UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Escola d'Enginyeria de Telecomunicació
i Aeroespacial de Castelldefels

TREBALL DE FI DE GRAU

TÍTOL DEL TFG: Estudi del consum energètic de convertidors analògic-digital integrats en microcontroladors

TITULACIÓ: Grau en Enginyeria de Sistemes Aeroespacials

AUTORS: Jose Francisco Merchan Holmes
Mandeep Singh

DIRECTOR: Ferran Reverter

DATA: 29 de juliol de 2019

Títol: Estudi del consum energètic de convertidors analògic-digital integrats en microcontroladors

Autors: Jose Francisco Merchan Holmes
Mandeep Singh

Director: Ferran Reverter

Data: 29 de juliol de 2019

Resum

Els microcontroladors són circuits integrats programables destinats a governar una aplicació determinada. Aquests estan presents en molts dels dispositius electrònics que ens envolten: a la feina, a casa nostra i en la nostra vida en general. Es poden trobar controlant el funcionament de mòbils, ordinadors, tablets, TV, rellotges intel·ligents, etc. També són els dispositius principals de control en nodes sensors de baix consum aplicats a ciutats i edificis intel·ligents.

Els microcontroladors disposen de recursos hardware (anomenats perifèrics) integrats en el mateix xip i que poden funcionar simultàniament amb l'execució d'instruccions feta per la CPU. Un d'aquests perifèrics és el convertidor analògic-digital (ADC) que té la funció de convertir una tensió d'entrada analògica en una dada digital.

Aquest treball de final de grau està destinat a estudiar el consum energètic d'aquests ADC integrats. Per fer aquest estudi, es presentaran dos convertidors integrats en diferents microcontroladors comercials i es realitzaran diferents experiments per mesurar el seu consum. S'estudiarà el consum d'energia dels dos convertidors ADC per separat i posteriorment es farà una comparativa. Finalment s'extrauran conclusions sobre el mode de funcionament més adequat (per exemple, en quant a la freqüència de treball) per obtenir una conversió ADC de baix consum que permeti, per exemple allargar el temps de vida de nodes sensors.

Title : Study of the energy consumption of analog-digital converters integrated into microcontrollers

Authors: Jose Francisco Merchan Holmes
Mandeep Singh

Advisor: Ferran Reverter

Date: July 29, 2019

Overview

Microcontrollers are programmable integrated circuits designed to control any particular application. Nowadays are present in many of the electronic devices that surround us: at work, at home and in our life in general. They can be found by controlling the operation of mobiles, computers, tablets, TVs, smart clocks, etc. These are also the main control devices in low-energy sensing nodes applied to smart cities and intelligent buildings.

The microcontrollers have integrated hardware modules (called peripherals) in the same chip and that can work simultaneously with the execution of instructions made by the CPU. One of these peripherals is the analog-digital converter (ADC) that has the function of converting an analog input voltage to digital data.

This bachelor thesis is intended to study the energy consumption of these integrated ADCs. To do this study, two integrated converters will be presented in different commercial microcontrollers and different experiments will be carried out to measure their consumption. The power consumption of the two ADCs will be studied separately and later a comparison will be made. Finally, conclusions will be drawn on the most appropriate mode of operation (for example, in terms of working frequency of this peripheral) for a low-consumption ADC conversion that allows, for example, increasing the life time of sensor nodes.

Aquest treball no seria el que és sense la paciència i l'ajuda que ens han brindat principalment els nostres pares.
També volem donar les gràcies al nostre director Dr. Ferran Reverter per la seva feina, el seu suport, la seva guia, els seus consells i les estones dedicades en tot el que hem necessitat.
Gràcies per fer-ho possible.

La nostra recompensa es troba en l'esforç i no en el resultat.
Un esforç total és una victòria completa.
-**Mahatma Gandhi**

ÍNDEX

ÍNDEX DE FIGURES

ÍNDEX DE TAULES

CAPÍTOL 1. INTRODUCCIÓ

Aquest projecte ha estat realitzat com a treball de fi de grau de la titulació de Grau en Enginyeria de Sistemes Aeroespacials. És un projecte que pertany més específicament al departament d'electrònica i ha estat guiat pel director Dr. Ferran Reverter.

En aquest projecte s'estudiarà principalment el consum energètic de diferents ADCs, integrats en diferents microcontroladors (un de la companyia Atmel i l'altre de Microchip), amb l'objectiu de donar guies de disseny per allargar el temps de vida de nodes sensors controlats per microcontroladors.

1.1. Revolució tecnològica

La revolució tecnològica que actualment estem vivint té molt a veure amb conceptes com: xarxes de sensors sense fils, nodes sensors, internet de les coses (IoT), sensors autònoms, captació d'energia, etc. S'analitzarà amb més detall el concepte IoT, ja que pràcticament engloba els altres.

L'anomenat "Internet de les coses" (IoT) va ser empleat per primera vegada pel britànic Kevin Ashton en el 1999 per descriure un sistema, pel qual els objectes del món físic es podien connectar a internet per diferents tipus de sensors com s'observa en la Figura 1.1. Ashton va analitzar la seva idea des del punt de vista de poder connectar a Internet les etiquetes d'identificació per radiofreqüència (RFID) que s'utilitzaven en les cadenes de subministrament, per contar i fer un seguiment de la mercaderia sense la necessitat d'intervenció humana. Avui en dia, aquest terme "IoT" s'ha popularitzat tant per descriure qualsevol tipus d'escenari que tracti d'un objecte que es podria automatitzar.

Sense cap mena de dubte és un tema emergent d'importància tècnica, social i econòmica, i avui en dia ja s'està intentant connectar productes de consum, automòbils, camions, màquines industrials i de serveis públics, sensors amb connectivitat a Internet, donant pas a la pretensió de transformar la manera en què treballem i vivim. Les projeccions de l'impacte de l'IoT sobre Internet i l'economia són impressionants: n'hi ha que anticipen que l'any 2025 hi haurà fins a cent mil milions de dispositius connectats a la IoT i que el seu impacte serà d'uns 11 mil milions de dòlars.

Figura 1.1: La Internet de les coses

No obstant això, la IoT també planteja importants reptes que podrien dificultar la realització dels seus potencials beneficis. Notícies sobre atacs a dispositius connectats a Internet, la por a la vigilància i les preocupacions relacionades amb la privacitat, ja han captat l'atenció del públic. Els desafiaments tècnics segueixen allà, però a més estan sorgint nous reptes polítics, jurídics i de desenvolupament.

La IoT és un tema important en la indústria de la tecnologia, les polítiques i els cercles d'enginyeria, i s'ha convertit en notícia de primera plana, tant en la premsa especialitzada com en els mitjans populars. Aquesta tecnologia s'encarna en una àmplia gamma de productes, sistemes i sensors en xarxa, que aprofiten els avenços en la potència de càlcul, la miniaturització dels components electrònics i les interconnexions de xarxa per oferir noves capacitats que abans no eren possibles. Una gran quantitat de conferències, informes i articles de notícies estan discutint i debatent el potencial impacte de la "revolució de la IoT", des de noves oportunitats de mercat i models de negoci fins a les preocupacions pel que fa a la seguretat, la privacitat i la interoperabilitat tècnica.

La implementació a gran escala de dispositius de la IoT promet transformar molts aspectes de la forma en què vivim. Per als consumidors, els nous productes de la IoT -Electrodomèstics, components d'automatització de la llar i dispositius de gestió d'energia amb connexió a Internet ens estan portant cap a una visió de la *casa intel·ligent* que ofereix més seguretat i eficiència energètica. Altres dispositius personals de la IoT -entre ells els dispositius portàtils per monitoritzar i gestionar l'activitat física i els dispositius mèdics amb connexió a Internet estan transformant la forma en què s'ofereixen els serveis de salut. Aquesta tecnologia promet ser beneficiosa per a les persones grans o amb discapacitats, millorant els seus nivells d'independència i qualitat de vida a un cost raonable.

Els sistemes de la IoT com els vehicles connectats en xarxa, els sistemes de trànsit intel·ligents i els sensors integrats en carreteres i ponts, ens apropen més a la idea de *ciutats intel·ligents*, que ajuden a minimitzar la congestió i el consum d'energia. La tecnologia de la IoT ofereix la possibilitat de transformar l'agricultura, la indústria i la producció i distribució d'energia mitjançant l'augment de la disponibilitat d'informació al llarg de la cadena de valor de la producció per mitjà de sensors connectats en xarxa. No obstant això, la IoT planteja moltes preguntes i desafiaments que s'han de tenir en compte i abordar perquè es puguin realitzar els seus potencials beneficis.

Aquesta tecnologia no només la trobem en les ciutats i en l'entorn en què diàriament caminem, ja que sense cap mena de dubte el món aeri s'està alimentant de molts coneixements i moltes aplicacions que de cara al futur podrien gaudir els vehicles aeris, a saber:

- Ales intel·ligents
- Avions solars
- Sensors de detecció més desenvolupats
- Avions elèctrics/intel·ligents
- Robots com a tripulació

1.2. Microcontroladors i mesures

És innegable que els avenços tecnològics que hem experimentat a les ultimes dècades han facilitat enormement la vida de l'ésser humà, ja sigui a la vida quotidiana o en diferents tipus de projectes que actualment els enginyers dels diferents camps de la ciència estan desenvolupant.

Un dels impulsors més destacables d'aquest creixement han sigut els MCU [Figura 1.2], aquells components petits, programables, capaços d'executar ordres gravades a la seva memòria i que han permès la facilitat en els diferents dissenys electrònics i l'abaratiment dels costos de producció.

Si ens aturem i observem la quantitat d'aparells que estan dotats de MCU, diríem que vivim en un món on l'esser humà és l'estranger en comparació amb aquest components electrònics, ja que exemples d'això són: smartwatchs, mòbils, rentadores, portàtils, etc., per tant cal dir que la versatilitat d'aquests dispositius ha propiciat que siguin imprescindibles tant en l'electrònica de consum com en l'industrial.

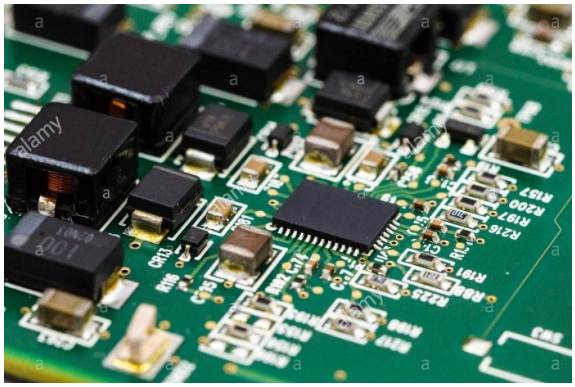

Figura 1.2: Microcontroladors

A més a més, aquests aparells/sistemes electrònics controlats per microcontroladors sovint han de fer mesures de variables analògiques del seu entorn, per exemple corresponents a la temperatura, humitat relativa, nivell de soroll, etc. Per fer aquestes mesures es precisa d'un ADC que generalment es troba integrat en el mateix xip. Malauradament, el consum d'aquests ADC integrat no està especificat (o només parcialment) en els corresponents fulls d'especificacions.

1.3. Problemàtica

En aquest apartat del projecte s'explicarà la raó principal del perquè s'ha triat aquest projecte i quins problemes s'intenten resoldre a nivell de recerca i aplicacions.És important saber que els fabricants no donen informació o bé una informació escassa sobre el consum del ADC i això es el punt de partida pel qual s'intenta buscar la principal resposta.

Sabent el punt de partida, s'ha fet un estudi previ sobre els fabricants de microcontroladors per decidir amb quins treballar, tenint en compte diferents característiques que aquests han de complir:

- Han de ser categoritzats com a MCUs de baix consum.
- Han de permetre fer les mesures d'una forma quasi directe (disponibilitat de jumpers) de corrent amb la mínima modificació de la placa.
- Han de permetre la connexió port sèrie per poder compilar i descarregar els programes al hardware corresponent.
- S'ha d'intentar que les plaques siguin de diferents fabricants per poder extreure el màxim de conclusions possibles
- Es puguin complimentar amb l'entorn de programació a baix nivell per fer treballar la placa com es desitgi.

Aquestes són les característiques que haurien de tenir les plaques de desenvolupament però cal dir que principalment s'estudiarà l'ADC, per tant les especificacions que s'haurien de complir són les següents:

- Tipus de ADC en relació amb la forma de conversió: SAR, Sigma Delta o altres.
- L'ADC ha de poder treballar en *mode Sleep* i/o *Idle* de manera que es puguin realitzar les mesures desitjades.
- Els bits de conversió de l'ADC.
- Rang de freqüències en què pot treballar l'ADC.

A la pregunta de quin microcontrolador és el millor o quin és el millor fabricant, no trobaríem resposta única, ja que depèn de quina aplicació se li donarà. Existeixen diversos fabricants i multitud de models que dificulten trobar resposta a la pregunta anterior, tot i així es poden establir diferents criteris que ens faciliten aquesta comparativa de fabricants que avui en dia es troben al mercat. Les principals empreses segons les ventes i la diversitat dels microcontroladors són:

- Intel Corp.
- Microchip Technology Corp.
- STMicroelectronics
- Texas Instruments
- Atmel Corp. (Actualment pertany a Microchip Technology Corp.)
- Freescale.

1.4. Objectius

Aquest treball consta d'un objectiu principal i d'una sèrie d'objectius secundaris necessaris per poder portar a terme el treball de manera exitosa.

1.4.1. Objectiu Principal

Tenint en compte la poca informació facilitada pels fabricants de microcontroladors en quant al consum dels ADCs integrats, l'objectiu principal d'aquest treball és estudiar-lo experimentalment en dos microcontroladors comercials diferents a fi i efecte de donar consells que permetin reduir l'energia necessària per fer una conversió ADC i d'aquesta manera, allargar el temps de vida dels nodes sensors.

Per assolir aquest objectiu cal crear un codi capaç de realitzar el control absolut del microcontrolador: fent adormir tot i cadascun dels perifèrics per seguidament activar l'ADC i fer-lo córrer de tal manera que realitzi conversions en mode lliure, és a dir, que treballi de manera contínua (*Free Running*, *Continuos Conversion*, etc.).

Una vegada s'aconsegueixi el programa adaptat en aquestes condicions s'hauria de poder controlar els registres de freqüència, tant de l'ADC com de la CPU, de tal manera que variant aquesta s'aconsegueixi veure el consum corresponent de cadascuna d'elles mitjançant un aparell de mesura.

1.4.2. Objectius Secundaris

1. Fer un estudi dels fabricants de microcontroladors que més s'adapten a la nostra necessitat per decidir amb quins MCUs treballar.

2. Seleccionar les plaques de desenvolupament segons les característiques comentades.

3. Entendre l'entorn de programació que s'utilitzarà així com familiaritzar-se amb el full d'especificacions, registres, esquemàtics i tota mena d'informació virtual i escrita que serveixi d'ajuda en tots els programes que es realitzaran al llarg del treball.

4. Crear diferents programes per aplicar el que s'ha comentat anteriorment, per poder:

    - Entendre com funcionen les entrades, sortides, ports, pins, etc.
    - Aprendre el funcionament de la comunicació serial mitjançant l'UART/USART
    - Treballar amb l'ADC i prendre lectures analògiques senzilles en un determinat port. Posteriorment ser capaços de veure-les mitjançant la consola.

1.5. Estructura de la memòria

Una vegada plantejat el problema i definits els objectius, s'explica el contingut dels diferents capítols de la memòria.

- Capítol 2: en aquest capítol s'esplicarà què és un MCU i els seus components més importants. També s'explicarà què és l'ADC juntament amb les seves característiques estàtiques i dinàmiques. S'analitzarà el tipus de MCU utilitzat, a saber el d'aproximacions succesives.

- Capítol 3: en aquest capítol s'analitzaran les característiques, l'entorn de programació, els modes de repòs i l'ADC dels MCUs utilitzats juntament amb les seves eïnes de desenvolupament.

- Capítol 4: serà el capítol més important ja que s'explicaran els resultats obtinguts mitjançant tots i cadascun dels experiments realitzats. Es veuran el registres i els bits corresponents que ens han ajudat a fer els programes.

CAPÍTOL 2. MICROCONTROLADORS I ADC

2.1. Què és un microcontrolador?

El MCU és un circuit integrat digital basat en un processador programable, utilitzat en els sistemes electrònics de control i mesura. Depenent del disseny, el MCU serà el sistema digital encarregat d'executar i fer la mesura del sensor. Tot i que es podrien utilitzar altres sistemes digitals, el MCU és el mes recomanat degut el seu cost i el consum d'energia.

Els microcontroladors tenen bàsicament tres part importants integrats: Unitat Central de processament (CPU), Memòria i els Perifèrics. [8]

2.1.1. CPU

És el nucli del MCU, és l'encarregat d'executar les instruccions de forma seqüencial, per dur a terme una feina específica. Incorpora una unitat lògica aritmètica (ALU) que processa dades (de 8, 16 o 32 bits) durant l'execució.

La potència de càlcul serà major a mesura que el processament de dades sigui de més bits, però també el consum d'energia serà major. Algunes CPUs tenen el mode repòs (*sleep*) de tal manera que suspèn la seva tasca i, per lo tant disminueix el seu consum d'energia.

2.1.2. Memòria

Està integrada en el MCU i guarda les instruccions per ser executades i les dades per ser processades.

La majoria de MCUs comercials actuals tenen una arquitectura *Harvard* amb dos memòries separades: una memòria d'instruccions (o programa) i una memòria de dades, sent la primera generalment programable una sola vegada (OTP) o tipo *flash*, mentre que la segona és la memòria d'accés aleatori (RAM).

2.1.3. Perifèrics

Els perifèrics són el recurs hardware integrats en el MCU que donen lloc a interactuar amb l'exterior del xip. Estan controlats per la CPU, però funcionen en paral·lel amb la CPU.

Els perifèrics més comuns integrats en el MCU són: ports d'entrada/sortida (E/S), temporitzadors/comptadors, ports de comunicació sèrie (per exemple: l'UART, SPI, I2C), ADC, entre altres.

2.2. ADC

La majoria de dades del món real són analògiques. Tant si es tracta de temperatura, pressió, tensió, etc., la seva variació és sempre de naturalesa analògica. Un cop mesurades aquestes dades, han de ser processades, però el processament de senyal analògic és bastant ineficient en quant a precisió, velocitat i sortida desitjada; per tant, es converteixen en un format digital mitjançant un ADC. Aquesta funció fa que els passos per obtenir el valor digital equivalent al de l'entrada analògica es realitzin de forma òptima per no perdre informació.

Segons el tipus de component i aplicació existeixen diferents paràmetres que el caracteritzen, aquest poden ser: la velocitat de conversió, la resolució, els rangs d'entrada, etc.

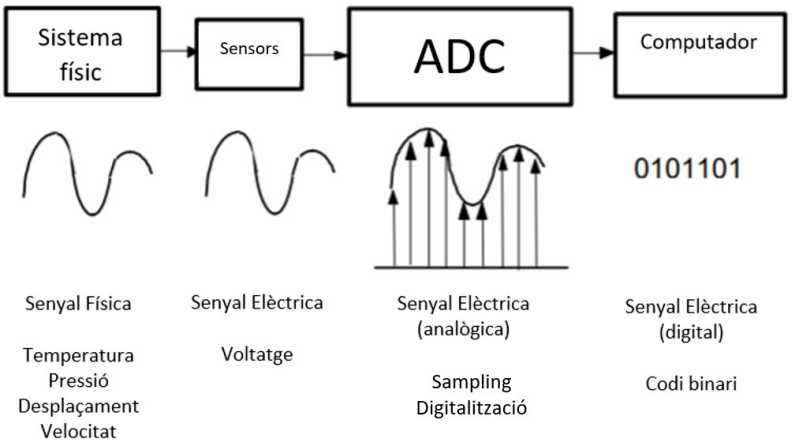

Figura 2.1: Funcionament ADC

A la Figura 2.1 es mostra la seqüència des de la senyal física fins la senyal digital (Codi binari). Per que aquesta senyal s'introdueixi en el ADC, ha de ser mostrada, és a dir, s'agafen diversos valors en períodes de temps de la senyal anàloga, aquest procés s'anomena "sampling" i com a resultat s'obté un tren de polsos amb amplituds limitades per l'envoltant de la senyal anàloga.

