R
 K  T        E  E  B                 R
             Q  Q  L
             |  |  E
             R  M
             E
             F
-------------------     -----------------
POWER ON RESET
0001: P  1  X  X  X  X  0     1  0  0  0  0  0
0002: C  1  X  X  X  X  0     H  L  L  L  L  L
0003: C  1  X  X  X  X  0     H  L  L  L  L  L
CHANGE OF STATE FROM INIT TO LOOP
0004: C  0  0  0  0  0  0     L  H  L  L  L  L
0005: C  0  0  0  0  0  0     L  H  L  L  L  L
LOOP TO ERROR
0006: C  0  1  0  0  0  0     L  L  L  L  H  L
STAYS IN ERROR
0007: C  0  0  0  0  0  0     L  L  L  L  H  L
TAKE EVASIVE ACTION CALLED RESET TO INIT
0008: C  1  0  0  0  0  0     H  L  L  L  L  L
CHANGE STATE FROM INIT TO LOOP
0009: C  0  0  0  0  0  0     L  H  L  L  L  L
CHANGE STATE TO INCR
0010: C  0  0  1  1  0  0     L  L  L  L  L  H
CHANGE STATE TO LOOP
0011: C  0  0  0  0  0  0     L  H  L  L  L  L
STATE CNG FROM LOOP TO ERROR ON BIT<>REF
0012: C  0  0  1  0  0  0     L  L  L  L  H  L
RESET TO GO TO INIT FROM ERROR
0013: C  1  0  0  0  0  0     H  L  L  L  L  L
INIT TO LOOP
0014: C  0  0  0  0  0  0     L  H  L  L  L  L
LOOP TO TEST
0015: C  0  0  1  1  1  0     L  L  H  L  L  L
REMAIN IN STATE TEST
0016: C  0  0  0  0  0  0     L  L  H  L  L  L
TEST TO ERROR STATE
0017: C  0  0  1  0  0  0     L  L  L  L  H  L
COME OUT OF ERROR STATE TO INIT
0018: C  1  0  0  0  0  0     H  L  L  L  L  L
PROCEDURE TO GO TO STATE TEST
0019: C  0  0  0  0  0  0     L  H  L  L  L  L
0020: C  0  0  1  1  1  0     L  L  H  L  L  L
TEST TO STATE OK!
0021: C  0  1  0  0  0  0     L  L  L  H  L  L
REMAIN IN STATE OK
0022: C  0  0  0  0  0  0     L  L  L  H  L  L
RESET TO STATE INIT
time: 12 secs
total time: 13 secs

C:\CUPL>
```

(45) OUTPUT OF CUPL SOFTWARE LOCK.JED

```
CUPL            2.15b  Serial# 6-00002-198
Device          p20r6  Library DLIB-h-24-10
Created         Thu Oct 12 18:02:07 1989
Name            LOCK
Partno          12345
Revision        00
Date            12/10/89
Designer        VENKA
Company         MICROLAB
Assembly        NIL
Location        ICTP
*QP24
*QF2560
*QV22
*G0
*F0
*L0320 11111111111011101110111011101110
*L0352 11111111101111111111111111111111
*L0384 11111111111111110000000000000000
*L0640 11110111111011111111111111111110
*L0672 11111111111111110110111111111111
*L0704 11111110111111111111111111101110
*L0736 11111111111111101111111101111111
*L0768 11111111111111111111111111111111
*L0960 11111111101111111110111111111111
*L0992 11111111111111101111110111111111
*L1024 11111111111111111111111101111111
*L1056 10111111111111111111111111111111
*L1088 01111011111111111111111111111111
*L1120 11110111111111111111111111111111
*L1152 11111110111111111111111111111111
*L1184 11111111111111110000000000000000
*L1280 11111011111111111111111011111111
*L1312 11111111111111111111111111101110
*L1344 11111111111111111111111101111111
*L1376 11111110111111111111111101111111
*L1408 11111111111111111111111111111111
*L1600 11111011111101111101111111101111
*L1632 11111111111110111011111111111111
*L1664 11101111111111111111111111111110
*L1696 11101111111101111111111111111111
*L1728 10111110111111111111011111111111
*L1760 01111111111111111111111111111111
*L1792 11111111000000000000000000000000
*L1920 11110111111111111111111111111111
*L1952 11111110111111111111111111111111
*L1984 11111111111111111111111110111111
*L2016 11111111111111111111111111111111
*L2048 11111101111111111111111111111111
*L2080 11111111111111110111111111111111
*L2112 11111111111111111111101111111111
*L2144 11111111111111110000000000000000
*C7E12
*P 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
*V0001 P1XXXXXXXXXNOXX000001XXN
*V0002 C1XXXXXXXXNOXXLLLLLHXXN
*V0003 C1XXXXXXXXNOXXLLLLLHXXN
*V0004 C00000XXXXNOXXLLLLHLXXN
*V0005 C00000XXXXNOXXLLLLHLXXN
*V0006 C01000XXXXNOXXLHLLLLXXN
*V0007 C00000XXXXNOXXLHLLLLXXN
*V0008 C10000XXXXNOXXLLLLLHXXN
*V0009 C00000XXXXNOXXLLLLHLXXN
*V0010 C00110XXXXNOXXHLLLLLXXN
*V0011 C00000XXXXNOXXLLLLHLXXN
*V0012 C00100XXXXNOXXLHLLLLXXN
*V0013 C10000XXXXNOXXLLLLLHXXN
*V0014 C00000XXXXNOXXLLLLHLXXN
*V0015 C00111XXXXNOXXLLLHLLXXN
*V0016 C00000XXXXNOXXLLLHLLXXN
*V0017 C00100XXXXNOXXLHLLLLXXN
*V0018 C10000XXXXNOXXLLLLLHXXN
*V0019 C00000XXXXNOXXLLLLHLXXN
*V0020 C00111XXXXNOXXLLLHLLXXN
*V0021 C01000XXXXNOXXLLHLLLXXN
*V0022 C00000XXXXNOXXLLLLHLXXN
```