

Università degli Studi di Trieste

Facoltà di Ingegneria

Corso di Laurea Specialistica in Ingegneria Elettronica

Dipartimento di Elettrotecnica, Elettronica e Informatica



Tesi di Laurea

**PROGETTO HARDWARE E SOFTWARE
DEL SISTEMA DI CONTROLLO
DEL SATELLITE ATMOCUBE**

Relatore:

Chiar.mo Prof. Sergio CARRATO

Correlatore:

Chiar.mo Prof. Mario FRAGIACOMO

Laureando:

Stefano PUNIS

Anno Accademico 2007 – 2008

1	<i>Il satellite Atmocube</i>	4
1.1	Descrizione generale dell'intero sistema	5
2	<i>Funzionamento e specifiche del modulo OBDH & HK</i>	7
2.1	Silicon Drift Detector SDD	8
2.2	Global Position System GPS	10
2.3	Magnetometro	13
2.4	On Board Radio OBR	17
2.5	Telemetria e storage dei dati di strumentazione	22
2.6	Power Management	23
3	<i>Scelta della memoria esterna e del microcontrollore</i>	24
3.1	Memoria esterna	25
3.2	Microcontrollore Microchip 18LF8722	30
4	<i>Architettura hardware del sistema di controllo</i>	42
4.1	Bus parallelo per la comunicazione del μC con la memoria esterna	44
4.2	Bus SPI per la comunicazione del μC con gli strumenti e il modulo OBR	59
4.3	Comunicazione seriale asincrona del μC con il GPS ed il PC	66
4.3.1	Gestione del segnale RS-232	67
4.3.2	Gestione del segnale RS-422	67
4.4	Misura dei segnali analogici per mezzo del convertitore AD	68
4.5	Abilitazione dei blocchi funzionali della Power Supply Unit	71
4.6	Considerazioni generali e scelte comuni	72
5	<i>Sistema di alimentazione del modulo OBDH & HK</i>	73
5.1	Regolazione della tensione della batteria	74
5.2	Circuito supervisore del Reset Hardware del μC	76
5.3	Sistema di carica della batteria e di ibernazione dello stato del μC	87
5.3.1	Modalità a basso consumo del microcontrollore	91
6	<i>Driver software per le periferiche del microcontrollore</i>	94
6.1	Scrittura e lettura della memoria SRAM di telemetria	95
6.2	Scrittura e lettura sul bus SPI	103
6.3	Scrittura e lettura mediante la periferica EUSART	111
6.4	Acquisizioni analogiche	118
7	<i>Real Time Operating System di Atmocube</i>	123
7.1	Operazioni del μ Kernel	124
7.2	Stack del PIC18LF8722	129
7.3	Struttura degli interrupt	132
7.4	Timer 0	134

7.5	Struttura di ARTOS (Atmocube Real Time Operating System)	136
7.5.1	Assegnazione delle priorità	137
7.5.2	Procedura di avvio	139
7.5.3	Procedure di regime e misura	140
7.5.4	Procedura in modalità di sicurezza	141
7.5.5	Stack software	142
Conclusioni		145
Ringraziamenti		146
Bibliografia		148
Appendice		150

1 Il satellite Atmocube

Atmocube rappresenta un innovativo sistema di misura per studiare l'ambiente spaziale vicino alla Terra su altitudini minime di 350 km. Esso è costituito da un nano satellite di forma cubica di 10 cm di lato con una massa totale massima di 1 kg.

Il progetto viene supportato e portato avanti dal Dipartimento di Fisica, il Dipartimento di Elettrotecnica, Elettronica ed Informatica e il Dipartimento di Ingegneria Meccanica.

L'obiettivo principale della missione spaziale è quello di costruire una mappa del flusso di radiazione presente, correlata a misure di campo magnetico e posizione. Per svolgere questo compito il payload del satellite include uno spettro-dosimetro, un magnetometro e un GPS.

Un'analisi continua lungo l'orbita del satellite fornisce una copertura totale di misura e dà la possibilità di studiare i legami tra i tre sistemi principali che caratterizzano la Terra, cioè il sole, l'attività solare e l'atmosfera terrestre.

In accordo con le specifiche del lanciatore VEGA, l'orbita percorsa è ellittica ed è compresa tra 350 e 1200 km con un'inclinazione di 71 °.

La vita del satellite è variabile e dipende fortemente dall'attività solare presente; a tal proposito tra gli obiettivi della missione compare anche il termine minimo di funzionamento dell'intero sistema che è stato imposto pari a 6 mesi.

1.1 Descrizione generale dell'intero sistema

Il satellite Atmocube è composto sostanzialmente da sei moduli principali:

- ***il Silicon Drift Detector (SDD);***
- ***il Global Position System (GPS);***
- ***il magnetometro;***
- ***il ricetrasmittitore On Board Radio (OBR);***
- ***la Power Supply Unit (PSU);***
- ***I'On Board Data Handling and House Keeping (OBDH & HK).***

I primi tre moduli compongono la parte sperimentale del satellite e sono gli strumenti di bordo necessari ad effettuare tutte le misurazioni delle grandezze fisiche di interesse.

Il ricetrasmittitore ha il compito di instaurare la comunicazione tra il satellite e la stazione di terra permettendo di analizzare i dati raccolti durante le varie misure e i dati sullo stato dell'intero sistema. Tutto ciò è possibile solamente in una condizione di visibilità tra satellite e stazione terrestre ovvero in una finestra temporale limitata.

La Power Supply Unit alimenta il satellite e permette di ottenere le tensioni e le correnti indispensabili per fornire energia ai vari moduli. Essa dispone di varie linee di abilitazione che consentono l'esclusione di alcuni sottosistemi per questioni di risparmio energetico.

Il modulo OBDH & HK gestisce e controlla tutte le parti che compongono Atmocube attraverso il microcontrollore installato ed è l'argomento su cui è centrato completamente questo elaborato.

Oltre a questi sei componenti, il satellite dispone di alcune celle solari, di una serie di sensori di luce, di uno o più sensori di temperatura, un dispositivo di sgancio dell'antenna e alcuni interruttori che isolano galvanicamente l'intero satellite dalla batteria.

Le celle solari permettono di immagazzinare energia nell'accumulatore principale costituito da una batteria agli ioni di litio, i sensori di luce sono posizionati sui lati del cubo e consentono di determinarne l'assetto, i sensori di temperatura fanno parte delle misure di housekeeping e tengono sotto controllo sia la struttura che l'accumulatore.

Il dispositivo di sgancio dell'antenna è costituito da un componente elettromeccanico che consente di espandere l'antenna alle sue dimensioni originali una volta espulso nello spazio.

In ultimo gli interruttori permettono di fornire energia al sistema solo a lancio concluso.

Nella pagina seguente si riporta uno schema elettrico del satellite Atmocube; le particolarità delle connessioni è parzialmente comprensibile in questo momento e risulteranno chiare solo dopo la lettura dei prossimi capitoli.

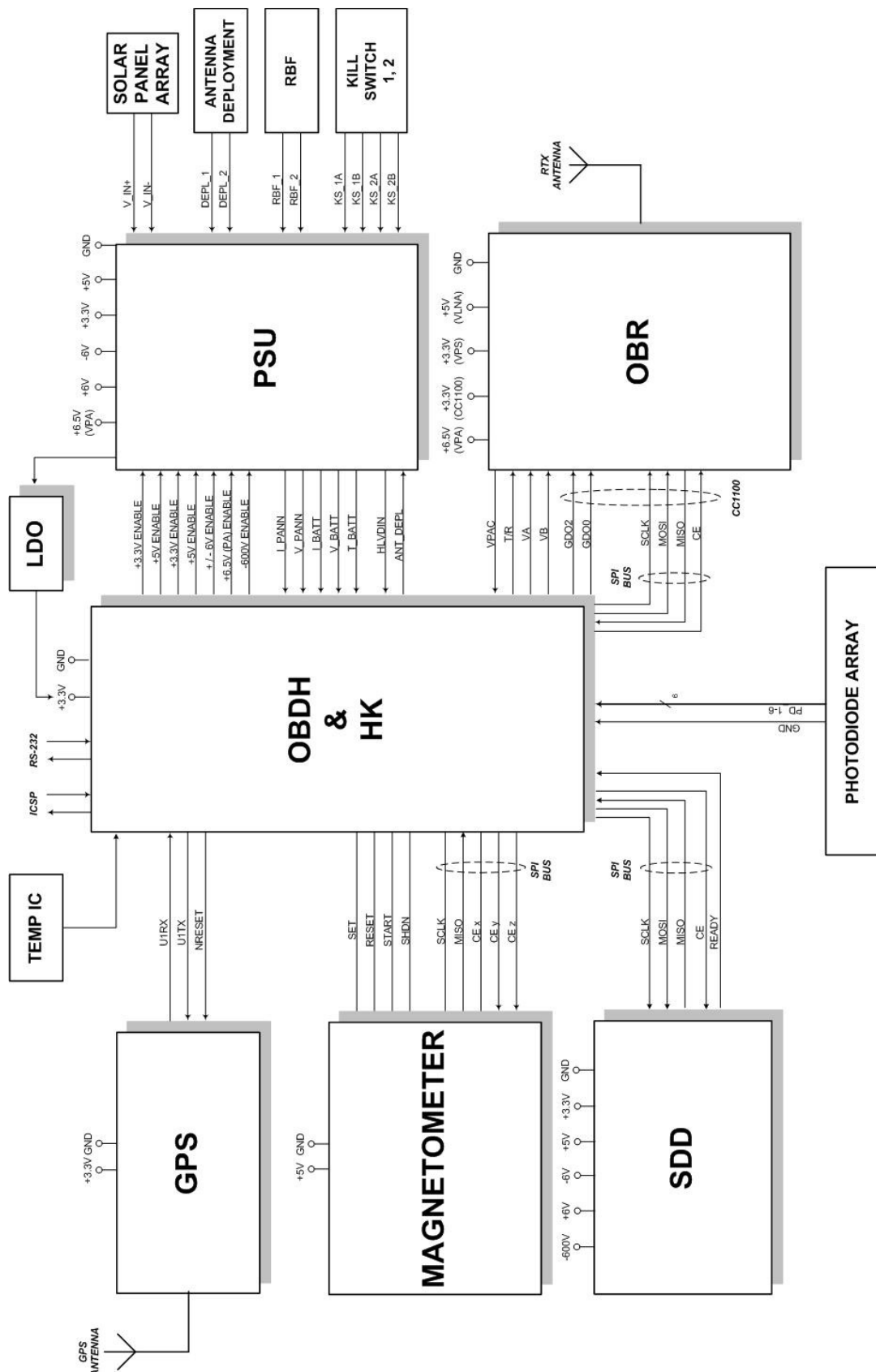


FIG. 1.1 – SCHEMA ELETTRICO DEL SATELLITE ATMO-CUBE

2 Funzionamento e specifiche del modulo OBDH & HK

L'OBDH & HK costituisce il nucleo del satellite, al quale è assegnato il compito di gestire l'intero sistema. Essa permette di far funzionare i singoli moduli e di metterli in comunicazione tra loro consentendo al satellite di svolgere completamente le funzioni per il quale è stato progettato. Ad essa inoltre è affidato l'importante compito di gestione della telemetria svolta per mezzo degli strumenti di bordo e per mezzo dei sensori di cui è dotato il satellite.

Le funzioni essenziali per cui il modulo OBDH & HK deve essere progettato sono le seguenti:

- ***gestione ed acquisizione dei dati dal SDD;***
- ***gestione ed acquisizione dei dati dal GPS;***
- ***gestione ed acquisizione dei dati dal magnetometro;***
- ***invio e ricezione dei dati tramite il modulo OBR;***
- ***lettura e scrittura dei dati di telemetria;***
- ***Power Management per il risparmio energetico;***

Nei paragrafi successivi verrà fornita una breve descrizione di ogni modulo del satellite Atmocube allo scopo di avere una visione complessiva del funzionamento del sistema.

2.1 Silicon Drift Detector SDD

Lo strumento di Atmocube denominato SDD è uno spettro-dosimetro; la sua funzione principale è quella di calcolare la dose di radiazione che viene depositata sul rivelatore a partire dallo spettro di energia della radiazione elettromagnetica o corpuscolare.

L'ulteriore caratteristica di questo strumento è quella di poter rilevare anche il passaggio di particelle con energia superiore rispetto al fondo scala dello spettro-dosimetro; per tal motivo è presente quindi anche la funzione di contatore di eventi ionizzanti associato a particelle ad alta energia.

Lo spettro dosimetro svolge il suo compito grazie alla presenza di un rivelatore chiamato Silicon Drift Chamber (camera a deriva in silicio). Essendo costruito in materiale semiconduttore, esso si presta a rilevare i fotoni nello spettro dei raggi X, dato il basso valore dell'energia di ionizzazione necessaria a creare una coppia lacuna–elettrone. Dato che il rivelatore trasduce l'energia della radiazione elettromagnetica in carica, è necessario effettuare un'elaborazione del segnale presente alla sua uscita.

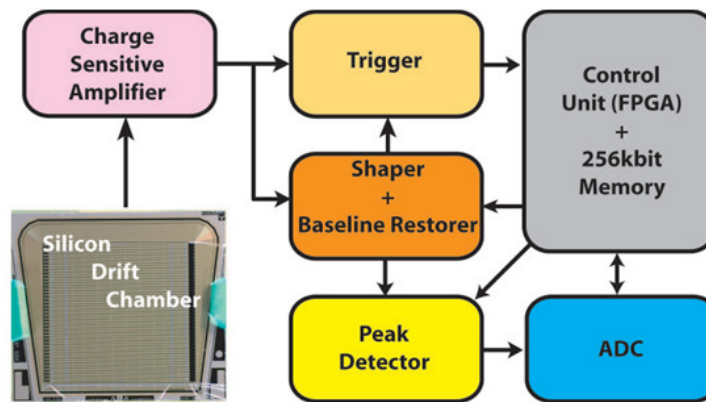


FIG. 2.1 - SCHEMA A BLOCCHI DEL MODULO SDD

La carica elettrica fornita dalla camera a deriva viene convertita in una tensione grazie alla presenza di un amplificatore CSA, Charge Sensitive Amplifier. Successivamente questo segnale fa capo ad un filtro shaper in modo da ridurre i disturbi e migliorare il rapporto segnale–rumore. Quello che si ottiene ora è un impulso che mantiene le caratteristiche di proporzionalità diretta con la carica fornita dalla camera a deriva. Il Peak Detector, a questo punto, mantiene il valore dell'impulso e lo invia al convertitore analogico–digitale. Quest'ultimo, assieme a parte dell'unità di controllo, funge da analizzatore multicanale in modo da contare le particelle che hanno energia compresa in un certo intervallo. Il fondo scala è tarato per coprire energie fino a 70 keV, in tal maniera, con una risoluzione dell'AD converter di 8bit, ogni canale ha un'ampiezza pari a circa 274 eV. L'informazione ottenibile quindi da questo tipo di strumento è uno spettro di energia discreto dei fotoni che si scontrano con la camera a deriva. Il tutto viene immagazzinato nella memoria presente nell'unità di controllo e aggiornato all'arrivo di ogni particella. Dallo schema a blocchi si nota un ulteriore circuito di trigger che ha come compito di segnalare l'inizio di una nuova misura in corrispondenza di ogni nuovo fotone.

Le particelle con energia maggiore di 70 keV non vengono processate nella modalità appena descritta, ma vengono solamente contate; il conteggio pertanto sfrutta solamente il CSA, il trigger e l'unità di controllo digitale.

Per ciò che riguarda la misura sperimentale questa caratteristica è meno utile, ma fornisce ugualmente informazioni sull'attività del sole.

L'interfacciamento con la scheda OBDH & HK avviene sostanzialmente instaurando una comunicazione di tipo digitale con l'unità di controllo del modulo SDD, ovvero con la FPGA di cui è equipaggiato. A tal proposito è disponibile un protocollo seriale di tipo SPI con l'ausilio di ulteriori comandi. Le funzioni di controllo sostanzialmente sono le seguenti:

- **impostazione del tempo di integrazione:** la durata delle misure viene prestabilita dalla stazione di terra, di conseguenza è opportuno prevedere questo controllo (bus SPI);
- **START:** il modulo OBDH & HK invia il comando per iniziare la misura sulle particelle incidenti;
- **lettura dello spettro di energia:** il microcontrollore invia un comando che indica la volontà di ricevere i dati misurati, quindi successivamente a questo la FPGA invia il pacchetto di dati dello spettro discreto (bus SPI);
- **lettura del numero di eventi ionizzanti:** la procedura per lo scambio di questa informazione avviene in modo analogo al punto precedente (bus SPI);
- **interrogazione dello stato:** tramite questa funzione si informa il microcontrollore dello stato del modulo SDD, per esempio se è in fase di misura o meno (bus SPI);
- **reset:** si prevede un ulteriore comando in modo da porre in reset lo spettrodosimetro.

<i>n° pin</i>	<i>nome del pin</i>	<i>I/O</i>	<i>descrizione</i>
1	SCLK	I	Clock del bus SPI
2	MISO	O	Segnale SPI in uscita dalla FPGA
3	MOSI	I	Segnale SPI in ingresso alla FPGA
4	START	I	Inizio della misura
5	RESET	I	Reset del modulo SDD

FIG. 2.2 - SEGNALI DEL MODULO SDD DISPONIBILI PER L'INTERFACCIAMENTO A MICROCONTROLLORE

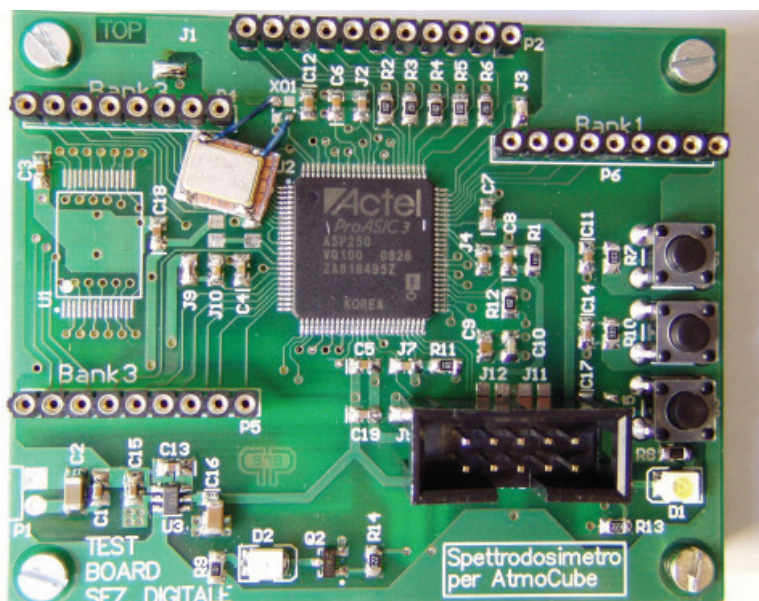


FIG. 2.3 - UNITA' DI CONTROLLO DEL MODULO SDD CON FPGA ACTEL A3P250

2.2 Global Position System GPS

Il GPS installato viene interrogato dalla modulo OBDH & HK per:

- ottenere dati di tempo, posizione e velocità in istanti specifici, da memorizzare assieme alle misure degli altri strumenti;
- ottenere dati di tempo, posizione e velocità ad intervalli regolari per determinare l'assetto del satellite e per rilevare il passaggio sopra la stazione di Basovizza; questi dati andranno inseriti in telemetria e permetteranno di attivare il ricetrasmittitore solo in condizioni di visibilità con la stazione di terra.

Il GPS sarà sempre attivo, eccetto nel caso la batteria sia scarica, poiché l'acquisizione della costellazione dei satelliti richiede almeno una decina di minuti.

Il GPS installato è un SGR-05 della Surrey Satellite Technology, appositamente progettato per applicazioni di tipo spaziale, con caratteristiche sia di dimensioni ridotte che di bassi consumi; esso riceve e decodifica da quattro o più satelliti GPS il segnale e fornisce la posizione di Atmocube con errori di 5-15 metri. Uno specifico circuito integrato presente nella sezione RF converte, filtra e amplifica il segnale GPS a 1,57542 GHz, mentre nella sezione digitale un processore con altre periferiche permette l'interfacciamento con l'esterno. Il codice macchina è presente in una memoria flash che viene copiata dopo la fase di inizializzazione in un'altra memoria di tipo RAM; tale operazione è permessa grazie ad un *loader program*. Dopo aver fornito l'alimentazione il codice viene copiato direttamente nella memoria RAM ed eseguito dal processore. Il software usa un sistema operativo real time a basso livello che commuta tra i vari task per mezzo di un interrupt programmabile.

I segnali messi a disposizione dall'interfaccia del GPS sono i seguenti:

- **UART1:** permette un collegamento punto a punto con il microcontrollore interno del GPS e costituisce il metodo principale di controllo e comunicazione;
- **PPS (Pulse per second output):** è un'uscita di tipo TTL che può essere impiegata per sincronizzare segnali di clock esterno al ricevitore GPS;
- **NRESET:** è un comando a logica TTL impiegato per mandare un segnale di reset al microcontrollore del GPS;
- **LOADER_MODE (GPIO7):** è una linea a logica TTL che permettere di controllare la modalità di boot del programma del GPS;
- **ROM_MODE (MFIO):** è una linea a logica TTL che consente di riprogrammare la memoria flash in cui è presente il codice di gestione del GPS;
- **VIN:** la versione del GPS usata in questa applicazione consente un'alimentazione a 3.3 V.

<i>n° pin</i>	<i>nome del pin</i>	<i>descrizione</i>
1	U1RX	Ricezione UART1
2	U1TX	Trasmissione UART1
3	NC	Non connesso
4	GPIO7	Selezione del boot TCD_Loader
5	MFIO	Selezione della programmazione TCD_ROM
6	NRESET	Reset
7	VIN	Alimentazione 3.3V a 150mA
8	GND	Massa
9	PPS	Pulse per Second

FIG. 2.4 - PIEDINATURA DEL GPS ACCESSIBILE DAL CONNETTORE VOLANTE

<i>n° pin</i>	<i>nome del pin</i>	<i>I/O</i>	<i>note</i>
1	U2TX	O	Non implementato
2	U2RX	I	Non implementato
3	PPS	O	OFF da default
4	U1TX	O	
5	U1RX	I	
6	NRESET	I	Attivo basso
7	MFIO	I	Solo in caso di flash corrotta
8	GND	-	
9	VIN	-	
10	VIN	-	
11	NC	-	
12	U1TX	O	Non implementato

FIG. 2.5 - PIEDINATURA DEL GPS ACCESSIBILE DAL CONNETTORE J1 PRESENTE NEL PCB

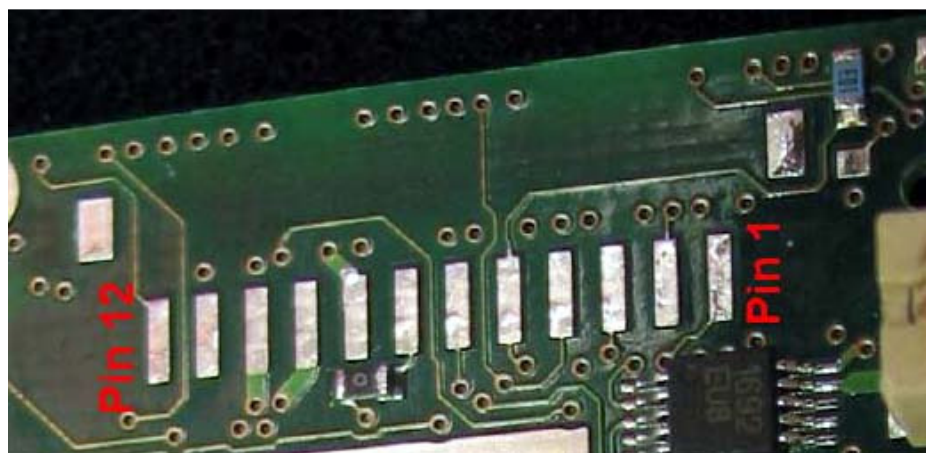


FIG. 2.6 - INGRANDIMENTO DEL PCB DEL GPS CON EVIDENZIATO IL CONNETTORE J1

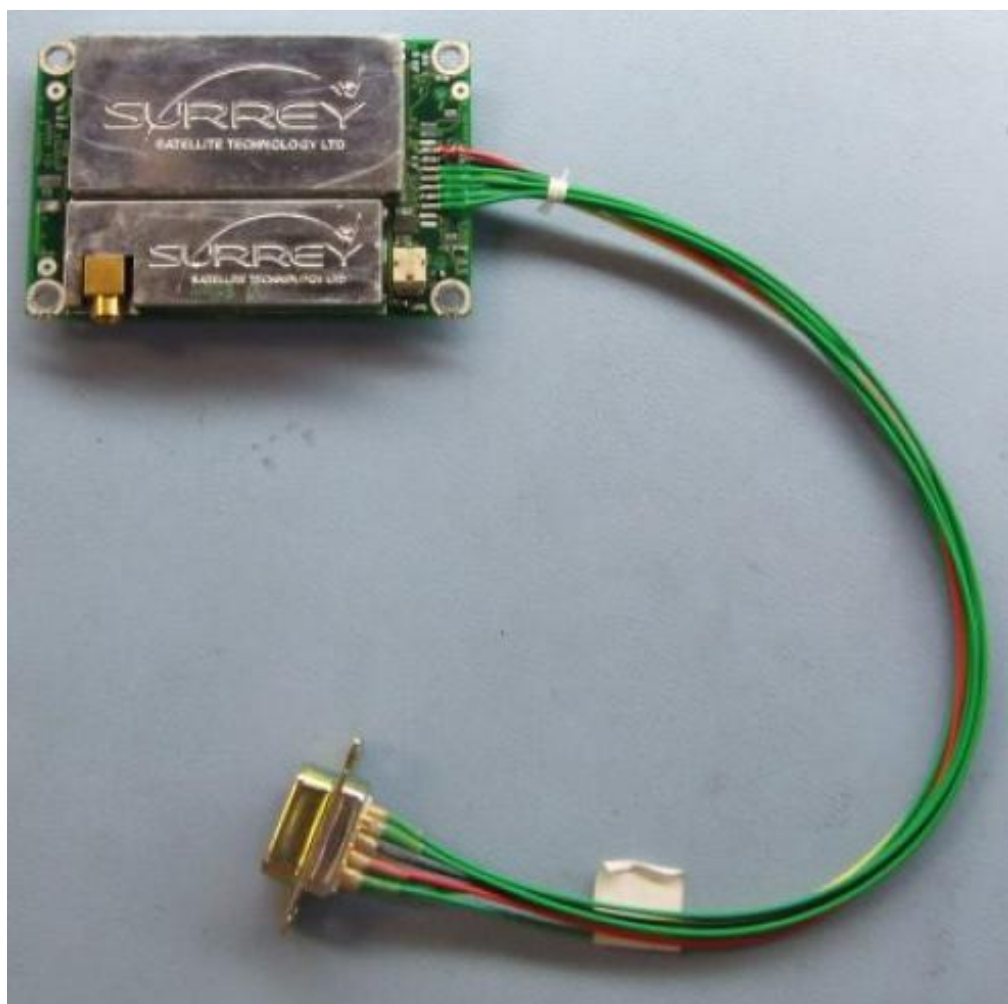


FIG. 2.7 - UN'IMMAGINE DEL GPS DELLA SURREY

2.3 Magnetometro

Il magnetometro permette di acquisire le tre componenti del vettore campo magnetico terrestre (B_x , B_y , B_z). L'interfacciamento al microcontrollore è consentito per mezzo di convertitori analogici-digitali che dialogano direttamente nello standard seriale SPI; poiché la misura delle 3 componenti deve essere contemporanea, si rende necessario utilizzare altrettanti circuiti sample hold da abilitare al medesimo istante. Questo strumento viene alimentato solo quando deve essere effettuata la misura e, dato che il magnetometro è fisso al satellite, il vettore campo magnetico terrestre è relativo al sistema di riferimento solidale della struttura meccanica di Atmocube. Per questo ultimo motivo infatti risulta indispensabile corredare ciascun dato di campo anche con i dati di tempo, posizione e velocità ottenuti dal modulo GPS in modo da passare dal sistema di riferimento solidale a quello terrestre.

Il modello del magnetometro impiegato è l'HMC2003 della Honeywell il quale contiene tre sensori ad alta sensibilità costituiti da dei ponti di Wheatstone realizzati con elementi magnetoresistivi di Permalloy (NiFe). I segnali relativi alle componenti del campo sono disponibili su delle uscite analogiche pilotate da degli amplificatori a basso rumore opportunamente filtrati. I nodi di alimentazione dei ponti sono accessibili attraverso la piedinatura del circuito integrato e ciò risulta utile per poter regolare i valori di offset che influenzano erroneamente la misura. Il satellite infatti monta al suo interno dei magneti permanenti necessari per il controllo passivo dell'assetto di Atmocube una volta in orbita che creano inevitabilmente dei campi magnetici indesiderati.

Dato che i sensori di Permalloy presentano degli effetti di isteresi, prima di ogni utilizzo bisogna orientare i domini di Weiss al fine di limitare la memoria degli eventi passati e massimizzare la correlazione tra il segnale di uscita ed il valore effettivo del campo che deve essere misurato. A tale scopo si implementa un ulteriore circuito di *set* e *reset* controllato dalla scheda OBDH & HK in modo da far scorrere una corrente di 4 A per circa 2 μ s sul film di Permalloy.

L'alimentazione di questo modulo è compresa tra 6 e 15 V, di conseguenza è indispensabile un circuito di elevazione della tensione della batteria.

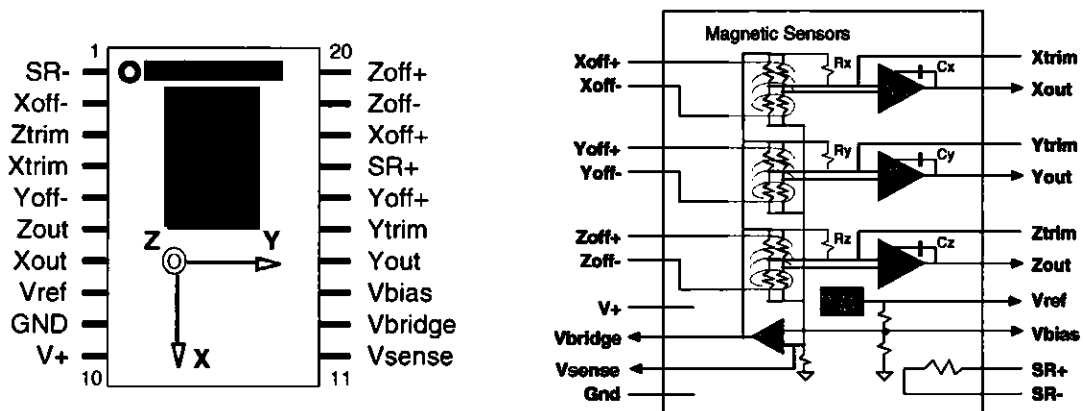


FIG. 2.8 - MODULO HMC2003, RIFERIMENTO DEGLI ASSI E SCHEMA INTERNO

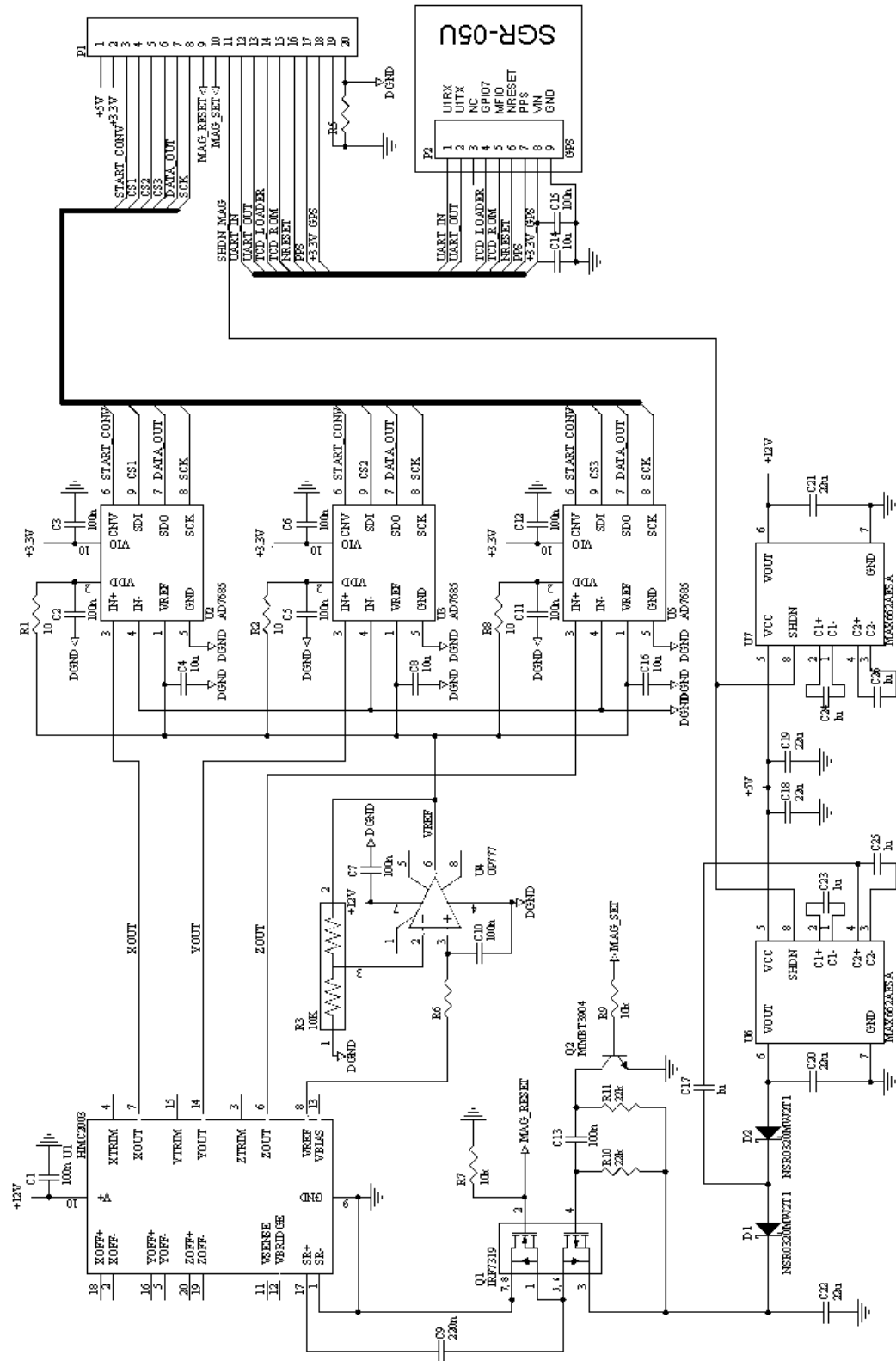


FIG. 2.9 - SCHEMA ELETTRICO COMPLETO DELLA SCHEDA MAGNETOMETRO

Dallo schema si notano i tre convertitori analogico-digitali AD7685 posti sulle uscite del modulo HMC2003, un amplificatore operazionale, due convertitori DC-DC MAX662 e una coppia di Mosfet.

I convertitori ADC sono stati scelti in base alla risoluzione a 12 bit necessaria per le misure ad alta sensibilità e in base al fatto che presentano alimentazioni separate per la parte analogica e quella digitale. Infatti la parte analogica, compresa la tensione di riferimento, viene alimentata dal magnetometro per mezzo di un amplificatore operazione mentre la parte digitale è vincolata a 3.3 V per consentire la compatibilità con il bus SPI della scheda OBDH & HK.

La coppia di Mosfet è il driver per la corrente di 4 A necessaria a ridurre l'isteresi del Permalloy. Per fornire una corrente di tale intensità è d'obbligo alimentare i Mosfet a circa 20 V e per questo è presente il primo convertitore DC-DC assieme ai due diodi posti in cascata. L'altro convertitore invece fornisce la tensione di alimentazione di 12 V del modulo HMC2003.

Il circuito integrato della Maxim eleva la tensione in ingresso di 5 V a 12 V garantendo una corrente di 30 mA.

Riassumendo le linee di accesso alla scheda magnetometro completa sono le seguenti:

- **+5V:** è la tensione di alimentazione dei convertitori DC-DC;
- **+3.3V:** è la tensione di alimentazione della parte digitale dei tre convertitori ADC necessaria per stabilire la comunicazione sul bus SPI;
- **START:** è il comando comune a tutti e 3 i convertitori ADC per iniziare la conversione e quindi l'acquisizione sui tre canali del modulo HMC2003;
- **MISO:** è la linea SPI sulla quale si presenta la parola a 12 bit di conversione;
- **SCLK:** è il clock seriale generato dal Master SPI del microcontrollore;
- **CSX, CSY, CSZ:** sono le abilitazioni che fanno capo ai singoli convertitori.
- **SET e RESET:** sono i comandi che agiscono sui gate del circuito a Mosfet;
- **SHDN:** permette di disabilitare i convertitori DC-DC garantendo consumi minimi;

<i>n° pin</i>	<i>nome del pin</i>	<i>I/O</i>	<i>note</i>
1	+5V	-	Alimentazione conv. DC – DC
2	+3.3V	-	Alimentazione bus SPI
3	START_CONV	I	Inizio dell'acquisizione
4	CE1 – CEX	I	Abilitazione convert. comp. X
5	CE2 – CEY	I	Abilitazione convert. comp. Y
6	CE3 - CEZ	I	Abilitazione convert. comp. Z
7	MISO	O	Dati in ingresso al μC
8	SCLK	I	Clock del bus SPI
9	MAGN_RESET	I	Comando dei MOS
10	MAGN_SET	I	Comando dei MOS
11	SHDN	I	Shutdown dei conv. DC - DC

FIG. 2.10 - PIEDINATURA DEL MODULO MAGNETOMETRO

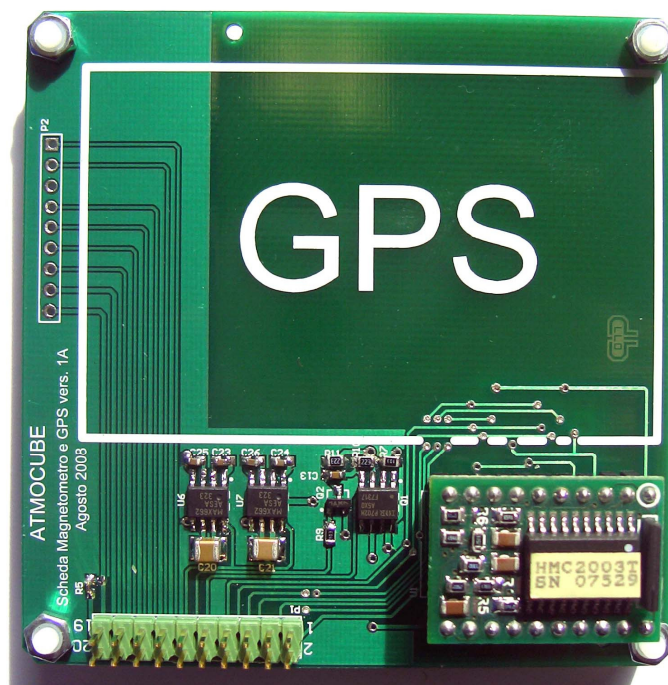


FIG. 2.11 - SCHEDA PROTOTIPO DEL GPS E DEL MAGNETOMETRO

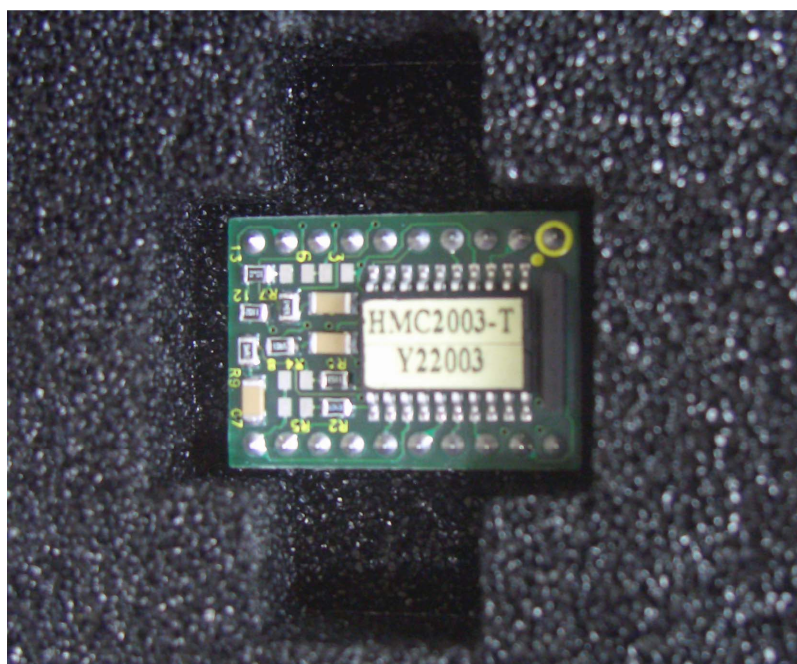


FIG. 2.12 - INGRANDIMENTO DEL SENSORE IMPIEGATO NEL MAGNETOMETRO HMC2003

2.4 On Board Radio OBR

Il ricetrasmittitore di bordo OBR è composto dai seguenti sottosistemi:

- *il ricevitore ed il trasmettitore a bassa potenza;*
- *il commutatore RX-TX in bassa potenza;*
- *il circuito del Front End (LNA) del ricevitore;*
- *l'amplificatore di potenza del trasmettitore;*
- *il commutatore di antenna;*
- *il circuito di misura della potenza emessa.*

Il ricevitore ed il trasmettitore a bassa potenza sono costituiti dal circuito integrato U102 CC1100 e dalla rete che va dalle sue uscite (piedini RF_N e RF_P , 12 e 13 del CC1100) fino al commutatore RX TX, U103 RF2536. La rete serve per la trasformazione da bilanciato a sbilanciato e per l'adattamento di impedenza verso il resto del circuito. Gli assorbimenti di corrente con Vcc = 3.3 V a +25 °C sono:

- 30 mA in modo TX , con livello di uscita pari a +10 dBm;
- 17 mA in modo RX;
- 1 μ A massimo in modo sleep.

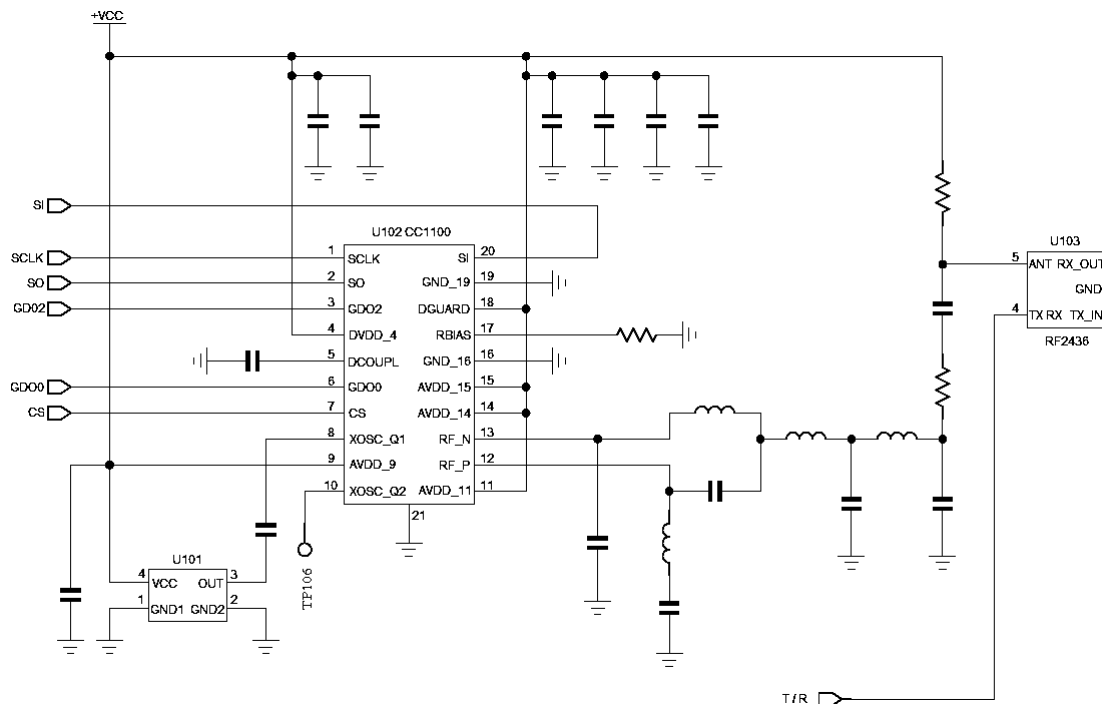


FIG. 2.13 - STADIO A BASSA POTENZA

Il commutatore RX-TX è formato dal circuito integrato della RF MicroDevice RF2436 denominato nello schema come U103. Si tratta di un dispositivo GaAs MESFET le cui caratteristiche sono:

- tensione di alimentazione compresa tra 1.5 V e 6 V;
- perdita di inserzione tipica 1 dB, massima 2 dB;
- isolamento 24 dB;
- massima potenza +27 dBm;
- la tensione di comando deve essere compresa tra 0 e 0.7 V in modo RX e tra 0.7 e 3.3 V in modo TX (Livelli logici CMOS);
- assorbimento di corrente massimo di 1 mA in modo TX e di 10 μ A in modo RX.

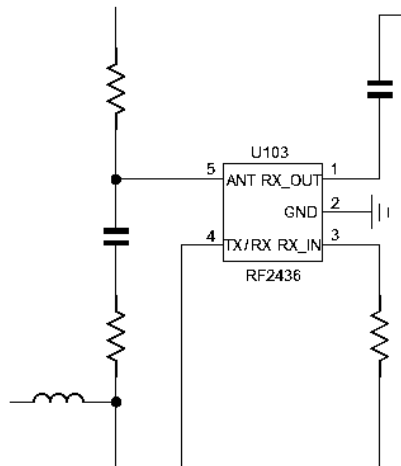


FIG. 2.14 - COMMUTATORE RX TX

Il Front End è la parte che comprende l'amplificatore a basso rumore LNA siglato Q101. Sono presenti inoltre i filtri passa banda FC101 e FC102 per l'attenuazione dei prodotti di intermodulazione di secondo ordine e per la limitazione della banda passante dell'amplificatore.

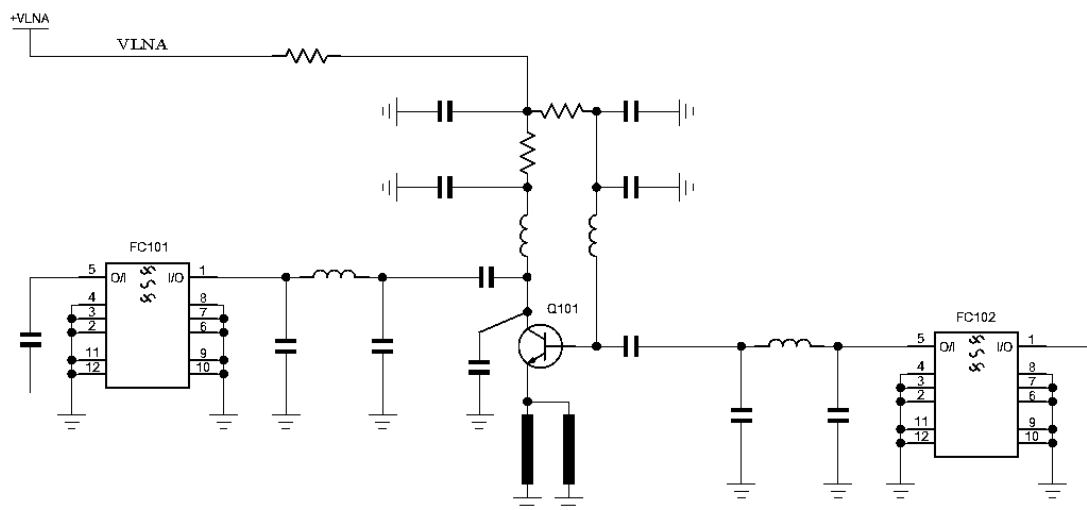


FIG. 2.15 - FRONT END E AMPLIFICATORE LNA

L'amplificatore di potenza del trasmettitore è realizzato con il modulo ibrido MHW704 della Motorola, U104. La potenza di uscita è di 3 W con una tensione di alimentazione di +6 V e con una potenza del segnale di ingresso di 1 mW.

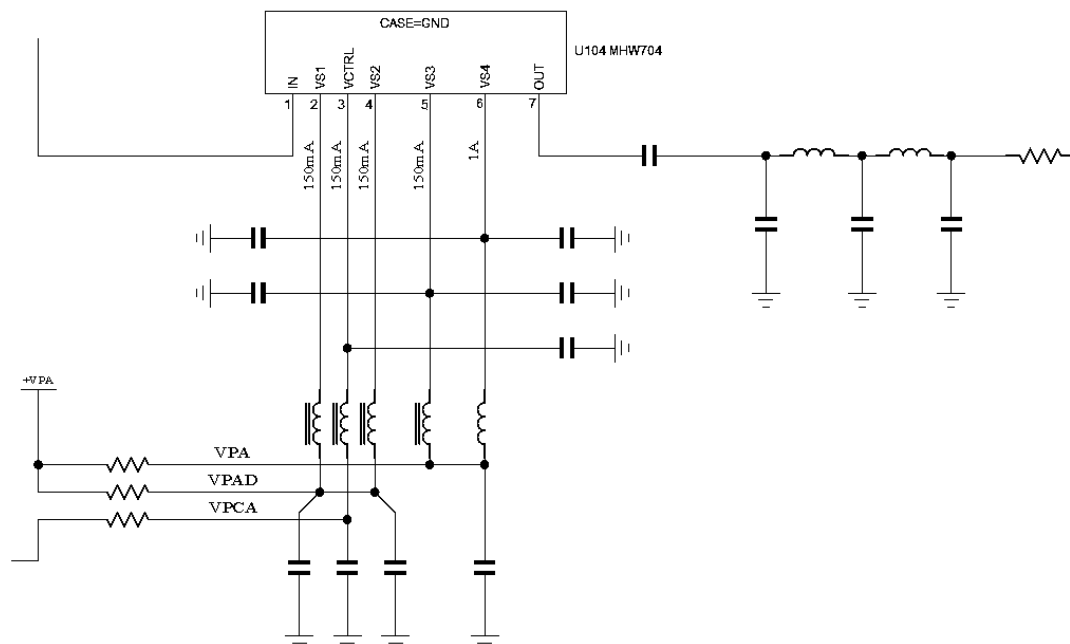


FIG. 2.16 - AMPLIFICATORE DI POTENZA IBRIDO MHW704

La funzione di commutatore di antenna invece è ottenuta con il circuito integrato MASWSS0143-V14 siglato U106. Si tratta di un dispositivo GaAs le cui principali caratteristiche sono:

- perdita di inserzione di 0.35 dB;
- isolamento pari a 22 dB;
- massima potenza +36 dBm;
- campo di temperatura compreso tra -40 °C e +85 °C.

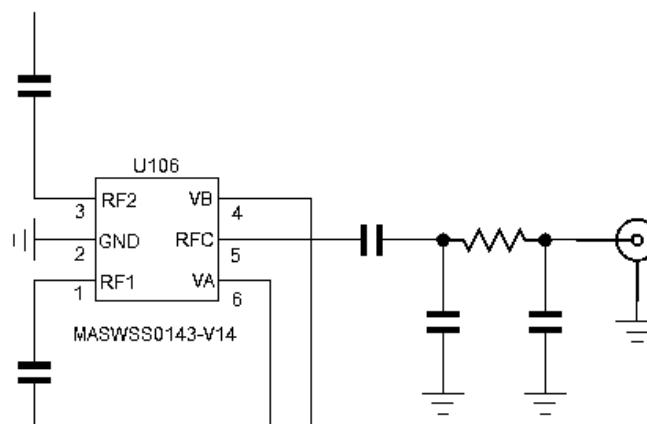


FIG. 2.17 - COMMUTATORE DI ANTENNA MASWSS0143 – V14

La misura della potenza emessa è ottenuta con il circuito integrato della Analog Device AD8307 siglato U105; la tensione in uscita fornisce direttamente il valore di tale misura.

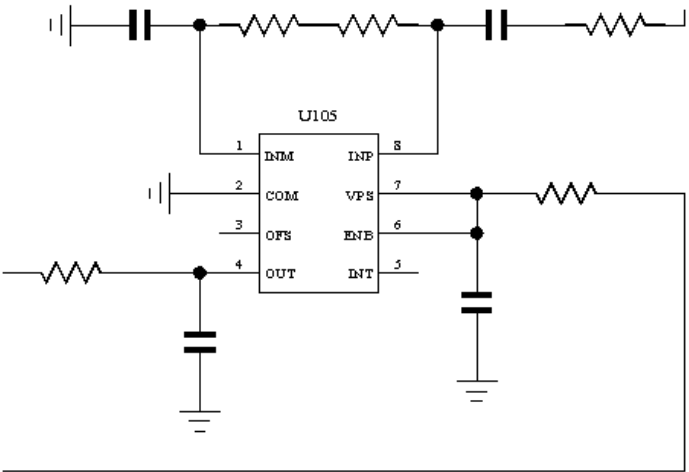


FIG. 2.18 - INTEGRATO AD8307 PER LA MISURA DELLA POTENZA EMESSA

Riassumendo, i segnali disponibili dal modulo del ricetrasmittitore vengono riportati nella seguente tabella:

N°Pin	Nome	Valore	Descrizione	Note
11,19,20	GND		Massa	
1,2	VCC	+3,3V 30 mA	Alimenta il CC1100, U102	Presente in ricezione e in trasmissione
9	VLNA	+5 V 26.2 mA	Alimenta il LNA, Q101.	Presente in ricezione. Temporizzato
13,14	VPA	+6V 2 A (+6.5 V max)	Alimenta il PA del TX, U104, MHW704.	Presente solamente in trasmissione. Temporizzato
12(?)	VPCA	Variabile da +3V a +6 V.	Tensione di controllo della potenza emessa dal PA. VCTRL,pin 3, del modulo U104, MHW704,	Presente solamente in trasmissione. Temporizzato come VPA. Viene fissata in fase di messa a punto del TX tramite la regolazione del relativo alimentatore.
18	VPS	+ 3.3 V	Alimenta il circuito di misura della potenza emessa dal TX, U105, AD8307.	Presente solamente in trasmissione. Temporizzato come VPA.
16	VAPC		Uscita della tensione funzione della potenza emessa dal TX.	Valore di tensione misurato dal OBDH per verificare il valore della potenza emessa.
17	VA	Tabella	Tensione di comando del commutatore di antenna	Temporizzato
15	VB	Tabella	Tensione di comando del commutatore di antenna	
10	T/R	+3.3 V - 1 mA 0V - 10µA	Tensione di comando del commutatore RxTx a bassa potenza	Livello logico CMOS Temporizzato
4	SCLK	Livello logico 1 = VCC 0 = GND	Clock Input	Pin 1 CC1100, U102
5	SO		Data Out	Pin 2 CC1100, U102
3	SI		Data Input	Pin 20 CC1100, U102
8	GDO0		Data In/Out	Pin 6 CC1100, U102
7	CSn		Chip Select, Input	Pin 6 CC1100, U102

FIG. 2.19 - CONNESSIONI DEL RICETRASMETTITORE

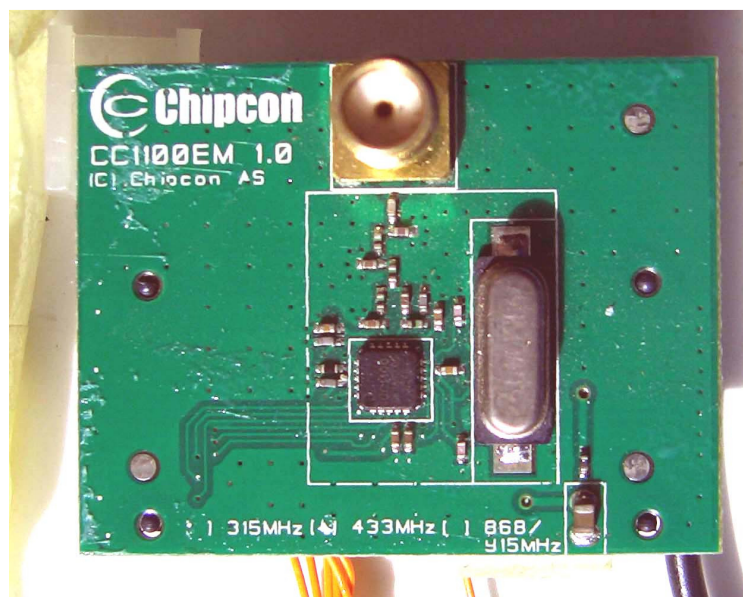


FIG. 2.20 - IMMAGINE DELLA BOARD DI PROVA DELL'INTEGRATO CC1100

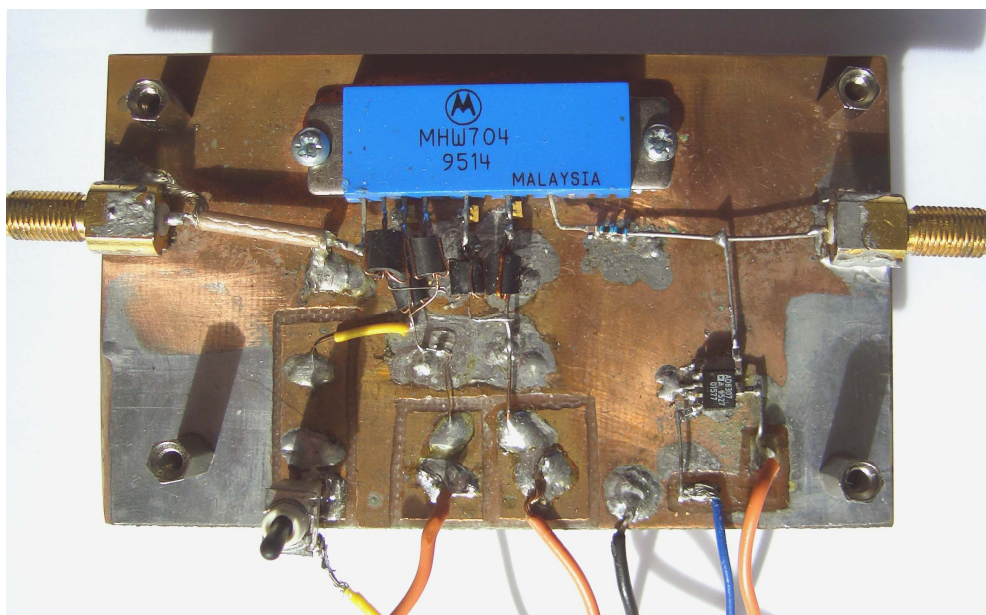


FIG.2.21 - IMMAGINE DEL PROTOTIPO DELLO STADIO DI POTENZA DEL MODULO OBR

2.5 Telemetria e storage dei dati di strumentazione

La telemetria consiste nella misurazione e la trascrizione di informazioni utili al progettista del sistema o all'operatore.

Nel caso del satellite Atmocube, la telemetria è l'insieme dei dati provenienti dai vari sensori e dai vari moduli. In una missione spaziale tali dati hanno un'importanza fondamentale in quanto servono come studio delle misure sperimentali e permettono il monitoraggio dei vari sottosistemi per poter studiare ed eventualmente agire su alcuni tipi di guasti. Per effettuare tutta questa serie di operazioni bisogna poter disporre del cosiddetto telecomando che è la controparte del sistema di telemetria ed opera nella stazione di terra. Esso consiste nell'invio di particolari messaggi atti ad alterare le impostazioni del satellite, nonché a richiedere l'invio delle sue impostazioni e dei dati acquisiti lungo la sua orbita. Tutte queste grandezze di telemetria vengono gestite dal microcontrollore presente nel modulo OBDH & HK di Atmocube e pertanto un'opportuna parte di questo sistema deve poter integrare delle particolari funzioni dedicate a tal scopo. Le grandezze di telemetria considerate nel progetto in esame sono:

- ***i dati strumentali;***
- ***l'occupazione della memoria da parte di dati strumentali;***
- ***la memoria residua;***
- ***la potenza emessa dal segnale del trasmettitore;***
- ***orientazione del satellite*** (informazioni ricavate dai fotodiodi posti su ogni faccia del cubo della struttura meccanica);
- ***le tensioni di alimentazione*** (tensione dei pannelli solari, tensione della batteria);
- ***le correnti di alimentazione*** (corrente dei pannelli solari, corrente della batteria);
- ***le temperature*** (temperatura della batteria, temperatura dei componenti di potenza).

Le ultime tre grandezze presenti nell'elenco vengono classificate come telemetria di housekeeping, da cui deriva parte del nome del modulo trattato in questo elaborato.

Dal momento che lo scopo della telemetria è quello di trasmettere alla stazione di terra le grandezze citate e visto che il satellite non è sempre in contatto radio, è necessario disporre di un'opportuna struttura di stoccaggio dei dati, ovvero un'adeguata memoria. Le specifiche sulla capacità di questo componente sono note già nella fase iniziale di progetto del modulo OBDH & HK e sono state calcolate ad un ammontare di 2 Mbyte di dati. Noto che lo scopo fondamentale della missione di questo satellite è quello di effettuare misure e registrarle con un'assidua frequenza, la memoria in dotazione è uno dei componenti maggiormente coinvolti e sollecitati; si vedrà a tal proposito che i cicli di lettura e scrittura sono una specifica importante per la scelta della memoria dato che il numero supera di gran lunga il milione.

2.6 *Power Management*

È prevista una gestione completa dell'energia disponibile a bordo del satellite in modo da massimizzare l'efficienza e la durata della batteria agli ioni di litio.

Il controllo del piano energetico prende il nome di Power Management e nel progetto in esame agisce sia sugli alimentatori sia sui consumi del microcontrollore stesso.

Per quanto riguarda la Power Supply Unit del satellite, a livello dell'hardware, sono disponibili diverse linee di abilitazione dei singoli alimentatori. Questo è stato previsto al fine di permettere al microcontrollore di abilitare o meno alcune parti e sottosistemi di Atmocube in modo da gestire tramite un software, e quindi un piano programmato, l'energia disponibile nell'accumulatore.

Per ciò che concerne invece la parte di risparmio energetico sui consumi del microcontrollore, si cerca di implementare sul satellite un componente a basso assorbimento di corrente con eventuali funzioni di gestione software degli oscillatori, degli stati di Idle e dello stato di Sleep.

3 Scelta della memoria esterna e del microcontrollore

Si dedica un intero capitolo alla discussione della selezione di questi componenti perché sono stati i punti di partenza cruciali e fondamentali per il progetto completo del modulo OBDH & HK. In particolare, la memoria esterna di immagazzinamento dei dati è stato il componente che ha condizionato molte parti del progetto stesso, dalla sintesi di altre architetture hardware correlate, alla scelta stessa del microcontrollore impiegato. Si può dire quindi che la memoria sia stata il dispositivo che ha fatto dipendere a livello globale molteplici e svariate scelte progettuali dell'intero sistema di controllo.

3.1 Memoria esterna

Le prime ricerche riguardo questo dispositivo si sono orientate fin da subito verso una memoria di tipo non volatile di capacità compresa tra 1 e 2 Mbyte; questo perché è noto che l'algoritmo di governo del satellite prevede uno stato di emergenza nel quale è presente una condizione di bassa energia con una possibile perdita dei dati non opportunamente salvati.

L'elevato numero di cicli di scrittura e lettura dichiarati nelle specifiche di telemetria del capitolo 2 portano ad eliminare come possibili candidate la maggior parte delle memorie di tipo non volatile, quali *EEPROM* e *FLASH*. Infatti le prime ricerche evidenziano che il numero di programmazioni garantito dai costruttori oscilla tra 100.000 e 1.000.000.

Uno dei primi componenti analizzati nella fase di selezione è stata una memoria flash seriale M25P80 della ST gestibile per mezzo dello standard SPI; con essa però non viene soddisfatto il numero minimo di cicli di programmazione e cancellazione.

Un secondo tipo di memoria appartenente alla categoria delle non volatili è stata analizzata: la Am29LV081B della AMD; si tratta sempre di un componente di tipo flash ma con un'architettura parallela. In questo caso il numero minimo di programmazioni è pari a 1.000.000, valore più vicino alle soglie accettabili ma non ancora sufficienti.

L'analisi effettuata su altri campioni di memorie portano alla conclusione di non poter soddisfare pienamente le specifiche di telemetria per mezzo di un unico componente con capacità pari a quella necessaria. Una soluzione abbastanza originale pensata per evitare questo problema consiste nell'impiegare una memoria con capacità pari ad un certo numero di volte quella originale, ad esempio 4 o 8 volte più ampia. Dato che generalmente i cicli di programmazione sono riferiti per ogni singolo settore, avendo più settori disponibili si può di conseguenza moltiplicare il numero di cicli totali disponibili.

Una seconda soluzione parziale al problema è ottenuta utilizzando due memorie, una flash ad alta capacità e una ram a bassa capacità. La procedura di immagazzinamento consiste nell'accumulare prima diversi dati in ram per poi trasferirli in flash, facendo accesso a quest'ultima un numero inferiore di volte.

Le due soluzioni elaborate vengono scartate, in quanto nel primo caso non si intende sviluppare un software troppo complicato di gestione dei settori, mentre nel secondo caso non si vogliono utilizzare più componenti.

A questo punto si passa alla ricerca di una memoria di capacità adeguata ma di tipo RAM, in modo da svincolarsi totalmente, causa la natura di questi dispositivi, dal numero di programmazioni possibili.

Le memorie disponibili di tipo ram seriale raggiungono una capacità pari a frazioni di 2 Mbyte risultando inadeguate al sistema in uso; una delle prime memorie ram seriali reperibili nei cataloghi dei più noti distributori, la Ramtron FM25040, ha una capacità massima di appena i 4 kbit. Questa memoria ram, denominata FRAM cioè Ferromagnetic RAM, ha la caratteristica di essere allo stesso tempo non volatile, conservando i dati anche in condizioni di assenza di alimentazione. I dati vengono mantenuti grazie alla tecnologia di costruzione della memoria stessa che impiega materiali ferromagnetici.

Alla fine di queste considerazioni, si evince che l'unica tipologia con capacità di 2 Mbyte e con nessun limite riguardo il numero di cicli di programmazioni è una memoria RAM di tipo parallelo. Si tralasciano sia memorie ram di tipo dinamico, per evitare continui refresh, sia memorie ram di tipo sincrono, per evitare segnali di sincronismo e visto il fatto che in questo progetto non sono necessari tempi di accesso molto brevi. La scelta finale cade quindi su una SRAM asincrona, cioè una memoria ram statica priva di segnali di

sincronizzazione. La caratteristica di essere statica, inoltre, permette lo stoccaggio dei dati almeno fino a quando non viene completamente tolta l'alimentazione, quindi, per certi versi e in certe condizioni, essa può essere considerata non volatile.

Generalmente le memorie statiche hanno il grosso inconveniente di consumare parecchia corrente, di conseguenza è indispensabile ricercarne una a bassi consumi di tipico impiego per applicazioni a batteria.

Dopo un'accurata ricerca è stata individuata una memoria SRAM parallela adeguata all'immagazzinamento dei dati di telemetria: si tratta di una Cypress CY62167DV30 con capacità di 16 Mbit. La Cypress CY62167DV30 è una memoria ad alte prestazioni costruita in tecnologia CMOS. Le caratteristiche avanzate di progetto dei circuiti che la compongono fanno sì che la corrente attiva di funzionamento sia estremamente bassa rendendola ideale per applicazioni di tipo portatile come ad esempio telefoni cellulari. Tra i dati caratteristici compare infatti la sigla MoBL che identifica la caratteristica di More Battery Life. La tensione di alimentazione è compresa tra 2.20 V e 3.60 V. La memoria inoltre dispone di una funzione di power down che permette di ridurre i consumi praticamente a zero; questa modalità viene attivata ogni volta che il pin CE disabilita il componente.