Per garantir la mostra i la conversió de forma correcte, s'ha de considerar la velocitat de mostreig, per tant el Teorema de Nyquist, estableix que la freqüència *sampling* (*fs*), ha de ser com a mínim el doble de la màxima freqüència d'entrada (*fm*). [Equació 2.1].

$$f_S > 2 * f_m \tag{2.1}$$

Cada ADC, està determinat per la funció de transferència de E/S [Figura 2.2], que mostra l'equivalència entre el món digital i l'analògic.

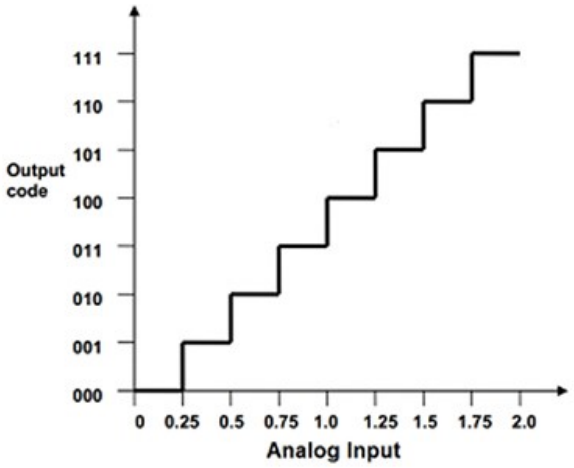

Figura 2.2: Funció de transferència d'E/S

En el cas d'una senyal unipolar entre 0-2 [V] i *n* = 3, el seu equivalent digital per n bits seria entre 000 per 0 [V] i 111 per 2 [V].

2.2.1. Característiques estàtiques

- Resolució: està expressada en unitats de tensió, dependrà del esglaó escollit com a referència respecte els nivells de tensió donats per l'últim bit. [Equació **??**] Per exemple, amb *n* bits, hi haurà 2^n nivells de tensió. A la pràctica correspon al valor del bit menys significatiu (*LSB*)

$$resolució = \frac{Fullscale}{2^n} \tag{2.2}$$

- Linealitat integral / Linealitat diferencial: si s'analitza la funció de transferència E/S, el resultat és una recta formada pels punts de transició dels valors d'entrada que determinen canvis de nivell a la sortida. L'ADC serà més precís si el comportament real s'ajusta a una recta. La màxima desviació entre la gràfica real i la recta ideal es defineix com linealitat integral. La linealitat diferencial correspon a la desviació màxima a partir de l'amplitud ideal.

- *Monotonicity*: un ADC és monotònic quan un increment de la tensió a l'entrada provoca un increment a la sortida.

- Error de guany, desplaçament i quantificació: correspon a la comparació i diferència màxima entre la curva de transferència ideal i real. L'error és un paràmetre que mostra la precisió de la funció de transferència.

- Velocitat: és important tenir una velocitat que garanteixi la conversió de manera correcte, tenint en compte el teorema de mostreig. En alguns ADC, la velocitat depèn del número de bits a la sortida.

2.2.2. Característiques dinàmiques

- Temps de conversió: és el temps des de que s'adquireix la senyal analògica fins que es proporciona el valor digital a la sortida.

- Temps d'adquisició: és el temps durant el qual, el sistema de mostreig/retenció està en estat de mostreig.

- Temps d'establiment: és l'interval de temps entre la senyal de retenció i l'establiment de la senyal.

- *Conversion rate*: és la velocitat a la qual el valor de la sortida de *Sample and hold* convergeix al valor desitjat.

2.3. SAR

Per tal que el conjunt de bits obtinguts a la sortida sigui el més exacte possible al valor analògic corresponent, s'utilitza una gran quantitat de mètodes per convertir senyals analògiques a digitals. Els més utilitzats són: Rampa d'escala, aproximacions successives, paral·lel (*flash*), doble rampa, voltatge a freqüència, *Sigma-Delta*, tipus sèrie.

Un dels mètodes més s'utilitzats degut a la seva combinació d'alta resolució i velocitat és el d'aproximacions succesives. El seu funcionament es basa en un comptador dintre del registre que compte de manera programable augmentant o disminuint en funció del bit de més pes [Figura 2.3]. La velocitat de conversió està limitada pel comparador.

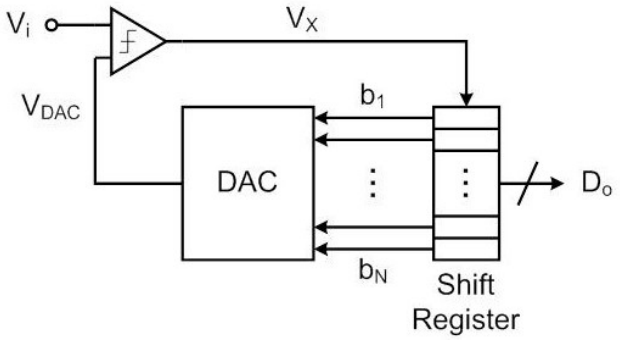

Figura 2.3: ADC mitjançant aproximacions succesives

El SAR posa el bit de major pes (MSB) a "1" i la resta a "0". El convertidor digital analògic (DAC) pren el valor digital generat internament, de tal manera que el seu equivalent analògic es compara amb la senyal d'entrada. Si la sortida del DAC és major que l'entrada, s'elimina el "1" del bit MSB i es posa a "1" el bit immediatament anterior deixant la resta a "0" i així successivament fins que s'aconsegueixi una seqüència analògica menor a l'entrada del senyal. Aquest procediment es repeteix fins que es posi a "1" tots els bits del comptador. A la Figura 2.4 es mostra la sortida característica d'aquest tipus de conversió.

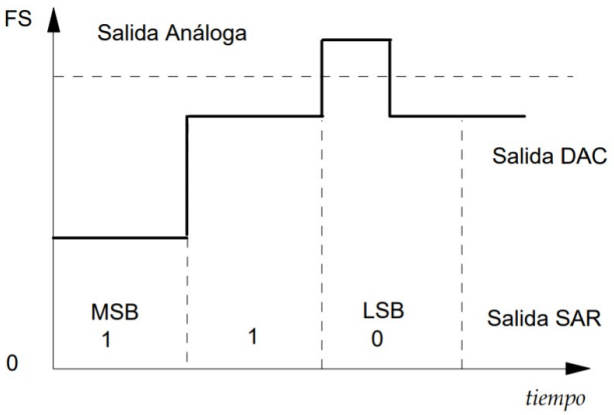

Figura 2.4: Procediment SAR

CAPÍTOL 3. MICROCONTROLADORS SELECCIONATS I EINES DE DESENVOLUPAMENT

En aquest capítol s'explicaran quins MCU s'han triat i les seves característiques.

La primera ha estat la placa de desenvolupament *ATmega324 Xplained Pro* que consta d'un MCU integrat *ATmega324PB* i la segona placa de desenvolapament és la *PIC24FJ256GA7 Curiosity Development Board* la qual té integrat el MCU *PIC24FJ256GA705*.

3.1. Atmel

En aquest apartat s'explicaran les característiques d'ATmega conjuntament amb l'entorn de programmació i es realitzarà una introducció de l'ADC del dispositiu.

3.1.1. Característiques

A continuació primer s'explicaran les característiques més destacades de la placa de desenvolupament i seguidament del MCU del fabricant Atmel.

3.1.1.1. ATmega324PB Xplained PRO

La placa *ATmega324PB Xplained Pro* és una plataforma de hardware per tal d'avaluar el MCU integrat ATmega324PB. Aquesta permet a l'usuari utilitzar els perifèrics del MCU d'una forma senzilla i comprendre de com integrar el dispositiu en un disseny propi.

Aquesta placa conté el depurador *Atmel Embedded* (EDBG) que permet compilar el programari d'una forma directa en connectar el dispositiu a l'ordinador. L'EDBG és un dispositiu USB compost de tres interfícies: un depurador, el port COM virtual i una interfície de passarel·la de dades (DGI). Juntament amb l'*Atmel Studio*, la interfície del depurador EDBG pot programar i depurar el MCU. A la placa, la interfície JTAG està connectada entre el EDBG i el microcontrolador, i el port COM virtual està connectat a una UART del MCU i proporciona una manera fàcil d'aconseguir comunicar-se amb l'aplicació d'orientació a través del programari del terminal.

La placa es pot alimentar de dues formes diferents:

- Alimentació externa: 5V +/-2% (100mV) al connector PWR.
- Alimentació per cable USB (Debug USB): alimentació de 4.4V a 5.25V depenent de les especificacions de USB.

En referència a l'estructura hardware, la placa té una o més capçaleres d'extensió de 20 pins i 100 mil files; els pins dels ports del MCU tenen capçaleres mascles. A més a més, el MCU conté altres connectors i capçaleres, dels quals els del nostre interès és l'encapçalament per mesurar la intensitat, localitzat a la vora superior de la placa a través

de la qual circula tot el corrent del dispositiu. Per mesurar el corrent s'ha de desconnectar el *Jumper*, que fa la funció de tancar el circuit entre dos pins, i posteriorment s'ha de substituir per un amperímetre.

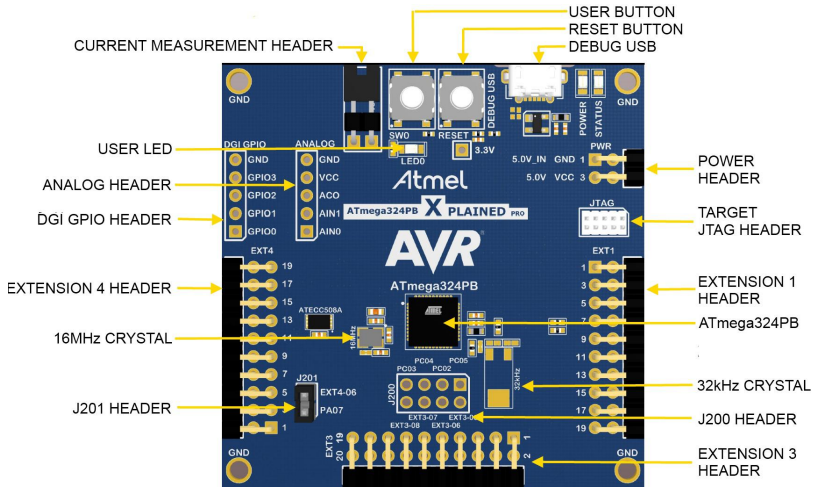

Figura 3.1: Connectors i mòduls d'ATmega324PB

La placa també inclou dos polsadors -un destinat per l'usuari per programació o diferents usos i l'altre per realitzar la funció de Reset-, dos LEDs indicadors de diferents estats i dos oscil·ladors; un de 16 MHz i l'altre de 32.768kHz, que es poden utilitzar com a rellotges del MCU. [Figura 3.1]

3.1.1.2. Microcontrolador ATmega324PB

L'Atmega324PB és un microcontrolador de 8 bits de la família Atmel (recentment comprada per Microchip) basat en l'arquitectura AVR. Es tracta d'un microcontrolador d'alt rendiment i amb un baix consum de potència.

Les característiques més significatives del MCU són:

- 1 MIPS per MHz
- Aquitectura *Harvard*
- Oscillador intern calibrat de 8 MHz
- Memòries de treball i de dades no volàtils
- Característiques dels perifèrics:
  - Dos temporitzadors de 8 bits amb programació i modes de comparació independents.
  - Dos temporitzadors de 16 bits amb programació, modes de comparació i de captura independents.
  - 6 canals PWM
  - ADC de 6 canals

  - Conversió ADC de 10 bits. Tipus SAR.
  - USART programable
  - Xip comparador analògic

- Caracteristiques especials:
  - Interupcions internes i externes
  - 6 modes de baix consum: *Idle*, *ADC Noise Reduction*, *Power-save*, *Power-down*, *Standby i Extended Standby*.
  - Circuit intern de reset sobre alimentació estable

- 44 pins d'entrada i sortida programables.
- Tensió d'operació: 1,8V - 5,5V.
- Rangs de temperatura operativa: -40^{o}C a 85^{o}C.
- Consum a 1MHz, 1.8V, 25^{o}C:
  - *Active Mode*: 0.24 mA
  - *Power-Down Mode*: 0.2 μA
  - *Power-Save Mode*: 1.3 μA

3.1.1.3. Ports d'E/S

Tots els MCUs tenen una sèrie de pins destinats a l'entrada i sortida de dades o senyals digitals. Cada pin d'un port pot ser configurat com a sortida o entrada independentment de la resta. En la Figura 3.2 s'observa que hi ha 44 pins i que aquesta placa té 5 ports (port A, B, C, D i E) que s'explicaran a continuació.

Port A [7:0]

Aquest port serveix com a entrades analògiques per l'ADC, per tant quan es treballi amb aquest, perifèric, serà el port que s'utilitzarà. És un port bidireccional de 8 bits amb resistències pull-up internes (estableixen un estat lògic en un pin o entrada d'un circuit quan aquest es troba en repòs) que es seleccionen individualment per cada pin. Si treballen com a sortida, el consum disminuirà si s'activen les resistències *pull-up*. Els pins poden tenir tres estats en una condició de reset encara que el rellotge no estigui funcionant.

Port B [7:0]

Aquest port és semblant al port A, és a dir, bidireccional, de 8 bits amb resistències *pull-up* internes que es seleccionen individualment per cada pin. Si es treballa com a sortida, el consum disminuirà si s'activen les resistències *pull-up*. A diferència del port A, aquest pot arribar a complir amb condicions especials, però aquest ja no és un port d'entrades analògiques.

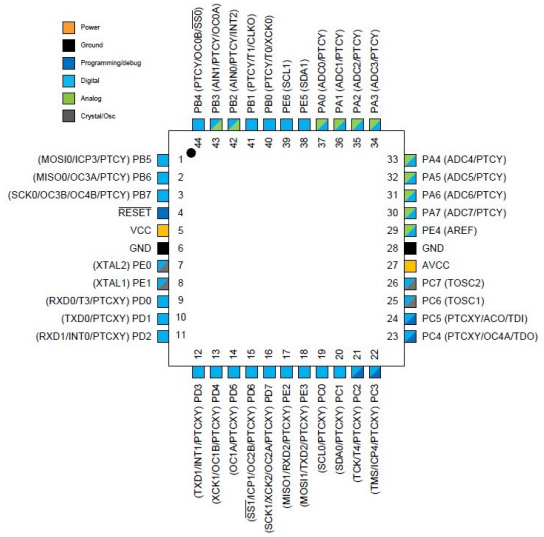

Figura 3.2: Esquemàtic Ports

Ports C/D/E [7:0]

Aquests ports igual que el Port B, són bidireccionals i de 8 bits amb resistències *pull-up* internes que es seleccionen individualment per cada pin.

3.1.1.4. Arquitectura

La família AVR utilitza una arquitectura *Harvard* amb memòries i busos diferents pel programa i les dades, de forma que maximitzen el rendiment, paral·lelisme i execució de les instruccions. Les ordres de la memòria del programa són executats amb un nivell senzill de solapament, també anomenat *pipelining*. S'executa una instrucció per a cada cicle de rellotge, gràcies a que mentre una instrucció s'està executant, la següent està sent presa de la memòria i preparada per a la seva execució.

3.1.2. Entorn de Programació

En aquest apartat del projecte s'explicarà l'entorn de programació que s'ha utilitzat per duu a terme els diferents procediments i programes que s'explicaran a continuació.

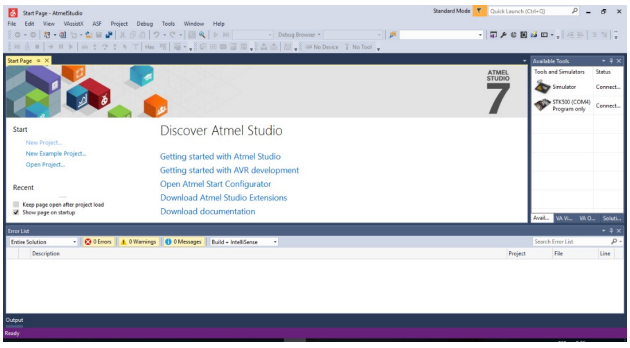

Figura 3.3: Atmel Studio

En el cas de l'*ATmega324PB*, s'ha utilitzat per programar el software *Atmel Studio* en llenguatge C [Figura 3.3]. Atmel és un dels pocs fabricants de MCU que ens proporciona de manera gratuïta totes les eines software necessàries per poder treballar amb els seus microcontroladors. Aquestes eines com podrien ser: l'editor, l'assemblador, etc., les trobem integrades en el mateix programari *Atmel Studio*.

Cal remarcar que l'objectiu d'aquest treball és controlar a un baix nivell les funcions de l'ADC, és a dir, es programa amb l'objectiu de donar instruccions utilitzant els conceptes bàsics del hardware.

3.1.3. Estudi dels modes de consum energètic

Els modes de repòs permeten desactivar els mòduls que no s'utilitzen a la placa, és a dir, tenen l'objectiu d'estalviar energia desactivant funcions innecessàries en diferents configuracions de treball. El dispositiu proporciona diversos modes que permeten a l'usuari adaptar el consum d'energia als requisits d'aplicació.

L'ATmega324PB permet treballar en 6 modes de repòs diferents:

- *Idle*: aquest tipus de mode desactiva la CPU, permetent el funcionament dels següents mòduls: CPI, la USART, el comparador analògic, la interfície sèrie de dos fils, l'ADC, els temporitzadors/comptadors, el watchdog, i el sistema d'interrupcions, és a dir, aquest mode bàsicament desactiva els rellotges de la CPU.

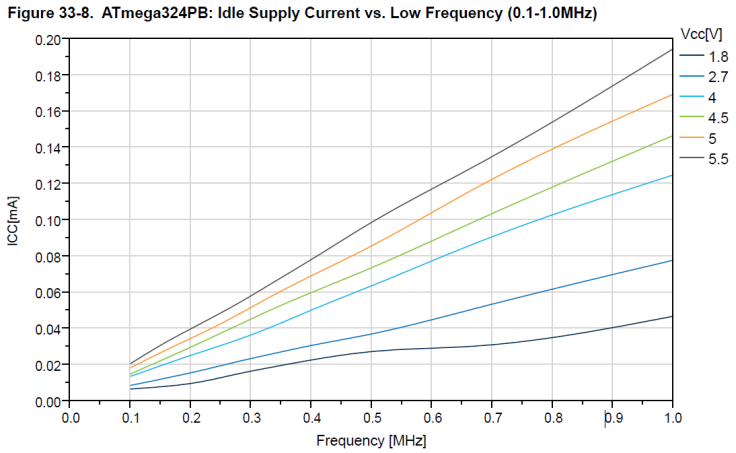

Figura 3.4: Consum en mode idle en funció de la freqüència

Tal com es mostra a la Figura 3.4 en les freqüències que treballa l'ADC, en el full d'especificacions està indicat que el consum està entre 20 μA - 140 μA. Aquests consums depènen de la tensió d'alimentació, tal com es pot veure el corrent augmenta amb la freqüència.

- *ADC Noise Reduction*: mode que permet treballar amb el mínim consum necessari utilitzant l'ADC, per lo tant es converteix en el mode que s'utilitzarà per aconseguir l'objectiu principal del treball. Aquest mode atura la CPU, però permet deixar en funcionament l'ADC, les interrupcions externes, algun timer i el wachtdog en cas d'estar habilitat. Bàsicament aquest mode deshabilita els següents rellotges: clkI/O, clkCPU and clkFLASH. Això millora l'entorn de soroll de l'ADC permetent mesuraments de major precisió. Si l'ADC està desactivat, la conversió s'inicia automàticament quant s'introdueix

aquest mode. El full d'especificacions no indica cap dada respecte al consum.

- *Power-Down*: aquest mode de repòs atura tots el rellotges generats, cosa que permet operar només amb mòduls asíncrons. Juntament amb el mode *Power-Save* és un mode que redueix completament el consum de la placa.

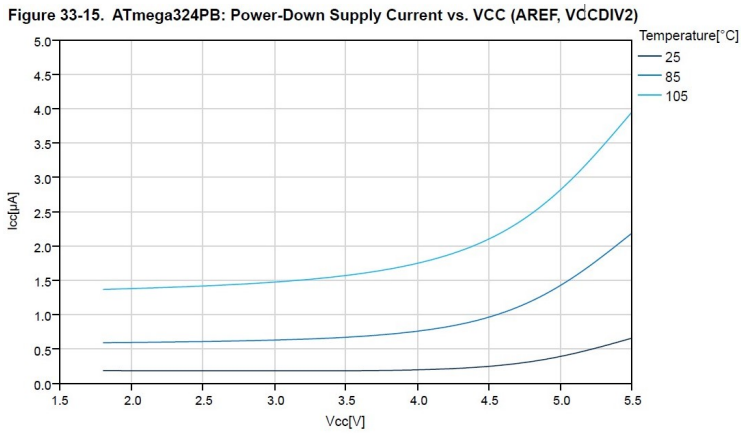

Figura 3.5: Consum del mode Power-Down en funció de Vcc

Depenent de la tensió d'alimentació, el consum d'aquest mode oscil·la entre 0.5-2.5 μA. [Figura 3.5]

Aquest consum també depèn de la temperatura ambiental variant en l'ordre de dècimes.

