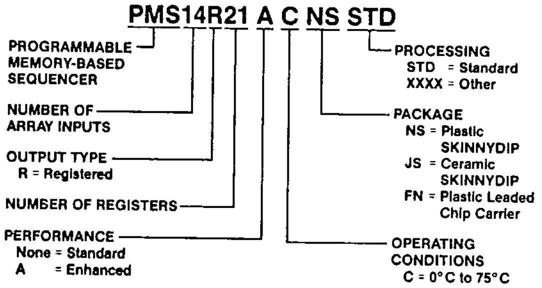
PROSE™ Programmable Sequencer

PMS14R21/A

Features/Benefits

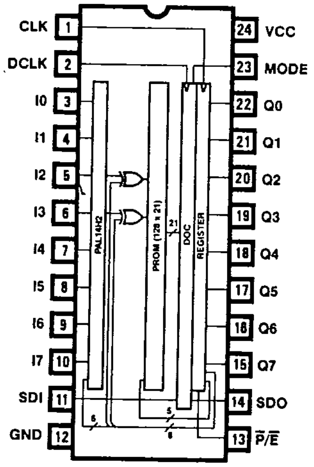
- User-programmable synchronous state machine
- 30 MHz maximum internal frequency, 25 MHz external
- Allows up to 128 states for complex designs
- 8 inputs and 8 outputs with 13 buried feedback signals
- PAL® array optimizes input decoding
- Four-way branching in one cycle
- User-selectable asynchronous preset or enable saves pins
- Power-up preset for start-up in known state
- Diagnostics-On-Chip™ shadow register eases chip and board testing by allowing preload and observation of output register
- Supported by PALASM®2 software and other development tools
- Programmable on standard logic programmers
- Security fuse prevents pattern duplication
- Space-saving 24-pin SKINNYDIP® and 28-pin PLCC packages

Ordering Information



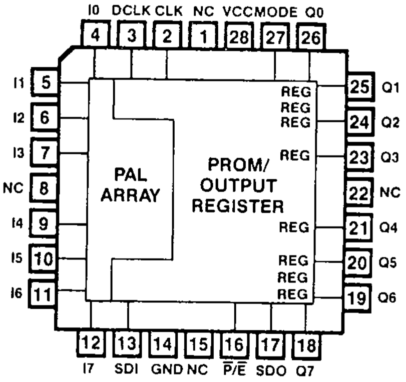
Pin Configurations

PMS14R21/A



SKINNYDIP Package

PMS14R21/A



Plastic Leaded Chip Carrier

Package Drawings

(refer to PAL Device Package Outlines)

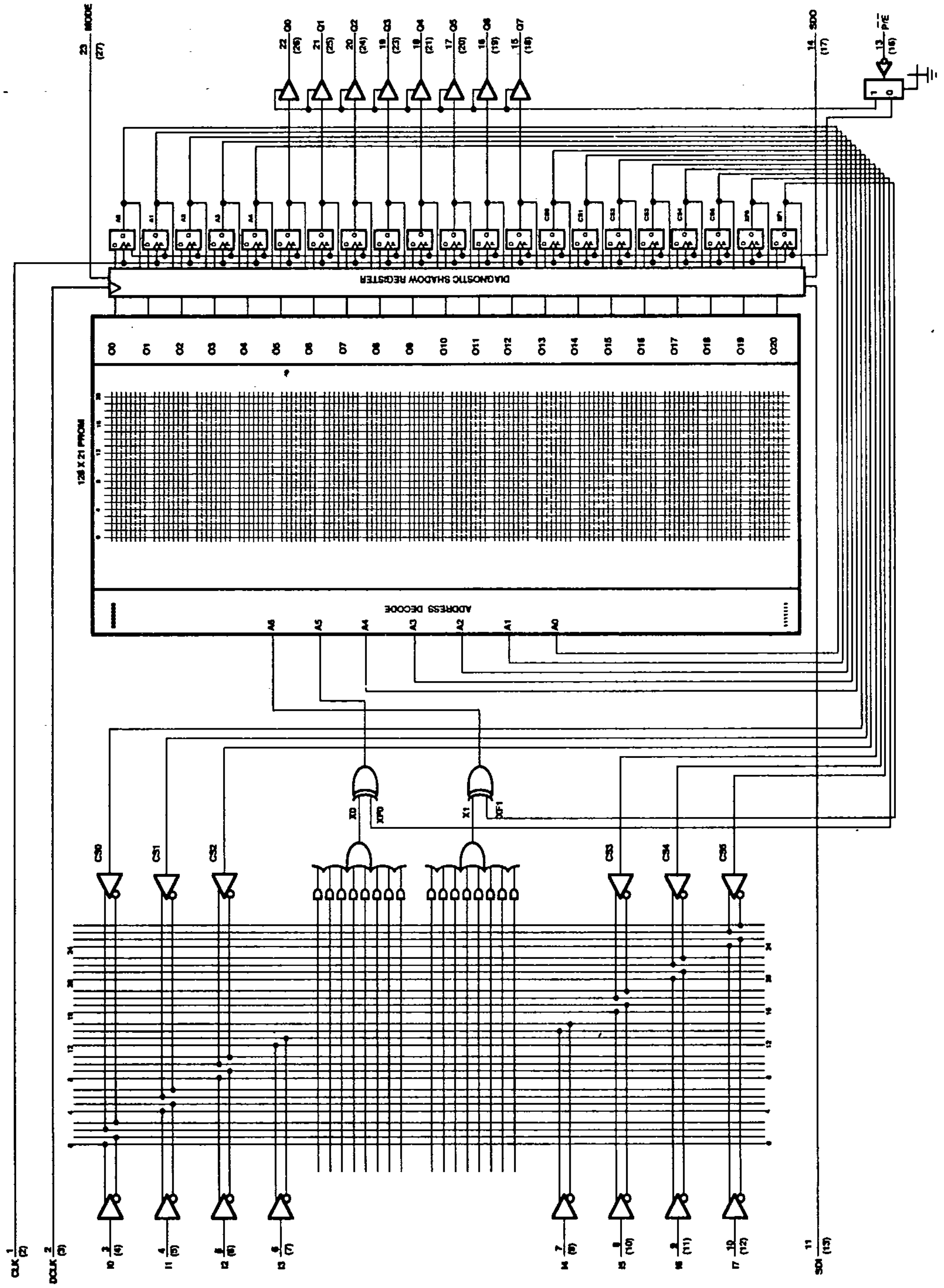
Description

The PMS14R21 combines the basic elements of a sequencer — the conditional input logic and state memory — into one programmable device. A 14H2 PAL array provides programmable input condition decoding while a 128-word by 21-bit PROM array stores the state information and outputs. The combination of both PAL and PROM arrays on a single chip provides very high performance.

