'1LT8'

CLARKE - CPU/LCD DRIVER/MEMORY CONTROLLER

External Reference Specification

Revision A

HEWLETT-PACKARD COMPANY

Preston Brown
Dennis Engelbrecht

February 8, 1990

A-1LT8-0601-1

CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

This document contains a description of the '1LT8' IC. This chip is designed as the single custom chip in the Charlemagne calculator. The chip also supports a commercial bus interface that can be used to drive up to 5 commercial memory ICs or ports. The '1LT8' IC is divided into the following functional areas:

1. A 4-bit CPU. The CPU operates on the Saturn bus and is 100% object code compatible with the 1LF2 and 1LK7 CPU's.

2. A MEMORY CONTROLLER capable of interfacing to 5 commercial RAMs, ROMs, or plug-in ports.

3. A Liquid Crystal Display (LCD) DRIVER capable of driving a 64-way multiplexed display.

4. A 32 bit, quartz-crystal controlled TIMER.

5. A 1200-9600 baud full-duplex UART capable of driving RS232 or infrared level signals.

6. A IR LED driver with built-in REDEYE formatting and pacing.

7. A collection of ANALOG CIRCUITRY including: Power Supply; Low Battery Indicator (LBI); Power-on Reset (POR) circuit; Crystal Oscillator; Frequency Multiplier; and Display Voltage Generation.

A chapter has been dedicated to each of these functional areas. Also included are chapters describing the BLOCK DIAGRAM and PIN OUT of the chip.

1LT8 is a highly leveraged IC based on the LEWIS design. The major differences are outlined below:

1. 1LT8 has no built in ROM; and the LEWIS mask options were also removed.

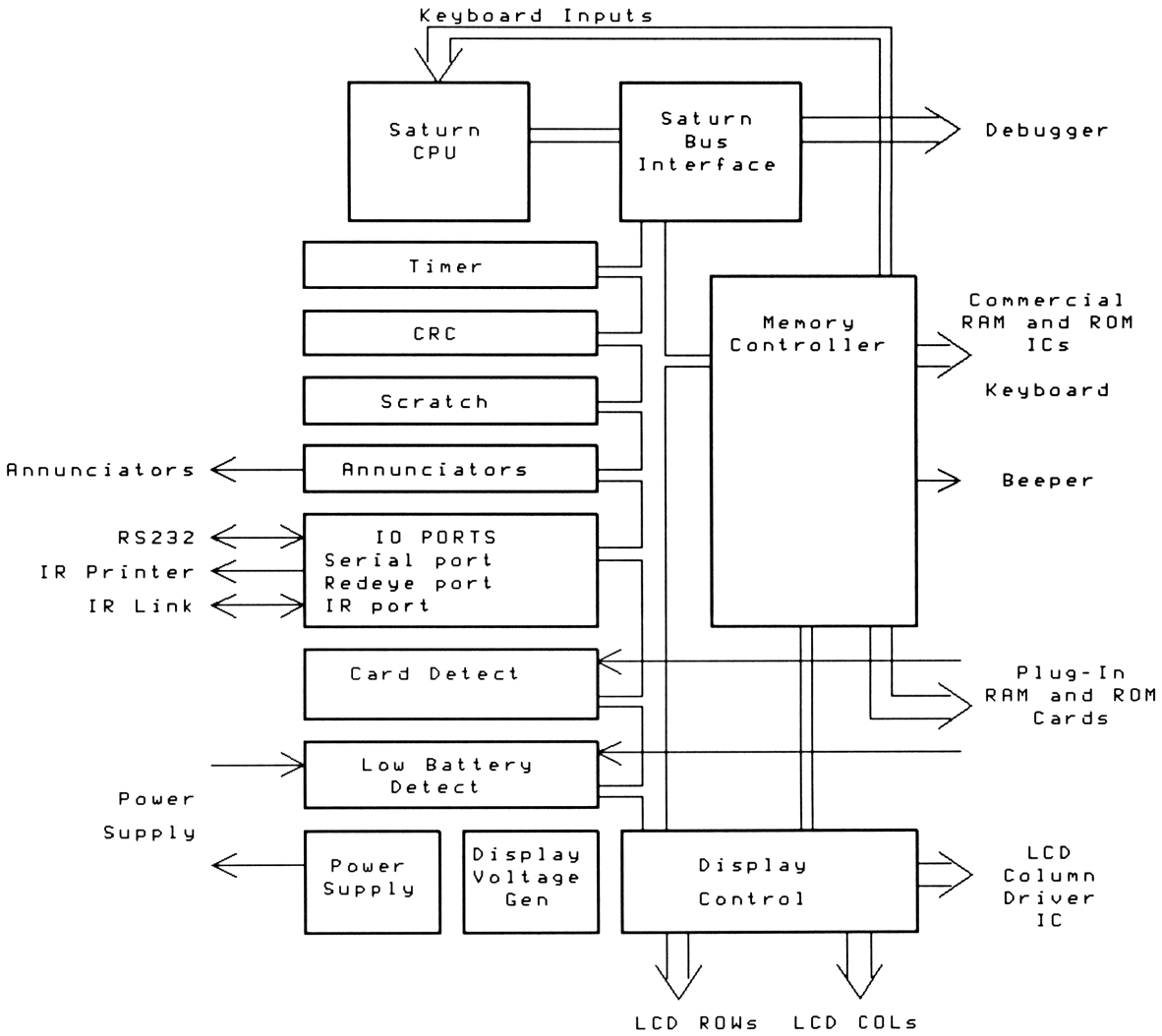
2. The display bit map is not built into the IC, but is contained in external RAM. The bit map ordering has changed.

3. 1LT8 requires one or two external LCD column drivers.

CHAPTER 2
BLOCK DIAGRAM

The following is a simplified block diagram of the 1LT8 chip.

Figure 2.1. 1LT8 Block Diagram



CHAPTER 3
PIN OUT

3.1 PIN Description

This chapter gives a brief description of each pin's function.

PIN	#	DESCRIPTION
Power Supplies:		
VDD	1	Power Supply: 4 to 5.5 Volts.
GND	4	0 Volts.
TGND	1	Burn-in Ground; isolated from GND by a diode and resistor.
VH	1	Most positive display voltage : VDD to 9 volts.
VSSH	1	Most negative display voltage : GND to VDD
VCO	1	Card power = VDD when CPU is running.
VB[0:2]	3	Battery voltage inputs.
Power supply generation:		
PMP	1	Power-supply connection to the gate of an external transistor used to charge the external inductor.
DIODE	1	A p-channel device from this pin to VDD is used to discharge the inductor into the VDD supply.
Display:		
LC[1:4]	4	Display Columns.
LR[1:64]	64	Display Rows.
LA[1:6]	6	Annunciator (non-multiplexed segment) drivers.
LN	1	Common (Row) line for Annunciators.

Interface to external display drivers:

LD[0:1]	2	Data output to external display drivers, synchronized to XSCL.
XSCL	1	Clock output.
LP	1	Latch output pulse.
LFR	1	Frame clock output, sets driver polarity.
VNCS	1	Column non-select voltage.

Saturn Bus:

NSTR	1	NOT STROBE; System clock, active low.
NCD	1	NOT COMMAND-DATA; low during NSTR for a command, high for data.
BUS[0:3]	4	4 bit multiplexed address, data, and command bus.
WAIT	1	Wait input; causes CPU to wait with NSTR low; for slow devices.
HALT	1	When asserted high, the CPU completes the current instruction, tristates the BUS, NSTR, and NCD lines and waits.

Memory Controller:

MA[0:17]	18	Address lines, aliased as IR[0:8], OR[0:8] (Keyboard I/O)
MD[0:7]	8	Data lines.
NCE[1]	1	ROM chip enable, read operations only.
NCE[2:3]	2	RAM or ROM Chip enables; active low. NCE[3] can also be mapped to NOT MA[18] by setting the DA19 bit.
CE[1:2]	2	Card enables; active high.
NWE	1	Read/Write Control output; active low.
NOE	1	Card output enable; active low.
CDT[1:2]	2	Card Detect inputs; senses NWPOUT (Write protect output) of plug-in cards. High indicates card

present and writable, Low indicates card present and write protected, Float indicates that no card is present. If a card is inserted or removed the CPU is interrupted and HS3 is set.

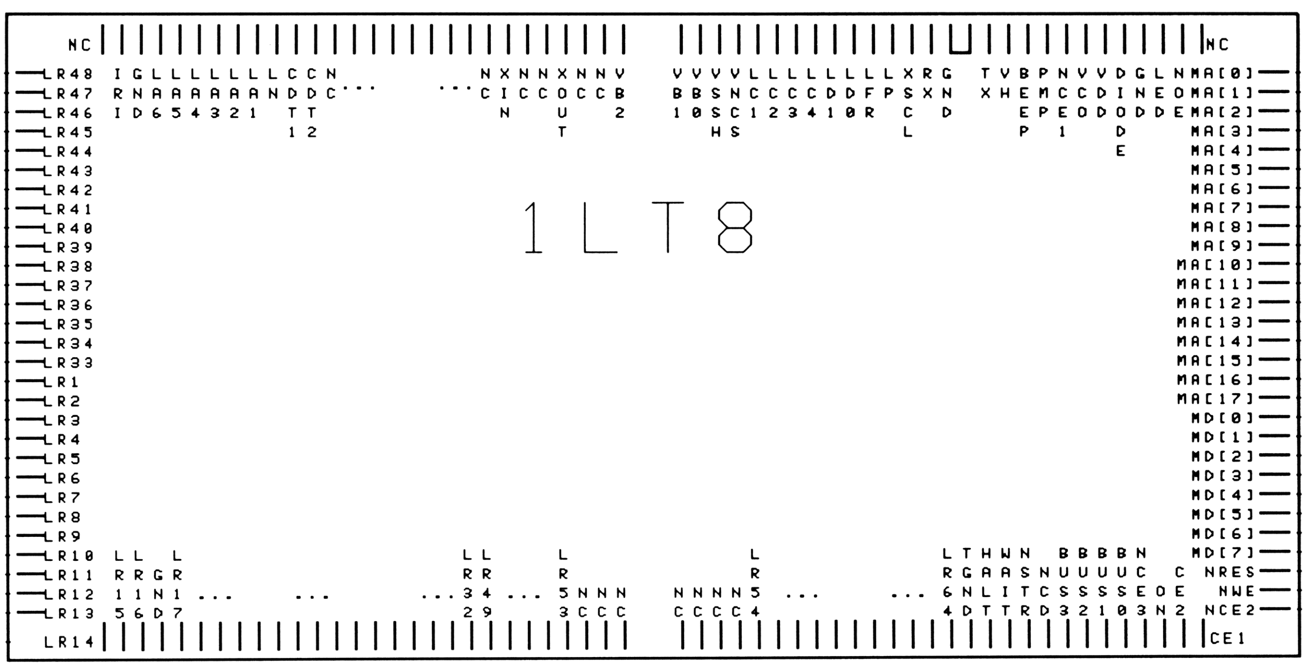
Other:

NRES	1	Reset; active low input.
ON	1	On Key; drives the CPU signal IR15X.
BEEP	1	Beeper Driver, high voltage (VH) OR11
LED	1	LED output driver.
IRI	1	Infrared I/O input pin.
RX,TX	2	Serial I/O.
XIN, XOUT	2	Quartz crystal connections.
TOTAL	**147**	

3.2 **Package**

1LT8 is packaged in a TAB package with outer leads and size identical to the LEWIS TAB package. The inner leads have changed to allow 1LT8 to have 147 pins in a reasonable die size of nnnn by sorrynnnnnons. The following figure gives the pin-out of the package. The TAB package includes 166 pins on a 20 mil pitch.

Figure 3.1. Pin Out



CHAPTER 4
SYSTEM BUS DESCRIPTION

This chapter is a generalized discussion of the 1LT8 system bus. The 1LT8 chip includes a CPU and several peripheral devices. The communication between the CPU and on-chip devices is the same as for external devices except that the communication between the CPU and on-chip devices is not driven onto the external bus unless the chip is in VERBOSE mode. See the section on "Device Modes" in the chapter entitled "TESTING" for a full discussion of when the external bus is active.

The 1LT8 CPU is functionally compatible with the LEWIS CPU. This chapter describes the 1LT8 CPU system bus structure, protocol, and timing. Also included is a description of the bus commands, device addressing, and the power down and wakeup characteristics. A description of the addressing and configuration of specific on-chip devices is found in the chapter entitled "SATURN BUS INTERFACE".

4.1 Bus Structure

The 1LT8 CPU bus consists of 6 lines including:

4	BUS[0:3]	- data lines, driven by the CPU or system devices.
1	NSTR	- driven by the CPU.
1	NCD	- usually driven by the CPU.

Soft configured peripheral chips have two additional lines:

1	DAISYIN	- (DIN) input to device
1	DAISYOUT	- (DOUT) output from device, may be tied to next device's DIN

An additional line may be used by slow peripherals:

1	WAIT	- input to CPU

NOTE : A 'N' before a signal name denotes negative true logic.

4.1.1 General Protocol

NSTR is driven by the CPU and serves to synchronize all bus transfers. It is not a true system clock since it can remain inactive for several cycles while the CPU is performing internal manipulations. All address, data, and bus commands are transferred on the BUS[0:3]. The BUS[0:3] is driven during NSTR low by either the CPU or the device the CPU is accessing. This data is latched either by the CPU or by the receiving device on the rising edge of NSTR.

All bus operations are initiated by the CPU. The CPU starts a specific transfer on the bus by driving the NCD line low before NSTR goes low. While NCD and NSTR are low the CPU drives a bus command on the BUS[0:3] and all devices in the system latch the command on the rising edge of NSTR. This strobe is referred to as a command strobe. The bus command issued during a command strobe specifies the operation that is to be performed on each succeeding NSTR until another bus command is issued. At all times when data or address is being transferred NCD is held high. A strobe issued while NCD is high is referred to as a data strobe.

4.1.2 Bus Commands

The bus commands are:

COMMAND		DESCRIPTION
0	NOP	All devices ignore NSTR until a new command is loaded.
1	ID	The unconfigured device that has its DAISY-IN high sends its 5-nibble ID on the following data strobes, starting with the low- order nibble of the ID.
2	PC READ	(PC)->BUS or read using the Program Counter (PC). The device addressed by its program counter sends data pointed to by its local program counter on each following data strobe and all devices increment their local program counters once each data strobe. A dummy strobe immediately follows the issuance of this bus command (see Section 4.1.4).
3	DP READ	(DP)->BUS or read using the Data Pointer (DP). The device addressed by its data pointer sends data pointed to by its local data pointer on each following data strobe and

		all devices increment their local data pointers once each data strobe. A dummy strobe immediately follows the issuance of this bus command (see Section 4.1.4).
4	SET MODE	All data strobes following this command load data into the MODE register to set test modes. See Chapter 17 for details on test modes.
5	DP WRITE	BUS->(DP) or write using DP. The device addressed by its data pointer loads the data on the following data strobes into the location pointed to by its local data pointer and all devices increment their local data pointer once each data strobe.
6	LOAD PC	BUS->PC or load PC. All devices load the data on the following 5 data strobes into their local program counter, starting with the low-order nibble. After all 5 nibbles are transferred the command code is automatically changed to a 2, PC READ (see Section 4.1.3).
7	LOAD DP	BUS->DP or load DP. All devices load the data on the following 5 data strobes into their local data pointer, starting with the low- order nibble. After all 5 nibbles are transferred the command code is automatically changed to a 3, DP READ (see Section 4.1.3).
8	CONFIGURE	The unconfigured device that has its DAISY-IN high loads the following 5 data nibbles into its configuration register starting with the low-order nibble.
9	UNCONFIGURE	The device currently addressed by its data pointer unconfigures itself. The device then responds to CONFIGURE and ID bus commands only. The local data pointers must be loaded immediately preceding an unconfigure command.
A	POLL	All peripherals that require service pull one data line high during the next NSTR low (see Section 4.6).
B	Reserved	This command is the same as NOP for the internal peripherals and is not issued by the CPU.
C	BUSCC	The device currently addressed by its local data pointer performs a specific operation as

		defined by the individual device. This command is the same as NOP for the internal peripherals and is not issued by the CPU.
D	Reserved	This command is the same as NOP for the internal peripherals and is not issued by the CPU.
E	SHUTDOWN	When the CPU has received a SHUTDN instruction it issues this command before shutting down. Each device responds based on its own special requirements to this command (see Section 4.5).
F	RESET	All devices reset their configuration flags (if applicable) and perform other local resets based on their own special requirements.

4.1.3 **Command Auto-Switch**

There exists one special case in which all devices change their current bus command. This is called 'auto-switch' and occurs following the load of either the PC or the DP. On the rising edge of NSTR after the 5th nibble of address has been loaded all devices clear bit 2 of their command latch changing the bus command from either a LOAD PC to PC READ or LOAD DP to DP READ.

4.1.4 **Dummy Strobe**

Immediately following a PC READ bus command, a DP READ bus command, and a command auto-switch the CPU issues a 'dummy strobe'. This dummy strobe appears as a data strobe except that no data is transferred during this period and devices do not increment their local address registers. The dummy strobe provides memory devices a full strobe cycle for the first access and therefore allows data pipelining.

4.2 **Addressing**

All bus peripherals have local auto-incrementing address pointers for accessing instructions or data. These local pointers are referred to as PC (program counter) and DP (data pointer). As their names imply, the PC is used for accessing instructions and the DP for data. To bus peripherals however, there is no distinction between the two. The CPU automatically updates (loads) these local pointers whenever necessary for instruction fetches or data transfers.

Each bus peripheral is either hard addressed at a specific address (hard configured) or capable of being dynamically located within the address space (soft configured). A bus peripheral only responds to READ and WRITE commands if its local address register (PC or DP depending on the particular command) is within its configured address space.

All bus peripherals are designed so that the upper-order bits of the local address register can be compared with the upper-order bits of the device's configuration register (hard or soft). If these bits are identical, the device has an address match and will respond to read and write commands (and the unconfigure and BUSCC commands if applicable). Each device with a given address space size compares a given number of the upper-order bits of address. For example, a device with an address space size of 2K nibbles requires 11 bits of address leaving the upper 9 bits for its configuration register.

4.2.1 **Soft Configuration**

A soft configured device powers up unconfigured. When unconfigured a device responds only to the ID and CONFIGURE commands and drives its DAISYOUT low. A device's ID code is used to identify the device before it is configured. If a soft configured device is unconfigured and has its DAISYIN line high, it sources its 5-nibble ID code starting with the low-order nibble on the 5 data strobes immediately following the issuance of an ID command (no dummy strobe is issued). The 5-nibble ID code contains information on the device type and the address space required by the device.

Since the BUS[0:3] is precharged low before each strobe, the CPU will read an ID of all zeros if all devices are configured (or are unconfigured but have DAISYIN low).

A soft configured device is assigned its address configuration by the CONFIGURE command. If an unconfigured device has its DAISYIN line high, it loads the configuration address that is issued on the 5 data strobes immediately following the CONFIGURE command (low-order nibble first) into its configuration register. A device may actually latch only the number of high-order bits it requires as determined by its address space size.

After being configured a device no longer responds to either an ID or CONFIGURE command. A configured device drives DAISYOUT to the same logic level as DAISYIN. The DAISYOUT of one device may be tied to the DAISYIN of second device. In this way many devices may be daisy-chained together in a way that they can be configured one at a time to different addresses. After being configured a device waits until the next command strobe to set its configuration flag

in order to delay DAISYOUT so that the next device on the daisy-chain will not be configured simultaneously.

