LSI/CSI

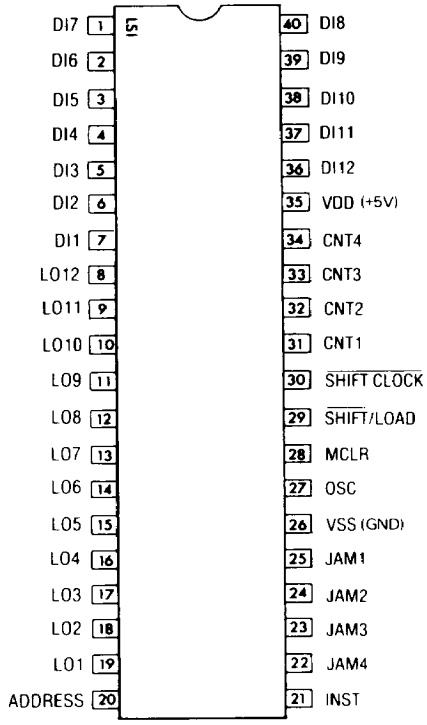
LS7270

Revised July 1989

PROGRAMMABLE INTEGRATED CONTROLLER/SEQUENCER

FEATURES:

- Hardware oriented simple instruction set
- 4 on-chip 12 bit programmable down-counters
- 4 priority interrupt (JAM) inputs
- 12 discrete inputs
- 12 latched outputs
- 12 discrete memory bit registers
- Anti-bounce circuits on DI, CNT and JAM inputs for direct interface with mechanical switches, keyboards, etc.
- Simple serial interface to external program memory (PROM or ROM)
- External program memory up to 2048 instructions
- On-chip clock generator
- Inputs TTL, NMOS and CMOS compatible
- Outputs TTL, NMOS and CMOS compatible
- Single power supply operation. +4.75 VDC to +12 VDC
- 40 pin plastic DIP



TOP VIEW

GENERAL DESCRIPTION:

The LS7270 is a monolithic, ion implanted MOS logic controller/sequencer, designed to satisfy a wide variety of timing, sequencing and controlling functions in small to medium sized systems requiring low cost electronic control hardware. A "basic controller/sequencer" type machine can be thought of as a simple "black box" with inputs, outputs and various chip support functions such as power supply, oscillator, etc. As in any sequential logic machine, the present state of the machine is logically combined with the present state of the inputs to produce a new machine state with its corresponding outputs. Hence, as inputs change, the machine reacts generating new outputs depending on its previous state and the new inputs.

In a traditional hardwired logic machine, the sequence of the machine for all possible combinations of inputs is determined by the design of various random logic units all permanently wired so that the results is not very flexible or amenable to change. The solution to this problem as implemented in the LS7270, is to utilize some form of computer or microprocessor type architecture that executes a series of instructions (the program steps) held in a memory (external to the chip) to perform the intended logical combinations of the inputs with the current machine state. In contrast to computers or microprocessors, however, the internal architecture of the LS7270 is geared to individual bit processing, Boolean processing, turn-on and turn-off functions, counting and timing operations as opposed to numeric computations. Broadly speaking, (see Fig. 1

and Description of General Architecture) the LS7270 has discrete inputs (DI) that can be addressed and operated upon at the individual bit level, internal flags (T) and storage cells (M) also addressed and operated upon at the bit level, addressable internal counters that can be clocked by external sources and a group of individually addressable output registers (LO). Boolean processing is done by selecting and multiplexing various inputs into the Logic Unit (LU) along with the working Accumulator Flag (AF), thus performing sequentially the required Boolean expression and then outputting the result to the appropriate output. This is done under control of a sequence of instructions (a table of logical "0"s and "1"s) fetched from the external program memory.

In operation, the LS7270 serially shifts out a memory address when the chip is in the "shift cycle." An external shift register has to be provided in which the address can be shifted and set up for addressing the memory (see fig. 2). During the shift cycle, clocks are generated at the shift clock output which are in synchronism with the address bit changes at the address output. At the end of the shift cycle, instruction from the memory is loaded into the interface shift register. A new shift cycle begins, and the instruction from the Interface Shift Register is now shifted into the LS7270; simultaneously a new instruction address is shifted out into the interface shift register. The LS7270 continuously alternates between the "shift" and the "load" cycles executing the instruction in between whenever a complete instruction has been fetched. The address is automatically incremented by 1 in every shift cycle so that instructions from higher locations of a memory can be fetched sequentially. This general rule of address sequencing is broken only when an instruction involving an address jump is executed. When an instruction is executed one of the following events may take place (see the instruction set for details):

1. Load 1 of 4 counters with a 12 bit number specified in the instruction field,
2. Decrement one of the counters,
3. Set or reset one of the internal registers,
4. Load the AF with the true or complement value of one of the internal registers or discrete inputs,
5. Combine AF with the true or complement value of any of the internal registers or discrete inputs in Boolean operation,
6. Store the true or complement value of the AF in any of the internal registers or output latches, and finally,
7. Branch out from normal addressing sequence and jump to an address specified in the instruction field.

