

## UNIVERSITÀ DI PISA

#### DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

Corso Di Laurea Magistrale In INGEGNERIA ELETTRONICA

TESI MAGISTRALE

#### Sviluppo del *firmware* della scheda di Memoria Associativa del processore Fast Tracker all'esperimento ATLAS del CERN

Candidato: Riccardo Cipriani

Relatori: Prof. Roberto Saletti

Dr.ssa Paola Giannetti

Dr. Simone Donati

Anno Accademico 2012-2013

## Indice

т		1	•	
In	tro	du	<b>Z1C</b>	ne

1	1 IL LARGE HADRON COLLIDER E IL RIVELATORE ATLAS				
	AL CERN 1				
	1.1	IL LA	RGE HAI	DRON COLLIDER	12
	1.2	IL RI	VELATOI	RE ATLAS	16
		1.2.1	I MAGN	ETI	20
		1.2.2	IL SIST	EMA DI TRACCIATURA	21
			1.2.2.1	IL RIVELATORE A <i>PIXEL</i>	23
			1.2.2.2	IL RIVELATORE A <i>MICROSTRIP</i>	24
			1.2.2.3	IL RIVELATORE A RADIAZIONE DI TRAN-	
				SIZIONE	24
		1.2.3	IL CAL	ORIMETRO	28
		1.2.4	IL RIVE	CLATORE PER MUONI	31
		1.2.5	IL SIST	EMA DI TRIGGER E DI ACQUISIZIONE	
			DATI		32

2	IL I	PROC.	ESSORE FAST TRACKER	37
	2.1	PRIN	CIPI DI FUNZIONAMENTO DI FTK	38
	2.2	ARCH	IITETTURA DI FTK	45
3	SUI	DDIVI	SIONE DI FTK IN 128 <i>PROCESSING UNIT</i>	53
	3.1	ARCH	IITETTURA E FUNZIONI DELLA SCHEDA AMB-FTK	54
		3.1.1	CONTROL - FPGA	57
		3.1.2	PIXEL - FPGA E SCT - FPGA	59
		3.1.3	ROADOUT-FPGA	62
		3.1.4	BUS PARALLELI E LINK SERIALI	63
		3.1.5	DISTRIBUZIONE DEL CLOCK	64
		3.1.6	GESTIONE SIMULTANEA DI DUE EVENTI	66
	3.2	A UXI	LIARY BOARD	67
4	IL	CHIP	DI MEMORIA ASSOCIATIVA E LA SCHEDA	
	$\mathbf{LA}$	MB-FI	ſK	69
	4.1	IL CH	IP DI MEMORIA ASSOCIATIVA	69
	4.2	ARCH	IITETTURA DELLA SCHEDA LAMB-FTK	74
		4.2.1	ALIMENTAZIONE DEL CHIP DI MEMORIA ASSO-	
			CIATIVA	75
		4.2.2	DISTRIBUZIONE DEI DATI AI CHIP DI MEMORIA	
			ASSOCIATIVA	76
		4.2.3	ESTRAZIONE DELLE ROAD DAI CHIP DI MEMO-	
				78

	4.3	BREV	E STOR	IA DEL CHIP DI MEMORIA ASSOCIATIVA	79
<b>5</b>	$\mathbf{L}\mathbf{A}$	FAMI	GLIA S	PARTAN-6 DELLA XILINX	82
	5.1	MAN	AGER DI	EL CLOCK	83
	5.2	PIN E	DI INPUT	P/OUTPUT	84
	5.3	BLOC	CCHI DI 1	MEMORIA	85
		5.3.1	MEMOI	RIA RAM	85
		5.3.2	MEMOI	RIA FIFO	86
	5.4	LOW	POWER	GIGABIT TRANSCEIVER	90
		5.4.1	GTP TI	RANSCEIVER	91
			5.4.1.1	TRASMITTER	95
			5.4.1.2	RECEIVER	98
6	٨R	СНІТІ	RTTIR	A LOCICA DEL CLUE-EPCA DELLA SC	HF_
U	DA LAMB-FTK 101				
	6.1 ELEMENTI LOGICI DEL GLUE - FPGA				108
		6.1.1	LA LOO	GICA DEL <i>MERGER</i>	115
			6.1.1.1	CONTROLLER	119
			6.1.1.2	LOGICA DELLA FINITE STATE MACHINI	E124
7	$\mathbf{AR}$	CHITI	ETTURA	A LOGICA DEL ROADOUT-FPGA DEL	-
	$\mathbf{L}\mathbf{A}$	SCHE	DA AM	B-FTK	130
	7.1	FUNZ	ZIONI SV	OLTE DAL ROADOUT-FPGA	130
	7.2	ELEN	IENTI LO	OGICI DEL ROADOUT - FPGA	132

		7.2.1	MONITOR DEL FLUSSO DEI DATI CON GLI $SPY$		
			BUFFER	134	
8	DES	SCRIZ	IONE DEI TEST DEL SISTEMA 1	.37	
	8 1	TEST	DEL GLUE-FPGA	140	
	0.1	ILDI		110	
	8.2	TEST	DEL ROADOUT-FPGA 1	142	
9	COI	NCLU	SIONI 1	45	
Bi	Bibliografia 167				

## Introduzione

Il Large Hadron Collider (LHC) al CERN di Ginevra è la macchina acceleratrice di particelle più grande e complessa mai costruita dall'uomo. Questo apparato è in grado di accelerare fasci di protoni lungo un anello di 27 km ad una velocità prossima a quella della luce, per poi farli scontrare alla frequenza di 20 milioni di collisioni al secondo.

Il rivelatore ATLAS è, a sua volta, un sistema sofisticato di enormi dimensioni, realizzato con il principale scopo di analizzare e studiare i prodotti delle collisioni protone-protone prodotte da LHC. In particolare, ATLAS consente di ricostruire le traiettorie delle particelle prodotte e di misurarne i parametri cinematici, quali impulso ed energia. Uno dei risultati più importanti ottenuti da ATLAS è stata la scoperta del Bosone di Higgs, avvenuta nell'anno 2012.

Poiché gli eventi interessanti per gli studi di Fisica sono assai rari e nascosti nella enorme quantità di eventi di fondo prodotti nell'elevatissimo numero di collisioni protone-protone che si verificano ogni secondo, non ha alcun senso cercare di registrare su memoria permanente tutti gli eventi prodotti. Si deve, invece, cercare di selezionare *online* gli eventi che possono essere interessanti, ricostruendo in tempo reale la maggior parte possibile delle informazioni provenienti dal rivelatore.

Presso la Sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare è in fase di progettazione e costruzione un nuovo sistema di ricostruzione online delle traiettorie delle particelle rivelate nel sistema di tracciatura di ATLAS, denominato Fast Tracker (FTK). Questo nuovo dispositivo sarà installato in tempo per la presa dati prevista per l'anno 2015.

Il funzionamento di FTK è complesso, ma può essere schematizzato in due fasi. Nella prima fase, detta *pattern recognition*, FTK organizza i dati ricevuti dal rivelatore in silicio e individua le combinazioni di punti colpiti nel rivelatore che hanno buona probabilità di essere dovute al passaggio di una particella. Nella seconda fase, FTK esegue un *fit* delle coordinate dei punti colpiti all'interno di ogni *pattern*, che consente di effettuare una ricostruzione tridimensionale delle traiettorie delle particelle che hanno attraversato il rivelatore e di determinarne i parametri cinematici. Il *pattern recognition* è la operazione più complessa ed è effettuata mediante un chip di memoria associativa appositamente realizzato. FTK è quindi un sistema estremamente complesso, capace di analizzare una enorme mole di dati in tempi estremamente ristretti. Per fare questo, è stato necessario programmare opportunamente numerosi dispositivi FPGA, ampiamente utilizzati in FTK.

Nella Tesi mi sono dedicato allo sviluppo ed alla messa in opera del firm-

ware di alcuni FPGA della famiglia Spartan-6 della casa produttrice Xilinx utilizzati sulla scheda di memoria associativa. Il mio contributo al progetto comprende anche l'intensa attività di test e di verifica del funzionamento del *firmware* sviluppato.

Nel seguito diamo una breve sintesi del contenuto della Tesi.

Il **Capitolo 1** descrive il Large Hadron Collider del CERN di Ginevra ed i rivelatori principali che costituiscono ATLAS. Nel descrivere l'apparato abbiamo dedicato maggiore spazio ai rivelatori in silicio che compongono il sistema di tracciatura ed al sistema di *trigger* e di acquisizione dati. Il lavoro descritto nella Tesi si colloca nel piano di miglioramenti del sistema di *trigger* per la presa dati programmata per l'anno 2015.

Il **Capitolo 2** illustra i principi di funzionamento e l'architettura del nuovo processore FTK. FTK è un progetto dell'Istituto Nazionale di Fisica Nucleare realizzato in collaborazione con numerose Istituti di ricerca internazionali, tra i quali The Enrico Fermi Institute della University of Chicago ed il CERN di Ginevra. La funzione di FTK è ricostruire in tempo reale le traiettorie delle particelle che attraversano il sistema di traccitura di ATLAS e fornire questa importante informazione al sistema di trigger per migliorare la selezione online degli eventi che si registrano su memoria permanente. In questo Capitolo viene descritta nelle linee essenziali l'architettura del processore che è costituito da 128 Processing Unit indipendenti e parallele.

Il **Capitolo 3** descrive l'architettura della singola *Processing Unit*. Essa è composta da due schede VME connesse per il *backplane*. La scheda principale è la scheda di memoria associativa ed effettua il *pattern recognition*, una ricostruzione preliminare ed a risoluzione ridotta delle traiettorie delle particelle. In pratica, la memoria associativa riconosce quali combinazioni di punti colpiti sul rivelatore hanno buona probabilità di essere dovute al passaggio di una stessa particella. Questa scheda è connessa ad una scheda ausiliaria, detta *Auxiliary Board*, che ha varie funzioni, tra le quali la principale è effettuare il *fit* a piena risoluzione delle traiettorie parzialmente riconosciute dalla memoria associativa e fornire questa informazione al sistema di *trigger*.