- *Power-Save*: és un mode de repòs similar al que s'ha esmentat anteriorment amb la diferencia que en aquest hi ha un comptador (timer) que està activat.

- *Standby / Extended Standby*: són uns modes idèntics al mode *Power-Down* amb l'excepció que en aquests l'oscil·lador segueix funcionant. En aquests modes el dispositiu es desperta en 6 cicles de rellotge.

A més a més hi ha diverses possibilitats per tal de minimitzar el consum d'energia en l'*ATmega324PB*. En general, els modes de repòs s'utilitzen per reduir el consum tant com sigui possible, tot i així es pot arribar a una reducció de consum desactivant els mòduls que segueixen funcionant. A la Figura 3.6 es mostren els diferents modes de repòs amb els mòduls que estan habilitats durant l'execució.

	Active Clock Domains						Oscillators		Wake-up Sources								Software BOD Disable
Sleep Mode	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	clk_{PTC}	Main Clock Source Enabled	Timer Oscillator Enabled	INT and PCINT	TWI Address Match	Timer2	SPM/ EEPROM Ready	ADC	WDT	USART[4]	Other I/O	
Idle			Yes	Yes	Yes	Yes	Yes	Yes[2]	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	
ADC Noise Reduction				Yes	Yes	Yes	Yes	Yes[2]	Yes[3]	Yes	Yes[2]	Yes	Yes	Yes	Yes		
Power-down									Yes[3]	Yes				Yes	Yes		Yes
Power-save					Yes	Yes	Yes[5]	Yes[2]	Yes[3]	Yes	Yes			Yes	Yes		Yes
Standby[1]							Yes		Yes[3]	Yes				Yes	Yes		Yes
Extended Standby					Yes[2]	Yes	Yes	Yes[2]	Yes[3]	Yes	Yes			Yes	Yes		Yes

Figura 3.6: Taula modes de repòs

El MCU té uns registres especials anomenats registres de reducció d'energia (PRRx) que permeten desactivar els rellotges d'una forma individual. Aquests registres s'han d'activar

abans d'entrar en el mode de repòs ja que després el MCU no permet executar aquestes instruccions.

3.1.4. Convertidor Analògic-Digital

En aquest apartat s'introduirà el perifèric ADC explicant les seves característiques principals i els seus paràmetres de funcionament.

3.1.4.1. Característiques

El MCU *ATmega328PB* té un ADC d'aproximacions successives de 10 bits. El convertidor està connectat a un multiplexor analògic de 8 canals. Permet 8 entrades de tensió unipolar corresponents als pins del port A [Figura 3.7]. L'ADC té un circuit de mostreig-retenció (*sample and hold*) per assegurar que l'estabilitat de la tensió de l'entrada durant el cicle de conversió sigui correcta.

L'ADC s'alimenta amb una tensió diferent a la del MCU a través del pin AVCC. Aquest no pot diferir-se més de 0.3V respecte a la tensió d'alimentació.

La tensió de referència pot ser interna (AREF, 1.1V o 2.56V) o bé externa (AVCC amb un condensador al pin AREF). La mínima tensió és el terra (GND) i el valor màxim és la tensió de VREF – 1 LSB.

L'ADC únicament pot realitzar una conversió al mateix temps, perquè només disposa d'un únic ADC. El resultat de la conversió es guarda en dos registres, ADCH i ADCL, això és necessari, perquè el convertidor és de 10 bits.

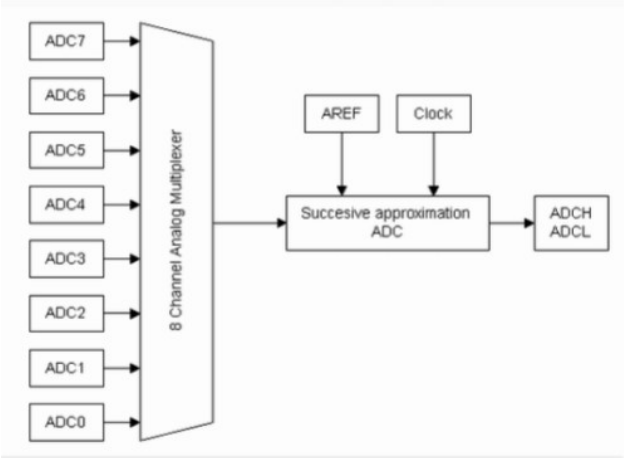

Figura 3.7: Esquemàtic ADC del ATmega324PB

Les característiques principals de l'ADC del MCU són:

- Resolució de 10 bits
- Conversió mitjançant aproximacions successives
- Temps de conversió d'entre 13-260 μs

- Fins a 15 ksps a màxima resolució
- 8 canals d'entrada
- Ajustament esquerra opcional per a la lectura de resultats de l'ADC
- Alimentació de l'ADC a 3.3 V
- Voltatges interns de referència d'ADC de 1.1V o 2.56V seleccionables
- *Free Running Mode* o Conversió individual
- Interrupció de la conversió ADC completa
- *Sleep Mode Noise Canceler*
- Marges de freqüència: 50kHz - 1MHz
- 13 cicles de rellotge per conversió

En referència a les característiques mencionades anteriorment, els 8 canals impliquen que hi ha 8 pins ADC multiplexats junts. Com es pot veure fàcilment, aquests pins es troben al porta (PA0 ... PA7). La resolució de 10 bits implica que hi ha 210 = 1024 passos.

3.1.4.2. Registres de l'ADC

L'ADC converteix la senyal analògica en senyal digital en algun interval de forma regular. Aquest interval es determina per la freqüència del rellotge. En general, l'ADC opera dins d'un interval de freqüències de 50 kHz a 1 MHz. Però la freqüència del rellotge de la CPU és molt més gran. Per aconseguir-ho, s'ha de realitzar una divisió de freqüències, per la qual cosa s'ha utilitzat el registre ADC PRESCALER.

El *prescaler* actua com un factor de divisió, produint la freqüència desitjada a partir de la freqüència del MCU. Hi ha alguns factors de divisió predefinits: 2, 4, 8, 16, 32, 64 i 128. Per exemple, un *prescaler* de 64 implica F_ADC = F_CPU / 64.

Per exemple: a F_CPU = 16MHz, F_ADC = 16 MHz /64 = 250 kHz.

A continuació es mencionen altres registres importants amb una breu explicació de la funció de cadascun, que s'han tingut en compte alhora de programar l'ADC per aconseguir els objectius.

ADMUX – Registre de selecció de multiplexor de l'ADC

Bit	7	6	5	4	3	2	1	0
	REFS [1:0]		ADLAR	MUX [4:0]				
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Figura 3.8: Bits del registre ADMUX

Bits 7:6 - REFS [1:0] – bits que s'utilitzen per triar la tensió de referència.

A continuació es mostren les possibles combinacions, de les quals per l'experiment del treball s'ha utilitzat la referència externa *AVcc with external capacitor at ARef pin*. [Figura 3.9]

REFS[1:0]	Voltage Reference Selection
00	AREF, Internal V_{ref} turned off
01	AV_{CC} with external capacitor at AREF pin
10	Internal 1.1V Voltage Reference with external capacitor at AREF pin
11	Internal 2.56V Voltage Reference with external capacitor at AREF pin

Figura 3.9: Taula de voltatges de referència

Bit 5 – ADLAR – Aquest bit serveix per ajustar a l'esquerra el resultat de l'ADC quan se li assigna un "1".

Bit 4:0 – MUX [3:0] – Segons diferents combinacions d'aquests bits es selecciona el canal desitjat (PA0...PA7).

Hi ha 8 canals analògics però hi ha altres combinacions especials, com en el cas de connexió a terra (0V - *Ground*), aquest s'ha utilitzat per obtenir els resultats d'aquest treball.

ADCSARA – Registre de control i estat A

Bit	7	6	5	4	3	2	1	0
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS [2:0]		
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Reset	0	0	0	0	0	0	0	0

Figura 3.10: Bits del registre ADMUX

Bit 7 – ADEN – *ADC Enable* – Tal com indica el seu nom, habilita el convertidor analògic digital. En cas de que aquest bit no se li assigni un 1 les operacions de l'ADC no es realitzaran a traves del PortA, és a dir, aquest port es comportarà com a pins digitals (GPIOA).

Bit 6 – *ADC conversions Start* - Se li assigna un "1"abans de començar qualsevol conversió. Un cop activat aquest bit, es comença la conversió després de la qual torna a "0". Es necessiten 13 cicles de rellotge ADC per aquesta operació, però quan es tracta de la primera conversió es necessiten 25, degut a què l'ADC s'ha d'inicialitzar.

Bit 5 – ADATE – *ADC Auto Trigger Enable* – En configurar aquest bit a 1 permet activar automàticament l'ADC.

Bit 4 – ADIF – *Indicador d'interrupcions de l'ADC* – És un bit relacionat amb les interrupcions, el qual no era el propòsit del projecte.

Bit 3 – ADIE – ADC interrupt Enable – Bit per habilitar interrupcions

Bit 2:0 – ADPS [2:0] – Prescaler ADC – Tal com s'ha explicat anteriorment aquests bits determinen el factor de divisió pel rellotge de l'ADC. [Figura 3.11]

ADPS[2:0]	Division Factor
000	2
001	2
010	4
011	8
100	16
101	32
110	64
111	128

Figura 3.11: Factors de divisió del prescaler

ADCSARB - Registre de control i estat B

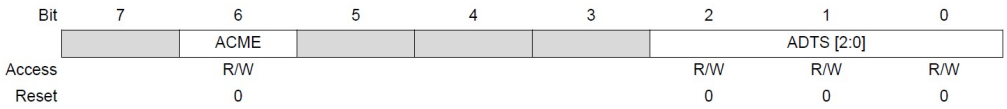

Figura 3.12: Bits del registre ADCSARB

Bit 6 – Multiplexor de comparador analògic

Bit 2:0 – ADTS [2:0] ADC Auto trigger Source – Si se li assigna un ”1”al bit ADATE del registre ADCSRA, automàticament es detecta en quin valor estan els bits ADTS, és a dir, en quin tipus de conversió es configura l'ADC. A la Figura 3.13 es troben els diferents tipus de fonts. Per l'objectiu del treball s'ha seleccionat la configuració *Free Running mode*.

ADTS[2:0]	Trigger Source
000	Free Running mode
001	Analog Comparator
010	External Interrupt Request 0
011	Timer/Counter0 Compare Match A
100	Timer/Counter0 Overflow
101	Timer/Counter1 Compare Match B
110	Timer/Counter1 Overflow
111	Timer/Counter1 Capture Event

Figura 3.13: Modes de conversió

DIDR0 – Registre de desactivació de entrades digitals

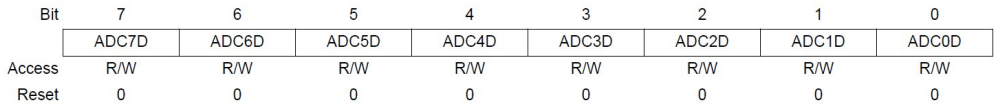

Figura 3.14: Bits del registre DIDR0

Quan els bits tenen el valor de 1 el buffer d'entrada de l'ADC es desactiva. Aquests bits per defecte tenen un zero lògic.

3.1.4.3. Funcionament de l'ADC

Per tal d'obtenir els resultats de la conversió analògica digital, l'ADC utilitza l'equació (3.1).

$$S = \frac{1024 * Vin}{Vref} \tag{3.1}$$

On *Vin* és la tensió d'entrada a través del pin seleccionat com pot ser qualsevol pin analògic del Port A; *Vref* és la tensió de referència i *S* és el valor obtingut en la conversió.

Respecte a la resolució del convertidor analògic-digital, aquesta depèn de la tensió de referència. Com que el resultat és un número entre els valors 0 i 1023, per exemple si la tensió de referencia (*Vref*) es configura a 5 Volts, això significa que el canvi de tensió més petit que es pot detectar és de 4.9 mV.

El convertidor ADC té dos modes d'operació diferents, però principalment els podem dividir en dos grans grups: conversió única i conversió continua (*Free running mode*). A la conversió única, el programa que utilitza el convertidor analògic digital ha d'iniciar la conversió cada vegada que sigui necessari. Un cop finalitzada la conversió el resultat es guarda en els registres ADCH i ADCL com s'ha comentat anteriorment. Al Bit ADIF del registre se li assigna un "1ï s'atura el convertidor.

Per començar una altre cop la conversió al Bit ADCS se li assigna un "1"en el registre ADCSR. Mentre el convertidor està treballant, aquest bit es mantindrà amb el valor "1ï quan el convertidor acabi, aquest bit passarà automàticament a "0".

Per poder realitzar la conversió, es necessita especificar el canal analògic que es vol utilitzar.

Per altre banda, tenim el mode de conversió continua (*Free running mode*), en aquest cas el programa inicia la primera conversió de la mateixa forma que la conversió única, però després no es necessari tornar a iniciar la conversió, és a dir, el convertidor realitza conversions de manera continua. Si es vol canviar de canal treballant en aquest mode s'ha de fer abans de començar la conversió.

3.2. Microchip

En aquest apartat s'expliquen les característiques de la placa de desenvolupament PIC conjuntament amb l'entorn de programació i es realitzarà una introducció de l'ADC del dispositiu.

3.2.1. Característiques

A continuació s'explicaran les característiques més destacades de la placa de desenvolupament i seguidament del MCU.

3.2.1.1. PIC24FJ256GA7 Curiosity Development Board

La placa *Curiosity PIC24FJ256GA7* és una plataforma de desenvolupament de 16 bits totalment integrada. Aquesta placa està adaptada per aquells usuaris que desitgin treballar amb MCUs per primera vegada, de manera que es podran fer prototips i aplicacions de manera ràpida i fàcil.

La placa *Curiosity PIC* s'integra perfectament amb l'ecosistema de software de Microchip que inclou *IDE MPLAB® X*, IDE basat en *Xpress de MPLAB*, compiladors X16 i configuració de codi MPLAB. Aquest entorn no requereix cap hardware addicional, ja que té un programador a la placa que el fa perfecte per treballar i explorar la família de MCUs PIC.

El disseny i les connexions externes del dispositiu ens proporcionen els diferents accessos als perifèrics del nucli (CIP's). Aquests CIPs ens permetran integrar diferents funcions del sistema en el MCU, simplificant el disseny i mantenint el baix consum d'energia del sistema. Figura 3.15

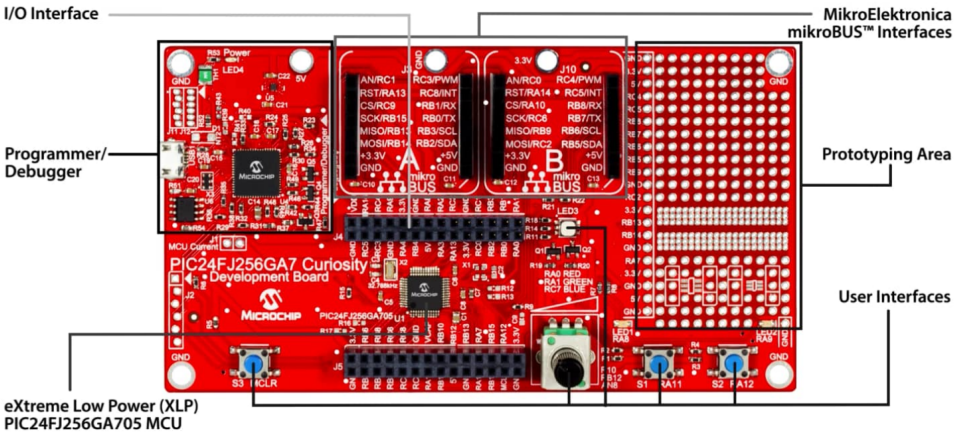

Figura 3.15: PIC24FJ256GA7 Curiosity Development Board

La placa ofereix diverses opcions per la interfície de l'usuari que inclouen interruptors, LEDs RGB i LEDs d'usuari.

3.2.1.2. Microcontrolador PIC24FJ256GA705

El MCU PIC té 16 bits i és del fabricant Microchip. Es tracta d'un MCU d'una potència extremadament baixa (XLP); això fa que sigui una placa ideal per treballar en aplicacions de baix consum com a objectiu primordial.

Les característiques més importats del MCU són:

- 0.5 MIPS per MHz
- Arquitectura *Harvard*
- Oscil·lador intern *Fast RC* de 8MHz
- Dues unitats de generació d'adreça per la lectura i escriptura de la memòria de dades

- Característiques dels perifèrics:
  - Un temporitzador de 16 bits amb un oscil·lador extern
  - Dos temporitzadors de 16 bits.
  - Tres canals PWM, cadascún amb un timer de 16 bits
  - ADC de 19 canals (interns i externs)
  - Conversió ADC de 10 o 12 bits. Tipus SAR
  - UART programable
  - 3 xips comparadors analògics
- Característiques especials
  - Interrupcions internes i externes
  - 6 modes de baix consum: *Doze*, *Idle*, *Fast Wake-up*, *Sleep*, *Fast Retention*, *Retention Sleep*
  - Circuit intern de reset
- 44 pins d'entrades i sortides programables
- Tensió d'operació: 2.0V - 3.6V
- Rangs de temperatura operativa: -40^{o}C a 125^{o}C

3.2.1.3. Ports d'E/S

A continuació s'explicaran breument els diferents ports i les possibles funcions de cadascun. [Figura 3.16]

Port A [15:0]

Tots els pins d'aquest port tenen bits de direcció de dades que es poden configurar com a entrades o sortides, es poden establir com un estat o l'altre mitjançant el registre TRISx que s'explicarà posteriorment.

Port B [15:0]

Tots els pins d'aquest port tenen bits de direcció de dades que es poden configurar com a entrades o sortides, es poden establir com a un estat o l'altre mitjançant el registre TRISx que s'explicarà posteriorment.Aquest port serveix com a entrades analògiques per l'ADC, per tant serà el port d'interès quan es realitzin experiments amb l'ADC.

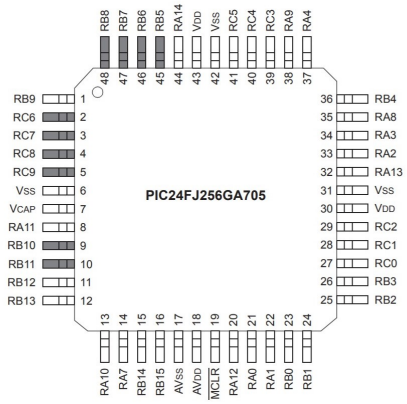

Figura 3.16: Esquemàtic Ports

Port C [15:0]

Aquest port és semblant al Port A, tots els pins d'aquest port tenen bits de direcció de dades que es poden configurar com a entrades o sortides.

3.2.1.4. Arquitectura

El MCU PIC utilitza la mateixa arquitectura *Harvard* que l'Atmel. Un processador integrat amb memòries i busos diferents pel programa. Les ordres del programa són executats amb nivell senzill de solapament, s'executa una instrucció a cada cicle de rellotge.

3.2.2. Entorn de programació

A diferència de l'entorn de programació d'*Atmel Studio* de la placa *ATmega324PB*, la placa PIC utilitza com a recurs hardware i software el programa *MPLAB X IDE*, que és un entorn de programació creat i dissenyat per Microchip, i que ens permet desenvolupar programes en C i ensamblar pels MCUs d'aquesta família.

Aquest entorn de programació es caracteritza per ser un entorn lliure, que pot ser empleat en Windows, MAC OS i Linux. Es podran realitzar feines de compilar, ensemblar i finalment pujar aquest codi a la placa.