A device may be unconfigured by either a RESET or UNCONFIGURE bus command. The bus RESET command unconfigures all soft configured devices in the system. A device responds to an UNCONFIGURE command by clearing its configuration flag if the DP is within its address configuration.

A device DAISYIN may be tied to the CPU's OR lines to provide software control over when that device will respond to ID, CONFIGURE, and UNCONFIGURE commands. When multiple soft configured devices are required they are typically chained together; DAISYOUT to DAISYIN.

4.2.2 **Hard Configuration**

A hard configured device powers-up configured to a specific address. It will not respond to an ID, CONFIGURE, or UNCONFIGURE command and a bus RESET will not affect its configuration. If the device has a DAISYOUT, it is always driven to the same logic level as its DAISYIN.

4.3 **Data Transfer**

All information that is transferred on the BUS[0:3] is latched on the rising edge of NSTR.

The CPU loads all devices' local address registers by issuing a LOAD PC or LOAD DP bus command followed by 5 data strobes of address, least significant nibble first. After the last nibble of address has been loaded all devices auto-switch to a PC READ or DP READ bus command. The CPU may then read the contents of that address location by issuing one dummy strobe followed by 1 to 16 data strobes during which the CPU latches the BUS[0:3] data. The CPU may read without first loading the local address registers by issuing a PC READ or DP READ, followed by a dummy strobe, followed by 1 to 16 data strobes. The CPU precharges the BUS[0:3] low each cycle before NSTR goes low. Therefore if no device responds the CPU reads zeros.

The CPU writes the contents of a specific addressed location similarly. It is not required to load the local address registers immediately before issuing a PC WRITE or DP WRITE command. The write command is followed by 1 to 16 data strobes during which the addressed device latches the BUS[0:3] data.

All devices increment their active address register (PC or DP) once each data strobe during read and write operations. It is

possible for a read or write operation to begin in one device and cross the address boundary into another device. In general, data transfers should not be limited to 16 nibbles.

Two other types of data transfers are ID, which is simply a 5-nibble read with no address load or dummy strobe, and CONFIGURE, which is a 5-nibble write with no address load. Both these data transfers require that a device be unconfigured and that the DAISYIN line be high. POLL is a unique read and is discussed in Section 4.6.

4.4 **Wait**

A slow peripheral device can cause the CPU to wait during a data strobe by driving WAIT high. While WAIT is high the CPU will hold NSTR low and will hold BUS[0:3] valid if a data nibble is being output. After WAIT goes low, the CPU will raise NSTR and latch BUS[0:3] if data is being input. Wait is resistively held low by the CPU. It should only be driven by a device if that device's address configuration matches the current address.

4.5 **Shutdown; Wakeup**

The 1LT8 CPU can be shutdown under software control. The CPU executes a SHUTDN instruction by issuing a SHUTDOWN bus command and on the next cycle stopping the system clock (NSTR) and its own local oscillator. While in shutdown mode all data stored in CPU resident memory is preserved. The CPU is brought out of shutdown mode by either pulling an Input Register line high, or by driving NCD low in a special manner.

NCD is driven low to wake up the CPU primarily by a bus peripheral that needs service while the system is in shutdown mode. The peripheral must release NCD when the CPU begins an operation by driving NSTR low. If a peripheral wakes up the CPU and the CPU shuts down without satisfying its service request the device must not wake up the CPU again until its service request has been satisfied and it needs service again. This prevents a situation where the operating system does not know how to handle a device's service request and therefore cannot shutdown. For more information on power down and wake up see Section 5.3.2.

4.6 **Service Poll**

If a device needs service while the CPU is operating it must wait until the CPU executes a service request instruction (SREQ), or, if it has the capability, interrupt the CPU. The SREQ instruction causes the CPU to issue a POLL bus command followed by one data strobe during which the CPU latches the BUS[0:3] data in the manner of a usual read. A device may respond to the service POLL by pulling one of the BUS[0:3] lines high. Since the CPU precharges the BUS[0:3] low every cycle before NSTR goes low the data read by the CPU is a binary OR of all devices' responses.

CHAPTER 5
1LT8 CPU

5.1 INTRODUCTION

The 1LT8 CPU is identical to the SACAJAWEA/LEWIS CPU which is a leveraged redesign of the 1LK7 CPU. This CPU provides faster instruction execution times but still maintains full compatibility with the 1LF2/1LK7 and the HP-71B bus architecture.

The CPU internal and external data paths are 4 bits wide. Memory is accessed in 4-bit quantities (referred to as nibbles) using 20 bit addresses yielding a physical address space of 512K bytes. The CPU internal word size is 64-bits. Operations are performed on data strings up to 16 nibbles in length.

Section 5.6 is a summary of differences between the 1LT8 CPU and the 1LK7 for those already familiar with the 1LK7.

5.2 ARCHITECTURE

5.2.1 CPU Overview

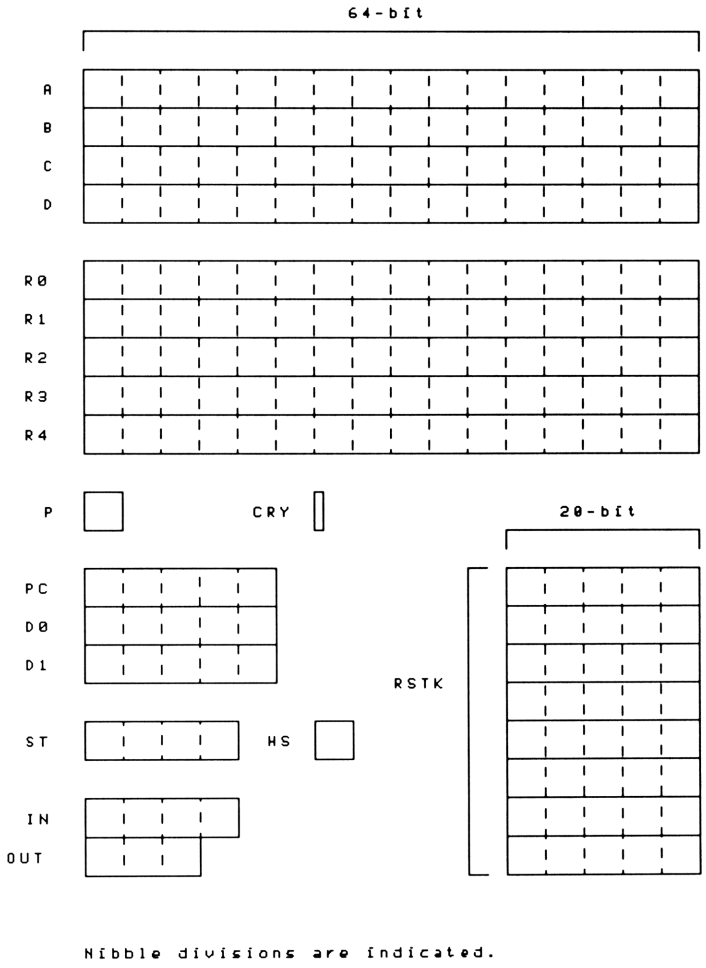
The 1LT8 CPU is a proprietary CPU optimized for high-accuracy BCD math and low power consumption.

Operations performed by the 4-bit ALU include ADD, SUBTRACT, AND OR, and ones, twos, nines and tens complements. A shifter allows bit-shift, nibble-shift and nibble rotate. The CPU internal word size is 64-bits (16 nibbles). Selected portions of a word (fields) may be operated on while leaving the remainder of the word unchanged. Operations on fields longer than one nibble are performed serially with each successive nibble requiring an additional CPU cycle. For information regarding the full CPU instruction set see Section 5.3 INSTRUCTION SET.

5.2.2 **Registers**

Figure 5.1 shows the programmers model of the CPU. A description of the register groups and their functions follows.

Figure 5.1. CPU Register Set



There are 198 nibbles of CPU resident RAM organized to provide four 64-bit working registers (A,B,C,D), five 64-bit scratch registers (R0,R1,R2,R3,R4), two 20-bit data pointer registers (D0,D1), an eight level hardware return stack (RSTK) for 20-bit addresses, and 16 bits of program status.

There are two types of registers on the CPU; those used for data and arithmetic operations, and those used for program and system control.

ARITHMETIC/DATA REGISTERS:

A	64-bits	Working register - I/O register
B	64-bits	Working register
C	64-bits	Working register - I/O register
D	64-bits	Working register
R0	64-bits	Scratch register
R1	64-bits	Scratch register
R2	64-bits	Scratch register
R3	64-bits	Scratch register
R4	64-bits	Scratch register
CRY	1-bit	Flag adjusted by arithmetic operations and tests

CONTROL REGISTERS:

P	4-bits	Pointer register
D0	20-bits	Address pointer register
D1	20-bits	Address pointer register
PC	20-bits	Program Counter
RSTK	20-bits	8-level subroutine stack
ST	16-bits	Program status flags
HS	4-bits	Hardware status flags
OUT	12-bits	Keyscan/write only Output Register
IN	16-bits	Keyscan/read only Input Register

For the 1LT8 chip, only 10 bits of the OUT register and 10 bits of the IN register are used (including IR[15] (ON) and OR11 (BEEP)). When the unused bits of the OUT register are written to, nothing will happen external to the chip.

All arithmetic operations are performed using the 4 working registers: A, B, C, and D. Data transfers are performed principally with the A and C registers. The scratch registers are

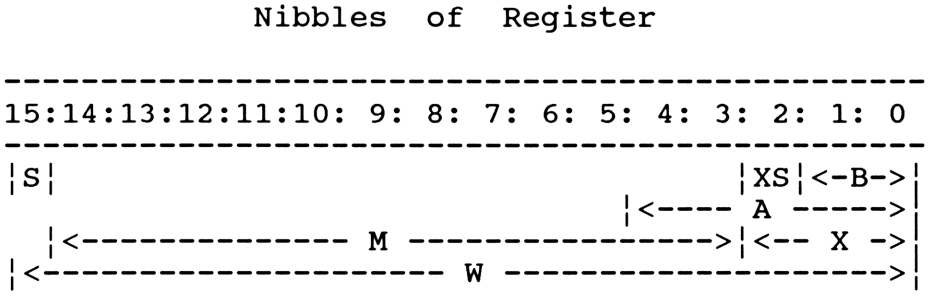
used to temporarily hold the contents of the working registers.

5.2.2.1 **Field Selection**

Subfields of the working registers may be manipulated using field selection. The possible field selections range from the entire register to any single nibble of the register. Certain subfields are designed for use in BCD calculations and others are designed for use in general data manipulation and data access. Nibble 0 is considered to be least significant by the ALU.

FIELD SELECTION FIELDS

P	nibble pointed to by P register
WP	nibble 0 through nibble pointed at by P
XS	nibble 2 - Exponent sign
X	nibbles 0-2 - Exponent and exponent sign
S	nibble 15 - Mantissa sign
M	nibbles 3-14 - Mantissa
B	nibbles 0-1 - Exponent or byte field
W	nibbles 0-15 - Whole word
A	nibbles 0-4 - Address field



5.2.2.2 **Carry Bit**

The Carry bit is adjusted when an arithmetic operation or test is performed. During a calculation, such as incrementing or decrementing a register, it is set if the calculation overflows or borrows and cleared if it does not. During a arithmetic test, such as comparing two registers for equality, it is set if the test is true and cleared if it is not.

5.2.2.3 Pointer Registers

The CPU uses D0 and D1 to provide the source addresses for all

external data transfers. During execution of a data transfer instruction the local DPs of all peripheral chips are loaded with either D0 or D1. The data access will then occur utilizing the auto-increment feature of the local DPs. See SYSTEM BUS DESCRIPTION. It should be noted that D0 and D1 are merely registers and do not increment during data transfer.

The Pointer register is used in Field Selection operations with the working registers.

5.2.2.4 **Program Counter and Return Stack**

The CPU resident PC is a true counter that increments in real time during instruction fetches. It can be accessed only using GOTO, GOSUB and RETURN instructions. Relative GOTOs and GOSUBs are handled using the ALU to modify the PC. Absolute GOTOs and GOSUBs are handled using a parallel load feature. All non-increment modifications to the PC are eventually followed by a load to the local PCs on all peripheral chips.

The subroutine return stack consists of eight 20-bit registers implemented as a LIFO. Subroutine call and return instructions automatically push and pop addresses on this stack. If a ninth address is pushed onto the stack, the oldest address will be lost and will be replaced by zero if it is eventually popped from the stack. This feature insures that repeated returns will eventually end at address zero. The return stack can also be manipulated through use of the push (RSTK=C) and pop (C=RSTK) instructions.

5.2.2.5 **Status Bits**

Additional program control is provided by the 16-bit Program Status register and the 4-bit Hardware Status register. Each Program status bit can be individually set, reset, and tested. The upper four bits may only be accessed individually. The lower 12 Program status bits may be collectively manipulated as the ST register.

The four Hardware Status bits are set (but not cleared) by hardware related events, and must therefore be cleared beforehand in order to detect a particular occurrence. They are individually accessible by name. The Module Pulled bit (MP) is set when the NINTX line is pulled low (regardless of whether an interrupt is actually executed). The Sticky Bit (SB) is set when a non-zero bit or digit shifts off the right end of a working register as the result of a shift right instruction, or the least significant nibble of a working register is non-zero prior to a shift right circular instruction. The Service Request (SR) bit is set as a result of a response to the SREQ? instruction. The external

Module Missing bit (XM) is set by execution of a "00" opcode (RTNSXM instruction). Since the BUS[0:3] is precharged low, the CPU will receive a RTNSXM instruction if no device responds to a PC READ bus command.

HARDWARE STATUS: 4 bits

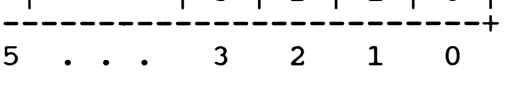
Bit	Symbol	Name
3	MP	Module Pulled
2	SR	Service Request
1	SB	Sticky Bit
0	XM	External Module Missing

5.2.3 **Loading Data from Memory**

When data is read from an external device into a register, the CPU places the lowest addressed nibble in the least significant nibble of the selected field in the register. For example, if the data shown below in memory is read into the C register using the **C=DAT1 4** instruction, the data in the register will be arranged as shown:

Memory Location	Value
1000	0
1001	1
1002	2
1003	3
.	
.	
.	

C Register



This principle also applies to loading constants into a CPU register such as C, D0,or D1, since the CPU must read the constant from the instruction opcode in memory. For example, the instruction LCHEX 3210 produces the opcode 330123 and the C register is loaded as shown above (assuming P=0).

5.2.4 **Storing Data in Memory**

When data is written from a register to an external device, the CPU places the least significant nibble of the register in the first addressed nibble of the memory location. For example, if the data shown above in the C register is written to memory using the **DAT1=C 4** instruction, the data will be written to memory as shown.

5.2.5 **HEX/DEC Modes**

All arithmetic operations, except for those listed below, are performed according to the HEX or DEC mode setting. The mode is set using the SETHEX or SETDEC instruction. The following operations are performed in HEX regardless of the mode setting.

```
C+P+1
D0=D0+ n          D0=D0- n
D1=D1+ n          D1=D1- n
P=P+1             P=P-1
A=A+CON fs,N      A=A-CON fs,N
B=B+CON fs,N      B=B-CON fs,N
C=C+CON fs,N      C=C-CON fs,N
D=D+CON fs,N      D=D-CON fs,N
```

5.3 **INSTRUCTION SET**

5.3.1 **General Information**

Many CPU instructions modify only a selected field of a register. Nibbles not explicitly changed by an instruction are left unchanged.

The offset addresses for the relative jump and gosub instructions are in two's compliment form with the least significant nibble first. Offsets for jump instructions are relative to the least significant nibble of the offset. Offsets for gosub instructions are relative to the first nibble of the next instruction.

Conditional instructions have two cycle counts listed. The second (and smaller count) is if the condition (test) is false.

5.3.1.1 **Glossary of Symbols**

DATn Memory location addressed by either D0 or D1.

fs Field selection specifier.

D Number of nibbles operated upon. Used in calculation of cycle counts.

5.3.2 **Instruction Description**

MNEMON	CODE	CYC	FLG	DESCRIPTION
RTNSXM	00	9	HS0	Return and set HS0
RTN	01	9	---	Return from subroutine
RTNSC	02	9	CRY	Return and set CRY
RTNCC	03	9	CRY	Return and clear CRY
SETHEX	04	3	---	Set ALU to HEX mode
SETDEC	05	3	---	Set ALU to BCD mode
RSTK=C	06	8	---	Push C(A) onto stack
C=RSTK	07	8	---	Pop stack to C(A)
CLRST	08	5	ST	Clear ST[0:11]
C=ST	09	5	---	Load C(X) with ST[0:11]
ST=C	0A	5	ST	Load ST[0:11] with C(X)
CSTEX	0B	5	ST	Exchange ST[0:11] with C(X)
P=P+1	0C	3	CRY	Hexadecimal increment of P. F wraps to 0.
P=P-1	0D	3	CRY	Hexadecimal decrement of P. 0 wraps to F.

r=r&s fs 0Eft 4+D --- Logical AND
r=r!s fs 0Eft 4+D --- Logical OR

f:3210	fs	t: HEX	t: HEX
X000	P	0 A=B&A	8 A=B!A
X001	WP	1 B=C&B	9 B=C!B
X010	XS	2 C=A&C	A C=A!C
X011	X	3 D=C&D	B D=C!D
X100	S	4 B=B&A	C B=B!A
X101	M	5 C=C&B	D C=C!B
X110	B	6 A=A&C	E A=A!C
0111	W	7 C=C&D	F C=C!D
1111	A		

MNEMON	CODE	CYC	FLG	DESCRIPTION
RTI	0F	9	---	Return from interrupt and enable interrupt handling.
ss=r	10s	19	---	Copy A or C to scratch register.
r=ss	11s	19	---	Copy scratch register to A or C.
rssEX	12s	19	---	Exchange A or C with scratch register.

s:3210

X000	scratch register R0
X001	scratch register R1
X010	scratch register R2
X011	scratch register R3
X100	scratch register R4
X101	scratch register R1
X110	scratch register R2
X111	scratch register R3
0XXX	working register A
1XXX	working register C

MNEMON	CODE	CYC	FLG	DESCRIPTION
d=r	13t	8	---	Copy A field of A or C to D0 or D1
d=rS	13t	7	---	Copy nibbles 0-3 of A or C to nib. 0-3 of D0 or D1
rdEX	13t	8	---	Exchange A field of A or C r<-dwith D0 or D1
rdXS	13t	7	---	Exchange nibbles 0-3 of A r<-d or C with nib. 0-3 of D0 or D1.

t: HEX		t: HEX	
0	D0=A	8	D0=AS
1	D1=A	9	D1=AS
2	AD0EX	A	AD0XS
3	AD1EX	B	AD1XS
4	D0=C	C	D0=CS
5	D1=C	D	D1=CS
6	CD0EX	E	CD0XS
7	CD1EX	F	CD1XS

MNEMON	CODE	CYC	FLG	DESCRIPTION
DATn=r A	14t	17	---	Write A field of A or C to location addressed by D0 or D1.
DATn=r B	14t	14	---	Write B field of A or C to location addressed by D0 or D1.
r=DATn A	14t	18	---	Read A field of A or C from location addressed by D0 or D1.
r=DATn B	14t	15	---	Read B field of A or C from location addressed by D0 or D1.

