ALMA MATER STUDIORUM
UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

Scuola di Ingegneria e Architettura

Corso di Laurea Magistrale in Ingegneria Elettronica e Telecomunicazioni per l'Energia

IMPLEMENTAZIONE SIMULINK DI ALGORITMI PER COMUNICAZIONI SATELLITARI E COMPATIBILITÀ CON VHDL

Tesi in
Progetto di Elettronica e Telecomunicazioni LM

Relatore:
Prof. Ing.
ENRICO PAOLINI

Correlatore:
Dr. Ing.
SIMONE MORETTI
Prof. Ing.
GIANNI PASOLINI

Tesi di Laurea di:
EUGENIO BARTOLETTI

SESSIONE III
ANNO ACCADEMICO 2015–2016

PAROLE CHIAVE

C-Band

Standard DVB-S2

Simulink

VHDL

Indice

Introduzione

Negli ultimi anni si sta assistendo allo sviluppo sempre più crescente di missioni satellitari. In particolare, si riscontra un aumento considerevole nell'uso di satelliti di piccole dimensioni (*small satellites*), come ad esempio i cosiddetti *CubeSat*. Questi satelliti vengono adoperati principalmente per missioni di natura scientifica, tuttavia, ESA (European Space Agency) ha intenzione di sviluppare nanosatelliti in grado di fornire servizi commerciali, come ad esempio la connessione Internet, o servizi di natura televisiva. Questi *small satellites* dovranno comunicare con le stazioni a Terra utilizzando frequenze allocate all'uso commerciale via satellite. La banda selezionata da ESA è la banda C, che è stata la prima porzione di spettro pensata per i servizi commerciali via satellite. Nasce quindi la necessità di progettare dispositivi operanti in banda C compatibili con le dimensioni di un nanosatellite, nello specifico il transceiver che si occupa della comunicazione con le stazioni base a Terra. Il lavoro svolto durante la tesi rientra all'interno di un progetto voluto da ESA e denominato progetto SCAT. Esso riguarda lo sviluppo a livello software degli algoritmi di comunicazione con cui verrà programmato il transceiver, in particolare il sistema di trasmissione per il segnale di *Feeder Link* ad elevato data rate. Lo standard di comunicazione utilizzato per la trasmissione del segnale di *Feeder Link* è il DVB-S2, ovvero lo standard ETSI di seconda generazione per la televisione digitale terrestre. Il dispositivo hardware su cui verranno implementati gli algoritmi è un FPGA (Field Programmable Gate Array) programmato in linguaggio VHDL.

Per la generazione dei codici VHDL relativi al sistema di trasmissione DVB-S2, si è pensato di seguire una via alternativa rispetto alla programmazione diretta in linguaggio HDL. L'ambiente di programmazione Simulink permette, infatti, di convertire i propri modelli in linguaggio VHDL mediante un tool denominato *HDL Coder*. Lo scopo principale dell'elaborato è stato, perciò, quello di progettare un modello Simulink che fosse convertibile in codice VHDL, in modo da programmare l'FPGA direttamente con il codice derivante da Simulink. Il lavoro di tesi si è concentrato essenzialmente sul rispetto dei requisiti necessari a rendere possibile questo passaggio, riservan-

do la fase di ottimizzazione del sistema al task successivo del progetto. In parallelo alla progettazione del modello compatibile con l'*HDL Coder*, si è reso necessario lo sviluppo di un modello di riferimento che permettesse la valutazione delle prestazioni del sistema e un confronto sui segnali generati con la catena di trasmissione convertibile in VHDL.

La tesi è suddivisa nei seguenti capitoli:

- Capitolo 1: Il primo capitolo riguarda una panoramica globale del progetto SCAT, partendo dallo scenario complessivo, passando per la descrizione specifica del mio compito all'interno del progetto e terminando con le considerazioni effettuate riguardo alla scelta dello standard di comunicazione.

- Capitolo 2: Viene fornita una descrizione dettagliata dello standard DVB-S2 in trasmissione, in particolare su tutte le sezioni dello standard che sono state sviluppate in Simulink.

- Capitolo 3: Il capitolo centrale dell'elaborato provvede ad una descrizione generale dell'ambiente di programmazione Simulink, concentrandosi sullo strumento necessario alla conversione in codice VHDL, ossia l'*HDL Coder*

- Capitolo 4: Questo capitolo racchiude la spiegazione delle catene implementate in Simulink (quella di riferimento per le simulazioni e quella convertibile in VHDL), oltre ai principi su cui ci si è basati nel progettarle.

- Capitolo 5: Il capitolo finale fornisce i risultati raccolti durante lo svolgimento della tesi. Oltre alla realizzazione di un modello compatibile coi linguaggi HDL, si sono verificate le prestazioni dei codici di canale BCH e LDPC al variare della modulazione digitale e lo spettro del segnale trasmesso se soggetto ad errore di quantizzazione.

Capitolo 1

Progetto SCAT: C-Band Transceiver for small satellites

Il numero di missioni spaziali che impiegano small satellites ha riscontrato un notevole incremento negli ultimi anni, in particolare a livello europeo. L'impiego di nanosatelliti coinvolge principalmente attività di ricerca scientifiche, tuttavia, ha ultimamente attratto l'interesse dei provider commerciali. Le notevoli funzionalità e la compattezza di questi dispositivi permettono di fornire servizi di uso comune come, ad esempio, connessioni ad Internet a banda larga e la trasmissione digitale in broadcasting. Un elevato numero di queste missioni satellitari di tipo commerciale utilizza frequenze allocate all'uso radioamatoriale, complicando l'impiego delle stesse a terra a causa delle interferenze introdotte. Per risolvere questa problematica, l'Agenzia Spaziale Europea (ESA) ha dato il via ad un progetto, chiamato progetto SCAT, nel quale l'Università di Bologna è partner di supporto. Lo scopo del progetto è quello di elaborare, sviluppare ed infine installare un transceiver di dimensioni ridotte operante in banda C all'interno di nanosatelliti orbitanti sulle cosiddette *LEO* (Low Earth Orbit). Le 'orbite terrestri basse', fanno parte della categoria delle orbite non geostazionarie e rappresentano le orbite più vicine alla superficie terrestre. La banda C è stata la prima porzione di spettro pensata per i servizi commerciali via satellite e dopo la WRC-15 (World's Radio Conference del 2015), questo range di frequenze è diventato permanentemente disponibile come feeder link per MSS (Mobile Satellite System) non geostazionari. Nonostante questi fattori, le tecnologie attuali a livello europeo non permettono l'immediata installazione e utilizzo di dispositivi in banda C su satelliti di dimensioni ridotte, come i cosiddeti *CubeSat*. L'utilizzo sempre più frequente di nanosatelliti ha incentivato perciò l'idea di progettare transceiver di dimensioni compatibili con i dispositivi in questione. L'impiego della banda C rispetto allo spettro radioamatoriale comporta

notevoli vantaggi. Tuttavia, tali applicazioni sono in fase sperimentale e un largo utilizzo potrà essere sfruttato risolvendo alcune problematiche attualmente limitanti. Una delle problematiche più comuni dipende dal fatto che queste frequenze vengono già usate da tempo per collegamenti di trasmissione terrestri a microonde. Le trasmissioni terrestri possono avvenire in porzioni di banda sovrapposte o adiacenti causando fenomeni di interferenza e la conseguente degradazione della comunicazione satellitare. Per questo motivo l'impiego della banda C per applicazioni satellitari commerciali è consigliato solamente quando la stazione a terra in comunicazione col satellite si trova in zone remote e lontane da possibili interferenze terrestri. Un notevole vantaggio della banda C, rispetto alle altre frequenze usate per le comunicazioni satellitari, è che risulta più robusta all'attenuazione dovuta ai fenomeni atmosferici. Queste ragioni hanno reso le trasmissioni satellitari in banda C ideali nell'offrire servizi satellitari alle applicazioni marittime.

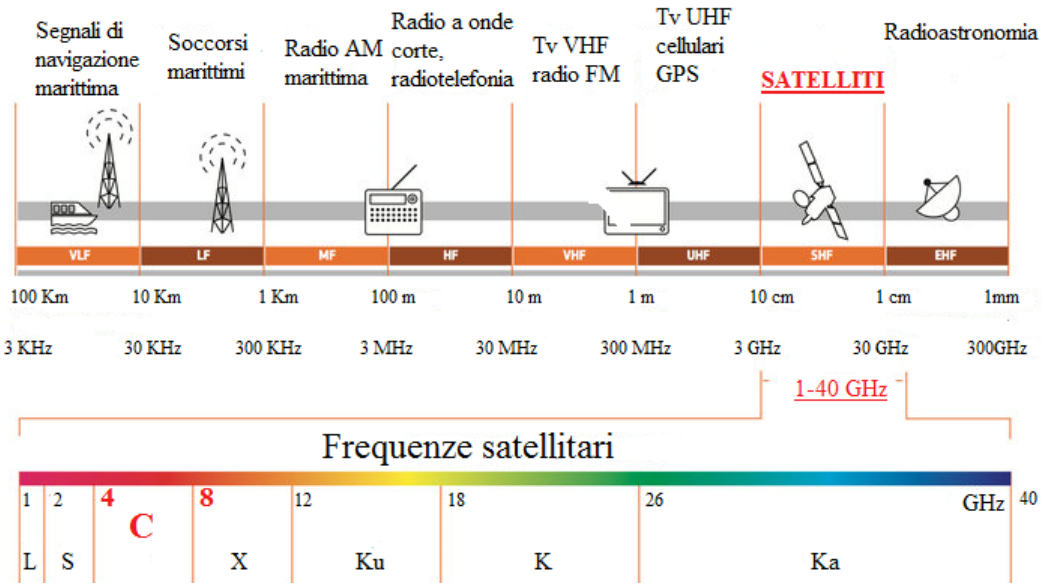

Figura 1.1: La suddivisione delle frequenze radio e le diverse applicazioni. In basso la porzione dedicata alle comunicazioni satellitari.

In figura 1.1 sono mostrate tutte le bande di frequenza utilizzate per le trasmissioni satellitari, in particolare la banda C compresa tra i 4 e gli 8 GHz [1]. Questo range di valori standard definisce il limite minimo e massimo entro cui è compresa la banda, tuttavia nelle comunicazioni satellitari i valori di frequenza vengono definiti in maniera più specifica, in particolare distinguendo le tratte di *downlink* da quelle di *uplink* e associando a questi segnali dei valori ben specifici e distinti tra loro. Con il termine *downlink* si inten-

de la tratta da satellite verso Terra, corrispondente alla fase di trasmissione del transceiver. Con il termine *uplink* si intende il percorso inverso, ovvero la fase di ricezione del segnale da parte del transceiver. Lo scenario che ho dovuto affrontare durante lo svolgimento della mia tesi, prevede la presenza di tre segnali distinti: *Feeder Link*, *Telemetria (TM)* e *Telecomando (TC)*. Il *Feeder Link* rappresenta il segnale relativo allo scambio di informazioni ad elevato *data rate*. Per un corretto funzionamento del satellite e più in particolare del ricetrasmettitore presente all'interno di esso, c'è bisogno anche di segnali aggiuntivi che permettano la gestione del satellite (TT&C). Con la sigla TT&C si intende Telemetry, Tracking & Command, ovvero l'insieme dei segnali in uplink e downlink che consentono la fondamentale gestione del satellite dalla stazione base terrestre. Questo avviene mediante lo scambio di informazioni come ad esempio i valori misurati dai sensori nel satellite, il posizionamento di quest'ultimo o i segnali di riconfigurazione del satellite da parte della stazione a terra. Tipicamente la funzionalità di tracking risulta facoltativa, e quindi il progetto SCAT si basa esclusivamente sui segnali di *Telecomando* in uplink e di *Telemetria* in downlink. Questi due segnali, essendo di controllo e gestione del satellite, hanno una *data rate* molto bassa, in quanto il loro utilizzo è differente rispetto al *Feeder Link*.

Lo scenario definitivo su cui si basa il progetto SCAT è quello raffigurato in figura 1.2, nel quale la funzione di *Feeder Link* (downlink) è allocata nella stessa banda C delle funzionalità di *Telecomando* (uplink) e *Telemetria* (downlink). Inoltre, lo scenario complessivo può includere la presenza di più satelliti e stazioni a terra (G/S) rendendo necessario il riuso delle risorse a radiofrequenza. Quello di cui il progetto SCAT non deve occuparsi è, invece, la comunicazione con i terminali mobili. In aggiunta ai valori di frequenza associati ai tre segnali precedenti, il progetto si basa su altri vincoli relativi all'impiego degli stessi. In particolare, i segnali di *Feeder Link* e TM non devono essere attivi nello stesso tempo e la ricezione del TC deve essere disponibile costantemente e caratterizzata da un alto tasso di affidabilità. L'alternanza tra la trasmissione del *Feeder Link* e del segnale di *Telemetria* viene gestita mediante un comando d'istruzione da Terra. L'Agenzia Spaziale Europea ha poi fornito indicazioni riguardo ai desiderati valori di *data rate* (per tutte e tre le funzionalità del satellite) che il progetto SCAT dovrà rispettare implementando il ricetrasmettitore. Con il termine *data rate* si definisce il valore di bit rate a valle della modulazione digitale. I vincoli da rispettare sono i seguenti:

- *Telecomando*

    1. elevato data rate di almeno 128 Kbit/s

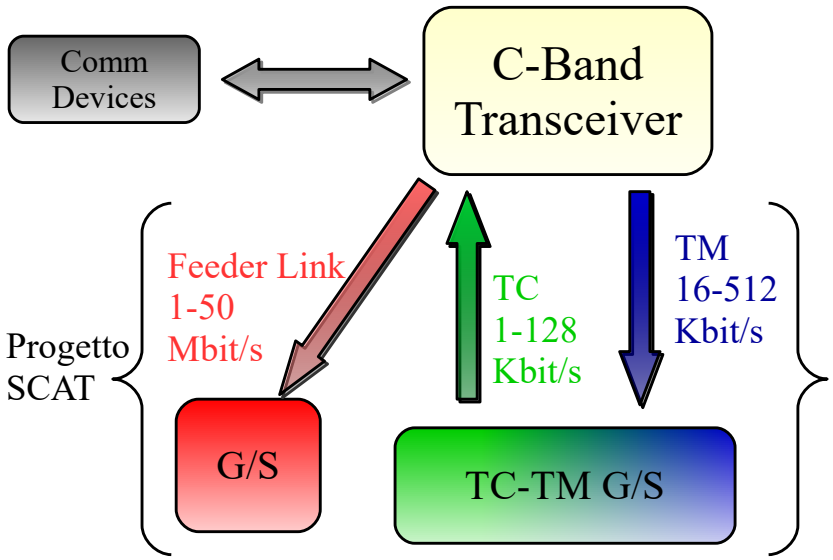

Figura 1.2: L'architettura principale del progetto SCAT

2. basso data rate di almeno 1 Kbit/s

- *Telemetria*

1. elevato data rate di almeno 512 Kbit/s
2. basso data rate di almeno 16 Kbit/s

- *Feeder Link*

1. elevato data rate di almeno 50 Mbit/s
2. basso data rate di almeno 1 Mbit/s

La scelta del sistema di comunicazione utilizzato per elaborare i tre segnali di interesse dipenderà essenzialmente dal tipo di codifica di canale e dalla modulazione digitale. Il sistema dovrà avere un buon compromesso tra le prestazioni (correzione degli errori, efficienza spettrale, etc) e la complessità. Inoltre, un'altra funzionalità considerata all'interno del progetto SCAT, riguarda la possibilità di riprogrammare il sistema in funzione delle condizioni variabili del canale satellitare. Questo requisito interessa principalmente il

segnale di *Feeder Link*. Le considerazioni precedenti rappresentano il fattore stimolante che ha portato alla scelta dello standard di comunicazione satellitare utilizzato nel progetto, e su cui mi sono concentrato durante lo sviluppo dell'elaborato. La descrizione del sistema di comunicazione implementato è stata fatta nel secondo capitolo dell'elaborato.

1.1 Sviluppo della parte digitale del transceiver

La realizzazione di un apparato di comunicazione satellitare comprende lo sviluppo di differenti segmenti di lavoro. La mia attività durante lo svolgimento della tesi si è limitata, tuttavia, alla sola parte digitale del transceiver. La circuiteria digitale del transceiver a livello hardware verrà realizzata implementando su *FPGA* il codice sorgente per il processamento dei segnali. Con il termine *FPGA (Field Programmable Gate Array)* si intende un circuito integrato le cui funzionalità sono programmabili via software. In questo caso, inoltre, l'*FPGA* deve essere interfacciato adeguatamente con le restanti parti del ricetrasmettitore, come ad esempio i convertitori A/D e D/A.

Gli algoritmi implementati su FPGA riguardano, nello specifico, il sistema di comunicazione in banda base e frequenza intermedia (IF). La conversione da *IF* a *RF* avviene a livello analogico e non fa parte dei compiti che ho svolto all'interno del progetto SCAT. Il diagramma raffigurato in figura 1.3, mostra l'interfacciamento tra l'FPGA e la restante componentistica che va a completare la parte digitale del transceiver, oltre all'interfacciamento con il Computer di Bordo. In particolare, l'FPGA sarà dotato delle seguenti interfacce:

- Segnale a IF in ingresso. Questo è il segnale a frequenza intermedia in ingresso proveniente dal modulo a RF.

- Segnale a IF in uscita. Segnale in uscita a frequenza intermedia trasmesso al modulo a RF.

- CAN BUS. Questa è l'interfaccia di controllo e di comando per il Computer di Bordo (OBC, On Board Computer), la cui gestione non fa parte del progetto SCAT.

- Segnali di controllo per il modulo a RF. Insieme di segnali utilizzati per il controllo delle diverse funzionalità implementate nel modulo a radiofrequenza, includendo possibilmente:

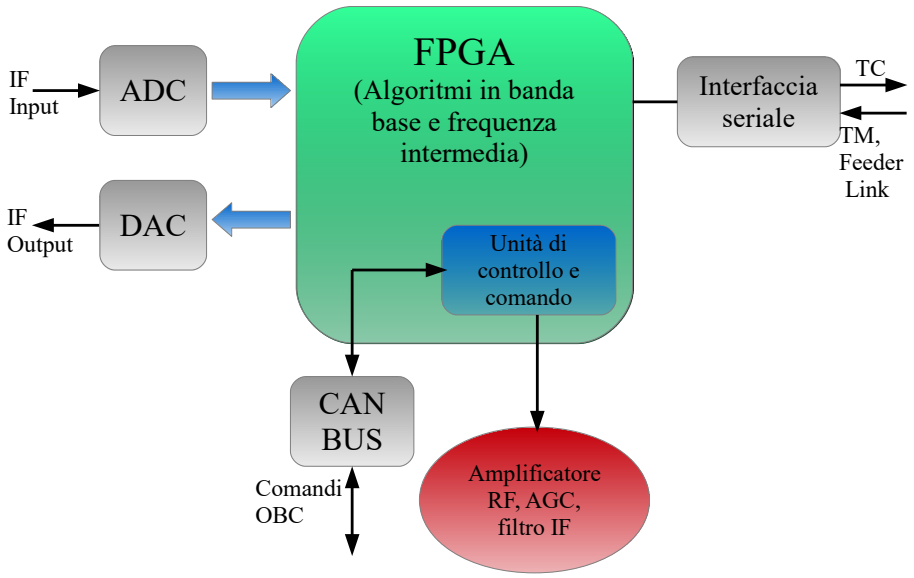

Figura 1.3: Architettura per la parte di comunicazione, composta da FPGA e relative interfacce

1. Comando per l'amplificatore a RF, un comando per selezionare il guadagno dell'amplificatore a radiofrequenza tra un set di valori prefissati.
2. Regolazione filtro IF, per modificare il filtro a frequenza intermedia allo scopo di ottimizzare la ricezione del segnale TC o la trasmissione dei segnali TM e *Feeder Link*
3. Comando per l'AGC, per variare il guadagno dell'AGC (Automatic Gain Control) esterno allo scopo di ottimizzare la dinamica disponibile del segnale senza limitazioni.

- Connessione seriale bidirezionale con OBC. Un'interfaccia seriale che supporta una comunicazione full-duplex con il computer di bordo e permette il trasferimento dei dati ricevuti o da trasmettere.

L'obiettivo della mia tesi si è focalizzato, quindi, nel progettare algoritmi a livello software che rappresentassero il sistema di comunicazione satellitare scelto, in particolare quello di trasmissione. Per realizzare gli algoritmi con cui programmare l'FPGA è stata scelta una via alternativa che non

prevede di programmare direttamente l'hardware a disposizione. La strada alternativa consiste nell'impiego di Simulink, un software ottimizzato per la modellazione, simulazione e analisi di sistemi dinamici di vario tipo, sviluppato da MathWorks e strettamente integrato con MATLAB. Il vantaggio nell'uso di Simulink, risiede nel suo ambiente di programmazione, basato sull'utilizzo di blocchi spesso preesistenti nelle librerie. La presenza di nutrite librerie Simulink permette di implementare in maniera rapida il proprio sistema per una preliminare simulazione e analisi dell'algoritmo scelto. La scelta di questo software, tuttavia, è stata presa soprattutto grazie ad una potenzialità molto utile che fornisce Mathworks, ossia la presenza di un tool che permette, partendo da un modello realizzato in Simulink o MATLAB, di generare un codice *VHDL* o Verilog. Con *VHDL* e Verilog si intendono due esempi di cosiddetti linguaggi di descrizione dell'hardware. Mathworks permette inoltre di verificare il corretto funzionamento del proprio modello con un altro tool chiamato *HDL Verifier*. Lo strumento di MathWorks che consente questa operazione è chiamato *HDL Coder* (dove la sigla HDL sta proprio per Hardware Description Language), e rappresenta il punto focale dell'intero elaborato, in quanto il suo utilizzo rappresenta il punto di congiuntura tra il mondo software e ideale di Simulink, e il mondo hardware dell'FPGA. L'effettivo compito che ho eseguito durante lo svolgimento della tesi è stato perciò quello di implementare gli algoritmi Simulink relativi al sistema di comunicazione scelto, per poi convertire il codice Simulink in codice VHDL con cui poter programmare l'FPGA. In particolare, lo sviluppo di questi sistemi di comunicazione riguarda tutte le operazioni che vengono realizzate partendo dall'elaborazione dello stream binario di dati provenienti dalla seriale (quindi dall'OBC) fino alla trasmissione del segnale digitale a frequenza intermedia verso il modulo analogico del transceiver (le operazioni opposte in fase di ricezione). Facendo riferimento a figura 1.3, in ambiente Simulink non mi sono occupato di realizzare l'unità di controllo e comando con cui interfacciare la componentistica a RF del transceiver.

1.2 Scelta degli Standard di Comunicazione Satellitare

La decisione finale relativa ai protocolli di comunicazione da utilizzare non è stata immediata e ha portato a considerare diverse possibili opzioni. Questa variabilità nella scelta degli standard di comunicazione è dovuta in primo luogo alla possibilità di impiegare protocolli diversi a seconda del segnale preso in considerazione (*Feeder Link*, TC o TM) e quindi a seconda del flusso

dati associato. In secondo luogo, è dovuta al fatto che gli small satellites pensati da ESA serviranno alla diffusione di servizi commerciali per aree isolate e difficilmente raggiungibili, rendendo necessario tenere in considerazione la presenza di stazioni a Terra già adattate alla ricezione o alla trasmissione dello standard usato. L'idea di base da cui il progetto SCAT è partito è la seguente:

- Standard ECSS per i segnali di TM e TC.

- Standard DVB-S o DVB-S2 per il segnale di *Feeder link*.

Tuttavia, durante lo svolgimento della tesi, mi sono concentrato sulla realizzazione del solo sistema di trasmissione relativo al segnale di *Feeder Link* in downlink. Per questo motivo, l'argomento principale dell'elaborato è il protocollo che è stato approvato da ESA per implementare il segnale ad elevato *data rate*: lo standard DVB-S2. Alcune considerazioni vanno fatte però anche sugli standard che inizialmente erano stati pensati ma che poi, dopo attente valutazioni (in accordo con ESA) sono stati sostituiti, o comunque non implementati a livello software.

1.2.1 ECSS Standard

In primo luogo si era pensato all'utilizzo, per i segnali di controllo del transceiver, ovvero *Telecomando* e *Telemetria*, di impiegare protocolli regolamentati dall'ECSS (European Cooperation for Space Standardization). ECSS è un'organizzazione che opera allo scopo di standardizzare e ottimizzare il settore spaziale europeo e collabora frequentemente con l'Agenzia Spaziale Europea, tant'è che i contraenti che lavorano per conto di ESA spesso si basano sugli standard forniti da ECSS. Per questo motivo, l'implementazione di questi standard era sembrata una possibile scelta da attuare nelle fasi iniziali del progetto.

In figura 1.4 è mostrato uno schema complessivo raffigurante l'architettura software relativa all'FPGA (analoga a quella di figura 1.3), nel caso in cui vengano utilizzati protocolli inerenti alle regole fornite da ECSS. Come si può vedere dalla figura, la catena in trasmissione, quindi relativa alla elaborazione del segnale di Telemetria sarà basata sui seguenti sottosistemi:

1. Codifica di canale, a sua volta suddivisa in:

    - RS Encoder
    - Encoder Convoluzionale

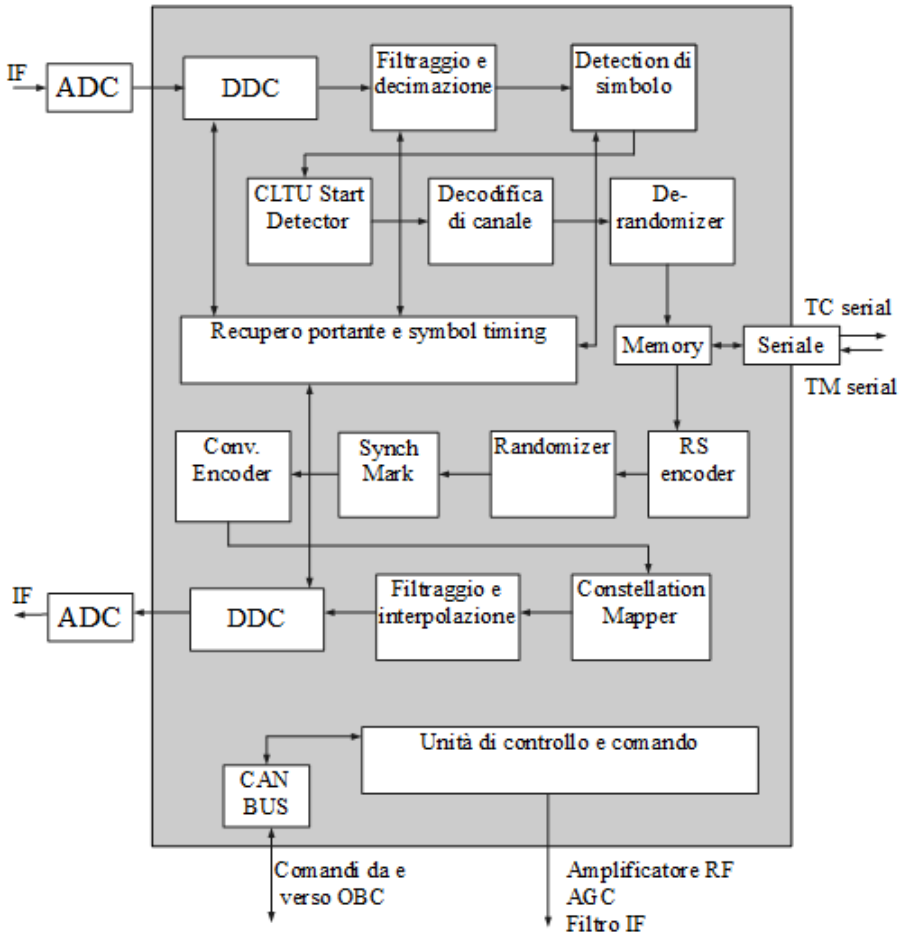

Figura 1.4: Schema a blocchi dell'architettura software comprendente gli algoritmi basati sugli standard ECSS

2. Randomizer, che si occupa di 'randomizzare' la sequenza di bit.

3. Synch Mark, ovvero la concatenazione alla trama di un prefisso noto, utile alla sincronizzazione di frame in ricezione

4. Constellation Mapper, la modulazione digitale scelta.

5. Filtraggio e Interpolazione, ovvero il formatore d'impulsi con annessa interpolazione digitale.

6. DUC, il Digital Up Converter necessario al passaggio del segnale dalla banda base alla frequenza intermedia.

I punti dell'elenco precedente dal primo al terzo, si basano sulle regole standardizzate da ECSS descritte in [2], mentre la tipologia di modulazione digitale scelta è basata su [3] e il filtraggio del segnale in banda base si appoggia sui criteri descritti in [4]. Nel diagramma di figura 1.4 è mostrata anche la catena in ricezione del segnale di Telecomando, nella quale sono rappresentati i sottosistemi relativi alle varie operazioni di detection di simbolo, recupero della portante, sincronizzazione di frame, demodulazione e decodifica, descritti sempre negli standard ECSS [3] e [5]. Come anticipato precedentemente, tuttavia, si è convenuto nel non utilizzare protocolli ECSS né per quanto riguarda i segnali di controllo e nemmeno nel caso del *Feeder Link*. Essenzialmente, la motivazione principale riguarda il fine effettivo per cui questo transceiver verrà realizzato, ovvero rendere fruibili servizi commerciali (come la connessione Internet, per esempio) in zone remote del globo o difficilmente raggiungibili da stazioni base terrestri (applicazioni marittime). Gli standard ECSS, infatti, non vengono solitamente usati per applicazioni di questo tipo, ma piuttosto per missioni satellitari di natura scientifica. La scelta è ricaduta quindi su degli standard di comunicazione satellitare frequentemente usati per la trasmissione di segnali in modalità broadcast: lo standard *DVB-S* e il suo successore, lo standard *DVB-S2*.

1.2.2 Standard DVB-S

Come già anticipato in precedenza, lo standard effetivamente implementato durante lo svolgimento della tesi è stato il *DVB-S2*. Prima di arrivare alla decisione definitiva, tuttavia, alcune valutazioni sono state fatte anche sullo standard *DVB-S*. Questo standard rappresenta il primo sistema di trasmissione satellitare impiegato per la diffusione della televisione satellitare ad alta definizione, realizzato a partire dal 1993 dal consorzio europeo DVB (Digital Video Broadcasting). Il sistema prevede la trasmissione in modalità broadcast di un flusso digitale audio/video della famiglia MPEG-2 (Moving Picture Experts Groups-2). Con il termine MPEG-2 si intende un sistema di codifica digitale che definisce la codifica di sorgente, ovvero la compressione audio, video e il formato di multiplazione e trasporto per servizi multimediali diffusivi a qualità superiore. La parte dello standard di interesse per una possibile implementazione del sistema all'interno del progetto SCAT non riguarda tuttavia la codifica di sorgente e l'elaborazione delle trame MPEG-2, bensì le operazioni successive a livello fisico. La generazione delle trame MPEG-2 è compito invece del Computer di Bordo (OBC). Come descritto in [6], l'algoritmo di trasmissione dello standard DVB-S non differisce molto da quello previsto dallo standard ECSS precedentemente illustrato.

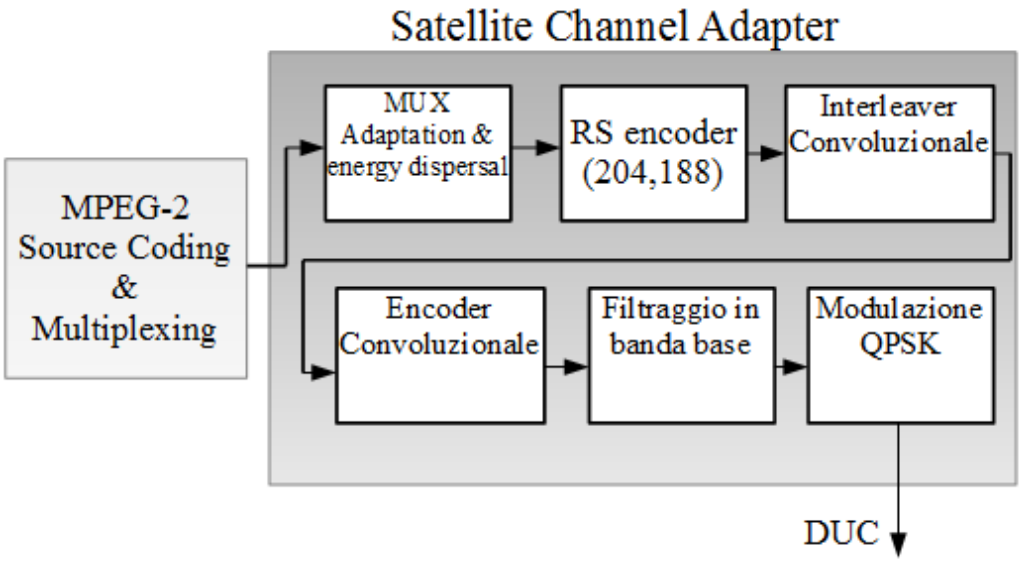

Figura 1.5: Schema a blocchi del sistema DVB-S

Se si osserva la figura 1.5, concentrandosi sulla parte destra dello schema a blocchi, chiamata *Satellite Channel Adapter*, si identificano i blocchi funzionali che compongono il sistema di trasmissione a livello fisico. In particolare:

- MUX adaptation & energy dispersal, ovvero il blocco che effettua lo scrambling della trama MPEG-2 in arrivo dalla seriale, randomizzando il flusso di dati.

- RS (204,188) Encoder, il codificatore di canale sistematico a blocchi esterno che si occupa di aggiungere 16 byte di ridondanza alla frame.

- Interleaver Convoluzionale, tecnica di interleaving che associata alla codifica Reed-Solomon rende la trasmissione più robusta a possibili errori a burst.

- Encoder Convoluzionale, codificatore convoluzionale esterno, possibilmente punturato.

- Filtraggio in banda base, ovvero un filtro a radice di coseno rialzato che funge da sagomatore del segnale in banda base.

- Modulazione QPSK, effettua il mapping dei bit in ingresso e genera la via in fase e quadratura.

L'impiego di questo standard per la trasmissione del segnale ad elevato *data rate*, ovvero il *Feeder Link*, risulta non ideale e come anticipato precedentemente non è stato usato. I motivi sono essenzialmente tre:

- Una capacità trasmissiva ridotta, soprattutto rispetto ai più moderni sistemi di trasmissione satellitare.

