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

JANUARY 2014 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 7 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) Briefly explain the function of oscillator in microcontroller.

(2 marks)

(b) Explain the differences between microcontroller and general purpose microprocessor.

(4 marks)

(c) Given the following table:

Table 1: List of ROM address

ROM ADDRESS	MACHINE LANGUAGE	ASSEMBLY LANGUAGE
0000	7D23	MOV R5, #23H
i	7F34	MOV R7, #34H
ii	7400	MOV A, #0
iii	2D	ADD A, R5
iv	2F	ADD A, R7

i, ii, iii and *iv* represent the addresses of the ROM for the instructions listed in Table 1. Give the values of those addresses.

(4 marks)

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) Write a program to extract the digits, tens and hundreds from register ACC and put them into register R1, R2 and R3 respectively.

(5 marks)

Question 3

Given the following code segment for the subtraction of two signed number:

```
MOV   R3, #0F7H
MOV   R5, #05DH
MOV   A, R5
SUBB  A, R3
```

According to the code given:

(a) Briefly explain the reason PSW register becomes 0x80 after executing the last instruction.

(4 marks)

(b) Does microcontroller give you the correct answer? 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, is there any method that can be used to get the correct answer? Justify your answer.

(2 marks)

Question 4

Given the following segment of code:

```
      MOV TMOD, #2H
      MOV TH0, #-150
AGAIN:SETB P1.3
      ACALL DELAY
      ACALL DELAY
      CLR P1.3
      ACALL DELAY
      SJMP AGAIN

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

Answer the following questions:

(a) Find the duty cycle of this wave and calculate the time of high portion and low portion of the pulse. Please justify. Exclude overhead due the instructions in the loop.

(6 marks)

(b) Based on the answer in Question 4(b), calculate the frequency of the square wave.

(2 marks)

(c) Modify the segment code above to change the duty cycle to 50% and frequency to 3Hz.

(2 marks)

SECTION B (Total: 60 marks)

INSTRUCTION: Answer only THREE (3) questions.