t: HEX		t: HEX	
0	DAT0=A A	8	DAT0=A B
1	DAT1=A A	9	DAT1=A B
2	A=DAT0 A	A	A=DAT0 B
3	A=DAT1 A	B	A=DAT1 B
4	DAT0=C A	C	DAT0=C B
5	DAT1=C A	D	DAT1=C B
6	C=DAT0 A	E	C=DAT0 B
7	C=DAT1 A	F	C=DAT1 B

MNEMON	CODE	CYC	FLG	DESCRIPTION
DATn=r fs	15tn	16+D	---	Write selected field of A or C to location addressed by D0 or D1.
DATn=r N	15t(N-1)	16+N	---	Write nib. 0-(N-1) of A or C to location addressed by D0 or D1.
r=DATn fs	15tn	17+D	---	Read selected field of A or C from location addressed by D0 or D1.
r=DATn N	15t(N-1)	17+N	---	Read nib. 0-(N-1) of A or C to location addressed by D0 or D1.

t: 3210		n: 3210	
X000	DAT0=A	X000	P
X001	DAT1=A	X001	WP
X010	A=DAT0	X010	XS
X011	A=DAT1	X011	X
X100	DAT0=C	X100	S
X101	DAT1=C	X101	M
X110	C=DAT0	X110	B
X111	C=DAT1	X111	W

0XXX	fs specified by n
1XXX	nibbles 0-N

MNEMON	CODE	CYC	FLG	DESCRIPTION
D0=D0+ n	16n	7	CRY	Hex increment of D0 by n+1. F wraps to 0.
D1=D1+ n	17n	7	CRY	Hex increment of D1 by n+1. F wraps to 0.
D0=D0- n	18n	7	CRY	Hex decrement of D0 by n+1. 0 wraps to F.
D0= nn	19nn	4	---	Load B field of D0 with nn.
D0= nnnn	1Annnn	6	---	Load nib. 0-3 of D0 with nnnn.
D0= nnnnn	1Bnnnnn	7	---	Load D0 with nnnnn.
D1=D1- n	1Cn	7	CRY	Hex decrement of D1 by n+1. 0 wraps to F.
D1= nn	1Dnn	4	---	Load B field of D1 with nn.
D1= nnnn	1Ennnn	6	---	Load nib. 0-3 of D1 with nnnn.
D1= nnnnn	1Fnnnnn	7	---	Load D1 with nnnnn.
P= n	2n	2	---	Load pointer with n.
LC nn..n	3tnn..n	3+t	---	Load C with t+1 nibbles nn...n beginning at the pointer position with wrap around capability. This mnemonic is not supported by most assemblers.
LCHEX n.n	Same.	3+t	---	These mnemonics are supported by most assemblers
LC(i) nnnnn	Same.	3+t	---	Same.
LCASC 'c.c'	Same.	3+t	---	Same.
RTNC	400	10 3	---	Return if CRY is set.
GOC	4aa	10 3	---	Jump relative if CRY is set.
RTNNC	500	10 3	---	Return if CRY is clear.
GONC	5aa	10 3	---	Jump relative if CRY is clear.

MNEMON	CODE	CYC	FLG	DESCRIPTION
GOTO	6aaa	11	---	Relative jump.
GOSUB	7aaa	12	---	Relative gosub.
OUT=CS	800	4	---	Load nib. 0 of OR with nib. 0 of C.
OUT=C	801	6	---	Load OR with X field of C.
A=IN	802	7	---	Load nib. 0-3 of A from IR.
C=IN	803	7	---	Load nib. 0-3 of C from IR.
UNCNFG	804	12	---	Unconfigure a configured device addressed by A field of C.
CONFIG	805	11	---	Configure an unconfigured device that has DAISYIN=1 at the address in A field of C.
C=ID	806	11	---	Load the A field of C with the ID of an unconfigured device that has DAISYIN=1.
SHUTDN	807	6	---	CPU issues the NOP bus command to eliminate all memory ops then issues the SHUTDOWN bus command and enters the low power state.
INTON	8080	5	---	Unmask maskable IR interrupts.
RSI	80810	6	---	Causes any IR line high to be considered a new interrupt regardless of whether all IR lines have returned low.
LAHEX n.n	8082tn.	6+t	---	Load A with t+1 nibbles nn...n beginning at the pointer position with wrap around capability.
LA(i) nnnnn	Same.	6+t	---	Same.
LAASC 'c.c'	Same.	6+t	---	Same.
BUSCB	8083	7	---	Issue the BUSCB bus command.

MNEMON	CODE	CYC	FLG	DESCRIPTION
ABIT=0 n	8084n	6	---	Clear specified bit (0 to F) of A register.
ABIT=1 n	8085n	6	---	Set specified bit (0 to F) of A register.
?ABIT=0 n RTNYES	8086n00	16 9	CRY	Return and set CRY if specified bit (0 to F) of A register is 0.
?ABIT=0 n GOYES	8086naa	16 9	CRY	Jump relative and set CRY if specified bit (0 to F) of A register is 0.
?ABIT=1 n RTNYES	8087n00	16 9	CRY	Return and set CRY if specified bit (0 to F) of A register is 1.
?ABIT=1 n GOYES	8087naa	16 9	CRY	Jump relative and set CRY if specified bit (0 to F) of A register is 1.
CBIT=0 n	8088n	6	---	Clear specified bit (0 to F) of C register.
CBIT=1 n	8089n	6	---	Set specified bit (0 to F) of C register.
?CBIT=0 n RTNYES	808An00	16 9	CRY	Return and set CRY if specified bit (0 to F) of C register is 0.
?CBIT=0 n GOYES	808Anaa	16 9	CRY	Jump relative and set CRY if specified bit (0 to F) of C register is 0.
?CBIT=1 n RTNYES	808Bn00	16 9	CRY	Return and set CRY if specified bit (0 to F) of C register is 1.
?CBIT=1 n GOYES	808Bnaa	16 9	CRY	Jump relative and set CRY if specified bit (0 to F) of C register is 1.

MNEMON	CODE	CYC	FLG	DESCRIPTION
PC=(A)	808C	23	---	Indirect Jump. Data addressed by the A field of the A register is placed in the PC and execution resumes there.
BUSCD	808D	7	---	Issue the BUSCD bus command.
PC=(C)	808E	23	---	Indirect Jump. Data addressed by the A field of the C register is placed in the PC and execution resumes there.
INTOFF	808F	5	---	Mask the maskable IR interrupts.
C+P+1	809	8	CRY	Hex increment of A field of C by P+1. Wraps to 0.
RESET	80A	6	---	Issue the RESET bus command. All chips perform a local reset.
BUSCC	80B	6	---	Issue the BUSCC bus command. The device addressed by its DP performs a specialized function.
C=P n	80Cn	6	---	Load nibble n of C with P.
P=C n	80Dn	6	---	Load P with nibble n of C.
SREQ?	80E	7	HS2	Issue POLL bus command and latch system bus into nib 0 of C. HS2 is set if this nibble is not zero.
CPEX n	80Fn	6	---	Exchange the pointer with nibble n of C.

MNEMON	CODE	CYC	FLG	DESCRIPTION
rSLC	81s	21	---	Left circular nibble shift of entire 16 nibble register.
rSRC	81s	21	HS1	Right circular nibble shift of entire 16 nibble register. Sticky bit is set if a non-zero nibble is shifted off.
rSRB	81s	20	HS1	Right bit shift of entire 64 bit register. Sticky bit is set if a non-zero bit is shifted off.

s:3210	
XX00	A register.
XX01	B register.
XX10	C register.
XX11	D register.
00XX	rSLC
01XX	rSRC
11XX	rSRB

MNEMON	CODE	CYC	FLG	DESCRIPTION
r=r+CON fs,n	818frn	5+D	CRY	Hex increment of register r by n+1.
r=r-CON fs,n	818frn	5+D	CRY	Hex decrement of register r by n+1.

f: 3210		r: 3210	
		XX00	A register
X011	X	XX01	B register
X101	M	XX10	C register
X110	B	XX11	D register
0111	W	0XXX	addition
1111	A	1XXX	subtraction

MNEMON	CODE	CYC	FLG	DESCRIPTION
rSRB.F fs	819fr	6+D	HS1	Right bit shift of register field. Sticky bit is set if a non-zero bit is shifted off.

f: 3210		r: 3210	
X000	P	XX00	A register
X001	WP	XX01	B register
X010	XS	XX10	C register
X011	X	XX11	D register
X100	S		
X101	M		
X110	B		
0111	W		
1111	A		

MNEMON	CODE	CYC	FLG	DESCRIPTION
ss=r.F fs	81Afrn	6+D	---	Copy selected field of A or C to scratch register.
r=ss.F fs	81Afrn	6+D	---	Copy selected field of scratch register to A or C.
rssEX.F fs	81Afrn	6+D	---	Exchange selected field of A or C with scratch register.

f:3210		r: 3210	n: HEX		
X000	P	0000	ss=r	X000	R0
X001	WP	0001	r=ss	X001	R1
X010	XS	0010	rssEX	X010	R2
X011	X			X011	R3
X100	S			X100	R4
X101	M			X101	R1
X110	B			X110	R2
0111	W			X111	R3
1111	A			0XXX	A
				1XXX	C

MNEMON	CODE	CYC	FLG	DESCRIPTION
PC=A	81B2	16	---	Absolute jump to the address contained in the A field of the A register.
PC=C	81B3	16	---	Absolute jump to the address contained in the A field of the C register.
A=PC	81B4	9	---	Move the address of the next instruction to the A field of the A register.
C=PC	81B5	9	---	Move the address of the next instruction to the A field of the C register.
APCEX	81B6	16	---	Exchange the A field of A and the current PC (pointing to the next instruction).
CPCEX	81B7	16	---	Exchange the A field of C and the current PC (pointing to the next instruction).
CLRHSN n	82n	3	HSn	Clear the hardware status flags specified by the bitmap of n.
XM=0	821	3	HS0	
SB=0	822	3	HS1	
SR=0	824	3	HS2	
MP=0	828	3	HS3	
CLRHST	82F	3	HS0-3	

n:3210	
XXX1	Missing module; set by RTNSXM
XX1X	Sticky bit; set by rotate and shift right
X1XX	Service request response
1XXX	Module interrupt line (*INT) pulled low

MNEMON	CODE	CYC	FLG	DESCRIPTION
?HS=0 n	83n00	13	CRY	Return and set CRY if all hardware status
RTNYES		6		bits specified by the bit-map of n are 0.
?HS=0 n	83naa	13	CRY	Relative jump and set CRY if all hardware
GOYES		6		status bits specified by the bit-map of n are 0.
?XM=0	83100	13	CRY	Same.
RTNYES		6		
?XM=0	831aa	13	CRY	
GOYES		6		
?SB=0	83200	13	CRY	
RTNYES		6		
?SB=0	832aa	13	CRY	Same.
GOYES		6		
?SR=0	83400	13	CRY	Same.
RTNYES		6		
?SR=0	834aa	13	CRY	Same.
GOYES		6		
?MP=0	83800	13	CRY	Same.
RTNYES		6		
?MP=0	838aa	13	CRY	Same.
GOYES		6		

MNEMON	CODE	CYC	FLG	DESCRIPTION
ST=0 n	84n	4	STn	Clear program status bit n.
ST=1 n	85n	4	STn	Set program status bit n.
?ST=0 n **RTNYES**	86n00	14 7	CRY	Return and set CRY if program status bit n is 0.
?ST=0 n **GOYES**	86naa	14 7	CRY	Jump relative and set CRY if program status bit n is 0.
?ST=1 n **RTNYES**	87n00	14 7	CRY	Return and set CRY if program status bit n is 1.
?ST=1 n **GOYES**	87naa	14 7	CRY	Jump relative and set CRY if program status bit n is 1.
?P# n **RTNYES**	88n00	13 6	CRY	Return and set CRY if the pointer is not equal to n.
?P# n **GOYES**	88naa	13 6	CRY	Jump relative and set CRY if the pointer is not equal to n.
?P= n **RTNYES**	89n00	13 6	CRY	Return and set CRY if the pointer is equal to n.
?P= n **GOYES**	89naa	13 6	CRY	Jump relative and set CRY if the pointer is equal to n.

MNEMON	CODE	CYC	FLG	DESCRIPTION
?XXX A RTNYES	8At00	18 11	CRY	Return and set CRY if test selected in GROUP A below is true for A field.
?XXX A GOYES	8Ataa	18 11	CRY	Jump relative and set CRY if test in GROUP A below is true for A field.
?XXX A RTNYES	8Bt00	18 11	CRY	Return and set CRY if test selected in GROUP B below is true for A field.
?XXX A GOYES	8Btaa	18 11	CRY	Jump relative and set CRY if test in GROUP B below is true for A field.

t:3210	GROUP A	GROUP B
0000	?A=B	?A>B
0001	?B=C	?B>C
0010	?C=A	?C>A
0011	?D=C	?D>C
0100	?A#B	?A<B
0101	?B#C	?B<C
0110	?C#A	?C<A
0111	?D#C	?D<C
1000	?A=0	?A>=B
1001	?B=0	?B>=C
1010	?C=0	?C>=A
1011	?D=0	?D>=C
1100	?A#0	?A<=B
1101	?B#0	?B<=C
1110	?C#0	?C<=A
1111	?D#0	?D<=C

MNEMON	CODE	CYC	FLG	DESCRIPTION
GOLONG	8Caaaa	14	---	Long relative jump. [LOAD PC] BUS,PC<-0+PC+CY0
GOVLNG	8Daaaaa	14	---	Absolute jump. [LOAD PC] BUS,PC<-0+PC+CY0
GOSUBL	8Eaaaa	14	---	Long relative gosub. [LOAD PC] BUS,PC<-RSTK+PC+CY0 RSTK<-PC
GOSBVL	8Faaaaa	15	---	Absolute gosub. [LOAD PC] BUS,PC<-RSTK+0+CY0 RSTK<-PC
?XXX fs **RTNYES**	9ft00	13+D 6+D	CRY	Return and set CRY if the selected test is true on the selected field.
?XXX fs **GOYES**	9ftaa	13+D 6+D	CRY	Jump relative and set CRY if the selected test is true on the field selected.

f:3210

X000	P
X001	WP
X010	XS
X011	X
X100	S
X101	M
X110	B
X111	W
0XXX	t selects test from GROUP A (see 8At00 opcode)
1XXX	t selects test from GROUP B (see 8Bt00 opcode)

MNEMON	CODE	CYC	FLG	DESCRIPTION
r=r+s fs	Aft	3+D	CRY	Arithmetic operation selected by t on field
r=r+r fs	Aft	3+D	CRY	selected by f. CRY is set on carry or borrow.
s=r+s fs	Aft	3+D	CRY	
r=r-1 fs	Aft	3+D	CRY	
r=0 fs	Aft	3+D	---	Arithmetic operation selected by t on field
r=s fs	Aft	3+D	---	selected by f. CRY is not altered.
s=r fs	Aft	3+D	---	
rsEX fs	Aft	3+D	---	

f:3210

X000	P
X001	WP
X010	XS
X011	X
X100	S
X101	M
X110	B
X111	W
0XXX	Selects GROUP C arithmetic operations.
1XXX	Selects GROUP D arithmetic operations.

t:3210	GROUP C	GROUP D
0000	A=A+B	A=0
0001	B=B+C	B=0
0010	C=C+A	C=0
0011	D=D+C	D=0
0100	A=A+A	A=B
0101	B=B+B	B=C
0110	C=C+C	C=A
0111	D=D+D	D=C
1000	B=B+A	B=A
1001	C=C+B	C=B
1010	A=A+C	A=C
1011	C=C+D	C=D
1100	A=A-1	ABEX
1101	B=B-1	BCEX
1110	C=C-1	CAEX
1111	D=D-1	DCEX

MNEMON	CODE	CYC	FLG	DESCRIPTION
r=r-s fs	Bft	3+D	CRY	Arithmetic operation selected by t on field
r=r+1 fs	Bft	3+D	CRY	selected by f. CRY is set on carry or borrow.
s=s-r fs	Bft	3+D	CRY	
r=s-r fs	Bft	3+D	CRY	
rSL fs	Bft	4+D	---	Left nibble shift selected by t on field selected by f. CRY is not altered.
rSR fs	Bft	4+D	HS1	Right nib. shift selected by t on field selected by f. HS1 is set if a non-zero nib is shifted off.
r=-r fs	Bft	3+D	CRY	Arithmetic operation selected by t on field
r=-r-1 fs	Bft	3+D	CRY	selected by f. CRY is set on borrow.

f:3210	
X000	P
X001	WP
X010	XS
X011	X
X100	S
X101	M
X110	B
X111	W
0XXX	Selects GROUP E arithmetic operations.
1XXX	Selects GROUP F arithmetic operations.

t:3210	GROUP E	GROUP F
0000	A=A-B	ASL
0001	B=B-C	BSL
0010	C=C-A	CSL
0011	D=D-C	DSL
0100	A=A+1	ASR
0101	B=B+1	BSR
0110	C=C+1	CSR
0111	D=D+1	DSR
1000	B=B-A	A=-A
1001	C=C-B	B=-B
1010	A=A-C	C=-C
1011	C=C-D	D=-D
1100	A=B-A	A=-A-1
1101	B=C-B	B=-B-1
1110	C=A-C	C=-C-1
1111	D=C-D	D=-D-1

MNEMON	CODE	CYC	FLG	DESCRIPTION
r=r+s A	Ct	7	CRY	GROUP C operation selected by t is performed
r=r+r A	Ct	7	CRY	on A field. CRY is set on carry or borrow.
s=r+s A	Ct	7	CRY	
r=r-1 A	Ct	7	CRY	
r=0 A	Dt	7	---	GROUP D operation selected by t is performed
r=s A	Dt	7	---	on the A field.
s=r A	Dt	7	---	
rsEX A	Dt	7	---	
r=r-s A	Et	7	CRY	GROUP E operation selected by t is performed
r=r+1 A	Et	7	CRY	on the a field.
s=s-r A	Et	7	CRY	
r=s-r A	Et	7	CRY	
rSL A	Ft	8	---	GROUP F operation selected by t is performed
rSR A	Ft	8	HS1	on the A field.
r=-r A	Ft	7	CRY	
r=-r-1 A	Ft	7	CRY	

5.4 **INTERRUPT SYSTEM**

The 1LT8 CPU supports several types of non-prioritized interrupts. Interrupts are handled one at a time without nesting. The CPU can be interrupted in the following manners:

1. Pulling the NINTX line low.
2. Pulling the NINT2X line low.
3. Pulling the IR15X (ON) line high.
4. Pulling the KDN line high.

Any interrupt source must remain active for a minimum interval to ensure recognition. This interval corresponds to the longest CPU instruction execution time (32 cycles).

If interrupted, the CPU completes the current instruction, pushes the PC onto the return stack (decrements the SP) and forces the PC to the interrupt routine starting address of 0000F, where F is in the least significant nibble of the PC. The instructions starting at this address are referred to as the interrupt routine. While in the interrupt routine the CPU cannot be reinterrupted. The RTI instruction defines the end of the interrupt routine and re-enables interrupt processing.

Each interrupt type is somewhat specialized for a specific purpose and each is described separately.

5.4.1 **NINTX Interrupt**

NINTX line - level sensitive - active low.

If pulled low the interrupt routine is entered and all interrupts are disabled. Hardware status bit 3 is set any time NINTX is pulled low regardless of whether an interrupt actually occurs. If NINTX is held low the CPU will be repeatedly reinterrupted after each RTI instruction is executed. This is a non-maskable interrupt.