Il **Capitolo 4** descrive in dettaglio i componenti principali della scheda di memoria associativa, in particolare il *chip* di memoria associativa, un dispositivo *full custom* appositamente progettato per svolgere la funzione di *pattern recognition*. Viene descritta la *Little Associative Memory Board* (LAMB), che ospita i chip di memoria associativa ed i numerosi FPGA necessari per la distribuzione dei dati in ingresso ed in uscita ai chip. Il mio lavoro di Tesi è stato dedicato alla programmazione di alcuni FPGA utilizzati sulla scheda LAMB-FTK e sulla scheda AMB-FTK.

Il **Capitolo 5** discute brevemente la famiglia di dispositivi programmabili Spartan-6 della casa produttrice Xilinx che sono ampiamente utilizzati in FTK. In particolare, sono descritti i numerosi elementi funzionali interni agli Spartan-6, quali le memorie RAM o FIFO ed i *Low Power Gigabit Transceiver* per trasmissioni veloci dei dati.

Il **Capitolo 6** descrive in dettaglio la prima parte del lavoro che ho svolto. Ho sviluppato l'architettura logica di un FPGA della *Little Associative Me*- *mory Board* utilizzato per raccogliere le candidate tracce trovate dai chip di memoria associativa e trasmettere questa informazione alla scheda di memoria associativa principale. Si tratta di una logica complessa che ha richiesto vari mesi di lavoro.

Il **Capitolo 7** descrive la seconda parte del mio lavoro. Ho sviluppato la logica di un FPGA della scheda di memoria associativa principale utilizzato per ricevere le candidate tracce dalle quattro *Little Associative Memory Board* presenti sulla scheda e spedire la lista completa alla *Auxiliary Board* che esegue la ricostruzione tridimensionale delle traiettorie delle particelle utilizzata nelle selezioni di *trigger*.

Il **Capitolo 8** descrive i test della logica da me sviluppata. Il *firmware* ha superato tutte le prove di funzionamento che abbiamo effettuato in laboratorio.

Il **Capitolo 9** presenta le conclusioni del lavoro svolto nella Tesi e le prospettive per possibili futuri sviluppi del mio lavoro.

## Capitolo 1

# IL LARGE HADRON COLLIDER E IL RIVELATORE ATLAS AL CERN

In questo Capitolo descriveremo brevemente il Large Hadron Collider (LHC) ed il rivelatore A Thoroidal LHC Apparatus (ATLAS). Il mio lavoro di Tesi è parte dell'intensa attività di miglioramento del sistema di acquisizione dati del rivelatore attualmente in corso presso la Sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare ed il Dipartimento di Fisica dell'Università di Pisa.

#### 1.1 IL LARGE HADRON COLLIDER

LHC fa parte del complesso di acceleratori del CERN di Ginevra ed è uno dei più sofisticati strumenti di ricerca nel campo della fisica delle particelle elementari realizzati dall'uomo [1]. LHC è stato installato nel tunnel usato in precedenza dal Large Electron-Proton Collider (LEP). Il tunnel ha una circonferenza di 27 km e si trova ad una profondità che oscilla tra i 100 m ed i 150 m. LHC consente di accelerare due fasci di protoni in direzione opposta fino all'energia di 4 TeV per fascio, e di farli collidere nelle regioni di interazione mostrate in Figura 1.1, nelle quali sono installati i rivelatori ATLAS, CMS, LHCb, Alice, TOTEM e LHCf.



Figura 1.1: Veduta aerea del CERN di Ginevra. Sono stati messi in evidenza il tracciato del Large Hadron Collider, i punti in cui si intersecano e collidono i fasci e sono collocati gli esperimenti ATLAS, ALICE, CMS, LHCb, TOTEM ed LHCf.

Per mantenere i protoni nell'orbita all'energia prevista, si utilizzano 1232 dipoli magnetici superconduttori raffreddati da elio liquido superfluido alla temperatura di 1.9 K. Il campo magnetico generato localmente è di circa 8.4 T. I due fasci circolano in direzioni opposte, in tubi a vuoto separati e sono fatti collidere in quattro punti lungo l'orbita, in corrispondenza di caverne nelle quali il tunnel si allarga per lasciare spazio alle grandi sale sperimentali che ospitano i rivelatori (Figura 1.1 e 1.2).



Figura 1.2: Geometria dei fasci in corrispondenza di una regione di interazione.

I tubi nei quali circolano i fasci sono mantenuti alla pressione di vuoto di circa  $10^{-3}$  atm, in modo da ridurre al minimo le collisioni con le molecole di gas residuo. Ciascun fascio è costituito da circa 1400 pacchetti, detti *bunch*, ognuno dei quali contiene al momento dell'iniezione circa 1.15 x  $10^{11}$  protoni. Ogni *bunch* è lungo circa 7 cm, ha dimensioni trasverse dell'ordine di 1 mm che vengono ridotte a circa 25 µm in corrispondenza della regione di interazione. La distanza temporale tra due *bunch* consecutivi è 50 ns, ma si prevede di ridurla a 25 ns nella presa dati programmata per l'anno 2015 [2]. La collisione tra due *bunch* che circolano in direzioni opposte è detta *bunch crossing*. Dato l'elevato numero di protoni presenti all'interno di ogni *bunch*, in ogni *bunch crossing* si verificano in media alcune decine di collisioni protone-protone che producono complessivamente migliaia di particelle. La

Figura 1.3 mostra la ricostruzione delle traiettorie delle particelle prodotte in un *bunch crossing* e fa capire quale livello di complessità possa raggiungere l'analisi di questi dati.



Figura 1.3: Ricostruzione delle traiettorie delle particelle prodotte in un *bunch crossing* ad LHC.

L'accelerazione dei protoni avviene in più stadi (Figura 1.4). Il Linear Accelerator (LINAC) produce un fascio di protoni di energia pari a 50 MeV che il Proton Synchrotron (PS) e il Super Proton Synchrotron (SPS) accelerano fino a raggiungere, rispettivamente, l'energia di 26 GeV e 450 GeV. Successivamente, due fasci di protoni vengono estratti dall SPS e iniettati nell'anello acceleratore principale (LHC), dove ruotano in direzione opposta e raggiungono l'energia di 4 TeV, che corrisponde ad una energia nel centro di massa dei fasci di 8 TeV [3]. Si prevede che i futuri miglioramenti dell'acceleratore consentiranno di ottenere fasci di energia pari a 8 TeV e, di conseguenza, un'energia nel centro di massa di 16 TeV. Il CERN ospita 6 esperimenti principali, installati sull'anello di LHC. Essi si trovano ai quattro punti di interazione situati lungo l'anello: ATLAS [4] (A Toroidal LHC ApparatuS), CMS [5] (Compact Muon Solenoid), ALICE [6] (A Large Ion Collider Experiment), LHCb [7] (Large Hadron Collider beauty), LHCf [8] (Large Hadron Collider forward) e TOTEM [9] (TOTal Elastic and diffractive cross section Measurement). TOTEM è installato vicino al punto di interazione di CMS, mentre LHCf è posizionato accanto ad ATLAS.

**CERN** Accelerator Complex



Figura 1.4: Rappresentazione schematica di LHC e della catena completa di acceleratori. Sono inoltre indicate le posizioni di ATLAS, CMS, ALICE ed LHCb.

Nella Tabella 1.1 sono riportati alcuni parametri dei fasci utilizzati nella presa dati dell'anno 2012 e nella futura presa dati prevista per l'anno 2015.

PARAMETRI DEL FASCIO	2012	2015
Energia del fascio	4 TeV	8 TeV
Energia nel centro di massa	8 TeV	$16  { m TeV}$
Tempo tra due bunch crossing	50  ns	25  ns
Numero di <i>bunch</i>	1400	2800
Vita media del fascio	10 h	10 h

Tabella 1.1: Parametri del fascio di LHC nella presa dati dell'anno 2012 e dell'anno 2015.

### 1.2 IL RIVELATORE ATLAS

ATLAS [10] è installato in una grande caverna sotteranea a circa 150 m di profondità ed è un rivelatore di tipo *general purpose*, cioè è stato progettato per essere estremamente versatile e consentire di investigare il più ampio spettro possibile di processi di fisica. ATLAS ha una lunghezza di 46 metri, un'altezza di 25 m, pesa circa 7000 tonnellate, ed ha una forma approsimativamente cilindrica con una simmetria *forward-backward* rispetto alla regione di interazione (Figura 1.5).



Figura 1.5: Veduta schematica di ATLAS. Sono indicati i componenti principali del rivelatore.

ATLAS è costituito da numerosi rivelatori diversi e indipendenti e permette una copertura di quasi tutto l'angolo solido intorno alla regione di interazione. Procedendo dall'interno verso l'esterno, si incontrano, in successione, il sistema di tracciatura, il solenoide, i calorimetri e le camere per la rivelazione dei muoni.

Il sistema di tracciatura è utilizzato per ricostruire le traiettorie delle particelle cariche prodotte nelle collisioni protone-protone [11]. Per far questo, si combinano misure spaziali ad elevata risoluzione spaziale, effettuate tramite rivelatori in silicio posti in prossimità della regione in cui avvengono le collisioni, detta regione luminosa, con un gran numero di misure a risoluzione ridotta, effettuate mediante un rivelatore a radiazione di transizione, che occupa la regione più esterna. Il calorimetro è costituito da una sezione elettromagnetica ed una adronica. La sezione elettromagnetica identifica e misura con precisione energia e direzione di elettroni e fotoni, la sezione adronica completa l'informazione fornita dalla sezione elettromagnetica e consente di misurare energia e direzione dei getti di particella adroniche. I calorimetri misurano l'energia delle particelle prodotte assorbendole completamente, perciò la misura energetica è distruttiva. Solamente i neutrini ed i muoni sfuggono all'assorbimento nel calorimetro. L'impulso dei muoni viene misurato in uno spettrometro costituito da un grande toroide superconduttore che avvolge tutto il calorimetro. Combinando le informazioni raccolte dai diversi rivelatori è possibile stabilire con buona approssimazione l'identità delle particelle, misurarne l'energia e la quantità di moto.

ATLAS utilizza un sistema di riferimento destrorso, con asse x che punta al centro del tunnel di LHC, asse y che punta verso l'alto, mentre la direzione positiva dell'asse z coincide con la direzione del fascio che circola nel tunnel in direzione antioraria (Figura 1.6).