- Si tratta di un protocollo poco flessibile da un punto di vista della variabilità del sistema di codifica di canale e modulazione. Questo punto risulta molto svantaggioso per applicazioni installate su satelliti orbitanti in orbite LEO, in quanto si possono avere durante l'arco della giornata variazioni notevoli nelle condizioni del canale satellitare.

- Nonostante tutt'ora la maggior parte delle ground station disponga di decoder in grado di ricevere segnali DVB-S, il rischio è che questo standard, ancora molto usato, ma comunque datato, diventi obsoleto col tempo.

La soluzione a tutte queste limitazioni riscontrate nel DVB-S, prende il nome di standard DVB-S2 e verrà trattato nel seguente capitolo.

Capitolo 2

Standard DVB-S2

Il protocollo denominato DVB-S2, altro non è che il sistema di seconda generazione per la trasmissione satellitare definito dal consorzio DVB, ovvero l'evoluzione dello standard DVB-S. Il sistema DVB-S2 è stato progettato per varie applicazioni satellitari a banda larga, quali: servizi di diffusione televisivi a definizione standard ma anche ad alta definizione (HDTV, High Definition Television), applicazioni interattive per l'utenza professionale, compreso l'accesso ad internet, servizi professionali di contribuzione TV e SNG (Satellite News Gathering), distribuzione di segnali TV a trasmettitori digitali terrestri VHF/UHF, distribuzione di dati e di siti Internet (Internet trunking) [7]. Le caratteristiche che caratterizzano il DVB-S2 e lo rendono molto più performante rispetto a quello di prima generazione, sono un'aumentata capacità trasmissiva e una maggiore variabilità dal punto di vista dei parametri di modulazione e codifica. Questo è dovuto all'utilizzo di una tipologia di codici di canale non implementabili a livello hardware ai tempi del DVB-S su dispositivi di dimensioni limitate, ovvero i codici LDPC (Low Density Parity Check), e alla possibilità di impiegare quattro diversi tipi di modulazione digitale (QPSK, 8-PSK, 16-APSK, 32-APSK) combinandoli con diversi valori di R_c (code rate) dell'encoder LDPC. Il sistema DVB-S2 può prevedere tre diverse tipologie di funzionamento:

- **CCM**:Constant Coding & Modulation, ovvero un sistema statico in cui non si modificano né modulazione digitale, né parametri dei codici di canale.

- **VCM**:Variable Coding & Modulation, ossia un sistema che prevede, mediante la ricezione di un comando esterno, la possibilità di variare parametri di modulazione e codifica.

- **ACM**:Adaptive Coding & Modulation, ovvero un sistema più complesso che mediante, ad esempio, la stima di canale è in grado di modificare dinamicamente la combinazione di modulazione e codifica; così da riuscire ad avere le prestazioni migliori in termini di efficienza spettrale e di potenza al variare delle condizioni del canale satellitare.

EIRP (dBW)	**51**		**53.7**	
Sistema	DVB-S	DVB-S2	DVB-S	DVB-S2
Mod & Cod	QPSK 2/3	QPSK 3/4	QPSK 7/8	8PSK 2/3
V di simbolo (Mbaud)	27.5 (α=0.35)	30.9 (α=0.20)	27.5 (α=0.35)	29.7 (α=0.25)
C/N (dB)	5.1	5.1	7.8	7.8
Bit-rate utile (Mbit/s)	33.8	46 (gain=36%)	44.4	58.8 (gain=32%)

Tabella 2.1: Esempio di confronto tra sistemi DVB-S e sistemi DVB-S2 in termini di capacità trasmissiva(bit-rate utile)

All'interno del progetto SCAT, si è scelto di adottare una soluzione VCM che comprenda la possibilità di modificare la configurazione trasmissiva variando la modulazione e la code rate dell'encoder LDPC. La possibilità di modificare i parametri di sistema a seconda delle condizioni di canale, permette di aumentare ancora il guadagno in termini di capacità di trasmissione, che già in condizioni statiche (CCM) è molto elevato. In tabella 2.1, è mostrato il confronto tra un sistema DVB-S e un sistema DVB-S2 aventi lo stesso valore di EIRP (Equivalent Isotropic Radiated Power) in due situazioni differenti [7]. Le performance dei sistemi DVB-S2 sono nettamente migliori in termini di bit-rate utile e presentano un guadagno superiore al 30%. Questo miglioramento nelle prestazioni è uno dei motivi principali, elencati nel capitolo precedente, per cui lo standard DVB-S2 è stato scelto a discapito del DVB-S per il *Feeder Link*.

2.1 Sistema di Trasmissione

Per descrivere nel dettaglio la catena in trasmissione di un sistema DVB-S2 è stato preso in considerazione lo standard ufficiale fornito da *ETSI* (European Telecommunications Standards Institute) [8]. Come già anticipato in precedenza, il progetto SCAT si basa su una versione VCM del sistema DVB-S2, ignorando perciò le differenti problematiche da tenere in considerazione nei sistemi ACM. Inoltre, in accordo con l'Agenzia Spaziale Europea,

è stato deciso di realizzare un sistema di trasmissione che non preveda l'implementazione delle operazioni definite 'non rilevanti' nello standard stesso, in riferimento alle applicazioni broadcast. Il tailoring dello standard ha riguardato in particolare le funzionalità che è possibile osservare tratteggiate nella figura seguente: *Padder* e *Dummy PLFRAME Insertion*.

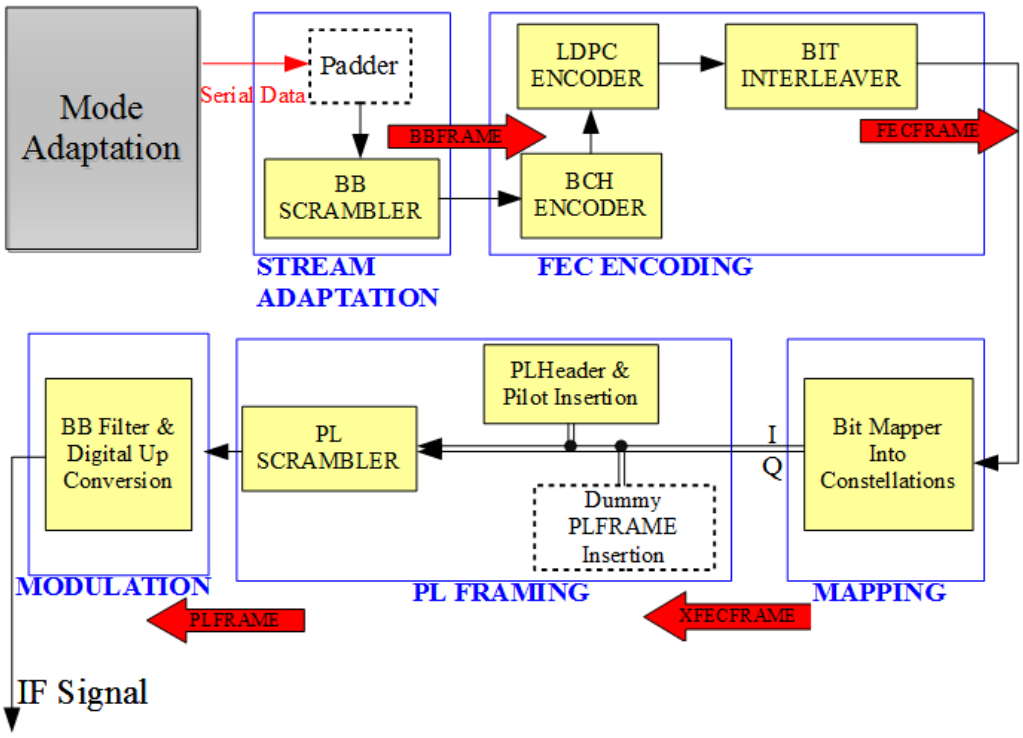

Figura 2.1: Schema a blocchi funzionale del sistema di trasmissione DVB-S2

In figura 2.1, è mostrato lo schema a blocchi di un sistema di trasmissione DVB-S2. Lo standard suddivide la catena di trasmissione in diverse sezioni. La parte denominata *Mode Adaption* in figura, e definita in [8], non riguarda gli algoritmi da implementare all'interno dell'FPGA in quanto si occupa della codifica di sorgente e dell'elaborazione dei dati a livello superiore. In particolare, all'interno del satellite, sarà compito dell'On Board Computer gestirla. Le sezioni dello standard che riguardano il mio lavoro di tesi sono quelle scritte in blu sempre in figura 2.1, ovvero:

- *Stream Adaptation*. Questo blocco si occupa di definire le trame di bit in ingresso alla sezione successiva, partendo dallo stream seriale in arrivo dal *Mode Adaptation*. Nello specifico si ha una randomizzazione della frame, che in uscita dal sottosistema prende il nome di *BBFrame*.

- *FEC Encoding.* Viene realizzata la codifica di canale sulla *BBFrame*, aggiungendo i bit di ridondanza necessari alla correzione degli errori in ricezione. Infine avviene un'operazione di interleaving sulla codeword. La trama in uscita dalla sezione di *FEC Encoding* ha il nome di *FECFRAME*.

- *Mapping.* In questa sezione, i bit in ingresso vengono mappati sulla costellazione nel piano complesso a seconda del tipo di modulazione in banda base scelta. In uscita si hanno trame composte da numeri complessi, rappresentati da due vie, una in fase (I) e una in quadratura (Q), che prendono il nome di *XFECFRAME*.

- *PL Framing.* Vengono composte le trame definitive a livello fisico, concatenando alla frame in ingresso un header fisico e dei toni pilota facoltativi, e realizzando un ulteriore operazione di scrambling (sui simboli complessi). La trama in uscita prende il nome di *PLFRAME*.

- *Modulation.* Nell'ultimo blocco si ha il filtro sagomatore e l'interpolazione in banda base; poi la conversione a frequenza intermedia. Quest'ultimo punto non è in realtà descritto nello standard, ma rappresenta un punto necessario per la trasmissione di un segnale a IF.

2.2 Stream Adaptation

Il sottosistema chiamato *Stream Adaptation* rappresenta la parte iniziale dello standard da programmare su FPGA. Esso riceve uno stream binario di dati dalle applicazioni a livello superiore, nel caso del progetto SCAT, da parte del Computer di Bordo. Nello standard ufficiale, è composta da due blocchi, il primo chiamato *Padder* e il secondo *BB Scrambler*. Lo scopo di questa prima sezione della catena di trasmissione è quello di formare la trama in ingresso all'encoder BCH, di dimensione costante K_{bch} e chiamata *BBFRAME*. Per fare questo, il blocco *Padder* aggiunge il numero necessario di bit a '0' in modo che la somma dell'header binario di 80 bit (BBHeader), del campo dati (Data Field) e dei bit di Padding raggiunga il valore K_{bch}. Il *BB Scrambler* si occupa invece di randomizzare l'intera trama prima di porla in uscita.

Come è mostrato in figura 2.2, esistono due possibilità per formare la *BBFRAME*: con l'eventuale presenza di Padding e senza mai aggiungere dei bit di Padding. Come già anticipato ad inizio capitolo, il progetto SCAT ha optato per una soluzione più consona alle caratteristiche delle applicazioni broadcast; ovvero senza mai applicare del Padding allo stream binario in ingresso dalla seriale. L'unico blocco del sistema che è stato implementato

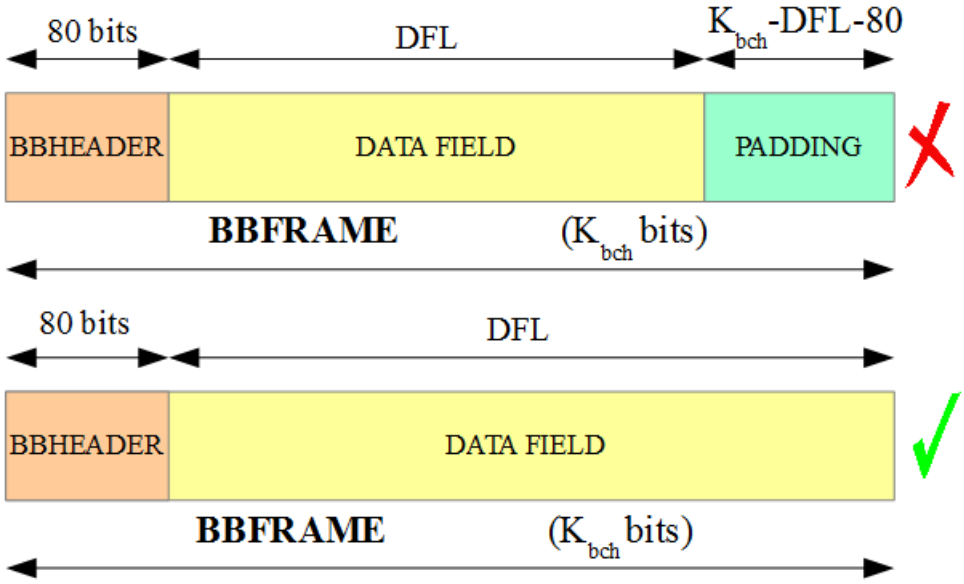

Figura 2.2: Formato della BBFRAME sia con l'aggiunta del Padding, sia nel caso opposto (sotto)

su FPGA per quanto riguarda la parte di *Stream Adaptation* è, perciò, il *BB Scrambler*.

2.2.1 BB Scrambler

Lo scrambling in banda base consiste nel randomizzare la trama in ingresso effettuando l'operazione di xor esclusivo con una sequenza binaria nota anche al ricevitore. Lo stream binario viene randomizzato essenzialmente per facilitare i dispositivi in ricezione. In questo modo, infatti, si è in grado di evitare all'apparato ricevente di dover attuare gli algoritmi di sincronizzazione di simbolo o di recupero della portante, in presenza di una lunga serie di bit costantemente a '0' o '1'. La trasmissione di lunghe sequenze binarie a valore costante complica infatti il corretto funzionamento dei circuiti adattativi in ricezione come i circuiti di Timing Recovery.

In figura 2.3 è mostrata una possibile architettura hardware del circuito in grado di generare la sequenza di scrambling. Il circuito che viene usato fa parte della famiglia dei PRBS (Pseudo Random Binary Sequence) encoder. Esso è formato da uno shift register in retroazione avente 15 celle elementari

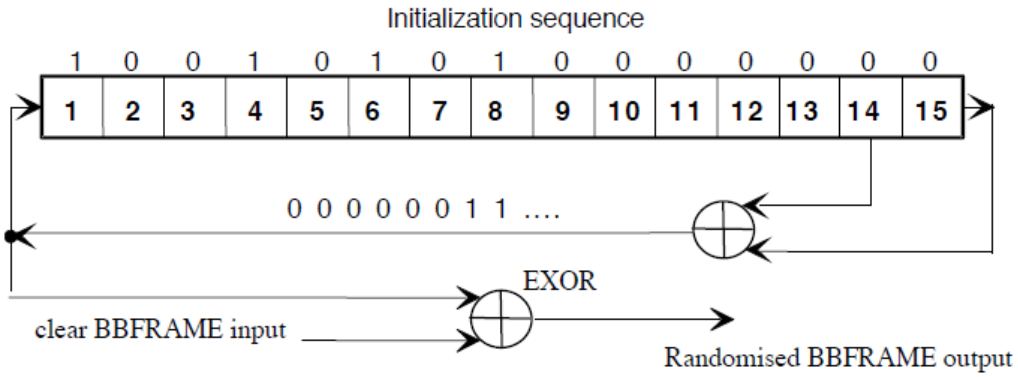

Figura 2.3: Architettura dell'encoder PRBS

e basato sul seguente polinomio generatore:

$$1 + X^{14} + X^{15}.$$

La sequenza deve essere sincrona con la trama in ingresso.Questo significa che il MSB (Most Significant Bit) di ogni trama deve essere scramblato mediante lo xor esclusivo con il primo bit della sequenza Pseudo Random. La sequenza PRBS va quindi reinizializzata ogni K_{bch} bit caricando nelle celle elementari la sequenza di 15 bit di inizializzazione:

$$100101010000000$$

L'impiego dello stesso circuito in ricezione permette di recuperare la sequenza binaria trasmessa, sfruttando la proprietà dello xor esclusivo:

$$[b_i \oplus b_o] \oplus b_o = b_i$$

2.3 FEC Encoding

Con la sezione denominata *FEC Encoding* si entra nel cuore del sistema di trasmissione DVB-S2. Essa consiste in tre blocchi: un encoder esterno BCH (Bose Ray-Chaudhuri), un encoder interno LDPC (Low-Density Parity-Check) e un interleaver classico. Il sottosistema che ha reso questo standard così performante in termini di capacità trasmissiva è l'encoder LDPC. L'annesso decoder LDPC in ricezione è in grado infatti di correggere anche da solo la quasi totalità degli errori. L'encoder BCH è stato tuttavia pensato per garantire un certo margine di sicurezza in situazioni particolarmente sfavorevoli di canale, mentre l'interleaver assicura una robusta protezione dagli

errori a burst. Come già anticipato ad inizio capitolo, la grande potenzialità del sistema DVB-S2 consiste nella possibilità di combinare un elevato numero di possibili code rate (R_c) del sistema di codifica, con quattro differenti modulazioni digitali in banda base. La variabilità del sistema in trasmissione si fonda però su un parametro costante qualsiasi sia la combinazione scelta, ovvero la lunghezza della codeword LDPC. Quest'ultima può assumere due possibili valori, 16200 nella versione short del sistema e 64800 nella versione classica. Il progetto SCAT ha optato per la versione classifica, da cui:

$$n_{ldpc} = 64800.$$

Questo parametro costante funge da base per definire tutte le altre grandezze del sistema *FEC Encoding*, in funzione del valore di code rate dell'encoder LDPC e della costellazione scelta per il mapping.

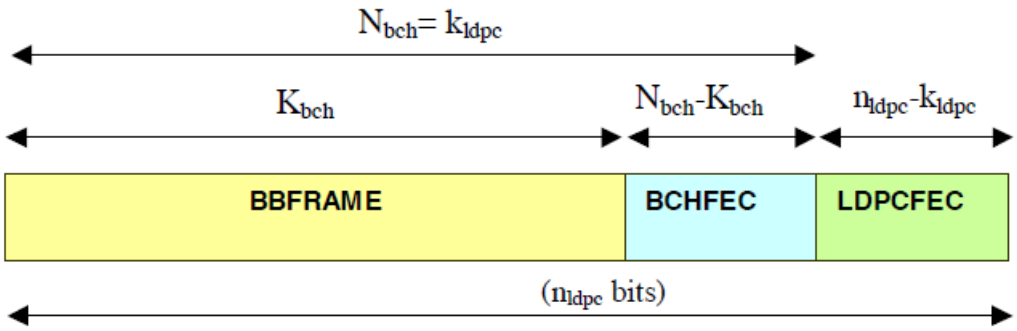

Figura 2.4: Formato dei dati prima dell'interleaving

In particolare, il valore di R_c del codice LDPC, va a definire ricorsivamente la lunghezza del messaggio in ingresso al LDPC (k_{ldpc}), che coincide con la codeword del BCH (N_{bch}) e la dimensione del messaggio in ingresso al codificatore esterno (K_{bch}), che corrisponde alla lunghezza della *BBFRAME*. In figura 2.4 è rappresentato il formato di una *FECFRAME* prima dell'interleaving.

Nella tabella 2.2 sono invece mostrati tutti i valori dei parametri sopra citati al variare della code rate LDPC. Come si può notare, la scelta della code rate dell'encoder LDPC va ad influenzare anche il valore di code rate del codificatore esterno BCH e quindi il numero di errori t che questo codice è in grado di correggere.

LDPC R_c	BCH Message K_{bch}	BCH Codeword N_{bch}	BCH t	LDPC Codeword n_{ldpc}
1/4	16008	16200	12	64800
1/3	21408	21600	12	64800
2/5	25728	25920	12	64800
1/2	32208	32400	12	64800
3/5	38688	38880	12	64800
2/3	43040	43200	10	64800
3/4	48408	48600	12	64800
4/5	51648	51860	12	64800
5/6	53840	54000	10	64800
8/9	57472	57600	8	64800
9/10	58192	58320	8	64800

Tabella 2.2: Parametri di codifica per una FECFRAME normale, 64800 bit

2.3.1 BCH Encoder

Il codificatore esterno che va a processare preliminarmente la *BBFRAME* aggiungendo i primi bit di parità è un encoder BCH. In particolare il valore numerico della parità introdotta è dato dalla differenza tra la dimensione della codeword BCH e la lunghezza del messaggio in ingresso:

$$\text{Bit di parità} = N_{bch} - K_{bch}.$$

L'encoder BCH fa parte della categoria dei codici sistematici, il che significa che ogni codeword è la concatenazione della *BBFRAME* in ingresso e dei bit di parità (o ridondanza) calcolati dal codice. Delle $2^{N_{bch}}$ possibili sequenze di bit in uscita dal sistema, solamente $2^{K_{bch}}$ rappresentano quindi parole di codice valide. I codici BCH fanno parte dei cosiddetti codici a correzione di errore, il che significa che la ridondanza introdotta è utile al ricevitore per rilevare o correggere errori che sono occorsi durante la trasmissione. Inoltre questi codici godono di altre proprietà particolari. Essi sono infatti codici lineari, a blocchi e ciclici. L'ultima caratteristica significa che se viene fatto uno shift circolare dei bit di ogni codeword per un numero arbitrario di posizioni, il risultato è sempre una delle codeword valide. Rispetto alle altre categorie di codici lineari a blocchi, i codici ciclici hanno inoltre il vantaggio di permettere una implementazione dell'encoder molto efficiente basata su un shift register in retroazione, come descritto in seguito.

Il calcolo dei bit di ridondanza per una data *BBFRAME* in ingresso è basato su un polinomio detto polinomio generatore del codice BCH, solitamente chiamato $g(x)$. Il polinomio generatore ha grado complessivo pari a

g_1(x)	$1+x^2+x^3+x^5+x^{16}$
g_2(x)	$1+x+x^4+x^5+x^6+x^8+x^{16}$
g_3(x)	$1+x^2+x^3+x^4+x^5+x^7+x^8+x^9+x^{10}+x^{11}+x^{16}$
g_4(x)	$1+x^2+x^4+x^6+x^9+x^{11}+x^{12}+x^{14}+x^{16}$
g_5(x)	$1+x+x^2+x^3+x^5+x^8+x^9+x^{10}+x^{11}+x^{12}+x^{16}$
g_6(x)	$1+x^2+x^4+x^5+x^7+x^8+x^9+x^{10}+x^{12}+x^{13}+x^{14}+x^{15}+x^{16}$
g_7(x)	$1+x^2+x^5+x^6+x^8+x^9+x^{10}+x^{11}+x^{13}+x^{15}+x^{16}$
g_8(x)	$1+x+x^2+x^5+x^6+x^8+x^9+x^{12}+x^{13}+x^{14}+x^{16}$
g_9(x)	$1+x^5+x^7+x^9+x^{10}+x^{11}+x^{16}$
g_{10}(x)	$1+x+x^2+x^5+x^7+x^8+x^{10}+x^{12}+x^{13}+x^{14}+x^{16}$
g_{11}(x)	$1+x^2+x^3+x^5+x^9+x^{11}+x^{12}+x^{13}+x^{16}$
g_{12}(x)	$1+x+x^5+x^6+x^7+x^9+x^{11}+x^{12}+x^{16}$

Tabella 2.3: Polinomi BCH necessari al calcolo del polinomio generatore nel caso di una FECFRAME normale, 64800 bit

$N_{bch} - K_{bch}$ ed è costruito su un campo finito di Galois, GF(2); questo significa che il coefficiente di ogni termine x^j è un valore binario pari a 0 o 1. Per il calcolo del polinomio generatore si prosegue come descritto di seguito. Se si osserva la tabella 2.2, ogni codice BCH è caratterizzato, oltre che dalla dimensione del messaggio in ingresso e della parola di codice in uscita, anche da un terzo parametro chiamato t. Questo parametro rappresenta il massimo numero di bit errati che il codice BCH è in grado di correggere a prescindere dalla posizione in cui l'errore si trova nella codeword. Il polinomio generatore di ogni singolo codice BCH, dipende dal corrispondente valore di t. Infatti per calcolare $g(x)$, viene fatta la moltiplicazione (in GF(2)) dei primi t polinomi riportati in tabella 2.3. Ad esempio, per i codici in grado di correggere 12 errori si prosegue nel seguente modo:

$$g(x) = g_1(x) \cdot g_2(x) \cdot ... \cdot g_{11}(x) \cdot g_{12}(x).$$

Il risultato è un polinomio generatore di grado 192, che può essere rappresentato mediante uno shift register retroazionato come quello di figura 2.5. Lo shift register è formato da 192 celle elementari di memoria. In base al polinomio generatore, determinati valori delle celle sono soggetti ad un'operazione di xor esclusivo con il valore in uscita dall'ultima cella andando ad aggiornare la cella successiva. Essendo un codice sistematico, per i primi K_{bch} colpi di clock, i bit in uscita dall'encoder sono i corrispondenti bit del messaggio in ingresso. Questi bit, nel frattempo, entrano nello shift register e ad ogni colpo di clock producono l'elaborazione dei bit di parità (i due switch di figura sono nella posizione (1)). Al $K_{bch} + 1$ colpo di clock, gli

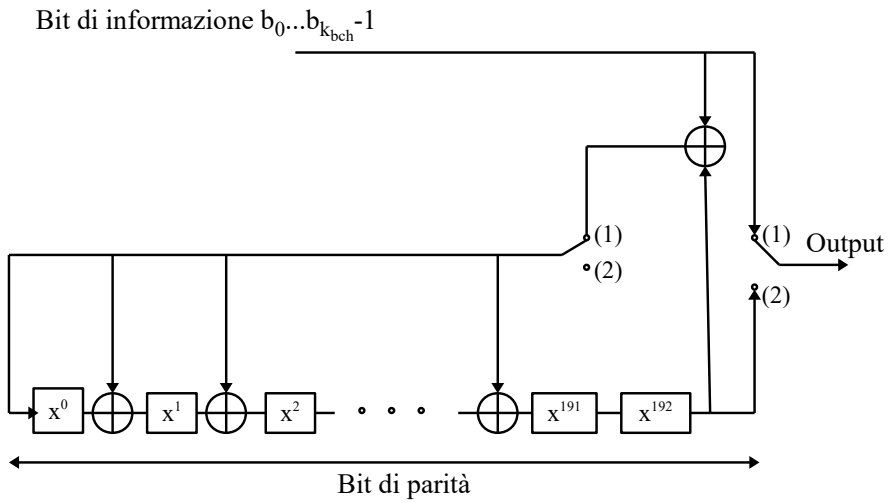

Figura 2.5: Implementazione hardware dell'encoder BCH

switch passano nella posizione (2). A questo punto lo shift register non viene più aggiornato con nuovi ingressi e in uscita all'encoder si hanno i bit di parità, ovvero i 192 valori che compongono il registro a scorrimento. Come già detto in precedenza, questo codice rappresenta un margine di sicurezza aggiuntivo per sopperire a situazioni di canale molto sfavorevoli e non rappresenta il sottosistema principale del *FEC Encoding*. A causa dell'elevato R_c di questo codice, esso è in grado di correggere infatti un valore massimo di solo 12 errori su codeword di dimensioni notevoli (fino a 58000 bit).

2.3.2 LDPC Encoder

Il codice che permette al sistema DVB-S2 di avere performance elevatissime è il codice a correzione d'errore introdotto da Robert G. Gallager nel 1962 e chiamato *Low-Density Parity-Check Code*. Il vantaggio principale di questi codici è che hanno prestazioni prossime alla capacità di canale definita dal teorema fondamentale dell'informazione di Shannon nel 1948. Nonostante sia stato introdotto negli anni '60, il suo utilizzo pratico è molto più recente e lo standard DVB-S2 è stato uno dei primi sistemi ad adottarlo. Il motivo che ha ritardato il successo dei codici LDPC è l'elevata complessità del sistema di ricezione. L'apparato che si occupa della decodifica prevede infatti unità di elaborazione molto complesse. Inoltre la fase di decodifica è iterativa e

non immediata, e il numero di iterazioni per sistemi complessi come il DVB-S2 è pari ad almeno 50. Le risorse hardware in grado di gestire operazioni così complesse sono comparse solamente di recente. Come i codici BCH, anche i codici LDPC sono sistematici e quindi i bit di ridondanza calcolati seguono in uscita i bit di informazione. L'elemento principale che descrive un codice LDPC è la cosiddetta matrice di parità H. Questa matrice, formata da un elevato numero di '0' e un bassissimo numero di '1' (da cui il nome Low-Density), prende il nome di matrice sparsa. Si sfrutta infatti l'elevato numero di '0' per risparmiare della memoria, andando a memorizzare solamente le posizioni (in termini di colonne e righe) degli elementi diversi da '0', ovvero i valori a '1'. Uno dei primi codici LDPC realizzati da Gallager, è un codice avente una proprietà particolare, ovvero, tutte le righe della matrice hanno lo stesso numero di '1'; lo stesso dicasi per le colonne. Una matrice di questo tipo si dice regolare e può essere descritta da quattro parametri: il numero di colonne, il numero di righe, il numero di '1' nelle righe e il numero di '1' nelle colonne. In particolare, una matrice di parità LDPC ha un numero di colonne pari alla dimensione della codeword e un numero di righe pari al numero di bit di parità.

Nello standard DVB-S2 è possibile utilizzare un elevato numero di codici LDPC diversi, a seconda del valore di code rate scelto, come mostrato in tabella 2.2. Tuttavia, i differenti codici che dipendono dalla scelta di R_c, godono delle stesse proprietà particolari [9]:

- Sono codici sistematici, quindi il la parola di codice BCH rappresenta il prefisso della codeword LDPC.

- Sono codici LDPC irregolari, ovvero il peso associato ad ogni colonna della matrice di parità non è costante.

- La matrice H ha una struttura ciclica nella parte di informazione (prime k_{ldpc} colonne).

- La matrice H una struttura a scala nella parte di parità (ultime $n_{ldpc} - k_{ldpc}$ colonne).

Le matrici di parità necessarie al calcolo dei bit di ridondanza introdotti dal codice godono quindi di un'altra peculiarità, oltre al fatto di essere sparse. Infatti, considerando gruppi di 360 colonne adiacenti, esse sono differenti solo per una traslazione costante di un determinato numero di righe. Queste due proprietà rendono quindi possibile implementare un algoritmo iterativo semplificato per il calcolo dei bit di parità, che permette di risparmiare le risorse utilizzate ed eliminare le operazioni il cui risultato sarebbe a priori

nullo. L'encoder riceve in ingresso una trama di informazione di lunghezza k_{ldpc},$I = (i_0, i_1, ..., i_{k_{ldpc}-1})$ e la codifica in una codeword **c**:

$$c = (i_0, i_1, ..., i_{k_{ldpc}-1}, p_0, p_1, ..., p_{n_{ldpc}-k_{ldpc}-1})$$

code rate	Q
1/4	135
1/3	120
2/5	108
1/2	90
3/5	72
2/3	60
3/4	45
4/5	36
5/6	30
8/9	20
9/10	18

Tabella 2.4: Valori di Q al variare della code rate nel caso di FECFRAME classica

L'algoritmo necessario al calcolo della ridondanza, ovvero dei bit da p_0 a $p_{n_{ldpc}-k_{ldpc}-1}$, si basa sui seguenti passaggi:

1. Inizializzazione a 0 dei bit di parità:

$$p_0 = p_1 = p_2 = ... = p_{n_{ldpc}-k_{ldpc}-1} = 0$$

2. Somma in GF(2) tra il primo bit d'informazione i_0 e i bit di parità uguali a 1 indicati dalla prima riga della tabella relativa alla codifica desiderata e definita in [8], ad esempio:

$$p_0 = p_0 \oplus i_0$$
$$p_{240} = p_{240} \oplus i_0$$

3. Per i successivi 359 bit (i_j), j=1,2,...,359, somma in GF(2) tra il bit i_j e i bit di parità di indice:

$$\{x + jmod360 \cdot Q\}mod(n_{ldpc} - k_{ldpc}),$$

dove con x si intendono gli indirizzi già utilizzati per il primo bit i_0 e con Q un valore costante che dipende dalla codifica scelta, secondo la tabella 2.4.

4. Per il successivo bit di informazione, i_{360}, si procede come per il bit i_0 ma utilizzando la seconda riga della tabella definita in [8]. Mentre per i successivi 359 bit si ripete il procedimento utilizzando la formula del punto precedente ma sostituendo i valori di j e i valori di x che ora corrispondono agli indirizzi utilizzati per i_{360}.

5. Questa operazione viene ripetuta per ogni gruppo di 360 bit, cambiando ad ogni gruppo la riga della tabella considerata.

6. Una volta che tutti i bit di informazione sono stati processati si deve procedere con la seguente operazione partendo da i=1:

$$p_i = p_i \oplus p_{i-1}$$
$$i = 1, 2, ..., n_{ldpc} - k_{ldpc} - 1$$

7. I valori finali di p_i con $i = 0, 1, ..., n_{ldpc} - k_{ldpc} - 1$ corrispondono ai bit di parità che andranno concatenati ai bit di informazione in uscita dall'encoder.

In uscita all'encoder LDPC si hanno già trame di bit di lunghezza pari alla *FECFRAME*, tuttavia, prima di passare alla modulazione digitale in banda base viene realizzata un'altra operazione che determina la *FECFRAME* definitiva, ovvero l'interleaving sulla trama lunga 64800 bit.

2.3.3 Bit Interleaver

L'ultimo sottosistema che fa parte della sezione *FEC Encoding*, e che non introduce ridondanza è chiamato *Bit Interleaver*. Questa operazione viene solitamente adottata nei sistemi di trasmissione per prevenire eventuali errori a 'burst'. Errori di questo tipo occorrono quando si ha un improvviso degrado delle condizioni di canale e gli errori sui bit in ricezione avvengono tutti ravvicinati all'interno di una singola parola di codice, rendendo impossibile per il sistema di decodifica correggere tutti gli errori nella singola codeword. L'interleaving consiste in una permutazione ordinata dei bit all'interno di una parola di codice prima che venga trasmessa sul canale, distribuendo lungo tutta la codeword gli eventuali errori a burst. In fase di ricezione, la codeword trasmessa viene ricostruita andando a svolgere l'operazione analoga. Per fare questo, l'operazione che si fa è definire una cosiddetta matrice di interleaving, il cui prodotto tra righe e colonne sia pari alla dimensione del blocco in ingresso alla matrice. La parola di codice in ingresso viene scritta serialmente all'interno dell'interleaver colonna per colonna, mentre viene letta in uscita

riga per riga. Questo determina la permutazione dei bit all'interno della codeword.