T-46-13-35

Logic Diagram

DIP (PLCC) Pinouts



The device has a balanced architecture, with eight inputs for sequencer control and eight outputs for system control. Thirteen internal signals feed back from the state registers to control next state selection. The next state is selected by the five primary address signals and the two branch address signals, allowing up to four-way branching in one cycle.

Two branch control signals are generated by the PAL array as a function of the eight device inputs and six feedback signals. The feedback signals allow programmable condition select decoding on the inputs. The programmable decoding provides efficiency and speed even in complex designs. Exclusive-OR (XOR) gates on the PAL outputs, controlled by two additional feedback lines, help save device resources by allowing merging of states.

A 21-bit output register is paralleled by a 21-bit shadow register. The shadow register is a parallel/serial register that implements Diagnostics-On-Chip. In normal mode, the shadow register does not affect the operation of the state machine. In diagnostic mode, the shadow register allows control and observation of the output register.

The user can select either asynchronous preset or asynchronous output enable, whichever is required for the application. Preset is automatically performed on power up. A security fuse, once programmed, prevents a competitor from reading the PAL array pattern.

Definition of signals

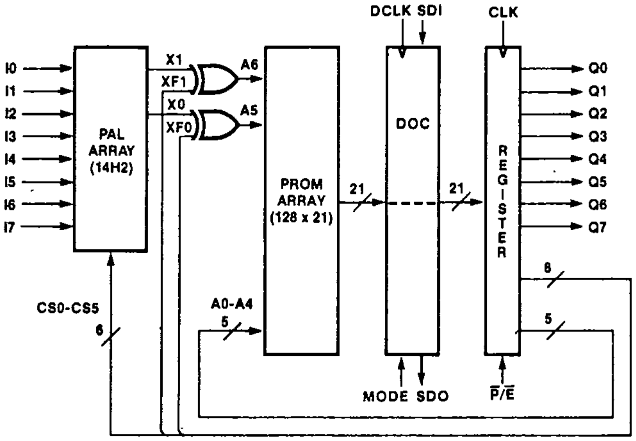
Pin Signals:

I0-I7	Inputs to PAL array
Q0-Q7	Registered outputs from PROM
P̅/E̅	Programmable asynchronous function pin: default is active low Preset (output register goes high), programmed is active low output enable
CLK	Output register clock
DCLK	Diagnostic register clock
MODE	Diagnostic mode selection
SDI	Serial Data Input to diagnostic register
SDO	Serial Data Output from diagnostic register

Internal Signals:

CS0-CS5	Condition Select feedback from output register to PAL array
X0-X1	PAL array outputs; inputs to XOR gates
XF0-XF1	XOR gate control feedback from output register
A5-A6	XOR gate outputs providing two highest order address bits to PROM array
A0-A4	Feedback from output register providing five lowest order address bits to PROM array
O0-O20	Output register contents
S0-S20	Shadow diagnostic register contents

Block Diagram



5

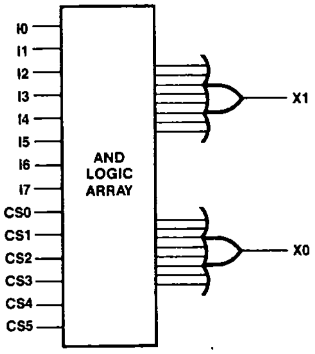
Architecture Description

Detailed knowledge of the PMS14R21 architecture is not necessary for use of the device; a design is entered into a software development system which automatically optimizes the design and creates a programming file. The user only needs to create a design description for the software to use. The following architecture description is provided for better knowledge of the device, but the software automatically handles the features described.

The following sections of the PROSE architecture are described in detail:

- PAL array
- Exclusive-OR (XOR) gates
- PROM array
- Output register
- Feedback to PAL array (CS0-CS5)
- Feedback to XOR gates (XF0-XF1)
- Feedback to PROM array (A0-A4)
- Programmable asynchronous Preset or Enable (P̅/E̅)
- Power-up preset
- Security fuse

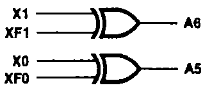
PAL Array



The PAL array implements the equivalent of a 14H2 circuit with fourteen inputs, two active high outputs, and eight product terms per output. Eight inputs come from pins (I0-I7) while the other six feed back from the output register (CS0-CS5).The outputs (X0-X1) feed two XOR gates.

The PAL array provides conditional input decoding, using the current inputs and state feedback to help control the two highest-order address bits to the PROM. The feedback signals enable the desired function of the inputs to force a branch to a given state. Default branches are handled easily because no product term will be on, providing 00 for X0-X1. Unconditional branches can be handled the same way, with no product terms on.

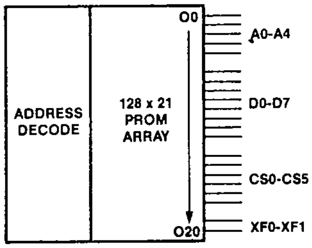
Exclusive-OR Gates



X0/1	XF0/1	A5/6
0	0	0
0	1	1
1	0	1
1	1	0

The two PAL array outputs (X0-X1) feed two Exclusive-OR (XOR) gates, with the other inputs (XF0-XF1) feeding back from the output register. The XOR gate outputs are the two highest-order address bits to the PROM. The XOR gates allow both the PAL array and the state to control the two address bits to the PROM, which provides four-way branching. This combination role allows the optimum use of device resources.

