Nom i Cognoms:_____Una posible solució_____

1) Raoneu si les següents afirmacions són certes o falses:

- Les memòries estàtiques no necessiten d'alimentació per a mantenir la informació.

Fals. Si són volàtils necessiten alimentació per mantenir les dades.

- El principal inconvenient de les memòries dinàmiques és que cada cel.la ocupa molta superfície.

Fals. Les memòries dinàmiques permeten una alta escala d'integració. De fet només cal una capacitat per a implementar cada cel.la.

2) Per què les instruccions de salt condicional triguen un cicle en executar-se si la condició és falsa però 2 cicles si la condició és certa?

Quan la condició és certa, es provoca un salt. En aquest cas, la següent instrucció a executar no és la que es troba ja carregada en el pipe-line, sino que s'ha de fer un nou fetch per a anar a buscar la instrucció correcta. Per tant, s'ha de buidar el pipe-line i executar un nop mentre es fa el fetch de la següent instrucció.

3) En comptes d'executar una instrucció a cada cicle de rellotge, el PIC18F4550 ha d'esperar 4 cicles de rellotge per tenir un cicle d'instrucció. Per quin motiu?

LA CPU necessita quatre fases per a executar una instrucció. (concretament: decodificació, cerca d'operands, operació i desar resultats)

4) Un enginyer de telecomunicacions ha fet el següent codi per omplir el Bank0 amb termes de la sèrie de Fibonacci:

```
        MOVLW 1              ;posem el Wreg a 1
        MOVWF 0,A            ;primer element a 1
        MOVWF 1,A            ;segon element a 1
        LFSR 0,0             ;FSR0  apunta a 0
        LFSR 1,1             ;FSR1  apunta a 1
        LFSR 2,2             ;FSR2  apunta a 2
bucle:  MOVF POSTINC0,W,A    ;posem element k-2 al Wreg, incrementa índex
        ADDWF POSTINC1,W,A    ;hi afegim element k-1, incrementa índex
        MOVWF POSTINC2,A     ;posem el resultat a l'element k, incrementa índex
        TSTFSZ   FSRL2,A       ;quan arribem a 256 elements, sortim
        BRA bucle
```

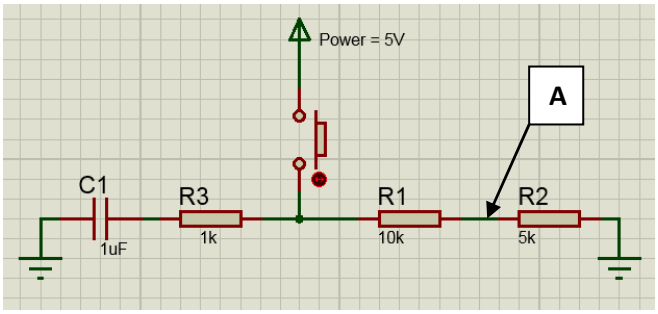
Quants termes correctes calcularà amb el PIC18f4550? Justifica la resposta.

Els termes de la sèrie de Fibonacci segueixen la seqüència:

1,1,2,3,5,8,13,21,34,55,89,144,233,377...

Donat que els registres, la ALU i el bus de dades del PIC18F4550 són de 8 bits, el terme de valor 377 ja no el podrem representar.

5) Observeu el següent esquema electrònic fet amb Proteus. En temps t=0 es prem el botó. Si no s'ha deixat anar el botó, quina tensió hi haurà al punt A en temps t= 0.16 segons?



Mentre el botó estigui premut, tindrem dos circuits independents:

- un amb la R i la C que seguirà l'equació de càrrega del condensador, però que no ens interessa per res.
- Un altre amb un divisor de tensió i que involucra el punt A. La tensió al punt A serà:

$$V_A = 5V \cdot (5KOhm / (5\ Kohm + 10\ Kohm)) = 1{,}66\ Volts$$

Nom i Cognoms:_____**Una posible solució**_____

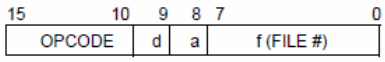
6) Omple la taula de registres següent amb els valors resultants de l'execució d'aquest codi (la h indica que el valor expressat és hexadecimal):

File	Initvalue	Final value
WREG	00h	12h
0, BANK 0	07h	19h
1, BANK 0	12h	12h
0, BANK 1	11h	19h
1, BANK 1	23h	23h

```
MOVF    01h,0, 0      // posició 1 de l'access va al Wreg: Wreg=12h
ADDWF   00h, 1, 0     // sumem el Wreg amb la posició 0 de l'access i ho
                      //  guardem a la posició 0 de l'access:
                      // @000=19h,Wreg=12h
MOVFF   000h, 100h    // movem el contingut de la posició 0 del bank 0 a la
                      //  posició 0 del bank 1: @100=19h
```

GENERAL FORMAT FOR INSTRUCTIONS

Byte-oriented file register operations — **Example Instruction**




```
ADDWF MYREG, W, B
```

d = 0 for result destination to be WREG register
d = 1 for result destination to be file register (f)
a = 0 to force Access Bank
a = 1 for BSR to select bank
f = 8-bit file register address

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 1111	ffff ffff	ffff 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	

7) Per la **secció de codi assemblador en negreta** de la part dreta, calcula el seu temps d'execució tenim present que la freqüència de l'oscil·lador del micro és de 1 MHz.

El bucle interior (loop1) consta de 9 instruccions que s'executen en 10 cicles. (excepte l'ultima volta que s'executa en 9 cicles). Count1=250 voltes (bucle interior).
El bucle interior triga: 249*10 + 9 cicles= 2499 cicles.

El bucle exterior (loop0), a més del bucle interior, consta de 4 instruccions que s'executen en 5 cicles. Count2=100 voltes (bucle exterior).
El bucle exterior triga: 100*(2499 + 5) cicles = 250400 cicles

Com la freqüència de l'oscil·lador és de 1MHz, el Tcicle=4µs.
Texecució= 250400*4 µs= 1.0016 s.

8) Indiqueu el nombre total de bytes en memòria de programa que ocupa la totalitat del codi de la part dreta en un PIC 18F.

Al codi adjunt tenim 14 instruccions de 16 bits i 2 instruccions de 32 bits.
(Les directives d'assemblador, la definició de constants i les etiquetes és informació que s'utilitza en el procés d'assemblat i/o linkat però NO són instruccions que executi el microprocessador.)

El total de bytes necessaris per guardar el codi en memòria de programa d'un PIC18 és de 36 bytes.

```
        ....
Count1 equ 0x20
Count2 equ 0x21
Delay1 equ d'250'
Delay2 equ d'100'

movlw Delay2
movff WREG, Count2

; *** negreta ***
loop0
        movlw Delay1
        movwf Count1, A
loop1
        nop
        nop
        nop
        nop
        nop
        nop
        nop
        decfsz  Count1, F, A
        bra loop1
        decfsz Count2, F, A
        GOTO loop0
; *** Fi negreta ***
        movlw d'0'
        .....
```

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	2	1110	110s	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
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	2	1110	1111	kkkk	kkkk	None	
		2nd word		1111	kkkk	kkkk	kkkk		
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,	