Nel caso specifico del sistema DVB-S2, l'operazione di interleaving a blocchi dipende dal tipo di modulazione digitale scelta tra la QPSK, la 8PSK, la 16APSK e la 32APSK.

Modulazione	Righe	Colonne
QPSK	64800	1
8PSK	21600	3
16APSK	16200	4
32APSK	12960	5

Tabella 2.5: Dimensioni della matrice di interleaver per una FECFRAME classica, al variare della modulazione

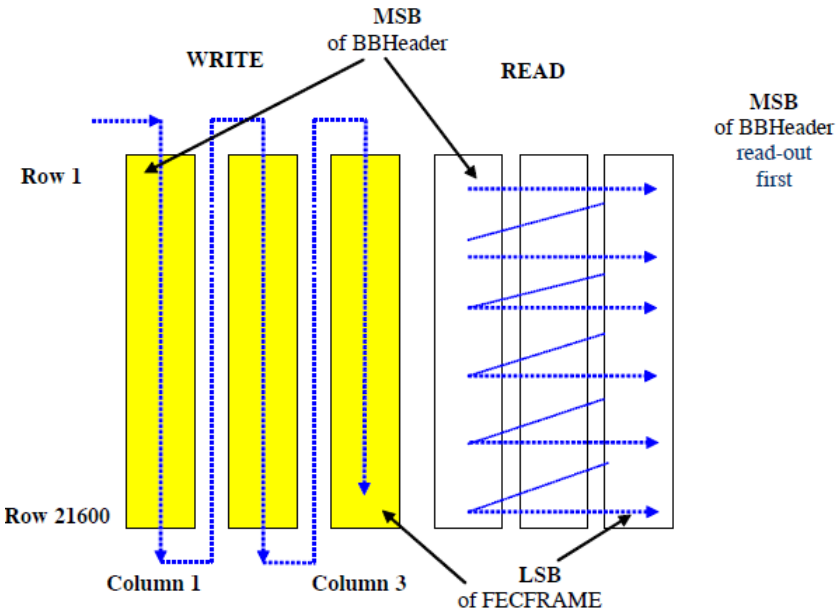

Figura 2.6: Schema di interleaving a blocchi nel caso di modulazione 8PSK per una FECFRAME classica

In particolare, la scelta della costellazione in cui mappare i bit, va a definire le dimensioni della matrice di interleaving, come mostrato in tabella 2.5. Nel caso di modulazione QPSK, di fatto, non avviene nessuna permutazione dei bit, in quanto la matrice si riduce ad essere un vettore colonna formato da 64800 righe. In tutti gli altri casi vige invece la regola descritta

in precedenza riguardo all'ordine di scrittura e di lettura dei bit all'interno della matrice. In figura 2.6 è mostrato l'esempio riguardante la struttura di interleaving nel caso 8PSK. L'unica eccezione riguarda il caso particolare 8PSK rate 3/5, nel quale la lettura avviene per righe ma non partendo dal MSB bensì dall'ultimo elemento di ogni riga.

2.4 Mapping

Lo step successivo nella catena di trasmissione consiste nel passare da una rappresentazione puramente binaria delle trame trasmesse, a una rappresentazione fisica con dei valori numerici complessi. Viene a tal proposito eseguita una modulazione numerica in banda base. Questa operazione è determinante per quantificare la larghezza di banda del segnale trasmesso, in particolare l'efficienza spettrale del sistema. L'operazione che viene eseguita è una conversione seriale-parallela della *FECFRAME* in ingresso. Di fatto, a seconda del parametro η che dipende dalla modulazione scelta, le *XFECFRAME* in uscita hanno lunghezza pari a $64800/\eta$. Il valore di η corrisponde al numero di bit consecutivi della *FECFRAME* che vanno a concatenarsi formando il valore complesso da mappare nel piano (I,Q). Nel sistema DVB-S2 le possibilità sono le seguenti:

- QPSK: η=2
- 8PSK: η=3
- 16APSK: η=4
- 32APSK: η=5

Se si suppone di avere lo stesso valore di bit-rate in ingresso a modulazioni digitali differenti, un valore più elevato di η corrisponde ad una minore symbol rate del segnale in uscita e quindi ad una larghezza di banda inferiore. Il valore η coincide, in particolare, con l'efficienza spettrale delle singole modulazioni, valutata in bits/s/Hz. Tuttavia, valori più elevati di η corrispondono a costellazioni più fitte che necessitano di rapporti segnale rumore in termini di E_s/N_0 più alti. La scelta tra una modulazione o l'altra dipende quindi dal compromesso tra l'efficienza in termini di banda occupata e l'efficienza in termini di potenza impiegata per la trasmissione.

Durante lo svolgimento della tesi, i sistemi di trasmissione che sono stati principalmente affrontati, hanno previsto l'impiego della modulazione QPSK e della 16ASPK. Per questo motivo, di seguito sono descritti solamente questi

due casi. La modulazione 8PSK risulta concettualmente simile all QPSK, essendo anch'essa ad inviluppo costante, ma con un numero maggiore di punti sulla costellazione. Lo stesso si può dire per la 32APSK nei confronti della 16APSK.

2.4.1 Costellazione QPSK

La costellazione QPSK prevede la concatenazione di 2 bit per formare il simbolo corrispondente sul piano complesso. Il sistema impiega una codifica di Gray, nella quale simboli vicini nella costellazione differiscono per il valore di un solo bit. Inoltre, la costellazione deve avere energia media per simbolo uguale ad 1; per questo motivo, essendo i simboli complessi rappresentativi posti su una circonferenza, la circonferenza ha raggio uguale a 1.

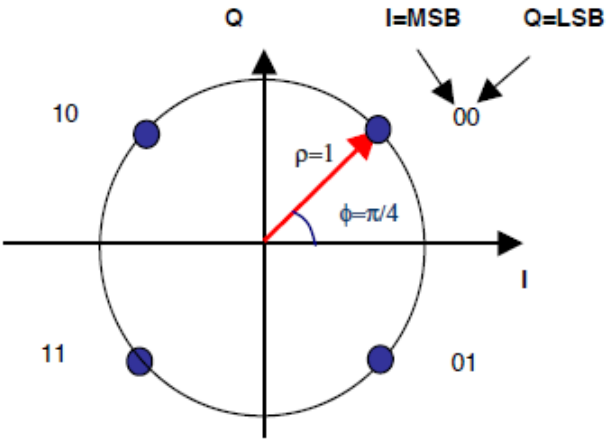

Figura 2.7: Costellazione QPSK secondo una codifica di Gray

I simboli sono sfasati di $\pi/4$ l'uno dall'altro sulla circonferenza, come si può vedere in figura 2.7. Il MSB rappresenta il primo bit in ingresso al mapper, mentre il LSB rappresenta il bit successivo.

2.4.2 Costellazione 16APSK

Questo tipo di modulazione, insieme alla 32APSK, rappresenta una modulazione digitale ibrida. Infatti, si tratta allo stesso tempo di una modulazione di fase e di ampiezza (Amplitude and phase-shift keying). Per formare un simbolo della costellazione, necessita della concatenazione di 4 bit consecutivi, di cui il MSB è il primo in ingresso al modulatore. La costellazione è

formata da 2 anelli concentrici, quello interno di raggio R_1 e sul quale sono posti 4 simboli uniformemente sfasati e quello esterno di raggio R_2 e sul quale giacciono 12 simboli della costellazione sempre uniformemente sfasati. Il rapporto tra il raggio esterno e quello interno vale γ.

Code rate	γ
2/3	3.15
3/4	2.85
4/5	2.75
5/6	2.70
8/9	2.60
9/10	2.57

Tabella 2.6: Rapporto tra raggio esterno e interno nella costellazione 16APSK, al variare della code rate LDPC

Per determinare il valore di R_1 e R_2 bisogna combinare l'equazione seguente con i valori di tabella 2.6.

$$R_1{}^2 + 3R_2{}^2 = 4$$

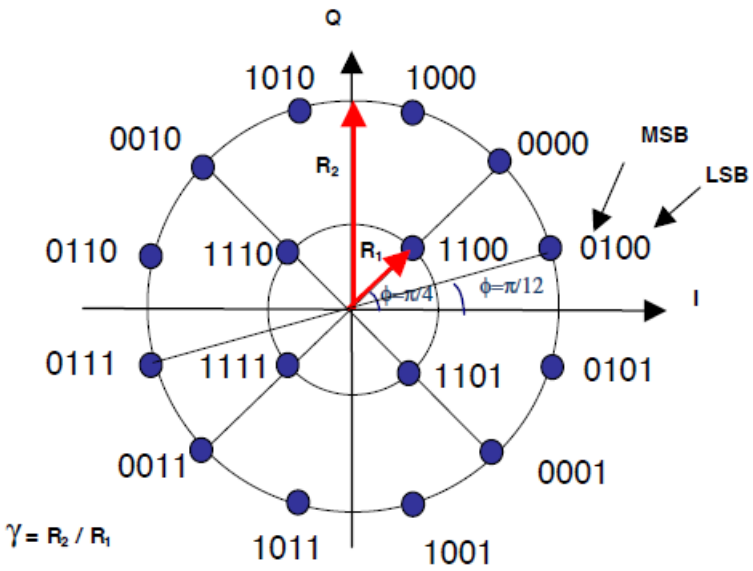

Figura 2.8: Costellazione 16APSK

In tabella 2.6 sono riportati i valori di γ in funzione della code rate del codice LDPC scelto, considerando che la modulazione 16APSK può essere

implementata solo insieme ai valori di code rate di tabella. La costellazione è mostrata in figura 2.8.

2.5 PL Framing

Una volta convertite le *FECFRAME* binarie in *XFECFRAME* formate da valori complessi, il sistema prevede alcune operazioni sullo stream di dati fisico. Queste operazioni risultano fondamentali in ricezione per facilitare il recupero della portante, la sincronizzazione di simbolo e per permettere la sincronizzazione di frame. L'insieme di questi sottosistemi prende il nome di *PL Framing*, dove con PL si intende Physical Layer, ovvero strato fisico. Questa sezione dello standard si occupa di formare le definitive *PLFRAME* che verranno poi processate dal filtro interpolatore in banda base. I blocchi che formano il *PL Framing* sono i seguenti quattro:

1. *Inserzione delle Dummy PLFRAME.* Inserimento di trame fittizie quando nessuna *XFECFRAME* è pronta ad essere processata e trasmessa.

2. *Inserzione del PLHeader.* Concatenazione di un header di strato fisico all'inizio di ogni *XFECFRAME.*

3. *Inserzione dei toni pilota.* Inserimento facoltativo di toni pilota all'interno di ogni trama complessa.

4. *PL Scrambling.* Operazione di randomizzazione della *XFECFRAME.*

L'inserimento delle Dummy PLFRAME, tuttava, risulta facoltativo nelle applicazioni broadcast che fanno uso del DVB-S2. Per questo motivo, come già anticipato ad inizio capitolo, il progetto SCAT non prevede questa funzionalità. La sezione di *PL Framing* consiste quindi solamente nelle altre tre elaborazioni della trama fisica.

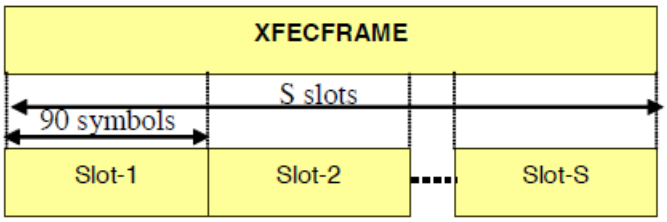

Figura 2.9: Suddivisione della *XFECFRAME*

Prima di passare alla descrizione di ogni singolo blocco, lo standard fa notare come, qualsiasi sia la modulazione scelta, la dimensione della *XFECFRAME* è tale da poter essere frazionata in un numero S (variabile a seconda della costellazione) di slot fisiche aventi lunghezza costante e pari a 90 simboli complessi, come mostrato in figura 2.9.

2.5.1 Inserzione PLheader

Questo blocco, obbligatorio in qualsiasi sistema DVB-S2, consiste nella generazione e nell'inserzione di un header di strato fisico (formato da simboli complessi) all'inizio di ogni *XFECFRAME*. Il preambolo, di dimensione costante e uguale a quella delle altre slot (90 simboli), viene inserito per due motivi: permettere la sincronizzazione di frame in ricezione e rendere noto al sistema ricevente la combinazione di modulazione e codifica utilizzata, oltre alla presenza o meno dei toni pilota. Il *PLHeader* é composto dai seguenti due campi: SOF (Start of Frame) formato da 26 simboli costanti qualsiasi sia la combinazione di modulazione e codifica e PLS (Physical Layer Signalling) code composto da 64 simboli che variano a seconda della configurazione scelta. Questi due campi sono generati prima come sequenza binaria e poi mappati su una costellazione $\pi/2$-BPSK convertendo i bit in simboli complessi. Il campo SOF corrisponde alla sequenza esadecimale 18D2E82 in notazione binaria (01-1000-...-0010) e il suo bit più significativo rappresenta anche il MSB del *PLHeader*.

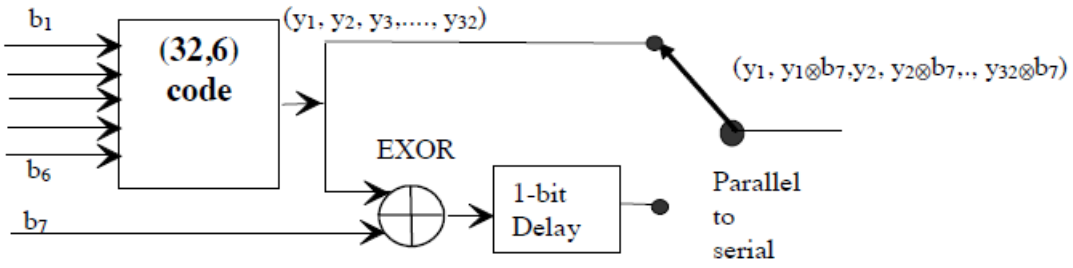

Figura 2.10: Schema relativo alla generazione del PLS code

La generazione del campo PLS code è invece più elaborata, in quanto questo campo dell'header è variabile e la sua rilevazione in ricezione permette al sistema di determinare quale configurazione è stata implementata in trasmissione. I 64 bit che formano la sequenza rappresentano la codeword di un codice binario non sistematico equivalente ad un codice di Reed-Muller. I passaggi che portano all'elaborazione dei 64 bit sono descritti di seguito.

I bit di informazione che rappresentano il messaggio in ingresso al codice sistematico sono 7 e sono a loro volta suddivisi nei seguenti due campi:

- MODCOD: 5 bit che identificano la modulazione in banda base e la code rate del codice LDPC.
- TYPE: 2 bit che informano se la *FECFRAME* è di 64800 bit o 16200 bit(short) e se sono stati usati toni pilota o no.

Mode	MODCOD	Mode	MODCOD
QPSK 1/4	00001	8PSK 9/10	10001
QPSK 1/3	00010	16APSK 2/3	10010
QPSK 2/5	00011	16APSK 3/4	10011
QPSK 1/2	00100	16APSK 4/5	10100
QPSK 3/5	00101	16APSK 5/6	10101
QPSK 2/3	00110	16APSK 8/9	10110
QPSK 3/4	00111	16APSK 9/10	10111
QPSK 4/5	01000	32APSK 3/4	11000
QPSK 5/6	01001	32APSK 4/5	11001
QPSK 8/9	01010	32APSK 5/6	11010
QPSK 9/10	01011	32APSK 8/9	11011
8PSK 3/5	01100	32APSK 9/10	11100
8PSK 2/3	01101	Reserved	11101
8PSK 3/4	01110	Reserved	11110
8PSK 5/6	01111	Reserved	11111
8PSK 8/9	10000	Dummy PLFRAME	00000

Tabella 2.7: Valori possibili di MODCOD

Tutte le possibili combinazioni di modulazione e codifica sono riassunte in tabella 2.7, in cui sono rappresentati i valodi di MODCOD associati alla configurazione scelta. Come si può vedere, non tutti i valori di code rate LDPC sono associabili alle varie modulazioni, rendendo solamente pari a 28 il numero di combinazioni valide. Inoltre, tre valori di MODCOD sono relativi a configurazioni riservate e uno alle Dummy PLFRAME. Il campo TYPE formato da due soli bit viene usato nel seguente modo. Il bit più significativo indica se la *FECFRAME* ha lunghezza normale di 64800(bit=0) o accorciata (bit=1). Il bit meno significativo invece indica la presenza (bit=1) o l'assenza (bit=0) dei toni pilota. Definendo come b_1 il MSB del campo MODCOD e b_6 il MSB del campo TYPE, il circuito rappresentato in figura 2.10 mostra come, a partire dai 7 bit di informazione si producano i 64 bit del PLS code

tenendo conto che i bit in uscita dal sistema raffigurato in figura andranno poi randomizzati eseguendo l'operazione di xor esclusivo con una sequenza fissa definita in [8]. Sempre in [8] è fornita la matrice generatrice del codice lineare (32,6) di figura 2.10.

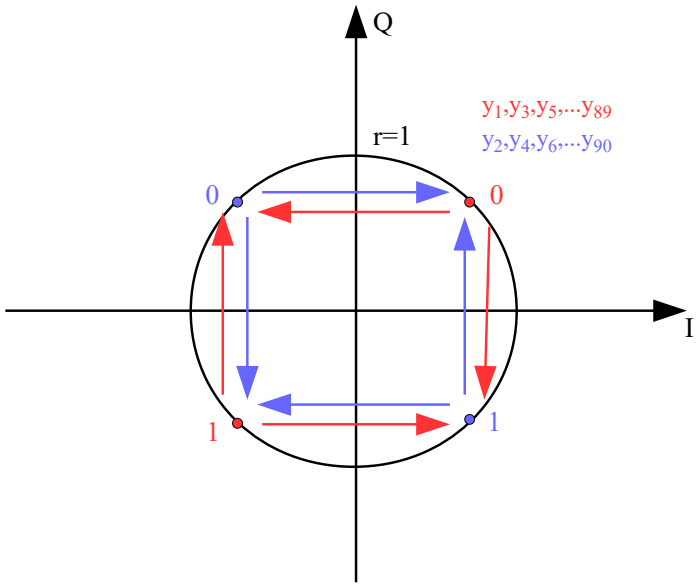

Figura 2.11: Costellazione $\pi/2$-BPSK

Come anticipato in precedenza, ogni bit dei campi SOF e PLS code, va mappato secondo una modulazione particolare chiamata $\pi/2$-BPSK. La costellazione $\pi/2$-BPSK è raffigurata in figura 2.11. Nonostante sia una costellazione di tipo BPSK (Binary Phase-Shift Keying), ovvero ad ogni bit sia associato un simbolo complesso, i possibili punti nella costellazione sono quattro. Questo è dovuto al fatto che per ogni bit successivo si ha una rotazione di 90 gradi della costellazione. Per i bit dispari dell'header (partendo dal MSB del SOF) i possibili valori sono quelli del primo e terzo quadrante, mentre per i bit pari si ha una rotazione di 90 gradi della costellazione e i simboli corrispondenti sono nel secondo e quarto quadrante. Il motivo per cui viene impiegata una costellazione di questo tipo, è evitare la transizione per l'origine della costellazione nel passaggio da un bit all'altro. Questo provoca infatti effetti di non linearità e la presenza di frequenze spurie che allargano lo spettro del segnale.

2.5.2 Inserzione toni pilota

Una volta aver concatenato alla *XFECFRAME* il preambolo di strato fisico, il passaggio successivo consiste nell'inserimento dei cosiddetti toni pilota. Questa operazione è opzionale e non rappresenta una normativa obbligatoria dello standard, tuttavia, la sua implementazione facilita notevolmente la fase di sincronizzazione di simbolo in ricezione. Il progetto SCAT in accordo con ESA ha, perciò, deciso di tenerne conto. Il tono pilota consiste in un blocco di 36 simboli pilota. Ogni simbolo pilota corrisponde al seguente numero complesso:

$$p = \frac{\sqrt{2}}{2} + j\frac{\sqrt{2}}{2}$$

L'inserimento dei toni pilota avviene nel seguente modo. Il primo tono pilota viene inserito dopo 16 slot di 90 simboli della *XFECFRAME*, il secondo dopo i successivi 16 slot e cosi via. Se la posizione del tono coincide con l'inizio della *XFECFRAME* successiva, il tono pilota non viene inserito. L'inserimento dei toni pilota, per quanto fondamentali in ricezione, ha come effetto negativo (seppur quasi trascurabile) una diminuzione dell'efficienza spettrale del sistema, in quanto si ha un aumento della symbol rate a parità di bit di informazione trasmessi a valle della fase di Mapping. La ridondanza, in termini di simboli, introdotta dai toni pilota è calcolabile con la seguente formula:

$$\text{simboli pilota} = 36 \cdot int\{\frac{(S-1)}{16}\}$$

dove S rappresenta il numero di slot di 90 simboli per *XFECFRAME* e *int* rappresenta la funzione integer.

2.5.3 PL scrambling

L'ultimo blocco che lo standard prevede nella fase di *PL Framing* e che genera la definitiva *PLFRAME* è lo scrambling a livello fisico. Questa operazione viene effettuata allo scopo di distribuire l'energia del segnale su tutto lo spettro disponibile, evitando che, in presenza di segnali pressoché costanti si abbiano problemi di interferenza con canali adiacenti a causa del fenomeno dell'intermodulazione. La randomizzazione della frame di strato fisico avviene moltiplicando ogni simbolo della stessa, con il corrispondente simbolo complesso di una sequenza pseudo-casuale.

Questa sequenza corrisponde ad una sequenza di Gold e può essere generata a livello hardware implementando due registri a scorrimento e combinando adeguatamente le uscite degli stessi per formare il numero complesso

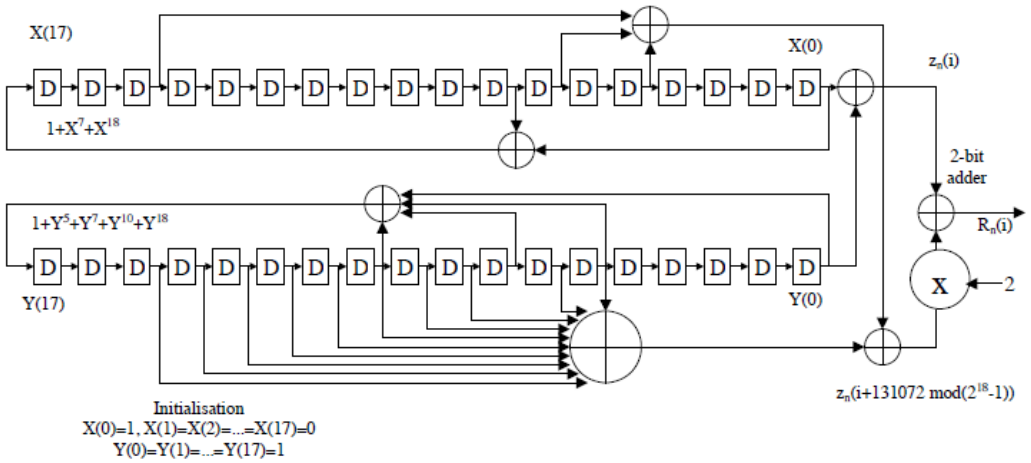

Figura 2.12: Circuito generatore della sequenza di scrambling per applicazioni broadcast

adeguato. Un esempio di implementazione hardware è quello di figura 2.12. Tuttavia lo standard DVB-S2 indica la possibilità di realizzare ben 262141 sequenze complesse differenti. Quello di figura 2.12 rappresenta il circuito utilizzato nel caso di applicazioni broadcast e che ho implementato nella mia tesi [8]. L'uscita del circuito rappresentato in figura è un numero intero compreso tra 0 e 3. Ad ognuno di questi quattro valori è associato un numero complesso nell'uscita finale dello scrambler. I possibili numeri complessi che vengono generati sono +1,-1,+j e -j. Lo scrambling consiste nella moltiplicazione dei simboli della trame per il corrispondente numero complesso tra i quattro possibili. Questi valori fanno si che dopo la moltplicazione non cambi la mappatura dei simboli sulla costellazione. La randomizzazione riguarda tutta la *PLFRAME* escludendo il *PLHeader*. Il circuito generatore va quindi reinizializzato alla fine di ogni header.

In figura 2.13, è rappresentata l'operazione di scrambling a livello fisico sulla trama. L'uscita di questo circuito rappresenta la *PLFRAME* che andrà poi filtrata nella parte successiva del trasmettitore.

2.6 Modulation

L'ultima sezione della catena di trasmissione è denominata *Modulation* e comprende il filtraggio in banda base del segnale mediante filtro sagomatore a radice di coseno rialzato e la modulazione in quadratura. In realtà lo standard [8] fornisce vincoli solamente per quanto riguarda il filtro sagomatore

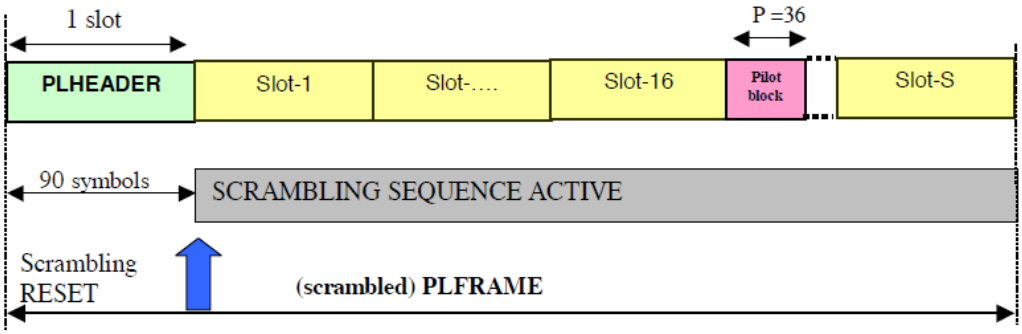

Figura 2.13: PLFRAME dopo l'operazione di PL scrambling

d'impulsi, mentre la fase di modulazione in quadratura non è descritta. Il sistema che ho implementato prevede, in particolare, la traslazione del segnale dalla banda base ad una frequenza intermedia. Per fare questo è necessario prevedere, prima della moltiplicazione per la portante a IF, una interpolazione del segnale in modo tale da soddisfare il Teorema del Campionamento di Nyquist-Shannon. Il sistema che si occupa dell'interpolazione del segnale e della modulazione in quadratura viene detto Digital Up Converter (DUC). Nel seguito ne verrà data una breve descrizione, anche se se il compito del DUC non è esplicitamente compreso nello standard DVB-S2.

2.6.1 Filtraggio in banda base

Questo segmento dello standard ha il compito di 'sagomare' il segnale in banda base, ovvero di andare a modificare la forma d'onda del segnale trasmesso. L'obiettivo è modellare e adattare il segnale trasmesso (filtrando il segnale in ingresso) in funzione dei requisiti imposti dallo standard e dalle caratteristiche del canale di comunicazione. Così facendo, si fa in modo che l'effettiva larghezza di banda del segnale non ecceda lo spettro disponibile. Questo processo è fondamentale per mantenere i fenomeni di distorsione sotto controllo, minimizzando gli effetti dell'interferenza inter-simbolo (ISI). Più specificatamente avviene un sovracampionamento del segnale e il filtraggio dello stesso usando un filtro FIR (Filtro a risposta finita) con risposta impulsiva a radice di coseno rialzato (Square Root Raised Cosine-SRRC). La risposta impulsiva di un filtro SRRC soddisfa la condizione di Nyquist (essa si annulla ad ogni multiplo del tempo di simbolo, tranne per t=0 in cui ha il picco), che rappresenta la condizione sufficiente a contrastare l'ISI.

Per valutare l'effettiva compatibilità del sistema implementato con i vincoli imposti dallo standard, bisogna valutare lo spettro del segnale in uscita,

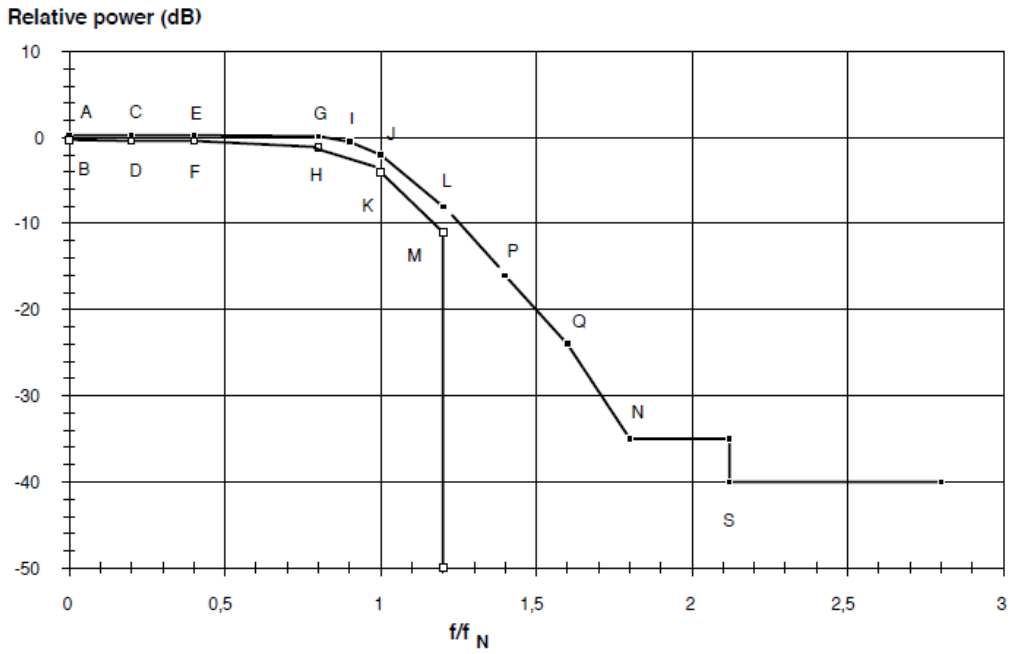

Figura 2.14: Maschera dello spettro in banda base fornita dallo standard ETSI

il quale deve rispettare la maschera riportata in [8] e rappresentata in figura 2.14. La forma dello spettro in uscita dipende dalla funzione di trasferimento del filtro sagomatore, qui riportata:

$$\begin{cases} H(f) = 1 & \text{per } |f| < f_N(1-\alpha) \\ H(f) = \{\frac{1}{2} + \frac{1}{2} \cdot \sin\left(\frac{\pi}{2f_N} \cdot \frac{f_N - |f|}{\alpha}\right)\}^{1/2} & \text{per } |f| = f_N(1-\alpha) \\ H(f) = 0 & \text{per } |f| > f_N(1-\alpha) \end{cases}$$

dove α è detto fattore di roll-off e f_N rappresenta la metà di B_s, ovvero la symbol rate in ingresso al filtro. La larghezza di banda del segnale equivale a $B_s \cdot (1+\alpha)$. La larghezza di banda dipende perciò dal parametro α, in quanto più il fattore di roll off è elevato, più la banda è larga. Con un fattore di roll-off pari a 0 lo spettro sarebbe quello di impulso rettangolare a banda minima, tuttavia questa possibilità è negata dalla fisica impossibilità nel realizzare un filtro del genere. I valori ammessi dallo standard DVB-S2 per α sono 0.20, 0.25 e 0.35.

Per quanto riguarda il processo di interpolazione (sovracampionamento), questo avviene mediante la moltiplicazione di ogni simbolo complesso in ingresso con i campioni della risposta impulsiva del filtro SRRC all'interno di

un periodo di simbolo $T_s = 1/B_s$. Il numero di campioni per periodo di simbolo con cui la riposta è campionata determina il fattore di interpolazione del filtro. Più questo valore è elevato, più si ha una rappresentazione adeguata del segnale. Allo stesso tempo, però, si ha un aumento di complessità del filtro (numero di stadi del filtro) con conseguente aumento della latenza introdotta.

2.6.2 Digital Up Conversion

L'ultimo elemento della catena di trasmissione non è esplicitato direttamente dallo standard, il quale indica come, a seguito del filtraggio a coseno rialzato, debba essere eseguita una modulazione in quadratura (da cui il nome *Modulation* nello standard). Prima di eseguire la moltiplicazione del segnale per una determinata frequenza portante, il segnale può necessitare di un'ulteriore interpolazione per non violare il Teorema del Campionamento di Nyquist-Shannon. Il sistema che si occupa di fare questo prende il nome di Digital Up Converter (DUC). Esistono diversi schemi per implementare un DUC. Il progetto SCAT ha optato per una struttura che comprende i seguenti stadi:

- Primo filtro interpolatore. Si tratta di un filtro FIR polifase avente fattore di interpolazione pari ad M_1.

- Secondo filtro interpolatore. Un altro filtro FIR polifase con fattore di interpolazione M_2.

- Terzo filtro interpolatore. Viene utilizzato un filtro CIC (Cascade Integrator-Comb). Questo filtro permette di effettuare l'operazione di interpolazione limitando il quantitativo di risorse hardware da impiegare. Introduce un guadagno maggiore di 1.

- Scaling. Serve a normalizzare il guadagno complessivo del DUC, annullando il guadagno introdotto dal CIC.

- Moltiplicazione per la portante f_0 che rappresenta la frequenza intermedia ed è generata da un oscillatore libero.

Un esempio di struttura, suddivisa per stadi, del Digital Up Converter è mostrata in figura 2.15.

