Projecte de Fi de Carrera
Enginyer Industrial

Sistema de monitorització remota de temperatures mitjançant un enllaç bidireccional de radiofreqüència

ANNEX A: Selecció de components
ANNEX B: Aspectes addicionals de hardware i software
ANNEX C: Càlculs i mesures
ANNEX D: Pressupost
ANNEX E: Impacte ambiental
ANNEX F: Codi de programació
ANNEX G: Catàlegs de components

Autor: Gabriel Torrens Caldentey
Director: Emili Lupón Rosés

Convocatòria: Març de 2004 (pla 94)

Escola Tècnica Superior
d'Enginyeria Industrial de Barcelona

Resum

En aquest volum d'annexos es recullen els continguts que per la seva extensió o naturalesa no ha estat possible incloure en el volum de la memòria.
S'hi inclouen diferents annexos amb especificacions i descripcions de parts del sistema, càlculs, i mesures experimentals. Un annex amb les més de 4000 línies de codi necessàries per a que el microcontrolador dugui a terme les tasques que permeten al sistema fer correctament la seva funció. I un últim annex amb el catàlegs d'alguns dels components utilitzats per si el lector necessita referir-s'hi en qualsevol moment.

Sumari

Volum 2: Annexos

Índex de figures

Índex de taules

A.- Selecció de components

En aquest annex es descriuen els diferents circuits avaluats per a l'elecció definitiva del sensor de temperatura utilitzat i del mòdul de radiofreqüència, així com el procés seguit per la seva selecció.

A.1.- Selecció del sensor de temperatura

Per a l'elecció final del sensor de temperatura utilitzat es varen tenir en compte diferents opcions. L'especificació més difícil d'aconseguir (i la que acabarà determinant la elecció definitiva del sensor) és que tingui una precisió relativament bona en un rang ampli de temperatures. En un principi s'havia pensat en poder obtenir una precisió en torn a 1ºC. El sensors avaluats es detallen a continuació:

AD22100K (Anolog Devices):

És un sensor de temperatura de sortida analògica, subministra una tensió proporcional a la temperatura, la qual cosa requereix un microcontrolador amb un conversor A/D, això no representa un problema greu ja que és un perifèric present en molts microcontroladors. La tensió que subministra varia entre 0,25 V i 4,75V, i per tant es podria entrar directament al conversor A/D sense necessitat d'amplificació. El principal desavantatge d'aquest sensor és la seva precisió que típicament és de ±0,75 ºC però que pot arribar a ser de ±2 ºC, la qual cosa és excessiva. Per aquest motiu va ser descartat.

AD590K (Analog Devices)

És un sensor de temperatura de sortida analògica que subministra un corrent proporcional a la temperatura. Subministra un corrent de 298,2μA a 25ºC que varia linealment amb la temperatura creixent 1μA/K. El problema que representa això és que es necessita convertir aquesta magnitud de corrent a tensió mitjançant una resistència que, per tal d'obtenir tensions fàcils de mesurar, ha de ser de l'ordre dels kΩ. D'aquesta manera, és fonamental utilitzar resistències amb una bona tolerància, el fabricant recomana que almenys s'utilitzin resistències de tolerància 1%. L'error que intrínsecament comet el sensor, si no es calibra, arriba fins a ±5,5ºC i amb calibració es pot reduir fins a ±2ºC. La magnitud d'aquests errors fa que es descarti aquest sensor.

AD592CN (Analog Devices)

És un sensor de sortida analògica que proporciona una sortida en corrent proporcional a la temperatura. Igual que l'anterior sensor, necessita convertir el senyal de corrent a tensió, però aquest presenta una millor precisió ja que a 25ºC té un error de només 0,5ºC. Tanmateix quan ens allunyem d'aquesta temperatura l'error creix essent de més d'1ºC per a temperatures més baixes. Per aquest motiu la precisió que ens ofereix no és suficient.

FM20 i FM50 i FM75 (Fairchild)

És un sensor de temperatura analògic que proporciona una sortida en tensió que representa la temperatura. La precisió que en ofereix és ±1ºC a 25ºC, però a 0ºC la precisió és ja de només d'uns ±2ºC. Aquesta pobra precisió fa que no sigui adient per a l'aplicació.

TC1046 (Microchip)

Es tracta d'un sensor de temperatura analògic que subministra una sortida en tensió proporcional a la temperatura. Presenta un error màxim de ±2ºC a 25ºC, fent-se més gran encara quan ens allunyem dels 25ºC. La present aplicació requereix major precisió, per aquest motiu el descartem.

TC77 (Microchip)

És un sensor digital que ens proporciona una sortida digital seriada amb un clock. La precisió és de ±1ºC a temperatures superiors a 25ºC, però per a temperatures inferiors a 25ºC la precisió es degrada fins a valors de ±2ºC i pitjors. Per aquest motiu, no es considera com a candidat.

TCN75 (Microchip)

És un sensor digital que proporciona una sortida seriada utilitzant el sistema *2-wire*. La precisió que ens ofereix a 25ºC és típicament de ±0,5ºC, però pot arribar a ser de ±3ºC, per temperatures més baixes el fabricant afirma que la precisió típica és de només ±3ºC. Per aquest motiu no el considerem en l'elecció final.

LM135 LM235 LM335 LM135A LM235A LM335A (National)

És un sensor analògic en tensió. Requereix calibració, ja que sense ella, la precisió fins i tot a 25ºC, no és gaire interessant (1ºC). El fabricant assegura que si es calibra a 25ºC la precisió pot arribar a 1ºC en tot el seu rang de mesura. La necessitat de calibració junt amb una precisió no excepcionalment bona tot i haver realitzat calibració fa que el descartem per a l'elecció final.

LM92 (National)

És un sensor digital que es comunica mitjançant el sistema *2-wire*. La precisió és la següent:

±0,33ºC a 30ºC

±0,5ºC entre 10ºC i 50ºC

±1ºC entre –10ºC i 85ºC

Fins ara és el sensor més precís que s'ha trobat, a temperatures properes als 30ºC té una precisió excel·lent, entre 10ºC i 50ºC, la precisió és prou bona, l'únic inconvenient és que la precisió es degrada a temperatures inferiors a10ºC. A la vista d'això decidim considerar-lo per a l'elecció final.

SMT160-30 (Smartec)

És un sensor digital que proporciona el valor de la temperatura codificat en el duty-cicle d'una forma d'ona quadrada. És a dir que mitjançant la modulació d'aquesta ona aconsegueix que un microcontrolador sigui capaç de llegir la informació digital continguda en ella, sense haver d'utilitzar un conversor A/D. A més a més, l'ona modulada d'aquesta manera conserva, en el seu valor mitjà, un valor (analògic) de tensió proporcional a la temperatura.

La precisió d'aquest sensor no és gaire interessant en principi, ja que és de ±0,7ºC, el que la fa digne de menció és que el fabricant assegura que es manté sobre un rang molt ampli (des de -30ºC fins a 100ºC). El principal problema d'aquest sensor és que no es va poder esbrinar cap manera fàcil d'aconseguir-lo, per aquest motiu, en principi, no el tindrem en compte per a l'elecció final.

DS1620 (Dallas-Maxim Semiconductors)

És un sensor digital que es comunica seriadament mitjançant el sistema 3-wire (clock, dades i reset). La precisió que ofereix és molt interessant per a temperatures compreses entre 0ºC i 70ºC, ja que proporciona ±0,5ºC d'error màxim. Per a temperatures inferiors a 0ºC la precisió cau fins a valors de prop de ±2ºC a –10ºC. Tot i la falta de precisió a temperatures baixes, el bon comportament a temperatures altes ens fa tenir-lo en compte per a l'elecció final.

DS1821 (Dallas-Maxim Semiconductors)

És un sensor de sortida digital amb comunicació sèrie mitjançant un sistema exclusiu de Dallas-Maxim Semiconductors anomenat *1-wire*®. Aquest sistema utilitza un sola línia de comunicació per a enviar i rebre dades sèrie. Aquest protocol és el mateix que utilitza el sensor DS18B20, per a més informació veure la discussió del sensor DS18B20.

La precisió que ofereix el DS1821 és de ±1ºC en un interval que va des de -35ºC fins a 125ºC. Aquesta precisió relativament bona en un rang tan ampli de temperatures ens fa tenir-lo en compte per a l'elecció final.

DS18B20 (Dallas-Maxim Semiconductors)

És un sensor de sortida digital. Estableix una comunicació sèrie utilitzant un sistema exclusiu de Dallas_Maxim Semiconductors anomenat *1-wire*®. Aquest sistema utilitza una sola línia de comunicació per a establir una comunicació bidireccional sèrie. El principal avantatge d'aquest sistema és que requereix una sola línia de comunicació i que, a més, permet que diferents sensors la comparteixin. Això representa un clar avantatge en quan a estalvi de cable si es desitja instal·lar una quantitat important de sensors allunyats uns dels altres. En el present projecte no serà possible aprofitar aquest fet, ja que en cada unitat remota de mesura de temperatura disposem d'un sol sensor, que a més a més, es troba a molt poca distància del microcontrolador.

El fet que el sensor utilitzi aquest protocol de comunicació, en el nostre cas representa un desavantatge, ja que és un protocol difícil d'implementat en un microcontrolador ja que per a transmetre la informació de la temperatura s'han de transmetre gran quantitat de dades complementàries, i a més els *timings* del protocol són crítics.En quant a la precisió que ofereix el sensor, aquesta és molt interessant, ja que des de –10ºC fins a 85ºC proporciona ±0,5ºC de

precisió, si sortim d'aquest rang la precisió cau fins a ±2ºC. La bona precisió que ofereix aquest sensor fa que el tinguem en compte per a l'elecció final.

Les principals característiques ja esmentades, junt amb altres característiques dels sensors avaluats fins ara, es poden veure en la següent taula:

sensor	**analògic / digital**	***sortida***	**precisió A 25ºC (ºC)**	**precisió a 70ºC (ºC)**	**precisió a –10ºC (ºC)**	**rang mesura (ºC)**	**tensions aliment. (V)**
AD22100K	analògic	tensió	±2	±2	±2	(-50,150)	(4,6)
AD590K	analògic	corrent	0 (*)	±2 (*)	±2 (*)	(-55,150)	(4,30)
AD592CN	analògic	corrent	0.5	0.8	1	(-25,105)	(4,30)
FM20/50/75	analògic	tensió	±1	±2.5	±1.5	(-55.130)	(2.4,6)
TC1046	analògic	tensió	±2	±3	-	(-40,125)	(2.7,4.4)
TC77	digital	sèrie	±1	±2	±3	(-40,125)	(2.7,5.5)
TCN75	digital	sèrie	±0.5	±3	±3	(-55,125)	(2.7,5.5)
LM135...335	analògic	tensió	0 (*)	1 (*)	1 (*)	(-55,150)	-
LM92	digital	sèrie	±0,33	±1	±1	(-55,150)	(2.7,5.5)
SMT160-30	digital	duty-cicle	±0.7	±0.7	±0.7	(-45,130)	(4.75,7)
DS1620	digital	sèrie	±0,5	±0,5	±2	(-55,125)	(2.7,5.5)
DS1821	digital	sèrie	±1	±1	±1	(-55,125)	(2.7,5.5)
DS18B20	digital	sèrie	±0,5	±0,5	±0,5	(-55,125)	(3,5.5)

Taula A.1. Comparativa de les característiques dels diferent sensors avaluats

(*) Mitjançant calibració a 25ºC.

D'entre tots els sensors avaluats se n'han descartat la majoria i n'han quedat quatre per a l'elecció definitiva, aquests són:

- LM92 (National)
- DS1620 (Dallas-Maxim Semiconductors)
- DS1821 (Dallas-Maxim Semiconductors)
- DS18B20 (Dallas-Maxim Semiconductors)

Si resumim les característiques dels quatre en una taula:

sensor	analògic / digital	*sortida*	precisió A 25ºC (ºC)	precisió a 70ºC (ºC)	precisió a –10ºC (ºC)	rang mesura (ºC)	tensions aliment. (V)
LM92	digital	sèrie	±0,33	±1	±1	(-55,150)	(2.7,5.5)
DS1620	digital	sèrie	±0,5	±0,5	±2	(-55,125)	(2.7,5.5)
DS1821	digital	sèrie	±1	±1	±1	(-55,125)	(2.7,5.5)
DS18B20	digital	sèrie	±0,5	±0,5	±0,5	(-55,125)	(3,5.5)

Taula A.2. Taula de característiques dels sensors considerats en l'elecció final

Recordem que aquesta elecció s'ha fet bàsicament tenint en compte la precisió del sensor, malgrat això en la taula anterior s'aprecia que tots els sensor seleccionats són de sortida digital seriada, que tenen uns rangs de mesura molt semblants i que les tensions d'alimentació són també semblants. El rang de temperatura no és un factor determinant en l'elecció, ja que el sensor haurà d'anar a la vora d'un emissor de radiofreqüència que serà, de ben segur, més restrictiu en quant a les temperatures de funcionament, i per tant serà aquest els que ens determinarà els límits de temperatura del sistema.

L'elecció definitiva s'haurà de prendre en funció de la precisió que ens ofereix cada sensor, per tant seria convenient estudiar-les millor:

LM92		DS1620		DS1821		DS18B20	
interval (ºC)	precisió (ºC)	interval (ºC)	precisió (ºC)	interval (ºC)	precisió (ºC)	interval (ºC)	precisió (ºC)
30	±0.33	(0,70)	±0.5	(-35,125)	±1	(-10,85)	±0.5
(10,50)	±0.5	(70,125)	>±1	(-55,-35)	±2	(-55,125)	±2
(-10,85)	±1	(-55,0)	>±1				
125	±1.25						
(-25,150)	±1.5						

Taula A.3. Detall de les característiques de precisió dels sensors seleccionats

Si s'analitza la taula s'aprecia que:

- El sensor LM92 és el més precís però només al voltants de 30ºC
- El sensor LM92 i el DS1620 són bastant precisos (±0,5ºC) en un rang ampli però per sobre de 0ºC
- El sensor DS1821 té un rang molt ampli on la precisió és de ±1ºC
- El sensor DS18B20 és bastant precís (±0,5ºC) en un rang relativament ampli (de –10ºC fins a 85ºC.

Per tant si interessés molta precisió en torn de 30º triarien el LM92, però això no és el que es busca, ja que es desitja que la precisió s'estengui en un rang més gran.
A la vista d'això existeix l'opció de DS1821 que en ofereix ±1ºC de precisió en un rang molt ampli, això tampoc és el que es necessita, ja que aquest rang és massa ampli. El factor limitant del marge de temperatures del sistema serà, de ben segur, el xip de radiofreqüència, i per tant no s'aprofitaria el fet que el sensor pugui operar en un rang tan ampli.
Així, només queda l'opció del DS1620 i del DS18B20, en aquest cas, l'elecció és clara ja que el dos tenen com a màxima precisió ±0,5ºC, que és una precisió prou bona, però el DS18B20 la ofereixen un rang de temperatures més ampli i a més s'estén per sota de 0ºC fins a una temperatura que és perfectament suportable pels sistemes de RF i per tant es aprofitable.

A la vista de tot això, de tots els sensor avaluats, el que millor compleix les especificacions és el DS18B20, i aquest serà el que es farà servir en el disseny del sistema.

A.2.- Selecció del circuit de radiofreqüència

A.2.1.- Introducció

Per a la selecció del circuit de radiofreqüència s'ha de tenir en compte que, en el present projecte, es pretén dissenyar un sistema on diferents elements (les unitats sensores) es comuniquen amb una unitat base utilitzant el mateix canal, l'aire. Això fa que s'hagi d'establir algun tipus de mecanisme que eviti que es produeixin col·lisions entre els missatges que s'envien.

Cada unitat sensora necessita poder enviar dades a la unitat base per tal que aquesta última rebi les mesures de les temperatures, en canvi, no és imprescindible que les unitats sensores puguin rebre cap tipus de missatge. Això es deu a que la principal missió del sistema és monitoritzar temperatures. El fet que la unitat base no pugui comunicar-se amb les sensores, i que per tant no tingui cap tipus de control sobre elles, no impedeix que el sistema pugui dur a terme la seva funció.

A.2.2.- Estudi d'alternatives

Existeixen dues grans alternatives a l'hora de seleccionar el tipus d'enllaç i per tant els circuits de radiofreqüència.

- **Enllaç unidireccional:** per a establir aquest enllaç és suficient que els circuits de ràdiofreqüència siguin o bé emissors o bé receptors. Els circuits de les unitats sensores serien els emissors, ja que són els que han de transmetre la mesura de la temperatura. El circuit de la unitat base seria únicament receptor, per tal de poder rebre els senyals que envien les unitats sensores.

- **Enllaç bidireccional:** per establir aquest enllaç els circuits de radiofreqüència han de ser emissors i receptors alhora.

En principi el sistema no necessita un enllaç bidireccional, ja que no és imprescindible que la unitat màster transmeti dades a les unitats sensores. El que ens farà decidir per un tipus d'enllaç o un altre serà el mode d'evitar les col·lisions de missatges:

Col·lisions entre missatges:

Existeixen dues alternatives per evitar col·lisions entre missatges quan diferents emissors utilitzen un mateix canal:

- **Multiplexació en freqüència:** aquesta tècnica consisteix en que el diferents emissors utilitzen freqüències diferents per a transmetre els seus missatges de tal manera aquests no col·lideixen. Aquesta opció es va haver de descartar ja que no es trobaren en el mercat circuits integrats receptors capaços de rebre alhora missatges en diferents freqüències. A més, aquesta tècnica presenta el problema de que el nombre de sensors instal·lables està limitat al nombre de freqüències disponibles.

- **Multiplexació en temps:** aquesta tècnica consisteix en fer que els missatges de les unitats sensores es distribueixin en el temps de manera que mai no es superposin. Això té l'inconvenient de que els emissors no poden transmetre de manera continuada, ja que han de deixar pas a altres emissors i per tant no es pot aprofitar al màxim la capacitat de transmissió. La tècnica de multiplexació en el temps presenta l'avantatge de que si el volum d'informació a transmetre és reduït, es pot instal·lar un elevat nombre de sensors. Aquest és el cas del present projecte, un senyal de temperatura representa molt poca informació sempre i quan només es vulgui obtenir espaiadament en el temps. Per tal d'aconseguir que el missatges no coincideixin en el temps s'ha avaluat dues tècniques:

    - **Aleatoritzar:** Cada emissor emet el seu missatge en moments aleatoris. De forma que si els missatges són curts i el nombre de sensors no és molt elevat, la probabilitat de que es produeixin col·lisions és baixa. Aquesta tècnica presenta l'avantatge de la seva simplicitat però presenta l'inconvenient de la seva poca robustesa, ja que la probabilitat de col·lisions existeix i per tant ocasionalment es perdrien missatges. Aquest fet fa que descartem aquesta possibilitat.

    - **Dividir el temps en *slots*:** A cada sensor se li assigna un espai de temps (*slot*) on ell i només ell pot emetre, Això fa que no es puguin produir col·lisions entre els missatges de diferents sensors.

El principal problema que presenta aquesta tècnica és que, degut a que els sensors són unitats autònomes, poden dessincronitzar-se uns respecte dels altres i, per tant, no emetre en la finestra de temps que cada un té assignada. Aquest problema es pot resoldre fent que totes les unitats sensores i base siguin alhora emissores i receptores, de manera que cada cert temps la unitat base pugui sincronitzar mitjançant un missatge a les unitats sensores. Aquesta tècnica presenta l'inconvenient de que és més complexa i que requereix que les unitats sigui emissores i receptores, però aconsegueix resoldre el problema de les col·lisions. Aquesta serà doncs la tècnica escollida.

A.2.3.- Requisits del circuit de radiofreqüència

A l'haver escollit dissenyar un sistema que eviti les col·lisions entre els missatges dividint el temps en slots, es fa necessari que el circuit a elegir sigui emissor i receptor alhora. Aquest tipus de circuits existeixen en el mercat baix el mon de *transceivers*.

Un altre requisit important és que els integrats tinguin entrada digital per tal de facilitar la comunicació amb el microcontrolador. Aquest requisit no representa una gran limitació, ja que el mercat es troben un bon nombre de circuits de radiofreqüència amb entrades i sortides totalment digitals, molt d'ells utilitzant protocols de comunicació integrats en microcontroladors de propòsit general.

A.2.4.- Avaluació de circuits integrats

Per a l'elecció del circuit de radiofreqüència s'han avaluat diversos circuits, a continuació es detallen les característiques dels més significatius:

TRF6900 (Texas Instruments)

És un *transceiver* que pot funcionar en les freqüències compreses entre els 868 MHz i els 915 MHz de la banda ISM. La màxima potència de transmissió és de 4.5dBm. Es comunica seriadament amb l'element de control. Es pot alimentar entre 2.2V i 3.6V, aquest marge estret pot representar un problema a l'hora de utilitzar-lo conjuntament amb altres elements. El principal inconvenient és que necessita un considerable nombre de components externs (condensadors, inductàncies, resistències, filtres,...).

XM1201 (Xemics)

És un *transceiver* que funciona a 433.92MHz en la banda ISM, no necessita cap tipus de component extern i es subministra muntat sobre una placa de circuit imprès amb la antena construïda sobre la mateixa placa. La màxima potència de transmissió és de 8.5dBm. Es pot alimentar entre 2.4 i 5.5 V, la qual cosa representa un avantatge ja que suporta un ampli rang de tensions. La comunicació amb el microcontrolador es porta a terme de manera seriada. El principal avantatge és que no necessita cap tipus de component extern, ja que tot es subministra incorporat a la placa.

RXQ1 (Rf Solutions)

És un *transceiver* que funciona en la banda ISM a 433 MHz, pot funcionar en dues freqüències diferents (433.92 MHz i 433.33 MHz), i transmet una potència de 5dBm. Es subministra muntat sobre una placa de circuit imprès de manera que no necessita cap tipus de component extern a excepció de la antena. Es comunica seriadament amb l'element de control, i es pot alimentar en un ampli rang de tensions, de 3V a 5V.

RTX-RTLP-434 (Aurel)

És un *transceiver* que funciona a 433.92 MHz en la banda ISM, transmetent una potència de 10dBm. No necessita gairebé cap tipus de component extern a excepció de l'antena i d'una resistència externa opcional per a controlar el nivell de soroll. L'entrada i sortida són digitals i amb protocol sèrie. S'ha d'alimentar entre 2.8 i 3.2V, això pot representar un problema per integrar aquest element en una sistema amb altres components que s'alimenten a tensions que no cauen dins aquest rang.

XTR-434 i XTR-869 (Aurel)

Són dos *transceivers* que funcionen respectivament a 433.92 MHz i a 869.85 MHz en la banda ISM i emeten amb una potència màxima de 10dBm. Es subministren en un encapsulat de tal manera que no necessiten cap tipus de component extern a excepció de l'antena. L'entrada i sortida és digital seriada, i el circuit s'ha d'alimentar entre 4,5 i 5,5 V.

nRF-401 (Nordic)

Es tracta d'un *transceiver* que funciona en la banda ISM a 433.93 MHz i 434.33 MHz, de manera que pot funcionar utilitzant una d'aquestes dues freqüències. La interfície és sèrie, i transmet una potència de 10dBm.

Es subministra de dues maneres diferents:

- L'integrat sol (nRF-401) de manera que necessita components externs per a que pugui funcionar (condensadors, resistències, bobines, cristall,...) així com l'antena.
- Un kit de desenvolupament (nRF-401-LOOPKIT) que porta tots els components externs necessaris per al funcionament del *transceiver* així com una antena construïda a la mateixa placa de circuit imprès. El kit consta de 6 unitats completes amb 3 tamanys d'antena diferents.

Les tensions d'alimentació han d'estar compreses entre 2.7 V i 5.25 V, aquest ample marge representa un avantatge de cara a la compatibilitat de tensions amb altres circuits.

nRF-903 (Nordic)

És un *transceiver* que opera en la banda ISM entre:

433.05 MHz i 434.87 MHz: on obté 10 canals diferents

868 MHz i 870 MHz: on obté 7 canals diferents

902 MHz i 928 MHz: on obté 169 canals diferents

L'elecció d'un dels tres grups de freqüències es fa a través dels components externs, mentre que dins cada grup els diferents canals es selecciones via software. La interfície és digital i seriada. Pot emetre amb diferents potències configurables via software i compreses entre –8dBm i 10 dBm.

Es subministra de dues maneres diferents:

- L'integrat sol (nRF-903) de manera que necessita components externs (resistències, condensadors, bobines, cristall, filtre ceràmic,...) així com una antena.
- Un kit d'avaluació (nRF903-Evaluation Board) que conté tots els elements externs necessaris per al funcionament del *transceiver*.

S'ha d'alimentar entre 2.7 i 3 V, aquest estret marge representa un problema de compatibilitat amb altres dispositius.

Tots els *transceivers* avaluats són semblants, funcionen en la mateixa banda (ISM) en diferents grups de freqüències. Tots tenen sistemes de comunicació amb els elements de control utilitzant protocols sèrie. Un dels paràmetres que els diferència, és la potència màxima d'emissió, aquest paràmetre és important, ja que és un dels que determinen l'abast del dispositiu, però no és l'únic ja que les característiques de l'antena també són fortament determinants. Les principals diferències es troben el la quantitat de perifèria que necessiten per funcionar. Pel que fa a aquest últim factor els circuits avaluats es poden dividir en dues grans categories: els *transceivers* que necessiten circuiteria externa i els *transceivers* que la porten ja muntada en una placa de circuit imprès.

Els avantatges i inconvenients de les dues opcions són els següents:
Dissenyar i construir la circuiteria externa, representa un esforç important, que si bé alguns fabricants donen certes pautes per a dur-ho a terme, és un treball que requereix temps i experiència. L'avantatge d'aquesta opció és que el preu de compra és baix, de l'ordre de 6 a 10 €. L'avantatge dels mòduls que ja porten a circuiteria incorporada és que es redueix considerablement el temps i l'esforç necessari, l'inconvenient és que el preu augmenta considerablement, de l'ordre de 30 a 50 €.

Per aquest motiu, no sembla una bona opció decantar-se ni pels mòduls de radiofreqüència, que encaririen l'hipotètic producte final, ni pels *transceivers* que requereixen molta circuiteria externa. L'opció escollida ha estat un terme mig, s'ha optat pel *transceiver* de Nordic nRF-401, ja que pertany al grup dels que necessiten components externs. Aquest, presenta l'avantatge de que es pot subministrar amb un kit de desenvolupament. Aquest kit és l'nRF-401-LOOPKIT que, per a la fase de prototipus, fa les funcions de mòdul de radiofreqüència sense necessitat de cap tipus de circuiteria externa i amb l'antena ja incorporada. El fabricant recomana que per a la primer fase de prototipus s'utilitzi el kit de desenvolupament i que un cop superada aquesta es procedeixi al disseny de la circuiteria externa.

A la vista de tot això, per al present projecte s'utilitzarà el kit nRF401-LOOPKIT, però amb la intenció de deixar la porta oberta per a que, en un futur, si s'arribés a superar la fase de prototipus, es pugui utilitzar el *transceiver* nRF-401. Per tal de fer això s'hauria de dissenyar i incorporar la circuiteria externa requerida, però no seria necessari fer cap altre tipus de modificació en el sistema ni de hardware ni de software.

S'ha d'esmentar també que, en el hardware que es construeix per fer funcionar l'nRF-401, s'hi inclourà també un connector i tolts el elements que siguin necessaris per a fer funcionar en un futur un altre *transceiver* de Nordic, l'nRF-903. Això es fa així, ja que els dos *transceivers* funcionen de manera no molt diferent i, per tant, paga la pena dissenyar un hardware que sigui compatible amb aquest altre component. Així es deixa el camí preparat per a l'aprofitament del hardware per a altres projectes emmarcats dins l'àmbit de la radiofreqüència.

B.- Aspectes addicionals de hardware i software

En aquest annex es recullen aspectes de hardware no tractats en la memòria però que poden ser d'interès, així com una breu descripció del software utilitzar per a la realització del projecte.

B.1.- Aspectes addicionals del sensor DS18B20

B.1.1.- La memòria interna

La memòria del sensor de temperatura DS18B20 s'organitza de la següent manera:

- Una memòria volàtil del tipus SRAM, anomenada *scratchpad*, que conté 9 bytes.
- Una memòria no volàtil del tipus EEPROM, que conté 3 bytes.

Els tres bytes de la EPROM es copien automàticament als tres bytes homònims de *l'scratchpad* quan s'engega el sensor. Els noms i l'ordre dels bytes es pot veure a la següent taula:

byte	**Scratchpad**		**EEPROM**
byte 0	LSB de la temperatura		
byte 1	MSB de la temperatura		
byte 2	Alarma de temperatura alta (T_H)	↔	Alarma de temperatura alta (T_H)
byte 3	Alarma de temperatura baixa (T_L)	↔	Alarma de temperatura baixa (T_L)
byte 4	Registre de configuració	↔	Registre de configuració
byte 5	byte reservat 1		
byte 6	byte reservat 2		
byte 7	byte reservat 3		
byte 8	CRC		

Taula B.1. Organització de la memòria interna del sensor DS18B20

La funció de cada byte es detalla a continuació:

- **LSB de la temperatura:** És el byte menys significatiu dels 16 bits que formen el registre de temperatures que s'utilitza per emmagatzemar el valor de la temperatura un cop feta la conversió A/D. Veure "Format del registre de temperatures" a la pàgina 24.
- **MSB de la temperatura:** És el byte més significatiu dels 16 bits que s'utilitzen per al valor de la temperatura un cop feta la conversió A/D. Veure "Format del registre de temperatures" a la pàgina 24.
- **Alarma de temperatura alta (T_H):** Són 8 bits on es guarda el valor màxim que pot assolir la temperatura sense que el sensor generi una alarma.
- **Alarma de temperatura baixa (T_L):** Són 8 bits on es guarda el valor mínim que pot assolir la temperatura sense que el sensor generi una alarma.
- **Registre de configuració:** Són 8 bits on es guarda la configuració del nombre de bits amb que s'ha de dur a terme la conversió de la temperatura. Es pot configurar des de 9 bits fins a 12 bits.
- **Bytes reservats 1, 2 i 3:** Són bits reservats per a l'ús intern del sensor.
- **CRC:** És el un codi CRC (Cyclic Redundancy check) generat amb els 8 bytes anteriors. Serveix per a detectar possibles errors en la transmissió de les dades.

Format del registre de temperatures

Un cop el sensor de temperatura ha efectuat la conversió A/D de la temperatura ha d'emmagatzemar aquestes dades en la seva memòria interna per a possibilitar que el microcontrolador les pugui llegir. La dada de temperatura convertida i expressada en graus centígrads (ºC), es guarda en un registre de 16 bits (2 bytes) codificada en complement a dos amb el bit de signe estès (repetit 5 vegades), això s'il·lustra millor en la següent taula:

	bit 7	**bit 6**	**bit 5**	**bit 4**	**bit 3**	**bit 2**	**bit 1**	**bit 0**
LS byte (byte 0)	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
	bit 15	**bit 14**	**bit 13**	**bit 12**	**bit 11**	**bit 10**	**bit 9**	**bit 8**
MS byte (byte 0)	S	S	S	S	S	2^6	2^5	2^4

Taula B.2. Format del registre de temperatures

On els bits que tenen una S són els bits de signe (el mateix bit repetit 5 cops), S=0 indica un número positiu, mentre que S=1 indica un número negatiu. Si la conversió de temperatura es

fa amb resolució de 12 bits, tots el bits són vàlids, mentre que si es fa amb 11 bits el bit 0 té un valor indefinit, si es fa amb 10 bits el valor indefinit és per als bits 0 i 1, i si es fa amb 9 bits els bits 0, 1 i 2 estan indefinits.

Format dels bytes d'alarma de temperatura

El sensor DS18B20, es capaç de generar senyals d'alarma quan la temperatura que ha convertit és superior o inferior a uns certs límits programables. En el present projecte no s'aprofita aquesta funcionalitat ja que disposem d'un microcontrolador capaç de dur a terme aquestes tasques amb major versatilitat. Els dos registres s'organitzen de la mateixa manera, la temperatura expressada en graus centígrads (ºC) es codifica en complement a 2 com a un número enter. Això es pot veure en la següent taula:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
S	2^6	2^5	2^4	2^3	2^2	2^1	2^0

Taula B.3. Estructura dels registres d'alarma de temperatura

El bit S és el bit de signe, si S=0 el número és positiu, mentre que si S=1 és negatiu.

Format del registre de configuració

El registre de configuració serveix per a configurar la resolució amb la que el sensor durà a terme la conversió de la temperatura. El registre s'organitza de la següent manera:

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
-	R1	R0	-	-	-	-	-

Taula B.4. Format del registre de configuració

On els únics bits configurables són R1 (bit 6) i R0 (bit 5). La relació entre els seus valors i la resolució, juntament amb els temps de conversió, es pot veure en la següent taula:

R1	R0	Resolució	Màxim temps de conversió
0	0	9 bits	93,75 ms
0	1	10 bits	187,5 ms
1	0	11 bits	375 ms
1	1	12 bits	750 ms

Taula B.5. Relació entre el valor del registre de configuració, la resolució i el màxim temps de mesura i conversió

El valor per defecte de la resolució és 12 bits i aquest serà el que es faci servir per a la mesura de la temperatura.

Funcionament del byte de CRC

El sensor DS18B20 té una funcionalitat molt útil per a fer el sistema de mesura més robust. Té la possibilitat d'enviar junt amb la lectura de la temperatura un codi detector d'errors conegut com a CRC (Cyclic Redundancy Check). Això permet al microcontrolador verificar si les dades han estat rebudes sense errors, i en cas de que es detecti un error tornar a sol·licitar l'enviament de les dades.

El codi de CRC consisteix en 8 bits que es guarden al byte 8 de *l'scratchpad* del sensor (veure "La memòria interna" pàgina 23). Aquest codi es calcula a partir dels 8 bytes anteriors de *l'scratchpad* , és a dir a partir dels 56 bits anteriors. D'aquesta manera s'hi inclouen els bits on es guarda la temperatura i el byte de configuració. Per tant el contingut del byte de CRC canvia a mesura que qualsevol d'aquests també ho fa.

El polinomi equivalent de la funció que s'utilitza per al càlcul del CRC és:

$$CRC(x) = x^8 + x^5 + x^4 + 1 \qquad \textbf{(Eq. B.1)}$$

D'aquesta manera s'aconsegueix generar un codi de 8 bits a partir dels 56 bits dels bytes 0 a 7. Això permet que el microcontrolador pugui verificar si s'ha produït un error en la transmissió de les dades, ja que el sensor de temperatura envia els 8 primers bytes seguit del byte de CRC. Per les propietats del codi CRC, si es recalcula el CRC d'un conjunt de bits seguit del codi CRC d'aquest conjunt de bits, el resultat és un codi CRC format tot per zeros.
En el nostre cas:

$$CRC(byte0,\ byte1, \ldots byte8,\ byteCRC) = 00000000 \qquad \textbf{(Eq. B.2)}$$

D'aquesta manera el microcontrolador pot calcular el CRC dels 72 bits (8 bytes de dades més el byte de CRC) i verificar si és igual a 00000000. En cas de que no sigui igual determina que s'ha produït un error i es torna a demanar al sensor que enviï les dades.

B.1.2.- Configuració hardware del bus

El bus 1-Wire® té per definició una sola línia de dades. Tant el microcontrolador com el sensor DS18B20 s'han de comunicar amb el bus a través d'un port que pugui posar-se en estat d'alta impedància. Això permet a cada element "deixar anar el bus", és a dir que, posant el pin que es connecta amb el bus en alta impedància s'aconsegueix deixar lliure el bus per a que l'utilitzi un altre. En el cas del sensor de temperatura s'aconsegueix deixar lliure el bus amb un pin *open-drain*, mentre que en el cas del microcontrolador el pin que es comunica amb el bus es posa en mode d'entrada i per tant en alta impedància.

Pel fet que els elements connectats al bus es poden posar en alta impedància, el bus necessita una resistència de *pull-up*, això fa que quan ningú no escriu al bus, és a dir quan està lliure, prengui el valor lògic de "1". El valor recomanat pel fabricant per a al resistència de *pull-up*, és de 4,7kΩ.

El sensor de temperatura té tres pins: el de terra (GND), el d'alimentació (Vdd), i el de comunicació (DQ). Per tal d'alimentar el sensor hi ha dues alternatives:

La primera, anomenada d'alimentació externa, requereix tres línies per alimentar el sensor i establir amb ell la comunicació. Aquestes són: Una línia d'alimentació (connectada al pin Vdd), una de terra (connectada al pin GND) i una de comunicació (connectada a DQ). Tot això es pot veure al següent esquema:

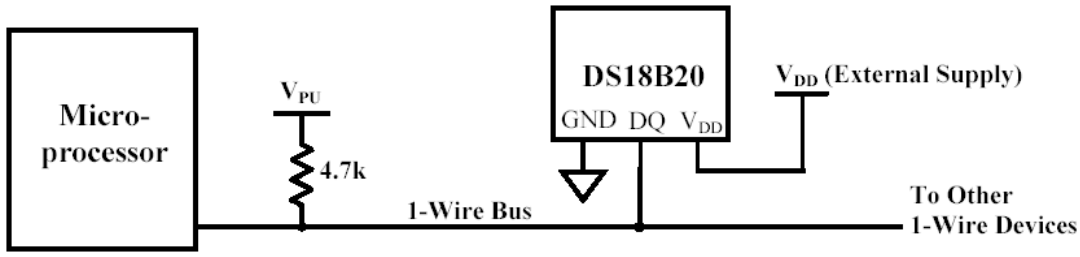


Figura B.1. Esquema del connexionat del bus en mode d'alimentació externa

La segona, anomenada d'alimentació paràsita, permet connectar el sensor mitjançant només dues línies. Aquestes són la de terra connectada al pin GND i la de comunicació connectada al pin DQ. El pin Vdd es connecta al GND. Aquest mètode és útil quan la temperatura es vol prendre a un lloc molt allunyat, i on per tant les línies de connexió són molt llargues, ja que

s'aconsegueix estalviar-ne una. En el nostre cas això no representa cap estalvi significatiu, ja que la temperatura es pren prop del microcontrolador.

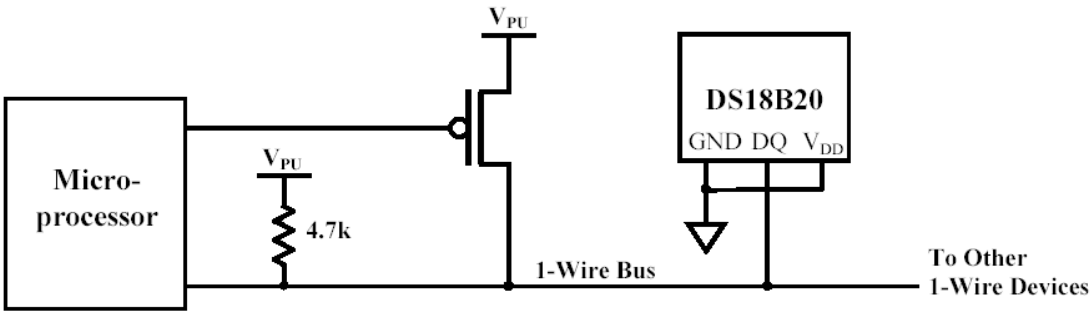


Figura B.2. Esquema del connexionat del bus en mode d'alimentació paràsita

Amb aquest mètode la potència arriba al sensor aprofitant el bus de comunicació mentre aquest es troba a "1", quan es troba a "0" la potència s'extreu d'un condensador intern que s'ha carregat prèviament quan el bus estava a "1". Tanmateix aquest mètode és útil per a subministrar poc corrent, és a dir que només és útil per a que el sensor realitzi certes operacions. Per a d'altres com per exemple dur a terme una conversió de temperatura s'ha d'utilitzar el que s'anomena un *strong pull-up*, ja que la operació pot arribar a involucrar corrents de l'ordre de 1,5mA. Si aquest corrent s'hagués de subministrar a través del bus de dades i per tant a través de la resistència de *pull-up*, la caiguda de tensió que experimentaria el bus seria inacceptable.

Per solucionar això, s'ha de disposar de l'esmentat *strog pull-up*, que el fabricant recomana que s'implementi amb un MOSFET, tal i com es pot veure a la Figura B.2 a la. El microcontrolador ha de ser l'encarregat de fer que el MOSFET subministri el corrent necessari en el moment oportú.

En el cas del present projecte, el mode d'alimentació triat és el d'alimentació externa, ja que el sensor de temperatura i el microcontrolador es troben un prop de l'altre. Per tant no representa cap avantatge significatiu estalviar connectar el sensor a l'alimentació a través d'una línia dedicada a tal efecte.

B.2.- Pantalla LCD. Protocol de comunicació

B.2.1.- Introducció

Per visualitzar la informació rebuda per la unitat base s'ha decidit utilitzar una pantalla LCD de 2 línies i 16 caràcters a cada línia. D'aquesta manera es pot visualitzar una quantitat d'informació suficient per a poder llegir les temperatures dels diferents sensors. La pantalla LCD elegida és la PC 1602-F de Powertip, és una pantalla retroil·luminada molt comuna entre les pantalles de caràcters i que es pot trobar fàcilment al mercat.

La interfície amb l'element de control, un microcontrolador en el cas del present projecte, es pot dur a terme de dues maneres diferents:

- La primera manera és utilitzant una comunicació mitjançant paraules de 8 bits, on cada paraula representa un caràcter a visualitzar. A més d'aquests senyals se'n necessiten 3 més de control, la qual cosa fa que es necessitin 11 pins del microcontrolador per a poder comunicar-se amb l'LCD.
- La segona manera és utilitzant paraules de 4 bits, de manera que un caràcter a visualitzar s'especifica mitjançant una successió de dues paraules de 4 bits. A part d'aquests 4 senyals se'n requereixen 3 més de control, tot això fa que siguin necessaris un total de 7 pins del microcontrolador.

Degut a que el nombre de pins lliures del microcontrolador utilitzat és limitat s'ha optat per la opció de realitzar la comunicació amb paraules de 4 bits ja que, tot i requerir un protocol lleugerament més complex, aconsegueix reduir en 4 el nombre de pins ocupats.

B.2.2.- Descripció del hardware

La pantalla LCD ve muntada de fàbrica sobre una placa on ja s'incorpora tota la circuiteria necessària per al seu control, en particular incorpora un controlador de pantalles LCD (Hitachi HD44780U). Aquest controlador s'encarrega de convertir els senyals que s'envien des de l'element de control de la pantalla en els senyals que l'LCD necessita per al seu funcionament.

Els pins de la pantalla accessibles des de l'exterior són els següents:

Número	Símbol	Funció
1	Vss	Connexió a GND
2	Vdd	Connexió a Vcc
3	V_0	Ajust del contrast
4	RS	Selecció de registre
5	R/W	Selecció lectura / escriptura
6	E	Enable
7	DB0	Bus de dades (no s'utilitza quan la longitud de paraula és de 4 bits)
8	DB1	Bus de dades (no s'utilitza quan la longitud de paraula és de 4 bits)
9	DB2	Bus de dades (no s'utilitza quan la longitud de paraula és de 4 bits)
10	DB3	Bus de dades (no s'utilitza quan la longitud de paraula és de 4 bits)
11	DB4	Bus de dades
12	DB5	Bus de dades
13	DB6	Bus de dades
14	DB7	Bus de dades

Taula B.6. Especificació dels pins de la pantalla LCD PC 1602-F

De tots aquests senyals els que serveixen per comunicar-se amb el microcontrolador són: els tres senyals de control (RS, R/W i E) i els senyals de bus de dades. Quan el sistema de comunicació és de 4 bits els senyals DB0, DB1, DB2 i DB3 no s'utilitzen.

B.2.3.- Protocol de comunicació

El sistema de comunicació triat, com ja s'ha justificat anteriorment, és el de 4 bits. Amb aquest sistema els senyals necessaris són: RS, R/W, E, DB4, DB5, DB6 i DB7. Les dades s'envies a través de DB4, DB5, DB6 i DB7, metre que RS, R/W i E són senyals de control. El valor del senyal R/W (Read / Write) indica si s'han de llegir o escriure dades a l'LCD. El senyal RS (Register Selection) s'utilitza per seleccionar el tipus de dades que es llegeixen o escriuen ja que hi ha dos tipus de dades, les comandes de funcions i els caràcters que es visualitzaran per l'LCD [HITACHI, 2000].

Les comandes de funcions serveixen per a que l'LCD realitzi diferents accions, esborrar la pantalla, moure el cursor, canviar de línia, ... o per esbrinar l'estat en que es troba l'LCD, posició del cursor, estat ocupat o lliure,...

Per tant combinant els senyals RS i R/W es poden dur a terme les següents 4 accions:

RS	R/W	Funció
0	0	Enviar (escriure) una certa funció a l'LCD
0	1	Rebre (llegir) l'estat en que es troba l'LCD
1	0	Enviar (escriure) un caràcter a l'LCD
1	1	Rebre (llegir) un caràcter de l'LCD

Taula B.7. Funcions del senyals de control de l'LCD

El senyal E (Enable) s'utilitza per a indicar al sistema de control de l'LCD quan se li envien les dades a través de DB4, DB5, DB6 i DB7 o quan es volen llegir dades a través dels mateixos pins.

Seqüència d'escriptura de dades

Per escriure una dada a la unitat LCD s'ha de realitzar el següent procés:

1.- Posar el senyal R/W a "0" per poder escriure a la pantalla LCD

2.- Flanc de pujada del senyal E almenys 40ns després de que R/W estigui a "0", E s'ha de mantenir a "1" durant almenys 230ns.

3.- Sortida per DB4 ... DB7 dels bits més significatius de la dada a enviar.

4.- Manteniment a "1" del senyal E durant almenys 80ns mentre a DB4...DB7 hi ha dades vàlides.

5.- Flanc de baixada de E

6.- Manteniment de les dades a DB4 ... DB7 durant almenys 10ns després del flanc de baixada.

7.- Repetir els punts 2 a 6 però fent sortir els bits menys significatius per DB4 ... DB7, i vigilant que hagin transcorregut més de 500ns entre flancs de pujada de E.

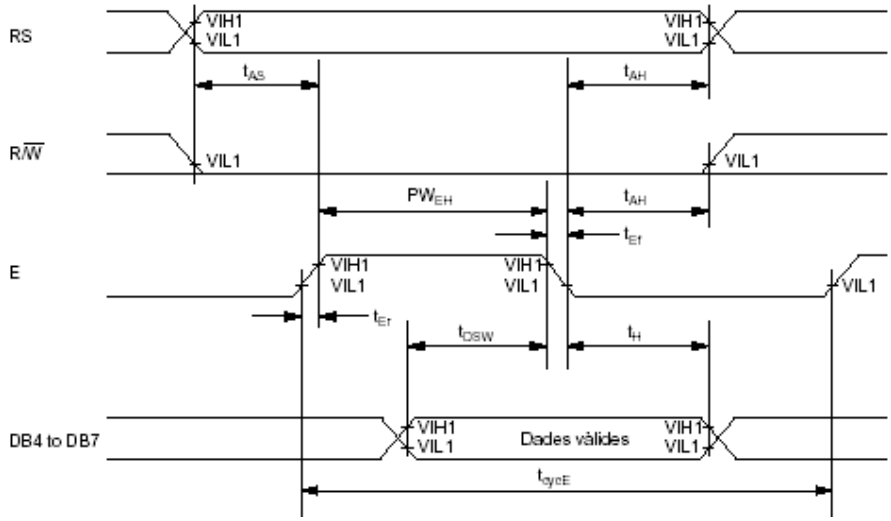


Figura B.3. *Timings* d'una operació d'escriptura del 4 primers o últims bits d'un byte de dades o d'instruccions.

RS estarà a "0" o a "1" segons es vulguin escriure instruccions o dades

Item	Symbol	Min	Typ	Max	Unit
Enable cycle time	t_{cycE}	500	—	—	ns
Enable pulse width (high level)	PW_{EH}	230	—	—	
Enable rise/fall time	t_{Er}, t_{Ef}	—	—	20	
Address set-up time (RS, R/$\overline{W}$ to E)	t_{AS}	40	—	—	
Address hold time	t_{AH}	10	—	—	
Data set-up time	t_{DSW}	80	—	—	
Data hold time	t_H	10	—	—	

Taula B.8. Valors dels timings per a una operació d'escriptura

Seqüència de lectura de dades.

Per llegir una dada a la unitat LCD s'ha de realitzar el següent procés:

1.- Posar el senyal R/W a "1" per a poder llegir de l'LCD.

2.- Flanc de pujada de E almenys 40ns després de que R/W estigui a "1", s'ha de mantenir E a "1" durant almenys 230ns.

3.- Llegir a través de DB4 ... DB7 els bits més significatius del byte a llegir, això s'ha de fer després d'almenys 160ns del flanc de pujada de E.

4.- Flanc de baixada de E.

5.- Repetir els passos 2 a 4 però tenint en compte que el bis que es llegiran ara seran els menys significatius. Entre els dos flancs de pujada han d'haver transcorregut almenys 500ns.

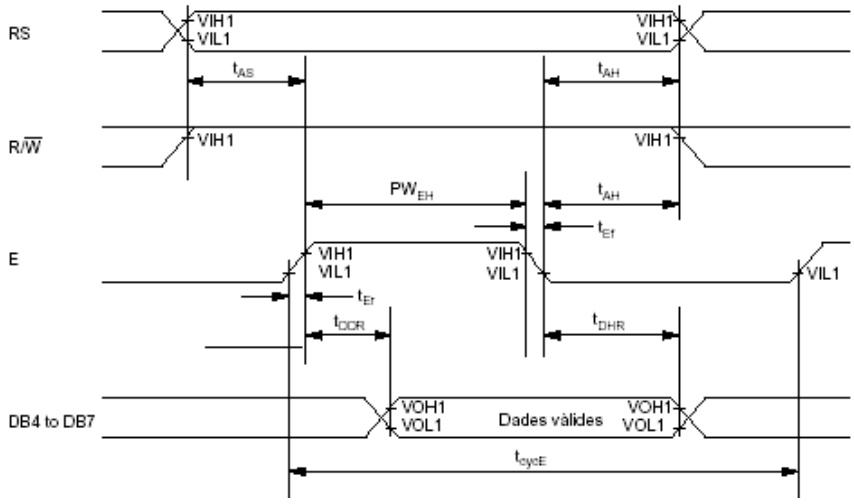


Figura B.4. Timings d'una operació de lectura del 4 primers o últims bits d'un byte de dades o d'instruccions.

RS estarà a “0” o a “1” segons es vulguin escriure instruccions o dades

Item	Symbol	Min	Typ	Max	Unit
Enable cycle time	t_{cycE}	500	—	—	ns
Enable pulse width (high level)	PW_{EH}	230	—	—	
Enable rise/fall time	t_{Er}, t_{Ef}	—	—	20	
Address set-up time (RS, R/$\overline{W}$ to E)	t_{AS}	40	—	—	
Address hold time	t_{AH}	10	—	—	
Data delay time	t_{DDR}	—	—	160	
Data hold time	t_{DHR}	5	—	—	

Taula B.9. Valors dels *timings* per a una operació de lectura

Consulta del Bussy Flag (BF)

Cada cop que s’envia una instrucció o una dada, l’LCD està ocupat durant un cert temps que depèn de la naturalesa de la dada o de la instrucció. Per aquest motiu, abans d’enviar una nova dada o instrucció s’ha d’estar segur que el controlador de l’LCD ja no està ocupat en tasques de procés.

Per a realitzar això hi ha dues opcions: la primera consisteix en esperar com a mínim el temps que el fabricant indica que es tarda en processar la tasca prèvia. L’altra consisteix en llegir un *flag* indicador que la pantalla subministra sota requeriment. Aquest *flag* s’anomena *Bussy Flag* (BF) i val “1” mentre l’LCD està ocupat i “0” quan està lliure. Per a llegir aquest *flag* s’ha d’emetre una ordre de lectura de l’estat de la pantalla (RS=0 i R/W=1). El que s’obtindrà, serà un byte que conté el BF com a bit més significatiu, la resta de bits són el comptador d’adreça que no interessa si només volem consultar el BF.

Com que es treballa amb interfície de 4 bits s’haurà d’emetre dos polsos del senyal E, amb el primer s’obtenen els 4 bits més significatius del byte i amb el segon els 4 menys significatius. S’ha d’anar consultant repetidament el BF fins a trobar-lo a “0”, un cop això ja s’ha produït es por emetre la següent instrucció o bé enviar la següent dada.

El la implementació software del protocol de comunicació s’ha utilitzat gairebé sempre la opció d’esperar el temps especificat pel fabricant abans des fer la següent transacció de dades. S’ha implementat d’aquesta manera en tots els casos excepte en l’ordre d’esborrat de la pantalla ja que, a part del cas d’aquesta última comanda, els temps a esperar són molt curts.

Si s'utilitza el microcontrolador en el mode que la CPU funciona a la freqüència més baixa configurable per software (amb l'oscil·lador de 4.9152MHz), els temps d'execució d'una sola instrucció és ja més gran que el temps d'espera que especifica el fabricant. En el cas de la comanda d'esborrat de la pantalla s'ha utilitzat el mètode de consulta del *bussy flag* ja que el temps d'espera és notablement més llarg.

Exemple de cicle de lectura i d'escriptura amb interfície de 4 bits

Per a il·lustrar el funcionament la interfície de 4 bits es mostra un exemple on primer s'envia una instrucció a l'LCD, es verifica que el BF estigui a "0" i per últim es llegeix una dada.

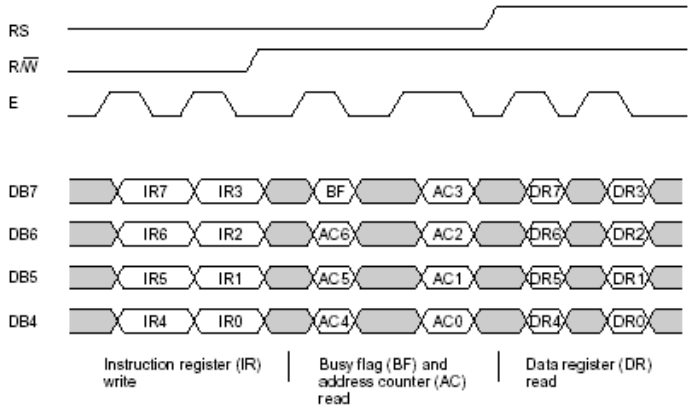


Figura B.5. Exemple del funcionament de la interfície de 4 bits

- La instrucció s'envia durant l'slot anomenat al diagrama "Instruction register (IR) write" i està formada pels bits IR7 ... IR0. Per a poder escriure, RW ha de valer "0" i com que el que s'envia és una instrucció i no una dada, RS ha de ser "0". Amb el primer pols de E s'envien els 4 bits més significatius (IR7 ... IR4), amb els segon pols s'envien la resta (IR3 ... IR0).
- La lectura de l'estat de l'LCD es realitza durant l'slot anomenat "Busy flag (BF) and address counter (AC) read". RS ha de continuar essent "0" ja que estem tractant amb instruccions i no amb dades, mentre que R/W ha de ser "1" ja que ara pretenem llegir Amb el primer pols de E s'obtenen els 4 bits més significatius del byte, entre els quals està el BF. Amb el segon pols s'obtenen la resta de bits.
- La lectura de la dada es realitza durant l'slot anomenat al diagrama "Data register (DR) read". En aquest cas tant RS com R/W han de ser "1" ja que pretenen llegir una dada. Els 4 primers bits del byte de la dada s'obtenen amb el primer pols de E, mentre que la resta s'obté amb el segon pols de E.