5.4.2 **NINT2X Interrupt**

NINT2X line - level sensitive - active low.

Identical to NINTX except that no hardware status bits are set.

5.4.3 **IR15X Interrupt**

IR15X line - level sensitive - active high.

Non-maskable interrupt. IR15X state is readable using the A=IN and C=IN instructions.

5.4.4 **KDN Interrupts**

KDN line - rising edge sensitive with qualifications - see text.

KDN is the "logic OR" of IR15X (ON) and the IR[0:8] lines. IR[0:8] lines held high will generate only one initial interrupt and will not repetitively reinterrupt the CPU as do the NINTX, NINT2X and IR15X lines. IR[0:6] interrupts are maskable using the INTON and INTOFF instructions.

A rising edge of KDN will generate an interrupt if the CPU is outside of the interrupt routine (RTI) and interrupts are ON (INTON). Once this interrupt has occurred, no further KDN interrupts can occur until another rising edge of KDN occurs outside of the interrupt routine with interrupts ON or until both an RTI instruction and an RSI instruction (see below) have been executed.

A rising edge of KDN will occur if IR[0:6] and IR15X were all momentarily low and if : 1) a rising edge occurs on IR15X; or 2) one or more of the IR[0:6] goes high.

The interrupt state machine ignores edges on KDN (and does not latch a request for a new interrupt) while the CPU is in the interrupt routine. This allows software to scan the keyboard by setting each OR line high in succession and executing an A=IN or C=IN without latching the need for another interrupt. During this keyboard scanning, KDN may be toggling low and high based on the keys that are down and the OR line that is high. In order to avoid "spurious" interrupts, software must ensure that the KDN signal has been updated to match the current state of the keyboard-OR lines before exiting an interrupt routine where keyboard scanning has occurred. This is normally done by restoring all keyboard-OR lines to their normal state using OUT=C or OUT=CS prior to the RTI. This ensures that if any key is down the KDN signal will be HIGH when the RTI is executed.

5.4.5 **RSI Instruction**

The RSI (ReSet Interrupts) instruction resets the KDN interrupt state machine. After an RSI is executed the CPU will interrupt immediately if KDN is high and interrupt processing is enabled

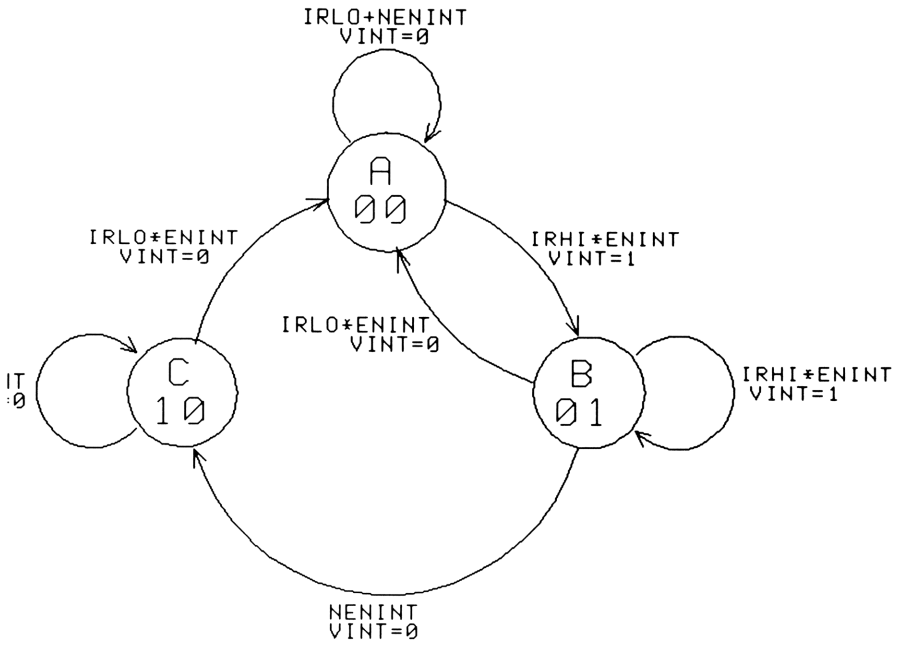
(INTON and RTI). If KDN is high and interrupt processing is disabled, the CPU will interrupt as soon as interrupt processing is enabled (INTON and/or RTI). This bypasses the need for KDN to go low before a new interrupt can be generated.

The RSI instruction should be used prior to RTI when keyboard scanning shows no keys are down. This will detect keys that are depressed between the time of the last scan and the RTI by causing an immediate reinterrupt.

5.4.6 **INTON/INTOFF Instructions**

The INTON and INTOFF instructions control the masking of KDN generated interrupts. If KDN goes high while interrupts are off it must remain high until INTON is executed to generate an interrupt.

Figure 5.2. Keyboard Interrupt State Machine



Notes:

1. State A is: Waiting for an IR line to be high while not in the interrupt routine.

2. State B is: An IR line went high. Wait for CPU to vector to 0000F or all IR line to return low.

3. State C is: Presently in the interrupt routine. Waiting for all IR lines to be low outside of the interrupt routine.

4. The RSI instruction resets State C to State A.

5. The INTON and INTOFF instructions do not affect the state machine. They merely control the masking of the VINT signal.

6. NENINT is true when you are in the interrupt routine and ENINT is true when you are outside the interrupt routine.

7. * means logic AND. + means logic OR.

8. VINT=1 requests the CPU execution unit to interrupt at the completion of the current instruction.

5.5 **SPECIAL FEATURES**

5.5.1 **CPU Reset**

A hardware reset performs the following CPU initialization:

- o Shuts down if running. Ignores all wakeup stimulus while NRST is low.
- o Reset PLA output control lines to zero (inactive) and hold PLA in precharge.
- o Enable interrupt processing.
- o INTON is asserted.
- o Input register interrupt circuitry (state machine) is reset.
- o All Hardware status bits are cleared.
- o The RSTK pointer (SP) is initialized to 0.

During reset the CPU output ports have the following characteristics:

NCDX	High impedance
NSTRX	High impedance
COUT[0:3]	Low
TRI	High
NBPC	High (inactive)
CREQ	Low
KPC	Low
IOCK	Low
NEKPOL	Low (High if HALT=1)
OUTB	Low
MBUS[0:3]	High impedance
INM	Low
TIRED	Low
SDTA	Low

5.5.2 **CPU Shutdown; Wakeup**

The CPU supports a low-power standby mode. The CPU state, including all CPU resident memory, is preserved during standby. To place the system in standby, a SHUTDN instruction is issued. The CPU then issues the SHUTDOWN bus command with TIRED=1 and on the next cycle shuts down internally. During CPU shutdown the CPU output ports have the reset conditions shown above.

The CPU may be brought out of standby mode in a number of ways if NRST=1 and HALTX=0. KDN, BIN and NCD (NINTX and NINT2X are mask options) are level sensitive wakeup sources. IR15X (ON) is an edge sensitive wakeup source. Wakeup stimulus is ignored when NRST=0 or when the CPU is already awake. Wakeups are latched if NRST=1 and the CPU is asleep. Wakeups are delayed until HALTX=0. Level sensitive wakeups must be active for a minimum interval to be recognized. Once awakened, the CPU will set CREQ high and wait for the high frequency oscillator to start and supply pulses on the HFO line. If a reset (NRST=0) occurred while the CPU was in standby then the PC is set to zero, hex mode arithmetic is asserted and a RESET bus command is issued. Otherwise the CPU status is unchanged and a NOP bus command is issued.

After this initial bus command (RESET or NOP) a LOAD PC command is issued and the CPU PC is loaded into the local PCs of all peripheral chips. At this point the standard instruction fetch sequence is initiated. If the CPU was awakened by IR15X (optionally: NINTX or NINT2X), or by KDN with interrupts ON and interrupt processing is enabled then an interrupt may occur prior to the first instruction fetch if the stimulus remains active for the minimum required interval. See also INTERRUPT SYSTEM.

If a SHUTDN instruction is executed while any of the level sensitive wakeup sources is active then the CPU will wakeup immediately as described above. A rising edge of IR15X just prior to CPU shutdown will not cause an immediate wakeup since wakeup sources are ignored until after shutdown.

The BIN signal causes a CPU wakeup for manufacturing burnin.

5.5.3 **Power Supply Handshake**

The 1LT8 IC includes a power supply circuit which is enabled during Light Sleep (CPU sleeping with the display on) or when running. When the system wakes up from the Deep Sleep state the power supply requires time to regulate VDD. The power supply handshake circuit provides this time before allowing the CPU to run. When the CPU wishes to wakeup it issues a wake request to the power supply; after the supply is up and running the power supply signals the CPU that it may now run.

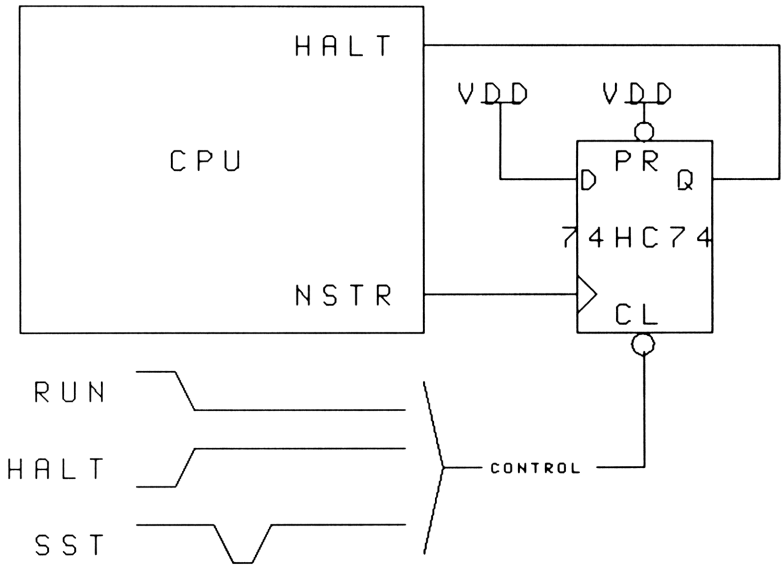
If the main TIMER is running the CPU can time-out its wake request. If the power supply can not get VDD up and running within 2 to 4 seconds then the wakeup request is aborted.

5.5.4 **HALT Function**

The 1LT8 CPU may have instruction execution halted by driving the HALTX pin to a logic one. The CPU will complete the current instruction before entering an idle state. In this state the CPU sets TRI and NBPC high and tristates NSTRX and NCDX. This idle state is not a low power state since CREQ remains high (active). HALTX is not latched internally and therefore must be held high until the idle condition has served its purpose and is no longer needed. The CPU ignores interrupts in this state. However, a pending interrupt will be acknowledged (prior to the first instruction fetch) when HALTX returns low.

A simple hardware single step may be implemented using HALT with a D flip-flop as shown in Figure 5.3.

Figure 5.3. CPU Single-Step Circuit



5.5.5 **WAIT Function**

A slow peripheral device can cause the CPU to wait during a CPU cycle by driving WAITX high. While WAITX is high the CPU will hold the current bus condition unchanged until WAITX returned low. In this way, WAITX allows the slow peripheral to extend the CPU cycle until data is transferred without degrading the overall bus rate.

5.5.6 **Input and Output Registers**

The CPU has two registers which correspond to I/O ports on the 1LT8 IC used for keyboard scanning. Like SACAJAWEA, 1LT8 implements only a portion of the Input and Output register lines supported by the CPU.

A 12-bit write only Output Register holds the data to be driven on the OR[0:8] output pins (on 1LT8, bit 11 is used for the beeper, bits 9 and 10 are not used). The Output Register pins have an asymmetrical drive scheme. This allows multiple keys to be

depressed at the same time without excessive power consumption and non-deterministic states. The Output register bits require a periodic precharge to drive the pads low if the corresponding Output Register data is zero. The precharge occurs during the last cycle of each CPU instruction.

A 16-bit read only Input Register provides data corresponding to the logic levels sensed on the 9 input lines IR[0:8] and IR15 (bits 9 through 14 are not used). ON (IR15) is dedicated for a hard-wired ON-key.

The Input Register can also generate wakeups and interrupts. IR15X is an edge triggered wakeup source to allow software to shutdown the system and save battery power in the event the ON-key is held down while the calculator is in storage (in a brief case for example). All other keys in the keyboard can be disabled by setting the output register low. See INTERRUPT SYSTEM for input register interrupt information.

5.5.7 **System Safeguards**

The CPU incorporates several additional features to help maintain integrity in typical systems.

5.5.7.1 **Hardware ON-Key**

The ON pin drives the CPU's IR15X signal and is a non-maskable interrupt intended for connection to the ON-key in the keyboard matrix. The ON-key switch connects ON to VDD. This feature provides a non-maskable interrupt to guarantee that keyboard intervention is never disabled.

5.5.7.2 **Bus Precharge and 00 Instruction**

Since the architecture of 1LT8 systems might include soft-configured add-on devices, the CPU provides three special features to provide recovery mechanisms in the event execution is attempted at addresses where no memory resides.

During each CPU cycle (prior to NSTRX going low) the CPU momentarily sets NBPC low signaling that the bus lines (D[0:3] and BUS[0:3] should be precharged low. If information is read from a non-existent device zeros will be read. The opcode for RTNSXM was purposely chosen to be 00 (HEX) to assure that a GOSUB to non-existent code would result in an immediate return, discernible by testing the XM hardware status bit.

Repeated returns are guaranteed to end up at address zero by a

special feature of the subroutine return stack as described in PROGRAM COUNTER AND RETURN STACK.

5.5.8 **Hardware Options**

The LEWIS, SACA, and 1LT8 CPUs contain three options that are set during design. IN the 1LT8 CPU the following options were selected.

-------OPTION----------		-BLOCK-	-CHOICE-
NINTX	causes wakeup - yes or no	POR	no
NINT2X	causes wakeup - yes or no	POR	yes
HALTX	delays wakeup - yes or no	POR	yes

5.6 **1LT8 CPU/1LK7 DIFFERENCES**

This section provides a description of the outward differences between the 1LT8 CPU and the 1LK7.

1. The 1LT8 CPU uses an on-board oscillator except in test mode.

2. Timing of the NCDX with respect to NSTRX was changed slightly to provide greater timing margins at 2MHz.

3. The timing of CPU bus reads and writes was changed to provide greater timing margins at 2MHz.

4. The input and output register operation has changed slightly since not all of the bits are used. When the CPU writes to unused output register bits the data is lost. When the CPU reads unused input register bits the data is always forced to logic zero.

5. DRI and NIF pins were eliminated.

6. The power supply interface circuitry has been changed to provide for a direct interface with the 1LT8 chip analog.

7. A SCAN path through the CPU PLA inputs and outputs has been added for improved testability. See chapter on TEST.

8. IR15X interrupts have changed. There is a mask option for 1LK7 compatibility. See section on INTERRUPT SYSTEM.

9. The following instructions were added to the instruction set:

MNEMONIC	OPCODE
LAHEX	8082tnn..n
BUSCB	8083
ABIT=0 n	8084n
ABIT=1 n	8085n
?ABIT=0 n	8086naa
?ABIT=1 n	8087naa
CBIT=0 n	8088n
CBIT=1 n	8089n
?CBIT=0 n	808Anaa
?CBIT=1 n	808Bnaa
BUSCD	808D
PC=(C(A))	808E
r=r+CON fs,n	818frn
r=r-CON fs,n	818frn
rSRB.F fs	819fr
r=ss.F fs	81Afrn
ss=r.F fs	81Afrn
rssEX.F fs	81Afrn
PC=A	81B2
PC=C	81B3
A=PC	81B4
C=PC	81B5
APCEX	81B6
CPCEX	81B7

10. The cycle times for the following instructions changed:

MNEMONIC	OPCODE	CYCLE COUNT
GOSUBL	8Ennnn	14
LCHEX	3tnn..n	3+t
CLRST	08	5
C=ST	09	5
ST=C	0A	5
CSTEX	0B	5

CHAPTER 6
SATURN BUS INTERFACE

The 1LT8 chip has its own Bus Interface that decodes commands and addresses from the 1LT8 CPU for the peripheral circuits. This interface contains 9 separate configuration registers and behaves like 6 separate devices on the bus.

Each of the five devices controlled by the Memory Controller except NCE[1] (the ROM) can be configured to fill an address space of from 4K to 1024K nibbles. In addition to the standard address configuration a size mask specifying the size of each device is provided.

6.1 Device IDs

The device IDs are listed below in the order of the internal daisy chain. The pri (priority) shown below determines which device is selected in the case of overlapping addresses. The device with the highest priority is selected.

DEVICE	ID	Pri	DESCRIPTION
NCE[1]	none	0	The ROM memory controller is selected by default; it has lowest priority and is hard configured to the entire address space.
CONTROL	xxx#9	5	Control register address, the size is 64 nibbles. This is the first device seen on power-up. #=xx01
SIZE2	xx003		RAM size mask.
NCE[2]	xx0F4	4	RAM address.
SIZE3	xx005		Plug in 1 size mask.
CE[1]	xx0F6	2	Plug in 1 address.
SIZE4	xx007		Plug in 2 size mask.
CE[2]	xx0F8	3	Plug in 2 address.
SIZE5	xx001		Extra size mask.

NCE[3]	xx0F2	1	Extra address. If DA19 is clear, this configuration can go to an additional ROM or RAM on the NCE[3] chip select. If DA19 is set, space reserved by this configuraton can be used by an external Saturn Bus device.

6.2 **Control Addressing**

ADDRESS	**DESCRIPTION**
00	Display Bit Offset and DON [DON OFF2 OFF1 OFF0]
01	Contrast Control [CON3 CON2 CON1 CON0]
02-03	Display Test [VDIG LID TRIM CON4] [LRT LRTD LRTC BIN]
04-07	CRC Register
08-09	Low Power Registers [LB2 LB1 LB0 VLBI] (read-only) [ELBI EVLBI GRST RST] (read/write)
0A	Mode Register (read-only)
0B-0C	Annunciator Control [LA4 LA3 LA2 LA1] [AON XTRA LA6 LA5]
0D	SERIAL Baud Rate [UCK BD2 BD1 BD0] (bit 3 is read-only)
0E-0F	Card Detect [ECDT RCDT SMP SWINT] (read/write) [P2W P1W P2C P1C] (read-only)
10	SERIAL I/O Control [SON ETBE ERBF ERBZ]
11	SERIAL Receive Control/Status [RX RER RBZ RBF] (bit 3 is read-only)
12	SERIAL Transmit Control/Status [BRK LPB TBZ TBF]
13	SERIAL Clear RER

	[writing anything clears RER bit]
14-15	SERIAL Receive Buffer Register [Reading this register clears RBF bit]
16-17	SERIAL Transmit Buffer Register [Writing this register sets TBF bit]
18-19	Service Request Register (read-only) [ISRQ TSRQ USRQ VSRQ] [KDN NINT2 NINT LSRQ]
1A	IR Control Register [IRI EIRU EIRI IRE] (bit 3 is read-only)
1C	Led Control Register [LED ELBE LBZ LBF]
1D	Led Buffer Register [0 0 0 LBO] (bits 1-3 read zero)
1E-1F	Scratch pad
20-24	Display Start Address (write only)
25-27	Display Line Offset (write only)
28-29	Display Line Counter and miscellaneous [LC3 LC2 LC1 LC0] [DA19 M32 LC5 LC4]
2E	TIMER1 Control [SRQ WKE INT XTRA]
2F	TIMER2 Control [SRQ WKE INT RUN]
30-34	Display Secondary Start Address (write only)
37	TIMER1
38-3F	TIMER2

The bus interface responds to all commands as was described in the chapter entitled "SYSTEM BUS DESCRIPTION".