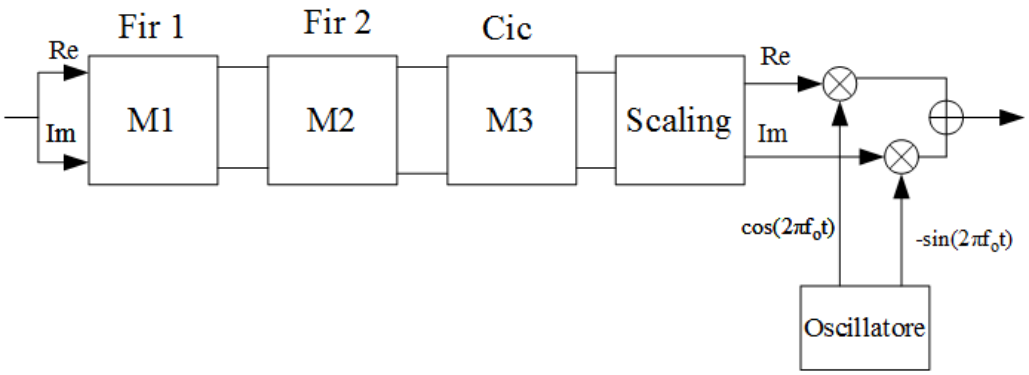

Figura 2.15: Struttura a stadi del Digital Up Converter

Capitolo 3

Modellazione Simulink

L'obiettivo della fase iniziale del progetto SCAT, ovvero quella che mi ha coinvolto in prima persona, è quello di sviluppare gli algoritmi di comunicazione satellitare per le fasi di downlink e uplink. In un secondo momento, questi algoritmi andranno implementati sulle risorse hardware a disposizione, in particolare l'FPGA di cui si è parlato nella prima parte dell'elaborato. Uno degli ambienti software più adeguati per realizzare sistemi di comunicazione completi è Simulink. Simulink, sviluppato dalla compagnia americana Mathworks, è un ambiente di programmazione grafico per la modellazione, la simulazione e l'analisi di sistemi dinamici multi-dominio. L'impiego di questo software per la fase di sviluppo degli algoritmi è dovuto a diverse ragioni. In primo luogo Simulink è strettamente integrato con l'ambiente MATLAB, permettendo l'accesso immediato ad un elevato numero di tools e fornendo la possibilità di potenziare il proprio modello Simulink con funzioni MATLAB. Nonostante sia un ambiente di programmazione visuale, è possibile quindi integrarlo con un linguaggio di programmazione vero e proprio, quello creato dalla Mathworks. In secondo luogo, Simulink fornisce una collezione personalizzata di librerie di blocchi che spesso sono sufficienti a realizzare il proprio modello senza il bisogno di intervenire progettando manualmente il proprio blocco (ad esempio con una funzione MATLAB ma anche con del codice C). Tra il vasto set di librerie fornite, oltre a quelle riguardanti blocchi dinamici continui e discreti, blocchi algoritmici e blocchi strutturali, è possibile fare uso anche di librerie più sofisticate. In particolare, installando i seguenti toolbox forniti da Mathworks, si può fare uso di un elevato numero di blocchi fondamentali per la progettazione di un sistema di comunicazione:

- *Communications System Toolbox*. I blocchi che fanno parte di questa libreria rappresentano i principali elementi di un sistema di comuni-

cazione, come ad esempio i blocchi relativi alla codifica di canale o la modulazione in banda base.

- *Data Acquisition Toolbox*. Insieme di blocchi che permettono l'acquisizione di segnali esterni e la conversione del segnale (A/D e D/A).

- *DSP System Toolbox*. Comprende tutte le funzionalità principali impiegate nel processamento digitale dei segnali, come ad esempio le trasformazioni FFT e IFFT o il filtraggio digitale (filtri FIR e CIC).

L'ulteriore caratteristica di Simulink (in generale dei software Mathworks) che rende questo ambiente di programmazione ideale per i fini del progetto SCAT, è la possibilità di convertire il proprio modello Simulink nel corrispondente codice VHDL, e quindi poter programmare l'FPGA con tale codice. Lo strumento Mathworks che si occupa di questa conversione è chiamato *HDL Coder* e verrà descritto nel proseguo del capitolo. La sottolineatura importante da fare riguarda il fatto che, durante lo svolgimento della tesi, mi sono occupato solamente della realizzazione di un modello Simulink che rappresentasse il sistema di comunicazione scelto e che avesse le caratteristiche necessarie per essere convertito in un linguaggio di descrizione dell'hardware. L'interazione tra il mondo hardware (FPGA) e l'ambiente software (Simulink), allo scopo di ottimizzare l'implementazione degli algoritmi sul dispositivo, è un task successivo del progetto.

3.1 Caratteristiche principali di Simulink

Per poter utilizzare un ambiente di programmazione come Simulink, il primo passo è quello di prendere conoscenza con le librerie che esso mette a disposizione. L'installazione di tutti i toolbox (tra cui quelli sopra citati) che Mathworks fornisce, in aggiunta alle librerie base già presenti in Simulink, permette di modellare e simulare sistemi di svariata natura, dall'elaborazione dei segnali, fino alle applicazioni aerospaziali. Il software permette, inoltre, di ampliare le proprie librerie con blocchi auto prodotti.

In figura 3.1 sono mostrate tutte le librerie presenti nel *Communications System Toolbox*, tra cui è possibile notare librerie relative alla codifica di sorgente, alla codifica di canale, alla modulazione e a tutte le principali operazioni di un sistema di telecomunicazione.

Integrazione con MATLAB

Una delle modalità più efficaci per ampliare le proprie librerie con blocchi che non siano nativi di Simulink, consiste nell'introdurre nel proprio modello

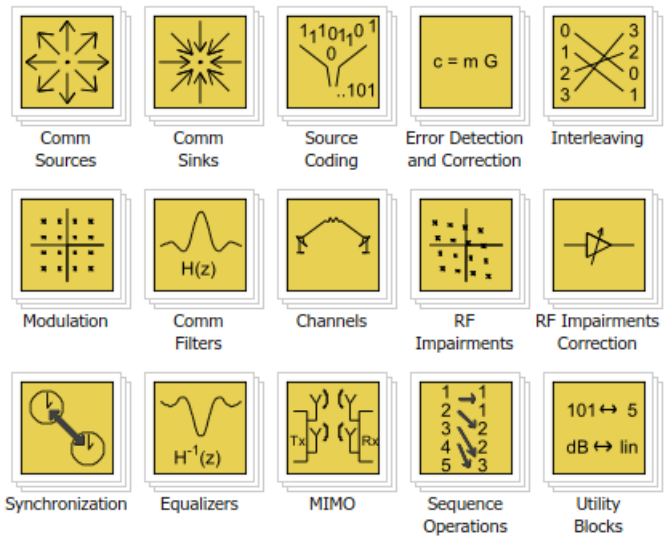

Figura 3.1: Librerie presenti all'interno del Communications System Toolbox

delle vere e proprie funzioni MATLAB. La MATLAB function è inserita nel modello come se fosse un normale blocco (ed effettivamente è un blocco presente in libreria) le cui uscite e ingressi sono propagati nel sistema. Si riesce perciò ad integrare l'ambiente grafico con il classico ambiente di programmazione MATLAB. Inoltre, è possibile inserire anche del codice C, Fortran ed Ada direttamente nel modello, abilitando l'utente a creare blocchi personalizzati. L'interazione con l'ambiente MATLAB non si limita all'utilizzo delle MATLAB functions. Un modello Simulink, infatti, è strettamente connesso con MATLAB in quanto è possibile passare determinati parametri o segnali al modello attraverso il workspace di MATLAB, e allo stesso tempo visualizzare i segnali in uscita dal sistema sempre sul workspace. Tra i tanti tools per l'analisi dei risulati, tra cui display e scope, si trova anche un blocco chiamato 'to workspace' che permette di esportare i valori del segnale Simulink nel workspace di MATLAB in forma vettoriale, matriciale o di timeseries. Blocchi analoghi permettono di generare un segnale sul modello importando grandezze vettoriali dal workspace, ad esempio il blocco 'signal from workspace'.

Struttura gerarchica del modello

Un'altra funzionalità particolarmente utile dell'ambiente Simulink è la possibilità di gestire progetti particolarmente complessi distribuendo i modelli in maniera gerarchica. Questo avviene utilizzando sottosistemi e referenziando i modelli. I sottosistemi racchiudono un insieme di blocchi e di segnali in un

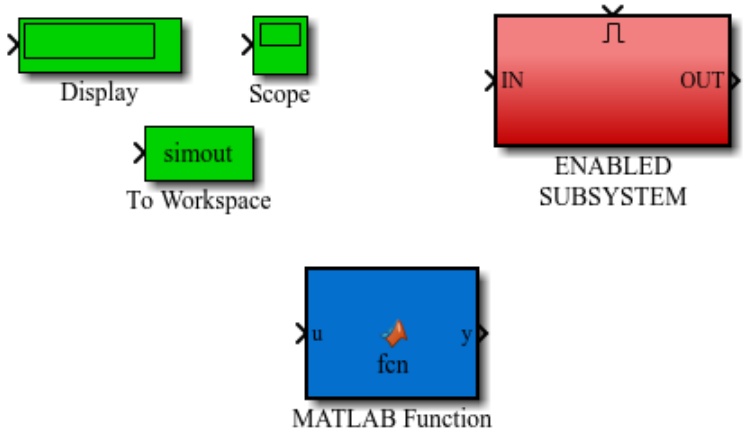

Figura 3.2: Esempi di tools per l'analisi dei risultati, di una MATLAB Function e di un Enabled Subsystem all'interno dell'ambiente di programmazione Simulink

singolo blocco. Inoltre, l'utente può associare al sottosistema un'interfaccia personalizzata in modo da poter nascondere il contenuto del sottosistema e facendolo apparire come un blocco atomico con una propria icona. Tra le varie tipologie di sottosistemi forniti da Simulink, una molto interessante è quella dei sottosistemi ad esecuzione condizionata dall'ingresso (Enabled Subsystem). Mediante il controllo di segnali logici, è possibile abilitare o disabilitare sezioni specifiche del progetto in maniera dinamica. L'impiego di blocchi logici all'interno di un modello risulta perciò spesso fondamentale. Una possibilità più elegante (ma anche più compelssa) per implementare controlli logici in un sistema consiste nell'utilizzo di macchine a stati modellabili grazie allo Stateflow Toolbox. In figura 3.2 è mostrata l'interfaccia grafica di un sottosistema condizionato, oltre alla rappresentazione base di una MATLAB Function e di alcuni esempi di strumenti per la visualizzazione dei segnali.

Model Explorer

Spesso, in un modello Simulink, è conveniente definire delle variabili numeriche predefinite per andare a progettare più blocchi del sistema che necessitano dello stesso parametro. Inoltre, un approccio parametrizzato è utile per caratterizzare i segnali del modello. I segnali sono quantità tempo-varianti rappresentate dalle linee di collegamento tra i blocchi. I parametri sono invece coefficienti che aiutano a definire la dinamica e il funzionamento del modello. Il software mette a disposizione uno strumento chiamato Model

Explorer, mediante il quale è possibile definire un database di variabili utili a caratterizzare il proprio modello. Il Model Explorer permette di avere una chiara panoramica del proprio modello, soprattutto se quest'ultimo dovesse avere una struttura gerarchica particolarmente complessa. Attraverso il Model Explorer si può, in aggiunta, sfruttare l'integrazione tra MATLAB e Simulink descritta in precedenza. Esiste, infatti, la possibilità di associare delle 'callback functions', ovvero delle funzioni MATLAB (o semplicemente degli script), al modello. Queste funzioni possono essere richiamate in fase di compilazione del modello, ad inizio simulazione, durante la simulazione (stoppandola per esempio) o a fine analisi. Questo può essere fatto per importare o esportare nel workspace di MATLAB variabili dal modello in maniera dinamica e rapida. All'interno del Model Explorer è possibile visualizzare non solo i valori associati a parametri e segnali, ma anche le caratteristiche che li descrivono, ovvero:

- Dimensioni: scalare, vettore, matrice, matrice multidimensionale.
- Complessità: valori reali o complessi.
- Range: valore minimo e massimo di un segnale o parametro
- Tipo di dato: single, double, interi con o senza segno (8,16 e 32 bit), booleani, virgola fissa (detti fixed-point in ambiente Mathworks).

Numeri in virgola fissa

Una menzione particolare va fatta sui dati a virgola fissa, in quanto il loro utilizzo si è reso necessario nell'implementazione di un modello compatibile con l'HDL Coder. Innanzitutto, è necessario un toolbox apposito per la rappresentazione dei dati in virgola fissa, chiamato Fixed-Point Designer. Parametri e segnali di un modello vanno necessariamente rappresentati in virgola fissa se si ha la necessità di esportare il modello su un dispositivo hardware come ad esempio l'FPGA. Questo tipo di dati, riproduce il valore numerico della variabile come una sequenza di bit. Il quantitativo di bit impiegabili per rappresentare un certo valore numerico dipende solitamente dalle risorse hardware a disposizione, tuttavia Simulink fornisce fino a 128 bit. Più elevato è il numero di bit utilizzati per descrivere il valore numerico, minore sarà l'errore di quantizzazione compiuto. Un numero in virgola fissa è caratterizzato da 3 parametri, la presenza o meno del bit di segno, la dimensione totale della parola di bit e il numero di bit utilizzati per descrivere la parte frazionaria del numero.

In figura 3.3, è mostrato l'esempio di un numero in virgola fissa senza segno. Se un numero fixed-point prevede di tenere in considerazione il segno,

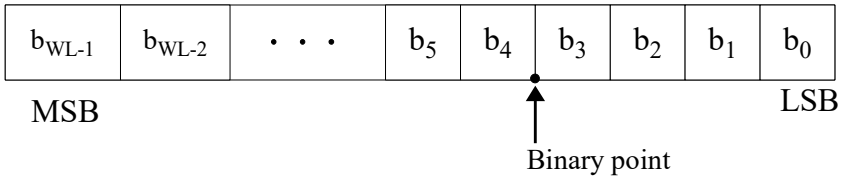

Figura 3.3: Rappresentazione grafica di un numero in virgola fissa senza segno

allora, il MSB della sequenza binaria serve a definire proprio se il segno sia positivo (0) o negativo (1). In questo caso, se WL (Word Length) rappresenta la lunghezza della parola di bit, solamente WL-1 sono i bit utilizzabili per rappresentare parte intera e frazionaria del numero. In riferimento alla figura 3.3, la virgola fissa si ritrova 4 bit alla sua destra. Questi bit sono quelli utilizzati per descrivere la parte frazionaria, mentre i restanti, da b_4 a $b_{\text{WL-1}}$, rappresentano la parte intera del numero. Come anticipato in precedenza, la scelta del numero di bit di quantizzazione introduce un errore più o meno significativo. In particolare, bisogna porre attenzione ai limiti di overflow e underflow, e valutare l'errore di troncamento compiuto. Di seguito sono mostrati i limiti di precisione imposti dai bit scelti:

Limiti di Overflow :

$$-2^{WL-FL-1} \leq N \leq 2^{WL-FL-1} - 1$$

Limiti di Underflow :

$$|N| \geq 2^{-FL}$$

Errore di troncamento :

$$|e| < 2^{-FL}$$

dove, N indica il valore numerico da riprodurre, WL rappresenta il numero di bit totali della parola, FL (Fraction Length) invece indica quanti bit si stanno usando per la parte frazionaria. Le problematiche principali che possono incorrere riguardano un errore di Overflow, quando non si hanno

bit sufficienti per rappresentare la parte intera (si ha una saturazione) e un errore di troncamento, quando la parte frazionaria non è sufficientemente precisa nella rappresentazione delle cifre a destra della virgola. In Simulink e MATLAB esistono tools appositi che permettono di convertire un modello definito da parametri e segnali in virgola mobile, in un modello composto da elementi in virgola fissa. Tuttavia, spesso conviene implementare il proprio modello direttamente con variabili fixed-point per avere un totale controllo delle dimensioni delle parole di bit utilizzate e quindi della precisione del sistema.

Configurazione della simulazione

Prima di poter effettuare l'analisi di un modello Simulink, è bene settare i parametri relativi alla simulazione che permettono di riprodurre dinamicamente il comportamento del modello in tempo reale. Per fare questo bisogna selezionare e programmare adeguatamente il risolutore più idoneo al tipo di sistema implementato. Esistono in Simulink differenti tipologie di risolutore. La prima distinzione riguarda il passo di campionamento dei segnali, che può essere fisso o variabile. La scelta di un risolutore a passo fisso è conveniente (talvolta necessaria) quando si ha a che fare con modelli multirate aventi frequenze di campionamento multiple tra di loro. In questo caso, lo step di campionamento può essere fissato manualmente o selezionato automaticamente dal risolutore, che sceglierà come valore un time-step basato alla rate più elevata del modello. Se, invece, viene scelto un risolutore a passo variabile, lo step di campionamento del segnale è automaticamente deciso da Simulink, in funzione di tutte le frequenze dei segnali. In generale, diminuire il time-step corrisponde ad una simulazione più accurata ma più lenta. Una simulazione con passo variabile viene gestita dal software in modo tale che, nelle fasi in cui le grandezze del modello subiscono variazioni rapide, il passo sia molto fine, mentre quando il modello si trova in situazioni pressoché statiche gli istanti di campionamento possano essere più distanziati. I risolutori Simulink si distinguono poi in risolutori continui e discreti. La scelta tra uno e l'altro dipende dal tipo di segnali propagati all'interno del modello. Risolutori continui utilizzano tecniche di integrazione numerica per calcolare le grandezze continue in un modello al time-step corrente, basandosi sui valori agli istanti precedenti e le derivate all'istante d'interesse. Risolutori continui sono anche in grado di valutare le grandezze discrete del sistema. Risolutori discreti, d'altra parte, possono essere impiegati solamente per campionare segnali discreti del modello. Infine, Simulink permette di configurare la propria simulazione in modalità Single Tasking o Multi Tasking.

Differenze tra un sistema Sample-Based e Frame-Based

In base alle considerazioni fatte nei paragrafi precedenti, un modello Simulink si distingue in base alle caratteristiche dei parametri e dei segnali propagati e in base alla configurazione di simulazione; ma non solo. Simulink, infatti, dà la possibilità di decidere la modalità di campionamento del segnale. Si può avere, perciò la propagazione di un segnale in modalità *Sample Based* o in modalità *Frame Based*. La scelta di un metodo o l'altro può dipendere da requisiti diversi, come ad esempio la velocità di simulazione del modello desiderata. Spesso, tuttavia, il sistema da implementare necessita obbligatoriamente di un determinato metodo di campionamento del segnale e l'utente è vincolato a seguire un certo approccio. Il mio lavoro di tesi si è basato proprio su vincoli di questo tipo, come verrà descritto nel seguito dell'elaborato.

Un segnale di tipo Sample-Based rappresenta il modello ideale di segnale utilizzato per implementare sistemi fisici e non puramente simulativi. Per creare un segnale di questo tipo, basta infatti campionare un segnale fisico ad una certa frequenza di campionamento ed emettere ogni singolo campione ricevuto alla volta. In generale, la maggior parte delle uscite dei convertitori D/A sono segnali Sample-Based.

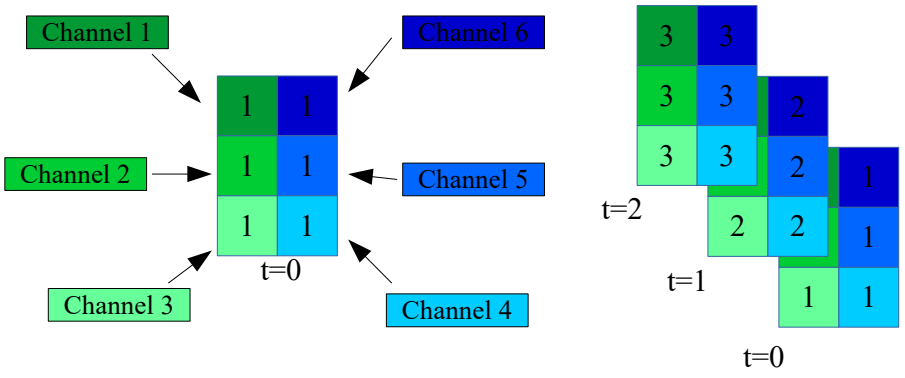

Figura 3.4: Esempio di propagazione di un segnale in modalità Sample-Based

La creazione di un segnale Frame-Based avviene elaborando il corrispondente segnale Sample-Based. Per fare questo basta bufferizzare un gruppo di N campioni del segnale originale. In uscita dal buffer si avranno sequenzialmente delle frame di dati ad una frequenza pari a 1/N volte la frequenza

originale del segnale Sample-Based. Questa nuova rate prende il nome di frequenza di frame. All'interno di Simulink, la maggior parte dei blocchi facenti parte di toolbox come DSP System Toolbox, dà la possibilità all'utente di decidere la tipologia di elaborazione del segnale in ingresso attraverso il parametro *Input Processing*, il quale può essere settato come :

- Elements as channel (Sample-Based)

- Columns as channel (Frame-Based)

Nell'elaborazione Sample-Based, i blocchi processano i segnali un campione alla volta. Ogni elemento del segnale in ingresso rappresenta un campione in un distinto canale. In figura 3.4, è mostrato un esempio di processamento Sample-Based, nel quale è rappresentata una matrice 3x2 contenente il primo campione in ognuno dei 6 canali indipendenti. Se si ha un ingresso scalare, il blocco lo interpreta come un segnale a singolo canale. Nel caso di una matrice, invece, il blocco interpreta l'input come un segnale formato da un numero di canali pari al prodotto di righe e colonne.

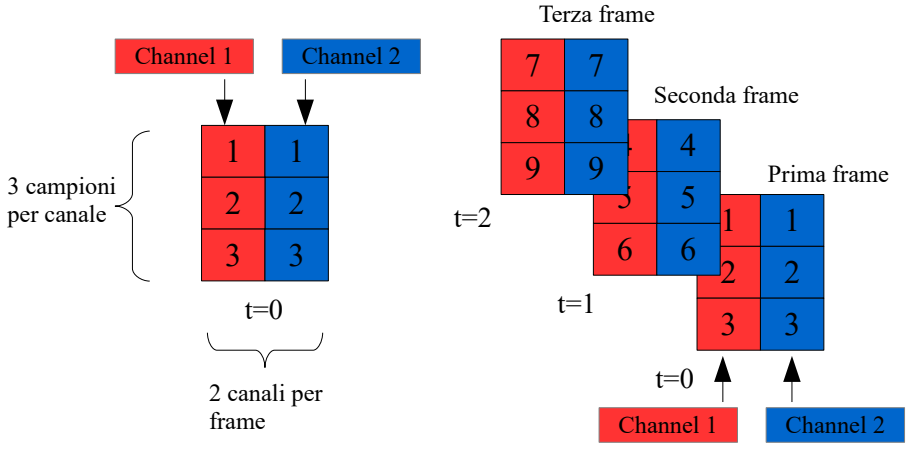

Figura 3.5: Esempio di propagazione di un segnale in modalità Frame-Based

In figura 3.5, è rappresentato un processamento in modalità Frame-Based. In questo caso, una matrice 3x2 in ingresso viene trattata dal blocco Simulink in maniera differente. Infatti ogni colonna della matrice viene intesa come un canale differente avente un numero di campioni per frame pari al numero

di righe. Ad ogni istante di campionamento si ha, quindi, la propagazione di una colonna intera.

L'utilizzo di una modalità di campionamento di tipo Frame-Based, ha lo scopo di accellerare notevolmente sia l'analisi di sistemi real time, sia la simulazione di modelli. Il vantaggio nell'elaborare frame di dati durante la simulazione di un modello sta nella diminuzione dell'overhead durante la comunicazione tra blocchi rispetto all'elaborazione di un singolo campione alla volta. Nei sistemi real-time, la propagazione di segnali Frame-Based rappresenta il formato più comunemente usato. La maggior parte dei sistemi di acquisizione dei dati a livello hardware opera accumulando un elevato numero di campioni ad elevata frequenza per poi propagare questi campioni al sistema real-time come blocchi di dati. Questo tipo di propagazione massimizza l'efficienza del sistema velocizzando l'acquizione dei dati.

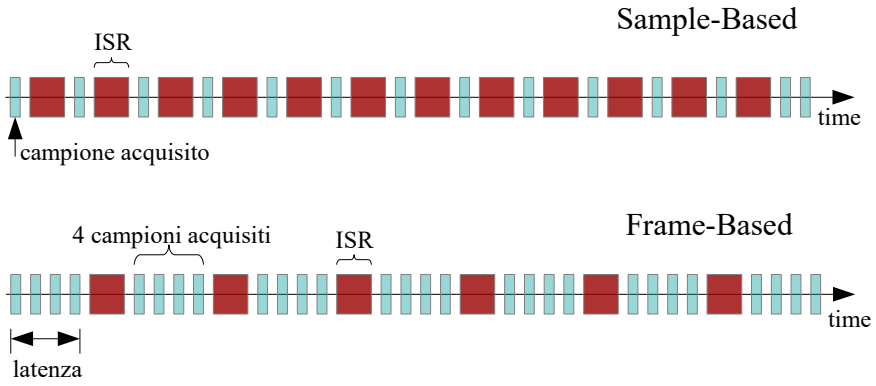

Figura 3.6: Differenza nell'acquisizione dei dati nei sistemi real-time tra la modalità Sample-Based e la modalità Frame-Based

Le operazioni di Interrupt Service Routine (ISR) necessarie a leggere i dati trasmessi dall'hardware vengono realizzate, infatti, alla fine di ogni frame. Nel caso Sample-Based, invece, si ha l'operazione di ISR dopo l'acquisizione di ogni singolo campione. In figura 3.6 è mostrata la differenza tra le due situazioni ed è possibile osservare il *throughput* maggiore nel caso Frame-Based.

Nonostante la propagazione in modalità Frame-Based si riveli molto vantaggiosa grazie ai motivi precedentemente descritti, esistono situazioni particolari in cui un suo impiego risulta impossibile da implementare. L'esempio

più lampante è quello che verrà descritto nel prossimo sottocapitolo e che ha riguardato il mio lavoro di tesi.

3.2 HDL Coder

Fino ad ora è stata data una descrizione delle principali caratteristiche che vanno a caratterizzare la simulazione e l'analisi di un generico modello Simulink. Ad inizio capitolo, è stata però introdotta la vera motivazione che ha spinto il progetto SCAT ad utilizzare questo ambiente di programmazione per la fase di realizzazione degli algoritmi di comunicazione satellitare. La ragione è un tool sviluppato da Mathworks (utilizzabile sia in MATLAB che in Simulink) che prende il nome di *HDL Coder*. Questo fondamentale strumento permette di convertire il proprio modello Simulink, la propria MATLAB function o la propria macchina a stati (Stateflow) in un codice VHDL o Verilog sintetizzabile. I codici di linguaggio di descrizione dell'hardware (HDL) generati dal tool di conversione possono poi essere utilizzati per programmare un FPGA (come nel caso del progetto SCAT). Mathworks fornisce, in aggiunta al *HDL Coder* anche un altro strumento chiamato *HDL Verifier* utile a verificare il corretto funzionamento del proprio codice VHDL rispetto al modello implementato su Simulink o MATLAB.

Durante lo svolgimento della tesi, lo strumento *HDL Verifier* non è stato però utilizzato. Lo scopo della prima parte del progetto SCAT, di cui mi sono occupato, è infatti quello di riuscire a progettare un modello Simulink che sia convertibile con successo in codice VHDL. La fase successiva di validazione del modello e di ottimizzazione dello stesso in termini di occupazione d'area e massima frequenza di lavoro del FPGA non ha riguardato lo svolgimento della mia tesi. Tuttavia di seguito sono elencate alcune funzionalità automatiche che l'*HDL Coder* e l' *HDL Verifier* forniscono a loro volta per poter ottimizzare e validare il lavoro svolto, ad esempio:

- Operazioni necessarie all'ottimizzazione come l'inserimento automatico di *Pipeline Register* per incrementare la frequenza utilizzabile dal dispositivo, funzionalità di *Resource Sharing* e *RAM Mapping* per diminuire l'occupazione d'area e il numero di risorse necessarie.

- Un tool chiamato *Delay Balancing* utilizzato per rilevare la presenza di ritardi dovuti a percorsi critici e tenerne conto su tutti i percorsi in parallelo del proprio modello. Questo si fa per evitare che il funzionamento del codice HDL si differenzi dal modello Simulink.

- Generazione di un *Validation Model* necessario per poter confrontare il funzionamento del modello originale e del modello generato.

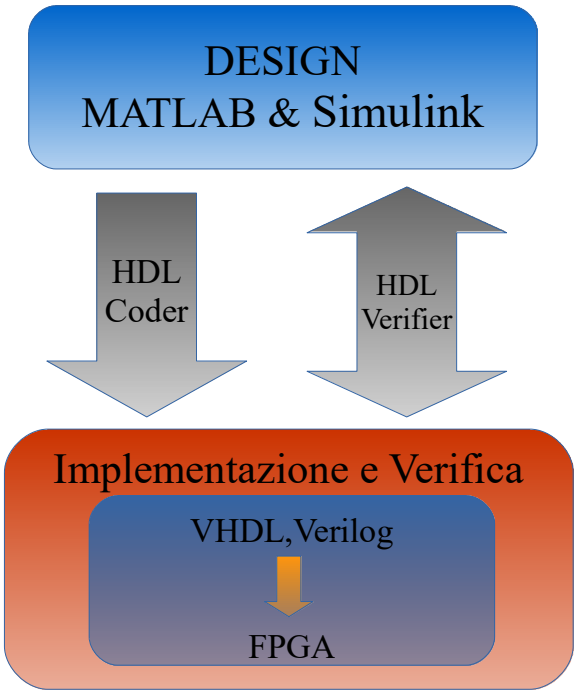

Figura 3.7: Schema di flusso per la conversione di modelli Mathworks in codice HDL

- Generazione di *Test Bench* per testare in ambiente Simulink il modello o il sottosistema specificato.

In figura 3.8, è mostrato l'esempio di un modello Simulink [10] al quale sono stati aggiunti dei blocci di ritardo unitario (colorati in arancione) che rappresentano in linguaggio VHDL dei pipeline register. Questo *pipeling* distribuito riduce i collegamenti critici del modello permettendo un utilizzo del FPGA con un clock rate maggiore, aumentando lo througput. Si ha, tuttavia, un compromesso tra le performance migliorate in termini di velocità e un conseguente aumento dell'occupazione d'area.

HDL Workflow Advisor

Lo strumento fornito dal *HDL Coder* che ho effettivamente utilizzato durante lo svolgimento della tesi è chiamato *HDL Workflow Advisor*. Questo tool

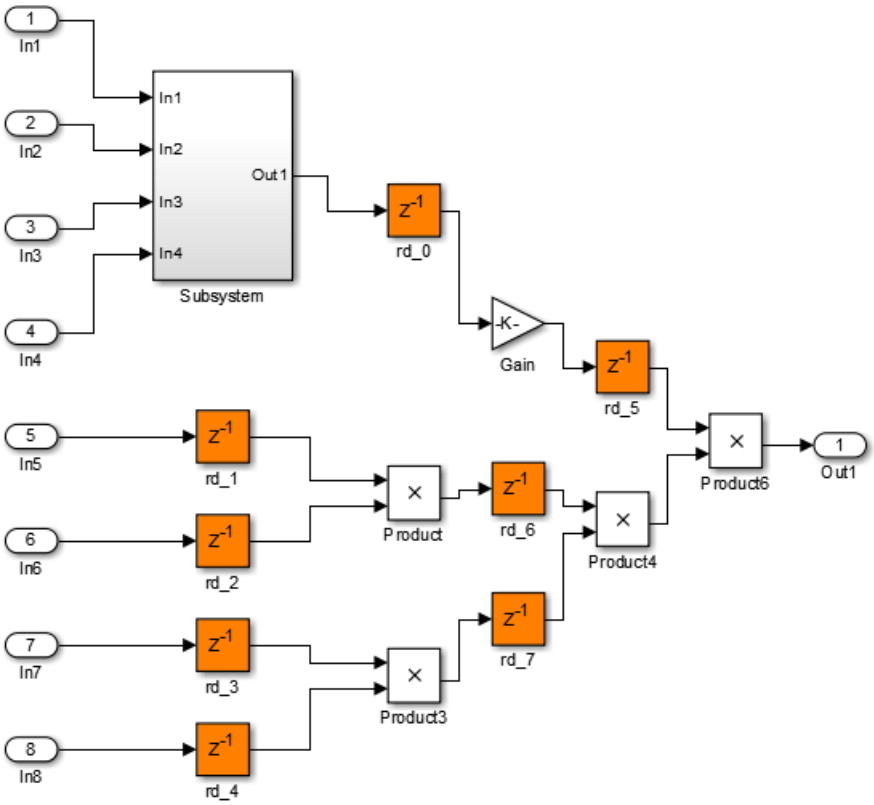

Figura 3.8: Esempio di modello Simulink con aggiunta di Pipeline Register distribuiti

permette di convertire il proprio modello o sottosistema Simulink nel desiderato codice VHDL o Verilog, attraverso una procedura suddivisa in più passaggi. I 3 passaggi da effettuare prima di aver completato la generazione del codice sono i seguenti:

- *Set Target*. Il primo passaggio consiste nella creazione della cartella di lavoro in cui verranno salvati i codici HDL generati e nel settaggio dei parametri del dispositivo (target) su cui il codice HDL sarà caricato, come ad esempio la frequenza massima di funzionamento. Se il target non è noto vengono lasciate le impostazioni di default.

- *Prepare Model for HDL Generation*. Questa fase è la più delicata, in quanto avviene un check sul modello realizzato, in particolare l'*HDL Workflow Advisor* definisce 3 campi da controllare: l'esistenza di loop algebrici, la compatibilità dei blocchi e i tempi di campionamento del

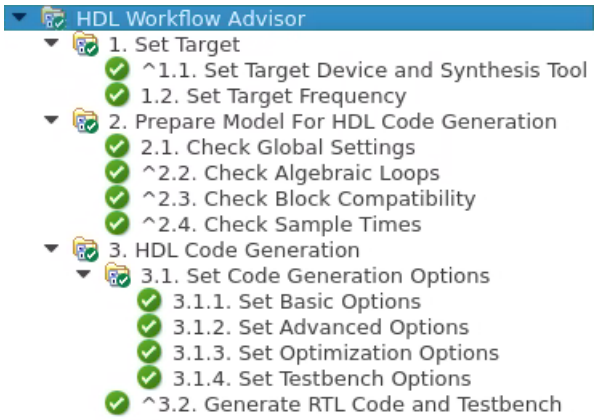

Figura 3.9: Processo di generazione del codice HDL mediante HDL Workflow Advisor

sistema. Solitamente, durante questo passaggio si ha la conferma di aver realizzato un modello realmente convertibile o no.

- *HDL Code Generation.* L'ultimo step è quello che effettivamente si occupa della generazione del codice RTL (Register Transfer Level). In questo passaggio è possibile settare a proprio piacimento diverse opzioni, da quelle basilari, ad opzioni più avanzate. Questo può essere fatto per cercare di ottimizzare il codice (RAM Mapping, Resource Sharing, Pipeling Registers), ma anche per cercare di renderlo più leggibile all'utente finale. Nella fase finale, inoltre, è possibile generare Modelli di Validazione, di TestBench e di Cosimulation, in aggiunta al codice HDL.