La pseudorapidità di una particella è definita come  $\eta = -\ln(\frac{\theta}{2})$ , con  $\theta$ angolo polare, ed è comunemente usata per misurare l'angolo relativo tra la direzione di moto della particella e l'asse del fascio. La pseudorapidità è un parametro importante e consente di effettuare una suddivisione convenzionale del rivelatore in quattro regioni: regione del *barrel* o centrale, *transition region* o regione di transizione, regione di *end-cap* e *forward* (Figura 1.7).



Figura 1.6: ATLAS utilizza un sistema di coordinate destrorso. L'asse x punta al centro del tunnel di LHC, l'asse y punta verso l'alto, l'asse z indica la direzione del fascio. Il verso positivo dell'asse z è definito dal fascio che circola nel tunnel in senso antiorario.



Figura 1.7: Suddivisione convenzionale del rivelatore in quattro regioni, *barrel, transition, end-cap* e *forward*, in funzione della pseudorapidità.

Riportiamo nel seguito una breve descrizione degli elementi principali che costituiscono il rivelatore ATLAS.

#### 1.2.1 I MAGNETI

ATLAS dispone di due sistemi magnetici principali. Il sistema di tracciatura è immerso in un campo magnetico di intensità pari a 2 T, allineato con il fascio e generato da un solenoide superconduttore (Figura 1.8).

I campi magnetici utilizzati nello spettrometro per muoni nella regione del *barrel* sono generati invece da magneti toroidali, costituiti da bobine superconduttrici che generano un campo magnetico dal valore massimo di 4 T. Nella regione di *end-cap* il campo magnetico è generato da un secondo sistema di magneti toroidali. Le bobine del *barrel* sono lunghe 25 m e alte 4.5 m; quelle della regione di *end-cap* hanno una lunghezza di 5 m ed un'altezza di 4.5 m.



Figura 1.8: A sinistra, geometria degli avvolgimenti dei magneti: sono visibili il solenoide (in viola), le 8 bobine del magnete toroidale del *barrel* alternate alle bobine della regione di *end-cap* (in rosso). A destra, veduta dei toroidi superconduttori della regione del barrel. La scala dimensionale è indicata dalla persona fotografata tra le due bobine in basso.

#### 1.2.2 IL SISTEMA DI TRACCIATURA

Il sistema di tracciatura è il rivelatore più interno di ATLAS (Figura 1.9) ed è costituito da una struttura cilindrica a strati di circa 2,3 m di diametro e 7 m di lunghezza totale.



Figura 1.9: Il sistema di tracciatura di ATLAS. Sono indicati il rivelatore a *pixel*, a *microstrip* centrale e in avanti ed il tracciatore a radiazione di transizione, nella regione del *barrel* e nella regione di *end-cap*.

Ogni strato, o *layer*, del rivelatore è formato da matrici di sensori in grado di segnalare il passaggio di una particella mediante un segnale elettrico.



Figura 1.10: Sezione longitudinale del sistema di tracciatura. Sono indicati il rivelatore in silicio a *pixel* e ad *microstrip*, ed il rivelatore a radiazione di transizione.

Il sistema di tracciatura è composto da tre rivelatori che si distinguono per il tipo di sensori utilizzati, per la loro geometria e per la distanza dalla regione di interazione. Procedendo dall'interno verso l'esterno si incontrano il rivelatore a *pixel* in silicio, il rivelatore a *microstrip* di silicio (SCT), e il *Transition Radiation Tracker* (TRT). Il rivelatore è diviso longitudinalmente in tre parti, come si vede nell'immagine in sezione riportata in Figura 1.10. La parte centrale, o *barrel*, è costituita dai *layer* concentrici ed a geometria cilindrica. Nella regione di e*nd-cap* la struttura del rivelatore è a dischi paralleli posti nel piano ortogonale al fascio.

#### 1.2.2.1 IL RIVELATORE A PIXEL

Il rivelatore a *pixel* in silicio è stato progettato per fornire un'elevata granularità (circa 100 milioni di *pixel*) e delle misure di posizione molto accurate nella zona più vicina alla regione di interazione. La Figura 1.11 ne mostra la struttura. Il rivelatore nella regione del *barrel* è costituito da 3 strati cilindrici concentrici centrati sull'asse del fascio e posti ad una distanza di 5, 10, 12 cm dal fascio, mentre il rivelatore nella regione di *end-cap* è costituito da 4 dischi per lato, posizionati nel piano perpendicolare alla direzione del fascio.



Figura 1.11: Rappresentazione schematica del rivelatore a *pixel* di silicio. Si notano gli strati concentrici della regione del *barrel* ed i dischi della regione di *end-cap*.

La risoluzione ottenuta sulla misura della posizione del punto nel quale le particelle attraversano i piani di *pixel* è di circa 10  $\mu m$  nel piano r- $\varphi$  e circa 116  $\mu m$  nel piano r-z.

#### 1.2.2.2 IL RIVELATORE A MICROSTRIP

La struttura del rivelatore a *microstrip* è molto simile a quella del rivelatore a *pixel*, con la differenza che in questo caso si utilizzano sensori a *microstrip* di silicio. La parte centrale è formata da 4 strati concentrici di *microstrip* di silicio, mentre la regione di *end-cap*, è formata da 18 dischi in totale, 9 dischi per lato. Nel *barrel* ognuno dei 4 strati è costituito da moduli di strip di silicio di 4 rivelatori aventi ciascuno una superficie di 63x63 mm<sup>2</sup>. La risoluzione sulla misura della posizione di un *hit* nel *barrel* è di circa 16  $\mu m$ nel piano  $r-\varphi$  e di circa 580  $\mu m$  nel piano r-z.

#### 1.2.2.3 IL RIVELATORE A RADIAZIONE DI TRANSIZIONE

Questo rivelatore utilizza la tecnologia degli *straw-tubes*, riempiti di una miscela gassosa a base di xenon. Quando una particella carica attraversa il rivelatore, nel gas si creano coppie elettrone-ione che, sotto l'influenza del campo elettrico presente nel volume interno del rivelatore, si muovono verso gli elettrodi. La carica generata e moltiplicata dall'intenso campo elettrico viene rivelata mediante un sottile filo anodico di tungsteno posto al centro del tubo e collegato direttamente all'elettronica di lettura. Il diametro degli *straw-tubes* è di soli 4 mm, in modo da ridurre l'occupazione nel singolo rivelatore, cioè il numero di hit per tubo per evento, pur mantenendo la necessaria stabilità meccanica. Allo stesso tempo, un sottile radiatore in polipropilene, posto tra i tubi, consente di generare la radiazione di transizione. Questa radiazione è prodotta quando una particella carica attraversa la superficie di separazione tra due materiali con diversa costante dielettrica. La probabilità di produrre questa radiazione dipende dalla velocità della particella. Siccome l'effetto diventa significativo solo per velocità relativistiche, la radiazione è di fatto generata solo dagli elettroni ed è quindi usata per riconoscerli tra tutte le altre particelle. I fotoni dovuti alla radiazione di transizione sono assorbiti nei tubi dallo xenon e contribuiscono a generare impulsi di carica di intensità maggiore rispetto ai rilasci di energia dovuti alla sola ionizzazione. L'elettronica è poi in grado di discriminare tra i segnali di traccia, nei quali il rilascio è maggiore di una soglia 'bassa', e i segnali dovuti alla radiazione di transizione nei quali il rilascio è maggiore di una soglia 'alta'. Un elevato numero di *hit*, tipicamente 36 per particella, è prodotto all'interno del rivelatore, e contribuisce in misura significativa alla ricostruzione delle tracce. Il rivelatore consente anche di identificare gli elettroni in un ampio intervallo di energia, e fornisce un'informazione complementare a quella del calorimetro.

Il sistema di tracciatura completo è mostrato nelle Figure 1.12 e 1.13. Si può vedere in entrambe le Figure una traccia di impulso 10 GeV/c attraversare il rivelatore. Nella Tabella 1.2 sono riportati alcuni parametri caratteristici del sistema di tracciatura.



Figura 1.12: Lo schema mostra i rivelatori e gli elementi strutturali attraversati da una traccia carica di  $p_T = 10 \text{ GeV/c}$  nel *barrel*. La traccia attraversa la *beam-pipe* di berillio, i 3 strati cilindrici di *pixel*, i 4 strati cilindrici di *microstrip* in silicio, ed approssimativamente 36 tubi assiali del tracciatore a radiazione di transizione.



Figura 1.13: Nell'immagine sono mostrate due tracce cariche di  $p_T = 10$  GeV/c che attraversano la zona del *barrel* e quella di *end-cap*.

Rivelatore		Raggio(mm)	Lunghezza(mm)
Dimensioni ID		0 < R < 1150	0 <  z  < 3512
Beam pipe		29 < R < 36	
Pixel	Involucro complessivo	45.5 < R < 242	0 <  z  < 3092
3 layer cilindrici	Barrel attivo	50.5 < R < 122.5	0 <  z  < 400.5
2×3 dischi	End-cap attivo	88.8 < R < 149.6	495 <  z  < 650
SCT	Involucro complessivo	255 < R < 549 (barrel)	0 <  z  < 805
	_	251 < R < 610 (end-cap)	810 <  z  < 2797
4 layer cilindrici	Barrel attivo	299 < R < 514	0 <  z  < 749
$2 \times$ 9 dischi	End-cap attivo	275 < R < 560	839 <  z  < 2735
	-		
TRT	Involucro complessivo	544 < R < 1082 (barrel)	0 <  z  < 780
		617 < R < 1106 (end-cap)	827 <  z  < 2744
73 piani di straw	Barrel attivo	563 < R < 1066	0 <  z  < 712
160 piani di tubi	End-cap attivo	644 < R < 1004	848 <  z  < 2710

Tabella 1.2: Parametri principali dei rivelatori che costituiscono il sistema di tracciatura.

Per semplificare la trattazione possiamo pensare che il sistema di tracciatura sia schematicamente costituito da un certo numero di strati o *layer*, ciascuno dei quali è segmentato in un certo numero di canali. In questo modo, un qualunque piano passante per l'asse del sistema individua una matrice di sensori in grado di segnalare il passaggio di particelle cariche. Quando le particelle cariche prodotte nelle collisioni protone-protone lo attraversano, colpiscono un canale per strato, e provocano l'attivazione dei sensori ai vari livelli. Si è quindi in grado di identificare dei punti nello spazio appartenenti alla traiettoria della particella che attraversa strati successivi del rivelatore, e per ogni strato si ha un punto della traiettoria che può essere stimata per interpolazione. Dalla traiettoria ricostruita è possibile determinare le variabili cinematiche della particella, per esempio la quantità di moto. In particolare, le particelle a bassa quantità di moto hanno traiettorie più spiralizzate a causa della presenza del campo magnetico solenoidale, mentre quelle ad elevata quantità di moto sono più rettilinee.