B.2.4.- Comandes de control de l'LCD

Una resum de les comandes junt amb els seus codis es pot trobar a la Taula B.11 a la pàgina 38, les descripcions de les principals comandes per a controlar la pantalla LCD són:

Clear Display: Esborra la pantalla, per a fer-ho fa sortir el caràcter "espai" en totes les posicions.
Cursor Home: Fa retornar el cursor a l'esquerra de la primera línia.
Entry mode set: Configura si a cada nova escriptura de caràcter el cursor s'ha de moure cap a la dreta o cap a l'esquerra, així com si a cada nova escriptura tots els caràcters s'han de desplaçar en la direcció prèviament determinada.
Display control: Configura si la pantalla es troba encesa o apagada, si el cursor es mostra i si el cursor parpelleja.
Cursor / Display shift: possibilita que sense escriure cap nova dada el cursor es mogui cap a la dreta o cap a l'esquerra, per exemple per corregir dades o que tots els caràcters de cada línia es desplacin cap a la dreta o cap a l'esquerra. En aquest cas, cada línia es desplaça però les dues ho fan a la vegada.
Function set: Configura que la interfície sigui de 8 bits o de 4 bits (veure "Seqüència d'inicialització de l'LCD" a la pàgina 39), que s'utilitzin una o dues línies i que el tamany de caràcter sigui de 5x7 o de 5x10 píxels. Quan s'utilitza 5x10 píxels només es pot utilitzar la primera línia i el joc de caràcters que es poden visualitzar només són els de les dues últimes columnes de la Taula B.10 a la pàgina 37.
Set DDRAM address: Especifica la posició a la qual s'ha de desplaçar el cursor. Per a fer això la comanda té 7 bits destinats a indicar la posició, la posició de més a l'esquerra de la primera línia és la 0000000, la següent és la 0000001 i així successivament.
La posició de més a l'esquerra de la segona línia és la 1000000, la següent és la 1000001...
Busy flag & address: Consulta l'esta de l'LCD, en particular l'estat del BF.
Read data: Llegeix la dada present a la posició actual del cursor
Write data: Permet escriure un caràcter a la pantalla que s'especifica a través dels 8 bits que per a això te destinats aquesta comanda. La taula amb els diferents caràcters que es poden escriure és la següent:

Taula B.10. Codis dels diferent caràcters que es poden escriure amb la pantalla LCD

La taula amb els codis de les comandes de control de l'LCD es pot veure a continuació:

Instruction	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	Description	Clock-Cycles
NOP	0	0	0	0	0	0	0	0	0	0	No Operation	0
Clear Display	0	0	0	0	0	0	0	0	0	1	Clear display & set address counter to zero	165
Cursor Home	0	0	0	0	0	0	0	0	1	x	Set adress counter to zero, return shifted display to original position. DD RAM contents remains unchanged.	3
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Set cursor move direction (I/D) and specify automatic display shift (S).	3
Display Control	0	0	0	0	0	0	1	D	C	B	Turn display (D), cursor on/off (C), and cursor blinking (B).	3
Cursor / Display shift	0	0	0	0	0	1	S/C	R/L	x	x	Shift display or move cursor (S/C) and specify direction (R/L).	3
Function Set	0	0	0	0	1	DL	N	F	x	x	Set interface data width (DL), number of display lines (N) and character font (F).	3
Set CGRAM Address	0	0	0	1	CGRAM Address						Set CGRAM address. CGRAM data is sent afterwards.	3
Set DDRAM Address	0	0	1	DDRAM Address							Set DDRAM address. DDRAM data is sent afterwards.	3
Busy Flag & Address	0	1	BF	Address Counter							Read busy flag (BF) and address counter	0
Write Data	1	0	Data								Write data into DDRAM or CGRAM	3
Read Data	1	1	Data								Read data from DDRAM or CGRAM	3

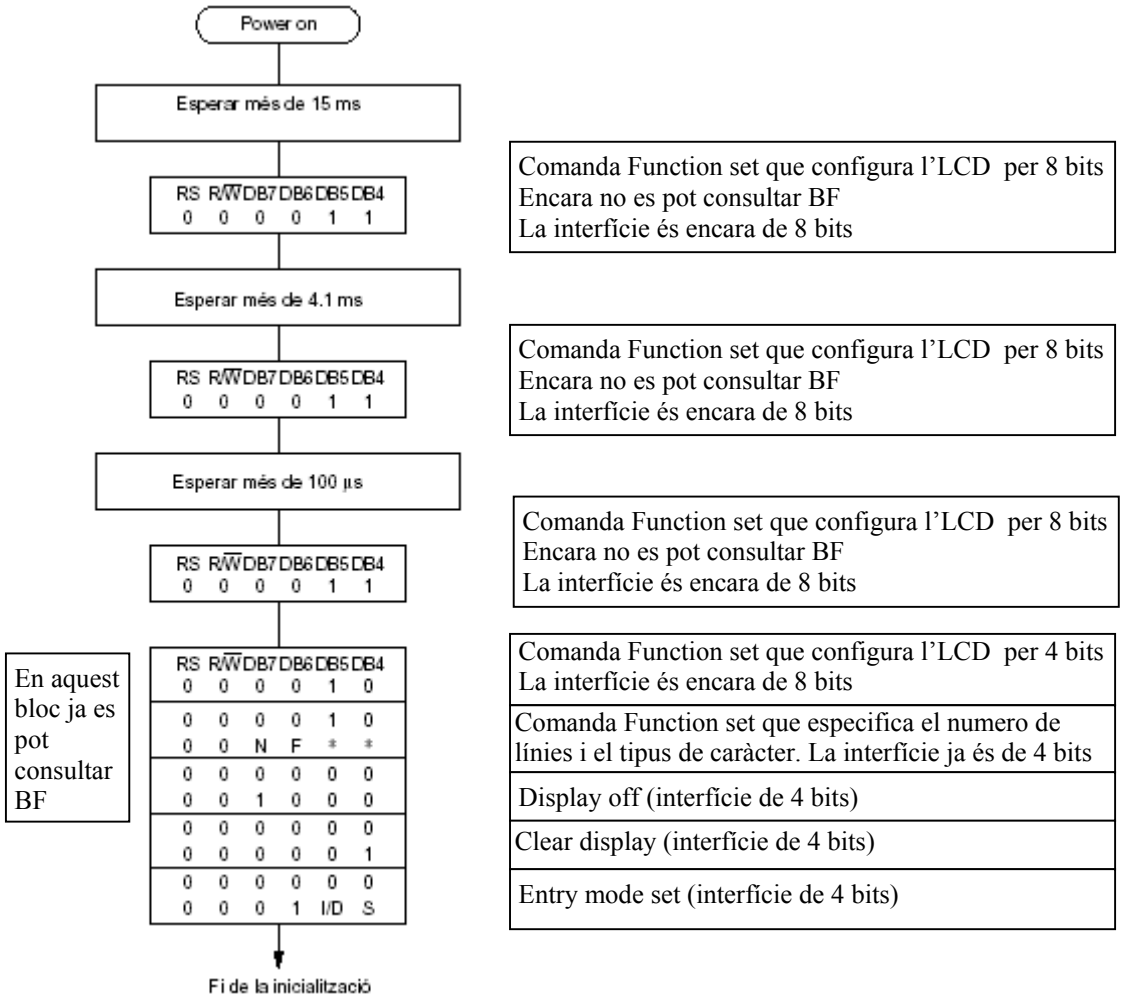
x : Don't care						
	I/D	1 0	Increment Decrement	R/L	1 0	Shift to the right Shift to the left
	S	1 0	Automatic display shift	DL	1 0	8 bit interface 4 bit interface
	D	1 0	Display ON Display OFF	N	1 0	2 lines 1 line
	C	1 0	Cursor ON Cursor OFF	F	1 0	5x10 dots 5x7 dots
	B	1 0	Cursor blinking	DDRAM : Display Data RAM CGRAM : Character Generator RAM		
	S/C	1 0	Display shift Cursor move			

Taula B.11. Resum de les comandes de control de l'LCD

B.2.5.- Seqüència d'inicialització de l'LCD

Per tal d'assegurar una bona inicialització de la pantalla quan es connecta l'alimentació el fabricant recomana que es duguin a terme una sèrie d'accions. A continuació s'esquematitza per al cas d'interfície de 4 bits.

Al principi de la inicialització la interfície és per defecte de 8 bits i per tant les dades s'han de transmetre utilitzant un sol pols de E (DB3 ... DB0 no es fan servir).
En els primers passos de la inicialització no es pot consultar encara el BF, i per tant s'han de deixar passar el intervals que recomana el fabricant entre l'emissió de dues comandes.



B.3.- Hardware utilitzat en el desenvolupament

A part de tot el hardware que s'ha descrit a la memòria (plaques del microcontrolador, placa de l'LCD i plaques de configuració de les unitats sensores) s'ha n'ha construït més. Aquest, en concret, són dues plaques anomenades plaques de test que serviren per a bescanviar informació amb el microcontrolador mentre no s'havia desenvolupat ni el hardware de la pantalla LCD ni el software per controlar-la.

La placa de test està formada principalment per:

- 4 LEDs per poder visualitzar informació del microcontrolador.
- 8 microinterruptors i un polsador per poder entrar informació al microcontrolador.
- 4 pins per fer accessibles externament senyals d'interès.

I es connecten a la placa del microcontrolador a través de connector J5.

El circuit de la placa pot veure's a la següent figura:

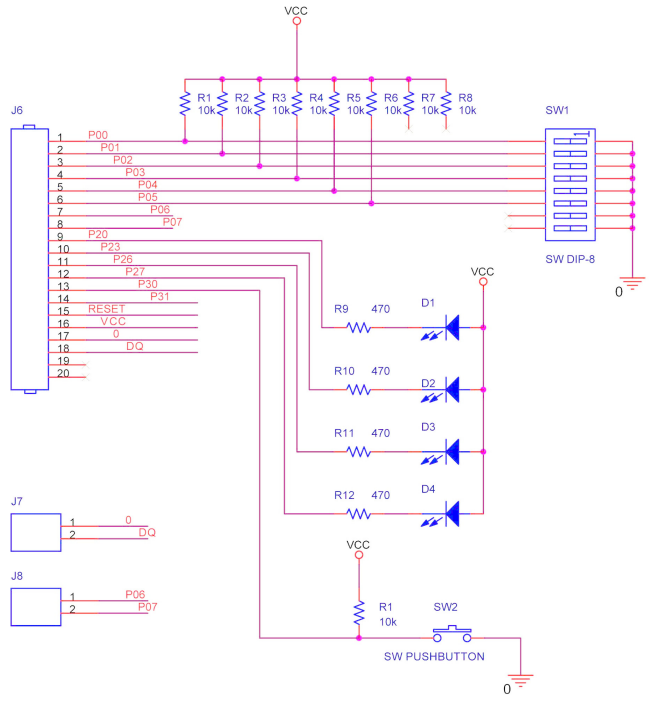


Figura B.6. Circuit de la placa de test

L'aspecte de la placa pot veure's a la següent fotografia:

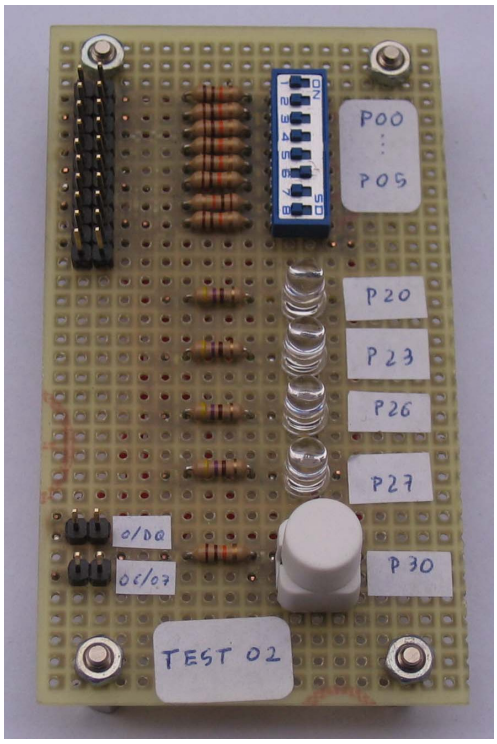


Figura B.7. Fotografia de la placa de test

Un joc de dues d'aquestes plaques amb dues plaques del microcontrolador i dos mòduls de ràdio freqüència han estat les eines amb les que s'ha realitzat, durant bona part del projecte, les tasques de posada apunt del sistema. Posteriorment es a desenvolupar el hardware de la placa de l'LCD i els software per al seu control i va ser possible visualitzar molta més informació gràcies al 32 caràcters que pot visualitzar l'LCD.

B.4.- Descripció del software utilitzat

En aquest annex es descriuen els diferents programes que s'han utilitzat per a la realització del projecte. Entre ells cal destacar dos grans grups:

El primer fa referència als programes de CAD utilitzats per al disseny dels circuits i de la placa de circuit imprès del microcontrolador (paquet ORCAD). El segon fa referència al software de compilació, simulació i programació del microcontrolador de NEC μPD78F9076.

B.4.1.- Disseny de les plaques de circuit imprès. ORCAD

L'ORCAD és un paquet que conté múltiples aplicacions. En el present projecte se n'han fer servir dues d'elles:

- *Orcad Capture*
- *Orcad Layout*

Orcad Capture és una aplicació a través de la qual és possible crear circuits electrònics per tal de després poder-los utilitzar en altres aplicacions. Aquestes poden ser, entre d'altres, simuladors del comportament del circuit o aplicacions per a la realització física del circuit a sobre d'una placa de circuit imprès. Això últim, en el cas del paquet ORCAD es fa a través de Orcad Layout.

Tots els circuits que s'han utilitzat en el projecte han estat creats amb Orcad Capture, tant aquells que posteriorment s'han plasmat en una placa de circuit imprès (placa del microcontrolador) com aquells que s'han realitzat artesanament sobre una placa de topos (placa de l'LCD i placa de configuració de les unitats sensores).

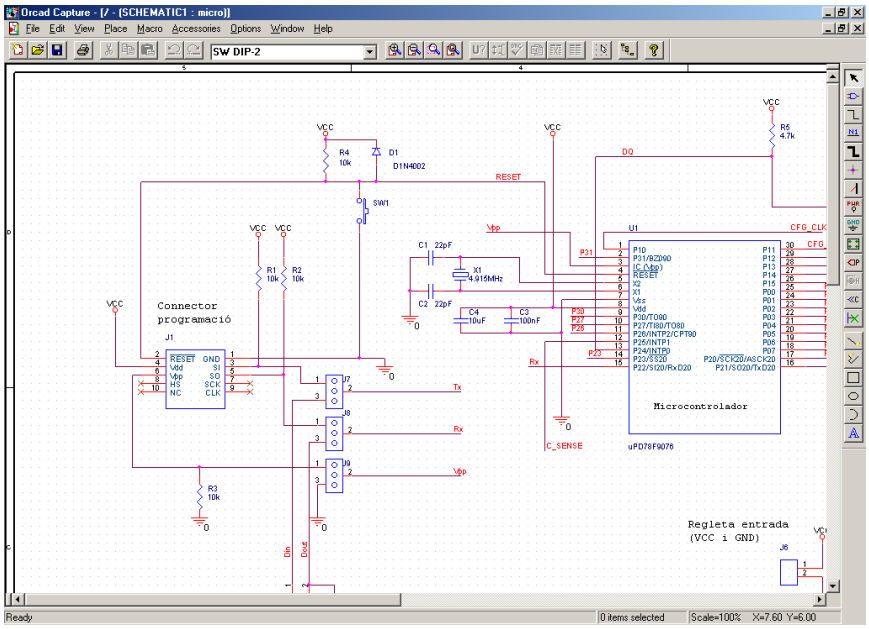


Figura B.8. Aspecte de la interfície d'Orcad Capture

Orcad Layout és una aplicació a través de la qual es pot dissenyar la implementació en placa de circuit imprès dels circuits que s'han creat amb *Orcad Capture.* El disseny de la placa del microcontrolador s'ha fet amb aquesta aplicació. La distribució en planta dels components s'ha realitzat manualment seguint criteris per facilitar la interconnexió dels diferents elements entre sí i amb elements exteriors (connectors). També s'han tingut en compte factors tèrmics que es concreten en allunyar tan com ha estat possible el sensor de temperatura del elements que més calor generen.

Orcad Layout incorpora la opció de fer el disseny de les pistes que uneixen els diferents elements de manera automàtica, per a la realització de la placa del microcontrolador no s'ha utilitzat aquesta opció. Ja que per optimitzar millor la distribució de les pistes i vies s'ha preferit fer aquesta tasca manualment. *Orcad Layout* conté un considerable nombre de llibreries de components electrònics, però per a la realització del projecte s'han hagut de modificar alguns dels components d'aquestes llibreries per tal d'adaptar-los als components que formen part de la placa del microcontrolador.

Un cop s'ha acabat el disseny de la placa s'han de generar uns fitxers anomenats *gerber files* que són els fitxers que utilitza la fresadora per fabricar les plaques de circuit imprès.

B.4.2.- Software del microcontrolador µPD78F9076

El fabricant del microcontrolador de NEC µPD78F9076, posa a disposició dels seus clients diferents paquets de software per possibilitar les tasques de programació i facilitar les de posada a punt del sistema. Els paquets són:

- *Iar Embedded Workbench*
- *Integrated Debugger*
- *Flashmaster*

Iar Embedded Workbench és un compilador per als microcontroladors de NEC, es capaç de compilar llenguatge C i ensamblador. La programació del microcontrolador del projecte s'ha realitzat amb ensamblador.

S'encarrega de compilar, "*linkar"* i fer l'executable del programes per als microcontroladors de NEC. En la realització del projecte s'han utilitzat dos tipus d'executables. El primer té l'extensió *.lnk i són els arxius que necessita el simulador. El segon té l'extensió *.hex i és el fitxer hexadecimal que utilitza el programador flashMASTER per programar el microcontrolador.

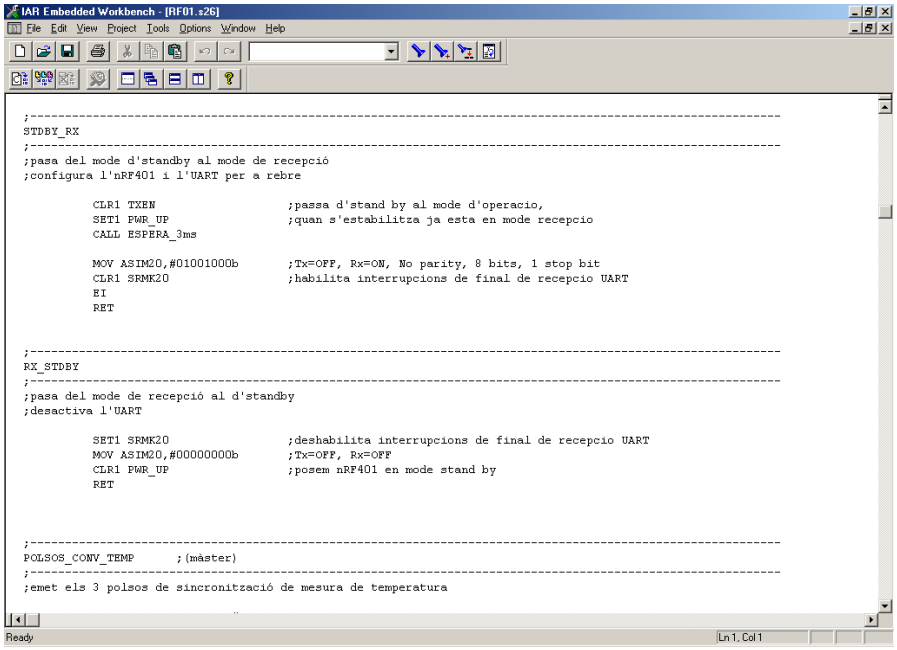


Figura B.9. Aspecte de la interfície d'Iar Embedded Workbench

Integrated Debugger SM78KOS en la versió 2.30 és l'entorn de simulació escollit per simular el funcionament del microcontrolador. Cal dir que per la naturalesa del projecte en cap moment s'ha pogut simular tot el conjunt. Això és així ja que el simulador no és capaç de simular l'execució de dos programes diferents que s'executen en dos microcontroladors diferents i que es comuniquen entre ells (en aquest cas via radiofreqüència). Per aquest motiu les simulacions que s'han portat a terme han estat només simulacions parcials. Ha hagut de ser en el laboratori, amb proves reals amb els microcontroladors, on s'han hagut de resoldre el problemes de posada a punt del programa per a que el establir l'enllaç de radiofreqüència.

El simulador realitza la seva tasca a partir del fitxers amb extensió *.lnk que genera *Iar Embedded Workbench*. Incorpora la simulació de les instruccions, així com dels perifèrics (UART, TIMERS,...) i monitoritza els registres i variables. També té una petita interfície d'usuari a través de la qual es poden generar senyals externs que actuïn en el microcontrolador i es poden visualitzar els que aquest genera.

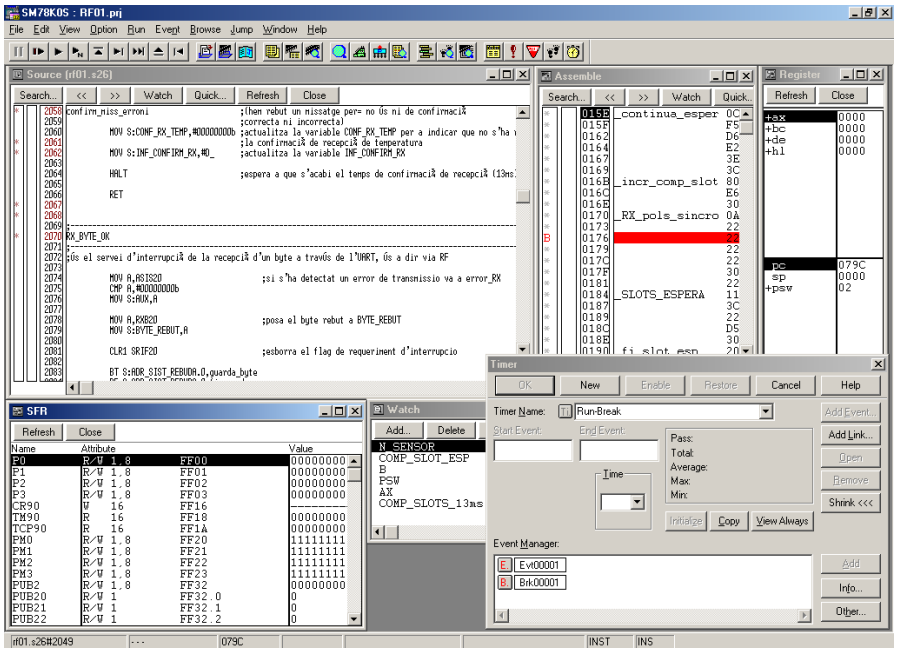


Figura B.10. Aspecte de la interfície d'usuari d'Integrated Debugger

El software del programador *flashMASTER* és l'encarregat de comunicar el PC amb el hardware que realitza la programació del microcontrolador, el pròpiament dit *flashMASTER*. Utilitza els fitxers amb extensió *.hex que genera *l'Iar Embedded Workbench*.

L'esquema de connexió és el següent: el PC es comunica amb el *flashMASTER* a través del port sèrie. El *flashMASTER* es comunica amb el connector J1 de la placa del microcontrolador a través del cable que el fabricant anomena cable 1. A la fotografia pot veure's el muntatge necessari per a la programació.

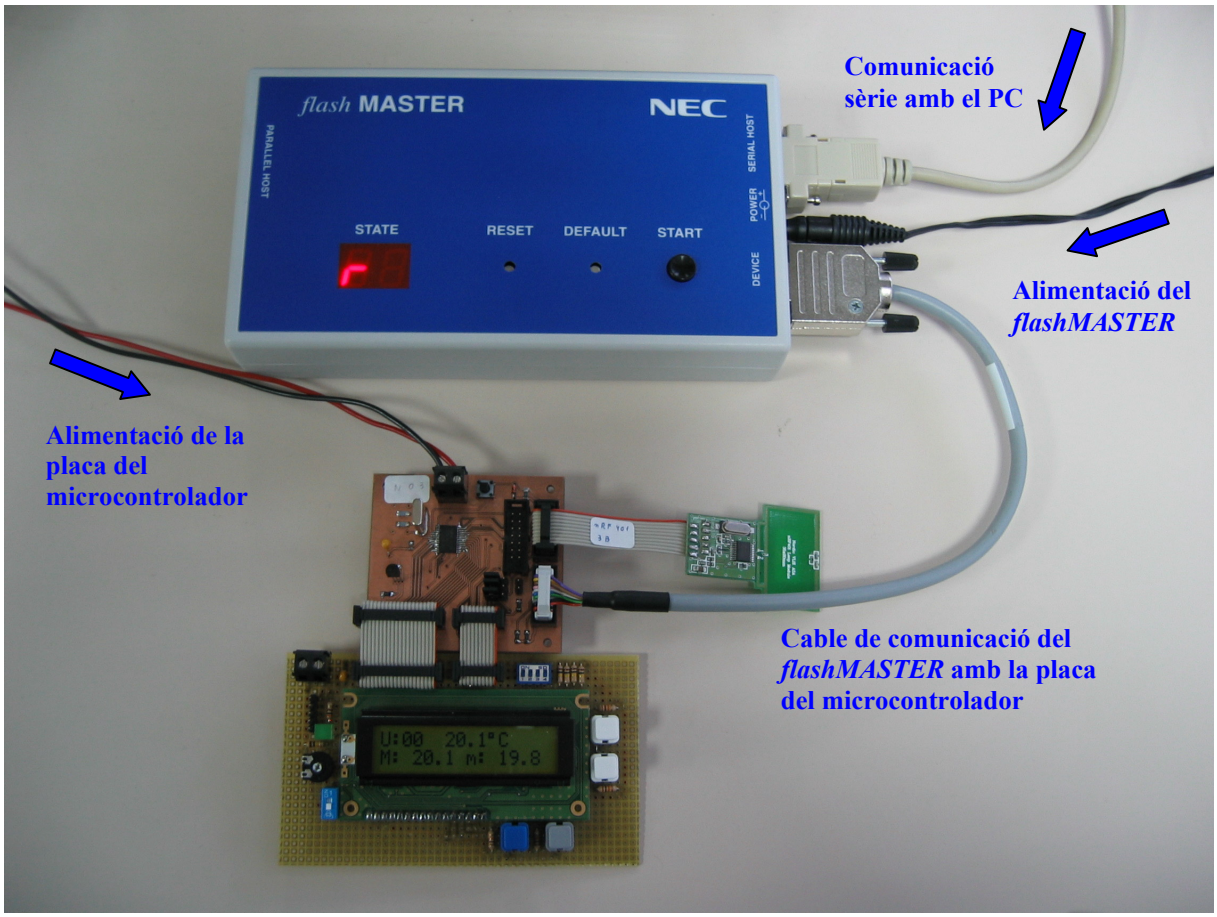


Figura B.11. Fotografia del muntatge de programació del microcontrolador

Per tal de programar el microcontrolador és necessari fer primer un esborrat de la memòria i procedir després a la gravació de la nova versió del programa. Totes aquestes acciones es controlen a través de la interfície d'usuari del software del *flashMASTER*.

C.- Càlculs i mesures

C.1.- Unitats de mesura habituals en radiofreqüència

En radiofreqüència, una de les mesures més habituals d'un senyal és la potència que aquest lliura a una càrrega determinada. La unitat de potència és el watt, ara bé, en radiofreqüència resulta més còmode la utilització de magnituds logarítmiques com ara el dB. El decibel es defineix com una relació logarítimica entre dues potències P_1 i P_2:

$$dB = 10\ log \frac{P_1}{P_2} \qquad \textbf{(Eq. C.1)}$$

Amb aquesta definició el que es té és una mesura relativa de P_1 respecte de P_2. Si s'assigna a P_2 un valor de referència adequat per al rang de magnituds que es vol mesurar, per exemple 1mW, la relació anterior deixa de ser una magnitud relativa i passa a ser absoluta. En el cas que el valor de referència es prengui igual a 1mW la magnitud (ara absoluta) s'anomena dBm.

$$dBm = 10\ log \frac{P}{1mW} \qquad \textbf{(Eq. C.2)}$$

De la mateixa manera es poden definir altres unitats com per exemple el dBW, per a una potència de referència d'1W.

Si en lloc de mesurar directament potències es volen mesurar tensions s'ha d'utilitzar la definició de potència i la llei d'Ohm amb una impedància de referència a sobre de la que mesurar les tensions. Si aquesta impedància no té part imaginària es té:

$$\left.\begin{array}{l} P(t) = v(t) \cdot i(t) \\ v(t) = R \cdot i(t) \end{array}\right\} \Rightarrow P(t) = \frac{v^2(t)}{R} \qquad \textbf{(Eq. C.3)}$$

i per tant la definició de decibel és:

$$dB = 10\ log \frac{P_1}{P_2} = 10\ log \frac{{v_1}^2 / R_1}{{v_2}^2 / R_2} = 10\ log \frac{{v_1}^2}{{v_2}^2} + 10\ log \frac{R_1}{R_2} = 20\ log \frac{v_1}{v_2} + 10\ log \frac{R_1}{R_2} \qquad \textbf{(Eq. C.4)}$$

si les dues impedàncies de referència són iguals l'expressió es redueix a:

$$dB = 10\ log \frac{P_1}{P_2} = 20\ log \frac{v_1}{v_2} \qquad \textbf{(Eq. C.5)}$$

si ara es pren com a tensió de referència per exemple un μV s'obté una altra magnitud absoluta anomenada dBμV:

$$dB\mu V = 20\ log \frac{V}{1\mu V} \qquad \textbf{(Eq. C.6)}$$

C.2.- Càlcul de l'abast de la comunicació amb una *loop antenna*

C.2.1.- Introducció

Loop antenna és la denominació que utilitza el fabricant dels mòduls de radiofreqüència (Nordic) per a referir-se a les antenes en forma de llaç que ell mateix fabrica. Recomana que s'utilitzin amb els seus integrats quan es vulgui desenvolupar un producte de baix cost i de tamany reduït. L'nRF401-LOOPKIT utilitza una *loop antenna*. El principal avantatge d'aquest tipus d'antena rau en el fet de que es construeixen amb coure a sobre del mateix circuit imprès, la qual cosa representa un clar estalvi en quant a espai i en quant a cost. El fabricant proposa un mètode de càlcul de l'abast utilitzant una *loop antenna*. Aquest mètode és el que s'ha fet servir i és el que es detalla a continuació [Nordic, 2000].

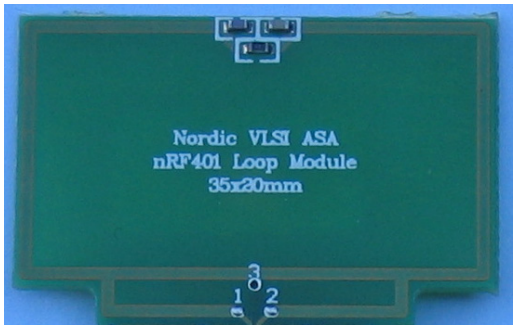


Figura C.1. Fotografia d'una Loop antenna

Per a més informació relacionada amb els càlculs que es duran a terme en aquesta secció referir-se a:C. A. Balanis, *"Antenna Theory, Analysis and Desig*n", second edition, John Wiley& Sons, Inc., 1997.

C.2.2.- Paràmetres físics d'una loop antenna

Les antenes que analitzarem tenen forma rectangular i estan construïdes al damunt de la placa de circuit imprès. La seva geometria es pot caracteritzar, en línies generals, pels següents paràmetres:

a_1 (m):	amplada de l'antena
a_2 (m):	alçada de l'antena
b_1 (m):	gruix del conductor de l'antena
b_2 (m):	amplada del conductor de l'antena

Taula C.1. Paràmetres físics d'una *loop antenna*

On b_1 és el gruix de la capa de coure de la placa de circuit imprès

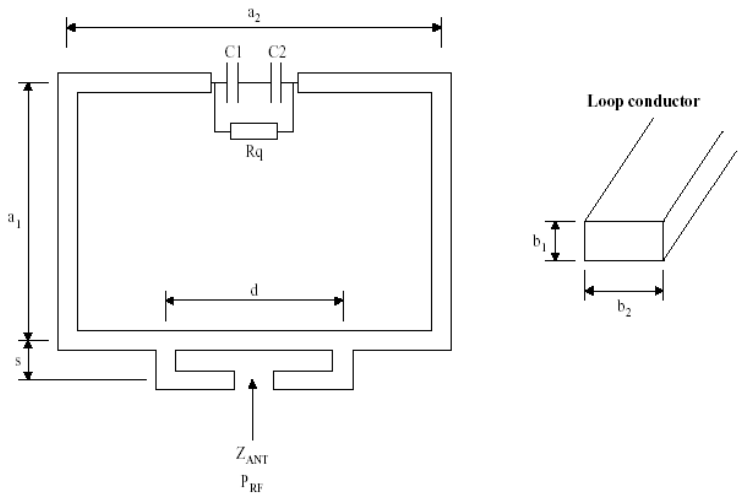


Figura C.2. Esquema de les dimensions físiques d'una *loop antenna*

Per fer els càlculs d'abast de l'antena és necessari trobar un model equivalent (d'antena quadrada i secció circular) per a l'antena rectangular i de secció rectangular. Això es fa mitjançant la introducció dels següents paràmetres:

a (m):	costat d'una antena quadrada equivalent a la *loop antenna*
b (m):	radi d'una antena de secció circular equivalent a la *loop antenna*
A (m^2)	àrea de l'antena quadrada equivalent a la *loop antenna*

Taula C.2. Paràmetres geomètrics equivalents a la *loop antenna*

Aquests paràmetres es calculen de la següent manera:

$$a = \sqrt{a_1 \cdot a_2} \qquad \textit{(Eq. C.7)}$$

$$b = 0{,}35 \cdot b_1 + 0{,}24 \cdot b_2 \qquad \textit{(Eq. C.8)}$$

$$A = a^2 \qquad \textit{(Eq. C.9)}$$

C.2.3.- Circuit equivalent de la *loop antenna*

Per a poder determinar l'abast de l'antena és necessari conèixer l'eficiència de la mateixa, i per a poder determinar això cal trobar primer els paràmetres del circuit equivalent a l'antena.

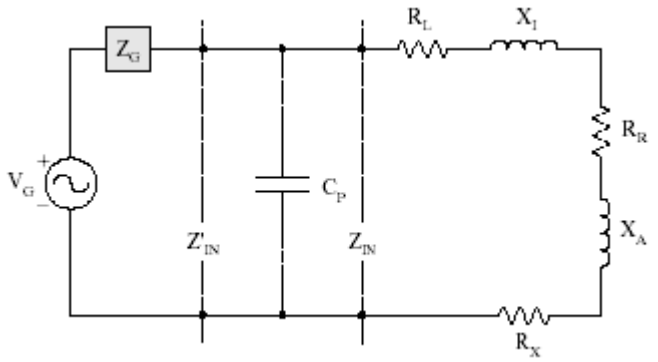


Figura C.3. Circuit equivalent de la loop antenna en mode de transmissió

On:

R_R (Ω)	Resistència de radiació
R_l (Ω)	Resistència de pèrdues de l'antena (*loop antenna*)
R_X (Ω)	Pèrdues òhmiques degudes a la resistència equivalent sèrie del condensador Cp
L_a (H)	Inductància de l'antena (*loop antenna*)
L_l (H)	Inductància del conductor de l'antena (*loop antenna*)
Z_{IN} (Ω)	Impedància d'entrada de la antena

Taula C.3. Paràmetres del circuit equivalent de la loop antenna

La impedància d'entrada de l'antena ve donada per l'expressió:

$$Z_{IN} = \left(R_R + R_L + R_X\right) + j \cdot 2\pi \cdot f_0 \left(L_A + L_I\right) \qquad \textit{(Eq. C.10)}$$

La resistència de radiació és:

$$R_R \approx 31171 \cdot \left(\frac{A^2}{\lambda^4}\right) \qquad \textit{(Eq. C.11)}$$

on λ és la longitud d'ona de la freqüència de ressonància:

$$\lambda = \frac{c}{f_0} \qquad \textit{(Eq. C.12)}$$

c: velocitat de la llum ($3 \cdot 10^8$m/s)

f_0: freqüència de ressonància

la resistència de pèrdues de l'antena és:

$$R_L = \frac{a_1 + a_2}{b_1 + b_2} \sqrt{\frac{\pi \cdot f_0 \cdot \mu_0}{\sigma}} \qquad \textit{(Eq. C.13)}$$

on:

σ (S/m): conductivitat del conductor, pel cas del coure val $5{,}8 \cdot 10^7$ S/m

μ_0 (H/m): permeabilitat magnètica del buit ($4\pi \cdot 10^{-7}$ H/m)

les pèrdues òhmiques degudes a la resistència equivalent sèrie del condensador Cp es poden calcular segons:

$$R_X = \frac{2\pi \cdot f_0 (L_A + L_I)}{Q} - R_R - R_L \qquad \textit{(Eq. C.14)}$$

On Q és el factor de qualitat de l'antena que es controla mitjançant la resistència R_Q en paral·lel amb el condensador C_p (format pels condensadors C_1 i C_2). Veure Figura C.2 a la pàgina 49. El fabricant recomana que el factor de qualitat sigui 50.

La inductància de l'antena es calcula amb l'expressió:

$$L_A = 2\mu_0 \frac{a}{\pi} \left(ln\left(\frac{a}{b}\right) - 0.774 \right) \qquad \textit{(Eq. C.15)}$$

I la inductància del conductor de l'antena amb l'expressió:

$$L_I = \mu_0 \frac{A}{2a} \qquad \textit{(Eq. C.16)}$$

C.2.4.- Càlcul de l'abast

Un cop s'han calculats els paràmetres del circuit equivalent estem en disposició de calcular l'eficiència de l'antena mitjançant l'expressió:

$$\eta = \frac{R_R}{R_R + R_L + R_X} \qquad \textit{(Eq. C.17)}$$

Que junt amb els següents paràmetres ens portarà a la determinació teòrica de l'abast:

P (dBm): potència de sortida del transmissor

S (dBm): sensibilitat de receptor

L'abast es calcula com:

$$R = \frac{\lambda}{4\pi\sqrt{\frac{S}{\eta^2 P}}} \qquad \textit{(Eq. C.18)}$$

Aquest abast és el que s'anomena abast en espai lliure, és a dir sense tenir en compte l'efecte de la presència d'obstacles. A més, l'abast així calculat fa referència a l'abast amb les antenes emissora i receptora alineades per tal d'aprofitar la seva direccionalitat.

Per aplicar aquestes expressions al càlcul de l'abast per a una de les *loop antena* del l'nRF401-LOOPKIT cal saber el valor de certs paràmetres que el fabricant subministra:

b_1=35μm	gruix del conductor de l'antena
b_2=1mm	amplada del conductor de l'antena
f_0=433,93 MHz	freqüència de ressonància (si s'utilitza l'altre canal pot valer també f_0=434,33 MHz)
Q=50	factor de qualitat de l'antena
σ=58μS/m	conductivitat del conductor (coure)
P=10dBm	potència de sortida del transmissor
S=-103dBm	sensibilitat de receptor

Taula C.4. Valors dels paràmetres dels LOOPKITs subministrats pel fabricant

Tots el càlculs es fan prenent com a f_0=433,93 MHz tenint així una longitud d'ona de λ=691,3mm. Amb la freqüència de l'altre canal les variacions són mínimes.

Amb aquests valors ja s'està en disposició de calcular els diferents abasts teòrics per als 3 tamanys d'antena del LOOPKIT:

	a_1	a_2
Tamany 1	20mm	35mm
Tamany 2	15mm	22mm
Tamany 3	10mm	18mm

Taula C.5. Tamanys de les antenes dels LOOPKITs

Amb les mides del tamany 1 s'obtenen els següents valors geomètrics i de circuit equivalent:

a=26,5 mm	costat d'una antena quadrada equivalent a la *loop antenna*
b=252,25μm	radi d'una antena de secció circular equivalent a la *loop antenna*
A=700 mm^2	àrea de l'antena quadrada equivalent a la *loop antenna*
R_r=0,067 Ω	Resistència de radiació
R_l=0,29 Ω	Resistència de pèrdues de l'antena (*loop antenna*)
R_X=5,03 Ω	Pèrdues òhmiques degudes a la resistència equivalent sèrie del condensador Cp
L_a=82,1 nH	Inductància de l'antena (*loop antenna*)
L_l=16,62 nH	Inductància del conductor de l'antena (*loop antenna*)

Taula C.6. Valors geomètrics i de circuit equivalent per a l'antena de tamany 1 de l'nRF401-LOOPKIT

Obtenint per tant una eficiència de l'antena de:

$$\eta = 0{,}01242 = -19\text{dB} \qquad \textit{(Eq. C.19)}$$

i un abast de:

$$R = 305\ m \qquad \textit{(Eq. C.20)}$$

Si ara es repeteixen els càlculs per als altres tamanys, s'obtenen els següents resultats:

	Tamany 1	**Tamany 2**	**Tamany 3**
a_1 (mm)	20	15	10
a_2 (mm)	35	22	18
a (mm)	26,45	18,17	13,42
b (μm)	252,25	252,25	252,25
A (mm^2)	700	330	180
R_r (Ω)	0,067	0,015	0,0044
R_l (Ω)	0,29	0,19	0,15
R_X (Ω)	5,03	3,19	2,18
L_a (nH)	82,1	50,90	34,34
L_I (nH)	16,62	11,41	8,43
η	0,01242 = -19dB	0,00437 = -23,5dB	0,00190 = -27dB
R (m)	305	107	47

Taula C.7. Comparativa entre els diferents paràmetres i abasts teòrics segons els tres tamanys d'antena del LOOPKIT

Els abasts per a cada tamany d'antena estan calculats per a parelles d'antenes amb el mateix tamany. En aquesta taula es pot veure com el tamany de l'antena influeix fortament en l'abast del sistema. Els abasts aquí calculats són, com ja hem dit abasts, en espai lliure, és a dir sense tenir en compte el efectes dels obstacles, si es vol tenir en compte això s'ha d'introduir un nou factor a l'expressió del càlcul de l'abast, aquest factor s'anomena *factor de pèrdues addicionals (L_X)*.

$$R = \frac{\lambda}{4\pi\sqrt{\frac{S}{\eta^2 P \cdot L_X}}} \qquad \textit{(Eq. C.21)}$$

Alguns valors d'aquest factor, i només a títol orientatiu, es poden trobar a la següent taula. Amb la introducció d'aquest coeficient l'abast calculat pot ser substancialment més petit.

Objecte	**Pèrdues típiques (dB)**
Paret interior	De 10 a 15
Paret exterior	De 2 a 38
Presència del terra	De 12 a 27
Finestra	De 2 a 30

Taula C.8. Pèrdues típiques per presència d'obstacles

Per fer-se una idea de com afecten aquest coeficients a l'abast només cal tenir en compte que unes pèrdues de 6dB redueixen l'abast a la meitat.

C.3.- Càlculs de consum

C.3.1.- Introducció:

Per poder fer una mesura directa del consum del sistema es requeriria un aparell que fos capaç d'integrar aquest consum al llarg de tot un període de funcionament (de l'ordre de minuts). Això es deu a que el sistema té pics de consum molt més alts que la mitjana (emissions i recepcions). Aquestes puntes de consum tenen una durada de l'ordre de milisegons mentre que un període complet del sistema és de l'ordre de minuts. Un amperímetre no és adequat per a realitzar aquests mesures, ja que el seu període d'integració és curt. L'amperímetre només serveix per mesurar el consum quan el sistema roman durant un temps suficientment llarg, això succeeix per exemple durant els períodes d'espera.

La opció escollida és fer les mesures de dues maneres diferents segons el temps durant el que es manté el nivell de consum

- Consum constat durant llargs períodes de temps: es realitzen amb un amperímetre.
- Consum durant períodes curts: es realitzen mesurant amb l'oscil·loscopi les caigudes de tensió en una resistència shunt. L'oscil·loscopi permet visualitzar la caiguda de tensió en els moments exactes on es produeixen els pics de consum i per tant, a partir de la caiguda de tensió, es pot conèixer el valor del corrent consumit pel sistema.

La resistència shunt té un valor (mesurat) de 1.2Ω per tant, per tal de conèixer el corrent consumit, s'ha de dividir per aquest valor la magnitud de la caiguda de tensió:

$$I = \frac{\Delta V_{shunt}}{1.2\Omega} \qquad \textit{(Eq. C.22)}$$

Un cop es coneguts tots aquests consums i sabent durant el temps que es produeixen es pot calcular una mitjana ponderada i obtenir el consum mitjà. D'aquesta manera el resultat que s'obté no és completament experimental sinó que és fruit d'un càlcul que, com és natural, conté suposicions de caire teòric, i per tant ha de ser pres només com a resultat estimatiu.

C.3.2.- Consum de d'una unitat sensora

Per calcular el consum d'una unitat sensora es divideix un cicle de funcionament (període comprés entre dues mesures de temperatura consecutives) en els següents períodes:

- Recepció del missatge de mesura de temperatura i mesura de temperatura.
- Sincronització, enviament de temperatura, recepció de confirmació, possible reemissió de la temperatura i slots d'espera (durant els períodes d'emissió de les altres unitats).
- Períodes d'espera entre sincronitzacions de 30s.
- Sincronitzacions cada 30s.

Les mesures de consum de la unitat sensora s'han fet amb els LEDs de la unitat deshabilitatats. És a dir que s'ha utilitzat la possibilitat de treure el jumper a través del que s'alimenten els LEDs per tal de minimitzar el consum de la unitat. Les mesures s'han fet també en absència d'interferències i per tant tots el missatges es reben correctament i al primer intent. La unitat sensora, abans de rebre cap missatge, està en mode de recepció a l'espera del primer missatge de conversió de temperatura. El càlcul del consum es fa en règim permanent i per tant aquest període no es té en compte.

Períodes d'espera

Amb un amperímetre s'ha mesurat el consum de la unitat sensora durant el període d'espera (Microcontrolador en mode *halt*, sensor en *standby* i nRF401-LOOPKIT també en *standby*). Aquest consum és de 0.5mA. Aquesta mesura s'ha fet durant el període entre sincronitzacions (30s) però és extrapolable als slots d'espera de 13.33ms, ja que tots els elements del sistema es troben el mateix estat (microcontrolador, sensor i nRF401-LOOPKIT).

Sincronització, enviament de temperatura, recepció confirmació, possible reemissió de la temperatura i slots d'espera.

Els slots d'espera es distribueixen abans i després del procés d'emissió de la temperatura en funció del número que tingui assignada la unitat. Un slot d'espera equival al temps que triga una unitat en emetre la temperatura, rebre el missatge de confirmació, i efectuar la possible reemissió de la temperatura (3·13.33ms = 40ms). El nombre total d'slots d'espera que realitza una unitat és 6, ja que el sistema està dissenyat per a 7 unitats sensores, la distribució dels slots és la següent:

Número d'unitat	Slots d'espera abans d'emetre	Slots d'espera després d'emetre
1	0	6
2	1	5
3	2	4
4	3	3
5	4	2
6	5	1
7	6	0

Taula C.9. Disposició dels slots d'espera segons el número d'unitat

Les mesures s'han fet per a la unitat 1, però són extrapolables a les altres unitats ja que l'ordre dels processos no influeix en el consum.

Les mesures de consum del període "Sincronització, enviament de temperatura, recepció confirmació, possible reemissió de la temperatura i slots d'espera" per a la unitat 1 es pot veure a la següent figura (a la part superior hi ha els senyals que rep la unitat sensora). Es pot apreciar com es rep el primer dels 3 missatges de sincronisme i, per tant, després s'esperen els slots corresponents a la possible recepció dels següents missatges de sincronisme (4 slots de 13.33ms). S'emet el missatge amb la temperatura, s'espera la recepció de la confirmació i com que el missatge indica que la temperatura s'ha rebut correctament no es torna a emetre, i s'espera l'slot corresponent (13.33ms).

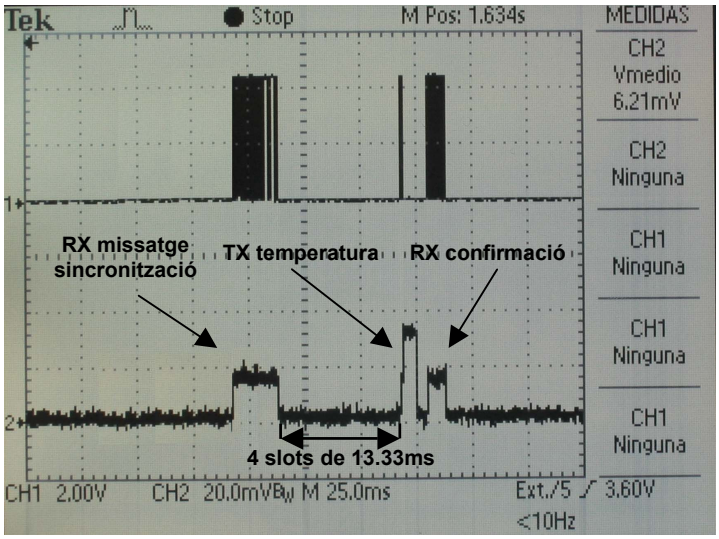


Figura C.4. Mesures del consum d'una unitat sensora en el període comprès entre la sincronització i el procés d'emissió de la temperatura

A les següents figures es pot veure més detalladament l'emissió de la temperatura i la recepció de la confirmació:

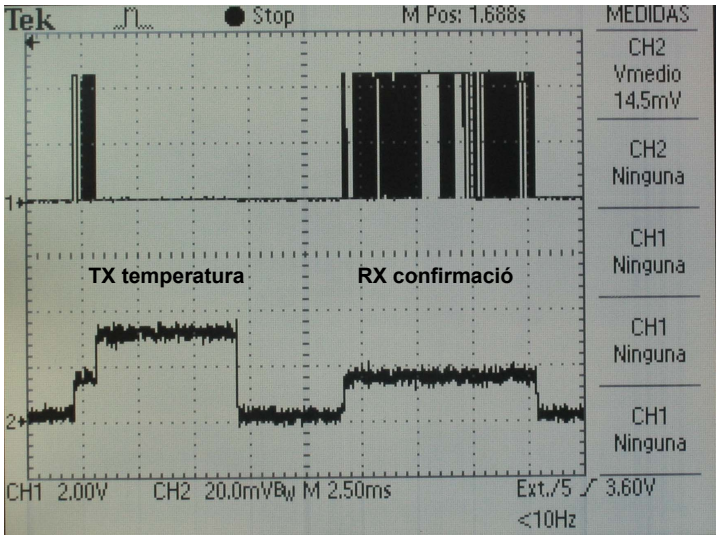


Figura C.5. Mesures de consum i senyal RX durant l'emissió de la temperatura i la recepció del missatge de confirmació

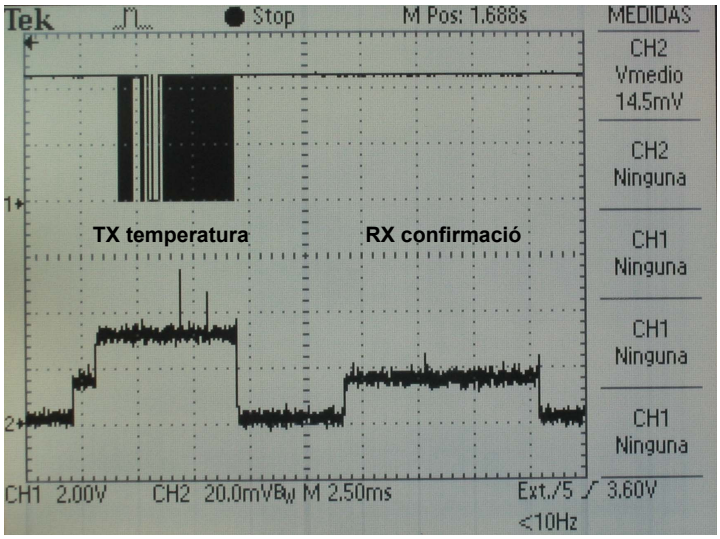


Figura C.6. Mesures de consum i senyal TX durant l'emissió de la temperatura i la recepció del missatge de confirmació

Els consum són:

- 13.3mA (16mV) durant 20ms [1] (RX missatge de sincronisme)
- 0.5mA durant 53.3ms (slots d'espera, ja que s'ha rebut el 1r missatge de sincronisme)
- 13.3mA (16mV) durant 1ms (preparació per transmissió)
- 27mA (32mV) durant 6.2ms (TX)
- 0.5mA durant 4.5ms (espera)
- 13.3mA (16mV) durant 8.8ms (RX confirmació)
- 0.5mA durant 13.33ms (slot espera, ja que no es reemet la temperatura)
- 0.5mA durant 6·40ms (slots d'espera)

Això suma 20.8ms, per tant s'hi ha de sumar 6.2ms en mode d'espera (0.5mA) per completar 2 slots de 13.3ms.

si se sumen ponderadament aquests consum s'obté:
2mA durant 353.3ms

[1] El temps de recepció depèn del nivell de dessincronització entre la unitat sensora i la màster

El pas *d'standby* a emissió es realitza segons els *timings* del següent diagrama:

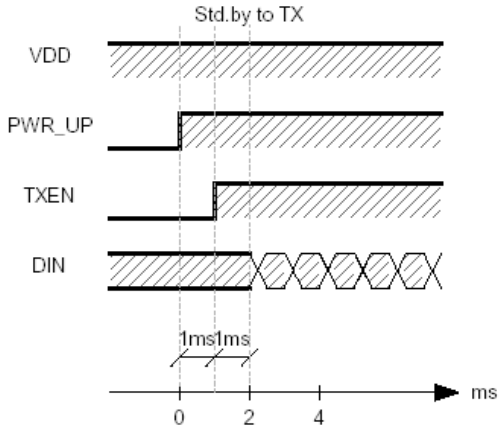


Figura C.7. *Timings* del pas a emissió de l'nRF401

És a dir que abans d'emetre el *transceiver* es troba durant 1ms en mode d'emissió. Aquest és el motiu pel qual a la part superior de la Figura C.4 i de la Figura C.5 s'aprecia com, abans d'emetre, el *transceiver* es està 1ms rebent senyals i com, durant aquest temps, el consum és el de recepció.

Sincronització entre mesures de temperatura (30s)

La sincronització entre mesures de temperatura es produeix de manera anàloga a la sincronització descrita en l'apartat anterior, amb la única diferència que després d'ella en continua en mode d'espera durant 30s més.

Així que el consum és:

13.3mA (16mV) durant uns 20ms [2]

Recepció missatge de mesura de temperatura i mesura de temperatura

La primera recepció del missatge de mesura de temperatura es produeix després d'un període indeterminat de temps (el temps que trigui l'usuari del sistema en iniciar el procés de mesura). Totes les altres recepcions del missatge de mesura de temperatura es produeixen de manera anàloga a la recepció del missatge de sincronisme. De fet tal i com es detalla a l'estructura del programa a la memòria, la recepció d'aquest missatge es fa amb la mateixa funció que per a

[2] El temps de recepció depèn del nivell de dessincronització entre la unitat sensora i la màster

rebre el missatge de sincronisme (ESPERA_SINCRO). La única diferència és que si es rep un missatge de mesura de temperatura es procedeix a la mesura de la mateixa.
Per tant el consum de la recepció del missatge de mesura de temperatura és de:
13.3mA (16mV) durant uns 20ms [3]

Pel que fa al consum durant la mesura de la temperatura amb l'oscil·loscopi s'han recollit les formes de la següent figura. La forma d'ona superior correspon al pin DQ (pin de comunicació amb el sensor de temperatura).

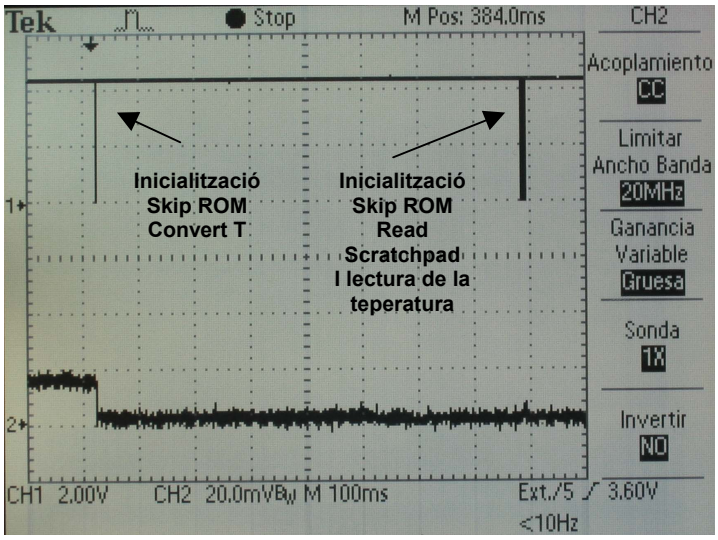


Figura C.8. Mesures del consum durant la mesura de la temperatura

Entremig, el sensor fa la conversió de la temperatura, com que el consum d'aquest període és petit s'obté una tensió petita difícil de mesurar amb l'oscil·loscopi. Tot i això fent una estimació del consum, es pot xifrar en uns 1.5mA. Aquest consum s'estén durant els 775ms que dura la el procés de mesura.

El sistema contempla la possibilitat d'una segona mesura en cas de que el codi CRC detecti un error en el primer intent. Per a fer aquesta càlcul del consum s'assumeix que la mesura es fa de manera correcta al primer intent, per tant durant els següents 775ms s'està en mode d'espera (0.5mA de consum).

[3] El temps de recepció depèn del nivell de dessincronització entre la unitat sensora i la màster

Per tant el procés de recepció del missatge de conversió de temperatura i la conversió de la mateixa consumeix:
13.3mA durant uns 20ms
1.5mA durant 775ms
0.5mA durant 775ms

que sumat ponderadament fa 1.2mA durant 1570ms

Consum mitjà de la unitat sensora

El consum de la unitat en un període de funcionament (període entre mesures de temperatura) depèn de la freqüència en que es facin aquestes mesures:

Mesures cada 30s: El consum és calcula ponderant:

- Consum de "Recepció missatge de mesura de temperatura i mesura de temperatura" (1.2mA durant 1570ms)
- Consum de "Sincronització, enviament de temperatura, recepció confirmació i possible reemissió de la temperatura" (2mA durant 353.3ms)
- Consum d'1 slot d'espera de 30s (0.5mA durant 30s)

Això fa un total de 0.55mA

Mesures cada 1 minut: El consum és calcula ponderant:

- Consum de "Recepció missatge de mesura de temperatura i mesura de temperatura" (1.2mA durant 1570ms)
- Consum de "Sincronització, enviament de temperatura, recepció confirmació i possible reemissió de la temperatura" 2mA durant 353.3ms)
- Consum d'1 slot d'espera de 30s (0.5mA durant 30s)
- Consum de "Sincronització entre mesures de temperatura" (13.3 durant 20ms)
- Consum d'1 slot d'espera de 30s (0.5mA durant 30s)

Això fa un total de 0.53mA

Mesures cada 5minuts: El consum és calcula ponderant:

- Consum de "Recepció missatge de mesura de temperatura i mesura de temperatura" (1.2mA durant 1570ms)
- Consum de "Sincronització, enviament de temperatura, recepció confirmació i possible reemissió de la temperatura" (2mA durant 353.3ms)
- Consum d'1 slot d'espera de 30s (0.5mA durant 30s)
- 9 vegades :
    - Consum de "Sincronització entre mesures de temperatura" (13.3 durant 20ms)
    - Consum d'1 slot d'espera de 30s (0.5mA durant 30s)

Això fa un total de 0.51mA

Per a períodes més llargs el consum s'estabilitza en 0.5mA És a dir que el consum, és pràcticament independent del període de mesures de temperatura. Això és així gràcies a l'estratègia de només estar en estat de recepció en els moments en que s'espera rebre. S'ha aconseguit reduir el consum del sistema pràcticament al consum en estat d'espera (0.5mA), en front dels 13.3mA que consumeix estar sempre en estat de recepció.

C.3.3.- Consum de la unitat màster

Per calcular el consum de la unitat màster es divideix un cicle de funcionament (període comprés entre dues mesures de temperatura consecutives) en els següents períodes:

- Emissió dels 3 missatges de mesura de temperatura
- Mesura de temperatura
- Emissió dels 3 missatges de sincronització
- Per a cada una de les 7unitats sensores (encara que no estiguin presents)
    - 1r intent de recepció de la temperatura
    - emissió missatge de confirmació
    - 2n intent de mesura de la temperatura
- Gestió de la interfície d'usuari.

Les mesures s'han fet també en absència d'interferències i per tant tots el missatges es reben correctament i al primer intent. Les mesures s'han fet amb tots els elements en funcionament,

inclòs el LED verd del màster. El càlcul del consum es fa en règim permanent, és a dir sense tenir en compte el període que transcorre fins que l'usuari del sistema decideix iniciar la seqüència de mesures.

Consum en mode d'espera

En el mode d'espera el microcontrolador està en mode *halt*, el sensor en mode *standby*, el mòdul de RF també en mode *standby* i la pantalla LCD en funcionament. La mesura del consum en mode d'espera s'ha fet amb l'amperímetre ja que és possible configurar el microcontrolador per a que estigui llargs períodes de temps en aquest mode. El consum mesurat és de 1,2mA. Aquest consum és superior al del mode d'espera de la unitat sensora degut a que, en la unitat màster, s'hi afegeix el consum de la pantalla LCD.

Consum en mode gestió interfície d'usuari

En el mode de gestió de la interfície d'usuari el microcontrolador està en mode actiu, el sensor en mode *standby*, el mòdul de RF també en mode *standby* i la pantalla LCD en funcionament. La mesura del consum en mode de gestió d'interfície d'usuari s'ha fet amb l'amperímetre ja que la unitat està llargs períodes en aquest mode. El consum mesurat és de 3.5mA. aquest consum és superior al consum en mode d'espera ja que el microcontrolador està en mode actiu gestionant la interfície d'usuari.

