EE 2920

Dr. Johnson

Midterm Exam

No calculators, notes, …

NAME:____

1 – Write each of the following numbers using the designated representation you must show your work .	10pts
2 – Evaluate each of the following expressions	10pts
3 – Provide the value of a,b,c after executing the following code snippits. evaluate each problem individually	10pts
4 – Given the following memory map – evaluate each item	10pts
5 – Provide the final values after executing the following code snippit	15pts
6 – Write a snippit of code … I/O	20pts
7 – Clock setup, code question	15pts
8 - Given a x bit D/A	10pts

See next 2 pages

Table 5-5. CSCTL1 Register Description

Bit	Field	Type	Reset	Description
31	Reserved	R	0h	Reserved. Always reads as 0.
30-28	DIVS	RW	0h	SMCLK source divider. 000b = f(SMCLK)/1 001b = f(SMCLK)/2 010b = f(SMCLK)/4 011b = f(SMCLK)/8 100b = f(SMCLK)/16 101b = f(SMCLK)/32 110b = f(SMCLK)/64 111b = f(SMCLK)/128
27	Reserved	R	0h	Reserved. Always reads as 0.
26-24	DIVA	RW	0h	ACLK source divider. 000b = f(ACLK)/1 001b = f(ACLK)/2 010b = f(ACLK)/4 011b = f(ACLK)/8 100b = f(ACLK)/16 101b = f(ACLK)/32 110b = f(ACLK)/64 111b = f(ACLK)/128
23	Reserved	R	0h	Reserved. Always reads as 0.
22-20	DIVHS	RW	0h	HSMCLK source divider. 000b = f(HSMCLK)/1 001b = f(HSMCLK)/2 010b = f(HSMCLK)/4 011b = f(HSMCLK)/8 100b = f(HSMCLK)/16 101b = f(HSMCLK)/32 110b = f(HSMCLK)/64 111b = f(HSMCLK)/128
19	Reserved	R	0h	Reserved. Always reads as 0.
18-16	DIVM	RW	0h	MCLK source divider. 000b = f(MCLK)/1 001b = f(MCLK)/2 010b = f(MCLK)/4 011b = f(MCLK)/8 100b = f(MCLK)/16 101b = f(MCLK)/32 110b = f(MCLK)/64 111b = f(MCLK)/128

Bit	Field	Type	Reset	Description
12	SELB	RW	0h	Selects the BCLK source. 0b = LFXTCLK 1b = REFOCLK
11	Reserved	R	0h	Reserved. Always reads as 0.
10-8	SELA	RW	0h	Selects the ACLK source. 000b = LFXTCLK 001b = VLOCLK 010b = REFOCLK 011b-111b = Reserved for future use. Defaults to REFOCLK. Not recommended for use to ensure future compatibilities.
7	Reserved	R	0h	Reserved. Always reads as 0.
6-4	SELS	RW	3h	Selects the SMCLK and HSMCLK source. 000b = LFXTCLK 001b = VLOCLK 010b = REFOCLK 011b = DCOCLK 100b = MODOSC 101b = HFXTCLK 110b-111b = Reserved for future use. Defaults to DCOCLK. Not recommended for use to ensure future compatibilities.
3	Reserved	R	0h	Reserved. Always reads as 0.
2-0	SELM	RW	3h	Selects the MCLK source. 000b = LFXTCLK 001b = VLOCLK 010b = REFOCLK 011b = DCOCLK 100b = MODOSC 101b = HFXTCLK 110b-111b = Reserved for future use. Defaults to DCOCLK. Not recommended for use to ensure future compatibilities.

Figure 5-11. CSSTAT Register

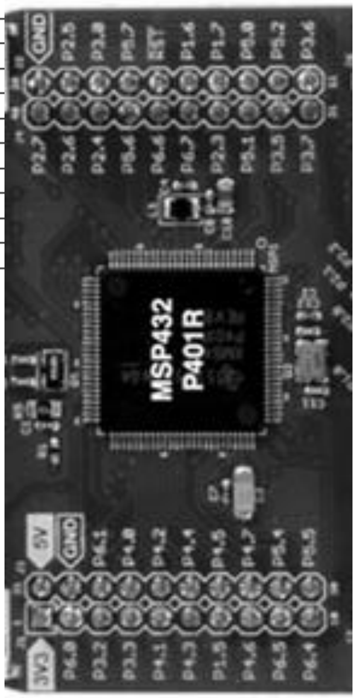
31	30	29	28	27	26	25	24
Reserved			BCLK_READY	SMCLK_READY	HSMCLK_READY	MCLK_READY	ACLK_READY
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
23	**22**	**21**	**20**	**19**	**18**	**17**	**16**
REFOCLK_ON	LFXTCLK_ON	VLOCLK_ON	MODCLK_ON	SMCLK_ON	HSMCLK_ON	MCLK_ON	ACLK_ON
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Reserved							
r-0	r-0	r-0	r-0	r-0	r-0	r-0	r-0
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
REFO_ON	LFXT_ON	VLO_ON	MODOSC_ON	Reserved	HFXT_ON	DCOBIAS_ON	DCO_ON
r-0	r-0	r-0	r-0	r-0	r-0	r-1	r-1