#### 1.2.3 IL CALORIMETRO

All'esterno dello sistema di tracciatura e del solenoide si incontra il calorimetro, costituito da una sezione elettromagnetica, utilizzata per l'identificazione e la misura di energia di elettroni e fotoni, e da una sezione adronica, per la misura dell'energia dei getti adronici (Figura 1.14).



Figura 1.14: Il calorimetro di ATLAS. Sono indicati i principali componenti della sezione elettromagnetica e di quella adronica.

**Calorimetro Elettromagnetico** Il calorimetro elettromagnetico, similmente agli altri elementi di ATLAS, è suddiviso nella regione del *barrel* e nella regione di *end-cap*. Il *barrel* è ospitato in un grande criostato a forma



Figura 1.15: Struttura ad organetto del calorimetro elettromagnetico.

di toroide cilindrico. Il criostato è utilizzato per mantenere allo stato liquido l'argon che costituisce il materiale sensibile del rivelatore. L'asse del toroide coincide con l'asse dei fasci, ha raggio interno pari a 1.15 m, e raggio esterno di 2.25 m ed è lungo 6 m. Nella regione di *end-cap* ci sono due toroidi cilindrici simmetrici rispetto al punto di interazione. Ognuno è costituito da 16 moduli, in modo che ogni modulo sottenda 1/16 dell'angolo azimutale. Ciascun modulo è a sua volta formato dalla sovrapposizione di 64 strati radiali di assorbitore ed elettrodi a forma di fisarmonica. Il rivelatore utilizza argon liquido come materiale sensibile, in cui sono immersi elettrodi che hanno la particolare forma di organetto (Figura 1.15).

Nella regione del barrel, all'interno del criostato, è posto il precampiona-

tore, che misura la ionizzazione prodotta dagli sciami elettromagnetici originati nel tracciatore e nel materiale di cui è costituito il criostato. I segnali analogici indotti sugli elettrodi sono condotti all'esterno attraverso i cosiddetti passanti freddo-caldo, dove vengono letti, elaborati e digitalizzati da un sistema elettronico di *front-end*. Gli stessi segnali analogici vengono contemporaneamente inviati all'elettronica di trigger di primo livello, mentre quelli digitalizzati vengono letti dalla elettronica di *back-end*, posta a circa 70 m dal rivelatore, per essere utilizzati nel *trigger* di alto livello e venire registrati nel caso in cui l'evento sia selezionato.

Nella regione di *end-cap*, il calorimetro elettromagnetico ha la forma di una ruota con raggio esterno di 2.10 m e interno di 0.33 m. Anche in questo caso gli assorbitori e gli elettrodi hanno una struttura a fisarmonica e sono disposti radialmente. Ogni elettrodo è segmentato in tre diverse sezioni nella direzione di incidenza delle particelle. La granularità in senso azimutale è determinata, ed eventualmente ridotta, collegando in parallelo più elettrodi. Davanti alla ruota esterna è posto un precampionatore che ha la stessa funzione di quello utilizzato nel *barrel*.

**Calorimetro adronico** La sezione adronica del calorimetro centrale si trova immediatamente all'esterno della sezione elettromagnetica ed ha una forma cilindrica, anch'essa suddivisa nella regione del *barrel* e nella regione di *end-cap*. Il rivelatore utilizza tegole di scintillatore quale materiale sensibile e ferro quale assorbitore [12, 13].

#### 1.2.4 IL RIVELATORE PER MUONI

A differenza dei neutrini, che attraversano l'apparato sperimentale senza interagire, e di tutte le altre particelle che, attraversando i rivelatori, generano sciami di particelle, i muoni attraversano la materia perdendo lungo il cammino solo una piccola frazione della loro energia iniziale per ionizzazione. Questo è chiaramente visibile in Figura 1.16, in cui è raffigurato il deposito di energia delle varie particelle nei diversi rivelatori di ATLAS. I muoni si possono facilmente riconoscere poiché sono le sole particelle, provenienti dalla regione di interazione, che raggiungono ed attraversano i rivelatori posti all'esterno del calorimetro. Le camere per la misura della traiettoria dei muoni e i rivelatori per il *trigger* sono posti all'interno del campo magnetico del toroide centrale. La rivelazione dei muoni è effettuata nelle zone in avanti dell'esperimento con l'aggiunta di due toroidi superconduttori più piccoli. Lo spettrometro muonico definisce le dimensioni complessive di ATLAS.

I rivelatori per la misura della posizione e per il *trigger* dei muoni nel *barrel* sono raggruppati in tre strati posti a distanza radiale crescente dall'asse del fascio. Nella stazione più interna sono montate solo camere per la misura di posizione, nelle altre due alle camere di posizione sono aggiunti i rivelatori utilizzati nel *trigger*. Una suddivisione simile, con tre camere per la misura di posizione e per il *trigger*, è mantenuta anche nella zona in avanti dell'esperimento [14].



Figura 1.16: Sezione trasversale dei vari rivelatori utilizzati in ATLAS ed illustrazione della interazione con i diversi tipi di particelle. La linea tratteggiata indica che la particella attraversa il rivelatore senza interagire ed è, di fatto, invisibile.

## 1.2.5 IL SISTEMA DI TRIGGER E DI ACQUISIZIO-NE DATI

Come anticipato nei paragrafi precedenti, la collisione tra due pacchetti di protoni, detta *bunch crossing*, o semplicemente "evento", produce alcune migliaia di particelle che attivano un numero enorme di canali nei rivelatori attraversati. Data l'elevata frequenza di *bunch crossing* (20 MHz) e la mole di dati relativa al singolo evento, non è possibile memorizzare su disco i dati relativi a tutti gli eventi prodotti ma, anzi, se ne può registrare solo una modesta frazione, dell'ordine di 100-200 al secondo, che corrisponde in media ad un evento registrato ogni centomila eventi prodotti. Si deve, inoltre, tener conto del fatto che solo una piccola frazione degli eventi prodotti contiene processi interessanti per le misure di fisica. Questo significa che, non solo il numero di eventi che si possono registrare su memoria permanente è una frazione assai limitata del numero totale di eventi prodotti, ma che si deve anche riuscire a selezionare in tempo reale i rari eventi interessanti.

Per ottenere questi due obiettivi, ATLAS effettua la selezione in tempo reale degli eventi mediante un complesso sistema detto *trigger* che è costituito da tre livelli successivi di analisi, Livello 1, Livello 2 ed Event Filter (Figura 1.17) [15, 16]. Ogni livello del *trigger* filtra ulteriormente gli eventi selezionati dal livello precedente, sulla base di una ricostruzione via via più accurata dei dati provenienti dal rivelatore. Vediamo brevemente quali sono le funzioni svolte da ciascun livello di *trigger*:

• Livello 1: Il trigger di Livello 1 esamina i dati relativi a tutti i bunch crossing, che si verificano alla frequenza di 20 MHz, ed effettua una selezione iniziale basata su una ricostruzione effettuata a risoluzione ridotta dei dati provenienti dal calorimetro e dai rivelatori per muoni. Nel Livello 1 non è usata alcuna informazione del sistema di tracciatura a causa delle restrizioni di timing e della natura eccessivamente complessa di questa informazione. La decisione del trigger di Livello 1 è basata su una combinazione logica delle informazioni raccolte. La sua implementazione è comunque molto flessibile e può essere programmata per selezionare eventi usando segnature complesse. Il trigger di Livello 1 ha inoltre la funzione di identificare il bunch-crossing e sincronizzare tutte le diverse parti del rivelatore per garantire che i dati provenienti dalle diverse parti del rivelatore non appartengano a bunch crossing diversi. Questo non è un obiettivo semplice dato il breve intervallo di tempo (50 ns) tra due *bunch crossing* consecutivi. Problematica è, ad esempio, la gestione dei dati provenienti dal rivelatore per muoni, in cui le sole dimensioni dello spettrometro implicano tempi di volo delle particelle paragonabili ai 50 ns che separano due bunch crossing successivi. La latenza del trigger di Livello 1, misurata dall'istante di collisione protone-protone fino all'istante in cui la decisione del trigger è resa disponibile all'elettronica di front-end, deve essere minore di 2.5  $\mu$ s. Inoltre, per ogni evento, il Livello 1 cerca di individuare delle regioni di interesse nel rivelatore, regioni nelle quali è stata riconosciuta una segnatura di particolare interesse, per esempio la presenza di un leptone energetico, cosicché i livelli successivi di *trigger* possano concentrarsi in un'analisi particolarmente accurata e limitata ai dati di questa regione del rivelatore. Durante il tempo di processamento del Livello 1, le informazioni complete sono lette dall'elettronica di front-end del rivelatore e registrate in *pipeline*. I dati degli eventi accettati dal Livello 1 sono trasferiti al Livello 2 per l'analisi successiva. Ogni evento accettato dal Livello 2 viene successivamente trasferito dal sistema di acquisizione dati ad una appropriata memoria per essere processato dall'Event Filter che esegue il terzo livello della selezione degli eventi. La frequenza degli eventi accettati dal *trigger* di Livello 1 e trasmessi al Livello 2 è,

in media, dell'ordine di 75 KHz.

- Livello 2: Il trigger di Livello 2 utilizza l'informazione del sistema di tracciatura oltre a quella del calorimetro e del rivelatore di muoni. A questo livello si possono eseguire algoritmi di vario genere, inclusa la ricostruzione dei vertici di interazione e di decadimento delle particelle. La decisione di Livello 2 è presa in circa 40 ms, e la frequenza massima di accettazione degli eventi di circa 1 − 2 KHz.
- Event Filter: Gli eventi che passano anche la selezione di Livello 2 vengono riorganizzati dal modulo Event Builder che raggruppa e riordina i dati provenienti da tutte le parti del rivelatore e li invia all'Event Filter, il terzo livello del trigger. L'Event Filter esegue un'analisi dettagliata dell'evento con algoritmi simili a quelli utilizzati nell'analisi off-line usando la maggior parte dei dati di calibrazione, le informazioni di allineamento del sistema di tracciatura e la mappatura dei campi magnetici. L'Event Filter ricostruisce completamente l'evento, ed effettua la selezione finale degli eventi che devono essere scritti sulle memorie di massa per le successive analisi off-line. Il tempo disponibile per la decisione dell'Event Filter è di circa 1 s. La dimensione finale dell'evento è di circa 1 Mbyte, corrispondente ad una frequenza di dati in uscita di circa 100 Mbyte/s e ad una mole annua di 10<sup>15</sup> byte, registrati su memoria permanente e pronti per l'analisi (Figura 1.17).