Emissió dels 3 missatges de mesura de temperatura

Les mesures del consum durant aquest període pot veure's a la figura:

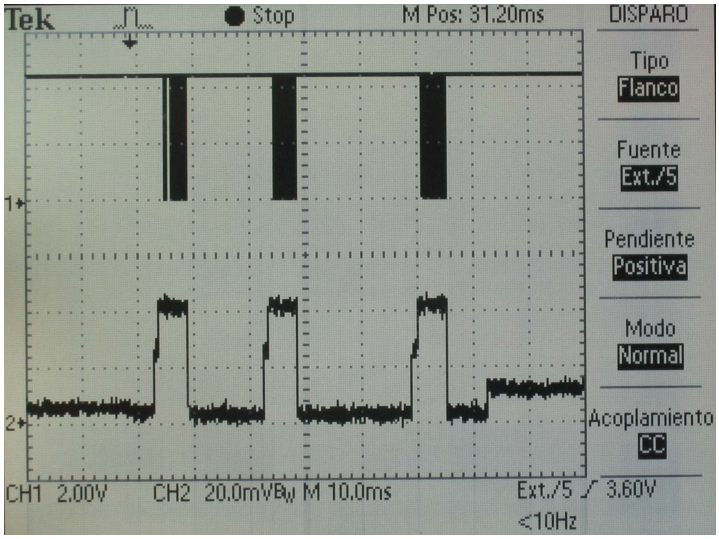


Figura C.9. Mesures de consum i senyal TX durant l'emissió dels 3 missatges de mesura de temperatura

Si s'analitza en detall el consum durant l'emissió d'un missatge es té la següent figura:

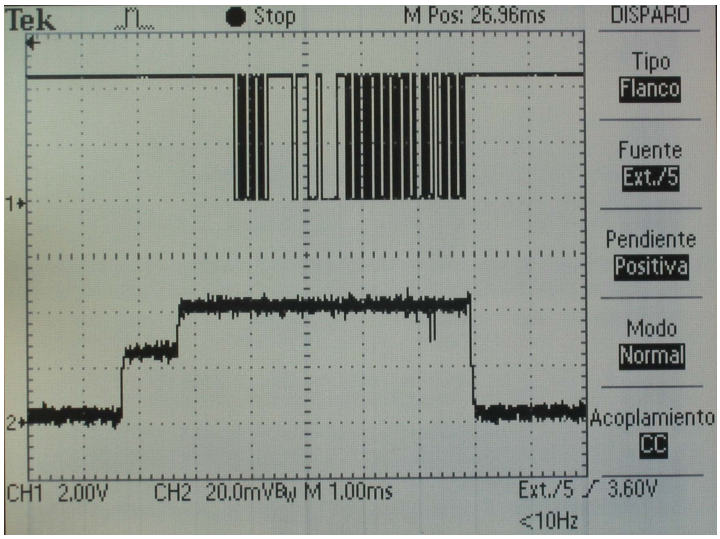


Figura C.10. Detall de les mesures del consum i senyal TX durant l'emissió d'un missatge de mesura de temperatura

A la vista de les figures emetre 1 missatge de mesura de temperatura té un consum que es pot desglossar en:

- 22mA (26mV) durant 1ms (preparació de transmissió)
- 34mA (40mV) durant 5.2ms (transmissió)
- 1.2mA durant 7.1ms (el que resta per completar l'slot de 13.3ms)

Això fa un total de 15.6mA durant 13.3ms.
Al ser els 3 missatges de sincronisme iguals, emetre'ls consumeix 15.6mA durant 40ms

El consum durant l'emissió i la recepció és superior a quan aquestes es fan en la unitat sensora, això es deu a que en la unitat màster les mesures s'han fet amb el LED verd del màster actiu, i durant aquests períodes està encès.

Mesura de temperatura

El consum durant el període de mesura de temperatura pot veure's a la figura:

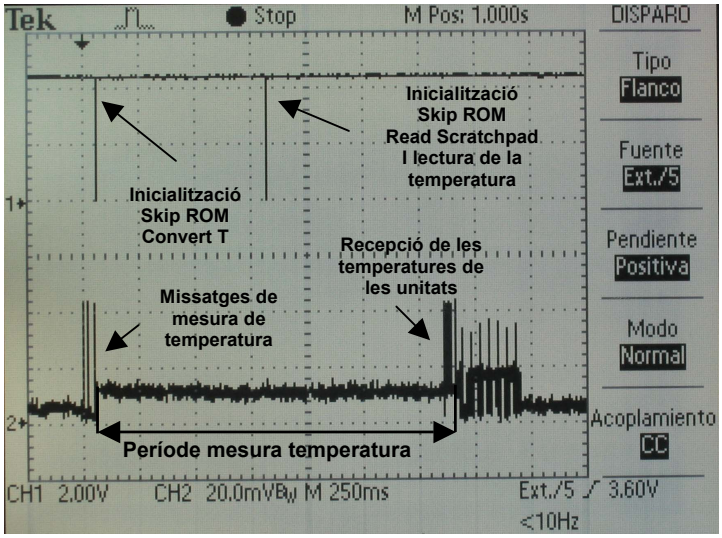


Figura C.11. Mesures de consum i senyal DQ durant la mesura de temperatura

El consum durant el període de mesura és pot xifrar en 10mA (12mV) durant 1550ms. Aquest consum és degut majoritàriament al LED, ja que aquest consumeix uns 8mA. En la figura es poden veure també els consums dels missatges de mesura de temperatura, de sincronisme, i el període de recepció de les mesures de temperatura de les 7 unitats sensores.

Emissió dels 3 missatges sincronisme

El consum per a emetre els 3 missatges de sincronisme és el mateix que per emetre els 3 missatges de mesura de temperatura, ja que l'únic que els diferencia és el seu contingut. Per tant el consum és de: 15.6mA durant 40ms.

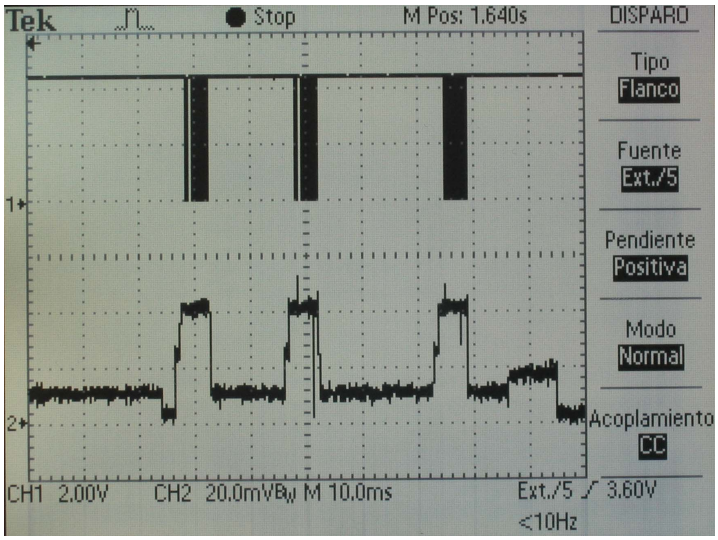


Figura C.12. Mesures del consum durant l'emissió dels missatges de sincronització

Període de recepció de les temperatures de les unitats

Aquest període consisteix en repetir 7 cops (un cop per cada unitat del sistema, encara que no estiguin presents) la seqüència:

- 1r intent de recepció de la temperatura
- emissió missatge de confirmació.
- 2n intent de mesura de la temperatura.

El segon intent de mesura només es porta a terme si en el primer intent no s'ha aconseguit. Si no es produeix aquest segon intent, es realitza un slot d'espera. Per això, el consum del període depèn del número d'unitats presents en el sistema i de si és possible comunicar-se amb les que estan presents al primer intent. Per a fer els càlculs de consum sempre s'ha considerat que el sistema funciona en absència d'interferències i que, per tant, tots els missatges de les unitats presents es reben al primer intent. Pel que fa al nombre d'unitats presents els càlculs es fan pels dos extrems de consum:

- 1 sola unitat present (consum màxim, ja que per les no presents s'intenta rebre 2 cops).
- 7 unitats presents (consum mínim, ja que totes estan presents i per tant suposem que es reben totes les temperatures al primer intent).

A la part superior de la següent figura es pot veure la forma d'ona de la transmissió de missatges des de l'emissió dels 3 missatges de sincronisme fins a la conclusió de la recepció de les temperatures de les unitats. A la part inferior pot veure's la forma d'ona de la caiguda de tensió al shunt. Les úniques unitats presents són la 1 i la 6, això es pot veure ja que no es fa el segon intent de mesura i per tant el consum durant aquell slot és menor.

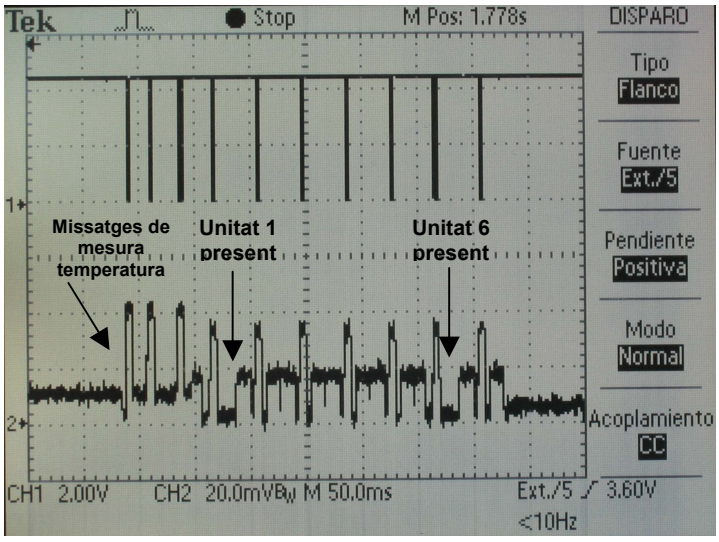


Figura C.13. Mesures de consum i senyal TX durant el període de recepció de temperatures

Per estudiar més en detall un slot on la unitat està present i un on no hi està es pot veure la següent figura:

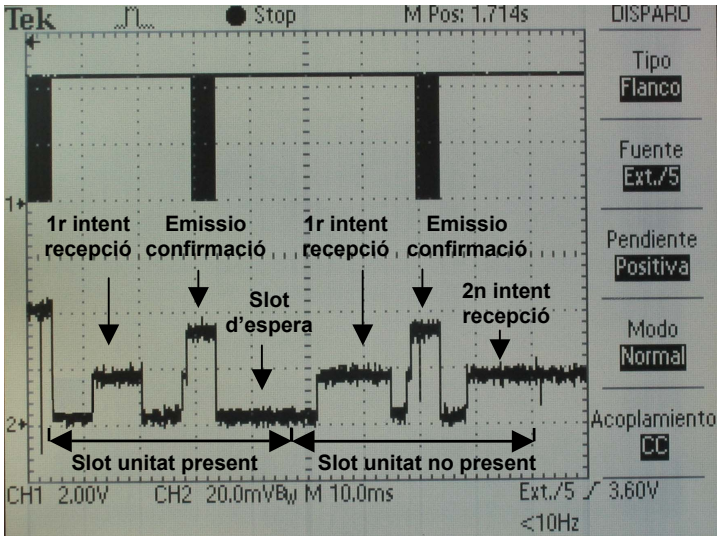


Figura C.14. Mesures de consum i senyal TX en un slot amb una unitat present i una no present

Slot amb la unitat present

El consum és el següent:

- 22mA durant 12ms (1r intent de recepció) [4]
- 1.2mA durant 5.3ms (el que resta per completar l'slot de 13.3ms)
- 22mA durant 1ms (preparació de transmissió)
- 34mA durant 5.2ms (transmissió)
- 1.2mA durant 7.1ms (el que resta per completar l'slot de 13.3ms)
- 1.2 durant 13.3ms (slot d'espera ja que no es fa el 2n intent de recepció de la temperatura)

Això fa un consum de 10mA durant 40ms

[4] El temps de recepció depèn del grau de dessincronització entre la unitat sensora i la màster

Slot amb la unitat no present

El consum és el següent:

- 22mA durant 12ms (1r intent de recepció) . Aquest temps és la pràctica totalitat de l'slot ja que no s'aconsegueix rebre la temperatura
- 1.2mA durant 1.3ms (el que resta per completar l'slot de 13.3ms)
- 22mA durant 1ms (preparació de transmissió)
- 34mA durant 5.2ms (transmissió)
- 1.2mA durant 7.1ms (el que resta per completar l'slot de 13.3ms)
- 22mA durant 12ms (1r intent de recepció) . Aquest temps és la pràctica totalitat de l'slot ja que no s'aconsegueix rebre la temperatura
- 1.2mA durant 1.3ms (el que resta per completar l'slot de 13.3ms)

Això fa un consum de 18,5mA durant 40ms

Per tant el consum d'un període de recepció de temperatures on només una unitat està present és de: 17mA durant 280ms. I el consum d'un període de recepció de temperatures on totes les unitats estan presents és de: 10mA durant 280ms.

Per fer els càlculs del consum total s'utilitza el consum d'un període de recepció de temperatures on hi ha 4 unitats presents i 3 no presents (13,5mA).

Consum mitjà de la unitat màster

El consum de la unitat màster en un període de funcionament (període entre mesures de temperatura) depèn de la freqüència en que es facin aquestes mesures:

Mesures cada 30s. El consum es calcula ponderant:

- 15.6mA durant 40ms (emissió dels 3 missatges de mesura de temperatura)
- 10mA durant 1550ms (mesura de temperatura)
- 15.6mA durant 40ms (emissió dels 3 missatges de sincronisme)
- 13,5mA durant 280ms (recepció temperatures unitats, considerem 4 presents i 3 no presents)
- 3.5mA durant 30s. (gestió interfície d'usuari)

Això fa un total de 4mA

Mesures cada 1 minut. El consum es calcula ponderant:

- 15.6mA durant 40ms (emissió dels 3 missatges de mesura de temperatura)
- 10mA durant 1550ms (mesura de temperatura)
- 15.6mA durant 40ms (emissió dels 3 missatges de sincronisme)
- 13,5mA durant 280ms (recepció temperatures unitats, considerem 4 presents i 3 no presents)
- 3.5mA durant 30s. (gestió interfície d'usuari)
- 15.6mA durant 40ms (emissió dels 3 missatges de sincronisme)
- 3.5mA durant 30s. (gestió interfície d'usuari)

Això fa un total de 3,7mA

Mesures cada 5 minuts. El consum es calcula ponderant:

- 15.6mA durant 40ms (emissió dels 3 missatges de mesura de temperatura)
- 10mA durant 1550ms (mesura de temperatura)
- 15.6mA durant 40ms (emissió dels 3 missatges de sincronisme)
- 13,5mA durant 280ms (recepció temperatures unitats, considerem 4 presents i 3 no presents)
- 3.5mA durant 30s. (gestió interfície d'usuari)
- 9 vegades
    - 15.6mA durant 40ms (emissió dels 3 missatges de sincronisme)
    - 3.5mA durant 30s. (gestió interfície d'usuari)

Això fa un total de 3.6mA
Per a períodes més llargs el consum s'estabilitza a 3.5mA

El consum de la unitat màster pot considerar-se que, amb les condicions en que s'ha fet els càlculs, oscil·la entre els 3,5mA i els 4mA.

C.4.- Mesura del desfasament entre unitats

En aquest apartat s'exposa la metodologia i els resultats de les mesures fetes per estimar el desfasament entre les diferents unitats.

C.4.1.- Introducció

Les unitats sensores han d'emetre els seus missatges de temperatura dins unes finestres de temps precises per tal de no interferir amb els missatges d'altres unitats. Cada unitat sensora porta a terme la comptabilització del temps amb el TIMER80 del microcontrolador. Aquest *timer* funciona a partir del senyal de *clock* del cristall de quars de la placa del microcontrolador. La precisió del cristalls és finita així que, amb el temps, es van dessincronitzant uns respecte els altres. Per aquest motiu, tal i com ja s'ha discutit en la memòria, és necessari que la unitat màster emeti missatges de sincronització per tal que les unitat sensores obtinguin punts de referència temporals a partir dels quals tornar a començar la comptabilització del temps.

C.4.2.- Metodologia

Per tal de mesurar els desfasaments s'executa en dues unitats diferents un mateix programa. Aquest genera un senyal consistent en un tren de polsos de període 13.33ms. El tren de polsos es genera amb el TIMER80 a partir del moment en que s'aplica un senyal extern comú a les dues unitats. Així, la generació del tren de polsos s'inicia a la vegada, i per tant al començament els senyals estan en fase.

Visualitzant per canals diferents de l'oscil·loscopi els senyals generats per les dues unitats es pot mesurar com augmenta el seu desfasament.

La descripció i el codi del programa es pot veure a continuació:

El programa realitza les següents tasques:

1.- Configuracions

2.- No continua fins a trobar el pin P07 a nivell baix

3.- Configura el timer80 per generar interrupcions cada 13.33ms sense que s'executi el seu servei d'interrupció.

4.- Genera un flanc de pujada amb senyal de trigger (P30) per a sincronitzar l'oscil·loscopi amb que es fan les mesures.

5.- Posa el senyal a mesurar (P31) a "1"

6.- Decrementa el comptador C per a tenir P31 a "1" durant un cert temps

7.- Posa el senyal a mesurar (P31) a "0"

8.- Espera en mode *halt* a que s'acabi l'slot de 13.33ms

9.- Torna a progpr per a repetir cíclicament la generació de polsos amb el senyal P31

El codi del programa es pot veure a continuació:

```
          MOV PM0,#11111111b ;configura el P0 com a entrada
          MOV PM3,#00000000b ;configura el P3 com a sortida
          MOV PCC,#00000000b ;fcpu=fx
          CLR1 P3.0          ;trigger a 0
          BT P0.7,$          ;quan es pitja el polsador passa a la següent instrucció

          CLR1 TMMK80        ;habilita les interrupcions del TIMER80
          DI                 ;deshabilita el servei d'instruccions
          CLR1 TCE80         ;atura el TIMER80
          CLR1 TMC80.2       ;configura el TIMER80 amb Tmax=13.33ms
          SET1 TMC80.1
          MOV CR80,#0FFh     ;posa el Compare Register a FF
          SET1 TCE80         ;posa en marxa el TIMER80

          SET1 P3.0          ;trigger a 1
progpr
          SET1 P3.1          ;posa el senyal a mesurar a "1"
          MOV C,#255d        ;espera un cert temps decrementant el comptador C
          DBNZ C,$
          CLR1 P3.1          ;posa el senyal a mesurar a "0"

          HALT               ;espera en mode halt a que s'acabi l'slot de 13.33ms
          CLR1 TMIF80        ;esborra el flag de requeriment d'interrupció

          BR progpr          ;torna a progpr
```

C.4.3.- Resultats

Si es mesuren i grafiquen els desfasaments obtinguts considerant diferents períodes de temps, s'obté es següent resultat:

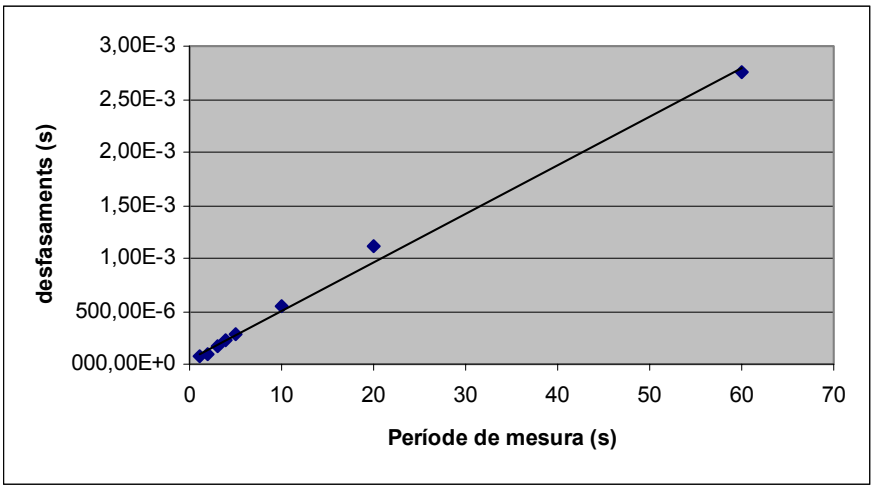


Figura C.15. Gràfic dels desfasaments de les unitat sensores en funció del temps

Es pot comprovar, tal i com era d'esperar, que els desfasaments són aproximadament lineals en el temps.
El pendent d'aquesta recta és de $45.65 \cdot 10^{-6}$ s/s.
Per tant les unitats sensores és dessincronitzen de l'ordre d'uns 2.7ms/minut.

Per a cada període mesura s'han promitjat els resultats de 4 mesures. Tot i això, aquests resultats són fruit d'una sola sessió de mesures i amb dues unitats concretes a temperatura ambient. El fet de ser mesures d'una sola sessió, influeix en que el comportament a altres temperatures pot diferir del mesurat, aquest fenomen és especialment acusat quan els cristalls es troben a temperatures diferents entre ells.

Per tots aquest fets, els resultats aquí exposats, s'han de prendre amb precaució i ser utilitzats només com a referència per a tenir un ordre de magnitud del desfasament entre les unitats. A l'hora d'implementar protocols que tinguin en compte aquests desfasaments, és recomanable utilitzar coeficients de seguretat amplis.

D.- Pressupost

El present projecte no pretén superar la primera fase de prototipus. En aquesta fase s'han utilitzat components dissenyats només per a tasques de desenvolupament i no pensats per a la incorporació en un producte acabat.

Aquest és el cas de l'nRF401-LOOPKIT, es tracta del kit de desenvolupament del transceiver nRF401. S'han utilitzat aquests kits però, en el producte definitiu (fora de l'abast d'aquest projecte), s'utilitzaria el *transceiver* nRF401. Aquest és bastant més econòmic però per poder ser utilitzar requereix el disseny i la construcció de la circuiteria annexa, tasca no recomanada per a les primeres fases de desenvolupament sinó que sol quedar relegada a estadis més avançats.

Una altre cost important de la fase de prototipus és la confecció de les plaques de circuit imprès. Aquestes han estat construïdes al Laboratori Comú d'Enginyeria Mecànica de la Universitat Politècnica de Catalunya situat a l'edifici de l'Escola Tècnica superior d'Enginyeria Industrial de Barcelona. La tecnologia de la que es disposa allà és una tecnologia per fabricar plaques de prototipus de manera que el cost és superior al que tindria la fabricació de grans sèries de plaques per a la incorporació en un producte definitiu.

A la vista de tot això, el cost dels prototipus construïts, és molt superior al que tindrien elements amb la mateixa funcionalitat però en fases més de desenvolupament més avançades.

Per avaluar el cost d'una unitat màster i d'una unitat sensora convé dividir el cost en els diferents elements que les integren:

- La placa del microcontrolador.
- El mòdul de radiofreqüència.
- La placa de l'LCD.
- La placa de configuració de les unitats sensores.

En aquest costos no s'hi ha inclòs els costos de la mà d'obra, tant per a la construcció com per al seu desenvolupament.

Els costos dels materials dels diferents circuits es poden veure a continuació:

Cost dels components de la placa del microcontrolador:

component	quantitat	(€/u)	preu (€)
placa circuit imprès	1	24	24
uPD789074	1	4	4
DS18B20	1	2	2
connector mascles 5x2 pins	2	0.15	0.3
connector mascle 7x2 pins	1	0.22	0.22
tira pins	1	0.23	0.23
cristall quars 4.915MHz	1	0.9	0.9
condensador tàntal 10uF	1	1.4	1.4
polsador	1	0.1	0.1
regleta	1	0.64	0.64
portapiles 3 piles AA	1	0.6	0.6
altres	1	0.3	0.3
			34.69

Taula D.1. Cost dels components de la placa del microcontrolador

El cost total de la placa del microcontrolador és pot xifrar en uns 35€. En aquest cos s'hi ha inclòs el portapiles per a alimentar la placa però no les piles. L'apartat altres inclou els preus de les resistències SMD, dels condensadors SMD i del díode.

Cost dels components del mòdul de radiofreqüència:

component	quantitat	(€/u)	preu (€)
nRF-LOOPKIT	1/6	147,32	23.57
connector femella cable pla 5x2	1	0.16	0.16
altres	1	0.3	0.3
			24.03

Taula D.2. Cost dels components del mòdul de radiofreqüència

L'nRF-LOOPKIT té un preu de 147.32€ i costa de 6 mòduls de radiofreqüència, per tant el cost de cada mòdul és de 23.57€. En el cost del mòdul de RF es pot xifrar en uns 24€, en aquest cost s'hi ha inclòs el del cable de connexió i el del connector amb la placa del microcontrolador.

Cost de la placa de l'LCD:

component	quantitat	(€/u)	preu (€)
LCD PC1602-F	1	10	10
placa topos	1	4	4
microinterruptor 4	1	0.64	0.64
microinterruptor 1	1	0.32	0.32
polsador	4	0.26	1.04
LED verd quadrat	1	0.31	0.31
condensador tàntal 10uF	1	1.4	1.4
potenciòmetre	1	0.5	0.5
regleta	1	0.64	0.64
tira pins	1	0.23	0.23
connector femella cable pla 5x2	2	0.16	0.32
connector femella cable pla 10x2	2	0.23	0.46
altres	1	0.8	0.8
			20.66

Taula D.3. Cost dels components de la placa de l'LCD

En el cost de la placa de l'LCD es pot xifrar en uns 20€, en aquest cost s'hi ha inclòs els dels dos connectors amb la placa del microcontrolador. L'apartat altres inclou el cost de les resistències i dels cables de connexió.

Cost de la placa de configuració de les unitats sensores

component	quantitat	(€/u)	preu (€)
placa topos	1/4	4	1
LED	2	0.1	0.2
microinterruptor 4	1	0.64	0.64
condensador tàntal 10uF	1	1.4	1.4
altres	1	0.8	0.8
			4.04

Taula D.4. Cost dels components de la placa de configuració de les unitats sensores

Una placa de configuració de les unitats sensores utilitza aproximadament una quarta part de placa de topos. El cost de la placa de configuració de les unitats sensores pot xifrar-se en uns 4€, aquest cost inclou el de les resistències utilitzades així com els dels cables de connexió.

Un cop avaluats els costos dels diferents elements que integren les unitats pot procedir-se al càlcul del cost d'una unitat màster i d'una unitat sensora:

Cost d'una unitat màster:

La unitat màster incorpora: una placa del microcontrolador, una placa de l'LCD i un mòdul de radiofreqüència.

circuit	quantitat	(€/u)	preu (€)
Placa del microcontrolador	1	34.69	34.69
Placa de l'LCD	1	20.66	20.66
Mòdul de radiofreqüència	1	24.03	24.03
			78.38

Taula D.5. Cost de la unitat màster

El cost d'una unitat màster pot xifrar-se en un 78€

Cost d'una unitat sensora:

Una unitat sensora incorpora: una placa del microcontrolador, una placa de configuració de les unitats sensores i un mòdul de radiofreqüència.

circuit	quantitat	(€/u)	preu (€)
Placa del microcontrolador	1	34.69	34.69
Placa de configuració	1	4.04	4.04
Mòdul de radiofreqüència	1	24.03	24.03
			62.76

Taula D.6. Cost d'una unitat sensora

El cost d'una unitat sensora pot xifrar-se en uns 62€

El cost d'un mòdul de RF és d'uns 23€, en canvi el cost del *transceiver* que incorpora és de només 4€. Per tant, el cost de les unitat sensores i màster en fases més avançades de desenvolupament podria rebaixar-se, pel cap baix, uns 15€ en concepte de mòdul de RF. El preu de les plaques de circuit imprès en la fase de prototipus és de 24€, però és d'esperar que construïdes amb tècniques de producció a més gran escala aquest preu pogués rebaixar-se de l'ordre de 15 o 20€. En fases més avançades de desenvolupament, el cost d'una unitat màster podria rondar els 40€ mentre que el cost d'una unitat sensora podria estar en torn dels 30€.

E.- Impacte Ambiental

Per avaluar l'impacte ambiental del present projecte cal diferenciar dos tipus d'impacte. El primer fa referència a l'impacte de la fabricació dels circuits impresos i dels circuits integrats, així com de la influència en el medi un cop acabada la seva vida útil. El segon es relaciona amb l'emissió de radiacions electromagnètiques per part del mòduls de radiofreqüència que integra el sistema.

E.1.- Impacte de la fabricació dels circuits integrats i dels circuits impresos

En la fabricació de circuits integrats i circuits impresos intervenen molts elements metàl·lics com ara alumini, coure, ferro, kapton (fulla de poliamida), Monel 404, níquel, acer inoxidable, acer i altres [Arnaldos J, 1998, p. 285-295]. Aquests materials, durant la seva fabricació i manipulació han estat recoberts amb olis protectors per a la conservació en front d'agents corrosius. En el moment de la utilització, aquests elements metàl·lics s'han de rentar per eliminar les seves capes protectores. Històricament la neteja s'efectuava amb un vapor desengreixant anomenat Freó TMS (CFC-113, metil alcohol i additius). En un primer esforç es va eliminar l'ús del CFC-113, reemplaçant-lo per TCA (1-1-1 tricloroetà). L'any 1990 es recomanà evitar l'ús tant del CFC-113 com del TCA, ara es plantegen diverses alternatives com ara terpens, barreges d'hidrocarburs alcohol isopropil i sabons aquosos. Cadascún d'aquests elements substitutius presenta avantatges i inconvenients, alguns són inflamants, altres són escassos i el sabó està limitat per polítiques ja existents de respecte al medi ambient.

Un altre element contaminant en el procés de construcció de circuits electrònics és el plom. Actualment és encara un dels principals elements en el procés de soldatge, hi ha una tendència a substituir-lo entre d'altres per adhesius conductors. Aquests adhesius contenen metalls com or, plata o coure, que cal reciclar. Avui en dia encara es continua utilitzant més el plom ja que els elements substitutius són substancialment més cars. En el procés de soldadura de les plaques del projecte s'ha fet servir una aleació d'estany i plom que per tant, un cop acabada la vida útil dels circuits, caldria tractar adientment.

Les unitats sensores del sistema s'alimenten amb piles elèctriques, és ben sabut per tots la toxicitat per al medi ambient dels elements que contenen. Per aquest motiu és del tot necessari que, un cop s'hagin esgotat, siguin tractades adientement. Una bona manera de fer-ho és depositant-les en contenidors habilitats per a tal efecte.

E.1.1.- Recuperació i tractament del materials utilitzats

Els circuits electrònics incorporen materials que, un cop acabada la seva vida útil, han de ser recuperats i tractats ja que representen un perill per al medi ambient. Exemples d'aquest materials són el plom, el níquel, el cadmi,... Per tal de poder tractar aquest materials és necessari dur a terme un procés de desmuntatge dels circuits. El procés és costós i difícil, en molts casos es fa manualment, encara que s'està començant a automatitzar.

Amb el materials recuperats hi ha dos possibles tractaments. El primer consisteix en el reciclatge, quan el valor dels materials és elevat, el procés de reciclatge pot arribar a ser rendible econòmicament. El segon es basa en l'abocament controlat dels materials recuperats.

E.2.- Impacte de les radiacions electromagnètiques

Hi ha gran quantitat d'estudis i informes dels efectes de les radiacions electromagnètiques en la salut, sense que cap d'ells extregui cap conclusió definitiva de si aquestes poden produir o no efectes perjudicials. Molts d'ells es basen en estudis experimentals a curt i mitjà termini sobre diverses poblacions exposades a diferents tipus de radiació [Huidobro,J. 2003, p. 303-312].

Les ones de ràdio i les microones utilitzades en els sistemes de comunicació són ones electromagnètiques que, per la seva freqüència (0.1MHz – 300GHz), pertanyen al grup de les radiacions no ionitzants. És a dir que no són capaces de produir ions a l'interactuar amb els àtoms d'un material. Això és així ja que per poder ionitzar l'estructura atòmica d'una molècula és necessari una energia superior als 12.4eV, en radiofreqüència i microones aquesta energia està compresa entre els $1.24 \cdot 10^{-9}$ eV i els $1.24 \cdot 10^{-3}$ eV [Arnaldos J, 1998, p. 277]. L'energia llindar per ionitzar les molècules només s'assoleix per freqüències superiors a uns $3 \cdot 10^{15}$ Hz que pertanyen a l'ultraviolat. Per aquest motiu els possibles efectes biològics de la radiofreqüència i les microones són només de tipus tèrmic.

El transceiver nRF401 emet en les freqüències de 433.92MHz i 434.33MHz i transmet amb una potència màxima de 10dBm. Aquesta potència és la que subministra el transceiver a l'antena, però la potència radiada és molt inferior ja s'ha de tenir en compte l'eficiència de l'antena. A l'apartat "Càlcul de l'abast" a la pàgina 52 s'han calculat aquestes eficiències, essent la major d'elles (la de l'antena més gran) de $\eta = -19dB$ (0.013). Per tant la potència radiada en la direcció de màxima direccionalitat és:

$$P_{radiada} = P_{transceiver} \cdot \eta_{antena} = 0.13mW = -8.9dBm \qquad \textbf{(Eq. E.1)}$$

El fabricant declara que els seus circuit compleixen la normativa EN 300 220-1 V1.2.1 de l'Institut Europeu d'Estàndards en Telecomunicacions (ETSI), que bàsicament limita la potència màxima radiada en la banda ISM d'operació del transceiver a 10dBm.

Existeixen vàries normes, estàndards i recomanacions internacionals que fan referència a l'exposició electromagnètica en la gamma de radiofreqüència i microones. Els límits solen ser bastant semblants i es basen en les recomanacions de la Organització Mundial de la Salut, de l'Associació Internacional per a la Protecció de les Radiacions (IRPA) i de l'ANSI (American National Standard Institute) entre d'altres.

Tal i com ja s'ha dit, els possibles efectes d'aquestes radiacions, al ser no ionitzants, es limiten a efectes tèrmics. El paràmetre que s'utilitza per avaluar aquests efectes sobre el éssers vius, i en particular sobre el cos humà, és l'anomenada taxa d'absorció específica (TAE). Es mesura en W/Kg i és una mesura de l'energia electromagnètica que absorbida per la unitat de massa biològica en la unitat de temps. La TAE és un paràmetre difícil d'avaluar i per això es defineixen altres paràmetres més operatius anomenats valors llindar o nivells de referència. El més habitual és el nivell de referència per a la densitat de potència equivalent d'ona plana i s'expressa en W/m^2 [Huidobro, J. 2003, p.303-312].

Arnaldos et al. (1998, p. 278), basant-se en els valor establerts per l'IRPA, fixa el valor límit d'exposició de potència equivalent d'ona plana a un valor (per al rang de ràdio freqüència) de $10W/m^2$.

Segons Huidobro (2003, p. 311), i basant-se en la recomanació 1999/519/CE, per a la freqüència de 433MHz, el nivell de referència per a la densitat de potència equivalent d'ona

plana es fixa en un valor de 2.2W/m^2. Com que aquest valor és més restrictiu que l'anterior serà l'utilitzat com a nivell de referència.

Un nRF401-LOOPKIT, amb l'antena de menors pèrdues, emet una potència de $0.13 \cdot 10^{-3}$W. Per tant, a una distància R, considerant radiació isotròpica, la densitat de potència equivalent d'ona plana és calcula com:

$$\frac{P}{S} = \frac{P_{radiada}}{4\pi R^2} \qquad \textit{(Eq. E.2)}$$

A una distància d'1m la densitat de potència equivalent d'ona plana val 10μW/m^2, 200000 cops per sota del nivell de referència. A una distància de 10cm el seu valor puja fins a 1mW/m^2, però que està encara 2000 cops per sota del nivell de referència.

Amb aquests càlculs concloem que les emissions de radiació electromagnètica del sistemes de radiofreqüència del projecte compleixen de sobres els nivells que marquen les autoritats competents i per tant no haurien de tenir cap tipus d'efecte sobre la salut.

