Document No : UniKL MFI_SD_AC41
Revision No: 02
Effective Date: 01 December 2008

CONFIDENTIAL

SET A

UNIVERSITI KUALA LUMPUR
Malaysia France Institute

FINAL EXAMINATION
SEPTEMBER 2013 SESSION

SUBJECT CODE	:	**FSB23203**
SUBJECT TITLE	:	**MICROCONTROLLER**
LEVEL	:	**BACHELOR**
TIME / DURATION	:	**(3 HOURS)**
DATE	:	

INSTRUCTIONS TO CANDIDATES

1. **Please read the instructions given in the question paper CAREFULLY.**
2. **This question paper is printed on both sides of the paper.**
3. **Please write your answers on the answer booklet provided.**
4. **Answer should be written in blue or black ink except for sketching, graphic and illustration.**
5. **This question paper consists of TWO (2) sections. Section A and B. Answer all questions in Section A. For Section B, answer three (3) questions only.**
6. **Answer all questions in English.**

THERE ARE 10 PAGES OF QUESTIONS AND 8 APPENDIXES, EXCLUDING THIS PAGE.

SECTION A (Total: 40 marks)

INSTRUCTION: Answer ALL questions.
Please use the answer booklet provided.

Question 1

(a) Describe briefly two criteria to be considered for choosing a microcontroller for a target application.

(6 marks)

(b) List down two registers in 8051 and explain their function.

(2 marks)

(c) A 16 bits data equals to ___________ nibble(s) or __________ word(s).

(1 mark)

(d) State the most important factor in choosing a microcontroller for a battery-based embedded product.

(1 mark)

Question 2

(a) Write a program to add two 16 bits hexadecimal numbers which are 0x3AB2 and 0x4F0D. Put the higher byte in R6 and the lower byte in R7.

(5 marks)

(b) Given the following segment of code:

```
MOV   R3, #200
MOV   R5, #0xF3
MOV   R6, #11000011B
PUSH  3
PUSH  6
PUSH  5
POP   1
POP   7
POP   0
```

List all the content of the memory from address #00 to #0A.

(5 marks)

Question 3

Given the following segment of code:

```
MOV   R3, #0EFH
MOV   R5, #07DH
MOV   A, R5
SUBB  A, R3
```

According to the code given:

(a) Briefly explain why PSW register becomes 0xC4 after executing the last instruction.

(4 marks)

(b) Is the answer given by microcontroller correct? Please elaborate the method used to know the correctness of the answer.

(2 marks)

(c) Prove the answer given in Question 3(b) by performing manual calculation.

(2 marks)

(d) Assume that the answer given by microcontroller is wrong. Add some lines of code to make the given answer correct.

(2 marks)

Question 4

(a) Calculate the frequency of the square wave generated on P1.5:

```
HERE:       MOV TL0,#0F2H
            MOV TH0, #0FFH
            CPL P1.5
            ACALL DELAY
            SJMP HERE

DELAY:      SETB TR0
AGAIN:      JNB TF0, AGAIN
            CLR TR0
            CLR TF0
            RET
```

(5 marks)

(b) Find the delay generated by Timer 0 in the following code. Do not include the overhead due to instructions.

```
            CLR P2.3
            MOV TMOD, #1
HERE:       MOV TL0, #3EH
            MOV TH0, #0B8H
            SETB P2.3
            SETB TR0
AGAIN:      JNB TF0, AGAIN
            CLR TR0
            CLR TF0
            CLR P2.3
```

(5 marks)

SECTION B (Total: 60 marks)

INSTRUCTION: Answer only THREE (3) questions.

Please use the answer booklet provided.

Question 5

In a bank, there are **4 counters** opened to serve the customer. Each customer has his/her **turn number** which goes from **0 to 255**. It will **restart to 0** once it reaches 255. However, there is only a **4-digit 7-segment multiplex display** which is used to call the number of customer to be served. It is divided into **two (2) sections**. The first section is the most significant digit (the most left) which represents the **number of counter** who calls the customer (1 to 4). The other sections represent the **turn number** of customer (0 to 255). Given in the following figure the architecture of the system:

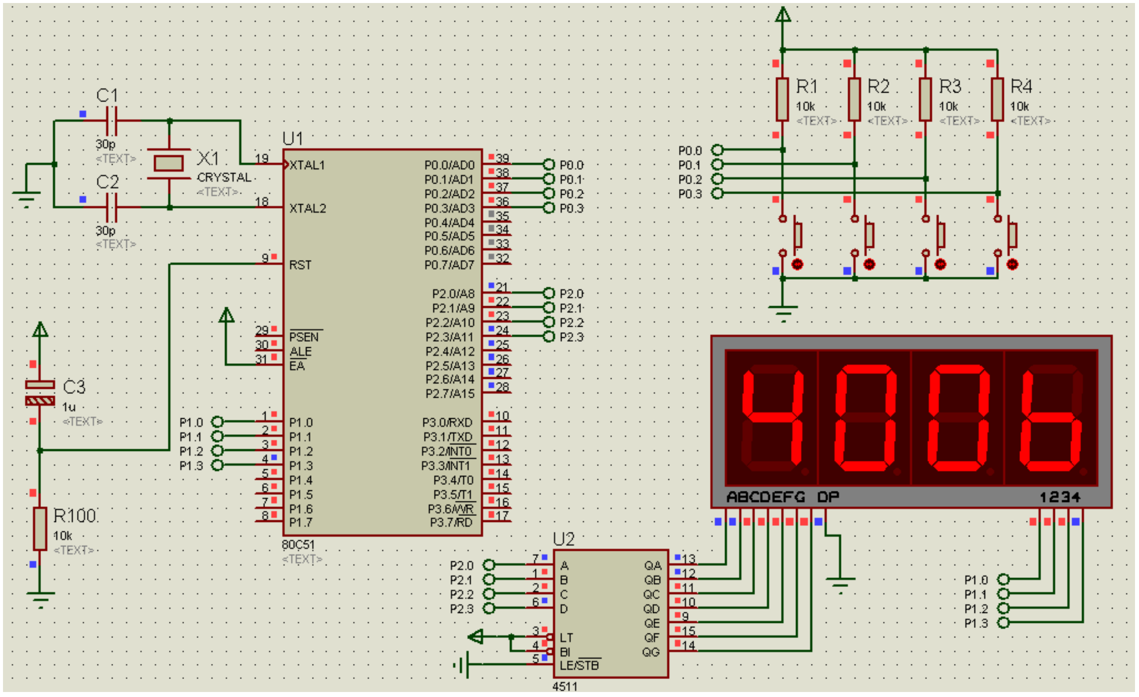


Figure 1: The architecture of customer queue system

There are **4 normally open push buttons** which represent each counter (**1 to 4**). The 7-segment is connected to the microcontroller via a **4511** decoder. Each time one of the push buttons is pressed, the turn number will **be incremented to one**. 4006 which is displayed on the 7-segments means as the following:

- 4: push button is pressed by counter 4
- 006: the current number to be served is 006.

(a) According to the Figure 1, state which type of 7-segment does the system uses (common cathode or anode). Please justify your answer.

(2 marks)

(b) Develop the assembly code of the system described above.

(18 marks)

Question 6

Figure 2 shows a **conveyor system** that is used to **transport goods** from **left to right position** and **vice versa**. Two **12 V_{DC} motor**, M1 and M2 (connected in parallel to each other) are used to drive the system. **Four switches** (2 push buttons normally opened and 2 limit switches) will be used to control the movement of the system. When **F** switch is pushed, the motor M1 and M2 will rotate **clock wise** and the goods will be sent **forward**. Once the good reach the destination, the limit switch, **s2** will trigger and both motors will **stop**. However, when **R** switch is pushed, the motor M1 and M2 will rotate **counter clock wise** and the goods will be sent **backward**. Once the goods reach the destination, the limit switch, **s1** will trigger and the both motor will **stop**.

A motor driver, **L293D** is connected to the microcontroller 8051 with X'TAL=11.0592 MHz to drive the motor rotation.

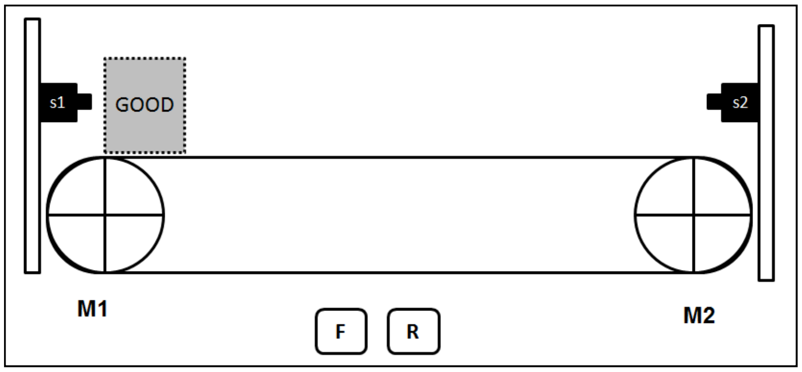


Figure 2: Conveyor system

(a) Design a motor **driving circuit** using L293D motor driver. Clearly highlight which pin should be connected to VCC, GDN and 8051 I/O pins.
Note: The specification for L293D motor driver is given in Appendix 3.

(8 marks)

(b) Draw the **logic configuration table** for driving the motor **forward, reverse and stop**.

(4 marks)

(c) Create an **assembly program** for the system.

(8 marks)

Question 7

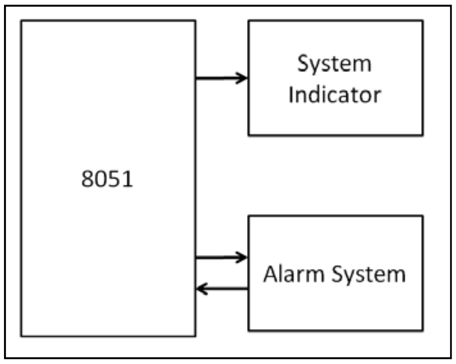


Figure 3: Alarm System

Figure 3 shows the concept of alarm system. The 8051 microcontroller needs to control **two (2) applications** which are **system indicator** and **alarm system**.

System indicator contains **an LED** that will blink continuously while the system runs.

Alarm system contains a sensor (replaced by **a push button**) as an **input** of microcontroller and a **5V buzzer** as an **output** of microcontroller. When the button is pushed, the system will be **interrupted** and the buzzer will be **triggered to on**. The buzzer will **stop** when the button is **pushed again**.

(a) State the differences of using **polling** or **interrupt** method.

(2 marks)

(b) Draw the **electronic circuit diagram** of the system described above (System indicator uses **timer interrupt 0** and alarm system uses **external interrupt 1**).

(5 marks)

(c) Based on the circuit designed in Question 7(a), write a program to develop **system indicator module** which uses **timer interrupt 0** to generate **2ms** of delay and **alarm system module** which use **external interrupt 1** as method to trigger the buzzer when the pushed button is pressed.

(13 marks)

Question 8

RB-41 is a peripheral control panel device developed by RRS group for their new national car, Proton Tuah. The RB-41 is equipped with LCD touch screen module-Main Controller and sub-controller in which they are interfaced by using serial communication. All accessories modules such as power windows module, air-conditioner, indoor lamp, rear sunshield are controlled via the touch screen panel (refer Figure below).