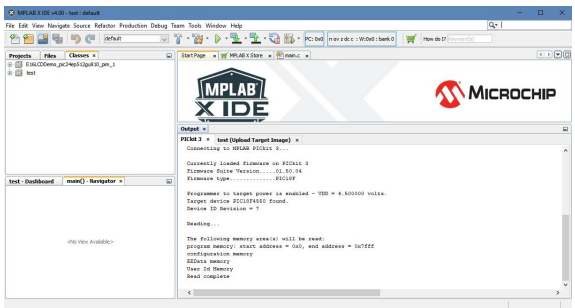

Figura 3.17: MPLAB X IDE

Com s'observa a la Figura 3.17 hi ha una zona on es mostra el contingut dels projectes, una altra zona per l'edició del codi i una zona inferior on es podran observar els problemes, sortides de la consola i variables de la consola. A la part superior es troben les icones per fer la compilació i pujar el codi a la placa. Tots aquest elements de la pantalla es poden configurar i personalitzar segons l'usuari.

3.2.3. Estudi dels modes de consum energètic

Un paràmetre important en els MCUs és el consum de corrent. Hi ha un gran nombre d'aplicacions que requereixen que el dispositiu estigui alimentat mitjançant bateries i un baix consum de corrent comporta una major duració a la bateria.

El consum de corrent en un circuit integrat depèn fonamentalment de tres factors: la tecnologia de fabricació, la freqüència de l'oscil·lador i la tensió d'alimentació.

El primer mecanisme principal per estalviar energia és la manipulació del rellotge del sistema. A continuació es mostra la Figura 3.18 del sistema oscil·lador pel nostre dispositiu.

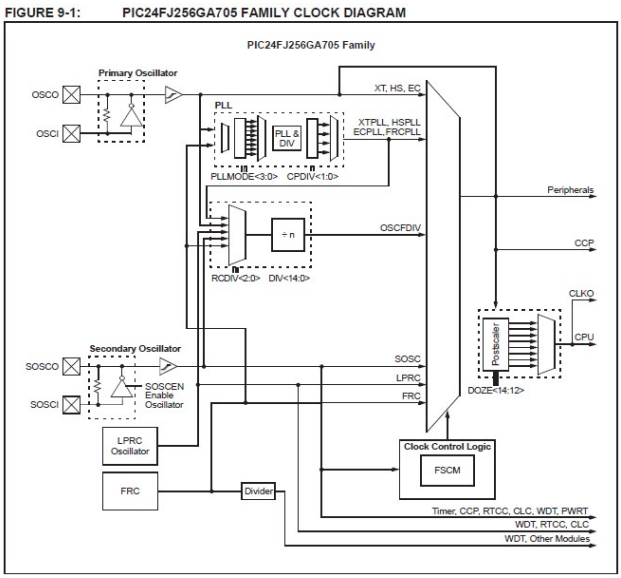

Figura 3.18: Esquemàtic funcionament dels rellotges

En el treball s'ha utilitzat el "*Fast internal Oscillator*": oscil·lador intern (RC) ràpid (7.3 o 8 MHz nominal), que es pot dividir per l'usuari. La sortida es pot dividir amb els bits RCDIV programables.

La sortida del rellotge intern pot ser dividida per a ser utilitzada com a rellotge del sistema (OSCFDIV).

Apart de la configuració del rellotge, *Microchip Technology* proporciona uns consums especificats en diferents situacions de treball. [Figura 3.19]

TABLE 10-1: LOW-POWER SLEEP MODES

RETEN	VREGS	MODE	Relative Power
0	1	Sleep	**A Few µA Range**
0	0	Fast Wake-up	**100 µA Range**
1	1	Retention Sleep	**Less than 1 µA**
1	0	Fast Retention	**A Few µA Range**

Figura 3.19: Taula de modes de repòs

- *Doze Mode*: permet que el programa d'aplicació rellenteixi el rellotge de la CPU deixant el rellotge perifèric sense canvis. Quan es canvia al mode *Doze* no cal cap ajustament als registres de configuració de perifèrics individuals. El funcionament dels perifèrics no es fa mitjançant el canvi de la velocitat del rellotge de la CPU.

- *Idle Mode*: aquest mode desactiva el rellotge de la CPU, però el rellotge perifèric és manté en funcionament. [Figura 3.20]

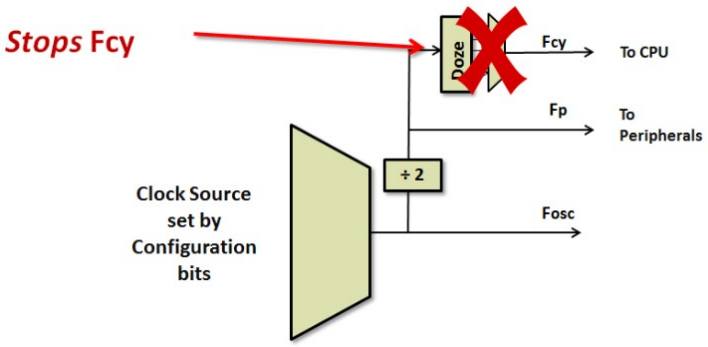

Figura 3.20: Mode Idle

L'oscil·lador del sistema es deixa d'executar i es conserven totes les ubicacions de memòria. És el mode en què es treballarà amb l'ADC per tal de fer conversions continues a diferents freqüències. Per entrar a treballar en aquest mode s'ha de posar la instrucció "Idle()". Aquest mode finalitzarà quan passin alguns d'aquests esdeveniments: un reset, una habilitació del "*Watchdog* (WTD)"o bé una interrupció amb una prioritat superior a la prioritat actual de la CPU.

- *Sleep Mode / Fast Wake up*: aquest mode de repòs desactiva el rellotge del sistema. La CPU i tots els perifèrics que requereixen aquest rellotge del sistema deixaran de funcionar [Figura 3.21. El contingut de la memòria i els registres es mantenen en mode de repòs. Aquesta instrucció s'habilita mitjançant "*Sleep()*". En funció dels registres de baixa tensió VREGS, el consum dependrà de la configuració del mode *Sleep*. Aquest mode finalitzarà quan passin alguns d'aquests esdeveniments: un reset, una habilitació del "*Watchdog (WTD)*"o bé qualsevol sol·licitud d'interrupció.

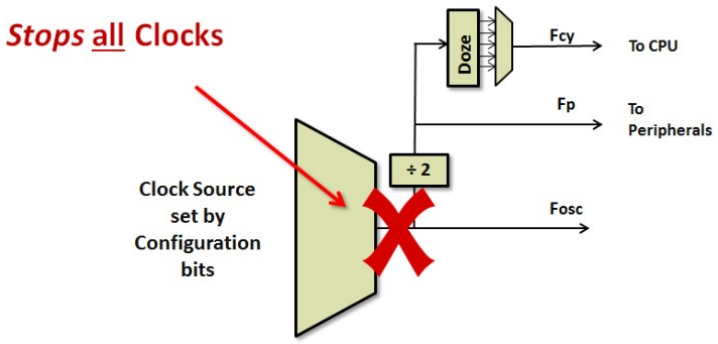

Figura 3.21: Mode Sleep

- *Retention Sleep / Fast Retention*: aquest mode atura el rellotge del sistema de la mateixa manera que el mode *sleep*. Utilitza un regulador de baixa tensió per alimentar el nucli. El mode *Retention Sleep* utilitza menys energia que el mode *Sleep*, però amb la diferència que triga més a despertar-se a causa del regulador de tensió.

  El regulador de baixa tensió està controlat pel bit de configuració LPCFG. Aquest bit fa que el regulador de baixa tensió estigui disponible per ser controlat per RCON; i la configuració de baixa tensió fa que el consum del MCU sigui diferent. Aquest mode finalitzarà quan passin alguns d'aquests esdeveniments: un reset, una habilitació del "*Watchdog (WTD)*"o bé qualsevol sol·licitud d'interrupció.

A continuació es mostra la Figura 3.22 indicant els diferents modes i les seves característiques principals.

Mode	CPU Clock (Fcy)	System Clock (Fosc)	Peripheral Clock (Fp)	All Memory Retained
Doze	Slower	Full Speed	Full Speed	Yes
Idle	Stopped	Full Speed	Full Speed	Yes
Sleep	Stopped	Stopped	Stopped	Yes
Low Voltage Sleep	Stopped	Stopped	Stopped	Yes

Figura 3.22: Taula de diferents estats dels rellotges en modes de repòs

3.2.4. Convertidor Analògic-Digital

En aquest apartat s'introduirà el perifèric ADC explicant les seves característiques principals i la configuració de funcionament.

3.2.4.1. CARACTERÍSTIQUES

Semblant al MCU d'Atmel, l'ADC PIC és del tipus SAR, però en aquest cas la resolució pot ser de 10 bits o bé 12 bits. Planteja un circuit de mostreig-retenció (*sample and hold*) per assegurar que l'estabilitat de la tensió de l'entrada durant el cicle de conversió sigui correcta.

L'ADC únicament pot realitzar una conversió al mateix temps, perquè nomes es disposa d'un únic ADC. El resultat de la conversió es guarda en els registres corresponents de la memòria.

El MCU PIC consta d'un ADC amb les següents característiques principals:

- Resolució de 10 bits o 12 bits
- Conversió mitjançant aproximacions successives
- Velocitat de conversió de 200 Ksps (12-bit)
- 19 canals d'entrades (externes i internes)
- Quatre opcions per l'alineació de la paraula. Longitud fixa (una paraula per canal)
- Alimentació 3.3 V
- Pins d'entrada de referència de voltatge extern
- *Continuos Mode* o Conversió individual
- Generació d'interrupció configurable
- Canals d'entrada de referència múltiple
- Escaneig i comparació de llindars automatitzats
- Operatiu durant CPU *Sleep Mode* i *Idle Mode*
- Màxima freqüència de treball 4MHz
- 15 cicles de rellotge per conversió

3.2.4.2. Registres de l'ADC

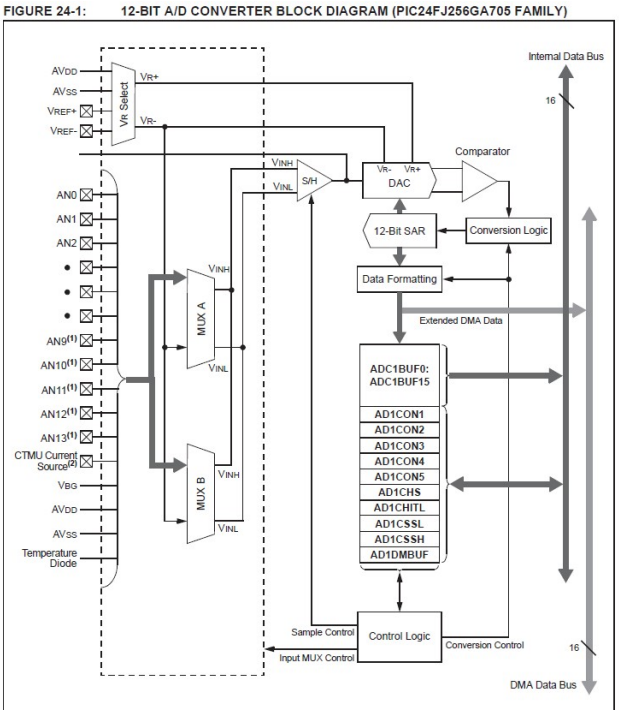

Figura 3.23: Esquemàtic ADC del *PIC24FJ256GA705*

El MCU PIC té les entrades analògiques, que es multiplexen en una única mostra com es veu en la Figura 3.23. L'ADC genera un resultat binari de 10 o 12 bits mitjançant aproximacions successives i emmagatzema el resultat de la conversió en els corresponents registres. L'ADC utilitza un voltatge de referència que es pot programar per ser generat internament o subministrada externament. També pot generar una interrupció un cop finalitzada la conversió. Aquesta interrupció es pot utilitzar per despertar el dispositiu des del *mode sleep*.

Per configurar l'ADC s'han de tenir en compte els registres següents:

AD1CHS - Registres de selecció de canals de l'ADC

REGISTER 24-6: AD1CHS: A/D SAMPLE SELECT REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NB2	CH0NB1	CH0NB0	CH0SB4	CH0SB3	CH0SB2	CH0SB1	CH0SB0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
CH0NA2	CH0NA1	CH0NA0	CH0SA4	CH0SA3	CH0SA2	CH0SA1	CH0SA0
bit 7							bit 0

Figura 3.24: Bits del registre AD1CHS

El multiplexor ADC ha ser connectat al pin d'E/S abans de començar el procés de mostra i mantenir-lo. Això es fa amb un conjunt de bits del registre AD1CHS: *ADC SAMPLE SELECT REGISTER*. Abans de sol·licitar una mostra ADC, aquests bits de selecció de canals es configuren per connectar-se al pin E/S desitjat. Només es pot connectar un pin

a l'ADC alhora i després de completar el procés, es poden canviar els bits de selecció per connectar-se al següent pin i tornar a començar el procés.

Bit 12:8 - CH0SB [4:0] *Sample B Channel 0 Positive Input Select bit*s: mitjançant aquests bits del registre AD1CHS es connecta el pin d'E/S per el canal desitjat. [Figura 3.25]

bit 12-8 **CH0SB<4:0>:** Sample B Channel 0 Positive Input Select bits
11110 = AV_{DD}(1)
11101 = AV_{SS}(1)
11100 = Band Gap Reference (V_{BG})(1)
10000-11011 = Reserved
01111 = No external channels connected (used for CTMU)
01110 = No external channels connected (used for CTMU temperature sensor)
01101 = AN13
01100 = AN12
01011 = AN11
01010 = AN10
01001 = AN9
01000 = AN8
00111 = AN7
00110 = AN6
00101 = AN5
00100 = AN4
00011 = AN3
00010 = AN2
00001 = AN1
00000 = AN0

Figura 3.25: Canals de l'ADC

AD1CON2 - Registres de selecció de voltatge de referència de l'ADC

REGISTER 24-2: AD1CON2: A/D CONTROL REGISTER 2

R/W-0	R/W-0	R/W-0	r-0	R/W-0	R/W-0	U-0	U-0
PVCFG1	PVCFG0	NVCFG0	—	BUFREGEN	CSCNA	—	—
bit 15							bit 8

R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
BUFS	SMPI4	SMPI3	SMPI2	SMPI1	SMPI0	BUFM	ALTS
bit 7							bit 0

Figura 3.26: Bits del registre AD1CON2

L'ADC integrat en el MCU PIC pot utilitzar diferents fonts de referència de tensió com a base per a mesures del voltatge analògic.

Els bits de selecció de referència de tensió es troben al registre AD1CON2 i les opcions de selecció es mostren a continuació.

Bits 15:14 - PVCFG [1:0]: Configuració de voltatge de referència positiu del ADC. Aquests bits proporcionen el control de la referència de tensió positiva. La referència de la tensió positiva pot ser:

- "01" = Externa *VREF+*
- "00" = *AVDD*

Bit 15:14 - NVCFG0: Configuració de voltatge de referencia negatiu del ADC. Aquests bits proporcionen el control de la referència de tensió negativa. La referència de la tensió negativa pot ser:

- "1" = Externa *VREF-*
- "0" = *VREF- AVSS*

VREF+ i VREF- són pins d'E/S específics del dispositiu. Es connecta una referència de tensió externa a aquests pins.

AD1CON3 - Registre de selecció de rellotge per la conversió ADC

REGISTER 24-3: AD1CON3: A/D CONTROL REGISTER 3

R/W-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADRC(1)	EXTSAM	PUMPEN(2)	SAMC4	SAMC3	SAMC2	SAMC1	SAMC0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADCS7	ADCS6	ADCS5	ADCS4	ADCS3	ADCS2	ADCS1	ADCS0
bit 7							bit 0

Figura 3.27: Bits del registre AD1CON3

El rellotge per fer la conversió es pot programar seleccionant el bit 15 (ADRC) del registre AD1CON3. En el cas de què aquest bit tingui el valor de "1" hi ha un rellotge dedicat a l'ADC treballant a una freqüència de 4MHz nominals. Pel contrari si aquest bit assoleix el valor de "0" es treballarà amb la font de rellotge que prové del sistema i que prèviament hem comentat.

El rellotge és fonamental per produir la conversió analògica-digital més ràpida i precisa. El temps per completar la conversió d'un bit es defineix com T_AD. Una conversió completa de 10 bits requereix 15 cicles de rellotge.

Per una conversió correcte, s'ha de complir les especificacions del T_AD adequada. La selecció de l'oscil·lador intern FRC farà que es realitzi un conversió més lenta però garanteix el compliment dels requisits del T_AD.

Al full d'especificacions del microcontrolador PIC ens adonem que el ADC pot treballar com a màxim a 4MHz, per lo tant mitjançant el conjunt de bits ADCS[7:0] del registre AD1CON3 s'adaptarà la freqüència de treball de l'ADC per aconseguir els objectius d'aquest treball.

INTCON 2 - Registre de Control d'interrupcions de l'ADC

REGISTER 8-4: INTCON2: INTERRUPT CONTROL REGISTER 2

R/W-1	R-0	R/W-0	U-0	U-0	U-0	U-0	R/W-0
GIE	DISI	SWTRAP	—	—	—	—	AIVTEN
bit 15							bit 8

U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	—	INT4EP	INT3EP	INT2EP	INT1EP	INT0EP
bit 7							bit 0

Figura 3.28: Bits del registre INTCON2

El mòdul de l'ADC té la capacitat de generar una interrupció després de completar la conversió analògica-digital. Aquesta interrupció també es pot generar mentre el dispositiu estigui en funcionament, en mode *Idle* o en mode *Sleep*. Si el dispositiu es troba en mode

*Slee*p, la interrupció despertarà el dispositiu i processarà la rutina de la interrupció (*ISR – Interrupt Service Routine*) sempre que els bits d'interrupció estiguin activats.

Donat que les interrupcions no s'han d'activar per complir els requisits d'aquest projecte, s'ha desactivat el bit d'interrupció global que està en el registre INTCON2, ja que en el full d'especificacions per defecte es troba activat.

AD1CON1 - Registre de Format de resultats de l'ADC

Els resultats de la conversió ADC s'emmagatzema de manera que el mòdul ADC té la flexibilitat de justificar a l'esquerre o a la dreta el resultat d'una conversió de 10 o 12 bits al registre de resultats.

Bit 9:8 - FORM [1:0] Format de bits a la sortida de dades: Aquests bits del registre AD1CON1 seleccionen el format amb el qual s'obtenen el valors convertits. El resultat pot ser copiat en una variable o utilitzat en alguna equació per implementar una funció basada en el resultat de l'ADC.

3.3. Taula comparativa de prestacions

A continuació es presenten les característiques més importants dels MCUs i dels seus ADCs corresponents.

Taula 3.1: Taula comparativa

Característiques	**PIC24FJ256GA705**	**ATmega324PB**
Bits CPU	16	8
Freqüència CPU	8 MHz	8 MHz
Tipus ADC	SAR	SAR
Bits ADC	10-12	10
Marge freq. ADC	max. 4 MHz	50 kHz - 1 MHz
Canal d'entrada utilitzat	RB13 *(1)*	GND
Tensió de referència ADC (V)	Externa	Externa
Alimentació ADC (V)	3.3 V	3.3 V
Cicles de rellotge per conversió	15	13

(1) Pin connectat a 0 V.

CAPÍTOL 4. RESULTATS EXPERIMENTALS

En aquest capítol s'explicarà el procediment seguit i els resultats experimentals obtinguts.

4.1. Instrumentació i mesures

Part imprescindible d'aquest treball ha estat utilitzar instruments de mesura per adquirir les dades de l'ADC. Cal dir que molts instruments utilitzats per fer aquestes mesures s'han estudiat al llarg de la carrera, però en les mesures definitives s'han utilitzat uns instruments més precisos dels quals es desconeixia el funcionament.

Per mesurar el corrent s'ha utilitzat el multímetre *Agilent 34410A* [Figura 4.1, aquest instrument està dotat de fins a 6 dígits de resolució.

La precisió és de 0.0030% en DC i 0.06% en AC. Pot realitzar 14 funcions de mesures, incluïdes la temperatura i mesures capacitives. Pot realitzar fins a 10.000 lectures/segon per a 5 dígits, o bé 1.000 lectures/segon a 6 dígits. En el cas de les mesures de corrent, en aquest multímetre es poden aconseguir mesures de l'ordre de nanoamperes sent aquestes molt precises. Per prendre mesures del treball, el multímetre s'ha configurat per la mesura de corrent en contínua (DC) i en 100 NPLC (*Number of Power Line Cycles*).