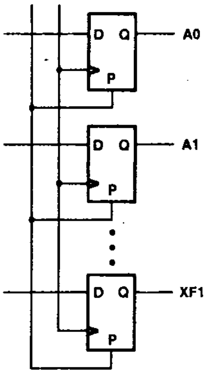
PROM Array



The PROM array is organized 128x21, or 128 words of 21 bits each, for a total of 2688 bits. The address lines are A0-A6, including A0-A4 from the output register and A5-A6 from the XOR gates. The PROM outputs (O0-O20) feed the 21-bit output register.

The PROM stores the state information just as a discrete PROM would in a discrete sequencer design. The 128 locations provide up to 128 possible states in an optimum design. The PROM outputs include eight fed to output pins for system control. The architecture directly implements a Moore machine, although a Mealy machine can be created by delaying the outputs from the state by one cycle. Thirteen PROM outputs feed back for extensive next-state control.

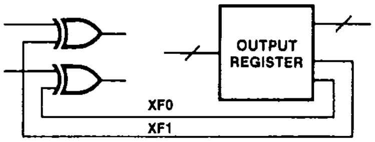
Output Register



The output register consists of twenty-one D-type flip-flops which are loaded from the PROM on the rising edge of the clock signal, CLK. Eight of the outputs lead to device output pins (Q0-Q7) while the other thirteen are fed back internally (A0-A4, XF0-XF1, CS0-CS5).

The thirteen buried flip-flops help control the sequencer, while the eight dedicated registered outputs provide system control resulting from the sequencer operation.

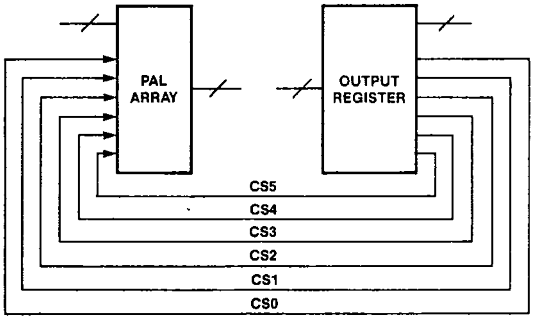
Feedback to XOR Gates (XF0-XF1)



Two bits of the output register (XF0-XF1) feed back to the XOR gates.

These two feedback signals allow either output of the PAL array to be inverted. This helps software optimization of resources by allowing the same product terms or PAL outputs to be used for different branches.

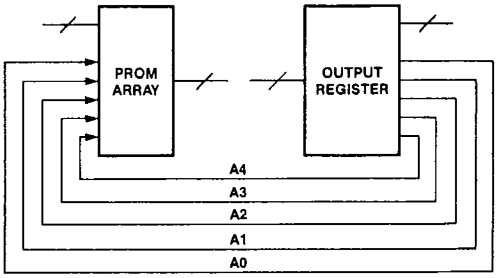
Feedback to PAL Array (CS0-CS5)



Six bits of the output register feed back to the PAL array. They are used as regular PAL array inputs.

These six Condition Select bits define the function to be performed on the eight external condition inputs. Any Boolean function may be performed, up to the limits of the PAL array architecture. The feedback signals generally select given combinations of inputs (product terms) necessary for branching to next states. For unconditional branches or default branches, they will not select a product term.

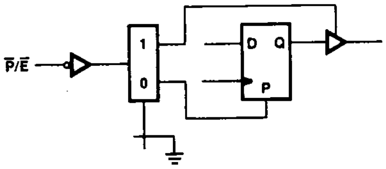
Feedback to PROM Array (A0-A4)



Five bits of the output register feed back to the PROM array. These signals become address lines A0-A4 for the PROM.

The register feedback to the PROM provides primary next state selection based only upon the current state. Address lines A0-A4 define four of the 128 specific locations within the PROM. The actual branch to one of these locations is selected by A5-A6 based on the current inputs.

Programmable Asynchronous Preset or Enable

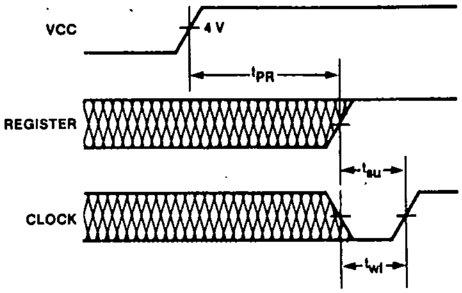


A programmable function pin allows the user to select either asynchronous preset or asynchronous output enable. The choice is programmed into the device when the arrays are programmed. The unprogrammed state is preset.

The programmable function allows the user to choose the function required for the design, saving precious pins. If preset is selected, a low signal on the pin will preset all twenty-one output flip-flops to the high state asynchronously. This is equivalent to the power-up state, and adds the capability of presetting without powering down. The outputs will always be enabled, and act as two-state outputs.

If enable is selected (i.e., the select fuse is programmed), a low signal on the pin will enable the eight output signals (Q0-Q7) from the output register. When the pin is high, the outputs are disabled, and the output pins are in the high-impedance state.

Power-Up Preset



The twenty-one output flip-flops will power up to the high state. This allows easy initialization, since the starting state is always known. The power-up preset time, t_{PR}, is 1 µs maximum. The required setup time and clock widths are listed in the specifications.

Security Fuse

After programming and verification, a PMS14R21 design can be secured by programming the security fuse. Once programmed, this fuse defeats readback of the PAL array fuse pattern by a device programmer, making proprietary designs very difficult to copy.

