Nom i Cognoms: _____

Computer Interfacing. Primer parcial, 9/10/2017

Responeu les preguntes a l'espai reservat. Justifiqueu les respostes.
Poseu el nom a tots els fulls. Totes les preguntes valen 1 punt.
(Recordeu, Destí: W=WORKING, F= FILE, Bank: A=ACCESS, B=BSR)

1. Volem ampliar la memòria de programa del PIC184550 de 32K a 128K. Quines modificacions haurem de fer a l'arquitectura del PIC per tal de que tot funcioni correctament?

2. El següent programa en *assembler* del PIC18F4550 implementa una suma de dos enters de 24 bits emmagatzemats en les posicions de memòria 0x10-0x12 i 0x13-0x15 respectivament i guarda el resultat en la posicions de memòria 0x20-0x22 . Emplena els forats que falten.

```
#include <p18F8720.inc>
ORG 0x00
GOTO
ORG 0x08
RETFIE
ORG 0x18
RETFIE
start: MOVF 0x10 , W, A
ADDWF _____, W, A
MOVWF 0x20 , A
MOVF 0x11 , W, A
_____ 0x14 , W, A
MOVWF 0x21 , A
MOVF 0x12 , W, A
_____ 0x15 , W, A
MOVWF _____, A
End
```

3. Calcula de temps que triga en executar-se la següent subrutina en un PIC18F4550 funcionant amb un oscil·lador a 4MHz.

```
Delay:
MOVLW d'250'
Loop:
ADDLW 0xFF
BTFSS STATUS, Z
BRA Loop
RETURN
```

Nom i Cognoms: _____

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
BYTE-ORIENTED OPERATIONS						
ADDWF	f, d, a	Add WREG and f	1	0010 01da ffff ffff	C, DC, Z, OV, N	1, 2
ADDWFC	f, d, a	Add WREG and Carry bit to f	1	0010 00da ffff ffff	C, DC, Z, OV, N	1, 2
ANDWF	f, d, a	AND WREG with f	1	0001 01da ffff ffff	Z, N	1,2
CLRF	f, a	Clear f	1	0110 101a ffff ffff	Z	2
COMF	f, d, a	Complement f	1	0001 11da ffff ffff	Z, N	1, 2
CPFSEQ	f, a	Compare f with WREG, Skip =	1 (2 or 3)	0110 001a ffff ffff	None	4
CPFSGT	f, a	Compare f with WREG, Skip >	1 (2 or 3)	0110 010a ffff ffff	None	4
CPFSLT	f, a	Compare f with WREG, Skip <	1 (2 or 3)	0110 000a ffff ffff	None	1, 2
DECF	f, d, a	Decrement f	1	0000 01da ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
DECFSZ	f, d, a	Decrement f, Skip if 0	1 (2 or 3)	0010 11da ffff ffff	None	1, 2, 3, 4
DCFSNZ	f, d, a	Decrement f, Skip if Not 0	1 (2 or 3)	0100 11da ffff ffff	None	1, 2
INCF	f, d, a	Increment f	1	0010 10da ffff ffff	C, DC, Z, OV, N	1, 2, 3, 4
INCFSZ	f, d, a	Increment f, Skip if 0	1 (2 or 3)	0011 11da ffff ffff	None	4
INFSNZ	f, d, a	Increment f, Skip if Not 0	1 (2 or 3)	0100 10da ffff ffff	None	1, 2
IORWF	f, d, a	Inclusive OR WREG with f	1	0001 00da ffff ffff	Z, N	1, 2
MOVF	f, d, a	Move f	1	0101 00da ffff ffff	Z, N	1
MOVFF	f_s, f_d	Move f_s (source) to 1st word f_d (destination) 2nd word	2	1100 ffff ffff ffff 1111 ffff ffff ffff	None	
MOVWF	f, a	Move WREG to f	1	0110 111a ffff ffff	None	
MULWF	f, a	Multiply WREG with f	1	0000 001a ffff ffff	None	1, 2
NEGF	f, a	Negate f	1	0110 110a ffff ffff	C, DC, Z, OV, N	
RLCF	f, d, a	Rotate Left f through Carry	1	0011 01da ffff ffff	C, Z, N	1, 2
RLNCF	f, d, a	Rotate Left f (No Carry)	1	0100 01da ffff ffff	Z, N	
RRCF	f, d, a	Rotate Right f through Carry	1	0011 00da ffff ffff	C, Z, N	
RRNCF	f, d, a	Rotate Right f (No Carry)	1	0100 00da ffff ffff	Z, N	
SETF	f, a	Set f	1	0110 100a ffff ffff	None	1, 2
SUBFWB	f, d, a	Subtract f from WREG with Borrow	1	0101 01da ffff ffff	C, DC, Z, OV, N	
SUBWF	f, d, a	Subtract WREG from f	1	0101 11da ffff ffff	C, DC, Z, OV, N	1, 2
SUBWFB	f, d, a	Subtract WREG from f with Borrow	1	0101 10da ffff ffff	C, DC, Z, OV, N	
SWAPF	f, d, a	Swap Nibbles in f	1	0011 10da ffff ffff	None	4
TSTFSZ	f, a	Test f, Skip if 0	1 (2 or 3)	0110 011a ffff ffff	None	1, 2
XORWF	f, d, a	Exclusive OR WREG with f	1	0001 10da ffff ffff	Z, N	