La figura 3.1 rappresenta lo schema a blocchi della struttura della memoria:

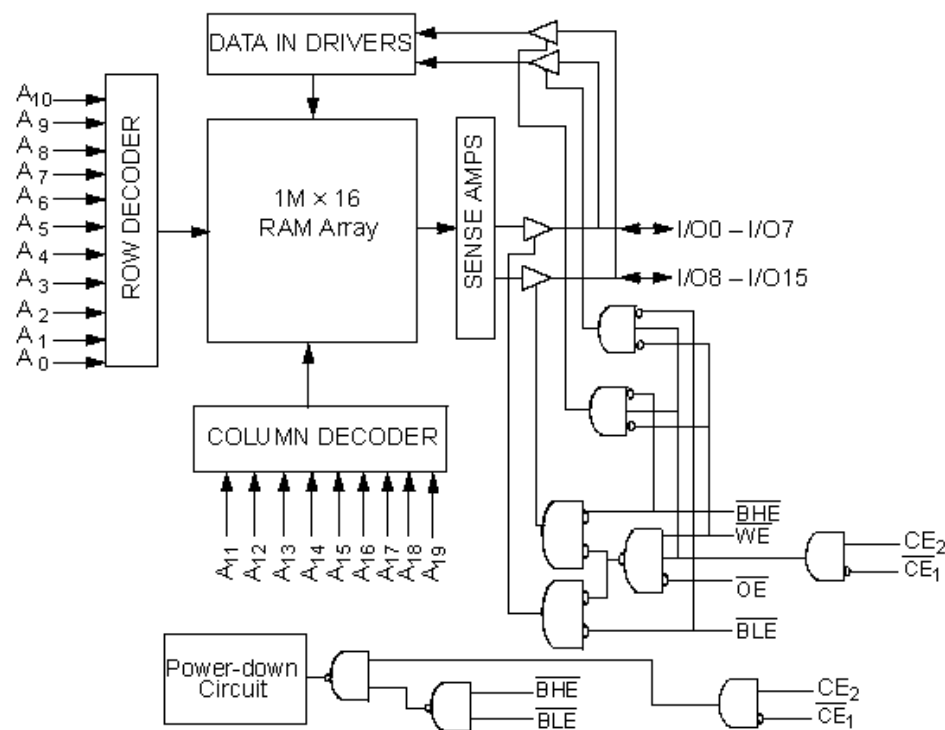


FIG. 3.1 - SCHEMA A BLOCCHI DELL'ARCHITETTURA DELLA MEMORIA CYPRESS CY62167DV30

La lettura dei dati in memoria avviene secondo la seguente temporizzazione:

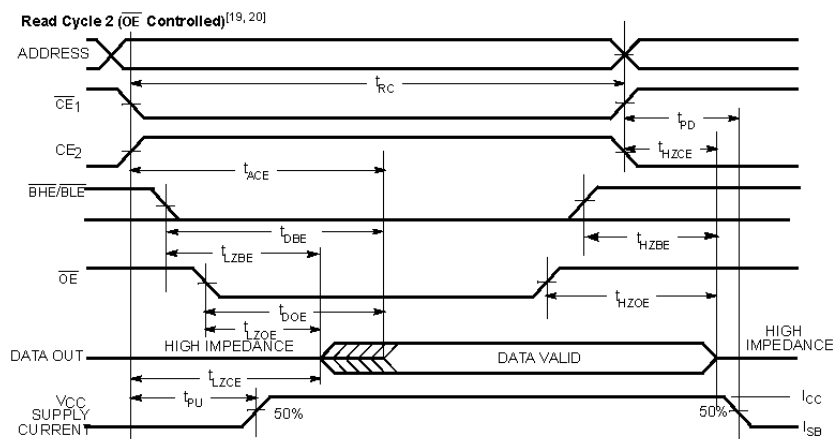


FIG. 3.2 - CICLO DI LETTURA DELLA MEMORIA CYPRESS

La lettura di un dato in memoria è effettuata selezionando la memoria ($\overline{CE_1}=0$ e $CE_2=1$), abilitando l'uscita ($\overline{OE}=0$) e disabilitando la scrittura ($\overline{WE}=1$).

Se l'abilitazione del byte meno significativo è attivato ($\overline{BLE}=0$), allora i dati della memoria all'indirizzo specificato compaiono sulle linee I/O_{0-7} . Se l'abilitazione del byte più significativo è attivato ($\overline{BHE}=0$), allora i dati della memoria all'indirizzo specificato, compaiono sulle linee I/O_{8-15} .

La scrittura dei dati in memoria avviene secondo la seguente temporizzazione:

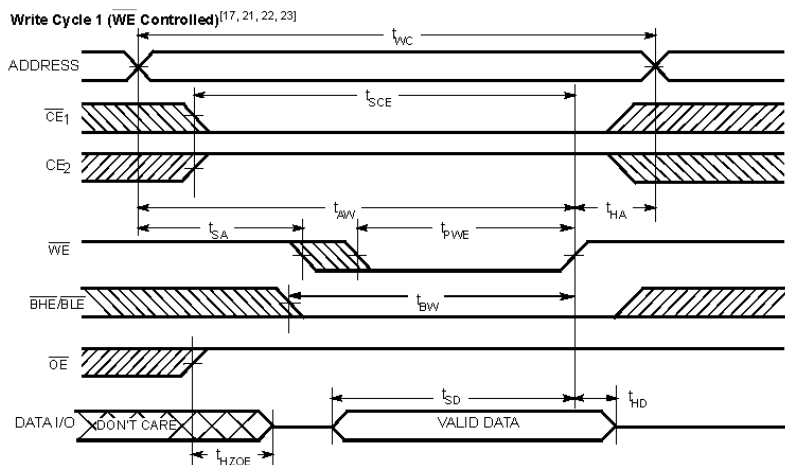


FIG. 3.3 - CICLO DI SCRITTURA DELLA MEMORIA CYPRESS

La scrittura di un dato in memoria è effettuato selezionando la memoria ($\overline{CE_1}=0$ e $CE_2=1$) e abilitando la scrittura ($\overline{WE}=0$).

Se l'abilitazione del byte meno significativo è attivato ($\overline{BLE}=0$), allora i dati all'indirizzo specificato, vengono scritti sulla memoria utilizzando le linee I/O_{0-7} . Se l'abilitazione del byte più significativo è attivato ($\overline{BHE}=0$), allora i dati all'indirizzo specificato, vengono scritti sulla memoria utilizzando le linee I/O_{8-15} .

Il dispositivo può essere posto in modalità a basso consumo per mezzo degli ingressi di abilitazione $\overline{CE_1}$ e $\overline{CE_2}$ o per mezzo degli ingressi di selezione dei byte $\overline{BHE}$ e $\overline{BLE}$. Le linee di dato I/O_{0-15} possono essere messe in condizione di alta impedenza deselectionando la memoria con gli ingressi $\overline{CE_1}$ e $\overline{CE_2}$, disabilitando le uscite con il controllo $\overline{OE}$, disabilitando i byte con $\overline{BHE}$ e $\overline{BLE}$ e ancora effettuando un'operazione di scrittura per mezzo del comando $\overline{WE}$.

Le combinazioni possibili dei vari controlli vengono riportate nella tabella di verità:

$\overline{CE_1}$	$\overline{CE_2}$	$\overline{WE}$	$\overline{OE}$	$\overline{BHE}$	$\overline{BLE}$	Inputs/Outputs	Mode	Power
H	X	X	X	X	X	High Z	Deselect/Power-Down	Standby (I_{SB})
X	L	X	X	X	X	High Z	Deselect/Power-Down	Standby (I_{SB})
X	X	X	X	H	H	High Z	Deselect/Power-Down	Standby (I_{SB})
L	H	H	L	L	L	Data Out ($I/O_0 - I/O_{15}$)	Read	Active (I_{CC})
L	H	H	L	H	L	Data Out ($I/O_0 - I/O_7$); High Z ($I/O_8 - I/O_{15}$)	Read	Active (I_{CC})
L	H	H	L	L	H	High Z ($I/O_0 - I/O_7$); Data Out ($I/O_8 - I/O_{15}$)	Read	Active (I_{CC})
L	H	H	H	L	H	High Z	Output Disabled	Active (I_{CC})
L	H	H	H	H	L	High Z	Output Disabled	Active (I_{CC})
L	H	H	H	L	L	High Z	Output Disabled	Active (I_{CC})
L	H	L	X	L	L	Data In ($I/O_0 - I/O_{15}$)	Write	Active (I_{CC})
L	H	L	X	H	L	Data In ($I/O_0 - I/O_7$); High Z ($I/O_8 - I/O_{15}$)	Write	Active (I_{CC})
L	H	L	X	L	H	High Z ($I/O_0 - I/O_7$); Data In ($I/O_8 - I/O_{15}$)	Write	Active (I_{CC})

FIG. 3.4 - TABELLA DI VERITA' DEI CONTROLLI DELLA MEMORIA CYPRESS

Riassumendo, le caratteristiche che prediligono l'uso di questo tipo di memoria vengono elencate di seguito assieme alla giustificazione della scelta delle stesse:

- **memoria RAM:** una memoria ad accesso casuale non ha nessun vincolo riguardo il numero di cicli di scrittura e lettura;
- **memoria asincrona:** non sono richieste velocità di accesso particolarmente elevate, di conseguenza non servono segnali di sincronismo;
- **memoria statica:** si vuole ottenere un'unità di immagazzinamento stabile senza il bisogno di alcun refresh;
- **memoria parallela:** questo tipo di configurazione è una scelta obbligata visto la capacità consistente di 2 Mbyte.

Il fatto di non disporre di una memoria non volatile viene parzialmente compensato dal fatto che i dati vengono persi solo se l'alimentazione scende al di sotto di 1.5 V, cioè la tensione minima di stoccaggio. Dato che questa condizione a meno di qualche guasto non si dovrebbe verificare mai, l'integrità dei dati viene praticamente quasi sempre garantita.

48-Lead TSOP I (12 mm x 18.4 mm x 1.0 mm) Z48A

DIMENSIONS IN INCHES[MM] MIN.

MAX.

JEDEC # MO-142

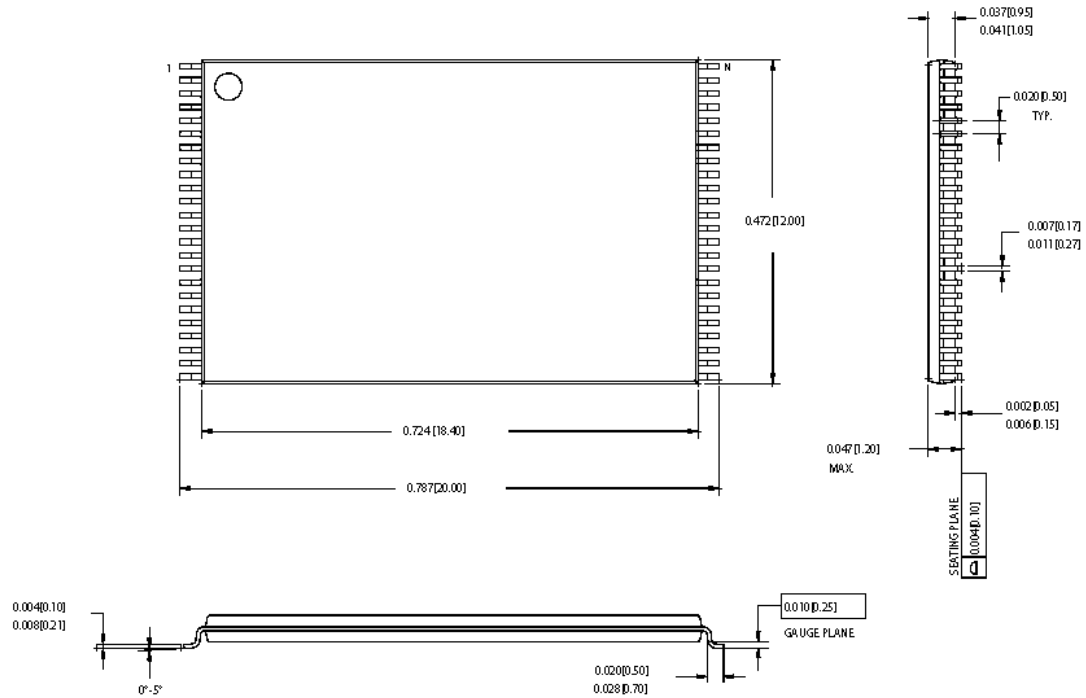


FIG. 3.5 - DISPOSIZIONE DELLA PIEDINATURA E DIMENSIONI DELLA MEMORIA NEL CONTENITORE TSOP A 48 PIN

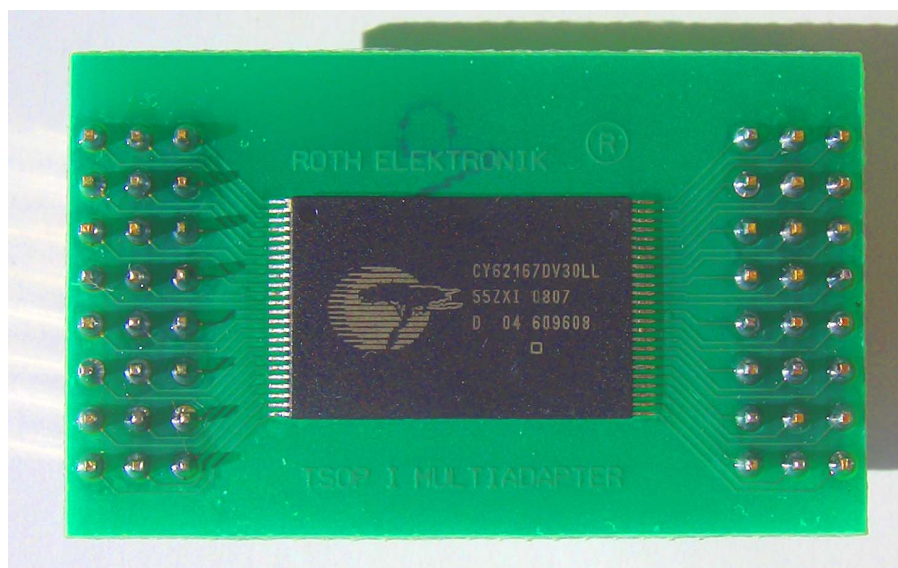


FIG. 3.6 - MEMORIA MONTATA SU UNO ZOCCOLO DI ADATTAMENTO PER IL PROTOTIPO DEL MODULO OBDH & HK

3.2 **Microcontrollore Microchip 18LF8722**

Per poter effettuare una scelta adeguata del microcontrollore da impiegare come nucleo del modulo OBDH & HK è necessario avere a priori una visione particolareggiata dell'intero sistema. Di conseguenza, oltre allo studio della memoria è stato fatto uno studio dei vari sottosistemi del satellite con lo scopo di identificare quante linee sono necessarie per il completo interfacciamento e che tipo di periferiche si rendono utili o indispensabili per svolgere le varie funzioni. Questi appena visti sono gli aspetti fondamentali per un'adeguata ricerca del componente e sono stati i capisaldi sui quali essa si è basata.

Di seguito si riportano le caratteristiche che deve possedere il microcontrollore per adempiere alle specifiche progettuali:

- **numero adeguato di porte:** uno degli svantaggi nell'utilizzo di una memoria ad accesso parallelo è dato dal gran dispendio dal punto di vista del cablaggio. Generalmente una memoria di questo tipo della capacità vista ha bisogno di 20 o 21 linee di indirizzo, 8 o 16 linee di dato e alcune linee di controllo. Considerando l'aggiunta delle linee di tutti gli altri sottosistemi, si arriva alla conclusione che è necessario implementare un microcontrollore a 80 o 100 pin con approssimativamente una sessantina di input/output;
- **periferica di controllo della memoria SRAM:** è chiaro fin da subito che risulta non fattibile costruire manualmente un protocollo di comunicazione tra il microcontrollore e la memoria. Questo perché risulta estremamente scomodo, e a maggior ragione lento, caricare il dato di indirizzamento della locazione di memoria. Essendo questo infatti una parola di 20 o 21 bit, esso si estende generalmente su tre porte del microcontrollore rendendosi quindi necessarie diverse istruzioni macchina per gestire le porte, inclusi shift dei singoli bit nei registri. E' quindi indispensabile ricercare un microcontrollore con la disponibilità di una periferica di gestione delle memorie di tipo parallelo;
- **periferica di comunicazione seriale SPI:** dato che ben tre moduli del satellite impiegano lo standard SPI per la trasmissione e la ricezione dei dati, è utile disporre di una periferica di gestione di tale protocollo;
- **periferica di comunicazione seriale UART:** si rende comoda anche l'uso di questo tipo di periferica vista l'intenzione di creare un'interfaccia di debug con il PC e visto il fatto che il GPS si serve pure di questa modalità di interfacciamento;
- **convertitore analogico-digitale:** nel sistema in questione sono presenti diversi segnali analogici che devono essere acquisiti e analizzati; è utile quindi scegliere un microcontrollore che disponga di questo tipo di convertitore già integrato nel core;
- **consumi ridotti:** criterio generale, del resto già adottato prima per la memoria ma anche per gli altri moduli del sistema, è l'utilizzo di componenti a basso consumo in modo da massimizzare l'autonomia della batteria. Si ricerca quindi un

microcontrollore a basso consumo con magari la disponibilità di commutare tra i vari stati di potenza;

- **memoria programma ampia:** non avendo ancora ben a disposizione tutte le specifiche software, si tendono a prediligere microcontrollori in grado di contenere un codice macchina pesante in modo da poter garantire l'implementazione di qualsiasi tipo di software;
- **tensione di funzionamento:** visto l'alimentazione primaria a 3.7 V e i protocolli di comunicazione SPI a 3.3 V, in particolare quello della radio, si necessita di un microcontrollore il cui funzionamento sia garantito a queste tensioni di lavoro.

All'inizio del progetto del modulo OBDH & HK, tra la documentazione disponibile, era indicato il possibile impiego del microcontrollore Renesas H8 38076R. Allora non era stato ancora affrontato il problema della memoria di telemetria sia per quanto riguarda la tipologia sia per quanto riguarda l'interfacciamento. I problemi preponderanti di questo tipo di microcontrollore sono che non dispone di alcuna periferica di gestione della memoria parallela e che non ha una memoria programma estesa.

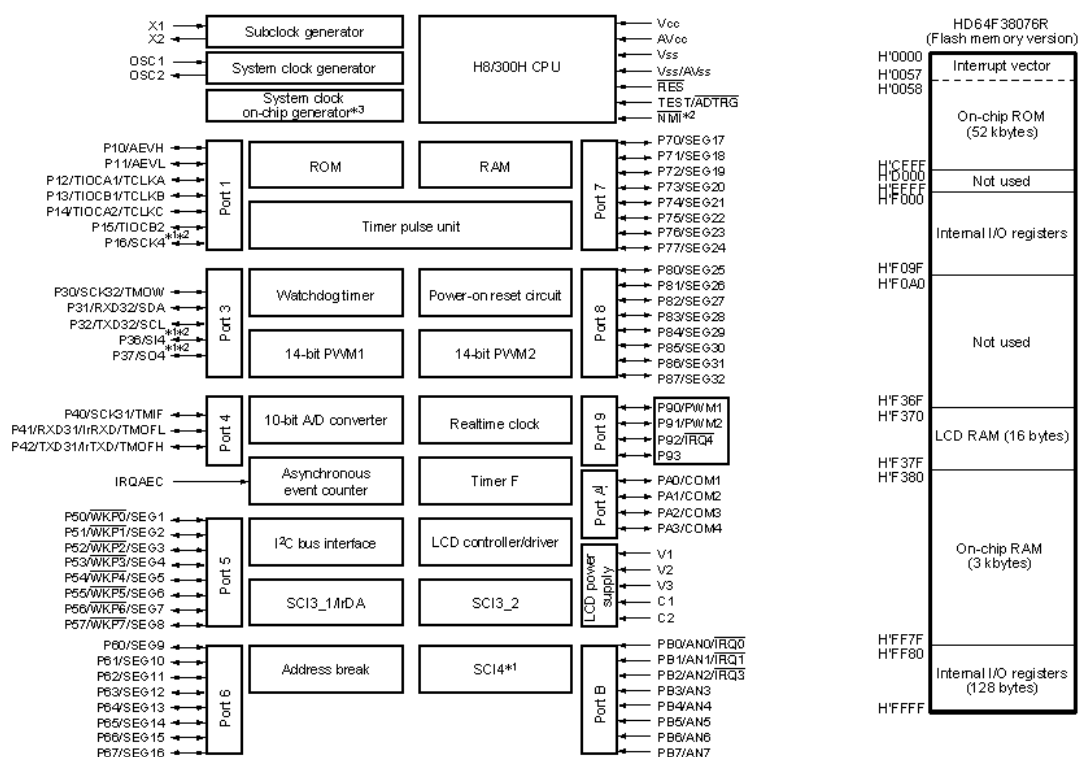


FIG. 3.7 - SCHEMA A BLOCCHI DELL'ARCHITETTURA DEL MICROCONTROLLORE RENESAS E DELLA MEMORIA RAM E ROM

Dallo schema a blocchi si nota l'ampio numero di porte utili per poter interfacciare tutti i sottosistemi del satellite, come anche la presenza delle periferiche di comunicazione seriale sincrone e asincrone siglate SCI3 e SCI4. È presente inoltre un controller per la gestione di un display LCD che però non trova impiego nel progetto in esame. Come accennato è assente una periferica di controllo per una memoria parallela.

La colonna a destra della figura 3.7 rappresenta l'organizzazione della memoria del microcontrollore Renesas. Si notano che gli indirizzi da 0x0058 a 0xCFFF identificano l'intervallo nel quale è residente la memoria ROM e cioè la memoria riservata al codice macchina eseguibile dal microcontrollore. Tale area di 52 kbyte risulta limitata o comunque non sufficientemente ampia per lo sviluppo del software di controllo globale.

Dopo l'analisi effettuata, il precedente microcontrollore viene abbandonato come anche il nucleo basato sull'Hitachi e cioè l'H8 300. L'orientamento cambia completamente verso altri tipi di microcontrollori e si considerano quelli della Microchip. Quest'ultimi sono microcontrollori generalmente più costosi, con un'architettura per certi versi più particolare e più spartana, tuttavia ben si adattano allo scopo.

Visto il fatto che in nucleo 18F è già stato visto in alcuni corsi universitari e visto che la documentazione e le application notes sono facilmente reperibili e ampiamente disponibili, questi microcontrollori sono dei validi candidati per svolgere il compito di controllo del satellite Atmocube. La famiglia PIC18F identifica la categoria a più alte prestazioni dei microcontrollori a 8 bit; essa utilizza un'architettura che utilizza parole di programma a 16 bit e incorpora un'architettura RISC con una stack a 32 livelli di profondità, un moltiplicatore hardware 8x8 e interrupt multipli interni ed esterni. Essa raggiunge inoltre velocità di esecuzione fino a 16 MIPS e mette a disposizione una memoria interna di tipo lineare. Altra caratteristica importante è che la famiglia PIC18F risulta essere la più nota per progetti basati su una piattaforma di programmazione in ambiente C.

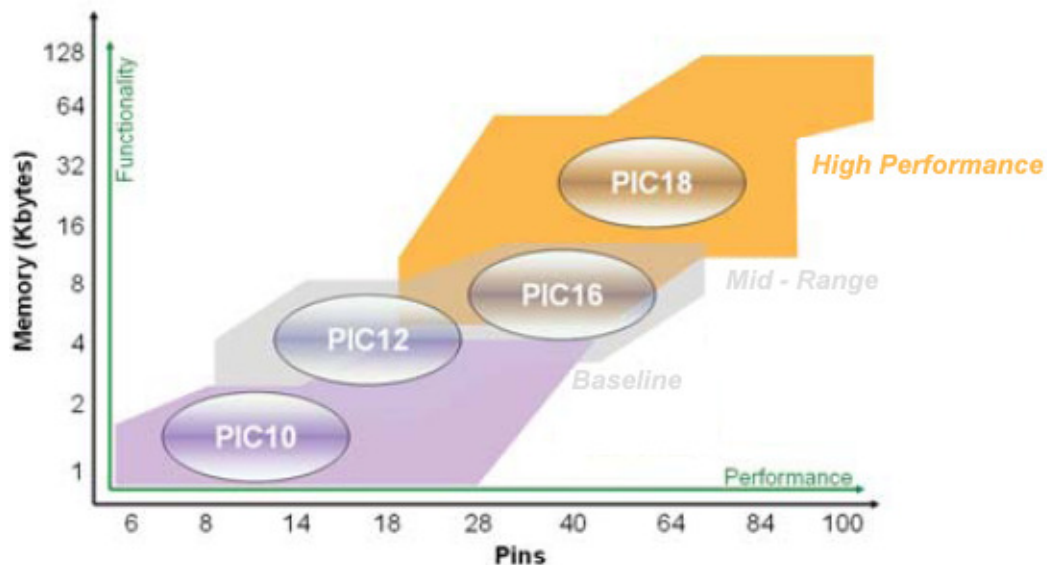


FIG. 3.8 - GRAFICO DI CONFRONTO DELLE VARIE FAMIGLIE A 8 BIT

In conclusione si cerca un microcontrollore Microchip della famiglia PIC18F basandosi sulle specifiche elencate a inizio paragrafo. La scelta cade sul PIC18F8722, in particolare sul PIC18LF8722, che è la versione equivalente ma con un intervallo di tensioni di alimentazione più vasto.

In figura 3.9 si riporta lo schema a blocchi dell'architettura del PIC18LF8722.

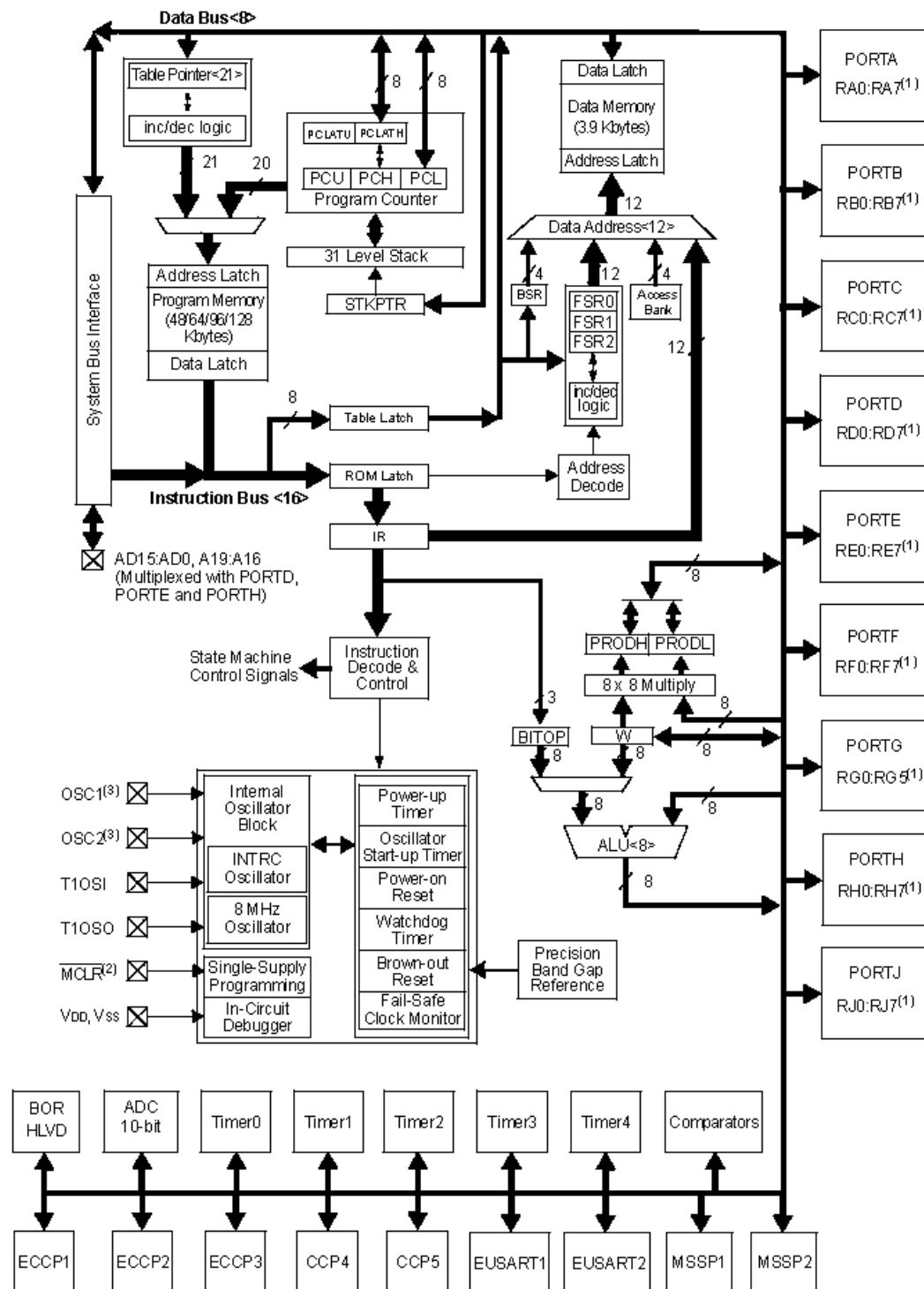


FIG. 3.9 - SCHEMA A BLOCCHI DELL'ARCHITETTURA DEL PIC18LF8722

Nota ora, in forma semplificata, la struttura del dispositivo, si riprendono dall'inizio del capitolo i requisiti che il microcontrollore in uso nel modulo OBDH & HK deve possedere e li si ripercorrono evidenziando il loro soddisfacimento attraverso la descrizione degli stessi nel PIC18LF8722.

- **Numero adeguato di porte:** il microcontrollore in questione dispone di 80 pin e ben 70 sono impiegati come Input ed Output; a tal scopo infatti sono accessibili via software 9 porte denominate dalla lettera A alla lettera J. In verità nelle versioni a 80 pin, alcuni di essi vengono condivisi per più funzionalità, di conseguenza il numero effettivo di pin disponibili per collegamenti esterni sono leggermente inferiori. Il pin 5 della porta G, cioè l'RG5, ad esempio viene condiviso con il reset hardware del microcontrollore, quindi volendo implementare quest'ultima funzione esso non sarà utilizzabile al di fuori di questo scopo. I pin RA6 e RA7 invece sono condivisi con gli ingressi per l'oscillatore esterno primario e di conseguenza anche quest'ultimi verranno esclusi da funzioni di interfacciamento perché verrà utilizzato un oscillatore esterno come sorgente di clock;
- **periferica di controllo della memoria SRAM:** la famiglia PIC18F8722 implementa la EMI, External Memory Interface. Nasce originariamente con lo scopo di espandere la memoria programma nello sfortunato caso in cui la memoria flash sia insufficiente ad ospitare l'intero codice macchina. A tal fine, questa periferica permette al Program Counter del controllore di indirizzare fino a 2 Mbyte di memoria. La EMI, tramite le sue modalità di gestione, offre diverse funzionalità:
 - gestione del microcontrollore interamente dalla memoria esterna;
 - impiego di una combinazione di memoria interna ed esterna sempre con limite massimo a 2 Mbyte;
 - uso di una memoria flash per applicazioni di riprogrammazione del codice o tabelle di dati molto ampie;
 - uso di dispositivi RAM per l'immagazzinamento di grandi quantità di dati o variabili.

L'ultima opzione disponibile ben si adatta al controllo della SRAM impiegata per i dati di telemetria;

- **periferica di comunicazione seriale SPI:** per quanto concerne la comunicazione seriale sincrona, da questo tipo di dispositivo vengono messi a disposizione due moduli MSSP, Master Synchronous Serial Port, completamente indipendenti tra loro, capaci di supportare entrambi gli standard SPI e I^2C in modalità Master e Slave;
- **periferica di comunicazione seriale UART:** le comunicazioni asincrone invece, come quella utilizzata per l'interfacciamento con il PC e quella utilizzata con il GPS, vengono permesse grazie alla presenza di altri due moduli denominati EUSART, Enhanced Universal Receiver Transmitter, anche loro completamente indipendenti;

- **convertitore analogico-digitale:** tra le varie periferiche presenti all'interno del nucleo del PIC18LF8722 è disponibile un convertitore AD a 10 bit. I canali analogici che fanno capo alle porte del microcontrollore sono selezionabili tramite un multiplexer interno e sono disponibili in una quantità pari a sedici. Come si vedrà successivamente non si sfrutterà questo multiplexer ma se ne adotterà uno esterno per non occupare troppi pin del dispositivo;
- **consumi ridotti:** il PIC18LF8722 dispone della tecnologia nanoWatt; tramite essa si è in grado di ridurre drasticamente il consumo di potenza agendo in diversi modi:
 - **Run Modes alternati:** fornendo al controller la sorgente di clock dal Timer1 o dal blocco dell'oscillatore interno, il consumo di potenza durante l'esecuzione del codice può essere drasticamente ridotto;
 - **Idle Modes multipli:** un altro modo per ridurre i consumi consiste nel porre il microcontrollore nella modalità Idle, cioè mantenere attive tutte le periferiche e disabilitare la CPU;
 - **On-the-fly Mode Switching:** le varie modalità di power management possono essere richiamate dal codice permettendo all'utente di incorporare applicazioni di gestione della potenza nel progetto del software;
 - **basso consumo nei moduli base:** i consumi per quanto riguarda il Timer1, che è la sorgente di clock a bassa frequenza, e il Watchdog Timer sono per loro natura ridotti.
- **memoria programma ampia:** tutti i tipi di memoria disponibili nel microcontrollore impiegato sono presenti in quantità abbondanti. La memoria RAM interna adibita allo stoccaggio delle variabili programma e dei vari registri di configurazione e funzionamento è pari a quasi 4 kbytes, 3936 bytes per la precisione. La memoria EEPROM adibita all'immagazzinamento di dati non volatili è pari a 1 kbyte. La memoria programma invece, che è una memoria di tipo *FLASH*, è pari a più del doppio di quella vista per il primo microcontrollore in uso e raggiunge i 128 kbytes.
- **tensione di funzionamento:** come tutti i dispositivi Microchip 18F, anche i membri della famiglia PIC18F8722 sono disponibili sia nelle versioni standard che nelle versioni a bassa tensione. I dispositivi standard con la memoria di tipo Enhanced Flash, identificati con la lettera "F" nella part number, supportano una tensione di funzionamento V_{dd} che cade nell'intervallo tra 4.2 V e 5.5 V. I dispositivi a bassa tensione identificati con "LF", come nel caso del microcontrollore designato, garantiscono il funzionamento in un range di tensione esteso, cioè tra 2.0 V e 5.5 V.

Features	PIC18F8527	PIC18F8622	PIC18F8627	PIC18F8722
Operating Frequency	DC – 40 MHz	DC – 40 MHz	DC – 40 MHz	DC – 40 MHz
Program Memory (Bytes)	48K	64K	96K	128K
Program Memory (Instructions)	24576	32768	49152	65536
Data Memory (Bytes)	3936	3936	3936	3936
Data EEPROM Memory (Bytes)	1024	1024	1024	1024
Interrupt Sources	29	29	29	29
I/O Ports	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J	Ports A, B, C, D, E, F, G, H, J
Timers	5	5	5	5
Capture/Compare/PWM Modules	2	2	2	2
Enhanced Capture/Compare/ PWM Modules	3	3	3	3
Enhanced USART	2	2	2	2
Serial Communications	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART	MSSP, Enhanced USART
Parallel Communications (PSP)	Yes	Yes	Yes	Yes
10-bit Analog-to-Digital Module	16 Input Channels	16 Input Channels	16 Input Channels	16 Input Channels
Resets (and Delays)	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT	POR, BOR, RESET Instruction, Stack Full, Stack Underflow (PWRT, OST), MCLR (optional), WDT
Programmable High/Low-Voltage Detect	Yes	Yes	Yes	Yes
Programmable Brown-out Reset	Yes	Yes	Yes	Yes
Instruction Set	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled	75 Instructions; 83 with Extended Instruction Set enabled
Packages	80-pin TQFP	80-pin TQFP	80-pin TQFP	80-pin TQFP

FIG. 3.10 - CARATTERISTICHE DELLA FAMIGLIA DEL MICROCONTROLLORE IN USO

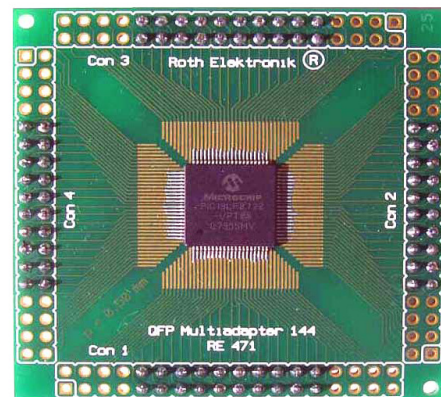
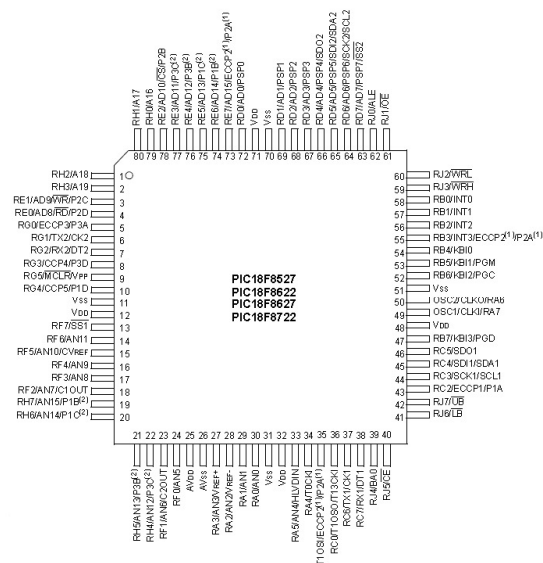


FIG. 3.11 - PIEDINATURA DEL PIC18F8722 E MONTAGGIO SU UNO ZOCCOLO DI ESPANSIONE

Si passa ora alla descrizione dettagliata dell'organizzazione della memoria del microcontrollore, in quanto essa è una parte indispensabile per capire a fondo il suo funzionamento. Ciò si renderà utile per la successiva comprensione di gran parte di questo elaborato, in particolar modo a cominciare dal prossimo capitolo, dove verrà trattata l'architettura del bus parallelo impiegato per la comunicazione con la memoria SRAM esterna.

Ci sono tre tipi di memoria nella famiglia di microcontrollori 18F: **la memoria programma, la memoria RAM dei dati e la memoria EEPROM dei dati.**

Come da manuale, seguendo un'architettura di tipo Harvard, la memoria dei dati e la memoria programma usano bus separati. La memoria *EEPROM* invece, per scopi pratici, può essere vista come una periferica del dispositivo dal momento che l'indirizzamento e l'accesso può venir eseguito attraverso alcuni registri di controllo.

Di seguito viene descritta **la Memoria programma.**

I microcontrollori della famiglia PIC18 impiegano un Program Counter a 21 bit, con il quale è possibile indirizzare fino a 2 Mbytes di spazio di memoria programma. Un accesso ad una locazione posta dopo il limite superiore della memoria fisica implementata nel particolare dispositivo ha come risultato una serie di zeri che equivalgono ad un'operazione di tipo *NOP*. Il limite superiore di indirizzamento per il PIC18LF8722 è pari appunto a 128 kbytes che corrisponde a 65.536 word di istruzioni a 16 bit.

Questa famiglia inoltre dispone di due vettori di interruzione: dopo l'indirizzo di reset posto alla locazione 0x0000, si incontra il primo interrupt alla locazione 0x0008 e il secondo alla locazione 0x0018.

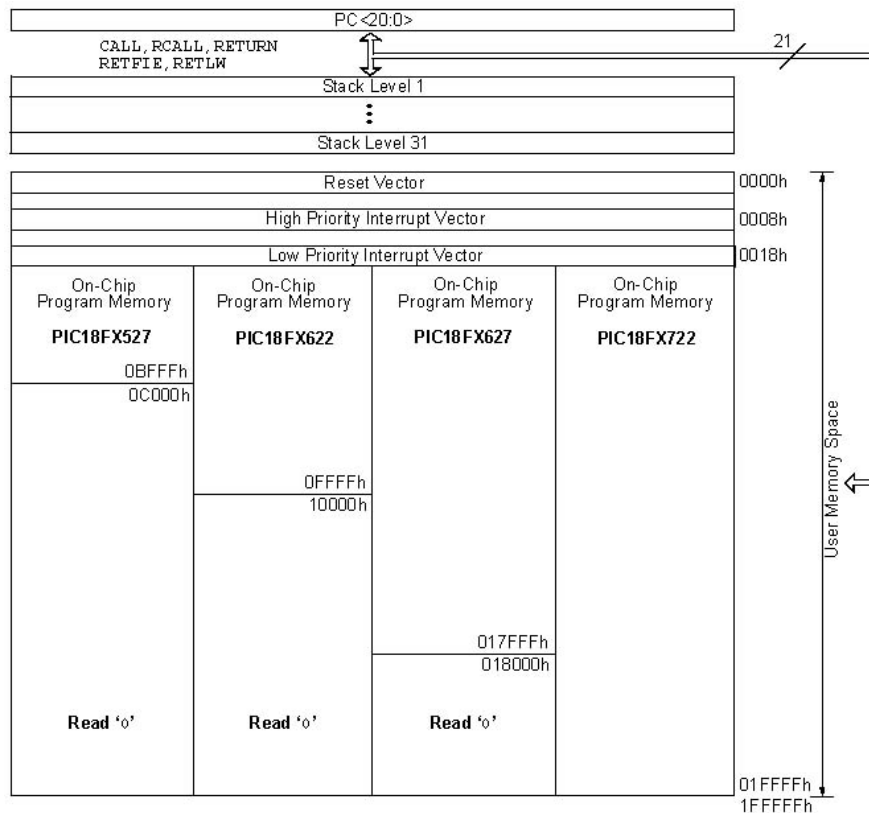


FIG. 3.12 - ORGANIZZAZIONE DELLA MEMORIA PROGRAMMA E INDIRIZZAMENTO TRAMITE PROGRAM COUNTER

Di rilevante importanza, proprio per la successiva integrazione nel progetto della memoria SRAM esterna, sono le modalità di come opera il microcontrollore. Tali modalità consistono infatti nell'indirizzamento del Program Counter fino a 2 Mbytes accedendo ad una memoria programma esterna attraverso la periferica EMI e superando di fatto il limite posto dalla dimensione della memoria fisica presente (128 kbytes).

Ci sono quattro modalità di operare con il microcontrollore in uso:

- **microprocessore (MP):** permette l'accesso solo alla memoria programma esterna; i contenuti della memoria programma interna vengono del tutto ignorati. Il Program Counter a 21 bit permette l'accesso ad uno spazio di 2 Mbytes di memoria programma lineare;
- **microprocessore con boot (MPBB):** in questa modalità si accede alla memoria interna del chip da un settore di boot. Sopra questo indirizzo il puntamento avviene verso la memoria esterna sempre rispettando il limite massimo di 2Mbytes. L'esecuzione del programma procede ad effettuare automaticamente la commutazione tra le due memorie presenti. Il settore di boot è configurabile tra 1, 2 o 4 kbytes;
- **microcontroller mode (MC):** in questa configurazione l'accesso avviene solamente verso la memoria interna del dispositivo. Il limite superiore è pertanto fissato dalla memoria fisica interna e ogni accesso ad un indirizzo superiore equivale ad un'istruzione NOP;
- **extended microcontroller mode (EMC):** in quest'ultimo modo, l'accesso è consentito ad entrambe le memorie, quella interna e quella esterna, che vengono viste dal software come un unico blocco. Il dispositivo può accedere interamente alla sua memoria interna e sopra il limite di quest'ultima, accedere alla memoria esterna rispettando ancora una volta il limite di 2 Mbytes imposto dalla natura del Program Counter a 21 bit. La commutazione tra le due memorie avviene in maniera del tutto trasparente per il programmatore in quanto l'appartenenza dei vari indirizzi nel range di 2 Mbytes viene assegnata e discriminata automaticamente.

Questa è la modalità con la quale viene controllato il microcontrollore al fine di gestire i dati di telemetria della SRAM esterna.

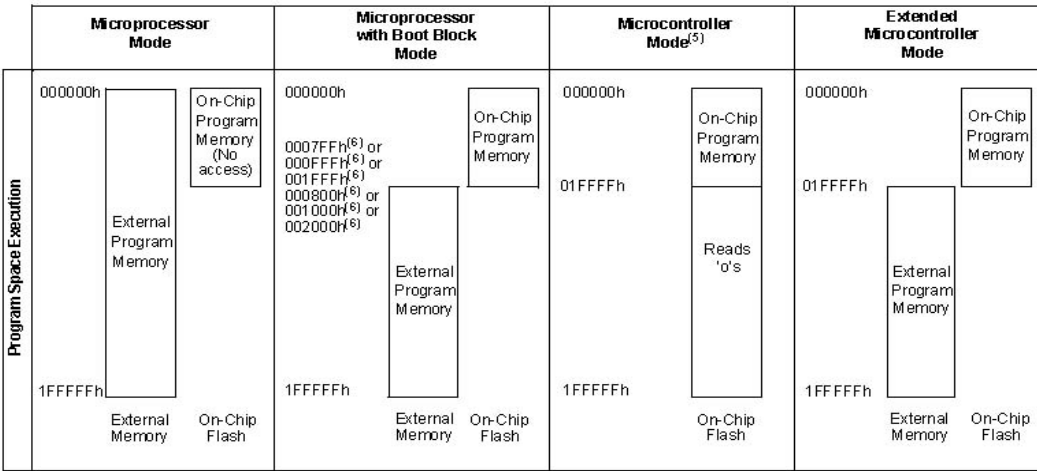


FIG. 3.13 - ORGANIZZAZIONE DELLA MEMORIA PROGRAMMA NELLE VARIE MODALITA' DI ESECUZIONE

I 21 bit che compongono il Program Counter (PC) sono contenuti in tre registri separati a 8 bit. Il byte più basso, noto come registro PCL, è sia leggibile che scrivibile. Il byte intermedio, il registro PCH, contiene i bit del Program Counter PC<15:8> e non è direttamente leggibile e scrivibile; esso viene modificato accedendo invece al registro PCLATH. Il byte più significativo è contenuto nel registro denominato PCU e contiene i bit PC<20:16>; esso viene modificato accedendo invece al registro PCLATU.

Come accennato nell'introduzione dei microcontrollori Microchip, la famiglia 18F possiede un'architettura che fa uso di istruzioni a 16 bit. Proprio per questo motivo, per prevenire allineamenti errati nei byte che compongono la singola istruzione, il Program Counter viene incrementato automaticamente di un fattore due ogni volta che esegue una singola istruzione. Ciò è permesso dal fatto che il bit meno significativo del registro PCL, cioè il bit meno significativo del PC, è tenuto fisso a 0.

Dalla figura 3.14 si nota come ogni istruzione venga suddivisa in due singoli byte e come il Program Counter incrementi di un fattore due per evitare di trovarsi in una locazione tra un byte e l'altro.

			Word Address		
			LSB = 1	LSB = 0	↓
Program Memory Byte Locations →					000000h
					000002h
					000004h
					000006h
Instruction 1:	MOVLW	055h	0Fh	55h	000008h
Instruction 2:	GOTO	0006h	EFh	03h	00000Ah
			F0h	00h	00000Ch
Instruction 3:	MOVFF	123h, 456h	C1h	23h	00000Eh
			F4h	56h	000010h
					000012h
					000014h

FIG. 3.14 - TIPICA DISPOSIZIONE DI UN'ISTRUZIONE DEI PIC DELLA FAMIGLIA 18F

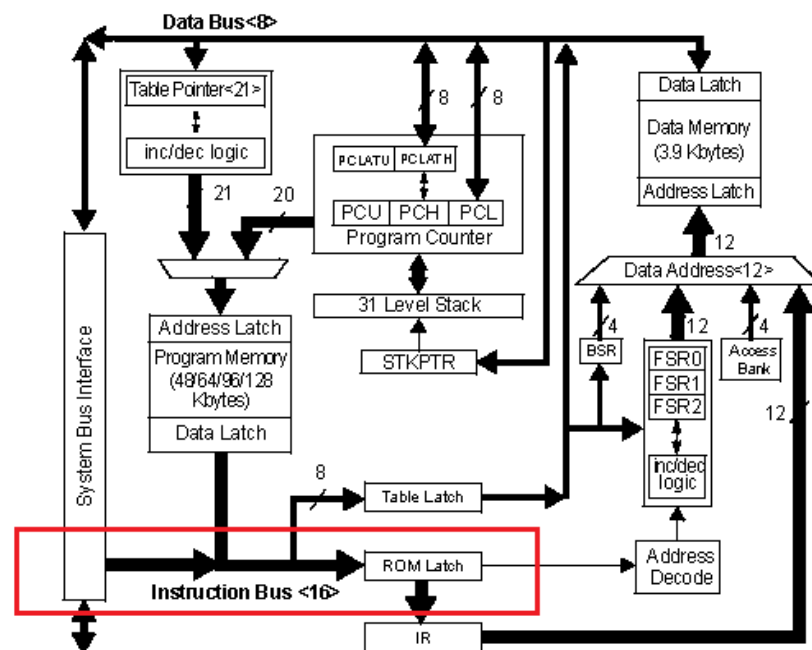


FIG. 3.15 - PARTICOLARE DELL'ARCHITETTURA DEL PIC18F8722 CON EVIDENZIATO IL BUS DI ISTRUZIONI A 16 BIT

Di seguito viene descritta **la Memoria RAM dei dati**.

La memoria RAM in questo tipo di microcontrollori viene implementata come RAM di tipo statico. Ogni registro nella memoria dei dati ha un indirizzo a 12 bit consentendo un'espansione fino a 4096 bytes. Questo spazio di memoria è diviso in 16 banchi, i quali contengono a loro volta 256 bytes.

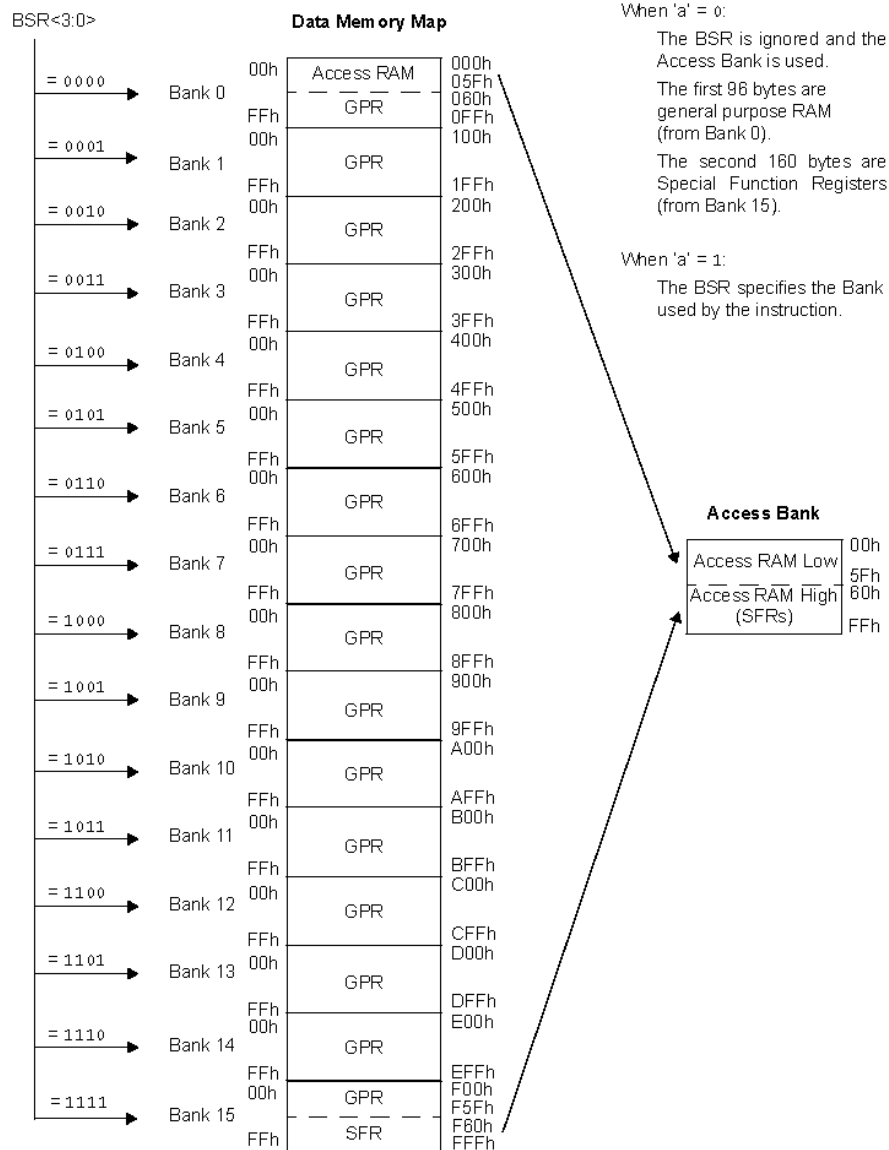


FIG. 3.16 - ORGANIZZAZIONE DELLA MEMORIA DATI DEL MICROCONTROLLORE

La memoria dati contiene dei registri cosiddetti speciali, Special Function Register, e dei registri cosiddetti generici, General Purpose Register. I registri SFR sono usati per controllare e per monitorare lo stato del controllore e le funzioni delle periferiche, mentre i registri GPR sono usati per l'immagazzinamento dei dati e le variabili del codice programma.

Il set di istruzioni e l'architettura del microcontrollore permettono l'accesso a qualsiasi banco della memoria dei dati attraverso varie modalità di puntamento.

Il puntamento dei vari banche avviene generalmente mediante il selezionatore di banco, il Bank Select Register, il quale permette di semplificare l'indirizzamento, evitando di usare una parola lunga 12 bit necessaria altrimenti a specificare il generico registro. Per identificare un registro in una locazione qualsiasi dello spazio della memoria dati, si fa ricorso a soli 4 bit per richiamare il banco, poi gli altri 8 bit rimanenti sono già inclusi nel codice dell'istruzione che si esegue; in sostanza gli 8 bit presenti nell'istruzione costituiscono un offset dal contorno del banco identificato.

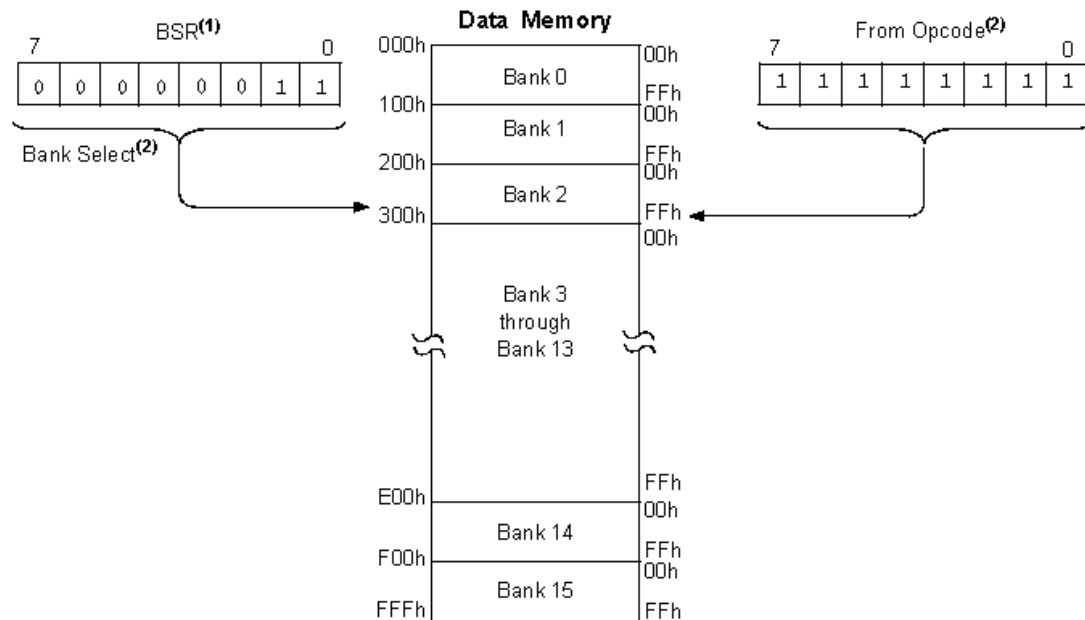


FIG. 3.17 - PUNTAMENTO DI UN GENERICO REGISTRO NELLO SPAZIO DI MEMORIA DATI

Per assicurare che i registri comunemente usati, cioè gli SFR e alcuni GPR, possano essere elaborati in un singolo ciclo macchina, quindi per assicurare buona velocità di accesso, questo tipo di microcontrollori impiega un particolare banco chiamato Access Bank. Questo è uno spazio di memoria a 256 bytes che provvede un rapido accesso a tutti gli SFR e alla porzione più bassa dei registri GPR del banco 0. In dettaglio, i primi 96 bytes della memoria dei dati (0x00 - 0x5F) costituiscono la prima parte dell'Access Bank, mentre gli ultimi 160 bytes della memoria (0xF60 - 0xFFFF) ne costituiscono la seconda parte. L'organizzazione del Access Bank è evidenziato in figura 3.16.

4 Architettura hardware del sistema di controllo

Note ora le specifiche del modulo OBDH & HK, note le caratteristiche dei singoli blocchi funzionali e nota la struttura del microcontrollore usato, si passa a tutti gli effetti al progetto dell'architettura hardware del sistema di controllo del satellite.

L'obiettivo che ci si pone fin da subito è quello di creare un sistema di tipo Embedded che sia più vicino possibile ad un microcomputer dal punto di vista sia dell'hardware che del software. Un sistema Embedded è tipicamente un progetto che fa uso della potenza di un microcontrollore. Quest'ultimo combina l'unità a microprocessore, come la CPU in un PC, con alcuni circuiti addizionali chiamati periferiche, per dar vita ad un piccolo modulo di controllo. Una volta progettato, un sistema di questo tipo viene successivamente inglobato in altri circuiti elettronici. La principale differenza tra un controllore Embedded ed un PC, è che il primo è dedicato ad uno specifico task o ad un set di task mentre un PC è progettato per elaborare differenti tipi di programmi e connette molti dispositivi esterni assorbendo di fatto molta più energia. Un controllore Embedded ha un singolo programma e in base a questo può essere integrata la potenza di calcolo e l'hardware strettamente necessari a quel dedicato task. Questi due differenti approcci che distinguono i due tipi di sistemi, nel progetto del satellite Atmocube vengono per certi versi fusi assieme o almeno avvicinati, in modo da ricavare un nuovo sistema con caratteristiche di affidabilità e stabilità.

Questo capitolo sarà dedicato esclusivamente all'hardware, di conseguenza si inizierà a descrivere la struttura hardware del satellite partendo proprio dalla parte che la rende simile all'architettura di un microcomputer, cioè l'interfacciamento con la memoria esterna per mezzo del bus parallelo.

Si riporta nella pagina seguente lo schema a blocchi della struttura hardware del modulo OBDH & HK in modo da avere una visione complessiva di come viene organizzato il sistema; tutti i particolari vengono giustificati nel corso di questo capitolo.

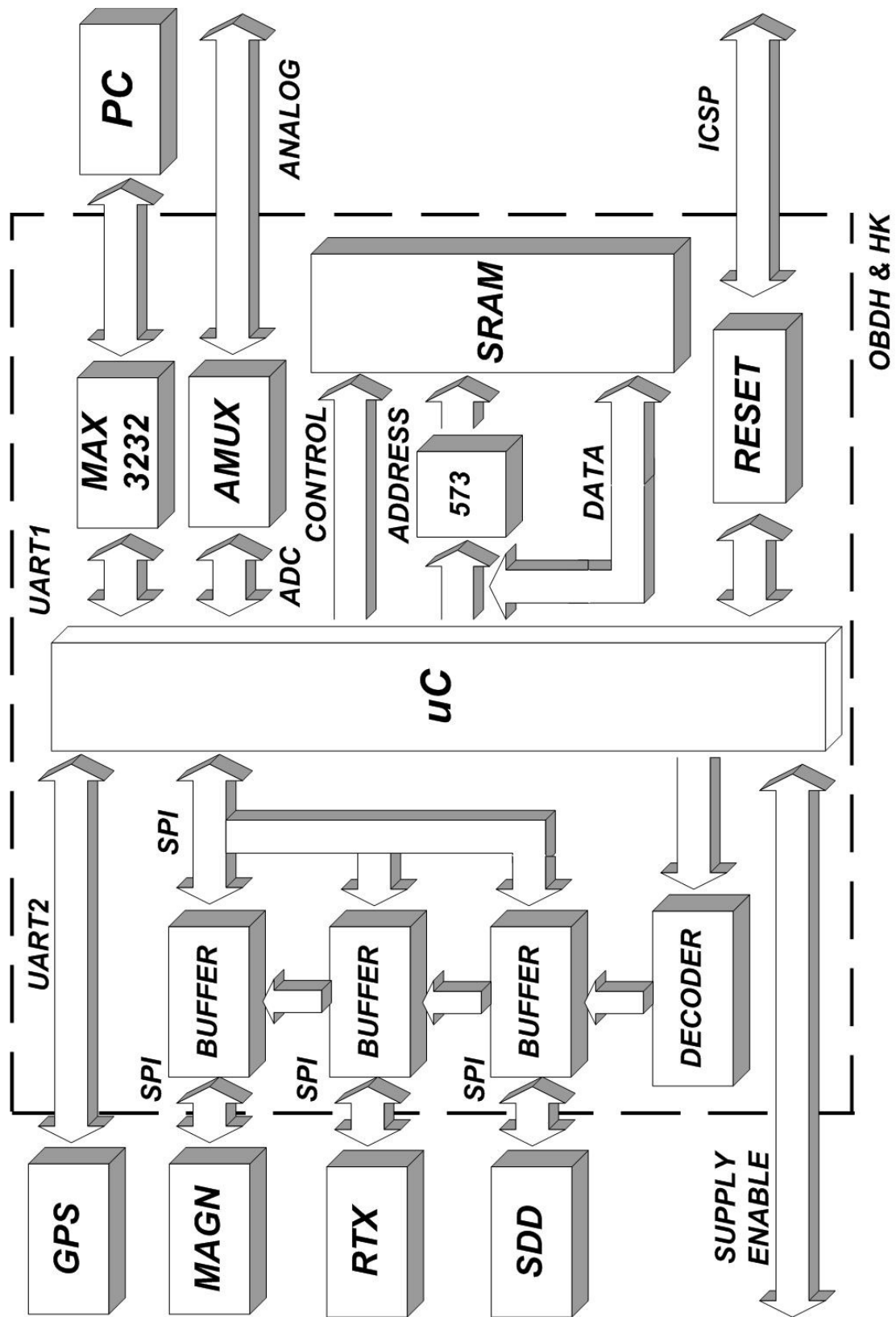


FIG. 4.1 – SCHEMA A BLOCCHI DEL MODULO OBDH & HK E RELATIVI INTERFACCIAMENTI

4.1 Bus parallelo per la comunicazione del μC con la memoria esterna

Si procede quindi all'analisi dell'interfacciamento della memoria SRAM Cypress CY62167DV30 di telemetria con il microcontrollore Microchip PIC18LF8722. Per fare ciò ci si serve della descrizione della memoria programma del microcontrollore, effettuata nel capitolo precedente, assieme alla descrizione della periferica EMI di seguito riportata. Come accennato la periferica EMI nasce con l'obiettivo di espandere la memoria programma, quindi in origine i dati contenuti in essa sono istruzioni di programma. Nel progetto in esame questo approccio viene ribaltato per sfruttare le caratteristiche di accesso tipiche di una CPU ma prelevando dati di immagazzinamento e variabili utili al contesto dell'applicazione.

L'interfacciamento tra microcontrollore e memoria SRAM avviene secondo questo schema generale:

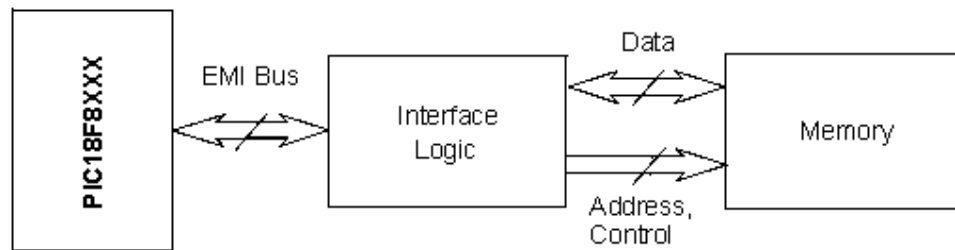


FIG. 4.2 - SCHEMA DI INTERFACCIAMENTO DI PRINCIPIO TRA PIC18LF8722 E SRAM

Per poter gestire adeguatamente la memoria esterna SRAM mantenendo intatta ed attiva la memoria programma interna di 128 kbytes, la modalità con la quale viene gestito il microcontrollore è l'Extended Microcontroller Mode EMC.

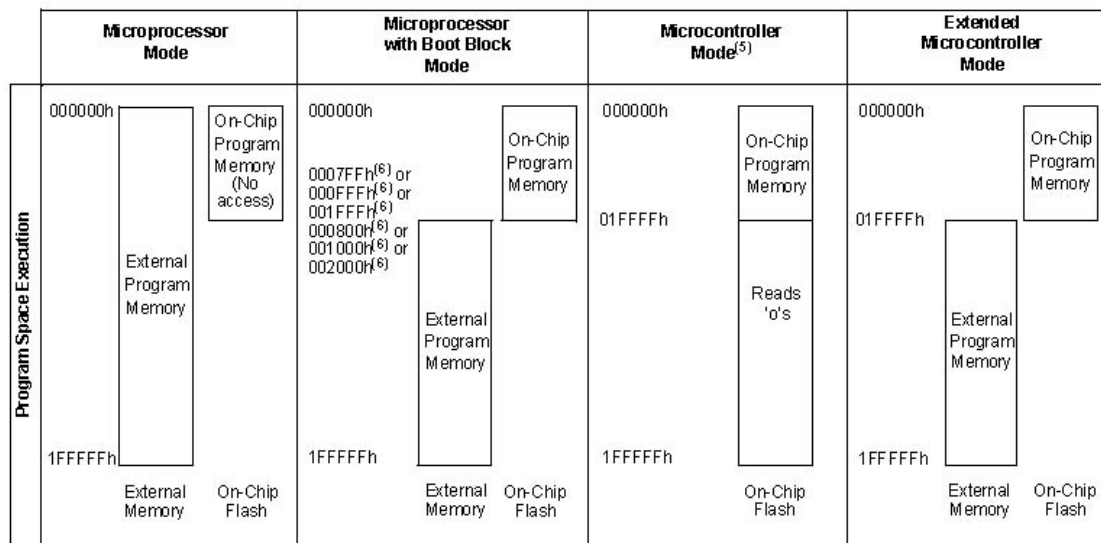


FIG. 4.3 - IL MICROCONTROLLORE VIENE GESTITO IN MODALITA' ESTESA

Si è visto che la memoria programma interna e la memoria SRAM esterna vengono interpretate dal microcontrollore come un unico blocco e l'indirizzamento è di tipo lineare. La natura di questa architettura implica il piccolo svantaggio che i primi 128kbytes della memoria SRAM esterna non sono accessibili per mezzo della periferica EMI in quanto gli indirizzi che fanno capo a questa prima porzione di memoria sono impiegati per accedere alla memoria programma interna.

Riassumendo, gli indirizzi compresi tra 0x000000 e 0x01FFFF sono dedicati alla memoria programma interna mentre i rimanenti indirizzi, cioè quelli compresa tra 0x020000 e 1FFFFFF, sono dedicati alla memoria SRAM esterna riducendo la capacità effettiva di quest'ultima a 1920 kbytes.

La periferica EMI dal punto di vista dell'hardware occupa quattro porte del microcontrollore, la D, la E, la H e la J per un totale di 28 pin. Questi pin sono utilizzati per le linee di indirizzo, di dati e di controllo e vengono condivisi con le funzioni e le periferiche comuni alle porte.

Name	Port	Bit	External Memory Bus Function
RD0/AD0	PORTD	0	Address bit 0 or Data bit 0
RD1/AD1	PORTD	1	Address bit 1 or Data bit 1
RD2/AD2	PORTD	2	Address bit 2 or Data bit 2
RD3/AD3	PORTD	3	Address bit 3 or Data bit 3
RD4/AD4	PORTD	4	Address bit 4 or Data bit 4
RD5/AD5	PORTD	5	Address bit 5 or Data bit 5
RD6/AD6	PORTD	6	Address bit 6 or Data bit 6
RD7/AD7	PORTD	7	Address bit 7 or Data bit 7
RE0/AD8	PORTE	0	Address bit 8 or Data bit 8
RE1/AD9	PORTE	1	Address bit 9 or Data bit 9
RE2/AD10	PORTE	2	Address bit 10 or Data bit 10
RE3/AD11	PORTE	3	Address bit 11 or Data bit 11
RE4/AD12	PORTE	4	Address bit 12 or Data bit 12
RE5/AD13	PORTE	5	Address bit 13 or Data bit 13
RE6/AD14	PORTE	6	Address bit 14 or Data bit 14
RE7/AD15	PORTE	7	Address bit 15 or Data bit 15
RH0/A16	PORTH	0	Address bit 16
RH1/A17	PORTH	1	Address bit 17
RH2/A18	PORTH	2	Address bit 18
RH3/A19	PORTH	3	Address bit 19
RJ0/ALE	PORTJ	0	Address Latch Enable (ALE) Control pin
RJ1/ $\overline{OE}$	PORTJ	1	Output Enable ($\overline{OE}$) Control pin
RJ2/ $\overline{WRL}$	PORTJ	2	Write Low ($\overline{WRL}$) Control pin
RJ3/ $\overline{WRH}$	PORTJ	3	Write High ($\overline{WRH}$) Control pin
RJ4/BA0	PORTJ	4	Byte Address bit 0 (BA0)
RJ5/ $\overline{CE}$	PORTJ	5	Chip Enable ($\overline{CE}$) Control pin
RJ6/ $\overline{LB}$	PORTJ	6	Lower Byte Enable ($\overline{LB}$) Control pin
RJ7/ $\overline{UB}$	PORTJ	7	Upper Byte Enable ($\overline{UB}$) Control pin

FIG. 4.4 - PIEDINATURA DELLA PERIFERICA EMI CONDIVISA CON LE ALTRE FUNZIONI DELLE PORTE

Per quanto riguarda il controllo del bus esterno sono presenti sette segnali: si tratta delle linee $\overline{OE}$, $\overline{WRH}$, $\overline{WRL}$, $\overline{CE}$, $\overline{UB}$, $\overline{LB}$ e ALE ; tutte le linee eccetto $\overline{OE}$ possono essere usate durante la scrittura, mentre tutte le linee eccetto $\overline{WRH}$ e $\overline{WRL}$ possono essere utilizzate durante la lettura. I controlli $\overline{UB}$ e $\overline{LB}$ invece selezionano il byte rispettivamente più e meno significativo.