Attraverso *HDL Workflow Advisor*, è possibile convertire interamente il proprio modello Simulink in linguaggio di descrzione dell'hardware, o convertire singoli sottosistemi separatamente. In un modello gerarchico particolarmente complesso, la generazione dei codici VHDL per i singoli sottosistemi può convenire in quanto risulta più semplice un controllo sul funzionamento del modello generato grazie agli strumenti forniti da *HDL Coder* e *HDL Verifier*. Tuttavia, *HDL Coder* ha la capacità di analizzare un intero modello e di gestire i vari sottosistemi che lo compongono. I vari blocchi o sottosistemi che formano un modello vengono infatti sintetizzati come singoli codici VHDL e importanti nel codice VHDL ad alto livello come componenti o entità dello stesso. Il file VHDL di alto livello si occupa poi di gestire la dichiarazione delle porte di I/O, i collegamenti tra le entità e tutte le operazioni necessarie

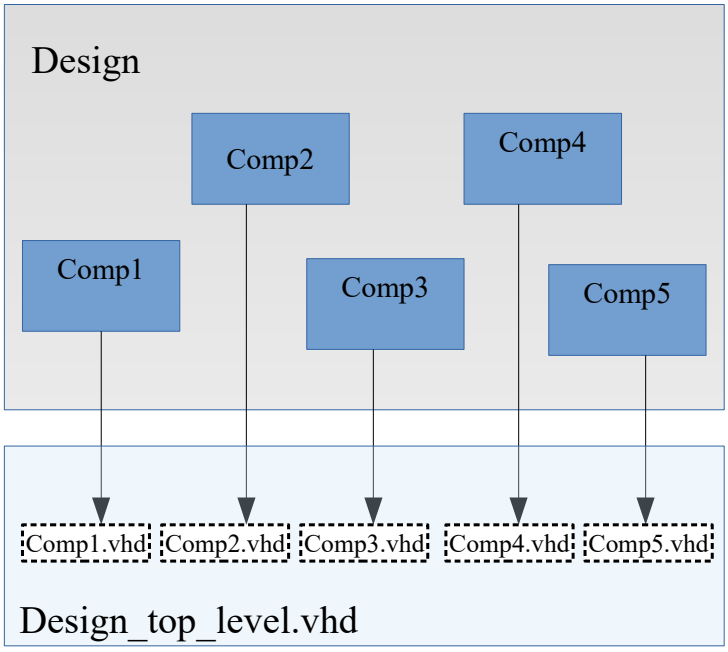

Figura 3.10: Gestione di una struttura gerarchica durante la generazione del codice VHDL

per integrare i vari blocchi del sistema. Un esempio di struttura gerarchica generata dal *HDL Coder* è mostrata in figura 3.10.

3.2.1 Requisiti del modello per la conversione da Simulink a VHDL

La progettazione di un modello Simulink che sia convertibile in linguaggi di descrizione dell'hardware è molto differente rispetto alla modellazione di un sistema utilizzato solamente sulla piattaforma software per analisi e simulazioni. Il modello convertibile deve infatti rispettare alcuni requisiti fondamentali imposti da Simulink, che lo rendano compatibile ad applicazioni hardware. Di seguito sono elencati alcuni vincoli fondamentali da dover rispettare per non incorrere in errori durante l'utilizzo del *HDL Workflow Advisor*.

Il modello Simulink compatile con l'*HDL Coder* deve, innanzitutto, essere composto da blocchi che appartengano a librerie tollerate dal coder. La

Figura 3.11: Differenze tra il blocco di codifica Reed-Solomon convertibile HDL e il blocco classico Simulink

maggior parte dei blocchi disponibili sono posizionati all'interno della libreria denominata proprio *HDL Coder*. Tuttavia, esistono anche blocchi ottimizzati per la conversione HDL che appartengono a toolbox specifici. Oltre al toolbox *Communications System Toolbox* esiste, ad esempio, il toolbox *Communications System Toolbox HDL Support*,che contiene blocchi appartenenti al mondo dei sistemi di telecomunicazioni convertibili in codice HDL. Come verrà descritto in seguito, le differenze tra un approccio puramente simulativo e un approccio vicino al mondo reale, sono notevoli. Un esempio lo si ha osservando la figura 3.11, che mostra due blocchi relativi alla codifica Reed-Solomon. Il blocco a destra è quello convertibile in HDL e quello a sinistra invece il blocco classico usato a livello simulativo. La principale differenza tra i due blocchi consiste nella tipologia di dati che accettano in ingresso. Il blocco classico della libreria Simulink accetta infatti trame dati, ovvero segnali campionati in modalità Frame-Based di lunghezza pari al messaggio di informazione. In uscita, fornisce invece (con la stessa frequenza di campionamento) una frame più lunga che corrisponde alla codeword generata dall'encoder. Il blocco HDL accetta invece un segnale di tipo Sample-Based; quindi non avrà in ingresso direttamente la trama completa, ma un campione di essa alla volta. Per poter effettuare l'operazione di codifica, necessita perciò di ulteriori segnali di controllo che definiscano l'inizio e la fine della frame e se il segnale in ingresso sia valido, ovvero un campione da codificare. L'esempio appena visto, mostra come un modello compatibile con l'*HDL Coder* debba seguire certi vincoli sulla tipologia di campionamento dei dati.

Segnali di tipo Frame-Based non sono vietati completamente, ma bisogna porre molta attenzione nella dimensione della frame, in quanto da un punto di vista hardware, una frame di N campioni verrà trattata sul FPGA come N collegamenti in parallelo. Per facilitare l'implementazione del codice HDL, è quindi consigliato un approccio seriale in modalità Sample-Based.

Un'altro requisito fondamentale da dover rispettare, riguarda il tipo di dato con cui segnali e parametri di un modello devono essere rappresentati. La rappresentazione mediante dati in virgola mobile (double, single) non è accettata dal coder. I tipi di dato accettabili dall'*HDL Coder* sono i numeri in virgola fissa, i booleani, gli interi con e senza segno a 8,16 e 32 bit. La maggior parte dei blocchi utilizzabili permettono, all'interno della loro maschera, di settare sia i parametri principali che il numero di bit per rappresentare i dati in uscita o i dati che vengono propagati all'interno del blocco. Ad esempio, in un filtro FIR, si possono fissare il numero di bit per definire l'uscita, l'accumulatore, i coefficienti del filtro e i prodotti delle moltiplicazioni. Spesso, gestire grandezze in virgola fissa può essere complicato se il modello è molto complesso. Per venire incontro a questa problematica, Simulink mette a disposizione un altro tool che permette di creare un modello in virgola mobile e di convertire tutte le grandezze in dati fixed-point.

Tra i vari passaggi che l'*HDL Workflow Advisor* deve superare, c'è anche il controllo di eventuali loop algebrici nel modello.

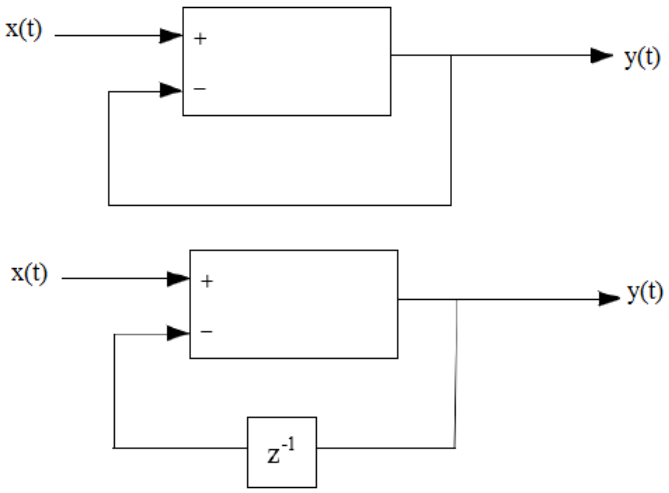

Figura 3.12: Esempio di loop algebrico

Un loop algebrico in un modello si ha quando l'ingresso di un blocco viene comandato direttamente dall'uscita, ad esempio in un sistema in retroazione come in figura 3.12. Il problema di base, è che l'uscita all'istante t è funzione di se stessa sempre all'istante t. Per risolvere la problematica, in figura 3.12 è rappresentata una possibile soluzione, che consiste nell'introdurre un ritardo unitario nel collegamento in retroazione. In questo modo l'*HDL Coder* non rileva problemi.

L'ultimo step nella seconda fase del *HDL Workflow Advisor* riguarda i tempi di campionamento del modello, o in generale, della temporizzazione che lo riguarda. Per effettuare una conversione HDL, è necessario, infatti, rispettare alcuni requisiti imposti dal software durante la fase di configurazione dei parametri di simulazione. Le impostazioni del risolutore devono sottostare a vincoli specifici. Per prima cosa, il risolutore deve avere un passo di campionamento del segnale fisso e non variabile. Questo è dovuto al fatto che il passo di campionamento dipende dal clock che comanda il dispositivo hardware e che chiaramente è un parametro fisso. Per lo stesso motivo, il risolutore deve essere discreto e il modello non deve prevedere stati continui. Infine, il sistema deve essere necessariamente settato in modalità Single Tasking. L'*HDL Coder* non supporta modelli che lavorano in modalità multitask. Una caratteristica che i modelli Simulink compatibili con il coder HDL hanno, ma che invece i modelli modelli MATLAB HDL non hanno, è la possibilità di gestire sistemi di tipo multirate. In questo caso il codice HDL generato si complica dovendo gestire blocchi con temporizzazioni diverse, ma il tutto è possibile grazie all'utilizzo di segnali di clock_enable appositamente generati.

Capitolo 4

Implementazione Simulink della catena di trasmissione DVB-S2

Nel presente capitolo dell'elaborato, viene fornita la descrizione del lavoro svolto mediante Simulink durante la prima fase del progetto SCAT. Tra i tre segnali che il transceiver deve gestire, il primo che è stato affrontato e di cui è stato implementato il sistema corrispondente a livello software, è il segnale di *Feeder Link* in downlink, quindi in trasmissione. Il protocollo di comunicazione che si è optato di utilizzare, per elaborare il segnale seriale dal computer di bordo fino al modulo analogico del transceiver, è lo standard DVB-S2, descritto nel secondo capitolo dell'elaborato. La ragione principale che ha portato a trattare per primo il *Feeder Link* , piuttosto che il *Telecommando* o la *Telemetria*, è l'elevata data rate con cui deve essere trasmesso dal satellite. I requisiti imposti da ESA indicano, infatti, una bit rate compresa tra 1 Mbit/s e 50 Mbit/s a valle della codifica di canale. Al contrario dei segnali di controllo del transceiver si ha, quindi, una velocità di trasmissione dei dati notevole che potrebbe mettere in difficoltà l'FPGA su cui sviluppare il sistema.

4.0.1 Valutazione delle frequenze di progetto

Il valore scelto di baud rate (B_s) a valle della sezione di *PL Framing*, in accordo con ESA, è pari a 12.875 Msymb/s. Nonostante il sistema da implementare su FPGA sia di tipo VCM, questo valore di baud rate rimane costante; il cambio nella configurazione di codifica e modulazione corrisponde, perciò, ad una variazione della bit rate di sorgente e non dei blocchi che si occupano dell'interpolazione del segnale e della conversione a frequenza intermedia. I modelli realizzati in Simulink che verranno descritti nel proseguo del capitolo sono in grado di gestire qualsiasi configurazione di codifica e

modulazione prevista dallo standard DVB-S2 [8]. Di seguito sono però elencate solamente le tre possibili combinazioni che il progetto SCAT prevede all'interno del proprio sistema VCM, ovvero:

1. QPSK, rate 1/2.
2. QPSK, rate 3/4.
3. 16APSK, rate 3/4.

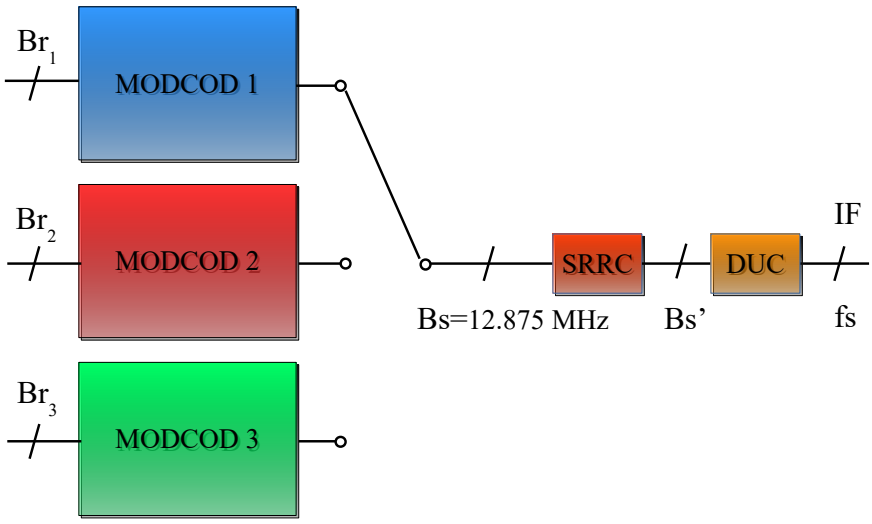

Figura 4.1: Possibile implementazione di un sistema VCM a baud rate costante

La scelta di una combinazione o l'altra va a modificare la bit rate di sorgente in ingresso dalla seriale (Br_i) e anche le sezioni a monte del filtro a radice di coseno rialzato (SRRC). Le sezioni in questione sono quelle definite nell standard DVB-S2 e chiamate *Stream Adaptation*, *FEC Encoding*, *Mapping* e *PL Framing*. In figura 4.1, l'insieme di questi blocchi è racchiuso, per ogni combinazione, all'interno del rettangolo denominato $MODCOD_i$. Nonostante alcune operazioni siano in comune tra 2 combinazioni differenti, come ad esempio la modulazione QPSK, le 3 configurazioni di figura 4.1 sono rappresentate come 3 catene parallele selezionabili singolarmente da un commutatore. I valori di bit rate per ognuna delle 3 configurazioni dipendono

dalla ridondanza introdotta dalla codifica di canale, dalla ridondanza del *PL Framing* e dalla costellazione scelta. Il calcolo è mostrato di seguito.

$$B_r = B_s \cdot R_{bch} \cdot R_{ldpc} \cdot \log_2 M \cdot \frac{XFEC}{XFEC + 90 + P},$$

dove B_r è la bit rate necessaria, B_s è la baud rate costante e pari a 12.875 Msymb/s, R_{bch} e R_{ldpc} sono le code rate dei due encoder, M è il numero di possibili punti sulla costellazione, XFEC è la dimensione della *XFECFRAME*, 90 sono i simboli aggiunti dall'header e P è il numero di toni pilota inseriti nella trama e il cui calcolo è mostrato nel secondo capitolo. Nel caso QPSK e rate 1/2, ricordando che durante lo svolgimento della tesi ho trattato solamente trame LDPC di lunghezza normale (64800 bit), la bit rate risultante è la seguente:

$$B_{r1} = 12.875 \cdot 10^6 \cdot \frac{32208}{32400} \cdot \frac{1}{2} \cdot \log_2 4 \cdot \frac{32400}{32400 + 90 + 36 \cdot 22} \simeq 12459500 \quad bit/s$$

I valori di bit rate negli altri due casi, sono i seguenti:

$$B_{r2} \simeq 18726400 \quad bit/s$$
$$B_{r3} \simeq 37351800 \quad bit/s$$

I blocchi a valle dello switch di figura 4.1, ovvero il filtro a radice di coseno rialzato (SRRC) e il Digital Up Converter (DUC), sono invariati qualsiasi siano i parametri di codifica e modulazione. La caratterizzazione di questi elementi della catena dipende dalla frequenza intermedia (IF) imposta da ESA per il segnale di *Feeder Link* e dalle specifiche tecniche del FPGA, in particolare la massima clock rate a cui può lavorare. La frequenza intermedia con cui il segnale deve uscire in digitale dal FPGA ed entrare nel convertitore digitale analogico è pari a 460 MHz. Teoricamente, la conversione del segnale dalla banda base alla IF avviene mediante una modulazione in quadratura, moltiplicando il proprio segnale complesso (via in fase e quadratura) per il segnale a frequenza intermedia generato da un oscillatore locale. Prima di traslare lo spettro del segnale dalla banda base alla IF, c'è bisogno però di aumentare la frequenza di campionamento del segnale, in modo da rispettare il Teorema del Campionamento di Nyquist-Shannon e non incorrere in fenomeni di aliasing. Per fare questo, il filtro a radice di coseno rialzato e il DUC si occupano di filtrare e interpolare il segnale dell'opportuno fattore di interpolazione. La frequenza di campionamento deve essere almeno il doppio della IF a 460 MHz, quindi vicina a 1 Gsps. Tuttavia, dopo alcune prime

analisi, è stato osservato come l'FPGA su cui verrà sviluppato il sistema di trasmissione non sia in grado di gestire frequenze di clock così elevate. Si è dovuto, perciò, pensare ad un approccio alternativo. La soluzione pensata si basa sul Teorema del Campionamento Passabanda.

Campionamento Passabanda e scelta dei fattori di interpolazione

Per riuscire ad avere lo spettro del segnale alla frequenza intermedia desiderata di 460 MHz, quello che si può fare, sfruttando il dominio digitale in cui si sta operando, è variare la frequenza del segnale generato dall'oscillatore locale e la frequenza di campionamento, in maniera opportuna. In particolare, la frequenza di campionamento che è stata utilizzata è pari a 360,5 Msps, quindi ben al di sotto della frequenza di Nyquist e addirittura inferiore alla IF. L'utilizzo della nuova frequenza di campionamento, combinata con una frequenza intermedia di 99.5 MHz, fa si che si generi una replica dello spettro esattamente a 460 MHz, senza incorrere in aliasing.

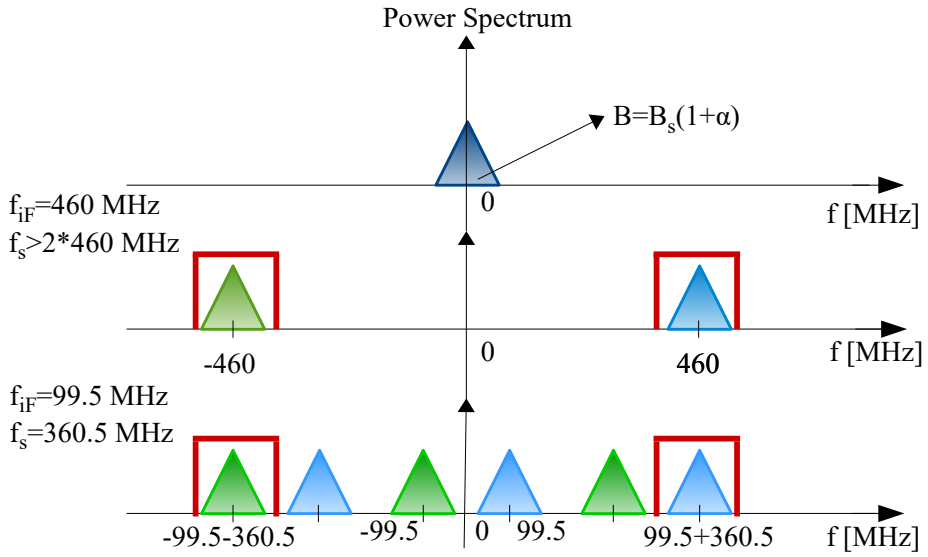

Figura 4.2: Utilizzo del campionamento Passa Banda

In figura 4.2 è rappresentato l'approccio utilizzato per risolvere la problematica precedentemente spiegata. L'immagine mostra come, effettuare un campionamento Passabanda con frequenza di campionamento pari a 360.5

Msps e IF pari a 99.5 MHz permetta di generare la replica desiderata dello spettro a 460 MHz. A questo punto, sempre in figura, è mostrato come sia sufficiente utilizzare un filtro passabanda per isolare la porzione di spettro desiderata. Il filtraggio in questione non fa parte del sistema digitale da implementare su FPGA, ma è una operazione di tipo analogico implementata a valle del DAC. Il filtraggio dovrà tenere conto della larghezza di banda dello spettro, in funzione del fattore di roll-off α, e pari a:

$$B = B_s \cdot (1 + \alpha) = 12.875 \cdot 10^6 \cdot (1 + 0.35) \simeq 17.38 \quad \text{MHz}$$

Questo approccio fornisce un ulteriore vantaggio, oltre all'avere diminuito notevolmente la frequenza di campionamento del segnale a IF, che rappresenta la frequenza massima del sistema e che quindi determina il segnale di clock del FPGA. Il rapporto tra la sample rate finale della catena e la baud rate a monte del SRRC è, infatti, un valore intero e pari a :

$$I = \frac{f_s}{B_s} = \frac{360.5 \cdot 10^6}{12.875 \cdot 10^6} = 28.$$

Il termine I rappresenta il fattore di interpolazione complessivo che deve essere distribuito tra il filtro interpolatore a radice di coseno rialzato, che funge anche da filtro sagomatore e i filtri interpolatori che vanno a comporre il Digital Up Converter. Esistono differenti possibilità per suddividire il fattore di interpolazione totale tra i due blocchi. La scelta di una o l'altra incide sulla complessità totale del sistema e anche sulla qualità dello spettro del segnale generato rispetto alla maschera fornita dallo standard, di figura 2.14. La configurazione iniziale che si è deciso di attuare, ma che in una seconda fase di ottimizzazione del codice VHDL potrà essere modificata, prevede la seguente ripartizione:

- I_1=14. Fattore di interpolazione del filtro SRRC.
- I_2=2. Fattore di interpolazione del DUC.

4.0.2 Introduzione ai modelli Simulink

Uno dei vantaggi principali dello standard DVB-S2, come già descritto nel secondo capitolo della tesi, è la possibilità di contemplare un elevato numero di combinazioni di codifica e modulazione in funzione delle caratteristiche del canale satellitare. Lo standard prevede, quindi, la modalità VCM (Variable Coding & Modulation) e la modalità ACM (Adaptive Coding & Modulation). Il sistema di trasmissione del *Feeder Link* previsto dal progetto SCAT è di tipo VCM. Il transceiver in orbita deve essere, perciò, in grado di modificare

i propri parametri di codifica e modulazione, ad avvenuta ricezione di un apposito comando d'istruzione da Terra. Una variazione real-time di questo tipo, in ambiente Simulink, non è possibile per sistemi di comunicazione così elaborati. La via che è stata scelta per poter realizzare un singolo modello Simulink che contemplasse tutte le possibili combinazioni di MODCOD è quella di progettare un sistema totalmente parametrizzato e configurabile mediante il Model Explorer di Simulink. Questo criterio permette di non dover gestire un elevato numero di modelli differenti, ma di poter operare sempre sulla stessa catena e con un numero di blocchi limitato. Un approccio di questo tipo è obbligatorio considerando l'ambiente di programmazione Simulink. L'implementazione su FPGA, tuttavia, dovrà seguire una strada differente, probabilmente molto simile a quella di figura 4.1, nella quale è prevista la diramazione di tre vie parallele. L'operatività di una catena o l'altra verrà decisa, probabilmente, mediante opportuno switch a livello hardware. La parametrizzazione del modello Simulink si basa su una cosiddetta *callback function* scritta in ambiente MATLAB, la quale viene richiamata in fase di inizializzazione del modello attraverso il Model Explorer, e denominata *parameter_list*.

```
pilots=1; % 0 = no pilots or 1 = presence of pilots
n_slots=16;
n_sym_per_slot=90;
n_sym_per_pilot=36;
PLHeader_length=90;
MPEG_packet=1504; % in bits
numOfPacketsToBeSent=1000000000; % if it is = -1 -> just 1 repetition



Rc = 1/2; %1/4, 1/3, 2/5, 1/2, 3/5, 2/3, 3/4, 4/5, 5/6, 8/9, 9/10
Modul= 'QPSK'; %QPSK, 8PSK, 16APSK, 32APSK
% Rc = 3/5 + 8PSK NOT AVAILABLE!
nldpc=64800; % 64800 or 16200 (ACTUALLY ONLY 64800)

str=combinationValid(Rc,Modul,nldpc);
if (~isempty(str))
    error(str)
end

clearvars str
S=retrieveBCH_LDPC_info(nldpc,Rc);
```

Figura 4.3: Porzione della callback function *parameter_list*

In figura 4.3, è mostrata una porzione del file MATLAB *parameter_list*, nel quale vengono definiti alcuni parametri fissi del sistema, come ad esempio la dimensione del PLHeader, la dimensione di una trama MPEG-2, il numero di toni pilota per singola finestra. Inoltre, sono mostrate anche le righe di codice necessarie a settare la combinazione di codifica e modulazione voluta. Ogni volta che si vuole cambiare la configurazione del sistema, la MATLAB function precedentemente descritta risulta il primo elemento del modello in cui apportare modifiche. La lista dei parametri non rappresenta, però, l'unica *callback function* da associare al modello. Per effettuare valutazioni sui risultati del sistema, è opportuno allegare al modello anche una o più funzioni che, terminata la simulazione Simulink, siano in grado di importare dal workspace i dati di interesse ed effettuare analisi appropriate, come ad esempio il calcolo della PER (Packet Error Rate) in un modello completo (trasmissione più ricezione).

Nel capitolo 3 dell'elaborato, è stato descritto lo strumento necessario a generare dal proprio file Simulink il codice VHDL corrispondente, ovvero l'*HDL Coder*. In particolare, sono stati elencati i principali requisiti che un modello Simulink deve rispettare per poter essere convertito correttamente in linguaggio hardware. Lo sviluppo di una catena compatibile con l'*HDL Coder* rappresenta l'obiettivo conclusivo e principale del mio lavoro di tesi, tuttavia, i modelli Simulink complessivamente progettati sono due, ossia:

- *Catena Frame-Based*. Il suo scopo principale è quello di valutare le performance del sistema in termini di PER (Packet Error Rate), infatti, il modello comprende anche la catena di ricezione. Inoltre, i segnali generati dal modello fungono da campioni di riferimento per i corrispondenti segnali della catena Sample-Based.

- *Catena Sample-Based*. Essa comprende la sola catena di trasmissione e rappresenta il vero modello Simulink compatibile con l'*HDL Coder* da convertire in codice VHDL.

Nel seguito del capitolo, verranno descritti nel dettaglio entrambi i modelli e i criteri che hanno guidato la loro implementazione.

4.1 Catena Frame-Based per la valutazione delle performance

Prima di realizzare il modello Simulink effettivamente di interesse per il progetto SCAT, è stato necessario prendere confidenza con lo standard DVB-S2

e implementare una catena che permettesse la valutazione del sistema in termini di PER in funzione del rapporto segnale rumore. Questo è stato fatto per poter confrontare i risultati rinvenuti, con le performance indicate in letteratura. Con il termine PER si indica, in questo caso, il numero di pacchetti da 1504 bit ciascuno (dimensione dei pacchetti MPEG-2) persi durante la trasmissione sul canale satellitare. I risultati riscontrati per ogni combinazione possibile di modulazione e codifica sono forniti nel successivo capitolo.

L'altra utilità di questa catena è stata sfruttata successivamente. Essa infatti ha funto da modello di riferimento per il corretto funzionamento della catena Sample-Based; i segnali in uscita da ogni suo singolo blocco della catena di trasmissione sono stati infatti confrontati con quelli della catena Sample-Based per verificarne l'identico comportamento. Questo è dovuto al fatto che, come verrà descritto in seguito, la maggior parte dei sottosistemi del modello compatibile con l'*HDL Coder* non sono nativi Simulink, ma sono stati realizzati mediante delle MATLAB Function. Al contrario, le librerie Simulink, in particolare il *Communication Systems Toolbox*, mettono a disposizione diversi blocchi compresi nello standard DVB-S2, come ad esempio l'encoder BCH, l'encoder LDPC e l'interleaver. Questi elementi non sono convertibili in codice HDL e prevedono una modalità di campionamento dei segnali di tipo Frame-Based.

Il primo passaggio della catena corrisponde, quindi, ad un'operazione di bufferizzazione dei dati seriali in ingresso dalla sorgente fittizia. Questa sorgente ideale propaga dei campioni binari forniti dalla *callback function* che si occupa degli input del modello. Il numero di bit della sorgente che vanno a formare la frame è pari alla dimensione della *BBFRAME* e quindi del messaggio di informazione in ingresso all'encoder BCH.

I blocchi successivi sono quelli previsti dallo standard, ovvero il *BB Scrambler* che si occupa della randomizzazione della *BBFRAME* e tutti i sottosistemi relativi alla sezione di *FEC Encoding*. Il vantaggio di una implementazione Frame-Based per la fase di analisi del sistema, sta nel fatto che tutti questi blocchi sono nativi Simulink. Risulta, perciò sufficiente, progettare adeguatamente il blocco attraverso i parametri della relativa maschera. In figura 4.4 è mostrata la sequenza di questi sottosistemi nel caso QPSK e rate 1/2, in cui si può notare come l'uscita di ogni blocco sia definita come un vettore colonna della dimensione della corrispondente frame dello standard DVB-S2. Il timing con cui i segnali di figura 4.4 vengono campionati rimane costante per ognuno di essi. L'aumento della bit rate introdotto dai bit di ridondanza si manifesta con l'aumento della dimensione delle frame, a parità di tempo di campionamento. Questa è la caratteristica principale di un sistema di comunicazione di tipo Frame-Based realizzato in Simulink.

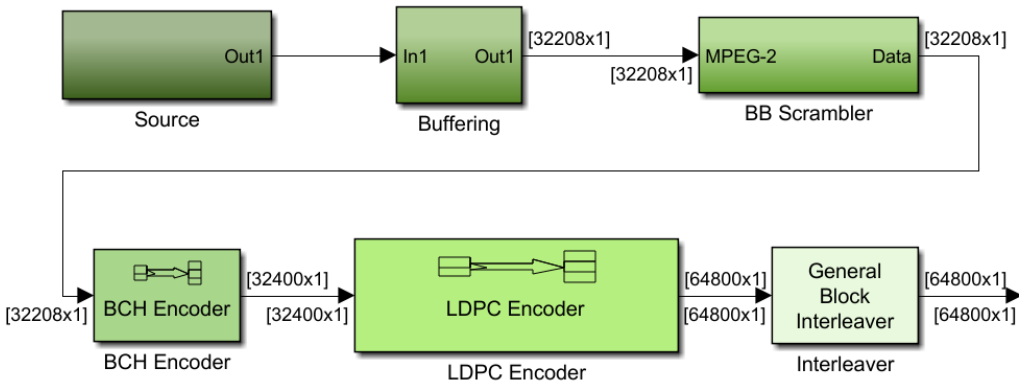

Figura 4.4: Stream Adaptation e FEC Encoding nella catena Frame-Based

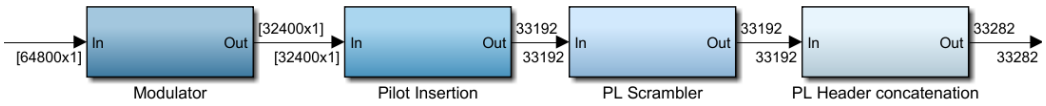

Figura 4.5: Mapping e PL Framing nella catena Frame-Based

L'immagine 4.5 mostra il proseguo della catena di trasmissione. Essa comprende un sottosistema relativo alla fase di *Mapping*, nel quale, mediante *callback function*, si può scegliere la costellazione desiderata. I blocchi successivi riguardano le operazioni di *PL Framing*. Al contrario degli elementi precedenti, queste operazioni dello standard non possono essere affidate a blocchi nativi Simulink. Tuttavia, sfruttando l'approccio Frame-Based, risulta agevole la concatenazione dell'header fisico, dei toni pilota e la fase di scrambling fisico. L'inserzione dei toni pilota avviene attraverso una MATLAB Function; lo scrambling mediante una moltiplicazione tra elementi corrispondenti di due vettori, di cui uno è la frame di sistema e l'altro la sequenza di scrambling calcolata offline e introdotta nel modello tramite Model Explorer. Infine, l'inserzione del PLHeader avviene semplicemente concatenando il vettore di 90 simboli complessi alla trama. Rispetto alla sequenza di operazioni definita nello standard DVB-S2 [8], in questo modello, l'ordine delle operazioni di *PL Framing* è stato alterato, effettuando prima l'inserzione dei toni pilota, poi lo scrambling e infine l'inserzione dell'header. Questo è dovuto all'approccio Frame-Based; infatti risulta più comodo concatenare

l'header per ultimo, non essendo soggetto a scrambling di livello fisico.

Dato che lo scopo principale di questo modello è valutare le performance del sistema in termini di PER, non è necessario contemplare i blocchi SRRC e DUC. Quest'ultimi, infatti, non influenzano le proprietà del sistema in termini di PER al variare del rapporto segnale rumore. I blocchi del modello da tenere in considerazione sono quelli di *FEC Encoding* che introducono la parità e quelli di *Mapping* e *PL Framing* che quantificano l'efficienza spettrale e l'efficienza energetica del sistema. La parte di *PL Framing*, tuttavia, influenza poco le prestazioni del sistema, che dipendono essenzialmente dai codici usati.

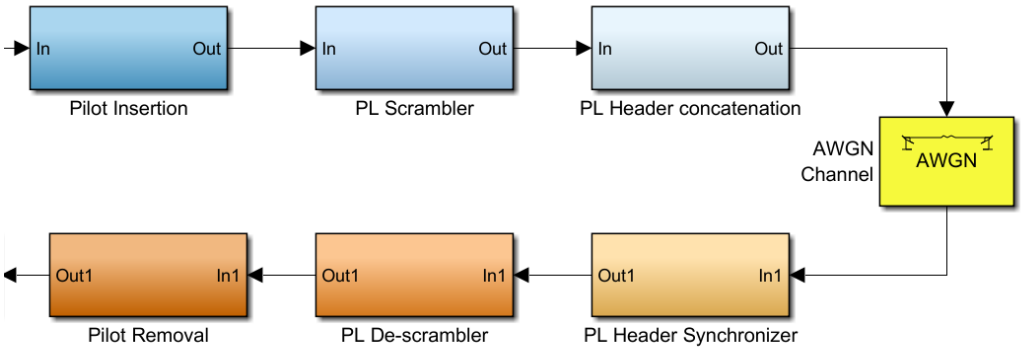

Figura 4.6: Simulazione della trasmissione in banda base su canale AWGN

L'immagine 4.6 raffigura gli elementi finali della catena di trasmissione, ossia un blocco che implementa un canale AWGN e i primi blocchi in ricezione che si occupano di ricostruire lo stream di bit trasmesso dalla seriale. La fase di ricezione è facilitata dalla totale sincronizzazione che una simulazione in banda base a livello software può offrire. In particolare, in figura, sono mostrati i blocchi che si occupano di rimuovere il PLHeader (in ricezione è nota la MODCOD a priori), di effettuare il de-scrambling e di rimuovere i toni pilota. L'inserimento della sezione di *PL Framing* in un modello che si occupa della valutazione della Packer Error Rate in funzione del rapporto segnale rumore, ha un impatto limitato in quanto l'inserzione dei simboli complessi ridondanti ha l'unico effetto di diminuire leggermente l'efficienza spettrale. La PER viene valutata, precisamente, in funzione del rapporto $E_s/N_0[dB]$. E_s indica l'energia del segnale in Joule, in particolare l'energia dei simboli complessi generati dalla costellazione. Lo standard DVB-S2 definisce delle costellazioni aventi energia media per simbolo pari a 1. N_0 rappresenta, invece, la densità spettrale di rumore. Le simulazioni sono state realizzate

su canali AWGN, attraverso un blocco nativo Simulink di cui settare, nella maschera, il valore di varianza di rumore legato al canale. Il valore di varianza (NoiseVariance) è legato al rapporto $E_s/N_0[dB]$ dalla seguente formula [11] :

$$\text{NoiseVariance} = \frac{\text{SignalPower} \cdot \text{SymbolPeriod}}{\text{SampleTime} \cdot 10^{\frac{E_s/N_0}{10}}}$$

In ingresso al blocco che simula il canale AWGN, i valori SymbolPeriod e SampleTime corrispondono; il valore SignalPower, ovvero la potenza dei simboli complessi è pari a 1 per definizione dello standard DVB-S2. La formuala necessaria al calcolo della varianza di rumore si semplifica, quindi, nel seguente modo:

$$\text{NoiseVariance} = \frac{1}{10^{\frac{E_s/N_0}{10}}}$$

Nella *callback function* che parametrizza il sistema è definito il valore di $E_s/N_0[dB]$ voluto e di conseguenza, in funzione dell'equazione precedente, la varianza di rumore con cui progettare il blocco Simulink.