F.- Codi complet del programa implementat en el microcontrolador μPD78F9076

```
;*****************************************************************************************************************
;*                                                                                                               *
;* SISTEMA DE MONITORITZACIÓ REMOTA DE TEMPERATURES MITJANÇANT UN ENLLAÇ BIDIRECCIONAL DE RÀDIOFREQÜÈNCIA        *
;*                                                                                                               *
;*****************************************************************************************************************


          name principal

#include  "Df9076.H"


;---definicio de constants-----------------------------------------------------------------------------------------

;---defincició dels pins del microcontrolador----------------------------------------------------------------------
DQ            EQU   P2.4              ;DQ es el pin P24
DQ_IO         EQU   PM2.4             ;DQ_IO es el bit de PM2 que controla si DQ és entrada o sortida
TXEN          EQU   P1.5              ;TXEN (transmit enable) es el pin P15
PWR_UP        EQU   P1.4              ;PWR_UP es el pin P14
CS            EQU   P1.2              ;CS (chanel selection) es el pin P12

MODE          EQU   P2.0              ;MODE és el pin P20
SET           EQU   P2.3              ;SET és el pin P23
MES           EQU   P2.6              ;MES és el pin P26
MENYS         EQU   P2.7              ;MENYS és el pin P27

LED_VERD_MAST EQU   P3.1              ;LED que s'encen quan el master emet o rep
LED_ROIG_SENS EQU   P3.0              ;LED que s'encen mentre el sensor emet
LED_GROC_SENS EQU   P3.1              ;LED que s'encen quan el sensor està en mode de recepció

En            EQU   P0.6              ;senyal En de la pantalla LCD
RS            EQU   P0.5              ;senyal RS de la pantalla LCD
RW            EQU   P0.4              ;senyal RW de la pantalla LCD
DB7           EQU   P0.3              ;senyal DB7 de la pantalla LCD
DB6           EQU   P0.2              ;senyal DB6 de la pantalla LCD
DB5           EQU   P0.1              ;senyal DB5 de la pantalla LCD
DB4           EQU   P0.0              ;senyal DB4 de la pantalla LCD

M_DB7         EQU   PM0.3             ;bits de control de las característiques (entrada o sortida)
M_DB6         EQU   PM0.2             ;dels pins DB7...DB4
M_DB5         EQU   PM0.1
M_DB4         EQU   PM0.0

;---flags de configuració
SENS_MAST     EQU   CONFIG_SENS_MAST.0 ;bit de configuració de la unitat com a sensor o com a master
LCD_CONNECT   EQU   CONFIG_SENS_MAST.1 ;bit que indica si el sensor te connectada la placa LCD
MODE_TEST     EQU   CONFIG_SENS_MAST.2 ;bit que indica si el master i el sensor han d'entrar en mode
                                       ;de test del cana
;---valors de la variable CONCL_RX
soroll        EQU   00000001b          ;hi ha hagut un error de recepció de byte abans d'haver rebut
                                       ;els dos bytes d'adreça del sistema
err_miss      EQU   00000010b          ;hi ha hagut un error de recepció de byte després
                                       ;d'haver rebut els dos
                                       ;bytes d'adreça del sistema o el codi detector d'errors
                                       ;ha detectat un error
ignorar       EQU   00000011b          ;s'ha rebut un missatge que cal ignorar
miss_mast_OK  EQU   00000100b          ;s'ha rebut correctament un missatge del master
miss_sens_OK  EQU   00000101b          ;s'ha rebut correctament un missatge d'una unitat sensor
adr_1_OK      EQU   00000110b          ;s'ha rebut correctament el primer byte de l'adreça del sistema
adr_2_OK      EQU   00000111b          ;s'ha rebut correctament el segon byte de l'adreça del sistema
parells_OK    EQU   00001000b          ;s'han rebut correctament els bits parells
adr_mast_OK   EQU   00001001b          ;s'ha rebut correctament l'adreça del master
adr_sens_OK   EQU   00001010b          ;s'ha rebut correctament l'adreça del sensor
byte1_OK      EQU   00001011b          ;s'ha rebut correctament el 1er byte del payload d'una unitat sensor
byte2_OK      EQU   00001100b          ;s'ha rebut correctament el 2on byte del payload d'una unitat sensor

;---adreces de RF
adr_sist_1    EQU   10001111b          ;adreça del sistema 1
```

```
adr_sist_2    EQU   01110000b          ;adreça del sistema 2

adr_master    EQU   00000000b          ;adreça master            ;00
adr_master1   EQU   01010101b                                    ;55
adr_master2   EQU   01010101b                                    ;55

adr_sens01    EQU   00000001b          ;adreça del sensor 01     ;01
adr_sens01p   EQU   01010110b                                    ;56
adr_sens01s   EQU   01010101b                                    ;55

adr_sens02    EQU   00000010b          ;adreça del sensor 02     ;02
adr_sens02p   EQU   01010101b                                    ;55
adr_sens02s   EQU   01010110b                                    ;56

adr_sens03    EQU   00000011b          ;adreça del sensor 03     ;03
adr_sens03p   EQU   01010110b                                    ;56
adr_sens03s   EQU   01010110b                                    ;56

adr_sens04    EQU   00000100b          ;adreça del sensor 04     ;04
adr_sens04p   EQU   01011001b                                    ;59
adr_sens04s   EQU   01010101b                                    ;55

adr_sens05    EQU   00000101b          ;adreça del sensor 05     ;05
adr_sens05p   EQU   01011010b                                    ;5A
adr_sens05s   EQU   01010101b                                    ;55

adr_sens06    EQU   00000110b          ;adreça del sensor 06     ;06
adr_sens06p   EQU   01011001b                                    ;59
adr_sens06s   EQU   01010110b                                    ;56

adr_sens07    EQU   00000111b          ;adreça del sensor 07     ;07
adr_sens07p   EQU   01011010b                                    ;5A
adr_sens07s   EQU   01010110b                                    ;56

;---missatges de RF
start_conv    EQU   01010100b          ;(T en ANSI)               54
start_conv1   EQU   10101001b                                    ;A9
start_conv2   EQU   01010101b                                    ;55

sincro1       EQU   01010011b          ;(S en ANSI)               53
sincro1p      EQU   10100110b                                    ;A6
sincro1s      EQU   01010110b                                    ;56

sincro2       EQU   01010101b          ;(U en ANSI)               55
sincro2p      EQU   10101010b                                    ;AA
sincro2s      EQU   01010101b                                    ;55

sincro3       EQU   01010110b          ;(V en ANSI)               56
sincro3p      EQU   10101001b                                    ;A9
sincro3s      EQU   01010110b                                    ;56

temp_OK       EQU   01000011b        ;(C en ANSI)                 43
temp_OK1      EQU   10010110b        ;valor per a RESUL_RX_TEMP1  96
temp_OK2      EQU   01010110b        ;valor per a RESUL_RX_TEMP2  56

temp_NOK      EQU   01100011b        ;(c en ANSI)                 63
temp_NOK1     EQU   10010110b        ;valor per a RESUL_RX_TEMP1  96
temp_NOK2     EQU   01100110b        ;valor per a RESUL_RX_TEMP2  66

;---valors de la variable ERROR_TEMP
no_error      EQU   00000000b        ;no s'ha produït error en la lectura de la temperatura
error_pres    EQU   00000001b        ;s'ha produït error de presència del senor
error_lec_bit EQU   00000010b        ;s'ha produït un error en la lecturad'un bit
error_CRC     EQU   00000100b        ;el codi CRC ha detectat un error

;sincronització sensors
max_num_sens  EQU   7d               ;màxim número de sensors presents en el sistema

;--- definició dels caràcter de l'LCD
dospunts_      EQU 00111010b
espai_         EQU 00100000b
menys_         EQU 00101101b
punt_          EQU 00101110b
grau_          EQU 11011111b
guiobaix_      EQU 01011111b
guio_          EQU 00101101b
claud1_        EQU 01011011b
claud2_        EQU 01011101b

N0_            EQU 00110000b
N1_            EQU 00110001b
N2_            EQU 00110010b
```

```
N3_             EQU 00110011b
N4_             EQU 00110100b
N5_             EQU 00110101b
N6_             EQU 00110110b
N7_             EQU 00110111b
N8_             EQU 00111000b
N9_             EQU 00111001b
A_              EQU 01000001b
B_              EQU 01000010b
C_              EQU 01000011b
D_              EQU 01000100b
E_              EQU 01000101b
F_              EQU 01000110b
G_              EQU 01000111b
H_              EQU 01001000b
I_              EQU 01001001b
J_              EQU 01001010b
K_              EQU 01001011b
L_              EQU 01001100b
M_              EQU 01001101b
N_              EQU 01001110b
O_              EQU 01001111b
P_              EQU 01010000b
Q_              EQU 01010001b
R_              EQU 01010010b
S_              EQU 01010011b
T_              EQU 01010100b
U_              EQU 01010101b
V_              EQU 01010110b
W_              EQU 01010111b
X_              EQU 01011000b
Y_              EQU 01011001b
Z_              EQU 01011010b
a_              EQU 01100001b
b_              EQU 01100010b
c_              EQU 01100011b
d_              EQU 01100100b
e_              EQU 01100101b
f_              EQU 01100110b
g_              EQU 01100111b
h_              EQU 01101000b
i_              EQU 01101001b
j_              EQU 01101010b
k_              EQU 01101011b
l_              EQU 01101100b
m_              EQU 01101101b
n_              EQU 01101110b
o_              EQU 01101111b
p_              EQU 01110000b
q_              EQU 01110001b
r_              EQU 01110010b
s_              EQU 01110011b
t_              EQU 01110100b
u_              EQU 01110101b
v_              EQU 01110110b
w_              EQU 01110111b
x_              EQU 01111000b
y_              EQU 01111001b
z_              EQU 01111010b

;---vectors d'interrupcio-----------------------------------------------------------------------------------

          rseg INTVECT

          org 0h
          dw  resetvect   ;vector d'interrupcio del RESET

          org 0Ah         ;vector d'interrupcio INTP2
          dw  intp2vect

          org 0Ch         ;vector d'interrupcio de final de recepcio UART
          dw  intsrvect

          org 0Eh         ;vector d'interrupcio de final de transmissio UART
          dw  intstvect

          org 14h         ;vector d'interrupció TIMER80
          dw ti80vect

          org 16h         ;vector d'interrupció TIMER90
          dw ti90vect
```

```
;---taula de convesió dels decimals binaris a codi LCD-------------------------------------------------------

          org 18h

taula_dec_LCD
          DB
00110000b,00110001b,00110001b,00110010b,00110011b,00110011b,00110100b,00110100b,00110101b,00110110b,00110110b,
00110111b,00111000b,00111000b,00111001b,00111001b

;---stack----------------------------------------------------------------------------------------------------

          stack STACKPP

endstack  ds   20h            ;stack del sistema
orgstack

;---variables------------------------------------------------------------------------------------------------
                rseg DADESAC

;---variables de configuració
Config           ds   1    ;Byte de configuracio del sensor DS18B20 (X X X X X X R1 R0)
CONFIG_SENS_MAST ds   1    ;Byte de configuracio de SENSOR i de MASTER

;---variables auxiliars
AUX              ds   1    ;variables auxiliars
AUX1             ds   1
AUX2             ds   1
AUX3             ds   1

;---bytes scratchpad
LSB              ds   1    ;Byte menys significatiu de la lectura de temperatura
MSB              ds   1    ;Byte més significatiu de la lectura de temperatura
TH               ds   1    ;Límit superior d'alarma de temperatura
TL               ds   1    ;Límit inferior d'alarma de temperatura
config_sens      ds   1    ;Byte de configuracio del sensor
reserv1          ds   1    ;Bytes reservats pel sensor
reserv2          ds   1
reserv3          ds   1
CRC_sens         ds   1    ;CRC calculat pel sensor

;---càlcul CRC
CRC              ds   1    ;CRC calculat pel micro
ACC              ds   1    ;Variable d'acumulacio pel calcul del CRC

;---presa de mostres en la lectura d'un bit de l'scratchpad
CC               ds   1    ;copia del registre C
DD               ds   1    ;copia del registre D
EE               ds   1    ;copia del registre E

;---variables d'errors en la mesura de la temperatura del sensor DS18B20
ERROR_TEMP       ds   1    ;errors en el procés de mesura de la temperatura

;---variables de recepció RF
BYTE_REBUT       ds   1     ;variable on es guarda el byte rebut per l'UART
ESTAT_RX         ds   1     ;variable que controla l'estat de recepcio dels bytes
                            ;00000001b  encara no s'ha rebut cap byte correcte que no sigui del preamble
                            ;00000010b  s'ha rebut el primer byte de l'adreça del sistema
                            ;00000011b  s'ha rebut el segon byte de l'adreça del sistema
                            ;00000100b  s'ha rebut el 1er nibble de l'adreça de dispositiu
                            ;00000101b  s'ha rebut el 2on nibble de l'adreça de dispositiu
                            ;00000110b  s'ha rebut el 1er nibble del 1er byte del payload
                            ;00000111b  s'ha rebut el 2on nibble del 1er byte del payload
                            ;00001000b  s'ha rebut el 1er nibble del 2on byte del payload
                            ;00001001b  s'ha rebut el 2on nibble del 2on byte del payload
                            ;00001010b  s'ha rebut el 1er nibble del 3er byte del payload
                            ;00001011b  s'ha rebut el 2on nibble del 3er byte del payload

CONCL_RX         ds   1     ;00000001b (soroll)      hi ha hagut un error de recepció de byte abans
                            ;                        d'haverrebut els dos bytes d'adreça del sistema
                            ;00000010b (err_miss)    hi ha hagut un error de recepció de byte després d'haver
                            ;                        d'haver rebut els dos bytes d'adreça del sistema
                            ;                        o el codi detector d'errors ha detectat un error
                            ;00000011b (ignorar)     s'ha rebut un missatge que cal ignorar
                            ;00000100b (miss_mast_OK)s'ha rebut correctament un missatge del master
                            ;00000101b (miss_sens_OK)s'ha rebut correctament un missatge d'una unitat sensora
                            ;00000110b (adr_1_OK)    s'ha rebut correctament el primer byte de
                            ;                        l'adreça del sistema
                            ;00000111b (adr_2_OK)    s'ha rebut correctament el segon byte de
                            ;                        l'adreça del sistema
                            ;00001000b (parells_OK)  s'han rebut correctament els bits parells
                            ;00001001b (adr_mast_OK) s'ha rebut correctament l'adreça del master
                            ;00001010b (adr_sens_OK) s'ha rebut correctament l'adreça del sensor
```

```
                              ;00001011b (byte1_OK)    s'ha rebut correctament el 1er byte del payload d'un
                              ;                        missatge d'una unitat sensor
                              ;00001100b (byte2_OK)    s'ha rebut correctament el 2on byte del payload d'un
                              ;                        missatge d'una unitat sensor

SENARS              ds   1    ;variable on es guarda el byte que conté els bits senars i els seus complements
PARELLS             ds   1    ;variable on es guarda el byte que conté els bits parells i els seus complements

BYTE_1              ds   1    ;el primer byte rebut
BYTE_2              ds   1    ;el segon byte rebut
BYTE_3              ds   1    ;el tercer byte rebut
BYTE_DECOD          ds   1    ;és el byte decodificat
N_BYTES_RX          ds   1    ;és el número de bytes rebuts
ADR_RX              ds   1    ;és l'adreça de l'emisor el missatge del qual s'està rebent

;---variable d'emissió RF
ADR_SENSp           ds   1    ;són els bits parells a transmentre de l'adreça del sensor
ADR_SENSs           ds   1    ;són els bits senars a transmentre de l'adreça del sensor

BYTE1_TX_TEMP       ds   1    ;primer byte de l'emissió de temperatura
BYTE2_TX_TEMP       ds   1    ;segon byte de l'emissió de temperatura
BYTE3_TX_TEMP       ds   1    ;tercer byte de l'emissió de temperatura
BYTE4_TX_TEMP       ds   1    ;quart byte de l'emissió de temperatura

SINCROp             ds   1    ;primer byte (bits parells) a emetre del missatge de sincronització corresponent
SINCROs             ds   1    ;segon byte (bits senars) a emetre del missatge de sincronització corresponent

RESUL_RX_TEMP1      ds   1    ;variable que emmagatzema el primer byte a transmetre en funció del resultat
                              ;de la recepció de la temperatura (temp_OK1,temp_NOK1)
RESUL_RX_TEMP2      ds   1    ;variable que emmagatzema el segon byte a transmetre en funció del resultat
                              ;de la recepció de la temperatura (temp_OK2,temp_NOK2)
RX_TEMP_OK          ds   1    ;variable que indica si s'ha rebut correctament la temperatura
                              ;(11111111b s'ha rebut correctament, 00000000b no s'ha rebut correctament)

CONF_RX_TEMP        ds   1    ;CONF_RX_TEMP=11111111b el sensor ha rebut la confirmació de recepció correcta
                              ;de la temperatura
                              ;CONF_RX_TEMP=00000001b el sensor ha rebut la confirmació de recepció incorrecta
                              ;de la temperatura
                              ;CONF_RX_TEMP=00000000b el sensor no ha rebut la confirmació de recepció
                              ;de la temperatura

;---variables dels slots de temporització
FI_SLOT             ds   1    ;variable que indica si ja s'ha acabat un slot de 13ms

N_SLOT_ESP1         ds   1    ;variable on es guarda el número d'slots d'espera del sensor abans
                              ;d'haver d'ementre (N_SLOT_ESP1=n°sensor-1)
N_SLOT_ESP2         ds   1    ;variable on es guarda el número d'slots d'espera del sensor després
                              ;d'ementre (N_SLOT_ESP2=max_num_sens - N_SENSOR)
COMP_SLOT_ESP       ds   1    ;comptador dels slots d'espera del sensor transcorreguts

N_SLOT_RX           ds   1    ;variable on es guarda el número d'slots de recepció a realitzar
COMP_SLOT_RX        ds   1    ;comptador dels slots de recepcio del master transcorreguts

N_SENSOR            ds   1    ;número del sensor
UNIT_ACT_RX         ds   1    ;número d'unitat de la qual actualmet estem rebent la temperatura

;---Variables per a la visualitzacio amb l'LCD
DADA_LCD            ds   1    ;variable on es guarda la dada a mostrar per l'LCD
BINARI              ds   1    ;variable on es guarda la part entera de la temperatura en binari
BCD                 ds   1    ;valor absolut de la part entera de la temperatura
DECIMALS            ds   1    ;variable que conté els 4 bits dels decimals de la temperatura
SIGNE               ds   1    ;expresió en codi LCD del signe de la temperatura " " o "-"
DESEN_LCD           ds   1    ;expresió en codi LCD de les desenes del valor absolut
                              ;de la part entera de la temperatura
UNIT_LCD            ds   1    ;expresió en codi LCD de les unitats del valor absolut
                              ;de la part entera de la temperatura
DECIMAL_LCD         ds   1    ;expresió en codi LCD del decimal de la temperatura
PUNT                ds   1    ;expresió en codi LCD del signe "."

BYTE_1_ACT          ds   1    ;byte 1 de la temperatura (o número d'unitat) de la unitat actual
                              ;que volguem visualitzar (temperatura, màxima o mínima)
BYTE_2_ACT          ds   1    ;byte 2 de la temperatura (o número d'unitat) de la unitat actual
                              ;que volguem visualitzar (temperatura, màxima o mínima)

TAU_TEMP_UNITATS    ds   16d  ;taula de 8*2 bytes on enmagatzemar les temperatures
                              ;de les diferents unitats del sistema
TAU_TEMP_MAX_MIN    ds   24d  ;taula de 8*3 bytes on enmagatzemar les temperatures
                              ;màximes i mínimes de les diferents unitats del sistema

TEMP1_ACT           ds   1    ;variable on es guarda la primera part del valor de la temperatura
                              ;que actualment estem visualitzant
TEMP2_ACT           ds   1    ;variable on es guarda la segona part del valor de la temperatura
```

```
                                ;que actualment estem visualitzant
MAX1_ACT          ds   1        ;variable on es guarda la primera part del valor de la temperatura
                                ;màxima que actualment estem visualitzant
MAX2_ACT          ds   1        ;variable on es guarda la segona part del valor de la temperatura
                                ;màxima que actualment estem visualitzant
MIN1_ACT          ds   1        ;variable on es guarda la primera part del valor de la temperatura
                                ;mínima que actualment estem visualitzant
MIN2_ACT          ds   1        ;variable on es guarda la segona part del valor de la temperatura
                                ;mínima que actualment estem visualitzant

VISU1_ACT         ds   1        ;variable on es guarda la primera part del valor numèric a visualitzar
VISU2_ACT         ds   1        ;variable on es guarda la segona part del valor numèric a visualitzar

CONVERTEIX_LCD    ds   1        ;variable on es guarda el byte a convertir per
                                ;visualitzar-lo en format hexadecimal
PART1             ds   1        ;variable on es guarden la primera part de CONVERTEIX_LCD
PART2             ds   1        ;variable on es guarda la segona part de CONVERTEIX_LCD
LCD1              ds   1        ;variable on es guarda el codi LCD de la primera part de CONVERTEIX_LCD
LCD2              ds   1        ;variable on es guarda el codi LCD de la segona part de CONVERTEIX_LCD

;---interfície d'usuari

UNIT_ACT_VISU     ds   1        ;valor en binari de la unitat que actualment estem visualitzant per l'LCD
ESTAT_INTERFICIE  ds   1        ;codificació de l'estat en que es troba la pantalla LCD
CODI_INTERVAL     ds   1        ;codificació de l'interval de mesura que es visualitza

INTERF_USUARI     ds   1        ;si val 11111111 estem en mode interficie d'usuari si val
                                ;00000000 estem en mode RF
;---variables de debug
INF_P_CONV_TX     ds   1        ;informació del resultat de l'emissió dels polsos
                                ;de conversió de temperatura (màster)
INF_P_CONV_RX     ds   1        ;informació del resultat de la recepció dels polsos
                                ;de conversió de temperatura (sensor)
INF_TEMPER        ds   1        ;informació del resultat de la conversió de temperatura (màster i sensor)
INF_P_SINCRO_TX   ds   1        ;informació del resultat de l'emissió dels polsos de sincronisme (màster)
INF_P_SINCRO_RX   ds   1        ;informació del resultat de la recepcio dels polsos de sincronisme (sensor)
INF_TX_TEMP       ds   1        ;informació del resultat de la última transmissió de la temperatura (sensor)
INF_TX_TEMP1      ds   1        ;informació del resultat del primer intent de
                                ;transmissió de la temperatura (sensor)
INF_TX_TEMP2      ds   1        ;informació del resultat del segon intent de
                                ;transmissió de la temperatura (sensor)
INF_RX_TEMP       ds   1        ;informació de la última recepció de la
                                ;temperatura de la unitat que estem rebent
                                ;actualment (màster). Aquesta informació s'ordena a
                                ;les taules TAU_INF_RX_TEMP 1 i 2
INF_CONFIRM_RX    ds   1        ;informació del resultat de la recepció del missatge de
                                ;confirmació del màster (sensor)

ADR_RX_CONV_TEMP  ds   1        ;adreça del missatge de conversió de temperatura rebut (sensor)
BYTE_1_CONV_TEMP  ds   1        ;primer byte del missatge de conversió de temperatura (sensor)
ADR_RX_SINCRO     ds   1        ;adreça del missatge de sincronització (sensor)
BYTE_1_SINCRO     ds   1        ;primer byte del missatge de sincronització (sensor)
ADR_RX_TEMP       ds   1        ;adreça de l'ultim missatge de temperatura rebut de la unitat escollida (màster)
BYTE_1_TEMP       ds   1        ;primer byte de l'ultim missatge de temperatura rebut
                                ;de la unitat escollida (màster)
BYTE_2_TEMP       ds   1        ;segon byte de l'ultim missatge de temperatura rebut
                                ;de la unitat escollida (màster)
ADR_RX_TEMP1      ds   1        ;adreça del missatge del 1r intent de RX de temperatura
                                ;de la unitat escollida (màster)
BYTE_1_TEMP1      ds   1        ;1r byte del missatge del 1r intent de RX de temperatura
                                ;de la unitat escollida (màster)
BYTE_2_TEMP1      ds   1        ;2n byte del missatge del 1r intent de RX de temperatura
                                ;de la unitat escollida (màster)
ADR_RX_TEMP2      ds   1        ;adreça del missatge del 2n intent de RX de temperatura
                                ;de la unitat escollida (màster)
BYTE_1_TEMP2      ds   1        ;1r byte del missatge del 2n intent de RX de temperatura
                                ;de la unitat escollida (màster)
BYTE_2_TEMP2      ds   1        ;2n byte del missatge del 2n intent de RX de temperatura
                                ;de la unitat escollida (màster)
ADR_RX_CONF       ds   1        ;adreça del missatge de confirmació rebut (sensor)
BYTE_1_CONF       ds   1        ;primer byte del missatge de confirmació rebut (sensor)

TAU_INF_RX_TEMP1  ds   7        ;taula on es guarden els INF_RX_TEMP del primer
                                ;de recepció de les temperatures (el seu comptador és COMP_TAU_INF_TEMP1)
TAU_INF_RX_TEMP2  ds   7        ;taula on es guarden els INF_RX_TEMP del segon
                                ;intent de recepció de les temperatures (el seu comptador és COMP_TAU_INF_TEMP2)

ADR_SIST_REBUDA   ds   1        ;variable que indica si ja s'ha rebut l'adreça del sistema,
                                ;serveix per a determinar si s'han de emmagatzemar els byters rebuts
GEN_ERRORS        ds   1        ;variable on es guraden els codis del tipus d'error a generar
```

```
;---variables de gestió de les sincronitzacions intermitges, i de la periodicitat de la mesura de temperatura

NOVA_MESURA       ds   1     ;variable que indica que si el sensor ha rebut l'ordre del màster
                             ;de realitzar una nova mesura de temperatura (sensor)
PSW_AUX           ds   1     ;còpia del PSW guardat a la pila del màster, es fa dins el servei d'interrupció
                             ;del TIMER80, conté el PSW que es tenia abans de que es produís la interrupció,
                             ;és a dir del lloc de la interfície d'usuari on s'estava
COMP_SLOTS_INTER_TEMP ds   1 ;comptador de "slots d'interval entre sincronitzacions" transcorreguts
                             ;entre mesures de temperatura (sensor i master)
N_SLOTS_INTER_TEMP    ds   1 ;número de "slots d'interval entre sincronitzacions" a fer

ALINEADORA        ds   1     ;variable alineadora
PC_AUX            ds   2     ;còpia del PC guardat a la pila del màster, es fa dins el servei d'interrupció
                             ;del TIMER80, conté el PC que es tenia abans d'entrar en el servei d'interrupció
                             ;és a dir del lloc de la interfície d'usuari on s'estava
COMP_SLOTS_13ms   ds   2     ;compta el número d'slots de 13.33ms (30s = 2250)
SP_PROGPR_MASTER  ds   2     ;SP per poder tornar al programa principal de master quan s'ha de reemetre
                             ;el pols de mesura de temperatures

;---comptadors de 16 bits
COMP_GUARDA_TEMP   ds   2    ;comptador d'escriptura de la taula TAU_TEMP_UNITATS
COMP_TEMP          ds   2    ;comptador de lectura de la taula TAU_TEMP_UNITATS
COMP_MAX_MIN       ds   2    ;comptador de lectura de la taula TAU_TEMP_MAX_MIN

COMP_DEBUG         ds   2    ;és el comptador de TAULA_DEBUG
COMP_TAU_INF_TEMP1 ds   2    ;és el comptador de TAU_INF_RX_TEMP1
COMP_TAU_INF_TEMP2 ds   2    ;és el comptador de TAU_INF_RX_TEMP2

;TAULA_SINC_MES    ds   16
;TAULA_SINC        ds   16
;TAULA_CONF        ds   16
;TAULA_TEMP1       ds   16
;TAULA_TEMP2       ds   16
TAULA_DEBUG1       ds   16   ;tula on es bolca el contingut de TAULA_DEBUG
TAULA_DEBUG        ds   16   ;taula on es guarden el bits rebuts


;---codi-----------------------------------------------------------------------------------------------------------
;-----------------------------------------------------------------------------------------------------------------

          rseg CODI

          org 80h

;---configuracio---------------------------------------------------------------------------------------------------
config

;-----configuració dels ports
          MOV PM0,#00000000b    ;Sotides (0): P00..P06 (DB4,DB5,DB6,DB7,RW,RS,E)
                                ;(adientment per a la placa LCD en mode escriptura) Sortida (0): P07
          MOV PM1,#11001011b    ;sortides (0): P12, P14, P15 (CS, PWR_UP, TXEN) (per a nRF 401)
                                ;entrades microinterruptors de configuració d'unitat (1): P10, P11, P13
          MOV PM2,#11111101b    ;entrades (1): P20, P23, P26 i P27 (MODE, SET, +, -)
                                ;P21 per a Tx (PM21=0 (sortida)), P22 pera Rx (PM22=1 (entrada))
                                ;P24 (DQ) com a entrada
                                ;P25 entrada per al microinterruptor de configuració de canal
          SET1 P2.1             ;P21=1 per a Tx (P21=1)
          MOV PM3,#11111100b    ;sortides(0): P30, P31. P30 és el trigger per al master
                                ;i LED_ROIG_SENS per al sensor     P31 és LED_VERD_MAST i LED_GROC_SENS

;-----inicialització de l'SP
          movw ax,#orgstack     ;ax=origen stack
          movw sp,ax            ;sp=origen stack

;-----configuració  de la velocitat del micro
          MOV PCC,#00000000b    ;fcpu=fx

;-----configuració UART i nRF401
          MOV BRGC20,#00110000b ;19200 bps    (configuracio UART)
          CLR1 PWR_UP           ;posa nRF401 en mode standby

          BF P2.5,CS0           ;a través del P25 tria el canal de transmissió
          BT P2.5,CS1
CS0       CLR1 CS
          BR fi_CS
CS1       SET1 CS
fi_CS

;-----apagat LEDs
          SET1 LED_VERD_MAST    ;apaga el LED verd del master o el groc del sensor
          SET1 LED_ROIG_SENS    ;apaga el LED vermell del sensor
```

```
;-----inicialització variables de debug
          CALL INI_VAR_DEBUG

          EI                    ;habilita les interrupcions



;---TRIA MODE DE FUNCIONAMENT-----------------------------------------------
;   a través dels microswitches tria el mode de funcionament MASTER, SENSOR01, ... SENSOR07
          BF P1.3,conf_0XX
          BT P1.3,conf_1XX
conf_0XX
          BF P1.1,conf_00X
          BT P1.1,conf_01X
conf_00X
          BF P1.0,conf_000
          BT P1.0,conf_001
conf_01X
          BF P1.0,conf_010
          BT P1.0,conf_011


conf_1XX
          BF P1.1,conf_10X
          BT P1.1,conf_11X

conf_10X
          BF P1.0,conf_100
          BT P1.0,conf_101
conf_11X
          BF P1.0,conf_110
          BT P1.0,conf_111



conf_000
          BR config_master
conf_001
          MOV S:N_SENSOR,#01d
          MOV S:ADR_SENSp,#adr_sens01p      ;configura el sistema per a transmentre com a sensor01
          MOV S:ADR_SENSs,#adr_sens01s
          BR config_sensor
conf_010
          MOV S:N_SENSOR,#02d
          MOV S:ADR_SENSp,#adr_sens02p      ;configura el sistema per a transmentre com a sensor02
          MOV S:ADR_SENSs,#adr_sens02s
          BR config_sensor
conf_011
          MOV S:N_SENSOR,#03d
          MOV S:ADR_SENSp,#adr_sens03p      ;configura el sistema per a transmentre com a sensor03
          MOV S:ADR_SENSs,#adr_sens03s
          BR config_sensor
conf_100
          MOV S:N_SENSOR,#04d
          MOV S:ADR_SENSp,#adr_sens04p      ;configura el sistema per a transmentre com a sensor04
          MOV S:ADR_SENSs,#adr_sens04s
          BR config_sensor
conf_101
          MOV S:N_SENSOR,#05d
          MOV S:ADR_SENSp,#adr_sens05p      ;configura el sistema per a transmentre com a sensor05
          MOV S:ADR_SENSs,#adr_sens05s
          BR config_sensor
conf_110
          MOV S:N_SENSOR,#06d
          MOV S:ADR_SENSp,#adr_sens06p      ;configura el sistema per a transmentre com a sensor06
          MOV S:ADR_SENSs,#adr_sens06s
          BR config_sensor
conf_111
          MOV S:N_SENSOR,#07d
          MOV S:ADR_SENSp,#adr_sens07p      ;configura el sistema per a transmentre com a sensor07
          MOV S:ADR_SENSs,#adr_sens07s
          BR config_sensor
```

```
;---programa principal de la unitat sensora-------------------------------------------------------------------

config_sensor
;-------------------------------------------------------------------------------------------------------------
          CALL INICIALIT_SENS           ;realitza les inicialitzacions necessàries per a la unitat sensora

          CALL MISSATGE_INI_SENS        ;escriu per pantalla el missatge inicial del sensor

          CALL ESPERA_3ms               ;s'ha d'assegurar que abans de rebre fa mes de 2ms
                                        ;que que VDD ha passat a "1"

          CALL PRE_SINC
          CALL ESPERA_SINCRO_TEMPER     ;espera el pols de sincronització de la mesura de temperatura

nova_mesura_temperatura

          MOV S:NOVA_MESURA,#00000000b  ;actualitza la variable NOVA_MESURA

          CALL MESURA_TEMP              ;fa una mesura de la temperatura amb el sensor DS18B20

          CALL GENERA_BYTES_TX_TEMP     ;crida la funció que genera els bytes per emetre la temperatura

          CALL PRE_SINC
          CALL ESPERA_SINCRO            ;espera el pols de sincronització
          CMP S:NOVA_MESURA,#11111111b  ;si es rep un pols de sincronització de temperatura
          BZ nova_mesura_temperatura    ;va a nova_mesura_temperatura
          CALL POST_SINC

          MOV A,N_SLOT_ESP1             ;realitza els slots d'espera necessaris abans d'emetre la temperatura
          MOV S:COMP_SLOT_ESP,A
          CALL SLOTS_ESPERA

          CALL SLOT_EMISSIO             ;realitza l'slot d'emissió de la temperatura

          MOV A,N_SLOT_ESP2             ;realitza els slots d'espera necessaris
          MOV S:COMP_SLOT_ESP,A         ;després d'emetre la temperatura
          CALL SLOTS_ESPERA

          CALL INFO_LCD_SENS1           ;ensenya per pantalla l'estat del sistema

          MOV S:COMP_SLOTS_INTER_TEMP,#01h ;inicialitza el comptador d'slots entre mesures de temperatura a 1

inicia_espera
          MOVW AX,#00h                  ;inicialització COMP_SLOTS_13ms
          MOVW S:COMP_SLOTS_13ms,AX
continua_espera
          HALT
          MOV S:FI_SLOT,#00000000b

          CALL INFO_LCD_SENS123         ;ensenya per pantalla l'estat del sistema

          MOVW AX,S:COMP_SLOTS_13ms     ;si COMP_SLOTS_13ms != 2249 incrementa el compt i continua l'espera
          CMPW AX,#2249d                ;si COMP_SLOTS_13ms  = 2249 (han passat 30s - epsilon)
          BNZ incr_comp_slots_13ms      ;va a rebre el pols sincro intermig
          BZ RX_pols_sincro_intermig
incr_comp_slots_13ms
          INCW AX
          MOVW S:COMP_SLOTS_13ms,AX
          BR continua_espera
RX_pols_sincro_intermig
          CLR1 TCE80                    ;atura el TIMER80
          CALL PRE_SINC
          CALL ESPERA_SINCRO            ;espera un pols de sincronització

          CMP S:NOVA_MESURA,#11111111b
          BZ nova_mesura_temperatura    ; -->

          CALL POST_SINC
          CALL INFO_LCD_SENS_INT        ;ensenya per pantalla l'estat del sistema
          INC S:COMP_SLOTS_INTER_TEMP   ;incrementem el comptador d'slots transcorreguts entre temperatures

          BR inicia_espera              ; -->
```

```
;---programa principal màster-------------------------------------------------------------------------------

config_master
;-------------------------------------------------------------------------------------------------------------
          CALL SELEC_ERRORS_MAST

          BF MODE_TEST,skip_mode_test_mast
          BR mode_test_mast
skip_mode_test_mast

          CLR1 P3.0                       ;trigger a 0

          CALL INICIALIT_MAST             ;realitza les inicialitzacions del master

          CALL MISSATGE_INI_MAST          ;escriu per pantalla el missatge d'inicialització del màster

          BT SET,$                        ;mentres el plosador (P23, SET) no està pitjat no es mou d'aquí,
                                          ;quan es pitja passa a la següent instrucció
progpr_master

          SET1 P3.0                       ;trigger a 1

          CALL MISSATGE_MESURA

          MOV PCC,#00000000b              ;fcpu=fx

          MOVW AX,SP                      ;guarda a SP_PROGPR_MASTER la posició de la pila per
          MOVW S:SP_PROGPR_MASTER,AX      ;poder tornar-hi quan s'hagin de resincronitzar les temperatures

          MOV S:INTERF_USUARI,#00000000b  ;configura la variable INTERF_USUARI per a indicar que
                                          ;no ens trobem mode interficie d'usuari
          MOV S:COMP_SLOTS_INTER_TEMP,#1d ;inicialitza a 1 el comptador d'slots entre mesures de temperatura

          CALL POLSOS_CONV_TEMP           ;emet el polsos de conversió de temperatura

          CALL MESURA_TEMP                ;realitza una mesura de temperatura amb el sensor DS18B20

          CALL ESP_MARGE                  ;1.2ms de marge
          CALL ESP_MARGE                  ;1.2ms de marge

          CALL POLSOS_SINCRO              ;crida la funció que emet el pols de sincronització

          CALL SLOTS_RECEPCIO             ;realitza els slots de recepció

          MOVW AX,#00h                    ;inicialització COMP_SLOTS_13ms
          MOVW S:COMP_SLOTS_13ms,AX

          CALL ACTUALITZA_MAX_MIN         ;a partir de les temperatures rebudes actualitza la taula
                                          ;de temperatures màximes i mínimes
          MOV S:INTERF_USUARI,#11111111b  ;configura la variable INTERF_USUARI per a indicar que
                                          ;ens trobem en mode interficie d'usuari
          BR INTERFICIE_USUARI



;-------------------------------------------------------------------------------------------------------------
INICIALIT_SENS
;-------------------------------------------------------------------------------------------------------------
;realitza les inicialitzacions necessàries per a la unitat sensor

          CALL SELEC_ERRORS_SENS        ;llegeix els polsadors i tria els errors a emetre

          BF MODE_TEST,skip_mode_test_sens ;en funció dels polsadors pitjats entra en el mode
          BR mode_test_sens               ;de test de l'enllaç
skip_mode_test_sens

          BT SET,LCD_connectat          ;si el pin SET es troba a 1, la pantalla LCD està connectada
          BR LCD_no_connectat           ;ja que SET té un pull-up extern quan l'LCD està connectat
LCD_connectat                           ;per això es posa LCD_CONNECT a 1
          SET1 LCD_CONNECT              ;si el pin SET es troba a 0 és la placa de configuració
          BR fi_LCD_con_no_con          ;la que està connectada ja que la placa del configuracio
LCD_no_connectat                        ;connecta SET a massa, per això es posa LCD_CONNECT a 0
          CLR1 LCD_CONNECT
          BR fi_LCD_con_no_con
fi_LCD_con_no_con

          SET1 SENS_MAST                ;activa el flag que indica que la unitat
                                        ;es troba configurada com a sensor
          MOV A,N_SENSOR                ;posa a N_SLOT_ESP1 el número d'slots d'espera que
          DEC A                         ;ha de realitzar el sensor
          MOV N_SLOT_ESP1,A             ;abans d'emetre la temperatura N_SLOT_ESP1 = N_SENSOR - 1
```

```
          MOV A,#max_num_sens           ;posa a N_SLOT_ESP1 el número d'slots d'espera que
          SUB A,N_SENSOR                ;ha de realitzar el sensor
          MOV N_SLOT_ESP2,A             ;després d'emetre la temperatura N_SOT_ESP2 = max_num_sens - N_SENSOR

          MOV S:INTERF_USUARI,#00000000b ;configura la variable INTERF_USUARI per a indicar que no ens trobem
                                        ;en mode interficie d'usuari
          RET


;-------------------------------------------------------------------------------------------------------------
INICIALIT_MAST
;-------------------------------------------------------------------------------------------------------------
;---realitza les inicialitzacions del master-------------------------------------------------------------------

          MOV S:N_SLOTS_INTER_TEMP,#4d      ;inicialitzem la variable que conté el número d'slots entre
                                            ;mesures de temperatura a realitzar
          MOVW AX,#TAU_TEMP_UNITATS         ;inicialitzem els comptadors de lectura de les taules
          MOVW S:COMP_TEMP,AX               ;de temperatura i de màximes i mínimes

          MOVW AX,#TAU_TEMP_MAX_MIN
          MOVW S:COMP_MAX_MIN,AX

          MOV S:ESTAT_INTERFICIE,#00000000b ;inicialitzem la variable que conté l'estat de la interf d'usuari
          MOV S:UNIT_ACT_VISU,#00d          ;inicialitzem la variable que conté la unitat a visualitzar
          MOV S:CODI_INTERVAL,#02d          ;inicialitzem la variable que conté l'interval que es visualitza

          CLR1 SENS_MAST                    ;activa el flag que indica que la unitat està config com a master
          MOV S:N_SLOT_RX,#max_num_sens     ;guarda a N_SLOT_RX el número d'slots de recepció a realitzar

          MOV S:INTERF_USUARI,#00000000b    ;configura la variable INTERF_USUARI per a indicar que no
                                            ;ens trobem mode interficie d'usuari
          CALL INICIALITZA_MAX_MIN          ;inicialitza la taula de les temperature màximes i mínimes
                                            ;a les màximes hi posa el nombre més petit  (1000 0000 0000)
                                            ;a les mínimes hi posa el nombre més gran   (0111 1111 1111)
          RET




;*************************************************************************************************************
;* RUTINES DE TEMPORITZACIÓ DE RF                                                                            *
;*************************************************************************************************************


;-------------------------------------------------------------------------------------------------------------
SLOTS_ESPERA    ;(sensor)
;-------------------------------------------------------------------------------------------------------------
;realitza COMP_SLOT_ESP slots d'espera consistents en 3 slots d'espera de 13.33ms

loop_slot_esp
          CMP S:COMP_SLOT_ESP,#0d
          BZ fi_slot_esp

          CALL SLOT_ESPERA

          DEC S:COMP_SLOT_ESP
          BR loop_slot_esp
fi_slot_esp

          RET

;-------------------------------------------------------------------------------------------------------------
SLOT_ESPERA      ;(sensor)
;-------------------------------------------------------------------------------------------------------------
;realitza 1 slot d'espera consistent en 3 slots d'espera de 13.33ms

          HALT
          MOV S:FI_SLOT,#00000000b
          HALT
          MOV S:FI_SLOT,#00000000b
          HALT
          MOV S:FI_SLOT,#00000000b

          RET
```

```
;-----------------------------------------------------------------------------------------------------------
SLOT_EMISSIO      ;(sensor)
;-----------------------------------------------------------------------------------------------------------
;realitza l'emissió, recepció del pols de confirmació i si s'escau reemissió de la temperatura

          CLR1 LED_ROIG_SENS          ;encen el LED vermell del sensor

          CALL ESP_MARGE              ;1.2ms de marge
          CALL ESP_MARGE              ;1.2ms de marge

          CALL PRE_TX_TEMP1
          CALL INTR_ERR_TX_TEMP1      ;si s'escau introdueix un error en el missatge
          CALL TX_TEMP                ;crida la funció d'enviar la temperatura al master
          MOV S:FI_SLOT,#00000000b
          CALL POST_TX_TEMP1

          CALL PRE_CONF
          CALL ESP_CONFIRM_RX         ;rep el missatge de confirmació de recepció de temperatura
          MOV S:FI_SLOT,#00000000b
          CALL POST_CONF

          CMP S:CONF_RX_TEMP,#11111111b ;si s'ha rebut el missatge de confirmació de recepció
          BNZ reemissio_temp            ;correcta de temperatura va a no_reemissio_temp
          BZ no_reemissio_temp          ;si no s'ha rebut el missatge de confirmació de recepció
                                        ;o s'ha rebut el de recepció incorrecta va a reemissio_temp
reemissio_temp
          CALL ESP_MARGE              ;1.2ms de marge

          CALL PRE_TX_TEMP2
          CALL INTR_ERR_TX_TEMP2      ;si s'escau introdueix un error en el missatge
          CALL TX_TEMP                ;crida la funció d'enviar la temperatura al master
          MOV S:FI_SLOT,#00000000b
          CALL POST_TX_TEMP2
          BR fi_reemissio

no_reemissio_temp
          MOV S:INF_TX_TEMP2,#B_
          HALT                        ;inicia un slot d'espera
          MOV S:FI_SLOT,#00000000b
          BR fi_reemissio

fi_reemissio
          SET1 LED_ROIG_SENS          ;apaga el LED vermell del sensor

          RET


;-----------------------------------------------------------------------------------------------------------
SLOTS_RECEPCIO      ;(master)
;-----------------------------------------------------------------------------------------------------------
;realitza els slots de recepció dels missatges de totes les unitats sensores (fa N_SLOT_RX slots)

          MOV S:UNIT_ACT_RX,#01d              ;inicialitzem la variable UNIT_ACT_RX a 01

          MOV A,N_SLOT_RX                     ;posa al COMP_SLOT_RX el número d'slots de recepció a realitzar
          MOV S:COMP_SLOT_RX,A

          MOVW AX,#TAU_INF_RX_TEMP1           ;inicialització del comptador de la taula on es guarden
          MOVW S:COMP_TAU_INF_TEMP1,AX        ;els INFs dels primers intents de recepció de temperatura
          MOVW AX,#TAU_INF_RX_TEMP2           ;inicialització del comptador de la taula on es guarden
          MOVW S:COMP_TAU_INF_TEMP2,AX        ;els INFs dels segons intents de recepció de temperatura

          MOVW AX,#TAU_TEMP_UNITATS           ;inicialització del comptador de la taula on es guarden
          INCW AX                             ;les temperatures
          INCW AX                             ;espai per a la temperatura de la unitat master
          MOVW S:COMP_GUARDA_TEMP,AX

loop_slot_RX
          CMP S:COMP_SLOT_RX,#0d
          BZ fi_slot_RX

          CALL SLOT_RECEPCIO                  ;crida la funció que efectua la recepció d'una unitat

          DEC S:COMP_SLOT_RX
          INC S:UNIT_ACT_RX
          BR loop_slot_RX
fi_slot_RX

          RET
```

```
;-------------------------------------------------------------------------------------------------------------------
SLOT_RECEPCIO      ;(master)
;-------------------------------------------------------------------------------------------------------------------
;efectua l'slot de recepció d'una unitat sensora que consisteix en:
;primer intent de recepció del missatge de temperatura
;emissió del missatge de confirmació de recepció
;si s'escau, segon intent de recepció del missatge de temperatura
;si s'ha rebut la temperatura la guarda a la taula de temperatures

          CALL PRE_TEMP1
          CALL RX_MESURA_TEMP               ;inicia la recepció de la temperatura
          MOV S:FI_SLOT,#00000000b
          CALL POST_TEMP1
          CALL GUARDA_TEMPER                ;si s'ha rebut correctament la temperatura
                                            ;la guarda a TEMP_UNITATS
          CALL ESP_MARGE                    ;1.2ms de marge
          CALL ESP_MARGE                    ;1.2ms de marge

          CALL CONFIRM_RX                   ;emet el missatge de confirmació de recepció
          MOV S:FI_SLOT,#00000000b

          CMP S:RX_TEMP_OK,#11111111b       ;si s'ha rebut correctament la temperatura va a no_nova_recepcio
          BNZ nova_recepcio                 ;si no s'ha rebut correctament va a nova_recepcio
          BZ no_nova_recepcio

no_nova_recepcio                            ;(s'ha rebut correctament la temperatura)
          CALL POST_TEMP2_NO_RECEP

          HALT                              ;deixa passar un slot
          MOV S:FI_SLOT,#00000000b
          BR fi_nova_recepcio
nova_recepcio                               ;(no s'ha rebut correctament la temperatura)

          CALL PRE_TEMP2
          CALL RX_MESURA_TEMP               ;inicia un nou slot de recepció
          MOV S:FI_SLOT,#00000000b
          CALL POST_TEMP2
          MOVW AX,S:COMP_GUARDA_TEMP        ;DEC COMP_GUARDA_TEMP
          DECW AX                           ;serveix per a tornar a escriure la temperatura
          DECW AX                           ;a la mateixa posició de la taula que en l'intent anterior
          MOVW S:COMP_GUARDA_TEMP,AX
          CALL GUARDA_TEMPER                ;si s'ha rebut correctament la temperatura al segon intent
                                            ;la guarda a TEMP_UNITATS
          BR fi_nova_recepcio

fi_nova_recepcio

          RET



;-------------------------------------------------------------------------------------------------------------------
INI_COMP_13ms
;-------------------------------------------------------------------------------------------------------------------
;inicia el comptatge d'un període de 13.33ms amb el TIMER80
          EI

          CLR1 TMMK80            ;habilita les interrupcions del TIMER80

          CLR1 TCE80             ;atura el TIMER80
          CLR1 TMC80.2           ;configura el TIMER80 amb Tmax=13.33ms
          SET1 TMC80.1
          MOV CR80,#0FFh         ;posa el Compare Register a FF
          SET1 TCE80             ;posa en marxa el TIMER80

          RET



;-------------------------------------------------------------------------------------------------------------------
FI_COMP_13ms    ;(master i sensor)
;-------------------------------------------------------------------------------------------------------------------
;finalització de l'slot de 13ms. Hi va en el servei d'interrupció del TIMER80
;si no estem en mode d'interficie d'usuari actualitza la variable FI_SLOT
;per a indicar que s'ha acabat un slot de 13.33ms i surt de la interrupció
;
;si estem en mode d'interficie d'usuari del màster, actualitza el comptador d'slots de 13.33ms
;i si ja han trancorregut 30s emet el missatge de sincronisme a les unitats sensores
;i torna al lloc de la interficie d'usuari on s'havia produït la interrupció

;si a més d'haver transcorregut els 30s s'ha de realitzar una mesura de temperatura
```

```
;enlloc d'emetre el missatge de sincronisme, emet el missatge de sincronització de temperatures
;i va a l'inici del programa recuperant lapila que hi havia allà

          MOV S:FI_SLOT,#11111111b          ;actualitza la variable FI_SLOT per a indicar que s'ha acabat
                                            ;un slot de 13.33ms, això s'ultilitza en les rutines de recepció
          CMP S:INTERF_USUARI,#11111111b    ;si estem en mode d'interf. usuari va a actualitza_comp_sincro
          BZ actualitza_comp_sincro         ;si no estem en mode d'interf. usuari surt del servei d'interrup.
          RETI                              ; ==>

actualitza_comp_sincro                      ;(estem en mode interficie d'usuari)

          MOV S:AUX1,A                      ;guardem còpies de A, X i B
          MOV A,X
          MOV S:AUX2,A
          MOV A,B
          MOV S:AUX3,A

          MOVW AX,S:COMP_SLOTS_13ms         ;si COMP_SLOTS_13ms  = 2250 (han passat 30s)
          CMPW AX,#2250d                    ;emet el pols de sincronització intermig
          BZ emet_pols_sincro               ;si COMP_SLOTS_13ms != 2250 incrementa el comptador
          INCW AX
          MOVW S:COMP_SLOTS_13ms,AX

          MOV A,AUX3                        ;recuperem les còpies de A, X i B
          MOV B,A
          MOV A,AUX2
          MOV X,A
          MOV A,AUX1

          RETI                              ; ==>

emet_pols_sincro

          MOV A,COMP_SLOTS_INTER_TEMP       ;si han transcorregut N_SLOTS_INTER_TEMP slots de l'interval
          CMP A,N_SLOTS_INTER_TEMP          ;entre sincronitzacions va a sincronitza mesures temperatura
          BZ sincronitza_mesures_temp       ;si encara no han transcorregut n slots incrementa el comptador
                                            ;d'slots i emet un pols de sincronització
          INC S:COMP_SLOTS_INTER_TEMP

          MOV PCC,#00000000b                ;fcpu=fx

          MOV S:INTERF_USUARI,#00000000b    ;passem a mode de no interficie d'usuari

          POP AX                            ;RETI
          MOVW S:PC_AUX,AX
          POP PSW
          MOV A,PSW
          MOV S:PSW_AUX,A
                                            ;(tens el PSW d'allà on eres abans de l'interrupció)
          EI                                ;fem EI per tenir el mateix indenpendentment d'allà on venguem

          CLR1 TCE80                        ;atura el TIMER80

          CALL POLSOS_SINCRO

          MOV A,PSW_AUX                     ;tornem a la interrupció primitiva
          MOV PSW,A                         ;(aquella que amb un RETI torna a la interfície d'usuari)
          PUSH PSW
          MOVW AX,S:PC_AUX
          PUSH AX

          MOVW AX,#00h                      ;reseteja el comptador d'slots de 13ms
          MOVW S:COMP_SLOTS_13ms,AX

          MOV S:INTERF_USUARI,#11111111b    ;passem a mode d'interficie d'usuari

          MOV A,AUX3                        ;recuperem les còpies de A, X i B
          MOV B,A
          MOV A,AUX2
          MOV X,A
          MOV A,AUX1


          RETI                              ;torna al lloc de la interf. usuari
                                            ;on s'havia produït la interrupció
sincronitza_mesures_temp

          MOV PCC,#00000000b                ;fcpu=fx

          MOVW AX,S:SP_PROGPR_MASTER        ;recupera la posició de la pila que es tenia a progpr_master
          MOVW SP,AX
```

```
          CLR1 P3.0                         ;trigger a 0

          MOV A,AUX3                        ;recuperem les còpies de A, X i B
          MOV B,A
          MOV A,AUX2
          MOV X,A
          MOV A,AUX1

          BR progpr_master                  ;va a progpr_master


;-----------------------------------------------------------------------------------------------------------------
INI_COMP_7ms_13ms
;-----------------------------------------------------------------------------------------------------------------
;inicia el comptatge d'un període de 7.17ms
          EI

          CLR1 TMMK80            ;habilita les interrupcions del TIMER80

          CLR1 TCE80             ;atura el TIMER80
          CLR1 TMC80.2           ;configura el TIMER80 amb Tmax=13.33ms
          SET1 TMC80.1
          MOV CR80,#138d         ;posa el Compare Register a 138d per a comptar en primer lloc 7.17ms
          SET1 TCE80             ;posa en marxa el TIMER80

          RET




;*****************************************************************************************************************
;* RUTINES DE RF                                                                                                 *
;*****************************************************************************************************************


;-----------------------------------------------------------------------------------------------------------------
STDBY_TX
;-----------------------------------------------------------------------------------------------------------------
;pasa del mode d'standby al mode d'emissió
;configura l'nRF401 i l'UART per a transmetre

          SET1 PWR_UP                ;passa d'stand by al mode d'operacio
          CALL ESPERA_1ms
          SET1 TXEN                  ;es posa en mode d'emissio
          CALL ESPERA_1ms

          MOV ASIM20,#10001000b      ;Tx=ON, Rx=OFF, No parity, 8 bits, 1 stop bit
          CLR1 STMK20                ;habilita interrupcions de final de transmisio UART
          DI                         ;deshabilita el servei d'interrupcio
          RET


;-----------------------------------------------------------------------------------------------------------------
TX_STDBY
;-----------------------------------------------------------------------------------------------------------------
;pasa del mode d'emissió al d'standby
;desactiva l'UART

          MOV ASIM20,#00000000b      ;Tx=OFF, Rx=OFF
          SET1 STMK20                ;deshabilita interrupcions de final de transmisio UART
          CLR1 PWR_UP                ;posem nRF401 en mode stand by
          CLR1 TXEN                  ;passa a recepcio
          RET


;-----------------------------------------------------------------------------------------------------------------
STDBY_RX
;-----------------------------------------------------------------------------------------------------------------
;pasa del mode d'standby al mode de recepció
;configura l'nRF401 i l'UART per a rebre

          CLR1 TXEN                  ;passa d'stand by al mode d'operacio,
          SET1 PWR_UP                ;quan s'estabilitza ja esta en mode recepcio
          CALL ESPERA_3ms

          MOV ASIM20,#01001000b      ;Tx=OFF, Rx=ON, No parity, 8 bits, 1 stop bit
          CLR1 SRMK20                ;habilita interrupcions de final de recepcio UART
          EI
          RET
```

```
;-----------------------------------------------------------------------------------------------------------------
RX_STDBY
;-----------------------------------------------------------------------------------------------------------------
;pasa del mode de recepció al d'standby
;desactiva l'UART

          SET1 SRMK20                ;deshabilita interrupcions de final de recepcio UART
          MOV ASIM20,#00000000b      ;Tx=OFF, Rx=OFF
          CLR1 PWR_UP                ;posem nRF401 en mode stand by
          RET



;-----------------------------------------------------------------------------------------------------------------
POLSOS_CONV_TEMP      ;(màster)
;-----------------------------------------------------------------------------------------------------------------
;emet els 3 polsos de sincronització de mesura de temperatura

          MOV S:INF_P_CONV_TX,#Z_

          CALL POLS_CONV_TEMP          ;crida la funció que emet el pols de sincronització
                                       ;per a la conversió de les temperatures

          CALL INI_COMP_13ms           ;inicia comptatge slots de 13.33ms
          MOV S:FI_SLOT,#00000000b
          EI                           ;inicia un slot d'espera
          HALT
          MOV S:FI_SLOT,#00000000b

          CALL POLS_CONV_TEMP          ;inicia un slot de pols de conversió de temperatura
          EI
          HALT
          MOV S:FI_SLOT,#00000000b

          EI                           ;inicia un slot d'espera
          HALT
          MOV S:FI_SLOT,#00000000b

          CALL POLS_CONV_TEMP          ;inicia un slot de pols de conversió de temperatura
          EI
          HALT
          MOV S:FI_SLOT,#00000000b

          CLR1 TCE80                   ;atura el TIMER80

          MOV S:INF_P_CONV_TX,#A_

          RET


;-----------------------------------------------------------------------------------------------------------------
POLS_CONV_TEMP       ;(màster)
;-----------------------------------------------------------------------------------------------------------------
;emet el un missatge de sincronització de mesura de temperatura

          CLR1 LED_VERD_MAST          ;encen el LED verd del master

          CALL STDBY_TX               ;passa d'standby al mode d'emisió

          SET1 P0.7

          CALL PREAM_ADRSIST          ;emet el preamble i la adreça del sistema

          CALL ADR_MASTER             ;emet l'adreça del master

          MOV A,#start_conv1          ;inicia transmissio del missatge start_conv (1er byte)
          CALL INTR_ERR_SINC_TEMP     ;si s'escau introdueix un error en el missatge
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,#start_conv2          ;inicia transmissio de missatge start_conv (2on byte)
          MOV TXS20,A
          HALT
          CLR1 STIF20


          SET1 STMK20                 ;deshabilita interrupcions de final de transmisio UART
          EI                          ;habilita el servei d'interrupcions

          CALL ESP_BIT_STOP           ;temps necessari per a que s'acabi d'enviar el bit d'stop
```

```
          CALL TX_STDBY               ;passa del mode de transmissió al mode d'standby

          SET1 LED_VERD_MAST          ;apaga el LED verd del master

          RET

;-------------------------------------------------------------------------------------------------------------
POLSOS_SINCRO      ;(màster)
;-------------------------------------------------------------------------------------------------------------
;emet els 3 polsos de sincronització

          CLR1 LED_VERD_MAST            ;encen el LED verd del master

          MOV S:SINCROp,#sincro1p       ;crida la funció que emet el primer pols desincronització
          MOV S:SINCROs,#sincro1s
          CALL INTR_ERR_SINC1           ;si s'escau introdueix un error en el missatge
          CALL POLS_SINCRO

          CALL INI_COMP_13ms            ;inicia el comptatge
          MOV S:FI_SLOT,#00000000b

          HALT
          MOV S:FI_SLOT,#00000000b

          MOV S:SINCROp,#sincro2p       ;crida la funció que emet el primer pols de sincronització
          MOV S:SINCROs,#sincro2s
          CALL INTR_ERR_SINC2           ;si s'escau introdueix un error en el missatge
          CALL POLS_SINCRO
          EI
          HALT
          MOV S:FI_SLOT,#00000000b

          HALT
          MOV S:FI_SLOT,#00000000b

          MOV S:SINCROp,#sincro3p       ;crida la funció que emet el primer pols de sincronització
          MOV S:SINCROs,#sincro3s
          CALL INTR_ERR_SINC3           ;si s'escau introdueix un error en el missatge
          CALL POLS_SINCRO
          EI
          HALT
          MOV S:FI_SLOT,#00000000b

          SET1 LED_VERD_MAST            ;apaga el LED verd del master

          RET


;-------------------------------------------------------------------------------------------------------------
POLS_SINCRO       ;(màster)
;-------------------------------------------------------------------------------------------------------------
;emet un pols de sincronització

          CALL STDBY_TX               ;passa d'standby al mode d'emisió

          SET1 P0.7

          CALL PREAM_ADRSIST          ;emet el preamble i la adreça del sistema

          CALL ADR_MASTER             ;emet l'adreça del master

          MOV A,S:SINCROp             ;inicia transmissio del missatge de sincro (1er byte) que pertoca
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,S:SINCROs             ;inicia transmissio de missatge de sincro (2on byte) que pertoca
          MOV TXS20,A
          HALT
          CLR1 STIF20

          SET1 STMK20                 ;deshabilita interrupcions de final de transmisio UART
          EI                          ;habilita el servei d'interrupcions

          CALL ESP_BIT_STOP           ;temps necessari per a que s'acabi d'enviar el bit
                                      ;d'stop de l'ultim byte
          CLR1 P0.7

          CALL TX_STDBY               ;passa del mode de transmissió al mode d'standby

          RET
```

```
;-------------------------------------------------------------------------------------------------------------
CONFIRM_RX
;-------------------------------------------------------------------------------------------------------------
;emet el missatge de confirmació de recepció de temperatura
;amb el contingut de les variable RESUL_RX_TEMP1 i RESUL_RX_TEMP2

          SET1 P0.7

          CALL STDBY_TX               ;passa d'standby al mode d'emisió

          CALL PREAM_ADRSIST          ;emet el preamble i la adreça del sistema

          CALL ADR_MASTER             ;emet l'adreça del master

          MOV A,RESUL_RX_TEMP1        ;inicia transmissio del missatge de confirmació
                                      ;de recepció de temperatura (1er byte)
          CALL INTR_ERR_CONF          ;si s'escau introdueix un error en el missatge
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,RESUL_RX_TEMP2        ;inicia transmissio del missatge de confirmació
          MOV TXS20,A                 ;de recepció de temperatura (2on byte)
          HALT
          CLR1 STIF20

          SET1 STMK20                 ;deshabilita interrupcions de final de transmisio UART
          EI                          ;habilita el servei d'interrupcions

          CALL ESP_BIT_STOP           ;temps necessari per a que s'acabi d'enviar el bit d'stop
                                      ;d'stop de l'ultim byte
          CLR1 P0.7

          CALL TX_STDBY               ;passa del mode de transmissió al mode d'standby

          ;MOV S:INF_CONFIRM_TX,#A_

          EI
          HALT

          RET


;-------------------------------------------------------------------------------------------------------------
TX_TEMP
;-------------------------------------------------------------------------------------------------------------
;emet un missatge de temperatura amb el contingut de les variables
;ADR_SENSp, ADR_SENSs, BYTE1_TX_TEMP, BYTE2_TX_TEMP, BYTE3_TX_TEMP i BYTE4_TX_TEMP

          CALL STDBY_TX               ;passa de mode stad-by al d'emissió

          SET1 P0.7

          CALL PREAM_ADRSIST          ;emet el preamble i la adreça del sistema

          MOV A,ADR_SENSp             ;inicia transmissio dels bits parells de l'adreça del sensor
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,ADR_SENSs             ;inicia transmissio dels bits senars de l'adreça del sensor
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,BYTE1_TX_TEMP
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,BYTE2_TX_TEMP
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,BYTE3_TX_TEMP
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,BYTE4_TX_TEMP
          MOV TXS20,A
```

```
          HALT
          CLR1 STIF20

          SET1 STMK20                ;deshabilita interrupcions de final de transmisio UART
          EI                         ;habilita el servei d'interrupcions

          CALL ESP_BIT_STOP          ;temps necessari per a que s'acabi d'enviar el bit d'stop
                                     ;de l'ultim byte
          CALL TX_STDBY              ;passa de mode transmissió a mode standby

          CLR1 P0.7

          MOV S:INF_TX_TEMP,#A_

          EI
          HALT

          RET


;-------------------------------------------------------------------------------------------------------------
PREAM_ADRSIST
;-------------------------------------------------------------------------------------------------------------
;emet el preamble i l'adreça del sistema

          MOV TXS20,#55h             ;inicia transmissio (55)
          HALT
          CLR1 STIF20                ;esborra el flag de requeriment d'interrupcio

          MOV TXS20,#0FFh            ;inicia transmissio (FF)
          HALT
          CLR1 STIF20

          MOV TXS20,#adr_sist_1      ;inicia transmissio de l'adreca del sistema 1
          HALT
          CLR1 STIF20

          MOV TXS20,#adr_sist_2      ;inicia transmissio de l'adreca del sistema 2
          HALT
          CLR1 STIF20

          RET


;-------------------------------------------------------------------------------------------------------------
ADR_MASTER
;-------------------------------------------------------------------------------------------------------------
;emet l'adreça del màster

          MOV A,#adr_master1         ;inicia transmissio de l'adreca del master (1er byte)
          MOV TXS20,A
          HALT
          CLR1 STIF20

          MOV A,#adr_master2         ;inicia transmissio de l'adreca del master (2on byte)
          MOV TXS20,A
          HALT
          CLR1 STIF20

          RET


;-------------------------------------------------------------------------------------------------------------
ESPERA_SINCRO_TEMPER
;-------------------------------------------------------------------------------------------------------------
;espera la recepció del missatge de sincronització de mesura de temperatura,
;mentre no el reb continua rebent, quan el reb surt de la funció

          CLR1 LED_GROC_SENS          ;encen el LED groc del sensor

          CALL STDBY_RX               ;passa de mode standby al mode recepció

          SET1 P0.7

          MOV S:ESTAT_RX,#00000001b   ;inicialització de la variable ESTAT_RX

espera_sincro_temp

          HALT                        ;espera a que l'UART acabi de rebre algun byte

          CMP S:CONCL_RX,#soroll      ;si hem rebut un error de recepció de byte abans d'haver rebut
```

```
          BZ espera_sincro_temp        ; els dos bytes d'adreça del sistema, estem rebent soroll

          CMP S:CONCL_RX,#err_miss     ;o hem rebut un error de RX de bytes quan ja hem rebut els dos bytes
          BZ reset_seq_sincro_temp     ;d'adreça del sistema, o el codi detect d'errors ha detectat un error

          CMP S:CONCL_RX,#ignorar      ;hem rebut un missatge a ignorar
          BZ espera_sincro_temp

          CMP S:CONCL_RX,#miss_mast_OK ;s'ha rebut correctament un missatge del master
          BZ sincro_temp_RX

          CMP S:CONCL_RX,#adr_1_OK     ;s'ha rebut correctament el primer byte de l'adreça del sistema
          BZ espera_sincro_temp

          CMP S:CONCL_RX,#adr_2_OK     ;s'ha rebut correctament el segon byte de l'adreça del sistema
          BZ espera_sincro_temp

          CMP S:CONCL_RX,#parells_OK   ;s'han rebut correctament els bits parells
          BZ espera_sincro_temp

          CMP S:CONCL_RX,#adr_mast_OK  ;s'ha rebut correctament l'adreça del master
          BZ espera_sincro_temp

          BR config                    ;si no es compleix cap de les condicions anteriors es fa un reset

sincro_temp_RX                         ;s'ha rebut el missatge del master

          MOV A,ADR_RX                 ;guarda l'adreça del missatge rebut i el seu contingut
          MOV S:ADR_RX_CONV_TEMP,A     ;a les variables de debug corresponents
          MOV A,BYTE_1
          MOV S:BYTE_1_CONV_TEMP,A

          CMP S:BYTE_1,#start_conv     ;comprova si el missatge rebut és de sicronització de conversió
          BZ sincro_temp_OK            ;de temperatura si el missatge és l'esperat va a sincro_temp_OK
                                       ;si el missatge no és l'esperat va reset_seq_sincro_temp
                                       ;per continuar rebent
          MOV S:INF_P_CONV_RX,#B_      ;actualitza la variable INF_P_CONV_RX

reset_seq_sincro_temp                  ;(s'ha rebut un missatge el payload del qual no conté el missatge
          MOV S:ESTAT_RX,#00000001b    ;esperat (start convert) o s'ha detectat un error en la recepció)
          CLR1 S:ADR_SIST_REBUDA.0     ;actualitza la variable ESTAT_RX per a iniciar una nova seqüència
          BZ espera_sincro_temp        ;de recepció del pols de sincronització
                                       ;modifica la variable ADR_SIST_REBUDA per a indicar que no s'ha rebut
                                       ;encara l'adreça del sistema
                                       ;va a espera_sincro_temp per a rebre un nou pols de sincronisme
sincro_temp_OK
          CALL RX_STDBY                ;passa de recepció a standby

          MOV S:INF_P_CONV_RX,#T_      ;actualitza la variable INF_P_CONV_RX

          CLR1 P0.7

          SET1 LED_GROC_SENS           ;apaga el LED groc del sensor

          RET


;-------------------------------------------------------------------------------------------------------------
ESPERA_SINCRO
;-------------------------------------------------------------------------------------------------------------
;espera un missatge de sincronització
;si rep un missatge de sincronització de mesura de temperatura, surt de la funció actualitzant
;la variable NOVA_MESURA de manera que al sortir de la funció és fa una nova mesura de temperatura
;si rep un missatge de sincronisme (sigui el 1r, el 2n o el 3r) es sincronitza amb el màster

          CLR1 LED_GROC_SENS           ;encen el LED groc del sensor
          CALL STDBY_RX                ;passa de mode standby al mode recepció

          SET1 P0.7

          MOV S:ESTAT_RX,#00000001b    ;inicialització de la variable ESTAT_RX

espera_sinc

          HALT                         ;espera a que l'UART acabi de rebre algun byte

          CMP S:CONCL_RX,#soroll       ;si hem rebut un error de recepció de byte abans d'haver rebut
          BZ espera_sinc               ;els dos bytes d'adreça del sistema, estem rebent soroll

          CMP S:CONCL_RX,#err_miss     ;o hem rebut un error de RX de bytes quan ja hem rebut els dos bytes
          BZ reset_seq_sincro          ;d'adreça del sistema,o el codi detector d'errors ha detectat un error
```

```
          CMP S:CONCL_RX,#ignorar      ;hem rebut un missatge a ignorar
          BZ espera_sinc

          CMP S:CONCL_RX,#miss_mast_OK ;s'ha rebut correctament un missatge del master
          BZ sincro_RX

          CMP S:CONCL_RX,#adr_1_OK     ;s'ha rebut correctament el primer byte de l'adreça del sistema
          BZ espera_sinc

          CMP S:CONCL_RX,#adr_2_OK     ;s'ha rebut correctament el segon byte de l'adreça del sistema
          BZ espera_sinc

          CMP S:CONCL_RX,#parells_OK   ;s'han rebut correctament els bits parells
          BZ espera_sinc

          CMP S:CONCL_RX,#adr_mast_OK  ;s'ha rebut correctament l'adreça del master
          BZ espera_sinc

          BR config                    ;si no es compleix cap de les condicions anterior es fa un reset

sincro_RX                              ;(s'ha rebut un missatge del master)

          MOV S:INF_P_SINCRO_RX,#A_    ;actualitza la variable INF_P_SINCRO_RX

          CMP S:BYTE_1,#start_conv     ;comprova si el missatge rebut és de conversió de temperatura
          BZ start_conv_OK             ;si el missatge és l'esperat va a start_conv_OK
          CMP S:BYTE_1,#sincro1        ;comprova si el missatge rebut és el primer de sicronització
          BZ sincro1_OK                ;si el missatge és l'esperat va a sincro1_OK
          CMP S:BYTE_1,#sincro2        ;comprova si el missatge rebut és el segon de sicronització
          BZ sincro2_OK                ;si el missatge és l'esperat va a sincro2_OK
          CMP S:BYTE_1,#sincro3        ;comprova si el missatge rebut és el tercer de sicronització
          BZ sincro3_OK                ;si el missatge és l'esperat va a sincro3_OK

reset_seq_sincro                       ;(s'ha rebut un missatge el payload del qual no conté el miss esperat
          MOV S:ESTAT_RX,#00000001b    ;(sincro1, sincro2 o sincro3) o s'ha detectat un error en la recepció)
          CLR1 S:ADR_SIST_REBUDA.0     ;actualitza la variable ESTAT_RX per a iniciar una nova seqüència