E' doveroso chiarire fin da ora che per poter utilizzare le prime sedici linee di indirizzo anche come linee di dato, tra il microcontrollore e la memoria sono interposti dei latch che mantengono al momento opportuno il valore dell'indirizzo proprio su questa parte del bus. A tal riguardo l' ALE è un segnale che permette di gestire le linee condivise tra i dati e gli indirizzi agendo sul segnale di abilitazione dei latch. L'applicazione che viene usata in un determinato momento invoca opportunamente gli adeguati segnali di controllo.

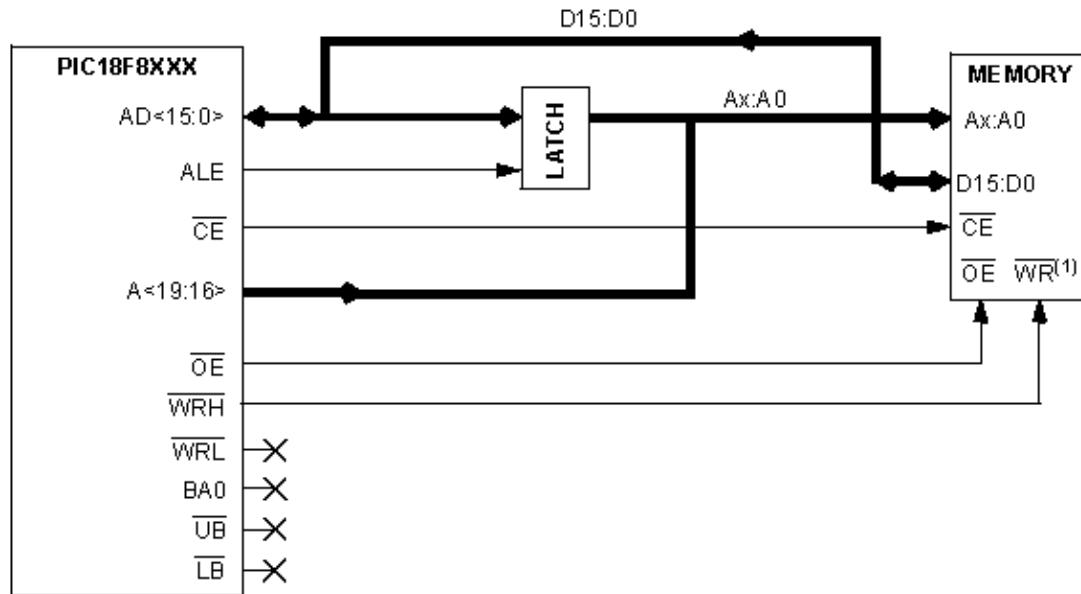


FIG. 4.6 - SCHEMA GENERALIZZATO DELLA CONNESSIONE TRA IL MICROCONTROLLORE E UNA MEMORIA ESTERNA

Lo schema a blocchi rappresenta una generica connessione tra il microcontrollore e una memoria per mezzo del bus a 16 bit, in modo da comprendere in linea di principio la differenziazione dei vari bus e l'organizzazione delle linee collegamento. Non ci si sofferma ora sui particolari segnali di controllo implementati; quello che si vuol far notare è la condivisione delle linee dati con le linee di indirizzo. Si vede infatti che dal microcontrollore si originano 16 linee comuni tra indirizzi e dati; le linee a monte del latch deviano dal bus e vengono dedicate alle informazioni del dato, mentre le linee a valle del latch diventano solamente un bus degli indirizzi. I 4 bit più significativi del bus indirizzi invece sono indipendenti e si portano direttamente ai pin della memoria.

In via di principio, il funzionamento di tale configurazione si basa sul fatto che una volta inviata l'opportuna locazione sulle 20 linee di indirizzo, tale informazione viene congelata a valle del latch per mezzo del segnale di abilitazione ALE, poi solo successivamente vengono modificati i segnali sulle linee AD<15:0> inviando l'opportuno dato sul bus denominato D<15:0>.

Gli aspetti dal punto di vista del software verranno opportunamente trattati nel capitolo dedicato; in ogni caso per la comprensione dei successivi diagrammi, si accenna al fatto che per accedere all'indirizzamento non si modifica direttamente il Program Counter ma bensì un registro di puntamento di tabella denominato TBLPTR suddiviso in 3 registri da 8 bit; per effettuare invece delle operazioni di lettura e scrittura si hanno a disposizione rispettivamente le istruzioni di TBLRD* e TBLWR*. Tutti i dati vengono scambiati in pacchetti di un singolo byte per mezzo del registro TABLAT.

Di seguito vengono analizzate le temporizzazioni di un ciclo di lettura mediante la periferica EMI. Al contrario di quanto avviene in scrittura, la modalità di lettura è unica indipendentemente dal tipo memoria utilizzata e dal modo in cui viene interfacciata al microcontrollore.

La famiglia PIC18 viene sincronizzata da un clock che è quattro volte più veloce rispetto al suo ciclo di istruzione. Quattro impulsi di clock sono un ciclo di istruzione e sono indicati come Q1, Q2, Q3 e Q4. Durante Q1, il segnale di ALE viene abilitato mentre la locazione di memoria viene posta sul bus AD<15:0>. Allo stesso tempo, i bit più significativi della locazione vengono posti sul bus A<19:16>. Successivamente, sul fronte di discesa del controllo ALE, l'indirizzo viene congelato dal latch. All'inizio di Q3, il segnale attivo basso $\overline{OE}$ abilita le uscite della memoria e viene generato il segnale BA0 corrispondente al bit meno significativo del Program Counter. Alla fine di Q4, l' $\overline{OE}$ ritorna a livello logico alto e la parola a 16 bit viene prelevata dalla memoria.

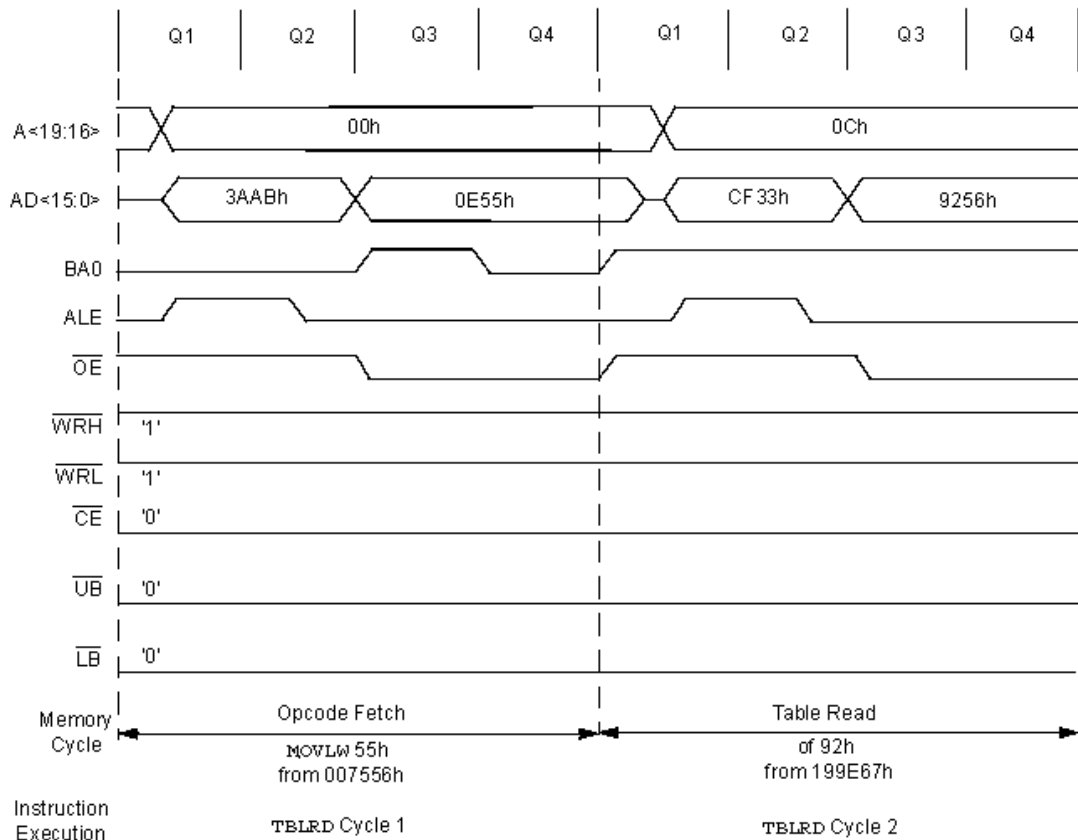


FIG. 4.7 - DIAGRAMMA DELLE TEMPORIZZAZIONI DI LETTURA NELLA PERIFERICA EMI

La prima lettura è un'operazione di fetch, mentre la seconda è un'operazione di read; questo è un diagramma generale, di conseguenza viene trattata anche un'operazione di fetch che in ogni caso è sempre una lettura.

Si analizzano ora i valori numerici che compaiono sui diagrammi e si considera come esempio l'operazione di Table Read sulla parte destra della figura precedente.

Tramite questa operazione si preleva il dato 0x92 dalla locazione di memoria 0x199E67.

Sui 4 bit più significativi dell'indirizzo compare il valore 0x0C mentre sui restanti 16 bit è presente il valore 0xCF33. Ad una prima grossolana occhiata questi valori sembrano non avere niente a che vedere con il valore della locazione 0x199E67, in verità il valore 0x0CCF33 presente sull'intero bus indirizzi è proprio il valore 0x199E67 senza il bit meno significativo shiftato a destra di una posizione. Matematicamente parlando, sull'indirizzo 0x199E67 è stata effettuata una divisione binaria per 2 in modo da ottenere un indirizzo a 20 bit e non 21, correlandosi adeguatamente al discorso dei bordi delle parole di memoria e al Program Counter incrementato di un fattore 2.

0x199E67	0	0	0	1	1	0	0	1		1	0	0	1	1	1	1	0		0	1	1	0	0	1	1	1
0x0CCF33	0	0	0	0	1	1	0	0		1	1	0	0	1	1	1	1		0	0	1	1	0	0	1	1

FIG. 4.8 - 0x199E67 E' IL VALORE DEL TBLPTR, MENTRE 0x0CCF33 E' IL VALORE PRESENTE SUI 20 BIT DELL'INDIRIZZO

Sul bus dei dati invece compare 0x9256 in quanto alla locazione di memoria 0x0CCF33 sono presenti 2 byte, uno è quello di interesse mentre l'altro no. Questa discriminazione viene effettuata dal circuito interno del registro TABLAT in base al bit meno significativo dell'indirizzo completo 0x199E67, cioè in base al bit meno significativo del TBLPTR o in maniera equivalente del Program Counter.

Il segnale BA0 cambia di stato a secondo di questo bit e nel diagramma in questione rimane sempre alto visto che l'indirizzo 0x199E67 è dispari cioè ha un valore pari a 1 nel bit meno significativo.

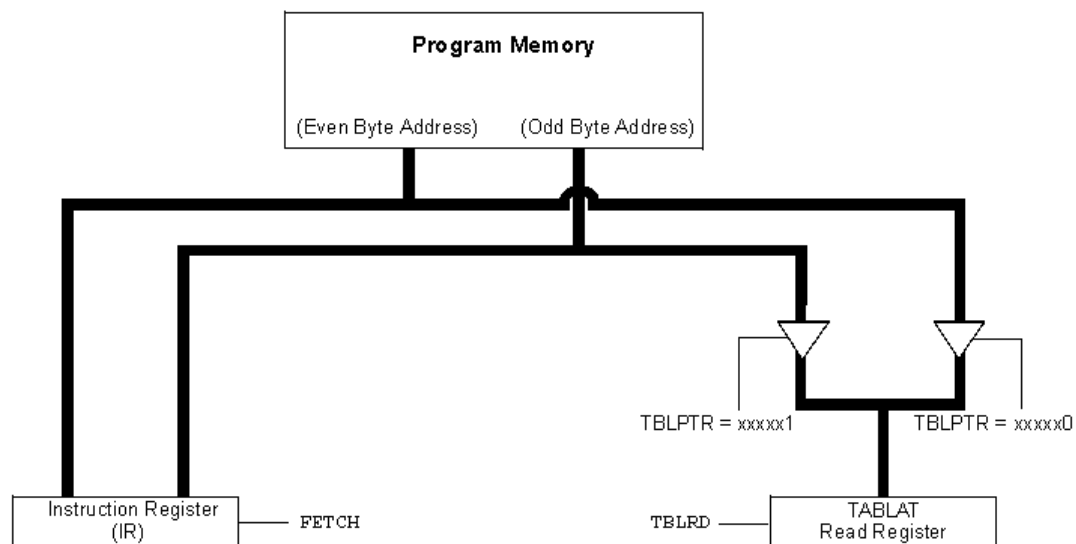


FIG. 4.9 - CIRCUITO DI DISCRIMINAZIONE IN BASE AL BIT MENO SIGNIFICATIVO DEL PROGRAM COUNTER

Dalla figura 4.9 si nota come il bus a 16 bit venga interpretato in due bus a 8 bit in base al bit meno significativo dell'indirizzo, cioè del Program Counter, e come l'indirizzo venga suddiviso nel tipo pari e nel tipo dispari.

Si passa ora alla descrizione della parte di scrittura della memoria.

Al contrario della lettura, le modalità di scrittura messe a disposizione dalla periferica EMI sono di tre tipologie; esse dipendono dal dispositivo di memoria impiegato e dal modo di connetterlo al microcontrollore. I tre tipi di configurazione del modulo EMI sono **il Word Write Mode, il Byte Select Mode e il Byte Write Mode**.

Si comincia con il primo, **il Word Write Mode**.

La figura mostra un esempio di interfacciamento nella modalità suddetta dove vengono usate memorie che impiegano parole di dati estese quanto la capacità del bus, quindi 16 bit. Questo metodo fa una differenziazione tra il ciclo che prevede un indirizzo pari e il ciclo che prevede un indirizzo dispari.

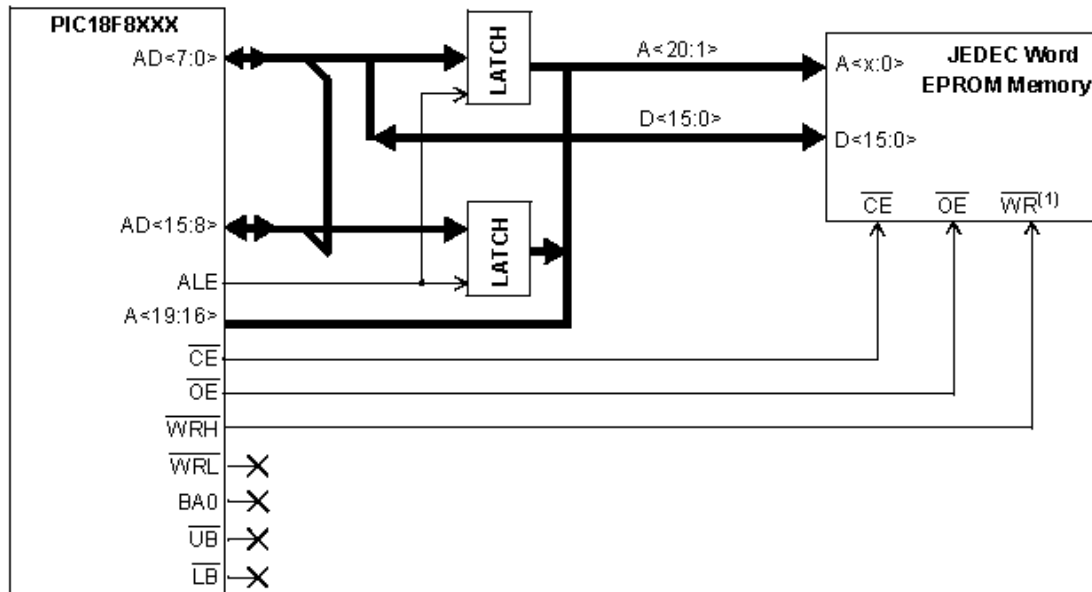


FIG. 4.10 - INTERFACCIAMENTO IN MODALITA' WORD WRITE MODE

Durante il ciclo di un indirizzo pari il dato viene salvato in un registro e il bus dei dati viene posto in una condizione di alta impedenza; ciò, associato al fatto che non viene abilitato alcun controllo, implica che momentaneamente non viene effettuata alcuna scrittura sulla memoria.

Durante il ciclo di un indirizzo dispari il dato viene posto subito sul bus AD<15:8> e allo stesso istante il dato precedentemente presente nel registro temporaneo viene posto sul bus AD<7:0>.

Il controllo $\overline{WRH}$ viene poi abilitato permettendo la scrittura contemporanea del dato a 16 bit nella memoria, mentre il controllo $\overline{WRL}$ non viene utilizzato.

La linea BA0 segue l'andamento del Program Counter ma non trova applicazione pertanto non viene implementato.

I segnali $\overline{UB}$ e $\overline{LB}$ invece vengono posti entrambi a livello logico basso per selezionare i due byte che compongono l'intero dato a 16 bit.

L'ovvia limitazione di questo metodo è che la scrittura deve avvenire per forza in coppia su uno specifico bordo della memoria.

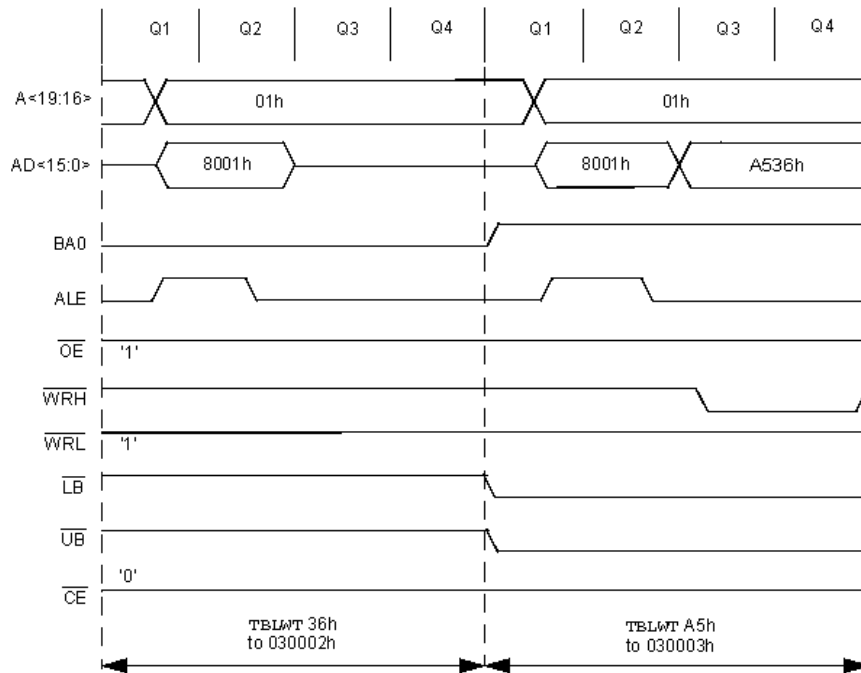


FIG. 4.11 - DIAGRAMMA DELLE TEMPORIZZAZIONI DI SCRITTURA NELLA MODALITA' WORD WRITE MODE

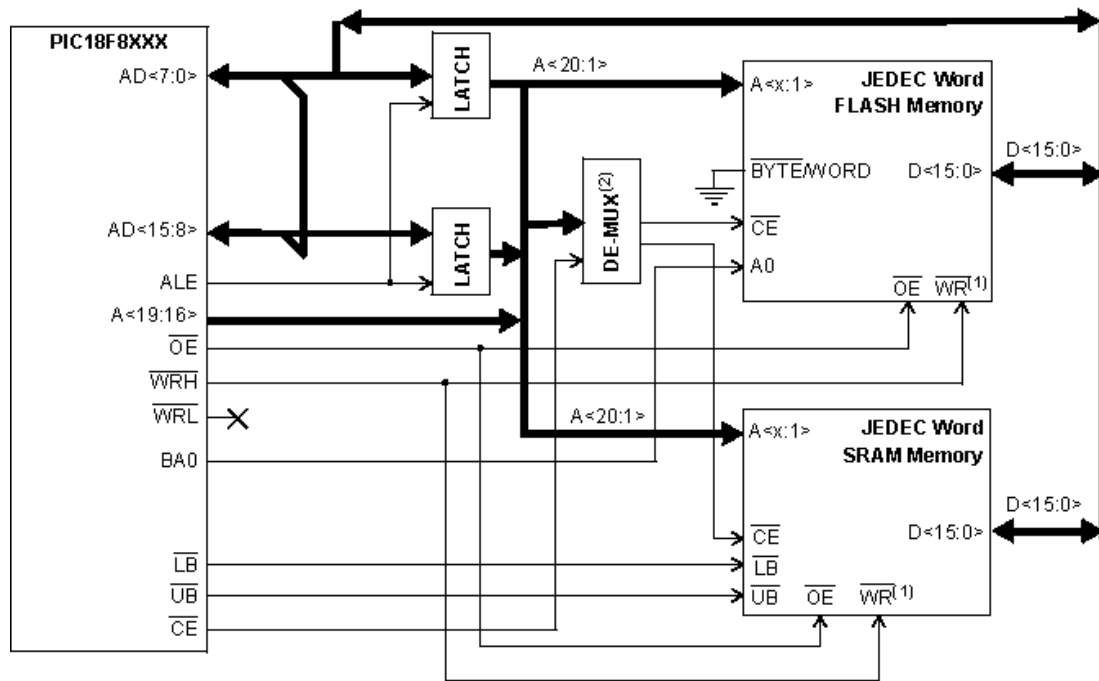


FIG. 4.12 - INTERFACCIAMENTO IN MODALITA' BYTE SELECT MODE

Il Byte Select Mode prevede la possibilità di effettuare accessi a memorie con parole di dati di larghezza pari alla capacità del bus, quindi sempre 16 bit, ma con la capacità di selezionare il singolo byte.

Durante un'operazione con questo metodo, il dato da scrivere viene copiato sia sulla parte più significativa del bus indirizzi, cioè su AD<15:8>, sia sulla parte meno significativa del bus indirizzi, cioè su AD<7:0>. Per ogni ciclo, il controllo $\overline{WRH}$ viene attivato assieme a

soltanto uno dei due segnali di selezione del byte, $\overline{UB}$ e $\overline{LB}$, e ciò viene fatto in base al bit meno significativo dell'indirizzo in esame.

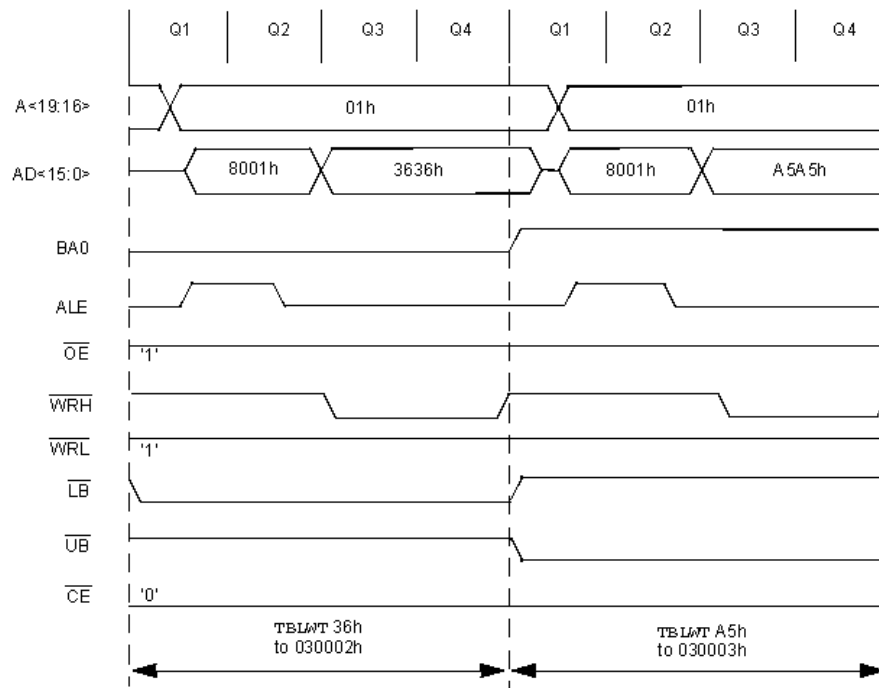


FIG. 4.13 - DIAGRAMMA DELLE TEMPORIZZAZIONI DI SCRITTURA NELLA MODALITA' BYTE SELECT MODE

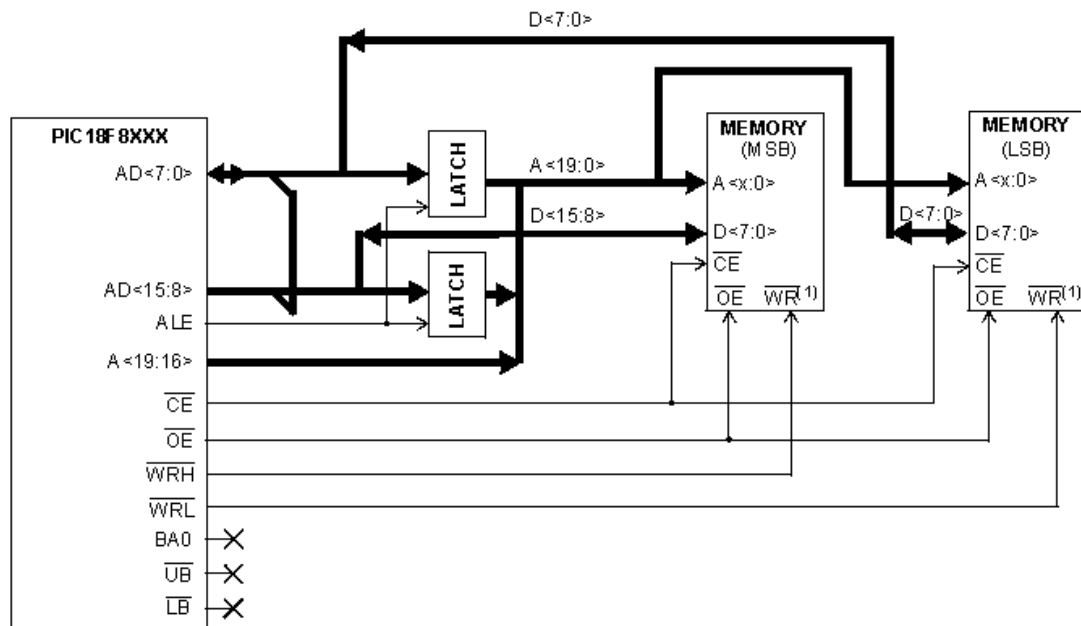


FIG. 4.14 - INTERFACCIAMENTO IN MODALITA' BYTE WRITE MODE

Il Byte Write Mode è l'ultima modalità disponibile nella periferica EMI e viene impiegata per interfacciare due memorie separate da 8 bit.

In questa configurazione, durante un ciclo il dato viene presentato come nella modalità precedente, copiato sia sul bus AD<15:8> che sul bus AD<7:0>. La discriminazione avviene per mezzo dei due controlli $\overline{WRH}$ e $\overline{WRL}$.

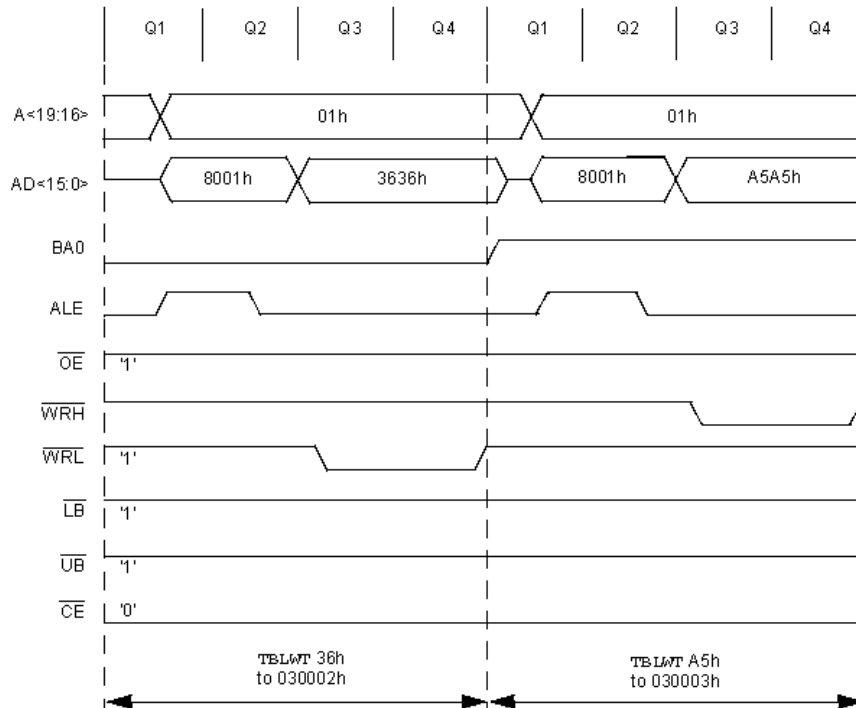


FIG. 4.15 - DIAGRAMMA DELLE TEMPORIZZAZIONI DI SCRITTURA NELLA MODALITA' BYTE WRITE MODE

Note ora le varie modalità e il loro funzionamento, si hanno a disposizione tutti gli strumenti per effettuare il progetto dell'interfacciamento tra la memoria Cypress CY62167DV30 e il microcontrollore PIC18LF8722.

L'ultima modalità vista, il Byte Write Mode, viene scartata come candidata della configurazione della periferica EMI in quanto è predisposta per l'implementazione di due memorie e vengono di conseguenza gestiti due segnali di scrittura, il $\overline{WRH}$ e $\overline{WRL}$.

Il primo metodo, il Word Write Mode può essere valido ma ci si vuole svincolare del tutto dal fatto che la scrittura possa avvenire solamente in coppie di byte e che ogni scrittura debba iniziare per forza su un indirizzo pari.

Tramite il Byte Select Mode invece è possibile effettuare la scrittura di un singolo byte per ogni ciclo. Per mezzo di questa configurazione quindi, si seleziona un certo indirizzo, pari o dispari, compreso nell'intervallo permesso e si va a scrivere un dato ampio un byte.

Questo tipo di accesso non risulta più lento del precedente, in quanto il Word Write Mode impiega comunque 2 cicli macchina per scrivere una word da 16 bit.

Viste le sue caratteristiche flessibili e generali, esso viene scelto per la gestione della memoria di telemetria di Atmocube.

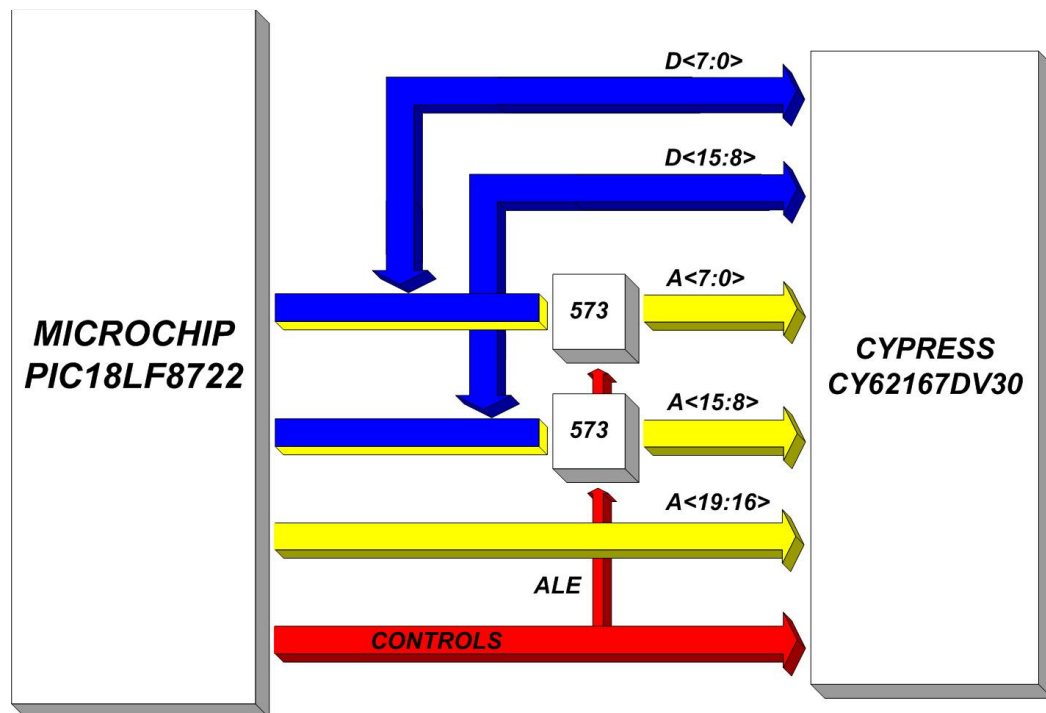


FIG. 4.16 - INTERFACCIAMENTO DELLA MEMORIA DI TELEMETRIA CON IL MICROCONTROLLORE DEL MODULO OBDH & HK

Dal punto di vista delle interconnessioni i segnali utilizzati per la connessione della memoria Cypress e il PIC Microchip sono: $AD<15:0>$ per i dati e i primi 16 bit dell'indirizzo, $A<19:16>$ per i restanti 4 bit dell'indirizzo e 6 segnali di controllo $\overline{OE}$, $\overline{WRH}$, $\overline{CE}$, $\overline{UB}$, $\overline{LB}$ e ALE ;

Si nota come la logica di connessione e controllo tra microcontrollore e memoria segue rigorosamente l'architettura hardware interna al microcontrollore per la gestione del bus di istruzioni. Si nota inoltre il dispendioso cablaggio dalla parte della memoria che viene abbondantemente ridotto dalla parte del microcontrollore grazie all'ausilio dei due latch.

La connessione rappresentata in figura è una valida soluzione per l'interfacciamento dei due dispositivi ma non è unica.

Il collegamento effettuato in questo progetto prevede infatti che le linee dati, i controlli e le 4 linee dei bit più significativi degli indirizzi connettano direttamente il microcontrollore e la memoria. Questa soluzione viene definita come parzialmente isolata, in quanto non sono presenti altri buffer di separazione. Generalmente, su un bus vengono interfacciati diversi dispositivi e di conseguenza possono insorgere problemi di fanout. Il fanout, è la misura del carico che può essere connesso all'uscita di un qualsiasi elemento logico della famiglia e identifica appunto il numero di ingressi che possono venir pilotati. Essendo presente in questa particolare applicazione solamente la memoria di telemetria, non insorgono problemi di questo tipo e pertanto non si rendono necessari ulteriori buffer.

Dato che le linee dati sono bidirezionali e dato che l'intero interfacciamento è arbitrato dalla periferica EMI, non insorgono problemi di conflitti hardware tra microcontrollore e memoria sul bus dati; in sostanza non si verifica mai l'errata condizione che sia le porte del microcontrollore che le porte della memoria siano poste entrambe come uscite. Resta il fatto che l'unica accortezza per evitare qualsiasi tipo di conflitto è stata presa riguardo ai primi istanti di avviamento del microcontrollore, cioè quando la periferica EMI non ha

ancora preso il controllo del bus della memoria. A tale scopo infatti, la linea di $\overline{CE}$ della memoria viene collegata all'alimentazione per mezzo di una resistenza di pull up, in modo da garantire lo stato di alta impedenza delle 16 linee dei dati fino ad avvio completo del sistema a microcontrollore.

L'impiego di una memoria parallela con l'ausilio di registri latch non è nuova, ma ha origine già con l'avvento dei processori Intel dotati di architettura x86. Di seguito si riportano delle configurazioni di interfacciamento di un processore 8086 con una memoria parallela ricavati direttamente dai manuali Intel. Si prende come esempio questo processore perché possiede un'architettura a 16 bit come il microcontrollore in dotazione. La figura 4.17 mostra un bus, gestito dal microcontrollore 8086, molto simile a quello presente nel progetto del modulo OBDH & HK. Si notano infatti due latch che permettono la condivisione tra le linee di indirizzo e le linee dei dati.

I latch vengono abilitati allo stesso modo, cioè servendosi di una linea di controllo di attivazione ALE.

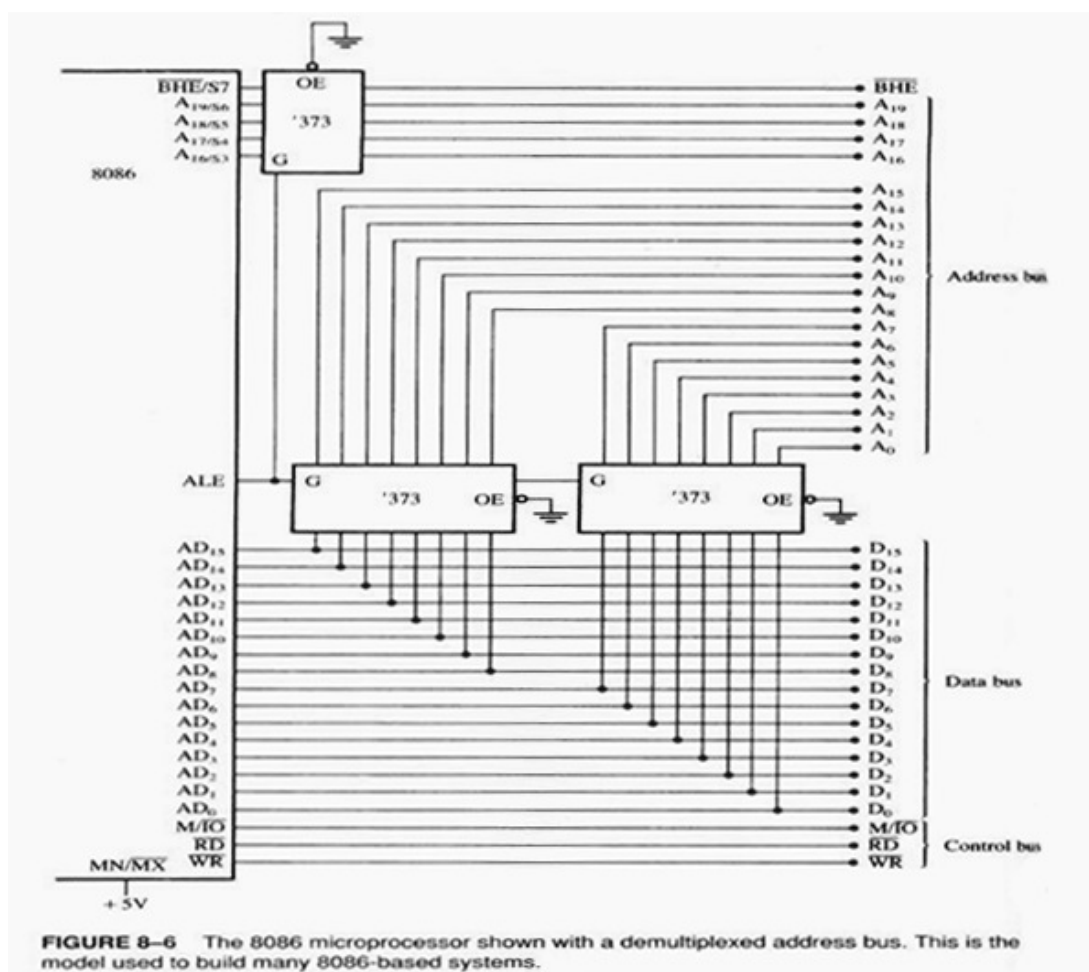


FIG. 4.17 - INTERFACCIAMENTO TRA MICROPROCESSORE INTEL 8086 E MEMORIA

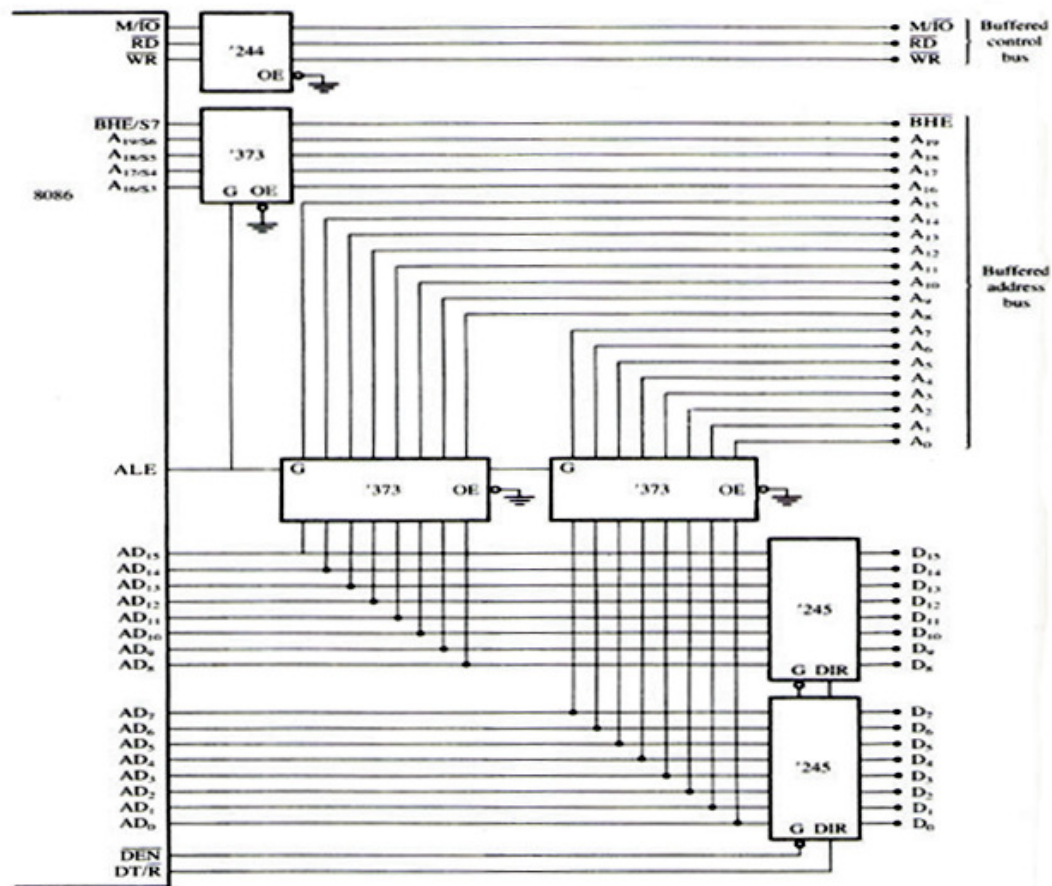


FIGURE 8-8 A fully buffered 8086 microprocessor

FIG. 4.18 - BUS DEL MICROPROCESSORE INTEL 8086 IN CONFIGURAZIONE COMPLETAMENTE ISOLATA

La figura 4.18 mostra l'implementazione di un bus completamente isolato. La configurazione di base rimane inalterata in quanto il principio con il quale avviene la condivisione tra linee dei dati e degli indirizzi rimane lo stesso. La differenza rispetto alla soluzione precedente è che ora sono presenti dei buffer bidirezionali anche sulle 16 linee dei dati. Per controllare la direzione dei dati è stato aggiunto un ulteriore controllo denominato nella figura come $DT/\bar{R}$. Anche i restanti controlli non fanno capo direttamente alla memoria ma vengono isolati interponendo dei buffer.

L'architettura per la gestione della memoria impiegata nel sistema Embedded di questo elaborato è la parte hardware che maggiormente avvicina il modulo OBDH & HK alla struttura di un microcomputer e gli schemi elettrici riportati dei manuali Intel ne danno piena testimonianza.

Ulteriori considerazioni vanno fatte per ciò che riguarda i tipi di latch scelti.

La parte che tratta l'alimentazione del modulo OBDH & HK viene esaminata nel capitolo 5 di questo documento; ciò che è indispensabile focalizzare ora è che l'intero sistema e tutti i protocolli di comunicazione funzionano con una tensione di 3.3 V. Questo è un dato importante in quanto condiziona i valori dei livelli logici e il tipo di famiglia logica dei latch.

La scelta cade fin da subito su componenti di tipo CMOS in modo da garantire consumi ridotti. La famiglia HC, cioè CMOS ad alta velocità, consente inoltre una buona flessibilità sul range di alimentazione, permettendo la gestione di segnali a 3.3 V.

L'insieme di queste caratteristiche portano a scegliere l'integrato SN74HC573 come dispositivo latch per l'interfacciamento della memoria al microcontrollore. La logica di controllo è compatibile con il bus EMI, in particolare il segnale di LE è adeguato per essere pilotato dal segnale ALE del microcontrollore.

A questo punto le ultime considerazioni dal punto di vista dell'hardware riguardano la compatibilità di tutti i livelli logici dei segnali tra il microcontrollore, la memoria e i latch stessi. Si fa notare che le linee della periferica EMI del microcontrollore vengono gestite da porte di tipo TTL. Per tale motivo si riportano in figura 4.19 tutte le tensioni dei livelli logici di questi tre componenti.

Le condizioni da verificare sono: $V_{OH}(\min) > V_{IH}(\min)$ e $V_{OL}(\max) < V_{IL}(\max)$.

	$V_{IL}(\max)$	$V_{IH}(\min)$	$V_{OL}(\max)$	$V_{OH}(\min)$
PIC18LF8722 (TTL)	$0.15V_{dd}=0.495V$	$0.25V_{dd}+0.8=1.625V$	$0.6V$	$V_{dd}-0.7=2.6V$
CYPRESS CY62167DV30 (CMOS)	$0.8V$	$2.2V$	$0.4V$	$2.4V$
SN74HC573 (CMOS)	$0.825V$	$2.475V$	$0.1V$	$3.2V$

FIG. 4.19 – LIVELLI LOGICI

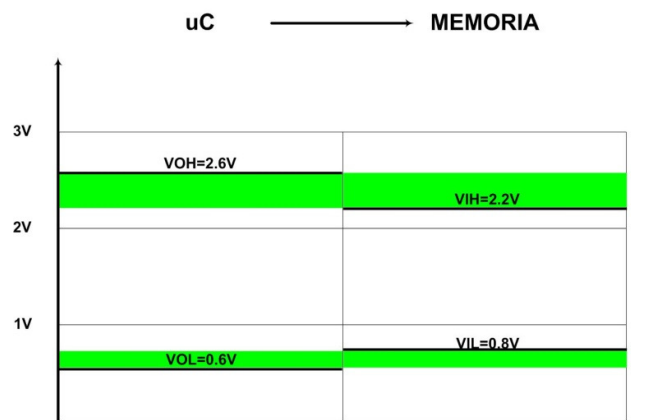


FIG. 4.20 - COMPATIBILITA' TRA LE USCITE DEL μC E GLI INGRESSI DELLA MEMORIA

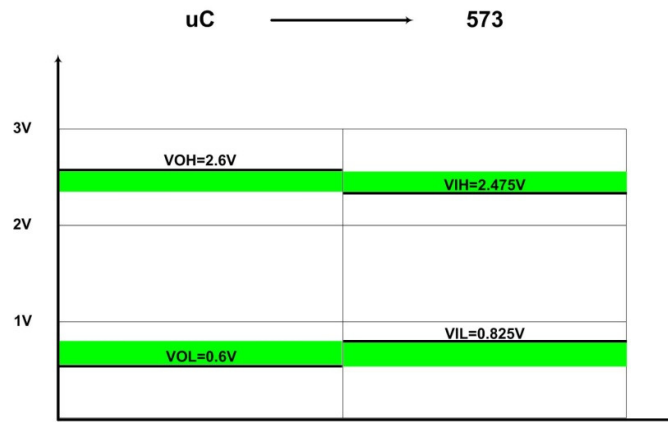


FIG. 4.21 - COMPATIBILITA' TRA LE USCITE DEL μC E GLI INGRESSI DEL LATCH

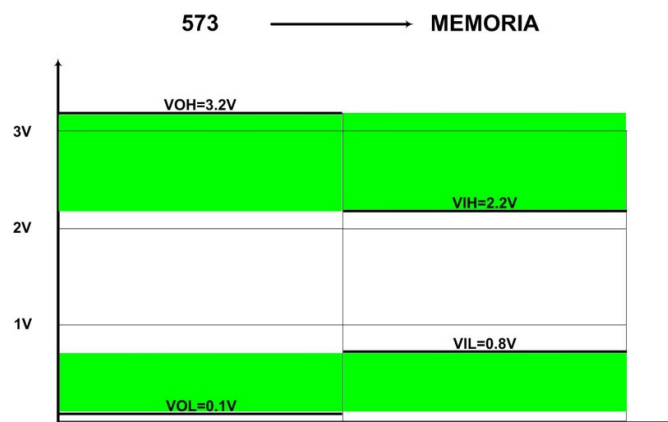


FIG. 4.22 - COMPATIBILITA' TRA LE USCITE DEL LATCH E GLI INGRESSI DELLA MEMORIA

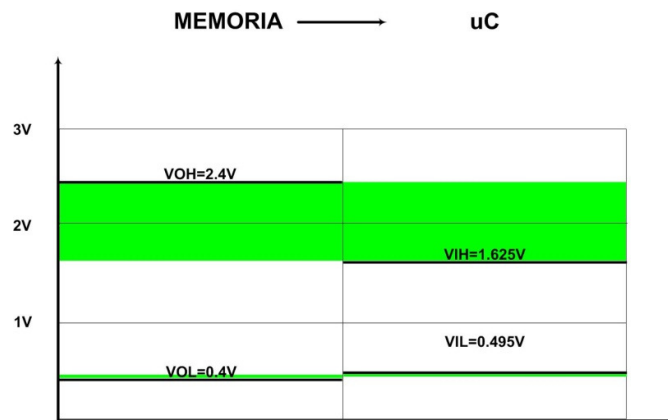


FIG. 4.23 - COMPATIBILITA' TRA LE USCITE DELLA MEMORIA E GLI INGRESSI DEL μC

I quattro diagrammi mostrano le possibili combinazioni dell'interfacciamento tra i dispositivi microcontrollore, memoria e latch.

Per tutti i casi, la compatibilità è garantita pienamente visto il fatto che sono stati considerati i valori dei livelli logici nelle condizioni peggiori, sia per le tensioni sia per il rumore. Le fasce colorate mettono in evidenza il margine di sicurezza che è presente nei vari casi e rappresentano la bontà dell'interfacciamento.

4.2 Bus SPI per la comunicazione del μ C con gli strumenti e il modulo OBR

Nel satellite Atmocube sono presenti tre moduli che impiegano lo standard seriale SPI per la comunicazione: il magnetometro, lo spettro - dosimetro SDD e il ricetrasmittitore OBR. Generalmente, come tutti i bus, ci sono a disposizione delle linee di comunicazione in comune con alcune periferiche che devono accedere all'unità centrale. La conseguenza principale è che le linee dei vari dispositivi sono fisicamente connesse e ciò solitamente non crea alcun tipo di problema. Nel sistema in esame però, è presente un piano di Power Management che prevede lo spegnimento dei vari moduli del satellite per ottimizzare l'autonomia della batteria al litio; tra questi moduli fanno parte sicuramente gli strumenti di bordo, quali il magnetometro, lo spettro dosimetro e lo stesso ricetrasmittitore.

Dato che la comunicazione con la stazione di terra avviene in una certa finestra temporale e solo in certe condizioni di visibilità, può accadere benissimo che gli strumenti di bordo siano accesi ed effettuino le misure sperimentali mentre non avvenga alcuna trasmissione e ricezione, con la conseguenza che il modulo OBR sia spento. In tal modo contemporaneamente sul bus SPI possono trovarsi delle periferiche alimentate ed altre no; questa particolare condizione implica che i dispositivi SPI privi di alimentazione continuino a ricevere segnali sulle linee di comunicazione creando di fatto delle condizioni non sicure e soprattutto non prevedibili; tra questi fanno parte anche danneggiamenti irreversibili dei componenti a causa di possibili correnti parassite e non previste createsi all'interno del dispositivo stesso. È buona norma quindi, dal punto di vista progettuale, evitare del tutto questa situazione.

La prima idea a scopo risolutivo per questo evidente problema è stata quella di creare tre bus SPI separati utilizzando tre linee di comunicazione dedicate per ogni modulo. In tal maniera infatti, quando viene tolta l'alimentazione ad un modulo, configurando via software tutte le linee SPI come ingressi del microcontrollore, è possibile creare una condizione di alta impedenza isolando completamente la periferica non alimentata.

Questa soluzione è valida, resta il fatto che risulta più onerosa dal punto di vista del cablaggio, dovendo infatti liberare un numero di piedini dalle porte del microcontrollore tre volte superiore alla configurazione classica del bus. Si rende obbligatoria quindi la scelta di condividere per tutti i sottosistemi di Atmocube le linee di comunicazione SPI; si raggiunge pertanto il problema e si creano sempre tre bus separati, ma facenti capo ad un'unica porta, quindi di fatto utilizzando sempre sole tre linee del microcontrollore. Il tutto si rende possibile grazie all'utilizzo di buffer di isolamento sulle linee SPI posti su ciascun modulo. Impiegando infatti dei buffer three state, per mezzo di un'abilitazione, è possibile creare una condizione di alta impedenza isolando le periferiche a valle del bus e allo stesso tempo è possibile impiegare un numero minimo di piedini del microcontrollore.

L'esigenza di impiegare dei buffer di isolamento è comunque duplice; infatti il secondo importante compito dell'isolamento di tutti i sottomoduli riguarda il fatto di prevenire l'occupazione permanente del bus da parte di qualche strumento accidentalmente guasto. Essendo il ricetrasmittitore di vitale importanza per l'intero satellite, si preferisce essere cauti utilizzando questo tipo di precauzione e minimizzare il possibile danno globale.

Per poter effettuare uno studio accurato di come implementare questo bus viene riportato sinteticamente l'elenco e una breve descrizione delle linee presenti per l'interfacciamento di ciascuna periferica SPI.

- **SDD:** la comunicazione con questo modulo avviene sia in trasmissione che in ricezione, di conseguenza oltre ovviamente la linea del clock SCLK, sono presenti sia la linea MOSI che la linea MISO. Dato che l'SDD gestisce la sua parte digitale per mezzo di una FPGA e visto che questo è l'unico componente che gestisce la comunicazione SPI, si procede con l'implementazione di un'unica linea di CE.
- **OBR:** anche per questo sottosistema, l'unico componente da gestire tramite il bus SPI è il CC1100; la comunicazione avviene in entrambe le direzioni. Si rendono disponibili quindi, in maniera uguale all'SDD, un totale di quattro linee per lo scambio dati del ricetrasmittitore: il SCLK, il MOSI, il MISO e un CE.
- **MAGNETOMETRO:** per questo sottosistema, l'interfacciamento è un po' più complicato e un po' diverso vista la presenza di 3 convertitori analogico - digitali. Dato che ci sono più dispositivi SPI su un unico modulo, si rende necessario l'utilizzo di più comandi di CE, uno per ciascun convertitore. In più, visto che da questo modulo si effettuano solo operazioni di lettura, cioè si vanno solo a leggere i valori digitali delle rispettive conversioni, non è necessaria la presenza della linea MOSI. L'unico segnale di comunicazione, oltre il SCLK, che il microcontrollore invia ai convertitori del magnetometro riguarda l'inizio della conversione. Questo controllo è unico ed è comune ai tre componenti. Riassumendo, per quanto riguarda strettamente la comunicazione SPI, le linee da rendere disponibili per il magnetometro sono: il SCLK, il MISO e tre CE .

L'interfacciamento sia per lo strumento SDD sia per il ricetrasmittitore è identica in quanto le linee necessarie sono uguali. Pertanto per entrambi i sottosistemi, si procede all'utilizzo di tre buffer di isolamento: uno per il SCLK, uno per il MOSI e uno per il MISO. Il comando di abilitazione dei buffer, cioè il comando per attivare la modalità di alta impedenza, viene reso unico e comune a tutti e tre i buffer. Risulta comodo e utile, inoltre accorpare il comando di abilitazione dei buffer con il segnale di abilitazione della periferica SPI, cioè il CE. Quindi il microcontrollore gestisce un'unica linea che ingloba sia l'abilitazione dello stato attivo dei buffer che lo stato attivo della periferica SPI. Per poter sfruttare questa semplificazione, ovviamente la polarità dell'abilitazione dei buffer deve essere la stessa di quella del CE del componente SPI.

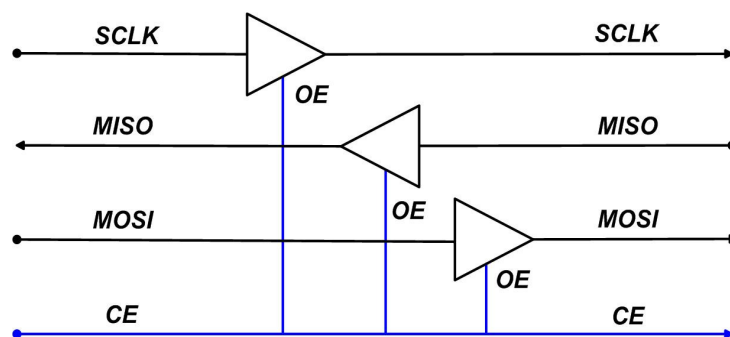


FIG. 4.24 - INTERFACCIAMENTO SPI DELL'SDD E DEL RICETRASMETTITORE

Per quanto riguarda il magnetometro, è necessario utilizzare dei blocchi come quelli appena riportati ma questa volta per ogni convertitore. Visto il tipo di applicazione, per ciascun dispositivo sono sufficienti solo due buffer, dato che la linea MOSI non è presente. Anche se le linee SCLK e MISO sono comuni, si implementano tre coppie di buffer vista comunque l'esigenza di controllare ciascun convertitore separatamente per mezzo di un segnale di CE indipendente. Potrebbe essere utilizzata una sola coppia di buffer con l'ausilio di una porta OR a tre ingressi, ma questa ipotesi viene scartata per non dover utilizzare altri circuiti integrati aggiuntivi.

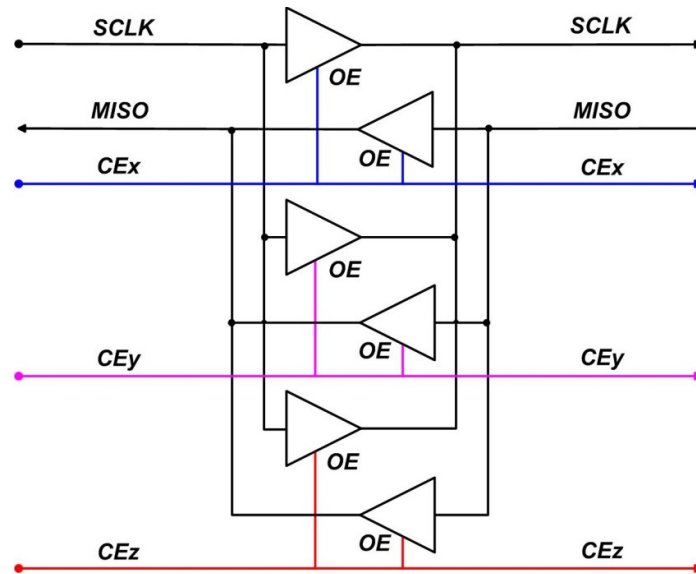


FIG. 4.25 - INTERFACCIAMENTO SPI DEL MAGNETOMETRO

I due circuiti progettati permettono quindi un interfacciamento completo delle varie periferiche con in aggiunta la caratteristica di isolare i dispositivi non utilizzati in quel momento nella comunicazione. In sintesi il funzionamento si basa sul seguente metodo: quando il modulo OBDH & HK deve iniziare uno scambio di dati con uno dei sistemi connessi al bus, viene per prima cosa attivata la linea del microcontrollore che fa capo sia all'abilitazione di tutti i buffer dedicati a quel particolare modulo sia al chip enable del dispositivo, poi successivamente si inizia a tutti gli effetti la comunicazione SPI. Procedendo in tal maniera infatti, si portano i buffer fuori dalla condizione di alta impedenza e si ottiene un collegamento tra il microcontrollore e la periferica abilitata. Si considera come stato iniziale che tutti i moduli si trovino in una condizione di isolamento in modo da non avere conflitti.

Dall'analisi circuitale, per l'abilitazione dei singoli dispositivi si notano un totale di cinque controlli: uno per l'SDD, uno per il ricetrasmittitore OBR e tre per il magnetometro. Dal momento che la comunicazione avviene con una periferica per volta, si rende utile al fine di risparmiare più cablaggio possibile, aggiungere all'architettura del bus un decoder dedicato al controllo delle linee di CE. Utilizzando infatti questo tipo di circuito integrato si possono risparmiare fino a 2 linee. Il guadagno non è estremamente vantaggioso, ma il decoder viene in ogni caso utilizzato dal momento che tutte le porte del microcontrollore sono quasi del tutto impegnate, ed ogni risparmio sotto questo punto di vista è ben accetto.

Con l'impiego del decoder, l'inizio di una comunicazione non avviene selezionando direttamente il rispettivo CE, ma selezionando il corrispondente indirizzo sul decoder. La

disabilitazione di tutti i moduli è comunque possibile e avviene indirizzando il decoder su un'uscita non implementata. L'orientamento sulla scelta dei buffer si rivolge fin da subito verso gates di tipo CMOS con consumi ridotti e tensioni compatibili con l'alimentazione del protocollo SPI. Si riserva particolare attenzione verso il numero di porte disponibili all'interno di ogni package, alla polarità del controllo di Output Enable OE per lo stato di alta impedenza e alla disponibilità di quest'ultimo per ogni singola porta. L'attenzione cade nuovamente su porte logiche della famiglia HC e il componente che soddisfa i precedenti requisiti è il 74HC126; l'intervallo di alimentazione è compreso tra 2 e 6 V ed è pienamente compatibile con l'alimentazione del bus SPI di 3.3 V; la polarità delle abilitazioni è positiva ed è perfettamente conforme con i CE dei dispositivi SPI. Per soddisfare tutte le esigenze dal punto di vista del cablaggio sono necessari 3 di questi circuiti integrati. Tra i vari componenti analizzati come candidati per l'isolamento del bus sono stati considerati inizialmente anche dei dispositivi nati proprio per svolgere in maniera dedicata questa funzione, gli IL600 della NVE Corporation. Questi componenti, permettono un isolamento più adeguato, in quanto separano galvanicamente ingressi e uscita per mezzo di un'induttanza. Ci sono però due inconvenienti che hanno portato a scartare questi circuiti integrati: uno è che ciascun elemento assorbe correnti non trascurabili fino a 6 mA, il secondo invece riguarda il fatto che, essendo impiegate delle induttanze, il componente è sensibile ai campi magnetici, modificando la risposta ai segnali. Considerando la presenza di magneti permanenti all'interno della struttura si evitano simili componenti privilegiando delle semplici porte CMOS come il 74HC126. La scelta del tipo di decoder cade ancora una volta verso la famiglia logica CMOS HC, fondamentalmente per le stesse motivazioni viste per i buffer. A causa delle polarità positive di tutti i segnali e per necessità di dover gestire almeno cinque linee di CE, viene scelto il componente 74HC238. Complessivamente, unificando tutti i circuiti visti, l'architettura globale del buffer SPI può essere rappresentata e riassunta mediante questo schema a blocchi.

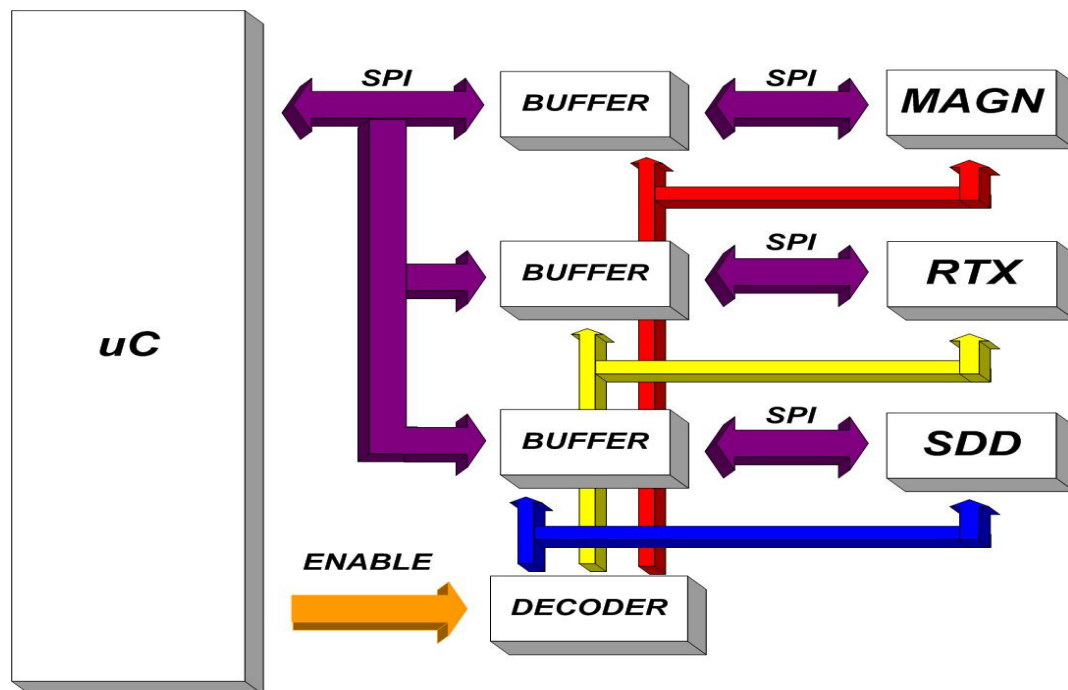


FIG. 4.26 - ARCHITETTURA GLOBALE DEL BUS SPI DEL SATELLITE ATMOCUBE

Anche per questa architettura vengono valutate adeguatamente le compatibilità dei livelli logici tra tutti i dispositivi. In questo caso le linee della periferica MSSP del bus SPI sono controllate da porte a Trigger di Schmitt con livelli logici di tipo CMOS.

Anche in questo caso le condizioni da verificare sono: $V_{OH}(\min) > V_{IH}(\min)$ e $V_{OL}(\max) < V_{IL}(\max)$.