Figura 1.17: Rappresentazione schematica dell'architettura in 3 livelli del *trigger* di ATLAS, e del sistema di acquisizione dati. Sono riportate le frequenze massime di acquisizione degli eventi consentite per ogni livello di *trigger*.
## Capitolo 2

# IL PROCESSORE FAST TRACKER

Il nuovo processore *Fast TracKer* (FTK) [17] è stato progettato per migliorare le prestazioni del *trigger* di ATLAS nella presa dati programmata per l'anno 2015. Infatti, per la prima volta in ATLAS, FTK consente di effettuare la ricostruzione tridimensionale delle traiettorie, o tracce, delle particelle prodotte nelle collisioni protone-protone che attaversano i rivelatori in silicio, e fornisce questa preziosa informazione al *trigger* di Livello 2.

Come abbiamo detto, FTK viene inserito tra gli attuali Livello 1 e Livello 2 del *trigger* e analizza tutti gli eventi selezionati dal *trigger* di Livello 1 (Figura 2.1). Inoltre, FTK è in grado di ricostruire e calcolare i parametri cinematici delle tracce con impulso trasverso superiore tipicamente a 1 GeV/c. La qualità delle tracce ricostruite da FTK è assai vicina alla qualità delle tracce ricostruite dai programmi *off-line* [18, 19]. Questo significa che le selezioni degli eventi che FTK consente di effettuare nel *trigger* è paragonabile in termini di efficienza sui segnali di fisica e capacità di reiezione degli eventi di fondo non interessanti, alle più sofisticate selezioni *off-line*.

Il tempo impiegato da FTK per ricostruire tutte le tracce presenti in un evento è di circa 30 microsecondi, un tempo del tutto compatibile con i vincoli imposti dal sistema di acquisizione dati di ATLAS.



Figura 2.1: Rappresentazione schematica del sistema di acquisizione dati e del *trigger* di ATLAS, suddiviso in tre livelli. FTK si colloca tra gli attuali Livello 1 e Livello 2.

## 2.1 PRINCIPI DI FUNZIONAMENTO DI FTK

La ricostruzione delle traiettorie delle particelle che attraversano il sistema di tracciatura viene effettuata nella analisi *off-line* mediante complessi algoritmi che sono tipicamente suddivisi in due fasi successive. Nella prima fase si effettua il riconoscimento dei punti sui piani del rivelatore, detti hit nel nostro linguaggio, che sono stati colpiti da una stessa particella. Nella seconda fase si esegue il fit delle coordinate degli hit, si ricostruisce la traiettoria seguita dalla particella e se ne misurano i parametri cinematici, in particolare la quantità di moto o impulso. Questa tecnica è sicuramente molto efficace sia per la qualità della ricostruzione delle tracce, sia per la risoluzione che si ottiene sulla misura dei parametri cinematici delle particelle. Tuttavia, i calcoli necessari sono assai complessi e infattibli nei limiti di potenza di calcolo e di tempo disponibili a livello di trigger. Per ottenere gli stessi risultati, FTK utilizza un approccio suddiviso in due fasi, nelle quali, in un certo senso, si parte dal risultato. Nella prima fase, FTK riceve i dati dai rivelatori in silicio, e individua le combinazioni di hit che hanno buona probabilità di essere dovute al passaggio della stessa particella nel rivelatore. Nella seconda fase, FTK esegue il *fit* di queste combinazioni di *hit* e effettua la ricostruzione tridimensionale delle tracce. Questa tecnica potrebbe apparire la stessa dell'analisi off-line, la differenza sta nel fatto che le combinazioni di hit, individuate nella prima fase, vengono individuate a risoluzione spaziale ridotta rispetto a quella consentita dal rivelatore in silicio mediante hardware dedicato estremamente veloce. Cerchiamo adesso di spiegare in dettaglio questa procedura.

Il numero di strati che compongono i rivelatori in silicio è elevato (Figura 2.2), per esempio, nella regione del *barrel* si hanno 3 strati di *pixel* e 4 di

*microstrip*. Tenendo conto del fatto che gli strati a *microstrip* sono a doppia faccia, si può dire che, in totale, si hanno 11 strati logicamente indipendenti e, di conseguenza, 11 misure indipendenti sulla traiettoria percorsa da ogni particella.



Figura 2.2: Configurazione degli strati nel rivelatore in silicio di ATLAS. Si possono riconoscere i 3 strati a *pixel* nella regione interna (azzurro, verde, blu) e i 4 strati a *microstrip* nella regione più esterna.

Un rivelatore in silicio è una giunzione p-n inversamente polarizzata. In un rivelatore, la carica rilasciata nella regione di svuotamento viene raccolta in corrispondenza della giunzione e viene registrata. Una particella, che attraversa il rivelatore, genera una certa quantità di carica di ionizzazione nel silicio. Il campo elettrico presente nel silicio trascina la carica in direzione degli elettrodi, generando quindi un segnale elettrico rivelabile. Questo segnale viene letto tramite l'elettronica di readout posta all'estremità del rivelatore (Figura 2.3).

Ogni strato del rivelatore è composto da un elevato numero di canali di lettura. Per ridurre artificialmente la risoluzione del rivelatore, invece di



Figura 2.3: (a) Veduta in sezione di una porzione di rivelatore a *microstrip*. Si può notare il campo elettrico interno, che trascina la carica in prossimità di un canale di lettura, che raccoglie la carica e genera un segnale elettrico, letto da opportuna elettronica. (b) Altra veduta in sezione di un rivelatore a *microstrip*.

usare la suddivisione nei canali reali del rivelatore, si può introdurre una suddivisione astratta in canali virtuali, detti *bin*. Un *bin* contiene più canali di lettura adiacenti ed in pratica corrisponde all'OR logico tra più canali. (Figura 2.4).

Ogni volta che una particella colpisce un *layer* del rivelatore può attivare uno o più canali di lettura. Nel primo caso, la particella interseca il *layer* esattamente in corrispondenza di un canale, determinando un *hit* (in blu della Figura 2.5) e l'accensione di un solo canale (in rosso). Nel secondo caso, invece, la particella colpisce il *layer* in un punto intermedio tra due canali adiacenti e distinti, ed attiva così due canali (in giallo ed arancione).



Figura 2.4: Veduta in sezione di uno strato del rivelatore in silicio. In blu sono stati rappresentati alcuni canali di lettura che appartengono ad uno stesso *layer*. In verde è mostrata la suddivisione astratta in *bin*. Il *bin* può essere visto, in pratica, come l'OR logico tra canali adiacenti.



Figura 2.5: Nella figura di sinistra, la particella colpisce il rivelatore in corrispondenza di un canale di lettura e l'hit è rivelato su un solo canale di lettura. Nella figura di destra, la particella colpisce il rivelatore nella regione compresa tra due canali adiacenti e l'hit è rivelato su due canali di lettura adiacenti.

In Figura 2.6.a, è mostrato un rivelatore ideale composto da 4 *layer*, suddivisi in *bin*. Quando una particella attraversa un *layer* e genera un *hit* ed attiva un canale di lettura, tutto il *bin* che contiene quel canale viene attivato (Fig 2.6.b). In blu è indicato l'*hit* contenuto nel *bin*. La combinazione di più *bin* attivati, uno per ogni *layer*, costituisce la *road* o *pattern* (in rosso in Figura 2.6.c). Naturalmente, la *road* non è una combinazione casuale di *bin*, ma una combinazione compatibile con la traiettoria di una particella reale, proveniente dalla regione nella quale si verificano le collisioni protoneprotone.



Figura 2.6: (a) Ogni strato del rivelatore è suddiviso in *bin* (in verde). (b) La presenza di un *hit* (blu) attiva l'intero *bin* nel quale è contenuto. (c) La combinazione di più *bin*, uno per ogni *layer*, determina una *road* o *pattern* riconducibile alla traiettoria reale della particella.

Si può quindi interpretare la *road* come una candidata traccia ricostruita a risoluzione spaziale ridotta rispetto a quella consentita dal rivelatore. Riconoscere la combinazione di *hit* dovuta al passaggio di una particella è un problema di grande complessità computazionale. FTK utilizza *hardware* dedicato per questa operazione che nel seguito chiameremo di *pattern recognition* e sfrutta un'importante semplificazione. Poiché la dimensione dei *bin* è finita, anche il numero di combinazioni casuali tra *bin* su diversi *layer*, seppur estremamente elevata, è finita. Mediante la simulazione si possono precalcolare tutte le possibili *road* riconducibili alle traiettorie di particelle reali che risultano essere un numero assai più limitato rispetto alle combinazioni casuali. Una volta calcolate, le *road* vengono registrate in una memoria, detta banca dei *pattern* o delle *road*. Al momento della presa dati, le *road*  memorizzate vengono confrontate con la lista di *hit* trovati nel rivelatore in ogni evento. Se un numero sufficiente di *hit* in un evento è stato trovato all'interno di una *road*, si può dire di aver individuato una candidata traccia nell'evento. Successivamente, nella seconda fase, le *road* e gli *hit* in esse trovati, sono trasmesse ad una batteria di processori che effettuano un fit tridimensionale delle traiettorie delle particelle e ne determinano i parametri cinematici. Il vantaggio di questa tecnica, che rende tutta la procedura estremamente veloce ed utilizzabile a livello di *trigger*, sta nell'effettuare mediante *hardware* dedicato la funzione di individuazione delle *road*, che è quella computazionalmente più complessa.