          BR espera_sinc               ;de recepció del pols de sincronització
                                       ;modifica la variable ADR_SIST_REBUDA per a indicar que no s'ha
                                       ;rebut encara l'adreça del sistema
                                       ;va a espera_sinc per a rebre un nou pols de sincronisme

start_conv_OK                          ;(s'ha rebut el pols de conversió de temperatura)
          CALL RX_STDBY                ;passa de recepció a standby
          CLR1 P0.7

          MOV S:INF_P_CONV_RX,#T_      ;actualitza la variable INF_P_CONV_RX

          MOV A,ADR_RX                 ;guarda l'adreça del missatge rebut i el seu contingut
          MOV S:ADR_RX_CONV_TEMP,A     ;a les variables de debug corresponents
          MOV A,BYTE_1
          MOV S:BYTE_1_CONV_TEMP,A

          MOV S:NOVA_MESURA,#11111111b ;actualitza NOVA_MESURA per a indicar que s'ha de
                                       ;realitzar una nova mesura
          SET1 LED_GROC_SENS           ;apaga el LED groc del sensor
          RET

sincro1_OK                             ;(s'ha rebut el primer pols de sincronització)
          CALL RX_STDBY                ;passa de recepció a standby
          CLR1 P0.7

          MOV S:INF_P_SINCRO_RX,#N1_   ;actualitza la variable INF_P_CONV_RX

          MOV A,ADR_RX                 ;guarda l'adreça del missatge rebut i el seu contingut
          MOV S:ADR_RX_SINCRO,A        ;a les variables de debug corresponents
          MOV A,BYTE_1
          MOV S:BYTE_1_SINCRO,A

          CALL INI_COMP_13ms           ;inicia el comptatge de 13.33ms
          MOV S:FI_SLOT,#00000000b

          HALT                         ;4 slots de 13.33ms d'espera
          MOV S:FI_SLOT,#00000000b

          HALT
          MOV S:FI_SLOT,#00000000b

          HALT
          MOV S:FI_SLOT,#00000000b

          HALT
          MOV S:FI_SLOT,#00000000b
```

```
          SET1 LED_GROC_SENS           ;apaga el LED groc del sensor

          RET

sincro2_OK                             ;(s'ha rebut el segon pols de sincronització)
          CALL RX_STDBY                ;passa de recepció a standby
          CLR1 P0.7

          MOV S:INF_P_SINCRO_RX,#N2_   ;actualitza la variable INF_P_CONV_RX

          MOV A,ADR_RX                 ;guarda l'adreça del missatge rebut i el seu contingut
          MOV S:ADR_RX_SINCRO,A        ;a les variables de debug corresponents
          MOV A,BYTE_1
          MOV S:BYTE_1_SINCRO,A

          CALL INI_COMP_7ms_13ms       ;inicia el comptatge de 1 slot de 7.17ms d'espera
          MOV S:FI_SLOT,#00000000b
          HALT
          MOV S:FI_SLOT,#00000000b

          CLR1 TCE80                   ;atura el TIMER80

          CALL INI_COMP_13ms           ;inicia el comptatge de 13.33ms

          HALT                         ;2 slots d'espera
          MOV S:FI_SLOT,#00000000b

          HALT
          MOV S:FI_SLOT,#00000000b

          SET1 LED_GROC_SENS           ;apaga el LED groc del sensor

          RET
sincro3_OK
          CALL RX_STDBY                ;passa de recepció a standby
          CLR1 P0.7

          MOV S:INF_P_SINCRO_RX,#N3_   ;actualitza la variable INF_P_CONV_RX

          MOV A,ADR_RX                 ;guarda l'adreça del missatge rebut i el seu contingut
          MOV S:ADR_RX_SINCRO,A        ;a les variables de debug corresponents
          MOV A,BYTE_1
          MOV S:BYTE_1_SINCRO,A

          CALL INI_COMP_7ms_13ms       ;inicia el comptatge de 1 slot de 7.17ms d'espera
          MOV S:FI_SLOT,#00000000b
          HALT
          MOV S:FI_SLOT,#00000000b

          CLR1 TCE80                   ;atura el TIMER80

          CALL INI_COMP_13ms           ;inicia el comptatge de 13.33ms

          SET1 LED_GROC_SENS           ;apaga el LED groc del sensor

          RET


;-------------------------------------------------------------------------------------------------------------
RX_MESURA_TEMP      ;(sensor)
;-------------------------------------------------------------------------------------------------------------
;espera durant l'slot assignat de 13.33ms la recepció de la temperatura
;si un cop acabat l'slot no s'ha rebut res es surt de la funció
;actualitza la variable RX_TEMP_OK en funció de si s'ha rebut o no la temperatura
;actualitza les variables RESULT_RX_TEMP1 i 2 per a transmetre el missatge
;de recepció correcta o incorrecta segons pertoqui
;si el missatge rebut no és de la unitat que tocava emetre en aquell slot es considera recepció incorrecta

          CALL STDBY_RX                ;passa del mode d'standby al mode de recepció

          SET1 P0.7

          MOV S:ESTAT_RX,#00000001b    ;inicialització de la variable ESTAT_RX

espera_RX_sens

          CMP S:FI_SLOT,#11111111b     ;si s'ha acabat l'slot va a surt_esp_conf_RX
          BZ surt_RX_temp

          HALT
```

```
          CMP S:FI_SLOT,#11111111b     ;si s'ha acabat l'slot va a surt_esp_conf_RX
          BZ surt_RX_temp

          CMP S:CONCL_RX,#soroll       ;si hem rebut un error de recepció de byte abans d'haver rebut
          BZ espera_RX_sens            ;els dos bytes d'adreça del sistema, estem rebent soroll

          CMP S:CONCL_RX,#err_miss     ;o hem rebut un error de RX de bytes quan ja hem rebut els dos bytes
          BZ miss_perdut_RX_temp       ;d'adreça del sistema,o el codi detector d'errors ha detectat un error

          CMP S:CONCL_RX,#ignorar      ;hem rebut un missatge a ignorar
          BZ espera_RX_sens

          CMP S:CONCL_RX,#miss_sens_OK ;s'ha rebut correctament un missatge d'un sensor
          BZ temp_RX_OK

          CMP S:CONCL_RX,#adr_1_OK     ;s'ha rebut correctament el primer byte de l'adreça del sistema
          BZ espera_RX_sens

          CMP S:CONCL_RX,#adr_2_OK     ;s'ha rebut correctament el segon byte de l'adreça del sistema
          BZ espera_RX_sens

          CMP S:CONCL_RX,#parells_OK   ;s'han rebut correctament els bits parells
          BZ espera_RX_sens

          CMP S:CONCL_RX,#adr_sens_OK  ;s'ha rebut correctament l'adreça d'un sensor
          BZ espera_RX_sens

          CMP S:CONCL_RX,#byte1_OK     ;s'ha rebut correctament el primer byte del missatge d'un sensor
          BZ espera_RX_sens

          BR config                    ;si no es compleix cap de les condicions anterior es fa un reset

miss_perdut_RX_temp                    ;(o s'ha detectat un error de byte quan ja s'havien rebut els 2 bytes
          CALL RX_STDBY                ;d'adreça de sistema o el codi detector d'errors ha detectat un error)
                                       ;passa a standby

          MOV S:INF_RX_TEMP,#C_        ;actualitza la variable INF_RX_TEMP

          MOV S:ADR_RX_TEMP,#0FFh      ;inicialitza les varibles de debug corresponents
          MOV S:BYTE_1_TEMP,#0FFh
          MOV S:BYTE_2_TEMP,#0FFh

          MOV S:RESUL_RX_TEMP1,#temp_NOK1  ;posa a RESUL_RX_TEMP1 i RESUL_RX_TEMP2 els valor adients per a
          MOV S:RESUL_RX_TEMP2,#temp_NOK2  ;transmetre el missatge de recepció incorrecta de temperatura
          MOV S:RX_TEMP_OK,#00000000b      ;actualitza la variable RX_TEMP_OK per a indicar que
                                           ;no s'ha rebut correctament la temperatura

          CLR1 P0.7

          EI                               ;espera a que s'acabi l'slot
          HALT

          RET

surt_RX_temp                               ;(s'ha acabat l'slot sense haver rebut bé cap temperatura)
          CALL RX_STDBY                    ;passa de recepció a standby

          MOV S:INF_RX_TEMP,#guiobaix_     ;actualitza la variable INF_RX_TEMP

          MOV S:ADR_RX_TEMP,#0DDh          ;inicialitza les variables de debug
          MOV S:BYTE_1_TEMP,#0DDh
          MOV S:BYTE_2_TEMP,#0DDh

          CLR1 P0.7

          MOV S:RESUL_RX_TEMP1,#temp_NOK1  ;posa a RESUL_RX_TEMP1 i RESUL_RX_TEMP2 els valor adients per a
          MOV S:RESUL_RX_TEMP2,#temp_NOK2  ;transmetre el missatge de recepció incorrecta de temperatura
          MOV S:RX_TEMP_OK,#00000000b      ;actualitza la variable RX_TEMP_OK per a indicar que
          RET                              ;no s'ha rebut correctament la temperatura


temp_RX_OK                                 ;(s'ha rebut correctament una mesura de temperatura)

          CALL RX_STDBY                    ;passa a standby

          MOV A,ADR_RX                     ;si el número de la unitat rebuda no correspon amb l'esperat
          CMP A,UNIT_ACT_RX                ;va a unitat incorrecta
          BNZ unitat_incorrecta

          MOV A,BYTE_2                     ;si els últims 4 bits del missatge no són 0000
          AND A,#00001111b                 ;va a error_fi_missatge
          CMP A,#00000000b
```

```
          BNZ error_fi_missatge

          MOV S:INF_RX_TEMP,#A_            ;actualitza la variable INF_RX_TEMP

          MOV A,ADR_RX                     ;guarda l'adreça i el contingut del missatge rebut
          MOV S:ADR_RX_TEMP,A              ;a les variables de debug
          MOV A,BYTE_1
          MOV S:BYTE_1_TEMP,A
          MOV A,BYTE_2
          MOV S:BYTE_2_TEMP,A

          MOV S:RESUL_RX_TEMP1,#temp_OK1   ;posa a RESUL_RX_TEMP1 i RESUL_RX_TEMP2 els valor adients
          MOV S:RESUL_RX_TEMP2,#temp_OK2   ;per a transmetre el missatge de recepció correcta de temperatura
          MOV S:RX_TEMP_OK,#11111111b      ;actualitza la variable RX_TEMP_OK per a indicar que
                                           ;s'ha rebut correctament la temperatura

          CLR1 P0.7

          EI                               ;espera a que s'acabi l'slot
          HALT

          RET

unitat_incorrecta                          ;(s'ha rebut un missatge correcte però d'una unitat incorrecta)

          MOV S:INF_RX_TEMP,#U_            ;actualitza la variable INF_RX_TEMP

          MOV A,ADR_RX                     ;guarda l'adreça i el contingut del missatge rebut
          MOV S:ADR_RX_TEMP,A              ;a les variables de debug
          MOV A,BYTE_1
          MOV S:BYTE_1_TEMP,A
          MOV A,BYTE_2
          MOV S:BYTE_2_TEMP,A

          MOV S:RESUL_RX_TEMP1,#temp_NOK1  ;posa a RESUL_RX_TEMP1 i RESUL_RX_TEMP2 els valor adients per a
          MOV S:RESUL_RX_TEMP2,#temp_NOK2  ;transmetre el missatge de recepció incorrecta de temperatura
          MOV S:RX_TEMP_OK,#00000000b      ;actualitza la variable RX_TEMP_OK per a indicar que
                                           ;no s'ha rebut correctament la temperatura

          CLR1 P0.7

          EI                               ;espera a que s'acabi l'slot
          HALT

          RET


error_fi_missatge                         ;(s'ha rebut un missatge amb el final de missatge incorrecte)

          MOV S:INF_RX_TEMP,#Y_            ;actualitza la variable INF_RX_TEMP

          MOV A,ADR_RX                     ;guarda l'adreça i el contingut del missatge rebut
          MOV S:ADR_RX_TEMP,A              ;a les variables de debug
          MOV A,BYTE_1
          MOV S:BYTE_1_TEMP,A
          MOV A,BYTE_2
          MOV S:BYTE_2_TEMP,A

          MOV S:RESUL_RX_TEMP1,#temp_NOK1  ;posa a RESUL_RX_TEMP1 i RESUL_RX_TEMP2 els valor adients per a
          MOV S:RESUL_RX_TEMP2,#temp_NOK2  ;transmetre el missatge de recepció incorrecta de temperatura
          MOV S:RX_TEMP_OK,#00000000b      ;actualitza la variable RX_TEMP_OK per a indicar que
                                           ;no s'ha rebut correctament la temperatura

          CLR1 P0.7

          EI                               ;espera a que s'acabi l'slot
          HALT

          RET



;-------------------------------------------------------------------------------------------------------------
ESP_CONFIRM_RX     ;(sensor)
;-------------------------------------------------------------------------------------------------------------
;espera durant l'slot assignat de 13.33ms la recepció del missatge de confirmació de recepció
;de la temperatura si un cop acabat l'slot no s'ha rebut res es surt de la funció
;actualitza la variable CONF_RX_TEMP en funció de si s'ha rebut el missatge de recepció correcta,
;incorrecta, si no s'ha rebut o si s'ha rebut però amb errors

          SET1 P0.7
```

```
          CALL STDBY_RX                ;passa de mode standby al mode recepció

          MOV S:ESTAT_RX,#00000001b    ;inicialització de la variable ESTAT_RX

espera_confir

          CMP S:FI_SLOT,#11111111b     ;si s'ha acabat l'slot va a surt_esp_conf_RX
          BZ surt_esp_conf_RX

          HALT                         ;espera a que l'UART acabi de rebre algun byte

          CMP S:FI_SLOT,#11111111b     ;si s'ha acabat l'slot va a surt_esp_conf_RX
          BZ surt_esp_conf_RX

          CMP S:CONCL_RX,#soroll       ;si hem rebut un error de recepció de byte abans d'haver rebut
          BZ espera_confir             ;els dos bytes d'adreça del sistema, estem rebent soroll

          CMP S:CONCL_RX,#err_miss     ;o hem rebut un error de recepció de bytes
          BZ miss_perdut_RX_conf       ;quan ja hem rebut els dos bytes
                                       ;d'adreça del sistema, o el codi
                                       ;detector d'errors ha detectat un error

          CMP S:CONCL_RX,#ignorar      ;hem rebut un missatge a ignorar
          BZ espera_confir

          CMP S:CONCL_RX,#miss_mast_OK ;s'ha rebut correctament un missatge del master
          BZ confirm_RX

          CMP S:CONCL_RX,#adr_1_OK     ;s'ha rebut correctament el primer byte de l'adreça del sistema
          BZ espera_confir

          CMP S:CONCL_RX,#adr_2_OK     ;s'ha rebut correctament el segon byte de l'adreça del sistema
          BZ espera_confir

          CMP S:CONCL_RX,#parells_OK   ;s'han rebut correctament els bits parells
          BZ espera_confir

          CMP S:CONCL_RX,#adr_mast_OK  ;s'ha rebut correctament l'adreça del master
          BZ espera_confir

          BR config                    ;si no es compleix cap de les condicions anterior es fa un reset


miss_perdut_RX_conf                    ;(o s'ha detectat un error de byte quan ja s'havien rebut els 2 bytes
          CALL RX_STDBY                ;d'adreça de sistema o el codi detector d'errors ha detectat 1 error)
                                       ;passa a standby

          MOV S:CONF_RX_TEMP,#00000000b ;actualitza la variable CONF_RX_TEMP per a indicar que
                                       ;no s'ha rebut la confirmació de recepció de temperatura
          MOV S:INF_CONFIRM_RX,#C_     ;actualitza la variable INF_CONFIRM_RX

          MOV S:ADR_RX_CONF,#0FFh      ;inicialitza les variables de debug
          MOV S:BYTE_1_CONF,#0FFh

          CLR1 P0.7

          EI                           ;espera a que s'acabi l'slot
          HALT

          RET

confirm_RX                             ;(s'ha rebut el miss de confirmació de recepció de la temperatura)
          CLR1 P0.7

          CALL RX_STDBY                ;passa de recepció a standby

          MOV S:INF_CONFIRM_RX,#A_     ;actualitza la variable INF_CONFIRM_RX

          MOV A,ADR_RX                 ;guarda l'adreça i el contingut del missatge rebut
          MOV S:ADR_RX_CONF,A          ;a les variables de debug
          MOV A,BYTE_1
          MOV S:BYTE_1_CONF,A

          CMP S:BYTE_1,#temp_OK        ;comprova si el missatge rebut és de confirmació de recepció correcta
          BZ confirm_OK                ;de temperatura, si el missatge és l'esperat va a confirm_OK

          CMP S:BYTE_1,#temp_NOK       ;comprova si el missatge rebut és de confirm de recepció incorrecta
          BZ confirm_NOK               ;de temperatura, si el missatge és l'esperat va a confirm_NOK

                                       ;(el missatge rebut no és de confirmació de recepció)
          BR confirm_miss_erroni       ;va a confirm_missatge_erroni
```

```
confirm_OK                          ;(s'ha rebut un missatge de confirmació de
                                    ;recepció correcta de la temperatura)
          MOV S:CONF_RX_TEMP,#11111111b ;actualitza la variable CONF_RX_TEMP per a indicar que s'ha rebut
                                    ;la confirmació de recepció correcta de temperatura
          MOV S:INF_CONFIRM_RX,#N1_     ;actualitza la variable INF_CONFIRM_RX

          HALT                      ;espera a que s'acabi el temps de confirmació de recepció (13ms)
          RET

confirm_NOK                         ;(s'ha rebut un missatge de confirmació de recepció incorrecta
                                    ;de la temperatura)
          MOV S:CONF_RX_TEMP,#00000001b ;actualitza la variable CONF_RX_TEMP per a indicar que s'ha rebut
                                    ;la confirmació de recepció incorrecta de temperatura
          MOV S:INF_CONFIRM_RX,#N0_     ;actualitza la variable INF_CONFIRM_RX

          HALT                      ;espera a que s'acabi el temps de confirmació de recepció (13ms)
          RET

surt_esp_conf_RX                    ;(s'ha acabat l'slot sense haver rebut cap missatge correcte)

          MOV S:CONF_RX_TEMP,#00000000b ;actualitza la variable CONF_RX_TEMP per a indicar que no s'ha rebut
                                    ;la confirmació de recepció de temperatura
          MOV S:INF_CONFIRM_RX,#B_      ;actualitza la variable INF_CONFIRM_RX

          MOV S:ADR_RX_TEMP,#0DDh   ;inicialitza les variables de debug
          MOV S:BYTE_1_TEMP,#0DDh

          CLR1 P0.7
          CALL RX_STDBY             ;passa de recepció a standby
          RET

confirm_miss_erroni                 ;(hem rebut un missatge però no és ni de confirmació
                                    ;correcta ni incorrecta)
          MOV S:CONF_RX_TEMP,#00000000b ;actualitza la variable CONF_RX_TEMP per a indicar que no s'ha rebut
                                    ;la confirmació de recepció de temperatura
          MOV S:INF_CONFIRM_RX,#D_      ;actualitza la variable INF_CONFIRM_RX

          HALT                      ;espera a que s'acabi el temps de confirmació de recepció (13ms)

          RET


;-------------------------------------------------------------------------------------------------------------
RX_BYTE_OK
;-------------------------------------------------------------------------------------------------------------
;és el servei d'interrupció de la recepció d'un byte a través de l'UART, és a dir via RF

          MOV A,ASIS20                  ;si s'ha detectat un error de transmissio va a error_RX
          CMP A,#00000000b
          MOV S:AUX,A

          MOV A,RXB20                   ;posa el byte rebut a BYTE_REBUT
          MOV S:BYTE_REBUT,A

          CLR1 SRIF20                   ;esborra el flag de requeriment d'interrupcio

          BT S:ADR_SIST_REBUDA.0,guarda_byte
          BF S:ADR_SIST_REBUDA.0,fi_guarda
guarda_byte
          MOVW AX,S:COMP_DEBUG          ;INC COMP_DEBUG
          CMPW AX,#TAULA_DEBUG+15
          BC inc_comp_debug_00
          BR fi_guarda
inc_comp_debug_00
          INCW AX
          MOVW S:COMP_DEBUG,AX
          MOVW HL,AX                    ;MOV [COMP_DEBUG],BYTE_REBUT
          MOV A,BYTE_REBUT
          MOV [HL],A
fi_guarda

          CMP S:AUX,#00000000b

          BZ no_error_RX
          BR error_RX
no_error_RX

          MOV A,ESTAT_RX                ;(s'ha rebut correctament un byte)
          CMP A,#00000001b              ;1)no s'havia rebut cap byte d'adreca va cap_adr_rebuda
          BZ cap_adr_rebuda
```

```
          CMP A,#00000010b                ;2)si ja s'havia rebut el primer byte d'adreca va a 1_adr_rebuda
          BZ adr1_rebuda

          CMP A,#00000011b                ;3)si ja s'han rebut els dos bytes d'adreca va a 2_adr_rebudes
          ;BZ adr2_rebudes
          BNC adr2_rebudes

;-------------

cap_adr_rebuda
          MOV A,BYTE_REBUT                ;(s'ha rebut correctament un byte sense haver rebut abans cap
          CMP A,#adr_sist_1               ;altre byte d'adreca (ESTAT_RX = 00000001b))
          BNZ errorRX1                    ;si el byte rebut es el byte d'adreça n°1 es prepara per a rebre
          INC S:ESTAT_RX                  ;el proper byte (ESTAT_RX = 00000010b)
          MOV S:CONCL_RX,#adr_1_OK        ;si el byte rebut no és el byte d'adreça n°1 va a error_RX
          RETI                            ;==>
errorRX1  BR error_RX

adr1_rebuda
          MOV A,BYTE_REBUT                ;(s'ha rebut correctament un byte despres d'haver rebut el 1r
          CMP A,#adr_sist_2               ;byte d'adreca (ESTAT_RX = 00000010b))
          BNZ errorRX2                    ;si el byte rebut es el byte d'adreca n°2 es prepara per a rebre
          INC S:ESTAT_RX                  ;el proper byte (ESTAT_RX = 00000011b)
          MOV S:CONCL_RX,#adr_2_OK        ;si el byte rebut no es el byte d'adreca n°2 va a error_RX

          SET1 S:ADR_SIST_REBUDA.0

          RETI                            ;==>
errorRX2  BR error_RX

adr2_rebudes
          INC S:ESTAT_RX                  ;hem rebut un nibble(1er o 2n)d'un  byte d'adreça de dispositiu
          BF S:ESTAT_RX.0,lbl_parells     ;o de payload, actualitza ESTAT_RX per a rebre el següent nibble
          BT S:ESTAT_RX.0,lbl_senars      ;si hem rebut nibble dels bits parells del byte va a lbl_parells
                                          ;si hem rebut nibble dels bits senars del byte va a llbl_senars

lbl_parells                               ;(hem rebut el nibble dels bits parells)
          MOV A,BYTE_REBUT                ;guardem el byte que conté els bits parells a PARELLS
          MOV S:PARELLS,A
          MOV S:CONCL_RX,#parells_OK
          RETI                            ;==>


lbl_senars                                ;(hem rebut el nibble dels bits senars)
          MOV A,BYTE_REBUT                ;posem el byte que conté els bits senars a SENARS
          MOV S:SENARS,A

;comprovació d'error de recepció amb el codi detector d'errors

          MOV A,SENARS                    ;posem a C X0X0X0X0 on X són els "bits senars complementats"
          AND A,#01010101b                ;descomplementats
          XOR A,#01010101b
          ROL A,1
          MOV C,A
          MOV A,SENARS
          AND A,#10101010b                ;posa a A X0X0X0X0 on X són els "bits senars sense complementar"
          CMP A,C                         ;comparem A i C
          BNZ error_RX3
          BR skip_e_RX3                   ;el codi detector d'errors ha detectat un error
error_RX3 BR error_RX                     ;en la recepció dels bits senars, salta a error_RX
skip_e_RX3

          MOV A,PARELLS                   ;posa a B X0X0X0X0 on X són els "bits parells complementats"
          AND A,#01010101b                ;descomplementats
          XOR A,#01010101b
          ROL A,1
          MOV B,A
          MOV A,PARELLS
          AND A,#10101010b                ;posa a A X0X0X0X0, X són els "bits parells sense complementar"
          CMP A,B                         ;comparem A i B
          BNZ error_RX4                   ;el codi detector d'errors ha detectat un error
          BR skip_e_RX4                   ;en la recepció dels bits parells, salta a error_RX
error_RX4 BR error_RX
skip_e_RX4
          ROR A,1                         ;posa a BYTE_DECOD el byte original recuperat
          ADD A,C                         ;a partir del senars i dels parells
          MOV S:BYTE_DECOD,A

;-------------

          CMP S:ESTAT_RX,#00000101b       ;si el byte llegit és un d'adreça de dispositiu va a
```

```
          BZ byte_adr_disp                ;byte_adr_disp  per a discriminar si es tracta d'un missatge del
          BR byte_dades                   ;master o d'un sensor, si és una dada va a
                                          ;byte_dades per a llegir-la
byte_adr_disp
          CMP S:BYTE_DECOD,#00000000b     ;(el byte rebut és una adreça de dispositiu)
          MOV S:N_BYTES_RX,#0d            ;inicialitza la variable N_BYTES_RX a 0
          BZ miss_master                  ;si el missatge és del master va a miss_master
          BNZ miss_sensor                 ;si el missatge és d'un sensor va a miss_sensor

miss_master                               ;(el missatge rebut és del master)
          BF SENS_MAST,ignora_miss        ;si la unitat es master i ha rebut un misstge d'un master,
          MOV A,BYTE_DECOD                ;va a ignora_miss
          MOV S:ADR_RX,A                  ;es guarda l'adreça del master a la variable ADR_RX
          MOV S:CONCL_RX,#adr_mast_OK
          RETI                            ;==>
miss_sensor                               ;(el missatge rebut és d'una unitat sensor)
          BT SENS_MAST,ignora_miss        ;si la unitat és sensor i ha rebut el missatge d'un sensor, va a
          MOV A,BYTE_DECOD                ;ignora_miss, es guarda l'adreça del sensor a la variable ADR_RX
          MOV S:ADR_RX,A
          MOV S:CONCL_RX,#adr_sens_OK
          RETI                            ;==>
byte_dades                                ;(el byte rebut és un byte de dades)
          BT SENS_MAST,lectur_sensor      ;si la unitat és sensor va a lectur_sensor
          BF SENS_MAST,lectur_master      ;si la unitat és master va a lectur_master

ignora_miss                               ;o la unitat està configurada com a sensor
                                          ;i ha rebut un missatge d'un altre sensor
                                          ;o la unitat està configurada com a master
                                          ;i ha rebut un missatge d'un altre master
          MOV S:CONCL_RX,#ignorar
          MOV S:ESTAT_RX,#00000001b
          CLR1 S:ADR_SIST_REBUDA.0
          MOVW AX,S:COMP_DEBUG            ;INC COMP_DEBUG
          CMPW AX,#TAULA_DEBUG+15
          BC inc_comp_debug_01
          BR fi_inc_comp_debug01
inc_comp_debug_01
          INCW AX
          MOVW S:COMP_DEBUG,AX
          MOVW HL,AX                      ;MOV [COMP_DEBUG],22h
          MOV A,#22h
          MOV [HL],A
fi_inc_comp_debug01
          RETI                            ;==>

;-------------

lectur_sensor                             ;(la unitat és un sensor) llegeix missatges del master
          MOV A,BYTE_DECOD                ;guarda el byte rebut a la variable BYTE_1
          MOV S:BYTE_1,A

          MOV S:CONCL_RX,#miss_mast_OK    ;actualitza adientment la variable CONCL_RX
          RETI                            ;==>
lectur_master                             ;(la unitat és master) llegeix missatges d'una unitat sensor
          MOV A,N_BYTES_RX

          CMP A,#0d                       ;si no hem rebut cap byte de dades (N_BYTES_RX=0) anem a
          BZ lbl_byte1                    ;llegir el primer byte
          CMP A,#1d                       ;si hem rebut un byte de dades (N_BYTES_RX=1) anem a
          BZ lbl_byte2                    ;llegir el segon byte

lbl_byte1
          MOV A,BYTE_DECOD                ;guarda el 1er byte rebut a BYTE_1
          MOV S:BYTE_1,A
          INC S:N_BYTES_RX                ;es prepara per a llegir el proper byte
          MOV S:CONCL_RX,#byte1_OK
          RETI                            ;==>
lbl_byte2
          MOV A,BYTE_DECOD                ;guarda el 2on byte rebut a BYTE_2
          MOV S:BYTE_2,A
          MOV S:CONCL_RX,#miss_sens_OK    ;actualitza adientment la variable CONCL_RX
          RETI                            ;==>
;-------------

error_RX
          MOV A,ESTAT_RX
          CMP A,#00000001b
          BZ soroll_rebut                 ;si es detecta un error en la recepcio del byte i encara no hem
                                          ;rebut cap byte d'adreça es que rebem soroll
          CMP A,#00000010b                ;si es detecta un error en la recepcio del byte pero ja hem
          BZ soroll_rebut                 ;rebut el primer byte de l'adreça del sistema va a soroll_rebut
```

```
          BR error_miss                     ;si ja hem rebut els dos bytes de l'adreça del sistema i es
                                            ;detecta  un error va a error_miss
soroll_rebut                                ;s'ha detectat un error de recepcio mentre no hem acabat
          MOV S:CONCL_RX,#soroll            ;de rebre els bytes d'adreca, i per tant estem rebent soroll
          MOV S:ESTAT_RX,#00000001b
          CLR1 S:ADR_SIST_REBUDA.0
          RETI                              ;===>

error_miss                                  ;o s'ha detectat un error de recepcio de byte un cop ja
                                            ;hem rebut els dos bytes d'adreca dels sistema
                                            ;o el codi detector d'errors ha detectat un error en els bytes
                                            ;de l'adreça de dispositiu o d'un byte del payload
          MOV S:CONCL_RX,#err_miss
          MOV S:ESTAT_RX,#00000001b
          CLR1 S:ADR_SIST_REBUDA.0
          MOVW AX,S:COMP_DEBUG              ;INC COMP_DEBUG
          CMPW AX,#TAULA_DEBUG+15
          BC inc_comp_debug_02
          BR fi_inc_comp_debug_02
inc_comp_debug_02
          INCW AX
          MOVW S:COMP_DEBUG,AX
          MOVW HL,AX                        ;MOV [COMP_DEBUG],00h
          MOV A,#00h
          MOV [HL],A
fi_inc_comp_debug_02
          RETI                              ;==>




;*****************************************************************************************************************
;* RUTINES DE MESURA DE TEMPERATURA DEL SENSOR DS18B20                                                           *
;*****************************************************************************************************************


;-----------------------------------------------------------------------------------------------------------------
MESURA_TEMP
;-----------------------------------------------------------------------------------------------------------------
;fa dos intents de mesura de temperatura del DS18B20
;si en el primer intent el codi CRC no detecta error, espera eltemps equivalent a lasegona mesura
;si el codi CRC detecta errors fa una segona mesura

          CLR1 LED_VERD_MAST            ;encen el LED verd del master o el groc del sensor

          MOV S:INF_TEMPER,#Z_          ;inicialitza la variable INF_TEMPER
          MOV S:ERROR_TEMP,#00000000b   ;inicialitza la variable ERROR_TEMP per indicar que no s'ha
                                        ;produit cap error
          CALL CONV_LECT                ;fa una mesura de temperatura

          CMP S:ERROR_TEMP,#no_error    ;si hi ha hagut un error fa una segona mesura
          BNZ segona_mesura             ;si no hi ha hagut error de CRC va a espera sincronit
          BZ espera_sincronit

segona_mesura
          MOV S:ERROR_TEMP,#00000000b   ;inicialitza la variable ERROR_TEMP per indicar que no s'ha
                                        ;produit cap error
          CALL CONV_LECT                ;fa una segona mesura de temperatura
          CMP S:ERROR_TEMP,#no_error    ;si s'ha tornat a produir un error de CRC va error_mesura
          BNZ error_mesura              ;si no va a fi_mesura
          BZ fi_mesura
error_mesura                            ;(s'ha tornat a produir un error en la lectura)
          MOV S:INF_TEMPER,#C_

          MOVW AX,#TAU_TEMP_UNITATS     ;emmagatzema a les dues primeres posicions de la taula
          MOVW HL,AX                    ;TAU_TEMP_UNITATS 01111111 i 11111111
          MOV A,#01111111b              ;aquests codis indiquen que no hi ha dada correcta de temperatura
          MOV [HL],A                    ;i es visualitzaran com a "--.-"
          INCW HL
          MOV A,#11110000b
          MOV [HL],A

          RET

fi_mesura                               ;(la segona mesura ha estat correcta)
          MOV S:INF_TEMPER,#B_

          CALL SCRT_VISU                ;passa les dades de temperatura de format SCRATCHPAD a format VISU

          MOVW AX,#TAU_TEMP_UNITATS     ;emmagatzema la temperatura en format visu a les dues primeres
          MOVW HL,AX                    ;posicions de la taula TAU_TEMP_UNITATS
```

```
          MOV A,VISU1_ACT
          MOV [HL],A
          INCW HL
          MOV A,VISU2_ACT
          MOV [HL],A
          SET1 LED_VERD_MAST              ;apaga el LED verd del master o el groc del sensor

          RET

espera_sincronit                          ;(la primera mesura ha estat correcta)
          MOV S:INF_TEMPER,#A_

          CALL SCRT_VISU                  ;passa les dades de temperatura de format SCRATCHPAD a format VISU

          MOVW AX,#TAU_TEMP_UNITATS       ;emmagatzema la temperatura en format visu a les dues primeres
          MOVW HL,AX                      ;posicions de la taula TAU_TEMP_UNITATS
          MOV A,VISU1_ACT
          MOV [HL],A
          INCW HL
          MOV A,VISU2_ACT
          MOV [HL],A

          CALL TEMPS_CONV_LECT             ;espera el mateix temps que es trigaria en fer una segona mesura

          SET1 LED_VERD_MAST               ;apaga el LED verd del master o el groc del sensor

          RET


;-----------------------------------------------------------------------------------------------------------------
CONV_LECT
;-----------------------------------------------------------------------------------------------------------------
;fa que el sensor DS18B20 faci una conversió de temperatura i llegeix el resultat
;el procés de conversió està encapsulat mitjançant el TIMER90 de manera que dura 760ms
;el procés de lectura està encapsulat amb el TIMER90 de manera que a l'acabar entra en mode halt
;fins que han transcorregut 15ms des de l'inici del procés de lectura (el procés real dura
;en condicions favorables uns 8.25ms)
;així s'aconseguiex que tot el procés duri 775ms

           CALL INI_TMR90_760ms

           CALL CONVERSIO          ;crida la funció que envia l'ordre de conversió de temperatura al sensor
           HALT

           CLR1 TMIF90             ;esborra el flag de requeriment d'interrupcio
           SET1 TMMK90             ;desactiva interrupcions TIMER90
           EI

           CALL INI_TMR90_15ms

           CALL LECTURA            ;cridala funció que lleeix les dades del sensor
           HALT

           CLR1 TMIF90             ;esborra el flag de requeriment d'interrupcio
           SET1 TMMK90             ;desactiva interrupcions TIMER90
           EI

           RET


;-----------------------------------------------------------------------------------------------------------------
CONVERSIO
;-----------------------------------------------------------------------------------------------------------------
;emet l'ordre de conversió de la temperatura (inicialització + skip ROM + ConvertT)

          CALL INICIALIT        ;inicialitzacio
          CALL SKIP_ROM
          CALL CONVERT_T        ;ordre d'inici de conversio (CC)

          RET

;-----------------------------------------------------------------------------------------------------------------
LECTURA
;-----------------------------------------------------------------------------------------------------------------
;emet l'ordre de lectura de la temperatura (inicialització + skip ROM + Read scratchpad)
;a més llegeix les dades de l'scratchpad i les verifica amb el codi CRC i les guarda en memòria

          CALL INICIALIT        ;inicialitzacio
          CALL SKIP_ROM
          CALL LLEGIR_SCRT      ;llegeix tot l'scratchpad i el guarda en memoria

          RET
```

```
;-------------------------------------------------------------------------------------------------------------
INICIALIT
;-------------------------------------------------------------------------------------------------------------
;iniciaitza el sensor
;Si s'obte el senyal de presència surt de la rutina
;Si no s'obte el senyal de presència actualitza adientment la variable ERROR_TEMP, es continua
;el procés de mesura però els resultat que s'obtinguin no es tindran en compte

          MOV PCC,#00000010b            ;fcpu=fx/4

          CLR1 DQ_IO                    ;DQ es de sortida
          CLR1 DQ                       ;pull-down del bus

          MOV B,#122d                   ;compta 600us (no exactes)
          DBNZ B,$

          SET1 DQ_IO                    ;DQ en alta impedancia (alliberament bus)


presencia
          MOV B,#12d                    ;compta 67.5us (68.2us)
          DBNZ B,$
          NOP                           ;per quadrar temps
          NOP

          BF DQ,pres_OK                 ;si DQ=0 Inicialitzacio OK, si DQ=1 repeteix inicialitzacio


pres_NOK
          MOV B,#109d                   ;compta 530us
          DBNZ B,$

          OR S:ERROR_TEMP,#error_pres   ;si no s'ha detectat presència del sensor s'actualitza
                                        ;adientment la variable ERROR_TEMP
          RET                           ;es continua executant tota la funció de lectura de la temperatura
                                        ;però els resultats que es produeixin no es tindran en compte

pres_OK
          MOV B,#109d                   ;compta 530us
          DBNZ B,$

          MOV PCC,#00000000b            ;fcpu=fx

          RET

;-------------------------------------------------------------------------------------------------------------
SKIP_ROM
;-------------------------------------------------------------------------------------------------------------
;emet l'ordre skip ROM (CCh)

          CALL WRITE0           ;CC
          CALL WRITE0
          CALL WRITE1
          CALL WRITE1
          CALL WRITE0
          CALL WRITE0
          CALL WRITE1
          CALL WRITE1

          RET


;-------------------------------------------------------------------------------------------------------------
CONVERT_T
;-------------------------------------------------------------------------------------------------------------
;emet l'ordre Convert_T (44h)

          CALL WRITE0           ;44
          CALL WRITE0
          CALL WRITE1
          CALL WRITE0
          CALL WRITE0
          CALL WRITE0
          CALL WRITE1
          CALL WRITE0

          RET
```

```
;-----------------------------------------------------------------------------------------------------------------
LLEGIR_SCRT
;-----------------------------------------------------------------------------------------------------------------
;emet l'ordre de llegir l'scratchpad i el guarda en memoria i verifica el codi CRC
;si el byte de CRC coincideix amb el calculat surt de la funció
;si no coincideix actualitza adientment la variable ERROR_TEMP

          CALL WRITE0                     ;BE  emet ordre READ_SCRATCPAD
          CALL WRITE1
          CALL WRITE1
          CALL WRITE1
          CALL WRITE1
          CALL WRITE1
          CALL WRITE0
          CALL WRITE1

          CALL LLEGIR_BYTE                ;llegeix cada byte de l'scratchpad i el guarda en memoria
          MOV LSB,A
          CALL LLEGIR_BYTE
          MOV MSB,A
          CALL LLEGIR_BYTE
          MOV TH,A
          CALL LLEGIR_BYTE
          MOV TL,A
          CALL LLEGIR_BYTE
          MOV config_sens,A
          CALL LLEGIR_BYTE
          MOV reserv1,A
          CALL LLEGIR_BYTE
          MOV reserv2,A
          CALL LLEGIR_BYTE
          MOV reserv3,A
          CALL LLEGIR_BYTE
          MOV CRC_sens,A


CALCUL_CRC                                ;verifica el byte de CRC
                                          ;tarda 3.73ms per 9 bytes a fx/4  0.93ms a fx
INI_CRC
          MOV S:CRC,#00000000b            ;inicialitzem el contingut del registre de desplaçament
          MOV C,#9d                       ;inicialitzem el comptador per comptar 9 bytes
          MOVW HL,#LSB
CRC_BYTE
          MOV A,[HL]                      ;fem una copia del byte a tractar a ACC
          MOV ACC,A
          MOV B,#8d                       ;inicilitzem el comptador per tractar 8 bits
LOOP_CRC
          XOR A,CRC                       ;ACC xor CRC ->A
          RORC A,1                        ;CY= LSB(CRC) xor LSB(ACC)
          MOV A,CRC
          BNC SKIP_XOR                    ;si [LSB(CRC) xor LSB(ACC)]=0 no cal fer xor
          XOR A,#00011000b                ;si [LSB(CRC) xor LSB(ACC)]=1 fem CRC xor 00011000b
SKIP_XOR
          RORC A,1                        ;rotem el registre de desplaçament (CRC)
          MOV CRC,A
          MOV A,ACC                       ;rotem ACC per tenir al LSB(ACC) el proper bit a entrar
          ROR A,1
          MOV ACC,A
          DBNZ B,LOOP_CRC                 ;encara no hem tractat els 8 bits saltem a LOOP_CRC

          INCW HL                         ;ja hem tractat els 8 bits incrementem l'apuntador
                                          ;als bytes a tractar
          DBNZ C,CRC_BYTE                 ;saltem a LOOP_CRC

TEST_CRC
          CMP S:CRC,#00000000b            ;(ja s'ha acabat el càlcul del CRC)
          BZ TEST_OK                      ;Si CRC=0 acabba la funcio
TEST_ERR
          OR S:ERROR_TEMP,#error_CRC      ;(CRC!=0) actualitza la variable ERROR_CRC per a indicar
                                          ;que hi ha hagut un error de CRC
TEST_OK
          RET                             ;(CRC=0)
```

```
;-----------------------------------------------------------------------------------------------------------------
LLEGIR_BYTE
;-----------------------------------------------------------------------------------------------------------------
;llegeix un byte de l'scratchpad i el deixa a X i a A
;és adir crida vuit cops la funció de llegir 1 bit (READ_01)

          MOV X,#00000000b       ;inicialitza X=00000000b
          MOV B,#8d              ;B es el comptador del numero de bits llegits
LOOP_LLEGIR

          CALL READ_01           ;llegeix 1 bit

          MOV A,H                ;si hi ha hagut un error llegint el bit, va a error_bit
          CMP A,#11111111b
          BZ error_bit

          MOV A,X                ;posa els 8 bits llegits per ordre a X
          ROR A,1
          ADD A,H
          MOV X,A

          DBNZ B,LOOP_LLEGIR

          RET

error_bit
          OR S:ERROR_TEMP,#error_lec_bit    ;actualitza la variable ERROR_TEMP per indicar
                                            ;que s'ha produit un error en la lectura del bit
          DBNZ B,LOOP_LLEGIR

          RET


;-----------------------------------------------------------------------------------------------------------------
READ_01
;-----------------------------------------------------------------------------------------------------------------
;llegeix un bit de l'scratchpad
;Es prenen tres mostres del bit a llegir
;H=00000000 si s'ha llegit un 0,   H=10000000 si s'ha llegit un 1  H=11111111 error

          MOV PCC,#00000000b     ;fcpu=fx
          CLR1 DQ                ;pull-down del bus
          CLR1 DQ_IO             ;DQ es de sortida  (durant 1.6us)

          NOP

          SET1 DQ_IO             ;DQ en alta impedancia (alliberament bus)

          MOV C,#4d              ;compta 7.3us   (durant 7.3us no es prenen mostres)
          DBNZ C,$

          MOV A,P2               ;copia a A el contingut de P2, només ens interessa P2.4
          MOV C,A                ;es prenen tres mostres en 4us
          MOV A,P2               ;es guarden a C,D i E
          MOV D,A
          MOV A,P2
          MOV E,A

          MOV A,C                ;copia C a CC
          MOV S:CC,A
          MOV A,D                ;copia D a DD
          MOV S:DD,A
          MOV A,E                ;copia E a EE
          MOV S:EE,A


          MOV A,#00000000b       ;posa a A 00000EDC
          BT S:CC.4,CC1
          BF S:CC.4,CC0
CC1       SET1 A.0
          BR fi_CC
CC0       CLR1 A.0
fi_CC

          BT S:DD.4,DD1
          BF S:DD.4,DD0
DD1       SET1 A.1
          BR fi_DD
DD0       CLR1 A.1
fi_DD

          BT S:EE.4,EE1
```

```
          BF S:EE.4,EE0
EE1       SET1 A.2
          BR fi_EE
EE0       CLR1 A.2
fi_EE

          MOV H,#11111111b      ;inicialitzem H a aquest valor de tal manera que si A no coincideix amb cap
                                ;de les sequencies donades detectarem que hi ha hagut error

test000   CMP A,#00000000b      ;si 000 H=00000000
          BNZ test001
          MOV H,#00000000b

test001   CMP A,#00000100b      ;si 001 H=00000000
          BNZ test111
          MOV H,#00000000b

test111   CMP A,#00000111b      ;si 111 H=10000000
          BNZ test011
          MOV H,#10000000b

test011   CMP A,#00000110b      ;si 011 H=10000000
          BNZ fi_test
          MOV H,#10000000b
fi_test
          RET



;-----------------------------------------------------------------------------------------------------------------
WRITE0
;-----------------------------------------------------------------------------------------------------------------
;escriu un "0" al sensor de temperatura DS18B20

          MOV PCC,#00000010b     ;fcpu=fx/4

          CLR1 DQ_IO             ;DQ es de sortida
          CLR1 DQ                ;pull-down del bus

          MOV B,#18d             ;compta 90us (no exactes)
          DBNZ B,$

          SET1 DQ_IO             ;DQ en alta impedancia (alliberament bus)

          MOV PCC,#00000000b     ;fcpu=fx
          RET


;-----------------------------------------------------------------------------------------------------------------
WRITE1
;-----------------------------------------------------------------------------------------------------------------
;escriu un "1" al sensor de temperatura DS18B20

          MOV PCC,#00000010b     ;fcpu=fx/4

          CLR1 DQ_IO             ;DQ es de sortida
          CLR1 DQ                ;pull-down del bus

          NOP                    ;compta 4.86us
          NOP
          NOP

          SET1 DQ_IO             ;DQ en alta impedancia (alliberament bus)

          MOV B,#16d             ;compta 85us
          DBNZ B,$

          MOV PCC,#00000000b     ;fcpu=fx
          RET
```

```
;*****************************************************************************************************************
;* RUTINES DE PROCESSAT DE LA TEMPERATURA                                                                        *
;*****************************************************************************************************************


;-----------------------------------------------------------------------------------------------------------------
GENERA_BYTES_TX_TEMP
;-----------------------------------------------------------------------------------------------------------------
;codifica VISU1_ACT i VISU2_ACT (provinents de SCRT_VISU) i troba
;BYTE1_TX_TEMP, BYTE2_TX_TEMP, BYTE3_TX_TEMP i BYTE4_TX_TEMP

          MOV A,VISU1_ACT
          ROL A,1
          AND A,#10101010b
          MOV S:BYTE1_TX_TEMP,A
          ROR A,1
          XOR A,#11111111b
          AND A,#01010101b
          ADD A,BYTE1_TX_TEMP
          MOV S:BYTE1_TX_TEMP,A

          MOV A,VISU1_ACT
          AND A,#10101010b
          MOV S:BYTE2_TX_TEMP,A
          ROR A,1
          XOR A,#11111111b
          AND A,#01010101b
          ADD A,BYTE2_TX_TEMP
          MOV S:BYTE2_TX_TEMP,A

          MOV A,VISU2_ACT
          ROL A,1
          AND A,#10101010b
          MOV S:BYTE3_TX_TEMP,A
          ROR A,1
          XOR A,#11111111b
          AND A,#01010101b
          ADD A,BYTE3_TX_TEMP
          MOV S:BYTE3_TX_TEMP,A

          MOV A,VISU2_ACT
          AND A,#10101010b
          MOV S:BYTE4_TX_TEMP,A
          ROR A,1
          XOR A,#11111111b
          AND A,#01010101b
          ADD A,BYTE4_TX_TEMP
          MOV S:BYTE4_TX_TEMP,A

          RET

;-----------------------------------------------------------------------------------------------------------------
GUARDA_TEMPER
;-----------------------------------------------------------------------------------------------------------------
;si s'ha rebut correctament una temperatura la guarda a TEMP_UNITATS, sinó guarda 01111111 11111111

          CMP S:RX_TEMP_OK,#11111111b
          BZ guarda_2bytes_temp
          BR no_guarda_2bytes_temp
guarda_2bytes_temp
          MOVW AX,S:COMP_GUARDA_TEMP    ;MOV [COMP_GUARDA_TEMP],BYTE_1
          MOVW HL,AX
          MOV A,BYTE_1
          MOV [HL],A

          MOVW AX,S:COMP_GUARDA_TEMP    ;INC COMP_GUARDA_TEMP
          INCW AX
          MOVW S:COMP_GUARDA_TEMP,AX

          MOVW AX,S:COMP_GUARDA_TEMP    ;MOV [COMP_GUARDA_TEMP],BYTE_2
          MOVW HL,AX
          MOV A,BYTE_2
          MOV [HL],A

          MOVW AX,S:COMP_GUARDA_TEMP    ;INC COMP_GUARDA_TEMP
          INCW AX
          MOVW S:COMP_GUARDA_TEMP,AX

          BR fi_guarda_temp

no_guarda_2bytes_temp
```

```
          MOVW AX,S:COMP_GUARDA_TEMP    ;MOV [COMP_GUARDA_TEMP],01111111
          MOVW HL,AX
          MOV A,#01111111b
          MOV [HL],A

          MOVW AX,S:COMP_GUARDA_TEMP    ;INC COMP_GUARDA_TEMP
          INCW AX
          MOVW S:COMP_GUARDA_TEMP,AX

          MOVW AX,S:COMP_GUARDA_TEMP    ;MOV [COMP_GUARDA_TEMP],11111111
          MOVW HL,AX
          MOV A,#11111111b
          MOV [HL],A

          MOVW AX,S:COMP_GUARDA_TEMP    ;INC COMP_GUARDA_TEMP
          INCW AX
          MOVW S:COMP_GUARDA_TEMP,AX

          BR fi_guarda_temp

fi_guarda_temp

          RET


;-------------------------------------------------------------------------------------------------------------------
INICIALITZA_MAX_MIN
;-------------------------------------------------------------------------------------------------------------------
;inicialitza la taula de les temperatures màximes i nínimes
;posa al lloc de les temperatures màximes la mínima temperatura possible(1000 0000 0000)
;posa al lloc de les temperatures mínimes la màxima temperatura possible(0111 1111 1111)

          MOVW DE,#TAU_TEMP_MAX_MIN    ;DE és localment el comptador de TAU_TEMP_MAX_MIN

          MOV B,#8d

loop_inic_max_min
          MOV A,#10000000b
          MOV [DE],A
          INCW DE
          MOV A,#00001111b
          MOV [DE],A
          INCW DE
          MOV A,#01111111b
          MOV [DE],A
          INCW DE

          DBNZ B,loop_inic_max_min

          RET



;-------------------------------------------------------------------------------------------------------------------
ACTUALITZA_MAX_MIN
;-------------------------------------------------------------------------------------------------------------------
;a partir de la taula de temperatures, actualitza la taula de temperatures màximes i mínimes

          MOVW DE,#TAU_TEMP_MAX_MIN    ;DE és localment el comptador de TAU_TEMP_MAX_MIN
          MOVW HL,#TAU_TEMP_UNITATS    ;HL és localment el comptador de TAU_TEMP_UNITATS

          MOV B,#8d


loop_actualit_max_min
                                       ;posa a MAX1_ACT, MAX2_ACT, MIN1_ACT i MIN2_ACT
          MOV A,[DE]                   ;les temperatures maximes i mínimes de la unitat n° B
          MOV S:MAX1_ACT,A

          INCW DE                            ;[posició 3B+1 de TAU_TEMP_MAX_MIN]

          MOV A,[DE]
          AND A,#11110000b
          MOV S:MAX2_ACT,A

          MOV A,[DE]
          AND A,#00001111b
          ROL A,1
          ROL A,1
          ROL A,1
          ROL A,1
          MOV S:MIN2_ACT,A
```

```
          INCW DE                          ;[posició 3B+2 de TAU_TEMP_MAX_MIN]

          MOV A,[DE]
          MOV S:MIN1_ACT,A

          DECW DE                          ;[posició 3B de TAU_TEMP_MAX_MIN]
          DECW DE

                                          ;guardem a TEMP1_ACT i TEMP2_ACT les temperatures actuals
          MOV A,[HL]
          MOV TEMP1_ACT,A

          INCW HL                          ;[posició 2B+1 de TAU_TEMP_UNITATS]

          MOV A,[HL]
          AND A,#11110000b
          MOV TEMP2_ACT,A

          DECW HL                          ;[posició 2B de TAU_TEMP_UNITATS]


          CMP S:TEMP1_ACT,#01111111b   ;si TEMP1_ACT i TEMP2_ACT contenen el codi de temperatura no rebuda
          BNZ complementa_bits_signe   ;(0111 1111 1111) no s'actualitzen les màximes i les mínimes
          MOV A,TEMP2_ACT              ;si no el conté va a complementa_bits_signe
          AND A,#11110000b
          CMP A,#11110000b
          BNZ complementa_bits_signe
          BR fi_actualitzacio_max_min


complementa_bits_signe
          XOR S:TEMP1_ACT,#10000000b   ;complementem el bit de signe de les emperatures
          XOR S:MAX1_ACT,#10000000b    ;així aconseguim que es pugui fer la comparació entre
          XOR S:MIN1_ACT,#10000000b    ;elles directament



verifica_maxima                        ;per a la unitat nº B compara la part entera de la temperatura actual
          MOV A,MAX1_ACT               ;amb la part entera de la màxima
          CMP A,TEMP1_ACT              ;(p.e. temperatura màxima - p.e. temp actual)
          BC actualitza_maxima         ;si p.e. temp actual > p.e. temp màxima, va a actualitza màxima
          BZ comprova_decimals_max     ;si p.e. temp actual = p.e. temp màxima, va a
                                       ;   comprova_decimals_max_pos
          BR verifica_minima           ;si p.e. temp actual < p.e. temp màxima, va a
                                       ;   verificar la temperatura mínima

comprova_decimals_max
                                       ;per a la unitat nº B compara la part decimal de la temperatura actual
          MOV A,MAX2_ACT               ;amb la part decimal de la màxima
          CMP A,TEMP2_ACT              ;(p.d. temperatura màxima - p.d. temp actual)
          BC actualitza_maxima         ;si p.d. temp actual >  p.d. temp màxima, va a actualitza màxima
          BR verifica_minima           ;si p.d. temp actual =< p.d. temp màxima, va a verifica_minima_pos



actualitza_maxima
          MOV A,[HL]                   ;posem "la p.e. de la temperatura actual"
          MOV [DE],A                   ;a "la posició de la p.e. de la màxima"

          INCW DE                           ;[posició 3B+1 de TAU_TEMP_MAX_MIN]
          INCW HL                           ;[posició 2B+1 de TAU_TEMP_UNITATS]

          MOV A,[HL]                   ;posem
          AND A,#11110000b             ;"el nibble de la part decimal de la temperatura de TAU_TEMP_UNITATS"
          MOV S:AUX,A                  ;a "el nibble de la part decimal de la màxima de TAU_TEMP_MAX_MIN"
          MOV A,[DE]
          AND A,#00001111b
          ADD A,AUX
          MOV [DE],A

          DECW DE                           ;[posició 3B de TAU_TEMP_MAX_MIN]
          DECW HL                           ;[posició 2B de TAU_TEMP_UNITATS]

          BR verifica_minima


fi_actualitzacio_max_min

         INCW DE
         INCW DE
         INCW DE
```

```
         INCW HL
         INCW HL

         DBNZ B,loop_actualit_max_min

         RET                          ;=======>

verifica_minima
                                      ;per a la unitat n° B compara la part entera de
          MOV A,TEMP1_ACT             ;la temperatura actual amb la part entera de la mínima
          CMP A,MIN1_ACT              ;(p.e. temp actual - p.e. temperatura mínima)
          BC actualitza_minima        ;si p.e. temp actual < p.e. temp mínima, va a actualitza mínima
          BZ comprova_decimals_min    ;si p.e. temp actual = p.e. temp mínima, va a
                                      ; comprova_decimals_min_pos
          BR fi_actualitzacio_max_min ;si p.e. temp actual > p.e. temp mínima, va a
                                      ;fi_actualització_max_min

comprova_decimals_min
                                      ;per a la unitat n° B compara la part decimal de
          MOV A,TEMP2_ACT             ;la temperatura actual amb la part decimal de la mínima
          AND A,#11110000b            ;(p.d. temp actual - p.d. temperatura mínima)
          CMP A,MIN2_ACT
          BC actualitza_minima        ;si p.d. temp actual <  p.d. temp mínima, va a actualitza mínia
          BR fi_actualitzacio_max_min ;si p.d. temp actual => p.d. temp mínima, va a
                                      ;fi_actualitzacio_max_min

actualitza_minima

         INCW DE                      ;posició 3B+2 de TAU_TEMP_MAX_MIN
         INCW DE

         MOV A,[HL]                   ;posem "la p.e. de la temperatura actual"
         MOV [DE],A                   ;a "la posició de la p.e. de la mínima"

         DECW DE                      ;posició 3B+1 de TAU_TEMP_MAX_MIN
         INCW HL                      ;posició 2B+1 de TAU_TEMP_UNITATS

         MOV A,[HL]                   ;posem
         ROR A,1                      ;"el nibble de la part decimal de la temp de TAU_TEMP_UNITATS"
         ROR A,1                      ;a
         ROR A,1                      ;"el nibble de la part decimal de la mínima de TAU_TEMP_MAX_MIN"
         ROR A,1
         AND A,#00001111b
         MOV S:AUX,A
         MOV A,[DE]
         AND A,#11110000b
         ADD A,AUX
         MOV [DE],A

         DECW DE                      ;posició 3B de TAU_TEMP_MAX_MIN
         DECW HL                      ;posició 2B de TAU_TEMP_UNITATS

         BR fi_actualitzacio_max_min


;-------------------------------------------------------------------------------------------------------------
TROBA_TEMPERS_ACTUALS
;-------------------------------------------------------------------------------------------------------------
;a partir UNIT_ACT_VISU troba TEMP1_ACT, TEMP2_ACT, MAX1_ACT, MAX2_ACT, MIN1_ACT i MIN2_ACT

          MOVW AX,S:COMP_TEMP         ;posa a HL el comptador de lectura de la taula de temperatures
          MOVW HL,AX

          MOV A,[HL]                  ;posa el primer byte de la temp de la unitat actual a TEMP1_ACT
          MOV S:TEMP1_ACT,A

          INCW HL

          MOV A,[HL]                  ;posa el segon byte de la temp de la unitat actual a TEMP2_ACT
          AND A,#11110000b
          MOV S:TEMP2_ACT,A

          MOVW AX,S:COMP_MAX_MIN      ;posa a HL el comptador de la taula de màximes i mínimes
          MOVW HL,AX

          MOV A,[HL]                  ;posa el primer byte de la màxima de la unitat actual a MAX1_ACT
          MOV S:MAX1_ACT,A

          INCW HL

          MOV A,[HL]                  ;posa el segon byte de màxima de la unitat actual a MAX2_ACT
```

```
          AND A,#11110000b
          MOV S:MAX2_ACT,A
          MOV A,[HL]                    ;posa el segon byte de mínima de la unitat actual a MIN2_ACT
          AND A,#00001111b
          ROL A,1
          ROL A,1
          ROL A,1
          ROL A,1
          MOV S:MIN2_ACT,A

          INCW HL

          MOV A,[HL]                    ;posa el primer byte de la mínima de la unitat actual a MIN1_ACT
          MOV S:MIN1_ACT,A

          RET



;-----------------------------------------------------------------------------------------------------------------
SCRT_VISU  ;(master i sensor)
;-----------------------------------------------------------------------------------------------------------------
;converteix el format de temperatura llegit de l'scratchpad al format d'emmagatzemament de temperatures
;a partir de MSB i LSB troba VISU1_ACT i VISU2_ACT
;format Scratchpad:     MSB:        S  S  S  S  S  B6 B5 B4   LSB:       B3 B2 B1 B0 D1 D2 D3 D4
;format Visu:           VISU1_ACT:  S  B6 B5 B4 B3 B2 B1 B0   VISU2_ACT: D1 D2 D3 D4 0  0  0  0

          MOV A,LSB
          ROL A,1
          ROL A,1
          ROL A,1
          ROL A,1
          MOV S:AUX,A
          AND A,#11110000b
          MOV S:VISU2_ACT,A

          MOV A,AUX
          AND A,#00001111b
          MOV S:AUX,A

          MOV A,MSB
          AND A,#00001111b
          ROL A,1
          ROL A,1
          ROL A,1
          ROL A,1

          ADD A,AUX
          MOV S:VISU1_ACT,A

          RET



;-----------------------------------------------------------------------------------------------------------------
CALC_TEMPER_LCD
;-----------------------------------------------------------------------------------------------------------------
;a partir de VISU1_ACT i VISU2_ACT troba SIGNE, DESEN_LCD, UNIT_LCD, PUNT i DECIMAL_LCD

          MOV A,VISU1_ACT               ;posa la variable BYTE_1_ACT (part entera de la temperatura) a BINARI
          MOV S:BINARI,A

          MOV A,VISU2_ACT               ;posa als primers 4 bits de DECIMALS els 4 bits
          AND A,#11110000b              ;dels decimals de la temperatura
          MOV S:DECIMALS,A


          CMP S:VISU1_ACT,#01111111b    ;si VISU1_ACT i VISU2_ACT contenen el codi de temperatura no
          BNZ comprova_altre_codi       ;disponible (0111 1111 1111 per a la temperatura o per a la mínima)
          MOV A,VISU2_ACT               ;va a visualit_temp_no_rebuda, si no el conté va a comprova_altre_codi
          AND A,#11110000b
          CMP A,#11110000b
          BNZ comprova_altre_codi
          BR visualit_temp_no_rebuda

comprova_altre_codi
          CMP S:VISU1_ACT,#10000000b    ;si VISU1_ACT i VISU2_ACT contenen el codi de temperatura no
          BNZ comprova_rang_temp        ;disponible (1000 0000 0000 per a la màxima)
          MOV A,VISU2_ACT               ;va visualit_temp_no_rebuda, si no el conté va a comprova_rang_temp
          AND A,#11110000b
          CMP A,#00000000b
```

```
          BNZ comprova_rang_temp
          BR visualit_temp_no_rebuda


comprova_rang_temp
          MOV A,VISU1_ACT             ;comprova si el núm. està dins el rang de visualització (-99.9,+99.9)
          XOR A,#10000000b            ;si no està dins el rang correcte va a overflow_temp
          CMP A,#11100100b
          BNC overflow_temp

          CMP A,#00011100b
          BC overflow_temp
          BZ overflow_temp



          MOV S:PUNT,#punt_           ;posa a la variable PUNT el byte correcte per a visualitzar "."


          BF S:BINARI.7,num_positiu   ;discrimina si la temperatura és positiva o negativa
          BT S:BINARI.7,num_negatiu

num_positiu
          MOV S:SIGNE,#espai_         ;prepara la variable signe per escriure " "
          BR valor_temp
num_negatiu                           ;obté el valor absolut de la temperatura
                                       ;(BINARI i DECIMALS(D3 D2 D1 D0 0000))
                                      ;resta 1 al conjunt format pels 8 bits de BINARI i els 4 de DECIMALS
          SUB S:DECIMALS,#00010000b   ;resta 1 al nibble format pels 4 bits més significatius de DECIMALS
          SUBC S:BINARI,#0d           ;resta a BINARI el carry de la resta anterior i 0d
          XOR S:DECIMALS,#11111111b   ;complementa DECIMALS
          AND S:DECIMALS,#11110000b
          XOR S:BINARI,#11111111b     ;complementa BINARI

          MOV S:SIGNE,#menys_         ;preparala variable signe per escriure "-"

          BR valor_temp
valor_temp

          CALL BINARI_BCD             ;passa la part entera de la temperatura a codi BCD,
                                      ;i el deixa a la variable BCD

          MOV A,BCD                   ;posa a DESEN_LCD el byte necessari per a mostrar
          ROR A,1                     ;les desenes de la temperatura
          ROR A,1
          ROR A,1
          ROR A,1
          AND A,#00001111b
          ADD A,#00110000b
          MOV S:DESEN_LCD,A

          CMP S:DESEN_LCD,#N0_        ;si les desenes contenen un "0" es mostra un espai en
          BNZ troba_unit_LCD          ;el lloc de les desenes
          MOV S:DESEN_LCD,#espai_

troba_unit_LCD
          MOV A,BCD                   ;posa a UNIT_LCD el byte necessari per a mostrar les
          AND A,#00001111b            ;unitats de la temperatura
          ADD A,#00110000b
          MOV S:UNIT_LCD,A


          MOV A,DECIMALS              ;posa a B 0000 D3 D2 D1 D0
          ROR A,1
          ROR A,1
          ROR A,1
          ROR A,1
          MOV B,A
                                      ;posa a DECIMAL_LCD byte necessari per a mostrar el decimal
                                      ;de la temperatura
          MOVW HL,#taula_dec_LCD      ;posa a HL l'adreça del primer element de taula_dec_LCD
          MOV A,L                     ;i li suma el valor de B (D3 D2 D1 D0)
          ADD A,B
          MOV L,A
          MOV A,H
          ADDC A,#0d
          MOV H,A

          MOV A,[HL]                  ;accedeix a l'element correcte de la taula_dec_LCD
          MOV S:DECIMAL_LCD,A         ;i guarda el valor que hi ha a DECIMAL_LCD

          RET                         ;--->
```

```
overflow_temp                          ;si la temperatura a visualitzar està fora de rang escriu " OV.F"

          MOV S:SIGNE,#espai_
          MOV S:DESEN_LCD,#O_
          MOV S:UNIT_LCD,#V_
          MOV S:PUNT,#punt_
          MOV S:DECIMAL_LCD,#F_

          RET                          ;--->

visualit_temp_no_rebuda                ;si no s'ha rebut la temperatura escriu " --.-"

          MOV S:SIGNE,#espai_
          MOV S:DESEN_LCD,#guio_
          MOV S:UNIT_LCD,#guio_
          MOV S:PUNT,#punt_
          MOV S:DECIMAL_LCD,#guio_


          RET                          ;--->



;-------------------------------------------------------------------------------------------------------------------
CALC_UNITAT_LCD
;-------------------------------------------------------------------------------------------------------------------
;a partir d'un número binari comprés entre 00 i 99 (VISU1_ACT) troba DESEN_LCD i UNIT_LCD

          MOV A,VISU1_ACT              ;posa la variable VISU1_ACT a BINARI
          MOV S:BINARI,A

comprova_rang_unitat_LCD
          MOV A,BINARI                 ;comprova si el número està dins el rang de visualització (0,+99)
          CMP A,#100d                  ;si no està dins el rang correcte va a overflow_temp
          BNC overflow_unit_LCD


          CALL BINARI_BCD              ;converteix BINARI a codi BCD, i el deixa a la variable BCD

          MOV A,BCD                    ;posa a DESEN_LCD el byte necessari per a mostrar les desenes BINARI
          ROR A,1
          ROR A,1
          ROR A,1
          ROR A,1
          AND A,#00001111b
          ADD A,#00110000b
          MOV S:DESEN_LCD,A

          MOV A,BCD                    ;posa a UNIT_LCD el byte necessari per a mostrar les unitats de BINARI
          AND A,#00001111b
          ADD A,#00110000b
          MOV S:UNIT_LCD,A

          RET                          ;--->

overflow_unit_LCD                      ;si el número a visualitzar està fora de rang escriu "OF"
          MOV S:DESEN_LCD,#O_
          MOV S:UNIT_LCD,#F_

          RET                          ;--->



;-------------------------------------------------------------------------------------------------------------------
BINARI_BCD
;-------------------------------------------------------------------------------------------------------------------
;converteix el número contingut a la variable BINARI de binari a codi BCD i el deixa a la variable BCD

          MOV B,#7d                   ;inicialitzem e comptador d'iteracions
          MOV S:BCD,#00000000b        ;inicialitzem la variable BCD

loop_binari_BCD
          MOV A,BINARI                ;shift a l'esquerra de BINARI
          ROLC A,1
          MOV S:BINARI,A

          MOV A,BCD                   ;shift a l'esquerra de BCD amb el carry de BINARI
          ROLC A,1
```

```
          MOV S:BCD,A


          MOV A,BCD                 ;si els quatre ultims bits de BCD són més grans que 5,
          AND A,#00001111b          ;suma 3 als últims quatre bits de BCD
          CMP A,#05h
          BC skip_suma3_unit
          ADD A,#03h
          MOV S:AUX,A
          MOV A,BCD
          AND A,#11110000b
          ADD A,AUX
          MOV S:BCD,A

skip_suma3_unit

          MOV A,BCD                 ;si els quatre primers bits de BCD són més grans que 5,
          AND A,#11110000b          ;suma 3 als primers quatre bits de BCD
          CMP S:BCD,#50h
          BC skip_suma3_desen
          ADD A,#30h
          MOV S:AUX,A
          MOV A,BCD
          AND A,#00001111b
          ADD A,AUX
          MOV S:BCD,A
skip_suma3_desen

          DBNZ B,loop_binari_BCD    ;actualitza el comptador d'iteracions

                                    ;shift de l'última iteració
          MOV A,BINARI              ;shift a l'esquerra de BINARI
          ROLC A,1
          MOV S:BINARI,A

          MOV A,BCD                 ;shift a l'esquerra de BCD amb el carry de BINARI
          ROLC A,1
          MOV S:BCD,A

          RET




;*************************************************************************************************************
;* RUTINES D'INTERFICIE D'USUARI                                                                             *
;*************************************************************************************************************


;-------------------------------------------------------------------------------------------------------------
INTERFICIE_USUARI
;-------------------------------------------------------------------------------------------------------------
;gestiona la interficie d'usuari del màster

inici_interficie                    ;selecció de la pantalla
compara00
          CMP S:ESTAT_INTERFICIE,#00d
          BNZ compara01
          BR pant_recepcio_unitats
compara01
          CMP S:ESTAT_INTERFICIE,#01d
          BNZ compara02
          BR pant_seleccio_interval
compara02
          CMP S:ESTAT_INTERFICIE,#02d
          BNZ compara00
          BR pant_esborrat_maxs_mins

;---------------------------------

pant_recepcio_unitats               ;pantalla 00

          MOV S:DADA_LCD,#00000001b ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          CALL ESCRIURE_TEMP_LCD    ;escriu per la pantalla la unitat actual,
                                    ;la seva temperatura, la màxima i la mínima

sens_inc_00                         ;sensat de +, -, MODE i SET
```

```
          BT MES,sens_dec_00              ;si es polsa + s'incrmeneta el número mostrat,
          BR inc_unit_visu                ;quan s'arriba a 07 es torna a 01
sens_dec_00                               ;si es polsa - es decrementa el número mostrat,
          BT MENYS,sens_mode_00           ;quan s'arriba a 01 es torna a 07
          BR dec_unit_visu                ;si es polsa MODE no es modifica el número mostrat
sens_mode_00                              ;i va a la pantalla següent
          BT MODE,sens_set_00             ;si el polsa SET durant un cert temps s'esborra
          BR passa_pantalla               ;la màxima i la mínima de la unitat actual
sens_set_00
          BT SET,sens_inc_00
          BR esborra_max_min

inc_unit_visu                             ;s'incrementa adientment la unitat actual,
          CMP S:UNIT_ACT_VISU,#07d        ;el comptador de lectura de la taula de temperatures
          BZ posa_a_0                     ;i el comptador de la taula de màximes i mínimes
          INC S:UNIT_ACT_VISU             ;quan s'acaba va a fi_sens_inc_dec_00
          MOVW AX,S:COMP_TEMP
          ADDW AX,#2d
          MOVW S:COMP_TEMP,AX
          MOVW AX,S:COMP_MAX_MIN
          ADDW AX,#3d
          MOVW S:COMP_MAX_MIN,AX
          BR fi_sens_inc_dec_00
posa_a_0
          MOV S:UNIT_ACT_VISU,#00d
          MOVW AX,#TAU_TEMP_UNITATS
          MOVW S:COMP_TEMP,AX
          MOVW AX,#TAU_TEMP_MAX_MIN
          MOVW S:COMP_MAX_MIN,AX
          BR fi_sens_inc_dec_00


dec_unit_visu                             ;es decrementa adientment la unitat actual,
          CMP S:UNIT_ACT_VISU,#00d        ;el comptador de lectura de la taula de temperatures
          BZ posa_a_7                     ;i el comptador de la taula de màximes i mínimes
          DEC S:UNIT_ACT_VISU             ;quan s'acaba va a fi_sens_inc_dec_00
          MOVW AX,S:COMP_TEMP
          SUBW AX,#2d
          MOVW S:COMP_TEMP,AX
          MOVW AX,S:COMP_MAX_MIN
          SUBW AX,#3d
          MOVW S:COMP_MAX_MIN,AX
          BR fi_sens_inc_dec_00
posa_a_7
          MOV S:UNIT_ACT_VISU,#07d
          MOVW AX,#TAU_TEMP_UNITATS
          ADDW AX,#14d
          MOVW S:COMP_TEMP,AX
          MOVW AX,#TAU_TEMP_MAX_MIN
          ADDW AX,#21d
          MOVW S:COMP_MAX_MIN,AX
          BR fi_sens_inc_dec_00
fi_sens_inc_dec_00

          CALL ESCRIURE_TEMP_LCD          ;escriu per la pantalla la nova unitat actual,
                                          ;la seva temperatura, la màxima i la mínima
          CALL ESPERA_POLSADOR

          BR sens_inc_00                  ;torna al sensat dels polsador de la pantalla 00

;-------------

pantalles_debug_
          BR pantalles_debug              ;si s'ha polsat MENYS, SET va a pantalles_debug
;-------------

esborra_max_min
          CALL ESPERA_POLSADOR            ;si després de 2 s, MODE està pitjat va a pantalles_debug
          CALL ESPERA_POLSADOR            ;si després de 2 s, SET encara està pitjat, va a
          CALL ESPERA_POLSADOR            ;esborra_registre
          CALL ESPERA_POLSADOR            ;si cap dels dos estan pitjats va a sens_inc_00

          BF MODE,pantalles_debug_
          BF SET,esborra_registre
          BR sens_inc_00
esborra_registre                          ;esborra la temperatura màxima i mínima de la unitat actual
          MOVW AX,S:COMP_MAX_MIN          ;posa a la temperatura màxima de la unitat actual
          MOVW DE,AX                      ;el codi de temperatura no disponible (1000 0000 0000
                                          ;posa a la temperatura mínima de la unitat actual
          MOV A,#10000000b                ;el codi de temperatura no disponible (0111 1111 1111)
          MOV [DE],A
          INCW DE
```

```
          MOV A,#00001111b
          MOV [DE],A
          INCW DE
          MOV A,#01111111b
          MOV [DE],A

          CALL ESCRIURE_TEMP_LCD          ;escriu les noves dades de temperatura

          CALL ESPERA_POLSADOR

          BR sens_inc_00
;--------------------------------

pant_seleccio_interval                    ;pantalla 01
          CALL MENU_SELEC_INTERVAL

sens_inc_01                               ;sensat de + - i MODE i SET
          BT MES,sens_dec_01              ;si es polsa + s'incrmeneta l'interval mostrat
          BR inc_interval                 ;si es polsa - es decrementa l'interval mostrat
sens_dec_01                               ;si es polsa MODE va a la pantalla següent
          BT MENYS,sens_mode_01           ;si es polsca SET queda fixat l'interval que actualment
          BR dec_interval                 ;s'està mostrant
sens_mode_01
          BT MODE,sens_set_01
          BR passa_pantalla
sens_set_01
          BT SET,sens_inc_01
          BR fixa_nou_interval


inc_interval
          CMP S:CODI_INTERVAL,#07d
          BZ torna_inici_interval
          INC S:CODI_INTERVAL
          BR fi_sens_inc_dec_01
torna_inici_interval
          MOV S:CODI_INTERVAL,#00d
          BR fi_sens_inc_dec_01

dec_interval
          CMP S:CODI_INTERVAL,#00d
          BZ torna_fi_interval
          DEC S:CODI_INTERVAL
          BR fi_sens_inc_dec_01
torna_fi_interval
          MOV S:CODI_INTERVAL,#07d
          BR fi_sens_inc_dec_01

fi_sens_inc_dec_01

          MOV S:DADA_LCD,#11000000b       ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          CALL ESCRIU_INTERVAL            ;escriu el nou interva

          CALL ESPERA_POLSADOR
          BR sens_inc_01                  ;torna al sensat dels polsadors de la pantalla 01

;-------------

fixa_nou_interval                         ;configura el sistema per a operar amb el nou interval

          CMP S:CODI_INTERVAL,#00d
          BZ fixa_interval_00
          CMP S:CODI_INTERVAL,#01d
          BZ fixa_interval_01
          CMP S:CODI_INTERVAL,#02d
          BZ fixa_interval_02
          CMP S:CODI_INTERVAL,#03d
          BZ fixa_interval_03
          CMP S:CODI_INTERVAL,#04d
          BZ fixa_interval_04
          CMP S:CODI_INTERVAL,#05d
          BZ fixa_interval_05
          CMP S:CODI_INTERVAL,#06d
          BZ fixa_interval_06
          CMP S:CODI_INTERVAL,#07d
          BZ fixa_interval_07

fixa_interval_00
          MOV S:N_SLOTS_INTER_TEMP,#1d
```

```
          MOV S:COMP_SLOTS_INTER_TEMP,#1d
          BR fi_fixa_interval
fixa_interval_01
          MOV S:N_SLOTS_INTER_TEMP,#2d
          MOV S:COMP_SLOTS_INTER_TEMP,#2d
          BR fi_fixa_interval
fixa_interval_02
          MOV S:N_SLOTS_INTER_TEMP,#4d
          MOV S:COMP_SLOTS_INTER_TEMP,#4d
          BR fi_fixa_interval
fixa_interval_03
          MOV S:N_SLOTS_INTER_TEMP,#10d
          MOV S:COMP_SLOTS_INTER_TEMP,#10d
          BR fi_fixa_interval