Table 2-25. NVIC Registers

Offset	Acronym	Register Name	Type	Reset
100h	ISER0	Irq 0 to 31 Set Enable Register	read-write	00000000h
104h	ISER1	Irq 32 to 63 Set Enable Register	read-write	00000000h
180h	ICER0	Irq 0 to 31 Clear Enable Register	read-write	00000000h
184h	ICER1	Irq 32 to 63 Clear Enable Register	read-write	00000000h
200h	ISPR0	Irq 0 to 31 Set Pending Register	read-write	00000000h

NVIC->ISER[0]
…
NVIC->ICER[0]
…
NVIC->ISPR[0]
…
NVIC->ICPR[0]
…
NVIC->IABR[0]
…
NVIC->IP[0]
…
NVIC->IP[63]

NVIC INTERRUPT INPUT	SOURCE	FLAGS IN SOURC
INTISR[25]	Timer32_INT1	Timer32 Interrupt for Timer1
INTISR[26]	Timer32_INT2	Timer32 Interrupt for Timer2
INTISR[27]	Timer32_INTC	Timer32 Combined Interrupt
INTISR[28]	AES256	AESRDYIFG
INTISR[29]	RTC_C	OFIFG, RDYIFG, TEVIFG, AIFG, RT0PSIFG, RT1PSIFG
INTISR[30]	DMA_ERR	DMA error interrupt
INTISR[31]	DMA_INT3	DMA completion interrupt3
INTISR[32]	DMA_INT2	DMA completion interrupt2
INTISR[33]	DMA_INT1	DMA completion interrupt1
INTISR[34]	DMA_INT0[(3)]	DMA completion interrupt0
INTISR[35]	I/O Port P1	P1IFG.x (x = 0 through 7)
INTISR[36]	I/O Port P2	P2IFG.x (x = 0 through 7)
INTISR[37]	I/O Port P3	P3IFG.x (x = 0 through 7)
INTISR[38]	I/O Port P4	P4IFG.x (x = 0 through 7)
INTISR[39]	I/O Port P5	P5IFG.x (x = 0 through 7)
INTISR[40]	I/O Port P6	P6IFG.x (x = 0 through 7)
INTISR[41]	Reserved	



Px->DS – Drive Strength Register
"0" for regular strength,
"1" for high drive strength
Px->SEL0, Px->SEL1
Mode Select Register

PxSEL1	PxSEL0	I/O Function
0	0	General purpose I/O is selected
0	1	Primary module function is selected
1	0	Secondary module function is selected
1	1	Tertiary module function is selected

Px->IE – Interrupt Enable Register
"1" for enable
Px->IES – Interrupt Edge Select Register
"0" for low → high, "1" for high → low
Px->IFG – Interrupt Flag Register
Set to "1" on selected edge transition
Px->IV – Interrupt Vector Register

C – Operator Precedence

Precedence	Operator	Description	Associativity
1	++ --	Suffix/postfix increment and decrement	Left-to-right
	()	Function call	
	[]	Array subscripting	
	.	Structure and union member access	
	->	Structure and union member access through pointer	
	(*type*){*list*}	Compound literal(C99)	
2	++ --	Prefix increment and decrement	Right-to-left
	+ -	Unary plus and minus	
	! ~	Logical NOT and bitwise NOT	
	(*type*)	Type cast	
	*	Indirection (dereference)	
	&	Address-of	
	sizeof	Size-of	
	_Alignof	Alignment requirement(C11)	
3	* / %	Multiplication, division, and remainder	Left-to-right
4	+ -	Addition and subtraction	
5	<< >>	Bitwise left shift and right shift	
6	< <=	For relational operators < and ≤ respectively	
	> >=	For relational operators > and ≥ respectively	
7	== !=	For relational = and ≠ respectively	
8	&	Bitwise AND	
9	^	Bitwise XOR (exclusive or)	
10	|	Bitwise OR (inclusive or)	
11	&&	Logical AND	
12	||	Logical OR	
13	?:	Ternary conditional	Right-to-Left
14	=	Simple assignment	
	+= -=	Assignment by sum and difference	
	*= /= %=	Assignment by product, quotient, and remainder	
	<<= >>=	Assignment by bitwise left shift and right shift	
	&= ^= |=	Assignment by bitwise AND, XOR, and OR	
15	,	Comma	Left-to-right