Please use the answer booklet provided.

Question 5

We want to develop a simple money (note) counter machine. This machine however, is capable only to count 255 pieces at a single time. The main components of the system are as the following:

- 1 switch
- 1 push button
- 1 seven segment 7SEG-MPX4-CC
- 1 decoder 4511
- 1 8051 microcontroller
- 1 10kOhm resistor

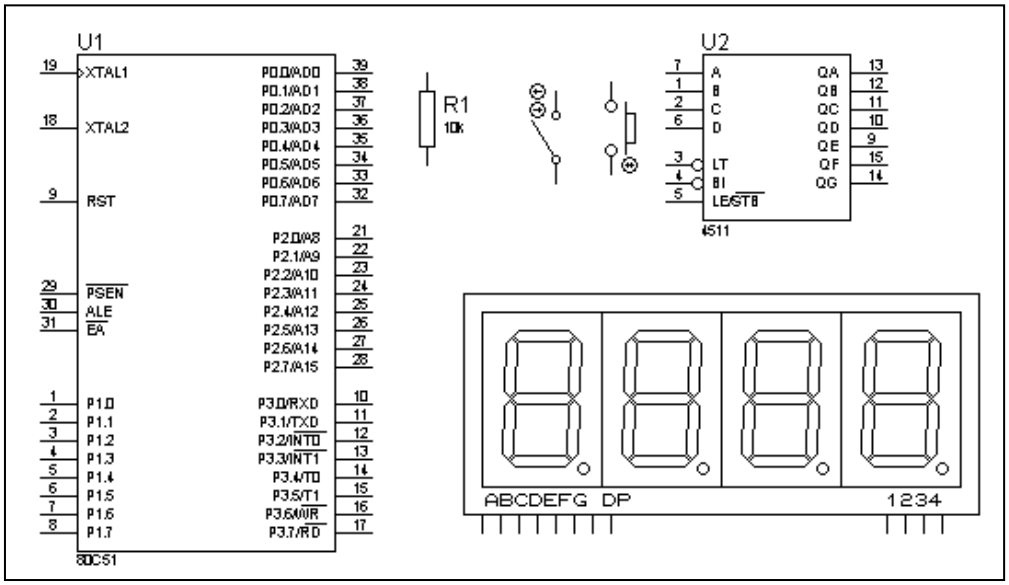


Figure 1: List of components

Switch is used to start the counting once it is pressed and reset the counter once it is released. Push button is used to replace the digital sensor that will generate pulse when the money is detected. Seven-segment displays the current quantity of the money counted. The 4511 is used as the decoder for the seven-segment. 8051 microcontroller is used as the controller of the system.

Based on the information above, answer the following question:

(a) Explain briefly the different concept between timer and counter in 8051 microcontroller.

(4 marks)

(b) Using Timer 1 and mode 2 as counter, design the circuit of the system (you do not need to design the basic circuit for 8051 such as the crystal, capacitor etc).

(4 marks)

(c) Based on Question 5(b), write the program for the system.

(12 marks)

Question 6

Figure 2 shows a pulley system that is used to transport goods from bottom to up level and vice versa. A $5V_{DC}$ motor, M1 is used on the pulley 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 S1 switch is pushed, the motor M1 will rotate clock wise and the goods will be sent down until S4 is triggered and the motor M1 will stop. However, when S2 switch is pushed, the motor M1 will rotate counter clock wise and the goods will be sent up. Once the goods reach the maximum high position, the limit switch, S3 will be triggered and the motor will stop.

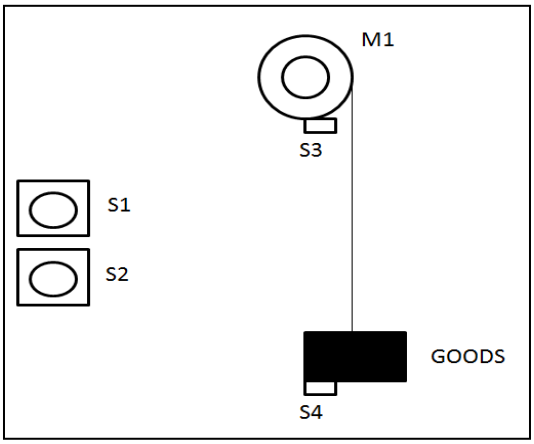


Figure 2: Pulley system to transfer goods vertically

(a) Based on Figure 2, design the circuit of the system. Clearly state which pins are being used. Motor is connected to 8051 microcontroller directly without using any motor driver.

(4 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.

(12 marks)