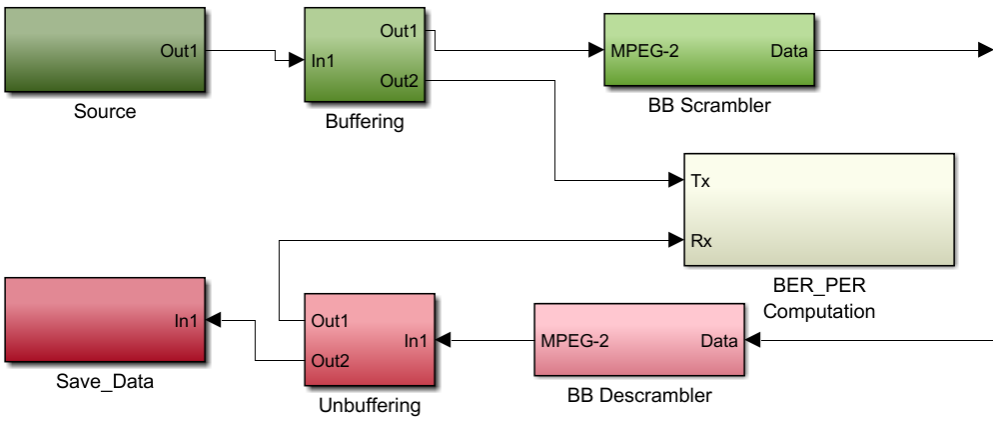

Figura 4.7: Ricezione dello stream binario e blocco che si occupa del calcolo della PER

Infine, in figura 4.7 è rappresentata la parte terminale della catena di ricezione, che prevede, l'operazione di de-scrambling della *BBFRAME* ricevuta e un'operazione di unbuffer per esportare lo stream binario in un sottosistema chiamato 'Save_Data'. Inoltre, in figura, è mostrato un subsystem chiamato *BER_PER_Computation*. Esso prende in ingresso i dati trasmessi prima del blocco di scrambling in trasmissione e i dati in ricezione dopo il de-scrambling. Entrambe le sequenze binarie vengono raggruppate in trame

di 1504 bit, ovvero la dimensione di una trama MPEG-2, e poi una *callback function* richiamata durante la simulazione si occupa del calcolo della Packet Error Rate.

4.2 Catena Sample-Based convertibile in codice VHDL

La realizzazione di questo modello Simulink rappresenta l'obbiettivo principale del mio lavoro di tesi e il punto di partenza per l'implementazione hardware del sistema di trasmissione. Il sistema che verrà descritto nel proseguo del capitolo rappresenta, infatti, il prototipo Simulink da convertire in VHDL per programmare l'FPGA. Le fasi successive del progetto, non descritte nell'elaborato, si occuperanno dell'ottimizzazione del codice VHDL, ritoccando ad esempio il modello Simulink stesso, in funzione delle specifiche fornite dal dispositivo hardware utilizzato. I criteri che hanno portato alla progettazione di questa catena Simulink si basano sui requisiti necessari alla compatibilità di un modello con l'*HDL Coder*. Per prima cosa, i blocchi Simulink utilizzati nel sistema devono far parte delle librerie ottimizzate alla conversione in linguaggio hardware. Questo vincolo esclude, ad esempio, l'intera sezione di *FEC Encoding* della catena e rende necessaria la progettazione dell'encoder BCH, dell'encoder LDPC e del General Interleaver. I blocchi nativi Simulink che sono stati utilizzati nella catena sono un numero limitato, ad esempio il modulatore QPSK, il generatore PRBS del *BB Scrambler* e il filtro SRRC.

Un altro vincolo stringente che caratterizza questa catena, al contrario di quella Frame-Based, è la necessità di utilizzare dei tipi di dato specifici per definire sia i parametri, che i segnali del modello. In particolare, risulta necessaria la rappresentazione dei numeri reali o complessi in virgola fissa, utilizzando un numero prefissato di bit. Il modello che è stato sviluppato non ha fatto uso del Fixed-Point Designer, il quale permette la conversione di un modello in virgola mobile in un equivalente modello in virgola fissa. I tipi di dato dei parametri e dei segnali della catena sono stati definiti passo dopo passo durante l'implementazione di ogni singolo blocco. In questo modo, si ha un controllo maggiore sul numero di bit impiegati per descrivere le varie grandezze. Come verrà descritto nel capitolo conclusivo della tesi, l'utilizzo di un numero limitato di bit di quantizzazione provoca un errore nella rappresentazione dei segnali. La scelta del tipo di dato in virgola fissa dipende, perciò, anche da questa problematica.

La differenza sostanziale tra i due modelli realizzati, da cui i file Simulink prendono il nome, è la modalità di campionamento dei segnali propagati tra i blocchi della catena, Frame-Based o Sample-Based. Il risolutore del precedente modello campiona i segnali con lo stesso tempo di campionamento ma campionando delle intere frame alla volta. I segnali in gioco risultano, perciò, dei veri e propri vettori colonna. In un modello HDL questo non è possibile, soprattutto a causa delle dimensioni notevoli delle frame. Il segnale viene campionato in maniera seriale, un elemento alla volta, che si tratti di bit o di simboli complessi. L'implementazione di codificatori a blocco come il BCH e l'LDPC, dell'interleaver, ma in generale di tutti i sottosistemi del modello si complica notevolmente. In ingresso al singolo blocco, non si ha più l'intero messaggio di informazione o l'intera codeword in forma di vettore, bensì un campione alla volta. Questo significa che la valutazione dei bit di parità di un encoder o la permutazione dei bit dell'interleaver non sono immediate ma avvengono ad avvenuto ingresso nel blocco del numero di bit necessario. Inoltre, ricevendo un campione alla volta, i blocchi non sono più in grado di riconoscere autonomamente l'inizio e la fine di una trama come nella catena Frame-Based. Il segnale dati deve essere, quindi, accompagnato da dei segnali logici di controllo sincronizzati ad esso, che specifichino la validità del dato e l'inizio e la fine di una trama.

Infine, un ulteriore limitazione nei modelli compatibili HDL sta nella configurazione del risolutore. Come descritto nel capitolo precedente, il risolutore deve essere obbligatoriamente discreto e di tipo Fixed-step, quindi con un passo di campionamento fisso che può essere deciso autonomamente dal software, in funzione dei segnali rilevati. Questi vincoli impongono una limitazione importante a livello software. Il modello può prevedere ancora la presenza di più frequenze di campionamento, tuttavia, la rate più elevata del sistema deve necessariamente essere un multiplo intero di ogni altra frequenza in gioco. Questo problema si pone nella realizzazione di sottosistemi che introducono della ridondanza, come gli encoder BCH e LDPC, ma anche i blocchi che si occupano dell'inserzione del PLHeader e dei toni pilota. Tutti questi sottosistemi, infatti, provocano un aumento non intero nella frequenza del sistema. Ad esempio, la code rate dell'encoder BCH nel caso QPSK e rate 1/2 è pari a 0.994074074, mentre l'inserzione del PLHeader, con la stessa configurazione MODCOD, provoca un aumento della symbol rate del 0.0023%. La risoluzione di problematiche di questo tipo a livello FPGA avviene grazie all'utilizzo di PLL (Phase-Locked Loop). Quest'ultimi, prendono in ingresso il segnale di clock di riferimento e generano a loro volta dei segnali di clock a frequenza inferiore mediante dei divisori interi ma anche frazionari. Ogni segnale di clock generato va a temporizzare il rispettivo blocco. Tuttavia, questo approccio non è possibile a livello Simulink volendo

convertire l'intera catena in VHDL, per i motivi precedentemente descritti. Nel seguente sottocapitolo è descritta la soluzione in ambiente Simulink che è stata pensata.

4.2.1 Control Manager

La soluzione alle problematiche introdotte precedentemente, in particolare la propagazione del segnale dati in modalità Sample-Based associato a dei segnali logici di controllo, e l'impossibilità nell'effettuare transizioni di rate frazionarie, è descritta di seguito.

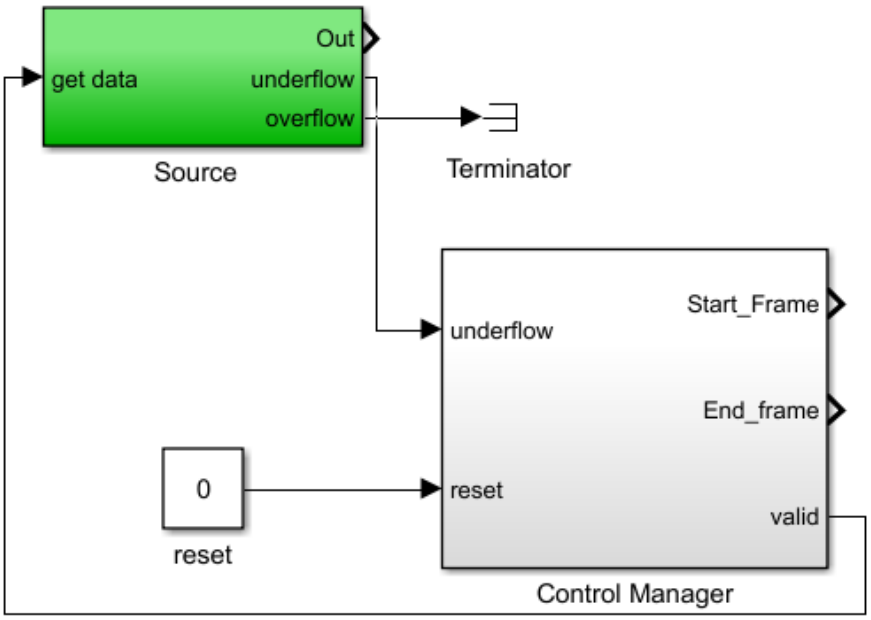

Figura 4.8: Trasmissione dei dati mediante sorgente seriale fittizia e Control Manager

L'approccio si basa sulla coesistenza di due blocchi iniziali della catena, mostrati in figura 4.8, ovvero:

- *Source*. Costituisce il sottosistema Simulink che implementa il buffer in uscita dalla seriale, dal quale estrapolare, mediante opportuno segnale di controllo, i bit dello stream da propagare nel sistema.

- *Control Manager*. Rappresenta il sistema mediante cui gestire il flusso di dati binari dalla seriale in maniera adeguata.

Questi due subsystem presenti nel modello, non fanno parte della catena Simulink da convertire in codice VHDL, ma servono a simulare il contesto nel quale ci si potrebbe trovare all'interno del FPGA. In particolare, il blocco denominato *Control Manager* verrà programmato su FPGA separatamente rispetto alla parte di codice derivata dalla conversione Simulink; mentre il blocco *Source* rappresenta la sorgente seriale esterna all'FPGA. Il blocco *Source* prevede un ingresso e tre uscite. L'ingresso è un segnale logico chiamato *valid* determinato dal *Control Manager* che indica se un nuovo bit presente nel buffer della seriale è pronto ad essere propagato nella catena DVB-S2. Le altre due uscite rappresentano i segnali logici di *underflow* e *overflow*, che indicano se il buffer risulta temporaneamente privo di dati binari o si trovi in saturazione. La definizione di queste due uscite è stata fatta per prevedere il più possibile le possibili condizioni di un ambiente hardware; tuttavia, durante le analisi sul corretto funzionamento della catena, questi segnali booleani sono stati lasciati costantemente a '0'.

Il *Control Manager* è il sottosistema che ha permesso di superare le problematiche legate alle variazioni di rate frazionarie non implementabili con l'*HDL Coder*. Grazie a questo blocco, questi cambi non interi di frequenza di campionamento, infatti, non si verificano più. Il *Control Manager* prevede al suo interno la presenza di un contatore. Questo contatore, conta da 1 fino ad un valore chiamato n_{frame}, dopo di che si reinizializza a 1 e ricomincia il conteggio fino a n_{frame}, continuando questa operazione in maniera ciclica. Il valore n_{frame} tiene conto di tutta la ridondanza aggiunta alla *BBFRAME* iniziale, in termini di bit di parità degli encoder e di simboli complessi inseriti durante la fase di *PL Framing*. L'idea è quella di prevedere ad inizio catena la successiva ridondanza introdotta nel sistema, la quale va ad aumentare la sample rate del sistema. Per fare questo, la *callback function* denominata *parameter_list* esegue il calcolo dei bit necessari da aggiungere alla *BBFRAME* iniziale per fare in modo che i vari blocchi della catena non necessitino di aumentare la rate del sistema per inserire la propria ridondanza. Il calcolo di n_{frame} è funzione delle code rate dei due encoder, dell'inserzione del PLHeader e dalla dimensione della *XFECFRAME* da cui dipende il numero di toni pilota inseriti. Inoltre, il valore numerico della frame complessivamente generata dal *Control Manager*, dipende dalla costellazione scelta in fase di *Mapping*. Ad un certo punto della catena, il segnale dati (e di conseguenza i segnali logici che lo accompagnano) dovranno essere decimati di un fattore pari al numero di bit per simbolo sulla costellazione. Questo non è un problema per quanto riguarda la *FECFRAME*; tuttavia, la ridondanza

associata al PLHeader e ai toni pilota (blocchi successivi al *Mapping*) da parte del *Control Manager*, sarà data dal numero di simboli complessi definiti dallo standard, moltiplicati per il numero di bit per simbolo definiti dalla modulazione digitale. Ad esempio, per una modulazione QPSK, il numero di bit associati dalla sorgente al PLHeader è pari a 90*2=180.

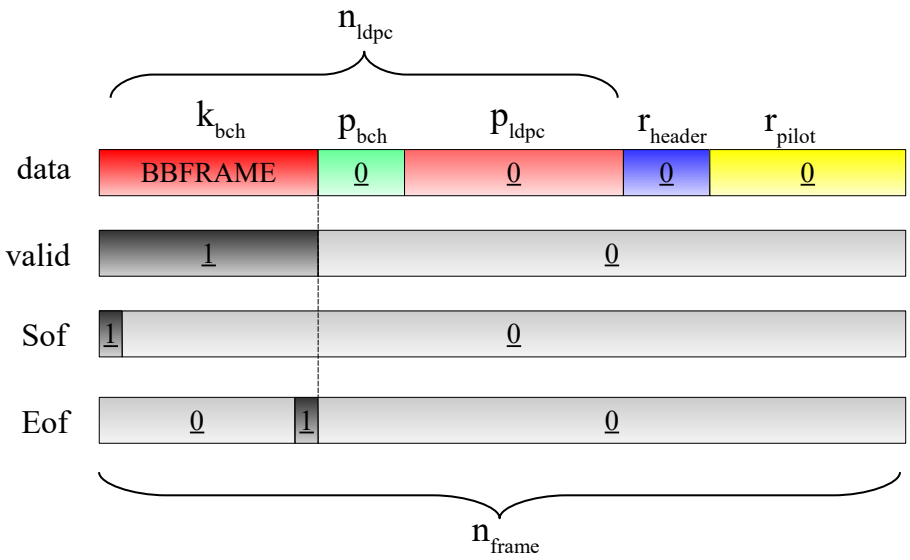

Figura 4.9: Formato della trama lunga n_{frame} relativa al segnale dati e i corrispondenti segnali di controllo, in uscita dal *Control Manager*

Il *Control Manager*, sfruttando il suo contatore e noto il valore di n_{frame}, fornisce alla seriale il segnale di *valid* a '1' per i primi k_{bch} colpi di clock; dopo di che, il segnale di *valid* torna basso e al posto dei dati vengono inseriti degli '0' fittizi che verranno sostituiti blocco per blocco dai bit di parità e dalle successive ridondanze. Inoltre, il *Control Manager* genera due ulteriori segnali logici oltre al *valid*:

- *Start of frame, Sof.* Indica il primo bit della *BBFRAME* e in tale caso vale '1'; altrimenti vale '0'.

- *End of frame, Eof.* Indica l'ultimo bit della *BBFRAME* e in tal caso vale '1'; altrimenti vale '0'.

In figura 4.9, è raffigurato l'insieme dei segnali in ingresso alla catena vera e propria. Il valore di n_{frame} è dato dal seguente calcolo:

$$n_{frame} = n_{ldpc} + r_{header} + r_{pilot},$$

con

$$r_{header} = 90 \cdot \log_2 M$$
$$r_{pilot} = P \cdot \log_2 M,$$

dove M rappresenta il numero di punti sulla costellazione e P il numero di toni pilota inseriti per *XFECFRAME*. Nel caso QPSK, il numero di bit rappresentativi di una frame iniziale in uscita dalla sorgente seriale è pari a:

$$n_{frame} = 64800 + 90 \cdot 2 + 36 \cdot 22 \cdot 2 = 66564$$

Oltre ai quattro segnali mostrati in figura 4.9, all'interno della catena avviene la propagazione del segnale di reset, posto anche in ingresso al *Control Manager*. Questo segnale, durante la fase di test sul corretto funzionamento della catena è posto costantemente a '0', e come nel caso dei segnali di *underflow* e *overflow* è stato aggiunto per rendere il modello ancora più hardware-friendly. Esso ha il compito di resettare l'intero sistema e di reinizializzare tutti i parametri della catena.

4.2.2 Caratteristiche della catena Sample-Based

Grazie al *Control Manager*, il modello Simulink è posto in una condizione ideale per poter realizzare tutti i blocchi della catena. Ogni sottosistema (a parte SRRC e DUC) prevederà cinque ingressi e conseguentemente cinque uscite, ovvero, il segnale dati, i segnali *valid*, *Start of frame*, *End of frame* e il segnale di reset (previsto nella catena ma posto sempre a '0' durante la progettazione del modello). In funzione dei 5 ingressi, in particolare di quelli logici di controllo, il sistema implementa il proprio algoritmo e fornisce in uscita la stessa tipologia di segnali di controllo modificati in funzione dell'operazione di cui si sta occupando. Ad esempio, la posizione del valore logico alto nel segnale *Start of frame* non subisce modifiche lungo la catena, in quanto il primo bit di una frame in ingresso all'encoder BCH corrisponde anche al primo bit in ingresso al LDPC, all'interleaver e così via. Il discorso cambia, invece, per quanto riguarda i segnali *valid* e *End of frame*. Blocchi come l'encoder BCH e l'encoder LDPC, che introducono della ridondanza, avranno un maggior numero di bit validi in uscita (l'aumento corrisponde al numero dei bit di parità). Inoltre, il segnale *End of frame* ha valore

logico alto non dopo k_{bch} bit, ma ad esempio dopo n_{bch} e infine dopo n_{ldpc} bit. Un comportamento analogo si ha per quanto riguarda i subsystem che introducono il PLHeader e i toni pilota.

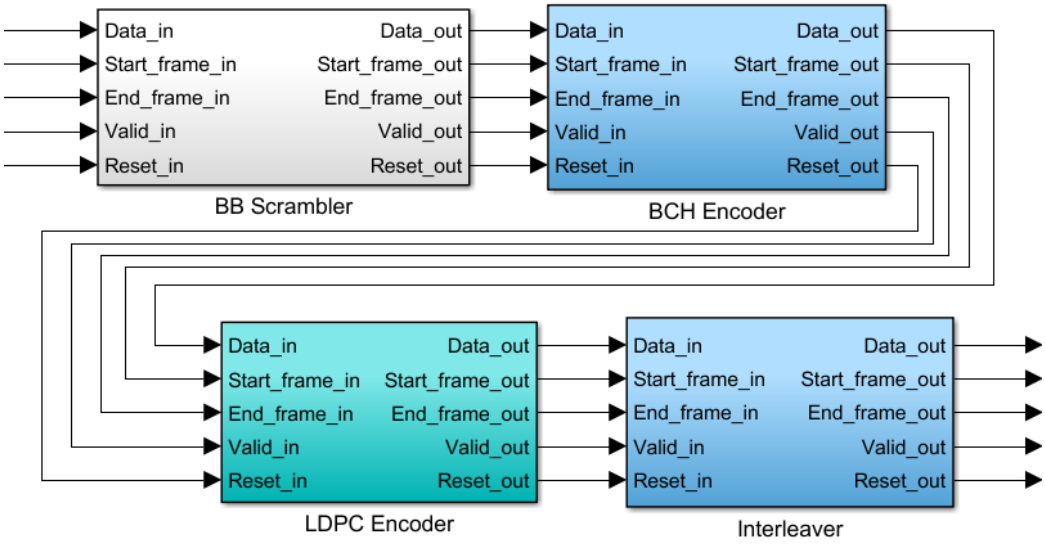

Figura 4.10: Stream Adaptation e FEC Encoding nella catena Sample-Based

I blocchi che si occupano di effettuare lo scrambling delle frame, ovvero il *BB Scrambler* e il *PL Scrambler*, e anche il General Interleaver, non introducono alcuna ridondanza e quindi non vanno a modificare la frequenza del segnale. I segnali logici di controllo non risultano, perciò, alterati all'uscita di questi sottosistemi.

In figura 4.10, sono posti sequenzialmente i primi quattro sottosistemi del modello, ovvero il *BB Scrambler* e i blocchi relativi al *FEC Encoding*. In figura 4.11, invece, è rappresentato il resto della catena. Come si può osservare, dall'ingresso del *BB Scrambler* fino all'ingresso del *PL Scrambler*, si ha la propagazione dei segnali logici di controllo (a parte *End of frame* che non è necessario al funzionamento del *PL Scrambler*). Questi segnali booleani vengono interrotti all'ingresso del SRRC, in quanto la gestione della frame generata dal *Control Manager* ed elaborata attraverso la catena, non è più necessaria.

I due blocchi che si occupano dello scrambling, sono realizzati attraverso degli shift registers come indicato nello standard [8]. In questi due subsystem, non solo non è introdotta ridondanza, non avviene nemmeno una elaborazione della frame in ingresso. I segnali logici di controllo vengono sfruttati,

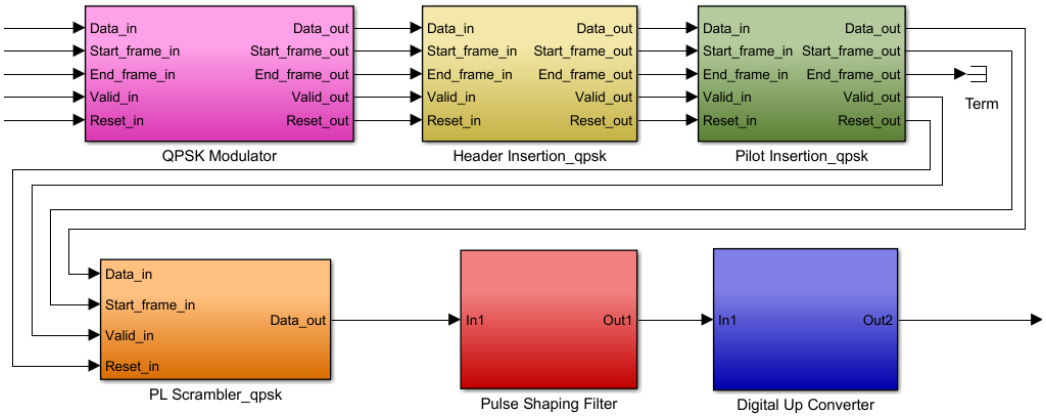

Figura 4.11: Mapping, PL Framing e Modulation nella catena Sample-Based, nel caso QPSK

quindi, per sincronizzare il primo bit (o simbolo) di una trama con il corrispondente bit (o simbolo) della sequenza generata dal registro a scorrimento. Inoltre, l'arrivo del segnale logico *Start of frame* a '1', indica al registro di reinizializzarsi, in modo da rigenerare la stessa sequenza precedente per la trama corrente. I registri a scorrimento sono realizzati mediante dei blocchi nativi Simulink, in particolare, il *BB Scrambler* è formato da un blocco Simulink nativo che implementa il generatore Pseudo Random. Gli shift register del *PL Scrambler* sono invece progettati manualmente con delle singole celle di memoria. Questo è possibile grazie al grado limitato dei polinomi generatori di questi registri. La stessa cosa non vale, ad esempio, per l'encoder BCH, il cui polinomio generatore può essere di grado 128, 160 o 192.

Approccio basato su MATLAB Function e macchina a stati

L'implementazione di tutti i sottosistemi che necessitano di una manipolazione della frame ricevuta e che inoltre dilatano la dimensione della frame in uscita, risulta più complessa. La modalità, comune a tutti questi elementi della catena, che è stata scelta per realizzarli, si basa su delle MATLAB Function. Il vantaggio nell'utilizzo di una funzione MATLAB, risiede nella possibilità di gestire sistemi molto complessi in maniera più compatta e con meno vincoli rispetto ad una struttura formata da blocchi Simulink. Inoltre, le librerie compatibili con l'*HDL Coder* sono in numero molto inferiore rispetto alla totalità delle librerie Simulink e quindi una funzione scritta manualmente incrementa le potenzialità del sistema. Gli algoritmi necessari

al calcolo della parità per BCH e LDPC, la permutazione dei bit nell'interleaver, l'inserimento del PLHeader e dei toni pilota e anche l'operazione necessaria alla concatenazione dei bit prima del *Mapping*, sono stati perciò realizzati mediante funzioni MATLAB. Queste funzioni prendono in ingresso tutti e cinque i segnali propagati lungo la catena, oltre a degli ingressi ricavati direttamente dal Model Explorer (solitamente dei parametri costanti). L'impiego di questo strumento è fattibile sfruttando l'utilizzo di variabili *persistent* nella funzione. Ad ogni time-step della simulazione Simulink, infatti, gli ingressi saranno successivi elementi delle frame e per tenere memoria dello stato del sistema al tempo di campionamento precedente, le variabili devono essere definite *persistent* così da rimanere memorizzate fino al time-step corrente. Ad esempio, elementi come contatori e registri sono sempre definiti nella MATLAB function come variabili persistenti. In questo modo, si ha la capacità di implementare tantissime funzionalità differenti. Molti vincoli che si hanno nel programmare una funzione MATLAB che sia convertibile in codice VHDL, sono gli stessi dei modelli Simulink. Ad esempio esiste un set limitato di funzioni elementari MATLAB utilizzabili, mentre altre non sono impiegabili (ad esempio le funzioni logaritmiche o goniometriche). Inoltre, i tipi di dato con cui definire le variabili della MATLAB function devono rispettare i requisiti HDL (virgola fissa, booleani, etc).

Nelle successive fasi del progetto SCAT, nelle quali si cercherà di ottimizzare il modello Simulink in funzione dell'implementazione vera e propria su FPGA, l'utilizzo delle MATLAB function potrebbe essere rivisto. Accanto agli innumerevoli vantaggi, esistono anche alcune problematiche che limitano l'ottimizzazione del modello. Ad esempio, si può pensare di sostituire la memorizzazione di alcuni dati negli shift register implementati con delle variabili *persistent*, con dei blocchi Simulink veri e propri che fungono da elementi di RAM. In questo modo, si utilizza più RAM del dispositivo ma allo stesso tempo si semplifica la complessità delle operazioni. Un altro limite che si può riscontrare utilizzando delle MATLAB Function particolarmente complesse, è la maggiore difficoltà nel risolvere le problematiche legate ai percorsi critici della catena con l'inserimento di pipeline registers. In questo caso, è possibile introdurre dei ritardi voluti nel sistema solamente in ingresso e uscita all'intera MATLAB Function. Realizzare il medesimo sistema con blocchi Simulink, permette di inserire i pipeline register in maniera più distribuita e quindi più ottimale.

L'impiego dei segnali logici di controllo propagati ad ogni blocco, permette di progettare delle vere e proprie macchine a stati all'interno delle MATLAB Function. Mediante la definizione di due variabili *persistent* che rappresentano lo stato corrente del sistema e lo stato successivo, è possibile gestire gli algoritmi di ogni sottosistema nella maniera più adeguata. In figu-

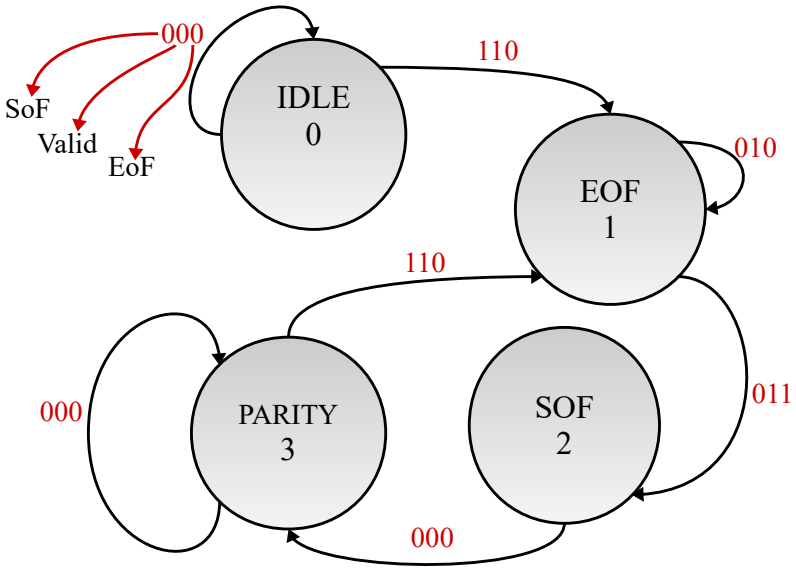

Figura 4.12: Macchina a stati semplificata utilizzata per gestire l'encoder BCH e l'encoder LDPC

ra 4.12, è mostrata la macchina a stati semplificata utilizzata per gli encoder BCH e LDPC. Gli stati possibili del sistema e i passaggi di stato, dipendono dalle varabili logiche di controllo. Ad esempio, l'arrivo di entrambi i segnali *valid* e *Start of frame* a '1' indica l'inizio di una nuova frame e il corrispondente passaggio di stato. La stessa cosa per l'arrivo del segnale *End of frame*. Tutte le MATLAB Function realizzate, prevedono al loro interno lo sviluppo di una macchina a stati.

In tabella 4.1 sono mostrati i valori dei segnali in uscita da uno dei due encoder (il principio è lo stesso sia per BCH che per LDPC) gestito da una macchina a stati come quella di figura 4.12. La tabella e la macchina a stati stessa, rappresentano una semplificazione del vero sistema implementato a livello software. Infatti non sono tenuti in considerazione i possibili cambi di stato dovuti ai segnali di reset o alla presenza di underflow nel buffer di sorgente e nemmeno la ridondanza aggiuntiva introdotta dal *Control Manager*. La macchina a stati raffigura il procedimento a regime del sistema, nel quale si passa da uno stato IDLE quando ancora non si hanno frame, ad uno stato chiamato SOF all'arrivo della *Start of frame*. La macchina resta nello

STATO IDLE 0	SoF_Out=0 Valid_Out=0 EoF_Out=0 Data_Out = Data_In
STATO SOF 1A	SoF_Out=1 Valid_Out=1 EoF_Out=0 Data_Out = Data_In
STATO SOF 1B	SoF_Out=0 Valid_Out=1 EoF_Out=0 Data_Out = Data_In
STATO EOF 2	SoF_Out=0 Valid_Out=1 EoF_Out=1 Data_Out = Data_In
STATO PARITY 3	SoF_Out=0 Valid_Out=1 EoF_Out=0 Data_Out = Parity_data

Tabella 4.1: Segnali in uscita da una MATLAB Function al variare degli stati della macchina a stati semplificata di figura 4.13

stato SOF fino all'arrivo del *End of frame*, dopo di che si passa nello stato EOF, per poi passare immediatamente nello stato PARITY in cui i dati in uscita non sono più i bit in ingresso, ma i bit di parità. Infine all'arrivo del successivo *Start of frame* si ritorna nello stato SOF.

Problema della concatenazione dei bit nella fase di Mapping

Una delle fasi più delicate nell'implementazione della catena, è stata quella relativa alla mappatura dei bit della *FECFRAME* nella costellazione scelta. Per fare questo, sono stati utilizzati blocchi nativi Simulink sia per la modulazione QPSK, che per la modulazione 8PSK. Il mapping nelle costellazioni 16APSK e 32APSK è stato realizzato manualmente mediante l'uso di look-up table. Questi blocchi originari di Simulink non prevedono in ingresso la sequenza binaria da mappare, bensì i valori interi, corrispondenti alla concatenazione del numero di bit utile a rappresentare un simbolo complesso. Ad esempio, nel caso QPSK, c'è bisogno di concatenare i bit della frame a 2 a 2 e valutare il corrispondente numero intero (formato da 2 bit) tra 0 e 3. Questa

operazione di concatenazione dei bit, seppur semplice concettualmente, si è dimostrata particolarmente problematica per i motivi descritti in seguito. In Simulink, la concatenazione di due bit può avvenire diramando il segnale dati in due vie, ritardando di un time-step la prima delle due. A questo punto, attraverso un blocco Simulink, è possibile effettuare la concatenazione dei due bit in arrivo dalle due vie, ad ogni time-step. In questo modo, però, per ogni coppia di bit concatenati correttamente, si ha la concatenazione di due bit in modo errato. Inoltre, al primo istante di campionamento, il MSB della *FECFRAME* non può essere concatenato con nessun bit e in questo modo si genera necessariamente una latenza di 1 time-step. Considerazioni analoghe, ma con differenti valori numerici in gioco, possono essere fatte per ogni tipo di modulazione digitale. A valle della concatenazione, si effettua una operazione di decimazione che cancella un campione del segnale ogni due. Tuttavia, il blocco decimatore deve sapere quale campione in arrivo salvare o cancellare. Se non fosse correttamente sincronizzato, verrebbero eliminati tutti i campioni corretti della trama. La risoluzione a questo problema si è avuta implementando una MATLAB Function apposita, il cui funzionamento è descritto in seguito.