GENERAL ARCHITECTURE OF LS7270 (See Fig. 1)

Program Counter (PC and PCB). The PC is a 12 bit register that holds the address for the next instruction. The external memory address is serially shifted out from the PC to the memory. The PCB is a back-up register for the PC used internally by the LS7270 chip.

Instruction Register (IR). The IR is a 16 bit register that holds the instruction currently being executed. Instructions from the external memory are serially shifted into the IR.

STACK 0-2. The LS7270 has a 3 level Last In-First Out (LIFO) stack. The next instruction address from the PCB is pushed onto the stack when a Jump to Subroutine (JS) instruction or a JAM 1 interrupt is executed. The address is returned to the PCB when a Return from Subroutine instruction is executed.

JAM Request Registers (JRR 1-4). The JRRs are 4 one-bit registers that are set by the corresponding JAM inputs. The outputs of the JRRs cause a Jump within the program sequence. Each JRR has a dedicated address assigned to it as its jump destination.

Counters (CNTR 1-4). The LS7270 has four 12 bit programmable down-counters. The counters can be clocked by either external count inputs or the internal clock under program control. Outputs from each counter are decoded for zero and testable under program control.

Logic Unit and the Accumulator Flag (LU and AF). The LU performs all the Boolean algebraic operations contained in the LS7270 instruction set and stores the result in the AF.

Temporary Flags (T1-3). The T's are three one-bit registers each of which can be accessed by the TEMP field of the LOGICAL CONTROL group instructions.

Memory Flags (M1-12). The M's are 12 one-bit registers. The output of the LU can be stored in any of these registers by program control. The outputs of the M's in turn can be logically combined with other inputs to the LU.

Latched Output Registers (LO 1-12). The LO's are 12 one-bit registers each of which can be loaded by the LU data. The LO outputs are available on the output pins.

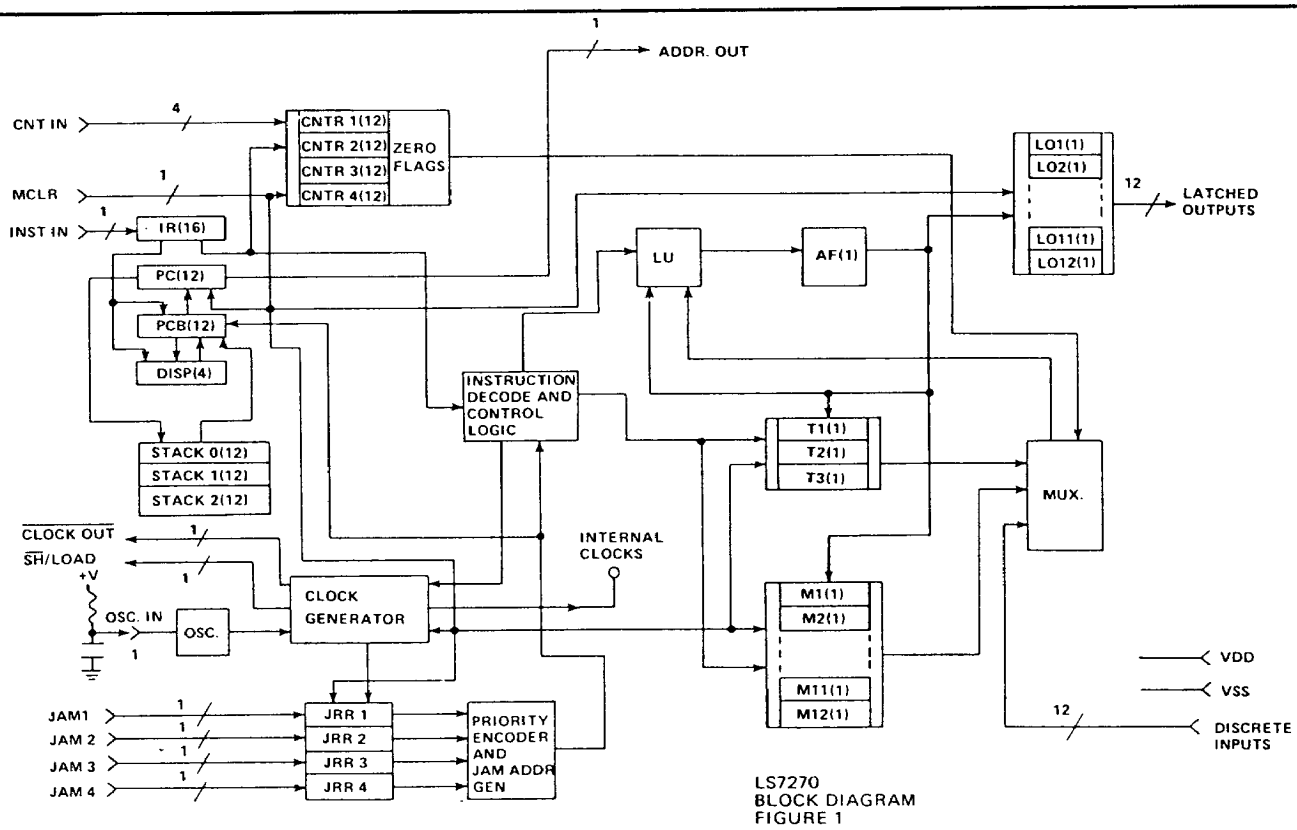
Multiplexer (MUX). The MUX performs all the steering operations of the T's, M's, AF and the Discrete inputs to LU.