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
LITERAL OPERATIONS						
ADDLW	k	Add Literal and WREG	1	0000 1111 kkkk kkkk	C, DC, Z, OV, N	
ANDLW	k	AND Literal with WREG	1	0000 1011 kkkk kkkk	Z, N	
IORLW	k	Inclusive OR Literal with WREG	1	0000 1001 kkkk kkkk	Z, N	
LFSR	f, k	Move Literal (12-bit) 2nd word to FSR(f) 1st word	2	1110 1110 00ff kkkk 1111 0000 kkkk kkkk	None	
MOVLB	k	Move Literal to BSR<3:0>	1	0000 0001 0000 kkkk	None	
MOVLW	k	Move Literal to WREG	1	0000 1110 kkkk kkkk	None	
MULLW	k	Multiply Literal with WREG	1	0000 1101 kkkk kkkk	None	
RETLW	k	Return with Literal in WREG	2	0000 1100 kkkk kkkk	None	
SUBLW	k	Subtract WREG from Literal	1	0000 1000 kkkk kkkk	C, DC, Z, OV, N	
XORLW	k	Exclusive OR Literal with WREG	1	0000 1010 kkkk kkkk	Z, N	
DATA MEMORY ↔ PROGRAM MEMORY OPERATIONS						
TBLRD*		Table Read	2	0000 0000 0000 1000	None	
TBLRD*+		Table Read with Post-Increment		0000 0000 0000 1001	None	
TBLRD*-		Table Read with Post-Decrement		0000 0000 0000 1010	None	
TBLRD+*		Table Read with Pre-Increment		0000 0000 0000 1011	None	
TBLWT*		Table Write	2	0000 0000 0000 1100	None	
TBLWT*+		Table Write with Post-Increment		0000 0000 0000 1101	None	
TBLWT*-		Table Write with Post-Decrement		0000 0000 0000 1110	None	
TBLWT+*		Table Write with Pre-Increment		0000 0000 0000 1111	None	

Nom i Cognoms: _____

4. Quines dues instruccions del PIC18F utilitzaríeu per passar dades de la ROM a la RAM?

5. Convertiu a C el següent programa fet en *assembler:*

```
                CLRF _x, A
                CLRF _i, A
        Loop:
                INCF    _i,  F, A
                MOVF    _i,  W, A
                ADDWF   _x, F, A
                MOVLW   9
                CPFSGT   _i, A
                GOTO     Loop
```

6. Quina funció implementa el següent codi?

```
                MOVLW       0
                MOVFF _     var1,_var2
                ADDWF       _var1, W, A
                BNN         final
                NEGF        _var2, A
        final:  ...
```

Nom i Cognoms: _____

TABLE 26-2: PIC18FXXXX INSTRUCTION SET (CONTINUED)