	$V_{IL}(\text{MAX})$	$V_{IH}(\min)$	$V_{OL}(\text{MAX})$	$V_{OH}(\min)$
PIC18LF8722 (CMOS)	$0.2V_{dd}=0.66V$	$0.8V_{dd}=2.64V$	$0.6V$	$V_{dd}-0.7=2.6V$
CD74HC126 / 238 (CMOS)	$0.825V$	$2.475V$	$0.1V$	$3.2V$
CC1100 (CMOS)	$0.7V$	$2.6V$	$0.5V$	$3.0V$
AD7685(CMOS)	$0.99V$	$2.31V$	$0.4V$	$3.0V$
FPGA A3P250 (LVTTL)	$0.8V$	$2.0V$	$0.4V$	$2.4V$

FIG. 4.27 – LIVELLI LOGICI

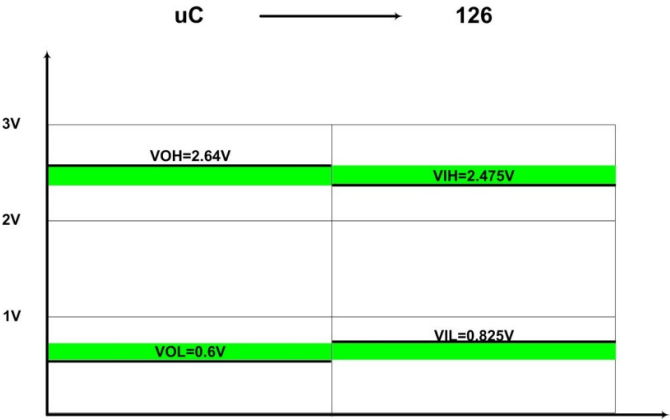


FIG. 4.28 - COMPATIBILITA' TRA LE USCITE DEL μC E GLI INGRESSI DEL BUFFER

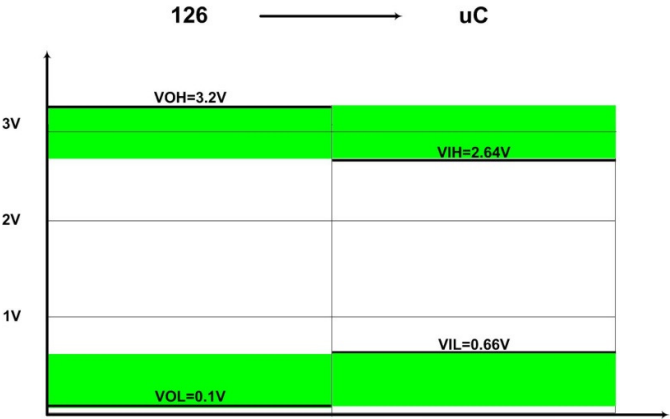


FIG. 4.29 - COMPATIBILITA' TRA LE USCITE DEL BUFFER E GLI INGRESSI DEL μC

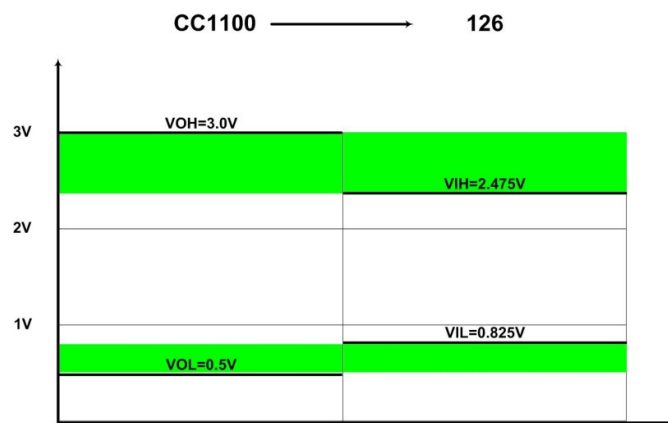


FIG. 4.30 - COMPATIBILITA' TRA LE USCITE DEL CC1100 DEL MODULO OBR E GLI INGRESSI DEL BUFFER

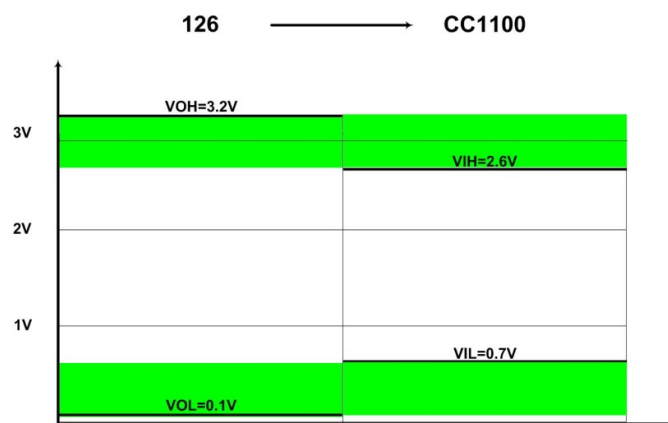


FIG. 4.31 - COMPATIBILITA' TRA LE USCITE DEL BUFFER E GLI INGRESSI DEL CC1100 DEL MODULO OBR

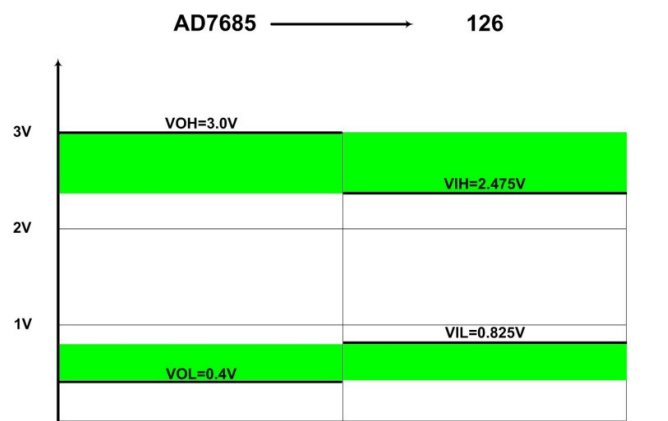


FIG. 4.32 - COMPATIBILITA' TRA LE USCITE DEI DAC DEL MAGNETOMETRO E GLI INGRESSI DEL BUFFER

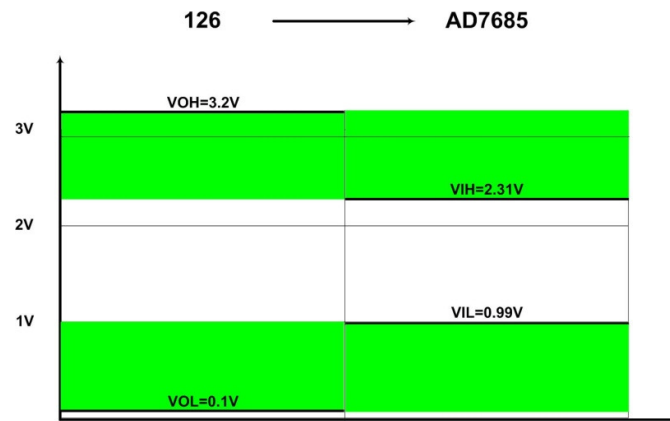


FIG. 4.33 - COMPATIBILITA' TRA LE USCITE DEL BUFFER E GLI INGRESSI DEI DAC DEL MAGNETOMETRO

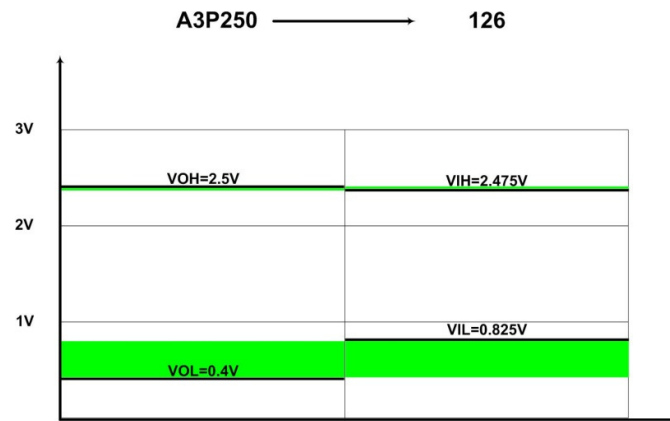


FIG. 4.34 - COMPATIBILITA' TRA LE USCITE DELLA FPGA E GLI INGRESSI DEL BUFFER

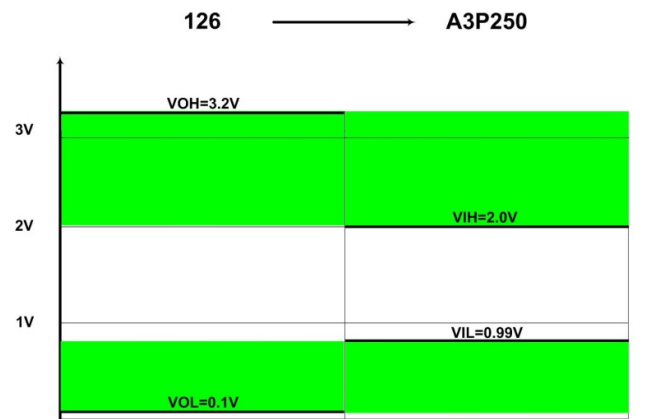


FIG. 4.35 - COMPATIBILITA' TRA LE USCITE DEL BUFFER E GLI INGRESSI DELLA FPGA

4.3 Comunicazione seriale asincrona del μC con il GPS ed il PC

Dopo aver visto il bus parallelo e il bus SPI, si passa ora alla descrizione dell'ultimo tipo di comunicazione presente sulla scheda OBDH & HK. Si tratta di uno scambio dati seriale di tipo asincrono implementato sia sul GPS di bordo sia su una porta RS-232. Questo tipo di porta non è un requisito fondamentale del modulo OBDH & HK, infatti non compare tra le specifiche del capitolo 2. In ogni caso viene aggiunta in quanto è di buona utilità come sistema di controllo del sistema; ciò avviene specialmente in fase in debug per mezzo di un Personal Computer.

Per quanto riguarda il GPS invece, esso utilizza per sua natura questo tipo di protocollo, di conseguenza il suo utilizzo non è una scelta opinabile ma una scelta obbligata. Il GPS e la porta RS-232 non condividono lo stesso cablaggio ma sono indipendenti come le periferiche messe a disposizione dal microcontrollore per questo tipo di comunicazione. Per tale motivo non si parla, come nei casi precedenti di bus, ma di semplice comunicazione punto a punto rispettivamente tra microcontrollore e GPS e tra microcontrollore e PC.

La separazione dal punto di vista dell'hardware viene effettuata sia per una questione di semplicità e ordine dell'architettura, ma anche per il fatto che dai datasheet del GPS si consiglia di utilizzare per esso uno standard di comunicazione di tipo RS-422.

Le comunicazioni di tipo seriale asincrono è permessa nei microcontrollori Microchip grazie alla presenza di due periferiche EUSART; con il loro ausilio di fatto è possibile instaurare le comunicazioni necessarie alle applicazioni della scheda OBDH & HK. Dato che i segnali di tensione del PIC18LF8722 sono compresi tra 0 e 3.3 V e invece gli standard RS-232 e RS-422 utilizzano tensioni più elevate e anche negative, sono necessari degli opportuni traslatori di livello, ovvero dei driver sia in trasmissione che in ricezione.

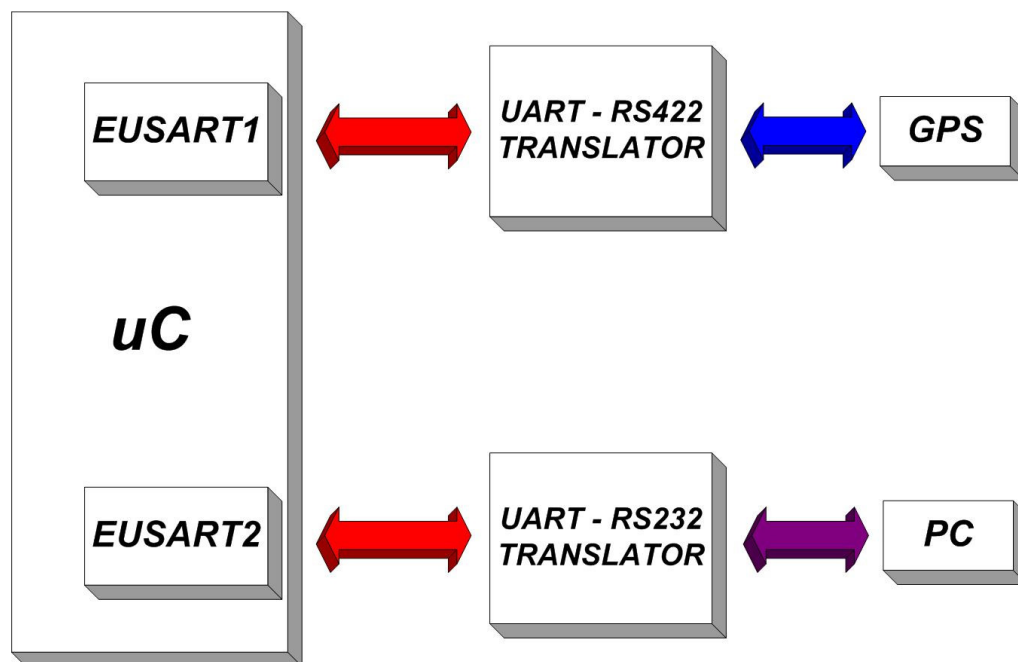


FIG. 4.36 - SCHEMA A BLOCCHI DEL CABLAGGIO DEL GPS E DELLA PORTA RS-232 DEL PC

4.3.1 Gestione del segnale RS-232

Un classico segnale RS-232 presenta la caratteristica di associare ad uno 0 logico una tensione tipica intorno ai +12 V (*min* +3 V, *MAX* +15 V), mentre ad un 1 logico una tensione tipica intorno ai -12 V (*min* -15 V, *MAX* -3 V). Il traslatore di livello quindi oltre a variare le tensioni rispetto a quelle del microcontrollore, deve invertire la logica. I componenti che svolgono questo compito sono di uso comune e in questo progetto viene utilizzato l'integrato della Maxim MAX3232.

L'alimentazione interna del circuito integrato consiste in una doppia pompa di carica in grado di fornire tensioni di uscita di +5.5 V (pompa di carica di duplicazione) e di -5.5 V (pompa di carica invertente) indipendentemente dalla stabilità della tensione di alimentazione. I trasmettitori e ricevitori interni sfruttano adeguatamente queste alimentazioni per poter effettuare le opportune traslazioni tra la porta RS-232 e la periferica EUSART. Il dispositivo utilizzato può essere collegato direttamente al microcontrollore seguendo lo schema riportato nel datasheet e non richiede componenti aggiuntivi oltre le capacità delle pompe di carica.

Resta il fatto che a livello di precauzione e protezione sono stati aggiunti dal lato della porta RS-232 dei diodi zener e delle resistenze, al fine di limitare possibili picchi di tensione pericolosi per l'integrità del MAX3232.

4.3.2 Gestione del segnale RS-422

La caratteristica principale dello standard di comunicazione RS-422 risiede nel fatto che i segnali viaggiano su una linea differenziale.

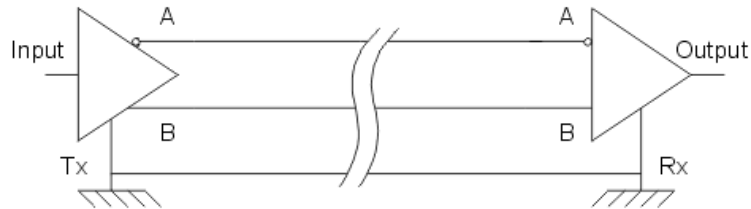


FIG. 4.37 - PRINCIPIO DI FUNZIONAMENTO DELLO STANDARD RS-422

I segnali vengono trattati nella seguente modalità:

- quando il terminale A è negativo rispetto a B, la linea rappresenta un 1 logico. Tale stato rappresenta anche l'assenza di segnale (Idle state);
- quando il terminale A è positivo rispetto a B, la linea rappresenta uno 0 logico.

Il trasmettitore produce una tensione a capo dei due terminali A e B che può variare tra i 2 e i 6 V e dispone di una connessione di massa. Sebbene un'adeguata connessione di massa sia importante, non è determinante per la discriminazione del livello logico. Per quanto riguarda il ricevitore, se la tensione tra i terminali A e B è superiore ai 200 mV, allora quest'ultimo dispone di uno specifico stato logico in uscita. Al contrario se la tensione è minore di -200 mV allora si ottiene lo stato logico opposto. Il fatto di usare una linea differenziale sostanzialmente aumenta l'immunità ai disturbi e garantisce un rapporto segnale rumore migliore. Come per lo standard RS-232 anche per l'RS-422 esistono opportuni traslatori di livello e un componente appropriato risulta essere il MAX3490.

4.4 Misura dei segnali analogici per mezzo del convertitore AD

Lo strumento del magnetometro impiega tre convertitori AD esterni alla scheda OBDH & HK principalmente per il fatto che è necessario acquisire le componenti del vettore campo magnetico separatamente e contemporaneamente. Inoltre si rende necessario una misura accurata, quindi vengono utilizzati dei convertitori di adeguata risoluzione, cioè a 12 bit. Per tutte le altre misure di tipo analogico necessarie al sistema satellite è sufficiente disporre del convertitore AD interno al microcontrollore, non essendoci l'esigenza di acquisizioni contemporanee ed essendo sufficienti 10 bit di risoluzione. Dal momento che il convertitore interno al microcontrollore è affiancato da un multiplexer analogico a 16 vie, l'obiettivo iniziale dal punto di vista dell'architettura hardware consisteva nel configurare più linee del microcontrollore come ingressi analogici. Questa opzione è stata ben presto abbandonata in virtù del fatto che vengono occupati troppi pin del PIC18LF8722.

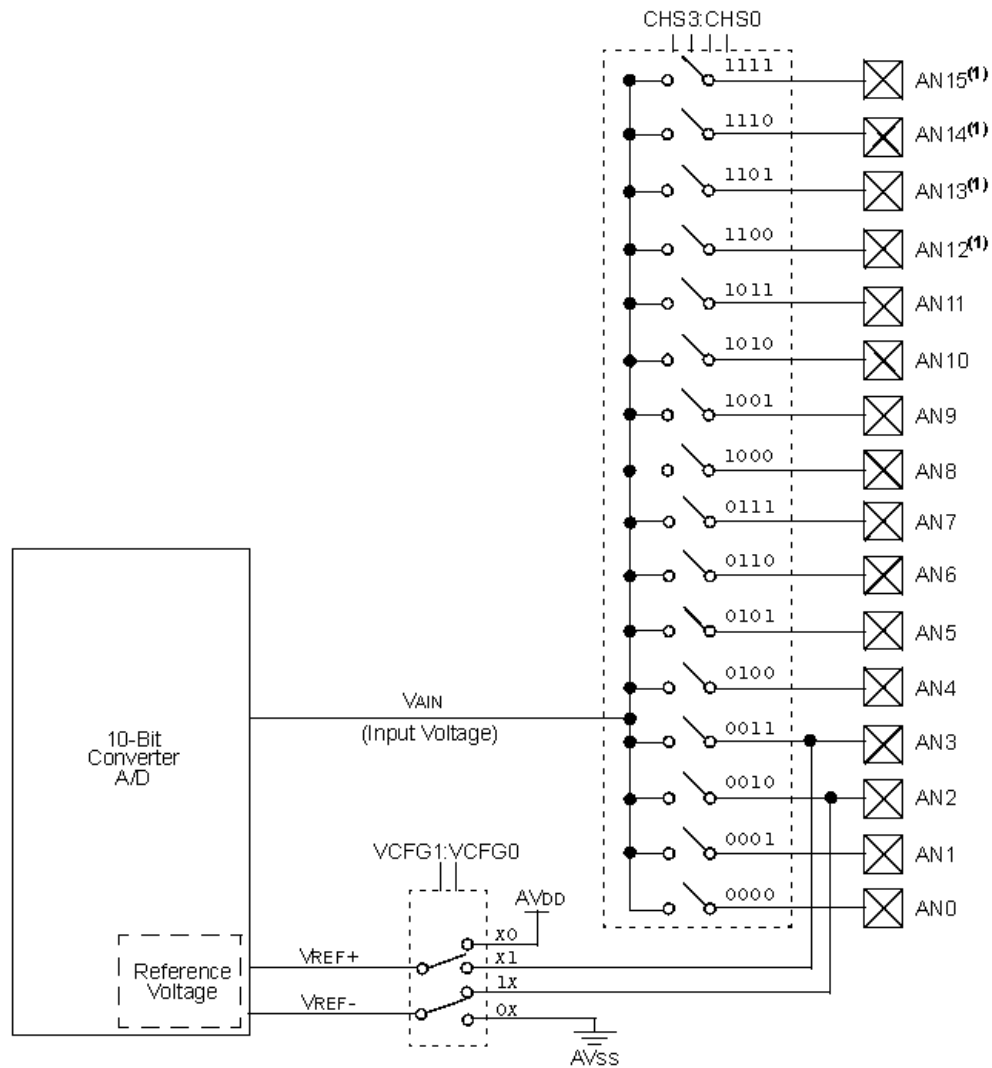


FIG. 4.38 - STRUTTURA DEL CONVERTITORE ANALOGICO DIGITALE DEL PIC18LF8722

Lo schema in figura 4.38 illustra la struttura interna del convertitore analogico-digitale del microcontrollore in uso. Si notano i 16 canali disponibili, instradati dal multiplexer configurato a sua volta via software. Tramite un'ulteriore configurazione è possibile inoltre scegliere le sorgenti dei riferimenti di tensione. Quindi, nonostante la disponibilità di un multiplexer interno alla struttura, si è reso necessario utilizzarne uno esterno in modo da occupare un solo ingresso analogico dal lato del microcontrollore.

Riassumendo, i segnali analogici di Atmocube che necessitano di essere convertiti ed elaborati dal microcontrollore sono 13:

- tensione della batteria V_{batt} ;
- corrente della batteria I_{batt} ;
- tensione dei pannelli solari V_{pann} ;
- corrente erogata dai pannelli solari I_{pann} ;
- temperatura delle batterie T_{batt} ;
- temperatura dello chassis del satellite T_{chas} ;
- potenza emessa dal blocco del ricetrasmittitore V_{pac} ;
- tensione proveniente da 6 fotodiodi dei lati del cubo *Photo1 - Photo6*.

Constatato ciò, si ricerca un multiplexer analogico con le seguenti caratteristiche:

- **disponibilità di almeno 13 canali** per soddisfare i requisiti del numero di misurazioni;
- **tensione di funzionamento di 3.3V** compatibili con l'alimentazione della scheda OBDH & HK;
- **bassa resistenza** equivalente degli switch per ridurre gli errori di acquisizione.

Il dispositivo ADG706 dell'Analog Device è un componente che soddisfa i requisiti appena elencati e viene scelto per il prototipo della scheda a microcontrollore.

La tensione di alimentazione copre pienamente il valore di 3.3 V in quanto il range si estende da +1.8 V a +5.5 V. La resistenza equivalente degli switch risulta inoltre bassa ed è dichiarata pari a 5 Ω nelle condizioni peggiori.

La disponibilità di 16 canali permette inoltre il pieno interfacciamento con tutti i segnali analogici.

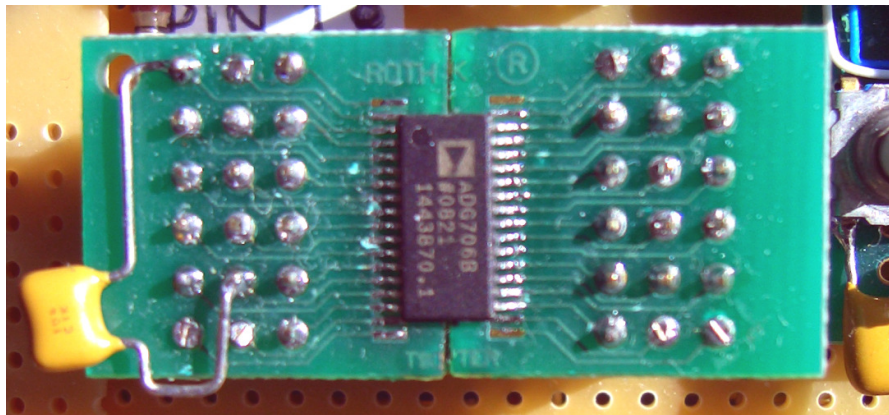


FIG. 4.39 - INGRANDIMENTO DEL DISPOSITIVO ADG706 MONTATO SU UNO ZOCCOLO DI ESPANSIONE

Il risparmio in termini di cablaggio è considerevole, dato che il PIC18LF8722 deve complessivamente mettere a disposizione 4 linee di indirizzamento per il canale più un ingresso analogico per le acquisizioni. Si rendono libere sul microcontrollore un totale di 8 linee.

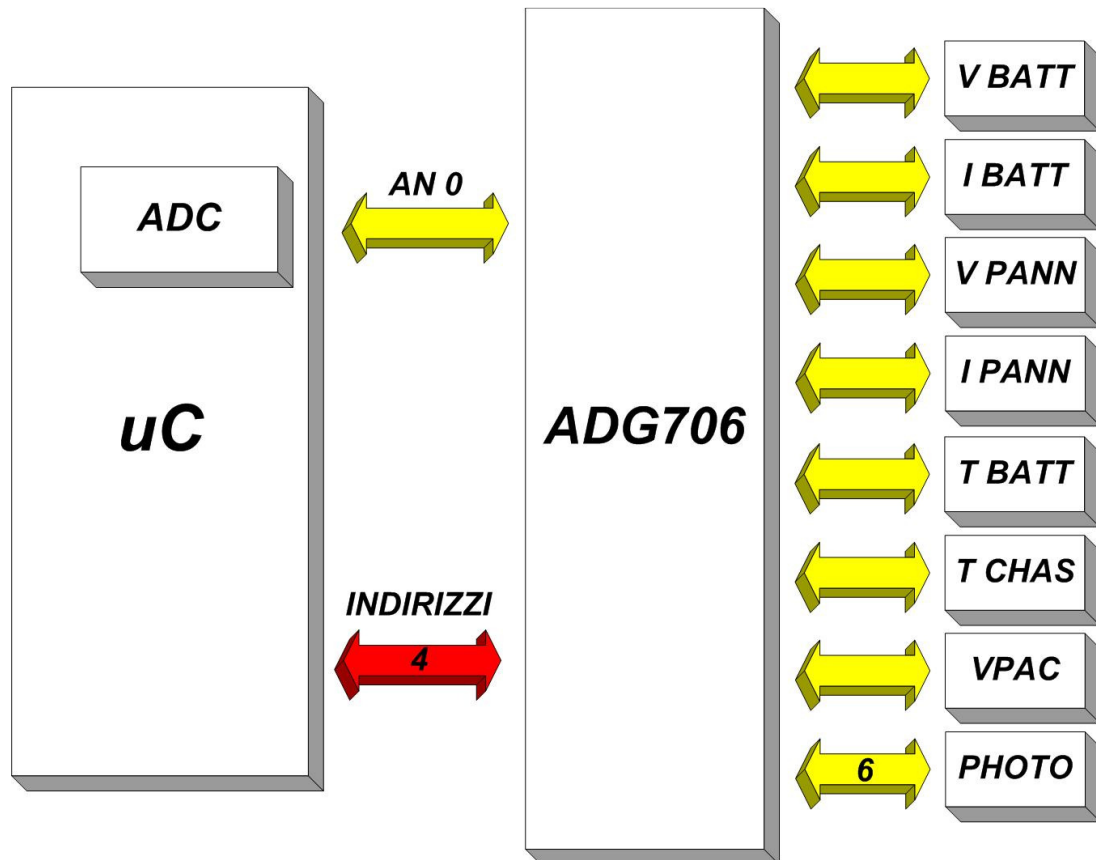


FIG. 4.40 - INTERFACCIAMENTO DEI SEGNALI ANALOGICI

4.5 Abilitazione dei blocchi funzionali della Power Supply Unit

Riguardo questa parte dell'architettura hardware non ci sono particolari considerazioni da fare. Quello che viene fatto consiste solamente nel dedicare alcune linee general purpose del microcontrollore a delle funzioni di abilitazione dei moduli costituenti la Power Supply Unit. I sottosistemi dell'alimentatore del satellite infatti sono per lo più formati da integrati indipendenti che generano le adeguate tensioni; ogni sorgente è attiva solamente se un MOS viene posto in uno stato di conduzione. Il suo gate viene controllato interamente dalle linee di abilitazione rese disponibili in questa sezione.

Sono accessibili in totale 9 controlli organizzati con il seguente criterio:

- 3 abilitazioni per i moduli a +3.3 V;
- 2 abilitazioni per i moduli a +5 V;
- 1 abilitazione per la tensione di alimentazione dell'SDD di +6 V;
- 1 abilitazione per l'alimentatore ad alta tensione dell'SDD di -600 V;
- 1 abilitazione per l'alimentazione del ricetrasmittitore di +6.5 V;
- 1 abilitazione per l'Antenna Deployment, ovvero per lo sgancio dell'antenna.

4.6 *Considerazioni generali e scelte comuni*

A livello dell'hardware sono state adottate delle generalità comuni per tutti i componenti del sistema. In particolare vengono impiegate delle capacità di 100 nF collegate in parallelo all'alimentazione di ogni circuito integrato al fine di ridurre i disturbi. L'alimentazione della parte analogica del microcontrollore è filtrata per mezzo di un filtro passa basso RC con $R=100\ \Omega$ e $C=100\ \text{nF}$ in modo da stabilizzare i riferimenti dell'AD converter.

La linea RA4 del microcontrollore è collegata all'alimentazione digitale positiva con una resistenza di pull up da 10 k Ω , dato che è un'uscita di tipo a drain aperto.

Tra gli ingressi SHDN del magnetometro e il microcontrollore e gli ingressi T/R, Va, Vb del ricetrasmittitore e il microcontrollore sono interposti dei buffer in quanto la compatibilità dei livelli logici non è completamente garantita.

Gli oscillatori a quarzo, sia il principale che il secondario, sono collegati come da manuale, in particolare con l'ausilio di due capacità ciascuno. La frequenza del clock primario è pari a 4 MHz e ciò implica che ogni istruzione venga eseguita in 1 μs . Questa frequenza condiziona molto il consumo del microcontrollore, di conseguenza si cerca di non spingersi particolarmente in alto.

5 Sistema di alimentazione del modulo OBDH & HK

Questa parte di progetto risulta fin dai primi momenti un po' critica in quanto gli stadi di alimentazione dell'intero satellite non sono ancora disponibili in via definitiva. In particolare, un alimentatore per il modulo OBDH & HK è totalmente assente dall'inizio del progetto della scheda a microcontrollore.

I principali aspetti che sono stati considerati riguardano il tipo di circuito integrato da utilizzare e il dimensionamento di tale componente in base alla corrente totale assorbita dal modulo OBDH.

Vengono inoltre trattati ulteriori aspetti correlati con l'alimentazione come il circuito di reset, l'avvio del microcontrollore ai primi istanti di applicazione della tensione e il sistema di carica della batteria.

5.1 Regolazione della tensione della batteria

Il punto di partenza da tenere in considerazione è che l'intero satellite è alimentato da una batteria agli ioni di litio di tensione nominale di 3.75 V. L'intero modulo OBDH & HK lavora completamente con tensioni di 3.3 V ed in particolare tale vincolo deve essere rigorosamente rispettato per quanto riguarda la tensione di riferimento dell'AD converter interno al microcontrollore e le tensioni dei livelli logici del bus SPI. L'AD converter richiede infatti una tensione stabile per poter effettuare delle misurazioni analogiche accurate, mentre il bus SPI deve essere compatibile con i diversi sottosistemi del satellite. In base a tali considerazioni si rende necessario l'uso di un integrato regolatore in modo da stabilizzare tutte le tensioni di lavoro della scheda a microcontrollore a 3.3 V.

Avendo a disposizione una batteria con tensione di scarica di 2.5 V, quindi che riesce ad erogare corrente fino a quando la tensione non scende al di sotto di tale valore, la prima idea è quella di implementare un regolatore step up-down in modo da far lavorare il satellite a piene funzioni anche quando la tensione di alimentazione scende al di sotto dei 3.3 V. Questo alimentatore però viene scartato in quanto prevale la considerazione che tale tipologia possa creare disturbi sulle tensioni di riferimento e sui vari bus dati di cui è dotato l'intero modulo.

Si opta di conseguenza per un regolatore lineare a bassa caduta, restringendo il campo di operatività nell'intervallo da 3.3 V a 3.7 V. Ciò non risulta essere una limitazione, in quanto nel momento in cui la tensione della batteria è scesa a circa 3.3 V le curve del grafico di figura 5.1 mostrano che ci si trova già in prossimità del ginocchio e quindi la batteria è già quasi completamente esaurita.

A questo punto per dimensionare il componente di regolazione si fa un'analisi degli assorbimenti di corrente del modulo OBDH & HK valutando la corrente di alimentazione di ogni singolo componente.

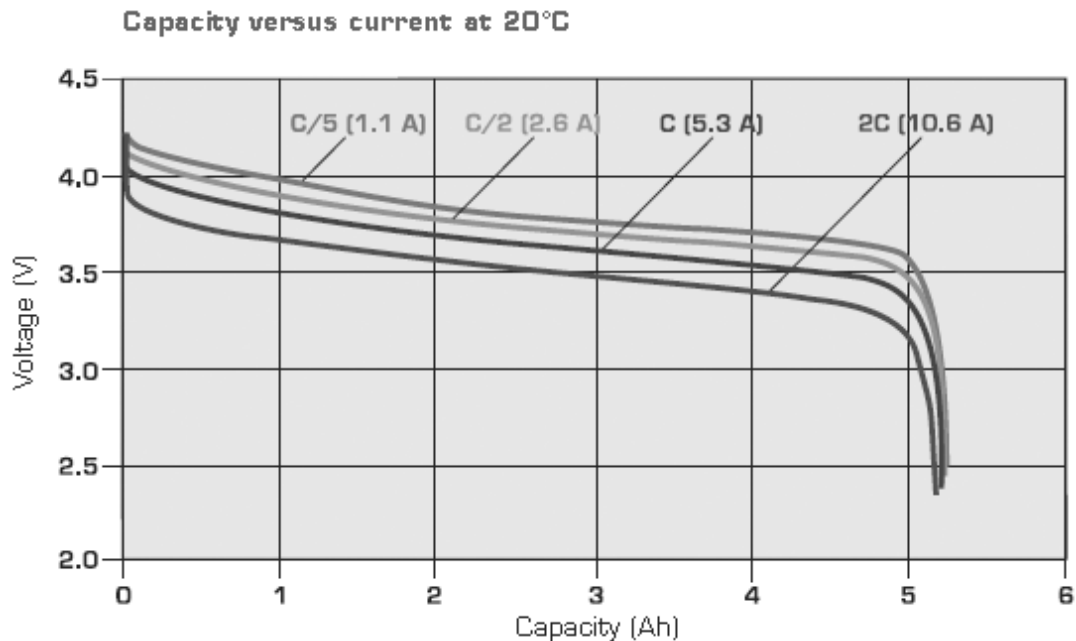


FIG. 5.1 - CURVE DI SCARICA DELLA BATTERIA IN USO SAFT MP174865

A tal proposito si nota che le maggiori correnti sono assorbite dalla memoria SRAM e dal microcontrollore; gli altri componenti non portano un budget di corrente sensibile al dimensionamento del regolatore. Il datasheet della memoria dichiara una corrente massima in condizioni di operatività di 30 mA, l'assorbimento del microcontrollore invece risulta molto variabile in quanto dipende dalle periferiche interne abilitate, dalla frequenza di clock e da come vengono caricate le varie porte. Non essendoci carichi pesanti sui vari pin, l'assorbimento totale del μC è comunque dell'ordine dei milliampere e in ogni caso molto più basso dell'assorbimento della corrente della memoria SRAM.

Ci sono svariati componenti che soddisfano tali requisiti; per questo impiego si utilizza un dispositivo della Micrel, il MIC5205 - 3.3, che presenta buone caratteristiche di rumore, caduta e sufficiente erogazione di corrente. Il MIC5205 è un componente progettato per sistemi alimentati a batteria, come in questo caso, e presenta una caduta di tensione di soli 17 mV con carichi leggeri e una caduta tipica di 165 mV in condizione di massima corrente erogabile, cioè 150 mA. Utilizzando quindi questo regolatore lineare il raggiungimento della soglia minima di regolazione, circa 3.3 V, deve essere monitorata in modo da poter saper agire opportunamente. Ciò verrà trattato nella paragrafo dedicato alla carica della batteria e all'ibernazione dello stato del microcontrollore.

5.2 *Circuito supervisore del Reset Hardware del μ C*

Con il termine supervisore si indicano tutti quei dispositivi che trattano la funzione POR (Power on reset), BOD (Brown - out detect) e Watchdog in sistemi di tipo Embedded.

È doveroso chiarire fin da subito che nonostante tutte le funzioni elencate siano presenti nella struttura del microcontrollore, le prime due, cioè il POR e BOD, vengono implementate in un dispositivo hardware esterno; ciò perché è buona norma in un sistema embedded, dove sono presenti più circuiti integrati, delegare tale compito ad un circuito che sincronizzi la partenza del sistema in maniera univoca e indipendente solo nel momento in cui l'alimentazione principale si sia stabilizzata del tutto.

La funzione Watchdog è anche utilizzata, ma a tal fine viene impiegata la risorsa interna al microcontrollore; la gestione effettuata per mezzo del software viene chiarita in maggior dettaglio nel capitolo 7 quando si tratta il sistema operativo del satellite.

Le funzioni POR e BOD invece vengono subito descritte e sono riassunte rispettivamente dalle due seguenti azioni:

- durante l'avvio di una sequenza di power up, cioè quando viene fornita tensione al sistema, il circuito supervisore mantiene il microcontrollore in stato di reset fino al momento in cui l'alimentazione non ha raggiunto il livello corretto e non si è stabilizzata;
- nel caso l'alimentazione cada sotto una determinata soglia, il dispositivo supervisore manda in stato di reset il microcontrollore.

Si devono considerare alcune situazioni che si possono verificare e in base a ciò effettuare la scelta di un componente adeguato:

- presenza di disturbi nell'alimentazione quando il sistema viene avviato;
- comportamenti anomali del μ C quando l'alimentazione scende sotto una soglia;
- comportamenti anomali del μ C quando viene tolta alimentazione al sistema.

Spesse volte accade che in fase di power up sia presente un glitch o del rumore sulla linea di alimentazione. Un glitch può molto spesso causare l'avvio inadeguato o comunque errato del microcontrollore ottenendo come risultato un comportamento anomalo dell'intero sistema embedded. In questa circostanza un circuito supervisore previene malfunzionamenti e lo fa mantenendo il microcontrollore in stato di reset fino a quando l'alimentazione non ha raggiunto un valore di soglia che garantisce la stabilità. Il tempo di azione di tale intervento varia a seconda dei dispositivi supervisor impiegati ma di solito è compreso tra i 150 e i 500 ms. Quando passa questo tempo, il dispositivo rilascia la linea di reset e permette al microcontrollore di iniziare l'esecuzione del suo codice.

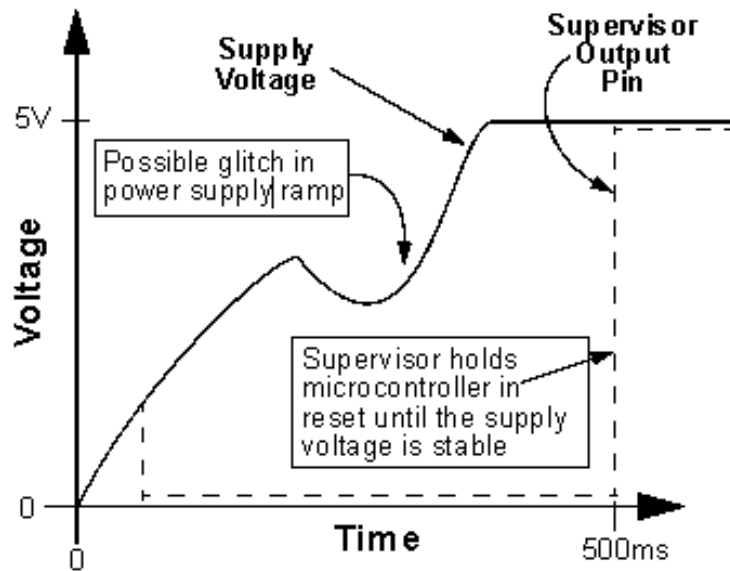


FIG. 5.2 - RAMPA DI ALIMENTAZIONE SOVRAPPOSTA AD UN GLITCH (FUNZIONE POR)

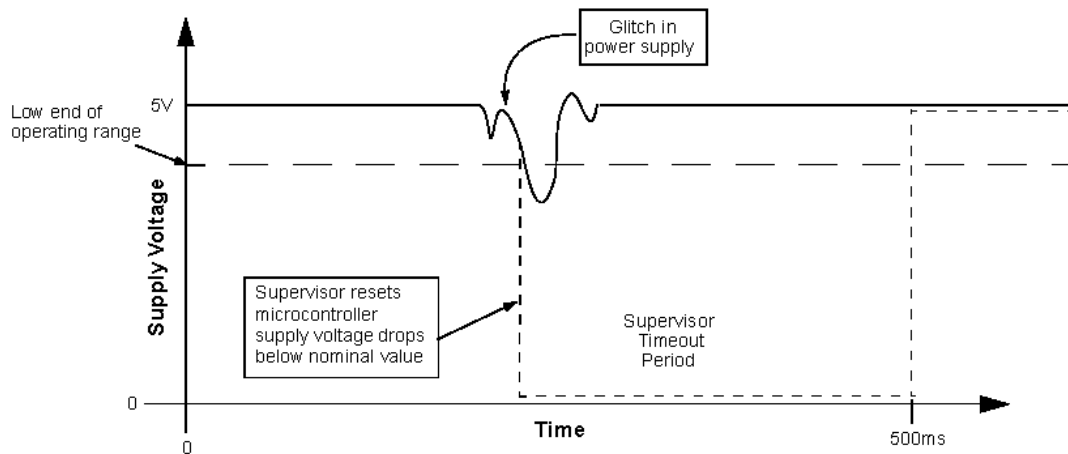


FIG. 5.3 - GLITCH DI TENSIONE CHE PORTA FUORI SOGLIA MINIMA L'ALIMENTAZIONE (FUNZIONE BOD)

Può accadere inoltre che la tensione di alimentazione si immerga in una zona dove non viene rispettata una certa soglia minima, prima di ritornare nuovamente in zona di sicurezza. Questa condizione può verificarsi ad esempio per un'inadeguata regolazione dell'alimentazione, dispositivi nel sistema che vengono accesi e spenti o malfunzionamenti di vario genere.

Sfortunatamente condizioni di brown-out di solito non si manifestano in fase di sviluppo ma avvengono in una fase finale quando sono presenti tutti i componenti. Ciò si manifesta in differenti modi come ad esempio livelli logici interpretati scorrettamente o locazioni della memoria RAM interna al microcontrollore corrotte o modificate senza nessun tipo di controllo.

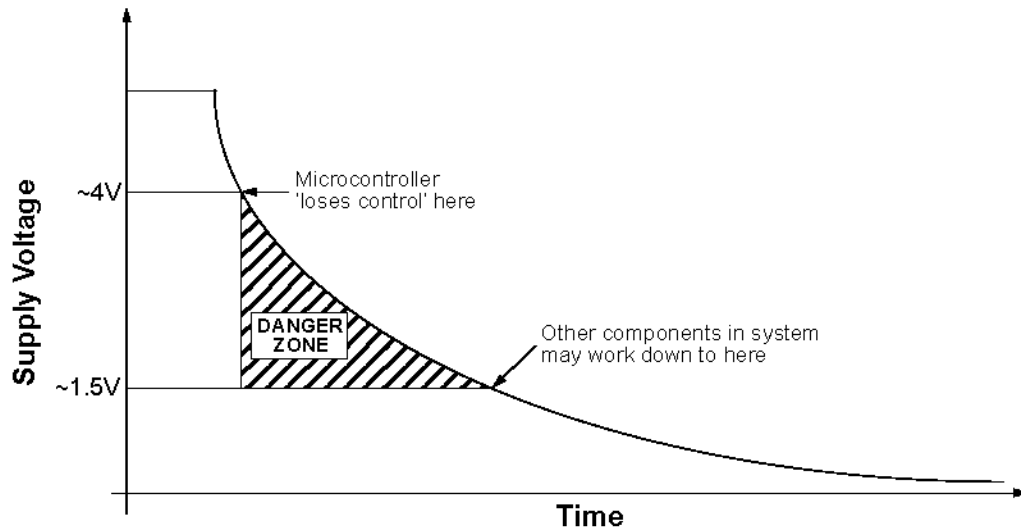


FIG. 5.4 - IL DECADIMENTO DELL'ALIMENTAZIONE COMPORTA UNA PERDITA DI CONTROLLO DA PARTE DEL MICROCONTROLORE

Un altro problema da tenere in considerazione è quando viene tolta l'alimentazione e la tensione scende a rampa in maniera molto lenta. Questa è una tipica situazione nella quale si ha a che fare con sistemi alimentati a batteria oppure con alimentatori da banco spenti manualmente.

Il codice del microcontrollore viene eseguito in maniera aleatoria e questo perché non c'è sufficiente tensione per alimentare la RAM interna la quale non carica opportunamente il Program Counter come anche altre variabili temporanee e registri. Più tempo l'alimentazione impiega a scendere lentamente, più grande è il disagio e il pericolo che una situazione come quella appena descritta si verifichi.

Se nel sistema embedded è presente un display si potranno verificare dei casi in cui alcuni caratteri vengono inviati in modo casuale. Se il sistema invece dispone di altri componenti come una memoria EEPROM esterna, la cui tensione di alimentazione può essere minore di quella del microcontrollore, allora possono incombere problemi più seri e gravosi. Queste memorie infatti lavorano con tensioni basse come 1.8 V e possono rispondere a comandi anche di 1.2 V.

Se il microcontrollore esegue una porzione del suo codice che esegue la scrittura della EEPROM, allora c'è la possibilità che dati casuali vengano scritti e non vengano rilevati al successivo avvio del sistema.

Note ora le situazioni di maggior criticità, la scelta del componente di monitor deve tenere in considerazione una breve serie di parametri, quali la tensione di reset (trip point), la tipologia dello stadio di uscita del dispositivo in questione e la polarità del reset del microcontrollore.

- La scelta del corretto trip point dipende principalmente dal range di tensione di operatività del microcontrollore in uso e dalla variazione della tensione di alimentazione.
- L'uscita dello stadio pilota può essere di vario tipo ma quello più comune impiega stadi push pull o stadi a collettore o drain aperto con o senza resistenza di pull up. La tipologia a collettore o drain aperto ha la caratteristica di permettere a più di una sorgente di mandare il microcontrollore in uno stato di reset. Altre sorgenti

comuni possono essere un pulsante per il reset manuale o un dispositivo di monitor della temperatura che entra in azione in caso di eccessivi riscaldamento.

- Infine la polarità del segnale di reset dipende strettamente dal microcontrollore in questione: alcuni segnali possono essere attivi alti, altri attivi bassi. È facile reperire componenti che dispongono anche di entrambe le uscite, una complementare dell'altra per l'appunto.

Oltre a queste considerazioni appena fatte bisogna tener conto che la scheda OBDH & HK dispone di un connettore dedicato per la programmazione del microcontrollore, cioè il *ICSP*, In Circuit Serial Programming; esso utilizza lo stesso piedino di reset gestito dal circuito hardware esterno di POR e BOD, quindi è necessario considerare ulteriori aspetti onde evitare conflitti tra il programmatore e il dispositivo di reset.

È opportuno isolare infatti il piedino $V_{pp}/\overline{MCLR}$ del microcontrollore in quanto la tensione di programmazione è dichiarata ad un valore tipico di 12 V con picchi massimi a 13.5 V. Ciò è un problema nel caso siano collegati come nel sistema trattato, circuiti di supervisione facenti capo proprio al piedino di reset. Quello che risulta evidente è che il circuito di uscita del dispositivo supervisore viene pilotato con una tensione ben superiore a quella di alimentazione. Questo vincolo obbliga a escludere sicuramente a priori dispositivi supervisor con stadio d'uscita push pull, in quanto comuni processi di produzione di circuiti integrati come l' N-Well permettono una circolazione extra di corrente nel MOS di tipo P attraverso il substrato.

I dispositivi a drain o collettore aperto non sono vincolati da tale problema, ma comunque bisogna considerare qual è la tensione massima sopra quella dell'alimentazione tollerata dal dispositivo.

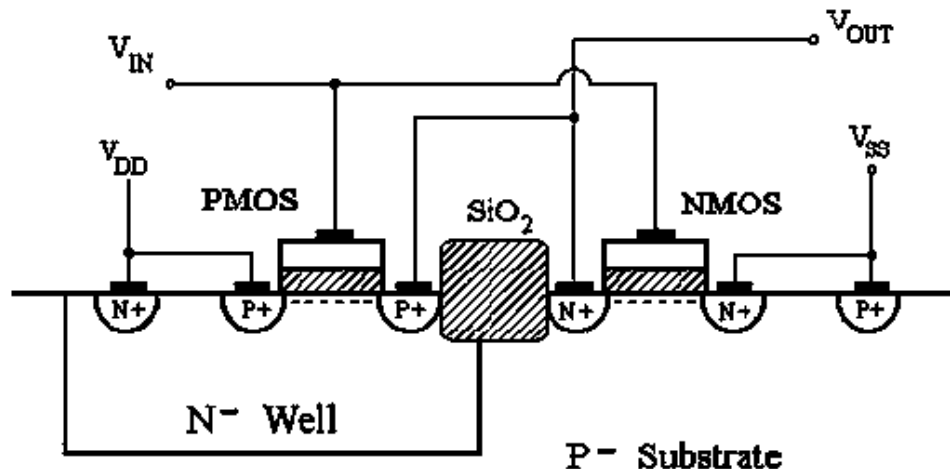


FIG. 5.5 - POLARIZZAZIONE DIRETTA DELLA GIUNZIONE DRAIN-SUBSTRATO PER IL MOS A CANALE P

In base alle considerazioni precedenti, le caratteristiche di base che deve avere il dispositivo di monitor del reset sono:

- **Trip point:** la tensione minima di funzionamento del microcontrollore è di 2 V, pertanto l'obiettivo che si vuole perseguire è quello di garantire l'avvio del core e quindi di garantire il controllo da parte del microcontrollore dell'intero sistema anche con batterie molto scariche. Viene scelto pertanto come trip point una tensione di 2.3 V vicina alla soglia minima ma leggermente superiore, in modo tale da garantire una certa tolleranza di sicurezza.

- **Stadio d'uscita:** non potendo di certo impiegare dispositivi con stadio d'uscita push pull a causa del problema del substrato si adotta un dispositivo a drain aperto. L'unica accortezza da considerare è la tensione massima sopra quella dell'alimentazione tollerabile dal componente ed eventualmente ricorrere ad un sistema di protezione.
- **Polarità:** il microcontrollore impiegato dispone di un reset attivo basso, di conseguenza deve essere reperibile un'uscita della stessa polarità.

Il dispositivo impiegato come supervisore di reset nel modulo OBDH & HK è l' *ISL88011* della Intersil con trip point di 2.3 V e timer POR di 200 ms

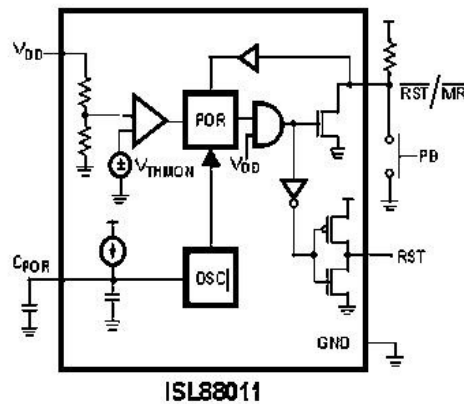


FIG. 5.6 - SCHEMA INTERNO DEL DISPOSITIVO DI SUPERVISIONE

Il dispositivo viene fornito in un piccolo package per montaggio superficiale; si notano il piedino di reset a logica negativa e il piedino per il collegamento di un'eventuale capacità per aumentare il ritardo di POR.

Dallo schema a blocchi si evidenzia il comparatore collegato alla tensione di riferimento per il trip point e lo stadio di uscita a drain aperto per il reset di tipo negativo.

È presente anche un'uscita di tipo push pull ma non viene impiegata e fa capo ad un piedino di reset a logica positiva.

I diagrammi che illustrano il funzionamento del supervisore vengono riportati in figura 5.7.

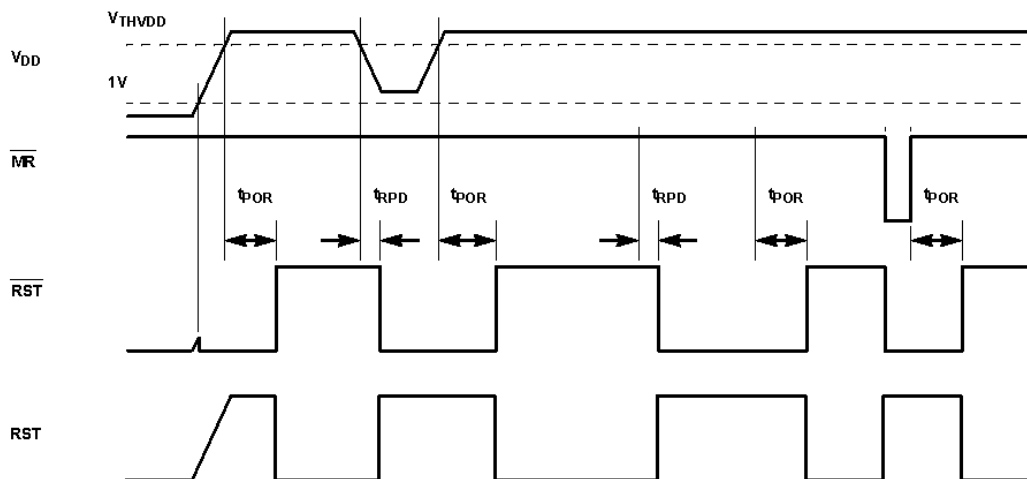


FIG. 5.7 - FORME D'ONDA D'INTERESSE PER IL DISPOSITIVO DELLA INTERSIL

Le considerazioni da fare riguardo questi diagrammi sono:

- si nota che il dispositivo comincia a svolgere il suo compito di monitor quando l'alimentazione raggiunge un valore minimo di 1 V. E' infatti solo a questo punto che il segnale di reset positivo (quello non utilizzato nell'applicazione del sistema) sale a rampa verso il livello logico alto. Non preoccupa lo stato d'incertezza che hanno le uscite di reset sotto la tensione minima di 1 V in quanto il microcontrollore si avvia appena a 2 V;
- quando la tensione di alimentazione raggiunge il trip point di 2.3 V (V_{thvdd}) parte il timer POR che garantisce da questo momento uno stato di reset del microcontrollore per 200 ms in modo da permettere una stabilizzazione dell'alimentazione;
- qualora l'alimentazione per un qualsiasi motivo dovesse ricadere al di sotto della soglia di 2.3 V, allora il microcontrollore viene riportato in uno stato di reset onde evitare comportamenti non prevedibili; questa è la condizione BOD. Quando la tensione di alimentazione risale sopra la soglia allora riparte il timer POR e il microcontrollore viene sbloccato nuovamente dopo 200 ms.

Come è stato detto, un parametro importante da considerare nel dispositivo supervisore è la tensione massima rispetto al riferimento tollerata da ciascun piedino. Dal datasheet questo valore è di 7 V, pertanto sono necessari degli accorgimenti per garantire una protezione dalla sovratensione della tensione di programmazione V_{pp} di 12 V.

L'idea di base è quella di servirsi di un diodo in modo da sfruttare la sua caduta di tensione fissa tra anodo e catodo. Si pone necessario anche l'ausilio di una resistenza di limitazione della corrente per il diodo stesso. La soluzione adottata viene riassunta nel seguente schema elettrico, ricavato da una simulazione in LTSpice descritta a breve:

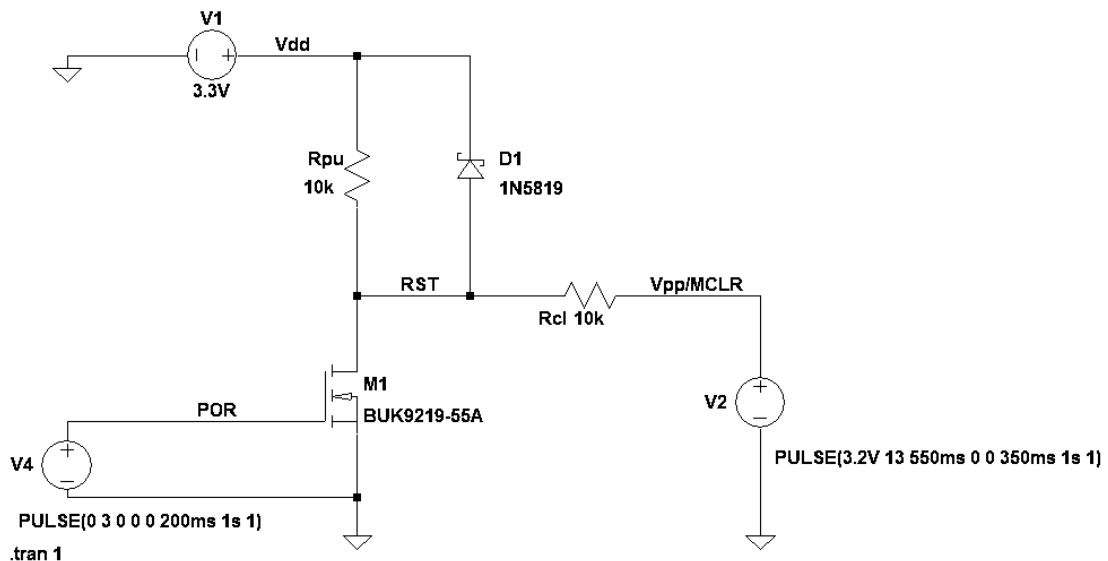


FIG. 5.8 - SCHEMA ELETTRICO UTILIZZATO PER LA SIMULAZIONE IN AMBIENTE LTSPICE

Si riporta ora lo studio analitico del circuito di reset e programmazione nelle due modalità di funzionamento, cioè nella modalità di normale operatività del sistema e nella modalità di programmazione e debug.

a. Normale operatività:

- *microcontrollore in stato di reset:* il mosfet è in conduzione dato che il circuito POR alimenta il gate e collega a massa il catodo del diodo polarizzandolo inversamente; il mosfet inoltre mette in pull down il piedino $V_{PP}/MCLR$ del microcontrollore per mezzo della resistenza R_{CL} garantendo lo stato di reset. La resistenza R_{PU} è alimentata ed in essa scorre una corrente di 0.33 mA.
- *microcontrollore attivo:* il mosfet è interdetto e il diodo è polarizzato inversamente consentendo alla linea di reset $V_{PP}/MCLR$ del microcontrollore di portarsi in uno stato logico attivo per mezzo della serie delle resistenze R_{PU} e R_{CL} . I consumi sono trascurabili visto l'ingresso CMOS del piedino del microcontrollore.

b. Programmazione e debug via ICSP:

in questa fase il microcontrollore non si trova in stato di reset pertanto il mosfet non conduce. Il potenziale dell'anodo del diodo viene a trovarsi ad un valore maggiore rispetto a quello del catodo in quanto la V_{PP} è maggiore della V_{DD} , quindi passa in conduzione vincolando la tensione di drain del mosfet.

Per quanto riguarda il dimensionamento dei componenti si sono fatte le seguenti considerazioni:

- **R_{PU} :** dal datasheet del supervisore ISL88011 la resistenza di pull up per il drain del mosfet deve essere compresa tra 5 kΩ e 100 kΩ. Si sceglie un valore di 10 kΩ in modo da stare sopra la soglia minima e garantire un rumore non eccessivo. Si tiene presente che il piedino di reset del microcontrollore va in pull up con una resistenza equivalente pari alla serie di questa resistenza più la resistenza di limitazione della corrente.
- **R_{CL} :** questa resistenza determina la corrente massima che scorre nel diodo. Inoltre, come da ispezione circuitale, tale resistenza determina il valore con il quale il reset del microcontrollore si pone in pull down e determina per metà il valore della resistenza totale con la quale lo stesso piedino si pone in pull up. Scegliendo anche per tale componente un valore di 10 kΩ si ottiene una bassa corrente di polarizzazione diretta del diodo e si garantisce un pull down non eccessivamente influenzato dal rumore.

Tendendo conto di queste considerazioni, il valore di pull up totale è di 20 kΩ.

$$V_D = V_{DD} + V_{AK} = 3.3 + 0.3 = 3.6V$$

$$I_{CL} = \frac{V_{PP_MAX} - V_D = V_{PP} - V_{DD} - V_{AK}}{R_{CL}} = \frac{13.25 - 0.3 - 3.3}{10 \cdot 10^3} = 0.96mA \approx 1mA$$

Dai calcoli si vede che, in fase di programmazione, la tensione di drain è vincolata a 3.6 V e il diodo conduce una corrente di 1 mA.

- **Diodo:** viene impiegato un diodo di tipo schottky in modo da ridurre la caduta di tensione e quindi non alzare di troppo la tensione di drain. Rispetto ai comuni diodi rettificatori, questa tipologia grazie alla giunzione metallo semiconduttore, garantisce una caduta di tensione tra anodo e catodo più che dimezzata.

Dopo lo studio analitico viene effettuata anche una simulazione per avere ulteriore conferma della corretta operatività del circuito;

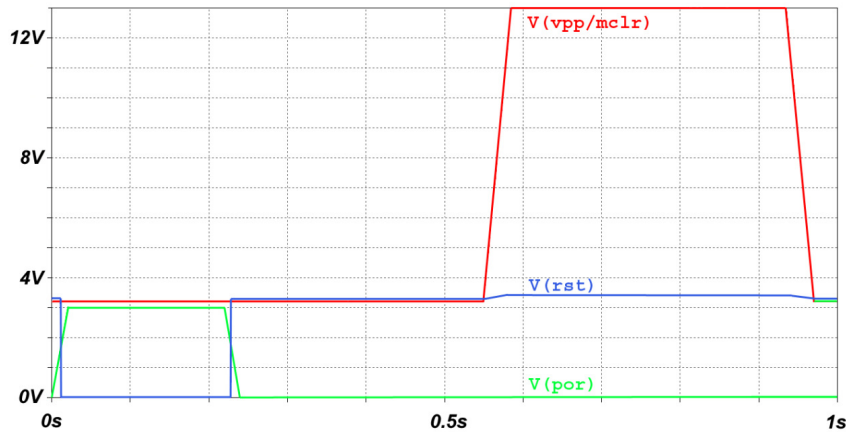


FIG. 5.9 - FORME D'ONDA PRINCIPALI RICAVATE DALLA SIMULAZIONE IN LTSPICE

Si suppone il circuito sia inizialmente spento, poi all'istante di tempo di 0 s viene fornita l'alimentazione di 3.3 V al sistema. Il sistema POR polarizza il gate del mosfet per 200 ms causando la chiusura della linea di reset verso massa per lo stesso periodo. Svolto il suo compito il timer POR rilascia il gate permettendo alla linea di reset di salire a 3.3 V per mezzo della resistenza di pull up di 10 k Ω portando il microcontrollore in uno stato di avvio. Successivamente tra 500 ms e 600 ms, si suppone di entrare in modalità debug per mezzo della porta ICSP, la quale fornisce direttamente al piedino $V_{pp}/\overline{MCLR}$ del microcontrollore la tensione di programmazione di 12 V. Si vede che effettivamente tale linea si porta a quest'ultimo valore di tensione ma la tensione che fa capo al drain del dispositivo supervisore è bloccata a 3.6 V; in ogni caso è ben al di sotto dei 7 V massimi consentiti.

In conclusione si verifica che, quando viene applicata la tensione di programmazione, il diodo passa in polarizzazione diretta bloccando il drain a poco più della tensione di alimentazione evitando danneggiamenti del supervisore.

Dopo la fase di progetto, si passa alla realizzazione pratica del circuito di reset. Il tutto viene montato su una basetta da laboratorio e viene testato l'adeguato comportamento del componente supervisore in tutte le condizioni descritte prima.

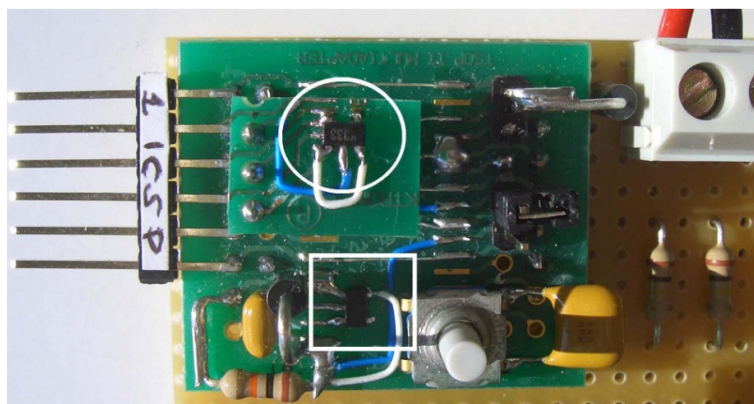


FIG. 5.10 - FOTO DELLA BASETTA SPERIMENTALE DEL CIRCUITO DI RESET CON EVIDENZIATO NEL QUADRATO IL DISPOSITIVO ISL88011 E NEL CERCHIO IL REGOLATORE MIC5205

FIG. 5.11 - SCHEMA ELETTRICO DI INTERFACCIAMENTO TRA CIRCUITO DI RESET, ICSP, MICROCONTROLLORE E REGOLATORE DELL'ALIMENTAZIONE



Per mezzo di un oscilloscopio triggerato sull'alimentazione si verifica che effettivamente il timer POR agisca per 200 ms mantenendo il reset ad uno stato basso dal momento che viene fornita alimentazione. Si riporta l'immagine tratta direttamente dall'oscilloscopio in figura 5.12.

Il canale 1, segnale in alto, mostra l'istante di applicazione dell'alimentazione. Il canale 2, segnale in basso, mostra come il POR agisca adeguatamente per 200ms mantenendo basso ($VOL_{max}=0.4\text{ V}$) il reset prima di rilasciarlo.

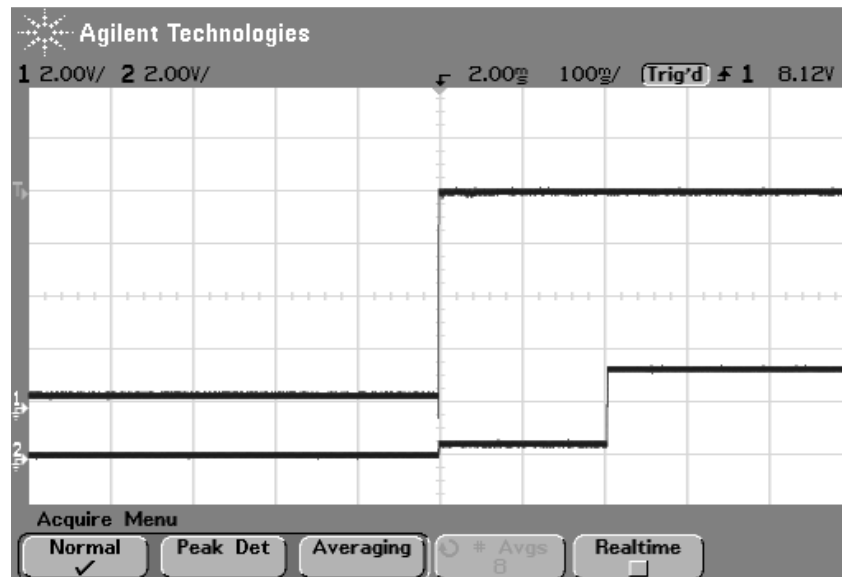


FIG. 5.12 - AZIONE DEL POR ALL'ISTANTE DI APPLICAZIONE DELL'ALIMENTAZIONE

Un'altra verifica effettuata in laboratorio riguarda l'operatività del POR dopo il reset manuale per mezzo del pulsante. Come dai diagrammi precedenti, si verifica che dal momento del rilascio del pulsante la linea di reset non si porti subito alta ma agisca nuovamente il POR per 200ms (figura 5.13).

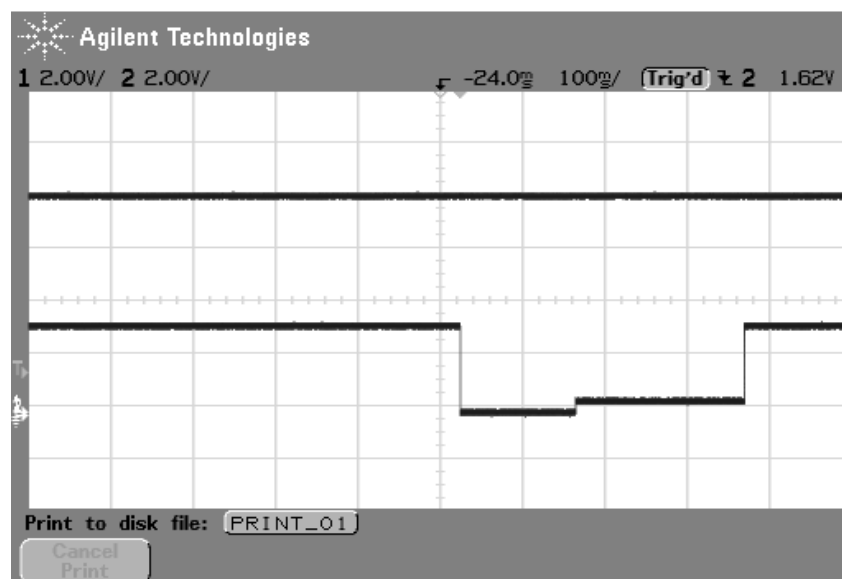


FIG. 5.13 - AZIONE DEL POR DOPO IL RILASCIO DEL PULSANTE DI RESET

Dal canale 2, segnale in basso, si nota la tensione del reset costante a 3.3 V. Immediatamente alla pressione del tasto il segnale si porta al valore di riferimento di massa e ci rimane fino a quando lo stesso viene rilasciato. Solo a questo punto il timer POR interviene mantenendo ancora a livello basso il reset per ancora 200 ms. Si nota una leggera differenza nei valori di livello basso, in quanto il primo tratto è generato dal collegamento fisico del piedino di reset verso massa per mezzo del pulsante, il secondo tratto invece è determinato dalla V_{ds} del mosfet in fase di conduzione ($VOL_{max}=0.4$ V).

L'ultima prova sperimentale effettuata, consiste nella verifica della funzione di brown out. Si verifica che quando l'alimentazione scende al di sotto della tensione di soglia di 2.3 V, il dispositivo della Intersil polarizza effettivamente il mosfet collegando la linea di reset verso massa.

Il canale 1, quello in alto, mostra la tensione di alimentazione che viene diminuita manualmente verso un valore sotto soglia. Il trigger impostato sul fronte di discesa di questo canale proprio sulla soglia di valore 2.3 V, mostra che quando viene raggiunto quest'ultimo valore, allora la linea di reset passa subito a livello logico basso. Ciò si visualizza sulla forma d'onda in basso sul canale 2 (figura 5.14).

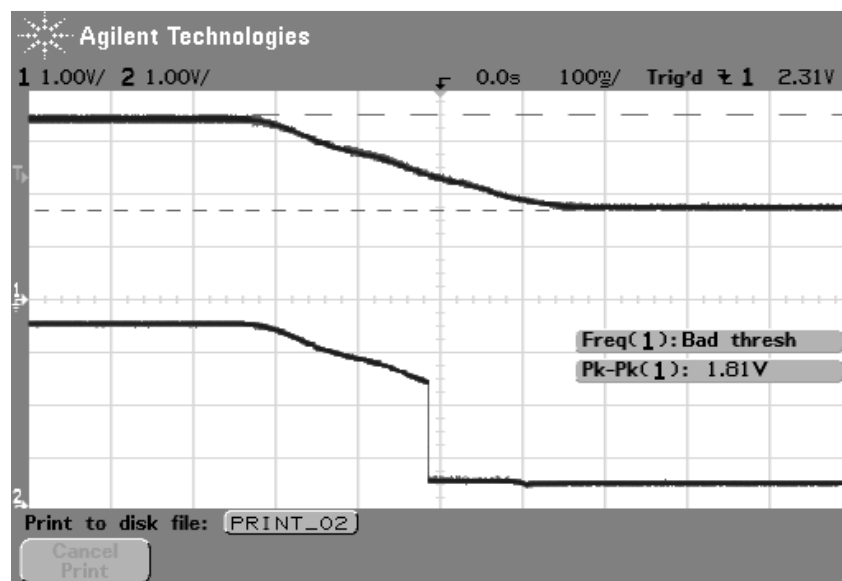


FIG. 5.14 - AZIONE DEL BOD AL RAGGIUNGIMENTO DELLA SOGLIA DI 2.3V

5.3 Sistema di carica della batteria e di ibernazione dello stato del μC

Il vincolo di partenza riguardo il sistema di ricarica è che i pannelli solari non sono permanentemente collegati all'accumulatore, ciò per evitare danneggiamenti dovuti ad un sovraccarico di energia in condizione della massima tensione raggiunta dalla batteria.