fixa_interval_04
          MOV S:N_SLOTS_INTER_TEMP,#20d
          MOV S:COMP_SLOTS_INTER_TEMP,#20d
          BR fi_fixa_interval
fixa_interval_05
          MOV S:N_SLOTS_INTER_TEMP,#30d
          MOV S:COMP_SLOTS_INTER_TEMP,#30d
          BR fi_fixa_interval
fixa_interval_06
          MOV S:N_SLOTS_INTER_TEMP,#60d
          MOV S:COMP_SLOTS_INTER_TEMP,#60d
          BR fi_fixa_interval
fixa_interval_07
          MOV S:N_SLOTS_INTER_TEMP,#120d
          MOV S:COMP_SLOTS_INTER_TEMP,#120d
          BR fi_fixa_interval

fi_fixa_interval
          CALL MISSATGE_NOU_INTERVAL        ;escriu per pantalla el missatge de fixació de nou interval

          CALL ESPERA_POLSADOR              ;espera 2.5 s
          CALL ESPERA_POLSADOR
          CALL ESPERA_POLSADOR
          CALL ESPERA_POLSADOR
          CALL ESPERA_POLSADOR

          CALL MENU_SELEC_INTERVAL          ;recupera per pantalla el menú de selecció d'interval

          BR sens_inc_01                    ;torna al sensat dels polsador de la pantalla 01
;----------------------------

pant_esborrat_maxs_mins                     ;pantalla 02

          CALL MENU_ESBORRAT_MAX_MIN        ;escriu per pantalla el menú d'esborrar de màximes i mínimes

sens_mode_02                                ;si es polsa MODE passa a la pantalla següent
          BT MODE,sens_set_02               ;si es polsa SET s'esborren les màximes i les mínimes
          BR passa_pantalla
sens_set_02
          BT SET,sens_mode_02
          BR esborrat_max_i_min


esborrat_max_i_min                          ;esborra les màximes i les mínimes
          CALL INICIALITZA_MAX_MIN

          CALL MISSATGE_ESBORRAT            ;escriu per pantalla el missatge d'esborrat

          CALL ESPERA_POLSADOR              ;espera 2.5 s
          CALL ESPERA_POLSADOR
          CALL ESPERA_POLSADOR
          CALL ESPERA_POLSADOR
          CALL ESPERA_POLSADOR

          BR passa_pantalla                 ;passa a la pantalla següent


;---------------------------------

pantalles_debug                             ;pantalles de debug

pant_debug_RX_tots

          CALL MOSTRA_INFs                  ;mostra per pantalla les variables INF del màster

          CALL ESPERA_POLSADOR              ;s'entra a aquesta pantalla amb MODE pitjat, per tant espera
          CALL ESPERA_POLSADOR              ;1.5 s per tal de que MODE no faci efecte de seguida
          CALL ESPERA_POLSADOR
```

```
          BT MODE,$                         ;espera a que es polsi MODE
          CALL ESPERA_POLSADOR

pant_debug_bytes_unit_act                   ;mostra per pantalla els bytes rebuts de la unitat triada

          CALL MOSTRA_BYTE_UNIT_ACT
          BT MODE,$                         ;espera a que es polsi MODE
          CALL ESPERA_POLSADOR

pant_debug_tots_bytes                       ;mostra per pantalla els bytes de temperatura (sense adreça)
                                            ;rebuts de totes les unitats inclosa la 00
          CALL MOSTRA_TOTS_BYTES
          BT MODE,$                         ;espera a que es polsi MODE
          CALL ESPERA_POLSADOR

pant_debug_bits1                            ;mostra per pantalla els bits rebuts en el 1r intent de mesura
                                            ;de la temperatura de la unitat triada
          CALL MOSTRA_BITS1
          BT MODE,$                         ;espera a que es polsi MODE
          CALL ESPERA_POLSADOR

pant_debug_bits2                            ;mostra per pantalla els bits rebuts en el 2n intent de mesura
                                            ;de la temperatura de la unitat triada
          CALL MOSTRA_BITS2
          BT MODE,$                         ;espera a que es polsi MODE
          CALL ESPERA_POLSADOR

          BR pant_recepcio_unitats          ;torna al lloc de selecció de pantalla,
                                            ;és a dir visualitza la mateixa
                                            ;pantalla que es visualitzava abans de entrar a pantalles_debug

;---------------------------------

passa_pantalla                              ;passa a la pantalla següent
          CMP S:ESTAT_INTERFICIE,#02d
          BZ torna_a_00d
          INC S:ESTAT_INTERFICIE
          CALL ESPERA_POLSADOR
          BR inici_interficie
torna_a_00d
          MOV S:ESTAT_INTERFICIE,#00d
          CALL ESPERA_POLSADOR
          BR inici_interficie


;-------------------------------------------------------------------------------------------------------------------
MENU_ESBORRAT_MAX_MIN
;-------------------------------------------------------------------------------------------------------------------
;Escriu el menú d'esborrat de màximes i mínimes

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#E_          ;E
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA
          MOV S:DADA_LCD,#b_          ;b
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_          ;o
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_          ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_          ;t
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_      ;" "
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_          ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_          ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#x_          ;x
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_          ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_          ;m
```

```
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#i_          ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_          ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_          ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_          ;n
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_          ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_          ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#claud1_     ;[
          CALL WRITE_DATA
          MOV S:DADA_LCD,#S_          ;S
          CALL WRITE_DATA
          MOV S:DADA_LCD,#E_          ;E
          CALL WRITE_DATA
          MOV S:DADA_LCD,#T_          ;T
          CALL WRITE_DATA
          MOV S:DADA_LCD,#claud2_     ;]
          CALL WRITE_DATA

          RET


;-------------------------------------------------------------------------------------------------------------------
MISSATGE_ESBORRAT
;-------------------------------------------------------------------------------------------------------------------
;escriu el missatge d'esborrat de màximes i mínimes

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#R_          ;R
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#g_          ;g
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_          ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_          ;t
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA
```

```
          MOV S:DADA_LCD,#b_          ;b
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_          ;o
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_          ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_          ;t
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA

          RET


;-----------------------------------------------------------------------------------------------------------------------
MENU_SELEC_INTERVAL
;-----------------------------------------------------------------------------------------------------------------------
;escriu el menú de selecció d'interval amb l'interval actual

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#I_          ;I
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_          ;n
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_          ;t
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#v_          ;v
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_          ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#l_          ;l
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_      ;" "
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_          ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA
          MOV S:DADA_LCD,#u_          ;u
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_          ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#dospunts_   ;":"
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11001011b   ;passa a la posició 11 de la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#claud1_     ;[
          CALL WRITE_DATA
          MOV S:DADA_LCD,#S_          ;S
          CALL WRITE_DATA
          MOV S:DADA_LCD,#E_          ;E
          CALL WRITE_DATA
          MOV S:DADA_LCD,#T_          ;T
          CALL WRITE_DATA
          MOV S:DADA_LCD,#claud2_     ;]
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          CALL ESCRIU_INTERVAL

          RET
```

```
;-------------------------------------------------------------------------------------------------------------------
ESCRIU_INTERVAL
;-------------------------------------------------------------------------------------------------------------------
;escriu l'interval actual
;en funció del valor de la variable CODI_INTERVAL escriu el valor correcte de l'interval

          CMP S:CODI_INTERVAL,#00d      ;en funció del valor de la variable CODI_INTERVAL
          BZ codi_interval_00_          ;escriu el valor correcte de l'interval
          CMP S:CODI_INTERVAL,#01d
          BZ codi_interval_01_
          CMP S:CODI_INTERVAL,#02d
          BZ codi_interval_02_
          CMP S:CODI_INTERVAL,#03d
          BZ codi_interval_03_
          CMP S:CODI_INTERVAL,#04d
          BZ codi_interval_04_
          CMP S:CODI_INTERVAL,#05d
          BZ codi_interval_05_
          CMP S:CODI_INTERVAL,#06d
          BZ codi_interval_06_
          CMP S:CODI_INTERVAL,#07d
          BZ codi_interval_07_

codi_interval_00_
          BR codi_interval_00
codi_interval_01_
          BR codi_interval_01
codi_interval_02_
          BR codi_interval_02
codi_interval_03_
          BR codi_interval_03
codi_interval_04_
          BR codi_interval_04
codi_interval_05_
          BR codi_interval_05
codi_interval_06_
          BR codi_interval_06
codi_interval_07_
          BR codi_interval_07

codi_interval_00
          MOV S:DADA_LCD,#N3_           ;3
          CALL WRITE_DATA
          MOV S:DADA_LCD,#N0_           ;0
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_            ;s
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA

          BR fi_codi_interval

codi_interval_01
          MOV S:DADA_LCD,#N1_           ;1
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_            ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_            ;n
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA

          BR fi_codi_interval

codi_interval_02
          MOV S:DADA_LCD,#N2_           ;2
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_            ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_            ;n
```

```
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA

          BR fi_codi_interval

codi_interval_03
          MOV S:DADA_LCD,#N5_           ;5
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_            ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_            ;n
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA

          BR fi_codi_interval

codi_interval_04
          MOV S:DADA_LCD,#N1_           ;1
          CALL WRITE_DATA
          MOV S:DADA_LCD,#N0_           ;0
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_            ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_            ;n
          CALL WRITE_DATA

          BR fi_codi_interval

codi_interval_05
          MOV S:DADA_LCD,#N1_           ;1
          CALL WRITE_DATA
          MOV S:DADA_LCD,#N5_           ;5
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_            ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_            ;n
          CALL WRITE_DATA

          BR fi_codi_interval

codi_interval_06
          MOV S:DADA_LCD,#N3_           ;3
          CALL WRITE_DATA
          MOV S:DADA_LCD,#N0_           ;0
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_            ;m
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_            ;n
          CALL WRITE_DATA

          BR fi_codi_interval

codi_interval_07
          MOV S:DADA_LCD,#N1_           ;1
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#h_            ;h
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
```

```
          CALL WRITE_DATA

          BR fi_codi_interval

fi_codi_interval

          RET


;-----------------------------------------------------------------------------------------------------------------
MISSATGE_NOU_INTERVAL
;-----------------------------------------------------------------------------------------------------------------
;escriu el missatge de nou interval fixat

          MOV S:DADA_LCD,#00000001b     ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#N_            ;N
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_            ;o
          CALL WRITE_DATA
          MOV S:DADA_LCD,#u_            ;u
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_            ;n
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_            ;t
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_            ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_            ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#v_            ;v
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_            ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#l_            ;l
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b     ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#f_            ;f
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_            ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#x_            ;x
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_            ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_            ;t
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;epai
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_            ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_        ;espai
          CALL WRITE_DATA

          CALL ESCRIU_INTERVAL

          RET


;-----------------------------------------------------------------------------------------------------------------
MISSATGE_MESURA     ;(master)
;-----------------------------------------------------------------------------------------------------------------
;escriu el missatge que es mostra durant el procés de mesura i recepció de les temperatures

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#M_          ;M
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
```

```
          MOV S:DADA_LCD,#s_          ;s
          CALL WRITE_DATA
          MOV S:DADA_LCD,#u_          ;u
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_          ;a
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_          ;n
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_          ;t
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#i_          ;i
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_      ;" "
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_          ;r
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#b_          ;b
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_          ;e
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_          ;n
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_          ;t
          CALL WRITE_DATA
          MOV S:DADA_LCD,#punt_       ;.
          CALL WRITE_DATA
          MOV S:DADA_LCD,#punt_       ;.
          CALL WRITE_DATA
          MOV S:DADA_LCD,#punt_       ;.
          CALL WRITE_DATA

          RET


;-------------------------------------------------------------------------------------------------------------
ESCRIU_UNIT_ACT     ;(master)
;-------------------------------------------------------------------------------------------------------------
;---A partir de la variable UNIT_ACT_VISU escriu dos digits amb el número de la unitat seguit d'un espai

          MOV A,UNIT_ACT_VISU         ;Calcula els codis LCD per visualitzar la unitat actual
          MOV S:VISU1_ACT,A
          CALL CALC_UNITAT_LCD

          MOV A,DESEN_LCD             ;desenes
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV A,UNIT_LCD              ;unitats
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA

          RET



;-------------------------------------------------------------------------------------------------------------
ESCRIURE_TEMP_LCD
;-------------------------------------------------------------------------------------------------------------
;escriu per la pantalla la unitat actual, la seva temperatura, la màxima i la mínima

          MOV S:DADA_LCD,#10000000b   ;va a l'origen de la línia 1
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          CALL TROBA_TEMPERS_ACTUALS  ;a partir UNIT_ACT_VISU troba
                                      ;TEMP1_ACT, TEMP2_ACT, MAX1_ACT, MAX2_ACT, MIN1_ACT i MIN2_ACT

          MOV S:DADA_LCD,#U_          ;U
          CALL WRITE_DATA
```

```
          MOV S:DADA_LCD,#dospunts_    ;:
          CALL WRITE_DATA

          CALL ESCRIU_UNIT_ACT

          MOV A,TEMP1_ACT              ;Calcula els codis LCD per visualitzar la temperatura i la visualitza
          MOV VISU1_ACT,A
          MOV A,TEMP2_ACT
          MOV VISU2_ACT,A
          CALL CALC_TEMPER_LCD
          CALL PRINT_TEMP

          MOV S:DADA_LCD,#grau_       ;"°"
          CALL WRITE_DATA

          MOV S:DADA_LCD,#C_          ;"C"
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#M_          ;"M"
          CALL WRITE_DATA

          MOV S:DADA_LCD,#dospunts_    ;:
          CALL WRITE_DATA

          MOV A,MAX1_ACT              ;Calcula els codis LCD per visualitzar la temp màxima i la visualitza
          MOV VISU1_ACT,A
          MOV A,MAX2_ACT
          MOV VISU2_ACT,A
          CALL CALC_TEMPER_LCD
          CALL PRINT_TEMP

          MOV S:DADA_LCD,#espai_       ;" "
          CALL WRITE_DATA

          MOV S:DADA_LCD,#m_           ;"m"
          CALL WRITE_DATA

          MOV S:DADA_LCD,#dospunts_    ;:
          CALL WRITE_DATA

          MOV A,MIN1_ACT              ;Calcula els codis LCD per visualitzar la temp mínima i la visualitza
          MOV VISU1_ACT,A
          MOV A,MIN2_ACT
          MOV VISU2_ACT,A
          CALL CALC_TEMPER_LCD
          CALL PRINT_TEMP

          RET


;-------------------------------------------------------------------------------------------------------------
PRINT_TEMP     ;(master i sensor)
;-------------------------------------------------------------------------------------------------------------
;escriu per pantalla una temperatura

          MOV A,SIGNE
          MOV S:DADA_LCD,A             ;" " o "-"
          CALL WRITE_DATA

          MOV A,DESEN_LCD
          MOV S:DADA_LCD,A            ;desenes
          CALL WRITE_DATA

          MOV A,UNIT_LCD
          MOV S:DADA_LCD,A            ;unitats
          CALL WRITE_DATA

          MOV A,PUNT
          MOV S:DADA_LCD,A            ;"."
          CALL WRITE_DATA

          MOV A,DECIMAL_LCD
          MOV S:DADA_LCD,A            ;decimal
          CALL WRITE_DATA

          RET
```

```
;-------------------------------------------------------------------------------------------------------------------
MISSATGE_INI_SENS
;-------------------------------------------------------------------------------------------------------------------
;escriu per pantalla el missatge inicial del sensor

          BF LCD_CONNECT,skip_miss_ini_sens_ ;si LCD_CONNECT està a "0" vol dir que la placa de configuració
          BR escriu_miss_ini_sens            ;està connectada i, al no haver-hi LCD, no s'executa la funció
skip_miss_ini_sens_                          ;si LCD_CONNECT està a "1" va a escriu_miss_ini_sens, és a dir
          BR skip_miss_ini_sens              ;executa la funció
escriu_miss_ini_sens

          CALL INICIALIT_LCD

          MOV S:DADA_LCD,#00000001b          ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF


          MOV S:DADA_LCD,#U_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b          ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#n_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#u_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA

          MOV A,N_SENSOR                     ;escriu el número del sensor
          MOV UNIT_ACT_VISU,A
          CALL ESCRIU_UNIT_ACT


skip_miss_ini_sens
          RET
```

```
;-------------------------------------------------------------------------------------------------------------------
MISSATGE_INI_MAST
;-------------------------------------------------------------------------------------------------------------------
;escriu per pantalla el missatge d'inicialització del màster

          CALL INICIALIT_LCD

          MOV S:DADA_LCD,#00000001b         ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#P_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#l_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#S_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#E_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#T_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#p_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_
          CALL WRITE_DATA

          MOV S:DADA_LCD,#11000000b         ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF


          MOV S:DADA_LCD,#i_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#c_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#i_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_
          CALL WRITE_DATA

          RET


;*******************************************************************************************************************
;* RUTINES DE CONTROL DE l'LCD                                                                                     *
;*******************************************************************************************************************


;-------------------------------------------------------------------------------------------------------------------
INICIALIT_LCD
;-------------------------------------------------------------------------------------------------------------------
;realitza la seqüència de pasos per a inicialitzar la pantalla LCD

          CALL CONFIG_8bits
          CALL ESPERA_5ms
          CALL CONFIG_8bits
          CALL ESPERA_5ms
          CALL CONFIG_8bits
          CALL ESPERA_5ms

          CALL CONFIG_4bits
          ;CALL ESPERA_BF
```

```
          MOV S:DADA_LCD,#00101000b    ;configura 4 bits, 2 línies i 5x8pixels
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#00001000b    ;display OFF
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#00000001b    ;clear display
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV S:DADA_LCD,#00000110b    ;cursor cap a la dreta, i no shift del LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_5ms

          MOV S:DADA_LCD,#00001100b    ;display ON, cursor OFF, blinking OFF
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          RET


;-----------------------------------------------------------------------------------------------------------
CONFIG_4bits
;-----------------------------------------------------------------------------------------------------------
;emet la comanda "function set" per tal de configurar l'LCD per funcionar amb interficie de 4 bits
          MOV PCC,#00000010b        ;fcpu=fx/4

          NOP
          CLR1 RS                   ;configura mode instuccions
          NOP
          CLR1 RW                   ;configura mode escriptura
          NOP

          SET1 En                   ;enable a 1

          CLR1 DB7                  ;configura el LCD per a funcionar amb 4 bits
          CLR1 DB6
          SET1 DB5
          CLR1 DB4
          NOP

          CLR1 En                   ;flanc de baixada de enable
          NOP

          MOV PCC,#0000000b         ;fcpu=fx

          RET


;-----------------------------------------------------------------------------------------------------------
CONFIG_8bits
;-----------------------------------------------------------------------------------------------------------
;emet la comanda "function set" per tal de configurar l'LCD per funcionar amb interficie de 8 bits

          MOV PCC,#00000010b        ;fcpu=fx/4

          NOP
          CLR1 RS                   ;configura mode instuccions
          NOP
          CLR1 RW                   ;configura mode escriptura
          NOP

          SET1 En                   ;enable a 1

          CLR1 DB7                  ;configura el LCD per a funcionar amb 8 bits
          CLR1 DB6
          SET1 DB5
          SET1 DB4
          NOP

          CLR1 En                   ;flanc de baixada de enable
          NOP

          MOV PCC,#00000000b        ;fcpu=fx

          RET
```

```
;-------------------------------------------------------------------------------------------------------------
WRITE_INSTRUC
;-------------------------------------------------------------------------------------------------------------
;envia a la pantalla LCD la instrucció continguda a la variable DADA_LCD

          MOV PCC,#00000010b        ;fcpu=fx/4

          NOP
          CLR1 RS                   ;configura mode instuccions
          NOP
          CLR1 RW                   ;configura mode escriptura
          NOP

          SET1 En                   ;enable a 1

          MOV A,DADA_LCD            ;posa a DB7, DB6, DB5 i DB4 els 4 primers bits de la instrucció
          AND A,#11110000b
          ROR A,1
          ROR A,1
          ROR A,1
          ROR A,1
          MOV S:AUX,A

          MOV A,P0
          AND A,#11110000b
          ADD A,AUX
          MOV P0,A
          NOP

          CLR1 En                   ;flanc de baixada de enable
          NOP

          SET1 En                   ;enable a 1

          MOV A,DADA_LCD            ;posa a DB7, DB6, DB5 i DB4 els 4 ultims bits de la instrucció
          AND A,#00001111b
          MOV S:AUX,A

          MOV A,P0
          AND A,#11110000b
          ADD A,AUX
          MOV P0,A

          NOP

          CLR1 En                   ;flanc de baixada de enable
          NOP

          MOV PCC,#00000000b           ;fcpu=fx

          RET


;-------------------------------------------------------------------------------------------------------------
WRITE_DATA
;-------------------------------------------------------------------------------------------------------------
;envia a la pantalla LCD la dada continguda a la variable DADA_LCD

          MOV PCC,#00000010b        ;fcpu=fx/4

          NOP
          SET1 RS                   ;configura mode dades
          NOP
          CLR1 RW                   ;configura mode escriptura
          NOP

          SET1 En                   ;enable a 1

          MOV A,DADA_LCD            ;posa a DB7, DB6, DB5 i DB4 els 4 primer bits de la instrucció
          AND A,#11110000b
          ROR A,1
          ROR A,1
          ROR A,1
          ROR A,1
          MOV S:AUX,A

          MOV A,P0
          AND A,#11110000b
          ADD A,AUX
          MOV P0,A
          NOP
```

```
          CLR1 En                   ;flanc de baixada de enable
          NOP

          SET1 En                   ;enable a 1


          MOV A,DADA_LCD            ;posa a DB7, DB6, DB5 i DB4 els 4 ultims bits de la instrucció
          AND A,#00001111b
          MOV S:AUX,A

          MOV A,P0
          AND A,#11110000b
          ADD A,AUX
          MOV P0,A
          NOP

          CLR1 En                   ;flanc de baixada de enable
          NOP

          MOV PCC,#00000000b        ;fcpu=fx

          RET


;-------------------------------------------------------------------------------------------------------------
ESPERA_BF
;-------------------------------------------------------------------------------------------------------------
;espera a que el flag BF de lapantalla LCD es posi a 0

          MOV PCC,#00000010b         ;fcpu=fx/4

          SET1 M_DB4                 ;configura DB4, DB5, DB6 i DB7 com a entrades
          SET1 M_DB5
          SET1 M_DB6
          SET1 M_DB7

          CLR1 RS                   ;configura mode instruccions
          SET1 RW                   ;configura mode lectura
          NOP

          SET1 En                   ;enable a 1
          NOP
test_BF
          BF DB7,skip_BF

          CLR1 En                   ;enable a 0
          NOP
          SET1 En                   ;enable a 1
          NOP
          CLR1 En                   ;enable a 0
          NOP
          SET1 En                   ;enable a 1

          BR test_BF

skip_BF
          CLR1 En                   ;enable a 0
          NOP
          SET1 En                   ;enable a 1
          NOP
          CLR1 En                   ;enable a 0
          NOP

          CLR1 RW                   ;configura mode escriptura
          NOP

          CLR1 M_DB4                 ;configura DB4, DB5, DB6 i DB7 com a sortides
          CLR1 M_DB5
          CLR1 M_DB6
          CLR1 M_DB7

          MOV PCC,#00000000b         ;fcpu=fx

          RET
```

```
;*****************************************************************************************************************
;* RUTINES D'ESPERA                                                                                              *
;*****************************************************************************************************************


;-----------------------------------------------------------------------------------------------------------------
ESPERA_1ms
;-----------------------------------------------------------------------------------------------------------------
;mitjançant el TIMER90, espera 1ms en mode halt

          MOV TMC90,#00000000b    ;configuracio TIMER90  (Tmin=0.8us Tmax=53.3ms)

          MOVW AX,TM90            ;guardem el valor actual del TIMER90 a AX
          ADDW AX,#01229d         ;incrementem AX en 1229d

          SET1 TMMK90             ;guardem el nou valor a AX a CR90
          MOVW CR90,AX

          MOV IF0,#00000000b      ;desactiva tots el flags de requeriment d'interrupció
          MOV IF1,#00000000b

          CLR1 TMMK90             ;activa interrupcions del TIMER90
          DI

          HALT


          CLR1 TMIF90             ;esborra el flag de requeriment d'interrupcio
          SET1 TMMK90             ;desactiva interrupcions TIMER90
          EI                      ;activa les interrupcions

          RET


;-----------------------------------------------------------------------------------------------------------------
ESPERA_3ms
;-----------------------------------------------------------------------------------------------------------------
;mitjançant el TIMER90, espera 3ms en mode halt

          MOV TMC90,#00000000b    ;configuracio TIMER90  (Tmin=0.8us Tmax=53.3ms)

          MOVW AX,TM90            ;guardem el valor actual del TIMER90 a AX
          ADDW AX,#03686d         ;incrementem AX en 3686d

          SET1 TMMK90             ;guardem el nou valor a AX a CR90
          MOVW CR90,AX

          MOV IF0,#00000000b      ;desactiva tots el flags de requeriment d'interrupció
          MOV IF1,#00000000b

          CLR1 TMMK90             ;activa interrupcions del TIMER90
          DI

          HALT


          CLR1 TMIF90             ;esborra el flag de requeriment d'interrupcio
          SET1 TMMK90             ;desactiva interrupcions TIMER90
          EI                      ;activa les interrupcions

          RET

;-----------------------------------------------------------------------------------------------------------------
ESPERA_5ms
;-----------------------------------------------------------------------------------------------------------------
;mitjançant el TIMER90, espera 3ms en mode halt

          MOV TMC90,#00000000b    ;configuracio TIMER90  (Tmin=0.8us Tmax=53.3ms)

          MOVW AX,TM90            ;guardem el valor actual del TIMER90 a AX
          ADDW AX,#06144d         ;incrementem AX en 6144d

          SET1 TMMK90             ;guardem el nou valor a AX a CR90
          MOVW CR90,AX

          MOV IF0,#00000000b      ;desactiva tots el flags de requeriment d'interrupció
          MOV IF1,#00000000b

          CLR1 TMMK90             ;activa interrupcions del TIMER90
          DI

          HALT
```

```
          CLR1 TMIF90             ;esborra el flag de requeriment d'interrupcio
          SET1 TMMK90             ;desactiva interrupcions TIMER90
          EI                      ;activa les interrupcions

          RET

;-----------------------------------------------------------------------------------------------------------
ESP_BIT_STOP
;-----------------------------------------------------------------------------------------------------------
;espera 60us per a donar temps a que s'acabi d'emetre el bit d'stop del darrer byte a transmetre

          MOV B,#50d              ;compta 60us
          DBNZ B,$

          RET


;-----------------------------------------------------------------------------------------------------------
ESP_MARGE
;-----------------------------------------------------------------------------------------------------------
;espera 1.224ms, s'utilitza com a marge de seguretat per a que el receptor estigui rebent quan l'emissor
;comença a emetre el seu missatge

          MOV PCC,#00000010b      ;fcpu=fx/4

          MOV B,#255d             ;compta 1.224ms
          DBNZ B,$

          MOV PCC,#00000000b      ;fcpu=fx

          RET



;-----------------------------------------------------------------------------------------------------------
ESPERA_POLSADOR
;-----------------------------------------------------------------------------------------------------------
;espera 500ms
;després de pitjar un polsador el programa executa una sèrie de instruccións.Si questes duren menys temps
;que el temps de polsació, pot succeir que el programa torni a consultar el polsador i el trobi
;encara pitjat. Això podria provocar s'executessin instruccions no desijades.
;Per aquest motiu, abans de tornar a consultar un polsador es deixa transcórrer 500ms, de manera que
;si la polsació dura manys de 500ms al tornar a consultar el polsador no se'l troba pitjat.

          MOV PCC,#00000010b      ;fcpu=fx/4

          MOV C,#250d
loop_espera_pols1
          MOV B,#203d             ;compta 1ms
          DBNZ B,$

          DBNZ C,loop_espera_pols1

          MOV C,#250d
loop_espera_pols2
          MOV B,#203d             ;compta 1ms
          DBNZ B,$

          DBNZ C,loop_espera_pols2

          MOV PCC,#00000000b      ;fcpu=fx

          RET


;-----------------------------------------------------------------------------------------------------------
TEMPS_CONV_LECT
;-----------------------------------------------------------------------------------------------------------
;deixa passar el mateix temps que el procés de converisió i lectura de la mesura de temperatura (775ms)

           CALL INI_TMR90_760ms
           HALT

           CLR1 TMIF90             ;esborra el flag de requeriment d'interrupcio
           SET1 TMMK90             ;desactiva interrupcions TIMER90
           EI

           CALL INI_TMR90_15ms
           HALT
```

```
            CLR1 TMIF90             ;esborra el flag de requeriment d'interrupcio
            SET1 TMMK90             ;desactiva interrupcions TIMER90
            EI

            RET

;-----------------------------------------------------------------------------------------------------------
INI_TMR90_760ms
;-----------------------------------------------------------------------------------------------------------
;comença un comptatge de 760ms amb el TIMER90

            MOV TMC90,#00000010b    ;configuracio TIMER90  (Tmin=13us Tmax=853ms)
            MOVW AX,TM90            ;guardem el valor actual del TIMER90 a DE
            ADDW AX,#58368d         ;(58368·Tmin= 760ms (>750 d'espera de conversió+ temps de càlculs))
            SET1 TMMK90             ;desactiva interrupcions TIMER90
            MOVW CR90,AX            ;actualitza el Compare Register

            MOV IF0,#00000000b      ;desactiva tots el flags de requeriment d'interrupció
            MOV IF1,#00000000b

            CLR1 TMMK90             ;activa interrupcions del TIMER90
            DI                      ;desactiva el servei d'interrupcions

            RET

;-----------------------------------------------------------------------------------------------------------
INI_TMR90_15ms
;-----------------------------------------------------------------------------------------------------------
;comença un comptatge de 15ms amb el TIMER90

            MOV TMC90,#00000010b    ;configuracio TIMER90  (Tmin=13us Tmax=853ms)
            MOVW AX,TM90            ;guardem el valor actual del TIMER90 a DE
            ADDW AX,#1152d           ;(1152·Tmin= 15ms)
            SET1 TMMK90             ;desactiva interrupcions TIMER90
            MOVW CR90,AX            ;actualitza el Compare Register

            MOV IF0,#00000000b      ;desactiva tots el flags de requeriment d'interrupció
            MOV IF1,#00000000b

            CLR1 TMMK90             ;activa interrupcions del TIMER90
            DI                      ;desactiva el servei d'interrupcions

            RET




;***********************************************************************************************************
;* RUTINES DE PREPROCESSAT I POST PROCESSAT DE RF PER A DEBUG                                              *
;***********************************************************************************************************

;-----------------------------------------------------------------------------------------------------------
PRE_SINC      ;(SENSOR)
;-----------------------------------------------------------------------------------------------------------
;preprocessat de les variables de debug abans de rebre el missatge de sincronització

          MOV S:INF_P_SINCRO_RX,#Z_
          MOVW AX,#TAULA_DEBUG          ;COMP_DEBUG=#TAULA_DEBUG
          MOVW S:COMP_DEBUG,AX
          CLR1 S:ADR_SIST_REBUDA.0
          CALL INI_TAULA_DEBUG

          RET
;-----------------------------------------------------------------------------------------------------------
POST_SINC      ;(SENSOR)
;-----------------------------------------------------------------------------------------------------------
;postprocessat de les variables de debug després de rebre el missatge de sincronització

          MOVW AX,#TAULA_DEBUG1
          MOVW DE,AX
          CALL BOLCA

          RET
;-----------------------------------------------------------------------------------------------------------
PRE_CONF       ;(SENSOR)
;-----------------------------------------------------------------------------------------------------------
;preprocessat deles variables de debug abans de rebre el missatge de confirmació

          MOV S:INF_CONFIRM_RX,#Z_
          MOVW AX,#TAULA_DEBUG          ;COMP_DEBUG=#TAULA_DEBUG
```

```
          MOVW S:COMP_DEBUG,AX
          CLR1 S:ADR_SIST_REBUDA.0
          CALL INI_TAULA_DEBUG

          RET
;-------------------------------------------------------------------------------------------------------------------
POST_CONF       ;(SENSOR)
;-------------------------------------------------------------------------------------------------------------------
;postprocessat de les variables de debug després de rebre el missatge de confirmació

          ;MOVW AX,#TAULA_CONF
          ;MOVW DE,AX
          ;CALL BOLCA

          RET

;-------------------------------------------------------------------------------------------------------------------
PRE_TEMP1       ;(MASTER)
;-------------------------------------------------------------------------------------------------------------------
;preprocessat de les variables de debug abans del primer intent de recepció de la temperatura

          MOV S:INF_RX_TEMP,#Z_

          MOVW AX,#TAULA_DEBUG          ;inicialitza COMP_DEBUG (COMP_DEBUG=#TAULA_DEBUG)
          MOVW S:COMP_DEBUG,AX

          CLR1 S:ADR_SIST_REBUDA.0      ;fixa a 0 el flag que indica si s'ha rebut l'adreça del sistema
          CALL INI_TAULA_DEBUG          ;inicialitza la TAULA_DEBUG

          RET

;-------------------------------------------------------------------------------------------------------------------
POST_TEMP1       ;(MASTER)
;-------------------------------------------------------------------------------------------------------------------
;postprocessat de les variables de debug després del primer intent de recepció de la temperatura

   ;emmagatzemament de INF_RX_TEMP
          MOVW AX,S:COMP_TAU_INF_TEMP1  ;MOV [COMP_TAU_INF_TEMP1],INF_RX_TEMP
          MOVW HL,AX
          MOV A,INF_RX_TEMP
          MOV [HL],A

          MOVW AX,S:COMP_TAU_INF_TEMP1  ;INC COMP_TAU_TEMP1
          INCW AX
          MOVW S:COMP_TAU_INF_TEMP1,AX
   ;fi emmagatzemament de INF_RX_TEMP

   ;emmagatzemament dels bytes rebuts
          MOV A,UNIT_ACT_RX             ;compara UNIT_ACT_RX amb UNIT_ACT_VISU
          CMP A,UNIT_ACT_VISU           ;si la unitat que estem rebent no coincideix amb la que volem
          BNZ skip_bytes_temp1_RX       ;visualitzar no guarda els bytes rebuts, si coincideixen,
                                        ;guarda els bytes rebuts
          MOV A,ADR_RX_TEMP
          MOV S:ADR_RX_TEMP1,A
          MOV A,BYTE_1_TEMP
          MOV S:BYTE_1_TEMP1,A
          MOV A,BYTE_2_TEMP
          MOV S:BYTE_2_TEMP1,A
   ;fi emmagatzemament dels bytes rebuts

   ;emmagatzemament dels bits rebuts
          MOVW AX,#TAULA_DEBUG1
          MOVW DE,AX
          CALL BOLCA
   ;fi emmagatzemament dels bits rebuts

skip_bytes_temp1_RX
          RET

;-------------------------------------------------------------------------------------------------------------------
PRE_TEMP2       ;(MASTER)
;-------------------------------------------------------------------------------------------------------------------
;preprocessat de les variables de debug abans del segon intent de recepció de la temperatura

          MOV S:INF_RX_TEMP,#Z_
          MOVW AX,#TAULA_DEBUG          ;COMP_DEBUG=#TAULA_DEBUG
          MOVW S:COMP_DEBUG,AX
          CLR1 S:ADR_SIST_REBUDA.0
          CALL INI_TAULA_DEBUG

          RET
```

```
;-------------------------------------------------------------------------------------------------------------------
POST_TEMP2      ;(MASTER)
;-------------------------------------------------------------------------------------------------------------------
;postprocessat de les variables de debug després del segon intent de recepció de la temperatura

   ;emmagatzemament de INF_RX_TEMP
          MOVW AX,S:COMP_TAU_INF_TEMP2  ;MOV [COMP_TAU_TEMP2],INF_RX_TEMP
          MOVW HL,AX
          MOV A,INF_RX_TEMP
          MOV [HL],A

          MOVW AX,S:COMP_TAU_INF_TEMP2  ;INC COMP_TAU_TEMP2
          INCW AX
          MOVW S:COMP_TAU_INF_TEMP2,AX
   ;fi emmagatzemament de INF_RX_TEMP

   ;emmagatzemament dels bytes rebuts
          MOV A,UNIT_ACT_RX             ;compara UNIT_ACT_RX amb UNIT_ACT_VISU
          CMP A,UNIT_ACT_VISU           ;si la unitat que estem rebent no coincideix amb la que volem
          BNZ skip_bytes_temp2_RX       ;visualitzar no guarda els bytes rebuts si coincideixen,
                                        ;guarda els bytes rebuts
          MOV A,ADR_RX_TEMP
          MOV S:ADR_RX_TEMP2,A
          MOV A,BYTE_1_TEMP
          MOV S:BYTE_1_TEMP2,A
          MOV A,BYTE_2_TEMP
          MOV S:BYTE_2_TEMP2,A

skip_bytes_temp2_RX

    ;fi emmagatzemament dels bytes rebuts

    ;emmagatzemament informació rebuda
          ;MOVW AX,#TAULA_TEMP2
          ;MOVW DE,AX
          ;CALL BOLCA
    ;fi emmagatzemament informació rebuda

          RET


;-------------------------------------------------------------------------------------------------------------------
POST_TEMP2_NO_RECEP   ;(MASTER)
;-------------------------------------------------------------------------------------------------------------------
;rutina d'emmagatzemamanet quan el segon intent de mesura no s'ha de portar a terme
;perquè ja s'ha rebut correctament la temperatura en el primer intent
;emmagatzema D_ a la posició corresponent de TAU_INF_RX_TEMP2

          MOVW AX,S:COMP_TAU_INF_TEMP2       ;MOV [COMP_TAU_TEMP2],#D
          MOVW HL,AX
          MOV A,#D_
          MOV [HL],A

          MOVW AX,S:COMP_TAU_INF_TEMP2       ;INC COMP_TAU_TEMP2
          INCW AX
          MOVW S:COMP_TAU_INF_TEMP2,AX

          RET
;-------------------------------------------------------------------------------------------------------------------
PRE_TX_TEMP1           ;(SENSOR)
;-------------------------------------------------------------------------------------------------------------------
;preprocessat de les variables de debug abans de fer el primer intent de transmissió de temperatura
          MOV S:INF_TX_TEMP,#Z_

          RET

;-------------------------------------------------------------------------------------------------------------------
POST_TX_TEMP1          ;(SENSOR)
;-------------------------------------------------------------------------------------------------------------------
;postprocessat de les variables de debug després de fer el primer intent de transmissió de temperatura

          MOV S:AUX,A
          MOV A,INF_TX_TEMP
          MOV S:INF_TX_TEMP1,A
          MOV A,AUX

          RET
```

```
;-----------------------------------------------------------------------------------------------------------------
PRE_TX_TEMP2            ;(SENSOR)
;-----------------------------------------------------------------------------------------------------------------
;preprocessat de les variables de debug abans de fer el segon intent de transmissió de temperatura

          MOV S:INF_TX_TEMP,#Z_
          RET

;-----------------------------------------------------------------------------------------------------------------
POST_TX_TEMP2           ;(SENSOR)
;-----------------------------------------------------------------------------------------------------------------
;postprocessat de les variables de debug després de fer el segon intent de transmissió de temperatura

          MOV S:AUX,A
          MOV A,INF_TX_TEMP
          MOV S:INF_TX_TEMP2,A
          MOV A,AUX

          RET



;-----------------------------------------------------------------------------------------------------------------
CONVERTEIX    ;(master i sensor)
;-----------------------------------------------------------------------------------------------------------------
;a partir de CONVERTEIX_LCD en troba la representació en hexadecimal (la emmagatzema a PART1 i PART2),
;i crida la funció que ho converteix a codi LCD (LCD1 i LCD2)

          MOV A,CONVERTEIX_LCD
          AND A,#11110000b
          ROR A,1
          ROR A,1
          ROR A,1
          ROR A,1
          MOV S:PART1,A

          MOV A,CONVERTEIX_LCD
          AND A,#00001111b
          MOV S:PART2,A

          MOV A,PART2
          CALL TROBA_LCD

          MOV A,LCD1
          MOV S:LCD2,A

          MOV A,PART1
          CALL TROBA_LCD

          RET

;-----------------------------------------------------------------------------------------------------------------
TROBA_LCD    ;(master i sensor)
;-----------------------------------------------------------------------------------------------------------------
;a partir de A (PART1 o PART2 en codi hexadecimal) troba el codi LCD

          CMP A,#010d
          BC menor10
          BNC major10
menor10
          ADD A,#00110000b
          MOV S:LCD1,A
          RET

major10
          ADD A,#07d
          AND A,#00001111b
          ADD A,#01000000b
          MOV S:LCD1,A
          RET


;-----------------------------------------------------------------------------------------------------------------
INI_VAR_DEBUG
;-----------------------------------------------------------------------------------------------------------------
;inicialitza a EEh les variables de debug

           MOV S:ADR_RX_CONV_TEMP,#0EEh
           MOV S:BYTE_1_CONV_TEMP,#0EEh
           MOV S:ADR_RX_SINCRO,#0EEh
           MOV S:BYTE_1_SINCRO,#0EEh
           MOV S:ADR_RX_TEMP,#0EEh
```

```
            MOV S:BYTE_1_TEMP,#0EEh
            MOV S:BYTE_2_TEMP,#0EEh
            MOV S:ADR_RX_TEMP1,#0EEh
            MOV S:BYTE_1_TEMP1,#0EEh
            MOV S:BYTE_2_TEMP1,#0EEh
            MOV S:ADR_RX_TEMP2,#0EEh
            MOV S:BYTE_1_TEMP2,#0EEh
            MOV S:BYTE_2_TEMP2,#0EEh
            MOV S:ADR_RX_CONF,#0EEh
            MOV S:BYTE_1_CONF,#0EEh

            RET


;-------------------------------------------------------------------------------------------------------------------
INI_TAULA_DEBUG          ;(master i sensor)
;-------------------------------------------------------------------------------------------------------------------
;inicialitza amb el valor 11h la taula de on es guarden els bits rebuts

          MOVW AX,#TAULA_DEBUG
          MOVW HL,AX

          MOV B,#16d
loop_ini_tau
          MOV A,#11h
          MOV [HL],A

          INCW HL

          DBNZ B,loop_ini_tau

          RET

;-------------------------------------------------------------------------------------------------------------------
BOLCA        ;(master i sensor)
;-------------------------------------------------------------------------------------------------------------------
;bolca el contingut de la TAULA_DEBUG a la taula que comença a l'adreça continguda a DE

          MOVW AX,#TAULA_DEBUG
          MOVW HL,AX

          MOV B,#16d
loop_bolca
          MOV A,[HL]
          MOV [DE],A

          INCW HL
          INCW DE

          DBNZ B,loop_bolca

          RET




;*******************************************************************************************************************
;* RUTINES  VISUALITZACIO DE DEBUG                                                                                 *
;*******************************************************************************************************************

;-------------------------------------------------------------------------------------------------------------------
MOSTRA_INFs
;-------------------------------------------------------------------------------------------------------------------
;mostra per pantalla les variables INF del màster:
;INF_P_CONV_TX,   INF_P_CONV_TX,  INF_P_SINCRO_TX,   TAU_INF_RX_TEMP1, TAU_INF_RX_TEMP2
;a més a més mostra la temperatura de la unitat màster

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV A,INF_P_CONV_TX         ;mostra els INFs del pols de conversió, de la conversio de temperatura
          MOV S:DADA_LCD,A            ;i del pols de sincronització
          CALL WRITE_DATA

          MOV A,INF_P_CONV_TX
          MOV S:DADA_LCD,A
```

```
          CALL WRITE_DATA

          MOV A,INF_P_SINCRO_TX
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA

          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA

          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA

          MOVW AX,#TAU_INF_RX_TEMP1
          CALL MOSTRA_INFs_RX_TEMP    ;7 caràcters dels INFs del primer intent de recepció

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          CALL SCRT_VISU              ;converteix la temperatura del format de l'scratchpad a
                                      ;VISU1_ACT i VISU2_ACT
          CALL CALC_TEMPER_LCD        ;converteix VISU1_ACT i VISU2_ACT en format LCD
                                      ;(SIGNE, DESEN_LCD, UNIT_LCD, PUNT i DECIMAL_LCD)
          CALL PRINT_TEMP             ;mostra per la pantalla LCD la temperatura del màster

          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA

          MOVW AX,#TAU_INF_RX_TEMP2
          CALL MOSTRA_INFs_RX_TEMP    ;7 caràcters dels INFs del segon intent de recepció

          RET


;-----------------------------------------------------------------------------------------------------------
MOSTRA_BYTE_UNIT_ACT
;-----------------------------------------------------------------------------------------------------------
;per a la unitat triada, mostra per pantalla en hexadecimal l'adreça i els dos bytes rebuts
;en cada intent de recepcció de la temperatura

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          CALL ESCRIU_UNIT_ACT

          MOV A,ADR_RX_TEMP1          ;mostra la recepció del la primera temperatura de la unitat escollida
          CALL MOSTRA_BYTE

          MOV A,BYTE_1_TEMP1
          CALL MOSTRA_BYTE

          MOV A,BYTE_2_TEMP1
          CALL MOSTRA_BYTE

          MOV S:DADA_LCD,#espai_      ;espai
          CALL WRITE_DATA

          MOV A,ADR_RX_TEMP2          ;mostra la recepció del la segona temperatura de la unitat escollida
          CALL MOSTRA_BYTE

          MOV A,BYTE_1_TEMP2
          CALL MOSTRA_BYTE

          MOV A,BYTE_2_TEMP2
          CALL MOSTRA_BYTE

          RET


;-----------------------------------------------------------------------------------------------------------
MOSTRA_TOTS_BYTES
;-----------------------------------------------------------------------------------------------------------
;mostra per pantalla els bytes rebuts de l atemperatura de cada unitat (sense l'adreça)
;també mostra els bytes de la temperatura de la unitat màster

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF
```

```
          CALL MOSTRA_2BYTES_TEMPERS

          RET


;-------------------------------------------------------------------------------------------------------------------
MOSTRA_BITS1
;-------------------------------------------------------------------------------------------------------------------
;mostra els bits rebuts en el primer intent de recepció de la temperatura de la unitat triada

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOVW DE,#TAULA_DEBUG1
          CALL MOSTRA_TAULA

          RET

;-------------------------------------------------------------------------------------------------------------------
MOSTRA_BITS2
;-------------------------------------------------------------------------------------------------------------------
;mostra els bits rebuts en el segon intent de recepció de la temperatura de la unitat triada

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOVW DE,#TAULA_DEBUG
          CALL MOSTRA_TAULA

          RET


;-------------------------------------------------------------------------------------------------------------------
MOSTRA_TAULA
;-------------------------------------------------------------------------------------------------------------------
;visualitza per la pantalla en format hexadecimal la taula que comença a l'adreça que continguda a DE
          MOV B,#8d
loop_mostra1
          MOV A,[DE]
          CALL MOSTRA_BYTE
          INCW DE

          DBNZ B,loop_mostra1

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF


          MOV B,#8d
loop_mostra2
          MOV A,[DE]
          CALL MOSTRA_BYTE
          INCW DE

          DBNZ B,loop_mostra2

          RET




;-------------------------------------------------------------------------------------------------------------------
INFO_LCD_SENS123    ;(sensor)
;-------------------------------------------------------------------------------------------------------------------
;subrutina que crida a les funcions que ensenyen per la pantalla del sensor les variables de debug
;si es polsa MODE crida a INFO_LCD_SENS1
;si es polsa MES crida a INFO_LCD_SENS2
;si es polsa MENYS crida a INFO_LCD_SENS3

          BF LCD_CONNECT,skip_i_LCD_sens123  ;si LCD_CONNECT està a "0" vol dir que la placa de configuració
                                             ;està connectada i, al no haver-hi LCD, no s'executa la funció
          BF MODE,info_LCD_sens_1
          BF MES,info_LCD_sens_2
          BF MENYS,info_LCD_sens_3

          RET                       ;-->

info_LCD_sens_1
```

```
          CALL INFO_LCD_SENS1
          RET                       ;-->
info_LCD_sens_2
          CALL INFO_LCD_SENS2
          RET                       ;-->
info_LCD_sens_3
          CALL INFO_LCD_SENS3

skip_i_LCD_sens123
          RET                       ;-->


;-------------------------------------------------------------------------------------------------------------
INFO_LCD_SENS1      ;(sensor)
;-------------------------------------------------------------------------------------------------------------
;ensenya per pantalla les variables INFs del sensor,
;els bytes rebuts del pols de sincronitació de mesura temperatura,
;els bytes rebuts del pols de sincronitació,
;els bytes rebuts del misstage de confirmació de recepció de temperatura
;i la temperatura de la unitat sensora

          BF LCD_CONNECT,skip_i_LCD_sens1  ;si LCD_CONNECT està a "0" vol dir que la placa de configuració
                                           ;està connectada i, al no haver-hi LCD, no s'executa la funció
          MOV S:DADA_LCD,#00000001b        ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV A,INF_P_CONV_RX              ;mostra els INFs (6 caràcteres)
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV A,INF_TEMPER
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV A,INF_P_SINCRO_RX
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV A,INF_TX_TEMP1
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV A,INF_CONFIRM_RX
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV A,INF_TX_TEMP2
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV S:DADA_LCD,#espai_           ;espai
          CALL WRITE_DATA

          MOV A,ADR_RX_CONV_TEMP           ;mostra la recepció dels polsos de sincronització
          CALL MOSTRA_BYTE

          MOV A,BYTE_1_CONV_TEMP
          CALL MOSTRA_BYTE

          MOV S:DADA_LCD,#espai_           ;espai
          CALL WRITE_DATA

          MOV A,ADR_RX_SINCRO
          CALL MOSTRA_BYTE

          MOV A,BYTE_1_SINCRO
          CALL MOSTRA_BYTE

          MOV S:DADA_LCD,#11000000b        ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV A,ADR_RX_CONF                ;mostra la recepció de la confirmació
          CALL MOSTRA_BYTE

          MOV A,BYTE_1_CONF
          CALL MOSTRA_BYTE

          MOV S:DADA_LCD,#espai_           ;espai
          CALL WRITE_DATA
```

```
          CALL SCRT_VISU                    ;mostra per la pantalla LCD la temperatura de la unitat sensora
          CALL CALC_TEMPER_LCD
          CALL PRINT_TEMP

skip_i_LCD_sens1
          RET


;-----------------------------------------------------------------------------------------------------------------
INFO_LCD_SENS2     ;(sensor)
;-----------------------------------------------------------------------------------------------------------------
;mostra la taula que conté els bits del pols de sincronització

          BF LCD_CONNECT,skip_i_LCD_sens2  ;si LCD_CONNECT està a "0" vol dir que la placa de configuració
                                           ;està connectada i, al no haver-hi LCD, no s'executa la funció
          MOV S:DADA_LCD,#00000001b        ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOVW DE,#TAULA_DEBUG1
          CALL MOSTRA_TAULA

skip_i_LCD_sens2

          RET


;-----------------------------------------------------------------------------------------------------------------
INFO_LCD_SENS3        ;(sensor)
;-----------------------------------------------------------------------------------------------------------------
;mostra la taula que conté els bits del missatge de sincronització

          BF LCD_CONNECT,skip_i_LCD_sens3  ;si LCD_CONNECT està a "0" vol dir que la placa de configuració
                                           ;està connectada i, al no haver-hi LCD, no s'executa la funció
          MOV S:DADA_LCD,#00000001b        ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOVW DE,#TAULA_DEBUG
          CALL MOSTRA_TAULA

skip_i_LCD_sens3

          RET


;-----------------------------------------------------------------------------------------------------------------
INFO_LCD_SENS_INT     ;(sensor)
;-----------------------------------------------------------------------------------------------------------------
;mostra per la pantalla el número que indica la quantitat de sincronitzacions que ha rebut el sensor,
;la variable INF de la sincronització (INF_P_SINCRO), i el missatge de sincronització rebut

          BF LCD_CONNECT,skip_i_LCD_sens_int ;si LCD_CONNECT està a "0" vol dir que la placa de configuració
                                             ;està connectada i, al no haver-hi LCD, no s'executa la funció
          MOV S:DADA_LCD,#00000001b          ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV A,COMP_SLOTS_INTER_TEMP        ;mostra el número en hexadecimal del pols de
          CALL MOSTRA_BYTE                   ;sincronització intermig

          MOV S:DADA_LCD,#espai_             ;espai
          CALL WRITE_DATA

          MOV A,INF_P_SINCRO_RX              ;mostra l'inf de la sincronització
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV S:DADA_LCD,#espai_             ;espai
          CALL WRITE_DATA

          MOV A,ADR_RX_SINCRO                ;mostra l'adreça rebuda
          CALL MOSTRA_BYTE

          MOV A,BYTE_1_SINCRO                ;mostra el missatge rebut
          CALL MOSTRA_BYTE

skip_i_LCD_sens_int

          RET
```

```
;-----------------------------------------------------------------------------------------------------------------
MOSTRA_2BYTES_TEMPERS  ;(master)
;-----------------------------------------------------------------------------------------------------------------
;---mostra els 8 conjunts de 2 bytes emmagatzemats a TAU_TEMP_UNITATS

          MOVW AX,#TAU_TEMP_UNITATS
          MOVW HL,AX
          MOV B,#4d
loop_mostra_2bytes_1
          MOV A,[HL]
          CALL MOSTRA_BYTE
          INCW HL
          MOV A,[HL]
          CALL MOSTRA_BYTE
          INCW HL

          DBNZ B,loop_mostra_2bytes_1

          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV B,#4d
loop_mostra_2bytes_2
          MOV A,[HL]
          CALL MOSTRA_BYTE
          INCW HL
          MOV A,[HL]
          CALL MOSTRA_BYTE
          INCW HL

          DBNZ B,loop_mostra_2bytes_2

          RET


;-----------------------------------------------------------------------------------------------------------------
MOSTRA_INFs_RX_TEMP    ;(master)
;-----------------------------------------------------------------------------------------------------------------
;mostra 7 caràcters informant de la recepció de les temperatures de les 7 unitats

          MOVW HL,AX
          MOV B,#7d
loop_mostra_infs
          MOV A,[HL]
          MOV S:DADA_LCD,A
          CALL WRITE_DATA
          INCW HL

          DBNZ B,loop_mostra_infs

          RET


;-----------------------------------------------------------------------------------------------------------------
MOSTRA_BYTE    ;(master i sensor)
;-----------------------------------------------------------------------------------------------------------------
;--- a partir de A en troba la representació hexadecimal i la visualitza amb 2 caràcters

          MOV S:CONVERTEIX_LCD,A
          CALL CONVERTEIX

          MOV A,LCD1
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          MOV A,LCD2
          MOV S:DADA_LCD,A
          CALL WRITE_DATA

          RET
```

```
;*****************************************************************************************************************
;* RUTINES DE SIMULACIÓ DE MISSATGES REBUTS AMB ERRORS                                                           *
;*****************************************************************************************************************

;-----------------------------------------------------------------------------------------------------------------
INTR_ERR_SINC1    ;(master)
;-----------------------------------------------------------------------------------------------------------------
;introdueix un error en l'emissió del primer missatge de sincronització

          BT S:GEN_ERRORS.0,gen_err_sinc1
          BF S:GEN_ERRORS.0,fi_gen_err_sinc1
gen_err_sinc1
          SET1 S:SINCROp.0
fi_gen_err_sinc1

          RET

;-----------------------------------------------------------------------------------------------------------------
INTR_ERR_SINC2    ;(master)
;-----------------------------------------------------------------------------------------------------------------
;introdueix un error en l'emissió del segon missatge de sincronització

          BT S:GEN_ERRORS.1,gen_err_sinc2
          BF S:GEN_ERRORS.1,fi_gen_err_sinc2
gen_err_sinc2
          CLR1 S:SINCROs.6
fi_gen_err_sinc2

          RET

;-----------------------------------------------------------------------------------------------------------------
INTR_ERR_SINC3     ;(master)
;-----------------------------------------------------------------------------------------------------------------
;introdueix un error en l'emissió del tercer missatge de sincronització

          BT S:GEN_ERRORS.2,gen_err_sinc3
          BF S:GEN_ERRORS.2,fi_gen_err_sinc3
gen_err_sinc3
          CLR1 S:SINCROp.7
fi_gen_err_sinc3

          RET

;-----------------------------------------------------------------------------------------------------------------
INTR_ERR_SINC_TEMP  ;(master)
;-----------------------------------------------------------------------------------------------------------------
;introdueix un error en l'emissió dels missatges de sincronització de mesura de temperatura

          BT S:GEN_ERRORS.3,gen_err_sinc_temp1
          BF S:GEN_ERRORS.3,fi_gen_err_sinc_temp1
gen_err_sinc_temp1
          CLR1 A.0
fi_gen_err_sinc_temp1

          RET

;-----------------------------------------------------------------------------------------------------------------
INTR_ERR_CONF       ;(master)
;-----------------------------------------------------------------------------------------------------------------
;introdueix un error en l'emissió del missatge de confirmació de recepció de la mesura de la temperatura

          BT S:GEN_ERRORS.4,gen_err_conf
          BF S:GEN_ERRORS.4,fi_gen_err_conf
gen_err_conf
          SET1 A.0
fi_gen_err_conf
          RET


;-----------------------------------------------------------------------------------------------------------------
INTR_ERR_TX_TEMP1    ;(sensor)
;-----------------------------------------------------------------------------------------------------------------
;introdueix errors en l'emissió del 1r missatge de de temperatura

          BT S:GEN_ERRORS.0,gen_err_TX_temp1
          BF S:GEN_ERRORS.0,fi_gen_err_TX_temp1
gen_err_TX_temp1
          MOV A,BYTE1_TX_TEMP
          XOR A,#10101010b
          MOV BYTE1_TX_TEMP,A
fi_gen_err_TX_temp1
          RET
```

```
;-------------------------------------------------------------------------------------------------------------------
INTR_ERR_TX_TEMP2    ;(sensor)
;-------------------------------------------------------------------------------------------------------------------
;desintrodueix els errors en l'emissió del 2n missatge de de temperatura

          BF S:GEN_ERRORS.1,desgenera_err_TX_temp2
          BT S:GEN_ERRORS.1,fi_desgenera_err_TX_temp2
desgenera_err_TX_temp2
          MOV A,BYTE1_TX_TEMP
          XOR A,#10101010b
          MOV BYTE1_TX_TEMP,A
fi_desgenera_err_TX_temp2
          RET


;-------------------------------------------------------------------------------------------------------------------
SELEC_ERRORS_SENS    ;(sensor)
;-------------------------------------------------------------------------------------------------------------------
;segons l'estat dels polsadors just després de fer el reset de la unitat sensora
;modifica adienmet la variable GEN_ERRORS per a simular l'efecte de la recepció
;incorrecta de missatges per part del màster
;també possibilita entrar en el mode de test del sensor

          MOV S:GEN_ERRORS,#00000000b
          CLR1 MODE_TEST               ;inicialitzem MODE_TEST a 0

          BT MODE,errors_TX_temp
          BF MODE,altres_errors_sens

errors_TX_temp
          BF MENYS,err_10X_sens
          BT MENYS,err_11X_sens
err_10X_sens
          BF MES,fi_err_sens           ;1 00
          BT MES,err_TX_temp1          ;1 01
err_11X_sens
          BF MES,err_TX_temp2          ;1 10
          BT MES,fi_err_sens           ;1 11


altres_errors_sens
          BF MENYS,err_00X_sens
          BT MENYS,err_01X_sens
err_00X_sens
          BF MES,fi_err_sens           ;0 00
          BT MES,fi_err_sens           ;0 01
err_01X_sens
          BF MES,fi_err_sens           ;0 10
          BT MES,test_canal_sens       ;0 11

err_TX_temp1
          SET1 S:GEN_ERRORS.0
          BR fi_err_sens
err_TX_temp2
          SET1 S:GEN_ERRORS.0
          SET1 S:GEN_ERRORS.1
          BR fi_err_sens
test_canal_sens
          SET1 MODE_TEST
          BR fi_err_sens
fi_err_sens

          RET


;-------------------------------------------------------------------------------------------------------------------
SELEC_ERRORS_MAST       ;(master)
;-------------------------------------------------------------------------------------------------------------------
;segons l'estat dels polsadors just després de fer el reset de la unitat màster
;modifica adienmet la variable GEN_ERRORS per a simular l'efecte de la recepció
;incorrecta de missatges per part del sensor
;també possibilita entrar en el mode de test del màster

          MOV S:GEN_ERRORS,#00000000b
          CLR1 MODE_TEST               ;inicialitzem MODE_TEST a 0

          BT MODE,errors_sincro
          BF MODE,altres_errors_mast

errors_sincro
          BF MENYS,err_0X_sinc
          BT MENYS,err_1X_sinc
```

```
err_0X_sinc
          BF MES,err_sincro1           ;1 00
          BT MES,err_sincro2           ;1 01
err_1X_sinc
          BF MES,err_sincro3           ;1 10
          BT MES,fi_err_mast           ;1 11


altres_errors_mast
          BF MENYS,err_0X_altres
          BT MENYS,err_1X_altres
err_0X_altres
          BF MES,fi_err_mast           ;0 00
          BT MES,err_conf              ;0 01
err_1X_altres
          BF MES,err_sinc_mes          ;0 10
          BT MES,test_canal_mast       ;0 11

err_sincro1
          SET1 S:GEN_ERRORS.0
          BR fi_err_mast
err_sincro2
          SET1 S:GEN_ERRORS.0
          SET1 S:GEN_ERRORS.1
          BR fi_err_mast
err_sincro3
          SET1 S:GEN_ERRORS.0
          SET1 S:GEN_ERRORS.1
          SET1 S:GEN_ERRORS.2
          BR fi_err_mast

err_sinc_mes
          SET1 S:GEN_ERRORS.3
          BR fi_err_mast
err_conf
          SET1 S:GEN_ERRORS.4
          BR fi_err_mast
test_canal_mast
          SET1 MODE_TEST
          BR fi_err_mast
fi_err_mast

          RET



;*****************************************************************************************************************
;* RUTINES VÀRIES                                                                                                *
;*****************************************************************************************************************

;-----------------------------------------------------------------------------------------------------------------
mode_test_sens
;-----------------------------------------------------------------------------------------------------------------
          CALL ESPERA_5ms

          CALL INICIALIT_LCD

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#M_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#d_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA
```

```
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#n_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_
          CALL WRITE_DATA

          SET1 PWR_UP                ;passa d'stand by al mode d'operacio
          CALL ESPERA_1ms
          SET1 TXEN                  ;es posa en mode d'emissio
          CALL ESPERA_1ms

          MOV ASIM20,#10001000b      ;Tx=ON, Rx=OFF, No parity, 8 bits, 1 stop bit
          CLR1 STMK20                ;habilita interrupcions de final de transmisio UART
          DI                         ;deshabilita el servei d'interrupcio

          MOV TXS20,#55h
          HALT
          CLR1 STIF20

          MOV TXS20,#0FFh
          HALT
          CLR1 STIF20

loop_mode_test_sens
          MOV TXS20,#10101010b
          HALT
          CLR1 STIF20

          BR loop_mode_test_sens


;-----------------------------------------------------------------------------------------------------------------
mode_test_mast
;-----------------------------------------------------------------------------------------------------------------

          CALL ESPERA_5ms

          CALL INICIALIT_LCD

          MOV S:DADA_LCD,#00000001b   ;esborra l'LCD
          CALL WRITE_INSTRUC
          CALL ESPERA_BF

          MOV S:DADA_LCD,#M_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#o_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#d_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#espai_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#m_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#a_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#s_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#t_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#e_
          CALL WRITE_DATA
          MOV S:DADA_LCD,#r_
          CALL WRITE_DATA
```

```
          CLR1 TXEN                   ;passa d'stand by al mode d'operacio,
          SET1 PWR_UP                 ;quan s'estabilitza ja esta en mode recepcio
          CALL ESPERA_3ms

          MOV ASIM20,#01001000b       ;Tx=OFF, Rx=ON, No parity, 8 bits, 1 stop bit
          CLR1 SRMK20                 ;habilita interrupcions de final de recepcio UART
          DI

          MOVW DE,#0000h
          MOVW HL,#0000h

loop_test_mast

          HALT

          MOV A,ASIS20              ;si s'ha detectat un error de transmissio va a error_test_mast
          CMP A,#00000000b
          BZ comprova_byte
          BR error_missatge
comprova_byte
          MOV A,RXB20               ;llegeix el byte rebut
          CLR1 SRIF20               ;!!!!!!! (no se si fa falta)esborra el flag de requeriment d'interrupcio
          CMP A,#10101010b
          BZ missatge_rebut_OK
          BNZ error_missatge

missatge_rebut_OK

          INCW DE
          INCW HL

          MOVW AX,HL
          CMPW AX,#0FFFh
          BC loop_test_mast
          BNC mostra_resultat
          BZ mostra_resultat

error_missatge

          MOV A,RXB20               ;llegeix el byte rebut
          CLR1 SRIF20               ;esborra el flag de requeriment d'interrupcio

          INCW HL

          MOVW AX,HL
          CMPW AX,#0FFFh
          BC loop_test_mast
          BNC mostra_resultat
          BZ mostra_resultat


mostra_resultat
          MOV S:DADA_LCD,#11000000b   ;passa a la línia 2
          CALL WRITE_INSTRUC
          ;CALL ESPERA_BF

          MOV A,D
          CALL MOSTRA_BYTE

          MOV A,E
          CALL MOSTRA_BYTE

          MOVW DE,#0000h
          MOVW HL,#0000h

          BR loop_test_mast
```

```
;-------------------------------------------------------------------------------------------------------------
;---rutines vectors d'interrupció-------------------------------------------------------------------------------

resetvect
          BR  config          ;rutina vector d'interrupció RESET

intp2vect
          NOP                 ;rutina vector d'interrupció INTP2
          RETI

intsrvect
          BR RX_BYTE_OK       ;rutina vector d'interrupcio de finalitzacio de recepcio UART


intstvect
          NOP                 ;rutina vector d'interrupcio de finalitzacio de transmissio UART
          RETI

ti90vect
          NOP                 ;rutina vector d'interrupció del TIMER90
          RETI

ti80vect
          BR FI_COMP_13ms     ;rutina vector d'interrupció TIMER80


;-------------------------------------------------------------------------------------------------------------


          endmod
          end
```