Questa prima fase utilizza un forte meccanismo di parallelizzazione, processando simultaneamente anche migliaia di *hit* per evento, che sono confrontati con di milioni di *road* allo stesso tempo. Naturalmente è necessario ottimizzare la dimensione della *road*. Se la *road* è troppo "sottile" e, di conseguenza, ne risulta necessario un numero eccessivamente elevato per coprire efficientemente l'intero rivelatore, le dimensioni e il costo della memoria sono troppo elevati. Se una *road* è troppo larga, è pur vero che il numero di *road* necessario per coprire il rivelatore risulta ridotto, ma il carico dei processori che eseguono i *fit* diviene eccessivo, a causa dell'elevato numero di *road* false e di *hit* scorrelati contenuti in ogni *road* che aumentano il numero di combinazioni di *hit* da elaborare.

### 2.2 ARCHITETTURA DI FTK

Vediamo adesso quali sono i principali blocchi funzionali che costituiscono il processore FTK.

Per poter gestire l'enorme flusso di dati provenienti dal rivelatore è stato necessario parallelizzare le funzioni di FTK. Per questo motivo, il rivelatore è stato suddiviso in 8 regioni che coprono 45° ciascuna lungo l'angolo azimutale. I dati provenienti da ogni regione sono inviati ad una porzione autonoma di FTK, detta *Core Processor*. FTK, di conseguenza, è composto 8 *Core Processor*, ognuno dei quali occupa un intero *Core Crate* VME (Figura 2.7). Il singolo *Core Processor* è ulteriormente suddiviso in 16 *Processing Unit*. 2 *Processing Unit* elaborano i dati provenienti da una porzione limitata di una regione del rivelatore, nel seguito chiamata "torre" (Figura 2.8). Il rivelatore risulta così suddiviso in un totale di 64 torri e FTK risulta suddiviso in 128 *Processing Unit*. In Figura 2.7 è mostrata l'intera struttura di FTK, con 64 torri, 8 *Core Crate* e 128 *Processing Unit*. In Figura 2.8 è mostrata la struttura di un *Core Crate*, che ospita un *Core Processor*, composto da 16 *Processing Unit*.



Figura 2.7: Struttura di FTK, composta da 8 *Core Processor* ospitati in 8 *Core Crate*. Ogni *Core Crate* contiene 16 *Processing Unit*, per un totale di 128 *Processing Unit*. I dati di una singola torre sono elaborati da una coppia di *Processing Unit*.



Figura 2.8: Schema di un *Core Crate*. Ogni *Core Crate* ospita 16 *Processing Unit*. I dati provenienti da una singola torre del rivelatore sono elaborati da una coppia di *Processing Unit*.

In Figura 2.9 è mostrato lo scheletro di un *Core Crate* VME nel quale possono essere ospitate le 16 *Processing Unit* che costituiscono un *Core Pro*- *cessor* di FTK. Il *Crate* VME è a disposizione del laboratorio del gruppo FTK presso la Sezione di Pisa dell'INFN.



Figura 2.9: Immagine di un *Core Crate* VME che può ospitare un *Core Processor*. Nel crate sono disponibili le locazioni per il montaggio della CPU e delle 16 *Processing Unit*.

Da quanto detto in precedenza, emerge chiaramente che il processore FTK è un dispositivo estremamente complesso. FTK è infatti composto da 128 *Processing Unit* indipendenti, che elaborano in parallelo i dati ricevuti dalle 64 torri in cui è suddiviso il rivelatore in silicio. Con questa architettura, FTK permette di elaborare una mole di dati enorme (Figura 2.10).



Figura 2.10: Schema semplificato di un *Core Processor* di FTK. Ogni *Core Processor* è costituito da 16 *Processing Unit* identiche. Ogni coppia di *Processing Unit* elabora i dati provenienti da una torre del rivelatore. Nello schema sono indicati i diversi blocchi funzionali che compongono una *Processing Unit*.

La Figura 2.10 mostra la struttura semplificata di un *Core Processor*. Ogni *Processing Unit* è composta da vari blocchi funzionali che abbiamo denominato *Data Organizer, Associative Memory, Track Fitter ed Hit Warrior*. Una coppia di *Processing Unit* analizza i dati provenienti dalla torre del rivelatore ad essa assegnata. Nelle zone di confine tra torri adiacenti viene effettuata una duplicazione dei dati. Due *Core Processor* o due *Processing Unit* adiacenti ricevono entrambi i dati della zona di sovrapposizione e questo consente di ridurre la possibile inefficienza nella ricostruzione delle tracce tra due regioni adiacenti (Figura 2.11). Se questo non fosse stato fatto, le tracce che attraversano per una parte una regione e per una parte la regione adiacente sarebbero state irrimediabilmente perdute. Con questa tecnica dei "confini allacciati" per ogni regione, questo problema risulta significativamente ridotto.



Figura 2.11: Suddivisione del rivelatore in regioni. In rosso è messa in evidenza la regione di sovrapposizione tra due regioni adiacenti. I dati della regione di sovrapposizione sono spediti simultaneamente alle *Processing Unit* corrispondenti alle due regioni adiacenti.

Descriveremo nel seguito il compito svolto da ciascun blocco funzionale che compone una *Processing Unit*. **Data Formatter** I dati provenienti dal rivelatore in silicio sono trasmessi ad FTK tramite fibre ottiche *S-Link*, uno standard di acquisizione dati ad elevate prestazioni sviluppato al CERN [20], che utilizza un bus a 32 bit con una frequenza di trasmissione dei dati fino a 66 MHz.

Questi dati sono trasmessi dall'elettronica di *front-end* ai *Data Formatter*, processori che eseguono il *clustering*, cioè calcolano la posizione degli *hit* sui piani del rivelatore in silicio. Come detto in precedenza, un *hit* è il deposito di carica dovuto ad una particella che attraversa lo strato di silicio. La coordinata dell'*hit*, determinata calcolando il baricentro del deposito di carica, è la stima della posizione nella quale la particella ha attraversato lo strato. Ogni *hit* è trasmesso alla *Processing Unit* dedicata alla elaborazione dei dati della regione del rivelatore nel quale l'*hit* è contenuto.

Data Organizer Gli hit trovati dal Data Formatter sono trasmessi al Data Organizer. Il Data Organizer proietta ogni hit nel bin che lo contiene, ed invia la sequenza di bin colpiti alle schede di memoria associativa. Il Data Organizer conserva anche una copia degli hit ricevuti, in attesa di ricevere la lista delle road trovata dalla memoria associativa. Quando la memoria associativa individua le road presenti nell'evento, le invia al Data Organizer, che procede a recuperare gli hit contenuti in ciascuna road (Figura 2.12). L'insieme di road e hit sono inviati al Track Fitter che esegue il fit tridimensionale delle tracce. Utilizzando questa procedura, il Track Fitter deve analizzare un numero limitato di combinazioni di hit, cioè solo quelle



ottenute con tutti e soli gli *hit* contenuti in una *road*.

Figura 2.12: Schema delle funzioni svolte da FTK. Il Data Organizer riceve la lista di *hit* trovati nel rivelatore in un evento e trasmette la lista dei *bin* colpiti. Questa lista viene confrontata con i *pattern* memorizzati nei *chip* di memoria associativa. Il Data Organizer riceve le *road* trovate nell'evento, ed invia *hit* e *road* al *Track Fitter* che esegue il *fit* delle tracce a piena risoluzione.

Il Data Organizer è stato progettato per processare i dati alla frequenza massima di eventi accettati dal trigger di Livello 1 di 100 KHz e può analizzare due eventi in parallelo. Il Data Organizer può ricevere gli hit appartenenti ad un evento e allo stesso tempo ricevere le road dell'evento precedente dalla memoria associativa e recuperare gli hit in esse trovati dalla memoria.

Memoria Associativa Questo dispositivo esegue il confronto tra la lista dei *bin* colpiti ed i *pattern* precalcolati in essa memorizzati. Così, la memoria associativa [21, 22] individua le candidate tracce a bassa risoluzione, dette *road*, presenti nell'evento e le trasmette al *Data Organizer*. Una *road*  viene selezionata se al suo interno sono presenti un numero di *bin* colpiti maggiore di una soglia programmabile.

**Track Fitter** Il Track Fitter esegue il fit per ogni combinazione di hit (uno per layer) contenuti nelle road e trova i parametri delle tracce con ottima precisione. Le dimensione delle road devono essere tali che il numero di hit trovati nella stessa road sia limitato. Questo riduce il combinatorio all'interno della road e di conseguenza il numero medio di fit che i Track Fitter devono eseguire per ogni road, limitando i tempi di esecuzione del Track Fitter. Il fit viene realizzato all'interno di un FPGA che consente di effettuare la ricostruzione di circa  $10^9$  tracce al secondo su ciascun dispositivo.

*Hit Warrior* Questo sistema elimina gli eventuali duplicati o tracce false, che si possono presentare dopo le operazioni di *pattern matching* e di *track fitting*. Una coppia di tracce duplicate condivide la maggior parte degli *hit*, ma non tutti. Il *Track Fitter* può generare i duplicati se all'interno delle *road* sono presenti *hit* di rumore o di altre particelle vicini a quella di interesse. L'*Hit Warrior* riconosce ed elimina le tracce duplicate mediante algoritmi di selezione appositamente studiati.

## Capitolo 3

# SUDDIVISIONE DI FTK IN 128 PROCESSING UNIT

L'unità logica elementare di FTK è detta *Processing Unit* (Figura 3.1) ed è costituita da due schede VME connesse sul *backplane*. La scheda principale è la scheda di memoria associativa, detta AMB-FTK. Essa ospita i *chip* di memoria associativa ed è affiancata da una scheda ausiliaria detta *Auxilary Board* che esegue le funzioni del *Data Formatter*, del *Track Fitter* e dell'*Hit Warrior* descritte nel Capitolo 2.

La scheda AMB-FTK è stata progettata e realizzata presso la Sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare, mentre la *Auxiliary Board* è stata progettata e realizzata dal gruppo di ricerca dell'Enrico Fermi Institute - The University of Chicago. I *chip* di memoria associativa non sono montati direttamente sulla scheda di memoria associativa, ma sono ospitati su 4 schede di dimensioni ridotte chiamate *Little Associative Memory Board*, o LAMB-FTK, montate sulla scheda principale AMB-FTK. Nel seguito di questo Capitolo descriveremo brevemente l'architettura e le funzioni della scheda AMB-FTK e della *Auxiliary Board*, mentre il chip di memoria associativa e la scheda LAMB-FTK saranno descritti nel Capitolo 4. Ricordiamo che la funzione principale di una *Processing Unit* è ricevere gli *hit* presenti nella torre del rivelatore ad essa associata e ricostruire le traiettorie delle particelle che l'hanno attraversata.