DESCRIPTION OF OPERATION: (see figs. 4, 5 and 6)

The LS7270 address consists of 12 bits, and the instruction of 16 bits. In normal operation an instruction cycle consists of 2 shift/load cycles involving 26 shift clocks.

After a reset, the Program Counter (PC) is cleared to address the first memory location (address 0). When the reset is removed, the 12 bit memory address of the location is serially sent out During this time, the shift/load output remains low to hold the memory interface shift register in the shift mode. At the end of the shift cycle consisting of 12 shift clocks, the shift/load output goes high placing the interface shift register in the load mode and the first instruction byte (lower byte) from the external memory matrix is loaded into the interface shift register on the thirteenth clock pulse. During every shift/load cycle the PC is incremented by 1 to address the next higher memory location. Then a shift cycle begins again. During this shift cycle while the address for the second byte (higher byte) of the instruction is shifted out to the interface shift register, the lower byte of the instruction, already in the interface shift register, is shifted into the Instruction Register (IR). Note that the internal shift clock for the IR occurs coincident with the first eight shift clocks only since one instruction byte consists of eight bits; there are no shift clocks for the IR corresponding to the remaining five clocks of the total shift/load cycle. At the end of the second shift cycle, the second instruction byte (upper byte) is loaded into the interface register. During the third shift cycle, the upper byte is shifted into the IR and, at the end of the cycle, the instruction is executed. It is important to note that if a smaller external memory is used which does not require all 12 bit addressing capability (4096 bytes or 2048 instructions), the interface register can be implemented with fewer bits and the higher order address bits will simply "fall off' the interface register during the shifting cycle. Thus if only 256 bytes of memory is required, the interface register could be implemented with one octal shift register.

The four programmable down-counters can be driven by either external clocks applied at the counter inputs or decremented under program control. The counters are programmable by instruction control only. During the execution of an instruction for loading or decrementing a counter, the external count input is blocked for a period of 2 shift clocks. The external count input is synchronized with the internal clock so that counter integrity is not lost during the blocking period. The blocking period for any count input lasts between trailing edges of 8th and 10th shift clocks of a high byte fetch cycle containing a load or decrement instruction for the corresponding counter.



LS7270
BLOCK DIAGRAM
FIGURE 1

A counter zero condition can be tested by program control to create decision branching within the program sequence.

The 12 discrete inputs (DI) can be combined with the accumulator flag (AF) to perform Boolean operations and the result steered to 12 single bit memory flags (M) or 12 output latches (LO), or 3 temporary storage flags (T).

I/O DESCRIPTION:

MCLR INPUT:

A high on this input initializes all the registers and holds the clock off. It clears the PC, the JAM request register (JRR), the output latches (LO), the temporary storage flags (T), and the memory flags (M). It presets the down-counters to all 1's. The MCLR input has an internal pull-down (to logic "0") resistor.

OSC INPUT:

An R/C network on this input sets up the frequency of the internal oscillator. The basic oscillator frequency as indicated by the ramp frequency developed on the OSC input is divided down by 4 for generating the internal system clock. The basic oscillator frequency is approximately given by the relation, $f = 1/RC$.

INST INPUT:

Instructions from external memory are serially shifted into the IR on this input with the LSB being input first. The INST input has an internal pull-down (to logic "0") resistor.

JAM INPUTS:

The JAM inputs are four vectored priority interrupts with JAM 1 having the highest priority and JAM 4 the lowest. A low to high transition of a JAM input forces a specific address into the PC at the end of the currently executing instruction. The four specific addresses allocated for JAM 1 through JAM 4 are 2, 4, 6, and 8, respectively. JAM 1 is different from the other JAMS in that it saves the address of the next instruction on the push down stack so that by including a RETURN instruction at the end of the JAM service routine, the original program sequence can be resumed. JAM request registers are set by a positive transition of the JAM inputs and are reset after the JAM has been serviced. If the JAM input remains high, it will not be serviced a second time. But a high level on a JAM input will inhibit all the lower priority JAM inputs. If a lower priority JAM is activated while a higher priority JAM request is being serviced (with the higher priority JAM input already returned low), the lower priority JAM request will be serviced at the end of the current instruction cycle. All JAM inputs have internal anti-bounce-circuits for direct interface with switches, relays, etc.

COUNTER INPUTS (CNT):

Each of the four counter inputs clocks one of the four 12 bit down counters. The counter advances on the positive transition of the counter input. All counter inputs have internal anti-bounce circuits.