NPLC és un terme de mesura que indica la precisió amb què es mostra una tensió o corrent en un dispositiu. Quan es determina el valor d'un senyal de corrent continu, el soroll d'AC és freqüentment present per les línies elèctriques. Atès que aquest soroll és periòdic, la integració del senyal de CC durant un cicle pot ajudar a eliminar aquest soroll. Un PLC són 50 Hz, que equivalen a 20 ms, per tant 100 NPLC són 2 segons. Així doncs en el cas d'aquest treball, el multímetre mostra una mitjana de valors de corrent cada 2 segons.

Figura 4.1: Agilent 34410A

Per tal d'observar i comprovar el corrent consumit del MCU mentre fa les conversions contínuament, s'ha utilitzat l'oscil·loscopi del fabricant *Keysight* de la sèrie *CX3300* [Figura 4.2]. Es tracta d'un oscil·loscopi que permet veure senyals a baixa intensitat. Té un ample de banda màxima de 200 MHz amb 1GSa/s. Permet detectar corrents desde 1 nA amb un ample de banda de 100 kHz fins a 100 A amb un ample de banda d'uns 100 MHz.

Figura 4.2: Keysight sèrie CX3300

Les últimes mesures s'han fet amb una alimentació externa, ja que quan s'intentava detectar la senyal amb l'oscil·loscopi, es veia un soroll sobreposat en la senyal a causa d'alimentar la placa amb el cable USB. Per millorar aquest fet s'ha utilitzat la font externa *Keysight E3631A* [Figura 4.3] amb una tensió de sortida de 5 Volts. Aquesta font d'alimentació està dotada de tres sortides amb un rang de +/-25 V. També té la capacitat d'emmagatzemar estats de funcionament.

Figura 4.3: Keysight E3631A

En referència a la connexió d'aquests aparells amb el MCU, en el cas de l'*ATmega324PB*, tant el multímetre com l'oscil·loscopi es connecten a la capçalera "*Current measurement header*", el qual es troba a la vora superior esquerra. Per alimentar amb la font d'alimentació externa s'ha de connectar aquesta amb el connector PWR (*Power header*), precisament en els pins de 5.0 V_IN (positiu) i GND (Terra). [Figura 3.1]

Les mesures de corrent en el MCU PIC s'han fet pel *Jumper 1*: on es mesura tot el corrent que circula pel MCU. Després de modificar aquest *jumper* i trencar la connexió s'ha soldat amb l'objectiu de poder connectar el multímetre per fer les mesures. A partir d'aquest moment les modificacions dels programes i la descàrrega d'aquests s'havien de fer amb un *jumper* adaptat provocant la interconnexió del circuit. [Figura 3.15]

Donat que el MCU PIC té pins adaptats de 5 Volts i GND (Terra) s'ha pogut alimentar el MCU amb una font externa mencionada anteriorment.

4.2. Atmel ATmega324PB

Paral·lelament amb el MCU del fabricant Microchip, s'han realitzat unes proves semblants amb el MCU *ATmega324PB* per tal de poder realitzar una comparació final entre aquestes dues.

4.2.1. Desenvolupament i proves prèvies

Per començar amb la part experimental s'ha familiaritzat amb les plaques, és a dir, s'ha realitzat un aprenentatge de les eines de programació i la simulació dels dos MCUs seleccionats. En relació amb el MCU d'Atmel, de forma similar al Microchip s'han realitzat les proves següents:

1. Configuració de registres dels ports: comprovació del funcionament correcte del MCU amb diferents programes i simulació en relació amb els pins d'entrada i sortida.
2. Comprovació de canvis de freqüència de treball del MCU
3. Lectures de la conversió ADC pel terminal USART
4. Verificació i valoració dels modes de repòs

A continuació es detallarà com s'han dut a terme les proves mencionades anteriorment.

4.2.1.1. Configuració de registres dels Ports

Com s'ha explicat en l'apartat de les característiques, *Atmel AVR* és un MCU de 8 bits. Tots els seus ports tenen una amplada de 8 bits. Cada port té 3 registres associats a cadascun d'ells. Cada bit d'aquests registres configura els pins del port en particular. El bit 0 d'aquests registres està associat amb el pin 0 del port, bit 1 d'aquests registres està associat amb pin 1 del port i així successivament amb els altres bits.

Aquests tres registres són els següents:

(x es pot substituir per A, B, C, D i E)

- Registre DDRx
- Registre PORTx
- Registre PINx

Registre DDRx

DDRx (*Data Direction Register*) configura la direcció de les dades dels pins de port. Significa que el seu paràmetre determina si els pins del port s'utilitzaran per a l'entrada o la sortida. L'escriptura d'un "1"a un bit de DDRx fa que el pin del port corresponent es comporti com una entrada, mentre que l'escriptura d'un "0"a un bit de DDRx fa que el pin del port corresponent sigui una sortida.

Exemple:

Per fer que tots els pins del port A siguin pins d'entrada: DDRA = 0b00000000;

Per fer que tots els pins del port A siguin pins de sortida: DDRA = 0b11111111;

Registre PINx

PINx (Port IN) s'utilitza per llegir dades de pins. Per tal de llegir les dades del pin d'un port, primer s'ha de canviar la direcció dels pins del port com a entrades. Això es fa si es fixen els bits de DDRx a zero. Si el port es configura com a sortida, el registre PINx proporciona les dades que s'emeten als pins del port.

Exemple:

Per llegir dades del port A:

DDRA = 0x00; - S'estableix el port A com a entrada

x = PINA; - Es llegeix el contingut del port a

Registre PORTx

PORTx s'utilitza per a dos propòsits.

1) Per la sortida de dades: quan el port està configurat com a sortida, s'estableixen els bits de DDRx a "1", els pins corresponents es converteixen en pins de sortida, permetent l'escriptura de dades en els respectius bits del registre PORTx. Això canviarà immediatament l'estat dels pins de sortida segons les dades que s'hagin escrit. En altres paraules, per emetre dades als pins del port, aquestes s' han d'escriure al registre PORTx. Tanmateix, no s'ha d'oblidar d'establir la direcció dels pins com a sortida. [Figura 4.4]

2) Per activar / desactivar les resistències *pull-up* quan el port es configura com a entrada.

	BITS								
BYTE	7	6	5	4	3	2	1	0	
255	1	1	1	1	1	1	1	1	BIN
	F				F				HEX
0	0	0	0	0	0	0	0	0	BIN
	0				0				HEX

REGISTROS DE DIRECCION								
DDRA	D7	D6	D5	D4	D3	D2	D1	D0
DDRD	ENTRADAS				SALIDAS			
DDRC	0	0	0	0	1	1	1	1

PUERTO								
PORTA	A7	A6	A5	A4	A3	A2	A1	A0
PORTD	D7	D6	D5	D4	D3	D2	D1	D0
PORTG	G7	G6	G5	G4	G3	G2	G1	G0
SALIDA	INICIA CERO				INICIA UNO			
	0	0	0	0	1	1	1	1
ENTRADAS	ALTA IMP				PULL UP			
	0	0	0	0	1	1	1	1

Figura 4.4: Taula resum dels registres per la configuració del ports

Aquests registres s'han utilitzat per realitzar diferents programes per encendre/apagar LEDS i per comprovar que el programa realitzi una compilació correcte en la placa.

4.2.1.2. Comprovació de canvis de freqüència de treball del MCU

Utilitzant un dels pins de la placa, i donant-li un valor lògic de “1” i “0” de manera simultània, s’ha pogut veure que mitjançant el registre *CLKPR* la freqüència d’aquest pin variava tal com s’ha vist connectant el pin a l’oscil·loscopi. [Figura 4.5]

Figura 4.5: Senyal observat en el pin analitzat

Al controlar aquest registre i sabent que l’ADC només pot treball en un rang de freqüències baixes, es podrà adaptar aquesta freqüència amb les especificacions de l’ADC i amb el seu factor de divisió propi (*PRESCALER*). El programa relacionat amb aquestes proves es troba a l’Annex A.

4.2.1.3. Lectura dels valors al terminal mitjançant l’USART

Aquesta prova es basa en poder realitzar una comunicació sèrie del MCU amb l’ordinador, és a dir, de realitzar una connexió entre aquests dos dispositius per tal d’enviar i rebre dades. El perifèric del MCU que permet interconnectar; enviar i rebre dades s’anomena USART (*Universal Synchronous and Asynchronous serial Receiver and Transmitter*). El MCU d’Atmel està dotat de 3 instàncies de USART: USART0, USART1, USART2.

Si s’analitza la secció USART del full d’especificacions del MCU, s’hi troben diverses característiques i funcions llistades. Algunes de les característiques principals de l’AVR USART són:

- Operació *Full Duplex* (Registres de recepció i transmissió de sèrie independents)
- Operació asíncrona o síncrona
- Operació sincrònica en ordre principal o esclau
- Generador de velocitat de transmissió d’alta resolución (*BAUD generator*)
- Suporta marcs de sèrie amb 5, 6, 7, 8 o 9 bits de dades i 1 o 2 Bits de *Stop*

Per configurar l'USART s'han seguit els pasos següents:

- El primer pas és establir la velocitat de transmissió. [Figura 4.6]

```
//--- F_CPU & BAUD RATE ---//

#define F_CPU       8000000 //--- CPU Clock Frequency
#define BAUD 9600       //--- Transmission Baud Rate
#define MYUBRR (F_CPU/16/BAUD-1)
#include "util/delay.h"
```

Figura 4.6: Configuració de freqüència i BaudRate

- El següent pas és inicialitzar l'USART amb un seguit de registres per habilitar el transmissor i el receptor; i seleccionar els bits de les dades amb què s'operarà. [Figura 4.7]

```
void usart_init()
{
    UBRR1H = (MYUBRR>>8);
    UBRR1L = MYUBRR;
    UCSR1B = (1<<TXEN) | (1<<RXEN);     //--- Transmit and Receive Enable
    UCSR1C = (1<<USBS) | (3<<UCSZ0);    //--- 8 bit data and UCSRC is selected
}
```

Figura 4.7: Inicialització de l'USART

- Posar el *b uffer*uffer a punt. En cas de transmissió (des de la placa a l'ordinador), s'ha de carregar-lo amb les dades que s'enviaran, mentre que en cas de recepció, s'han de desar les dades anteriors, perquè es puguin sobreescriure les noves dades rebudes. [Figura 4.8]

```
void usart_tx(int x)
{
    while (!(UCSR1A & (1<<UDRE)));      //--- Check whether UDR is empty
    UDR1 = x;                           //--- Move data to UDR Data Reg
}

int usart_rx()
{
    while (!(UCSR1A & (1<<RXC)));       //--- Check whether Receive is complete
    return(UDR1);                       //--- Return Data from UDR
}
```

Figura 4.8: Configuració del transmissor/receptor

- A continuació, s'activa el transmissor / receptor segons l'ús desitjat.

Un cop s'ha aconseguit un programa bàsic per funcionar el perifèric USART, s'ha donat el pas d'experimentar amb l'ADC, així doncs s'han realitzat lectures analògiques d'un pin analògic connectat amb un potenciòmetre. Aquestes dades són convertides per l'ADC i mitjançant l'USART són transmeses al Terminal propi d'Atmel Studio, és a dir, les dades són processades i enviades a l'ordinador. El programa relacionat amb aquest experiment es troba a l'Annex A.

4.2.1.4. Verificació i valoració dels modes de repòs

Com s'ha comentat en l'apartat dels modes de repòs, tenim sis diferents modes, que mitjançant diferents combinacions dels bits SM0, SM1 i SM2 del registre *SMCR* s'aconseguirà treballar en diferents modes com es mostra en la Figura 4.9.

```
//Selecció de SLEEP MODE:
SMCR = (0<<SM0)|(0<<SM1)|(0<<SM2); // Idle
SMCR = (1<<SM0); // ADC Noise Reduction
SMCR = (1<<SM1); // Power Down
SMCR = (1<<SM0)|(1<<SM1); // Power Save
SMCR = (1<<SM1)|(1<<SM2); // Standby
SMCR = (1<<SM0)|(1<<SM1)|(1<<SM2); // Extended Standby
```

Figura 4.9: Registres modes de repòs

L'estudi del consum dels diferents modes, s'ha realitzat amb l'objectiu d'entendre i controlar els diferents modes de repòs, per conseqüentment i amb l'analisis del full d'especificacions de l'ADC trobar el mode que millor s'adapta a les necessitats del ADC.

4.2.2. Funcionament del programa principal

Un cop realitzades diverses proves prèvies comprovant diferents mòduls i registres del MCU s'ha realitzat el programa final per tal de poder fer les proves de consum. Aquest programa s'estructura principalment en dos funcions, la inicialització de diverses llibreries i un *main*, on s'executen aquestes funcions i altres instruccions necessàries que s'explicaran a continuació.

Com es mostra a la Figura 4.10, a l'inici s'ha partit de dues funcions que es basen en unes configuracions que s'explicaran seguidament. Llavors s'han executat en el main les funcions i instruccions mostrades a la figura mencionada.

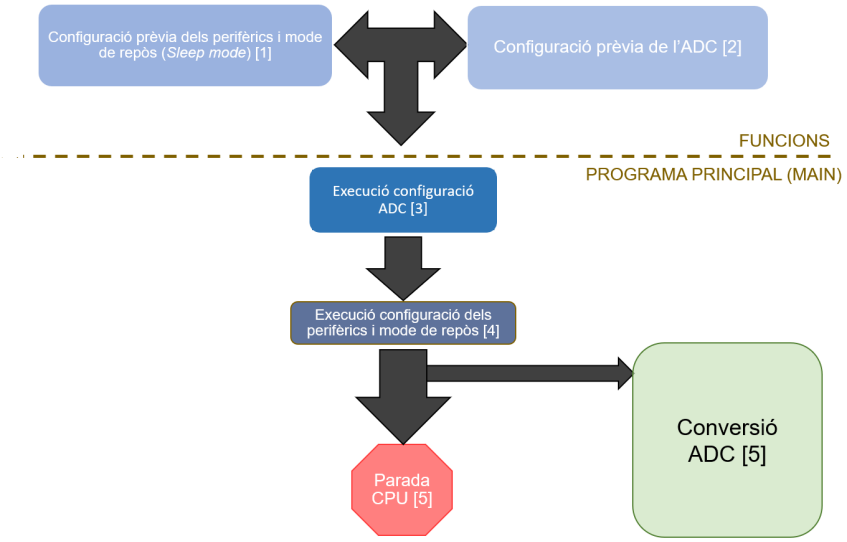

Figura 4.10: Fluxograma Atmel

La primera funció es basa en diverses configuracions relaciones amb la reducció de consum i en configurar el mode de repòs (*ADC Noise Reduction*) [Figura 4.11]:

- Registres DIDRx: desactiven el port A com a entrades digitals.
- Registres DDRx: configuren els ports com a entrades/sortides
- Registre MCUCR: desactiva les interrupcions i el JTAG.
- Registre WDTCSR: configuració del *Watchdog* (desactivació)
- Registres PRRx: desactiven diversos mòduls com timers, els usarts, etc. El bit PRADC desactiva el mòdul relacionat amb el ADC.
- Registre SMCR: selecció de mode de repòs; quan al bit SM0 se li assigna un "1", es selecciona el mode *ADC Noise Reduction*.

```
void sleepconfig()
{
    //Deshabilitar entrades digitals als diferents PORTS
    DIDR0 = (1 << ADC5D) | (1 << ADC4D) | (1 << ADC5D) | (1 << ADC4D) | (1 << ADC3D) | (1 << ADC2D) | (1 << ADC1D) | (1 << ADC0D); //Disable digital input buffer on ADC pins
    DIDR1 = (1 << AIN1D) | (1 << AIN0D); // Disable digital input buffer on Analog comparator pins */
    ACSR |= (1 << ACD); //Disable Analog Comparator
    DDRA = 0x00; // Conf. pins PORT A com a entrades
    DDRB = 0x00; // Conf. pins PORT B com a entrades
    DDRC = 0x00; // Conf. pins PORT C com a entrades
    DDRD = 0x00; // Conf. pins PORT D com a entrades
    DDRE = 0x00; // Conf. pins PORT E com a entrades
    PORTA = 0xFF;  // Conf. pins PORT A com Entrades Pull-up
    PORTB = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    PORTC = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    PORTD = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    PORTE = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    //Deshabilitar interupcions
    MCUCR = 0b01110001;
    MCUCR = 0b01010010;
    //Deshabilitar Watchdog
    MCUSR &= ~(1<<WDRF);
    WDTCSR |= (1<<WDCE) | (1<<WDE);
    WDTCSR = 0x00;
    //Registres de reducció de consum
    PRR0 = (1<<PRTWI0) | (1<<PRTIM2) | (1<<PRTIM0) | (1<<PRUSART1) | (1<<PRTIM1) | (1<<PRSPI0) | (1<<PRUSART0)| (1<<PRADC);//Power Reduction Register 0
    PRR1 = (1<<PRTIM4) | (1<<PRTIM3); //Power Reduction Register 1
    PRR2 = (1<<PRPTC) | (1<<PRUSART2) | (1<<PRSPI1) |(1<<PRTWI1); //Power Reduction Register 2
    //Selecció Sleep Mode:
    SMCR = (1<<SM0); // ADC Noise Reduction

}
```

Figura 4.11: Funció 1: *sleepconfig*

La segona funció és un seguit de registres que configuren l'ADC de tal forma que un cop activat, realitzi conversions de manera contínua [Figura 4.12]:

- ADMUX: es selecciona el canal *GND* com a canal d'entrada a convertir i la tensió de referència externa.
- ADCRA: amb la combinació dels 3 bits de ADPx s'aconsegueixen els diferents factors de divisió i el bit ADEN activa l'ADC.
- ADCSRB: en aquest mode es selecciona el *Free Running Mode*, que permet a l'ADC realitzar conversions de forma contínua.

```
void adc_init()     //FUNCIÓ PER CONFIGURAR L'ADC
{

    ADMUX = 0b01011111;     // Selecció de la tensió de referència (Externa) i el canal a llegir (GND)

    // Habilitar ADC i Divisió de freqüència (PRESCALER)
    ADCSRA = (1<<ADEN)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE);               //8 [1000 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADATE);                        //16 [500 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS0)|(1<<ADATE);             //32 [250 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADATE);             //64 [125 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE);  //128 [62.5 kHz]

    ADCSRB = 0b00000000; //FREE RUNNING MODE
}
```

Figura 4.12: Funció 2: *adc_init*

Aquestes funcions comentades anteriorment, es troben al final del programa; per començar el programa es defineixen o s'inclouen dues llibreries: *io.h* i *sleep.h*. La primera és una llibreria general dels registres del MCU, en canvi la segona és una llibreria concreta dels modes de repòs, que inclou diferents instruccions relacionats amb aquests modes.

Una vegada es defineixen les llibreries, s'inicialitzen o es defineixen les dues funcions que s'han creat, per tal de poder utilitzar-les posteriorment. [Figura 4.13]

```
#include <avr/io.h>    //Llibreria General
#include <avr/sleep.h> //Llibreria Registres de modes Sleep

//Inicialització funcions:
void adc_init(void);    // Funció Configuració ADC
void sleepconfig(void);  // Funció per configurar modes de consum i registres de reducció de consum
```

Figura 4.13: Llibreries

A continuació es troba la part principal del programa (Main) [Figura 4.14], en el qual primer s'executen les funcions *adc_init* i *sleepconfig* per tal de configurar l'ADC i els registres relacionats amb el consum. Un cop realitzades les configuracions corresponents, s'activa l'ADC assignant un "1 lògic als bits ADEN i ADSC del registre ADCSRA. En aquest moment, el MCU comença a convertir, però en un estat actiu. Per tant, seguidament s'activa el bit SE del registre SMCR i s'executa l'instrucció *sleep_cpu()* per activar el mode de repòs.

```
int main(void)
{

    adc_init(); // Executació de la configuració ADC
    sleepconfig(); //Executació Sleep and power consumption registers

    ADCSRA |= (1<<ADEN) | (1<<ADSC); // Inici de conversió

    SMCR  |= (1<<SE); // Posar en mode repòs
    sleep_cpu();    // Instrucció d'execució de Sleep Mode


}
```

Figura 4.14: Main

4.3. Microchip PIC24FJ256GA705

En primer lloc s'explicarà el desenvolupament i les proves prèvies, llavors es comentarà el funcionament del programa principal amb els resultats obtinguts.