CHAPTER 7
MEMORY INTERFACE

The purpose of the memory interface is to allow 1LT8 to drive commercial RAM and ROM ICs. Since commercial memory parts are byte-wide, this requires some careful interfacing to the nibble world of the Saturn bus. The same lines that go to the memory address are also used to scan the keyboard. The keyboard is connected through series resistors to the I/O lines while the memory is connected directly. This allows for commercial memory ICs to be connected to the 1LT8 chip with a minimum of added pads. On earlier models of the CPU the I/O lines were unidirectional and were labeled IR[0:15] and OR[0:11]. The IR15 line receives special treatment (non-maskable keyboard interrupt). OR0 was also unique on the earlier CPUs but this feature has been eliminated on the 1LT8 chip.

The memory controller has five chip enable lines, a write enable line (NWE), and eight memory data lines (MD[0:7]) that are not shared with the keyboard. The ON (IR15) pad is dedicated to the unmaskable interrupt input. The BEEP (OR[11]) output is a dedicated, high voltage output line that can drive a piezo-electric beeper.

7.1 **Configuration**

The system ROM memory controller (NCE[1]) is hard configured to the entire address space. The other memory controllers are soft configured. At power-up or after a RESET command only NCE[1] memory controller is configured. The other memory controllers are unconfigured. Each memory is configured with two configuration commands. The first is a mask which specifies the address bits to be used to generate the chip enable. The mask tells the IC which address bits it should care about when computing whether a particular address is within the address range of the controller. In this way the mask is used to specify the size of the device. The second configuration command specifies the device's address. For example, a 2K byte RAM at address 20000-20FFF would be configured FF000,20000. Overlapping address spaces are allowed; the higher priority chip selects take priority. The first memory controller (ROM) has no configuration registers but is the 'default' if no other devices are configured over it. The control register match takes priority over all the memory controllers. See Chapter 6 for the list of the controller ID's and priorities.

7.2 **Getting MA[18]**

A control bit, DA19, redefines NCE[3] as NOT MA[18]. This pin must be used as an address line if any device is larger then 256K bytes. For example, a 512K byte ROM would need this address signal. The signal is inverted because NCE[3] is active low and inverting the address output allows the cold start code to redefine the pin before it is needed. Note: Program the ROM with the address line inverted.

7.3 **Using a Saturn bus device**

When DA19 is set the address range defined by NCE[3]'s configuration registers maps to the external Saturn bus. This is normally the only way to access an external device over the Saturn bus.

The only other way to access an external device is by using JANET mode. This mode is designed for firmware development and maps the ROM memory controller to the external Saturn bus. This allows system ROM to be simulated with Saturn bus RAM boxes and debugger systems.

7.4 **Write Disable**

NCE[1], the ROM chip enable will not generate write cycles, it is read only. Write bus commands to the ROM are ignored, no waits are generated and no data is written. In addition, signals from the Card Detect circuit can disable writes to CE[1] and CE[2].

To increase shelf battery life the ROM is powered by VCO, which is low in deep sleep and coma modes. NCE[1] is also powered by VCO to prevent forward biasing any input diodes on the ROM.

7.5 **Memory Operations**

The memory controller can control five memory devices of up to 512 KByte. The names of the address pads correspond to the keyboard function. The memory controller pads are:

PAD	CONTROLLER USAGE	KEYBOARD USAGE
NCE[1:3]	chip enables	none
CE[1:2]	card enables	none
NWE	read/write line	none
MA[0:8]	memory address lines	IR[0:8]
MA[9:17]	memory address lines	OR[0:8]
MD[0:7]	memory data lines	none
ON	none	IR15
BEEP	none	OR11

The memory controller does not access the memory for every nibble operation but does byte operations on pairs of buffered nibbles from the Saturn bus. When the controller is writing to a RAM it will use a read/modify/write cycle as necessary to allow the memory to appear to the system as a nibble wide device.

7.6 **Timing Diagrams**

The timing diagrams of the various memory operations follow:

Figure 7.1. Memory Read Cycle

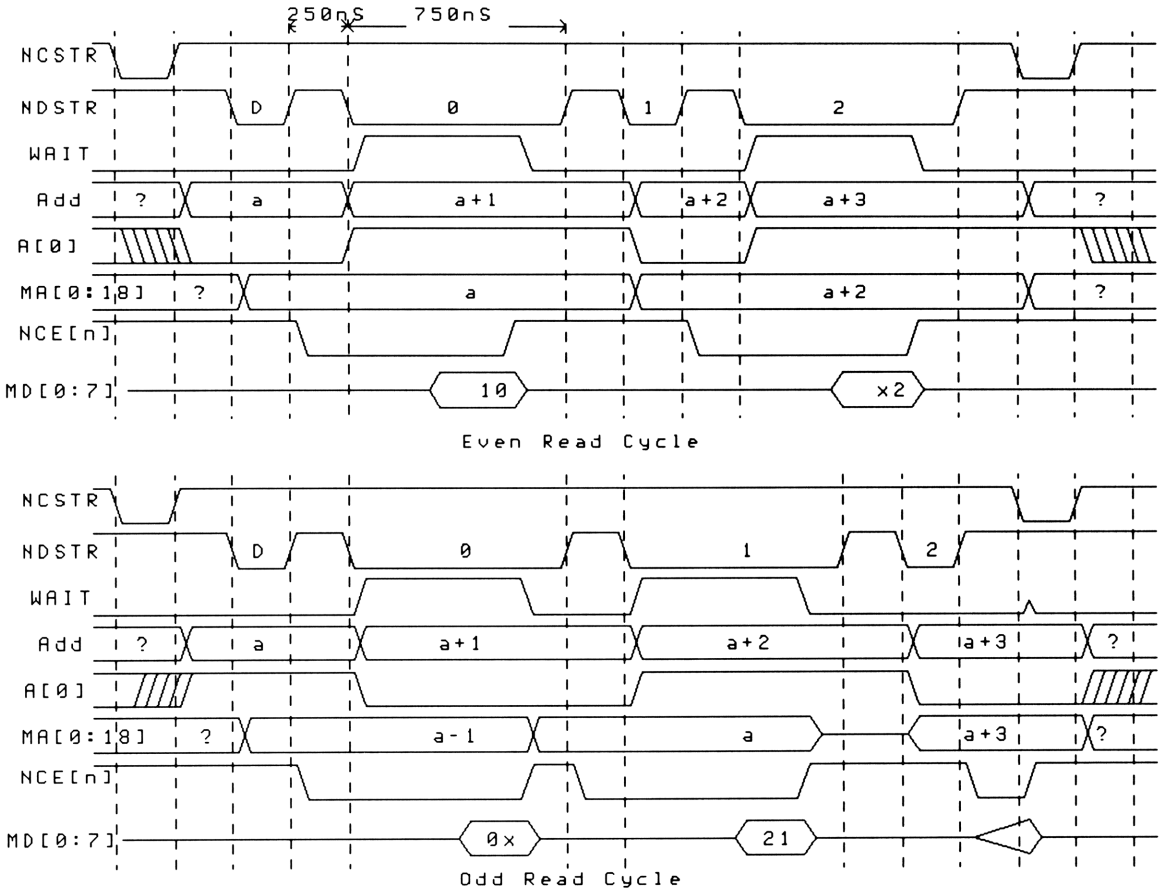
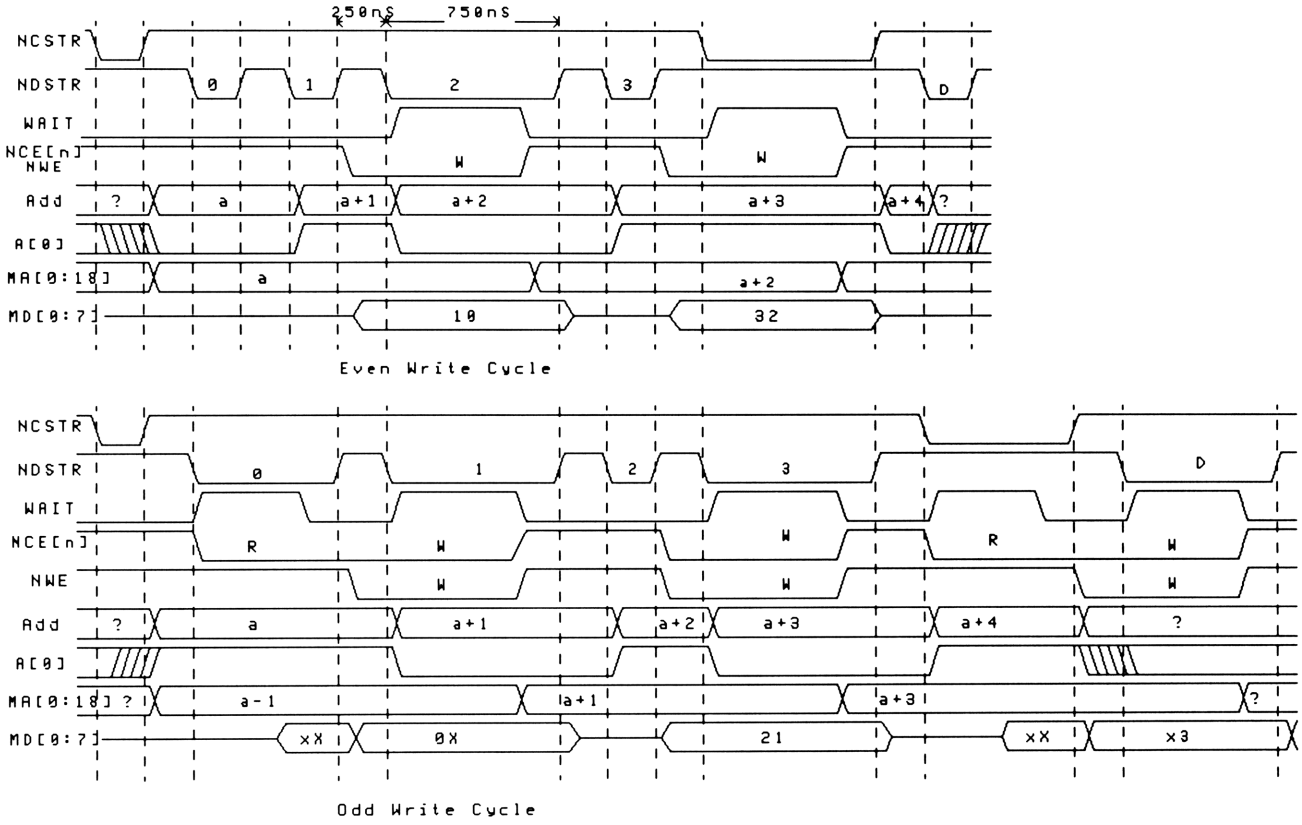


Figure 7.2. Memory Write Cycle



7.7 **Keyboard Operation**

The keyboard uses the common lines at a lower priority than the memory usage. Keyboard polling occurs asynchronously every millisecond and when the CPU initiates a read of the input register. In either case, the Memory Controller will assert WAIT to stop the CPU while the keyboard is polled.

Updates made by the CPU to the output register will have no external effect on the OR pads until the next keyboard poll. The BEEP pad (OR11) is not used for memory control and when the CPU outputs to this bit of the output register the external effect is immediate.

ON (IR15) is also not used for memory control. Pulling ON high will generate a non-maskable CPU interrupt.

7.8 **Special Considerations**

There is a bug in the interaction of the memory and keyboard controlling functions. This bug manifests itself only when an x=IN instruction is executed starting on an odd address of memory that is being controlled by the memory controller. When this occurs the first nibble of the following instruction read by the CPU will be corrupted. Therefore all x=IN instructions must be stored beginning at an even address of the memory controller.

The asynchronous key polls are triggered each millisecond by bit 2 of the timer. Therefore the timer R/S bit must be set allowing the timer to run in order for keyboard polling to occur.

A rising edge of ON wakes up the CPU. But due to the memory controller wait and the time required for the frequency multiplier to come up to speed, it is possible that ON will have gone away before software can look for interrupt sources.

A chip cannot reconfigure itself. If the ROM must be configured differently the configuration code must be run out of a different chip, perhaps by copying the code into RAM.

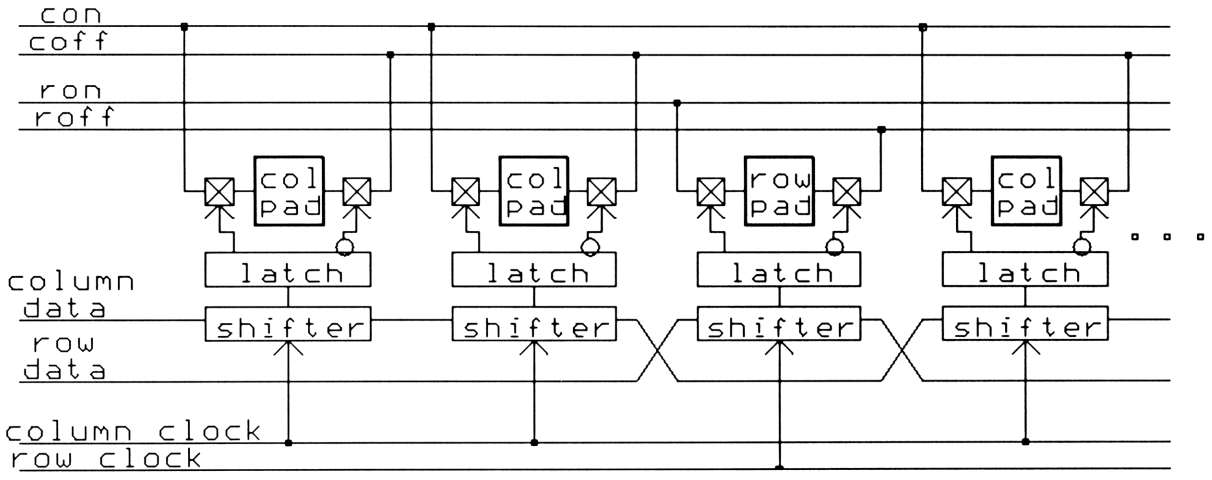
When the plug-in cards are powered down all inputs must be tied to ground. Clarke handles this by forcing all the signals low; including the OR signals. Therefore deep sleep keyboard wakeups are only possible with the ON key.

CHAPTER 8
LCD DRIVER

1LT8 contains 4 column and 64 ROW drivers designed to drive a LCD display. Sixty-four to 128 additional column drivers may be added externally.

Each LCD driver consists of two high voltage (up to 9V) transmission gates and a latch. The value in the latch selects which transmission gate will be on. Either an ON (voltage select) or an OFF (voltage non-select) signal will be driven out. The display analog circuitry generates the necessary ON and OFF signals to operate the display, outputting an ON signal results in a pixel being turned on. Sixty-four of the LCD drivers are row drivers. They are given the Ron and Roff levels necessary to drive the row lines of the LCD. The columns are given the Con and Coff levels that they need. The latch data is defined by the display memory. The data is loaded into the latches in parallel after being shifted from the display memory by a shift register. There are separate data streams for rows and columns.

Figure 8.1. LCD Drivers.



8.1 **Display Memory**

The display column data is stored in RAM. The display RAM contains the data for 8448 pixels (132*64) or 2112 nibbles. However, the display data is accessed in bytes. Therefore the actual minimum size of the display memory is (136*64) or 1088 bytes. The first bit of display memory maps to the upper left hand corner of the display, the second maps to the next COLUMN, the 137th bit may start the second ROW. Note that this organization is different from LEWIS.

The LCD display is refreshed at 64Hz. The refresh sequences through the display memory starting with the top ROW of the display and continues through each of the other ROWS in series and then begins again. Each ROW of data is driven for 1/(64 rows * 64 Hz) or 244 uS. The data for the ROW is read from memory while the previous ROW is being driven. This provides a 244 uS window to retrieve the necessary data. When the old ROW begins to drive, the display control logic issues a request for the next ROW's data. The CPU is halted for 22-23 uS every 244 uS to read the data. The display controller requires 116 uS to prepare the data; which leaves 244-23-116 = 105 uS of time during which the request for data could be held off without affecting the display. This time is used for the HALT latency period or other delay.

The display data access is postponed under the following conditions:

1. Waiting for the CPU to HALT.

2. Waiting for a keyscan to complete. During keyscans the CPU can not be halted.

3. If an external halt is applied and a NOP command is not active.

4. If the RAM is unconfigured.

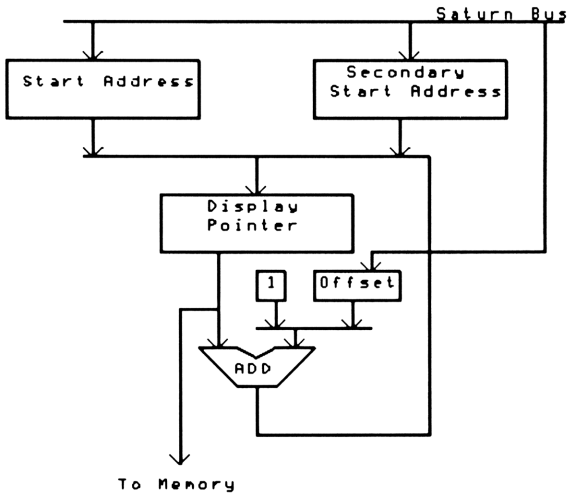
Postponing the display access for more then 105uS will cause the display to receive incorrect data. The display will recover on the next refresh cycle which is okay for the debugger system but not too great in the product.

8.2 **Display Pointers**

The data for the display is pointed to by the display pointer. This pointer is initialized at the beginning of each refresh cycle and incremented as each byte of data is read. In addition, a offset value may be added to the pointer at the end of each ROW

and it may be reloaded on a specified ROW.

Figure 8.2. Pointer Hardware Block Diagram.



8.2.1 **Display Start Address**

The display start address register points to start of memory where the pixel map is contained. This address must be within a valid RAM or ROM. The least significant bit of the start address register is ignored; all display accesses act on byte boundaries. At the beginning of each refresh cycle the display pointer is loaded from the display start address.

8.2.2 **Display Line Offset**

The three nibble offset register is added to the display pointer after each ROW has been read. This feature allows display memory to be built larger then the physical display and real time panning of the display simply by changing the display start address. Again, the least significant bit of this register is ignored; however, the MSB is sign extended allowing the offset register to actually subtract from the display pointer. This feature saves memory for a product which uses only one external column driver IC.

8.2.3 **Display Line Counter**

The display line counter may be used to specify a ROW which will reload the display pointer with a new absolute address. This feature can be used to point to a different area in memory for the product soft keys, etc. Once the display pointer has been reloaded it will be incremented as normal but all offsets are disabled. The display pointer will be reloaded with the normal display start address and offset re-enabled at the beginning of the next refresh cycle. The ROWS affected by this register are from (VALUE+2) through the bottom of the display. Writing to this register stores its value into a holding register which is loaded into the counter at the beginning of each refresh cycle. The counter is decremented after each ROW of display data is read. Reading this register returns the present value of the counter not the value in the holding register. The two MSBs of this register control other functions:

Bit	Reset Value	Description
[2]	0	M32 - Multiplex 32 way.
[3]	0	DA19 - Drive A[19] changes the definition of NCE[3] to NOT MA[18]

8.2.4 **Display Secondary Start Address**

The display secondary start address register specifies a second address for the display memory. This address is loaded into the display pointer for the ROW specified by the display line counter, above. The load takes place on the refresh cycle when the counter equals zero. The load of the Display Start Address takes priority, therefore the secondary start address cannot be loaded for the first ROW.