Absolute Maximum Ratings

	Operating	Programming
Supply voltage V_{CC}	-0.5 V to 7.0 V	-0.5 V to 12.0 V
Input voltage	-1.5 V to 5.5 V	-1.0 V to 12.0 V
Off-state output voltage	5.5 V	12.0 V
Storage temperature	-65°C to +150°C	

Sequencer Operating Conditions

SYMBOL	PARAMETER		COMMERCIAL[1]						UNIT
			STD			A			
			MIN	TYP	MAX	MIN	TYP	MAX	
V_{CC}	Supply voltage		4.75	5	5.25	4.75	5	5.25	V
t_w	Width of CLK	Low	25	15		20	15		ns
		High	10	5		10	5		
t_{su}	Setup time from I7-I0 to CLK		35	25		28	22		ns
t_h	Hold time for I7-I0 to CLK		0	-5		0	-5		ns
t_{aw}	Asynchronous preset width		10	5		10	5		ns
t_{ar}	Asynchronous preset recovery		20	10		20	10		ns
T_A	Operating free-air temperature		0	25	75	0	25	75	°C

Electrical Characteristics Over Operating Conditions

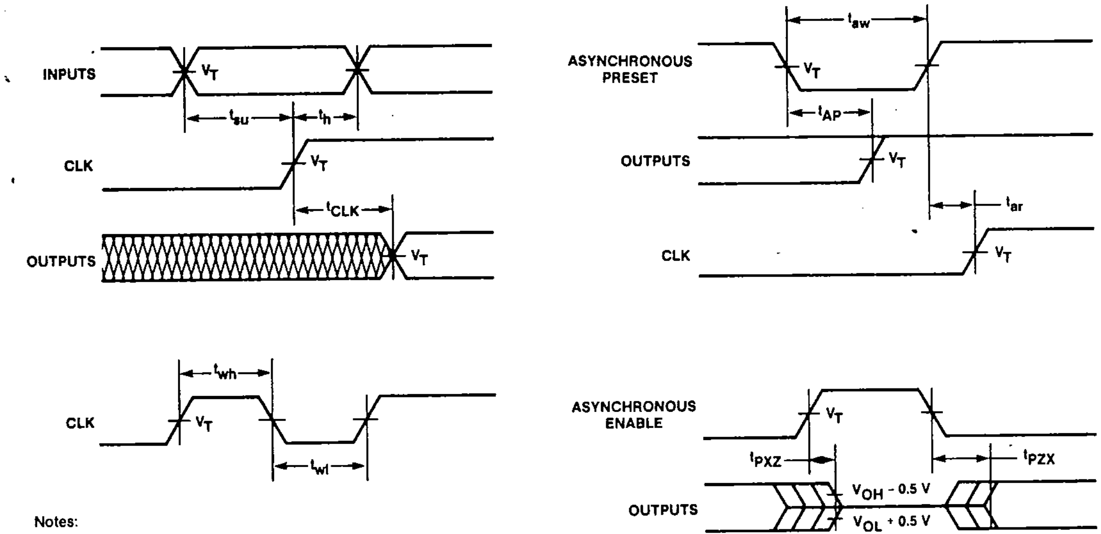
SYMBOL	PARAMETER	TEST CONDITION		MIN	TYP	MAX	UNIT
V_{IL}[2]	Low-level input voltage					0.8	V
V_{IH}[2]	High-level input voltage			2.0			V
V_{IC}	Input clamp voltage	V_{CC} = MIN	I_I = -18 mA		-0.8	-1.5	V
I_{IL}	Low-level input current	V_{CC} = MAX	V_I = 0.4 V		-0.02	-0.25	mA
I_{IH}	High-level input current	V_{CC} = MAX	V_I = 2.4 V			25	μA
I_I	Maximum input current	V_{CC} = MAX	V_I = 5.5 V			200	μA
V_{OL}	Low-level output voltage	V_{CC} = MIN	I_{OL} = 8 mA		0.3	0.45	V
V_{OH}	High-level output voltage	V_{CC} = MIN	I_{OH} = -3.2 mA	2.4	3.2		V
I_{OZL}	Off-state output current	V_{CC} = MAX	V_O = 0.4 V			-100	μA
I_{OZH}			V_O = 2.4 V			100	μA
I_{OS}[3]	Output short-circuit current	V_{CC} = 5 V	V_O = 0 V	-20	-50	-90	mA
I_{CC}	Supply current	V_{CC} = MAX			170	210	mA
C_{IN}	Input capacitance	V_{IN} = 2.0 V at f = 1 MHz.			8		pF
C_{OUT}	Output capacitance	V_{OUT} = 2.0 V at f = 1 MHz			11		pF
C_{CLK}	Clock capacitance	V_{CLK} = 2.0 V at f = 1 MHZ			13		pF

Notes:
1. The PMS14R21/A is designed to operate over the full military operating conditions. For availability and specifications, contact Monolithic Memories
2. These are absolute values with respect to the ground pin on the device and include all overshoots due to system and/or tester noise. Do not attempt to test these values without suitable equipment.
3. No more than one output should be shorted at a time and duration of the short-circuit should not exceed one second

Sequencer Switching Characteristics Over Operating Conditions

SYMBOL	PARAMETER	TEST CONDITIONS	COMMERCIAL						UNIT
			STD			A			
			MIN	TYP	MAX	MIN	TYP	MAX	
t_{CLK}	CLK to output	$R_1 = 560\ \Omega$ $R_2 = 1.1\ K\Omega$		10	20		9	12	ns
t_{PZX}	$\overline{E}$ to output enable			10	20		10	15	ns
t_{PXZ}	$\overline{E}$ to output disable			10	20		10	15	ns
t_{AP}	Asynchronous preset to output			14	25		14	25	ns
f_{MAX}	Maximum frequency, external $(1/(t_{su} + t_{CLK}))$		18	25		25	30		MHz
	Maximum frequency, internal		25	30		30	35		