G.- Catàlegs de components

En aquest annex s'inclouen els *datasheets* d'alguns dels components utilitzats en el projecte. En concret s'hi inclouen els següents *datassheets*:

- Sensor de temperatura DS18B20
- Transceiver nRF401
- Mòdul de radiofreqüència nRF401-LOOPKIT
- Pantalla LCD PC1602-F

G.1.- Datasheet del sensor de temperatura DS18B20

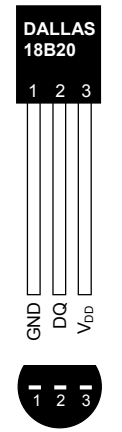
DALLAS SEMICONDUCTOR MAXIM

www.maxim-ic.com

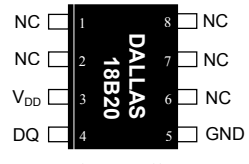
DS18B20
Programmable Resolution 1-Wire Digital Thermometer

FEATURES
- Unique 1-Wire® interface requires only one port pin for communication
- Each device has a unique 64-bit serial code stored in an onboard ROM
- Multidrop capability simplifies distributed temperature sensing applications
- Requires no external components
- Can be powered from data line. Power supply range is 3.0V to 5.5V
- Measures temperatures from –55°C to +125°C (–67°F to +257°F)
- ±0.5°C accuracy from –10°C to +85°C
- Thermometer resolution is user-selectable from 9 to 12 bits
- Converts temperature to 12-bit digital word in 750ms (max.)
- User-definable nonvolatile (NV) alarm settings
- Alarm search command identifies and addresses devices whose temperature is outside of programmed limits (temperature alarm condition)
- Available in 8-pin SO (150mil), 8-pin µSOP, and 3-pin TO-92 packages
- Software compatible with the DS1822
- Applications include thermostatic controls, industrial systems, consumer products, thermometers, or any thermally sensitive system

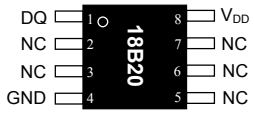
PIN ASSIGNMENT



TO-92
(DS18B20)



8-Pin 150mil SO
(DS18B20Z)



8-Pin µSOP
(DS18B20U)

PIN DESCRIPTION
GND - Ground
DQ - Data In/Out
V_{DD} - Power Supply Voltage
NC - No Connect

DESCRIPTION
The DS18B20 Digital Thermometer provides 9 to 12–bit centigrade temperature measurements and has an alarm function with nonvolatile user-programmable upper and lower trigger points. The DS18B20 communicates over a 1-Wire bus that by definition requires only one data line (and ground) for communication with a central microprocessor. It has an operating temperature range of –55°C to +125°C and is accurate to ±0.5°C over the range of –10°C to +85°C. In addition, the DS18B20 can derive power directly from the data line ("parasite power"), eliminating the need for an external power supply.

Each DS18B20 has a unique 64-bit serial code, which allows multiple DS18B20s to function on the same 1–wire bus; thus, it is simple to use one microprocessor to control many DS18B20s distributed over a large area. Applications that can benefit from this feature include HVAC environmental controls, temperature monitoring systems inside buildings, equipment or machinery, and process monitoring and control systems.

DETAILED PIN DESCRIPTIONS Table 1

SO*	µSOP*	TO-92	SYMBOL	DESCRIPTION
5	4	1	GND	**Ground.**
4	1	2	DQ	**Data Input/Output pin.** Open-drain 1-Wire interface pin. Also provides power to the device when used in parasite power mode (see "Parasite Power" section.)
3	8	3	V_{DD}	**Optional V_{DD} pin.** V_{DD} must be grounded for operation in parasite power mode.

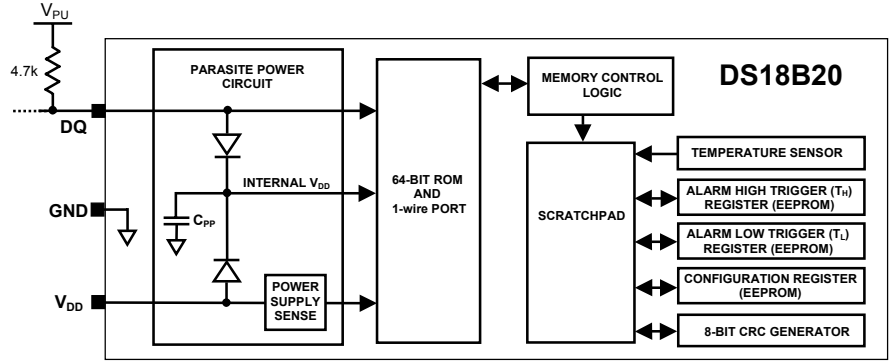
*All pins not specified in this table are "No Connect" pins.

OVERVIEW
Figure 1 shows a block diagram of the DS18B20, and pin descriptions are given in Table 1. The 64-bit ROM stores the device's unique serial code. The scratchpad memory contains the 2-byte temperature register that stores the digital output from the temperature sensor. In addition, the scratchpad provides access to the 1-byte upper and lower alarm trigger registers (T_H and T_L), and the 1-byte configuration register. The configuration register allows the user to set the resolution of the temperature-to-digital conversion to 9, 10, 11, or 12 bits. The T_H, T_L and configuration registers are nonvolatile (EEPROM), so they will retain data when the device is powered down.

The DS18B20 uses Dallas' exclusive 1-Wire bus protocol that implements bus communication using one control signal. The control line requires a weak pullup resistor since all devices are linked to the bus via a 3-state or open-drain port (the DQ pin in the case of the DS18B20). In this bus system, the microprocessor (the master device) identifies and addresses devices on the bus using each device's unique 64-bit code. Because each device has a unique code, the number of devices that can be addressed on one bus is virtually unlimited. The 1-Wire bus protocol, including detailed explanations of the commands and "time slots," is covered in the *1-WIRE BUS SYSTEM* section of this datasheet.

Another feature of the DS18B20 is the ability to operate without an external power supply. Power is instead supplied through the 1-Wire pullup resistor via the DQ pin when the bus is high. The high bus signal also charges an internal capacitor (C_{PP}), which then supplies power to the device when the bus is low. This method of deriving power from the 1-Wire bus is referred to as "parasite power." As an alternative, the DS18B20 may also be powered by an external supply on V_{DD}.

DS18B20 BLOCK DIAGRAM Figure 1



OPERATION — MEASURING TEMPERATURE

The core functionality of the DS18B20 is its direct-to-digital temperature sensor. The resolution of the temperature sensor is user-configurable to 9, 10, 11, or 12 bits, corresponding to increments of 0.5°C, 0.25°C, 0.125°C, and 0.0625°C, respectively. The default resolution at power-up is 12-bit. The DS18B20 powers-up in a low-power idle state; to initiate a temperature measurement and A-to-D conversion, the master must issue a Convert T [44h] command. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its idle state. If the DS18B20 is powered by an external supply, the master can issue "read time slots" (see the *1-WIRE BUS SYSTEM* section) after the Convert T command and the DS18B20 will respond by transmitting 0 while the temperature conversion is in progress and 1 when the conversion is done. If the DS18B20 is powered with parasite power, this notification technique cannot be used since the bus must be pulled high by a strong pullup during the entire temperature conversion. The bus requirements for parasite power are explained in detail in the *POWERING THE DS18B20* section of this datasheet.

The DS18B20 output temperature data is calibrated in degrees centigrade; for Fahrenheit applications, a lookup table or conversion routine must be used. The temperature data is stored as a 16-bit sign-extended two's complement number in the temperature register (see Figure 2). The sign bits (S) indicate if the temperature is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. If the DS18B20 is configured for 12-bit resolution, all bits in the temperature register will contain valid data. For 11-bit resolution, bit 0 is undefined. For 10-bit resolution, bits 1 and 0 are undefined, and for 9-bit resolution bits 2, 1 and 0 are undefined. Table 2 gives examples of digital output data and the corresponding temperature reading for 12-bit resolution conversions.

TEMPERATURE REGISTER FORMAT Figure 2

	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
LS Byte	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}

	bit 15	bit 14	bit 13	bit 12	bit 11	bit 10	bit 9	bit 8
MS Byte	S	S	S	S	S	2^6	2^5	2^4

TEMPERATURE/DATA RELATIONSHIP Table 2

TEMPERATURE	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+125°C	0000 0111 1101 0000	07D0h
+85°C*	0000 0101 0101 0000	0550h
+25.0625°C	0000 0001 1001 0001	0191h
+10.125°C	0000 0000 1010 0010	00A2h
+0.5°C	0000 0000 0000 1000	0008h
0°C	0000 0000 0000 0000	0000h
-0.5°C	1111 1111 1111 1000	FFF8h
-10.125°C	1111 1111 0101 1110	FF5Eh
-25.0625°C	1111 1110 0110 1111	FE6Fh
-55°C	1111 1100 1001 0000	FC90h

*The power-on reset value of the temperature register is +85°C

OPERATION — ALARM SIGNALING

After the DS18B20 performs a temperature conversion, the temperature value is compared to the user-defined two's complement alarm trigger values stored in the 1-byte T_H and T_L registers (see Figure 3). The sign bit (S) indicates if the value is positive or negative: for positive numbers S = 0 and for negative numbers S = 1. The T_H and T_L registers are nonvolatile (EEPROM) so they will retain data when the device is powered down. T_H and T_L can be accessed through bytes 2 and 3 of the scratchpad as explained in the *MEMORY* section of this datasheet.

T_H AND T_L REGISTER FORMAT Figure 3

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
S	2^6	2^5	2^5	2^5	2^2	2^1	2^0

Only bits 11 through 4 of the temperature register are used in the T_H and T_L comparison since T_H and T_L are 8-bit registers. If the measured temperature is lower than or equal to T_L or higher than T_H, an alarm condition exists and an alarm flag is set inside the DS18B20. This flag is updated after every temperature measurement; therefore, if the alarm condition goes away, the flag will be turned off after the next temperature conversion.

The master device can check the alarm flag status of all DS18B20s on the bus by issuing an Alarm Search [ECh] command. Any DS18B20s with a set alarm flag will respond to the command, so the master can determine exactly which DS18B20s have experienced an alarm condition. If an alarm condition exists and the T_H or T_L settings have changed, another temperature conversion should be done to validate the alarm condition.

POWERING THE DS18B20

The DS18B20 can be powered by an external supply on the V_{DD} pin, or it can operate in "parasite power" mode, which allows the DS18B20 to function without a local external supply. Parasite power is very useful for applications that require remote temperature sensing or that are very space constrained. Figure 1 shows the DS18B20's parasite-power control circuitry, which "steals" power from the 1-Wire bus via the DQ pin when the bus is high. The stolen charge powers the DS18B20 while the bus is high, and some of the charge is stored on the parasite power capacitor (C_{PP}) to provide power when the bus is low. When the DS18B20 is used in parasite power mode, the V_{DD} pin must be connected to ground.

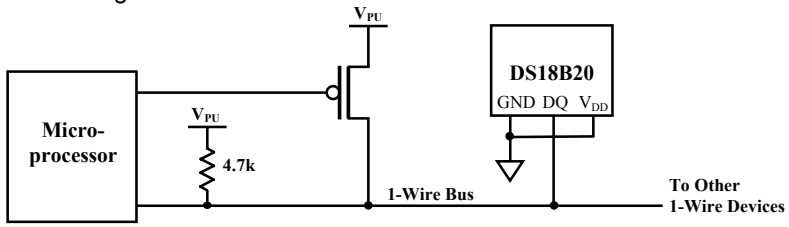
In parasite power mode, the 1-Wire bus and C_{PP} can provide sufficient current to the DS18B20 for most operations as long as the specified timing and voltage requirements are met (refer to *the DC ELECTRICAL CHARACTERISTICS* and the *AC ELECTRICAL CHARACTERISTICS* sections of this data sheet). However, when the DS18B20 is performing temperature conversions or copying data from the scratchpad memory to EEPROM, the operating current can be as high as 1.5mA. This current can cause an unacceptable voltage drop across the weak 1-Wire pullup resistor and is more current than can be supplied by C_{PP}. To assure that the DS18B20 has sufficient supply current, it is necessary to provide a strong pullup on the 1-Wire bus whenever temperature conversions are taking place or data is being copied from the scratchpad to EEPROM. This can be accomplished by using a MOSFET to pull the bus directly to the rail as shown in Figure 4. The 1-Wire bus must be switched to the strong pullup within 10µs (max) after a Convert T [44h] or Copy Scratchpad [48h] command is issued, and the bus must be held high by the pullup for the duration of the conversion (t_{conv}) or data transfer (t_{wr} = 10ms). No other activity can take place on the 1-Wire bus while the pullup is enabled.

The DS18B20 can also be powered by the conventional method of connecting an external power supply to the V_{DD} pin, as shown in Figure 5. The advantage of this method is that the MOSFET pullup is not required, and the 1-Wire bus is free to carry other traffic during the temperature conversion time.

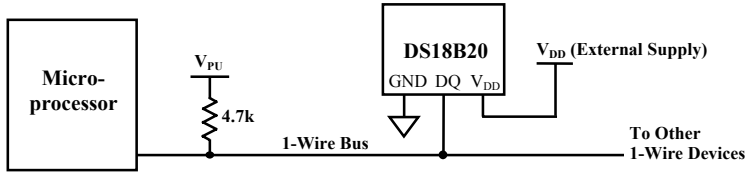
The use of parasite power is not recommended for temperatures above +100°C since the DS18B20 may not be able to sustain communications due to the higher leakage currents that can exist at these temperatures. For applications in which such temperatures are likely, it is strongly recommended that the DS18B20 be powered by an external power supply.

In some situations the bus master may not know whether the DS18B20s on the bus are parasite powered or powered by external supplies. The master needs this information to determine if the strong bus pullup should be used during temperature conversions. To get this information, the master can issue a Skip ROM [CCh] command followed by a Read Power Supply [B4h] command followed by a "read time slot". During the read time slot, parasite powered DS18B20s will pull the bus low, and externally powered DS18B20s will let the bus remain high. If the bus is pulled low, the master knows that it must supply the strong pullup on the 1-Wire bus during temperature conversions.

SUPPLYING THE PARASITE-POWERED DS18B20 DURING TEMPERATURE CONVERSIONS Figure 4



POWERING THE DS18B20 WITH AN EXTERNAL SUPPLY Figure 5



64-BIT LASERED ROM CODE

Each DS18B20 contains a unique 64–bit code (see Figure 6) stored in ROM. The least significant 8 bits of the ROM code contain the DS18B20's 1-Wire family code: 28h. The next 48 bits contain a unique serial number. The most significant 8 bits contain a cyclic redundancy check (CRC) byte that is calculated from the first 56 bits of the ROM code. A detailed explanation of the CRC bits is provided in the *CRC GENERATION* section. The 64-bit ROM code and associated ROM function control logic allow the DS18B20 to operate as a 1-Wire device using the protocol detailed in the *1-WIRE BUS SYSTEM* section of this datasheet.

64-BIT LASERED ROM CODE Figure 6

8-BIT CRC		48-BIT SERIAL NUMBER		8-BIT FAMILY CODE (28h)	
MSB	LSB	MSB	LSB	MSB	LSB

MEMORY

The DS18B20's memory is organized as shown in Figure 7. The memory consists of an SRAM scratchpad with nonvolatile EEPROM storage for the high and low alarm trigger registers (T_H and T_L) and configuration register. Note that if the DS18B20 alarm function is not used, the T_H and T_L registers can serve as general-purpose memory. All memory commands are described in detail in the *DS18B20 FUNCTION COMMANDS* section.

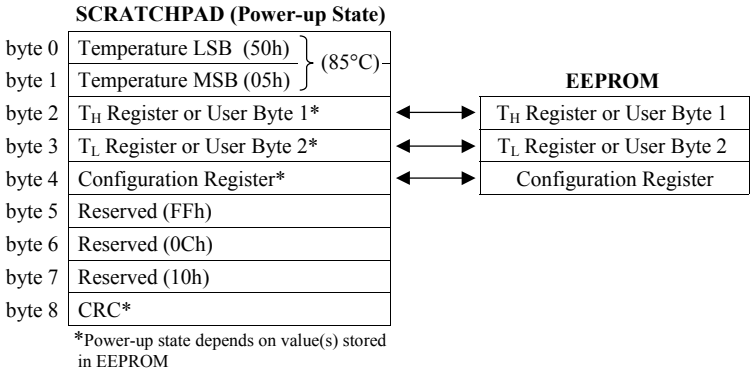
Byte 0 and byte 1 of the scratchpad contain the LSB and the MSB of the temperature register, respectively. These bytes are read-only. Bytes 2 and 3 provide access to T_H and T_L registers. Byte 4 contains the configuration register data, which is explained in detail in the CONFIGURATION REGISTER section of this datasheet. Bytes 5, 6, and 7 are reserved for internal use by the device and cannot be overwritten; these bytes will return all 1s when read.

Byte 8 of the scratchpad is read-only and contains the cyclic redundancy check (CRC) code for bytes 0 through 7 of the scratchpad. The DS18B20 generates this CRC using the method described in *the CRC GENERATION* section.

Data is written to bytes 2, 3, and 4 of the scratchpad using the Write Scratchpad [4Eh] command; the data must be transmitted to the DS18B20 starting with the least significant bit of byte 2. To verify data integrity, the scratchpad can be read (using the Read Scratchpad [BEh] command) after the data is written. When reading the scratchpad, data is transferred over the 1-Wire bus starting with the least significant bit of byte 0. To transfer the T_H, T_L and configuration data from the scratchpad to EEPROM, the master must issue the Copy Scratchpad [48h] command.

Data in the EEPROM registers is retained when the device is powered down; at power-up the EEPROM data is reloaded into the corresponding scratchpad locations. Data can also be reloaded from EEPROM to the scratchpad at any time using the Recall E^2 [B8h] command. The master can issue read time slots following the Recall E^2 command and the DS18B20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done.

DS18B20 MEMORY MAP Figure 7



*Power-up state depends on value(s) stored in EEPROM

CONFIGURATION REGISTER

Byte 4 of the scratchpad memory contains the configuration register, which is organized as illustrated in Figure 8. The user can set the conversion resolution of the DS18B20 using the R0 and R1 bits in this register as shown in Table 3. The power-up default of these bits is R0 = 1 and R1 = 1 (12-bit resolution). Note that there is a direct tradeoff between resolution and conversion time. Bit 7 and bits 0 to 4 in the configuration register are reserved for internal use by the device and cannot be overwritten; these bits will return 1s when read.

CONFIGURATION REGISTER Figure 8

bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
0	R1	R0	1	1	1	1	1

THERMOMETER RESOLUTION CONFIGURATION Table 3

R1	R0	Resolution	Max Conversion Time	
0	0	9-bit	93.75 ms	($t_{CONV}/8$)
0	1	10-bit	187.5 ms	($t_{CONV}/4$)
1	0	11-bit	375 ms	($t_{CONV}/2$)
1	1	12-bit	750 ms	(t_{CONV})

CRC GENERATION

CRC bytes are provided as part of the DS18B20's 64-bit ROM code and in the 9th byte of the scratchpad memory. The ROM code CRC is calculated from the first 56 bits of the ROM code and is contained in the most significant byte of the ROM. The scratchpad CRC is calculated from the data stored in the scratchpad, and therefore it changes when the data in the scratchpad changes. The CRCs provide the bus master with a method of data validation when data is read from the DS18B20. To verify that data has been read correctly, the bus master must re-calculate the CRC from the received data and then compare this value to either the ROM code CRC (for ROM reads) or to the scratchpad CRC (for scratchpad reads). If the calculated CRC matches the read CRC, the data has been received error free. The comparison of CRC values and the decision to continue with an operation are determined entirely by the bus master. There is no circuitry inside the DS18B20 that prevents a command sequence from proceeding if the DS18B20 CRC (ROM or scratchpad) does not match the value generated by the bus master.

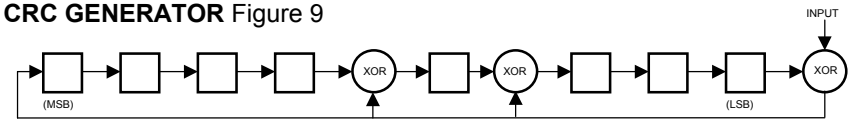
The equivalent polynomial function of the CRC (ROM or scratchpad) is:

$$CRC = X^8 + X^5 + X^4 + 1$$

The bus master can re-calculate the CRC and compare it to the CRC values from the DS18B20 using the polynomial generator shown in Figure 9. This circuit consists of a shift register and XOR gates, and the shift register bits are initialized to 0. Starting with the least significant bit of the ROM code or the least significant bit of byte 0 in the scratchpad, one bit at a time should shifted into the shift register. After shifting in the 56th bit from the ROM or the most significant bit of byte 7 from the scratchpad, the polynomial generator will contain the re-calculated CRC. Next, the 8-bit ROM code or scratchpad CRC from the DS18B20 must be shifted into the circuit. At this point, if the re-calculated CRC was correct, the shift register will contain all 0s. Additional information about the Dallas 1-Wire cyclic redundancy check

is available in *Application Note 27: Understanding and Using Cyclic Redundancy Checks with Dallas Semiconductor Touch Memory Products*.

CRC GENERATOR Figure 9



1-WIRE BUS SYSTEM

The 1-Wire bus system uses a single bus master to control one or more slave devices. The DS18B20 is always a slave. When there is only one slave on the bus, the system is referred to as a "single-drop" system; the system is "multidrop" if there are multiple slaves on the bus.

All data and commands are transmitted least significant bit first over the 1-Wire bus.

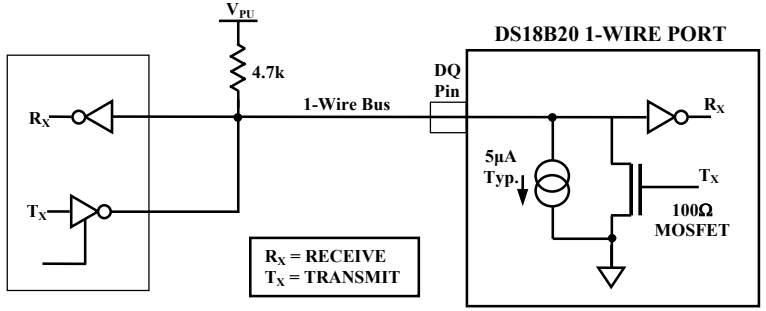
The following discussion of the 1-Wire bus system is broken down into three topics: hardware configuration, transaction sequence, and 1-Wire signaling (signal types and timing).

HARDWARE CONFIGURATION

The 1-Wire bus has by definition only a single data line. Each device (master or slave) interfaces to the data line via an open-drain or 3-state port. This allows each device to "release" the data line when the device is not transmitting data so the bus is available for use by another device. The 1-Wire port of the DS18B20 (the DQ pin) is open drain with an internal circuit equivalent to that shown in Figure 10.

The 1-Wire bus requires an external pullup resistor of approximately 5kΩ; thus, the idle state for the 1-Wire bus is high. If for any reason a transaction needs to be suspended, the bus MUST be left in the idle state if the transaction is to resume. Infinite recovery time can occur between bits so long as the 1-Wire bus is in the inactive (high) state during the recovery period. If the bus is held low for more than 480µs, all components on the bus will be reset.

HARDWARE CONFIGURATION Figure 10



TRANSACTION SEQUENCE

The transaction sequence for accessing the DS18B20 is as follows:

Step 1. Initialization

Step 2. ROM Command (followed by any required data exchange)

Step 3. DS18B20 Function Command (followed by any required data exchange)

It is very important to follow this sequence every time the DS18B20 is accessed, as the DS18B20 will not respond if any steps in the sequence are missing or out of order. Exceptions to this rule are the Search ROM [F0h] and Alarm Search [ECh] commands. After issuing either of these ROM commands, the master must return to Step 1 in the sequence.

INITIALIZATION

All transactions on the 1-Wire bus begin with an initialization sequence. The initialization sequence consists of a reset pulse transmitted by the bus master followed by presence pulse(s) transmitted by the slave(s). The presence pulse lets the bus master know that slave devices (such as the DS18B20) are on the bus and are ready to operate. Timing for the reset and presence pulses is detailed in the *1-WIRE SIGNALING* section.

ROM COMMANDS

After the bus master has detected a presence pulse, it can issue a ROM command. These commands operate on the unique 64-bit ROM codes of each slave device and allow the master to single out a specific device if many are present on the 1-Wire bus. These commands also allow the master to determine how many and what types of devices are present on the bus or if any device has experienced an alarm condition. There are five ROM commands, and each command is 8 bits long. The master device must issue an appropriate ROM command before issuing a DS18B20 function command. A flowchart for operation of the ROM commands is shown in Figure 11.

SEARCH ROM [F0h]

When a system is initially powered up, the master must identify the ROM codes of all slave devices on the bus, which allows the master to determine the number of slaves and their device types. The master learns the ROM codes through a process of elimination that requires the master to perform a Search ROM cycle (i.e., Search ROM command followed by data exchange) as many times as necessary to identify all of the slave devices. If there is only one slave on the bus, the simpler Read ROM command (see below) can be used in place of the Search ROM process. For a detailed explanation of the Search ROM procedure, refer to the *iButton® Book of Standards* at www.ibutton.com/ibuttons/standard.pdf. After every Search ROM cycle, the bus master must return to Step 1 (Initialization) in the transaction sequence.

READ ROM [33h]

This command can only be used when there is one slave on the bus. It allows the bus master to read the slave's 64-bit ROM code without using the Search ROM procedure. If this command is used when there is more than one slave present on the bus, a data collision will occur when all the slaves attempt to respond at the same time.

MATCH ROM [55h]

The match ROM command followed by a 64-bit ROM code sequence allows the bus master to address a specific slave device on a multidrop or single-drop bus. Only the slave that exactly matches the 64-bit ROM code sequence will respond to the function command issued by the master; all other slaves on the bus will wait for a reset pulse.

iButton is a registered trademark of Dallas Semiconductor.

SKIP ROM [CCh]

The master can use this command to address all devices on the bus simultaneously without sending out any ROM code information. For example, the master can make all DS18B20s on the bus perform simultaneous temperature conversions by issuing a Skip ROM command followed by a Convert T [44h] command.

Note that the Read Scratchpad [BEh] command can follow the Skip ROM command only if there is a single slave device on the bus. In this case time is saved by allowing the master to read from the slave without sending the device's 64-bit ROM code. A Skip ROM command followed by a Read Scratchpad command will cause a data collision on the bus if there is more than one slave since multiple devices will attempt to transmit data simultaneously.

ALARM SEARCH [ECh]

The operation of this command is identical to the operation of the Search ROM command except that only slaves with a set alarm flag will respond. This command allows the master device to determine if any DS18B20s experienced an alarm condition during the most recent temperature conversion. After every Alarm Search cycle (i.e., Alarm Search command followed by data exchange), the bus master must return to Step 1 (Initialization) in the transaction sequence. Refer to the *OPERATION — ALARM SIGNALING* section for an explanation of alarm flag operation.

DS18B20 FUNCTION COMMANDS

After the bus master has used a ROM command to address the DS18B20 with which it wishes to communicate, the master can issue one of the DS18B20 function commands. These commands allow the master to write to and read from the DS18B20's scratchpad memory, initiate temperature conversions and determine the power supply mode. The DS18B20 function commands, which are described below, are summarized in Table 4 and illustrated by the flowchart in Figure 12.

CONVERT T [44h]

This command initiates a single temperature conversion. Following the conversion, the resulting thermal data is stored in the 2-byte temperature register in the scratchpad memory and the DS18B20 returns to its low-power idle state. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for the duration of the conversion (t_{conv}) as described in the *POWERING THE DS18B20* section. If the DS18B20 is powered by an external supply, the master can issue read time slots after the Convert T command and the DS18B20 will respond by transmitting a 0 while the temperature conversion is in progress and a 1 when the conversion is done. In parasite power mode this notification technique cannot be used since the bus is pulled high by the strong pullup during the conversion.

WRITE SCRATCHPAD [4Eh]

This command allows the master to write 3 bytes of data to the DS18B20's scratchpad. The first data byte is written into the T_H register (byte 2 of the scratchpad), the second byte is written into the T_L register (byte 3), and the third byte is written into the configuration register (byte 4). Data must be transmitted least significant bit first. All three bytes MUST be written before the master issues a reset, or the data may be corrupted.

READ SCRATCHPAD [BEh]

This command allows the master to read the contents of the scratchpad. The data transfer starts with the least significant bit of byte 0 and continues through the scratchpad until the 9th byte (byte 8 – CRC) is read. The master may issue a reset to terminate reading at any time if only part of the scratchpad data is needed.

COPY SCRATCHPAD [48h]

This command copies the contents of the scratchpad T_H, T_L and configuration registers (bytes 2, 3 and 4) to EEPROM. If the device is being used in parasite power mode, within 10µs (max) after this command is issued the master must enable a strong pullup on the 1-Wire bus for at least 10ms as described in the *POWERING THE DS18B20* section.

RECALL E^2 [B8h]

This command recalls the alarm trigger values (T_H and T_L) and configuration data from EEPROM and places the data in bytes 2, 3, and 4, respectively, in the scratchpad memory. The master device can issue read time slots following the Recall E^2 command and the DS18B20 will indicate the status of the recall by transmitting 0 while the recall is in progress and 1 when the recall is done. The recall operation happens automatically at power-up, so valid data is available in the scratchpad as soon as power is applied to the device.

READ POWER SUPPLY [B4h]

The master device issues this command followed by a read time slot to determine if any DS18B20s on the bus are using parasite power. During the read time slot, parasite powered DS18B20s will pull the bus low, and externally powered DS18B20s will let the bus remain high. Refer to the *POWERING THE DS18B20* section for usage information for this command.

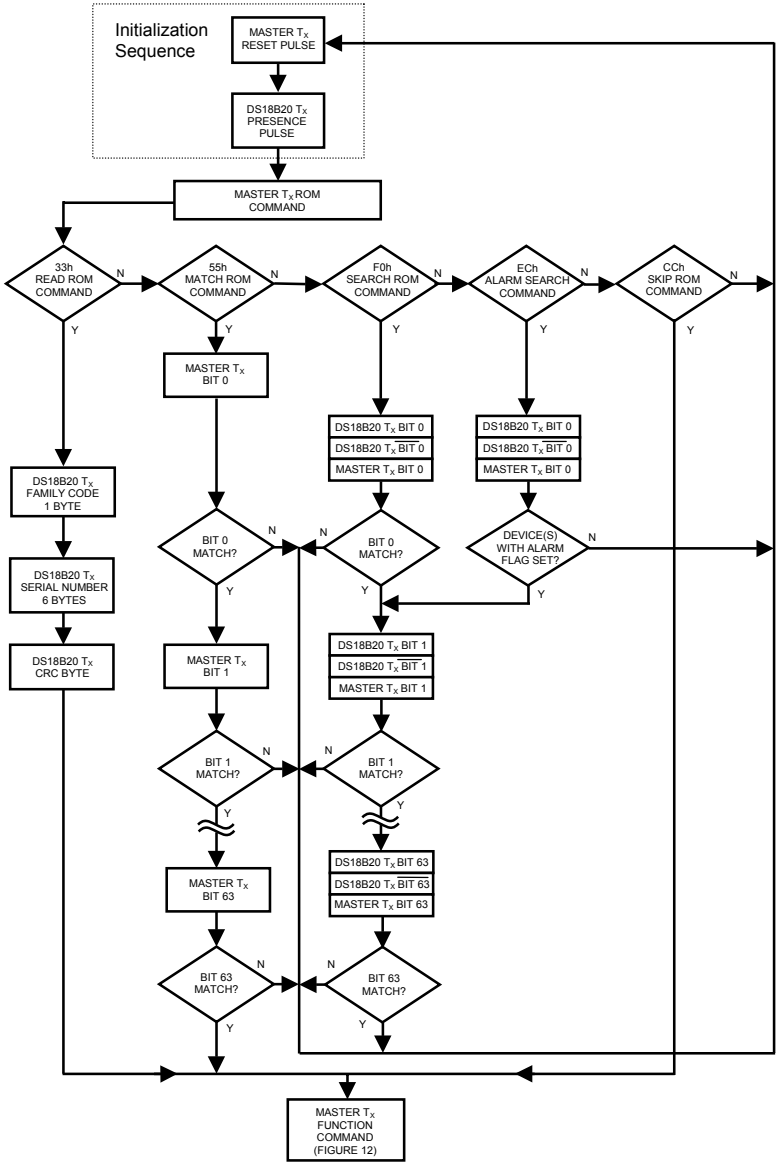
DS18B20 FUNCTION COMMAND SET Table 4

Command	Description	Protocol	1-Wire Bus Activity After Command is Issued	Notes
TEMPERATURE CONVERSION COMMANDS				
Convert T	Initiates temperature conversion.	44h	DS18B20 transmits conversion status to master (not applicable for parasite-powered DS18B20s).	1
MEMORY COMMANDS				
Read Scratchpad	Reads the entire scratchpad including the CRC byte.	BEh	DS18B20 transmits up to 9 data bytes to master.	2
Write Scratchpad	Writes data into scratchpad bytes 2, 3, and 4 (T_H, T_L, and configuration registers).	4Eh	Master transmits 3 data bytes to DS18B20.	3
Copy Scratchpad	Copies T_H, T_L, and configuration register data from the scratchpad to EEPROM.	48h	None	1
Recall E^2	Recalls T_H, T_L, and configuration register data from EEPROM to the scratchpad.	B8h	DS18B20 transmits recall status to master.	
Read Power Supply	Signals DS18B20 power supply mode to the master.	B4h	DS18B20 transmits supply status to master.	

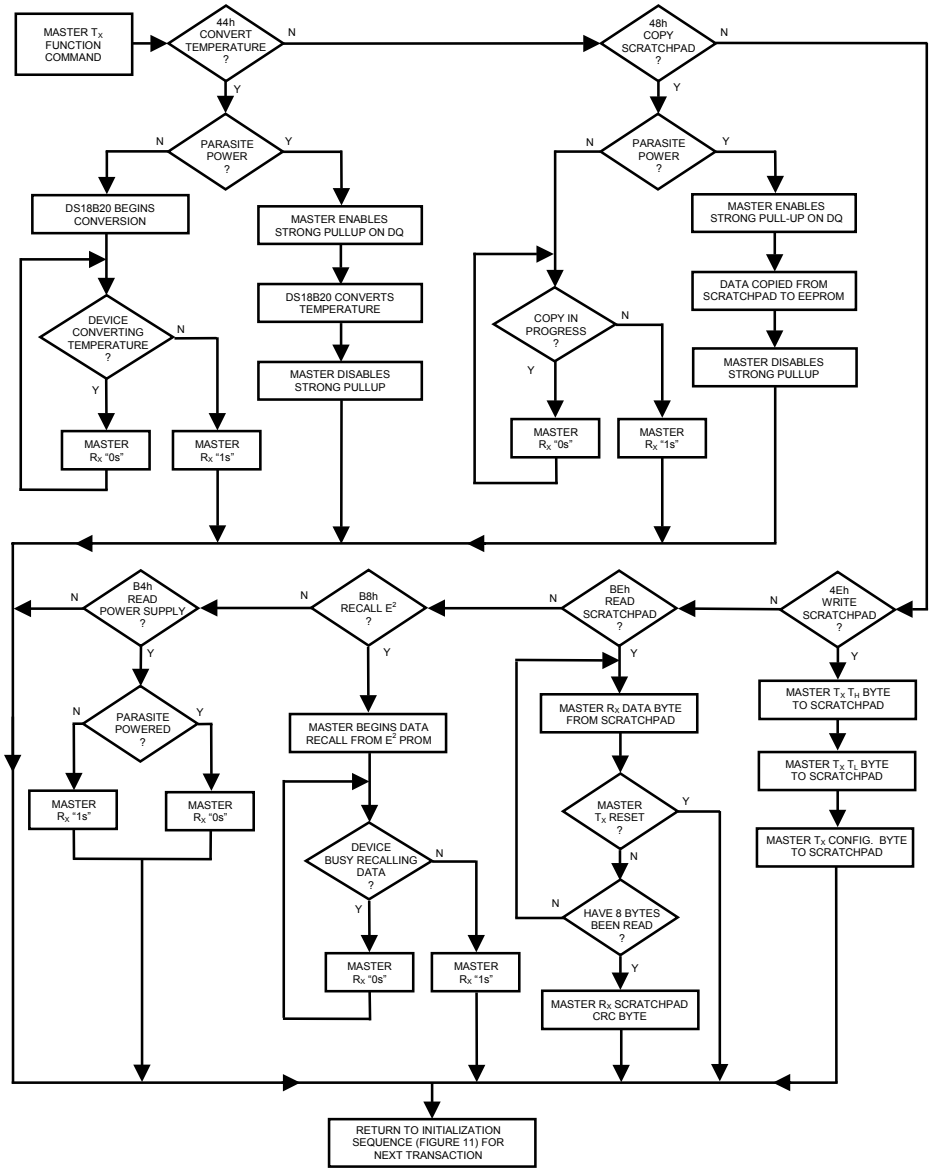
NOTES:

1) For parasite-powered DS18B20s, the master must enable a strong pullup on the 1-Wire bus during temperature conversions and copies from the scratchpad to EEPROM. No other bus activity may take place during this time.
2) The master can interrupt the transmission of data at any time by issuing a reset.
3) All three bytes must be written before a reset is issued.

ROM COMMANDS FLOW CHART Figure 11



DS18B20 FUNCTION COMMANDS FLOW CHART Figure 12



1-WIRE SIGNALING

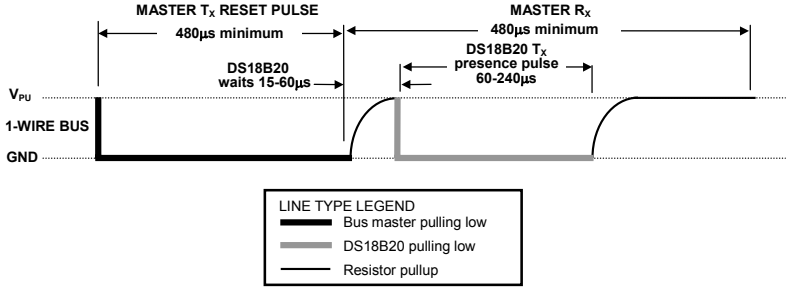
The DS18B20 uses a strict 1-Wire communication protocol to insure data integrity. Several signal types are defined by this protocol: reset pulse, presence pulse, write 0, write 1, read 0, and read 1. The bus master initiates all of these signals, with the exception of the presence pulse.

INITIALIZATION PROCEDURE: RESET AND PRESENCE PULSES

All communication with the DS18B20 begins with an initialization sequence that consists of a reset pulse from the master followed by a presence pulse from the DS18B20. This is illustrated in Figure 13. When the DS18B20 sends the presence pulse in response to the reset, it is indicating to the master that it is on the bus and ready to operate.

During the initialization sequence the bus master transmits (T_X) the reset pulse by pulling the 1-Wire bus low for a minimum of 480µs. The bus master then releases the bus and goes into receive mode (R_X). When the bus is released, the 5k pullup resistor pulls the 1-Wire bus high. When the DS18B20 detects this rising edge, it waits 15µs to 60µs and then transmits a presence pulse by pulling the 1-Wire bus low for 60µs to 240µs.

INITIALIZATION TIMING Figure 13



READ/WRITE TIME SLOTS

The bus master writes data to the DS18B20 during write time slots and reads data from the DS18B20 during read time slots. One bit of data is transmitted over the 1-Wire bus per time slot.

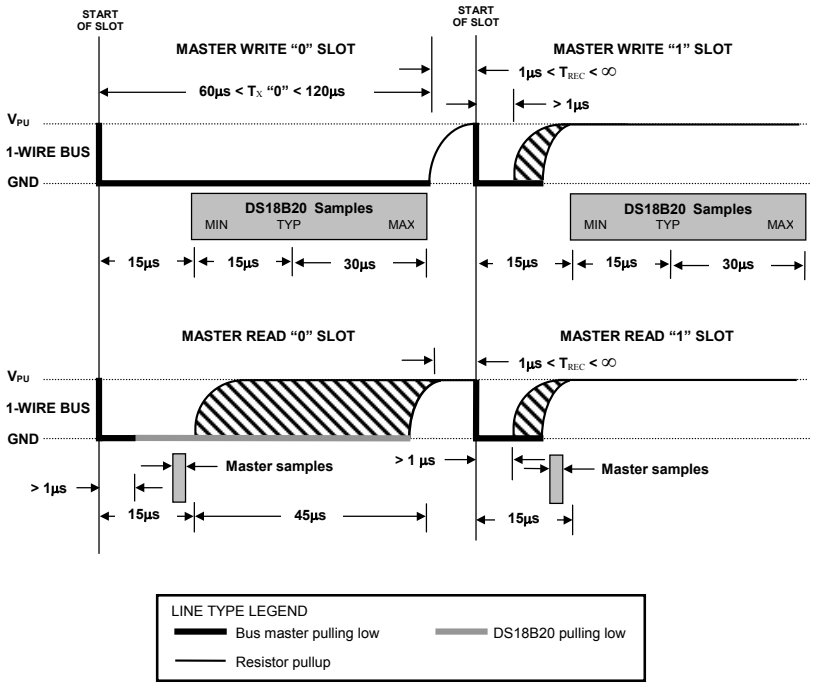
WRITE TIME SLOTS

There are two types of write time slots: "Write 1" time slots and "Write 0" time slots. The bus master uses a Write 1 time slot to write a logic 1 to the DS18B20 and a Write 0 time slot to write a logic 0 to the DS18B20. All write time slots must be a minimum of 60µs in duration with a minimum of a 1µs recovery time between individual write slots. Both types of write time slots are initiated by the master pulling the 1-Wire bus low (see Figure 14).

To generate a Write 1 time slot, after pulling the 1-Wire bus low, the bus master must release the 1-Wire bus within 15µs. When the bus is released, the 5k pullup resistor will pull the bus high. To generate a Write 0 time slot, after pulling the 1-Wire bus low, the bus master must continue to hold the bus low for the duration of the time slot (at least 60µs).

The DS18B20 samples the 1-Wire bus during a window that lasts from 15µs to 60µs after the master initiates the write time slot. If the bus is high during the sampling window, a 1 is written to the DS18B20. If the line is low, a 0 is written to the DS18B20.

READ/WRITE TIME SLOT TIMING DIAGRAM Figure 14



READ TIME SLOTS

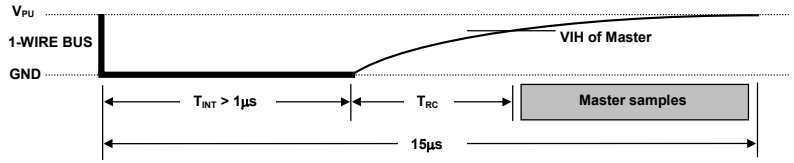
The DS18B20 can only transmit data to the master when the master issues read time slots. Therefore, the master must generate read time slots immediately after issuing a Read Scratchpad [BEh] or Read Power Supply [B4h] command, so that the DS18B20 can provide the requested data. In addition, the master can generate read time slots after issuing Convert T [44h] or Recall E^2 [B8h] commands to find out the status of the operation as explained in the *DS18B20 FUNCTION COMMAND* section.

All read time slots must be a minimum of 60µs in duration with a minimum of a 1µs recovery time between slots. A read time slot is initiated by the master device pulling the 1-Wire bus low for a minimum of 1µs and then releasing the bus (see Figure 14). After the master initiates the read time slot, the DS18B20 will begin transmitting a 1 or 0 on bus. The DS18B20 transmits a 1 by leaving the bus high and transmits a 0 by pulling the bus low. When transmitting a 0, the DS18B20 will release the bus by the end of the time slot, and the bus will be pulled back to its high idle state by the pullup resister. Output

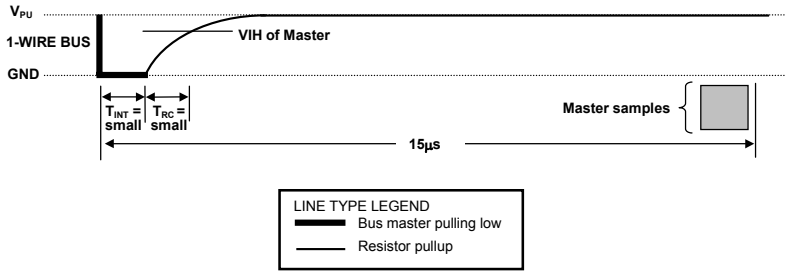
data from the DS18B20 is valid for 15µs after the falling edge that initiated the read time slot. Therefore, the master must release the bus and then sample the bus state within 15µs from the start of the slot.

Figure 15 illustrates that the sum of T_{INIT}, T_{RC}, and T_{SAMPLE} must be less than 15µs for a read time slot. Figure 16 shows that system timing margin is maximized by keeping T_{INIT} and T_{RC} as short as possible and by locating the master sample time during read time slots towards the end of the 15µs period.