4.3.1. Desenvolupament i proves prèvies

El procediment que s'ha realitzat per l'aprenentatge de les eines de programació i la simulació dels dos MCUs ha sigut:

1. Configuració dels registres dels Ports: comprovació del funcionament correcte del MCU amb diferents programes i simulació en relació amb els pins d'entrada i sortida.
2. Comprovació de canvis de freqüència de treball del MCU
3. Lectures de la conversió ADC pel terminal UART
4. Verificació i valoració dels modes de repòs

4.3.1.1. Configuració de registres dels Ports

Els ports són els perifèrics més simples i a la vegada els més importants, ja que ens permeten connectar la unitat de control del MCU, amb la resta de perifèrics. A més a més, alguns d'ells tenen la característica de poder combinar funcions, pel que ens aporta flexibilitat a l'hora de programar amb aquesta placa.

La majoria dels pins E/S es poden utilitzar com a entrada analògica o com a entrada digital. Quan es converteix la senyal analògica mitjançant l'ADC, els pins d'E/S s'ha de configurar com a entrada analògica establint els bits associats al registre TRISx i el registre ANSELx. El registre TRISx del pin E/S ha de tenir el seu bit associat en "1" per convertir-lo en una entrada. [Figura 4.15]

TABLE 11-1: CONFIGURING ANALOG/DIGITAL FUNCTION OF AN I/O PIN

Pin Function	ANSx Setting	TRISx Setting	Comments
Analog Input	1	1	It is recommended to keep ANSx = 1.
Analog Output	1	1	It is recommended to keep ANSx = 1.
Digital Input	0	1	Firmware must wait at least one instruction cycle after configuring a pin as a digital input before a valid input value can be read.
Digital Output	0	0	Make sure to disable the analog output function on the pin if any is present.

Figura 4.15: Configuració dels pins de l'ADC

REGISTER 11-3: TRISx: OUTPUT ENABLE FOR PORTx REGISTER(1)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TRISx<15:8>							
bit 15							bit 8

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TRISx<7:0>							
bit 7							bit 0

Figura 4.16: Bits del registre TRISx

El següent pas és establir el bit al registre ANSELx del pin d'E/S i donar-li el valor de 1 per activar l'ADC en aquest pin. [Figura 4.17]

REGISTER 11-7: ANSELx: ANALOG SELECT FOR PORTx REGISTER(1)

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSELx<15:8>							
bit 15							bit 8

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
ANSELx<7:0>							
bit 7							bit 0

Legend:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Figura 4.17: Bits del registre ANSELx

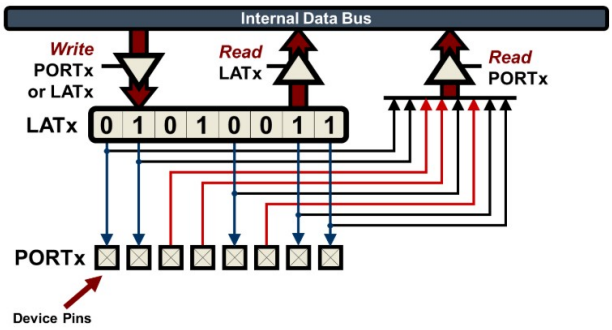

Figura 4.18: Escriptura i Lectura

4.3.1.2. Comprovació de canvis de freqüència de treball del MCU

Un dels passos previs per utilitzar els registres per aconseguir l'objectiu principal, era tenir clar com treballar amb els registres relacionats amb el canvi de freqüència [Figura 4.19] . A continuació es mostra un programa senzill en què s'ha configurat el pin RC1 del corresponent PORTC.

REGISTER 24-1: AD1CON1: A/D CONTROL REGISTER 1

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADON	—	ADSIDL	DMABM(1)	DMAEN	MODE12	FORM1	FORM0
bit 15							bit 8

R/W-0	R/W-0	R/W-0	R/W-0	U-0	R/W-0	HSC/R/W-0	HSC/R/C-0
SSRC3	SSRC2	SSRC1	SSRC0	—	ASAM	SAMP	DONE
bit 7							bit 0

Legend:	C = Clearable bit	U = Unimplemented bit, read as '0'	
R = Readable bit	W = Writable bit	HSC = Hardware Settable/Clearable bit	
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Figura 4.19: Bits del registre AD1CON1

Com es veu en aquesta part del programa, s'ha configurat l'oscil·lador mitjançant el bit RCDIV del registre CLKDIV i sense permetre cap efecte en les interrupcions mitjançant el bit ROI del mateix registre anterior. [Figura 4.20]

Com hem comentat anteriorment l'objectiu d'aquest codi senzill, però molt important és poder dominar el canvi de freqüència i això s'ha aconseguit mitjançant el bit DIV del registre OSCDIV.

```
/*
 * File:   Cambio_frecuencia.c
 */

#include "xc.h"

int main(void) {

    TRISA = 0x0000;
    TRISB = 0x0000;
    TRISC = 0x0000;

    CLKDIVbits.RCDIV = 0b000;
    CLKDIVbits.ROI = 0;
    //OSCDIVbits.DIV = 0b0000000000101000; //40 (50KHz)
    //OSCDIVbits.DIV = 0b0000000000010100; //20 (100KHz)
    //OSCDIVbits.DIV = 0b0000000000000100; //8 (250KHz)
    //OSCDIVbits.DIV = 0b0000000000000010; //4  (500KHz)
    OSCDIVbits.DIV = 0b0000000000000001; //2  (1MHz)
```

Figura 4.20: Programa principal del canvi de freqüència de la CPU

Posteriorment i fent canvis simultanis de valors lògics entre ”1ï ”0”al pin seleccionat, es pot observar la senyal, i segons la modificació del bit DIV del registre OSCDIV, es veurà que també hi ha una variació en el canvi de freqüència que pateix el MCU.

4.3.1.3. Lectura dels valors al terminal mitjançant l'UART

L'UART és el mòdul de comunicacions serial asíncron i/o síncron: en el primer cas, no hi ha necessitat de tenir un rellotge de sincronització, a diferència del segon tipus que sí que és necessari tenir un rellotge que sincronitzi el temps de comunicació que té el MCU. En el nostre cas la comunicació és asíncrona utilitzant dues terminals: el Tx i Rx. Aquests dos terminals són els que serviran per transmetre i rebre la informació respectivament. El MCU PIC està dotat de dues UARTs.

Aquest módul de transmissor/receptor asíncron universal (UART) és una de les E/S en sèrie disponibles en el dispositiu PIC. Té la característica de que pot rebre i transferir dades al mateix temps i utilitzar un canal de comunicació asíncron per comunicarse amb dispositius perifèrics i/o computadors utilitzant protocols RS-232 i RS-485.

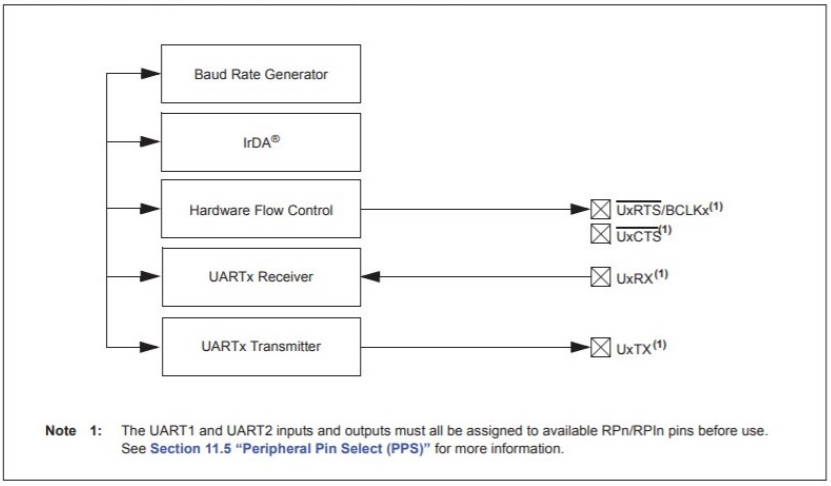

Figura 4.21: Diagrama de blocs

Pel correcte funcionament d'aquest mòdul cal fer el càlcul de la variable "*Baud Rate*" [Equació (4.1)] El registre UxBRG [Figura 4.21] controla el període d'un temporitzador de 16 bits de llibre execució; dit d'una altre manera, s'estableix la velocitat amb la què es transmeten i es rebran els bits.

$$BaudRate = F_{cy}/4 * (UxBRG + 1) \tag{4.1}$$

Segons l'equació, la màxima velocitat possible és:

$$BaudRate = F_{cy}/4 \tag{4.2}$$

I el mínim *Baud Rate* (4.3)

$$BaudRate = F_{cy}/4 * (65536) \tag{4.3}$$

Per habilitar l'UART s'ha de configurar el bit d'activació de la recepció UARTEN i el bit d'activació de la transmissió UTXEN. Quan ja el tenim habilitat, els pins UxTX i UxRX es configuren com a sortida i entrada respectivament. El terminal UxTX adquireix el valor lògic de "1"quan no s'està fent cap transmissió.

El programa relacionat amb la transmissió de dades via UART es troba a l'Annex B

4.3.1.4. Verificació i valoració dels modes de repòs

Com s'ha comentat en el l'apartat dels modes de repòs, tenim sis diferents modes, que mitjançant diferents combinacions dels bits VREG i RETEN, del registre *RCON* s'aconseguirà treballar en diferents modes com es mostra en la Figura 3.19.

L'estudi del consum dels diferents modes, s'ha realitzat amb l'objectiu de entendre i controlar els diferents modes de repòs, per conseqüentment i amb l'analisis del full d'especificacions de l'ADC trobar el mode que millor s'adapta a les necessitats del ADC.

4.3.2. Funcionament del programa principal

En aquesta part s'explicaran els registres que s'han utilitzat per donar lloc a complimentar amb el projecte.

Com es mostra a la Figura 4.22, a l'inici s'ha partit de dues funcions que es basen en unes configuracions que s'explicaran seguidament. Llavors s'han executat en el main les funcions i instruccions mostrades a la figura mencionada.

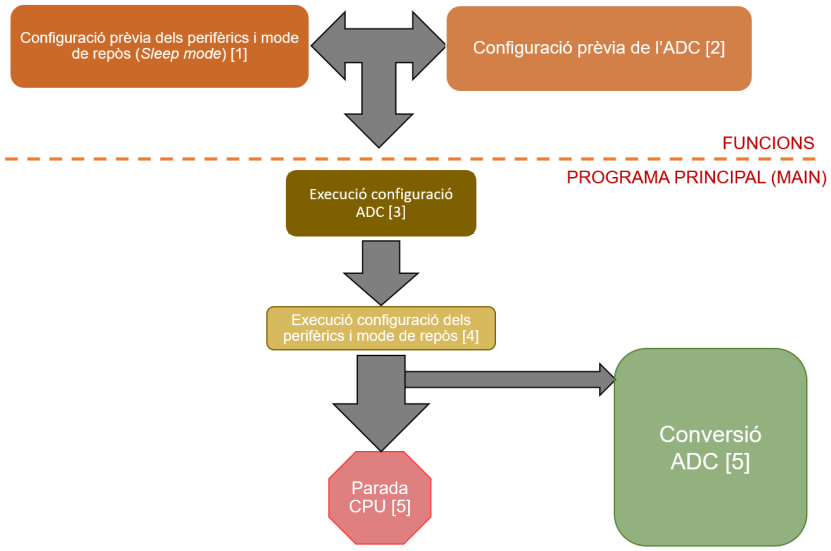

Figura 4.22: Fluxograma Microchip

- S'afegeixen les llibreries, perquè ens permetran utilitzar les funcions, registres, configuracions etc.. [Figura 4.23]

```
2  #include <xc.h>
3  #include <stdio.h>
4  #include <stdint.h>
```

Figura 4.23: Llibreries

- S'afegeixen les funcions utilitzades

- Desactivació dels perifèrics (Funció 1): Per desactivar els perifèrics han de ser configurats amb el valor lògic de "1". S'han desactivar tots els perifèrics de tal manera que el consum de corrent de l'ADC no es vegi afectat per cap perifèric. [Figura 4.24]

```
57  void disable_perifericos()
58  {
59    // Se desactivan todos los timer de la máquina
60    PMD1bits.T3MD = 1;
61    PMD1bits.T2MD = 1;
62    PMD1bits.T1MD = 1;
63
64    // Desactiva el I2C (comunicador serial sincrono)
65    PMD1bits.I2C1MD = 1;
66    PMD3bits.I2C2MD = 1;
67
68    // Se desactivan las UART
69    PMD1bits.U2MD = 1;
70    PMD1bits.U1MD = 1;
71
72    // Se desactivan los modulos SPI (comunicador serial sincrono)
73    PMD1bits.SPI2MD = 1;
74    PMD1bits.SPI1MD = 1;
75    PMD6bits.SPI3MD = 1;
76
77    // Se desactivan los modulos por captura
78    PMD2bits.IC3MD = 1;
79    PMD2bits.IC2MD = 1;
80    PMD2bits.IC1MD = 1;
81
82    PMD2bits.OC3MD = 1;
83    PMD2bits.OC2MD = 1;
84    PMD2bits.OC1MD = 1;
85    // Se desactivan los comparadores
86    PMD3bits.CMPMD = 1;
87
88    // Se desactiva el Puerto paralelo
89    PMD3bits.PMPMD = 1;
90
91    // Activa el ADC
92    PMD1bits.AD1MD = 0;
93  }
```

Figura 4.24: Funció Desactivació de perifèrics

- Configuració de l'ADC (Funció 2): "ADC_Configuración" permetrà configurar l'ADC segons els requeriments d'aquest treball. [Figura 4.25]

  - Bit CH0SB (reg. AD1CHS)
  - Bit LPEN (Reg. AD1CON5): adquirint el valor de "1" ens permet treballar en baix consum després del escaneig.
  - Bit PVCFG (Reg. AD1CON2): control de la referència de tensió positiva Vref+
  - Bit NVCFG0 (Reg. AD1CON2): control de la referència de tensió negativa Vref-
  - Bit ADSIDL (Reg. AD1CON1): configurat a "0" permet al ADC treballar estan en mode Idle.
  - Bit SSRC (Reg. AD1CON1): "0b011" el ADC entrarà en un estat en el qual contínuament estarà convertint
  - Bit ASAM (Reg. AD1CON1): "1" començarà el mostreig de manera automàtica
  - Bit ADRC (AD1CON3): "0" utilitzarà el rellotge del sistema seleccionat
  - Bit ASEN (AD1CON5): "1" s'activa l'auto escaneig

```
41  void ADC_Configuracion()
42  {
43      ANSA = 0b0000000100000000;
44      AD1CHSbits.CH0SB = 0b01101;
45      AD1CON5bits.LPEN = 0;
46      AD1CON2bits.PVCFG = 0b01;
47      AD1CON2bits.NVCFG0 = 1;
48      AD1CON1bits.ADSIDL = 0;
49      AD1CON1bits.SSRC = 0b0111;
50      AD1CON1bits.ASAM = 1;
51      AD1CON3bits.SAMC = 1;
52      AD1CON2bits.ALTS = 0;
53      AD1CON3bits.ADRC = 0;
54      AD1CON5bits.ASEN = 1;
55  }
```

Figura 4.25: Funció Configuració de l'ADC

- Programa principal (*Main*): Aquest programa té diferents instruccions explicades al llarg del projecte, però s'ha de destacar que mitjançant el bit anomenat DIV del registre OSCDIV es faran les divisions de freqüència tenint en compte el període dels 15 cicles del rellotge. [Figura 4.26]

    - Es criden les funcions utilitzades i comentades anteriorment.
    - Es configuren els ports com a entrades
    - Es configuren els rellotges
    - Es desactiven les interrupcions globals mitjançant el bit anomenat GIE del registre INTCON2
    - Es configura la freqüència de l'ADC
    - S'activa el mostreig en l'ADC mitjançant el bit 2 (SAMP) del registre AD1CON1.
    - S'activa l'ADC: permet que l'ADC s'activi i realitzi totes les configuracions esmentades anteriorment
    - Mode *Idle* apaga el rellotge a la CPU (Fcy), però el rellotge perifèric (Fp) es manté en funcionament, per aquesta causa l'ADC pot continuar treballant i fer conversions de manera contínua.

```
9   int main(void)
10  {
11      disable_perifericos();
12      ADC_Configuracion();
13      TRISA = 0b0000000100000000;
14      TRISB = 0x0000;
15      TRISC = 0x0000;
16
17      CLKDIVbits.RCDIV = 0b000;
18      CLKDIVbits.ROI = 0;
19      //OSCDIVbits.DIV = 0b0000000000101000; //40 (50KHz)
20      //OSCDIVbits.DIV = 0b0000000000010100; //20 (100KHz)
21      //OSCDIVbits.DIV = 0b0000000000000100; //8 (250KHz)
22      //OSCDIVbits.DIV = 0b0000000000000010; //4  (500KHz)
23      OSCDIVbits.DIV = 0b0000000000000001;   //2  (1MHz)
24
25      INTCON2bits.GIE = 0;
27      //AD1CON3bits.ADCS = 0b00100000;   //62,5KHz
28      //AD1CON3bits.ADCS = 0b00010000;   //125KHz
29      //AD1CON3bits.ADCS = 0b00001000;   //250KHz
30      //AD1CON3bits.ADCS = 0b00000100;   //500KHz
31      //AD1CON3bits.ADCS = 0b00000010;   //1MHz
32      AD1CON3bits.ADCS = 0b00000001;     //2MHz
33      AD1CON1bits.SAMP = 1;              //muestreo
34      AD1CON1bits.ADON = 1;              //Convertidor A/D trabajando
35
36      delay();
37      Idle();
38      return 1;
39  }
```

Figura 4.26: Programa principal

4.4. Resultats i Discussió

En aquest subcapítol es comentaran els resultats obtinguts, en primer lloc s'introduiran els resultats individuals de cadascun dels dos MCUs per separat i després es realitzarà una comparativa entre ells.

$$I_T = I_{s,m} + I_{s,ADC} + I_{d,ADC} \tag{4.4}$$

En referència als resultats que es mostraran en aquest apartat, s'ha d'aclarir que el corrent total és la suma dels corrents estàtics del mode de baix consum, de l'ADC i del corrent dinàmic de l'ADC, tal com es mostra a l'Equació 4.4.

4.4.1. Atmel

El resultats de la mesura de corrent del MCU *ATmega324PB* en els diferents Sleep Modes a una freqüència de 8 MHz es mostren a la Taula 4.1. Tal com s'esperava i ens indica el document d'especificacions del MCU, el mode amb el menor consum és el *Power-Down*, amb un consum de corrent de 307.9 nA; a diferència del mode *Idle*, que és el de major consum amb 783.82 μA.

Taula 4.1: Consum d'Intensitat en diferents modes de repòs

Idle	783.83 μA
ADC Noise Reduction	526.5 μA
Power Down	307.9 nA
Power Save	308.2 nA
Standby	541.5 nA
Extended Standby	540.2 nA

Com s'ha explicat anteriorment, l'objectiu és analitzar l'ADC en un dels modes de menor consum, per tant es descarta la possibilitat d'utilitzar els modes *Power* i *Standby*, ja que ens limiten el funcionament del ADC en un estat completament adormit. Així doncs, s'ha utilitzat el mode A*DC Noise Reduction*, el primer mode de baix consum que permet realitzar conversions amb l'ADC. A causa d'utilitzar aquest mode, s'obté un *offset* d'intensitat de 526.5 μA amb tots el perifèrics desactivats. Aquest consum relacionat amb els modes de baix consum és el corrent estàtic *Is,m* de l'Equació 4.4.Pel fet d'activar el mòdul que conté el ADC, aquest consum s'incrementa als 629.5 μA. Arribats en aquest punt, es pot concloure que el consum del ADC és de 103 μA, que equival al corrent estàtic *Is,ADC* de l'Equació 4.4.

Els resultats experimentals quan l'ADC està convertint contínuament i està funcionant a diferents valors de freqüència es mostren a la Taula 4.2.

Com s'il·lustra a la Taula 4.2 el consum és entre 812.5 μA a 62.5 kHz i 834.9 μA a 1 MHz, que únicament analitzant la diferència del corrent entre les dues freqüències extremes s'observen uns canvis notablement reduïts.

A mesura que la freqüència augmenta, també ho fa la intensitat, seguint gairebé una tendència lineal, excepte pel valor més petit de la freqüència, tal com es mostra a la Figura 4.27.