DISCRETE INPUTS (DI):

Each of the 12 discrete inputs can be read by the program as part of Boolean logical expression evaluation. All discrete inputs have internal anti-bounce circuits.

SHIFT CLOCK OUTPUT:

This output is used for clocking the memory interface shift registers. The negative edge of the clock output should be used to clock the shift register.

SHIFT/LOAD OUTPUT:

This output is used for shift/load control of the memory interface shift register. Each shift/load cycle encompasses 13 shift clocks. When the shift/load output is "low," 12 output clocks serially shift out a 12 bit memory address, while the instruction byte from the preceding address is simultaneously shifted into the IR. At the end of the 12 shift clocks, the shift/load output goes "high" for one clock period. During this period, the next instruction byte is loaded into the shift register and a new shift cycle begins.

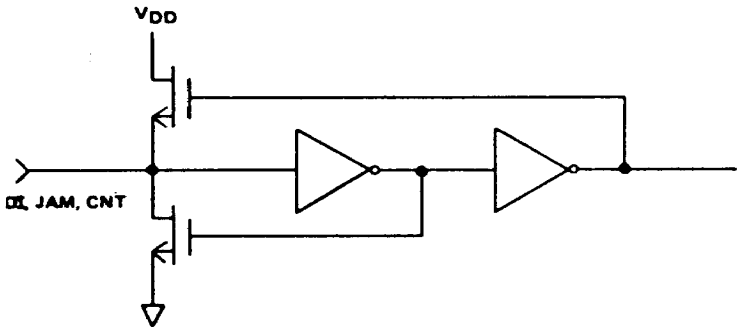


FIGURE 2. Input Anti-Bounce Circuit

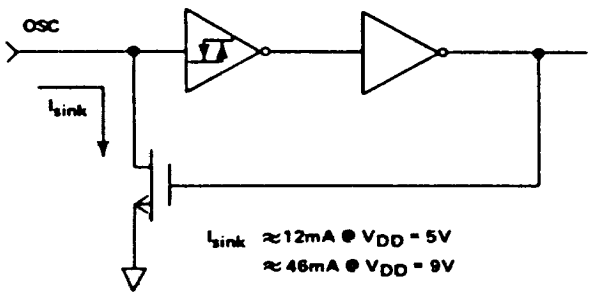


FIGURE 3. Internal Oscillator Circuit

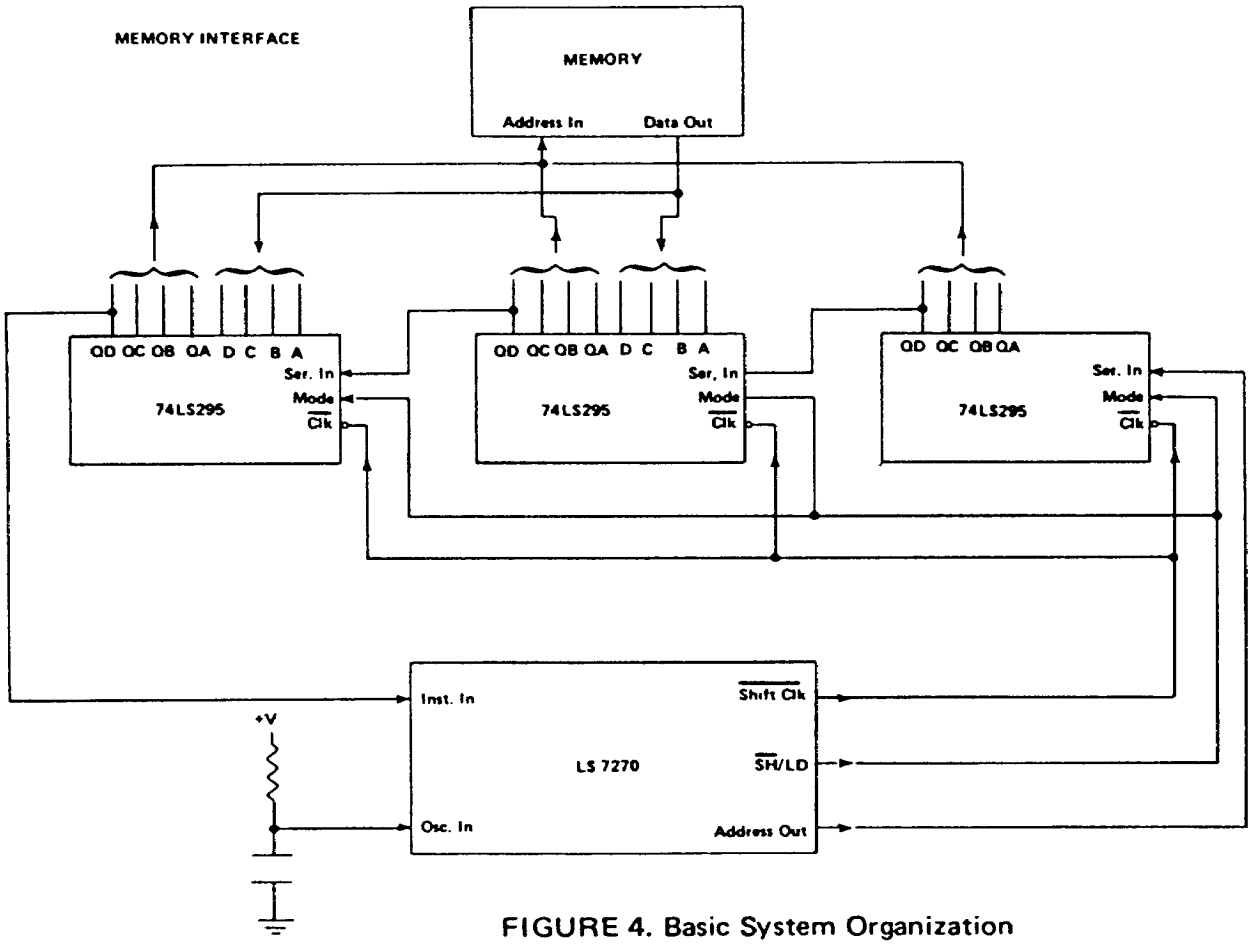
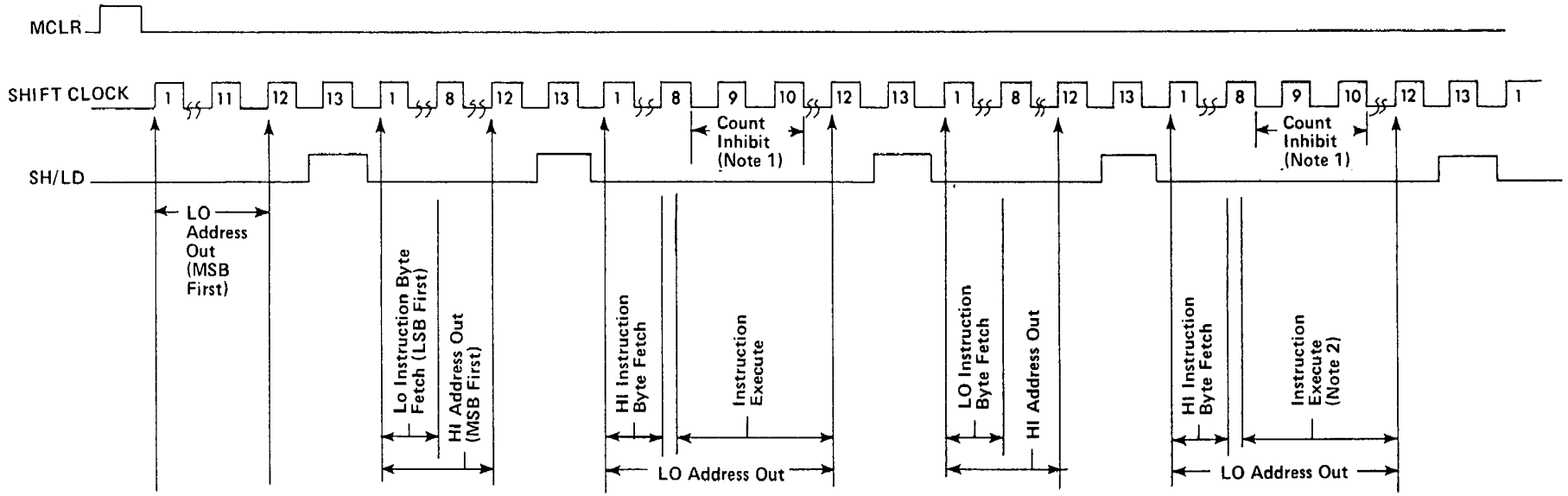