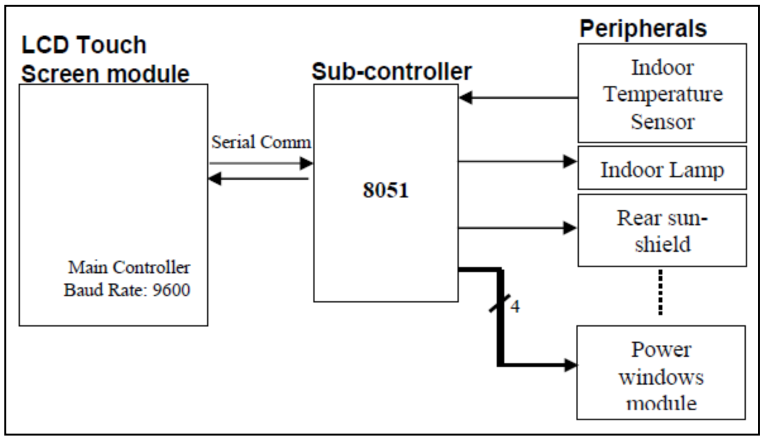


Figure 4: RB-41 architecture

The RRS group is instructed to upgrade the device by adding on central door lock module. The data assignment for the new module has been set by the project engineer as shown in Table 1 below:

Table 1: Central Lock Door Module Data Assignment

OUTPUT	TASK	ASCII character sent from LCD module	PIN
Front-right door, FD1	Lock	s	P2.0
	Unlock	t	
Front-left door, FD2	Lock	u	P2.1
	Unlock	v	
Rear-right door, RD3	Lock	w	P2.2
	Unlock	x	
Rear-right door, RD3	Lock	y	P2.3
	Unlock	z	

(a) Which **timer register** will be involved in calculating the **baud rate**? Calculate the **hexadecimal value** to be loaded into the register for **9600 bps**. Assume the crystal clock is **11.0592MHz**.

(5 marks)

(b) Single wire-power door lock actuators (single acting solenoid-spring return) are used for each door. If the wire is connected to **12Vdc**, the doors will **lock** and when it is **grounded**, then the doors will **unlock**. Draw the **electronic circuit diagram** for the central door lock system. ***Hint:*** *Use relay to control the 12V buzzer.*

(5 marks)

(c) Create a **program** for the system above based on the circuit designed in (b).

(10 marks)

END OF QUESTION

APPENDIX 1
DATA SHEET AND INSTRUCTION SET

ARITHMETIC OPERATORS

MNEMONIC	DESCRIPTION	BYTES	CYCLES	C	OV	AC
ADD A, Rn	Add register to ACC	1	1	x	x	X
ADD A, direct	Add direct byte to ACC	2	1	x	x	X
ADD A, @Ri	Add indirect RAM to ACC	1	1	x	x	X
ADD A, #data	Add immediate data to ACC	2	1	x	x	X
ADDC A, Rn	Add register to ACC with Carry	1	1	x	x	X
ADDC A, direct	Add direct byte to ACC with Carry	2	1	x	x	X
ADDC A, @Ri	Add indirect RAM to ACC with Carry	1	1	x	x	X
ADDC A, #data	Add immediate data to ACC with Carry	2	1	x	x	X
SUBB A, Rn	Subtract Register from ACC with borrow	1	1	x	x	X
SUBB A, direct	Subtract indirect RAM from ACC with borrow	2	1	x	x	X
SUBB A, @Ri	Subtract indirect RAM from ACC with borrow	1	1	x	x	X
SUBB A, #data	Subtract immediate data from ACC with borrow	2	1	x	x	X
INC A	Increment ACC	1	1			
INC Rn	Increment register	1	1			
INC direct	Increment direct byte	2	1			
INC @Ri	Increment direct RAM	1	1			
DEC A	Decrement ACC	1	1			
DEC Rn	Decrement Register	1	1			
DEC direct	Decrement direct byte	2	1			
DEC @Ri	Decrement indirect RAM	1	1			
INC DPTR	Increment Data Pointer	1	2			
MUL AB	Multiply A and B	1	4	0	x	
DIV AB	Divide A by B	1	4	0	x	
DAA	Decimal Adjust ACC	1	1	x		

BOOLEAN OPERATORS

MNEMONIC	DESCRIPTION	BYTES	CYCLES	C	OV	AC
CLR C	Clear carry flag	1	1	0		
CLR bit	Clear direct bit	2	1			
SETB C	Set carry flag	1	1	1		
SETB bit	Set direct bit	2	1			
CPL C	Complement carry flag	1	1	x		
CPL bit	Complement direct bit	2	1			
ANL C,bit	AND direct bit to carry	2	2	x		
ANL C,/bit	AND complement of direct bit to carry	2	2	x		
ORL C,bit	OR direct bit to carry	2	2	x		
ORL C,/bit	OR complement of direct bit to carry	2	2	x		
MOV C,bit	Move direct bit to carry	2	1	x		
MOV bit,C	Move carry to direct bit	2	2			
JC rel	Jump if carry is set	2	2			
JNC rel	Jump if carry is NOT set	2	2			
JB bit,rel	Jump if direct bit is set	3	2			
JNB bit,rel	Jump if direct bit is NOT set	3	2			
JBC bit,rel	Jump if direct bit is set and clear that bit	3	2			