Taula 4.2: Consum d'intensitat per diferents valors de freqüències

Freqüència (kHz)	Intensitat (μA)
62.5	812.5
125	815.8
250	818.4
500	823.7
1000	834.9

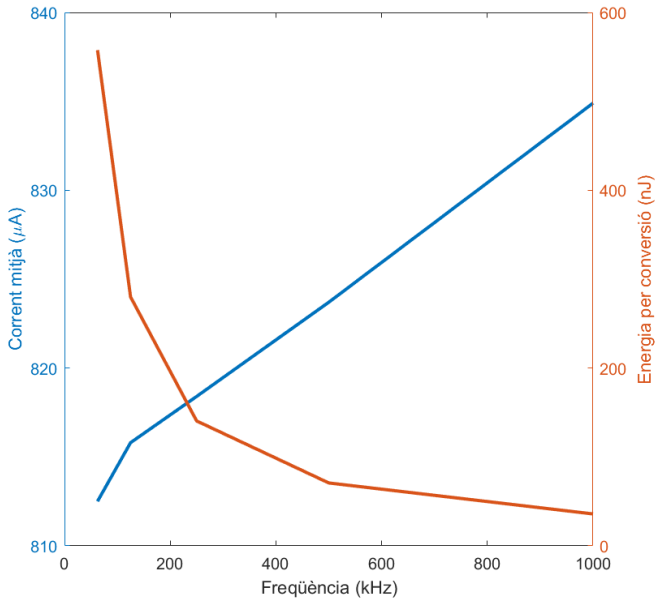

Figura 4.27: Energia per conversió i la intensitat envers a la freqüència

Tanmateix, la discussió del paràgraf anterior no implica que el consum d'energia per unitat de temps augmenti amb la freqüència, sabent que únicament la variació lineal amb la freqüència és pels resultats de la intensitat i no de l'energia. Aquesta energia per conversió es defineix a l'Equació (4.5).

$$E = \frac{I * V * n}{f} \tag{4.5}$$

I: intensitat mesurada en l'instrument de mesura

V: Voltatge d'alimentació de l'ADC

n: Nombre de cicles de rellotge per cada conversió

f: Freqüència de l'ADC

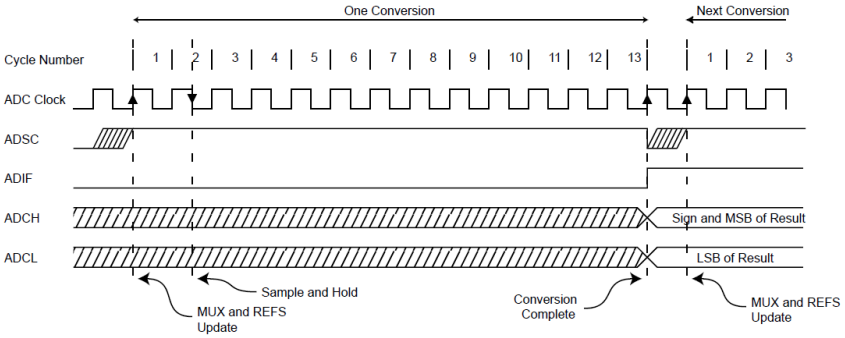

Figura 4.28: Cicles de rellotge per una conversió de l'ADC

Així doncs, tenint en compte que la placa s'alimenta a 3.3 V i que la conversió precisa 13

cicles [Figura 4.28] de rellotge, realitzant el càlcul de l'energia per conversió en diferents freqüències s'obté la Figura 4.27.

Analitzant els resultats de l'energia per conversió conjuntament amb la intensitat per diferents freqüències, es veu que la càrrega decreix de 553.92 nJ treballant a 62.5 kHz fins a 35.52 nJ a 1 MHz. Considerant els resultats, l'energia per conversió té una tendència inversament proporcional, independentment del valor del corrent en diferents freqüències.

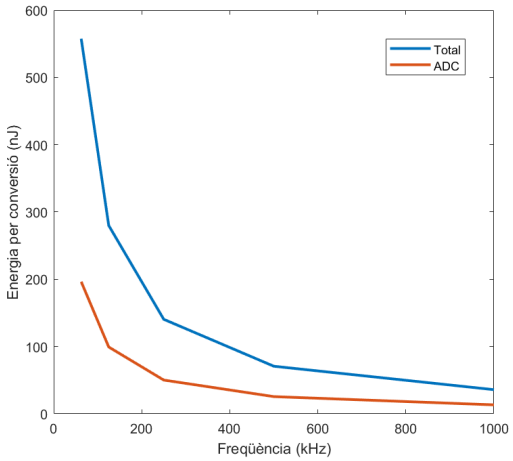

Figura 4.29: Energia per conversió envers la freqüència sense considerar l'offset

Amb l'objectiu de veure la comparació de l'energia únicament de l'ADC envers l'energia total del MCU s'ha obtingut la Figura 4.29. Tal i com es veu coherentment, el consum d'energia de l'ADC pel fet de realitzar la conversió en diferents freqüències és molt petit comparat amb el consum total del MCU.

Per tal d'assegurar que l'ADC fa les conversions a diferents freqüències, s'ha analitzat la forma de la senyal de la intensitat mitjançant l'oscil·loscopi. En provocar un retard a la senyal de 200 ms abans d'activar l'ADC, s'ha pogut veure que en els primers microsegons s'executen les instruccions per diferents configuracions i es posa en marxa l'ADC. Un cop executades les instruccions, al recórrer els 200 ms s'executa el mode *ADC Noise Reduction*. [Figura 4.30

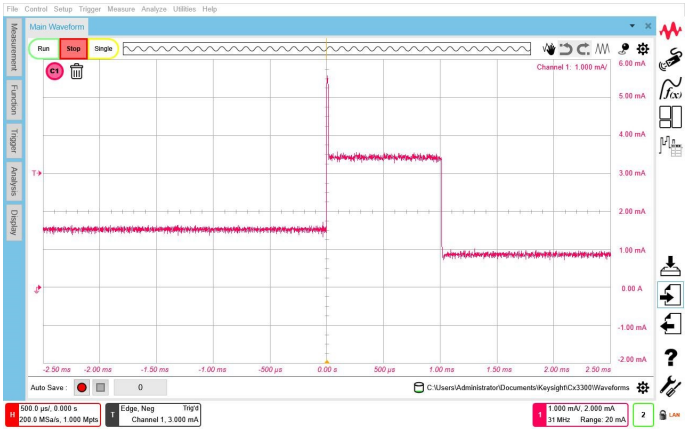

Figura 4.30: Provocant el reset i afegint un delay de 200 ms abans de l'ADC. Alimentació USB 3.3 V.

Amb la curiositat de poder veure la forma de la senyal de la conversió i el comportament de l'ADC per diferents freqüències s'ha monitoritzat la senyal de corrent a l'oscil·loscopi.

Al connectar la placa a una alimentació externa de 5V, s'ha evitat la sobreposició del soroll generat pel port sèrie, obtenint unes formes més clares de la senyal de conversió de l'ADC.

Pel cas d'una freqüència de 62.5 kHz s'ha vist una periodicitat cada 208 μs (13 clocks x 1/f) [Figura 4.31] i una repetició de pics cada 16 μs [Figura 4.32] Anàlogament per les freqüències de 125 kHz i 250 kHz s'ha vist una repetició cada 8 μs [Figura 4.33] i 4 μs [Figura 4.34] respectivament.

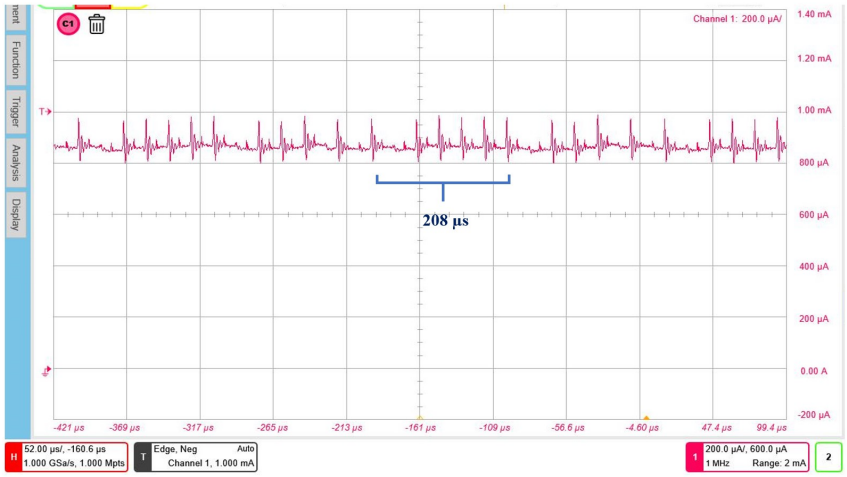

Figura 4.31: ADC a 62.5 kHz. Pics separats 16 μs. Periodicitat cada 13·16=208 μs

Figura 4.32: ADC a 62.5 kHz. “Activitat” cada 16 μs

Figura 4.33: ADC a 125 kHz. “Activitat” cada 8 μs.

Figura 4.34: ADC a 250 kHz. “Activitat” cada 4 μs.

4.4.2. Microchip

Els resultats de la mesura de corrent del microcontrollador *PIC24FJ256GA705* en els diferents modes de repòs a una freqüència de 4 MHz es mostren a la Taula 4.3. A diferència de la placa Atmel, no s’especifica clarament el consum dels diferents modes; però els resultats obtinguts mostren que el mode de menor consum és el *Fast Retention*, amb un consum de corrent de 0.7 μA; a diferència del mode *Idle*, que és el de major consum amb 305.7 μA

Taula 4.3: Consum d’Intensitat en diferents modes de repòs

Idle	305.7 μA
Sleep	44.87 μA
Fast Wake-up	5.31 μA
Sleep retention	2.87 μA
Fast retention	0.70 μA

Els modes en els quals es pot activar l’ADC per tal de convertir per diferents valors de freqüències són els modes *Sleep* i *Idle*. El mode *Sleep* és el més convenient en respecte el consum de corrent però presenta l’inconvenient de que s’ha de treballar amb un rellotge concret en el qual dificulta els canvis de freqüència per l’ADC. Per tant, s’han realitzat les mesures activant el mode *Idle*. A causa d’utilitzar aquest mode, s’obté un offset d’intensitat de 308.7 μA amb tots el perifèrics desactivats, aquest corrent és el corrent estàtic degut al mode de consum (*Is,m*) de l’Equació 4.4. Pel fet d’activar el mòdul que conté el ADC, aquest consum s’incrementa als 311.7 μA. Arribats en aquest punt, és pot concloure que el consum de l’ADC és 3 μA, corrent estàtic degut a l’ADC, (*Is,ADC*) de l’Equació 4.4. Els resultats experimentals quan l’ADC contínuament està convertit i està funcionant a diferents valors de freqüència es mostren a la Taula 4.4.

Taula 4.4: Consum d'Intensitat per diferents valors de freqüencies

Freqüència (kHz)	Intensitat (μA)
62.5	312
125	312.5
250	313.6
500	315.2
1000	317.6
2000	320.9

Com s'il·lustra, el consum és entre 312 μA a 62.5 kHz i 320.9 μA a 2 MHz, que únicament analitzant la diferència del corrent entre les dues freqüències extremes s'observen uns canvis extremadament reduïts, amb una diferència de 8.9 μA. A mesura que la freqüència augmenta, també ho fa la intensitat, seguint una tendència menys lineal que en el cas de l'Atmel, tal com es mostra a la Figura 4.35.

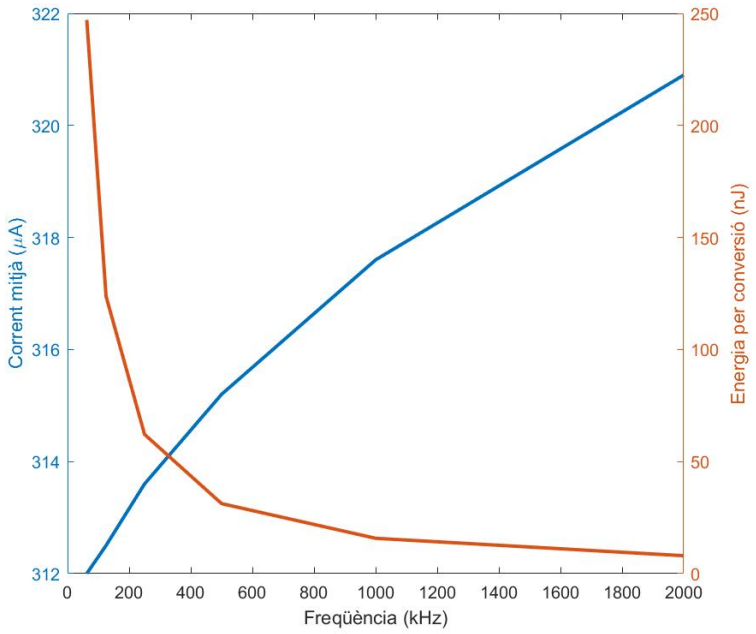

Figura 4.35: Energia per conversió i el corrent mitjà envers a la freqüència

A la Figura 4.35 també s'observa el càlcul d'energia per conversió en diferents freqüències, a diferència d'Atmel, en el PIC la conversió precisa cada 15 cicles de rellotge, però igualment l'ADC s'alimenta a 3.3V.

Analitzant els resultats de l'energia per conversió conjuntament amb la intensitat per diferents freqüències, es veu que l'energia decreix de 247.10 nJ treballant a 62.5 kHz fins a 7.94 nJ a 2 MHz. S'observa que l'energia per conversió té una tendència inversament proporcional, tal com s'esperava.

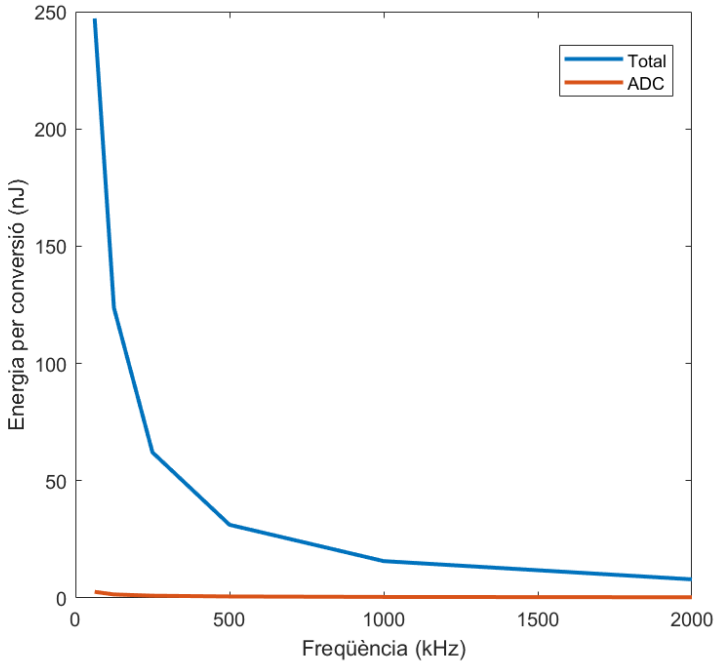

Figura 4.36: Energia per conversió envers la freqüència sense considerar l'offset

Amb l'objectiu de veure la comparació de l'energia únicament de l'ADC envers l'energia total del MCU s'ha obtingut la Figura 4.36.

El comportament de la senyal de la intensitat s'observa en la Figura 4.37; aquest anàlisi mitjançant l'oscil·loscopi mostra un retard a la senyal de 1.5 ms abans d'activar l'ADC. En els primers microsegons s'executen les instruccions per diferents configuracions i es posa en marxa l'ADC. Un cop executades les instruccions s'executa el mode *Idle*.

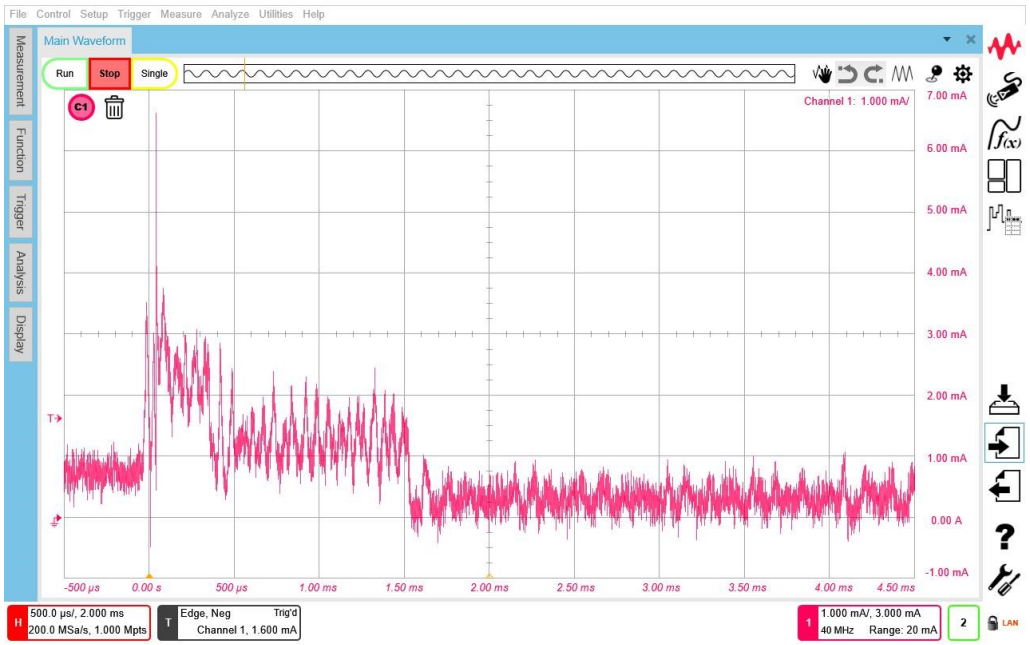

Figura 4.37: Senyal de conversió

4.4.3. Comparació

Un dels objectius d'aquest treball com s'ha comentat anteriorment és poder fer una comparació per analitzar el comportament del ADC en les diferents MCUs.

- Modes de repòs

Els dos MCUs treballen amb uns modes de repòs semblants però amb una nomenclatura diferent. En referència al consum de corrent del MCU PIC permet desactivar els perifèrics de tal forma que el consum arriba a ser menor que el del MCU *ATmega*. En els modes *ADC Noise Reduction* i *Idle* en què s'han realitzat les proves, veiem que l'Atmel presenta un consum de 783.82 μA, en canvi el PIC té un consum de 305.7 μA; és a dir, un 39% menor que l'Atmel.

- Consum ADC

Si es compara el consum del corrent estàtic de l'ADC (*Is,ADC*) pel fet d'estar activat sense realitzar cap conversió, es veu que l'ADC del MCU PIC presenta millors prestacions ja que únicament té un consum de 3 μA, en canvi l'ADC del MCU Atmel té un consum de 102 μA. La diferència entre les freqüències extremes en el cas del PIC és de 5.6 μA molt menor en comparació amb els 22 μA de l'Atmega. En el cas d'Atmel es veu un comportament més lineal en comparació amb el MCU de Microchip. [Figura 4.39]

	Intensitat (µA)		Energia per conversió (nJ)	
Freqüència (kHz)	Atmega324PB	PIC24F	Atmega324PB	PIC24F
62.5	808	312	553.92	247.10
125	810	312.5	277.99	123.75
250	813	313.6	139.51	62.09
500	819	315.2	70.27	31.20
1000	830	317.6	35.52	15.72

Figura 4.38: Valors numèrics consum Atmel - Microchip

- Energia per conversió

Com s'aprecia en la comparació anterior en el consum de corrent, en referència a l'energia per conversió, es veu un comportament similar obtenint unes gràfiques d'una forma inversament proporcional en ambdós casos. [Figura 4.39].