FIGURE 4. Basic System Organization

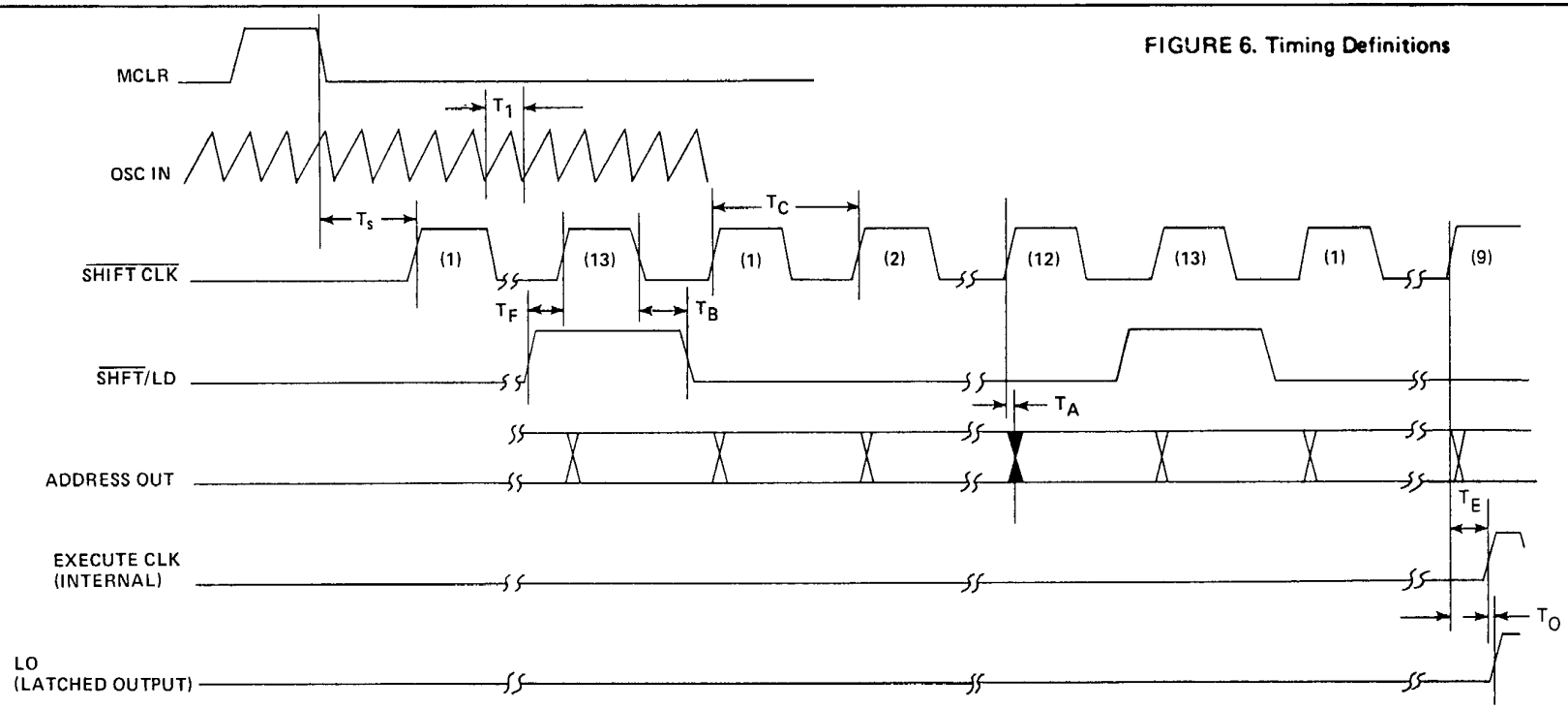


Note 1: External Count to a counter is inhibited during the execution of a load or decrement instruciton operating on that counter

Note 2: Following the execution of A J, JS, RET, SKIP and JAM, three address cycles will be generated instead of regular two, before the next instruction is executed.

FIGURE 5. Instruction Cycles

FIGURE 6. Timing Definitions



CONTROLLER SEQUENCER

ADDRESS OUTPUT:
The twelve bit memory addresses are serially shifted out on this output line with the MSB being output first. Address bits change with the rising edge of the shift clock.

LATCHED OUTPUTS (LO):
The twelve latched outputs are driven from 12 flip-flops within the chip. The state of each of these flip-flops is directly under program control so that they can be manipulated according to the required application.

INSTRUCTION SET:
There are two classes of instructions in the LS7270 Controller/Sequencer. The first is the internal control group and the second is the logical operation group. Each is 16 bit in length.

INTERNAL CONTROL GROUP:

0 (1-bit)	OPER (3-bits)	CONSTANT OR ADDRESS (N) (12-bits)

15 0

Operation Code	MNEMONIC	OPERATION
000	NOP	No operation
001	LC1 DC1*	CNTR1 ⧫ N (Load counter 1 with constant N)
010	LC2 DC2*	CNTR2 ⧫ N (Load counter 2 with N)
011	LC3 DC3*	CNTR ⧫ N (Load counter 3 with N)
100	LC4 DC4*	CNTR4 ⧫ N (Load counter 4 with N)
101	J/RT	If N ≠ FFF (HEX), PC ⧫ N (Jump to address N); If N = FFF, PC ⧫ $Stack_0$, $Stack_{0,1}$ ⧫ $Stack_{1,2}$ (Return from subroutine)
110	JS	$Stack_{1,2}$ ⧫ $Stack_{0,1}$, $Stack_0$ ⧫ PC, PC ⧫ N (Jump to subroutine address N and save the return instruction address on the stack)

*If N=0, a load counter instruction is decoded as a decrement counter instruction thereby creating the DC1, DC2, DC3, DC4 instructions.