Il problema principale della carica della batteria sta nel fatto che non è possibile, o in ogni caso non è sicuro, affidarne completamente la gestione al solo microcontrollore; infatti quest'ultimo per poter svolgere l'algoritmo di ricarica a sua volta deve essere alimentato.

Visto che il satellite viene tenuto fermo prima del lancio senza alcuna ricarica, il rischio maggiore è che esso venga espulso nello spazio con una condizione di carica iniziale non sufficiente a garantire le condizioni di avvio del sistema. Ci si troverebbe di conseguenza in una condizione nella quale la carica della batteria non verrebbe mai inizializzata.

Le soluzioni per evitare questo problema possono essere l'utilizzo di una batteria supplementare di tipo tampone o un circuito puramente hardware che sia indipendente dal resto del sistema. La batteria tampone ha generalmente tempi di scarica maggiore e permette quindi quasi sicuramente al microcontrollore di avviarsi. L'idea sostanzialmente è quella di fornire Atmocube di un'alimentazione secondaria come è presente nelle schede madri dei PC per mantenere vivo il real time clock del sistema. Questa prima soluzione seppur valida viene scartata per diversi motivi:

- si vuol evitare di limitare la vita del satellite stesso alla durata della batteria tampone;
- non si vogliono avere più fonti di energia;
- si vuole evitare di separare le alimentazioni all'interno del modulo OBDH & HK, scelta quasi obbligata dal momento che la memoria SRAM assorbe fino a 30 mA.

La soluzione che si propone di adottare invece è di progettare un circuito hardware che gestisca la carica quando la batteria si trovi in una condizione nella quale non è garantito, o perlomeno non è scontato, l'avvio del microcontrollore stesso. In tal maniera si assicura che la batteria riesca a rientrare nel range di tensione facendo riprendere il controllo al modulo OBDH & HK. Un integrato supervisore di reset del tipo visto in precedenza può svolgere a dovere tale compito e gestire direttamente un controllo sul circuito di carica.

Come visto già nel paragrafo precedente, questo tipo di integrato funziona già a tensioni basse di 1 V, di conseguenza ricopre pienamente la fascia di collasso del microcontrollore.

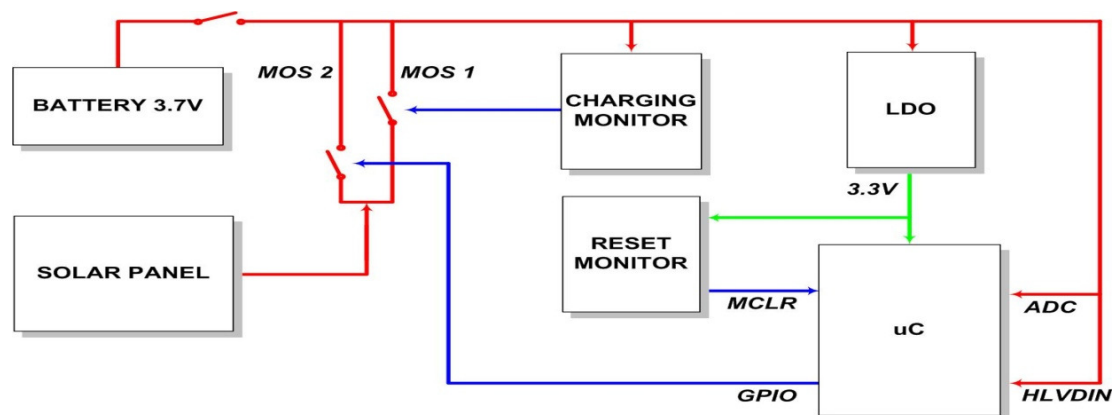


FIG. 5.15 - SCHEMA DI PRINCIPIO DI GESTIONE DELLA CARICA

Lo schema in figura 5.15 illustra una possibile soluzione per la gestione della carica. Il principio di funzionamento si basa in sostanza sulla possibilità di intervenire sul comando della carica, cioè sull'azionamento dei pannelli solari per mezzo di due comandi separati e indipendenti uno dall'altro. Il primo comando è fornito dal monitor di carica ed è costituito dall'integrato di supervisione del reset, mentre il secondo comando è controllato completamente dal modulo OBDH & HK.

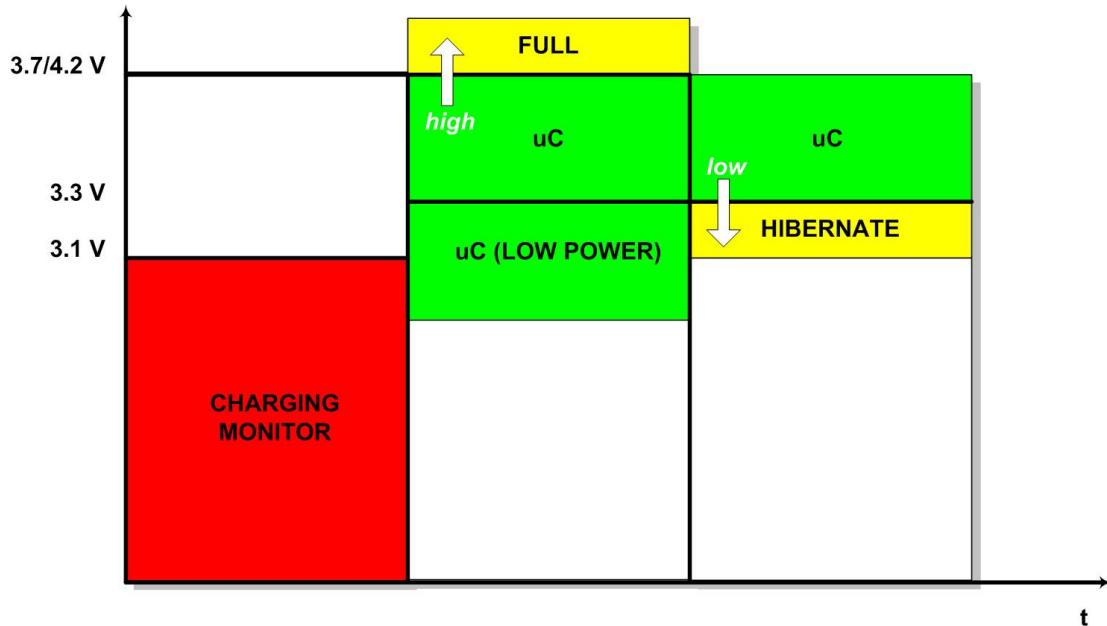


FIG. 5.16 - POSSIBILE CONTROLLO DELLA CARICA

Dal punto di vista concettuale è possibile suddividere il controllo della carica in varie modalità a seconda delle tensioni di lavoro.

Il monitor di carica prende il controllo quando la tensione di batteria è molto scarica e per tensioni inferiori di 3.1 V, mentre per tensioni superiori la carica è gestita interamente dal microcontrollore. La soglia di 3.1 V non è unica, ma è stata impostata a questo valore inferiore a 3.3 V ritenuto critico; la cosa importante da considerare è che essa sia superiore del valore di tensione in cui si attiva il microcontrollore, in modo da creare una finestra di isteresi tra quando si avvia il modulo OBDH & HK e quando si spegne il monitor hardware. La tensione di avvio del microcontrollore è legata al trip point del dispositivo ISL88011 e si ricorda che è pari a 2.3 V. All'avvio del modulo OBDH & HK, la carica della batteria è ancora minima, quindi può risultare utile far partire il microcontrollore in una modalità a basso consumo.

Una volta che il comando è passato al microcontrollore, è quest'ultimo a gestire e a controllarne l'andamento. Sono state fissate due soglie a tal riguardo, una che indica quando si raggiunge la carica massima (4.2 V) e una quando si raggiunge la tensione minima di funzionamento della scheda OBDH & HK (3.3 V).

La soglia alta è utile per interrompere la carica quando la batteria ha raggiunto il suo valore massimo evitando possibili danneggiamenti della stessa, mentre la soglia bassa permette di agire in tempo per mezzo di un algoritmo di ibernazione salvando lo stato del microcontrollore, prima di eventualmente perderne il controllo.

Il microcontrollore controlla le soglie per mezzo della periferica HLVD, High/Low Voltage Detect; quest'ultima è costituita da un circuito programmabile che permette di specificare

sia un trip point che un cambio di direzione da quest'ultimo. Qualora queste condizioni si verifichino il modulo imposta un flag di interrupt permettendo di gestire una routine dedicata. Quando il modulo HLVD è abilitato, esso si serve di un comparatore e di una tensione di riferimento generata internamente per adempiere alle sue funzioni.

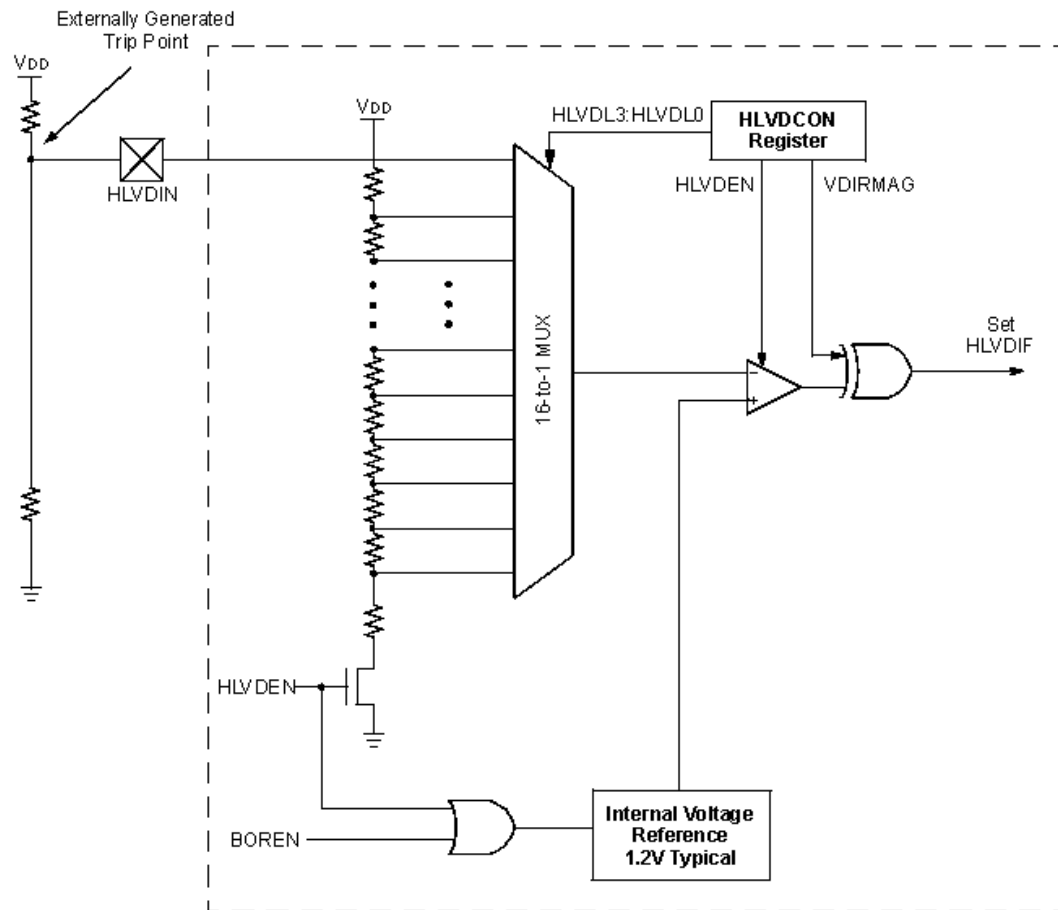


FIG. 5.17 - MODULO HIGH LOW VOLTAGE DETECTOR INTERNO AL PIC 18F8722

Come si nota in figura 5.17 il modulo è unico e pertanto risulta necessario agire via software per poter sensibilizzare il microcontrollore ad entrambe le soglie modificando sia il trip point per mezzo della rete resistiva sia il cambio di direzione per mezzo del registro di configurazione HLVDCON. Qualora la risoluzione della rete resistiva non è sufficiente o inadeguata si considera di tarare opportunamente il riferimento per mezzo del piedino esterno HLVDIN.

Come interruttori comandati per il controllo dei pannelli solari si utilizzano dei mosfet in modo da ottenere correnti di comando pressoché nulle.

Le caratteristiche che tali mosfet devono presentare sono:

- resistenza equivalente in conduzione più bassa possibile;
- tipologia di canale compatibile con la polarità di uscita del circuito supervisore;
- $V_{GS_TH_MAX}$ adeguata per garantire lo stato di conduzione;
- P_{D_MAX} sufficiente a dissipare la potenza data dal prodotto della corrente di carica dei pannelli con la tensione di caduta V_{DS} .

Essendo il monitor di carica attivo a livello logico basso quando la tensione si trova sotto soglia, il mosfet in questione deve essere a canale P in modo da portarsi in conduzione e collegare le celle solari alla batteria.

La $V_{GS_TH_MAX}$ è condizionata dal valore di tensione che il pannello solare riesce a fornire.

Le celle a disposizione forniscono 2.19 V massimi ciascuna; supponendo di collegarne due in serie la tensione di soglia del mosfet deve essere inferiore ad almeno 4.4 V circa.

La P_{D_MAX} dipende invece dalla corrente massima erogabile dalle celle solari, che risulta essere di 28 mA e dalla tensione V_{DS} che risulta essere di circa 0.7 V (differenza tra il valore di tensione fornito dai pannelli, 4.4 V, e il valore della tensione della carica, 3.7 V). Un valido componente in grado di soddisfare l'insieme di caratteristiche è l'IRLML6401 della International Rectifier.

Di seguito si riportano i grafici che illustrano le caratteristiche di maggior interesse.

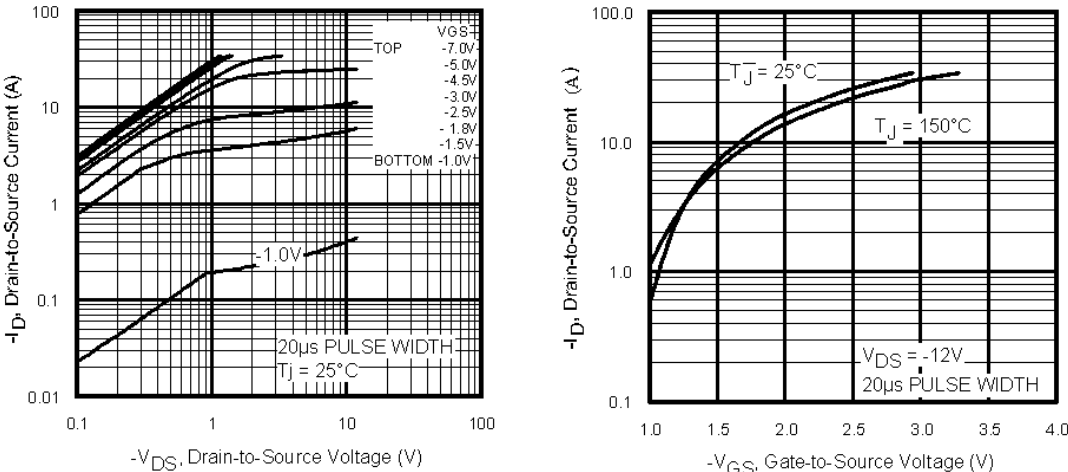


FIG. 5.18 - CARATTERISTICHE DEL MOSFET IRLML6401

<i>1 Sun, AM1.5G (100.0 mW/cm²) 25°C</i>	
$I_{sc} = 31 \text{ mA}$	$I_{mp} = 28 \text{ mA}$
$V_{oc} = 2.52 \text{ V}$	$V_{mp} = 2.19 \text{ V}$
$P_{mp} = 0.027 \text{ W/cm}^2$	Cff= 80 %
Efficiency= 27 ± 3% Absolute	Temp. Coeff. $V_{mp} = -6.2 \text{ mV/}^\circ\text{C}$

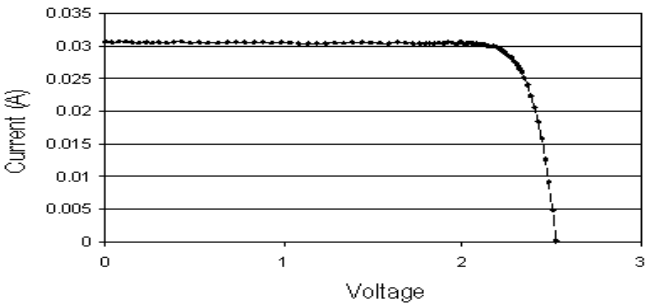


FIG. 5.19 - PARAMETRI DELLE CELLE SOLARI IN DOTAZIONE E CARATTERISTICA I-V

5.3.1 Modalità a basso consumo del microcontrollore

Per quanto riguarda la condizione di *Low Power* e il Power Management nelle diverse soglie viste precedentemente è utile descrivere la struttura interna del microcontrollore dedicata a questa parte. La famiglia del PIC18F8722 include caratteristiche che permettono alla sorgente di clock del dispositivo di commutare dall'oscillatore principale ad un clock alternativo. Quando una sorgente di clock viene attivata, le varie modalità di Power Management sono disponibili.

Essenzialmente ci sono tre sorgenti di clock: **l'oscillatore principale, l'oscillatore secondario e l'oscillatore interno.**

L'oscillatore primario è generalmente quello che comprende il clock ad alta frequenza, quello cioè dove l'esecuzione avviene a piena potenza.

L'oscillatore secondario è formato da tutte quelle sorgenti esterne non connesse ai pin OSC1 e OSC2 del microcontrollore. Come sorgente del clock secondario il PIC18LF8722 mette a disposizione l'oscillatore regolato dal Timer1. Spesso, un quarzo a 32.768 kHz viene connesso tra i piedini T1OSO e T2OSO.

L'oscillatore interno è presente nella struttura del microcontrollore e può essere utilizzato anche come sorgente di clock primario.

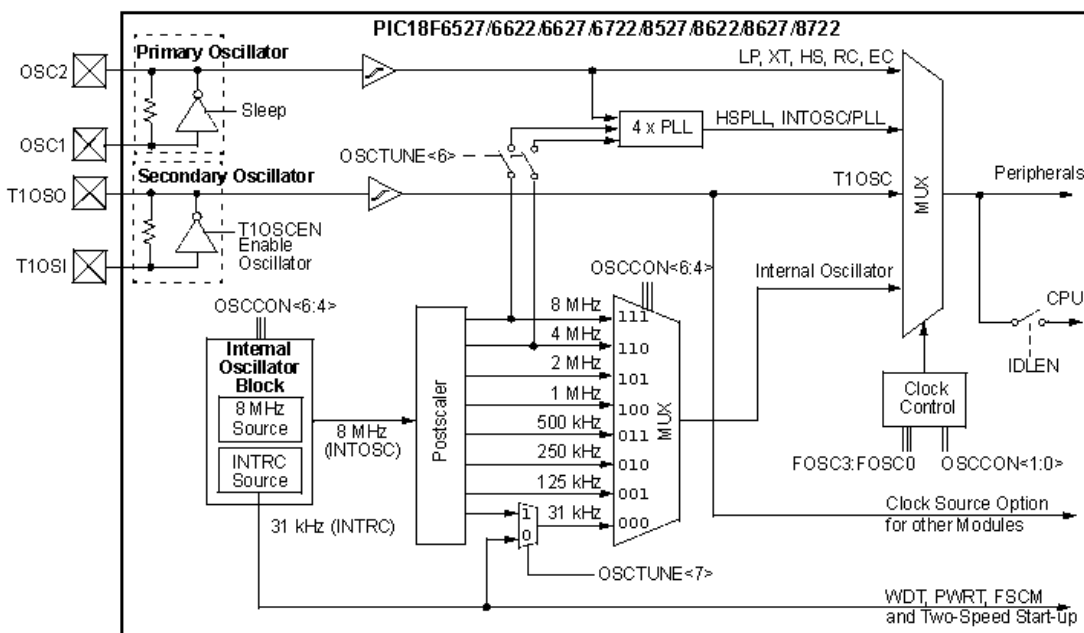


FIG. 5.20 - STRUTTURA INTERNA DELLE VARIE SORGENTI DI CLOCK DEL MICROCONTROLLORE

Le modalità di Power Management mettono a disposizione diverse opzioni per gestire il risparmio di potenza in applicazioni dove le risorse energetiche possono essere limitate.

Ci sono tre categorie di Power Management: **i Run modes, gli Idle modes e gli Sleep modes.** Queste categorie definiscono quali parti del dispositivo sono sincronizzate dal clock. I Run modes e l'Idle modes possono utilizzare qualsiasi delle tre sorgenti di clock, cioè la primaria, la secondaria o l'interna; la modalità Sleep non usa alcuna sorgente di clock.

Selezionare una modalità di Power Management richiede due decisioni e cioè se il clock agisce o meno sulla CPU e quale deve essere la sorgente del clock. Le combinazioni vengono riassunte nella tabella seguente.

Mode	OSCCON Bits		Module Clocking		Available Clock and Oscillator Source
	IDLEN<7> ⁽¹⁾	SCS<1:0>	CPU	Peripherals	
Sleep	0	N/A	Off	Off	None – All clocks are disabled
PRI_RUN	N/A	00	Clocked	Clocked	Primary – LP, XT, HS, HSPLL, RC, EC and Internal Oscillator Block ⁽²⁾ . This is the normal full power execution mode.
SEC_RUN	N/A	01	Clocked	Clocked	Secondary – Timer1 Oscillator
RC_RUN	N/A	1x	Clocked	Clocked	Internal Oscillator Block ⁽²⁾
PRI_IDLE	1	00	Off	Clocked	Primary – LP, XT, HS, HSPLL, RC, EC
SEC_IDLE	1	01	Off	Clocked	Secondary – Timer1 Oscillator
RC_IDLE	1	1x	Off	Clocked	Internal Oscillator Block ⁽²⁾

FIG. 5.21 - VARIE MODALITA' DI POWER MANAGEMENT

Il bit IDLEN controlla se la CPU viene sincronizzata o meno dal clock mentre i bit SCS<1:0> selezionano la sorgente del clock.

Per gestire le modalità di Power Management e commutare da una all'altra si carica opportunamente il registro di controllo OSCCON

R/W-0	R/W-1	R/W-0	R/W-0	R ⁽¹⁾	R-0	R/W-0	R/W-0
IDLEN	IRCF2	IRCF1	IRCF0	OSTS	IOFS	SCS1	SCS0
bit 7							bit 0

bit 7 **IDLEN:** Idle Enable bit

- 1 = Device enters an Idle mode when a SLEEP instruction is executed
- 0 = Device enters Sleep mode when a SLEEP instruction is executed

bit 6-4 **IRCF2:IRCF0:** Internal Oscillator Frequency Select bits⁽⁵⁾

- 111 = 8 MHz (INTOSC drives clock directly)
- 110 = 4 MHz
- 101 = 2 MHz
- 100 = 1 MHz⁽³⁾
- 011 = 500 kHz
- 010 = 250 kHz
- 001 = 125 kHz
- 000 = 31 kHz (from either INTOSC/256 or INTRC directly)⁽²⁾

bit 3 **OSTS:** Oscillator Start-up Time-out Status bit⁽¹⁾

- 1 = Oscillator Start-up Timer (OST) time-out has expired; primary oscillator is running
- 0 = Oscillator Start-up Timer (OST) time-out is running; primary oscillator is not ready

bit 2 **IOFS:** INTOSC Frequency Stable bit

- 1 = INTOSC frequency is stable
- 0 = INTOSC frequency is not stable

bit 1-0 **SCS1:SCS0:** System Clock Select bits⁽⁴⁾

- 1x = Internal oscillator block
- 01 = Secondary (Timer1) oscillator
- 00 = Primary oscillator

FIG. 5.22 - REGISTRO DI CONTROLLO DEL POWER MANAGEMENT

Una semplice soluzione per la gestione del Power Management nella fasi di *Low Power* può essere la commutazione tra la modalità PRI_RUN e la SEC RUN e viceversa, dove la modalità SEC_RUN rappresenta la condizione di Low Power.

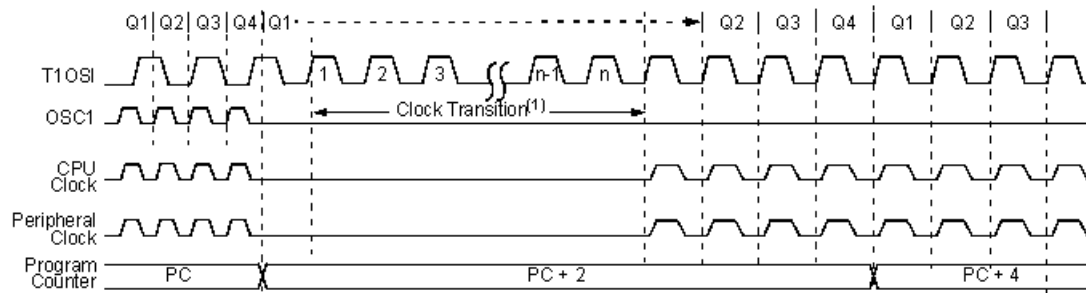
Ora vengono brevemente descritte queste due modalità e successivamente vengono mostrati dei diagrammi che illustrano la commutazione da un clock ad un altro.

La PRI_RUN è la modalità normale a piena potenza di esecuzione del microcontrollore ed è la modalità di default dopo un reset.

La SEC_RUN è la modalità in cui sia la CPU sia le periferiche vengono sincronizzate dal clock proveniente dal Timer1. In tal modo si passa ad una modalità a basso consumo avendo l'elevata accuratezza e stabilità di una sorgente a quarzo.

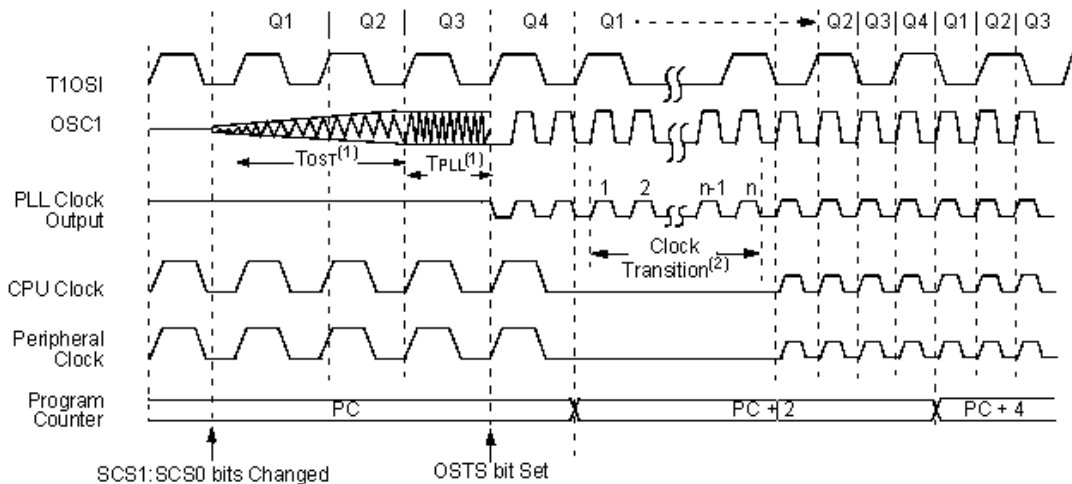
Si entra in questa modalità agendo sui bit SCS1 e SCS2; dopo tale commutazione il Timer1 viene selezionato e l'oscillatore primario viene spento.

Nella transizione dalla modalità SEC_RUN alla modalità PRI_RUN le periferiche e la CPU continuano ad essere sincronizzate dal clock del Timer1 mentre il clock primario viene avviato. Quando quest'ultimo si stabilizza avviene la commutazione dal Timer1 al clock principale.



Note 1: Clock transition typically occurs within 2-4 T_{osc} .

FIG. 5.23 - COMMUTAZIONE DALL'OSCILLATORE PRIMARIO ALL'OSCILLATORE SECONDARIO TIMER1



Note 1: $T_{ost} = 1024 T_{osc}$; $T_{PLL} = 2 \text{ ms}$ (approx). These intervals are not shown to scale.

2: Clock transition typically occurs within 2-4 T_{osc} .

FIG. 5.24 - COMMUTAZIONE DAL TIMER1 ALL'OSCILLATORE PRIMARIO

6 Driver software per le periferiche del microcontrollore

Nel seguente capitolo viene analizzata la struttura delle periferiche messe a disposizione dal microcontrollore e i software a basso livello creati per la gestione delle stesse. Per tal motivo, queste parti del software possono essere definiti come i driver necessari a instaurare tutte le comunicazioni nel modulo OBDH & HK.

Successivamente allo sviluppo dei driver sono state effettuate delle prove sperimentali in laboratorio, verificando per mezzo della strumentazione disponibile la correttezza di tutti i segnali digitali.

In particolare, in questo capitolo, verranno descritti i moduli e i driver di gestione della periferica EMI, MSSP1, EUSART1, EUSART2 e ADC.

6.1 Scrittura e lettura della memoria SRAM di telemetria

Assodato il fatto che la periferica EMI crea un'espansione del bus della memoria programma, non sorprende il fatto che le modalità di accesso della SRAM esterna sono molto simili a quelle della flash interna.

Ci sono due registri principali che permettono l'accesso al bus della memoria programma: si tratta dei due Special Function Register TABLAT (Table Latch) e TABPTR (Table Pointer).

Dato che lo spazio della memoria programma è largo 16 bit mentre lo spazio della memoria dei dati è largo 8 bit, ci si serve del registro TABLAT a 8 bit per effettuare lo scambio delle informazioni tra di essi.

Il registro TABPTR serve per il puntamento del dato sul bus della memoria programma e a sua volta è composto da tre Special Function Register a 8 bit ciascuno denominati TBLPTRU (Table Pointer Upper), TBLPTRH (Table Pointer High) e TBLPTRL (Table Pointer Low). Tramite il TBLPTR si può indirizzare il completo spazio di memoria di 2 Mbyte e si può accedere sia alla memoria interna che a quella esterna indiscriminatamente.

Questo registro è a 22 bit e contiene un bit supplementare oltre i 21 bit di indirizzamento; il bit supplementare serve solo per applicazioni della memoria interna e definisce, tramite il suo stato, se si intende accedere al *device ID*, lo *user ID* e i *configuration bit* del microcontrollore.

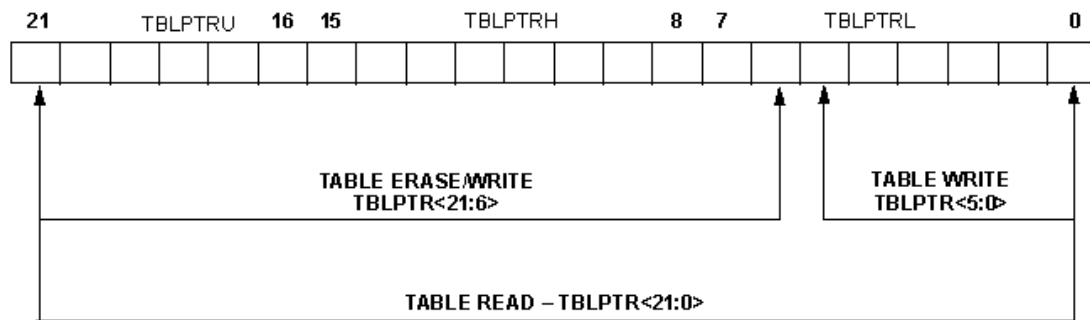


FIG. 6.1 - ORGANIZZAZIONE DEL REGISTRO DI INDIRIZZAMENTO

Le istruzioni di scrittura e lettura della memoria interna ed esterna fanno parte dell'Instruction Set Architecture e sono dedicate esclusivamente a questa funzione.

Per la lettura dei dati dal bus della memoria programma è stata implementata l'istruzione *TBLRD**. Dopo aver posto l'indirizzo nell'opportuno puntatore TBLPTR, eseguendo l'istruzione *TBLRD**, il dato della determinata locazione di memoria viene opportunamente prelevato e salvato nel registro TABLAT. Il circuito interno di controllo provvederà automaticamente a salvare il byte più significativo o meno significativo della parola a 16 bit basandosi sul bit meno significativo del TBLPTR.

Per la scrittura dei dati sul bus della memoria programma invece è disponibile l'istruzione *TBLWT**. In questo caso dopo aver sempre eseguito l'indirizzamento con il TBLPTR, si passa a scrivere il dato nel registro latch TABLAT e infine si esegue l'istruzione *TBLWT**. Le due istruzioni base per l'accesso alla memoria programma possono essere corredate dalle opzioni denominate "+" e "-" in modo da incrementare o decrementare automaticamente il TBLPTR. In questo modo non occorre nuovamente impostare l'indirizzo se si vuole accedere ad una locazione precedente o successiva a quella richiamata.

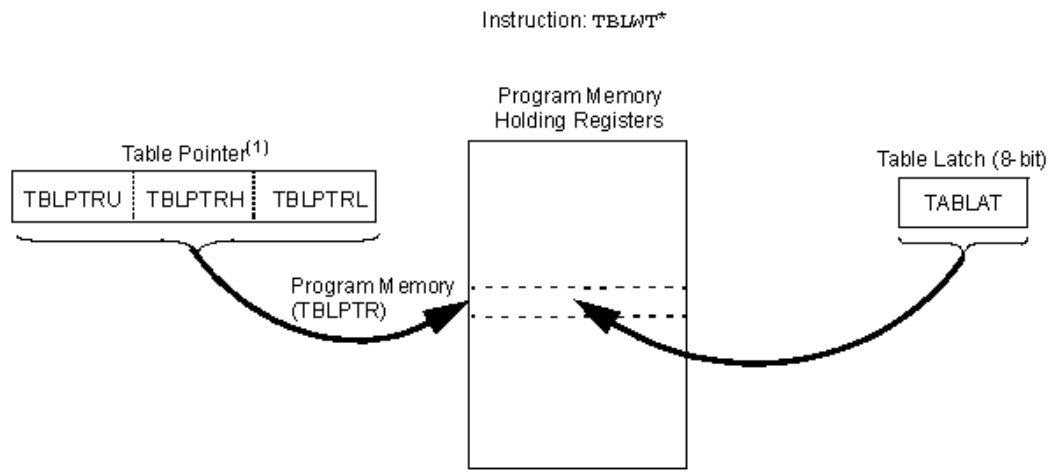


FIG. 6.2 - MODALITA' DI SCRITTURA PER MEZZO DEI REGISTRI TBLPTR E TBLWT

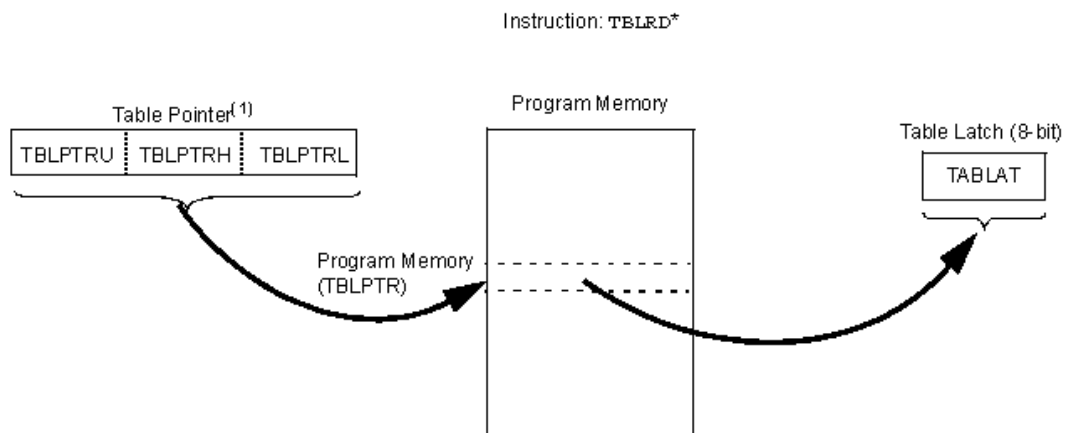


FIG. 6.3 - MODALITA' DI LETTURA PER MEZZO DEI REGISTRI TBLPTR E TBLRD

Prima di poter usufruire della periferica EMI è necessario configurare opportunamente ulteriori due registri. Essi sono il MEMCON e il CONFIG3L.

MEMCON REGISTER

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
bit7							bit0

FIG. 6.4 - CONTENUTO DEL REGISTRO MEMCOM

- **EBDIS:** impostando questo bit a 0 tutti i pin dedicati alla periferica EMI del microcontrollore vengono impiegati esclusivamente per la gestione della memoria SRAM e non sono disponibili per alcun altro compito. Impostandolo a 1, gli stessi pin vengono controllati dalla periferica EMI solamente quando si accede alla memoria SRAM, mentre in tutti gli altri casi possono venir utilizzati per le altre funzioni per cui sono stati progettati.
- **WAIT<1:0>:** generalmente le memorie esterne sono più lente della CPU del microcontrollore, di conseguenza possono essere introdotti degli stati di pausa nelle operazioni di TBLWT* e TBLRD* in modo da rendere il protocollo di comunicazione più lento.
- **WM<1:0>:** con questi due bit si selezionano le varie modalità di scrittura permesse dal bus e analizzate nel capitolo 4.

Dato che il bus EMI è stato dedicato esclusivamente alla memoria SRAM, la configurazione del bit EBDIS non è essenziale al funzionamento di questa parte del progetto.

Per quanto riguarda gli stati di attesa WAIT, essi non vengono aggiunti al protocollo in quanto il clock effettivo di istruzione di 1 MHz è sufficiente a rispettare tutte le temporizzazioni della memoria. I bit vengono di conseguenza impostati entrambi a 1.

La modalità di scrittura del bus EMI è la Byte Select Mode, di conseguenza i bit di WM vengono impostati a 0 e 1.

CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)⁽¹⁾

R/P-1	R/P-1	R/P-1	R/P-1	U-0	U-0	R/P-1	R/P-1
WAIT	BW	ABW1	ABW0	—	—	PM1	PM0
bit 7							bit 0

FIG. 6.5 - CONTENUTO DEL REGISTRO DI CONFIGURAZIONE CONFIG3L

- **PM:** con questi bit si selezionano le modalità di operazione del microcontrollore e permettono di usufruire dell'intero spazio di memoria di 2 Mbyte;
- **ABW:** è possibile variare la lunghezza dell'indirizzo e agire sulle rispettive linee disponibili;
- **BW:** si seleziona se utilizzare il bus in modalità a 8 bit o a 16 bit;
- **WAIT:** questa è un'ulteriore configurazione per gli stati di attesa e permette di escluderli del tutto o seguire la configurazione del registro precedente.

Gli stati di WAIT vengono disabilitati per le stesse motivazioni giustificate nella configurazione del registro precedente. Il bit BW viene posto a 1 in modo da avere a

disposizione un bus a 16 bit. I bit ABW vengono posti entrambi a 1 per poter indirizzare completamente i 2 Mbyte di memoria SRAM. Infine la modalità con cui far operare il microcontrollore è la Extended Microcontroller Mode per le motivazioni descritte nei capitoli precedenti, di conseguenza entrambi i bit vanno posti a 0.

Di seguito si riportano i diagrammi di flusso dei driver per la scrittura e la lettura della memoria SRAM esterna. Il codice implementato, anche se formulato in linguaggio C, è strutturato a basso livello, in modo da comprendere le operazioni descritte nella teoria delle periferiche. Queste procedure fanno uso infatti delle istruzioni TBLWT* e TBLRD* richiamando del codice Assembler all'interno del codice C.

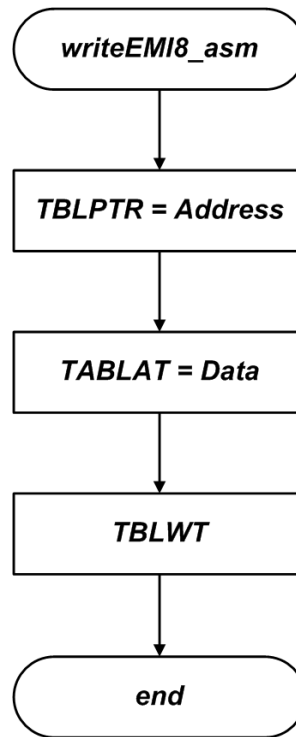


FIG. 6.6 - DIAGRAMMA DI FLUSSO PER LA SCRITTURA SUL BUS EMI

La routine di scrittura copia il valore dell'indirizzo nel registro di indirizzamento TBLPTR, successivamente carica il dato nel latch TABLAT e poi richiama l'istruzione Assembler di scrittura TBLWT.

Il trasferimento del dato, e quindi l'accesso al bus, comincia solamente all'esecuzione dell'istruzione TBLWT.

```

//-----
// Write data into RAM (Assembler instructions)
// INPUT: unsigned char Data, data to be written into memory
//         unsigned short long , address in RAM
//-----

void writeEMI8_asm ( unsigned char Data, unsigned short long Address )
{
    TBLPTR = Address;           // Load address
                                // into Table Pointer
    TABLAT = Data;              // Load data into Table Latch
    _asm TBLWT _endasm          // Write data
}
  
```

FIG. 6.7 - PROCEDURA DI SCRITTURA SUL BUS EMI

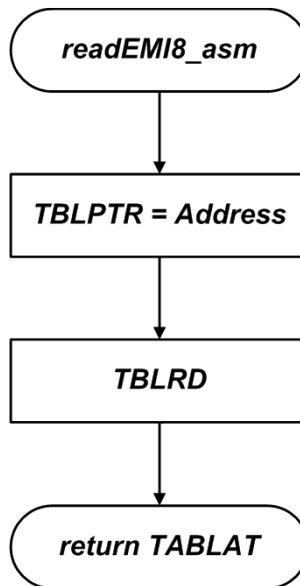


FIG. 6.8 - DIAGRAMMA DI FLUSSO PER LA LETTURA SUL BUS EMI

La routine di lettura, come quella di scrittura, seleziona per primo l'indirizzo in cui scrivere il dato, successivamente richiama l'istruzione Assembler TBLRD; a questo punto il dato viene prelevato dalla specifica locazione e trasferita nel registro di Latch TABLAT;

```

//-----
// Read data into RAM (Assembler instructions)
// INPUT: unsigned short long, address in RAM
// OUTPUT: unsigned char, data read from RAM
//-----

unsigned char readEMI8_asm (unsigned short long Address)
{
    TBLPTR = Address;           // Load address
                                // into Table Pointer
    _asm TBLRD _endasm          // Read data & save
                                // into Table Latch
    return TABLAT;              // Return data
}
  
```

FIG. 6.9 - PROCEDURA DI LETTURA DAL BUS EMI

Vengono riportate di seguito le prove sperimentali di laboratorio effettuate con il debugger del microcontrollore e un oscilloscopio digitale.

La fig. 6.10 rappresenta la scrittura della memoria SRAM all'indirizzo pari 0x20006 con il dato 0xAA. Alle sonde D0 a D7 sono state collegate le linee AD<7:0>, alla sonda D8 la linea $\overline{WRH}$, alla sonda D9 la linea $\overline{UB}$, alla sonda D10 la linea $\overline{LB}$, alla sonda D11 la linea $\overline{CE}$, alla sonda D12 la linea $\overline{OE}$ e infine alla sonda D13 la linea ALE .

Si nota il segnale ALE (D13) che si porta alto per congelare l'indirizzo, i segnali $\overline{UB}$ (D9) e $\overline{LB}$ (D10) rispettivamente a livello 1 e 0 che indicano la selezione di un indirizzo pari. Dopo che il $\overline{WRH}$ (D8) si porta basso, il dato 0xAA (0b10101010) viene posto sulle linee AD<7:0> (D0 – D7) e viene campionato sul fronte di salita dello stesso controllo.

Il segnale $\overline{OE}$ (D12) rimane per tutto il ciclo di scrittura a livello 1 disabilitando le uscite, mentre la linea $\overline{CE}$ (D11) resta bassa per garantire l'abilitazione della memoria.

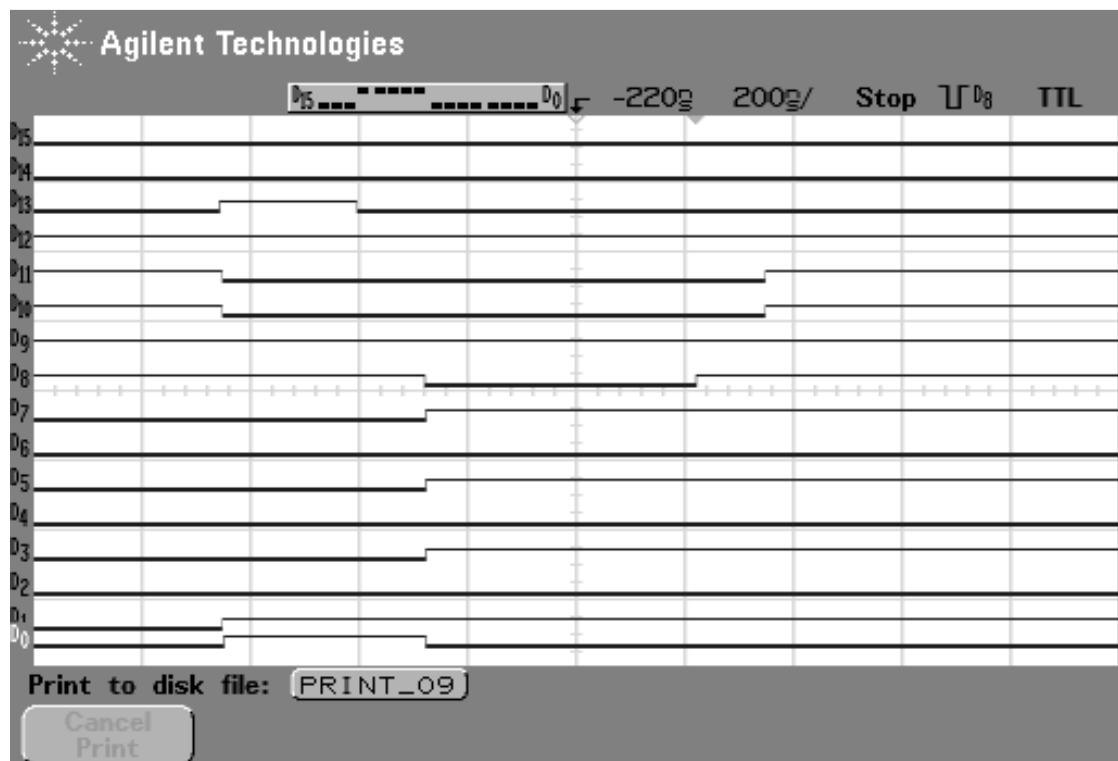


FIG. 6.10 - SCRITTURA DEL DATO 0xAA ALLA LOCAZIONE PARI 0x20006

Questa immagine rappresenta la scrittura della memoria SRAM all'indirizzo dispari 0x20007 con il dato 0xAA. Le sonde sono collegate nella stessa modalità di prima. Le considerazioni fatte per la figura precedente equivalgono anche per questo caso; l'unica differenza è che trattando un indirizzo dispari, gli stati logici dei controlli della selezione del byte risultano invertiti, cioè si constata che il segnale $\overline{UB}$ (D9) è a livello 0 mentre il segnale $\overline{LB}$ (D10) a livello 1.

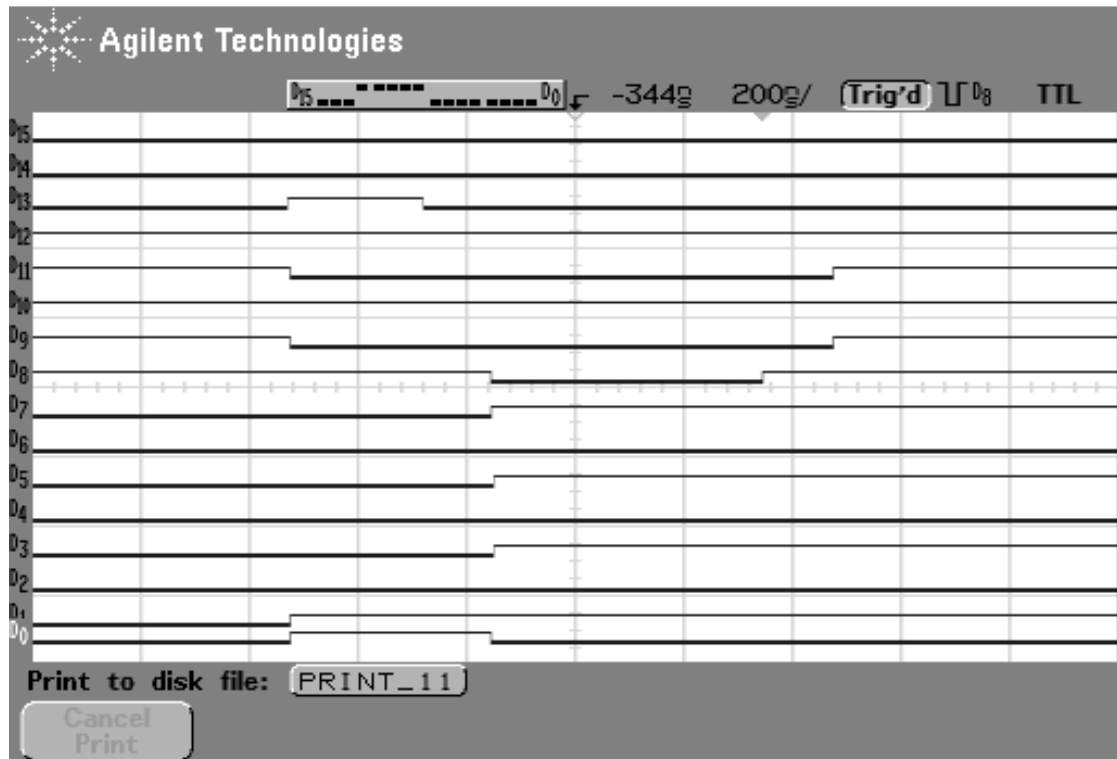


FIG. 6.11 - SCRITTURA DEL DATO 0xAA ALLA LOCAZIONE DISPARI 0x20007

Infine si mostra un'immagine della lettura dalla memoria SRAM all'indirizzo pari 0x20000 del dato 0x01. Le sonde sono collegate sempre nella stessa modalità.

Si nota che la linea di $\overline{WRH}$ (D8) resta permanentemente a livello 1 per disabilitare la scrittura. I dati si rendono disponibili invece dopo il fronte di salita del segnale di $\overline{OE}$ (D12) e vengono campionati sul fronte opposto dello stesso.

Entrambe le linee $\overline{UB}$ (D9) e $\overline{LB}$ (D10) sono a livello 0 per trasferire l'intero dato a 16 bit sul bus. Si ricorda infatti che il byte d'interesse è selezionato dal circuito di TABLAT in base al bit meno significativo del TBLPTR.

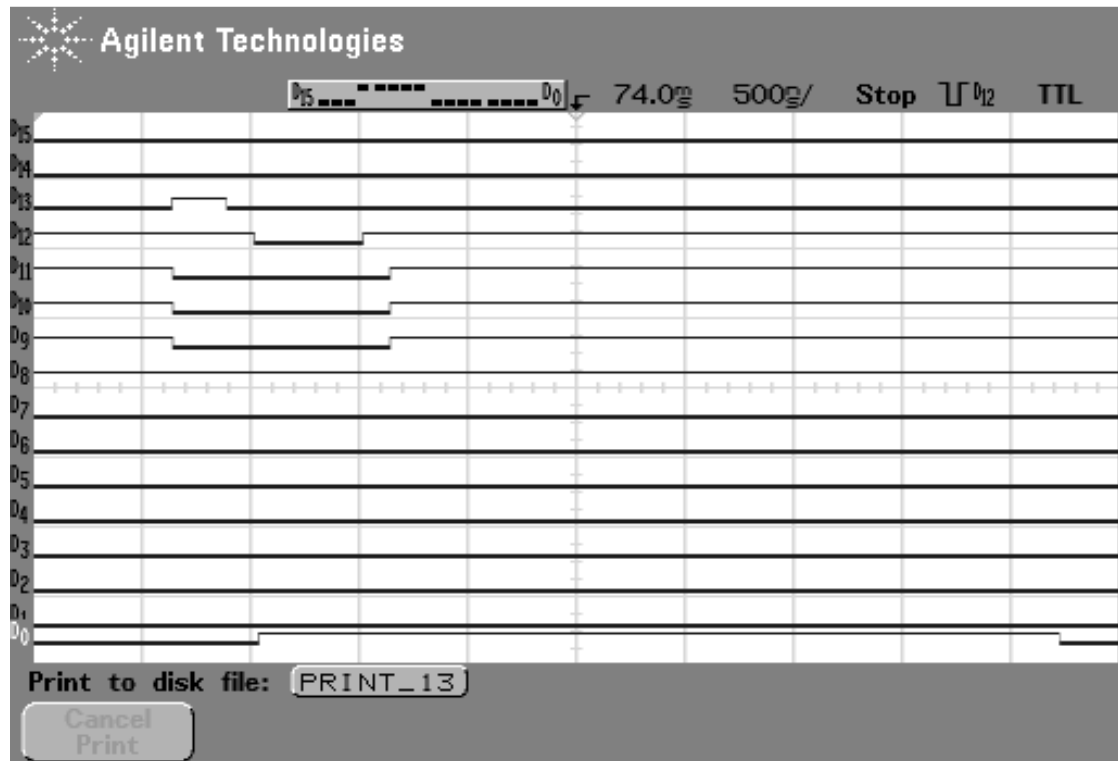


FIG. 6.12 - LETTURA DEL DATO 0x01 ALLA LOCAZIONE PARI 0x20000

6.2 Scrittura e lettura sul bus SPI

Si descrive ora il funzionamento della periferica MSSP, Master Synchronous Serial Port, utilizzata per la comunicazione SPI.

La MSSP è un'interfaccia seriale utile per lo scambio di dati con altre periferiche o altri microcontrollori e opera in questi modi:

- Serial Peripheral Interface (SPI);
- Inter – Integrated Circuit (IIC);
 - Full Master Mode;
 - Slave mode.

Ogni membro della famiglia PIC18LF8722 dispone di due periferiche seriali denominate rispettivamente MSSP1 e MSSP2 completamente indipendenti tra loro.

La modalità SPI permette ad un dato di 8 bit di essere trasmesso o ricevuto in maniera simultanea attraverso un segnale di sincronizzazione.

Per consentire la comunicazione tipicamente sono usati tre pin:

- Serial Data Out (SDOx);
- Serial Data In (SDIx);
- Serial Clock (SCKx).

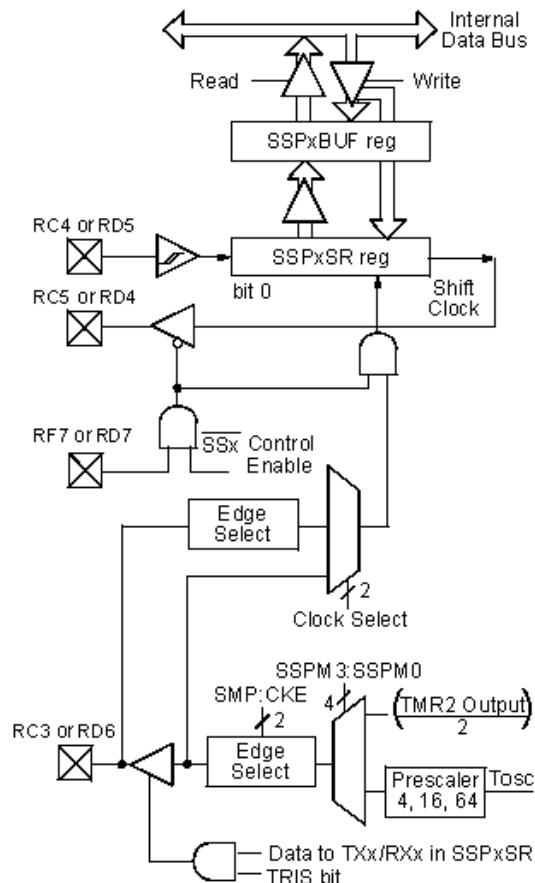


FIG. 6.13 - STRUTTURA INTERNA DELLA PERIFERICA MSSP

Ogni modulo MSSP possiede 4 registri per la configurazione e la gestione del protocollo SPI. Essi sono:

- MSSP Control Register 1 (SSPxCON1);
- MSSP Status Register (SSPxSTAT);
- Serial Receive/Transmit Buffer Register (SSPxBUF);
- MSSP Shift Register (SSPxSR).

I registri SSPxCON1 e SSPxSTAT sono registri di controllo e di stato della comunicazione SPI, il registro SSPxBUF è il registro di buffer con il quale i dati vengono scambiati all'interno della memoria del microcontrollore e il registro SSPxSR è lo shift register con il quale invece i singoli bit vengono inviati e ricevuti.

Lo shift register non è direttamente accessibile via software e lo si può modificare agendo solo sul registro di buffer SSPxBUF.

In ricezione i registri SSPxSR e SSPxBUF creano assieme un ricevitore a doppio buffer. Quando il SSPxSR riceve un byte, esso viene trasferito al SSPxBUF, il flag di interrupt SSPxIF si attiva assieme al bit di Buffer Full BF.

Dopo la ricezione dei dati, il registro SSPxBUF deve essere letto prima che il successivo dato da trasmettere venga scritto nuovamente sullo stesso.

In trasmissione non è presente il doppio buffer e una scrittura sul SSPxBUF causa una scrittura contemporanea su quest'ultimo e sullo shift register SSPxSR iniziando immediatamente l'invio dei dati.

Qualsiasi scrittura nel registro SSPxBUF durante una trasmissione o una ricezione viene completamente ignorata e il bit WCOL Write Collision Detect viene attivato.

Quando si inizializza l'SPI ci sono diverse opzioni che devono essere specificate; i vari bit di controllo permettono le seguenti impostazioni:

- Master mode (SCKx è l'uscita del clock);
- Clock Polarity (stato Idle del clock SCKx);
- Data Input Sample Phase (modalità di campionamento del dato in ingresso);
- Clock Edge (dato in uscita sul fronte di salita o discesa del clock);
- Clock Rate (frequenza di comunicazione).

Per abilitare la porta seriale il bit SSPEN deve essere posto a livello logico 1; in tal modo si dedicano le linee SDIx, SDOx, SCKx alla porta seriale. Mentre l'ingresso SDI è automaticamente gestito dalla periferica, i pin SDOx e SCKx devono avere i relativi buffer configurati come uscite digitali.

Per quanto riguarda il clock del protocollo SPI, esso è configurabile via software in uno dei seguenti modi:

- $F_{osc} / 4 (T_{CY})$;
- $F_{osc} / 16 (4T_{CY})$;
- $F_{osc} / 64 (16T_{CY})$;
- *Timer2 Output / 2.*

Nelle figure 6.14 e 6.15 vengono riportati i contenuti dei registri di configurazione della periferica MSSP.

SSPxSTAT: MSSPx STATUS REGISTER (SPI™ MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/A	P	S	R/W	UA	BF
bit 7							bit 0

FIG. 6.14 - REGISTRO DI STATO DELLA PERIFERICA MSSP

Dato che il modulo MSSP può gestire sia lo standard SPI che lo standard I^2C , i registri di configurazione condividono caratteristiche di entrambi i protocolli. Nel caso della comunicazione SPI i bit interessati sono il bit 0, il bit 6 e il bit 7.

- **SMP:** con questo bit si decide se campionare i bit in arrivo sulla linea SDI in corrispondenza dell'inizio o del mezzo del contemporaneo bit in uscita in quel momento sulla linea SDO;
- **CKE:** tramite questa opzione si seleziona se la trasmissione del dato in uscita avviene dallo stato attivo del clock a quello Idle oppure viceversa. Questo bit assieme al bit CKP del registro SSPxCON1 determina 4 diversi protocolli SPI;
- **BF:** effettuando un polling su questo bit è possibile testare se lo scambio di dati tra master e slave è concluso.

SSPxCON1: MSSPx CONTROL REGISTER 1 (SPI™ MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

FIG. 6.15 - REGISTRO DI CONTROLLO DELLA PERIFERICA MSSP

- **SSPM:** con questi bit si sceglie la modalità master o slave della periferica e l'eventuale clock con cui sincronizzare i dati;
- **CKP:** si seleziona lo stato inattivo del clock; in sostanza nella condizione in cui il modulo non operi, si determina se la linea del clock SCKx sia in uno stato alto o basso. Come descritto per il registro precedente, con questa configurazione si determinano le varie modalità di scambio dati;
- **SSPEN:** dal momento che sia attiva questo bit, la periferica MSSP viene abilitata e le rispettive linee vengono dedicate alla comunicazione SPI;
- **SSPOV:** se il registro di buffer SSPBUF contiene un dato e non viene salvato in un registro prima che venga iniziata un'altra trasmissione, allora viene attivato lo stato di overflow;
- **WCOL:** questo bit viene attivato se si scrive nel buffer SSPxBUF quando è in corso uno scambio una comunicazione.

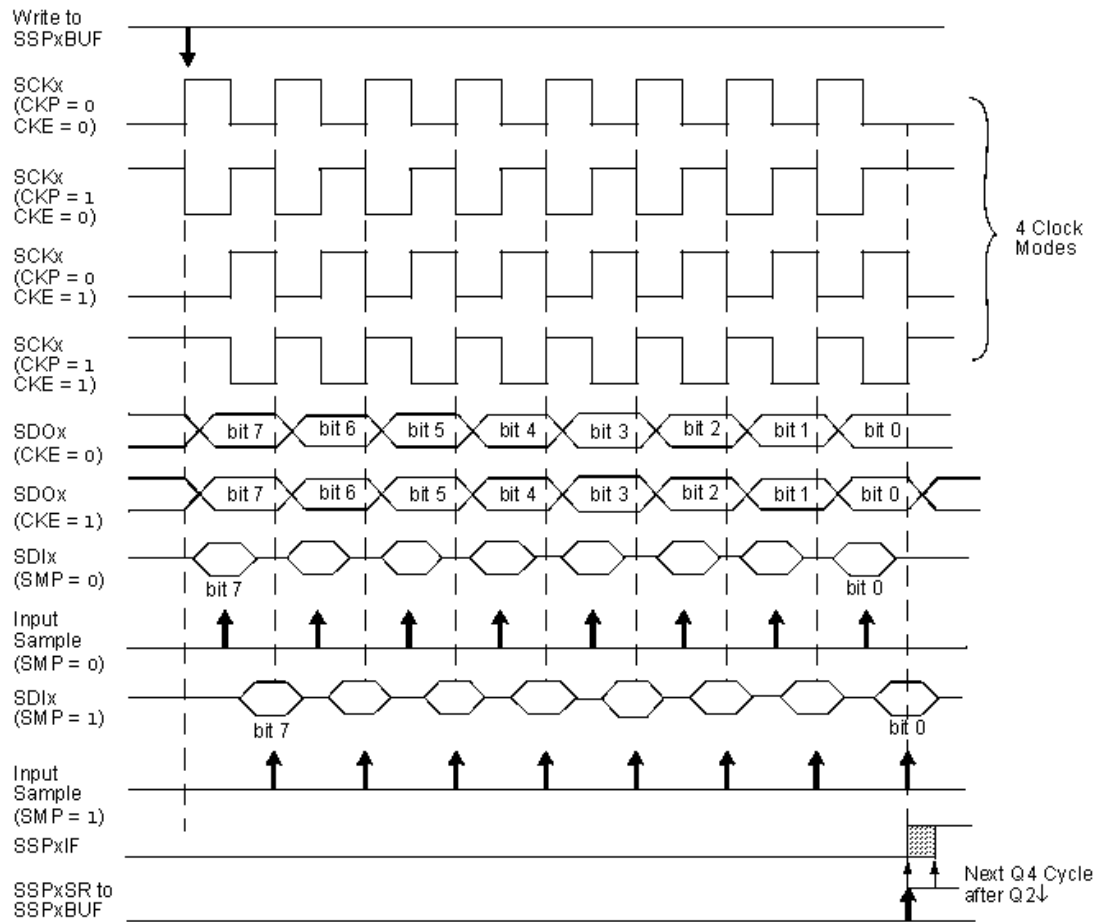


FIG. 6.16 - MODALITA' DI FUNZIONAMENTO DEL PROTOCOLLO SPI

Dal diagramma delle temporizzazioni di figura 6.16 si nota come il flag di interrupt si porta alto quando lo scambio di dati tra il microcontrollore e il dispositivo slave è completato. La stessa figura 6.16 illustra le quattro modalità di funzionamento del protocollo SPI. Esse si differenziano sostanzialmente per lo stato attivo del clock e per il fronte del clock con il quale si inviano e si ricevono i dati.

Le modalità vengono identificate nella tabella di figura 6.17.

Standard SPI™ Mode Terminology	Control Bits State	
	CKP	CKE
0, 0	0	1
0, 1	0	0
1, 0	1	1
1, 1	1	0

FIG. 6.17 - MODI DI FUNZIONAMENTO DELLA PERIFERICA SPI

Si riporta il diagramma di flusso del driver di gestione della periferica MSSP1.

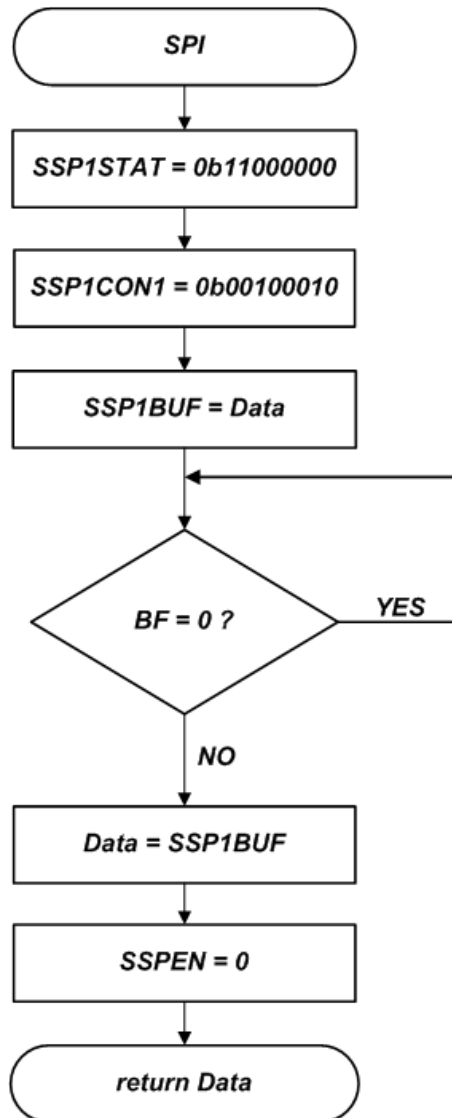


FIG. 6.18 - DIAGRAMMA DI FLUSSO DELLA COMUNICAZIONE SPI

Le prime operazioni eseguite dal driver sono quelle di impostazione dei registri SSP1STAT e SSP1CON1 al fine di configurare le varie opzioni: si sceglie la modalità SPI 0,0 con livello Idle del clock pari a 0 e shift dei singoli bit in uscita ad ogni fronte di discesa; i bit in ingresso invece vengono campionati ancora sul fronte di discesa. Inoltre per mezzo del registro SSP1BUF si attiva la periferica MSSP1. A questo punto il dato viene caricato nel registro di buffer SSP1BUF consentendo l'inizio della comunicazione tra master e slave. Il test continuo sul bit BF consente di sapere quando è stato scambiato l'intero byte in modo da poter leggere sempre sullo stesso registro di buffer SSP1BUF il dato ricevuto dal dispositivo slave. Il buffer viene salvato nella variabile Data, si spegne la periferica MSSP1 e si ritorna dalla funzione.

```

//-----
// Exchange data in SPI
// OUTPUT: unsigned char, data to be written into bus
// INPUT: unsigned char, data to be read from bus
//-----

unsigned char SPI (unsigned char Data)
{
    SSP1STAT = 0b11000000;           // SMP=0, CKE=1
    SSP1CON1 = 0b00100010;           // MSSP ON, CKP=0,
                                     // SCLK= 62,5 KHz
    SSP1BUF = Data;                   // Transmit data

    while ( SSP1STATbits.BF == 0 );  // Wait for end of exchange

    Data = SSP1BUF;                   // Save received data
    SSP1CON1bits.SSPEN = 0;           // Turn off MSSP

    return Data;                      // Return received data
}

```

FIG. 6.19 - CODICE C PER LO SCAMBIO DI UN BYTE MEDIANTE IL PROTOCOLLO SPI

Si riportano ora alcuni test effettuati in laboratorio per constatare l'effettivo funzionamento della periferica MSSP e dei relativi buffer di isolamento.

La prima verifica è eseguita con il prototipo della scheda OBDH & HK impiegata come dispositivo master e un'altra scheda di prova impiegata come dispositivo slave; quest'ultima è costituita a sua volta da un microcontrollore Microchip 18LF452.

La scheda OBDH & HK viene programmata con del codice C in modo da inviare il byte 0xCC, cioè 0b11001100 in binario, mentre il microcontrollore slave contemporaneamente risponde con il dato 0xAA, cioè 0b10101010 in binario.

La frequenza di clock selezionata è di Fosc/64, cioè 62.5 kHz.