JUMPS AND BRANCHES

MNEMONIC	DESCRIPTION	BYTES	CYCLES	C	OV	AC
ACALL addr11	Absolute call within 2K page	2	2			
LCALL addr16	Absolute call (Long call)	3	2			
RET	Return from subroutine	1	2			
RETI	Return from interrupt	1	2			
AJMP addr11	Absolute jump within 2K page	2	2			
LJMP addr16	Absolute jump (Long jump)	3	2			
SJMP rel8	Relative jump within +/- 127 bytes (Short jump)	2	2			
JMP @A+DPTR	Jump direct relative to DPTR	1	2			
JZ rel8	Jump if ACC is zero	2	2			
JNZ rel8	Jump if ACC is NOT zero	2	2			
CJNE A,direct,rel8	Compare direct byte to ACC, jump if NOT equal	3	2	x		
CJNE A,#data,rel8	Compare immediate to ACC, jump if NOT equal	3	2	x		
CJNE Rn,#data,rel8	Compare immediate to register, jump if NOT equal	3	2	x		
CJNE @Ri,#data,rel8	Compare immediate to indirect, jump if NOT equal	3	2	x		
DJNZ Rn,rel8	Decrement register, jump if NOT zero	2	2			
DJNZ direct,rel8	Decrement direct byte, jump if NOT zero	3	2			
NOP	No operation (Skip to next instruction)	1	1			

LOGICAL OPERATIONS

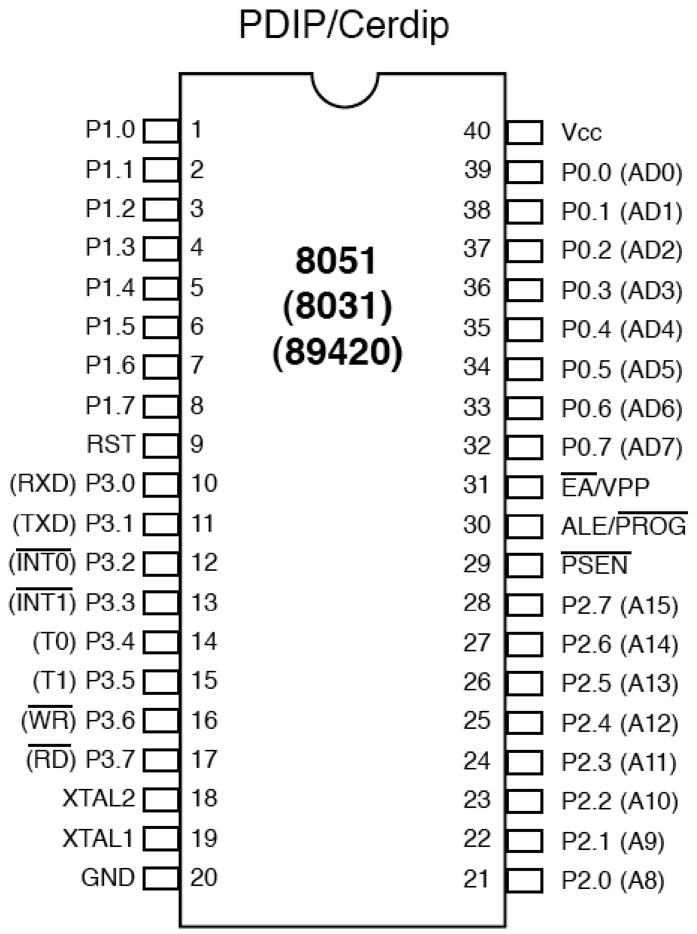
MNEMONIC	DESCRIPTION	BYTES	CYCLES	C	OV	AC
ANL A,Rn	AND register to ACC	1	1			
ANL A,direct	AND direct byte to ACC	2	1			
ANL A,@Ri	AND indirect RAM to ACC	1	1			
ANL A,#data	AND immediate data to ACC	2	1			
ANL direct,A	AND ACC to direct byte	2	1			
ANL direct,#data	AND immediate data to direct byte	3	2			
ORL A,Rn	OR register to ACC	1	1			
ORL A,direct	OR direct byte to ACC	2	1			
ORL A,@Ri	OR indirect RAM to ACC	1	1			
ORL A,#data	OR immediate data to ACC	2	1			
ORL direct,A	OR ACC to direct byte	2	1			
ORL direct,#data	OR immediate data to direct byte	3	2			
XRL A,Rn	XOR register to ACC	1	1			
XRL A,direct	XOR direct byte to ACC	2	1			
XRL A,@Ri	XOR indirect RAM to ACC	1	1			
XRL A,#data	XOR immediate data to ACC	2	1			
XRL direct,A	XOR ACC to direct byte	2	1			
XRL direct,#data	XOR immediate data to direct byte	3	2			
CLR A	Clear the ACC	1	1			
CPL A	Complement the ACC	1	1			
RL A	Rotate the ACC left	1	1			
RLC A	Rotate the ACC left through Carry	1	1	x		
RR A	Rotate the ACC right	1	1			
RRC A	Rotate the ACC right through Carry	1	1	x		
SWAP A	Swap nibbles in the ACC	1	1			