Sequencer Switching Waveforms

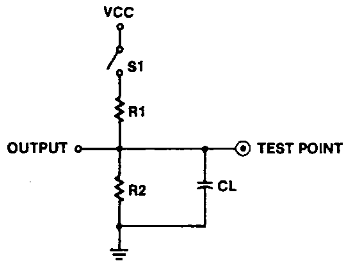


Notes:
1. $V_T = 1.5\ V$
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

Key to Timing Diagram

WAVEFORM	INPUTS	OUTPUTS
▨	DON'T CARE; CHANGE PERMITTED	CHANGING; STATE UNKNOWN
⟫⟪	NOT APPLICABLE	CENTER LINE IS HIGH IMPEDANCE STATE
—	MUST BE STEADY	WILL BE STEADY

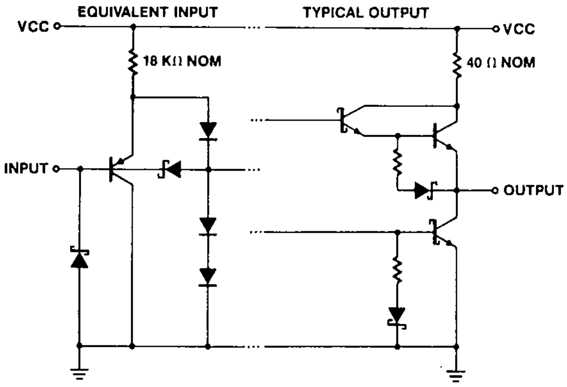
Switching Test Load



Notes:

1. Propagation delays are tested with switch S_1 closed. C_L = 30 pF and measured at 1.5 V output level.
2. t_{PZX} is measured at the 1.5 V output level with C_L = 30 pF. S_1 is open for high impedance to "1" test, and closed for high impedance to "0" test.
3. t_{PXZ} is tested with C_L = 5 pF. S1 is open for "1" to high impedance test, measured at V_{OH} -0.5 V output level: S_1 is closed for "0" to high impedance test measured at V_{OL} +0.5 V output level.

Schematic of Inputs and Outputs

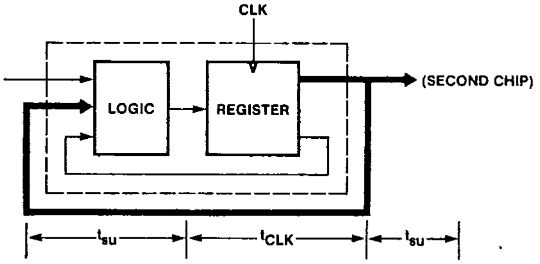


fMAX Parameters

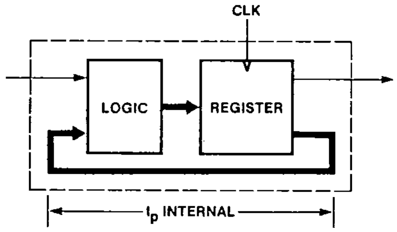
The parameter fMAX is the maximum clock rate at which the device is guaranteed to operate. Because flexibility inherent in programmable logic devices offers a choice of clocked flip-flop designs, fMAX is specified for two types of synchronous designs.

The first type of design is a state machine with feedback signals sent off-chip. This external feedback could go back to the device inputs, or to a second device in a multi-chip state machine. The slowest path defining the period is the sum of the clock-to-output time and the input setup time for the external signals. The reciprocal, fMAX, is the maximum frequency with external feedback or in conjunction with an equivalent speed device. This fMAX is designated "fMAX external."

The second type of design is a single-chip state machine with internal feedback only. In this case, flip-flop inputs are defined by the device inputs and flip-flop outputs. Under these conditions, the period is limited by the internal delay from the flip-flop outputs through the internal feedback and logic to the flip-flop inputs. This fMAX is designated "fMAX internal."



f_{MAX} EXTERNAL; 1/(t_{CLK} +t_{su})



f_{MAX} INTERNAL; 1/(t_p INTERNAL)

Diagnostics Function Table

INPUTS				OUTPUTS			OPERATION
MODE	SDI	CLK	DCLK	Q_{20}-Q_0	S_{20}-S_0	SDO	
L	X	↑	*	Qn ← PROM	HOLD	S_{20}	Load output register from PROM array
L	X	*	↑	HOLD	$S_n \leftarrow S_{n-1}$ $S_0 \leftarrow$ SDI	S_{20}	Shift shadow register data
L	X	↑	↑	Qn ←PROM	$S_n \leftarrow S_{n-1}$ $S_0 \leftarrow$ SDI	S_{20}	Load output register from PROM array while shifting shadow register data
H	X	↑	*	$Q_n \leftarrow S_n$	HOLD	SDI	Load output register from shadow register
H	L	*	↑	HOLD	$S_n \leftarrow Q_n$	SDI	Load shadow register from output register
H	L	↑	↑	$Q_n \leftarrow S_n$	$S_n \leftarrow Q_n$	SDI	Swap output and shadow registers
H	H	*	↑	HOLD	HOLD	SDI	No operation†

* Clock must be steady or falling.

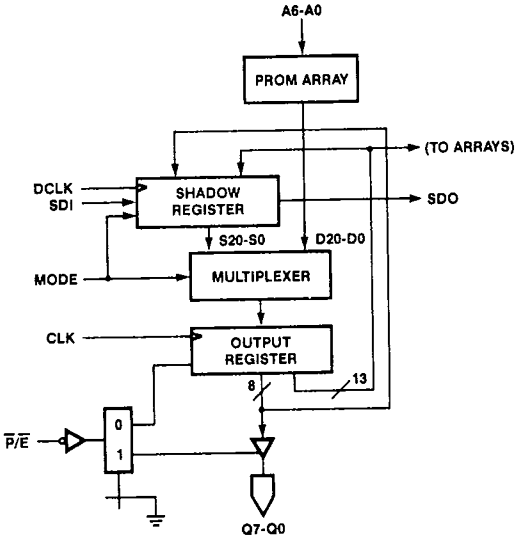
† Reserved operation for 54/74S818 8-Bit Diagnostic Register.