Mnemonic, Operands		Description	Cycles	16-Bit Instruction Word MSb LSb	Status Affected	Notes
BIT-ORIENTED OPERATIONS						
BCF	f, b, a	Bit Clear f	1	1001 bbba ffff ffff	None	1, 2
BSF	f, b, a	Bit Set f	1	1000 bbba ffff ffff	None	1, 2
BTFSC	f, b, a	Bit Test f, Skip if Clear	1 (2 or 3)	1011 bbba ffff ffff	None	3, 4
BTFSS	f, b, a	Bit Test f, Skip if Set	1 (2 or 3)	1010 bbba ffff ffff	None	3, 4
BTG	f, d, a	Bit Toggle f	1	0111 bbba ffff ffff	None	1, 2
CONTROL OPERATIONS						
BC	n	Branch if Carry	1 (2)	1110 0010 nnnn nnnn	None	
BN	n	Branch if Negative	1 (2)	1110 0110 nnnn nnnn	None	
BNC	n	Branch if Not Carry	1 (2)	1110 0011 nnnn nnnn	None	
BNN	n	Branch if Not Negative	1 (2)	1110 0111 nnnn nnnn	None	
BNOV	n	Branch if Not Overflow	1 (2)	1110 0101 nnnn nnnn	None	
BNZ	n	Branch if Not Zero	1 (2)	1110 0001 nnnn nnnn	None	
BOV	n	Branch if Overflow	1 (2)	1110 0100 nnnn nnnn	None	
BRA	n	Branch Unconditionally	2	1101 0nnn nnnn nnnn	None	
BZ	n	Branch if Zero	1 (2)	1110 0000 nnnn nnnn	None	
CALL	n, s	Call Subroutine 1st word 2nd word	2	1110 110s kkkk kkkk 1111 kkkk kkkk kkkk	None	
CLRWDT	—	Clear Watchdog Timer	1	0000 0000 0000 0100	$\overline{TO}$, $\overline{PD}$	
DAW	—	Decimal Adjust WREG	1	0000 0000 0000 0111	C	
GOTO	n	Go to Address 1st word 2nd word	2	1110 1111 kkkk kkkk 1111 kkkk kkkk kkkk	None	
NOP	—	No Operation	1	0000 0000 0000 0000	None	
NOP	—	No Operation	1	1111 xxxx xxxx xxxx	None	4
POP	—	Pop Top of Return Stack (TOS)	1	0000 0000 0000 0110	None	
PUSH	—	Push Top of Return Stack (TOS)	1	0000 0000 0000 0101	None	
RCALL	n	Relative Call	2	1101 1nnn nnnn nnnn	None	
RESET		Software Device Reset	1	0000 0000 1111 1111	All	
RETFIE	s	Return from Interrupt Enable	2	0000 0000 0001 000s	GIE/GIEH, PEIE/GIEL	
RETLW	k	Return with Literal in WREG	2	0000 1100 kkkk kkkk	None	
RETURN	s	Return from Subroutine	2	0000 0000 0001 001s	None	
SLEEP	—	Go into Standby mode	1	0000 0000 0000 0011	$\overline{TO}$, $\overline{PD}$	

Nom i Cognoms: _____

7. El mode Banked i l'Access Bank són necessaris perquè a la majoria d'instruccions tenim 8 bits per codificar el paràmetre (file) i la memòria de dades s'ha d'adreçar amb 12 bits.

Què caldria canviar a l'arquitectura del PIC per fer que totes les instruccions existents puguin tenir un paràmetre de 12 bits (no volem modificar el joc d'instruccions).

Marca amb un cercle **totes** les opcions certes (cada encert suma 0,5, cada error resta 0,33):

(a) l'amplada de l'Instruction Register (IR) (ara només té 16 bits).

(b) l'amplada de la pila del sistema (ara només té 20 bits).

(c) el nombre de nivells de la pila del sistema (ara només té 31 nivells).

(d) la mida de paraula de la memòria de programa (ara només té 16 bits).

(e) la freqüència del processador (al tenir més bits d'adreçament pot anar més ràpid).

8. Perquè és aconsellable fer servir la instrucció BRA en comptes del GOTO pels salts incondicionals on l'adreça de destí sigui propera?

9. En el mode indirecte ens cal un registre especial per fer d'apuntador a la memòria de dades. Com podem posar un valor a aquest registre per inicialitzar aquest punter?

10. Perquè en el joc d'instruccions del PIC hi ha dues instruccions NOP; una codificada amb tot 0 i l'altra codificada amb quatre 1 a l'inici?