DATA TRANSFER

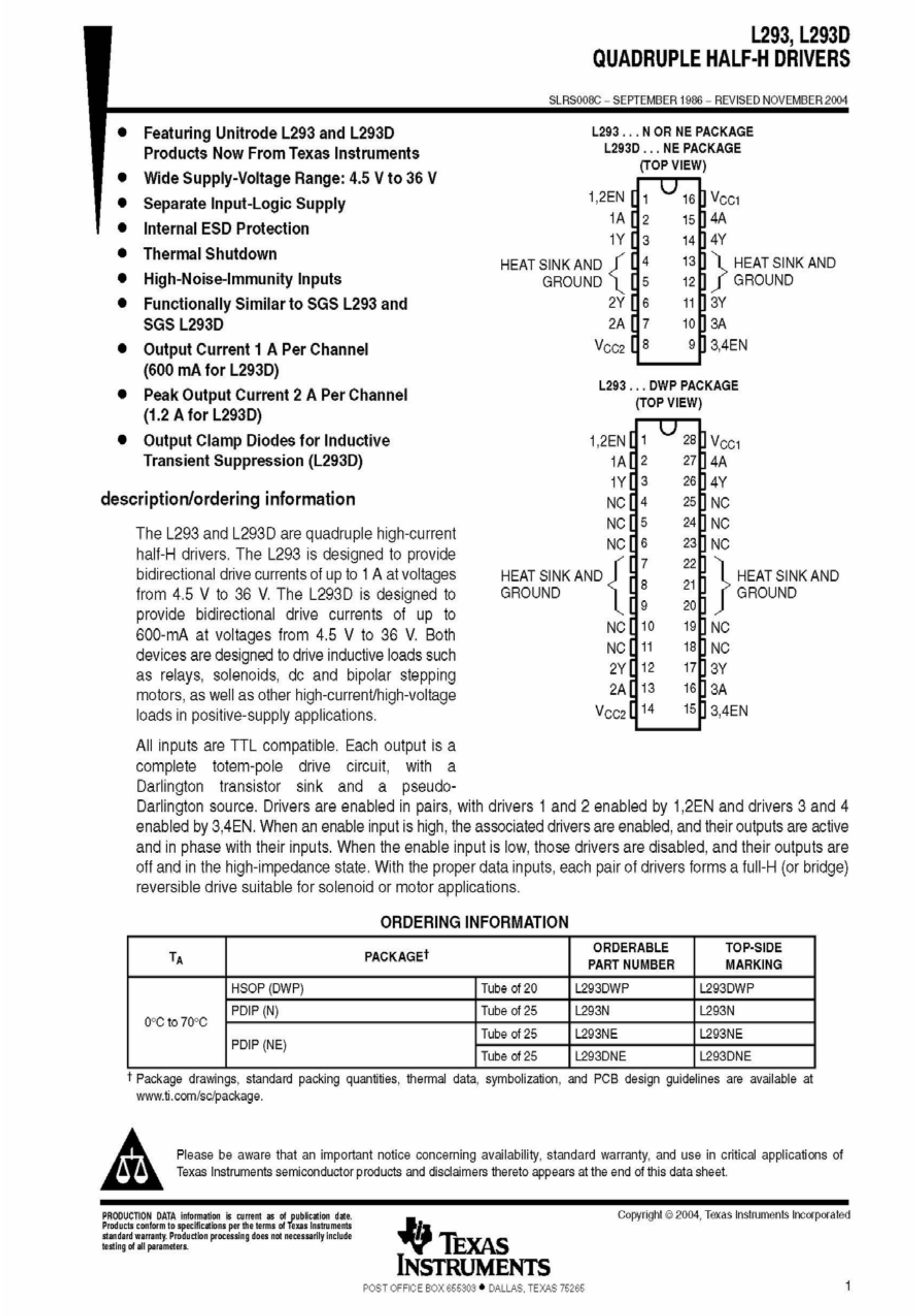
MNEMONIC	DESCRIPTION	BYTES	CYCLES	C	OV	AC
MOV A,Rn	Move Register to ACC	1	1			
MOV A,direct	Move Direct byte to ACC	2	1			
MOV A,@Ri	Move Indirect byte to ACC	1	1			
MOV A,#data	Move Immediate data to ACC	2	1			
MOV Rn,A	Mov ACC to Register	1	1			
MOV Rn,direct	Move Direct byte to Register	2	2			
MOV Rn,#data	Move Immediate data to Register	2	1			
MOV direct,A	Move ACC to Direct byte	2	1			
MOV direct,Rn	Move Register to Direct byte	2	2			
MOV direct,direct	Move Direct byte to Direct byte	3	2			
MOV direct,@Ri	Mov Indirect RAM to Direct byte	3	2			
MOV direct,#data	Move Immediate data to Direct byte	3	2			
MOV @Ri,A	Move ACC to Indirect RAM	1	1			
MOV @Ri,direct	Move direct byte to indirect RAM.	2	2			
MOV @Ri,#data	Move Immediate data to Indirect RAM	2	1			
MOV DPTR,#data16	Load datapointer with 16 bit constant	3	2			
MOVC A,@A+DPTR	Move code byte at ACC+DPTR to ACC	1	2			
MOVC A,@A+PC	Move code byte at ACC+PC to ACC	1	2			
MOVX A,@Ri	Move external RAM to ACC	1	2			
MOVX @Ri,A	Move ACC to external RAM	1	2			
MOVX A,@DPTR	Move external RAM to ACC	1	2			
MOVX @DPTR,A	Move ACC to external RAM	1	2			
PUSH direct	Push direct byte to stack	2	2			
POP direct	Pop direct byte from stack	2	2			
XCH A,Rn	Exchange register with ACC	1	1			
XCH A,direct	Exchange direct byte with ACC	2	1			
XCH A,@Ri	Exchange indirect RAM with ACC	1	1			
XCHD A,@Ri	Exchange low order digit indirect RAM with ACC	1	1			