Figura 3.1: (a) Rappresentazione schematica di una *Processing Unit* di FTK. Sono visibili le 4 schede LAMB montate sulla scheda AMB-FTK e la *Auxiliary Board*. (b) Ingrandimento della connessione su backplane tra la scheda AMB-FTK e la *Auxiliary Board*.

# 3.1 ARCHITETTURA E FUNZIONI DELLA SCHEDA AMB-FTK

La funzione della scheda AMB-FTK è ricevere dal *Data Organizer* la lista dei *bin* colpiti nel rivelatore, distribuirli ai *chip* di memoria associativa presenti sulle 4 schede LAMB-FTK, raccogliere le *road* trovate dai *chip* di memoria associativa ed inviarle di nuovo al *Data Organizer* sulla *Auxilary Board*. La Figura 3.2 mostra il percorso completo dei dati all'interno della *Processing Unit*.



Figura 3.2: Schema riepilogativo del percorso compiuto dai dati all'interno di una *Processing Unit*.

La Auxiliary Board è di fatto la scheda che riceve i dati in ingresso e spedisce i dati in uscita da FTK. Infatti, essa riceve la lista degli *hit* dal rivelatore e, in uscita, trasmette le tracce ricostruite al *trigger* di Livello 2. Ogni scheda LAMB-FTK ospita e gestisce 32 *chip* di memoria associativa, 16 per ogni lato. Attraverso la scheda LAMB-FTK, la lista di *bin* è distribuita a tutti i *chip* di memoria associativa, che provvedono ad individuare le *road* presenti nell'evento. Al termine di questa operazione, la scheda LAMB-FTK raccoglie le *road* trovate da tutti i *chip* di memoria associativa e le spedisce alla AMB-FTK. Questi dati vengono successivamente inviati dalla AMB-FTK alla *Auxiliary Board*, dove il *Data Organizer* recupera gli *hit* presenti nelle *road* ed il *Track Fitter* esegue il *fit* tridimensionale delle tracce. La scheda AMB-FTK comunica con l'esterno attraverso tre connettori, i connettori standard P1 e P2 comunemente utilizzati nel protocollo VME [23], ed il connettore P3, utilizzato per la trasmissione dati ad alta frequenza da e verso la *Auxiliary Board*.

Sia sulla scheda AMB-FTK che sulla scheda LAMB-FTK sono stati montati alcuni dispositivi programmabili prodotti dalla Xilinx, degli FPGA della famiglia Spartan 6<sup>1</sup> e CPLD<sup>2</sup>. Mediante questi *chip* le due schede riescono a svolgere tutte le funzioni necessarie per il funzionamento di FTK. In questo lavoro di Tesi, mi sono occupato di scrivere il microcodice di programmazione, o *firmware*, utilizzato in alcuni di questi dispositivi.

Descriviamo adesso le funzioni svolte da questi dispositivi programma-

<sup>&</sup>lt;sup>1</sup>La famiglia Xilinx Spartan 6 sarà descritta in dettaglio nel Capitolo 5.

<sup>&</sup>lt;sup>2</sup>Il Complex Programmable Logic Device, o comunemente CPLD, è un dispositivo programmable logic device, programmabile e cancellabile. La programmazione permette al CPLD di simulare un generico circuito digitale di complessità non elevata. I CPLD vengono usati prevalentemente per applicazioni ad alta velocità e basso costo o funzionalità di glue logic, come in questo caso, ovvero di interfacciamento tra due dispositivi complessi.

bili. A ciascun dispositivo abbiamo dato un nome che ricorda la funzione svolta. Avremo, naturalmente, cura di menzionare esplicitamente quali sono i dispositivi per i quali ho sviluppato personalmente il *firmware*.

#### 3.1.1 CONTROL - FPGA

La FPGA evidenziata in verde in Figura 3.3 è la cosidetta CONTROL-FPGA. Questa FPGA può essere considerata come il "cervello" della scheda. Essa è connessa a tutti i dispositivi presenti sulla *Processing Unit* e ne controlla le operazioni. In particolare, essa controlla:

- l'inizio e la fine dell'analisi di un evento.
- la trasmissione della lista dei *bin* colpiti dalla scheda AMB-FTK alle schede LAMB-FTK tramite Pixel-FPGA e SCT-FPGA.
- la memorizzazione nei *buffer* dei *chip* di memoria associativa della lista di *bin* ricevuta dalla scheda LAMB-FTK.
- la trasmissione delle *road* trovate nei *chip* di memoria associativa alla *Auxiliary Board* tramite i Roadout-FPGA.

In Figura 3.3 sono evidenziati i percorsi dei segnali bidirezionali scambiati tra Control-FPGA e i dispositivi sulle schede LAMB-FTK e *Auxiliary Board*. In particolare, in verde è indicato il percorso dei segnali verso il connettore utilizzato per la comunicazione tra le schede LAMB-FTK e AMB-FTK. Un importante segnale di controllo generato dal Control-FPGA è il segnale di l'Init<sup>3</sup> con cui si comunica a tutti i *chip* presenti sulla LAMB-FTK l'inizio di un nuovo evento o si asserisce un *reset* globale del sistema. Il Control-FPGA genera altri segnali necessari per operazioni di diagnostica. In rosso, sono evidenziati le connessioni tra Control-FPGA e Pixel-FPGA e SCT-FPGA. Fra queste sono presenti segnali necessari per il protocollo di comunicazione e di controllo della trasmissione dei dati in uscita dai due *chip*. In blu, sono evidenziati i collegamenti con i Roadout-FPGA. In arancio è riportato il percorso verso la Auxiliary Board.

<sup>&</sup>lt;sup>3</sup>Per maggiori dettagli sul segnale di Init si rimanda al Capitolo 6.



Figura 3.3: Schema della scheda AMB-FTK, in cui sono messi in evidenza la posizione e le connessioni di Control-FPGA, che è in pratica il "cervello" della scheda AMB-FTK. In verde è indicato il percorso dei dati verso il connettore utilizzato per la comunicazione tra la scheda LAMB-FTK e AMB-FTK. In rosso sono evidenziati le connessioni tra Control-FPGA e Pixel-FPGA e SCT-FPGA. In blu sono mostrati i collegamenti con i Roadout-FPGA. In arancio è evidenziata la connessione verso la *Auxiliary Board*.

#### 3.1.2 PIXEL - FPGA E SCT - FPGA

La scheda AMB-FTK riceve su dodici linee differenziali e in formato seriale i dati che, attraverso la *Auxiliary Board*, provengono dagli otto strati del rivelatore in silicio, quattro strati dal rivelatore a *pixel* e quattro strati dal rivelatore a *microstrip*. Poiché dal rivelatore a *pixel* si riceve un maggior flusso di dati, ad esso sono dedicate 8 linee differenziali, mentre solo 4 linee sono utilizzate per i dati provenienti dal rivelatore a *microstrip*. Le corrispondenti liste di *bin* colpiti, trovate dal Data Organizer, sono ricevuti sulla scheda AMB-FTK su due FPGA Xilinx Spartan 6 (XC6SLX75T e XC6SLX45T), che abbiamo denominato Pixel-FPGA e SCT-FPGA. La Pixel-FPGA riceve i dati dal rivelatore a *pixel*, la SCT-FPGA riceve i dati dal rivelatore a *microstrip*. (Figura 3.4).



Figura 3.4: Schema della scheda AMB-FTK: in rosso, è indicato il percorso dei dati verso le LAMB-FTK a partire da SCT-FPGA e Pixel-FPGA, che ricevono le liste dei *bin* colpiti sul rivelatore dalla *Auxiliary Board*.

La lista dei bin colpiti che provengono dalla Auxiliary Board attraversano

e giungono ai *chip* SCT-FPGA e Pixel-FPGA tramite i dispositivi Micrel SY58601U<sup>4</sup>, che sono *buffer* LVPECL differenziali a 800 mV con terminazioni interne. Questi FPGA replicano questi dati alle 4 schede LAMB-FTK. Nella versione della scheda attualmente disponibile nei laboratori della Sezione INFN di Pisa, i due FPGA distribuiscono i *bin* alle LAMB-FTK in parte su linee seriali e in parte su bus paralleli. Nella nuova versione della scheda, che è stata recentemente riprogettata [24] e sarà presto disponibile, saranno presenti solo connessioni di tipo seriale.

Il protocollo di comunicazione tra Auxiliary Board e la scheda AMB-FTK tiene conto del fatto che la trasmissione dei dati sui 12 link seriali è asincrona. Per questo motivo nel Pixel-FPGA e nel SCT-FPGA i dati sono ricevuti in 12 buffer FIFO, uno per ogni link seriale. Se una FIFO risulta parzialmente riempita superando una certa soglia programmata, un segnale di hold viene mandato alla Auxiliary Board, che sospende la trasmissione dei dati fino a che qualche altra locazione del buffer non diventa nuovamente disponibile. I 12 segnali di hold sono mandati alla Auxiliary Board attraverso il connettore P3. Allo stesso modo funziona anche la trasmissione verso la Auxiliary Board, ma questa volta sono stati previsti 16 buffer FIFO per ricevere i dati sulla Auxiliary Board. Anche in questo caso, 16 segnali di hold sono mandati dalla Auxiliary Board alla scheda AMB-FTK, per sospendere la trasmissione delle road se qualche FIFO dovesse risultare parzialmente riempita.

 $<sup>^{4}</sup>$ Il dispositivo Micrel SY58601U è un *buffer* ad elevata precisione che può gestire segnali di clock fino a 5 GHz e flussi di dati fino a 5 Gbps.

#### 3.1.3 ROADOUT-FPGA

In Figura 3.5 sono messi in evidenza i Roadout-FPGA presenti sulla scheda AMB-FTK. Questi dispositivi raccolgono i dati dalle LAMB-FTK per ritrasmetterli al Data Organizer presente sulla *Auxiliary Board* tramite il connettore P3. Ogni LAMB-FTK fornisce in uscita 4 link seriali differenziali, così ogni Roadout-FPGA riceve in totale 8 link seriali differenziali. Discuteremo in maggior dettaglio il Roadout-FPGA nel Capitolo 7, dove ne descriveremo l'architettura logica e i blocchi funzionali interni.