8.3 Control Registers

There are four control registers for the display:

8.3.1 Display Bit Offset and DON

A three bit value specifies a bit offset into the display memory. This allows the display to be located on any bit location in memory. Each byte of display memory is read normally but is shifted by the bit offset amount while being prepared for the display. Note that if the offset is greater then three another byte of data is required for each ROW; the line offset may need to be adjusted in this case. The MSB of this register is DON which turns the display on.

8.3.2 Contrast Control

A five bit register controls the contrast of the display. The higher the value the darker the display. This register must be initialized to a reasonable value.

8.3.3 Display Test

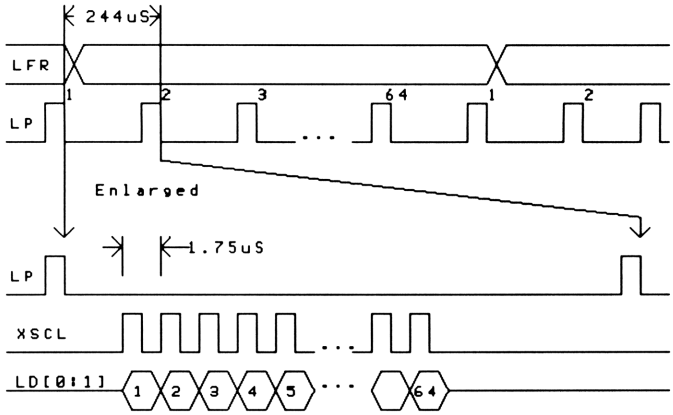
Seven bits control the mode of the display during testing. During normal display operation these bits are zero. See the chapter entitled "TESTING" for a description of their operation and use.

8.4 **External Column Driver Interface**

The 1LT8 IC includes an interface to external column driver ICs (128 columns). Display data is shifted out to the column drivers. This interface consists of the following output signals:

Signal	**Description**
LD[0:1]	Data, two bits of data are output at a time. A total of 128 bits are output for each ROW. (2bits*64shifts) LD[0] should be tied to the shift register leading to the left most column; LD[1] leads to the next column.
XSCL	The shift clock, data should be latched on the falling edge of XSCL.
LP	Latch Pulse, the data should be loaded into the output driver on the falling edge of the latch pulse.
LFR	Frame clock, specifies the polarity of the outputs.
VNCS	The analog column non-select voltage. If LFR is high VNCS=VSSH+VB; if LFR is low VNCS=VH-VB.

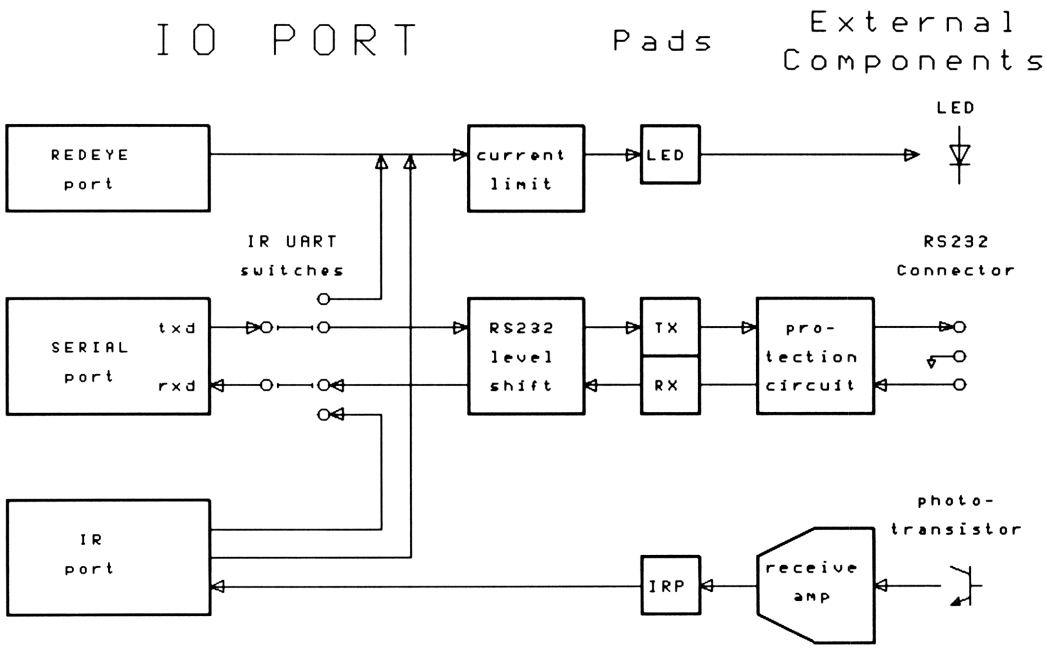
Figure 8.3. External Column Driver Interface Waveforms



CHAPTER 9
I/O PORTS

1LT8 has three built-in input/output ports for communication between systems over wires or without wires using infrared light. The SERIAL port is a full-duplex 1200-9600 baud UART combined with RS232 signal level shifting and uses the RX and TX pins for wired communication. The REDEYE port is an infrared REDEYE printer port that uses the LED pin and an external LED for wireless printer communcation. The IR port uses an IR input pin (IRI) and an external infrared receiver circuit for input, and the LED pin and an external LED for output. This port may be used to implement wireless I/O using infrared light.

Figure 9.4. IO PORT



The SERIAL and IR ports may be combined in IR UART mode. This mode may be used for half-duplex 1200-2400 baud wireless communication between systems.

In each port, software has full access to the actual I/O pins (RX, TX, LED, and IRI) and thus may use other I/O formats by providing a bit-level driver. This accessibility also facilitates testing. Also, each port provides an interrupt capability that allows communication routines to run in the background.

9.1 **SERIAL PORT**

The serial I/O port allows full-duplex communication between systems at 1200-9600 baud over wires with RS232 level signals. The serial i/o port is divided into separate receive and transmit sections which share a baud-rate generator and an interrupt mechanism. Both the receiver and transmitter sections are double-buffered to maximize the thru-put of the serial channel. The receiver portion consists of the receive pin (RX), the receiver control/status register (RCS), the receive buffer register (RBR), and the receive shift (accumulator) register. The transmitter consists of the transmit pin (TX), the transmitter control/status register (TCS), the transmitter buffer register (TBR), and the transmit shift register. The Baud-rate generator is controlled by the Baud rate register (BAU). An I/O control register (IOC) controls interrupts and wake-ups and enables the serial port.

9.1.1 **Serial Format**

A frame of serial data consists of a start-bit, eight data-bits, and at least one stop-bit. The start bit is equivalent to a zero data-bit and a stop bit is equivalent to a one data-bit. The data bits are sent least-significant-bit first. 1LT8's transmitter sends 2-3/16 stop bits with every frame. 1LT8's receiver requires at least 3/4 stop bit for proper frame synchronization.

A mark condition or one data-bit on the serial line corresponds to a negative voltage. A space condition or zero data-bit corresponds to a positive voltage. An idle condition corresponds to zero voltage and is a low power state. The line is normally held in the mark or idle condition. When data is to be transmitted it must start in the mark condition. When the line is to be returned to the idle condition it must pass through the mark condition first. A break condition is when the line is held in the space condition for at least one entire frame. A break should not start in the middle of a character.

9.1.2 **RBR - Receive Buffer Register**

The receive buffer register is an eight bit read-write register. The register is loaded from the receive accumulator after a valid start-bit, eight data bits, and a stop-bit have been received.

9.1.3 **RCS - Receive Control/Status Register**

The most significant bit of this four-bit register is read-only. Bits 0-2 are read/write. Writing bit 3 has no effect.

Bit	Reset	Description
[0]	0	RBF - Receive Buffer Full. This bit is set when the receive buffer register is loaded from the receiver shift register after a complete frame has been received. It is automaticaly cleared by reading the receive buffer register.
[1]	0	RBZ - Receiver Busy. This bit is automatically set/cleared at the beginning/end of a receive frame to indicate that the receive shift register is busy accumulating a frame.
[2]	0	RER - Receiver Error. This bit is automatically set to indicate that a framing error or an input overrun have occurred. A framing error is defined as an illegal (0) stop bit. Input overrun occurs if RBF is set when a new frame is transferred to the receive buffer register. This bit can be cleared by writing to the Clear RER address.
[3]	X	RX - RX pin state. This read-only bit reflects the state of the RX pin. It is high to indicate a marking condition (low or negative voltage) or an idle condition (zero voltage) on the RX pin. It is low to indicate a spacing condition (high or positive voltage) on the RX pin.

9.1.4 **CRER - Clear RER address**

Writing any value to this address causes RER to be cleared. The return value from reading this location is undefined.

9.1.5 **TBR - Transmit Buffer Register**

This eight bit read-write register contains the data to be transmitted. Writing to this register will automatically start the transmitter.

9.1.6 **TCS - Transmitter Control/Status Register**

All four bits of this register are read-write.

Bit	Reset	Description
[0]	0	TBF - Transmit Buffer Full. This bit indicates that the Transmit Buffer Register is not available for writing. It is automatically set by writing data to the Transmit Buffer Register. When the transmitter is not busy, the data in the TBR will be transferred to the Transmit Shift Register and this bit will be cleared.
[1]	0	TBZ - Transmitter Busy. This bit is automatically set and cleared to indicate that the Transmitter Shift register is busy shifting data to the TX pin.
[2]	0	LPB - Loopback. This bit causes a loop-back mode where the transmit pin is connected on-chip to the receive pin. Data that is transmitted is automatically circulated back to the receiver.
[3]	0	BRK - Break. When set, this bit forces the TX pin to go to the space condition. It will abort any transmission in progress.

9.1.7 **IOC - Serial I/O Control Register**

This four-bit read-write register is used to control interrupts from the serial I/O port. If an enable bit is set and its interrupting condition is met, a UART service request (USRQ) is generated.

Bit	Reset	Description
[0]	0	ERBZ - Interrupt on Receiver Busy (RBZ set).
[1]	0	ERBF - Interrupt on Receive Buffer Full (RBF set).
[2]	0	ETBE - Interrupt on Transmitter buffer Empty (TBF clear).
[3]	0	SON - Serial On. Must be set for any serial I/O operation to function. If clear, the RCS, RBR, TCS, TBR, and IOC registers (except SON bit) are cleared, the Receive and Transmit Shift registers and state-machines are cleared, and the TX and RX pins are set to a low-current state. The TX pin will be connected to Signal Ground (VDD) to set an idle condition. The BAU register is not cleared by SON=0 to allow changing baud rates while the receiver and transmitter are disabled.

9.1.8 **BAU - Baud Rate Register/Generator**

This four-bit register is used to control the baud-rate generator. Bits 0-2 are read-write and control the baud rate according to the table below. They default to zeros at reset.

Bits [0-2]	**Baud Rate**
000	1200
001	1920
010	2400
011	3840
100	4800
101	7680
110	9600
111	15360

The baud-rate generator output is a rising-edge clock that is 16-times the selected baud-rate. Bit 3 is a read-only bit that returns the current state of this clock for testing. Writing to bit 3 has no effect. To prevent glitching the UART's state machines, baud rate changes should be done with SON=0.

The values in the table depend on a crystal oscillator frequency of 32768 Hz as follows:

The crystal frequency is multiplied by the VCO circuitry:
    32768 x15 x16 /8 = 983040 Hz
This is divided by 1.6 and by 4 for multiples of 1200 baud:
    983040 /4 /1.6 = 153600 Hz
This is divided by 1, 2, 4, or 8 and then by 16:
    153600 /1 /16 = 9600 Hz
    153600 /2 /16 = 4800 Hz
    153600 /4 /16 = 2400 Hz
    153600 /8 /16 = 1200 Hz

9.1.9 **Receiver Operation**

With SON zero, the receive voltage level shifter is turned off to save power. The RX bit in RCS is one to show an idle line condition. The other bits in RCS, the RBR, the Receive Shift Register, the Receive State Machine (RSM), the Receive Clock Generator, and the Receive Start Bit Filter are all held clear. When SON is set, the receive level shifter is turned on and the receiver is ready to receive frames.

The RX pin goes to the space condition to indicate the start of a frame. The start-bit filter sees this and starts the receive clock. If the RX pin remains in space for at least half of a

bit-time, the start-bit is considered valid and the state machine is enabled. If the RX pin returns to mark condition before the half-bit timer expires, the start-bit filter stops the receive clock and aborts the frame.

If the state machine receives a start-bit-received signal from the start-bit filter, it initializes the receiver shift register and sets the RBZ bit in RCS. If the ERBZ bit in IOC is set, this causes a USRQ interrupt. Then the state machine enables the receive shift register and waits for the shift register to indicate completion.

All bit timing is relative to the leading-edge of the start bit. The resolution of this timing is 16 times the baud-rate. The data on the RX pin is shifted into the shift register at the center of the bit time. When the shift register has shifted in a start-bit, eight data-bits, and a stop bit (9-1/2 bit times from the leading edge of the start-bit), it indicates completion to the state machine.

The state machine then transfers the contents of the shift register to the buffer register (RBR), sets the receive buffer full flag (RBF in RCS), clears RBZ, and restores the shift register and start bit filter. If the RBF flag was already set, the RER flag is set to indicate overrun. If the stop bit was not a one (mark condition), the RER flag is set to indicate a framing error. Then the state machine returns to its idle state of waiting for a valid start bit.

If the ERBF bit in IOC is set, the RBF flag will cause a USRQ interrupt. While software is responding to the RBF condition, another incoming frame may start processing. Software has 9-1/2 bit-times of the new frame to read the old frame out of RBR. The RBF bit in RCS is automatically cleared when RBR is read.

```
Frame length:
    1 start-bit + 8 data-bits + 1/2 stop-bit
    + 4/16 bit for the state machine
              = 9.75 bits.
Maximum thru-put:
    Baud-rate / 9.75 = 985 cps at 9600 baud.
                     = 123 cps at 1200 baud.
```

The RER bit in RCS is used to indicate a framing error or overrun condition. It can be cleared by writing to the RCS register. However, since writing RCS may cause a receive frame to be missed (by clearing RBZ or RBF just as they are being set), another method of clearing RER is provided. Writing to the CRER (clear RER) address will clear RER without changing the other RCS status bits.

A break condition on the line will be interpreted as a valid start-bit followed by eight zero data-bits and at least one zero stop-bit. Thus a break condition will result in a null byte in RBR and setting RER and RBF. The start bit filter will prevent an extended break condition from being interpreted as additional characters until the line returns to the mark condition.

The RX bit in RCS provides low-level access to the RX pin for testing and for software generated I/O formats.

9.1.10 **Transmitter Operation**

With SON zero, the transmit voltage level shifter is turned off to save power. The TX pin is shorted to signal ground (VDD) to send a line idle condition. With SON zero, the TCS, TBR, the transmit shift register, the transmit state machine, and the transmit clock generator are all held clear. When SON is set, the transmit level shifter is enabled and the TX pin goes to the mark condition.

Software initiates a transmission by writing to the TBR. This automatically sets the TBF bit in TCS. The state machine transfers two stop bits, the start bit, and the contents of TBR into the shift register. Then it sets TBZ and clears TBF in TCS. Then it enables the shift register and the transmit clock generator and waits for the shift register to indicate completion.

When the shift register has shifted out 11 bits it indicates completion and the state machine returns to the idle condition of waiting for TBF. If the transmit buffer is empty (TBF clear) the state machine will clear TBZ as it returns to idle.

If the ETBE bit in IOC is set, the transmit buffer empty condition (TBF clear) will cause a USRQ interrupt. Software has 11 bit-times to write a new byte to TBR while the old frame is being shifted out. If software takes longer than 11 bit-times, the maximum thru-put will not be achieved.

After the last byte is written to TBR, software must wait for the serial port to complete transmission. The TBF flag will be cleared as the last byte is transferred to the shift register. Then the TBZ flag will be cleared when the last byte is shifted out. Software should wait at least an additional two bit-times before clearing SON to generate stop bits.

```
Frame length:
    2 stop-bits + 1 start-bit + 8 data-bits
    + 3/16 bit for the state machine
              = 11.375 bits
Maximum thru-put:
    Baud-rate / 11.375 = 844 cps at 9600 baud.
                       = 105 cps at 1200 baud.
```

The LPB (loopback) bit in TCS is provided for software testing. With LPB and SON set, the TX pin is shorted to the RX pin. Any frames transmitted in this mode will be routed back to the receiver. Any data transmitted to the receiver in LPB mode is also broadcast on the serial lines.

The BRK (break) bit in TCS allows for software control of the TX pin. When BRK and SON are set, the TX pin is set to the space condition. This can be used to generating break characters, for transmitting software defined I/O formats, or for testing. Software must time the length of break characters.

When BRK is set, it will abort any transmit frames in progress by clearing the state machine and the shift register. TBF and TBZ may be left undefined by this operation and thus should be written to zero when BRK is written. When BRK is set, it will start the transmit clock. When BRK is cleared, the break condition will continue for up to one bit-time, and then the transmit clock will be turned off.

9.2 **REDEYE (LED) PORT**

The REDEYE port consists of the Led Control Register (LCR), the Led Buffer register (LBR), the REDEYE formatter, and the LED pin (LED). The LED pin has an open drain device and thus may be driven low or tristated only. When driven low the drain current is somewhat regulated by a feedback circuit. The LCR and LBR registers form a double buffered handshake mechanism that allow automatic REDEYE half-bit formatting and pacing. An interrupt mechanism is provided to indicate completion of each half-bit.

9.2.1 **REDEYE format**

The REDEYE printer requires 15-bit frames of a very precise format. Each bit of the frame consists of two half-bits. The duration of each half-bit is 14 cycles of the 32768 kHz crystal oscillator. The half-bit is considered to be "on" if the LED is pulsed 6-8 times (out of the 14 possible) at the 32768 kHz rate. 1LT8's REDEYE port uses 8 pulses. The format of a complete REDEYE frame is shown below:

Start-bits	Three half-bits "on-on-on".
Hamming-bits	Four pairs of half-bits.
Data-bits	Eight pairs of half-bits. Each of the four hamming and eight data bits are encoded with two half-bits. A "one" data or hamming bit is encoded by "on-off" and a zero is encoded by "off-on".
Stop-bits	Three half-bits "off-off-off". This is the minimum idle time required between frames.

9.2.2 **LCR - Led Control Register**

This four-bit read-write register is used to pace the 14 cycle half-bits.

Bit	**Reset**	**Description**
[0]	0	LBF - Led Buffer Full. This bit is used to indicate that the contents of the Led Buffer Register have not yet been transmitted and should not be written at this time. Writing to the LBR automatically sets this bit. This bit is cleared when the LBR is transferred to the REDEYE formatter.
[1]	0	LBZ - Led Port Busy. Indicates that the REDEYE formatter is busy with a previous bit.
[2]	0	ELBE - Enable Interrupt on Led Buffer Empty (LBF clear). If this bit is set and LBF is clear, a Led Service Request (LSRQ) will occur.
[3]	0	LED - Turn on LED. When this bit is set, the LED pin will go low turning on the LED.

9.2.3 **LBR - Led Buffer Register**

Bit 0 of this four-bit register is read-write and is cleared at reset. Bits 1-3 read as zeros and have no effect when written. Bit 0 contains the half-bit to be transmitted in REDEYE format. Write a one to send an "on" half-bit or write a zero to send an "off" half-bit.