LOGICAL CONTROL GROUP:

1 (1)	OPER (3)	T/C (1)	ADDRESS,N (5)	TEMP (2)	SKIP/DISP. (4)

15 0

Input Mode

OPER FIELD	MNEMONIC	OPERATION
000	LD/LDC	AF ⧫ (N)/AF ⧫ ($\overline{N}$) Load AF with the true/complement of the value of register addressed by N.
001	AND/ANDC	AF ⧫ AF∧(N)/AF ⧫ AF∧ ($\overline{N}$) AND AF with true/complement value of register addressed by N.
010	OR/ORC	AF ⧫ AF V (N)/AF ⧫ AF V ($\overline{N}$) OR AF with true/complement value of register addressed by N.
011	XOR/XORC	AF ⧫ AF⊕(N)/AF ⧫ AF⊕($\overline{N}$) XOR AF with true/complement value of register address by N.

Output Mode

OPER FIELD	MNEMONIC	OPERATION
110	STR/STRC	(N) ⧫ AF/(N) ⧫ $\overline{AF}$ store true/complement value of AF into register addressed by N.
111	SET/CLR	If T/C=0 then (N) ⧫ 1; If T/C=1 then (N) ⧫ 0. Set or clear the register addressed by N.

T/C Field, True/Complement
0 – Select the true value of the addressed register.
1 – Select the complimented value of the addressed register.

ADDRESS, N FIELD – Input Mode Addressing Assignments
00000 to 01011 –Specify DI_1 to DI_{12}, respectively.
01100 to 01111 –Specify CNTR1 equal to zero flag through CNTR4 equal to zero flag,respectively.(see note 1)
10000 to 11011 –Specify M1 through M12, respectively.
11100 to 11110 –Specify T1 through T3, respectively.

ADDRESS, N Field – Output Mode Addressing Assignments
00000 to 01011 –Specifies LO1 through LO12, respectively.
10000 to 11011 –Specifies M1 through M12, respectively.

TEMP Field – Temporary Storage
00 – Do not store the output of the Logic Unit (LU)
01 – Store LU in T1
10 – Store LU in T2
11 – Store LU in T3

SKIP/DISP Field – Skip/Displacement Field
0000 – Continue to next instruction.
$n_3n_2n_1n_0$ – If the output of the LU is zero, then use $n_3n_2n_1n_0$ as 2's complement displacement to be added to the PC for forward and backward branching.
$n_3n_2n_1n_0 = -7$ to $+7$.

Note 1: A counter zero flag is set to "1" when the corresponding counter is reset to "0".

MAXIMUM RATINGS:

Parameter	Symbol	Value	Unit
Storage Temperature	T_{STG}	−55 to +150	°C
Operating Temperature	T_A	0 to 70	°C
Voltage (any pin to V_{SS})	V_{max}	+15 to −0.3	V

DC Electrical Characteristics

(V_{SS} = 0, T_A = 0°C to +70°C, unless otherwise specified)

Parameter	V_{DD}	Symbol	Min.	Max.	Units	Conditions
Supply Voltage	–	V_{DD}	+4.75	+12	V	
Supply Current		I_{DD}	–	28	mA	All outputs open ckt.
Input High Vltg	+5V	V_{IH}	+3.0	V_{DD}	V	
	+9V		+37	V_{DD}	V	
	+12V		+4.7	V_{DD}	V	
Input Low Vltg	+5V	V_{IL}	0	+1.0	V	
	+9V		0	+1.5	V	
	+12V		0	+2.4	V	
Output High Vltg		V_{OH}	V_{DD}−0.2	–	V	@I_{OH}=20uA
Output Low Vltg	+5V	VOL	–	+0.6	V	@I_{OL}=2.5mA
			–	+0.4	V	@I_{OL}=1.6mA
	+9V		–	+0.6	V	@I_{OL}=6.2mA
			–	+0.4	V	@I_{OL}=4.1mA
	+12V		–	+0.6	V	@I_{OL}=12mA
		–		+0.4	V	@I_{OL}=8mA
Output Source Current	+5V	I_{source}	−770	–	µA	@V_{OH}=+2.0V
			−20	–	µA	@V_{OH}=+4.8V
	+9V		−640	–	µA	@V_{OH}=+6.0V
			−25	–	µA	@V_{OH}=+8.7V
	+12V		−1	–	mA	@V_{OH}=+8V
			−50	–	µA	@V_{OH}=11.5V
Output Sink Current	+5V	I_{sink}	+3.5	–	mA	@V_{OL}=+0.8V
			+2.5	–	mA	@V_{OL}=+0.6V
			+1.6	–	mA	@V_{OL}=+0.4V
	+9V		+8.3	–	mA	@V_{OL}=+0.8V
			+6.2	–	mA	@V_{OL}=+0.6V
			+4.1	–	mA	@V_{OL}=+0.4V
	+12V		+15	–	mA	@V_{OL}=+.8V
			+12	–	mA	@V_{OL}=+.6V
			+8	–	mA	@V_{OL}=+.4V
Input High Current		I_{INH}				
INST IN and MCLR			–	+4.8	µA	@V_{IH}=V_{DD}−.5V
All other Inputs			–	−1.0	µA	@V_{IH}=V_{DD}−.5V
Input Low Current		I_{INL}				
All Inputs			–	+1.9	µA	@V_{IL}=+0.2V