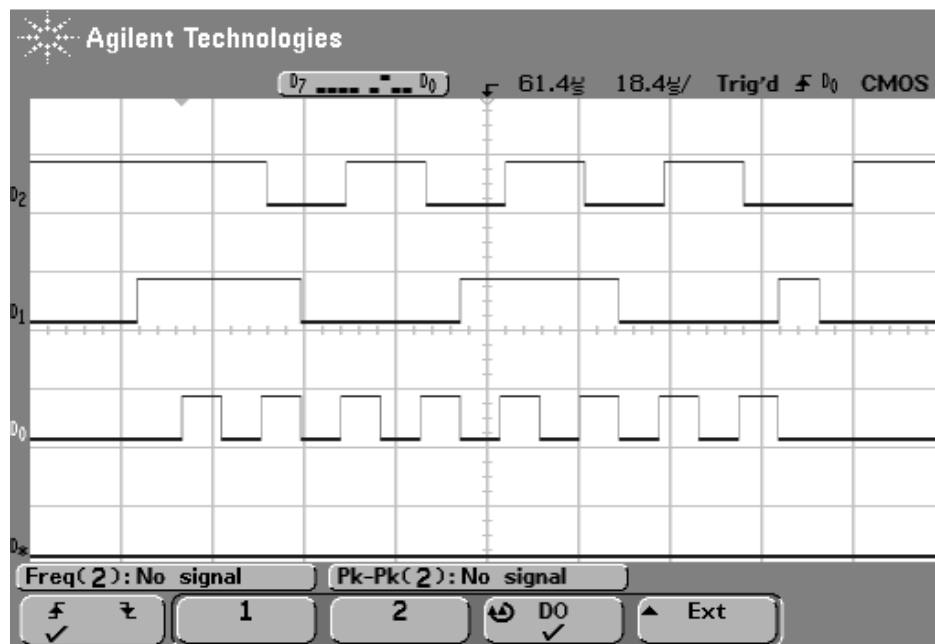


FIG. 6.20 - SCAMBIO DI DATI SPI TRA DISPOSITIVO MASTER E SLAVE

Nella figura 6.20, la sonda denominata D0 rileva il segnale di clock SCK, la sonda D1 rappresenta il dato in uscita dal pin SDO e infine la sonda D2 campiona i dati in ingresso al pin SDI. Questa prova viene eseguita quando ancora non sono stati montati sul prototipo i buffer di isolamento e ha come unico scopo la verifica del funzionamento della periferica MSSP e del relativo driver software implementato.

La seconda prova invece viene eseguita constatando l'effettivo funzionamento dei buffer. Viene verificato che lo scambio dei dati abbia buon fine e che la qualità dei segnali a monte e a valle dei buffer sia soddisfacente, il tutto a varie frequenza di lavoro. Nelle figure si riporta solo il segnale di clock.

La sonda D0 fa capo all'analizzatore degli stati logici e rappresenta il clock all'uscita della periferica MSSP, la sonda 1 è il primo canale dell'oscilloscopio e verifica la qualità del clock a monte del buffer, cioè direttamente sul pin SDO, e infine la sonda 2 è il secondo canale dell'oscilloscopio e mostra il clock dopo esser passato attraverso il buffer 74HC126. La frequenza di lavoro è di 1 MHz.

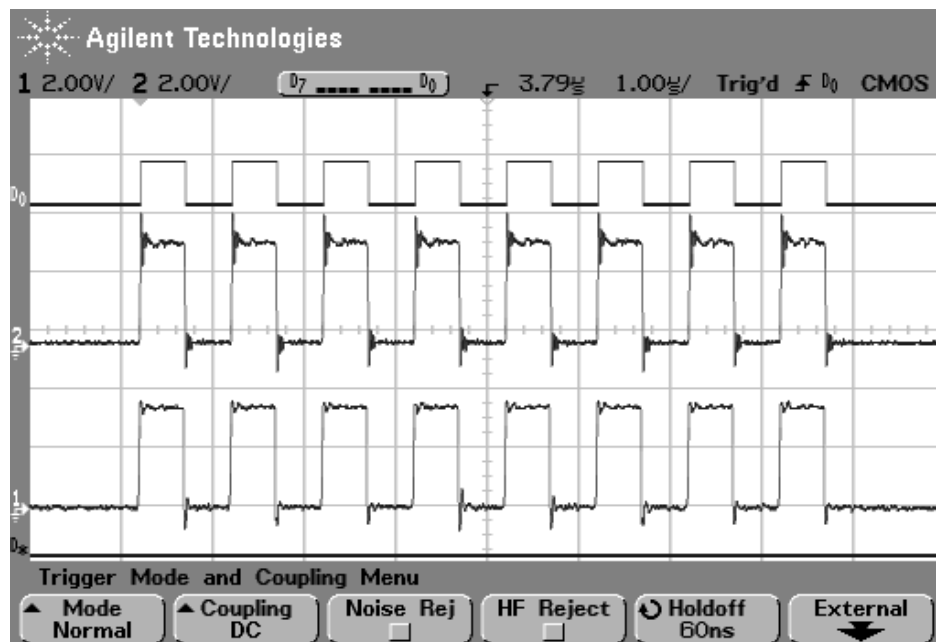


FIG. 6.21 - SEGNALI DI CLOCK A 1 MHz A MONTE E A VALLE DEL BUFFER 74HC126

Dato che in questo test la frequenza di clock è la più alta possibile per il tipo di quarzo adottato, è anche quella in cui anche i segnali sono maggiormente degradati. Si notano infatti, in maniera più accentuata sul clock a valle del buffer, delle oscillazioni ad ogni cambio di stato del segnale.

Le oscillazioni tendono a smorzarsi come la frequenza del clock scende; a tal riguardo vengono riportate nella pagina successiva altre due prove sperimentali in questa condizione.

Bisogna notare comunque che la scheda è un prototipo e il tutto è montato su una scheda preforata, quindi delle considerazioni definitive riguardo la qualità dei segnali non possono essere effettuate in questo momento.

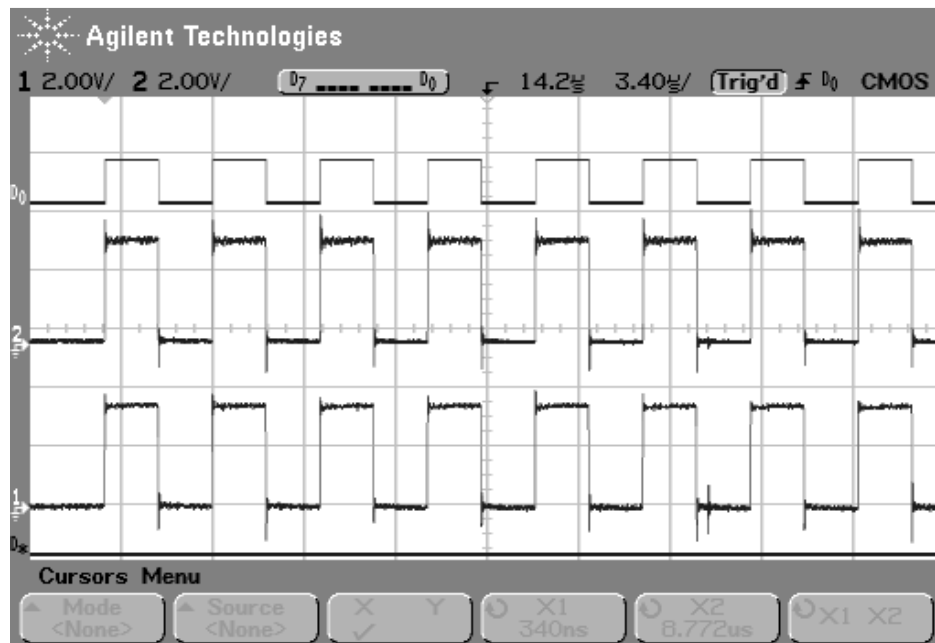


FIG. 6.22 - SEGNALE DI CLOCK ALLA FREQUENZA DI 250 kHz

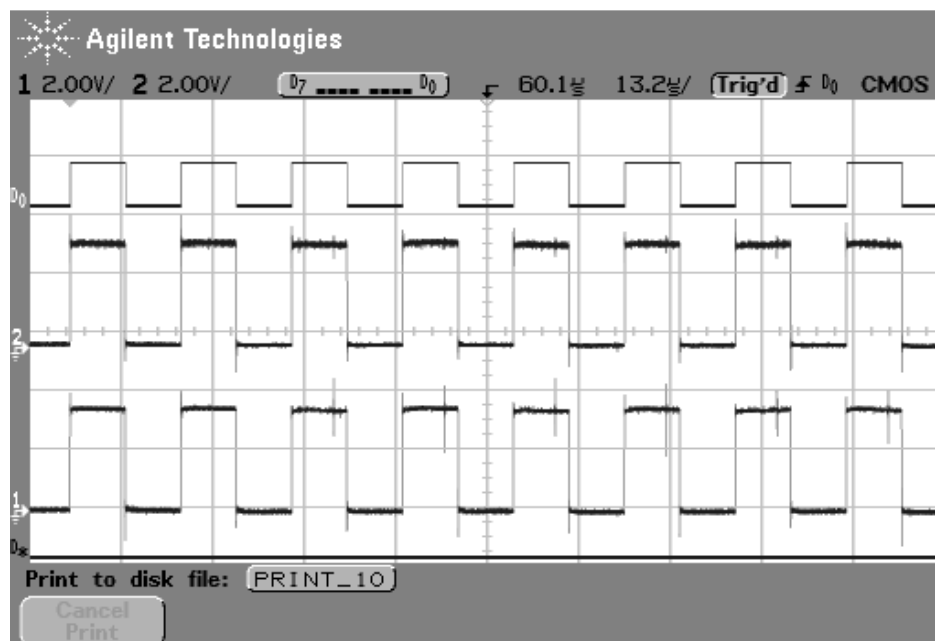


FIG. 6.23 - SEGNALE DI CLOCK A 62.5 kHz

6.3 Scrittura e lettura mediante la periferica EUSART

Il modulo EUSART, Enhanced Universal Synchronous Asynchronous Receiver Transmitter, è uno delle due periferiche seriali di I/O presenti nel microcontrollore PIC18LF8722; questo modulo è anche conosciuto come SCI ovvero Serial Communication Interface.

La periferica EUSART è utile nel progetto della scheda OBDH & HK in quanto può venir configurata come un sistema asincrono full-duplex tipicamente utilizzato in comunicazioni con Personal Computer. Le modalità sincrone vengono scartate perché non sono d'interesse in questo contesto.

Le operazioni di ciascuna periferica EUSART sono pienamente controllate mediante l'ausilio di tre Special Function Register: il Transmit Status and Control Register (TXSTAx), il Receive Status and Control Register (RCSTAx) e il Baud Rate Control (BAUDCONx).

La modalità asincrona è selezionata resettando il bit SYNC del registro TXSTA; in tal maniera la periferica EUSART impiega un formato Non-Return-to-zero (NRZ) costituito da 1 bit di start, 8 o 9 bit di dati e 1 bit di stop.

I pin della periferica EUSART1 ed EUSART2 sono multiplexati rispettivamente con le funzioni della porta PORTC (RC6/TX1/CK1 e RC7/RX1/DT1) e della porta PORTG (RG1/TX2/CK2 e RG2/RX2/DT2).

Per dedicare questi pin alla comunicazione asincrona le periferiche devono venir abilitate e i buffer interni al microcontrollore dei rispettivi pin devono venir configurati opportunamente come ingressi e uscite.

I blocchi funzionali fondamentali della periferica EUSART sono:

- il Baud Rate Generator;
- il trasmettitore asincrono;
- il ricevitore asincrono.

Il Baud Rate Generator crea le opportune frequenze di trasmissione e ricezione seguendo gli standard di uso comune; per mezzo della coppia di registri SPBRGHx e SPBRGx si controlla il periodo di un timer che si dedica a questa funzione. Questo valore può essere sia a 8 bit che a 16 bit, a seconda della precisione che si vuole ottenere.

BAUD RATE (K)	SYNC = 0, BRGH = 0, BRG16 = 0								
	F _{OSC} = 4.000 MHz			F _{OSC} = 2.000 MHz			F _{OSC} = 1.000 MHz		
	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)	Actual Rate (K)	% Error	SPBRG value (decimal)
0.3	0.300	0.16	207	300	-0.16	103	300	-0.16	51
1.2	1.202	0.16	51	1201	-0.16	25	1201	-0.16	12
2.4	2.404	0.16	25	2403	-0.16	12	—	—	—
9.6	8.929	-6.99	6	—	—	—	—	—	—
19.2	20.833	8.51	2	—	—	—	—	—	—
57.6	62.500	8.51	0	—	—	—	—	—	—
115.2	62.500	-45.75	0	—	—	—	—	—	—

FIG. 6.24 - VALORI DEL REGISTRO SPBRGx PER VARIE BAUD RATE E VARIE FREQUENZE DI CLOCK

Per quanto riguarda il trasmettitore asincrono, il cuore di questo sottosistema è costituito dal Transmit Shift Register TSRx; esso ottiene i dati dal Read/Write Transmit Buffer Register TXREG accessibile via software.

Una volta che il registro TXREGx trasferisce il dato nello shift register rimanendo vuoto, e questo avviene in un ciclo macchina, successivamente il flag di interrupt TXxIF viene settato.

Mentre il flag TXxIF indica lo svuotamento del registro di buffer, il bit TRMT indica lo stato dello shift register. Quest'ultimo è un bit a sola lettura che viene attivato quando il dato è shiftato completamente dal TSRx, quindi effettuando un polling su di esso si può verificare quando la trasmissione è conclusa.

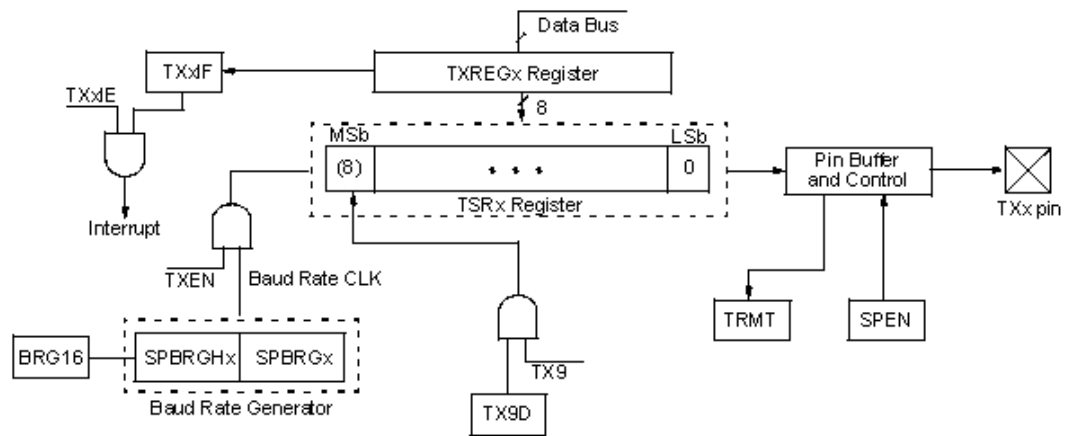


FIG. 6.25 - TRASMETTITORE ASINCRONO

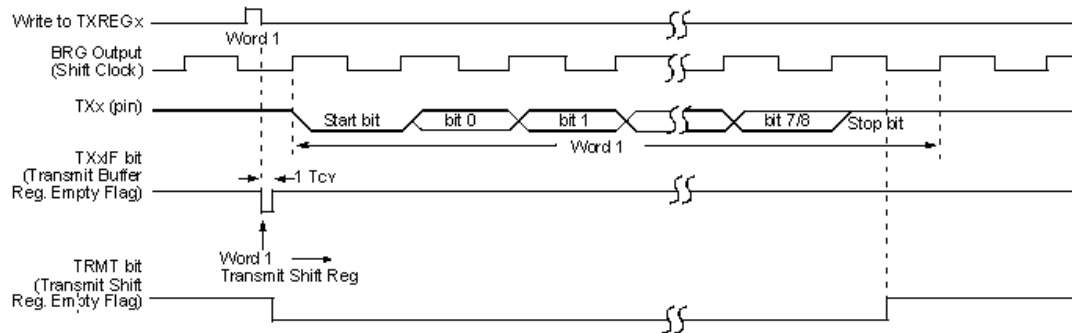


FIG. 6.26 - DIAGRAMMA DELLA TRASMISSIONE ASINCRONA

Dal diagramma in figura 6.26 si nota come il flag di interrupt TXxIF venga settato quando il buffer si svuota e come il bit TRMT si riporti alto a trasmissione completata.

Per quanto riguarda il ricevitore, i dati vengono ricevuti sul pin RXx e guidati subito nel Recovery Block. Questo blocco è composto da uno shift register che opera ad una frequenza 16 volte maggiore di quella del Baud Rate Generator e serve per campionare efficacemente i dati in ingresso. I singoli bit ricevuti vengono salvati nel Receive Shift Register e successivamente copiati nel buffer RCREGx. Il flag di interrupt RCxIF viene settato quando è stato ricevuto l'intero dato e viene resettato automaticamente quando viene eseguita la lettura del buffer RCREGx.

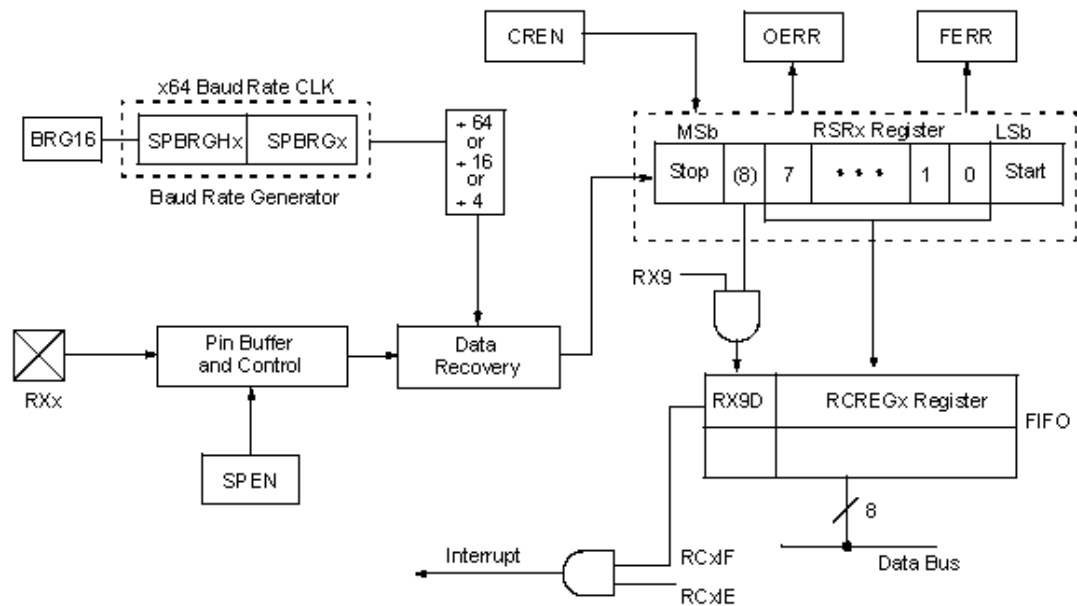


FIG. 6.27 - RICEVITORE ASINCRONO

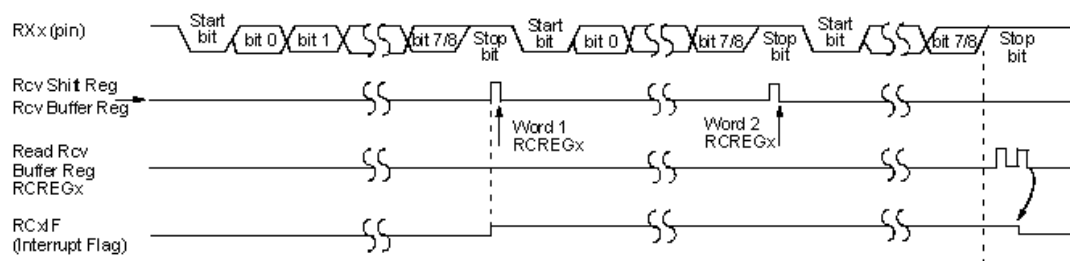


FIG. 6.28 - DIAGRAMMA DELLA RICEZIONE ASINCRONA

Il diagramma delle temporizzazioni mostra la ricezione di 3 byte. Dopo il primo byte il flag di interrupt viene settato e all'invio dei successivi byte non ritorna ad uno stato basso perché erroneamente non viene effettuata la lettura del registro di buffer.

I contenuti dei tre registri di controllo e configurazione sono i seguenti:

TXSTA_x: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

FIG. 6.29 - REGISTRO DI TRASMISSIONE DELLA PERIFERICA EUSART

I bit considerati sono:

- **TX9:** con questo bit si seleziona se la trasmissione è a 8 o 9 bit; in pratica si considera o meno il controllo di parità. Questo bit viene posto a 0 in modo da ottenere una trasmissione ad un singolo byte;
- **TXEN:** nel momento in cui viene attivato il modulo di trasmissione viene abilitato e si è pronti per la trasmissione;
- **SYNC:** ponendo a 0 questo bit si predispone la periferica EUSART ad operare in modalità asincrona;
- **BRGH:** viene posto a 0 in modo da avere una trasmissione di tipo Low Speed. Ciò condiziona la scelta del valore con cui caricare il Baud Rate Generator;
- **TRMT:** come detto effettuando un polling su questo bit si controlla quando la trasmissione si è conclusa.

RCSTA_x: RECEIVE STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7							bit 0

FIG. 6.30 - REGISTRO DI RICEZIONE DELLA PERIFERICA EUSART

I bit considerati sono:

- **SPEN:** attivando questo bit si abilita la periferica seriale EUSART;
- **RX9:** a seconda del valore di questo bit si considera o meno il bit di parità; viene posto a 0;
- **CREN:** dal momento che viene posto a 1 il modulo ricevitore viene abilitato.

BAUDCON_x: BAUD RATE CONTROL REGISTER

R/W-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

FIG. 6.31 - REGISTRO DI CONTROLLO DEL BAUD RATE DELLA PERIFERICA EUSART

Il bit considerato è:

- **BRG16:** con questo bit si seleziona se impostare il periodo del Baud Rate Generator a 8 o 16 bit. In questa configurazione viene escluso il registro SPBRGHx e si imposta un timer a 8 bit.

La figura 6.32 illustra il diagramma di flusso di trasmissione di un byte per mezzo della periferica EUSART1.

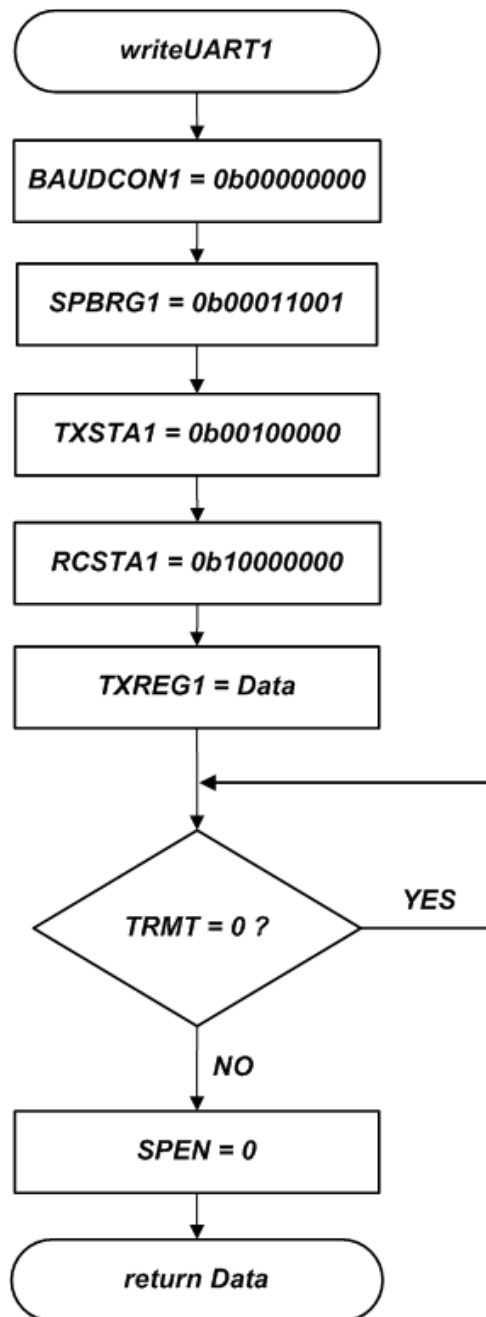


FIG. 6.32 - DIAGRAMMA DI FLUSSO PER LA TRASMISSIONE DI UN BYTE CON LA PERIFERICA EUSART1

Le assegnazioni nei primi quattro registri configurano la periferica EUSART1 in modo da instaurare una comunicazione con 1 bit di start, 8 bit di dati e 1 bit di stop, il tutto ad una velocità di 2400 baud; ulteriori configurazioni riguardano l'accensione del modulo EUSART1 e l'attivazione del trasmettitore.

L'inizio della trasmissione del byte inizia immediatamente con il caricamento del registro TXREG1; quando tutti i bit sono stati inviati lo stato dello shift register TRMT segnala la conclusione dell'operazione e solo successivamente si termina la procedura.

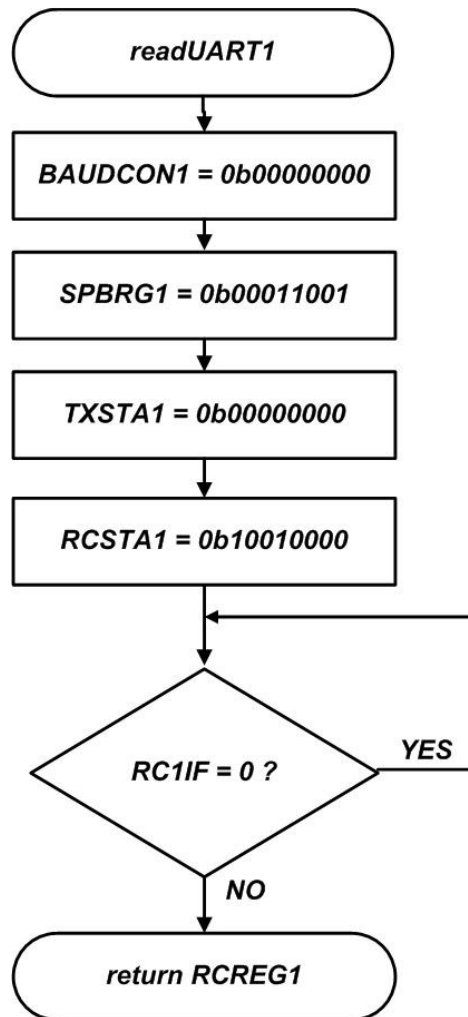


FIG. 6.33 - DIAGRAMMA DI FLUSSO PER LA RICEZIONE DI UN BYTE CON LA PERIFERICA EUSART1

Le assegnazioni nei primi quattro registri configurano nel medesimo modo del caso precedente la periferica EUSART1. Nella comunicazione sono sempre presenti 1 bit di start, 8 bit di dati e 1 bit di stop, il tutto ad una velocità di 2400 baud; ulteriori configurazioni riguardano l'accensione del modulo EUSART1 e attivano il ricevitore.

Il test eseguito sul flag di interrupt segnala il completamento della ricezione mentre l'istruzione di return trasferisce il dato salvato.

```

//-----
// Write data into UART1
// INPUT: unsigned char Data, data to be written into UART1
//-----

void writeUART1 (unsigned char Data)
{
    BAUDCON1 = 0b00000000;           // 8 bit Baud Rate Generator
    SPBRG1 = 0b00011001;             // 2400 Baud
    TXSTA1 = 0b00100000;             // Low Speed, Asynchronous Mode,
                                     // 8 bit, Transmission ON
    RCSTA1 = 0b10000000;             // Serial Port enable,
                                     // Reception OFF
    TXREG1 = Data;                   // Write buffer

    while ( TXSTA1bits.TRMT == 0 );   // Test end of transmission

    RCSTA1bits.SPEN = 0;              // Turn OFF EUSART1
}

```

FIG. 6.34 - CODICE C PER LA TRASMISSIONE DEI DATI SULLA PERIFERICA EUSART1

```

//-----
// Read data from UART1
// OUTPUT: unsigned char , data read from UART1
//-----

unsigned char readUART1 ()
{
    BAUDCON1 = 0b00000000;           // 8 bit Baud Rate Generator
    SPBRG1 = 0b00011001;             // 2400 Baud
    TXSTA1 = 0b00000000;             // Transmission OFF
    RCSTA1 = 0b10010000;             // Continuous receive enable,
                                     // 8 bit,
                                     // Serial Port enable
    while ( PIR1bits.RC1IF == 0 );   // Test interrupt flag

    Data = RCREG1;                   // Save data
    RCSTA1bits.SPEN = 0;              // Turn OFF EUSART1

    return RCREG1;                   // Return received value
}

```

FIG. 6.35 - CODICE C PER LA RICEZIONE DEI DATI DALLA PERIFERICA EUSART1

Entrambi i codici di ricezione e trasmissione sono stati scritti implementando la periferica EUSART1 che fa capo alla porta RS-232. La gestione della periferica EUSART2 viene controllata nella medesima modalità, consentendo di instaurare la comunicazione tra il microcontrollore ed il GPS.

6.4 Acquisizioni analogiche

Il modulo ADC permette la conversione di un segnale analogico in una corrispondente parola digitale a 10 bit.

Per ottenere un'adeguata conversione è indispensabile selezionare il clock della conversione. Esso infatti deve essere il più breve possibile ma in ogni caso maggiore del minimo tempo di conversione per singolo bit, definito come T_{ad} (1.4 μ s). A tal scopo è a disposizione la seguente tabella che permette di ottenere la giusta configurazione a partire dalle frequenza del clock dell'oscillatore principale.

AD Clock Source (T_{AD})		Maximum Device Frequency	
Operation	ADCS2:ADCS0	PIC18FXXX	PIC18LFXXX ⁽⁴⁾
2 TOSC	000	2.86 MHz	1.43 kHz
4 TOSC	100	5.71 MHz	2.86 MHz
8 TOSC	001	11.43 MHz	5.72 MHz
16 TOSC	101	22.86 MHz	11.43 MHz
32 TOSC	010	40.0 MHz	22.86 MHz
64 TOSC	110	40.0 MHz	22.86 MHz
RC ⁽³⁾	x11	1.00 MHz ⁽¹⁾	1.00 MHz ⁽²⁾

FIG. 6.36 - CONFIGURAZIONE DEL CLOCK DI CONVERSIONE

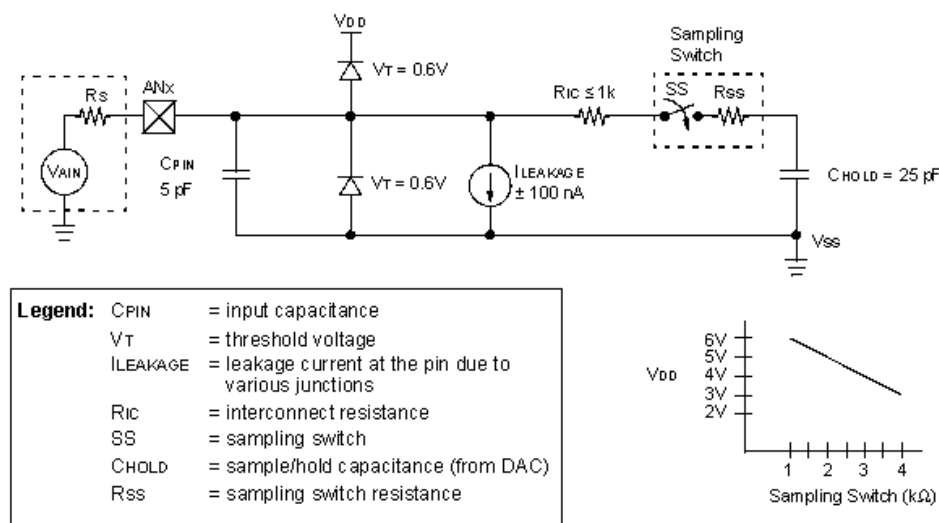


FIG. 6.37 - CIRCUITO SAMPLE & HOLD DEL CONVERTITORE ANALOGICO DIGITALE

Una volta che il modulo ADC viene configurato, il canale selezionato tramite il multiplexer interno deve essere acquisito prima che abbia inizio la conversione; bisogna assicurarsi inoltre che il pin del microcontrollore abbia il corrispondente buffer selezionato come ingresso.

La necessità di attendere un corretto tempo di acquisizione deriva dal fatto che il condensatore di mantenimento Chold deve essere completamente caricato al livello del segnale analogico in ingresso. L'impedenza della sorgente R_s e dello switch di campionamento R_{ss} condizionano direttamente il tempo richiesto per la carica del

condensatore Chold e quindi il tempo di acquisizione. A tal proposito dalle specifiche del datasheet l'impedenza massima della sorgente non deve superare i 2.5 k Ω .

Il tempo di acquisizione può essere impostato via software e varia tra $2T_{ad}$ e $20T_{ad}$.

Mantenendosi all'interno dell'intervallo appena citato, una volta avviata la conversione, la periferica provvederà automaticamente a garantire questo tempo, continuando a campionare l'ingresso analogico. Questa modalità viene definita acquisizione automatica. Esiste inoltre una modalità manuale, dove una volta selezionato il canale, è necessario attendere il tempo di acquisizione e ciò deve essere fatto mediante un ritardo di tipo software opportunamente calcolato.

In ogni caso, in entrambe le modalità, la condizione essenziale da garantire è che tra la selezione del canale e l'avvio della conversione sia rispettato il tempo di acquisizione; a questo punto la conversione può essere avviata mediante il settaggio del bit $\overline{GO/DONE}$ (registro ADCON0).

A conversione completata, il risultato viene caricato nella coppia di registri ADRESH e ADRESL, il bit $\overline{GO/DONE}$ si resetta e il flag di interrupt ADIF si attiva.

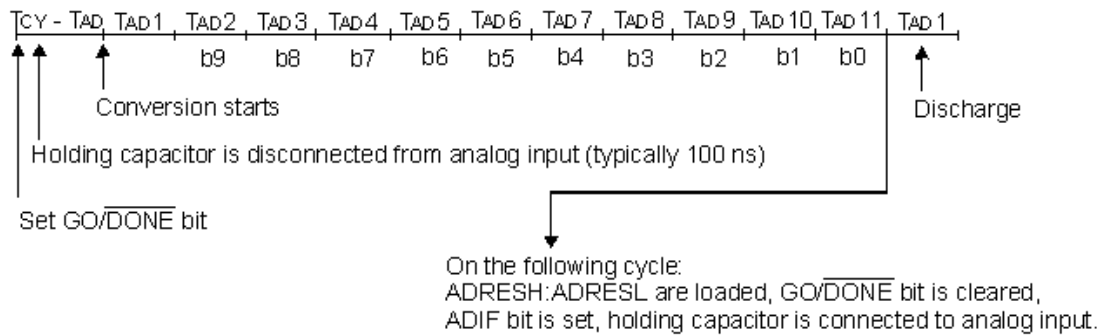


FIG. 6.38 - ACQUISIZIONE MANUALE

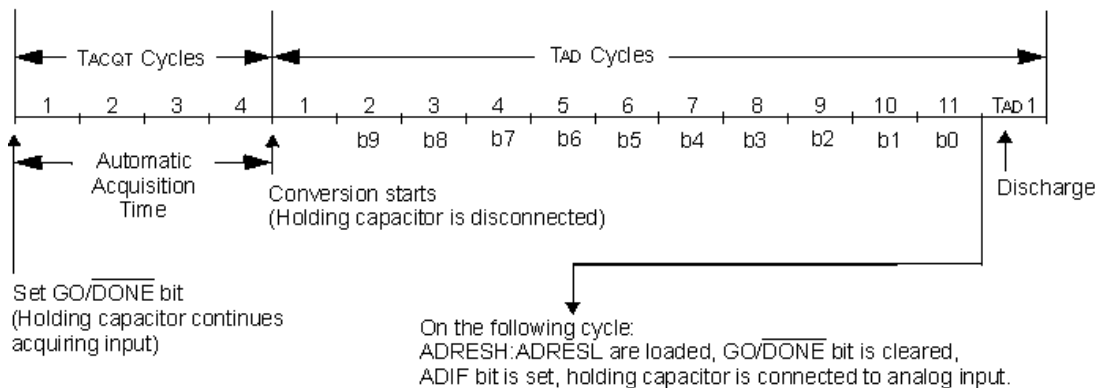


FIG. 6.39 - ACQUISIZIONE AUTOMATICA

I registri messi a disposizione dal modulo ADC sono l'A/D Result High Register (ADRESH), l'A/D Result Low Register (ADRESL), A/D Control Register 0,1 e 2 (ADCON0, ADCON1, ADCON2).

ADCON0: A/D CONTROL REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON
bit 7							bit 0

FIG. 6.40 - REGISTRO DI CONTROLLO ADCON0

I bit svolgono le seguenti funzioni:

- **CHS<3:0>**: questo nibble permette di commutare il multiplexer interno al microcontrollore e quindi provvede a selezionare il canale sul quale effettuare la conversione;
- **GO/DONE**: dal momento che viene settato la conversione ha inizio; è la periferica stessa che provvede a resettarlo al termine dell'operazione;
- **ADON**: attivando questo bit si abilita la periferica ADC.

ADCON1: A/D CONTROL REGISTER 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							bit 0

FIG. 6.41 - REGISTRO DI CONTROLLO ADCON1

I bit del registro ADCON1 svolgono invece queste funzioni:

- **VCFG<1:0>**: configurando questi due bit si selezionano quattro modalità con le quali ottenere le sorgenti delle tensioni di riferimento dell'ADC;
- **PCFG<3:0>**: è disponibile un ulteriore nibble per poter scegliere quanti canali analogici implementare. I pin dell'ADC che non vengono utilizzati come ingressi analogici possono svolgere un'altra qualsiasi funzione digitale.

ADCON2: A/D CONTROL REGISTER 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

FIG. 6.42 - REGISTRO DI CONTROLLO ADCON2

I bit del registro ADCON2 infine sono:

- **ADFM**: essendo la risoluzione dell'ADC pari a 10 bit ed essendoci due registri da 8 bit per salvare il risultato, tramite questo bit si seleziona se giustificare il tutto verso destra o sinistra dei 16 bit messi a disposizione;
- **ACQT<2:0>**: con questa configurazione è possibile scegliere il tempo di acquisizione;
- **ADCS<2:0>**: questi due ultimi bit selezionano invece il clock di conversione.

La figura 6.43 illustra la sequenza di istruzioni eseguita per svolgere una conversione analogica-digitale.

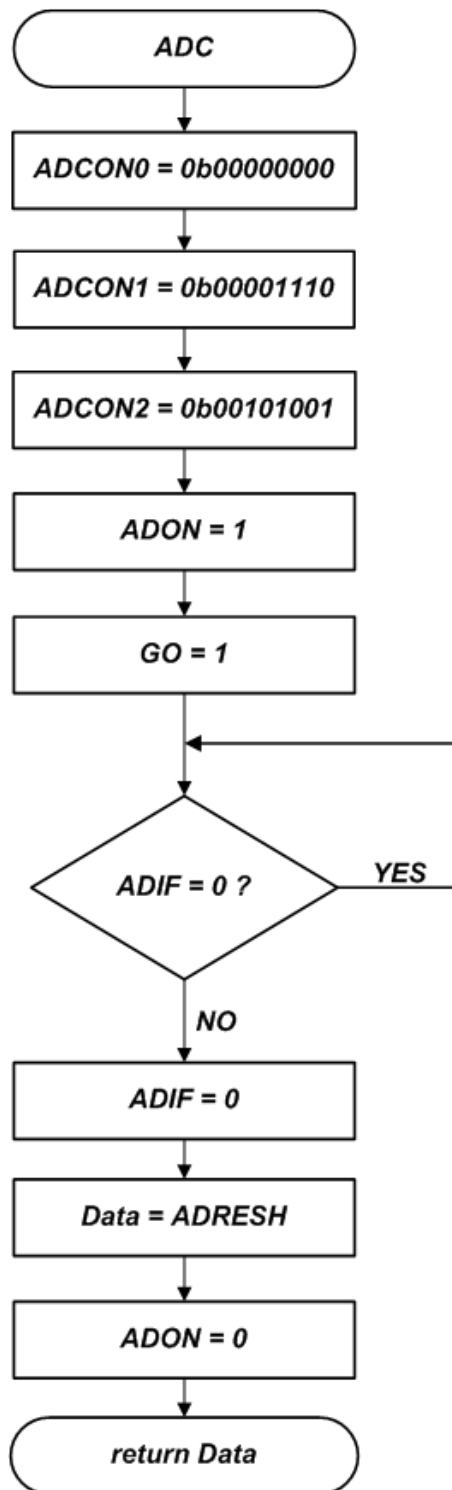


FIG. 6.43 - DIAGRAMMA DI FLUSSO DELLA CONVERSIONE ANALOGICA-DIGITALE

Le prime operazioni inizializzano e configurano i registri dell'ADC del microcontrollore. Si sceglie il pin analogico RA0 come ingresso, i riferimenti di tensione vengono posti pari alle

alimentazioni analogiche del microcontrollore, il tempo di acquisizione è di $12T_{ad}$ con un clock di conversione di $F_{osc}/8$.

L'operazione successiva consiste nell'accendere il modulo convertitore ed ad iniziare la conversione.

Il flag di interrupt ADIF viene sfruttato per verificare quando la conversione è terminata e viene resettato successivamente al suo test.

Le ultime operazioni invece consistono nel salvare il risultato dell'acquisizione e a restituire il suo valore.

```
//-----  
// AD conversion  
// OUTPUT: unsigned char, ADRESH  
//-----  
  
unsigned char ADC ()  
{  
    unsigned char Data;  
    ADCON0 = 0b00000000;           // CH0=RA0, ADC OFF  
    ADCON1 = 0b00001110;           // +VREF=AVdd, -VREF=AVss,  
    ADCON2 = 0b00101001;           // RA0 = Analog  
    ADCON0bits.ADON = 1;           // LEFT justified,  
    ADCON0bits.GO = 1;             // 12 TAD, FOSC/8  
    ADCON0bits.ADON = 1;           // ADC=ON  
    ADCON0bits.GO = 1;             // Start Conversion  
  
    while ( PIR1bits.ADIF == 0 );  // Wait end of conversion  
    PIR1bits.ADIF = 0;             // Reset interrupt flag  
  
    Data = ADRESH;                 // Save result  
    ADCON0bits.ADON = 0;           // ADC=OFF  
  
    return Data;                   // Return result of  
    // conversion(onlyMSbyte)  
}
```

FIG. 6.44 - CODICE C DELLA CONVERSIONE ANALOGICA DIGITALE

7 Real Time Operating System di Atmocube

Nell'introduzione del capitolo 4 sono state illustrate le principali differenze tra un sistema di tipo Embedded ed un PC; si è messo in evidenza che la potenza del microcontrollore in uso generalmente viene dedicata ad uno specifico task, mentre un PC gestisce simultaneamente più programmi e dispositivi esterni sfruttando più risorse. Passando ora ad un'analisi dal punto di vista del software, la seconda caratteristica che modella il prototipo del modulo OBDH & HK ad un microcomputer è l'installazione di un sistema operativo come software primario di controllo; il tipo implementato in questo progetto viene comunemente chiamato Real Time Operating System o più brevemente RTOS. Utilizzando questo approccio ci si svincola dalla scrittura di un unico programma globale; è possibile infatti spezzare il software del satellite in più parti, semplificandone lo studio e il progetto, e ottenere una maggior affidabilità e stabilità del sistema passando il controllo della CPU periodicamente al RTOS anche se un qualsiasi task è in esecuzione. Ci sono a disposizione numerosi RTOS open source per i più comuni microcontrollori sul mercato; essi possono avere una struttura del tutto generica, dove per il corretto funzionamento è necessario il cosiddetto lavoro di "porting" ovvero di adattamento, oppure possono essere specifici per un determinato dispositivo e quindi immediatamente utilizzabili.

Nel prossimo paragrafo si inizia trattando le operazioni che svolge un generico μ Kernel, ovvero il nucleo del RTOS.

Successivamente si passa allo studio dello stack del PIC18LF8722, della struttura degli interrupt e del Timer 0 del microcontrollore, tutti strumenti essenziali per il progetto del sistema operativo.

Gli ultimi paragrafi invece entrano più nello specifico dell'argomento trattando il RTOS ad alto livello progettato per Atmocube.

7.1 Operazioni del μ Kernel

Il kernel costituisce il nucleo di un sistema operativo ed è un software che ha il compito di fornire ai processi in esecuzione un accesso sicuro e controllato all'hardware. Dato che può essere eseguito simultaneamente più di un processo, il kernel ha la responsabilità di assegnare una parte del tempo-macchina e di accesso all'hardware a ciascun programma. Ogni programma eseguito viene definito come un task sotto il controllo del sistema operativo e l'esecuzione di un insieme di task a sua volta prende il nome di multitasking. I maggiori vantaggi che derivano da un'ingegneria del software di questo tipo, tra cui il primo già annunciato nell'introduzione di questo capitolo, vengono riassunti in questi tre punti:

- il multitasking e la comunicazione tra i vari task permettono ad applicazioni complesse di essere partizionate in un set di programmi più piccoli e più gestibili;
- la partizione in più task implica un lavoro di test software più agevole, una suddivisione dei compiti all'interno di un team di progetto e il riutilizzo di parti di codice comune a più applicativi;
- complesse temporizzazioni possono essere rimosse dal codice di ogni singolo task e diventano responsabilità del kernel.

Un processore convenzionale può eseguire un singolo task alla volta; ciò nonostante, passando rapidamente da un task ad un altro, il sistema operativo multitasking può dare l'impressione che tutti i programmi siano eseguiti contemporaneamente e parallelamente; questo è concesso al prezzo di una perdita di tempo.

Questo concetto viene esplicitato dai diagrammi di fig. 7.1:

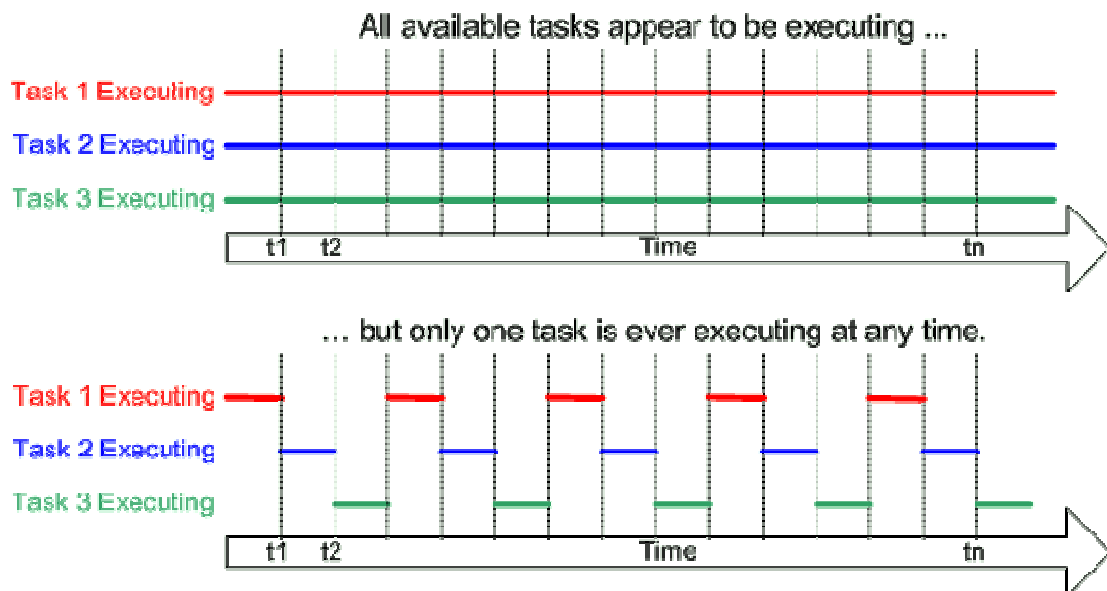


FIG. 7.1 - APPARENTE ESECUZIONE DI 3 TASK E REALE SWITCHING

Esistono sostanzialmente due modi di implementare il multitasking: *cooperative* e *preemptive*.

Nel cooperative multitasking sono i programmi che, spontaneamente, cedono il controllo al sistema non appena hanno terminato la singola operazione in corso. Questo metodo assorbe meno risorse di calcolo e ha un ritardo di commutazione trascurabile tra ogni task; esso non richiede nessuna struttura hardware dedicata, il che rende possibile l'implementazione su qualunque calcolatore; per contro è molto vulnerabile agli errori, in quanto generalmente il crash di un programma fa cadere l'intero sistema, e l'isolamento fra i processi è molto debole.

Nel preemptive multitasking è lo schedatore che ferma i programmi allo scadere del tempo assegnato e trasferisce il controllo dall'uno all'altro.

Lo schedatore è una parte fondamentale del kernel e ha il compito di far coabitare i differenti task; esso manda in esecuzione più processi, assegna loro a turno la CPU e sospende l'esecuzione dei programmi in attesa di eventi esterni (lettura/scrittura sulle periferiche, input da parte dell'utente ecc.) finché questi non si verificano. In linea di principio si possono usare questi tempi "morti" per far eseguire un altro programma.

Lo schedatore quindi è responsabile di decidere quale task debba essere eseguito ad ogni particolare istante di tempo, quando esso debba essere sospeso e quando riattivato.

Per comprendere meglio questo meccanismo viene analizzata la figura 7.2.

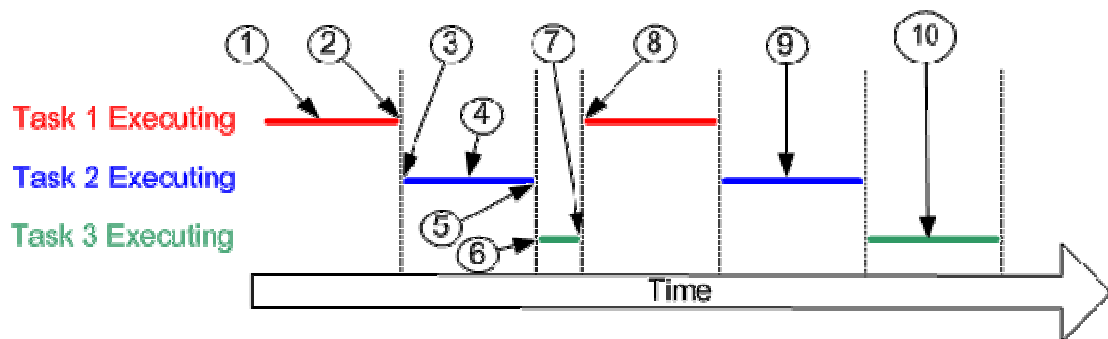


FIG 7.2 - ESEMPIO DI SOSPENSIONE E RIATTIVAZIONE DI DIVERSI TASK

Le operazioni svolte sono:

1. il task 1 è in esecuzione;
2. il kernel sospende il task 1;
3. il kernel riattiva il task 2;
4. il task 2 è in esecuzione e blocca una certa periferica per il suo accesso esclusivo;
5. il task 2 viene sospeso;
6. il task 3 viene riattivato;
7. il task 3 cerca di utilizzare la stessa periferica bloccata dal task 2 e dato che non può continuare si autosospende;
8. il task 1 viene riattivato e ricomincia il ciclo;
9. il task 2 finisce di usare la periferica e subito la rilascia;
10. ora il task 3 si impossessa della periferica e viene sospeso solo dal kernel.

Un task sospeso o bloccato non è in grado di essere eseguito, di conseguenza per esso non viene dedicato ed allocato alcun tempo di processo; queste situazioni vengono gestite assegnando uno stato ad ogni task.

I vari tipi di stati possono venir così riassunti:

- ***RUNNING***: tra tutti i task presenti in coda per essere eseguiti è quello effettivamente in esecuzione sulla CPU;
- ***READY***: in questo stato il nucleo considera il task pronto per essere eseguito dalla CPU;
- ***WAITING***: il task si trova in una condizione temporaneamente inattiva e può ritornare a far parte dei processi pronti ad essere mandati in esecuzione dalla CPU qualora sopraggiunga un particolare evento;
- ***SUSPENDED***: il task è presente nel progetto, ma non essendo attivo non è preso in considerazione dal nucleo.

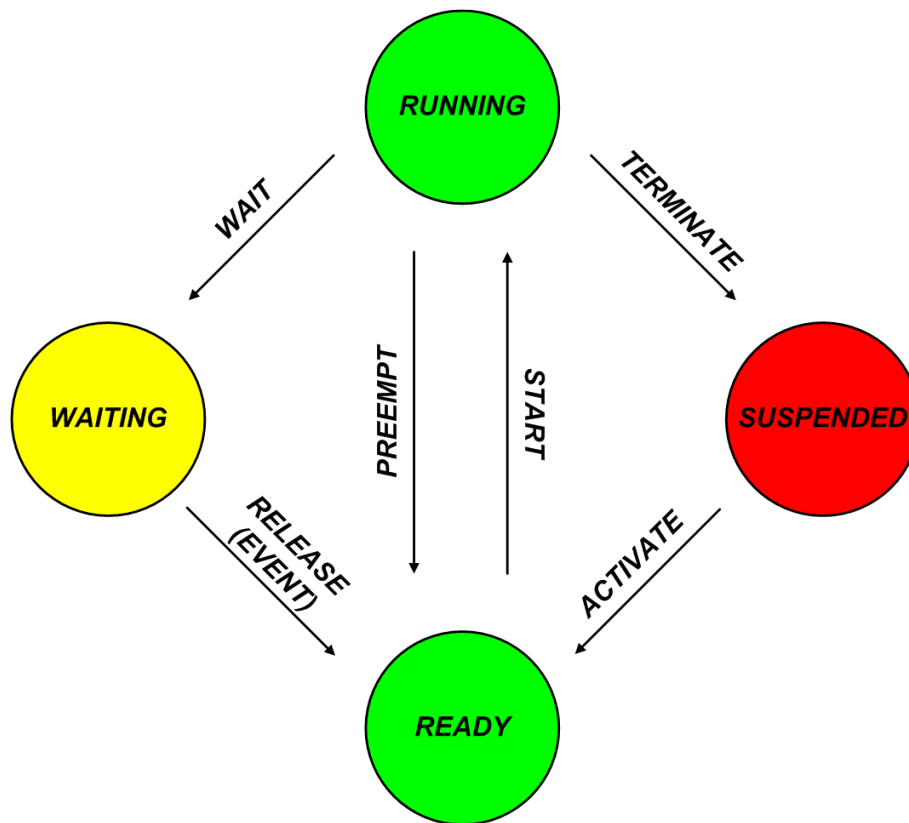


FIG. 7.3 - STATI FONDAMENTALI DEI TASK

In conclusione il preemptive multitasking necessita di CPU che possano implementare una logica specifica per il cambio di task eseguito dallo scheduler. Poiché l'interruzione dei programmi è arbitraria, al cambio di task il sistema operativo è costretto a salvare tutti o quasi i registri della CPU e ricaricarli con quelli salvati dal task che subentra perdendo molto tempo.

A fronte di queste maggiori richieste il preemptive multitasking offre una sicurezza del sistema molto maggiore e un'immunità ai crash causati da errori nei programmi.

Il Real Time Operating System, cioè il tipo di sistema operativo impiegato per Atmocube, si basa sui principi visti per il preemptive multitasking e il suo principale obiettivo è fornire

una risposta in tempi definiti ad eventi che si presentano nel sistema. Dal momento che si verifica un evento può accadere che la reazione dal parte del microcontrollore debba rispettare delle scadenze ben precise, quindi lo schedatore deve assicurare che esse siano rispettate.

In via di principio l'intervallo di tempo in cui il sistema operativo deve reagire non è importante, non deve essere quindi necessariamente veloce, la cosa importante è che risponda entro un tempo massimo determinato, cioè sia un sistema prevedibile.

A tal fine, il progettista del software deve assegnare una priorità in modo che il task con quella maggiore abbia il tempo di processo adeguato e necessario. Questo può voler dire di assegnare priorità uguali a task che sono pronti ad andare in esecuzione simultaneamente; tale particolare condizione prende il nome di Round Robin.

Nel terzo punto dell'elenco dei vantaggi di un sistema multitasking sono state prese in considerazione le temporizzazioni. Queste, all'interno del sistema software e quindi dei vari task, vengono controllate dal cosiddetto battito del sistema operativo o più precisamente dal RTOS tick. Si tratta di un interrupt periodico controllato da un timer del microcontrollore che incrementa un contatore; in questo modo al kernel è permesso di misurare tempi con una risoluzione pari alla frequenza dell'interrupt. Ogni volta che la variabile contatore viene incrementata il kernel verifica se è arrivato il momento di sbloccare o svegliare un certo task o di eseguire qualche operazione. Può accadere che un task appena sbloccato abbia una priorità più alta del task appena interrotto; una volta all'interno della routine di interrupt ISR, il ritorno quindi deve essere effettuato sul task a priorità più elevata.

Questa condizione viene riassunta nella seguente sequenza:

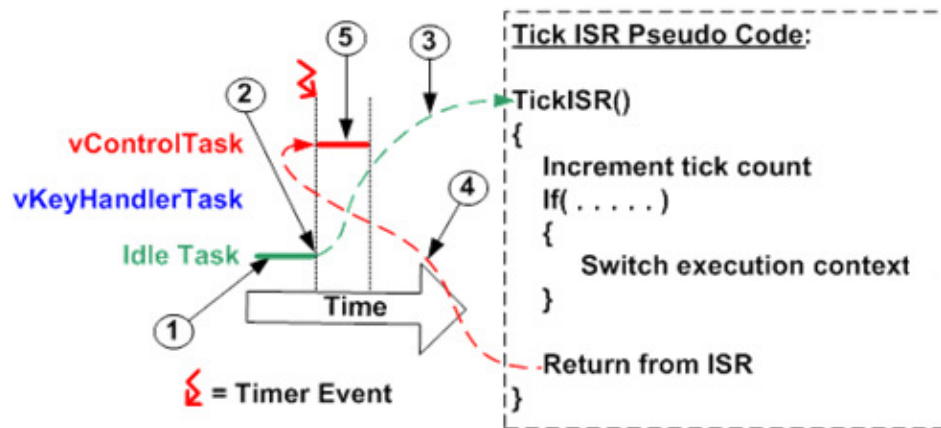


FIG. 7.4 - RTOS TICK GESTITO IN INTERRUPT E SWITCHING VERSO UN TASK A PRIORITA' PIU' ALTA

1. il task Idle è in esecuzione;
2. in questo punto agisce il tick e si punta al vettore di interrupt;
3. l'incremento del contatore causa il risveglio del task vControlTask;
4. vista la maggiore priorità di quest'ultimo task il ritorno avviene presso il suo indirizzo;
5. ora è quindi il vControlTask ad essere attivo e ad usufruire del tempo di processo.

Questa situazione appena descritta prende generalmente il nome di Program Redirection ed è permessa grazie alla manipolazione dello stack. Infatti, dal momento che ci si trova all'interno della routine ISR nel vettore di interrupt, a seconda delle operazioni che è necessario svolgere è possibile modificare il valore dell'indirizzo di ritorno con cui

impostare il Program Counter. In questa maniera si riduce inoltre l'ISR Processing Time, cioè il tempo per il quale l'ISR occupa la CPU, dato che l'evento non viene processato nella stessa ma al di fuori; infatti una volta impostato l'indirizzo di ritorno si esce immediatamente dalla routine di interrupt per consentire il controllo di ulteriori eventi. Un esempio specifico di Program Redirection per il PIC18LF8722 viene riportato nel paragrafo successivo inerente lo stack hardware.

La Program Redirection è integrata in un contesto ben più ampio all'interno del sistema operativo, ovvero fa parte dell'indispensabile funzione denominata Context Switching.

Fin da subito è stato detto che la caratteristica dei sistemi operativi è quella di passare da un task all'altro; visto che ogni task è in esecuzione come se tutta la CPU gli appartenesse e accede alla memoria e ai vari registri, prima di passare il controllo ad un altro task è necessario salvare lo stato della CPU, i principali registri ed in particolare lo stack. Se questi fossero lasciati inalterati dal nucleo ad ogni Context Switching, l'interruzione di un task e la successiva attivazione di un altro implicherebbe il puntamento di ogni istruzione di *return* presso il task appena interrotto.

La convivenza di più task è quindi permessa grazie alla presenza del Context Switching ovvero del cambio di contesto messo a disposizione dal nucleo. Ogni task pertanto è dotato del proprio contesto formato da uno stack software che congela lo stato dei registri e dello stack hardware, il tutto ripristinato ad ogni switch.

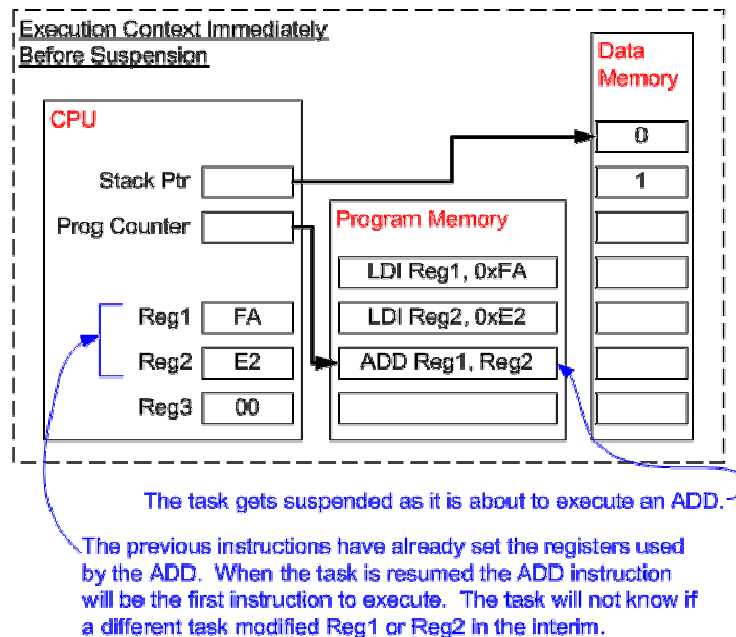


FIG. 7.5 – ASSENZA DI CONTEXT SWITCHING

Per quanto riguarda le interruzioni e il Context Switching c'è da dire che esistono delle condizioni in cui tutti gli interrupt vengono disabilitati. Ciò avviene quando ad esempio si sta effettuando una comunicazione dati per mezzo di una periferica; quello che si vuole evitare è di interrompere uno scambio dati in corso, situazione in cui si perderebbero delle informazioni. A tal proposito è possibile implementare un ulteriore controllo di tipo "time out" se l'operazione non viene compiuta in un certo tempo in modo da poter agire opportunamente. A livello generale tutte le routine che necessitano di essere eseguite senza interruzioni prendono il nome di Critical Time Section e vengono eseguite in un breve periodo di tempo. Queste parti di codice influenzano direttamente l'Interrupt Latency cioè il massimo tempo in cui gli interrupt sono disabilitati più il tempo che impiega la ISR a passare in esecuzione.

7.2 Stack del PIC18LF8722

Lo stack, o la pila, del microcontrollore è la struttura di memoria dove le modalità di accesso seguono una politica di tipo LIFO (Last In First Out), ovvero dove i dati vengono letti in ordine rigorosamente inverso rispetto a quello in cui sono stati inseriti. Questa organizzazione viene sfruttata per immagazzinare informazioni riguardo i sottoprogrammi attivi nel codice programma, cioè quelli chiamati ma non ancora terminati. Più precisamente, il motivo per cui utilizzare questo tipo di struttura deriva dal fatto che è indispensabile tenere traccia dei punti in cui ogni sottoprogramma attivo deve restituire il controllo al termine della sua esecuzione. La funzione chiamante pone l'indirizzo di ritorno del codice macchina in testa alla pila e il sottoprogramma chiamato, alla sua conclusione, lo toglie e restituisce il controllo tramite esso. Queste operazioni, oltre a essere valide per i sottoprogrammi, si estendono anche alla gestione degli interrupt: quando il microcontrollore riconosce un determinato flag, il salto al rispettivo vettore ed il successivo ritorno avvengono nelle stesse modalità.

La pila messa a disposizione dal PIC18LF8722 consente fino a 31 combinazioni di chiamate a sottoprogrammi ed interrupt. Il Program Counter viene salvato dentro allo stack ogniqualvolta viene eseguita un'istruzione di CALL o viene riconosciuto un flag di interrupt. Lo stesso valore del Program Counter salvato viene poi estratto dallo stack quando viene richiamata un'istruzione RETURN, RETLW o RETFIE. La pila del microcontrollore in uso è composta da 31 parole a 21 bit di memoria RAM ed è indipendente dalla memoria programma e da quella dei dati; ogni locazione viene indirizzata tramite un registro puntatore a 5 bit denominato STKPTR, Stack Pointer. Quest'ultimo è sia leggibile che scrivibile e permette di accedere al dato puntato nello stack attraverso un ulteriore registro, ovvero il TOS, Top-of-Stack. I due registri STKPTR e TOS permettono completamente la manipolazione dello stack inserendo dati ed estraendoli. La dimensione del registro TOS è uguale a quella della pila e cioè a quello del Program Counter; per contenere completamente questi dati, il TOS è composto a sua volta da 3 registri a 8 bit chiamati TOSU, TOSH e TOSL. Il registro STKPTR, oltre a contenere il valore a 5 bit per il puntamento delle singole celle, contiene ulteriori due bit che segnalano quando si è raggiunto il fondo o il tetto dello stack: si tratta dei bit di stato Stack Full STKFUL e Stack Underflow STKUNF.

Dopo che il Program Counter è stato inserito 31 volte senza alcuna estrazione il bit STKFUL si attiva e segnala che lo stack è completamente pieno.

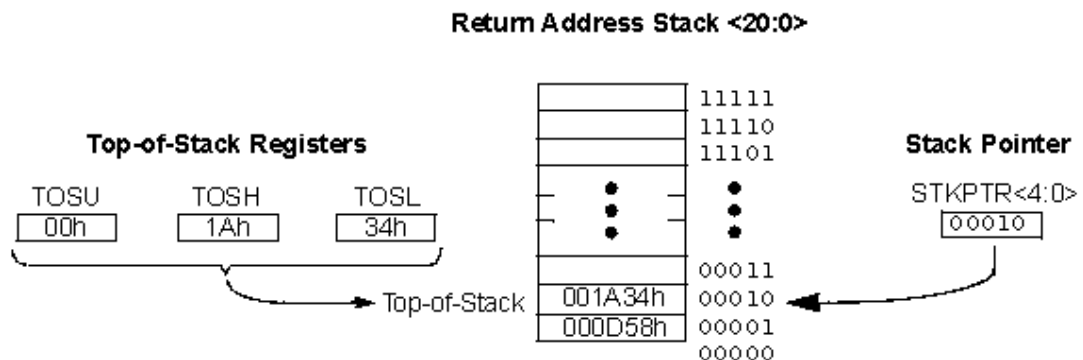


FIG. 7.6 - STRUTTURA ED ORGANIZZAZIONE DELLO STACK E RELATIVI REGISTRI DI ACCESSO

STKPTR: STACK POINTER REGISTER

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL ⁽¹⁾	STKUNF ⁽¹⁾	—	SP4	SP3	SP2	SP1	SP0
bit 7							bit 0

FIG. 7.7 - CONTENUTO DEL REGISTRO DI PUNTAMENTO ALLO STACK

In questa condizione è possibile definire il tipo di azione che si intende eseguire immediatamente dopo, ovvero un reset del microcontrollore e dello Stack Pointer oppure un'inibizione delle successive modifiche del livello più alto della pila. A tal scopo è disponibile il bit Stack Overflow Reset Enable STVREN nei registri di configurazione del microcontrollore. Nel caso si verifichi la situazione opposta, cioè si estraggano i dati dallo stack fino a svuotarla del tutto, si attiva il bit STKUNF ed ad ogni successiva operazione viene restituito un valore nullo.

L'Istruction Set Architecture del nucleo 18F, rispetto alla serie 16F, implementa due nuove tipologie di istruzioni dedicate alla manipolazione dello stack: esse sono la PUSH e la POP e permettono di agire sul TOS modificando gli indirizzi di ritorno durante l'esecuzione del codice. L'istruzione PUSH mette il valore corrente del Program Counter all'interno dello stack incrementando contemporaneamente lo Stack Pointer. L'istruzione POP scarta il valore corrente del TOS, decrementa lo Stack Pointer e aggiorna il TOS con il valore puntato. Il fatto di riuscire ad accedere completamente allo stack hardware implica la possibilità di creare uno stack di tipo software necessario all'implementazione del sistema operativo RTOS. Durante la manipolazione dello stack risulta importante disabilitare gli interrupt in modo da evitare corruzioni involontarie della stessa.

Nella pagina successiva viene riportato un listato Assembler specifico del microcontrollore in uso dove è implementato un Program Redirection; quello che viene fatto è condizionare l'indirizzo di ritorno dell'istruzione di RETFIE (return from interrupt) a seconda delle operazioni che si svolgono all'interno della ISR. La prima istruzione di RETFIE consente di ritornare nella posizione del codice precedente al salto di interrupt, mentre la seconda effettua un salto nella posizione selezionata con l'ausilio dello STKPTR e del TOS.

```

;-----
; Program Redirection Example
;-----

MyInterruptRoutine                                ; ISR
; .                                                ; Data collection interrupt
; .
; .
; .
; .

decfsz DATA_COUNT, F                            ; Check for 8 samples
retfie                                           ; Resume normal execution

movlw 0x08
movwf DATA_COUNT                               ; Reset counter

incf      STKPTR, F                             ; Increment stack pointer

movlw low MyAvgRoutine                          ; Load the TOS to point
; to averaging routine

movwf TOSL
movlw high MyAvgRoutine
movwf TOSH
movlw upper MyAvgRoutine
movwf TOSU

retfie                                           ; Do average
; PROGRAM REDIRECTION
; BRANCH to MyAvgRoutine

MyAvgRoutine
; .                                                ; Average
; .
; .
; .
; .

return

```

FIG. 7.8 - CODICE MICROCHIP ASSEMBLER DI RITORNO CONDIZIONATO

7.3 *Struttura degli interrupt*

Viene dedicato un paragrafo agli interrupt in quanto il loro uso in un sistema operativo RTOS è indispensabile, sia per quanta riguarda il tick di sistema sia per la gestione degli eventi che insorgono durante le varie applicazioni. La famiglia del PIC18LF8722 dispone di sorgenti multiple di interrupt ed una caratteristica di priorità che permette di assegnare via software un livello alto o basso di precedenza. Questa gerarchia a due livelli è permessa grazie alla struttura hardware del microcontrollore che dispone di altrettanti vettori di interrupt, uno appunto ad alta priorità ed uno a bassa priorità facenti capo rispettivamente agli indirizzi di memoria programma 0x0008 e 0x0018. Qualsiasi evento di interrupt ad alta priorità ha la capacità di interrompere un qualsiasi interrupt in corso a bassa priorità.