APPENDIX 2

8051 Pin Assignment



APPENDIX 3
L293D Data Specification

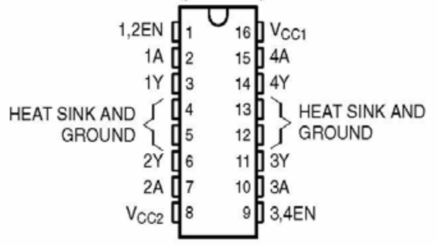


L293, L293D
QUADRUPLE HALF-H DRIVERS

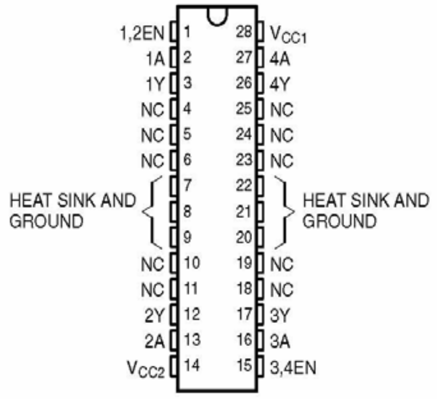
SLRS008C – SEPTEMBER 1986 – REVISED NOVEMBER 2004

- **Featuring Unitrode L293 and L293D Products Now From Texas Instruments**
- **Wide Supply-Voltage Range: 4.5 V to 36 V**
- **Separate Input-Logic Supply**
- **Internal ESD Protection**
- **Thermal Shutdown**
- **High-Noise-Immunity Inputs**
- **Functionally Similar to SGS L293 and SGS L293D**
- **Output Current 1 A Per Channel (600 mA for L293D)**
- **Peak Output Current 2 A Per Channel (1.2 A for L293D)**
- **Output Clamp Diodes for Inductive Transient Suppression (L293D)**

L293 . . . N OR NE PACKAGE
L293D . . . NE PACKAGE
(TOP VIEW)



L293 . . . DWP PACKAGE
(TOP VIEW)



description/ordering information

The L293 and L293D are quadruple high-current half-H drivers. The L293 is designed to provide bidirectional drive currents of up to 1 A at voltages from 4.5 V to 36 V. The L293D is designed to provide bidirectional drive currents of up to 600-mA at voltages from 4.5 V to 36 V. Both devices are designed to drive inductive loads such as relays, solenoids, dc and bipolar stepping motors, as well as other high-current/high-voltage loads in positive-supply applications.

All inputs are TTL compatible. Each output is a complete totem-pole drive circuit, with a Darlington transistor sink and a pseudo-Darlington source. Drivers are enabled in pairs, with drivers 1 and 2 enabled by 1,2EN and drivers 3 and 4 enabled by 3,4EN. When an enable input is high, the associated drivers are enabled, and their outputs are active and in phase with their inputs. When the enable input is low, those drivers are disabled, and their outputs are off and in the high-impedance state. With the proper data inputs, each pair of drivers forms a full-H (or bridge) reversible drive suitable for solenoid or motor applications.

ORDERING INFORMATION

T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	HSOP (DWP)	Tube of 20	L293DWP	L293DWP
	PDIP (N)	Tube of 25	L293N	L293N
	PDIP (NE)	Tube of 25	L293NE	L293NE
		Tube of 25	L293DNE	L293DNE

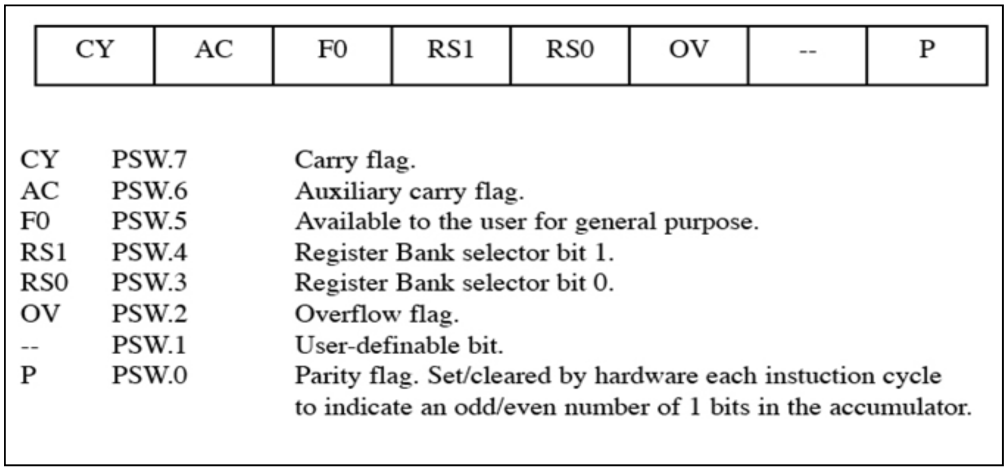
† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