DYNAMIC ELECTRICAL CHARACTERISTICS

(V_{DD} = +4.75 to +12V, T_A = 0 to +70°C, unless otherwise specified; see Fig. 6)

Parameter	Symbol	Min.	Typ.	Max.	Units	Condition
Osc. Frequency	f_{osc} (=1/T1)	–	–	2.0	MHz	
Reset Pulse width	T_{WR}	2.5	–	–	µs	
Reset to Shift Clock Delay	T_S	1.0	–	1.25	µs	@f_{osc}=2.0MHZ
Shift Clock Period	T_C(=4T1)	–	2.0	–	µs	@f_{osc}=2.0MHZ
Load Front Porch	T_F(=T1)	–	500		ns	@f_{osc}=2.0MHZ
Load Back Porch	T_B(=T1)	–	500	–	ns	@f_{osc}=2.0MHZ
Shift Clock To address out delay	T_A	30	–	80	ns	
Shift Clock To Execute Delay	T_E(=T_1)	–	500	–	ns	
Shift Clock To Output Delay	T_O	–	550	–	ns	
Counter Input frequency	f_{cnt}	–	–	100	KHz	
Counter Input pulse width:						
HI		2.0	–	–	µs	
LO		1.0	–	–	µs	

PROGRAM EXAMPLE

A simple example is given below to illustrate how the codes are constructed.

A momentary push-button switch, S is connected to the DI1 input of the ICS as shown in Fig. 7. It is required that every time S is pushed, the output LO1 will toggle (change state). Note that only the transition from the nondepressed to the depressed state should cause LO1 to toggle; if S is held depressed, it will have no further effect on the output.

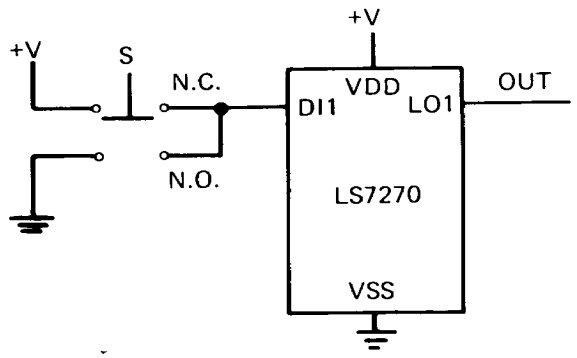


FIGURE 7

Let us assign ICS internal register M1 to store the status of S and M2 to store the status of the output latch LO1 during each sample cycle. A flow chart to describe the program steps is given in Fig 8. The program is in mnemonic code and its binary equivalent is given below.

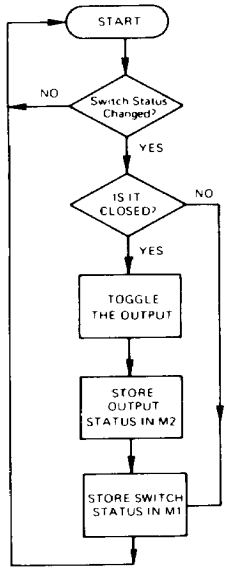


FIGURE 8

Mem. Address (Decimal)	Mnemonic	Binary	Comment
0	STRT: LD DI1, T2	0 0 1 0 0 0 0 0	Read status of S and save it in T2.
1		1 0 0 0 0 0 0 0	
2	XOR M2, −2	0 1 0 0 1 1 1 0	If S changed go to next step, otherwise back to start.
3		1 0 1 1 0 1 0 0	
4	LD T2	0 1 0 0 0 0 0 0	S changed; get ready to test for open/close.
5		1 0 0 0 0 1 1 1	
6	XOR T1, +1	0 0 0 0 0 0 0 1	Closed? If so, skip next step
7		1 0 1 1 0 1 1 1	
8	J UPDT	0 0 0 1 0 0 0 0	Not closed. Go to UPDT routine to update M2
9		0 1 0 1 0 0 0 0	
10	LD M1	0 0 0 0 0 0 0 0	S is closed; so get ready to toggle.
11		1 0 0 0 0 1 0 0	
12	STRC LO1	0 0 0 0 0 0 0 0	Toggled.
13		1 1 1 0 1 0 0 0	
14	STRC M1	0 0 0 0 0 0 0 0	Save current output status in M1
15		1 1 1 0 1 1 0 0	
16	UPDT: LD T2	0 1 0 0 0 0 0 0	Current status of S
17		1 0 0 0 0 1 1 1	
18	STR M2	0 1 0 0 0 0 0 0	Save status in M2
19		1 1 1 0 0 1 0 0	
20	J STRT	0 0 0 0 0 0 0 0	Start new sample cycle
21		0 1 0 1 0 0 0 0	

Note here that memory addresses for successive instructions have incremental value of 2. This is because each memory location can only hold a single byte (8 bits), whereas, an instruction consists of 2 bytes. The low byte of the first instruction is stored at address 0 and the high byte at address 1. The low byte of the second instruction at address 2 and the high byte at address 3 and so on.