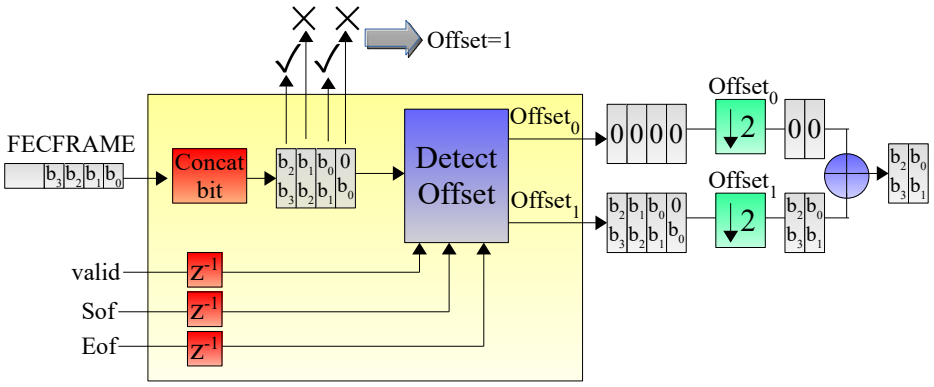

Figura 4.13: Schema a blocchi del funzionamento della MATLAB Function che si occupa della concatenazione dei bit e della decimazione del segnale nel caso QPSK

La MATLAB Function utilizza un contatore interno che è inizializzato a 0 all'arrivo del primo bit trasmesso dalla seriale. Al secondo istante di cam-

pionamento, il contatore viene incrementato a 1 per poi essere inizializzato nuovamente a 0 al terzo istante di campionamento. Ad ogni time-step, il bit in arrivo viene concatenato con quello precedentemente arrivato. La funzione prevede due uscite, una associata ad un offset pari a 0 (la concatenazione è corretta al primo time-step) e l'altra associata ad un offset pari ad 1 (la concatenazione è corretta al secondo time-step). Quando in ingresso alla funzione si ha il segnale di *Start of frame* ad '1', si va a vedere il valore del contatore e a seconda del suo valore una delle due uscite viene impostata a 0, mentre all'uscita 'corretta' viene associato il valore della concatenazione. I due segnali in uscita verranno poi decimati, uno senza introduzione di offset, l'altro con un offset pari a 1, ed infine sommati. La somma di un segnale utile e di un altro posto sempre a 0 fa si che in uscita si abbia il segnale utile. In figura 4.13 è rappresentato uno schema a blocchi che mostra concettualmente il funzionamento della MATLAB Function. I valori in uscita dal sommatore, che sono dei numeri interi senza segno, entrano poi in ingresso al modulatore digitale. In questo modo si riesce a risolvere senza errori la concatenazione dei bit e la transizione di rate per ogni tipo di modulazione digitale (utilizzando un numero superiore di uscite per le altre costellazioni previste dallo standard).

Implementazione del filtro SRRC e del DUC

Lo sviluppo degli ultimi due blocchi della catena Sample-Based, ha seguito un approccio differente. Sia il filtro interpolatore a radice di coseno rialzato che il Digital Up Converter, non necessitano di avere informazioni riguardo la validità dei dati e l'inizio e la fine della della *PLFRAME*, bensì effettuano le proprie operazioni su ogni campione ricevuto. La realizzazione dei due sistemi in Simulink ha riguardato principalmente le analisi effettuate ad inizio capitolo sulla scelta del fattore di interpolazione complessivo con cui interpolare il segnale a monte del SRRC e avente baud rate pari a 12.875 Msymb/s. Una volta definita la baud rate, sia l'SRRC che il DUC rimangono invariati qualsiasi sia la configurazione di MODCOD e la bit rate di sorgente. Come anticipato precedentemente, il fattore complessivo di interpolazione è pari a 28, distribuito nel seguente modo, 14 associato al SRRC e 2 al DUC. Entrambi i sottosistemi sono stati realizzati con dei blocchi nativi Simulink, in particolare, il filtro a radice di coseno rialzato non ha necessitato di alcuna modifica al proprio blocco, se non il settaggio dei parametri appropriati nella relativa maschera.

Nel filtro SRRC, che è implementato mediante un filtro FIR interpolatore polifase, oltre al fattore di interpolazione pari a 14, è stato settato il fattore di roll-off pari a 0.35, il numero di lobi della risposta impulsiva e il guadagno pari

a 1. Il numero dei lobi, il cui parametro è chiamato *FilterSpan in Symbols*, quantifica la complessità e la precisione del filtro. Il prodotto di questo parametro per il fattore di interpolazione quantifica il numero di prese del filtro, il numero di campioni con cui campionare la risposta impulsiva e il ritardo introdotto dal filtro. Idealmente, più il numero dei lobi è elevato, più la funzione di trasferimento del filtro è ottimale e il filtro risulta complesso da realizzare. Tuttavia, come verrà mostrato nel capitolo dei risultati, se i segnali e i parametri del filtro sono soggetti a quantizzazione (essendo definiti in virgola fissa), questa regola non è più garantita. Insieme ai parametri basilari della maschera, l'utente può settare anche il tipo di dato in virgola fissa con cui rappresentare i segnali e i parametri del filtro, in particolare:

- L'uscita complessiva del filtro
- I coefficienti dei tappi del filtro
- Il risultato dei moltiplicatori
- Il risultato dell'accumulatore.

Un approccio che spesso viene utilizzato, è quello di utilizzare un elevato numero di bit per rappresentare i dati interni al filtro, come i coefficienti, i moltiplicatori e l'accumulatore, ed effettuare infine un'operazione di cast sul segnale in uscita. La cosa più importante, infatti, è fare in modo che non si propaghino errori internamente al filtro durante il calcolo dell'uscita, della quale può essere poi diminuita la dinamica in maniera adeguata.

Le librerie Simulink mettono a disposizione il blocco chiamato Digital Up Converter. Tuttavia, questo blocco non è compatibile con l'*HDL Coder* e bisogna realizzare il sistema manualmente. Per fare questo, si sfrutta il System Object di MATLAB relativo al DUC. Fornendo i parametri di sistema corretti, in particolare la larghezza di banda dello spettro, la sample rate in ingresso, il fattore di interpolazione (in questo caso pari a 2) e la frequenza intermedia alla quale traslare lo spettro, è possibile generare in MATLAB il System Object che descrive il Digital Up Converter di interesse. Inoltre, è possibile visualizzare in che modo è realizzato questo blocco, ovvero i filtri interpolatori che ne fanno parte. I filtri FIR e il filtro CIC che lo implementano, sono realizzabili in Simulink mediante blocchi originali del software e compatibili con l'*HDL Coder*. Di questi filtri è possibile esportare i parametri, come i coefficienti, i fattori di interpolazione e il numero di sezioni del CIC e quindi settare questi valori nelle rispettive maschere. Come nel caso precedente, è fondamentale porre attenzione sul numero di cifre binarie con

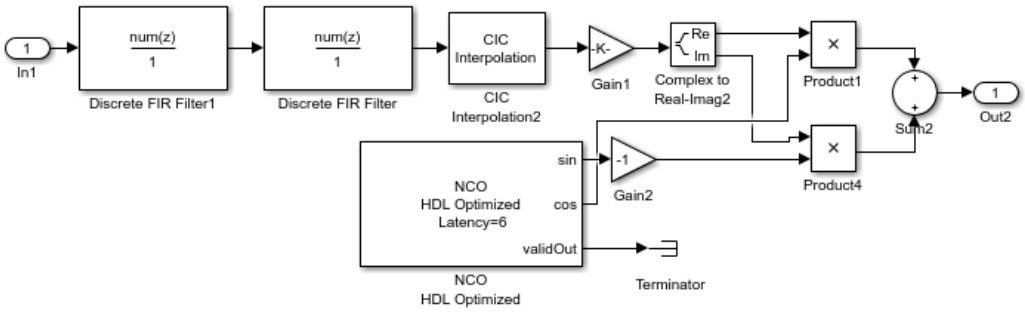

Figura 4.14: Schema in Simulink del Digital Up Converter

cui descrivere i segnali e i parametri di questi filtri. Un numero troppo basso di bit utilizzati può alterare la forma dello spettro del segnale.

In figura 4.14, è mostrato lo schema su Simulink che rappresenta il DUC, nel quale è possibile osservare i due filtri FIR interpolatori, un filtro CIC e il seguente attenuatore per normalizzare il guadagno introdotto dal CIC stesso ed infine la modulazione in quadratura. Per effettuare la modulazione in quadratura e traslare lo spettro a IF, viene utilizzato un NCO (Numerical Controlled Oscillator) ottimizzato per l'*HDL Coder*.

Capitolo 5

Risultati

L'obiettivo principale della tesi è stato quello di produrre un prototipo di modello Simulink da cui poter generare un codice VHDL. Oltre a progettare un modello che avesse le caratteristiche idonee a rispettare i requisiti del punto precedente, nel realizzare la catena di trasmissione, mi sono dovuto occupare, anche, di verificarne le corrette prestazioni rispetto alla letteratura relativa al sistema DVB-S2. In particolare, i risultati di questo capitolo riguardano due punti precisi, ovvero:

- Valutazione delle performance del sistema, in particolare riguardo alla capacità di rilevazione e correzione degli errori da parte dei codici.

- Analisi sull'impatto dell'errore di quantizzazione nel segnale trasmesso, in particolare valutando il rispetto della maschera relativa allo spettro di potenza, fornita dallo standard.

5.1 Prestazioni del sistema

Lo standard DVB-S2 permette di selezionare lo schema di modulazione digitale ed il tasso di codifica a seconda dei requisiti del servizio, delle caratteristiche del transceiver satellitare impiegato e delle condizioni del canale. In letteratura, ed in particolare nello standard ETSI [12], sono fornite valutazioni sulle prestazioni del sistema per differenti tipologie di canale, ad esempio per canali AWGN (Additive White Gaussian Noise) o per canali satellitari. Le considerazioni e i risultati mostrati nel proseguo del capitolo riguardano solamente simulazioni su canale AWGN. La scelta della configurazione di sistema (modulazione e codifica) è funzione di differenti parametri, come l'efficienza spettrale o il rapporto segnale rumore necessario a fornire deter-

minate prestazioni in termine di bit errati in ricezione (e conseguentemente pacchetti errati).

Mode	Efficienza Spettrale	Es/No (dB) ideale per una FECFRAME lunga=64800
QPSK 1/4	0.490243	-2.35
QPSK 1/3	0.656448	-1.24
QPSK 2/5	0.789412	-0.30
QPSK 1/2	0.988858	1.00
QPSK 3/5	1.188304	2.23
QPSK 2/3	1.322253	3.10
QPSK 3/4	1.487473	4.03
QPSK 4/5	1.587196	4.68
QPSK 5/6	1.654663	5.18
QPSK 8/9	1.766451	6.20
QPSK 9/10	1.788612	6.42
8PSK 3/5	1.779991	5.50
8PSK 2/3	1.980636	6.62
8PSK 3/4	2.228124	7.91
8PSK 5/6	2.478562	9.35
8PSK 8/9	2.646012	10.69
8PSK 9/10	2.679207	10.98
16APSK 2/3	2.637201	8.97
16APSK 3/4	2.966728	10.21
16APSK 4/5	3.165623	11.03
16APSK 5/6	3.300184	11.61
16APSK 8/9	3.523143	12.89
16APSK 9/10	3.567342	13.13
32APSK 3/4	3.703295	12.73
32APSK 4/5	3.951571	13.64
32APSK 5/6	4.119540	14.28
32APSK 8/9	4.397854	15.69
32APSK 9/10	4.453027	16.05

Tabella 5.1: Efficienze spettrali e valori di Es/No ideali per una $PER = 10^{-7}$ in un canale AWGN

Con il termine rapporto segnale rumore si possono intendere differenti valori, ad esempio il rapporto C/N (Carrier-to-Noise Ratio) che indica il rapporto segnale rumore a livello di segnali modulati (simboli complessi) e il E_b/N_0 che indica il rapporto in termini di bit ricevuti dopo la demodula-

zione. Infine il rapporto E_s/N_0, che è equivalente al C/N ma tiene conto dell'allargamento di banda dovuto al fattore di roll-off del filtro a radice di coseno rialzato. Questi rapporti sono legati tra loro dalle seguenti relazioni lineari:

$$\frac{C}{N} = \frac{E_b}{N_0} \cdot \frac{f_b}{B}$$

$$\frac{E_s}{N_0} = \frac{C}{N} \cdot \frac{B}{f_s}$$

dove E_b è l'energia per bit, E_s è l'energia per simbolo pari ad $E_b \cdot \log_2 M$, f_b è la data rate sul canale (bit rate netta), f_s è la symbol rate in campioni al secondo e B è la larghezza di banda del segnale. In particolare, la seconda relazione può essere semplificata nel seguente modo, tenendo conto del fattore di roll-off α, solitamente pari a 0.35:

$$\frac{E_s}{N_0} = \frac{C}{N} \cdot \frac{B}{f_s} = \frac{C}{N} \cdot \frac{f_s \cdot (1+\alpha)}{f_s} = \frac{C}{N} \cdot (1+\alpha)$$

In tabella 5.1 sono mostrati i valori di efficienza spettrale e di rapporto E_s/N_0 ideale per ogni tipo di combinazione MODCOD in condizioni QEF (Quasi-Error-Free), ovvero con una PER (Packet Error Rate) pari a 10^{-7}. Con PER si intende il rapporto tra pacchetti di dimensione 188 bytes (come ad esempio gli MPEG-2) ricevuti in maniera errata rispetto a quelli correttamente ricevuti dopo la rilevazione e la correzione degli errori di tipo FEC. La tabella mostra i risultati ricavati da simulazioni numeriche relative ad una *FECFRAME* di dimensione normale (64800), forniti nello standard DVB-S2 [8]. Le simulazioni si basano su un numero di iterazioni del decoder LDPC pari a 50, assenza di rumore di fase, ipotesi di perfetto recupero della portante e perfetta sincronizzazione di simbolo. L'efficienza spettrale (valutata con una symbol rate unitaria) è calcolata sempre per una *FECFRAME* normale e assenza di toni pilota (che la diminuirebbero leggermente). Osservando i valori in tabella si nota il compromesso tra l'efficienza spettrale e l'efficienza di potenza delle configurazioni. Le costellazioni QPSK e 8PSK, essendo modulazioni ad inviluppo costante, possono essere usate senza problemi su transponder da satellite non lineari, per le applicazioni broadcast. Le modalità 16APSK e 32APSK, al contrario, sono principalmente orientate ad applicazioni di tipo professionale; possono anche essere utilizzate per il broadcasting, ma richiedono la disponibilità di un rapporto segnale rumore più elevato al ricevitore e l'impiego di avanzati metodi di pre-distorsione nella

stazione di up-link per attenuare gli effetti di non linearità del transponder. Le costellazioni 16APSK e 32APSK, sebbene non permettano efficienze di potenza analoghe agli schemi ad inviluppo costante, offrono però maggiore capacità trasmissiva.

Considerazioni analoghe alle precedenti possono essere fatte, analizzando il grafico di figura 5.1, ricavato dallo standard ETSI [12]. In esso è mostrato l'andamento dell'efficienza spettrale R_u (considerando una symbol rate R_s unitaria) al variare del rapporto C/N. L'immagine mostra come, rispetto ai sistemi di prima generazione DVB-S e DVB-DSNG, a parità di rapporto segnale rumore si ha un incremento del 20-35 % della capacità trasmissiva (bit rate utile o efficienza spettrale); mentre per avere la stesse prestazioni in termini di efficienza spettrale, le configurazioni DVB-S2 necessitano, in media, di un C/N inferiore di 2-2.5 dB (miglior efficienza di potenza). Il grafico mostra anche come, l'utilizzo dei codici LDPC permetta di avere performance molto vicine al limite teorico imposto da Shannon (curve tratteggiate).

Le valutazioni sulle prestazioni del sistema sviluppato in Simulink sono

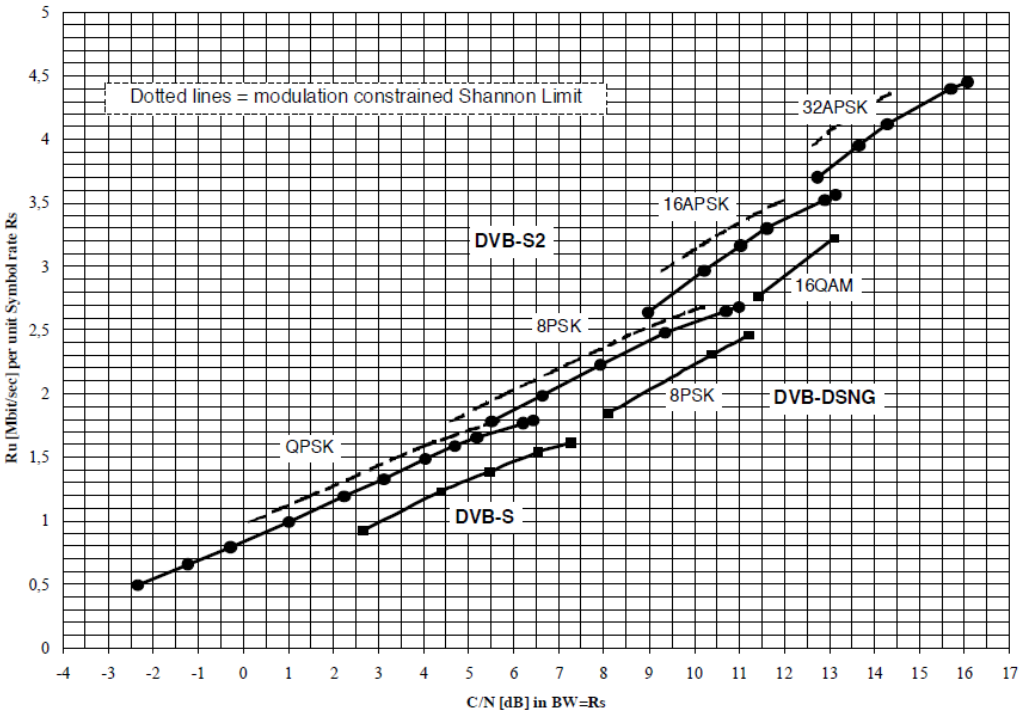

Figura 5.1: Grafico dell'efficienza spettrale in funzione del rapporto C/N (C/N si riferisce alla potenza media) richiesto, ottenuto attraverso simulazioni al calcolatore sul canale AWGN (demodulazione ideale).

state eseguite mediante simulazioni del modello Frame-Based nel quale non è stata inserita la sezione di *PL Framing*, per poter effettuare un confronto congruo con i riferimenti della letteratura [12]. In particolare, le simulazioni sono state realizzate al fine di valutare la PER sui pacchetti di dimensione 188 bytes al variare del rapporto E_s/N_0 su canale AWGN. Queste simulazioni sono state realizzate per ogni combinazione di modulazione e codifica possibile, secondo lo standard. I grafici di riferimento forniti dalla documentazione ufficiale ETSI [12] non includono tutte e 28 le possibili combinazioni descritte dallo standard stesso, tuttavia, sono stati lo strumento di confronto utile per valutare la bontà dei risultati simulati in Simulink.

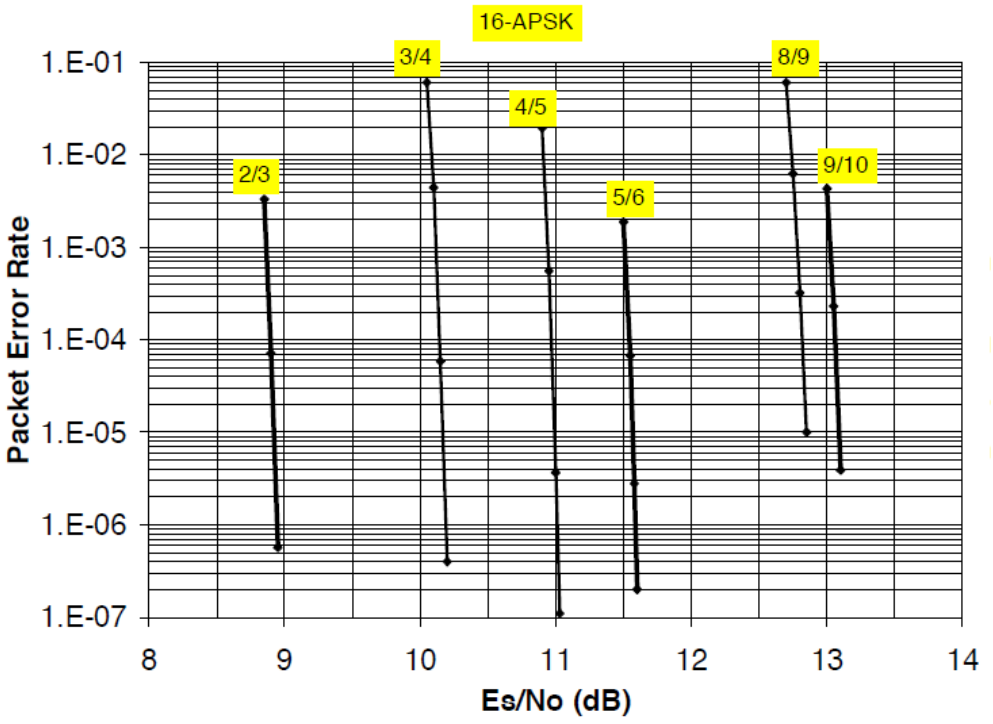

Figura 5.2: Curve prestazionali dei codici BCH+LDPC, con modulazione 16APSK, su un canale AWGN, fornite dallo standard ETSI.

Un esempio di grafico fornito dallo standard, è quello di figura 5.2. Esso rappresenta le curve della PER in funzione del rapporto E_s/N_0 nel caso 16APSK, per tutti i possibili valori di code rate dell'encoder LDPC. Dal grafico, si nota come aumentando la code rate del LDPC, e quindi aumentando l'efficienza spettrale R_u del sistema, si ha una diminuzione dell'efficienza

di potenza; ovvero per avere le stesse prestazioni in termini di PER, si ha bisogno di un rapporto E_s/N_0 maggiore.

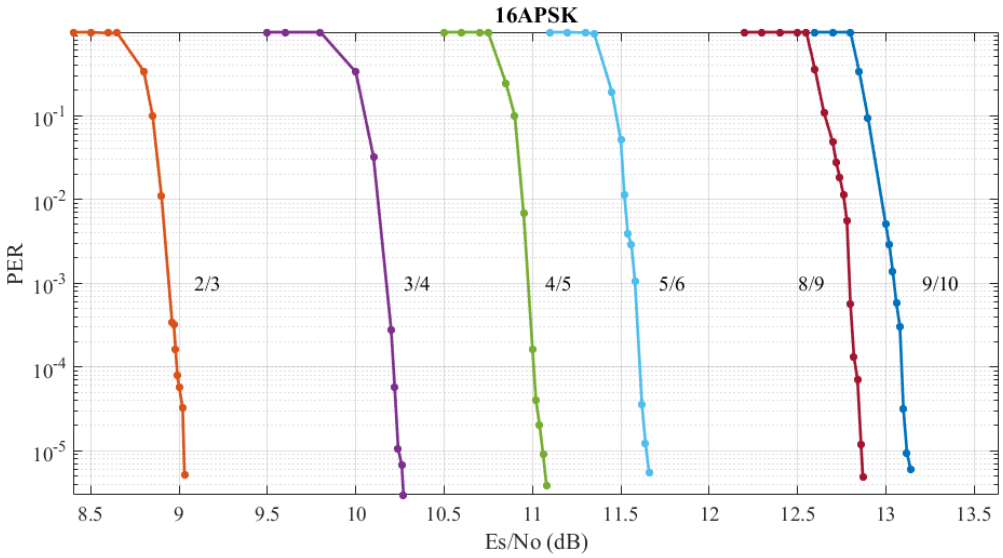

Figura 5.3: Curve prestazionali per la modulazione 16APSK, ricavate dalle simulazioni del modello Frame-Based

Il grafico di figura 5.2 è stato confrontato con le curve ricavate dalle simulazioni Simulink di figura 5.3, osservando le stesse prestazioni per ogni tipo di combinazione MODCOD. Di seguito sono riportate, nell'ordine, le prestazioni relative alla modulazione QPSK, 8PSK e 32APSK. Anche queste curve sono state confrontate con le performance dello standard [12], riscontrando risultati conformi alle aspettative.

In definitiva si può constatare che il modello Frame-Based, in particolare la parte relativa alla codifica di canale e alla modulazione digitale, è conforme in termini di prestazioni ai risultati forniti in letteratura. L'aver verificato a posteriori, attraverso la memorizzazione dei segnali generati sul workspace di MATLAB , che il modello Sample-Based convertito in codice VHDL, produca gli stessi simboli complessi del modello Frame-Based, certifica il corretto funzionamento del sistema che andrà a programmare l'FPGA.

A questo punto, la seconda verifica che è stata fatta e che è mostrata nella parte successiva del capitolo, riguarda i possibili errori di quantizzazione sui segnali generati, a causa dell'utilizzo di segnali e parametri in virgola fissa.

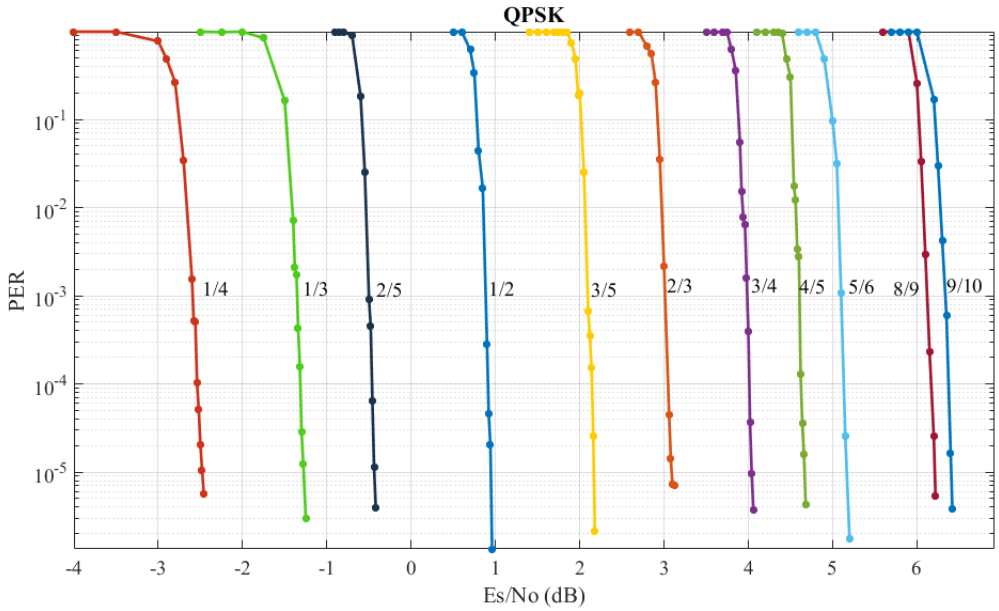

Figura 5.4: Curve prestazionali per la modulazione QPSK, ricavate dalle simulazioni del modello Frame-Based

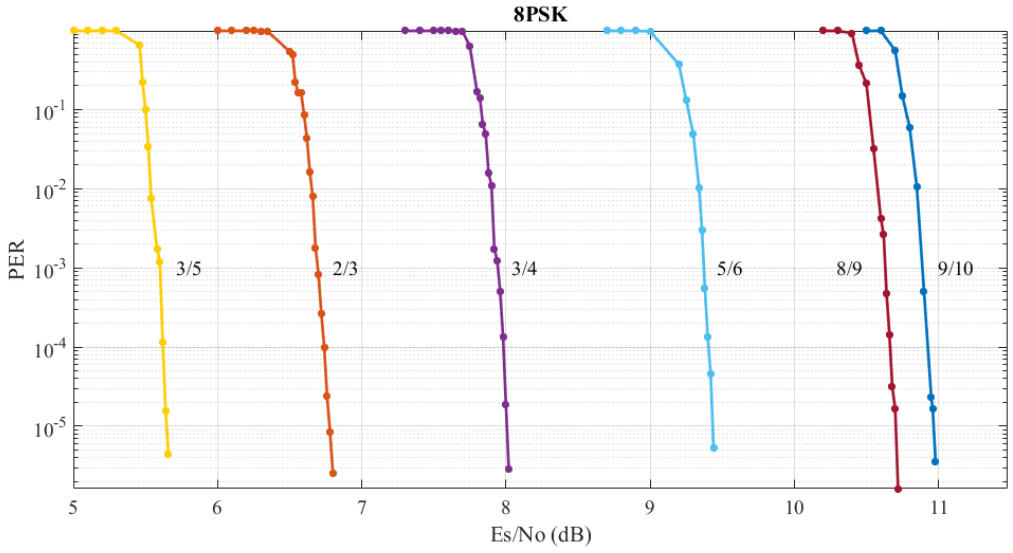

Figura 5.5: Curve prestazionali per la modulazione 8PSK, ricavate dalle simulazioni del modello Frame-Based

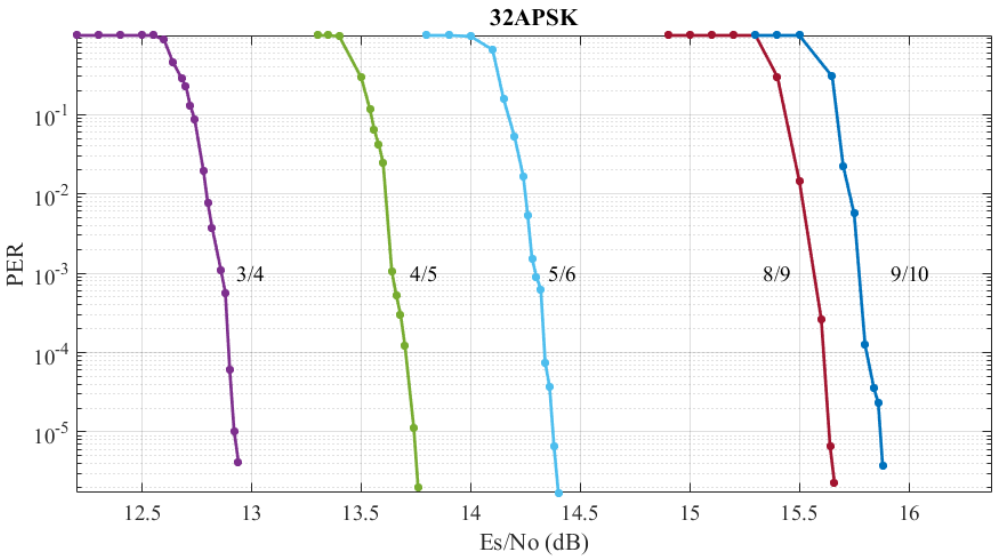

Figura 5.6: Curve prestazionali per la modulazione 32APSK, ricavate dalle simulazioni del modello Frame-Based

5.2 Effetti della quantizzazione sul segnale trasmesso

Il sistema di trasmissione implementato, oltre a dover fornire le performance desiderate in termini di Packet Error Rate, deve anche rispettare un'ulteriore condizione. Questo vincolo è rappresentato dal rispetto della maschera mostrata in figura 2.14 da parte dello spettro del segnale in banda base (e analogamente a frequenza intermedia). In condizioni ideali, l'utilizzo del fattore di roll-off del filtro a coseno rialzato, indicato dallo standard, permette il rispetto della maschera. Questo avviene anche grazie alle operazioni preliminari come il PL Scrambling e l'impiego di una costellazione $\pi/2$ BPSK per il PLHeader, che si occupano di evitare la possibile presenza di frequenze spurie. Tuttavia, la realizzazione di un modello Simulink in virgola fissa che sia compatibile con l'*HDL Coder*, fa si che tutti i segnali in gioco vengano quantizzati in funzione del numero di bit impiegati per rappresentarli. Nel terzo capitolo dell'elaborato è data una descrizione delle possibili problematiche legate all'utilizzo dei numeri in virgola fissa. Si possono avere fenomeni di overflow se il numero di bit usati per la parte intera è insufficiente, ed errori di troncamento quando il numero di bit è insufficiente per rappresentare la parte frazionaria del numero. Gli errori di saturazione non possono essere contemplati all'interno del proprio modello, perciò il numero di bit a sinistra

della virgola è sempre la minima quantità necessaria a descrivere un determinato valore. Lo stesso non si può dire per quanto riguarda la rappresentazione decimale del segnale o del parametro. A seconda del dispositivo hardware su cui verranno programmati i codici VHDL, si ha un determinato vincolo sul massimo numero di bit con cui rappresentare i propri segnali. Questo vincolo va ad impattare sulla rappresentazione frazionaria dei segnali, provocando necessariamente un errore di troncamento. All'interno del sistema di trasmissione DVB-S2, le problematiche legate alla quantizzazione vengono introdotte a partire dal mapping della *FECFRAME* sulla costellazione. Da questo punto della catena in poi si passa, infatti, dalla propagazione di segnali binari, alla propagazione di segnali complessi frazionari. La scelta del tipo di dato in virgola fissa con cui descrivere i simboli complessi della *XFECFRAME* e quindi della *PLFRAME* introduce un primo errore di quantizzazione. Tuttavia, i blocchi fino al PL Scrambler (compreso), non operano elaborazioni sui simboli, e quindi l'errore di troncamento non viene incrementato. Per quantificare l'impatto dell'errore di troncamento sui simboli complessi a partire dalla modulazione digitale, sono state realizzate tre catene parallele sostanzialmente identiche, se non per i valori WL (Word Length) e FL (Fraction Length) con cui sono descritti i simboli della costellazione. Questa sperimentazione è stata sviluppata solamente per una modulazione di tipo QPSK. Il motivo è dovuto al fatto che, analizzando i simboli complessi in gioco con le altre costellazioni, ci si aspetta risultati analoghi anche per 8PSK, 16APSK e 32APSK. Un'analisi di questo tipo è dovuta anche al fatto che non sono note a priori le caratteristiche tecniche dell'FPGA che verrà utilizzato. I tre casi sono i seguenti:

- WL=32, FL=30. Rappresenta un caso quasi ideale (soft), ma che utilizza un numero di bit compatibile con dispositivi hardware particolarmente sofisticati.