DETAILED MASTER READ 1 TIMING Figure 15



RECOMMENDED MASTER READ 1 TIMING Figure 16



RELATED APPLICATION NOTES

The following Application Notes can be applied to the DS18B20. These notes can be obtained from the Dallas Semiconductor "Application Note Book," via the Dallas website at http://www.dalsemi.com/, or through our faxback service at (214) 450–0441.

Application Note 27: Understanding and Using Cyclic Redundancy Checks with Dallas Semiconductor Touch Memory Product
Application Note 55: Extending the Contact Range of Touch Memories
Application Note 74: Reading and Writing Touch Memories via Serial Interfaces
Application Note 104: Minimalist Temperature Control Demo
Application Note 106: Complex MicroLANs
Application Note 108: MicroLAN — In the Long Run
Application Note 162: Interfacing the DS18X20/DS1822 1-Wire Temperature Sensor in a Microcontroller Environment

Sample 1-Wire subroutines that can be used in conjunction with AN74 can be downloaded from the Dallas website or anonymous FTP Site.

DS18B20 OPERATION EXAMPLE 1

In this example there are multiple DS18B20s on the bus and they are using parasite power. The bus master initiates a temperature conversion in a specific DS18B20 and then reads its scratchpad and recalculates the CRC to verify the data.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
TX	Reset	Master issues reset pulse.
RX	Presence	DS18B20s respond with presence pulse.
TX	55h	Master issues Match ROM command.
TX	64-bit ROM code	Master sends DS18B20 ROM code.
TX	44h	Master issues Convert T command.
TX	DQ line held high by strong pullup	Master applies strong pullup to DQ for the duration of the conversion (t_{conv}).
TX	Reset	Master issues reset pulse.
RX	Presence	DS18B20s respond with presence pulse.
TX	55h	Master issues Match ROM command.
TX	64-bit ROM code	Master sends DS18B20 ROM code.
TX	BEh	Master issues Read Scratchpad command.
RX	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.

DS18B20 OPERATION EXAMPLE 2

In this example there is only one DS18B20 on the bus and it is using parasite power. The master writes to the T_H, T_L, and configuration registers in the DS18B20 scratchpad and then reads the scratchpad and recalculates the CRC to verify the data. The master then copies the scratchpad contents to EEPROM.

MASTER MODE	DATA (LSB FIRST)	COMMENTS
TX	Reset	Master issues reset pulse.
RX	Presence	DS18B20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	4Eh	Master issues Write Scratchpad command.
TX	3 data bytes	Master sends three data bytes to scratchpad (T_H, T_L, and config).
TX	Reset	Master issues reset pulse.
RX	Presence	DS18B20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	BEh	Master issues Read Scratchpad command.
RX	9 data bytes	Master reads entire scratchpad including CRC. The master then recalculates the CRC of the first eight data bytes from the scratchpad and compares the calculated CRC with the read CRC (byte 9). If they match, the master continues; if not, the read operation is repeated.
TX	Reset	Master issues reset pulse.
RX	Presence	DS18B20 responds with presence pulse.
TX	CCh	Master issues Skip ROM command.
TX	48h	Master issues Copy Scratchpad command.
TX	DQ line held high by strong pullup	Master applies strong pullup to DQ for at least 10ms while copy operation is in progress.

ABSOLUTE MAXIMUM RATINGS*

Voltage on Any Pin Relative to Ground	-0.5V to +6.0V
Operating Temperature Range	-55°C to +125°C
Storage Temperature Range	-55°C to +125°C
Solder Temperature	See IPC/JEDEC J-STD-020A
Reflow Oven Temperature	+220°C

**These are stress ratings only and functional operation of the device at these or any other conditions above those indicated in the operation sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods of time may affect reliability.*

DC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD}=3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITION	MIN	TYP	MAX	UNITS	NOTES
Supply Voltage	V_{DD}	Local Power	+3.0		+5.5	V	1
Pullup Supply Voltage	V_{PU}	Parasite Power	+3.0		+5.5	V	1,2
		Local Power	+3.0		V_{DD}		
Thermometer Error	t_{ERR}	-10°C to +85°C			±0.5	°C	3
		-55°C to +125°C			±2		
Input Logic Low	V_{IL}		-0.3		+0.8	V	1,4,5
Input Logic High	V_{IH}	Local Power	+2.2		The lower of 5.5 or V_{DD} + 0.3	V	1, 6
		Parasite Power	+3.0				
Sink Current	I_L	$V_{I/O}$=0.4V	4.0			mA	1
Standby Current	I_{DDS}			750	1000	nA	7,8
Active Current	I_{DD}	V_{DD}=5V		1	1.5	mA	9
DQ Input Current	I_{DQ}			5		µA	10
Drift				±0.2		°C	11

NOTES:

1) All voltages are referenced to ground.
2) The Pullup Supply Voltage specification assumes that the pullup device is ideal, and therefore the high level of the pullup is equal to V_{PU}. In order to meet the V_{IH} spec of the DS18B20, the actual supply rail for the strong pullup transistor must include margin for the voltage drop across the transistor when it is turned on; thus: $V_{PU\_ACTUAL} = V_{PU\_IDEAL} + V_{TRANSISTOR}$.
3) See typical performance curve in Figure 17
4) Logic low voltages are specified at a sink current of 4mA.
5) To guarantee a presence pulse under low voltage parasite power conditions, V_{ILMAX} may have to be reduced to as low as 0.5V.
6) Logic high voltages are specified at a source current of 1mA.
7) Standby current specified up to 70°C. Standby current typically is 3µA at 125°C.
8) To minimize I_{DDS}, DQ should be within the following ranges: GND ≤ DQ ≤ GND + 0.3V or V_{DD} – 0.3V ≤ DQ ≤ V_{DD}.
9) Active current refers to supply current during active temperature conversions or EEPROM writes.
10) DQ line is high ("hi-Z" state).
11) Drift data is based on a 1000 hour stress test at 125°C with V_{DD} = 5.5V.

AC ELECTRICAL CHARACTERISTICS: NV MEMORY

(-55°C to +100°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITION	MIN	TYP	MAX	UNITS
NV Write Cycle Time	t_{wr}			2	10	ms
EEPROM Writes	N_{EEWR}	-55°C to +55°C	50k			writes
EEPROM Data Retention	t_{EEDR}	-55°C to +55°C	10			years

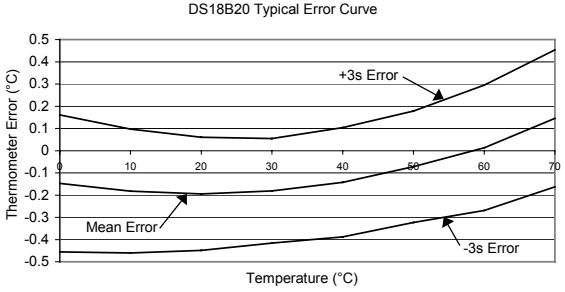
AC ELECTRICAL CHARACTERISTICS (-55°C to +125°C; V_{DD} = 3.0V to 5.5V)

PARAMETER	SYMBOL	CONDITION	MIN	TYP	MAX	UNITS	NOTES
Temperature Conversion Time	t_{CONV}	9-bit resolution			93.75	ms	1
		10-bit resolution			187.5	ms	1
		11-bit resolution			375	ms	1
		12-bit resolution			750	ms	1
Time to Strong Pullup On	t_{SPON}	Start Convert T Command Issued			10	µs	
Time Slot	t_{SLOT}		60		120	µs	1
Recovery Time	t_{REC}		1			µs	1
Write 0 Low Time	r_{LOW0}		60		120	µs	1
Write 1 Low Time	t_{LOW1}		1		15	µs	1
Read Data Valid	t_{RDV}				15	µs	1
Reset Time High	t_{RSTH}		480			µs	1
Reset Time Low	t_{RSTL}		480			µs	1,2
Presence Detect High	t_{PDHIGH}		15		60	µs	1
Presence Detect Low	t_{PDLOW}		60		240	µs	1
Capacitance	$C_{IN/OUT}$				25	pF	

NOTES:

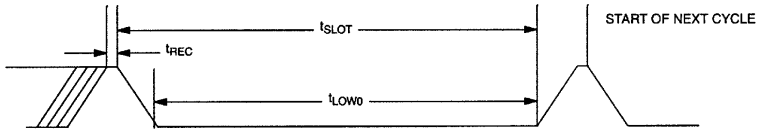
1) Refer to timing diagrams in Figure 18.
2) Under parasite power, if t_{RSTL} > 960µs, a power on reset may occur.

TYPICAL PERFORMANCE CURVE Figure 17

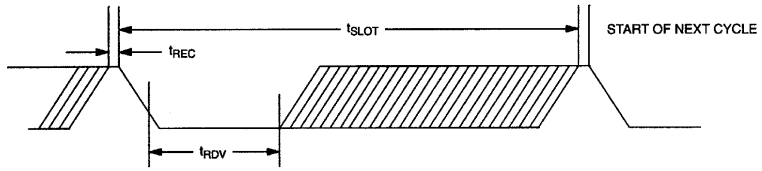


TIMING DIAGRAMS Figure 18

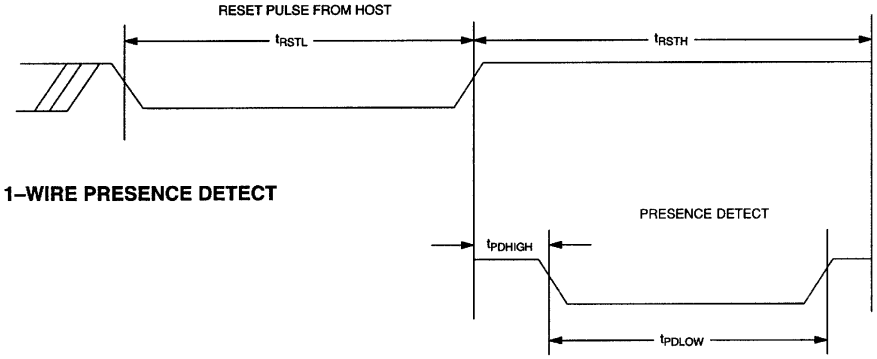
1–WIRE WRITE ZERO TIME SLOT



1–WIRE READ ZERO TIME SLOT



1–WIRE RESET PULSE



1–WIRE PRESENCE DETECT

G.2.- Datasheet del transceiver nRF401

Nordic VLSI ASA

433MHz Single Chip RF Transceiver **nRF401**

FEATURES

- True single chip FSK transceiver
- Few external components required
- No set up or configuration
- No coding of data required
- 20kbit/s data rate
- 2 channels
- Wide supply range
- Very low power consumption
- Standby mode

APPLICATIONS

- Alarm and Security Systems
- Automatic Meter Reading (AMR)
- Home Automation
- Remote Control
- Surveillance
- Automotive
- Telemetry
- Toys
- Wireless Communication

GENERAL DESCRIPTION

nRF401 is a true single chip UHF transceiver designed to operate in the 433MHz ISM (Industrial, Scientific and Medical) frequency band. It features Frequency Shift Keying (FSK) modulation and demodulation capability. nRF401 operates at bit rates up to 20kbit/s. Transmit power can be adjusted to a maximum of 10dBm. Antenna interface is differential and suited for low cost PCB antennas. nRF401 features a standby mode which makes power saving easy and efficient. nRF401 operates from a single +3-5V DC supply.

As a primary application, nRF401 is intended for UHF radio equipment in compliance with the European Telecommunication Standard Institute (ETSI) specification EN 300 220-1 V1.2.1.

QUICK REFERENCE DATA

Parameter	Value	Unit
Frequency, Channel#1/Channel#2	433.93 / 434.33	MHz
Modulation	FSK	
Frequency deviation	±15	kHz
Max. RF output power @ 400Ω, 3V	10	dBm
Sensitivity @ 400Ω, BR=20 kbit/s, BER<10^{-3}	-105	dBm
Maximum bit rate	20	kbit/s
Supply voltage	2.7 – 5.25	V
Receive supply current	250*	µA
Transmit supply current @ -10 dBm output power	8	mA
Standby supply current	8	µA

Table 1. nRF401 quick reference data.

ORDERING INFORMATION

Type number	Description	Version
nRF401-IC	20 pin SSOIC	A
nRF401-EVKIT	Evaluation kit with nRF401 IC	1.0

Table 2. nRF401 ordering information.

* The PWR_UP pin is used for power duty cycling. The duty-cycle is 2 % with a period of 200msec.

BLOCK DIAGRAM

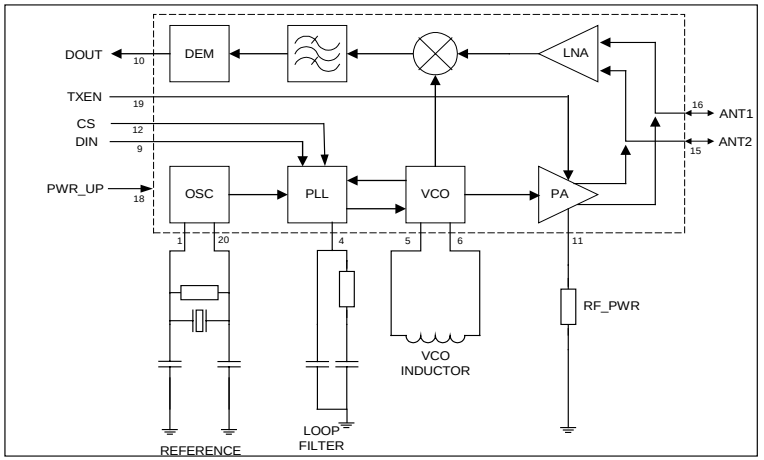


Figure 1. nRF401 block diagram with external components.

PIN FUNCTIONS

Pin	Name	Pin function	Description
1	XC1	Input	Crystal oscillator input
2	VDD	Power	Power supply (+3-5V DC)
3	VSS	Ground	Ground (0V)
4	FILT1	Input	Loop filter
5	VCO1	Input	External inductor for VCO
6	VCO2	Input	External inductor for VCO
7	VSS	Ground	Ground (0V)
8	VDD	Power	Power supply (+3-5V DC)
9	DIN	Input	Data input
10	DOUT	Output	Data output
11	RF_PWR	Input	Transmit power setting
12	CS	Input	Channel selection CS="0" ⇒ 433.93MHz (Channel#1) CS="1" ⇒ 434.33MHz (Channel#2)
13	VDD	Power	Power supply (+3-5V DC)
14	VSS	Ground	Ground (0V)
15	ANT2	Input/Output	Antenna terminal
16	ANT1	Input/Output	Antenna terminal
17	VSS	Ground	Ground (0V)
18	PWR_UP	Input	Power on/off PWR_UP = "1" ⇒ Power up (Operating mode) PWR_UP = "0" ⇒ Power down (Standby mode)
19	TXEN	Input	Transmit enable TXEN = "1" ⇒ Transmit mode TXEN = "0" ⇒ Receive mode
20	XC2	Output	Crystal oscillator output

Table 3. nRF401 pin functions.

ELECTRICAL SPECIFICATIONS

Conditions: VDD = +3V DC, VSS = 0V, T_A= -25°C to +85°C

Symbol	Parameter (condition)	Min.	Typ.	Max.	Units
VDD	Supply voltage	2.7	3	5.25	V
VSS	Ground		0		V
I_{DD}	Total current consumption Receive mode Transmit mode @ -10 dBm RF power Stand by mode		 11 8 8		 mA mA µA
P_{RF}	Max. RF output power @ 400Ω load		10		dBm
V_{IH}	Logic "1" input voltage	$0.7 \cdot V_{DD}$		V_{DD}	V
V_{IL}	Logic "0" input voltage	0		$0.3 \cdot V_{DD}$	V
V_{OH}	Logic "1" output voltage (I_{OH} = - 1.0mA)	$0.7 \cdot V_{DD}$		V_{DD}	V
V_{OL}	Logic "0" output voltage (I_{OL} = 1.0mA)	0		$0.3 \cdot V_{DD}$	V
I_H	Logic "1" input current (V_I = VDD)			+20	µA
I_L	Logic "0" input current (V_I = VSS)			-20	µA
f_1	Channel#1 frequency		433.93		MHz
f_2	Channel#2 frequency		434.33		MHz
	Dynamic range	90			dB
	Modulation type		FSK		
Δf	Frequency deviation		±15		kHz
f_{IF}	IF frequency		400		kHz
BW_{IF}	IF bandwidth	65		85	kHz
f_{XTAL}	Crystal frequency		4.0		MHz
	Crystal frequency stability requirement [1]			±45	ppm
	Loop filter voltage [3]	0.9	1.1	1.3	V
	Sensitivity @ 400Ω,BR=20 kbit/s, BER < 10^{-3}		-105		dBm
	Bit rate	0		20	kbit/s
Z_I	Recommended antenna port differential impedance		400		Ω
	Spurious emission	Compliant with EN 300-220-1 V1.2.1 [2]			

Table 4. nRF401 electrical specifications.

[1] Maximum 5dB sensitivity degradation at temperature extremes. See also page 11.
[2] With a PCB loop antenna or a differential to single ended matching network to a 50Ω antenna.
[3] See also page 9, Loop filter.

ABSOLUTE MAXIMUM RATINGS

Supply voltages

VDD .....- 0.3V to +6V
VSS 0V

Input voltage

V_I .....- 0.3V to VDD + 0.3V

Output voltage

V_O .....- 0.3V to VDD + 0.3V

Power dissipation

P_D (T_A=25°C) 250mW

Temperatures

Operating Temperature…. -25°C to +85°C
Storage Temperature…... -40°C to +125°C

Note: Stress exceeding one or more of the limiting values may cause permanent damage to the device.

ATTENTION!
Electrostatic Sensitive Device
Observe Precaution for handling

PIN ASSIGNMENT

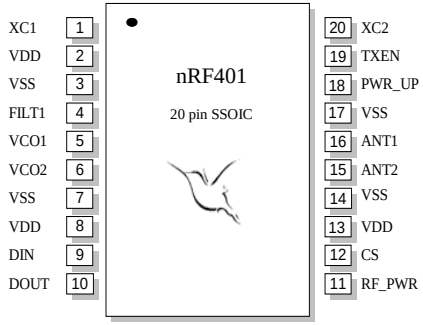
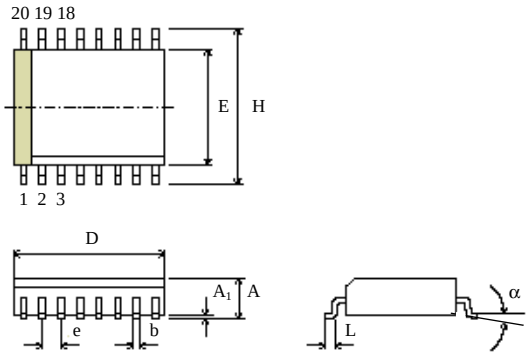


Figure 2. nRF401 pin assignment.

PACKAGE OUTLINE

nRF401, 20 pin SSOIC. (Dimensions in mm.)



Package Type		D	E	H	A	A_1	e	b	L	Copl.	α
20 pin SSOIC (Wide)	**Min**	6.90	5.00	7.40		0.05	0.65	0.22	0.55		0°
	Max	7.50	5.60	8.20	2.00			0.38	0.95	0.10	8°

Figure 3. SSOIC-20 Package outline.

IMPORTANT TIMING DATA

Timing information
The timing information for the different operations is summarised in Table 5. (TX is transmit mode, RX is receive mode and Std.by is Standby mode.)

Change of Mode	Name	Max Delay	Condition
TX ➔ RX	t_{TR}	3ms	Operational mode
RX ➔ TX	t_{RT}	1ms	
Std.by➔ TX	t_{ST}	2ms	
Std.by➔ RX	t_{SR}	3ms	
V_{DD}=0 ➔ TX	t_{VT}	4ms	Start-up
V_{DD} =0 ➔ RX	t_{VR}	5ms	

Table 5 Switching times for nRF401.

Switching TX ↔ RX (operational mode).
When switching from RX-mode to TX-mode data (DIN) may not be sent before the TXEN-input has been high for at least 1ms, see Figure 4(a).
When switching from TX-mode to RX-mode the receiver may not receive data (DOUT) before the TXEN-input has been low for at least 3ms, see Figure 4(b).

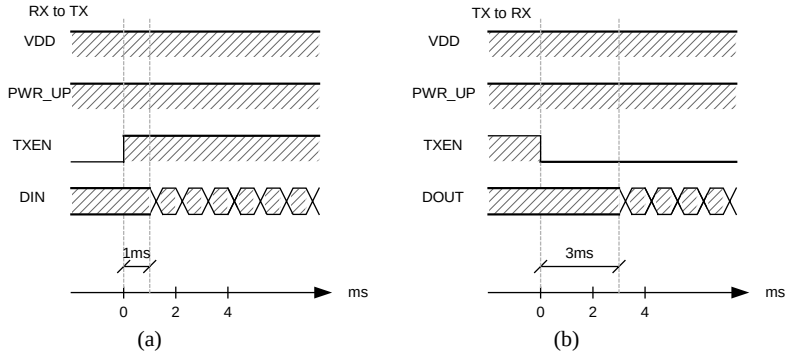


Figure 4. Timing diagram for nRF401for switching from RX to TX (a) and TX to RX (b).

Switching between standby and RX-mode (operational mode).
The time from the PWR_UP input is set to "1", until the data (DOUT) is valid is $t_{SR,}$, see Table 5. Worst case t_{SR} is 3ms for nRF401 as can be seen in Figure 5 (a).

Switching between standby and TX-mode (operational mode).
The time from the PWR_UP input is set to "1", until the synthesised frequency is stable is t_{ST}, see Table 5.

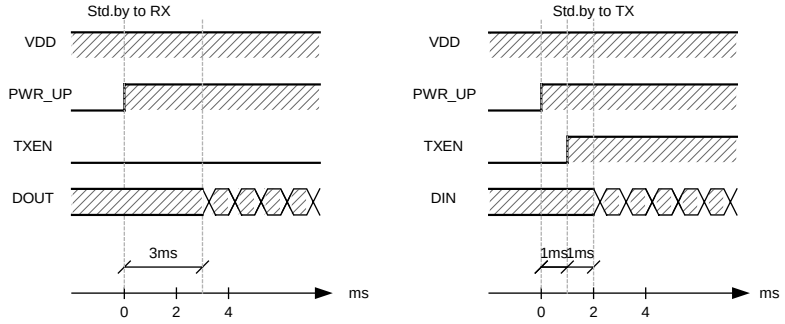


Figure 5 Timing diagram for nRF401 when going from standby to RX-mode (a) or TX-mode (b).

Power up to transmit-mode (start-up).
To avoid spurious emission outside the ISM-band when the power supply is switched on, the TXEN-input must be kept low until the synthesised frequency is stable, see Figure 6 (a).

When enabling transmit-mode, TXEN-input should be high for at least 1 ms before data (DIN) is transmitted, see Figure 6 (a).

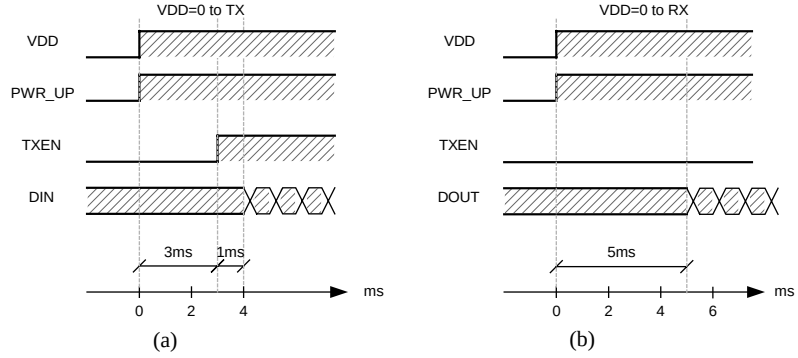


Figure 6. Timing diagram for nRF401 when powering up to TX-mode (a) or RX-mode (b).

Power up to receive mode (start up).
In transition from power up to receive mode, the receiver may not receive data (DOUT) until VDD has been stable (VDD > 2.7 V) for at least 5ms, see Figure 6(b). If an external reference oscillator is used, the receiver may receive data (DOUT) after 3ms.

APPLICATION INFORMATION

Antenna input/output
The ANT1 and ANT2 pins provide RF input to the LNA (Low Noise Amplifier) when nRF401 is in receive mode, and RF output from the PA (Power Amplifier) when nRF401 is in transmit mode. The antenna connection to nRF401 is differential and the recommended load impedance at the antenna port is 400Ω.

Figure 12 shows a typical application schematic with a differential loop antenna on a Printed Circuit Board (PCB). The output stage (PA) consists of two open collector transistors in a differential pair configuration. VDD to the PA must be supplied through the collector load. When connecting a differential loop antenna to the ANT1/ANT2 pins, VDD should be supplied through the centre of the loop antenna as shown in Figure 12.

A 50Ω single ended antenna or 50Ω test instrument may be connected to nRF401 by using a differential to single ended matching network (BALUN) as shown in Figure 7.

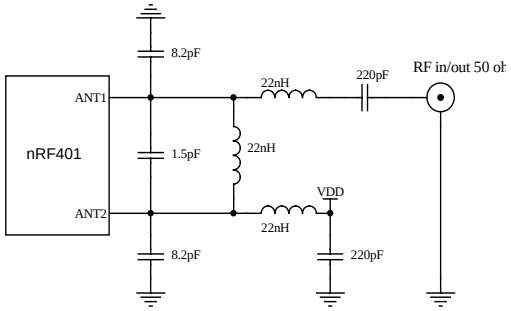


Figure 7. Connection of nRF401 to single ended antenna by using a differential to single ended matching network.

The value of the capacitor connected between ANT1 and ANT2 is dependent on parasitics in the layout. 1.5 pF is the optimal value when using Nordic VLSI layout and 1.6 mm, 2 layer, FR4 printed circuit board, see application note AN400-05.

The 22 nH inductor to VDD in Figure 7, need to have a Self-Resonance Frequency (SRF) above 868 MHz to be effective. Suitable inductors are listed in Table 6.

Vendors	WWW address	Part. no., 22 nH inductors, 0603 size
Pulse	http://www.pulseeng.com	PE-0603CD220GTT
Coilcraft	http://www.coilcraft.com	0603CS-22NXGBC
muRata	http://www.murata.com	LQW1608A22NG00
Stetco	http://www.stetco.com	0603G220GTE
KOA	http://www.koaspeer.com	KQ0603TE22NG

Table 6. Vendors and part. no. for suitable 22 nH inductors.

A single ended antenna may also be connected to nRF401 using an 8:1 impedance RF transformer. The RF transformer must have a centre tap at the primary side for VDD supply.

RF output power
The external bias resistor R3 connected between the RF_PWR pin and VSS in Figure 12 sets the output power. The RF output power may be set to levels up to +10dBm. In Figure 8 the output power is plotted for power levels down to, but not limited to, -8.5dBm for a differential load of 400Ω. DC power supply current versus external bias resistor value is shown in Figure 9.

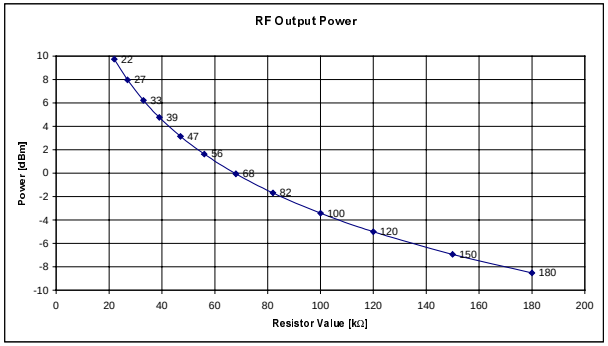


Figure 8. RF output power vs. external power setting resistor (R3) for nRF401.

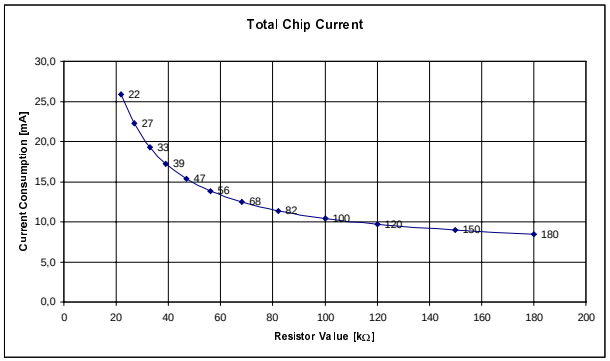


Figure 9. Total chip current consumption vs. external power setting resistor (R3) for nRF401.

PLL loop filter

The synthesiser loop filter is an external, single-ended second order lag-lead filter. The recommended filter component values are: C3 = 820 pF, C4 =15 nF, and R2 = 4.7 kΩ, see Figure 12.

The loop filter voltage, measured at pin 4, should be 1.1 ± 0.2 V. Measuring the specified loop filter voltage verifies that the VCO inductor value and placement are correct. This means optimal nRF401 performance.

VCO inductor

An external 22nH inductor connected between the VCO1 and VCO2 pins is required for the on-chip voltage controlled oscillator (VCO). This inductor must be a high quality chip inductor, Q > 45 @ 433 MHz, with a maximum tolerance of ± 2%. The following 22 nH inductors (0603) are suitable for use with nRF401.

Vendors	WWW address	Part. no., 22 nH inductors, 0603 size
Pulse	http://www.pulseeng.com	PE-0603CD220GTT
Coilcraft	http://www.coilcraft.com	0603CS-22NXGBC
muRata	http://www.murata.com	LQW1608A22NG00
Stetco	http://www.stetco.com	0603G220GTE
KOA	http://www.koaspeer.com	KQ0603TE22NG

Table 7. Vendors and part no. for suitable 22nH inductors.

The VCO inductor placement is important. The optimum placement of the VCO inductor gives a PLL loop filter voltage of 1.1 ±0.2 V, which can be measured at FILT1 (pin4). For a 0603 size inductor the length between the centre of the VCO1/VCO2 pad and the centre of the inductor pad should be 5.4 mm, see Figure 13 (c) (layout, top view), for a 2 layer, 1.6 mm thick FR4 PCB.

Crystal specification

To achieve an active crystal oscillator (XOSC) with low power consumption, certain requirements apply for crystal loss and capacitive load.

The crystal specification is:

$f = 4.0000\ MHz$	Crystal parallel resonant frequency
$Co \le 7\ pF$	Crystal parallel equivalent capacitance
$ESR \le 150\ ohm$.	Crystal equivalent series resistance
$C_L \le 14\ pF$	Total crystal load capacitance, including capacitance in PCB layout.

For the crystal oscillator shown in Figure 10 the load capacitance is given by:

$C_L = \frac{C1' \cdot C2'}{C1' + C2'}$, Where C1´ = C1 + C_{PCB1} and C2´ = C2 + C_{PCB2}

C1 and C2 are 0603 SMD capacitors as shown in the application schematic, see Figure 12 and Table 9. C_{PCB1} and C_{PCB2} are the layout parasitic capacitance on the circuit board. Layout parasitics are significant when using SMD crystals on PCBs with ground planes. Changes in C_L leads to changes in crystal frequency.

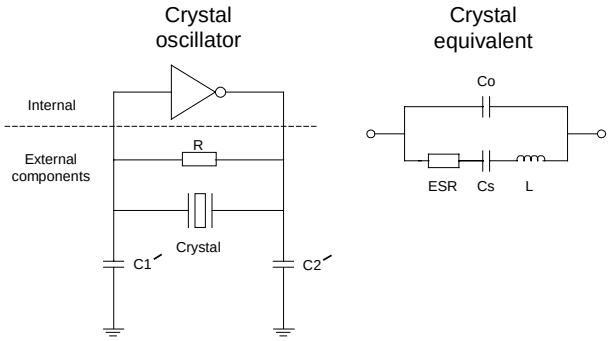


Figure 10. Crystal oscillator and crystal equivalent.

Sharing a reference crystal with a micro-controller

Figure 11 shows circuit diagram of a typical application where nRF401 and a micro controller share the reference crystal.

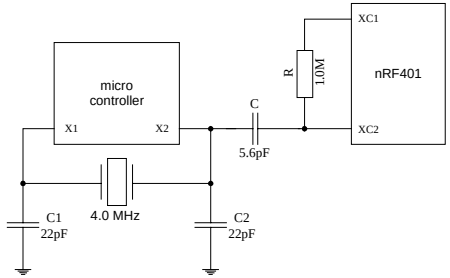


Figure 11. nRF401 and a micro-controller sharing the reference crystal.

The crystal reference line from the micro-controller must be shielded from noise, e.g. not be routed close to full swing digital data or control signals.

When sharing crystal, frequency (f) and frequency tolerance of the crystal is set by nRF401 specifications. C_L, C_0 and ESR are set by the microcontroller (MCU) specifications. The voltage amplitude at XC2 should be > 300 mV_{PP}. Changing the value of C, see figure 11, changes the XC2 voltage amplitude.

Frequency difference between transmitter and receiver
For optimum performance, the total frequency difference between transmitter and receiver should not exceed 70 ppm (30 kHz). This yields a crystal stability requirement of ±35 ppm for the transmitter and receiver. Frequency difference exceeding this will result in a -12dB/octave drop in receiver sensitivity. The functional frequency window of the transmission link is typically 450 ppm (200 kHz).

Example: A crystal with ±20 ppm frequency tolerance and ±25 ppm frequency stability over the operating temperature has a worst case frequency difference of ±45 ppm. If the transmitter and receiver operate in different temperature environments, the resulting worst-case frequency difference may be as high as 90 ppm. Resulting drop in sensitivity due to the extra 20 ppm, is then approx. 5dB

Transmit/receive mode selection
TXEN is a digital input for selection of transmit or receive mode.
TXEN = "1" selects transmit mode.
TXEN = "0" selects receive mode.

Channel#1 / Channel#2 selection
CS is a digital input for selection of either channel#1 (f_1=433.93MHz) or channel#2 (f_2=434.33MHz).
CS = "0" selects channel#1.
CS = "1" selects channel#2.

Power up
PWR_UP is a digital input for selection of normal operating mode or standby mode.
PWR_UP = "1" selects normal operating mode.
PWR_UP = "0" selects standby mode.

Input			Response	
TXEN	CS	PWR_UP	Channel #	Mode
0	0	1	1	RX
0	1	1	2	RX
1	0	1	1	TX
1	1	1	2	TX
X	X	0	--	Standby

Table 8. Required setting for standby and channel selection in RX and TX.

D_{IN} (data input) and D_{OUT} (data output)
The DIN pin is the input to the digital modulator of the transmitter. The input signal to this pin should be standard CMOS logic level at data rates up to 20 kbit/s. No coding of data is required.

DIN = "1" → f = f_0 + Δf
DIN = "0" → f = f_0 - Δf

The demodulated digital output data appear at the D_{OUT} pin at standard CMOS logic levels.

f_0 + Δf → DOUT="1",
f_0 - Δf → DOUT="0".

PCB layout and decoupling guidelines
A well-designed PCB is necessary to achieve good RF performance. A PCB with a minimum of two layers including a ground plane is recommended for optimum performance.

The nRF401 DC supply voltage should be decoupled as close as possible to the VDD pins with high performance RF capacitors, see Table 9. It is preferable to mount a large surface mount capacitor (e.g. 4.7 µF tantalum) in parallel with the smaller value capacitors. The nRF401 supply voltage should be filtered and routed separately from the supply voltages of any digital circuitry (star routed).

Long power supply lines on the PCB should be avoided. All device grounds, VDD connections and VDD bypass capacitors must be connected as close as possible to the nRF401 IC. For a PCB with a topside RF ground plane, the VSS pins should be connected directly to the ground plane. For a PCB with a bottom ground plane, the best technique is to have via holes in or close to the VSS pads.

Full swing digital data or control signals should not be routed close to the PLL loop filter components or the external VCO inductor.

The VCO inductor placement is important. The optimum placement of the VCO inductor gives a PLL loop filter voltage of 1.1 ±0.2 V, which can be measured at FILT1 (pin4). For a 0603 size inductor the length between the centre of the VCO1/VCO2 pad and the centre of the inductor pad should be 5.4 mm, see Figure 13 (c) (layout, top view), for a 2 layer, 1.6 mm thick FR4 PCB.

PCB layout example
Figure 13 shows a PCB layout example for the application schematic in Figure 12. A double-sided FR-4 board of 1.6mm thickness is used. This PCB has a ground plane on the bottom layer. Additionally, there are ground areas on the component side of the board to ensure sufficient grounding of critical components. A large number of via holes connect the top layer ground areas to the bottom layer ground plane. There must **NOT** be a ground plane shielding the radiation from the antenna.

For more layout information, please refer to application note nAN400-05, "nRF401 RF and antenna layout".

APPLICATION SCHEMATIC

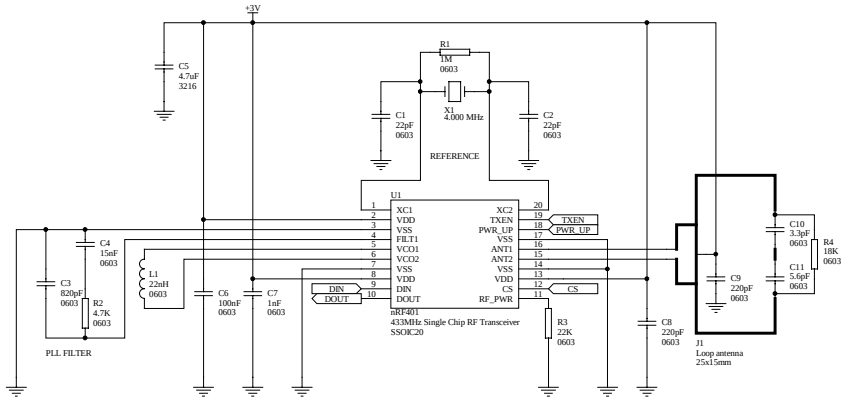
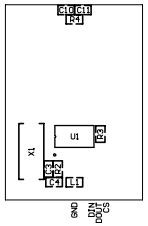


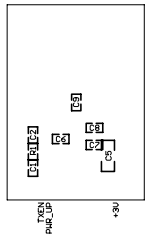
Figure 12. nRF401 application schematic.

Component	Description	Size	Value	Tolerance	Units
C1	NP0 ceramic chip capacitor, (Crystal oscillator)	0603	22		pF
C2	NP0 ceramic chip capacitor, (Crystal oscillator)	0603	22		pF
C3	X7R ceramic chip capacitor, (PLL loop filter)	0603	820		pF
C4	X7R ceramic chip capacitor, (PLL loop filter)	0603	15		nF
C5	Tantalum chip capacitor, (Supply decoupling)	3216	4.7		µF
C6	X7R ceramic chip capacitor, (Supply decoupling)	0603	100		nF
C7	X7R ceramic chip capacitor, (Supply decoupling)	0603	1		nF
C8	NP0 ceramic chip capacitor, (Supply decoupling)	0603	220		pF
C9	NP0 ceramic chip capacitor, (Supply decoupling)	0603	220		pF
C10	NP0 ceramic chip capacitor, (Antenna tuning)	0603	3.3	±0.1	pF
C11	NP0 ceramic chip capacitor, (Antenna tuning)	0603	5.6	±0.1	pF
L1	VCO inductor, Q>45 @ 433 MHz	0603	22	±2%	nH
R1	0.1W chip resistor, (Crystal oscillator)	0603	1.0		MΩ
R2	0.1W chip resistor, (PLL loop filter)	0603	4.7		kΩ
R3	0.1W chip resistor, (Transmitter power setting)	0603	22		kΩ
R4	0.1W chip resistor, (Antenna Q reduction)	0603	18		kΩ
X1	Crystal	-	4.000		MHz

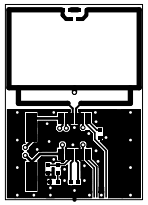
Table 9. Recommended External Components.



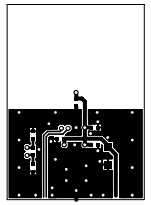
a) Top silk screen



b) Bottom silk screen



c) Top view



d) Bottom view

Figure 13. PCB layout (example) for nRF401 with loop antenna.

DEFINITIONS

Data sheet status	
Objective product specification	This datasheet contains target specifications for product development.
Preliminary product specification	This datasheet contains preliminary data; supplementary data may be published from Nordic VLSI ASA later.
Product specification	This datasheet contains final product specifications. Nordic VLSI ASA reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.
Limiting values	
Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Specifications sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.	
Application information	
Where application information is given, it is advisory and does not form part of the specification.	

Table 10. Definitions.

Nordic VLSI ASA reserves the right to make changes without further notice to the product to improve reliability, function or design. Nordic VLSI does not assume any liability arising out of the application or use of any product or circuits described herein.

LIFE SUPPORT APPLICATIONS

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Nordic VLSI ASA customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nordic VLSI ASA for any damages resulting from such improper use or sale.

Product specification: Revision Date: 28.02.2002.

Datasheet order code: 280202nRF401

YOUR NOTES

G.3.- Datasheet del nRF401-LOOPKIT

Nordic VLSI ASA

nRF401 Loop Kit **nRF401-LOOPKIT**

1. Introduction

The Loop Kit for the **nRF401** Single chip 433MHz RF transceiver has been developed to enable customers to get hands-on experience with the functionality of the device combined with loop antennas in their applications.

The **nRF401** Loop Kit is convenient for use in the prototyping phase when developing, testing and debugging PC software, microcontroller code and/or electronic circuitry for interfacing towards **nRF401** and a wireless communication link.

The **nRF401** Loop Kit consists of the following items:

- Two **nRF401** Loop Boards with the **nRF401** transceiver.
  Each **nRF401** Loop Board consists of three **nRF401** Loop Modules with different loop antenna sizes
- **nRF401-LOOPKIT** documentation

The **nRF401** datasheet can be downloaded from the Nordic VLSI web pages: http://www.nvlsi.no.

This document specifies the usage of **nRF401** Loop Modules as complete radio modules in customer's development of short-range wireless communication systems.

For information about loop antennas, please refer to application note nAN400-03, "Small loop antennas", and application note nAN400-05, "nRF401 RF and antenna layout".

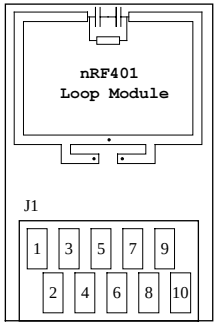
2. nRF401 Loop Module description

The **nRF401** Loop Module pin placement and description is shown in Figure 2.1. Appendix 1 shows the **nRF401** Loop Module circuit diagram, and the component list is given in Appendix 2.

All digital control and signal lines are available for connection to customer's application circuitry at connector footprint J1. Connections to the **nRF401** Loop Module can be soldered directly to the connector footprint J1 solder pads. A via hole has been placed in each solder pad to get a more secure connection.

Power and ground is applied to the **nRF401** Loop Module via connector footprint J1. **Voltage range on this input must be restricted to +2.7V to +5.25V**.

If it is desirable to use a cable with a connector, an SMD connector can be mounted on connector footprint J1. The PCB footprint corresponds to the recommended PCB layout for the AMP Micro-MaTch SMD 10 pin female connector (AMP Part number 8(1)-188275-0). The corresponding cable connector is the AMP Micro-MaTch 10 pin male connector (AMP Part number 8(1)-215083-0).



Pin	Name	Description
1	TXEN	Transmit enable TXEN = "1" ⇒ Transmit mode TXEN = "0" ⇒ Receive mode
2	PWR_UP	Power on/off PWR_UP = "1" ⇒ Power up (Operating mode) PWR_UP = "0" ⇒ Power down (Standby mode)
3	GND	Ground (0V)
4	CS	Channel selection CS="0" ⇒ 433.92MHz (Channel#1) CS="1" ⇒ 434.33MHz (Channel#2)
5	GND	Ground (0V)
6	DIN	Data input
7	GND	Ground (0V)
8	DOUT	Data output
9	GND	Ground (0V)
10	VDD	Power supply (+3-5V DC)

Figure 2.1. **nRF401** Loop Module pin placement and description

The applied 4.000MHz reference crystal X1 has an overall frequency tolerance (frequency tolerance at 25°C + temperature drift) that does not exceed ±35 ppm in the operating temperature range -25°C to +75°C.

The external bias resistor R3 connected between the RF_PWR pin and VSS sets the output power. On the **nRF401** Loop Module R3=22K which sets the output power to +10dBm.

3. Getting started

Before starting to use the **nRF401** Loop Kit, each **nRF401** Loop Module should be snapped out from the **nRF401** Loop Boards.

The **nRF401** Loop Modules are complete radio modules with a digital interface for connection to the customer's application circuitry. The **nRF401** Loop Modules enable customers to find the combination of loop antenna sizes to use in their communication system that fulfils the application range requirements.

As mentioned in application note nAN400-05, it is recommended that a system should not be designed with a longer communication range than the application requires. Estimations on communication range with combinations of the antennas used on the **nRF401** Loop Modules can be made based on the theory given in application note nAN400-03. Initial communication range tests should be carried out with the combination of the smallest antennas that, based on the estimations, satisfies the range requirements. If the achieved communication range does not satisfy the requirements, one or both of the Loop Modules in the communication link should be exchanged with a Loop Module with a larger antenna.

Figure 3.1 shows a typical set-up with two **nRF401** Loop Modules connected to the customer's application circuitry in order to develop and debug a complete wireless communication link.

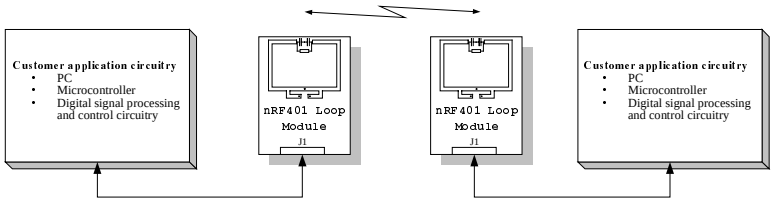


Figure 3.1. Set-up with two **nRF401** Loop Modules connected to the customer's application circuitry

The TXEN, PWR_UP, CS, DIN, DOUT and VDD signals available at connector footprint J1 on the **nRF401** Loop Module should be connected to the customer's application circuitry. The following precautions should be taken when connecting to the **nRF401** Loop Module:

- If the **nRF401** Loop Module is to be connected to the customer's application circuitry with a cable via connector footprint J1, this cable should be a twisted pair flat cable. Each signal wire must be twisted together with the adjacent GND wire (that is VDD/GND, DOUT/GND, DIN/GND and so on). Do remember to connect the GND-wires in the twisted pair cable to both the **nRF401** Loop Module and the customer's application circuitry. The cable length must be kept as short as possible.

- The power supply wire and the ground wire connected to the **nRF401** Loop Module should be twisted together.

- Ensure that the peak-to-peak voltage level of the data input signal DIN and the control signals never exceed the **nRF401** device power supply level.

Details regarding digital input/output voltage levels and timing requirements for control of the **nRF401** device can be found in the **nRF401** datasheet.

The **nRF401** Loop Module should not be located inside a conductive or metal enclosure. Care should be taken to keep the loop antenna away from metal objects such as batteries, PCB ground plane and traces, or any other metal object. Generally, mounting the **nRF401** Loop Module in front of a product with the loop antenna radiating into free space will yield the best result.

Be aware of that the layout of the rest of the communication system may influence the antenna tuning and require measurement of the antenna together with the complete system in order to find the values for the antenna tuning capacitors and Q reduction resistor for your system.

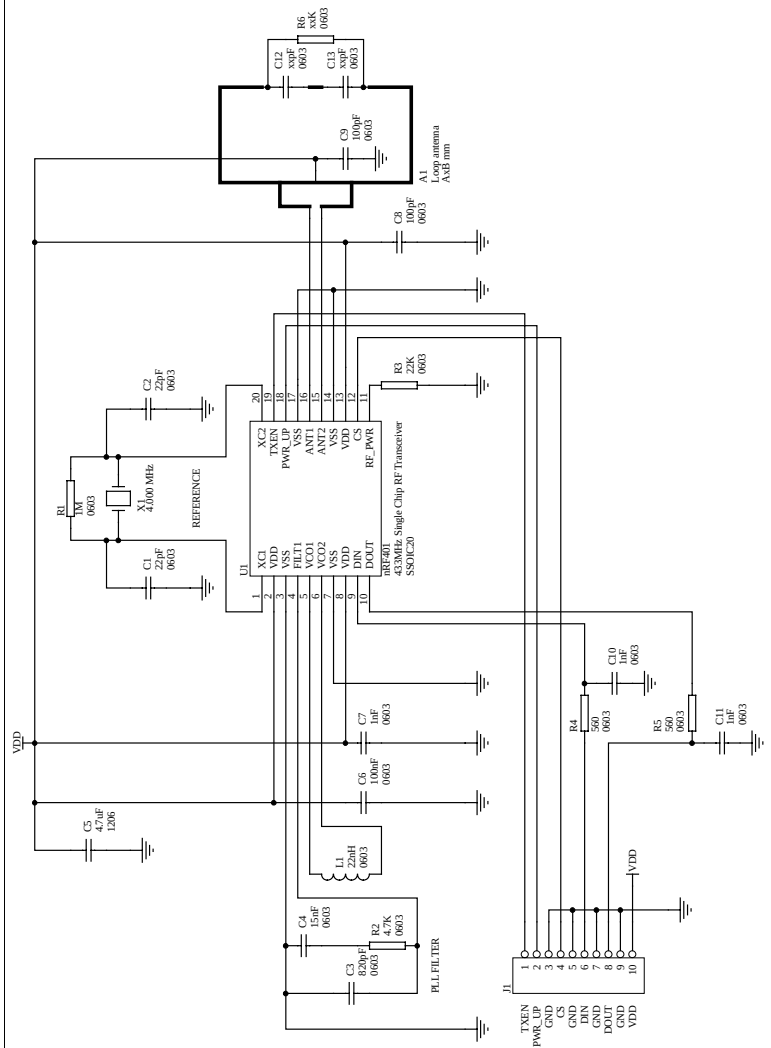


Figure A.1. Loop Module Circuit Diagram

Component	Description	Value	Tolerance	Units
C1	NP0 Ceramic 0603 Chip Capacitor (Crystal oscillator)	22	± 5%	pF
C2	NP0 Ceramic 0603 Chip Capacitor (Crystal oscillator)	22	± 5%	pF
C3	X7R Ceramic 0603 Chip Capacitor (PLL loop filter)	820	± 10%	pF
C4	X7R Ceramic 0603 Chip Capacitor (PLL loop filter)	15	± 10%	nF
C5	Y5V Ceramic 1206 Chip Capacitor (Supply decoupling)	4.7	+80% -20%	µF
C6	X7R Ceramic 0603 Chip Capacitor (Supply decoupling)	100	± 10%	nF
C7	X7R Ceramic 0603 Chip Capacitor (Supply decoupling)	1	± 10%	nF
C8	NP0 Ceramic 0603 Chip Capacitor (Supply decoupling)	100	± 5%	pF
C9	NP0 Ceramic 0603 Chip Capacitor (Supply decoupling)	100	± 5%	pF
C10	X7R Ceramic 0603 Chip Capacitor (EMI filter)	1	± 10%	nF
C11	X7R Ceramic 0603 Chip Capacitor (EMI filter)	1	± 10%	nF
C12	NP0 Ceramic 0603 Chip Capacitor (Antenna tuning) 18x10mm loop antenna 25x15mm loop antenna 35x20mm loop antenna	 5.6 3.3 1.8	 ± 0.1 ± 0.1 ± 0.1	 pF pF pF
C13	NP0 Ceramic 0603 Chip Capacitor (Antenna tuning) 18x10mm loop antenna 25x15mm loop antenna 35x20mm loop antenna	 10 5.6 4.7	 ± 1% ± 0.1 ± 0.1	 pF pF pF
L1	0603 chip inductor (VCO)	22	± 2% Q Min > 45@433MHz	nH
R1	0.1W 0603 chip resistor (Crystal oscillator)	1	± 1%	MΩ
R2	0.1W 0603 chip resistor (PLL loop filter)	4.7	± 1%	kΩ
R3	0.1W 0603 chip resistor (Transmitter power setting)	22	± 1%	kΩ
R4	0.1W 0603 chip resistor (EMI filter)	560	± 1%	Ω
R5	0.1W 0603 chip resistor (EMI filter)	560	± 1%	Ω
R6	0.1W 0603 chip resistor (Antenna Q reduction) 18x10mm loop antenna 25x15mm loop antenna 35x20mm loop antenna	 18 18 68	 ± 1% ± 1% ± 1%	 kΩ kΩ kΩ
X1	Crystal	4.000	<35 ppm (frequency tolerance at 25°C + temperature drift)	MHz
U1	**nRF401**			

Table A.2. **nRF401** Loop Modules Component List

DEFINITIONS

Product specification
This Loop Kit documentation contains final product specifications. Nordic VLSI ASA reserves the right to make changes at any time without notice in order to improve design and supply the best possible product.
Limiting values
Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Specifications sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.
Application information
Where application information is given, it is advisory and does not form part of the specification.

Table 1. Definitions.

Nordic VLSI ASA reserves the right to make changes without further notice to the product to improve reliability, function or design. Nordic VLSI does not assume any liability arising out of the application or use of any product or circuits described herein.

LIFE SUPPORT APPLICATIONS

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Nordic VLSI ASA customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Nordic VLSI ASA for any damages resulting from such improper use or sale.

Product specification. Revision Date : 31.10.2002.

nRF401 Loop Kit order code : 311002-nRF401-LOOPKIT.

YOUR NOTES

Nordic VLSI - World Wide Distributors

For Your nearest dealer, please see http://www.nvlsi.no

nordic
Nordic VLSI ASA

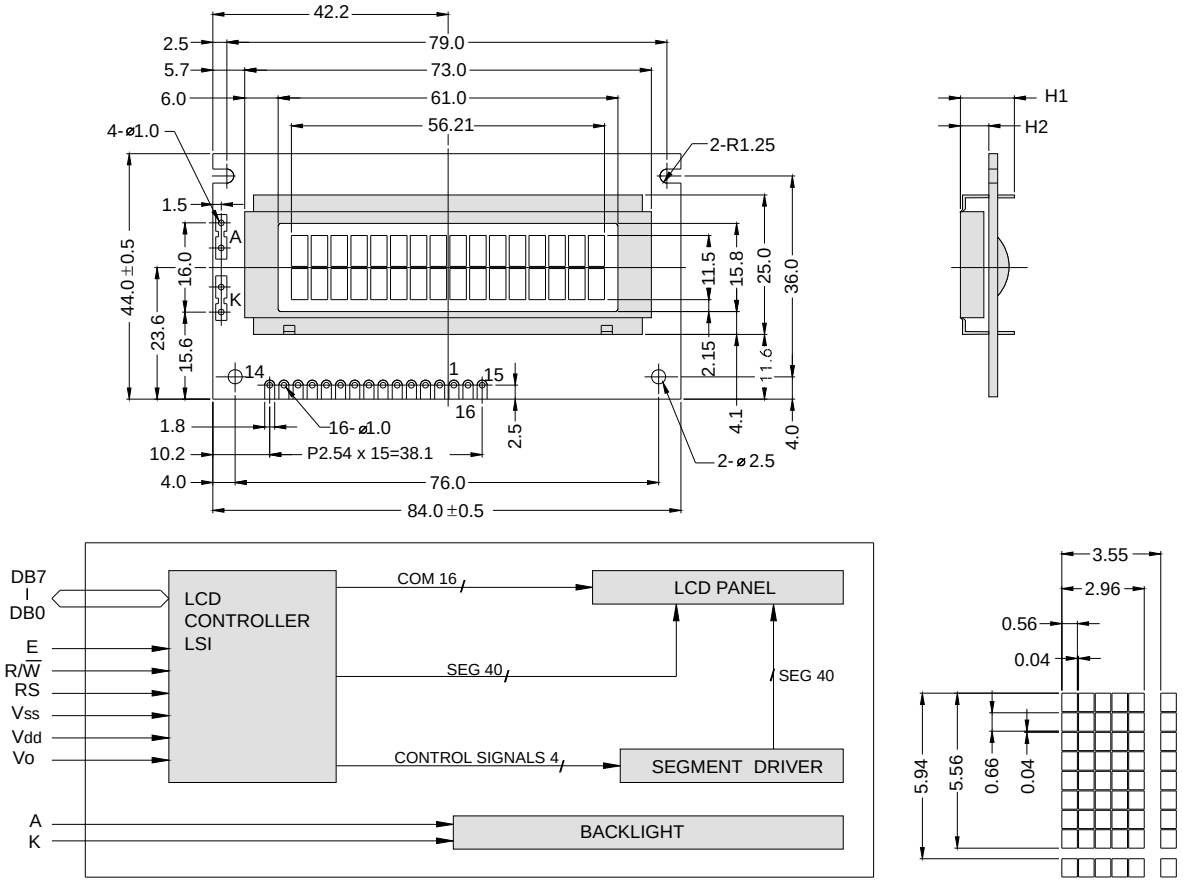
Main Office:
Vestre Rosten 81, N-7075 Tiller, Norway
Phone: +47 72 89 89 00, Fax: +47 72 89 89 89
E-mail: nRF@nvlsi.no
Visit the Nordic VLSI ASA website at http://www.nvlsi.no

G.4.- Datasheet de la pantalla LCD PC1602-F

POWERTIP

PC 1602-F

OUTLINE DIMENSION & BLOCK DIAGRAM



The tolerance unless classified ±0.3mm

MECHANICAL SPECIFICATION			
Overall Size	84.0 x 44.0	Module	H2 / H1
View Area	61.0 x 15.8	W /O B/L	5.1 / 9.7
Dot Size	0.56 x 0.66	EL B/L	5.1 / 9.7
Dot Pitch	0.60 x 0.70	LED B/L	9.4 / 14.0

PIN ASSIGNMENT		
Pin no.	Symbol	Function
1	Vss	Power supply(GND)
2	Vdd	Power supply(+)
3	Vo	Contrast Adjust
4	RS	Register select signal
5	R/W	Data read / write
6	E	Enable signal
7	DB0	Data bus line
8	DB1	Data bus line
9	DB2	Data bus line
10	DB3	Data bus line
11	DB4	Data bus line
12	DB5	Data bus line
13	DB6	Data bus line
14	DB7	Data bus line
15	A	Power supply for LED B/L (+)
16	K	Power supply for LED B/L (–)

ABSOLUTE MAXIMUM RATING					
Item	Symbol	Condition	Min.	Max.	Units
Supply for logic voltage	Vdd-Vss	25°C	-0.3	7	V
LCD driving supply voltage	Vdd-Vee	25°C	-0.3	13	V
Input voltage	Vin	25°C	-0.3	Vdd+0.3	V

ELECTRICAL CHARACTERISTICS									
Item	Symbol	Condition	Min.		Typical		Max.		Units
Power supply voltage	Vdd-Vss	25°C	2.7		–		5.5		V
LCD operation voltage	Vop	Top	N	W	N	W	N	W	V
		-20°C	–	7.1	–	7.5	–	7.9	V
		0°C	4.5	–	5.1	–	5.3	–	V
		25°C	4.1	6.1	4.7	6.4	4.9	6.7	V
		50°C	3.8	–	4.4	–	4.6	–	V
		70°C	–	5.7	–	6	–	6.3	V
LCM current consumption (No B/L)	Idd	Vdd=5V	–		2		3		mA
Backlight current consumption	LED/edge	VB/L=4.2V	–		40		–		mA
	LED/array	VB/L=4.2V	–		120		–		mA

REMARK

LCD option: STN, TN, FSTN

Backlight Option: LED,EL Backlight feature, other Specs not available on catalog is under request.