Ci sono tredici registri complessivamente che sono utilizzati per controllare le operazioni di interrupt. Essi sono:

- **RCON;**
- **INTCON;**
- **INTCON2;**
- **INTCON3;**
- **PIR1, PIR2, PIR3;**
- **PIE1, PIE2, PIE3;**
- **IPR1, IPR2, IPR3.**

Ogni sorgente di interrupt ha a disposizione 3 bit di controllo. Essi sono:

- **Flag bit:** indica che un interrupt è avvenuto;
- **Enable bit:** è un'abilitazione del rispettivo flag e permette al programma di dirottare la sua esecuzione all'indirizzo del vettore di interrupt;
- **Priority bit:** seleziona il tipo di priorità da assegnare a ciascun evento.

La discriminazione tra alta e bassa priorità è abilitata servendosi del bit IPEN, Interrupt Priority Enable, presente nel registro RCON. Quando IPEN è a livello logico alto si rendono disponibili due bit di abilitazione globale degli interrupt, il GIEL, Global Interrupt Enable Low, e il GIEH, Global Interrupt Enable High. Quando il Flag bit, l'Enable bit e l'appropriato bit di abilitazione globale GIEL o GIEH vengono settati, allora il Program Counter viene indirizzato verso uno dei vettori di interrupt a bassa o alta priorità.

Se il bit IPEN invece viene resettato la discriminazione tra alta e bassa priorità viene meno, gli interrupt sono di un unico tipo e vengono tutti dirottati verso la locazione di memoria di indirizzo 0x0008; ogniqualvolta si verifica un evento di interrupt il bit di abilitazione globale GIEL o GIEH viene resettato per evitare immediate successive interruzioni, l'indirizzo di ritorno viene allocato nello stack del microcontrollore mentre l'indirizzo del Program Counter viene caricato con il valore di uno dei due vettori. Dal momento che la routine ISR viene gestita, la sorgente dell'evento può essere determinata effettuando un polling sui bit di Flag; quest'ultimi devono essere resettati in software nella ISR in modo da evitare interrupt ricorsivi. L'istruzione RETFIE consente di uscire dal rispettivo vettore e attiva nuovamente il bit di abilitazione globale GIEL o GIEH.

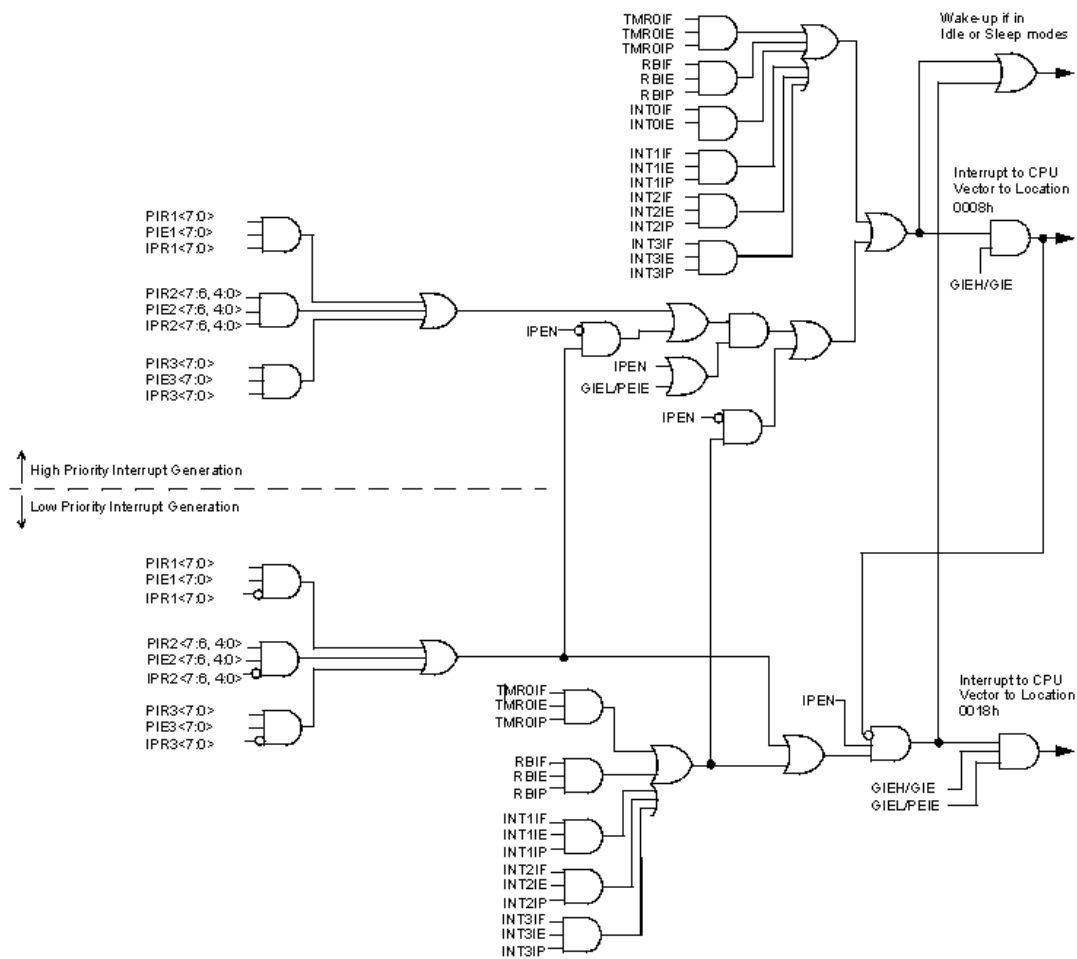


FIG. 7.9 - ARCHITETTURA DEGLI INTERRUPT NEL PIC18LF8722

7.4 Timer 0

Il Timer 0 è uno dei 5 timer messi a disposizione dal microcontrollore per gestire le temporizzazioni di precisione; viene descritto in questo capitolo in quanto è un buon candidato per svolgere la funzione del tick del RTOS.

Le caratteristiche di maggior interesse sono:

- funzionamento a 8 o 16 bit selezionabile via software;
- presenza di un prescaler a 8 bit;
- configurazione della sorgente di clock interna o esterna;
- segnalazione di overflow tramite un evento di interrupt.

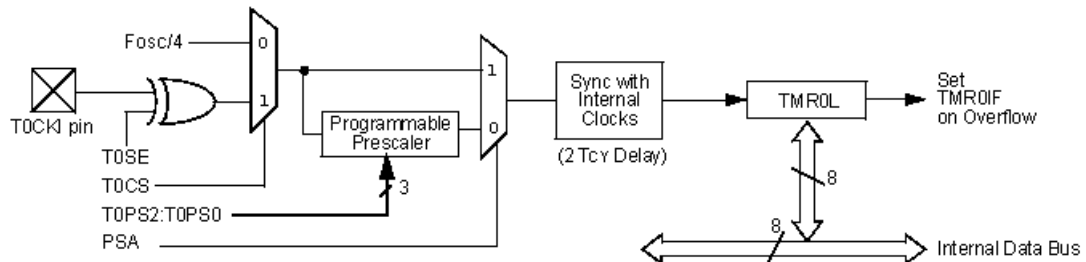


FIG. 7.10 - STRUTTURA DEL TIMER 0 IN CONFIGURAZIONE A 8 BIT

La prima configurazione del Timer 0 è in modalità 8 bit; nel contesto del tick del RTOS di controllo del satellite, la funzione che deve svolgere il Timer 0 è esclusivamente quella di timer e non di contatore di eventi, di conseguenza la sorgente che fa incrementare il valore immagazzinato è il clock interno del sistema indicato nello schema come Fosc/4; il prescaler è programmabile per mezzo dei bit TOPS e può essere escluso o meno dall'applicazione a seconda delle esigenze. È presente inoltre un modulo che ha lo scopo di sincronizzare la sorgente con i blocchi funzionali del Timer 0 e che causa un ritardo pari a due istruzioni di clock. Il registro che viene incrementato dal segnale d'ingresso è il TMR0L accessibile dal bus interno del microcontrollore.

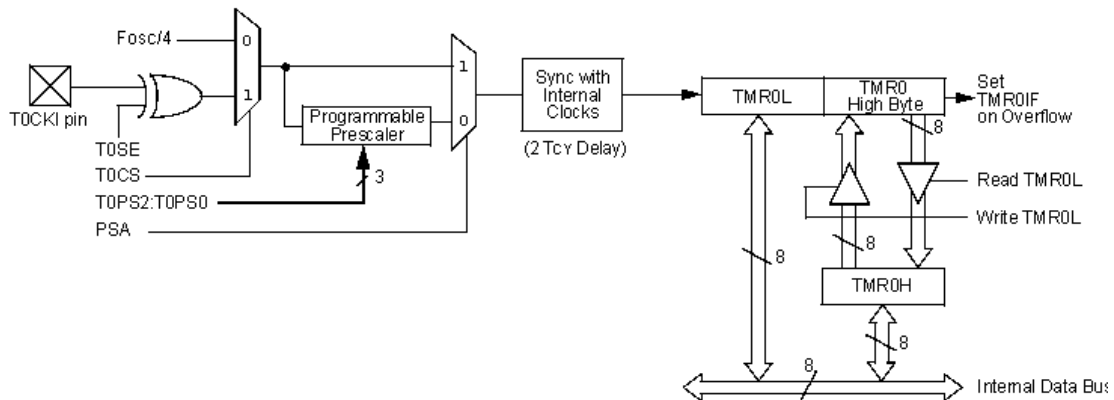


FIG. 7.11 - STRUTTURA DEL TIMER 0 IN CONFIGURAZIONE A 16 BIT

Per quanto riguarda la configurazione a 16 bit, i blocchi funzionali del Timer0 rimangono sostanzialmente gli stessi; la differenza consiste nella presenza di un registro di buffer denominato TMR0 High Byte accessibile via software per mezzo del registro TMR0H.

Quando si effettua una lettura sul registro TMR0L contemporaneamente il valore del registro TMR0H viene aggiornato con il valore contenuto nel buffer TMR0 High Byte; quando invece si effettua una scrittura nel registro TMR0L il valore del buffer TMR0 High Byte viene aggiornato con il valore del registro TMR0H. Questa modalità consente di gestire dati a 16 bit in un'unica istruzione.

Tutte le configurazioni del Timer 0 sono possibili per mezzo del registro di controllo T0CON.

T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7				bit 0			

FIG. 7.12 - REGISTRO DI CONFIGURAZIONE DEL TIMER 0

7.5 *Struttura di ARTOS (Atmocube Real Time Operating System)*

In questo paragrafo viene delineata ad alto livello l'architettura del software per il controllo del satellite Atmocube; seguendo la struttura di tipo multitasking vengono creati diversi task ciascuno con delle funzioni di base. La struttura analizzata comprende il nucleo del sistema operativo, le applicazioni di avvio del kernel ed i task.

I task creati sono 7:

- **CHARGE MODE()**: gestisce completamente la carica della batteria; si occupa di gestire i primi istanti di vita del satellite quando ancora l'energia accumulata è minima, la fase intermedia e la fase finale in cui la batteria è completamente carica;
- **TEST()**: è un task che verifica il funzionamento di tutti sottosistemi di Atmocube, specialmente il payload installato; fornisce di conseguenza informazioni sullo stato del satellite permettendo di conoscere possibili guasti in modo da intervenire ed eventualmente escludere qualche sottosistema;
- **SAFE MODE()**: è la modalità sicura che il satellite deve avere la possibilità di attivare; dalle norme stabilite dall'agenzia spaziale ESA è necessario avere la possibilità di porre il satellite in uno stato dormiente privo di misurazioni e soprattutto trasmissioni radio, onde evitare disturbi ed interferenze ad altri satelliti in orbita. Il ricevitore in ogni caso deve rimanere abilitato in modo da ripristinare lo stato ordinario di funzionamento;
- **MEASURE()**: con questo task si gestisce l'intero payload effettuando tutte le misure sperimentali e immagazzinando i dati raccolti; esso garantisce l'obiettivo primario del satellite ed è una risorsa fondamentale per l'intero sistema;
- **OBR()**: quando sono stati raccolti tutti i dati e si è ad una distanza adeguata dalla stazione terrestre, questo task instaura la comunicazione radio al fine di trasmettere e riceve tutte le informazioni ed i dati; tutto ciò avviene controllando opportunamente il modulo radio OBR;
- **IDLE()**: In qualsiasi momento in cui non sia necessario effettuare alcuna misura o trasmissione radio, il satellite può portarsi in una modalità di attesa a basso consumo risparmiando energia;
- **WATCHDOG()**: si reputa necessario implementare un sistema di sicurezza in modo da evitare possibili situazioni di stallo del software. Qualora si dovesse verificare un blocco del sistema causato da un possibile guasto, un bug di qualche task, un malfunzionamento del microcontrollore o dell'oscillatore principale, si prevede l'intervento del timer watchdog del microcontrollore per porre in reset l'intero modulo OBDH & HK. Questa funzione viene implementata sotto forma di task ed è caratteristica nei sistemi di tipo RTOS.

7.5.1 Assegnazione delle priorità

La figura 7.13 illustra l'architettura del sistema operativo implementato; è presente il μ Kernel con tutti task disposti in forma circolare. Tra essi compare anche uno denominato STARTUP: in verità non si tratta di un task, ma di un applicativo indispensabile per inizializzare il μ Kernel e avviare il microcontrollore.

Si passa ora quindi all'assegnazione delle varie priorità per consentire al RTOS di gestire adeguatamente il sistema; dalla figura si nota che compaiono dei task con priorità uguali: se lo stato è posto per entrambi in condizione di READY l'esecuzione avviene parallelamente, ovvero passando di continuo da uno all'altro.

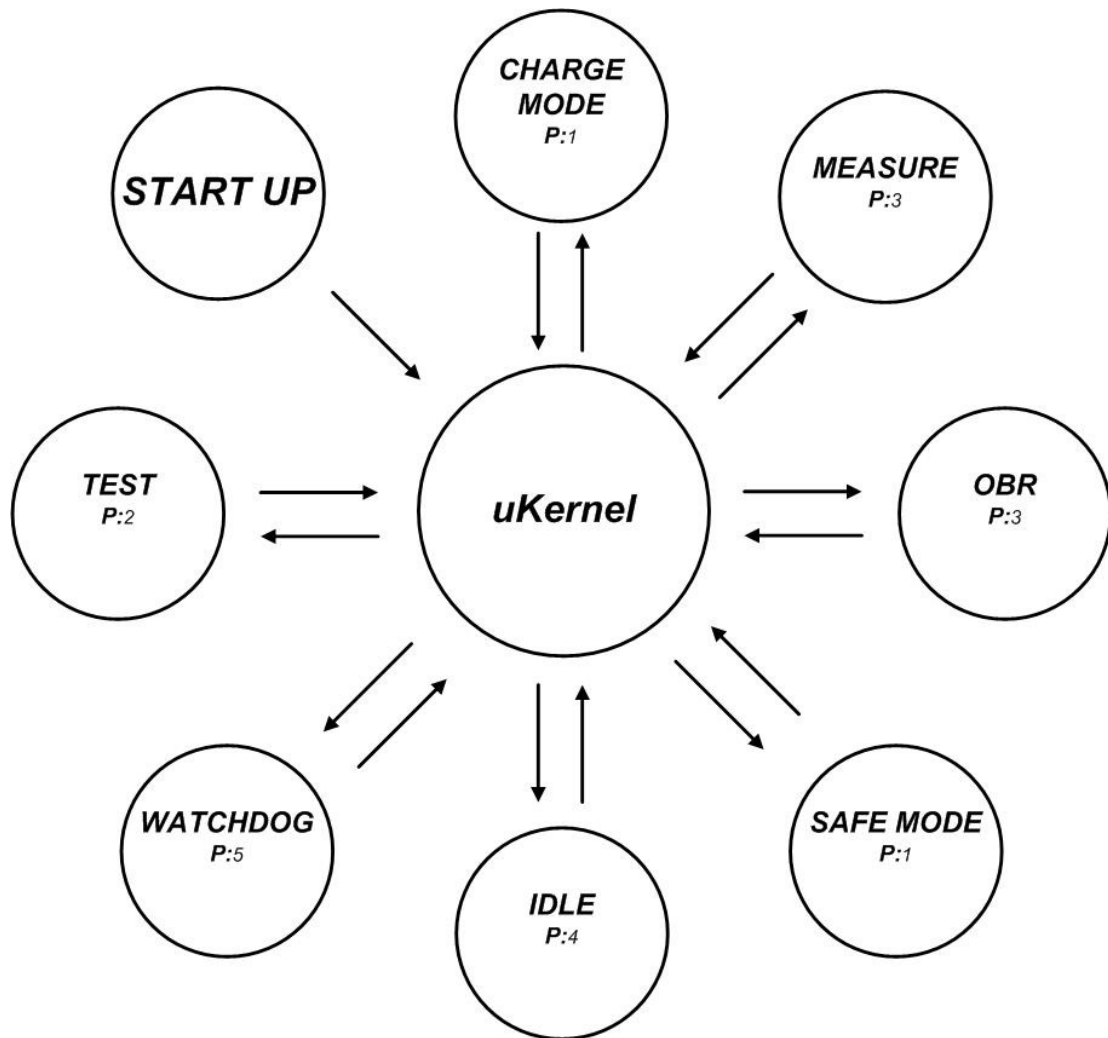


FIG. 7.13 - STRUTTURA DEL RTOS DI ATMOCUBE

Si parte dalla priorità più elevata:

- **PRIORITA' 1:** viene condivisa tra lo stato di carica della batteria **CHARGE MODE()** e la modalità sicura **SAFE MODE()**. Lo stato di carica è essenziale per garantire energia al sistema quindi deve essere gestito prima di qualsiasi altra operazione. La modalità sicura, in caso venga attivata, è l'unica in funzione assieme alla carica della batteria e non permette ad alcun altro task di subentrare senza un opportuno comando da parte della stazione di terra;
- **PRIORITA' 2:** successivamente si procede a svolgere un'attività di test sull'intero sistema con l'ausilio del task **TEST()**; si verifica lo stato del satellite e l'operatività dei vari moduli e del payload;
- **PRIORITA' 3:** solo a questo punto il satellite è pronto a iniziare le operazioni per cui è stato progettato; questa priorità viene assegnata sia al task di misura **MEASURE()** sia al task della comunicazione radio con la stazione di terra **OBR()**;
- **PRIORITA' 4:** qualora non si effettuino misure o comunicazioni radio si passa successivamente nella modalità di attesa a basso consumo passando il controllo al task **IDLE()**;
- **PRIORITA' 5:** l'ultima priorità viene assegnata al task **WATCHDOG()**; si tratta di un task che rimane attivo in sottofondo al sistema e viene comunque preso in considerazione dal sistema operativo; il tick di sistema provvede ad aggiornare periodicamente il watchdog evitando di porre il microcontrollore in uno stato di reset. La condizione di reset si verifica soltanto se il software si blocca da qualche parte del codice e non permette al tick di intervenire su questo contatore.

7.5.2 Procedura di avvio

Si analizza ora la struttura del software nella condizione particolare di avvio; ad ogni task, oltre alle priorità assegnate, viene assegnato uno stato iniziale. I task colorati rappresentano quelli che contribuiscono a svolgere le funzioni di avvio del satellite; il watchdog è anche attivo ma non prende parte alla normale procedura.

Il software avvia l'applicativo di start up e configura il μ Kernel; successivamente viene attivato il task di carica che provvede ad accumulare energia sufficiente per le misure e le comunicazioni; il task **CHARGE MODE()** avendo la priorità più alta ed essendo attivo si porta subito in una condizione di **RUNNING**, cioè di esecuzione; la modalità sicura, che presenta la medesima priorità non interagisce in quanto il suo stato iniziale è stato posto in una condizione di **SUSPENDED**.

Una volta che il task **CHARGE MODE()** rileva la carica completa della batteria, per esempio tramite l'evento di interrupt segnalato dalla periferica HLVD, esso si pone in uno stato di **WAITING**. L'evento che questo task aspetta successivamente è l'interrupt che segnala la batteria scarica per mezzo sempre dalla periferica HLVD configurata però sulla soglia opposta.

A questo punto è il task **TEST()** che viene eseguito dal μ Kernel in quanto è quello che possiede una priorità più alta con uno stato attivo di tipo **READY**. Quando tutte le procedure di test vengono eseguite il task **TEST()** si pone in stato di **SUSPENDED** permettendo l'esecuzione del task **IDLE()**.

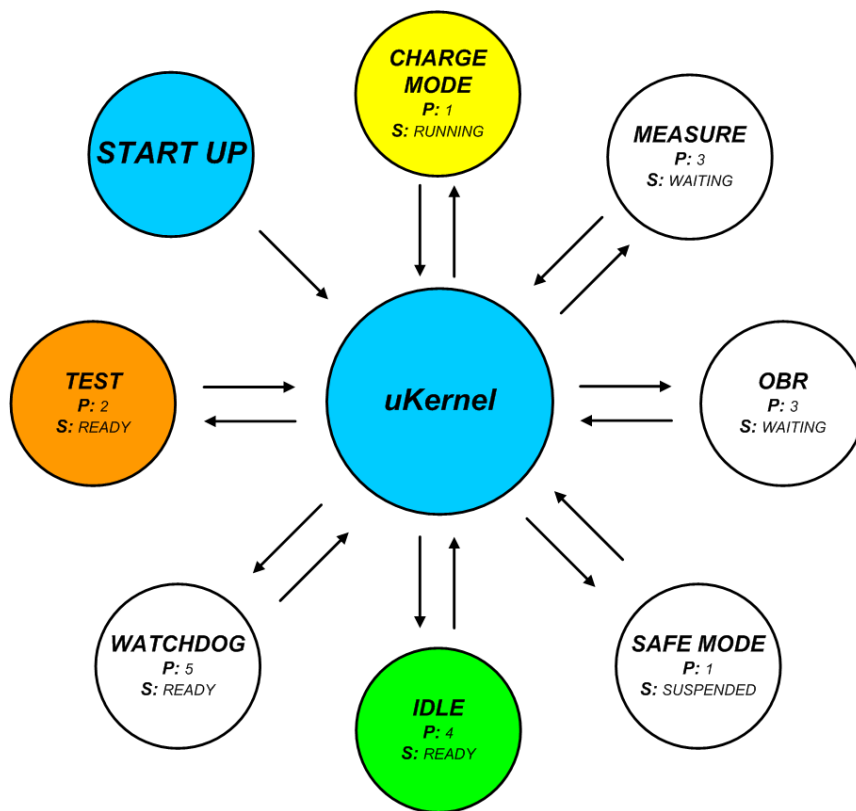


FIG. 7.14 - AVVIO E PRIME OPERAZIONI DI ATMOCUBE

7.5.3 Procedure di regime e misura

Si descrive ora la fase di regime quando l'accumulatore è completamente carico e si intendono svolgere le funzioni che garantiscono gli obiettivi della missione.

Ora il task attivo a priorità più alta è il **MEASURE()**; si possono quindi effettuare tutte le misure scientifiche. Il task **OBR()** si trova in uno stato iniziale di **WAITING** in quanto le comunicazioni radio avvengono solamente quando il satellite si trova ad una distanza adeguata dalla stazione di terra. L'evento che porta questo task da uno stato di **WAITING** ad uno stato di **READY** è fornito dal GPS che segnala il raggiungimento della corretta posizione; esso implica una gestione di tipo Round Robin tra il task **MEASURE()** e **OBR()** vista la medesima priorità assegnata. Lo stato di **IDLE()** è attivo e subentra solo quando i task precedenti hanno completato le loro funzioni.

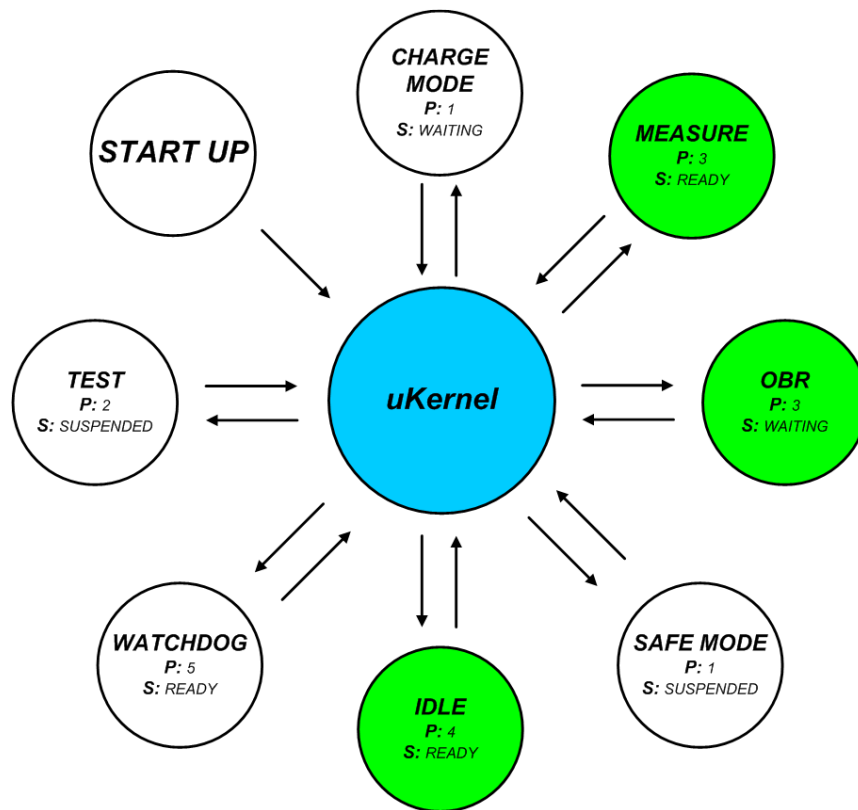


FIG. 7.15 - MISURE E COMUNICAZIONI RADIO

7.5.4 Procedura in modalità di sicurezza

Per entrare in questa modalità bisogna ricevere un comando specifico dalla stazione di terra. Il task **SAFE MODE()** viene portato da uno stato di **SUSPENDED** ad uno stato di **READY**; in tal modo, avendo la priorità più elevata, esso assieme al task **CHARGE MODE()** prende il controllo totale del satellite escludendo del tutto le altre funzioni. La politica di gestione tra il **CHARGE MODE()** e il **SAFE MODE()** è ancora di tipo Round Robin vista la priorità di entrambi pari a 1.

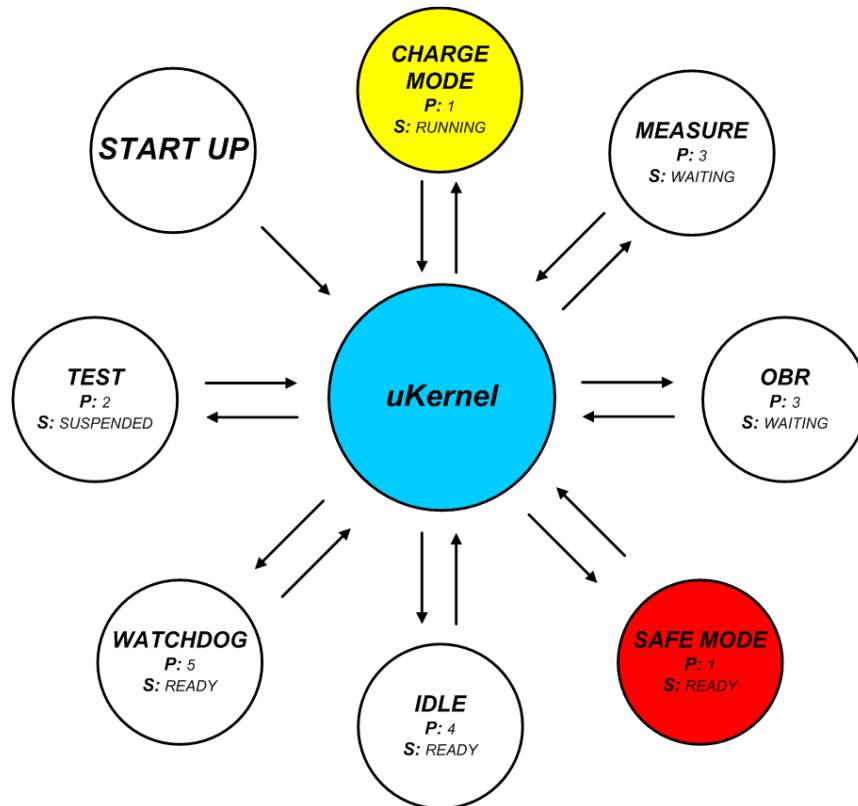


FIG. 7.16 - GESTIONE DELLA MODALITA' SICURA

7.5.5 *Stack software*

Si è visto che in un sistema operativo ogni task possiede il suo contesto; esso si compone sostanzialmente dei registri di stato della CPU, dei registri di controllo delle periferiche, dello stack hardware e dei registri di stoccaggio delle variabili locali; tutte queste informazioni vengono accorpate in un ulteriore stack di tipo software messo a disposizione del Context Switching.

Quello che si vuole fornire in questo paragrafo è un'indicazione preliminare di come organizzare la struttura della pila software e di che parte della memoria del microcontrollore dedicarle; si cercano inoltre di fornire delle informazioni riguardo la mole di dati coinvolta e di come distribuirla per ogni singolo task creato.

Un primo approccio per creare un ambiente di Context Switching è dato dalla possibilità di suddividere la pila hardware in modo da dedicare un suo sottoinsieme ad ogni task; di fatto questa soluzione è valida se la pila hardware è di dimensioni molto grandi, l'applicazione coinvolge pochi task e non ci sono molti livelli di chiamate a sottoprogrammi. Dato che lo stack hardware del PIC18LF8722 è a 31 livelli e i task coinvolti sono 7 si decide di implementare tutto lo stack software nella memoria ram interna del microcontrollore.

Il Context Switching e le pile software vengono organizzate nel modo seguente:

- si salva per ogni task il contenuto della pila hardware considerando tutti i 31 livelli. Dato che è possibile accedere a questi dati tramite i registri TOSU, TOSH e TOSL si creano altrettanti registri TOSUx, TOSHx e TOSLx per ogni livello della pila e per ogni task, dove "x" indica il rispettivo livello.
Il budget di memoria ram utilizzata è di 93 bytes per il contenuto di ogni pila hardware di ciascun task; in totale sono necessari 651 bytes.
- si salvano i registri di manipolazione della pila, dello stato della CPU, degli interrupt, della periferica MSSP1, del convertitore ADC, della periferica HLVD e delle periferiche EUSART.
I registri coinvolti sono: **STKPTR, STATUS, WREG, INTCON, INTCON2, PIR1, PIR2, PIE1, PIE2, IPR1, IPR2, RCON, SSP1STAT, SSP1CON1, ADCON0, ADCON1, ADCON2, HLVDCON, TXSTA1, RCSTA1, BAUDCON1, TXSTA2, RCSTA2, BAUDCON2**.
Il budget di memoria ram utilizzata è pari a 24 bytes per ogni task; in totale sono necessari 168 bytes.
- si riserva dello spazio per dei registri di uso generico e per lo stoccaggio delle variabili locali di ogni task. Il budget in questo momento non viene definito ma trova ampio spazio nella memoria rimasta.

Questa appena fornito è un calcolo non definitivo; in ogni caso vengono considerati tutti i registri impiegati nell'hardware e nel software coinvolto e viene considerata l'occupazione di memoria più alta possibile in modo da riflettere la condizione peggiore.

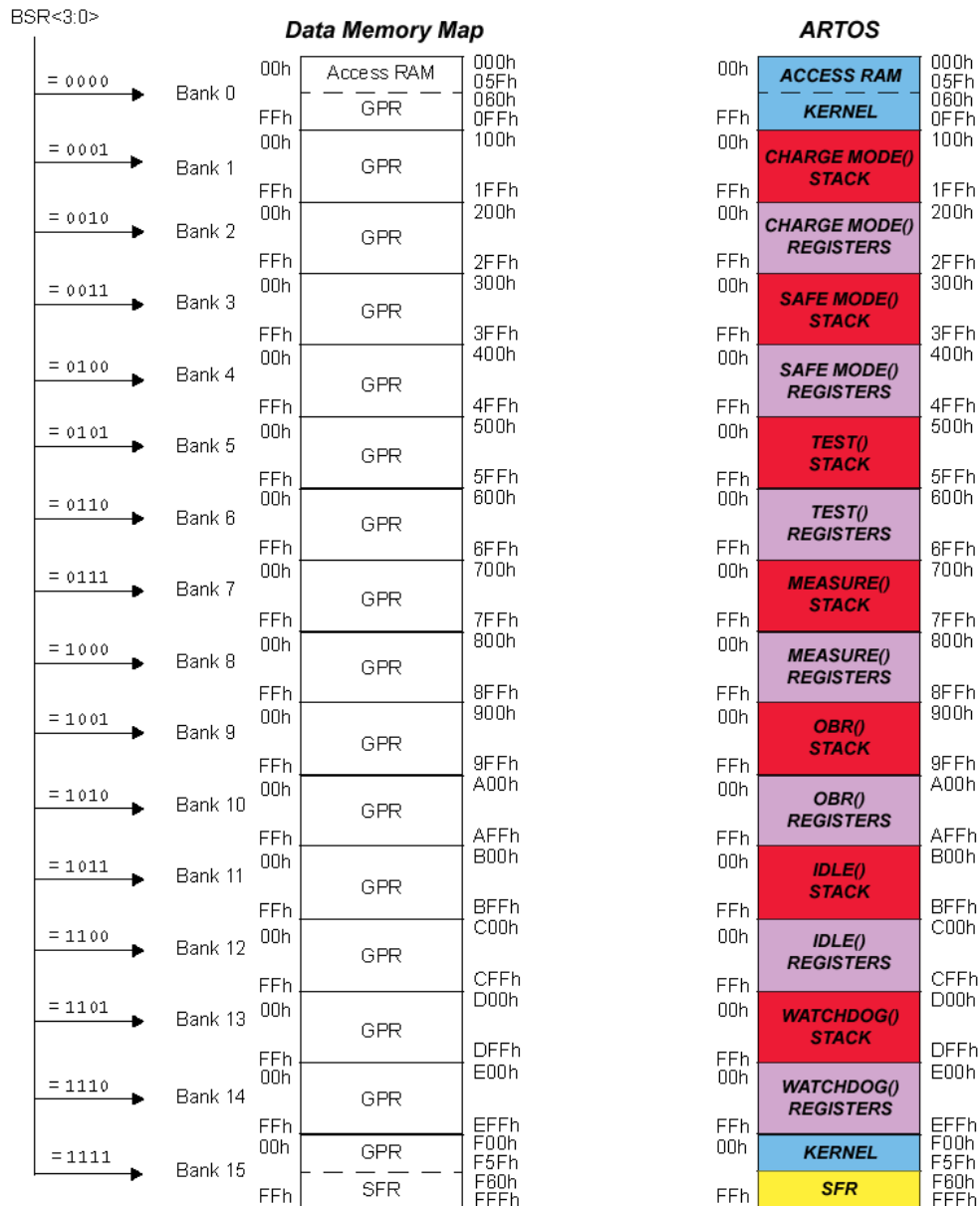


FIG. 7.17 – INSEDIAMENTO DI ARTOS NELLA MEMORIA RAM DEL MICROCONTROLLORE

La capacità di 4 kbytes e la disposizione a 16 banchi della memoria ram si rendono adeguati per organizzare le pile software come in figura 7.17.

Ad ogni banco di memoria è stato assegnata una particolare funzione; i blocchi in rosso contengono i dati salvati dallo stack hardware per ogni task; i blocchi in viola contengono le copie dei registri delle variabili locali e dei registri Special Function Register utilizzati, cioè i registri di manipolazione della pila, dello stato della CPU e delle periferiche coinvolte; il blocco in giallo è l'area di memoria riservata agli Special Function Register, cioè quella delle periferiche attive in un certo contesto; infine i blocchi in azzurro sono riservati al kernel e all'applicativo di avvio.

Ciascun blocco è ampio 256 bytes ed è molto più ampio delle specifiche iniziali per la creazione dello stack software; a tal riguardo si potrebbero accorpate i blocchi rossi e viola dedicando un singolo banco ad ogni task e liberando della memoria per il kernel.

Riassumendo la memoria ram viene così gestita e strutturata:

- ad ogni copia dello stack hardware di ciascun task viene riservato un banco di 256 bytes;
- ad ogni copia degli Special Function Register di ciascun task e delle variabili locali viene riservato un ulteriore banco di 256 bytes; sono disponibili quindi per le variabili locali più di 200 bytes;
- i banchi appartenenti al medesimo task sono stati resi adiacenti nello spazio di memoria;
- al kernel si riserva il primo blocco di memoria, tra cui l'Access Ram per l'accesso veloce, e parte dell'ultimo banco per un totale di 352 bytes.

Conclusioni

Il modulo OBDH & HK progettato e documentato in questo elaborato vuole essere un prototipo su cui basare completamente il circuito finale da installare sul satellite Atmocube. Infatti le specifiche iniziali di progetto risalgono a meno di un anno fa e comprendono tutte le modifiche più recenti effettuate ai vari sottosistemi del satellite.

L'unica modifica certa riguarda l'oscillatore primario; invece di utilizzare un semplice quarzo è consigliabile impiegare un oscillatore compensato in modo da garantire l'avvio in tutte le condizioni di temperatura. La frequenza di clock può essere aumentata ma bisogna considerare l'aumento di assorbimento di corrente.

Non essendo ancora definito l'alimentatore completo del satellite può essere necessaria qualche modificata alle linee di controllo della PSU. A tal proposito si è voluto intenzionalmente progettare un alimentatore per il modulo OBDH & HK in modo da accelerare in parte questo punto critico di avanzamento del progetto e allo stesso tempo rendere indipendente il sistema primario di controllo.

Per quanto riguarda il software è necessario sviluppare completamente il μ Kernel e i singoli task di controllo; risulta valido attenersi al lavoro svolto nel capitolo 7, per quanto riguarda l'algoritmo ad alto livello, e al lavoro del capitolo 6, per quanto concerne lo sviluppo dei singoli task e l'implementazione dei driver.

Ringraziamenti

Ci sono molte persone che desidero ringraziare; proprio per questo motivo non seguirò un ordine preciso ma un po' casuale a seconda della mia spontaneità e di come mi scorrono i pensieri in questo momento senza dar importanza alcuna all'ordine di citazione.

Ci tengo a ringraziare il Prof. Mario Fragiaco per avermi dato la possibilità di entrare a far parte del team di Atmocube e per avermi dato fiducia dall'inizio alla fine sullo sviluppo del progetto. Voglio ringraziarlo inoltre per l'instancabile disponibilità, impeccabile in questi mesi a partire dall'estate scorsa, e per l'opportunità concessami ad intraprendere la borsa di studio in azienda che ancora adesso sto svolgendo.

Ringrazio il Prof. Sergio Carrato per essere stato sempre molto attivo come coordinatore del mio corso di Laurea risolvendo con immediata risolutezza qualsiasi piccolo intoppo che si è presentato lungo questo cammino. Lo ringrazio inoltre per essere il mio relatore nella discussione della tesi e per i preziosi consigli riguardo l'impostazione della stessa.

Ringrazio tutta l'Enteos e la MyWave per avermi ospitato nei mesi dello svolgimento della tesi e per continuare ad ospitarmi nei mesi della borsa di studio. Un particolare grazie a Livio per gli instancabili consigli e aiuti sulla teoria e sul progetto dell'ultimo esame che mi ha permesso di consegnare il libretto in segreteria. Ringrazio William per le infinite risposte alle mie perplessità sulla parte hardware del progetto di tesi. Ringrazio Roberto per il contributo sullo sviluppo software in C, per le immancabili e raffinate indentazioni, per i caffè e per le risate. Ringrazio tutti gli altri colleghi, in particolare i due Francesco e Stefano, per qualsiasi momento positivo condiviso tra i vari uffici e in qualsiasi pausa.

Ringrazio gli amici della Elcon per avermi permesso di svolgere il tirocinio in azienda la scorsa estate; in particolare il Sig. Generali per le dritte riguardo il sistema di carica e Marco per aver risistemato lo schema elettrico del progetto di tesi.

Ringrazio tutti i membri del team di Atmocube per le ricorrenti "riunioni" e per qualsiasi altra cosa: saluto Alessandro, Marco e Veronica, sempre presenti.

Ringrazio i miei fedeli e insostituibili compagni di studio Riccardo e Michele per gli indimenticabili momenti passati e condivisi in questi anni da studenti dentro e fuori la biblioteca generale. Resteranno sempre impressi i lunghi banchi neri completamente pieni di scritte, i libri e i fogli sparsi, l'unica pagina del libro letta in tutto il giorno, le solite facce di ogni giorno, le persone conosciute, gli immancabili commenti, le irriducibili risate, le infinite pause, gli indispensabili caffè da Sandro e Alessia, i pranzi sui gradini di Piazzale Europa allo sbucare della bella stagione, tutte le serate, tutti i ritrovi, tutti i divertimenti. Che altro? Tanto ancora sicuramente. GRAZIE AMICI!

Ringrazio il mio amico Lorenzo per le superbe vacanze estive insieme, per un'amicizia unica sicuramente rara, per le continue chiamate al cell, per i messaggi ironici, per i messaggi malinconici, per i messaggi giusti, per i semplici messaggi di saluto, per le speranze, per la presunzione, per i sogni passati, per i sogni futuri. Ah sì, ...per le corse della domenica.

Ringrazio i miei genitori per avermi permesso di portare avanti i miei studi in completa serenità e tranquillità, per avermi sostenuto pienamente, per aver creduto sempre nelle mie capacità e potenzialità, per avermi dato fiducia su tutto, per avermi permesso di superare qualsiasi difficoltà, per avermi trasmesso i sani principi e valori! Ringrazio in particolare mio papà per avermi fatto conoscere lo sport, essenziale strumento di sopravvivenza e sfogo in questi anni, e mia mamma per la pazienza di prepararmi le cene a ogni orario possibile della sera dopo gli allenamenti.

Ringrazio i miei nonni, che con ansia attendono il compiersi di questo evento (e questo ...vale anche per me), per la loro insostituibile e immancabile presenza in tante occasioni a cominciare da quando ero piccolo, per l'essere sempre informati su di me, per le generose paghe mensili, per ancora, sì per ancora!

Bibliografia

- Microchip (2004) *PIC18F8722 Data Sheet*, Microchip Technology Inc.
- Microchip (2005) *MPLAB C18 C COMPILER LIBRARIES*, Microchip Technology Inc.
- Microchip (2005) *MPLAB C18 C COMPILER USER'S GUIDE*, Microchip Technology Inc.
- Microchip (2008) *25LC1024 Data Sheet*, Microchip Technology Inc.
- Microchip (1998) *AN686 Understanding and Using Supervisory Circuits*, Microchip Technology Inc.
- Valeanu, A. (2008) *AN1193 Basic PICmicro Oscillator Design*, Microchip Technology Inc.
- Duane, B. (2004) *AN949 Making Your Oscillator Work*, Microchip Technology Inc.
- Lourens, R. (2004) *AN943 Practical PICmicro Oscillator Analysis and Design*, Microchip Technology Inc.
- Farmer, J. (1997) *AN585 A Real – Time Operating System for PICmicro Microcontrollers*, Microchip Technology Inc.
- Valenti, C. (2001) *AN777 Multi – Tasking on the PIC16F877 with the Salvo RTOS*, Microchip Technology Inc.
- Fosler, R. M. (1997) *AN818 Manipulating the Stack of the PIC18 Microcontroller*, Microchip Technology Inc.
- Rovnak, T. (2003) *External Memory Interfacing Techniques for the PIC18FXXX*, Microchip Technology Inc.
- Matthews, D. (2002) *AN849 Using C to interface 8051 MCU with SPI Serial EEPROMs*, Microchip Technology Inc.
- Cypress (2003) *CY62167DV30*, Cypress Semiconductor Corporation.
- TI (2003) *CD74HC126 datasheet*, Texas Instruments Incorporated.
- TI (2003) *CD74HC238 datasheet*, Texas Instruments Incorporated.
- TI (2003) *CD74HC573 datasheet*, Texas Instruments Incorporated.
- TI (2008) *CC1100 datasheet*, Texas Instruments Incorporated.
- ANALOG DEVICES (2007) *AD7685 datasheet*, Analog Devices, Inc.
- ANALOG DEVICES (2002) *ADG706 datasheet*, Analog Devices, Inc.
- Intersil (2006) *ISL88011 Data Sheet*, Intersil Americas Inc.
- Micrel (2006) *MIC5205 datasheet*, Micrel Inc.
- MAXIM (2007) *MAX3232 datasheet*, Maxim Integrated Products Inc.

MAXIM (2007) *MAX3490 datasheet*, Maxim Integrated Products Inc.

MAXIM (1994) *MAX662A datasheet*, Maxim Integrated Products Inc.

MA – COM (2007) *MASWSS0143V4 datasheet*, MA – COM Technology Solutions.

RFMD (2001) *RF2436 datasheet*, RF Micro Devices.

IR (2008) *IRLML6401 datasheet*, International Rectifier

Intel (1990) *8086 datasheet*, Intel Corporation.

Intel (1990) *8088 datasheet*, Intel Corporation

Pragmatec (2005) PICos18 Sistema in real time per PIC18,
<http://www.picos18.com/index_it.htm>

Honeywell (2004) *HMC2003 datasheet*, Honeywell International Inc.

Surrey (2006) *SGR-05 Series User Interface Manual*, Surrey Satellite Technology Ltd

SAFT (2005) *MP174865 datasheet*, Saft Speciality Battery Group.

Renesas (2006) *H8 / 38076R Group datasheet*, Renesas Technology Corp.

Barry, R. (2009) *FreeRTOS Designed for Microcontrollers*, < <http://www.freertos.org>>.

Agilent (2000) *Mixed Signal Oscilloscope 54622D 100 MHz, 200 MSa/s*, Agilent Technologies.

Appendice A1

Codice C di test, codice C dei driver e registri del microcontrollore

```

// -----
// File      : TestOBDAH.c
//
// Synopsis  : Implementation of basic functionalities to test
//              the AtmoCube control board: UART and SPI
//
// Author    : Stefano Punis
// -----

#include "prova.h"

#pragma config OSC = HS      // Hi-speed oscillator
#pragma config WDT = OFF     // Watchdog off
#pragma config LVP = OFF     // Low voltage programming
#pragma config MODE = MC     // Microcontroller mode
#pragma config MCLRE = ON    // Use external reset pin

// -----
// Global variables
// -----

char SerialCommand[MAX_LEN_COMMAND];

// -----
// Serial port interrupt handling
// -----

#pragma code low_vector = 0x18
void low_interrupt (void)
{
    _asm
        goto interrupt232
    _endasm
}
#pragma code

// -----
// Command parser
// -----

#pragma interruptlow interrupt232
void interrupt232 (void)
{
    char temp;
    char RxChar; // Received char
    char Answer2Command[MAX_LEN_ANSWER];
    char Token[MAX_LEN_COMMAND];
    char *StrtokResult = NULL;
    unsigned char i = 0;

    RxChar = getc1USART();

    // If a CR is received...
    if ((RxChar == '\n' ||
        RxChar == '\r') )
    {
        // If the buffer is empty, simply ignore and answer with a prompt..
        if ( SerialCommand[0] == '\0' )
        {
            puts1USART ( NEWLINE );
            puts1USART ( PROMPT );
            return;
        }
        else
        {

```

```

//...otherwise check if the command can be handled
if (strcmp (strlwr(SerialCommand), Command_Id ) == 0)
{
    puts1USART ( NEWLINE );
    puts1USART ( DeviceId );
}
else if (strcmp (strlwr(SerialCommand), Command_Ver ) == 0)
{
    puts1USART ( NEWLINE );
    puts1USART ( VersionNumber );
}
else if (strcmp (strlwr(SerialCommand), Command_Reset ) == 0)
{
    _asm
    reset
    _endasm
}
else if (strcmp (strlwr(SerialCommand), Command_SpiOpen ) == 0)
{
    OpensPI1 ( SPI_FOSC_64, // 62.5 kHz
               MODE_00,    // Mode 00: CKP = 0, CKE = 1
               SMPEND );   // Inp. data sample at end of data

    puts1USART ( "\n\rSPI bus now open" );
}
else if (strcmp (strlwr(SerialCommand), Command_SpiClose ) == 0)
{
    CloseSPI1 ();
    puts1USART ( "\n\rSPI bus now closed" );
}
else if ( strstr (SerialCommand, Command_SpiTest ) != NULL )
{
    if ( SSP1CON1bits.SSPEN == 0 )
    {
        puts1USART ( "\n\rERROR: SPI bus closed" );
    }
    else
    {
        WriteSPI1 ( TestByte );
        puts1USART ( "\n\rOK" );
    }
}
else
{
    puts1USART ( "\n\rERROR" );
}

puts1USART ( NEWLINE );
puts1USART ( PROMPT );

// Reset the command line buffer
SerialCommand[0] = '\0';
}
}
else
{
    // Any non CR or LF char received is appended to the buffer
    i = strlen (SerialCommand);
    SerialCommand[i] = RxChar;
    SerialCommand[i+1] = '\0';

    // Echo the received char [ToDo: maybe echo could be disabled...]
    Write1USART (RxChar);
}

temp = RCREG;
}

```

```

void main (void)
{
    // Initialize the interrupt registers
    INTCON = 0;
    INTCON2 = 0;
    INTCON3 = 0;
    PIR1 = 0;

    RCONbits.IPEN = 1;        // Differentiate between hi and low prio.
    PIE1bits.RC1IE = 1;      // Enable USART1 interrupt flag
    IPR1bits.RC1IP = 0;      // Assign low prio interrupt to the serial port
    INTCONbits.GIEH = 1;     // Enable global interrupt (high priority)
    INTCONbits.GIEL = 1;     // Enable global interrupt (low priority)

    // -----
    // Configure USART 1
    // -----

    Open1USART ( USART_TX_INT_OFF & // Transmit interrupt OFF
                 USART_RX_INT_ON  & // Receive interrupt ON
                 USART_ASYNC_MODE & // Asynchronous Mode
                 USART_EIGHT_BIT  & // 8-bit transmit/receive mode
                 USART_CONT_RX    & // Continuous reception
                 USART_BRGH_LOW,  // Low baud rate
                 25,              // Baud rate value: 2400 Baud
                 );

    // Reset the command buffer
    SerialCommand[0] = '\0';

    // -----
    // When device is booting, send ID and SW version number
    // on the serial port
    // -----

    puts1USART ( NEWLINE );
    puts1USART ( DeviceId );
    puts1USART ( NEWLINE );
    puts1USART ( VersionNumber );
    puts1USART ( NEWLINE );
    puts1USART ( PROMPT );

    while(1)
    {
        // Idle loop
    }
}

```

```

// -----
// File      : DRIVER.c
//
// Synopsis  : Implementation of basic functionalities to test
//              the AtmoCube control board OBDH & HK
//
// Author    : Stefano Punis
// -----

// -----
// Preprocessor instructions
// -----

#pragma config OSC      = HS           // Hi-speed oscillator
#pragma config WDT      = OFF          // Watchdog off
#pragma config LVP      = OFF          // Low voltage programming
#pragma config MODE     = EM           // Microcontroller mode
#pragma config MCLRE    = ON           // Use external reset pin
#pragma config ADDR20   = ADDR20BIT    // 20-bit Address Bus
#pragma config DATA16  = DATA16BIT   // 16-bit External Bus mode
#pragma config WAIT     = OFF          // Wait selections unavailable
                                           // for table read and write

// -----
// Write data into RAM
// INPUT: unsigned char Data, data to be written into memory
//        far rom unsigned char, address in RAM
// -----

void writeEMI8 ( unsigned char Data, far rom unsigned char *Address )
{
    *Address = Data;
}

// -----
// Read data into RAM
// INPUT: far rom unsigned char, address in RAM
// OUTPUT: unsigned char, data read from RAM
// -----

unsigned char readEMI8 (far rom unsigned char *Address)
{
    return *Address;
}

// -----
// Write data into RAM (Assembler instructions)
// INPUT: unsigned char Data, data to be written into memory
//        unsigned short long , address in RAM
// -----

void writeEMI8_asm ( unsigned char Data, unsigned short long Address )
{
    TBLPTR = Address;           // Load address into Table Pointer
    TABLAT = Data;             // Load data into Table Latch
    _asm TBLWT _endasm         // Write
}

```

```

// -----
// Read data into RAM (Assembler instructions)
// INPUT: unsigned short long, address in RAM
// OUTPUT: unsigned char, data read from RAM
// -----

unsigned char readEMI8_asm (unsigned short long Address)
{
    TBLPTR = Address;          // Load address into Table Pointer
    _asm TBLRD _endasm         // Read data & save into Table Latch
    return TABLAT;             // Return data
}

// -----
// Exchange data in SPI
// OUTPUT: unsigned char, data to be written into bus
// INPUT:  unsigned char, data to be read from bus
// -----

unsigned char SPI (unsigned char Data)
{
    SSP1STAT = 0b11000000;      // SMP = 0, CKE = 1
    SSP1CON1 = 0b00100010;      // MSSP ON, CKP = 0, SCLK= 62,5 KHz
    SSP1BUF = Data;             // Transmit data

    while ( SSP1STATbits.BF == 0 ); // Wait for end of exchange

    Data = SSP1BUF;              // Save received data
    SSP1CON1bits.SSPEN = 0;      // Turn off MSSP

    return Data;                // Return received data
}

// -----
// Read data from UART1
// OUTPUT: unsigned char , data read from UART1
// -----

unsigned char readUART1 ( )
{
    unsigned char Data;

    BAUDCON1 = 0b00000000;      // 8 bit Baud Rate Generator
    SPBRG1 = 0b00011001;        // 2400 Baud
    TXSTA1 = 0b00000000;        // Transmission OFF
    RCSTA1 = 0b10010000;        // Continuous RX enable, 8 bit,
                                // Serial Port enable

    while ( PIR1bits.RC1IF == 0 ); // Test interrupt flag
    Data = RCREG1;                // Save data
    RCSTA1bits.SPEN = 0;          // Turn OFF MSSP1

    return Data;                 // Return received value
}

```

```

// -----
// Write data into UART1
// INPUT: unsigned char Data, data to be written into UART1
// -----

void writeUART1 (unsigned char Data)
{
    BAUDCON1 = 0b00000000;           // 8 bit Baud Rate Generator
    SPBRG1   = 0b00011001;           // 2400 Baud
    TXSTA1   = 0b00100000;           // Low Speed, Asynchronous Mode, 8
    bit,                                           //
    RCSTA1   = 0b10000000;           // Transmission ON
                                           // Serial Port enable, Reception OFF

    TXREG1 = Data;                     // Write buffer
    while ( TXSTA1bits.TRMT == 0 );    // Test end of transmission

    RCSTA1bits.SPEN = 0;               // Turn OFF MSSP1
}

// -----
// AD conversion
// OUTPUT: unsigned char, ADRESH
// -----

unsigned char ADC ( )
{
    unsigned char Data;

    ADCON0 = 0b00000000;           // CH0 = RA0, ADC OFF
    ADCON1 = 0b00001110;           // +VREF = AVdd, -VREF = AVss, RA0 =
    Analog
    ADCON2 = 0b00101001;           // LEFT justified, 12 TAD, FOSC/8

    ADCON0bits.ADON = 1;           // ADC = ON
    ADCON0bits.GO = 1;             // Start Conversion

    while ( PIR1bits.ADIF == 0 ); // Wait end of conversion
    PIR1bits.ADIF = 0;             // Reset interrupt flag

    Data = ADRESH;                 // Save result
    ADCON0bits.ADON = 0;           // ADC = OFF

    // Return result of conversion (MSbyte only)
    return Data;
}

```

```

void main (void)
{
    // Port initialization
    PORTA = 0b00000000;
    PORTB = 0b00000000;
    PORTC = 0b00000000;
    PORTD = 0b00000000;
    PORTE = 0b00000000;
    PORTF = 0b00001111;
    PORTG = 0b00000000;
    PORTH = 0b00000000;
    PORTJ = 0b00000000;

    // Tristate configurations
    TRISA = 0b00011111;
    TRISB = 0b11111001;
    TRISC = 0b10010010;
    TRISD = 0b11111111;
    TRISE = 0b11111111;
    TRISF = 0b00000000;
    TRISG = 0b00000100;
    TRISH = 0b00000000;
    TRISJ = 0b11111111;

    ADCON1 = 0b00001110;    // Only RA0 analog
    MEMCON = 0x00110001;    // Bus EMI configuration
}

```

PIC18F8722 FAMILY

7.1 External Memory Bus Control

The operation of the interface is controlled by the MEMCON register (Register 7-1). This register is available in all program memory operating modes except Microcontroller mode. In this mode, the register is disabled and cannot be written to.

The EBDIS bit (MEMCON<7>) controls the operation of the bus and related port functions. Clearing EBDIS enables the interface and disables the I/O functions of the ports, as well as any other functions multiplexed to those pins. Setting the bit enables the I/O ports and other functions but allows the interface to override everything else on the pins when an external memory operation is required. By default, the external bus is always enabled and disables all other I/O.

The operation of the EBDIS bit is also influenced by the program memory mode being used. This is discussed in more detail in **Section 7.4 “Program Memory Modes and the External Memory Bus”**.

The WAIT bits allow for the addition of wait states to external memory operations. The use of these bits is discussed in **Section 7.3 “Wait States”**.

The WM bits select the particular operating mode used when the bus is operating in 16-bit Data Width mode. These are discussed in more detail in **Section 7.5 “16-bit Data Width Modes”**. These bits have no effect when an 8-bit Data Width mode is selected.

REGISTER 7-1: MEMCON: EXTERNAL MEMORY BUS CONTROL REGISTER

R/W-0	U-0	R/W-0	R/W-0	U-0	U-0	R/W-0	R/W-0
EBDIS	—	WAIT1	WAIT0	—	—	WM1	WM0
bit7							bit0

bit 7 **EBDIS:** External Bus Disable bit

- 1 = External bus enabled when microcontroller accesses external memory; otherwise, all external bus drivers are mapped as I/O ports
- 0 = External bus always enabled, I/O ports are disabled

bit 6 **Unimplemented:** Read as ‘0’

bit 5-4 **WAIT1:WAIT0:** Table Reads and Writes Bus Cycle Wait Count bits

- 11 = Table reads and writes will wait 0 Tcy
- 10 = Table reads and writes will wait 1 Tcy
- 01 = Table reads and writes will wait 2 Tcy
- 00 = Table reads and writes will wait 3 Tcy

bit 3-2 **Unimplemented:** Read as ‘0’

bit 1-0 **WM1:WM0:** TBLWT Operation with 16-bit Data Bus Width Select bits

- 1x = Word Write mode: TABLAT0 and TABLAT1 word output, $\overline{WRH}$ active when TABLAT1 written
- 01 = Byte Select mode: TABLAT data copied on both MSB and LSB, $\overline{WRH}$ and $(\overline{UB}$ or $\overline{LB})$ will activate
- 00 = Byte Write mode: TABLAT data copied on both MSB and LSB, $\overline{WRH}$ or $\overline{WRL}$ will activate

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

PIC18F8722 FAMILY

REGISTER 25-4: CONFIG3L: CONFIGURATION REGISTER 3 LOW (BYTE ADDRESS 300004h)⁽¹⁾

R/P-1	R/P-1	R/P-1	R/P-1	U-0	U-0	R/P-1	R/P-1
WAIT	BW	ABW1	ABW0	—	—	PM1	PM0
bit 7				bit 0			

- bit 7 **WAIT:** External Bus Data Wait Enable bit
1 = Wait selections are unavailable for table reads and table writes
0 = Wait selections for table reads and table writes are determined by the WAIT1:WAIT0 bits
- bit 6 **BW:** Data Bus Width Select bit
1 = 16-bit External Bus mode
0 = 8-bit External Bus mode
- bit 5-4 **ABW<1:0>:** Address Bus Width Select bits
11 = 20-bit address bus
10 = 16-bit address bus
01 = 12-bit address bus
00 = 8-bit address bus
- bit 3-2 **Unimplemented:** Read as '0'
- bit 1-0 **PM<1:0>:** Processor Data Memory Mode Select bits
11 = Microcontroller mode
10 = Microprocessor mode
01 = Microprocessor with Boot Block mode
00 = Extended Microcontroller mode

Note 1: This register is unimplemented in PIC18F6527/6622/6627/6722 devices.

Legend:

R = Readable bit	P = Programmable bit	U = Unimplemented bit, read as '0'
-n = Value when device is unprogrammed		u = Unchanged from programmed state

PIC18F8722 FAMILY

19.3.1 REGISTERS

Each MSSP module has four registers for SPI mode operation. These are:

- MSSP Control Register 1 (SSPxCON1)
- MSSP Status Register (SSPxSTAT)
- Serial Receive/Transmit Buffer Register (SSPxBUF)
- MSSP Shift Register (SSPxSR) – Not directly accessible

SSPxCON1 and SSPxSTAT are the control and status registers in SPI mode operation. The SSPxCON1 register is readable and writable. The lower 6 bits of the SSPxSTAT are read-only. The upper two bits of the SSPxSTAT are read/write.

SSPxSR is the shift register used for shifting data in or out. SSPxBUF is the buffer register to which data bytes are written to or read from.

In receive operations, SSPxSR and SSPxBUF together create a double-buffered receiver. When SSPxSR receives a complete byte, it is transferred to SSPxBUF and the SSPxIF interrupt is set.

During transmission, the SSPxBUF is not double-buffered. A write to SSPxBUF will write to both SSPxBUF and SSPxSR.

REGISTER 19-1: SSPxSTAT: MSSPx STATUS REGISTER (SPI™ MODE)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/Ā	P	S	R/W	UA	BF
bit 7							bit 0

- bit 7 **SMP:** Sample bit
SPI Master mode:
 1 = Input data sampled at end of data output time
 0 = Input data sampled at middle of data output time
SPI Slave mode:
 SMP must be cleared when SPI is used in Slave mode.
- bit 6 **CKE:** SPI Clock Select bit
 1 = Transmit occurs on transition from active to Idle clock state
 0 = Transmit occurs on transition from Idle to active clock state
Note: Polarity of clock state is set by the CKP bit (SSPxCON1<4>).
- bit 5 **D/Ā:** Data/Address bit
 Used in I²C mode only.
- bit 4 **P:** Stop bit
 Used in I²C mode only. This bit is cleared when the MSSP module is disabled, SSPEN is cleared.
- bit 3 **S:** Start bit
 Used in I²C mode only.
- bit 2 **R/W:** Read/Write Information bit
 Used in I²C mode only.
- bit 1 **UA:** Update Address bit
 Used in I²C mode only.
- bit 0 **BF:** Buffer Full Status bit (Receive mode only)
 1 = Receive complete, SSPxBUF is full
 0 = Receive not complete, SSPxBUF is empty

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

PIC18F8722 FAMILY

REGISTER 19-2: SSPxCON1: MSSPx CONTROL REGISTER 1 (SPI™ MODE)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0

bit 7

bit 0

- bit 7 **WCOL:** Write Collision Detect bit
 1 = The SSPxBUF register is written while it is still transmitting the previous word (must be cleared in software)
 0 = No collision
- bit 6 **SSPOV:** Receive Overflow Indicator bit
SPI Slave mode:
 1 = A new byte is received while the SSPxBUF register is still holding the previous data. In case of overflow, the data in SSPxSR is lost. Overflow can only occur in Slave mode. The user must read the SSPxBUF, even if only transmitting data, to avoid setting overflow (must be cleared in software).
 0 = No overflow
Note: In Master mode, the overflow bit is not set since each new reception (and transmission) is initiated by writing to the SSPxBUF register.
- bit 5 **SSPEN:** Synchronous Serial Port Enable bit
 1 = Enables serial port and configures SCKx, SDOx, SDIx and $\overline{SSx}$ as serial port pins
 0 = Disables serial port and configures these pins as I/O port pins
Note: When enabled, these pins must be properly configured as input or output.
- bit 4 **CKP:** Clock Polarity Select bit
 1 = Idle state for clock is a high level
 0 = Idle state for clock is a low level
- bit 3-0 **SSPM3:SSPM0:** Synchronous Serial Port Mode Select bits
 0101 = SPI Slave mode, clock = SCKx pin, $\overline{SSx}$ pin control disabled, $\overline{SSx}$ can be used as I/O pin
 0100 = SPI Slave mode, clock = SCKx pin, $\overline{SSx}$ pin control enabled
 0011 = SPI Master mode, clock = TMR2 output/2
 0010 = SPI Master mode, clock = Fosc/64
 0001 = SPI Master mode, clock = Fosc/16
 0000 = SPI Master mode, clock = Fosc/4
Note: Bit combinations not specifically listed here are either reserved or implemented in I²C mode only.

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

PIC18F8722 FAMILY

REGISTER 20-1: TXSTAx: TRANSMIT STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-1	R/W-0
CSRC	TX9	TXEN	SYNC	SENDB	BRGH	TRMT	TX9D
bit 7							bit 0

bit 7 **CSRC:** Clock Source Select bit

Asynchronous mode:

Don't care.

Synchronous mode:

1 = Master mode (clock generated internally from BRG)

0 = Slave mode (clock from external source)

bit 6 **TX9:** 9-bit Transmit Enable bit

1 = Selects 9-bit transmission

0 = Selects 8-bit transmission

bit 5 **TXEN:** Transmit Enable bit

1 = Transmit enabled

0 = Transmit disabled

Note: SREN/CREN overrides TXEN in Sync mode.

bit 4 **SYNC:** EUSART Mode Select bit

1 = Synchronous mode

0 = Asynchronous mode

bit 3 **SENDB:** Send Break Character bit

Asynchronous mode:

1 = Send Sync Break on next transmission (cleared by hardware upon completion)

0 = Sync Break transmission completed

Synchronous mode:

Don't care.

bit 2 **BRGH:** High Baud Rate Select bit

Asynchronous mode:

1 = High speed

0 = Low speed

Synchronous mode:

Unused in this mode.

bit 1 **TRMT:** Transmit Shift Register Status bit

1 = TSRx empty

0 = TSRx full

bit 0 **TX9D:** 9th bit of Transmit Data

Can be address/data bit or a parity bit.

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

REGISTER 20-2: RCSTAx: RECEIVE STATUS AND CONTROL REGISTER

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R-0	R-0	R-x
SPEN	RX9	SREN	CREN	ADDEN	FERR	OERR	RX9D
bit 7						bit 0	

- bit 7 **SPEN:** Serial Port Enable bit
 1 = Serial port enabled (configures RXx/DTx and TXx/CKx pins as serial port pins)
 0 = Serial port disabled (held in Reset)
- bit 6 **RX9:** 9-bit Receive Enable bit
 1 = Selects 9-bit reception
 0 = Selects 8-bit reception
- bit 5 **SREN:** Single Receive Enable bit
Asynchronous mode:
 Don't care.
Synchronous mode – Master:
 1 = Enables single receive
 0 = Disables single receive
 This bit is cleared after reception is complete.
Synchronous mode – Slave:
 Don't care.
- bit 4 **CREN:** Continuous Receive Enable bit
Asynchronous mode:
 1 = Enables receiver
 0 = Disables receiver
Synchronous mode:
 1 = Enables continuous receive until enable bit CREN is cleared (CREN overrides SREN)
 0 = Disables continuous receive
- bit 3 **ADDEN:** Address Detect Enable bit
Asynchronous mode 9-bit (RX9 = 1):
 1 = Enables address detection, enables interrupt and loads the receive buffer when RSRx<8> is set
 0 = Disables address detection, all bytes are received and ninth bit can be used as parity bit
Asynchronous mode 9-bit (RX9 = 0):
 Don't care.
- bit 2 **FERR:** Framing Error bit
 1 = Framing error (can be updated by reading RCREGx register and receiving next valid byte)
 0 = No framing error
- bit 1 **OERR:** Overrun Error bit
 1 = Overrun error (can be cleared by clearing bit CREN)
 0 = No overrun error
- bit 0 **RX9D:** 9th bit of Received Data
 This can be address/data bit or a parity bit and must be calculated by user firmware.