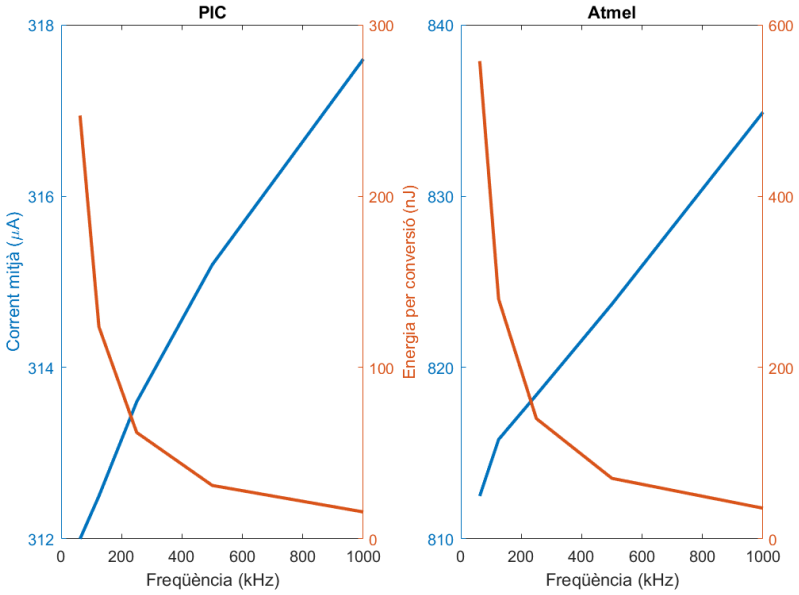

Figura 4.39: Comparació consum Atmel - Microchip

Vist els resultats i el comportament dels dos MCUs s'esdevé una conclusió important respecte al consum, donat que a freqüències altes el corrent mitjà és major, però això no implica que l'energia per conversió consumida per el MCU suposi el mateix comportament. Tot i que el corrent mitjà augmenta amb la freqüència de treball, el temps per fer la conversió és menor, i en conseqüència, l'energia requerida també és menor. Així doncs, a mesura que la freqüència augmenta, l'energia per conversió disminueix. Per tant, segons els microcontroladors estudiats és millor treballar a freqüències altes si es desitja una conversió energèticament eficient.

CONCLUSIONS

Una vegada fet aquest treball de final de grau podem extreure les següents conclusions. En primer lloc, hem sigut capaços de:

- Evaluar diferents microcontroladors comercials de baix consum

- Seleccionar i entendre el funcionament de dues plaques de desenvolupament comercials

- Aprendre noves eines de programació aplicades als microcontroladors seleccionats

- Entendre l'arquitectura interna d'aquests microcontroladors a nivell de perifèrics i dels corresponents registres, i dels modes de baix consum

El coneixement indicat en els punts anteriors ens ha permès assolir l'objectiu principal, és a dir, conèixer el consum energètics dels ADC integrats en els microcontroladors seleccionats. Tot i que el consum de corrent augmenta amb la freqüència de treball, el temps per fer la conversió és menor i, en conseqüència, l'energia requerida també és menor. Per tant, segons els microcontroladors estudiats, sembla recomanable fer treballar l'ADC a freqüències altes si es desitja una conversió energèticament eficient. Segons els resultats experimentats l'ADC integrat en el microcontrolador PIC necessita aproximadament la meitat d'energia de la de l'Atmel.

Aquesta informació sobre com millorar des del punt de vista energètic la conversió ADC mitjançant un microcontrolador ha de permetre estendre el temps de vida de nodes sensors alimentats amb piles i aplicats a ciutats, edificis,... intel·ligents.

BIBLIOGRAFIA

[1] Subhas Chandra Mukhopadhyay. *Wearable Sensors for Human Activity Monitoring: A Review.*: https://www.researchgate.net/publication/273393907_Wearable_Sensors_for_Human_Activity_Monitoring_A_Review

[2] Sean Dieter Tebje Kelly, Nagender Kumar Suryadevara, and Subhas Chandra Mukhopadhyay. *Towards the Implementation of IoT for Environmental Condition Monitoring in Homes.* " IEEE Sensors Journal, Vol. 13, No. 10, October 2013: http://pgembeddedsystems.com/securelogin/upload/project/IEEE/33/PGEMB0063

[3] F. Reverter and M. Gasulla, *Experimental characterization of the energy consumption of ADC embedded into microcontrollers operating in low power*. Proc. IEEE I2MTC 2019, Auckland, New Zealand, 20-23 May 2019

[4] Microchip Technology Inc. *PIC24.*: https://microchipdeveloper.com/16bit:doze-idle-sleep

[5] Microchip Technology Inc. *ATmega328PB Datasheet complete [PDF].*: http://ww1.microchip.com/downloads/en/DeviceDoc/40001906A.pdf

[6] Atmel Corp. . *AVR 8-bit Microcontrollers. AT15007.*: https://www.pololu.com/file/0J1464/Atmel-42559-Differences-between-ATmega328P-and-ATmega328PB_ApplicationNote_AT15007.pdff

[7] Microchip Technology Inc. *PIC24FJ256GA705 Datasheet complete [PDF].*: http://ww1.microchip.com/downloads/en/DeviceDoc/PIC24FJ256GA705-Family-Data-Sheet-DS30010118D.pdf

[8] F. Reverter, *Interfacing sensors to microcontrollers: A direct approach*, in: S. Nihtianov and A. Luque, Smart sensors and MEMS, Chapter 2, Woodhead Publishing, 2nd ed., Duxford, UK, 2018

[9] Microchip Technology Inc. *PIC24FJ256GA705 Datasheet complete [PDF].*: http://ww1.microchip.com/downloads/en/DeviceDoc/PIC24FJ256GA705-Family-Data-Sheet-DS30010118D.pdf

[10] Atmel Corp. *AVR 8-bit Microcontrollers. ATmega328/PB Summary.*: http://www.farnell.com/datasheets/2014286.pdf

[11] Luccio Di Jasio. *Programming 16-BIT PIC microcontrollers in C, Learning to Fly the PIC24.*Second Edition.: https://books.google.es/books?id=cBdWD4YOj3QC&pg=PR17&dq=microcontroller+pic24&hl=es&sa=X&ved=0ahUKEwjSmdyr9L7jAhXOa8AKHejoC4cQ6AEIKTAA#v=onepage&q=microcontroller20pic24&f=false

[12] Steven F. Barrett, Daniel J. Pack. *Atmel AVR Microcontroller, Programming and Interfacing.*Second Edition. : https://books.google.es/books?id=bAHGiKnwMO4C&pg=PA4&dq=microcontroller+ATmega&hl=es&sa=X&ved=0ahUKEwir7aus9b7jAhUOExQKHdOUBP0Q6AEIRTAD#v=onepage&q=microcontroller20ATmega&f=false

APÈNDIXS

APÈNDIX A. PROGRAMES D'ATMEGA324PB

A.1. Conversió ADC i comunicació sèrie via USART

```
#include <avr/io.h>         //--- Include Reg file of Atmega16

//--- F_CPU & BAUD RATE ---//

#define F_CPU       16000000    //--- CPU Clock Frequency
#define BAUD 9600       //--- Transmission Baud Rate
#define MYUBRR (F_CPU/16/BAUD-1)

#include "util/delay.h"



void usart_init(void);      //--- Usart Initialize
void usart_tx(uint16_t x);      //--- Usart Transmit
void adc_init(void);        // --- ADC initialize
uint16_t usart_rx(void);        //--- Usart Receive
uint16_t * adc_read(uint16_t ch); //--- ADC read

// Variables
int tx_data;
uint16_t resultADC[2];
```

```
//--- Main Program ---//

int main(void)
{

    usart_init();                       //--- Initialize usuart

    while(1)
    {
        uint16_t *adc_result;
        // initialize adc
        adc_init();

        adc_result = adc_read(0); // read ADC0
        usart_tx(adc_result[1]);
        usart_tx(adc_result[2]);
        _delay_ms(1000);
    }
}
```

```
void usart_init()
{
    UBRR1H = (MYUBRR>>8);
    UBRR1L = MYUBRR;
    UCSR1B = (1<<TXEN) | (1<<RXEN);      //--- Transmit and Receive Enable
    UCSR1C = (1<<USBS) | (3<<UCSZ0);     //--- 8 bit data and UCSRC is selected
}

void usart_tx(uint16_t x)
{

    while (!(UCSR1A & (1<<UDRE)));       //--- Check whether UDR is empty
    UDR1 = x;                            //--- Move data to UDR Data Reg
}



uint16_t usart_rx()
{
    while (!(UCSR1A & (1<<RXC)));        //--- Check whether Receive is complete
    return(UDR1);                        //--- Return Data from UDR
}
```

A.2. Canvis de freqüència

```
//

#define F_CPU 8000000UL //--- CPU Clock Frequency

#include <avr/io.h>
#include "avr/delay.h"
```

```c
int main(void)
{
    /* Replace with your application code */

    //CLKPR = 0x80;
    CLKPR= 0b00000001; // 2
    //CLKPR= 0b00000010; // 4
    //CLKPR= 0b00000011; // 8
    //CLKPR= 0b00000100; // 16
    //CLKPR= 0b00000101; // 32

    DDRA = 0b00000010;
    PORTA = 0xFF;

    while (1)
    {
        //PORTA = 0x00;
        //_delay_ms(500);
        PORTA = 0xFF;
        _delay_ms(1000);
        PORTA = 0x00;
        _delay_ms(1000);
        PORTA = 0xFF;
        _delay_ms(1000);
        PORTA = 0x00;

    }
}
```

A.3. Programa principal

```c
#include <avr/io.h>    //Llibreria General
#include <avr/sleep.h> //Llibreria Registres de modes Sleep

//Inicialització funcions:
void adc_init(void);    // Funció Configuració ADC
void sleepconfig(void);  // Funció per configurar modes de consum i registres de reducció de consum

//--- Main ---//
int main(void)
{
    adc_init(); // Executació de la configuració ADC
    sleepconfig(); //Executació Sleep and power consumption registers

    ADCSRA |= (1<<ADEN) | (1<<ADSC); // Inici de conversió

    SMCR  |= (1<<SE); // Posar en mode repòs
    sleep_cpu();    // Instrucció d'execució de Sleep Mode
}
```

```
void sleepconfig()
{
    //Deshabilitar entrades digitals als diferents PORTS
    DIDR0 = (1 << ADC5D) | (1 << ADC4D) | (1 << ADC5D) | (1 << ADC4D) | (1 << ADC3D) | (1 << ADC2D) | (1 << ADC1D) | (1 << ADC0D);
    ACSR |= (1 << ACD); //Disable Analog Comparator
    DDRA = 0x00; // Conf. pins PORT A com a entrades
    DDRB = 0x00; // Conf. pins PORT B com a entrades
    DDRC = 0x00; // Conf. pins PORT C com a entrades
    DDRD = 0x00; // Conf. pins PORT D com a entrades
    DDRE = 0x00; // Conf. pins PORT E com a entrades
    PORTA = 0xFF;  // Conf. pins PORT A com Entrades Pull-up
    PORTB = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    PORTC = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    PORTD = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    PORTE = 0xFF;   // Conf. pins PORT A com Entrades Pull-up
    //Deshabilitar interupcions
    MCUCR = 0b01110001;
    MCUCR = 0b01010010;
    //Deshabilitar Watchdog
    MCUSR &= ~(1<<WDRF);
    WDTCSR |= (1<<WDCE) | (1<<WDE);
    WDTCSR = 0x00;
    //Registres de reducció de consum
    PRR0 = (1<<PRTWI0) | (1<<PRTIM2) | (1<<PRTIM0) | (1<<PRUSART1) | (1<<PRTIM1) | (1<<PRSPI0) | (1<<PRUSART0)| (1<<PRADC);//Power Reduction Register 0
    PRR1 = (1<<PRTIM4) | (1<<PRTIM3); //Power Reduction Register 1
    PRR2 = (1<<PRPTC) | (1<<PRUSART2) | (1<<PRSPI1) |(1<<PRTWI1); //Power Reduction Register 2
    //Selecció Sleep Mode:
    SMCR = (1<<SM0); // ADC Noise Reduction
}
```

```
void adc_init()     //FUNCIÓ PER CONFIGURAR L'ADC
{

    ADMUX = 0b01011111;     // Selecció de la tensió de referència (Externa) i el canal a llegir (GND)

    // Habilitar ADC i Divisió de freqüència (PRESCALER)
    ADCSRA = (1<<ADEN)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE);               //8 [1000 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADATE);                        //16 [500 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS0)|(1<<ADATE);             //32 [250 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADATE);             //64 [125 kHz]
    //ADCSRA = (1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0)|(1<<ADATE);  //128 [62.5 kHz]

    ADCSRB = 0b00000000; //FREE RUNNING MODE
}
```

APÈNDIX B. PROGRAMES DE PIC24F

B.1. Conversió ADC i comunicació sèrie via USART

```
#include <xc.h>
#include <stdio.h>
#include <stdint.h>

void SYSTEM_Initialize(void);
void UART2_Initialize(void);
uint8_t UART2_Read(void);
void UART2_Write(char txData);
uint16_t UART2_StatusGet (void);
void UART2_Enable(void);
void UART2_Disable(void);
void ADC_Configuracion();
uint16_t ADC_leer10bit();
void delay(void);

int main(void)
{
    int m = 0,d = 0,c = 0,u = 0, j =0;
    int residuos1, residuos2, residuos3;
    uint16_t resultado;
    // initialize the device
    UART2_Initialize();
    ADC_Configuracion();

    while (j<=10)
    {
```

```
200     AD1CON3bits.ADCS = 0xFF;          //A/D Conversion Clock Select bits
201     AD1CON1bits.SSRC = 0x0;           //Sample Clock Source Select bits (SAMP is cleared by software=0)
202     AD1CON3bits.SAMC = 0b10000;       //Auto-Sample Time Select bits
203     AD1CON1bits.FORM = 0b00;          //Resultado decimal aboluto
204     AD1CON2bits.SMPI = 0x0;           //Interrupt Sample/DMA Increment Rate Select bits (Direct memory acces)
205     AD1CON1bits.MODE12 = 0;           //Conversion a 10 bits (=1 seria para 12 bits)
206     AD1CON1bits.ADON = 1;             //Convertidor A/D trabajando
207 }
208
209 uint16_t ADC_leer10bit()
210 {
211  uint16_t i ;
212  AD1CHS = 8;                       // Muestrea del por el terminal RB12_ Canal AN8 (8) 0b01000
213                                    // Empieza la Conversión
214  AD1CON1bits.SAMP = 1;             //Empieza el muestreo
215  for (i = 0 ; i < 1000 ; i++) ;    // Tiempo para estabilizar el muestreo
216  AD1CON1bits.SAMP = 0 ;            //Empieza la conversión
217  for (i = 0 ; i < 1000 ; i++) ;    // Tiempo de estabilización
218  while (!AD1CON1bits.DONE) ;       //Espera a que se termine la conversión
219  return ADC1BUF0 ;
220 }
221
222 void UART2_Initialize(void){
223 /**
224     Set the UART2 module to the options selected in the user interface.
225     Make sure to set LAT bit corresponding to TxPin as high before UART initialization
226 */
```

```
    // STSEL 1; IREN disabled; PDSEL 8N; UARTEN enabled; RTSMD disabled; USIDL
    // Data Bits = 8; Parity = None; Stop Bits = 1;
    U2MODE = (0x8008 & ~(1<<15));  // disabling UARTEN bit
    // UTXISEL0 TX_ONE_CHAR; UTXINV disabled; URXEN enabled; OERR NO_ERROR_cle
    U2STA = 0x1400;
    // BaudRate = 9600; Frequency = 4000000 Hz; U2BRG 103;
    U2BRG = 0x67;
    // ADMADDR 0; ADMMASK 0;
    U2ADMD = 0x00;

    UART2_Enable();  // enabling UARTEN bit
}

uint8_t UART2_Read(void){
    while(!(U2STAbits.URXDA == 1))
    {

    }

    if ((U2STAbits.OERR == 1))
    {
        U2STAbits.OERR = 0;
    }

    return U2RXREG;
}
```

```
void delay(void){
    int k;
        int j;
       for (j = 0; j< 1000; j++)
        {
           for(k = 0; k<1000; k++)
           {
           }
        }
}
```

B.2. Canvis de freqüència

```
#include "xc.h"

int main(void) {

    TRISA = 0x0000;
    TRISB = 0x0000;
    TRISC = 0x0000;

    CLKDIVbits.RCDIV = 0b000;
    CLKDIVbits.ROI = 0;
    //OSCDIVbits.DIV = 0b0000000000101000; //40 (50KHz)
    //OSCDIVbits.DIV = 0b0000000000010100; //20 (100KHz)
    //OSCDIVbits.DIV = 0b0000000000000100; //8 (250KHz)
    //OSCDIVbits.DIV = 0b0000000000000010; //4  (500KHz)
    OSCDIVbits.DIV = 0b0000000000000001; //2  (1MHz)

    PORTC = 0;

        while (1)
    {
        PORTCbits.RC1 = 1;
        PORTCbits.RC1 = 0;
        PORTCbits.RC1 = 1;
        PORTCbits.RC1 = 0;
        PORTCbits.RC1 = 1;
        PORTCbits.RC1 = 0;
        PORTCbits.RC1 = 1;
        PORTCbits.RC1 = 0;
```

```
        PORTCbits.RC1 = 1;
        PORTCbits.RC1 = 0;
        PORTCbits.RC1 = 1;
        PORTCbits.RC1 = 0;
    }
    return 0;
}
```

B.3. Programa principal

```
#include <xc.h>
#include <stdio.h>
#include <stdint.h>
void disable_perifericos(void);
void ADC_Configuracion(void);
void delay(void);

int main(void)
{
    disable_perifericos();
    ADC_Configuracion();
    TRISA = 0b0000000100000000;
    TRISB = 0x0000;
    TRISC = 0x0000;

    CLKDIVbits.RCDIV = 0b000;
    CLKDIVbits.ROI = 0;
    //OSCDIVbits.DIV = 0b0000000000101000; //40 (50KHz)
    //OSCDIVbits.DIV = 0b0000000000010100; //20 (100KHz)
    //OSCDIVbits.DIV = 0b0000000000000100; //8 (250KHz)
    //OSCDIVbits.DIV = 0b0000000000000010; //4  (500KHz)
    OSCDIVbits.DIV = 0b0000000000000001;   //2  (1MHz)

    INTCON2bits.GIE = 0;
```

```
    //AD1CON3bits.ADCS = 0b00100000;   //62,5KHz
    //AD1CON3bits.ADCS = 0b00010000;   //125KHz
    //AD1CON3bits.ADCS = 0b00001000;   //250KHz
    //AD1CON3bits.ADCS = 0b00000100;   //500KHz
    //AD1CON3bits.ADCS = 0b00000010;   //1MHz
    AD1CON3bits.ADCS = 0b00000001;     //2MHz
    AD1CON1bits.SAMP = 1;              //muestreo
    AD1CON1bits.ADON = 1;              //Convertidor A/D trabajando

    delay();
    Idle();
    return 1;
}

void ADC_Configuracion()
{
    ANSA = 0b0000000100000000;
    AD1CHSbits.CH0SB = 0b01101;
    AD1CON5bits.LPEN = 0;
    AD1CON2bits.PVCFG = 0b01;
    AD1CON2bits.NVCFG0 = 1;
    AD1CON1bits.ADSIDL = 0;
    AD1CON1bits.SSRC = 0b0111;
    AD1CON1bits.ASAM = 1;
    AD1CON3bits.SAMC = 1;
    AD1CON2bits.ALTS = 0;
```

```
53          AD1CON3bits.ADRC = 0;
54          AD1CON5bits.ASEN = 1;
55      }
56
57      void disable_perifericos()
58      {
59        // Se desactivan todos los timer de la máquina
60        PMD1bits.T3MD = 1;
61        PMD1bits.T2MD = 1;
62        PMD1bits.T1MD = 1;
63
64        // Desactiva el I2C (comunicador serial sincrono)
65        PMD1bits.I2C1MD = 1;
66        PMD3bits.I2C2MD = 1;
67
68        // Se desactivan las UART
69        PMD1bits.U2MD = 1;
70        PMD1bits.U1MD = 1;
71
72        // Se desactivan los modulos SPI (comunicador serial sincrono)
73        PMD1bits.SPI2MD = 1;
74        PMD1bits.SPI1MD = 1;
75        PMD6bits.SPI3MD = 1;
76
```

```
77        // Se desactivan los modulos por captura
78        PMD2bits.IC3MD = 1;
79        PMD2bits.IC2MD = 1;
80        PMD2bits.IC1MD = 1;
81
82        PMD2bits.OC3MD = 1;
83        PMD2bits.OC2MD = 1;
84        PMD2bits.OC1MD = 1;
85        // Se desactivan los comparadores
86        PMD3bits.CMPMD = 1;
87
88        // Se desactiva el Puerto paralelo
89        PMD3bits.PMPMD = 1;
90
91        // Activa el ADC
92        PMD1bits.AD1MD = 0;
93      }
94
95      void delay()
96      {
97          int k;
98              int j;
99             for (j = 0; j<500; j++)
100             {
101             }
102     }
```