Diagnostics-On-Chip

Diagnostics-On-Chip, or DOC™, is a test method that allows complete controllability and observability of a device, which results in complete testability. The DOC circuitry includes the serial diagnostic register and parallel connections to and from the output register.

The diagnostic (or shadow) register is a 21-bit serial/parallel register. It has its own clock, called DCLK. It can shift serially, or parallel transfer its contents to or from the output register. The MODE input selects serial or parallel mode (low or high, respectively). The serial shift input is SDI and the output is SDO. SDI doubles as a control input if not in serial mode (MODE is high), and is then transferred directly to SDO. See the function table above for details.

Diagnostics Block Diagram



Diagnostic Test Sequence

A typical diagnostic test sequence would proceed as follows. First, a test vector would be serially shifted into the diagnostic register. This is done as shown in the waveforms under Shifting Diagnostic Register, with MODE set low and clocking DCLK. During shifting, the sequencer operates completely independently.

The test vector is then transferred to the output register by switching MODE to high and clocking the output register. This is shown in the waveforms under Loading Output Register from Diagnostic Register. This function makes the output register controllable. MODE goes back low and the system is cycled with the test vector as during normal operation.

Test results in the output register are observed by again setting MODE high and clocking the diagnostic clock. This loads the diagnostic register from the output register. This process is shown in the waveforms under Loading Diagnostic Register from Output Register. The results can then be shifted out by switching MODE low and clocking the diagnostic clock. These functions provide complete observability of the test results in the output register. Note that while the test results are being shifted out, a new test vector may be shifted into the diagnostic register.

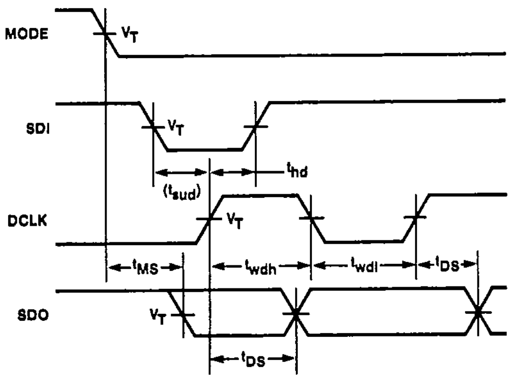
When a test vector is loaded into the output register, the current state of the output register may be required for test analysis or for system restart at the same state. In either case, the output register contents must be transferred to the diagnostic register for saving. Diagnostics-On-Chip allows the registers to swap their contents for this purpose. The CLK and DCLK signals can be separated by no more than t_{skew}, as shown in the waveforms under Swap Registers.

A readback function is reserved for the 8-bit Diagnostic Register device. The required signals create a No Operation condition in the PMS14R21 device. This is shown in the waveforms under No Operation.

Using Diagnostics-On-Chip in a System

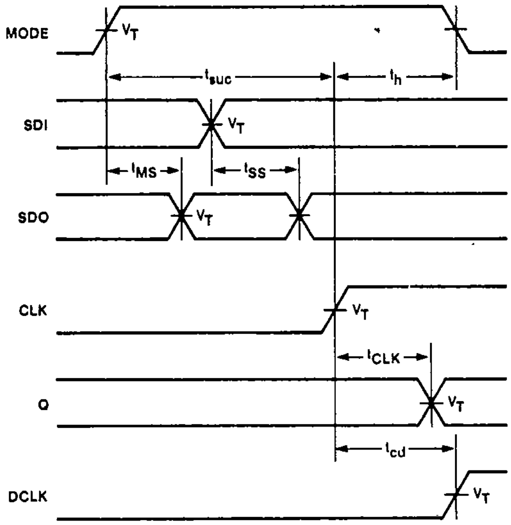
In a system environment, Diagnostics-On-Chip allows complete controllability and observability of registers buried within devices or within blocks of logic on the printed circuit board. Access is permitted through a scan path, requiring only one input to access a number of internal nodes. The scan path within each DOC chip can be connected to any other DOC chip by connecting SDO to SDI in series, and applying DCLK and MODE signals in parallel to all DOC chips. This allows a single scan path to be routed throughout a board, increasing the testability of the board and system.

Diagnostics Switching Waveforms

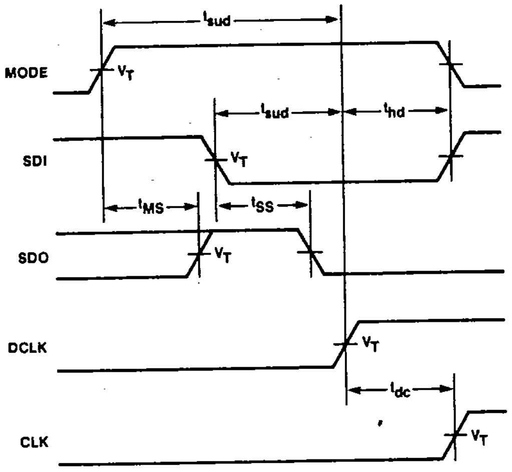


Note: Sequencer operates independently when MODE is low

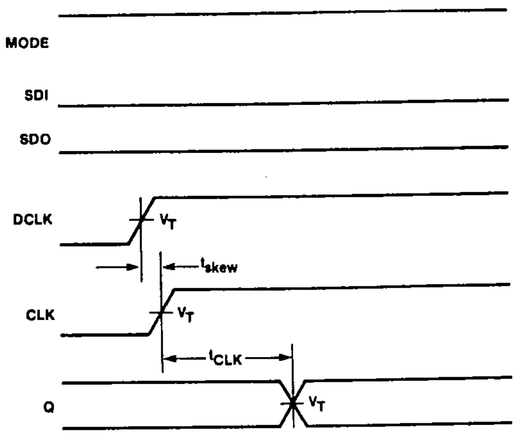
Shifting Diagnostic Register



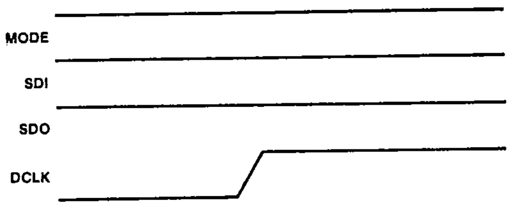
Load Output Register from Diagnostic Register