TEXAS INSTRUMENTS
POST OFFICE BOX 655303 ● DALLAS, TEXAS 75265

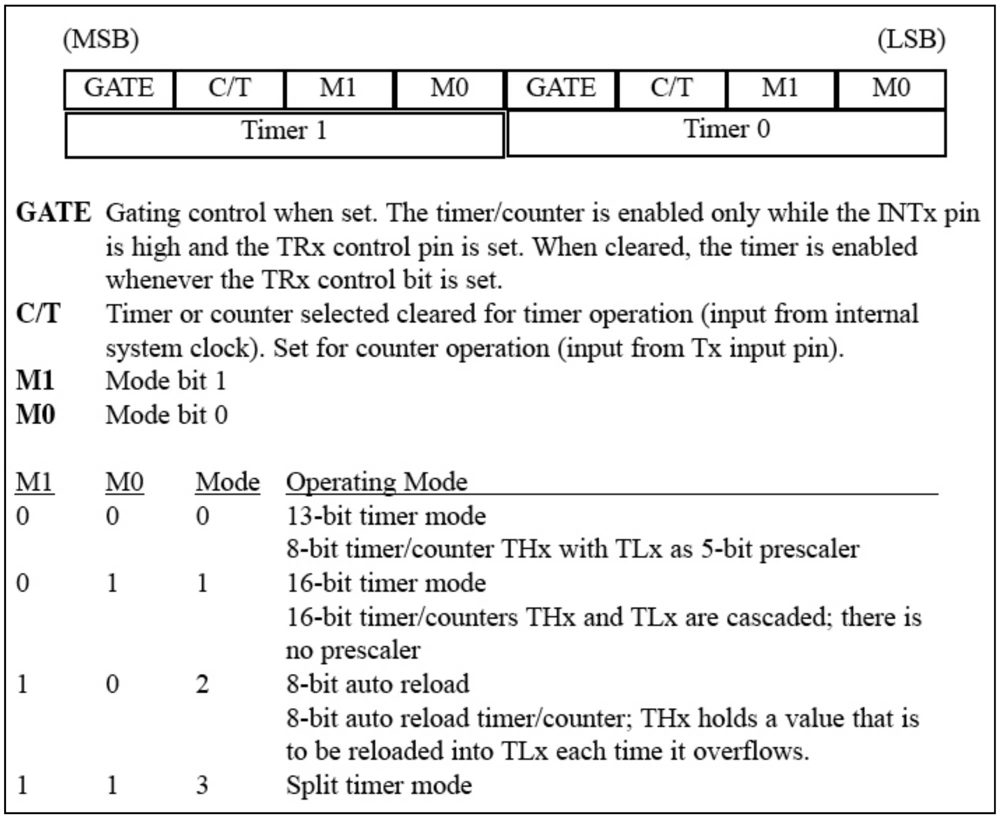
1

APPENDIX 4
PSW, TMOD and TCON Registers



CY	AC	F0	RS1	RS0	OV	--	P

CY	PSW.7	Carry flag.
AC	PSW.6	Auxiliary carry flag.
F0	PSW.5	Available to the user for general purpose.
RS1	PSW.4	Register Bank selector bit 1.
RS0	PSW.3	Register Bank selector bit 0.
OV	PSW.2	Overflow flag.
--	PSW.1	User-definable bit.
P	PSW.0	Parity flag. Set/cleared by hardware each instuction cycle to indicate an odd/even number of 1 bits in the accumulator.



(MSB) (LSB)

GATE	C/T	M1	M0	GATE	C/T	M1	M0
Timer 1				Timer 0			

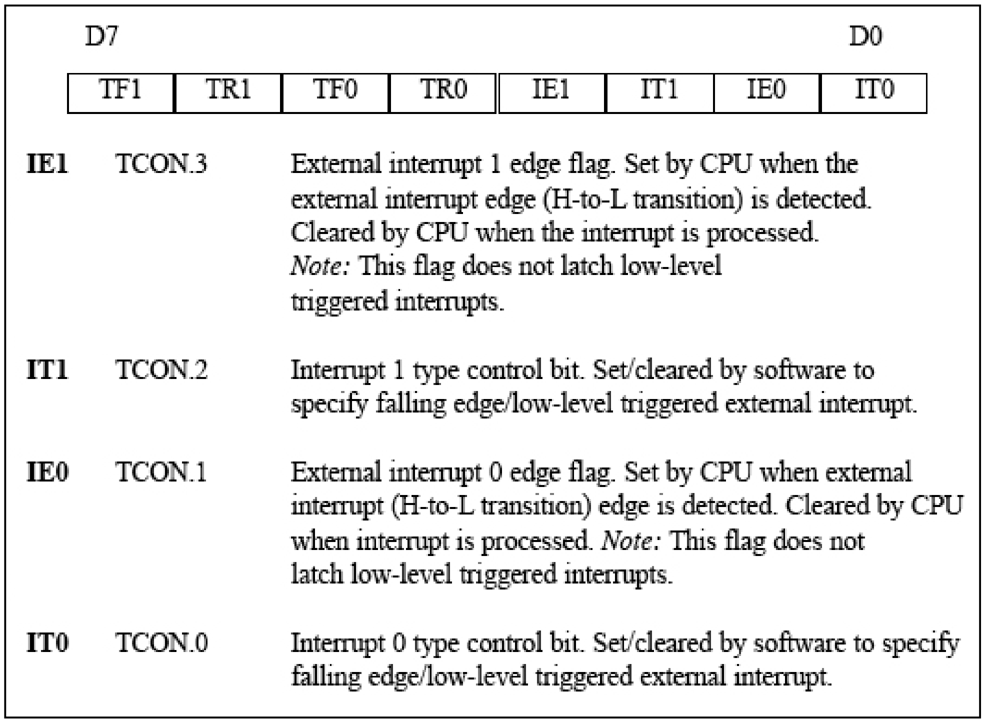
GATE Gating control when set. The timer/counter is enabled only while the INTx pin is high and the TRx control pin is set. When cleared, the timer is enabled whenever the TRx control bit is set.

C/T Timer or counter selected cleared for timer operation (input from internal system clock). Set for counter operation (input from Tx input pin).