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

PIC18F8722 FAMILY

REGISTER 20-3: BAUDCONx: BAUD RATE CONTROL REGISTER

R/W-0	R-1	U-0	R/W-0	R/W-0	U-0	R/W-0	R/W-0
ABDOVF	RCIDL	—	SCKP	BRG16	—	WUE	ABDEN
bit 7							bit 0

- bit 7 **ABDOVF**: Auto-Baud Acquisition Rollover Status bit
 1 = A BRG rollover has occurred during Auto-Baud Rate Detect mode (must be cleared in software)
 0 = No BRG rollover has occurred
- bit 6 **RCIDL**: Receive Operation Idle Status bit
 1 = Receive operation is inactive
 0 = Receive operation is active
- bit 5 **Unimplemented**: Read as '0'
- bit 4 **SCKP**: Synchronous Clock Polarity Select bit
Asynchronous mode:
 Unused in this mode.
Synchronous mode:
 1 = Idle state for clock (CKx) is a high level
 0 = Idle state for clock (CKx) is a low level
- bit 3 **BRG16**: 16-bit Baud Rate Register Enable bit
 1 = 16-bit Baud Rate Generator – SPBRGHx and SPBRGx
 0 = 8-bit Baud Rate Generator – SPBRGx only (Compatible mode), SPBRGHx value ignored
- bit 2 **Unimplemented**: Read as '0'
- bit 1 **WUE**: Wake-up Enable bit
Asynchronous mode:
 1 = EUSART will continue to sample the RXx pin – interrupt generated on falling edge; bit cleared in hardware on following rising edge
 0 = RXx pin not monitored or rising edge detected
Synchronous mode:
 Unused in this mode.
- bit 0 **ABDEN**: Auto-Baud Detect Enable bit
Asynchronous mode:
 1 = Enable baud rate measurement on the next character. Requires reception of a Sync field (55h); cleared in hardware upon completion.
 0 = Baud rate measurement disabled or completed
Synchronous mode:
 Unused in this mode.

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

21.0 10-BIT ANALOG-TO-DIGITAL CONVERTER (A/D) MODULE

The Analog-to-Digital (A/D) converter module has 12 inputs for the 64-pin devices and 16 for the 80-pin devices. This module allows conversion of an analog input signal to a corresponding 10-bit digital number.

The module has five registers:

- A/D Result High Register (ADRESH)
- A/D Result Low Register (ADRESL)
- A/D Control Register 0 (ADCON0)
- A/D Control Register 1 (ADCON1)
- A/D Control Register 2 (ADCON2)

The ADCON0 register, shown in Register 21-1, controls the operation of the A/D module. The ADCON1 register, shown in Register 21-2, configures the functions of the port pins. The ADCON2 register, shown in Register 21-3, configures the A/D clock source, programmed acquisition time and justification.

REGISTER 21-1: ADCON0: A/D CONTROL REGISTER

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	CHS3	CHS2	CHS1	CHS0	GO/DONE	ADON
bit 7							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-2 **CHS3:CHS0:** Analog Channel Select bits

0000 = Channel 0 (AN0)
 0001 = Channel 1 (AN1)
 0010 = Channel 2 (AN2)
 0011 = Channel 3 (AN3)
 0100 = Channel 4 (AN4)
 0101 = Channel 5 (AN5)
 0110 = Channel 6 (AN6)
 0111 = Channel 7 (AN7)
 1000 = Channel 8 (AN8)
 1001 = Channel 9 (AN9)
 1010 = Channel 10 (AN10)
 1011 = Channel 11 (AN11)
 1100 = Channel 12 (AN12)⁽¹⁾
 1101 = Channel 13 (AN13)⁽¹⁾
 1110 = Channel 14 (AN14)⁽¹⁾
 1111 = Channel 15 (AN15)⁽¹⁾

Note 1: These channels are not implemented on 64-pin devices.

bit 1 **GO/DONE:** A/D Conversion Status bit

When ADON = 1:
 1 = A/D conversion in progress
 0 = A/D Idle

bit 0 **ADON:** A/D On bit

1 = A/D converter module is enabled
 0 = A/D converter module is disabled

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

PIC18F8722 FAMILY

REGISTER 21-2: ADCON1: A/D CONTROL REGISTER 1

U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
—	—	VCFG1	VCFG0	PCFG3	PCFG2	PCFG1	PCFG0
bit 7							
							bit 0

bit 7-6 **Unimplemented:** Read as '0'

bit 5-4 **VCFG1:VCFG0:** Voltage Reference Configuration bits

	A/D VREF+	A/D VREF-
00	AVDD	AVSS
01	External VREF+	AVSS
10	AVDD	External VREF-
11	External VREF+	External VREF-

bit 3-0 **PCFG3:PCFG0:** A/D Port Configuration Control bits:

PCFG3: PCFG0	AN15 ⁽¹⁾	AN14 ⁽¹⁾	AN13 ⁽¹⁾	AN12 ⁽¹⁾	AN11	AN10	AN9	AN8	AN7	AN6	AN5	AN4	AN3	AN2	AN1	AN0
0000	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0001	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A	A
0010	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A	A
0011	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A	A
0100	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A	A
0101	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A	A
0110	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A	A
0111	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A	A
1000	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A	A
1001	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A	A
1010	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A	A
1011	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A	A
1100	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A	A
1101	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A	A
1110	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	A
1111	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D	D

A = Analog input

D = Digital I/O

Note 1: AN15 through AN12 are available only on 80-pin devices.

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

REGISTER 21-3: ADCON2: A/D CONTROL REGISTER 2

R/W-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
ADFM	—	ACQT2	ACQT1	ACQT0	ADCS2	ADCS1	ADCS0
bit 7							bit 0

bit 7 **ADFM:** A/D Result Format Select bit

1 = Right justified
0 = Left justified

bit 6 **Unimplemented:** Read as '0'

bit 5-3 **ACQT2:ACQT0:** A/D Acquisition Time Select bits

111 = 20 TAD
110 = 16 TAD
101 = 12 TAD
100 = 8 TAD
011 = 6 TAD
010 = 4 TAD
001 = 2 TAD
000 = 0 TAD⁽¹⁾

bit 2-0 **ADCS2:ADCS0:** A/D Conversion Clock Select bits

111 = FRC (clock derived from A/D RC oscillator)⁽¹⁾
110 = Fosc/64
101 = Fosc/16
100 = Fosc/4
011 = FRC (clock derived from A/D RC oscillator)⁽¹⁾
010 = Fosc/32
001 = Fosc/8
000 = Fosc/2

Note 1: If the A/D FRC clock source is selected, a delay of one Tcy (instruction cycle) is added before the A/D clock starts. This allows the SLEEP instruction to be executed before starting a conversion.

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

5.1.3.2 Return Stack Pointer (STKPTR)

The STKPTR register (Register 5-1) contains the Stack Pointer value, the STKFUL (Stack Full) status bit and the STKUNF (Stack Underflow) status bits. The value of the Stack Pointer can be 0 through 31. The Stack Pointer increments before values are pushed onto the stack and decrements after values are popped off the stack. On Reset, the stack pointer value will be zero. The user may read and write the stack pointer value. This feature can be used by a Real-Time Operating System (RTOS) for return stack maintenance.

After the PC is pushed onto the stack 31 times (without popping any values off the stack), the STKFUL bit is set. The STKFUL bit is cleared by software or by a POR.

The action that takes place when the stack becomes full depends on the state of the STVREN (Stack Overflow Reset Enable) configuration bit. (Refer to **Section 25.1 "Configuration Bits"** for a description of the device configuration bits.) If STVREN is set (default), the 31st PUSH will push the (PC + 2) value onto the stack, set the STKFUL bit and reset the device. The STKFUL bit will remain set and the Stack Pointer will be set to zero.

If STVREN is cleared, the STKFUL bit will be set on the 31st PUSH and the Stack Pointer will increment to 31. Any additional pushes will not overwrite the 31st PUSH and STKPTR will remain at 31.

When the stack has been popped enough times to unload the stack, the next POP will return a value of zero to the PC and set the STKUNF bit, while the Stack Pointer remains at zero. The STKUNF bit will remain set until cleared by software or until a POR occurs.

Note: Returning a value of zero to the PC on an underflow has the effect of vectoring the program to the Reset vector, where the stack conditions can be verified and appropriate actions can be taken. This is not the same as a Reset, as the contents of the SFRs are not affected.

5.1.3.3 PUSH and POP Instructions

Since the Top-of-Stack is readable and writable, the ability to push values onto the stack and pull values off the stack without disturbing normal program execution is a desirable feature. The PIC18 instruction set includes two instructions, PUSH and POP, that permit the TOS to be manipulated under software control. TOSU, TOSH and TOSL can be modified to place data or a return address on the stack.

The PUSH instruction places the current PC value onto the stack. This increments the Stack Pointer and loads the current PC value onto the stack.

The POP instruction discards the current TOS by decrementing the Stack Pointer. The previous value pushed onto the stack then becomes the TOS value.

REGISTER 5-1: STKPTR: STACK POINTER REGISTER

R/C-0	R/C-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STKFUL ⁽¹⁾	STKUNF ⁽¹⁾	—	SP4	SP3	SP2	SP1	SP0
bit 7			bit 0				

- bit 7 **STKFUL:** Stack Full Flag bit⁽¹⁾
 1 = Stack became full or overflowed
 0 = Stack has not become full or overflowed
- bit 6 **STKUNF:** Stack Underflow Flag bit⁽¹⁾
 1 = Stack underflow occurred
 0 = Stack underflow did not occur
- bit 5 **Unimplemented:** Read as '0'
- bit 4-0 **SP4:SP0:** Stack Pointer Location bits

Note 1: Bit 7 and bit 6 are cleared by user software or by a POR.

Legend:

R = Readable bit	W = Writable bit	U = Unimplemented	C = Clearable only bit
-n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

12.0 TIMER0 MODULE

The Timer0 module incorporates the following features:

- Software selectable operation as a timer or counter in both 8-bit or 16-bit modes
- Readable and writable registers
- Dedicated 8-bit, software programmable prescaler
- Selectable clock source (internal or external)
- Edge select for external clock
- Interrupt-on-overflow

The T0CON register (Register 12-1) controls all aspects of the module's operation, including the prescale selection. It is both readable and writable.

A simplified block diagram of the Timer0 module in 8-bit mode is shown in Figure 12-1. Figure 12-2 shows a simplified block diagram of the Timer0 module in 16-bit mode.

REGISTER 12-1: T0CON: TIMER0 CONTROL REGISTER

R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1	R/W-1
TMR0ON	T08BIT	T0CS	T0SE	PSA	T0PS2	T0PS1	T0PS0
bit 7							bit 0

- bit 7 **TMR0ON:** Timer0 On/Off Control bit
 1 = Enables Timer0
 0 = Stops Timer0
- bit 6 **T08BIT:** Timer0 8-bit/16-bit Control bit
 1 = Timer0 is configured as an 8-bit timer/counter
 0 = Timer0 is configured as a 16-bit timer/counter
- bit 5 **T0CS:** Timer0 Clock Source Select bit
 1 = Transition on T0CKI pin
 0 = Internal instruction cycle clock (CLKO)
- bit 4 **T0SE:** Timer0 Source Edge Select bit
 1 = Increment on high-to-low transition on T0CKI pin
 0 = Increment on low-to-high transition on T0CKI pin
- bit 3 **PSA:** Timer0 Prescaler Assignment bit
 1 = Timer0 prescaler is NOT assigned. Timer0 clock input bypasses prescaler.
 0 = Timer0 prescaler is assigned. Timer0 clock input comes from prescaler output.
- bit 2-0 **T0PS2:T0PS0:** Timer0 Prescaler Select bits
 111 = 1:256 Prescale value
 110 = 1:128 Prescale value
 101 = 1:64 Prescale value
 100 = 1:32 Prescale value
 011 = 1:16 Prescale value
 010 = 1:8 Prescale value
 001 = 1:4 Prescale value
 000 = 1:2 Prescale value

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

Appendice A2

Datasheet dei componenti, foto del prototipo e schema elettrico del modulo OBDH & HK



MICROCHIP

PIC18F8722 FAMILY

64/80-Pin, 1-Mbit, Enhanced Flash Microcontrollers with 10-Bit A/D and nanoWatt Technology

Peripheral Highlights:

- Two Master Synchronous Serial Port (MSSP) modules supporting 2/3/4-wire SPI™ (all 4 modes) and I²C™ Master and Slave modes
- Two Capture/Compare/PWM (CCP) modules
- Three Enhanced Capture/Compare/PWM (ECCP) modules:
 - One, two or four PWM outputs
 - Selectable polarity
 - Programmable dead time
 - Auto-Shutdown and Auto-Restart
- Two Enhanced Addressable USART modules:
 - Supports RS-485, RS-232 and LIN 1.2
 - Auto-Wake-up on Start bit
 - Auto-Baud Detect
- 10-bit, up to 16-channel Analog-to-Digital Converter module (A/D)
 - Auto-acquisition capability
 - Conversion available during Sleep
- Dual analog comparators with input multiplexing
- High-current sink/source 25 mA/25 mA
- Four programmable external interrupts
- Four input change interrupts

External Memory Interface

(PIC18F8527/8622/8627/8722 only):

- Address capability of up to 2 Mbytes
- 8-bit or 16-bit interface
- 8, 12, 16 and 20-bit Address modes

Power-Managed Modes:

- Run: CPU on, peripherals on
- Idle: CPU off, peripherals on
- Sleep: CPU off, peripherals off
- Idle mode currents down to 15 μ A typical
- Sleep current down to 0.2 μ A typical
- Timer1 Oscillator: 1.8 μ A, 32 kHz, 2V
- Watchdog Timer: 2.1 μ A

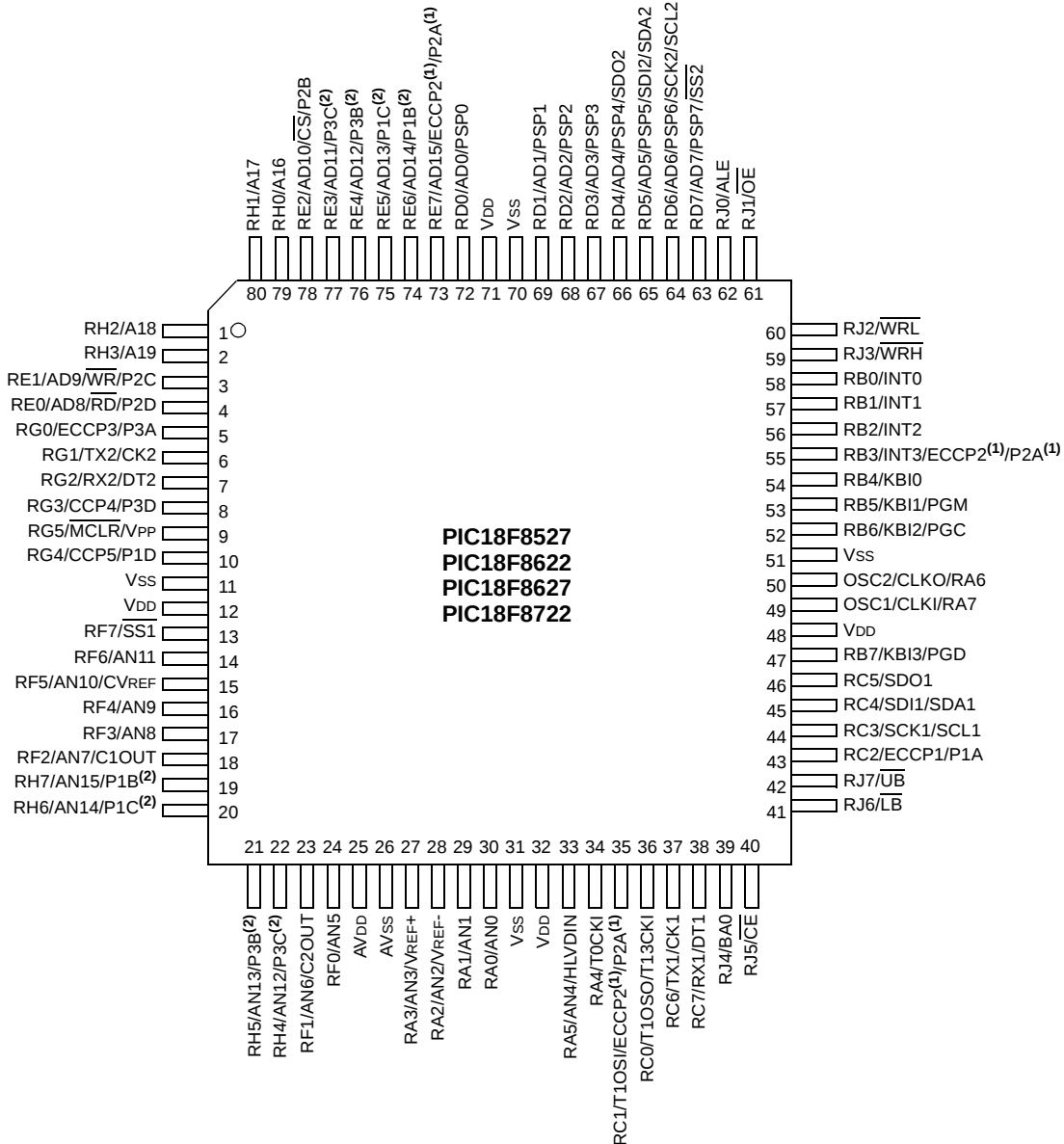
Special Microcontroller Features:

- C compiler optimized architecture:
 - Optional extended instruction set designed to optimize re-entrant code
- 100,000 erase/write cycle Enhanced Flash program memory typical
- 1,000,000 erase/write cycle Data EEPROM memory typical
- Flash/Data EEPROM Retention: 100 years typical
- Self-programmable under software control
- Priority levels for interrupts
- 8 x 8 Single-Cycle Hardware Multiplier
- Extended Watchdog Timer (WDT):
 - Programmable period from 4 ms to 131s
- Single-Supply In-Circuit Serial Programming™ (ICSP™) via two pins
- In-Circuit Debug (ICD) via two pins
- Wide operating voltage range: 2.0V to 5.5V
- Fail-Safe Clock Monitor
- Two-Speed Oscillator Start-up
- nanoWatt Technology

Device	Program Memory		Data Memory		I/O	10-bit A/D (ch)	CCP/ ECCP (PWM)	MSSP		EUSART	Comparators	Timers 8/16-bit	External Bus	
	Flash (bytes)	# Single-Word Instructions	SRAM (bytes)	EEPROM (bytes)				SPI™	Master I²C™					
PIC18F6527	48K	24576	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F6622	64K	32768	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F6627	96K	49152	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F6722	128K	65536	3936	1024	54	12	2/3	2	Y	Y	2	2	2/3	N
PIC18F8527	48K	24576	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y
PIC18F8622	64K	32768	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y
PIC18F8627	96K	49152	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y
PIC18F8722	128K	65536	3936	1024	70	16	2/3	2	Y	Y	2	2	2/3	Y

Pin Diagrams (Continued)

80-Pin TQFP

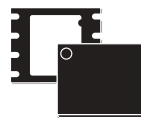


Note 1: The ECCP2/P2A pin placement is determined by the CCP2MX configuration bit and Processor mode settings.
Note 2: P1B, P1C, P3B and P3C pin placement is determined by the ECCPMX configuration bit.

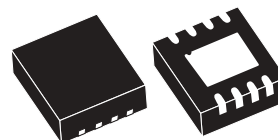
8-Mbit, page-erasable serial flash memory with
byte alterability, 75 MHz SPI bus, standard pinout

Features

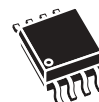
- SPI bus compatible serial interface
- 8-Mbit page-erasable flash memory
- Page size: 256 bytes
 - Page write in 11 ms (typical)
 - Page program in 0.8 ms (typical)
 - Page erase in 10 ms (typical)
- Subsector erase (4 Kbytes)
- Sector erase (64 Kbytes)
- Bulk erase (8 Mbits)
- 2.7 V to 3.6 V single supply voltage
- 75 MHz clock rate (maximum)
- Deep power-down mode 1 μ A (typical)
- Electronic signature
 - JEDEC standard two-byte signature (8014h)
 - Unique ID code (UID) with 16 bytes read-only, available upon customer request only in the T9HX process
- Software write protection on a 64-Kbyte sector basis
- Hardware write protection of the memory area selected using the BP0, BP1 and BP2 bits
- More than 100 000 write cycles
- More than 20 years data retention
- Packages
 - ECOPACK® (RoHS compliant)



VFQFPN8 (MP)
6 x 5 mm (MLP8)



QFN8L (MS)
6 x 5 mm (MLP8)



SO8W (MW)
208 mils width



SO8N (MN)
150 mils width

FM25040A

4Kb FRAM Serial Memory

RAMTRON

Features

4K bit Ferroelectric Nonvolatile RAM

- Organized as 512 x 8 bits
- High Endurance 1 Trillion (10^{12}) Read/Writes
- 45 year Data Retention
- NoDelay™ Writes
- Advanced High-Reliability Ferroelectric Process

Very Fast Serial Peripheral Interface - SPI

- Up to 20 MHz maximum Bus Frequency
- Direct hardware replacement for EEPROM
- SPI Mode 0 & 3 (CPOL, CPHA=0,0 & 1,1)

Sophisticated Write Protection Scheme

- Hardware Protection
- Software Protection

Low Power Consumption

- 10 μ A Standby Current

Industry Standard Configuration

- Industrial Temperature -40° C to +85° C
- 8-pin SOIC (-S)
- “Green” 8-pin SOIC (-G)

Description

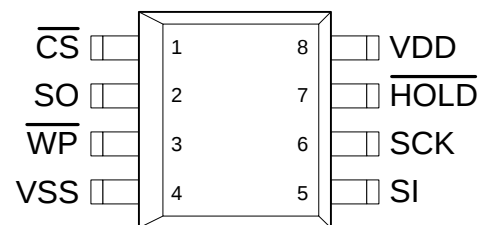
The FM25040A is a 4-kilobit nonvolatile memory employing an advanced ferroelectric process. A ferroelectric random access memory or FRAM is nonvolatile but operates in other respects as a RAM. It provides reliable data retention for 45 years while eliminating the complexities, overhead, and system level reliability problems caused by EEPROM and other nonvolatile memories.

Unlike serial EEPROMs, the FM25040A performs write operations at bus speed. No write delays are incurred. Data is written to the memory array in the cycle after it has been successfully transferred to the device. The next bus cycle may commence immediately without the need for data polling. In addition the product offers substantial write endurance compared with other nonvolatile memories. The FM25040A is capable of supporting up to 10^{12} read/write cycles -- far more than most systems will require from a serial memory.

These capabilities make the FM25040A ideal for nonvolatile memory applications requiring frequent or rapid writes. Examples range from data collection, where the number of write cycles may be critical, to demanding industrial controls where the long write time of EEPROM can cause data loss.

The FM25040A provides substantial benefits to users of serial EEPROM, in a hardware drop-in replacement. The FM25040A uses the high-speed SPI bus which enhances the high-speed write capability of FRAM technology. Device specifications are guaranteed over an industrial temperature range of -40°C to +85°C.

Pin Configuration



Pin Names	Function
/CS	Chip Select
/WP	Write Protect
/HOLD	Hold
SCK	Serial Clock
SI	Serial Data Input
SO	Serial Data Output
VDD	Supply Voltage 5V
VSS	Ground

Ordering Information	
FM25040A-S	8-pin SOIC
FM25040A-G	“Green” 8-pin SOIC

This product conforms to specifications per the terms of the Ramtron standard warranty. The product has completed Ramtron’s internal qualification testing and has reached production status.

Ramtron International Corporation
1850 Ramtron Drive, Colorado Springs, CO 80921
(800) 545-FRAM, (719) 481-7000
www.ramtron.com



Am29LV081B

8 Megabit (1 M x 8-Bit)

CMOS 3.0 Volt-only Uniform Sector Flash Memory

DISTINCTIVE CHARACTERISTICS

- **Single power supply operation**
 - 2.7 to 3.6 volt read and write operations for battery-powered applications
- **Manufactured on 0.32 μ m process technology**
 - Compatible with 0.5 μ m Am29LV081 device
- **High performance**
 - Access times as fast as 70 ns
- **Ultra low power consumption (typical values at 5 MHz)**
 - 200 nA Automatic Sleep mode current
 - 200 nA standby mode current
 - 7 mA read current
 - 15 mA program/erase current
- **Flexible sector architecture**
 - Sixteen 64 Kbyte sectors
 - Supports full chip erase
 - Sector Protection features:
 - A hardware method of locking a sector to prevent any program or erase operations within that sector
 - Sectors can be locked in-system or via programming equipment
 - Temporary Sector Unprotect feature allows code changes in previously locked sectors
- **Unlock Bypass Program Command**
 - Reduces overall programming time when issuing multiple program command sequences
- **Embedded Algorithms**
 - Embedded Erase algorithm automatically preprograms and erases the entire chip or any combination of designated sectors
 - Embedded Program algorithm automatically writes and verifies data at specified addresses
- **Minimum 1,000,000 write cycle guarantee per sector**
- **20-year data retention at 125°C**
 - Reliable operation for the life of the system
- **Package option**
 - 40-pin TSOP
- **Compatibility with JEDEC standards**
 - Pinout and software compatible with single-power supply Flash
 - Superior inadvertent write protection
- **Data# Polling and toggle bits**
 - Provides a software method of detecting program or erase operation completion
- **Ready/Busy# pin (RY/BY#)**
 - Provides a hardware method of detecting program or erase cycle completion
- **Erase Suspend/Erase Resume**
 - Suspends an erase operation to read data from, or program data to, a sector that is not being erased, then resumes the erase operation
- **Hardware reset pin (RESET#)**
 - Hardware method to reset the device to reading array data
- **Command sequence optimized for mass storage**
 - Specific addresses not required for unlock cycles



16 MB (1M x 16) Static RAM

Features

- Very high speed: 55 ns
- Wide voltage range: 2.20V – 3.60V
- Ultra-low active power
 - Typical active current: 2 mA @ f = 1 MHz
 - Typical active current: 15 mA @ f = f_{max}
- Ultra-low standby power
- Easy memory expansion with $\overline{CE}_1$, CE_2 , and $\overline{OE}$ features
- Automatic power-down when deselected
- CMOS for optimum speed/power
- Packages offered in a 48-ball BGA and 48-pin TSOP1

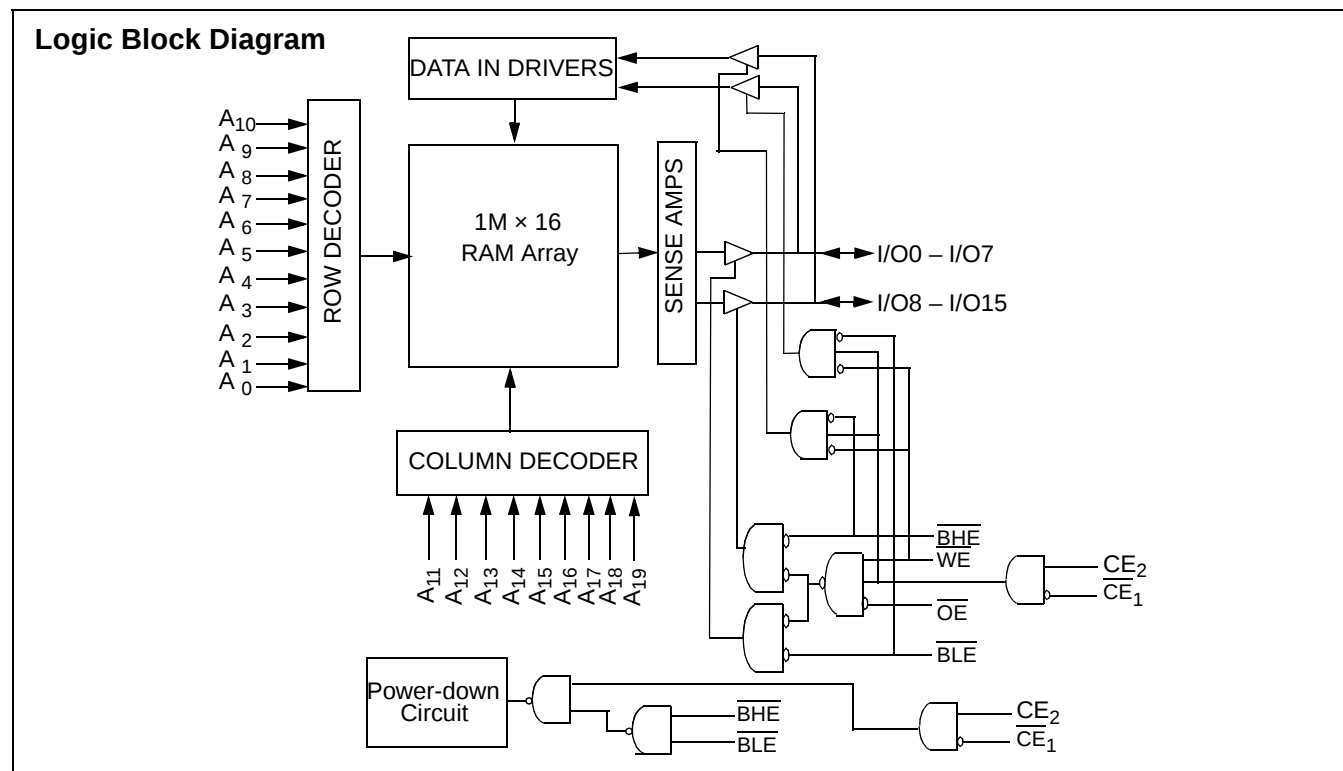
Functional Description^[1]

The CY62167DV30 is a high-performance CMOS static RAM organized as 1M words by 16 bits. This device features advanced circuit design to provide ultra-low active current. This is ideal for providing More Battery Life™ (MoBL[®]) in portable applications such as cellular telephones. The device also has an automatic power-down feature that significantly

reduces power consumption by 99% when addresses are not toggling. The device can also be put into standby mode when deselected ($\overline{CE}_1$ HIGH or CE_2 LOW or both BHE and BLE are HIGH). The input/output pins (I/O₀ through I/O₁₅) are placed in a high-impedance state when: deselected ($\overline{CE}_1$ HIGH or CE_2 LOW), outputs are disabled ($\overline{OE}$ HIGH), both Byte High Enable and Byte Low Enable are disabled (BHE, BLE HIGH), or during a write operation ($\overline{CE}_1$ LOW, CE_2 HIGH and WE LOW).

Writing to the device is accomplished by taking Chip Enables ($\overline{CE}_1$ LOW and CE_2 HIGH) and Write Enable (WE) input LOW. If Byte Low Enable (BLE) is LOW, then data from I/O pins (I/O₀ through I/O₇), is written into the location specified on the address pins (A₀ through A₁₉). If Byte High Enable (BHE) is LOW, then data from I/O pins (I/O₈ through I/O₁₅) is written into the location specified on the address pins (A₀ through A₁₉).

Reading from the device is accomplished by taking Chip Enables ($\overline{CE}_1$ LOW and CE_2 HIGH) and Output Enable ($\overline{OE}$) LOW while forcing the Write Enable (WE) HIGH. If Byte Low Enable (BLE) is LOW, then data from the memory location specified by the address pins will appear on I/O₀ to I/O₇. If Byte High Enable (BHE) is LOW, then data from memory will appear on I/O₈ to I/O₁₅. See the truth table at the back of this datasheet for a complete description of read and write modes.



Note:

1. For best practice recommendations, please refer to the Cypress application note "System Design Guidelines" on <http://www.cypress.com>.

November 1997 - Revised September 2003

High-Speed CMOS Logic Quad Buffer, Three-State

Features

- Three-State Outputs
- Separate Output Enable Inputs
- Fanout (Over Temperature Range)
 - Standard Outputs 10 LSTTL Loads
 - Bus Driver Outputs 15 LSTTL Loads
- Wide Operating Temperature Range . . . -55°C to 125°C
- Balanced Propagation Delay and Transition Times
- Significant Power Reduction Compared to LSTTL Logic ICs
- HC Types
 - 2V to 6V Operation
 - High Noise Immunity: $N_{IL} = 30\%$, $N_{IH} = 30\%$ of V_{CC} at $V_{CC} = 5V$
- HCT Types
 - 4.5V to 5.5V Operation
 - Direct LSTTL Input Logic Compatibility, $V_{IL} = 0.8V$ (Max), $V_{IH} = 2V$ (Min)
 - CMOS Input Compatibility, $I_I \leq 1\mu A$ at V_{OL} , V_{OH}

Description

The 'HC126 and 'HCT126 contain four independent three-state buffers, each having its own output enable input, which when "low" puts the output in the high-impedance state.

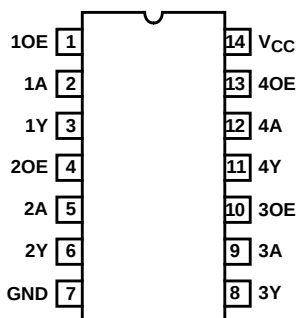
Ordering Information

PART NUMBER	TEMP. RANGE (°C)	PACKAGE
CD54HC126F3A	-55 to 125	14 Ld CERDIP
CD54HCT126F3A	-55 to 125	14 Ld CERDIP
CD74HC126E	-55 to 125	14 Ld PDIP
CD74HC126M	-55 to 125	14 Ld SOIC
CD74HC126MT	-55 to 125	14 Ld SOIC
CD74HC126M96	-55 to 125	14 Ld SOIC
CD74HCT126E	-55 to 125	14 Ld PDIP
CD74HCT126M	-55 to 125	14 Ld SOIC
CD74HCT126MT	-55 to 125	14 Ld SOIC
CD74HCT126M96	-55 to 125	14 Ld SOIC

NOTE: When ordering, use the entire part number. The suffix 96 denotes tape and reel. The suffix T denotes a small-quantity reel of 250.

Pinout

CD54HC126, CD54HCT126
(CERDIP)
CD74HC126, CD74HCT126
(PDIP, SOIC)
TOP VIEW



High-Speed CMOS Logic 3- to 8-Line Decoder/ Demultiplexer Inverting and Noninverting

October 1997 - Revised August 2004

Features

- Select One Of Eight Data Outputs
Active Low for 138, Active High for 238
- I/O Port or Memory Selector
- Three Enable Inputs to Simplify Cascading
- Typical Propagation Delay of 13 ns at $V_{CC} = 5\text{ V}$,
 $C_L = 15\text{ pF}$, $T_A = 25^\circ\text{C}$
- Fanout (Over Temperature Range)
 - Standard Outputs 10 LSTTL Loads
 - Bus Driver Outputs 15 LSTTL Loads
- Wide Operating Temperature Range . . . -55°C to 125°C
- Balanced Propagation Delay and Transition Times
- Significant Power Reduction Compared to LSTTL
Logic ICs
- HC Types
 - 2 V to 6 V Operation
 - High Noise Immunity: $N_{IL} = 30\%$, $N_{IH} = 30\%$ of V_{CC}
at $V_{CC} = 5\text{ V}$
- HCT Types
 - 4.5-V to 5.5-V Operation
 - Direct LSTTL Input Logic Compatibility,
 $V_{IL} = 0.8\text{ V (Max)}$, $V_{IH} = 2\text{ V (Min)}$
 - CMOS Input Compatibility, $I_I \leq 1\mu\text{A}$ at V_{OL} , V_{OH}

Description

The 'HC138, 'HC238, 'HCT138, and 'HCT238 are high-speed silicon-gate CMOS decoders well suited to memory address decoding or data-routing applications. Both circuits feature low power consumption usually associated with CMOS circuitry, yet have speeds comparable to low-power Schottky TTL logic. Both circuits have three binary select inputs (A0, A1, and A2). If the device is enabled, these inputs determine which one of the eight normally high outputs of the HC/HCT138 series go low or which of the normally low outputs of the HC/HCT238 series go high.

Two active low and one active high enables ($\overline{E1}$, $\overline{E2}$, and E3) are provided to ease the cascading of decoders. The decoder's eight outputs can drive ten low-power Schottky TTL equivalent loads.

Ordering Information

PART NUMBER	TEMP. RANGE (°C)	PACKAGE
CD54HC138F3A	-55 to 125	16 Ld CERDIP
CD54HC238F3A	-55 to 125	16 Ld CERDIP
CD54HCT138F3A	-55 to 125	16 Ld CERDIP
CD54HCT238F3A	-55 to 125	16 Ld CERDIP
CD74HC138E	-55 to 125	16 Ld PDIP
CD74HC138M	-55 to 125	16 Ld SOIC
CD74HC138MT	-55 to 125	16 Ld SOIC
CD74HC138M96	-55 to 125	16 Ld SOIC
CD74HC238E	-55 to 125	16 Ld PDIP
CD74HC238M	-55 to 125	16 Ld SOIC
CD74HC238MT	-55 to 125	16 Ld SOIC
CD74HC238M96	-55 to 125	16 Ld SOIC
CD74HC238NSR	-55 to 125	16 Ld SOP
CD74HC238PW	-55 to 125	16 Ld TSSOP
CD74HC238PWR	-55 to 125	16 Ld TSSOP
CD74HC238PWT	-55 to 125	16 Ld TSSOP
CD74HCT138E	-55 to 125	16 Ld PDIP
CD74HCT138M	-55 to 125	16 Ld SOIC
CD74HCT138MT	-55 to 125	16 Ld SOIC
CD74HCT138M96	-55 to 125	16 Ld SOIC
CD74HCT238E	-55 to 125	16 Ld PDIP
CD74HCT238M	-55 to 125	16 Ld SOIC
CD74HCT238M96	-55 to 125	16 Ld SOIC

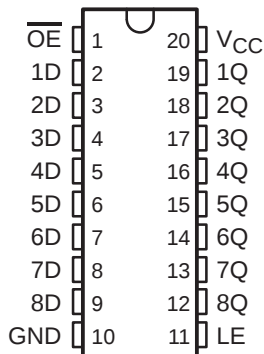
NOTE: When ordering, use the entire part number. The suffixes 96 and R denote tape and reel. The suffix T denotes a small-quantity reel of 250.

SN54HC573A, SN74HC573A OCTAL TRANSPARENT D-TYPE LATCHES WITH 3-STATE OUTPUTS

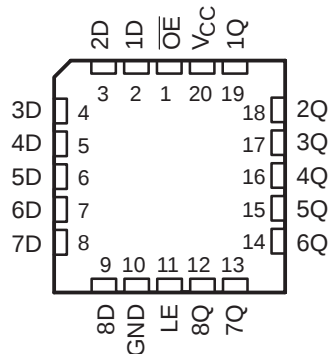
SCLS147E – DECEMBER 1982 – REVISED SEPTEMBER 2003

- Wide Operating Voltage Range of 2 V to 6 V
- High-Current 3-State Outputs Drive Bus Lines Directly or Up To 15 LSTTL Loads
- Low Power Consumption, 80- μ A Max I_{CC}
- Typical $t_{pd} = 21$ ns
- ± 6 -mA Output Drive at 5 V
- Low Input Current of 1 μ A Max
- Bus-Structured Pinout

SN54HC573A . . . J OR W PACKAGE
SN74HC573A . . . DB, DW, N, OR PW PACKAGE
(TOP VIEW)



SN54HC573A . . . FK PACKAGE
(TOP VIEW)



description/ordering information

These octal transparent D-type latches feature 3-state outputs designed specifically for driving highly capacitive or relatively low-impedance loads. They are particularly suitable for implementing buffer registers, I/O ports, bidirectional bus drivers, and working registers.

While the latch-enable (LE) input is high, the Q outputs respond to the data (D) inputs. When LE is low, the outputs are latched to retain the data that was set up.

A buffered output-enable ($\overline{OE}$) input can be used to place the eight outputs in either a normal logic state (high or low logic levels) or the high-impedance state. In the high-impedance state, the outputs neither load nor drive the bus lines significantly. The high-impedance state and increased drive provide the capability to drive bus lines without interface or pullup components.

ORDERING INFORMATION

T _A	PACKAGE [†]		ORDERABLE PART NUMBER	TOP-SIDE MARKING
-40°C to 85°C	PDIP – N	Tube of 25	SN74HC573AN	SN74HC573AN
	SOIC – DW	Tube of 40	SN74HC573ADW	HC573A
		Reel of 2500	SN74HC573ADWR	
	SSOP – DB	Reel of 2000	SN74HC573ADBR	HC573A
	TSSOP – PW	Reel of 2000	SN74HC573APWR	HC573A
		Reel of 250	SN74HC573APWT	
-55°C to 125°C	CDIP – J	Tube of 25	SNJ54HC573AJ	SNJ54HC573AJ
	CFP – W	Tube of 150	SNJ54HC573AW	SNJ54HC573AW
	LCCC – FK	Tube of 55	SNJ54HC573AFK	SNJ54HC573AFK

[†] Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.



Please be aware that an important notice concerning availability, standard warranty, and use in critical applications of Texas Instruments semiconductor products and disclaimers thereto appears at the end of this data sheet.

PRODUCTION DATA information is current as of publication date. Products conform to specifications per the terms of Texas Instruments standard warranty. Production processing does not necessarily include testing of all parameters.

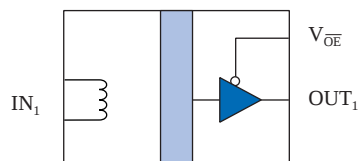


POST OFFICE BOX 655303 • DALLAS, TEXAS 75265

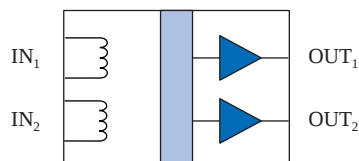
Copyright © 2003, Texas Instruments Incorporated
On products compliant to MIL-PRF-38535, all parameters are tested unless otherwise noted. On all other products, production processing does not necessarily include testing of all parameters.

Passive-Input Digital Isolators – CMOS Outputs

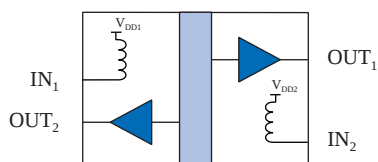
Functional Diagrams



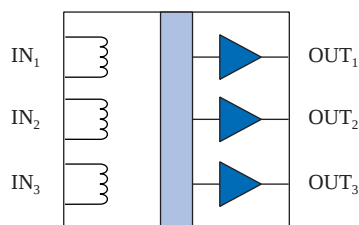
IL610



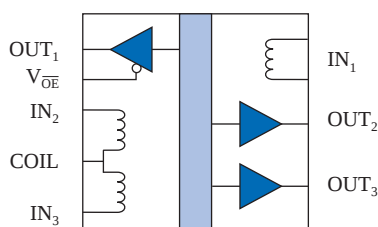
IL611



IL612



IL613



IL614

Features

- Up to 100 Mbps Data Rate
- Single-ended or Differential Input Operation
- Flexible Inputs with Very Wide Input Voltage Range
- Failsafe Output (logic high output for zero coil current)
- Output Enable (IL610)
- 3.3 V or 5 V Operation / Level Translation
- 2500 V_{RMS} Isolation (1 minute)
- Low Power Dissipation
- -40°C to 85°C Temperature Range
- 20 kV/μs Transient Immunity
- Low EMC Footprint
- UL1577 and IEC61010-2001 Approved
- 8-Pin MSOP, SOIC, and PDIP Packages
- Bare Die Available (IL610)

Applications

- CAN Bus / Device Net
- Differential Line Receiver
- Optocoupler Replacement
- SPI Interface
- RS-485, RS-422, or RS-232
- Digital Fieldbus
- Space-critical multi-channel applications

Description

The IL600 Series are passive input digital signal isolators with CMOS outputs. They have a similar interface but better performance and higher package density than optocouplers.

The devices are manufactured with NVE's patented* IsoLoop[®] spintronic Giant Magnetoresistive (GMR) technology for small size, high speed, and low power.

A single resistor sets the maximum input current for voltages above 0.5 V. A capacitor in parallel with the current-limit resistor provides improved dynamic performance.

These versatile components simplify inventory requirements by replacing a variety of optocouplers and functioning over a wide range of data rates, edge speeds, and power supply levels. The devices are available in MSOP, SOIC, and PDIP packages, as well as bare die.



3.0V to 5.5V, Low-Power, up to 1Mbps, True RS-232 Transceivers Using Four 0.1 μ F External Capacitors

General Description

The MAX3222/MAX3232/MAX3237/MAX3241 transceivers have a proprietary low-dropout transmitter output stage enabling true RS-232 performance from a 3.0V to 5.5V supply with a dual charge pump. The devices require only four small 0.1 μ F external charge-pump capacitors. The MAX3222, MAX3232, and MAX3241 are guaranteed to run at data rates of 120kbps while maintaining RS-232 output levels. The MAX3237 is guaranteed to run at data rates of 250kbps in the normal operating mode and 1Mbps in the MegaBaud™ operating mode, while maintaining RS-232 output levels.

The MAX3222/MAX3232 have 2 receivers and 2 drivers. The MAX3222 features a 1 μ A shutdown mode that reduces power consumption and extends battery life in portable systems. Its receivers remain active in shutdown mode, allowing external devices such as modems to be monitored using only 1 μ A supply current. The MAX3222 and MAX3232 are pin, package, and functionally compatible with the industry-standard MAX242 and MAX232, respectively.

The MAX3241 is a complete serial port (3 drivers/5 receivers) designed for notebook and subnotebook computers. The MAX3237 (5 drivers/3 receivers) is ideal for fast modem applications. Both these devices feature a shutdown mode in which all receivers can remain active while using only 1 μ A supply current. Receivers R1 (MAX3237/MAX3241) and R2 (MAX3241) have extra outputs in addition to their standard outputs. These extra outputs are always active, allowing external devices such as a modem to be monitored without forward biasing the protection diodes in circuitry that may have VCC completely removed.

The MAX3222, MAX3237, and MAX3241 are available in space-saving TSSOP and SSOP packages.

Applications

Notebook, Subnotebook, and Palmtop Computers
High-Speed Modems
Battery-Powered Equipment
Hand-Held Equipment
Peripherals
Printers

Typical Operating Circuits appear at end of data sheet.

MegaBaud and UCSP are trademarks of Maxim Integrated Products, Inc.

*Covered by U.S. Patent numbers 4,636,930; 4,679,134; 4,777,577; 4,797,899; 4,809,152; 4,897,774; 4,999,761; and other patents pending.



Maxim Integrated Products 1

For pricing, delivery, and ordering information, please contact Maxim/Dallas Direct! at 1-888-629-4642, or visit Maxim's website at www.maxim-ic.com.

Next Generation Device Features

- ◆ **For Smaller Packaging:**
MAX3228E/MAX3229E: +2.5V to +5.5V RS-232 Transceivers in UCSP™
- ◆ **For Integrated ESD Protection:**
MAX3222E/MAX3232E/MAX3237E/MAX3241E*/MAX3246E: ± 15 kV ESD-Protected, Down to 10nA, 3.0V to 5.5V, Up to 1Mbps, True RS-232 Transceivers
- ◆ **For Low-Voltage or Data Cable Applications:**
MAX3380E/MAX3381E: +2.35V to +5.5V, 1 μ A, 2 Tx/2 Rx RS-232 Transceivers with ± 15 kV ESD-Protected I/O and Logic Pins

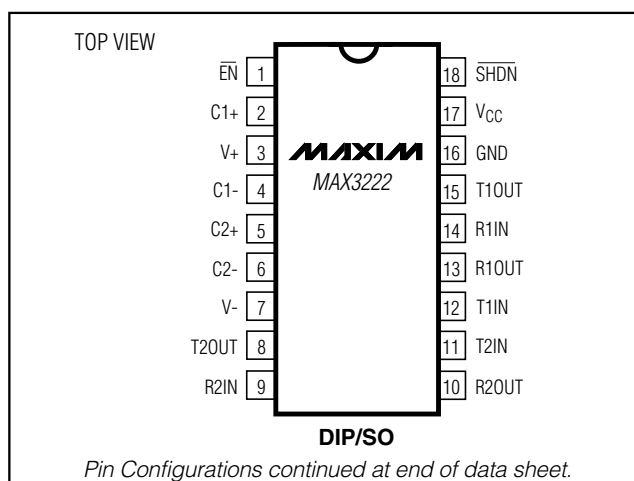
Ordering Information

PART	TEMP RANGE	PIN-PACKAGE	PKG CODE
MAX3222CUP+	0°C to +70°C	20 TSSOP	U20+2
MAX3222CAP+	0°C to +70°C	20 SSOP	A20+1
MAX3222CWN+	0°C to +70°C	18 SO	W18+1
MAX3222CPN+	0°C to +70°C	18 Plastic Dip	P18+5

+Denotes lead-free package.

Ordering Information continued at end of data sheet.

Pin Configurations



MAX3222/MAX3232/MAX3237/MAX3241 *



3.3V-Powered, 10Mbps and Slew-Rate-Limited True RS-485/RS-422 Transceivers

General Description

The MAX3483, MAX3485, MAX3486, MAX3488, MAX3490, and MAX3491 are 3.3V, low-power transceivers for RS-485 and RS-422 communication. Each part contains one driver and one receiver. The MAX3483 and MAX3488 feature slew-rate-limited drivers that minimize EMI and reduce reflections caused by improperly terminated cables, allowing error-free data transmission at data rates up to 250kbps. The partially slew-rate-limited MAX3486 transmits up to 2.5Mbps. The MAX3485, MAX3490, and MAX3491 transmit at up to 10Mbps.

Drivers are short-circuit current limited and are protected against excessive power dissipation by thermal shutdown circuitry that places the driver outputs into a high-impedance state. The receiver input has a fail-safe feature that guarantees a logic-high output if both inputs are open circuit.

The MAX3488, MAX3490, and MAX3491 feature full-duplex communication, while the MAX3483, MAX3485, and MAX3486 are designed for half-duplex communication.

Applications

Low-Power RS-485/RS-422 Transceivers
Telecommunications
Transceivers for EMI-Sensitive Applications
Industrial-Control Local Area Networks

Features

- ♦ Operate from a Single 3.3V Supply—No Charge Pump!
- ♦ Interoperable with +5V Logic
- ♦ 8ns Max Skew (MAX3485/MAX3490/MAX3491)
- ♦ Slew-Rate Limited for Errorless Data Transmission (MAX3483/MAX3488)
- ♦ 2nA Low-Current Shutdown Mode (MAX3483/MAX3485/MAX3486/MAX3491)
- ♦ -7V to +12V Common-Mode Input Voltage Range
- ♦ Allows up to 32 Transceivers on the Bus
- ♦ Full-Duplex and Half-Duplex Versions Available
- ♦ Industry Standard 75176 Pinout (MAX3483/MAX3485/MAX3486)
- ♦ Current-Limiting and Thermal Shutdown for Driver Overload Protection

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX3483CPA	0°C to +70°C	8 Plastic DIP
MAX3483CSA	0°C to +70°C	8 SO
MAX3483C/D	0°C to +70°C	Dice*
MAX3483EPA	-40°C to +85°C	8 Plastic DIP
MAX3483ESA	-40°C to +85°C	8 SO
MAX3485CPA	0°C to +70°C	8 Plastic DIP
MAX3485CSA	0°C to +70°C	8 SO
MAX3485C/D	0°C to +70°C	Dice*
MAX3485EPA	-40°C to +85°C	8 Plastic DIP
MAX3485ESA	-40°C to +85°C	8 SO

Ordering Information continued at end of data sheet.

* Contact factory for dice specifications.

Selection Table

PART NUMBER	GUARANTEED DATA RATE (Mbps)	SUPPLY VOLTAGE (V)	HALF/FULL DUPLEX	SLEW-RATE LIMITED	DRIVER/RECEIVER ENABLE	SHUTDOWN CURRENT (nA)	PIN COUNT
MAX3483	0.25	3.0 to 3.6	Half	Yes	Yes	2	8
MAX3485	10		Half	No	Yes	2	8
MAX3486	2.5		Half	Yes	Yes	2	8
MAX3488	0.25		Full	Yes	No	—	8
MAX3490	10		Full	No	No	—	8
MAX3491	10		Full	No	Yes	2	14





CMOS, +1.8 V to +5.5 V/ ± 2.5 V, 2.5 Ω Low-Voltage, 8-/16-Channel Multiplexers

ADG706/ADG707

FEATURES

- +1.8 V to +5.5 V Single Supply
- ± 2.5 V Dual Supply
- 2.5 Ω ON Resistance
- 0.5 Ω ON Resistance Flatness
- 100 pA Leakage Currents
- 40 ns Switching Times
- Single 16-to-1 Multiplexer ADG706
- Differential 8-to-1 Multiplexer ADG707
- 28-Lead TSSOP Package
- Low-Power Consumption
- TTL/CMOS-Compatible Inputs

APPLICATIONS

- Data Acquisition Systems
- Communication Systems
- Relay Replacement
- Audio and Video Switching
- Battery-Powered Systems

GENERAL DESCRIPTION

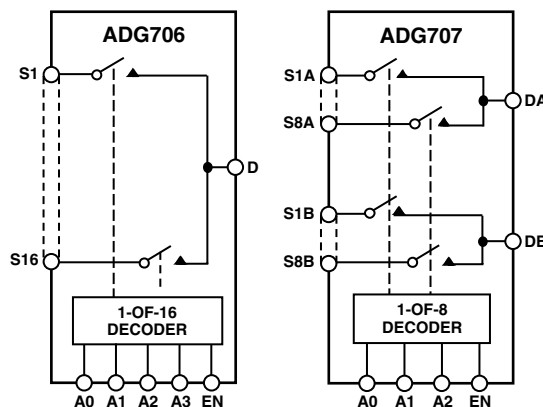
The ADG706 and ADG707 are low-voltage, CMOS analog multiplexers comprising 16 single channels and eight differential channels, respectively. The ADG706 switches one of 16 inputs (S1–S16) to a common output, D, as determined by the 4-bit binary address lines A0, A1, A2, and A3. The ADG707 switches one of eight differential inputs to a common differential output as determined by the 3-bit binary address lines A0, A1, and A2. An EN input on both devices is used to enable or disable the device. When disabled, all channels are switched OFF.

Low-power consumption and operating supply range of 1.8 V to 5.5 V make the ADG706 and ADG707 ideal for battery-powered, portable instruments. All channels exhibit break-before-make switching action preventing momentary shorting when switching channels. These devices are also designed to operate from a dual supply of ± 2.5 V.

These multiplexers are designed on an enhanced submicron process that provides low-power dissipation yet gives high switching speed, very low ON resistance, and leakage currents. ON resistance is in the region of a few ohms and is closely matched between switches and very flat over the full signal range. These parts can operate equally well as either multiplexers or demultiplexers and have an input signal range that extends to the supplies.

The ADG706 and ADG707 are available in small 28-lead TSSOP packages.

FUNCTIONAL BLOCK DIAGRAMS



PRODUCT HIGHLIGHTS

1. Single-/dual-supply operation. The ADG706 and ADG707 are fully specified and guaranteed with 3 V and 5 V single-supply and ± 2.5 V dual-supply rails.
2. Low ON resistance (2.5 Ω typical)
3. Low-power consumption (< 0.01 μ W)
4. Guaranteed break-before-make switching action
5. Small 28-lead TSSOP package

REV. A

Information furnished by Analog Devices is believed to be accurate and reliable. However, no responsibility is assumed by Analog Devices for its use, nor for any infringements of patents or other rights of third parties that may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Analog Devices.

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781/329-4700
Fax: 781/326-8703
www.analog.com
© Analog Devices, Inc., 2002



MIC5205

150mA Low-Noise LDO Regulator

General Description

The MIC5205 is an efficient linear voltage regulator with ultra low-noise output, very low dropout voltage (typically 17mV at light loads and 165mV at 150mA), and very low ground current (600fA at 100mA output). The MIC5205 offers better than 1% initial accuracy.

Designed especially for hand-held, battery-powered devices, the MIC5205 includes a CMOS or TTL compatible enable/shutdown control input. When shut down, power consumption drops nearly to zero. Regulator ground current increases only slightly in dropout, further prolonging battery life.

Key MIC5205 features include a reference bypass pin to improve its already excellent low-noise performance, reversed-battery protection, current limiting, and overtemperature shutdown.

The MIC5205 is available in fixed and adjustable output voltage versions in a small SOT-23-5 package.

For low-dropout regulators that are stable with ceramic output capacitors, see the μ Cap MIC5245/6/7 family.

Data sheets and support documentation can be found on Micrel's web site at www.micrel.com.

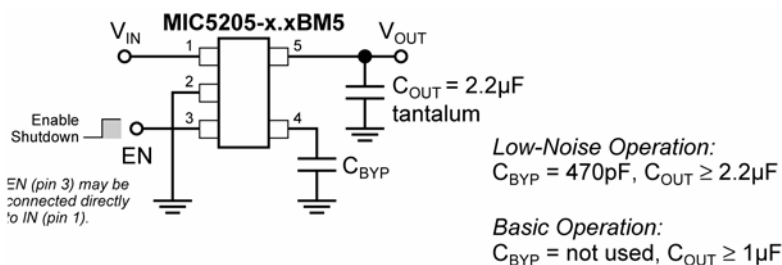
Features

- Ultra-low-noise output
- High output voltage accuracy
- Guaranteed 150mA output
- Low quiescent current
- Low dropout voltage
- Extremely tight load and line regulation
- Very low temperature coefficient
- Current and thermal limiting
- Reverse-battery protection
- "Zero" off-mode current
- Logic-controlled electronic enable

Applications

- Cellular telephones
- Laptop, notebook, and palmtop computers
- Battery-powered equipment
- PCMCIA VCC and VPP regulation/switching
- Consumer/personal electronics
- SMPS post-regulator/dc-to-dc modules
- High-efficiency linear power supplies

Typical Application



Ultra-Low-Noise Regulator Application

Xxxxx is a trademark of Micrel, Inc.
Xxxxx is a registered trademark of Micrel, Inc.

Micrel Inc. • 2180 Fortune Drive • San Jose, CA 95131 • USA • tel +1 (408) 944-0800 • fax + 1 (408) 474-1000 • <http://www.micrel.com>

February 2006

M9999-020806
(408) 955-1690

5 Ld Voltage Supervisors with Adjustable Power-On Reset, Dual Voltage Monitoring or Watchdog Timer Capability

The ISL88011 through ISL88015 family of devices offer both fixed and/or adjustable voltage-monitoring that combine popular functions such as Power On Reset control, Watchdog Timer, Supply Voltage Supervision, and Manual Reset assertion in a small 5 Ld SOT-23 package.

Unique features on the ISL88013 and ISL88015 include a watchdog timer with a 51s startup timeout and a 1.6s normal timeout duration. On the ISL88011 and ISL88014, users can increase the nominal 200ms Power On Reset timeout delay by adding an external capacitor to the C_{POR} pin. Both fixed and adjustable voltage monitors are provided by the ISL88012. Complementary active-low and active-high reset outputs are available on the ISL88011, ISL88012 and ISL88013 devices. All devices provide manual reset capability (see "Product Features Table" on page 4).

Seven preprogrammed reset threshold voltages accurate to $\pm 1.5\%$ over temperature are offered (see "Ordering Information" on page 3). The ISL88012, ISL88014 and ISL88015 have a user-adjustable voltage input available for custom monitoring of any voltage down to 0.6V. All parts are specifically designed for low power consumption and high threshold accuracy.

Features

- Single/Dual Voltage Monitoring Supervisors
- Fixed-Voltage Options Allow Precise Monitoring of +2.5V, +3.0V, +3.3V, and +5.0V Power Supplies
- Dual Supervisor Has One Fixed Voltage Input and Another That is User-Adjustable Down to 0.6V.
- Both RST and $\overline{\text{RST}}$ Outputs Available
- Adjustable POR Timeout Delay Options
- Watchdog Timer With 1.6s Normal and 51s Startup Timeout Durations
- Manual Reset Input on All Devices
- Reset Signal Valid Down to $V_{DD} = 1V$
- Accurate $\pm 1.5\%$ Voltage Threshold
- Immune to Power-Supply Transients
- Ultra Low 5.5 μ A Supply Current
- Small 5 Ld SOT-23 Pb-Free Package
- Pb-Free Plus Anneal Available (RoHS Compliant)

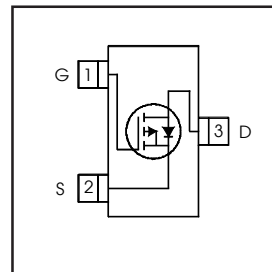
Applications

- Process Control Systems
- Intelligent Instruments
- Embedded Control Systems
- Computer Systems
- Critical μ P and μ C Power Monitoring
- Portable/Battery-Powered Equipment
- PDA and Handheld PC Devices

IRLML6401

HEXFET® Power MOSFET

- Ultra Low On-Resistance
- P-Channel MOSFET
- SOT-23 Footprint
- Low Profile (<1.1mm)
- Available in Tape and Reel
- Fast Switching
- 1.8V Gate Rated

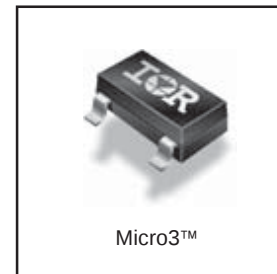


$V_{DS} = -12V$
$R_{DS(on)} = 0.05\Omega$

Description

These P-Channel MOSFETs from International Rectifier utilize advanced processing techniques to achieve extremely low on-resistance per silicon area. This benefit, combined with the fast switching speed and ruggedized device design that HEXFET® power MOSFETs are well known for, provides the designer with an extremely efficient and reliable device for use in battery and load management.

A thermally enhanced large pad leadframe has been incorporated into the standard SOT-23 package to produce a HEXFET Power MOSFET with the industry's smallest footprint. This package, dubbed the Micro3™, is ideal for applications where printed circuit board space is at a premium. The low profile (<1.1mm) of the Micro3 allows it to fit easily into extremely thin application environments such as portable electronics and PCMCIA cards. The thermal resistance and power dissipation are the best available.

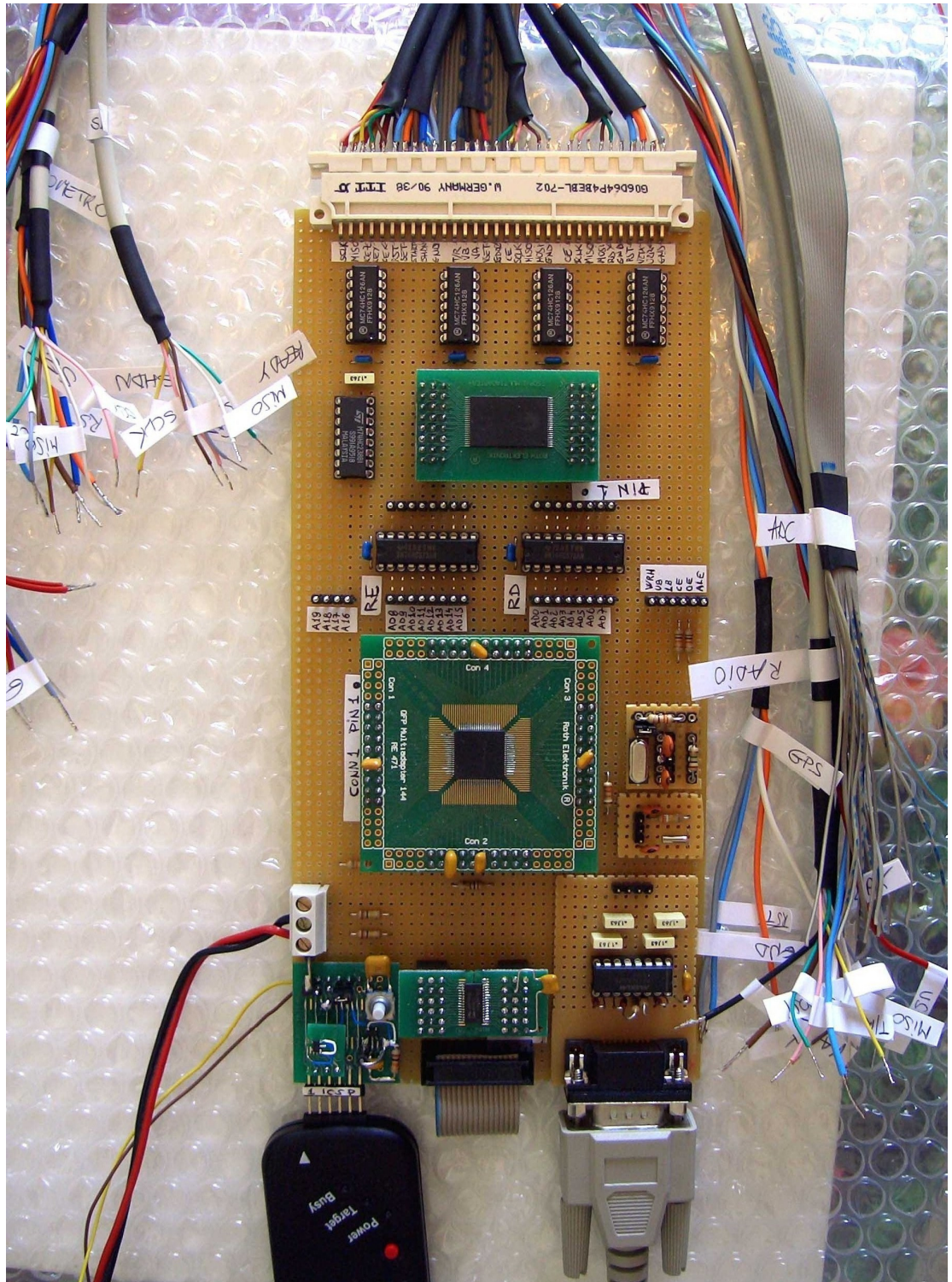


Absolute Maximum Ratings

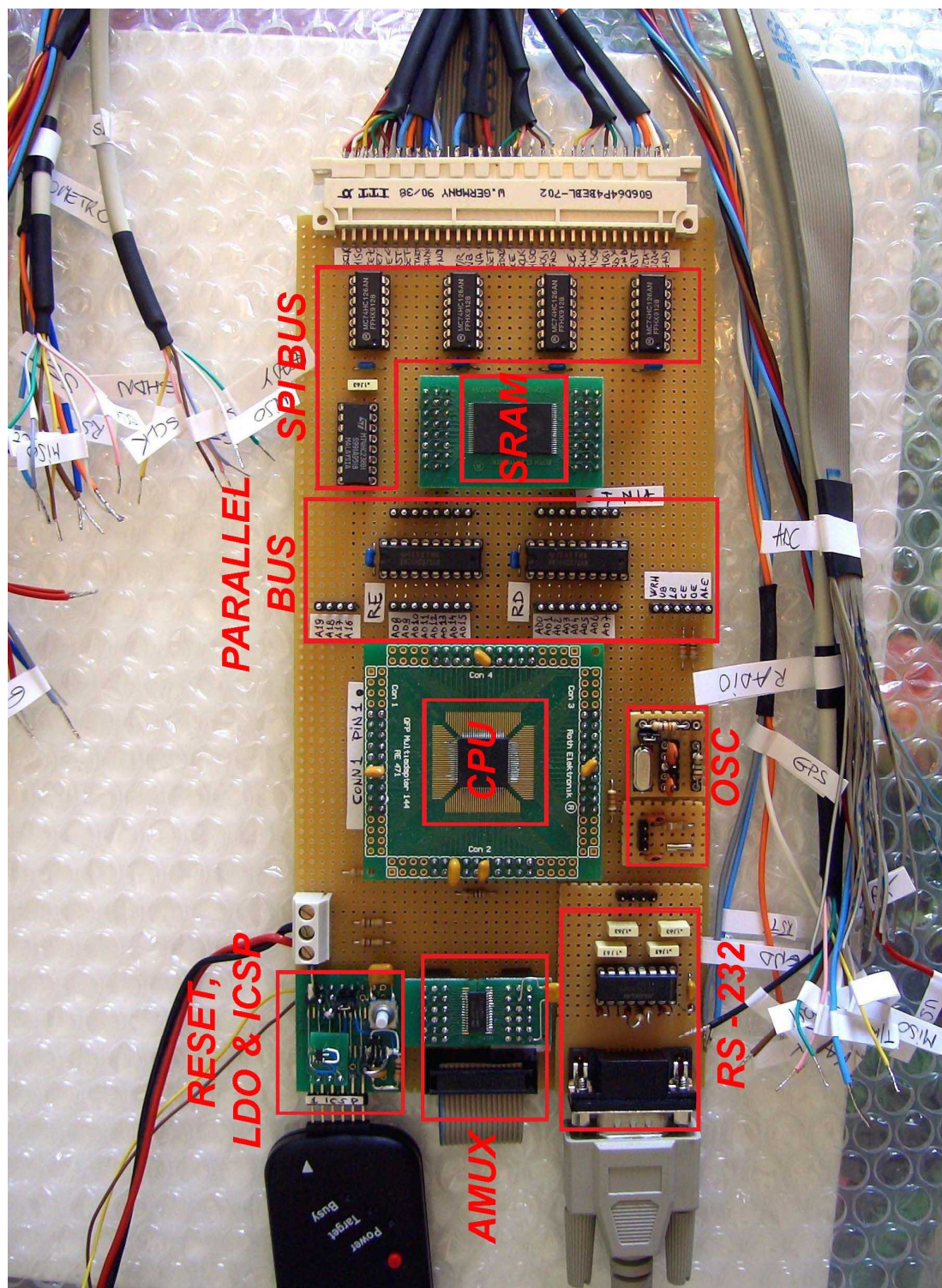
	Parameter	Max.	Units
V_{DS}	Drain- Source Voltage	-12	V
$I_D @ T_A = 25^\circ C$	Continuous Drain Current, $V_{GS} @ -4.5V$	-4.3	A
$I_D @ T_A = 70^\circ C$	Continuous Drain Current, $V_{GS} @ -4.5V$	-3.4	
I_{DM}	Pulsed Drain Current ①	-34	
$P_D @ T_A = 25^\circ C$	Power Dissipation	1.3	W
$P_D @ T_A = 70^\circ C$	Power Dissipation	0.8	
	Linear Derating Factor	0.01	
E_{AS}	Single Pulse Avalanche Energy④	33	mJ
V_{GS}	Gate-to-Source Voltage	± 8.0	V
T_J, T_{STG}	Junction and Storage Temperature Range	-55 to + 150	$^\circ C$

Thermal Resistance

	Parameter	Typ.	Max.	Units
$R_{\theta JA}$	Maximum Junction-to-Ambient③	75	100	$^\circ C/W$



PROTOTIPO DEL MODULO OBDH & HK



PROTOTIPO DEL MODULO OBDAH & HK CON IN EVIDENZA I SOTTOSISTEMI