Load Diagnostic Register from Output Register



Swap Registers



No Operation

Notes:

1. $V_T = 1.5$ V
2. Input pulse amplitude 0 V to 3.0 V
3. Input rise and fall times 2-5 ns typical

T-46-13-35

Diagnostics Operating Conditions

SYMBOL	PARAMETER		COMMERCIAL						UNIT
			STD			A			
			MIN	TYP	MAX	MIN	TYP	MAX	
t_{wd}	Width of DCLK	Low	30	15		30	15		ns
		High	30	15		30	15		
t_{suc}	Setup time from MODE to CLK		30	15		30	15		ns
t_{sud}	Setup time from MODE or SDI to DCLK		30	15		30	15		ns
t_h	Hold time for MODE to CLK		15	10		15	10		ns
t_{hd}	Hold time for MODE or SDI to DCLK		15	10		15	10		ns
t_{dc}	DCLK to CLK separation if not simultaneous	MODE = H	50	30		50	30		ns
t_{cd}	CLK to DCLK separation if not simultaneous	MODE = H	40	20		40	20		ns
t_{skew}	CLK to DCLK (DCLK to CLK) separation in swap mode	MODE = H		10	5		10	5	ns

Diagnostics Switching Characteristics Over Operating Conditions

SYMBOL	PARAMETER		TEST CONDITIONS	COMMERCIAL						UNIT
				STD			A			
				MIN	TYP	MAX	MIN	TYP	MAX	
t_{DS}	DCLK to SDO	MODE = L	R_1 = 560 Ω R_2 = 1.1 KΩ		20	35		20	35	ns
t_{SS}	SDI to SDO	MODE = H			10	25		10	25	ns
t_{MS}	MODE to SDO				10	25		10	25	ns
f_{MAXd}	Maximum diagnostic shift frequency $(1/(t_{wdh} + t_{wdl}))$			16.6	33		16.6	33		MHz

Design Example

As an introduction to designing with the PMS14R21, an example application will be used to demonstrate the state machine design file required for PALASM 2 software. PALASM 2 software can verify the design and then create a file for programming the device.

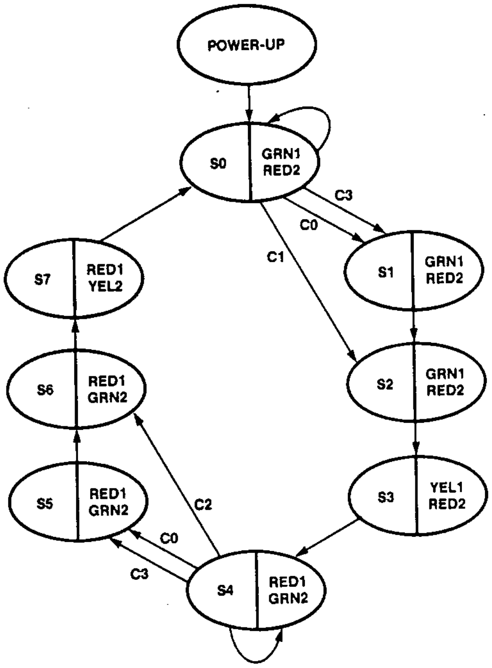
The example is a traffic signal controller for an intersection of two one-way streets. The inputs are active-high sensors for each direction and a clock, and the outputs drive the six active-high lights for the two signals.

The state diagram for this application is shown below. Each bubble represents a single state, with the state name on the left and the state outputs on the right. Each arrow represents a potential transition between states, with the conditions required for that transition (if any) next to the arrow.

Translation of this design to a PALASM 2 software file is straightforward. The following sections will be discussed in detail:

- Declaration and pin definitions (equivalent to other PALASM 2 software files)
- Other definitions required for state machines and the PMS14R21
- Transitions and their conditions
- State outputs
- Definitions of conditions
- Optional simulation section.

The complete design file is shown on the next page.



Traffic Controller State Diagram

DECLARATION Section

This section documents the design. The CHIP statement consists of the design name, the part number (PMS14R21), and the pin list for pins 1-24 of the device.

STATE Section

The STATE section begins with the keyword STATE and has three parts: a block of general definitions in the design, the transition equations, and the state outputs.

General Definitions

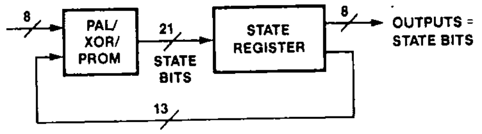
The first general definition specifies one of two types of state machine. Either MOORE_MACHINE or MEALY_MACHINE may be entered, according to the design. The difference between the two types is shown below. In this case, the outputs are a function of only the state, so MOORE_MACHINE is specified.

The next general definition specifies the programmable function pin. Preset or Enable are selected by specifying MASTER_RESET or OUTPUT_ENABLE. MASTER_RESET is used, although neither is functionally required in this design.

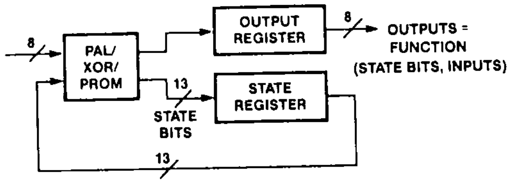
The next general definitions describe defaults for outputs not otherwise defined for a transition. First is an optional OUTPUT_HOLD statement that lists all outputs that are to hold value as a default. It is not used in this design.

Second is the optional DEFAULT_OUTPUT statement that lists default output values. High, Low, and Don't Care are selected with <pin name>, /<pin name>, and %<pin name>, respectively. In the example, the signal lights default to off, or Low.