9.2.4 **Operation**

The LCS and LBR registers are cleared at reset. The REDEYE port also uses a state machine and a formatter bit which are cleared at reset. Software initiates a half-bit transmission by writing a

bit to the LBR. This automatically sets the LBF and LBZ flags in LCS and starts the state machine. The state machine transfers the bit from LBR into the formatter and clears LBF. If ELBE is set, this will cause a LSRQ interrupt indicating that it is safe to write the next half-bit to LBR. The state machine then times the half-bit for 14 counts of the 32768 Hz crystal oscillator. If the bit in the formatter is a one, the LED is pulsed for the last eight of the 14 counts. Otherwise the LED is left off. If after the 14 counts LBF is clear, the state machine will clear LBZ and return to its idle state of waiting for LBF. Otherwise it will leave LBZ set and immediately transfer the next half-bit and start timing it.

When LBF is clear and ELBE is set, a LSRQ interrupt will occur. When the state machine clears LBF, software has 13 counts of the oscillator to write the next bit to LBR. Otherwise the length of the half-bits will not be correct.

```
Thru-put:
      32768 /14 = 2340.6 baud (half-bits/sec)
      32768 /28 = 1170.3 bps (bits/sec)
REDEYE Frame Length:
      1.5 start + 4 Hamming + 8 data + 1.5 stop = 15 bits
REDEYE Thru-put:
      1170.3 /15 = 78.02 cps
```

The LED bit in LCS is provided for software generated IR formats. This bit is OR-ed with the output of the REDEYE formatter, therefore, IR formats may not be used simultaneously with the REDEYE formatter.

9.2.5 **Special Considerations**

Due to LED current limitations, the LED output driver duty cycle must be limited to a time average of 20%. The duty cycle is automatically limited to 1/2 x 8/14 or 28.6% by the REDEYE formatter. The format of a full REDEYE frame yields a duty-cycle of only 14.3%. If a different format is used (by using the LED bit) software must limit the duty-cycle.

9.3 **IR PORT**

The IR port allows half-duplex communication between systems at 1200-2400 baud using infrared pulses of light instead of wires. It provides the IRC register and the IRI pin for IR input. The LED bit of the REDEYE port can be used for IR output. All bit formating and timing must be done by software. An interrupt mechanism is provided for completeness.

9.3.1 **IR format**

The format for IR transmission is similar to Serial transmission except that a pulse of infrared light of 50 usec duration (nominal) is used to transmit a zero-bit. The absence of a pulse indicates a one-bit or idle condition.

9.3.2 **IRC - IR Control Register**

Bits 0-2 of this four-bit register are read/write. Bit 3 is read-only and has no effect when written.

Bit	Reset	Description
[0]	0	IRE - IR Event. This bit is set by a logic low voltage on the IRI pin. It is set to indicate that an IR event has occurred.
[1]	0	EIRI - Enable IR Interrupt. If this bit and IRE are set an ISRQ (IR Service Request) is generated.
[2]	0	EIRU - Enable IR UART mode. If this bit is set, IR UART mode is enabled. See the section on IR UART mode for details.
[3]	X	IRI - IR Input. This read-only bit reflects the state of the IRI pin. It is true if the IRI pin is low (IR light is being received).

9.3.3 **Operation**

The IRC register is cleared at reset. When a pulse of infrared light is received, an external circuit should pulse the IRI pin low. This event is indicated by the IRI bit and is latched into the IRE bit. If EIRI is set, the IRE bit will cause an ISRQ interrupt. Software should then write the IRC register to clear the IRE bit.

9.4 **IR UART MODE**

When the EIRU bit in IRC is set, IR UART mode is enabled. IR UART mode uses the formating and pacing features of the SERIAL port and adds the additional formating for IR pulses. It uses the LED pin for IR output and the IRI pin and IRC register for IR input.

9.4.1 **IR UART Transmitter Operation**

SON must be set for the transmitter to work. The output of the UART is modulated and sent to the LED pin instead of the TX pin. The TX pin will be powered-up but will be held in the mark condition. The TCS and TBR registers will operate as in the SERIAL port. LPB will still short TX to RX but will also disable the IRI pin and route the transmitter output into the IRE and IRI bits of IRC. BRK will not be sent to the TX pin but will instead generate a series of IR pulses. The BAU register still selects the baud-rate but also selects the IR pulse duration as follows:

Bits [0-2]	**Baud Rate - Pulse Duration**
000	1200 - 52.08 usec
001	1920 - 32.55
010	2400 - 52.08
011	3840 - 32.55
100	4800 - 26.04
101	7680 - 16.28
110	9600 - 26.04
111	15360 - 16.28

9.4.2 **IR UART Receiver Operation**

SON must be set for the receiver to work. The RX pin and the receiver voltage level translator will be powered up but it will be ignored by the UART. Instead, IR pulses that have been latched by the IRE bit in IRC will be routed into the UART.

The receiver starts by waiting for a valid start-bit. An incoming pulse of light will set the IRE bit which will start the receiver clock generator. The clock generator will shift IRE into the shift register after half of a bit-time and then clear IRE for the next bit. Because this method stretches the pulses, any pulse of light that is long enough to set IRE will be considered a valid start-bit.

The remaining bits are shifted in in the same fashion. As in the SERIAL port, all bit timing is relative to the leading edge of the start bit and has 1/16-th bit resolution.

In loopback mode (LPB in TCS is set), the IRI pin is ignored and the modulated output of the transmitter is routed into the IRE bit. Any transmit data routed to the receiver in loopback mode will also be broadcast on the LED. Any data sent to the LED through the LED bit in LCS or through the REDEYE port will not be routed to the receiver in loopback mode.

Due to the possibility that reflections from the transmitter may be received and interpreted by the receiver, the receiver data should be ignored.

CHAPTER 10
TIMER

The TIMER consists of two read/write counters. TIMER1 is a 4-bit counter that is decremented 16 times per second. TIMER2 is a 32-bit counter that is decremented 8192 times per second. A crystal controlled oscillator generates the timer clock.

10.1 **Timer Control**

A TIMER Control register is associated with each timer. The bits in these two control registers are defined as follows:

Bit	Reset Value	Description
[0]	0	RUN - Active only on the TIMER2 control nibble. If set, both timers will decrement. The timers must be allowed to run in order for automatic keyboard polls to occur. Turning off the timers will save a small amount of power.
[1]	0	INT - If the corresponding timer's MSB is a one AND INT is set AND the CPU is awake THEN NINT2 is pulled low.
[2]	0	WAKE - If the corresponding timer's MSB is a one AND WAKE is set AND the CPU is asleep then NCD is pulled low. This bit is cleared on the first data strobe after a wakeup.
[3]	0	SREQ - Service is requested. This bit is set whenever the corresponding timer's MSB is a one AND either INT or WAKE is set. It is cleared when the MSB of the timer is a zero.

10.2 **Special Considerations**

The clock for TIMER1 is derived from bit 8 of TIMER2. Therefore TIMER2 has to be enabled for TIMER1 to run. Also, writing to bit 8 of TIMER2 can cause TIMER1 to decrement when it shouldn't or miss a decrement.

CHAPTER 11
ANNUNCIATORS

The LA[1:6] and LN pins may be used to directly drive up to 6 LCD segments with no multiplexing. Individual control for these segments is contained in the Annunciator Control Register. When the AON and TRUN bits are set, the annunciators are turned on. Annunciators may be driven while the rest of the display is off.

A segment is off when the corresponding LA pin is at the same logic level as the LN pin. A segment is on when its LA pin is as the opposite logic level as the LN pin. When the annunciators are on, the LN pin is toggled at 128 Hz to prevent damage to the LCD. When the annunciators are off, the LA[1:6] and LN pins go to the logic low level.

All 8 bits of the Annunciator Control Register (including XTRA) are read/write:

Bit	Reset Value	Description
[0]	0	LA1 - Shift
[1]	0	LA2 - ALT Shift
[2]	0	LA3 - Alpha
[3]	0	LA4 - Alert

Bit	Reset Value	Description
[4]	0	LA5 - Busy
[5]	0	LA6 - Transmitting
[6]	0	XTRA - does nothing.
[7]	0	AON - Annunciators on. The annunciators are enabled if the timer TRUN bit is also set.

CHAPTER 12
CYCLIC REDUNDANCY CHECK

The Cyclic Redundancy Code (CRC) Register is a readable/writable register containing a 16-bit CRC based on data read by the CPU. A CRC is a type of checksum that when used to detect errors is referred to as a signature. The CRC register is used in the production test to verify the ROM and may also be used by software.

The CRC Register is a 16-bit polynomial counter whose input is a function of the data on the internal data bus. Data is clocked into the CRC Register only when data is being read through the DP. This allows software to use the PC freely without affecting the CRC register. The data read by the DP from the control registers is also not clocked into the CRC. This allows the CRC to be read without modifying its contents, however the last nibble of the control registers (address 3F - Part of Timer2) can be clocked in.

The 1LT8 CRC conforms to a industry standard and is compatible with the CRC generated by a VAX CRC instruction and the Kermit type three checksum. The CRC calculation treats the data as a string of bits with the low-order bit of the first nibble first and the high-order bit of the last nibble last. The 16-bit CRC is the remainder after dividing the data bit string by the CCITT polynominal:
X^16 + X^12 + X^5 + 1

The CRC register will detect better than 99.997% of all errors, including:

- All single bit errors
- Any number of errors that occur within a single burst of 16 bits
- All but 31 ppm of errors that occur within a single burst of 17 bits
- All but 15 ppm of errors that occur within a burst of more than 17 bits

The algorithm for computing the CRC can be described compactly by the C program statement:

```
/* crc  - crc register                                  */
/* nib  - nibble to be accumulated into crc             */

crc = ( crc >> 4 ) ^ ( ( ( crc ^ nib ) & 0x000f ) * 0x1081 ) ;
```

Or less compactly by the algorithm:

A. Exclusive-OR the least significant nibble of the CRC register with the next nibble to be accumulated.
B. Multiply this result by the magic number 10201 (octal).
C. Shift the CRC register right by one nibble.
D. Exclusive-OR the result from B with the CRC register.
E. Store this result back into the CRC register.

12.1 **Examples**

If the CRC register is cleared and the character 'A' (41 hex) is read with DP0, the CRC register will be modified as follows:

	CRC register	result
nibble 1 - 0001:	0000 0000 0000 0000	
step A	0000 0000 0000 0000	0001
step B	0000 0000 0000 0000	0001 0000 1000 0001
step C	0000 0000 0000 0000	
step D	0000 0000 0000 0000	0001 0000 1000 0001
step E	0001 0000 1000 0001	
nibble 2 - 0100:	0001 0000 1000 0001	
step A	0001 0000 1000 0001	0101
step B	0001 0000 1000 0001	0101 0010 1000 0101
step C	0000 0001 0000 1000	
step D	0000 0001 0000 1000	0101 0011 1000 1101
step E	0101 0011 1000 1101	

The CRC register would end up with the value 538d (hex) in it.

If the CRC register was cleared and the string "Preston Brown" was read from DP0, the CRC result would be f577 (hex).

CHAPTER 13
CARD DETECT

The Card Detect Circuit uses the CDT[1:2] pins to sample the write protect outputs (NWPOUT) of plug-in cards. It can determine if a card is plugged in and if it is writable. A High on the CDT pins indicates that a card is present and writable, a Low indicates that a card is present and write protected, and a Float indicates that no card is present. When enabled the card detect circuit cycles through floating the CTD pads, passively driving them low, and passively driving them high to determine its state.

13.1 **Card Detect Register**

A two nibble register controls the circuit and shows its status:

Bit	Reset Value	Description
[0]	0	SWINT- Software interrupt. Used by software to generate an interrupt.
[1]	0	SMP - Set module pulled, when high NINT is pulled low to set the CPU module pulled bit. Set module pulled is set when the card detect logic detects that a card has been inserted or removed. This bit must be cleared to stop and acknowledge the interrupt. WARNING: Clearing this bit is asynchronous with the card detect control logic. The card status bits (P1C etc.) should be read after clearing this bit to get the latest information without missing an interrupt.
[2]	0	RCDT - Run Card Detect; If RCDT, ECDT and TRUN are set then the card detect logic runs continuously, updating its status every 122uS. This bit should be set while the CPU is running. In this mode the CDT pads are fighting with a cards write protect output 30uS every 122uS.

[3]	0	ECDT- Enable Card detect, If ECDT and TRUN are set then the card detect logic runs updating its status every 1mS. This bit should always be set. In this mode the CDT pads are fighting with a cards write protect output no more then 30uS every millisecond. When low P1C, P2C, P1W and P2W are all cleared.

Bit	Reset Value	Description
[0]	0	P1C- High when port 1 has a card inserted. Read Only.
[1]	0	P2C - High when port 2 has a card inserted. Read Only.
[2]	0	P1W - High when port 1 is writable. Read Only. When low all writes to CE[1] are disabled.
[3]	0	P2W- High when port 2 is writable. Read Only. When low all writes to CE[2] are disabled.

13.2 **Other Applications**

The Card detect pads can also be used for other purposes:

1. If OSCTM (see chapter on TESTING) is set the internal clocks are output on the card detect pads.
2. If TRUN is set and ECDT is low then NINT can be input on CDT1 and NINT2 can be input on CDT2. The inputs are ORed with the other on chip interrupt sources.

13.3 **Special Considerations**

1. The status bits (P1C etc) are initialized to zero. Therefore, if a card is plugged in when the circuits are enabled then a module pulled interrupt will be generated.
2. When TRUN is set if ECDT is low then the CDT pads are passively pulled high and NINTs are input. Since the CDT pads were in an unknown state before, a glitch may be generated on the interrupt signals. To prevent this ECDT could be set before TRUN.
3. The Plug-in cards are powered down when the system is in Deep sleep and when powered down they are write protected. The Card detect will detect their status and clear P1W and P2W. Therefore, after waking from Deep sleep allow time for the Card detect circuit to restore P1W and P2W.

CHAPTER 14
SCRATCH PAD

A one byte scratch pad is available in the control registers. This byte controls nothing and may be used to hold status information.

This register is especially useful for any code which reconfigures the RAM. While the RAM is unconfigured the scratch pad is the only RAM outside the CPU available.

CHAPTER 15
SERVICE REQUEST REGISTER

A two nibble read only status register helps sort interrupt causes quickly:

Bit	Reset Value	Description
[0]	0	VSRQ - Set when VLBI pulls on NINT2.
[1]	0	USRQ - Set when the UART pulls on NINT2.
[2]	0	TSRQ - Set when one of the Timers pulls on NINT2.
[3]	0	ISRQ - Set when the IR receiver pulls on NINT2.

Bit	Reset Value	Description
[0]	0	LSRQ - Set when the LED driver pulls on NINT2.
[1]	0	NINT - State of NINT.
[2]	0	NINT2 - State of NINT2.
[3]	0	KDN - Set when ON is high or by a keyboard interrupt.

CHAPTER 16
ANALOG CIRCUITRY

The major blocks of analog circuitry are: the power supplies, the oscillators, LCD voltage generation, reset generation, LBI detection, LED current regulation.

16.1 **Power Supply**

The power supply is a boost switching regulator that generates the three system power supplies VDD, VCO, and VH from 3 AAA batteries. Both the VDD and VH are regulated in two quadrants, meaning that they can source or sink current and maintain the regulated voltage. The power supply circuits require only two external diodes, an inductor, an n-channel power mosfet, and three filter capacitors.

The power supply uses the DIODE and PMP pins to generate the voltages and uses the VH and VDD pins to detect the voltages and close the feedback loop. The PMP pin drives the gate of the external power mosfet. When PMP is high, the mosfet is turned on and the inductor builds up a current charge from the battery. When PMP goes low, the mosfet is turned off and the inductor dumps its charge thru one of the diodes into either the VH filter capacitor or into the DIODE pin. There is an on-chip p-channel mosfet connected from the DIODE pin to the VDD pin. When this mosfet is on, the charge from the inductor is routed into the VDD filter capacitor instead of the VH capacitor. There is another on-chip p-channel mosfet connected from the VDD pin to the VCO pin. The VCO supply is derived from the VDD supply in light-sleep or run and is turned off in deep-sleep.

The supply runs at a frequency of 122.88 kHz derived from the VCO and crystal oscillator. When one of the supplies has a low voltage, the PMP pin starts pulsing the n-channel mosfet at 122.88 kHz and 75% duty-cycle. The DIODE mosfet switches so that the correct supply gets the inductor charge. If both supplies are enabled and have a low voltage, the DIODE pin switches between the VDD and VH supplies at 30.72 kHZ so that both supplies receive charge from the inductor. If both supplies have adequate voltage, the PMP pin stops pulsing and the DIODE mosfet turns on. If the VDD supply has too much voltage, an on-chip n-channel mosfet turns on to sink current from VDD into Ground. The DIODE mosfet is turned off in this case. If the VH supply has too much voltage,

an on-chip p-channel mosfet turns on to sink current from VH into the VDD supply. In deep-sleep, the supplies are turned off and the DIODE mosfet is left on so that VDD and VH are maintained from the batteries.

The power supply handshakes with the CPU allowing the CPU to run only after the VDD supply has reached the regulation point. Handshake signals to the display control logic prevent it from running until the VH supply has stabilized.

16.2 **Oscillator**

The oscillator section provides clocks for the chip. A crystal-controlled frequency of 32768 Hz is used for the timer, display and LED sections. When the CPU is running, a high frequency signal is generated at a multiple of the 32 kHz.

The frequency of the CPU clock is regulated by the 32 kHz crystal using a phase locked loop. The frequency is determined by a four-bit RATE value. In 1LT8, this value is set to 'E' and may not be changed by software. The RATE value selects a CPU frequency of (RATE + 1) * 524,288 Hz. For 1LT8, this frequency is 7.86 MHz and the strobe rate is 1.97 MHz. In the absence of a crystal (e.g. during reel burn-in) the oscillator will assume its lowest frequency (somewhere around 100 kHz).

During burn-in, the timer and display circuits are driven from the high frequency oscillator, through a divide by 16*(RATE+1) counter. If the 32kHz oscillator were running, this frequency would be 32kHz.

When BIN or a TEST mode is activated, the clock distribution is changed. Normally LFO, the low frequency oscillator, is driven by the inverse of XOUT. But with BIN (Burn in mode) set LFO is driven by the internal oscillator's divider. The TEST signal does not effect LFO. Normally HFO, the high frequency oscillator, is driven by the internal oscillator. But with TEST set HFO is driven by XIN. The BIN signal does not effect HFO.

16.3 **Low Power Detection**

The 1LT8 analog has four separate low power detection circuits. They are memory mapped into the Low Power Registers as follows:

Bit	Read Only Indicator Description
[0]	VLBI - Very Low Battery Indicator. True if the system battery voltage VB[0] is below the VLBI trip point. The

VLBI bit is read only. VLBI is intended to be used to shutdown the system to preserve memory. This bit is deglitched in hardware and requires that VB[0] be below the VLBI trip point continuously for 8mS-16mS before triggering an interrupt. If enabled VLBI will cause a NINT2 wake-up and interrupt and set the VSRQ bit when this condition is met.

[1] LB0 - Low Battery Indicator. True if the system battery voltage VB[0] is below the VLBS trip point.

[2] LB1 - Low Battery Indicator. True if the keep alive battery voltage VB[1] of memory port 1 is below the VLBC trip point.