M1 Mode bit 1

M0 Mode bit 0

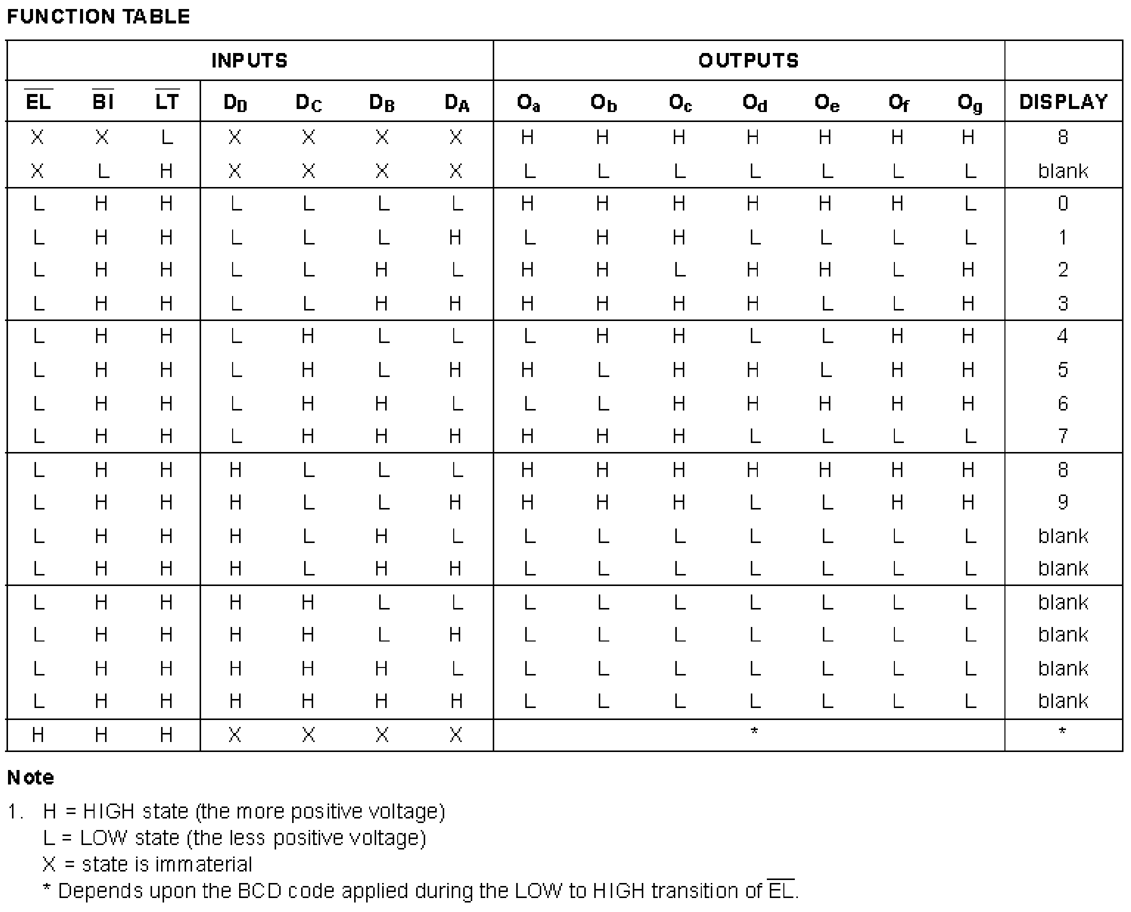
M1	M0	Mode	Operating Mode
0	0	0	13-bit timer mode 8-bit timer/counter THx with TLx as 5-bit prescaler
0	1	1	16-bit timer mode 16-bit timer/counters THx and TLx are cascaded; there is no prescaler
1	0	2	8-bit auto reload 8-bit auto reload timer/counter; THx holds a value that is to be reloaded into TLx each time it overflows.
1	1	3	Split timer mode



D7							D0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

IE1	TCON.3	External interrupt 1 edge flag. Set by CPU when the external interrupt edge (H-to-L transition) is detected. Cleared by CPU when the interrupt is processed. *Note:* This flag does not latch low-level triggered interrupts.
IT1	TCON.2	Interrupt 1 type control bit. Set/cleared by software to specify falling edge/low-level triggered external interrupt.
IE0	TCON.1	External interrupt 0 edge flag. Set by CPU when external interrupt (H-to-L transition) edge is detected. Cleared by CPU when interrupt is processed. *Note:* This flag does not latch low-level triggered interrupts.
IT0	TCON.0	Interrupt 0 type control bit. Set/cleared by software to specify falling edge/low-level triggered external interrupt.

APPENDIX 5
Truth Table of 4511



FUNCTION TABLE

INPUTS							OUTPUTS							
$\overline{EL}$	$\overline{BI}$	$\overline{LT}$	D_D	D_C	D_B	D_A	O_a	O_b	O_c	O_d	O_e	O_f	O_g	DISPLAY
X	X	L	X	X	X	X	H	H	H	H	H	H	H	8
X	L	H	X	X	X	X	L	L	L	L	L	L	L	blank
L	H	H	L	L	L	L	H	H	H	H	H	H	L	0
L	H	H	L	L	L	H	L	H	H	L	L	L	L	1
L	H	H	L	L	H	L	H	H	L	H	H	L	H	2
L	H	H	L	L	H	H	H	H	H	H	L	L	H	3
L	H	H	L	H	L	L	L	H	H	L	L	H	H	4
L	H	H	L	H	L	H	H	L	H	H	L	H	H	5
L	H	H	L	H	H	L	L	L	H	H	H	H	H	6
L	H	H	L	H	H	H	H	H	H	L	L	L	L	7
L	H	H	H	L	L	L	H	H	H	H	H	H	H	8
L	H	H	H	L	L	H	H	H	H	L	L	H	H	9
L	H	H	H	L	H	L	L	L	L	L	L	L	L	blank
L	H	H	H	L	H	H	L	L	L	L	L	L	L	blank
L	H	H	H	H	L	L	L	L	L	L	L	L	L	blank
L	H	H	H	H	L	H	L	L	L	L	L	L	L	blank
L	H	H	H	H	H	L	L	L	L	L	L	L	L	blank
L	H	H	H	H	H	H	L	L	L	L	L	L	L	blank
H	H	H	X	X	X	X				*				*

Note

1. H = HIGH state (the more positive voltage)
   L = LOW state (the less positive voltage)
   X = state is immaterial
   * Depends upon the BCD code applied during the LOW to HIGH transition of $\overline{EL}$.