Next, the optional DEFAULT_BRANCH statement specifies the next state when not otherwise specified. The state may be a state name, or can be HOLD_STATE or NEXT_STATE to hold or go to the next state in the design file. It is not needed in this design.



Moore Machine



Mealy Machine

T-46-13-35

```
TITLE       Traffic Controller               ;Description Section
PATTERN     State Machine
REVISION    1
AUTHOR      Jane Engineer
COMPANY     Monolithic Memories
DATE        January 30, 1987

CHIP        S_MACHINE PMS14R21
            CLK DCLK SEN1 SEN2 I2 I3 I4 I5 I6 I7 SDI GND
            PRESET SDO RED1 YEL1 GRN1 RED2 YEL2 GRN2 O1 O0 MODE VCC

;This example demonstrates the conversion of a state diagram to a
;PALASM 2 input file, or PDS file.  The example is a traffic signal
;controller for an intersection of two one-way streets.  The inputs
;are active high sensors for each direction and a clock, and the
;outputs drive the six active-high lights for the two signals.

STATE                         ;Specifies state machine format

;General Definitions
   MOORE_MACHINE              ;Outputs function of state only
   MASTER_RESET               ;Programmable pin selects preset

   DEFAULT_OUTPUT /RED1 /YEL1 /GRN1 /RED2 /YEL2 /GRN2
                              ;Outputs default to Low, or off
;Transition Equations
   POWER_UP:= VCC -> S0 ;Goto State0 after power-up (or preset)
   S0:= C3 -> S1        ;From State0: if C3 true, goto State1
      + C0 -> S1        ;             or if C0 true, goto State1
      + C1 -> S2        ;             or if C1 true, goto State2
         +-> S0         ;            Otherwise goto State0 (hold)
   S1:=VCC -> S2        ;From State1: goto State2 unconditionally
   S2:=VCC -> S3        ;From State2: goto State3 unconditionally
   S3:=VCC -> S4        ;From State3: goto State4 unconditionally
   S4:= C3 -> S5        ;From State4: if C3 true, goto State5
      + C0 -> S5        ;             or if C0 true, goto State5
      + C2 -> S6        ;             or if C2 true, goto State6
         +-> S4         ;            Otherwise goto State4 (hold)
   S5:=VCC -> S6        ;From State5: goto State6 unconditionally
   S6:=VCC -> S7        ;From State6: goto State7 unconditionally
   S7:=VCC -> S0        ;From State7: goto State0 unconditionally

;Output Equations
   S0.OUTF  := GRN1*RED2    ;In State0 outputs GRN1 and RED2 are High
   S1.OUTF  := GRN1*RED2    ;In State1 outputs GRN1 and RED2 are High
   S2.OUTF  := GRN1*RED2    ;In State2 outputs GRN1 and RED2 are High
   S3.OUTF  := YEL1*RED2    ;In State3 outputs YEL1 and RED2 are High
   S4.OUTF  := RED1*GRN2    ;In State4 outputs RED1 and GRN2 are High
   S5.OUTF  := RED1*GRN2    ;In State5 outputs RED1 and GRN2 are High
   S6.OUTF  := RED1*GRN2    ;In State6 outputs RED1 and GRN2 are High
   S7.OUTF  := RED1*YEL2    ;In State7 outputs RED1 and YEL2 are High

CONDITIONS
   C0 = /SEN1*/SEN2      ;C0 is true when SEN1 and SEN2 Low
   C1 = /SEN1* SEN2      ;C1 is true when SEN1 Low and SEN2 High
   C2 =  SEN1*/SEN2      ;C2 is true when SEN1 High and SEN2 Low
   C3 =  SEN1* SEN2      ;C3 is true when SEN1 and SEN2 High

;(SIMULATION is optional - not shown here)
```

Transition Equations

The next section of the file is the transition equations. These are directly equivalent to the arrows in the state diagram, using the state names and the conditions defined later in the file. For example, the second state equation defines the four transition arrows from state S0:

```
S0: = C3 -> S1
    + C0 -> S1
    + C1 -> S2
       +-> S0
```

which means: when in state S0, if condition C3 is true go to state S1, or if condition C0 is true go to state S1, or if condition C1 is true go to state S2, otherwise go to state S0 (hold). The last line has no condition, so that if none of the preceding conditions is true, this transition will occur by default. Note that conditions and states cannot be grouped by parentheses.

The first transition equation in the example specifies the transition from the power-up state. The condition is VCC because the transition always occurs on the clock signal. If a Mealy machine were being designed, the power-up state would also require definition of the outputs for transition to the next state. Note that the power-up state is also entered when the preset pin is asserted.

Output Equations

The next section of the design file lists the output equations, which specify the output values for each state. The first output equation, S0.OUTF := GRN1*RED2, means that in state S0, GRN1 and RED2 will be High (lights GRN1 and RED2 will be on) while the other outputs assume their default values (Low, or off). Note that for a Mealy machine, equations specify the outputs on transitions to next states, and can include conditions.

CONDITIONS Section

The next section of the design file defines the conditions used in the STATE section. In this case, C0 = /SEN1*/SEN2 means that condition C0 is true when the inputs SEN1 and SEN2 are Low (sensors are not activated). Conditions equal a sum of products of the device inputs.

SIMULATION Section

The SIMULATION section provides a means of verifying the design. State machine simulation is similar to Boolean equation simulation, with a few added features. This section is optional.

Other Design Considerations

PALASM 2 software automatically assigns states to PROM addresses, and automatically arranges the product terms and XOR feedback signals. The automatic assignment optimizes the use of both the PAL and PROM arrays for the most efficient use of resources.

Transition equations are limited to four-way branches, but up to sixteen-way branching can be performed in two clock cycles. This requires the addition of a temporary state after the first four-way branch.

Programmers/Development Systems

(refer to Programmer Reference Guide, page 3-81)