[3] LB2 - Low Battery Indicator. True if the keep alive battery voltage VB[2] of memory port 2 is below the VLBC trip point.

The LBI readings are not always accurate at a single reading, especially if the CPU is running. The following strategy should be used to overcome this problem:

1. The state of LBI should not be derived from a single reading. At least three readings, voting in unanimity should be used. These bits are updated continuously when the circuit is enabled, no delay is necessary between readings.

Bit Control Bit Description

[0] RST - Reset This bit is set when a hardware reset has been triggered by the NRES pin or by the POR circuit. A reset pulse is generated at initial power-up (POR circuit) and when the NRES pad is pulled low. The NRES input is de-glitched to prevent ESD sensitivity. It is intended to allow discrimination of hardware resets from software resets that both trap through address 00000.

[1] GRAM - Glitch Sensitive Ram. GRAM is a RAM cell that is designed to be sensitive to power supply glitches. This status bit is set when a reset has been triggered by GRAM. This bit is read/writable. GRAM is intended to reset the chip into a known state after a power supply glitch has occurred that may have flipped other RAM cells. GRAM also includes a watchdog circuit which will be triggered if the VDD supply can not be regulated for more than 2 seconds.

[2] EVLBI - This bit enables the VLBI detection analog; care should be taken to turn off the VLBI analog before going

into Deep sleep. EVLBI should be set when Running or while in Light sleep so that the VLBI circuit can interrupt or wake up the system. This allows the firmware the opportunity to clean up and power down the system to preserve memory.

[3] ELBI - This bit enables the LBI detection analog; care should be taken turn off the LBI analog when it is not in use.

Note that VLBI cannot cause a RESET, and GRAM cannot cause an interrupt.

16.4 **LCD Drive Voltages**

This section is mainly of interest to hardware, and may be safely skipped by those attempting to program.

The LCD material is intended for 64-way multiplexing with a room-temperature drive requirement of about 8V at 1/9th bias ratio. One voltage level is generated, from which six drive levels are defined. (See Figure 16.1.) VB is twice the smallest step voltage (the LCD bias voltage) from VH. It is derived from diode reference voltages. VH is the positive peak voltage. It is developed by the power supply, referenced to GND. VH is nominally 8.5V. VSSH is nominally VH-4.5*(VH-VB) or GND (which ever is higher) when 64 way multiplexing and VH-3.3*(VH-VB) when 32 way multiplexing. The six voltage levels used are: VSSH, VSSH+(VH-VB)/2, VSSH+(VH-VB), VH, VH-(VH-VB)/2, and VB. Only VSSH, VH, and two other voltages are ever used at a time.

Figure 16.1. shows the timing when 64-way multiplexing; for 32-way multiplexing each ROW is selected for 488uS (twice as long) and the pattern repeats after 32 ROWs.

The IC process used to fabricate the chip has voltage limits. To prevent harm to the chip, the VH supply has to be voltage limited, and VSSH cannot go below GND. To compensate for these limitations, VB includes feedback from the value of VH-VSSH. If VH-VSSH is lower than it should be, VB is boosted slightly to keep the VOFF level of the LCD constant.

This scheme should improve the display quality, because the VOFF level is maintained even under limiting conditions. The bias ratio will vary, but it is generally a slowly varying function with bias, and the best available contrast ratio should be obtained.

A digital contrast register is also available. Five bits provide 32 levels in the 1 to 4V range (for VB). The contrast control adjusts the desired level of VB.

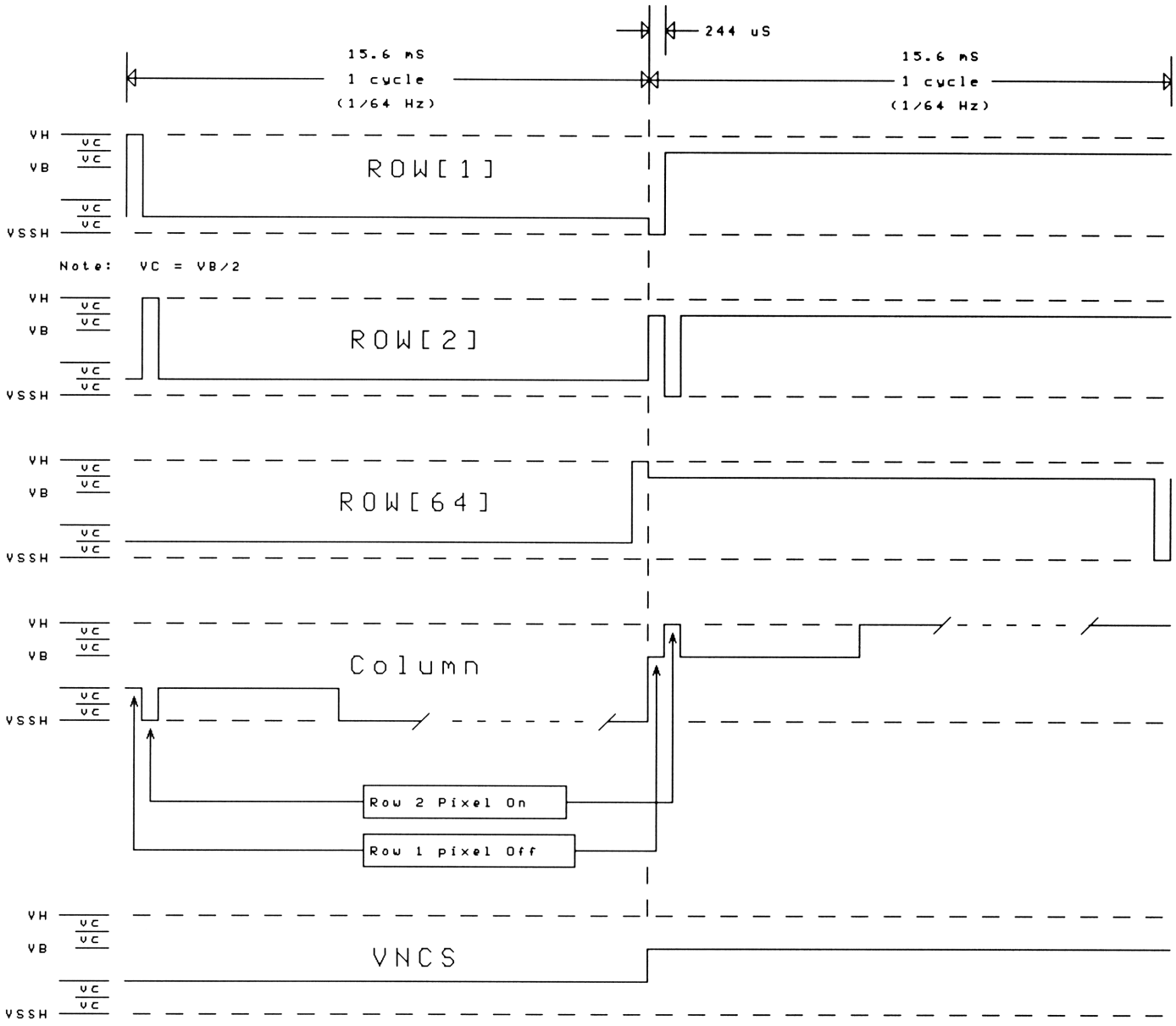


Figure 16.1. LCD drive waveforms.

CHAPTER 17
TESTING

17.1 Mode Register

The MODE register can be written using a PC WRITE bus command. All PC writes map to the MODE register. Since the CPU is not capable of writing through the program counter, these modes are usable only by test fixtures. The MODE REGISTER can be read using either pointer. The MODE value is decoded into individual control signals. These control signals have the following effect:

SIGNAL	DESCRIPTION
OD	Tristates all pads for the input leakage test. Also resets the CPU.
OSCTM	Oscillator test mode, causes HFO to be output on CDT1 and LFO on CDT2.
SCAN	Forces a test mode. The CPU is under the control of the CPU scan path (See section below).
TEST	When TEST is set HFO is input from XIN.
JANET	Forces a development mode intended to allow for exact ROM timing emulation, and rambox use. The ROM memory controller is disabled except to cause CPU wait states.
ESBI	Enables the Saturn bus interface.
VERBX	Forces a test mode during which all SBI activity is echoed onto the external bus. Necessary for any external Saturn bus device to operate.
PLEV	When set the CPU register bank is precharged low, used to test RAM retention.
DPWR	Disables the VDD power supply and forces the CPU/Power supply handshake lines to be ignored. Used to run the system on an external power supply or straight from the batteries.

17.2 **Mode Programming**

The decoding of the value of the MODE REGISTER into the 16 possible modes is defined below.

Mode Name	Mode Value	Outputs: OD	OSCTM	OSCAN	TEST	JANET	ESBI	VERBX	PLEV	DPWR
Quiet	0000	0	0	0	0	0	1	0	0	0
JanetDP	0001	0	0	0	0	1	1	1	0	1
Janet	0010	0	0	0	0	1	1	1	0	0
	0011	0	0	0	0	0	1	1	0	1
Osctm	0100	0	1	0	0	0	1	1	0	0
OsctmDP	0101	0	1	0	0	0	1	1	0	1
Verbose	0110	0	0	0	0	0	1	1	0	0
VerbosDP	0111	0	0	0	0	0	1	1	0	1
QuietT	1000	0	0	0	1	0	1	0	0	1
PowerT	1001	0	0	0	1	0	1	1	0	0
JanetT	1010	0	0	0	1	1	1	1	0	1
VerbT	1011	0	0	0	1	0	1	1	0	1
CPUT	1100	0	0	0	1	0	0	1	0	1
PreLow	1101	0	0	0	1	0	0	1	1	1
Scan	1110	0	0	1	1	0	0	1	0	1
OD	1111	1	0	0	1	0	0	1	0	1

17.3 **CPU Scan Path**

The CPU scan path is provided to make testing of the CPU PLA easier and more complete.

The CPU scan path is composed of a 55-bit shift register consisting of the 14 distinct PLA inputs and 41 PLA outputs. The shift register is clocked one bit position each CPU cycle and is designed to present input data to and latch output data from the PLA using the same timing that is used in normal CPU operation. This is intended to allow PLA speed testing with the scan path. ON serves as the scan path input and LN serves as the scan path output.

In scan mode the PLA is cycling as normal but the PLA outputs are never latched into the scan path or control line drivers. This allow the CPU to hold a particular state during the many cycles required to setup an input pattern. Leaving scan mode momentarily when the input shifting is complete allows the PLA output resulting from the input pattern to be latched into the scan path and control line drivers.

Since scan mode is controlled using a memory-mapped register, another control mechanism was required to allow precise timing control over scan mode. LA[1] provides direct timing control over CPU scan mode. If LA[1]=1 AND MODE=1110 then scan mode is active. If LA[1]=0 then scan mode is inactive.

The HALT pin enhances the versatility of the scan path. When HALT is low, the scan path operates as described above. When HALT is raised high the scan path shifting is halted and the PLA control line drivers are loaded from the scan path. This allows the CPU to be controlled with control patterns that are unavailable in the PLA.

17.4 **Display Testing**

Each LCD ROW driver (LR[1:64]) consists of two transmission gates, and one bit of a serial shift register. Row outputs may be connected to the Ron or Roff signals (VRS,VNRS). These on and off signals are normally analog voltages generated by the power supply. If the VDIG bit is set, these voltages are forced to logic levels. If a pad latch has been loaded with a "0", the off signal is connected to the pad. Loading a pad latch with a "1" will connect the on signal to the pad.

During normal operation, row data from the display control logic is shifted out to the pads. However, the test register allows direct control over these functions.

In normal operation the shift clock is generated by the display control logic but when testing it is controlled by LRTC.

In normal operation, the on and off signals reverse polarity after each display refresh cycle so that the LCD will not see a DC voltage. During the test modes, this polarity reversal is controlled by LID.

The display testing modes allow testing of all 64 LCD ROW output

drivers without probing all the pads.

Bit	Reset Value	Description
[0]	0	CONT[4] - High bit of the contrast control register.
[1]	0	TRIM - Tristates the analog non-select outputs.
[2]	0	LID - LCD invert data.
[3]	0	VDIG - Forces the analog outputs to digital levels.

Bit	Reset Value	Description
[0]	-	BIN - Burn-in. This read only bit is normally cleared. It is set when the TGND is pulled to GND.
[1]	0	LRTC - LCD ROW test clock.
[2]	0	LRTD - LCD ROW test data.
[3]	0	LRT - LCD ROW test signal.

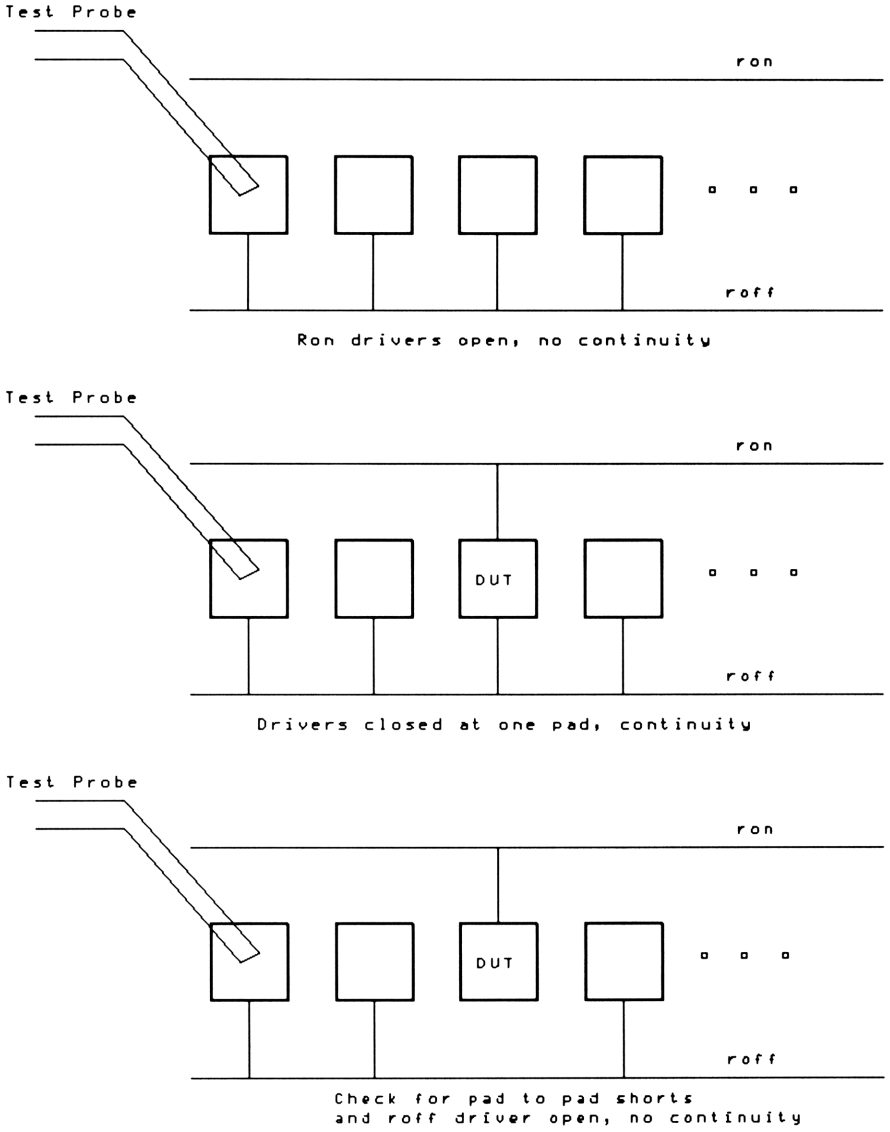
TRIM, LID, VDIG, LRTC, LRTD and LRT should normally be set to zero. If LRT, TRIM, or VDIG are set then the ROW data, clock and inversion signal are controlled by LRD, LRC and LID.

The first test checks that all of the display pad transmission gates will turn off, that is disconnect from Ron. When in this mode, the Roff signal is open circuited so that it may be driven or sensed from any output driver. First all zeros are loaded into the pad latches. If functioning properly all the pads should be connected to Roff through the Roff transmission gates and all of the Ron gates should be open. Then Ron is pulsed by using the LID bit, if an output follows LID then one or more of the Ron gates are bad.

Next, the LRT signal forces the pads into a test mode that causes both transmission gates to turn on if the latch value is a one. Roff is again open. First the pads latches are loaded with zeros, tying all of the pads to Roff, then a single one is shifted into the first pad. This one will turn on both transmission gates at that pad. Then if Ron is pulsed, pulses should be caused on Roff (through that one driver) and all the outputs should follow.

With a single one in the pad latches, pad to pad shorts and Roff opening can be checked for by switching off LRT and pulsing Ron with the LID bit. If the output follows LID then Ron is connected to Roff because the Roff transmission gate did not turn off or two adjacent pads are shorted.

Figure 17.1. Testing the LCD Drivers.



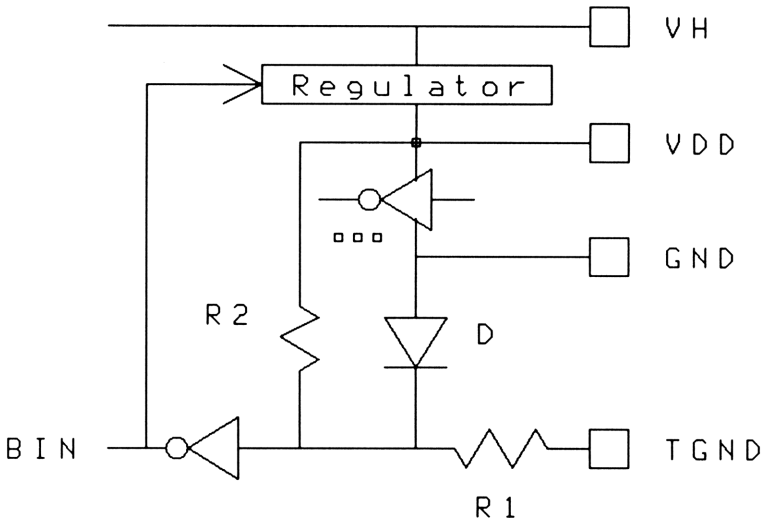
17.5 **Power Isolation Circuit**

A Power Isolation Circuit allows many 1LT8 chips to undergo burn-in and testing on a single TAB reel. When the chips are on the reel, all VH pads are tied together and TGND pins are tied together by the TAB substrate. During burn-in, power is applied to the entire reel across the VH and TGND lines. This forward biases the diode shown below in Figure 17.2 and provides power to each chip. Since TGND is at a low level, the BIN signal is true. The resistor R1 insures that a chip cannot short out the power supply. During burnin VDD is regulated on chip with a special regulator which divides down VH.

During production test, power is applied to a single chip across the VDD and GND lines. Since the diode is reversed biased the tester power supply is isolated from all other chips on the reel.

During normal operation the pull-up resistor on TGND provides a high level and the BIN signal is false.

Figure 17.2. Power Isolation Circuit.



17.6 **Burn-In**

As was described in the previous section, the 1LT8 chip is designed to allow the burn-in of many ICs on a single TAB reel. The clock generation circuitry is designed to provide clocks even when no quartz crystal is attached to the XIN and XOUT pads.

Under these conditions the CPU clock will be approximately 100KHz and the timer clock is the CPU clock divided by 16*(RATE+1).

During burn-in the BIN signal is a "1". This signal is routed through the control logic and starts as many circuits as possible. In this way, a dynamic burn-in can be accomplished on a TAB reel by applying only power.