- WL=16, FL=14. Questo è il caso classico (lo stesso Simulink imposta di default 16 cifre binarie) e più soventemente utilizzato.

- WL=12, FL=10. Rappresenta un caso limite sviluppato per verificare se sia possibile diminuire il numero di bit (e quindi la complessità della struttura) rispetto al caso precedente, senza introdurre errori significativi.

L'impiego di un differente numero di bit per rappresentare la parte decimale, introduce già un piccolo errore nella rappresentazione dei simboli della costellazione. Si consideri, ad esempio, la parte reale di un simbolo complesso QPSK:

Valore Soft =0.707106781186
Valore con 30 bit =0.707106781192
Valore con 14 bit =0.707092285156
Valore con 10 bit =0.70703125

Come si può notare, utilizzando 30 bit si ha un arrotondamento, rispetto al valore ideale, a partire dalla undicesima cifra decimale, mentre diminuendo il numero di bit il troncamento avviene ben prima, ovvero alla quarta cifra decimale. Errori di questo tipo, sono comunque fisiologici e quasi irrilevanti a livello hardware. Tuttavia la propagazione di errori, anche poco considerevoli, nei blocchi successivi può alterare in maniera più evidente il segnale trasmesso. Un'analisi è stata fatta quindi per valutare gli effetti della quantizzazione sulle parti più delicate della catena, ossia il filtro SRRC e il DUC (in particolare i filtri che lo compongono).

5.2.1 Errore introdotto dal SRRC

Il filtro a radice di coseno rialzato, come descritto nei capitoli precedenti dell'elaborato, è implementato in Simulink mediante un filtro FIR interpolatore, il cui fattore di interpolazione, per sottostare alle specifiche relative al progetto SCAT, è stato fissato a 14. Essendo un filtro FIR, le parti che lo compongono sono: i coefficienti dei tappi, i moltiplicatori per ogni presa del filtro e l'accumulatore che fornisce l'uscita. Per ognuno di questi elementi, Simulink fornisce la possibilità di settarne i parametri WL e FL. L'approccio semplificato che è stato utilizzato inizialmente, è quello di rappresentare ogni elemento interno al filtro FIR con il numero di bit della catena corrispondente. Ad esempio, nella catena più performante, sia i coefficienti, che le uscite dei moltiplicatori, che l'uscita dell'accumulatore, che l'uscita complessiva del filtro, sono rappresentati con 32 bit totali, di cui 30 relativi alla parte decimale. Essendo fissato il fattore di interpolazione (numero di campioni per periodo di simbolo) e il coefficiente di roll-off, l'altro parametro del filtro che può essere variato e da cui dipendono la risposta impulsiva, la funzione di trasferimento e perciò anche i valori dei simboli in uscita, è il *Filter Span in symbols*.

In figura 5.7 è rappresentata la risposta impulsiva di un filtro SRRC avente fattore di interpolazione uguale a 14 e *Filter Span in Symbols* pari a 12. In questo caso, si vede come a destra e a sinistra del lobo principale, ve ne siano altri sei. Questo valore di *Filter Span* fa si, che la risposta impulsiva associata ad un determinato simbolo vada ad intersecare la risposta impulsiva dei sette precedenti e dei sette successivi simboli in ingresso al filtro. La

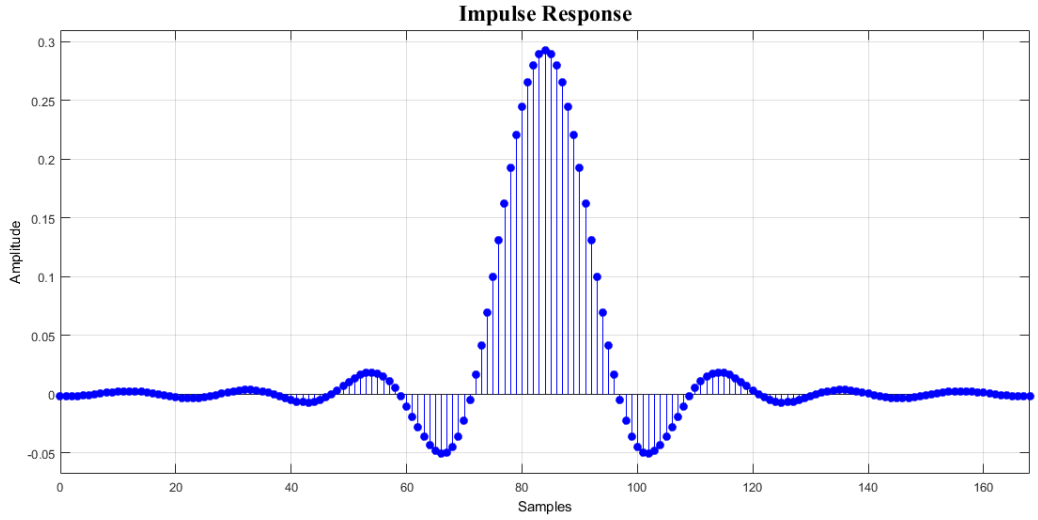

Figura 5.7: Esempio di risposta impulsiva di un filtro SRRC avente fattore di interpolazione pari a 14 e FilterSpan in symbols pari a 12

risposta impulsiva del filtro, tuttavia, soddisfa il criterio di Nyquist annullandosi ad ogni multiplo k del tempo di simbolo, tranne che per k=0. Questa condizione ideale permette di evitare l'interferenza intersimbolo (ISI) e di poter rappresentare la risposta impulsiva con un numero idealmente infinito di campioni, migliorandone la rappresentazione a livello digitale. Un sistema reale, tuttavia, deve tenere conto di alcune condizioni. In primo luogo, il filtro implementato deve avere un numero di prese finito (dato dal prodotto tra il fattore di interpolazione e il parametro *Filter Span*). In secondo luogo, l'utilizzare grandezze in virgola fissa, fa si che il valore dei campioni della risposta impulsiva venga approssimato. Non è più garantito, quindi, il soddisfacimento ideale del criterio di Nyquist, e in generale l'andamento teorico della risposta impulsiva. Per questo motivo, aumentare la lunghezza della risposta impulsiva con il parametro *Filter Span* può risultare sconveniente in caso di filtri soggetti ad errori di troncamento notevoli. Per valutare quale sia il valore ottimale di *Filter Span* da utilizzare, sono state realizzate simulazioni per ognuna delle tre catene a WL differente, impiegando cinque valori diversi del parametro: 4, 6, 8, 10 e 12. Per ogni simulazione, è stato plottato lo spettro del segnale in uscita dal filtro ed è stato calcolato il valore del MER (Modulation Error Ratio), utilizzando un filtro SRRC analogo in ricezione, al fine di ricostruire il segnale. Le simulazioni sono state effettuate in assenza di canale e con un filtro SRRC ideale in ricezione. In questo modo il MER risulta dipendente dal solo errore di quantizzazione del segnale trasmesso. La

figura di merito denominata MER, è calcolata come segue:

$$MER = 10 \cdot \log_{10} \frac{\sum_{n=1}^{N} (I^2 + Q^2)}{\sum_{n=1}^{N} (e_n)} \quad dB$$

dove, I rappresenta la parte reale del simbolo di riferimento trasmesso, Q rappresenta la parte immaginaria del simbolo di riferimento trasmesso, N è il numero dei campioni ricevuti presi in considerazione e e_n vale:

$$e_n = (I - I_n)^2 + (Q - Q_n)^2$$

con I_n e Q_n che indicano l'n-esimo simbolo ricevuto.

(WL,FL) / Filter Span	(12,10)	(16,14)	(32,30)
4	+28.5149	+28.6875	+28.6818
6	+41.8318	+53.9138	+54.1633
8	+37.8492	+41.8973	+41.8940
10	+37.4608	+46.4133	+46.5200
12	+36.5025	+57.6095	+61.6706

Tabella 5.2: Valori di MER (dB) relativi alla ricezione dei simboli nella costellazione QPSK, in funzione del parametro Filter Span in Symbols e in funzione del numero di bit di quantizzazione

In tabella 5.2, sono mostrati i valori di MER per i quindici casi esaminati, variando il numero di bit utilizzati per rappresentare i simboli in uscita dal filtro e il parametro *Filter Span*. Dalla tabella si nota come, aumentare il valore di *Filter Span* non implichi un automatico aumento del valore del MER e quindi un miglioramento delle prestazioni. Questo è dovuto alla problematica descritta in precedenza, ovvero l'effetto dell'errore di quantizzazione sulla rappresentazione della risposta impulsiva del filtro. Solitamente un MER accettabile in assenza di canale, quindi soggetto solo al rumore di quantizzazione, per sistemi di questo è tipo, deve essere vicino a +40 dB. In virtù di questa considerazione, i risultati mostrano come, qualsiasi sia il numero dei bit di quantizzazione, un valore di *Filter Span* pari a 4 aumenta troppo l'errore sul simbolo ricevuto. D'altro canto, aumentando a 6 il parametro si hanno prestazioni migliori in termini di MER, rispetto ai casi con *Filter Span* superiore e pari a 8 e 10. Se l'errore di quantizzazione è sensibile,

come nel caso a 12 bit, anche un valore di *Filter Span* pari a 12 peggiora la situazione. Questo non avviene se si utilizzano 16 e 32 bit, dove, allungando la risposta impulsiva fino a 12 lobi si migliora l'errore sul segnale ricevuto. Questa prima analisi dimostra come sia possibile utilizzare anche 12 bit di quantizzazione se il *Filter Span* del filtro è uguale a 6.

Dopo aver valutato l'errore sui simboli ricevuti, l'ulteriore analisi ha riguardato il rispetto della maschera imposta dallo standard DVB-S2, da parte dello spettro in uscita dal SRRC. Anche in questo caso sono state effettuate delle simulazioni relative ai quindici casi mostrati in tabella 5.2. Gli spettri sono stati confrontati con i valori della maschera di figura 2.14, per osservare eventuali sforamenti indesiderati, e il margine con cui la maschera è soddisfatta.

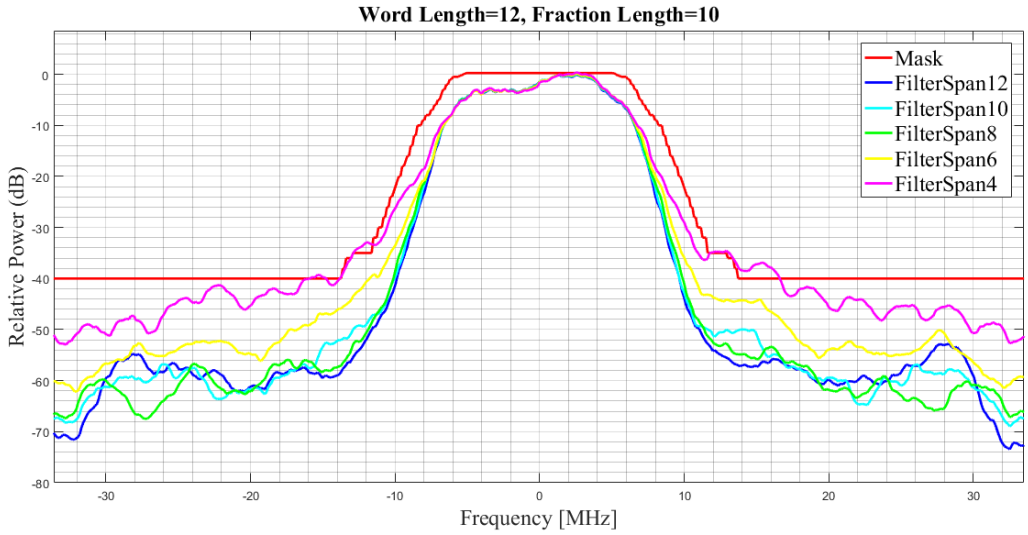

Figura 5.8: Spettro in banda base in uscita dal SRRC con 12 bit di quantizzazione, al variare del Filter Span

Osservando i grafici di figura 5.8, 5.9 e 5.10, si nota come qualsiasi sia il numero di bit utilizzato per descrivere lo spettro, la maschera è sforata solamente utilizzando un *Filter Span* uguale a 4. In tutti gli altri casi si ha il rispetto della maschera fornita dallo standard DVB-S2. In definitiva, si può perciò pensare di diminuire la complessità del filtro senza incappare in errori significativi, fornendo un segnale in uscita con WL=12, FL=10 e descrivendo il filtro con il *Filter Span* pari a 6. In questo modo il filtro interpolatore avrà complessivamente 85 prese, ovvero la metà del filtro mostrato in figura 5.7.

L'approccio utilizzato per valutare le figure di merito precedenti, non ha tenuto conto di una possibilità ulteriore, ovvero descrivere i parametri interni

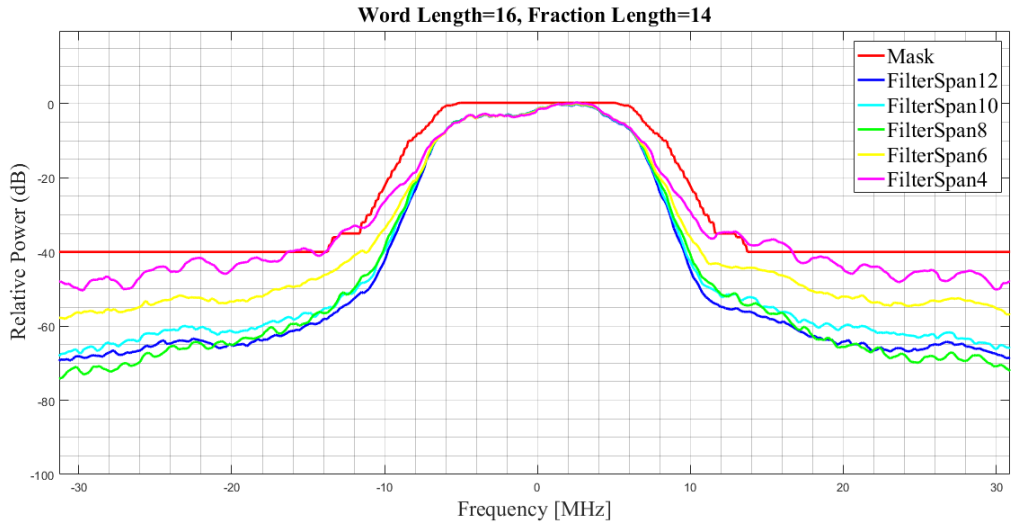

Figura 5.9: Spettro in banda base in uscita dal SRRC con 16 bit di quantizzazione, al variare del Filter Span

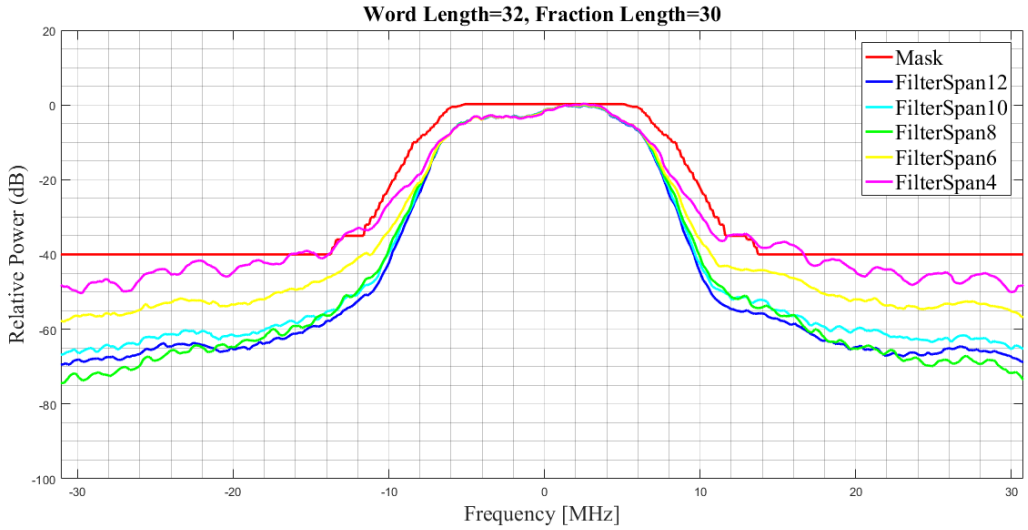

Figura 5.10: Spettro in banda base in uscita dal SRRC con 32 bit di quantizzazione, al variare del Filter Span

al filtro (coefficienti, moltiplicatori e accumulatore) con un numero di bit superiore rispetto all'uscita. Questo è sicuramente un criterio più corretto, in quanto, così facendo si riduce la propagazione degli errori internamente al filtro. L'uscita viene quindi calcolata con 16 bit, e del risultato ne viene fatto un cast a 12 bit, scartando i bit meno significativi. Si è dimostrato

che il filtro con uscita a 12 bit e *Filter Span* pari a 6, passa da un valore di MER di +41.8318 dB ad un valore di +51.6469 dB, solamente descrivendo le grandezze interne al filtro con 16 bit piuttosto che con 12.

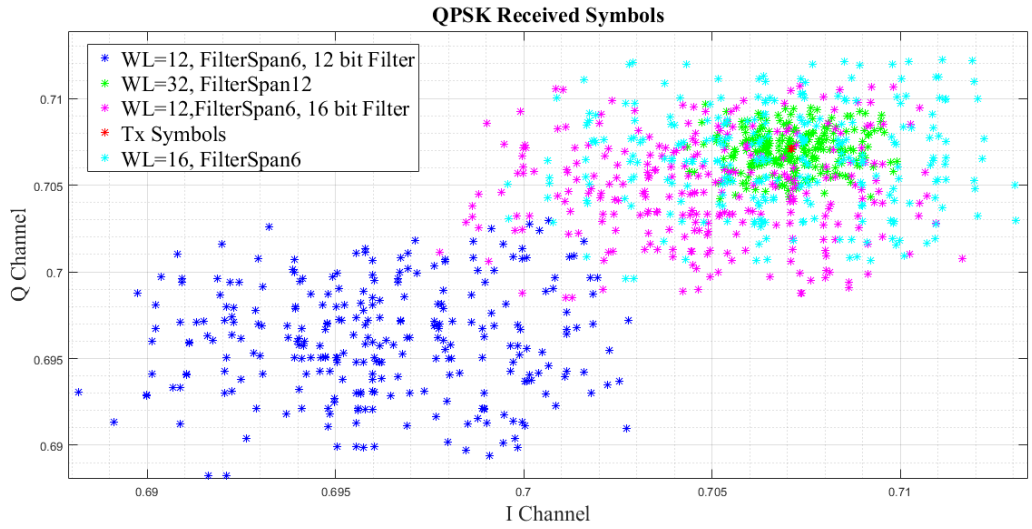

Figura 5.11: Errore nella ricezione di un simbolo QPSK dovuto alla quantizzazione, per quattro differenti casi

Il risultato precedente è confermato dalla figura 5.11. La figura mostra un punto (quello del primo quadrante) della costellazione QPSK in ricezione. In particolare, mette a confronto la posizione del simbolo trasmesso (punto rosso), con quattro differenti casi in ricezione. La nuvola verde rappresenta il caso migliore di tabella 5.2, ovvero 32 bit e *Filter Span*=12; La nuvola blu invece il caso a 12 bit e *Filter Span*=6, mentre quella magenta riproduce quest'ultimo caso con il miglioramento dei parametri del filtro a 16 bit. Il soddisfacimento della maschera, un valore di MER superiore a +50 dB e la propagazione di un segnale a soli 12 bit, rende quest'ultimo caso ottimale dal punto di vista del compromesso tra prestazioni e complessità. Le considerazioni fatte sul Digital Up Converter partono, perciò, dal presupposto che il filtro SRRC a monte sia stato implementato con queste specifiche.

5.2.2 Errore introdotto dal DUC e conversione del segnale a IF

Il Digital Up Converter realizzato in Simulink secondo le specifiche discusse nel capitolo 4, prevede una conversione ad una frequenza intermedia pari a 99.5 MHz e un fattore di interpolazione 2, in modo da avere una sample rate

di 360.5 Msps e la replica dello spettro alla effettiva IF richiesta da ESA di 460 MHz. Un fattore di interpolazione così limitato, fa si che dei due filtri FIR interpolatori che solitamente compongono il DUC, ne basti solo uno. Inoltre, l'effettiva interpolazione del segnale viene eseguita interamente dal filtro CIC. Il filtro FIR può essere realizzato con le stesse specifiche del precedente SRRC, lasciando l'uscita a 12 bit ed effettuando le operazioni interne a 16 bit. L'errore di quantizzazione sul segnale in uscita al primo filtro non viene, perciò, aumentato rispetto al segnale in ingresso. Lo stesso non vale per il filtro CIC. Il filtro CIC (Cascaded integrator-comb) in questione è un particolare tipo di filtro FIR formato da due parti distinte, separate da un blocco interpolatore intermedio. La prima parte è formata da filtri a pettine mentre la seconda da filtri integratori.

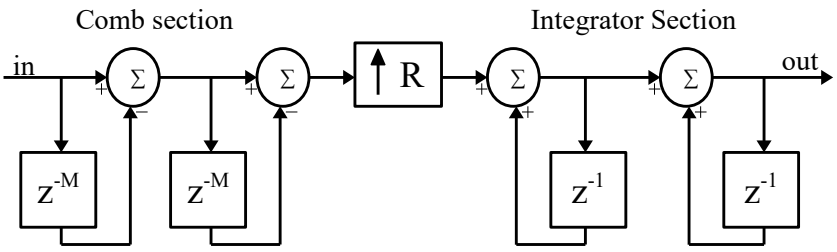

Figura 5.12: Schema logico di un filtro CIC interpolatore

In figura 5.12 è mostrato lo schema di un filtro CIC interpolatore avente complessivamente 4 sezioni. Il parametro M è detto ritardo differenziale e spesso, come nel modello Simulink realizzato, vale 1. R è invece il fattore di interpolazione. Il numero di stadi integratori e stadi pettine è equivalente e la loro somma quantifica il numero totale di sezioni del filtro. Il blocco Simulink permette di settare, oltre al numero di bit dell'uscita, anche il numero di bit con cui rappresentare le sezioni del filtro. Questa particolare tipologia di filtro FIR riduce notevolmente la complessità, diminuendo il numero di moltiplicatori e sommatori. Tuttavia, introduce un guadagno notevole in funzione del numero di sezioni. Per questo motivo, a valle del filtro è posto un blocco che normalizza l'ampiezza del segnale. Il guadagno molto elevato, fa si che la dinamica del segnale si allarghi molto. Per questo motivo non è più possibile propagare il segnale con soli 12 bit di quantizzazione, in quanto si avrebbero problemi di saturazione. In particolare, la scelta che è stata fatta, prevede l'utilizzo di WL=24 e FL=8 per descrivere le sezioni del filtro

e WL=16 e FL=2 per rappresentare l'uscita del CIC. Tuttavia, riportare il segnale a 12 bit, di cui 10 per la parte decimale, è possibile a valle del blocco di scaling. In questo modo l'uscita digitale del DUC e quindi del FPGA può essere effettivamente a 12 bit. Per valutare se il modello sia effettivamente conforme alle specifiche imposte dallo standard, si è andato a valutare lo spettro a valle del CIC, prima perciò della conversione a IF.

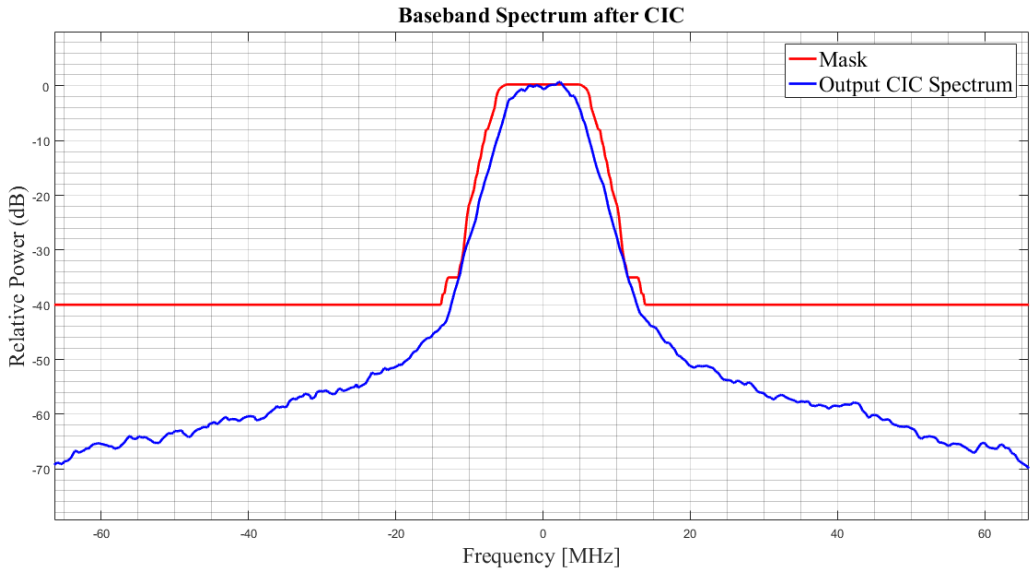

Figura 5.13: Spettro del segnale in banda base a valle dei filtri interpolatori del DUC

Il grafico di figura 5.13 mostra come lo spettro in banda base dopo l'interpolazione del CIC rispetti ancora la maschera fornita dallo standard [8]. Rispetto al segnale in uscita dal filtro a radice di coseno rialzato, tuttavia, lo spettro si avvicina notevolmente al limite consentito dalle normative, a causa dell'ulteriore errore di quantizzazione introdotto. Se l'FPGA su cui verrà realizzato il sistema sarà in grado di gestire segnali ad almeno 16 bit, l'aumento dei valori di WL e FL fin dal blocco SRRC permetterà, in ogni caso, di generare uno spettro più distante dalla maschera.

L'ultimo passaggio della catena consiste nella conversione a frequenza intermedia del segnale. Il modello Simulink si basa su un oscillatore locale libero in grado di generare campioni sinusoidali alla frequenza di campionamento del segnale da trasmettere. Il segnale in uscita al modulatore in quadratura, ovvero l'uscita del FPGA e l'ingresso del DAC interfacciato ad esso, è rappresentato sempre con 12 bit di quantizzazione.

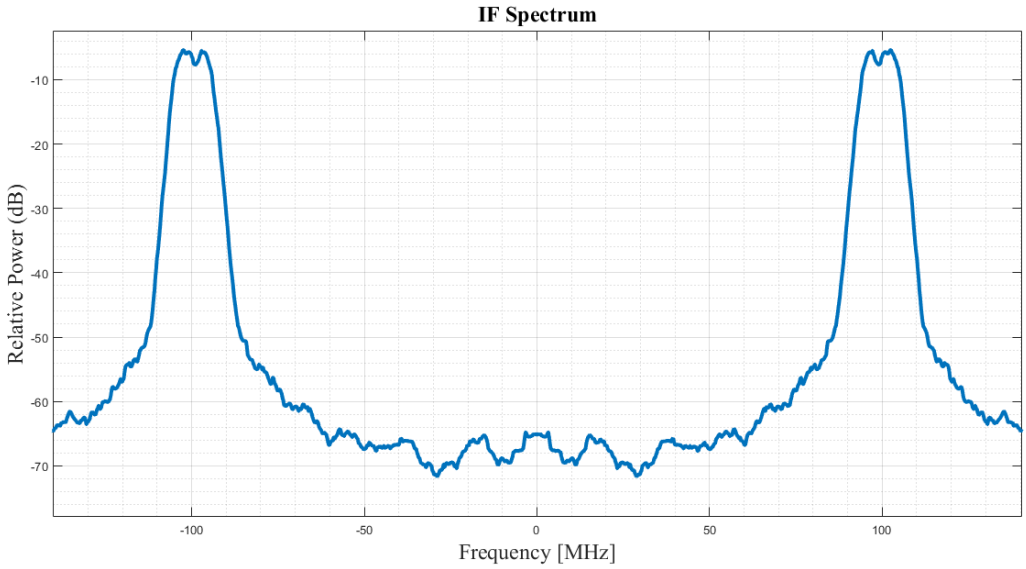

Figura 5.14: Spettro del segnale finale a frequenza intermedia

In figura 5.14, è mostrato lo spettro alla frequenza intermedia di 99.5 MHz. Grazie alla frequenza di campionamento di 360.5 Msps, ci si aspetta la replica dello spettro esattamente a 460 MHz.

Conclusioni

L'obiettivo di questa tesi è stato quello di sviluppare un modello Simulink relativo al sistema di trasmissione DVB-S2, che fosse convertibile in codice VHDL mediante il tool *HDL Coder*. Per arrivare al risultato definitivo della tesi, si è reso necessario uno studio preliminare dei possibili protocolli di comunicazione utilizzabili, optando in accordo con ESA, sull'utilizzo dello standard DVB-S2. In parallelo allo studio in letteratura degli algoritmi che definiscono lo standard DVB-S2, è stata presa confidenza con lo strumento *HDL Coder*. La programmazione in Simulink di un sistema compatibile con questo tool si differenzia notevolmente, infatti, dall'approccio utilizzato classicamente per la realizzazione di modelli Simulink impiegati per la sola simulazione e analisi di sistemi. Rispettando i requisiti imposti dal *HDL Coder* e attraverso uno strumento chiamato HDL Workflow Advisor, si è riusciti a generare il codice VHDL relativo alla catena di trasmissione DVB-S2 completa. Inoltre, grazie al modello di riferimento basato sulle librerie Simulink (in particolare quelle dell'encoder BCH e dell'encoder LDPC), si è verificato come il modello HDL fornisca gli stessi segnali rispetto a quello di riferimento. Questa verifica ha dimostrato come l'obiettivo principale della tesi sia stato raggiunto. Per poter affermare di aver realizzato un modello effettivamente valido, è stata fatta un'analisi sulle prestazioni del sistema, valutando per ogni configurazione di codifica e modulazione, l'andamento del PER (Packet Error Rate) al variare del rapporto E_s/N_0. Le curve tracciate si sono dimostrate conformi a quelle fornite in letteratura. Infine, l'ultima analisi ha riguardato lo studio degli spettri dei segnali trasmessi dopo la sagomatura a coseno rialzato, l'interpolazione e la conversione a frequenza intermedia. In particolare, si è visto come, anche fornendo in uscita al FPGA un segnale in virgola fissa descritto da soli 12 bit, si riesca a rispettare la maschera fornita dallo standard DVB-S2.

Il lavoro di tesi realizzato, rappresenta la base del progetto SCAT. In particolare, negli step successivi del progetto ci si occuperà di ottimizzare il codice VHDL, andando a modificare direttamente il modello Simulink laddove risulti necessario. Nello specifico, si cercherà di ottimizzare il sistema

in termini di utilizzo delle risorse (mediante l'utilizzo di più RAM), di occupazione di memoria (sviluppando il sistema DVB-S2 short) e di velocità (con l'utilizzo di pipeline register). Infine, la parte conclusiva del progetto riguarderà lo sviluppo della catena per la trasmissione del segnale di *Telemetria* e la ricezione del *Telecomando*, utilizzando molto probabilmente lo standard DVB-S di prima generazione.

Bibliografia

[1] *http://www.esa.int/Our_Activities/Telecommunications_Integrated_Applications/Satellite_frequency_bands*

[2] *Space data links - Telemetry synchronization and channel coding*,ref. ECSS-E.ST.50-01C.

[3] *Space engineering-Radio Frequency and Modulation*, ref. ECSS-E.ST.50-05C.

[4] *Bandwidth-Efficient Modulations: Summary of Definition, Implementation, and Performance*, ref. CCSDS 413.0-G-2.

[5] *Space Data Links-Telecommand Protocols Synchronization and Channel Coding*,ref. ECSS-E.ST.50-04C.

[6] *Digital Video Broadcasting (DVB); Framing structure, channel coding and modulation for 11/12 GHz satellite services*, ETSI EN 300 421 V1.1.2 (1997-08) .

[7] Alberto Morello, Vittoria Mignone: *Il sistema DVB-S2 di seconda generazione per la trasmissione via satellite e Unicast*, Rai Centro Ricerche e Innovazione Tecnologica Torino.

[8] *Digital Video Broadcasting (DVB); Second generation framing structure, channel coding and modulation systems for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; Part 1: DVB-S2*, ETSI EN 302 307-1 V1.4.1 (2014-11).

[9] Davide Biadene *Implementazione di un encoder LDPC DVB-T2 su piattaforma FPGA*, Università di Padova, Dipartimento di Ingegneria dell'Informazione.

[10] *https://lost-contact.mit.edu/afs/cs.stanford.edu/pkg/matlab-r2015b/matlab/r2015b/help/hdlcoder/ug/distributed-pipelining_btonpii.html*

[11] *https://it.mathworks.com/help/comm/ref/awgnchannel.html*

[12] *Digital Video Broadcasting (DVB); Implementation guidelines for the second generation system for Broadcasting, Interactive Services, News Gathering and other broadband satellite applications; Part 1: DVB-S2*, ETSI TR 102 376-1 V1.2.1 (2015-11).

Elenco delle figure

Elenco delle tabelle

Ringraziamenti

Ed eccomi al capitolo piú delicato della tesi, quello dei ringraziamenti. Un paio di righe vanno dedicate certamente a coloro che fin dal 2010 hanno fatto parte delle mie giornate in quel di Cesena e che ancora popolano questi luoghi: i vecchi compagni di maraffa, Fede e Sibo. Insieme a loro, per dovere di cronaca, tutti coloro che in questi anni mi hanno tenuto compagnia prima, durante e dopo le lezioni e il cui elenco é ahimé troppo lungo. Un saluto anche ai vecchi compagni della 5A, piú carichi che mai e un saluto pure ai Kakareri, sperando di non essere eliminato per qualche strana ragione. Un ringraziamento doveroso alla mia famiglia che mi ha assistito in questi anni e che talvolta ha pure cercato di capire qualcosa di quello che stavo studiando!! Una menzione speciale pure per Enrico Paolini e Gianni Pasolini, sempre disponibili per qualsiasi domanda o dubbio, e per Simone, senza il quale questi mesi sarebbero stati sicuramente meno divertenti in quel di Apice e senza i cui preziosi consigli avrei imparato molto meno da questa esperienza. Ma il ringraziamento piú importante é certamente quello per la mia ragazza Alessia. Nei momenti di stress, di fatica e di ansia é sempre riuscita a farmi stare meglio e a darmi la forza giusta per continuare il mio lavoro serenamente.