Question 7

Based on the Figure 3, answer the following questions:

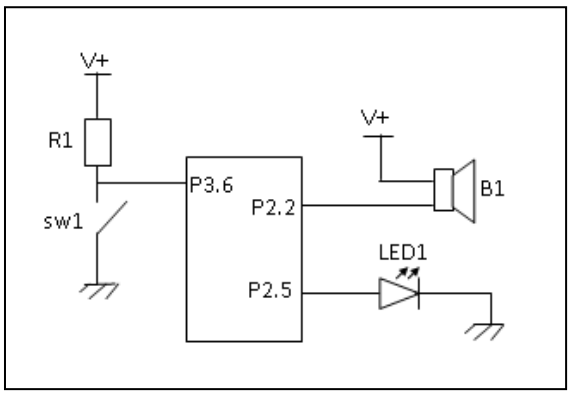


Figure 3: Input/output Connection

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

(4 marks)

(b) In this system, LED1 is blinking continuously while buzzer B1 will be on for a fraction second after SW1 is pressed. Write a program for this system using polling method for switch SW1 and buzzer B1, while using timer interrupt for LED1 to generate the longest delay possible of mode 1.

(10 marks)

(c) Modify the connection in Figure 3 and the code in 7(b) to allow using external interrupt for switch SW1.

(6 marks)

Question 8

A conveyor system with two parallel motors is being connected to 8051 microcontroller using L293D motor driver. The conveyor is capable to receive commands through serial communication via 8051 microcontroller. There a three (3) commands which are available, they are MOVE FORWARD, MOVE REVERSE and STOP. Each command is represented by an ASCII character which is 'a', 'b' and 'c' respectively.

(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.

(4 marks)

(b) Two (2) 12V_{DC} motors are being connected parallel to the microcontroller via L293D motor driver (Figure 4). You are ONLY allowed to use two microcontroller's pins which are P2.0 for the direction and P2.1 as motor enabler. Draw the electronic circuit diagram for the system.

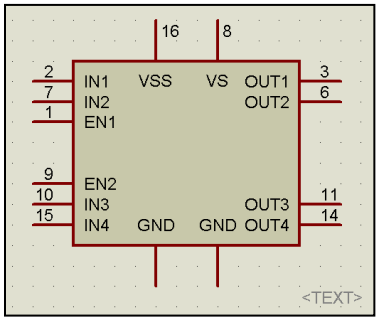


Figure 4: L293D proteus design

(4 marks)

(c) Create a program for the system above based on the circuit designed in Question 8(a) and 8(b).

(12 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			

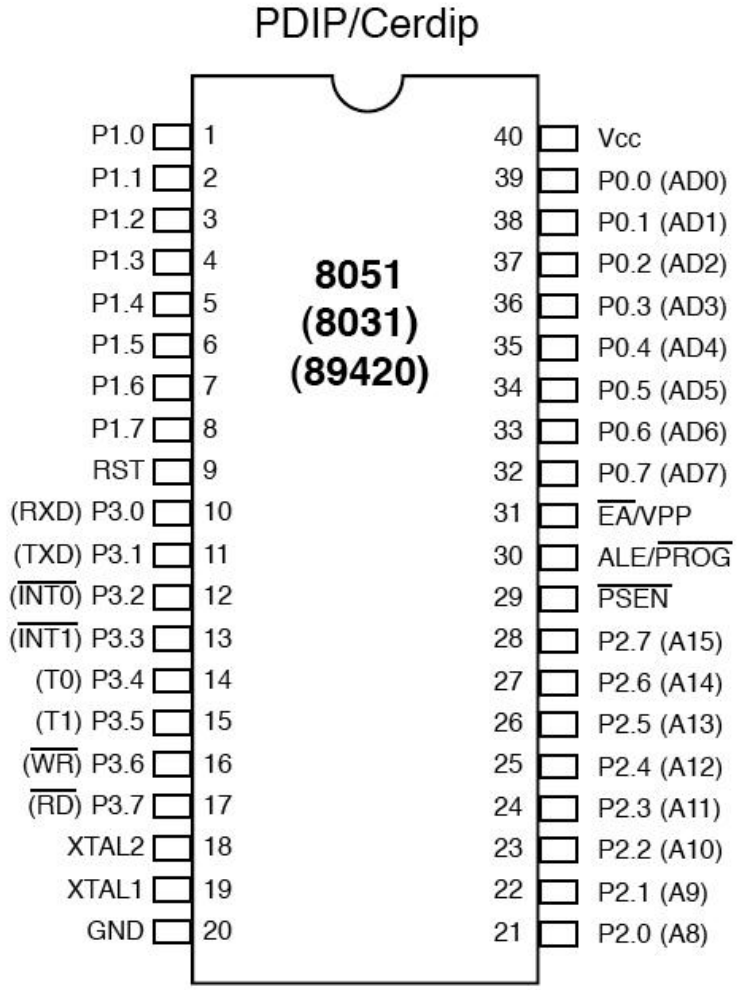
LOGICAL OPERATIONS

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



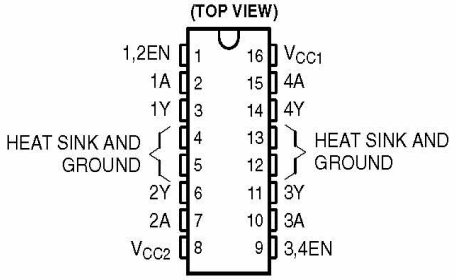
APPENDIX 3
L293D Data Specification

L293, L293D
QUADRUPLE HALF-H DRIVERS

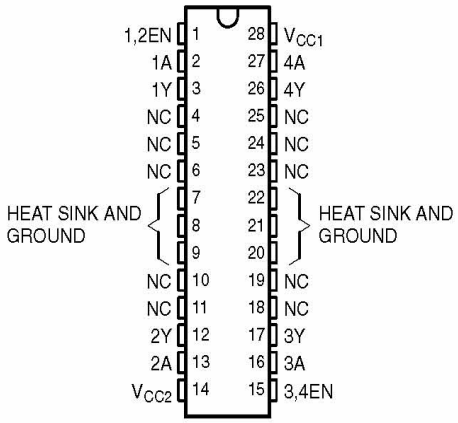
SLRS008C – SEPTEMBER 1986 – REVISED NOVEMBER 2004

- **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.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

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 $\overline{EL}$.