Figura 3.5: In blu sono messi in evidenza i Roadout-FPGA sulla scheda AMB-FTK. Ogni Roadout-FPGA riceve i dati da 2 schede LAMB-FTK.

#### 3.1.4 BUS PARALLELI E LINK SERIALI

Per minimizzare il costo dei primi prototipi della scheda, per i bus di dati sono stati inizialmente utilizzati solo bus paralleli. Nei prototipi successivi sono stati utilizzati sia bus paralleli sia bus seriali e l'obiettivo per il prossimo prototipo è la serializzazione di tutti i bus presenti sulla scheda. Nel prototipo utilizzato in questa Tesi, la PIXEL-FPGA riceve i dati in ingresso su 8 link seriali e trasmette alle schede LAMB-FTK tramite una combinazione di link seriali e di bus paralleli. La SCT-FPGA riceve i dati in ingresso su 4 link seriali e trasmette i dati in uscita verso le schede LAMB-FTK utilizzando link seriali e bus paralleli. In Figura 3.6, sono mostrate le connessioni seriali e parallele tra le due FPGA ed i connettori per la comunicazione con le 4 schede LAMB-FTK.



Figura 3.6: Schema delle connessioni seriali e dei bus paralleli in uscita da SCT-FPGA e Pixel-FPGA verso i connettori con le schede LAMB-FTK. In rosso sono illustrati i collegamenti su link seriale, in verde sono illustrati i collegamenti su bus paralleli. In blu sono illustrati i dispositivi Micrel, utilizzati per rigenerare il segnale.

#### 3.1.5 DISTRIBUZIONE DEL CLOCK

Sulla scheda AMB-FTK si distribuisce un clock a 100 MHz ai CPLD sulla scheda AMB-FTK e ai *chip* delle schede LAMB-FTK come segnale *single ended* CMOS, e si utilizzano *clock* differenziali per pilotare le Spartan FP- GA<sup>5</sup>. A partire da un oscillatore SI598, identificato dal rettangolo giallo in Figura 3.7, è stata fatta una duplicazione del segnale grazie ai *fan-out buffer* 1:8. Questi *buffer* forniscono un clock differenziale che viene mandato a due LAMB-FTK e alle FPGA presenti su di esse. Uno dei clock differenziali generato viene inviato anche ad un altro blocco, chiamato *RoboClock* (blocco viola) che genera un clock *single ended* CMOS destinato ai CPLD e ai *chip* presenti sulla scheda di memoria associativa. Ovviamente, nel fare questo, sono stati eseguiti tutti i controlli standard della lunghezza delle linee del clock, per evitare ritardi spuri fra i vari blocchi. Infine, seguendo i *datasheet* dei vari componenti interessati, sono state realizzate le terminazioni adeguate, per evitare di degradare il segnale di *clock*.

 $<sup>^5 {\</sup>rm Spartan-6}$ è una famiglia di FPGA della casa produttrice Xilinx. Per maggiori dettagli, si rimanda al Capitolo 5.



Figura 3.7: Schema dei generatori e rete di distribuzione del segnale di clock sulla scheda AMB-FTK. In giallo, abbiamo messo in evidenza l'oscillatore a 100 MHz; in verde, i generatori di clock per i GTP delle FPGA; in viola, il *RoboClock*; in blu, la rete di collegamento ed i segnali di clock *single ended*; in rosso, la rete di collegamento ed i segnali di clock differenziali.

#### 3.1.6 GESTIONE SIMULTANEA DI DUE EVENTI

Tra le caratteristiche particolari della scheda AMB-FTK, possiamo menzionare il fatto che essa gestisce simultaneamente due eventi consecutivi. Infatti, mentre gli *hit* di un evento sono caricati nella memoria associativa, le *road* dell'evento precedente sono trasmesse in uscita verso la *Auxiliary Board*. Vediamo in dettaglio cosa accade. Mentre gli *hit* dell'evento *N* sono trasferiti nei *buffer* dei *chip* di memoria associativa sulle schede LAMB-FTK, le *road* trovate nell'evento *N-1* sono trasmesse dai Glue-FPGA<sup>6</sup> verso i Roadout-FPGA. Successivamente, i Roadout-FPGA ritrasmettono le *road*, tramite il connettore P3, all'*Auxiliary Board*. Quando le LAMB-FTK hanno raccolto le *road* da inviare in uscita alla *Auxiliary board*, inclusa una parola di fine evento, l'evento *N-1* può essere considerato processato e terminato.

## 3.2 AUXILIARY BOARD

La Auxiliary Board è stata realizzata presso l'Enrico Fermi Institute - The University of Chicago. Questa scheda riceve la lista degli *hit* dal rivelatore in silicio e trasmette le tracce ricostruite al *trigger* di Livello 2, e svolge le funzioni del *Data Organizer*, del *Track Fitter* e del'*Hit Warrior* (Figura 3.2).

Il Data Organizer riceve la lista di hit che provengono dai vari strati del rivelatore, trova i bin colpiti e ne effettua la trasmissione alla AMB-FTK attraverso il connettore P3. Le road trovate dai chip di memoria associativa vengono indirizzate verso il Roadout-FPGA sulla scheda AMB-FTK, che effettua la ritrasmissione, attraverso il connettore P3, al Data Organizer sulla scheda Axiliary Board. A questo punto, il Data Organizer raccoglie le road trovate e gli hit in esse contenuti e invia questi dati al Track Fitter. Dopo il fit

<sup>&</sup>lt;sup>6</sup>Il *chip* di colla, o Glue-FPGA, è il dispositivo posto in uscita sulla scheda LAMB-FTK. Esso raccoglie le *road* trovate dai *chip* di memoria associativa, e le trasmette alla *Auxilary Board*. Per maggiori dettagli, dato che la programmazione del Glue-FPGA è stata parte del mio Lavoro di Tesi, si rimanda al Capitolo 6.

tridimensionale degli *hit* eseguito dal *Track Fitter*, e il filtro eseguito dall'*Hit Warrior*, le tracce ricostruite sono spedite dalla *Auxiliary Board* al *trigger* di Livello 2 per essere utilizzate nella selezione degli eventi interessanti per le misure di fisica.

## Capitolo 4

# IL CHIP DI MEMORIA ASSOCIATIVA E LA SCHEDA LAMB-FTK

In questo Capitolo descriveremo le funzioni svolte dal *chip* di memoria associativa e dalla scheda su cui esso è montato, che abbiamo denominato *Little Associative Memory Board* o, più brevemente, LAMB-FTK.

## 4.1 IL CHIP DI MEMORIA ASSOCIATIVA

Il compito di FTK è ricostruire le traiettorie compiute dalle particelle nel rivelatore di silicio con ottima risoluzione ed in tempi estremamente brevi. Per fare questo, l'intero sistema è caratterizzato da un'architettura che permette di eseguire l'algoritmo di *pattern matching* in parallelo sui dati che provengono dalle 64 torri del rivelatore. Come abbiamo spiegato nella Sezione 2.2, FTK è costituito da 128 *Processing Unit* che funzionano in parallelo, con 2 *Processing Unit* che elaborano i dati relativi alla torre ad esse associata.

Un pattern è associato ad 8 celle di una Content Addressable Memory o memoria CAM. A differenza di una normale RAM, in cui si fornisce l'indirizzo di una cella e si ottiene in uscita il dato contenuto in quella cella, nella CAM l'ingresso è costituito dal dato stesso e, se questo dato è presente in una qualunque cella della memoria, allora viene presentato in uscita l'indirizzo della cella in cui quel dato è contenuto. Se non vi è corrispondenza fra il dato in ingresso ed un dato localizzato nelle celle, in uscita non viene presentato alcun indirizzo. Inoltre, mentre in una normale CAM il dato deve giungere per intero ad un certo istante, nel chip custom dell'INFN il dato può giungere segmentato ed in momenti diversi.

La struttura del *chip* di memoria associativa può essere ricondotta ad una matrice bidimensionale, nella quale ogni riga rappresenta un *pattern* di memoria ed è composta da 8 celle CAM, una per ogni strato del rivelatore. Alle colonne della matrice corrispondono i bus sui quali scorrono i dati (Figura 4.1).



Figura 4.1: La struttura del *chip* di memoria associativa può essere ricondotta ad una matrice bidimensionale, nella quale ogni riga rappresenta un *pattern* di memoria ed è composta da 8 celle CAM, una per ogni strato del rivelatore. Alle colonne della matrice corrispondono i bus sui quali scorrono i dati.

Le colonne della matrice corrispondono ai bus di dati e su ogni colonna scorrono i dati provenienti da un singolo strato del rivelatore.

Come visto in precedenza nell'operazione di *pattern recognition*, non vengono utilizzati tutti gli 11 strati disponibili del rivelatore in silicio, ma solamente i dati provenienti da 8 strati. Questa scelta permette di limitare la dimensione della memoria associativa. Le simulazioni mostrano che questa scelta è un buon compromesso tra la qualità delle tracce ricostruite, il costo dell'intero sistema, e la velocità di elaborazione.

Ogni cella del pattern contiene una parola di 15 bit che individua la posizione del bin sul corrispondente strato del rivelatore.

Il confronto tra i dati in ingresso ed il contenuto di ogni cella CAM è fatto tramite un comparatore che confronta il dato memorizzato con il dato in ingresso. Il risultato del confronto viene salvato in un *flip-flop* chiamato *layer match*. Alla fine dell'analisi di un evento, tutti i *bin* colpiti su ogni *layer* sono stati confrontati con tutti i *pattern* presenti in memoria e, se si è trovato un numero sufficiente di *bin* colpiti in un *pattern*, si può dire che il *pattern* è scattato, ovvero che è stata trovata una candidata traccia o *road*. Per risalire a quale *bin* è scattato e su quale strato del rivelatore, ogni *road* trovata in un evento viene trasmessa unitamente alla *bitmap*, cioè alla mappatura che identifica quale *bin* di quale *layer* è scattato per quella *road*.

14 bit 14 bit						
ADDRESS	BITMAP	ADDRESS	BITMAP	•••	ADDRESS	BITMAP
ROAD 0	ROAD 0	ROAD 1	ROAD 1		ROAD n	ROAD n

Figura 4.2: Formato dei dati in uscita dal *chip* di memoria associativa. Nella prima parola è trasmesso l'indirizzo della *road* trovata e nella seconda parola è trasmessa la *bitmap*, cioè la mappatura degli strati sui quali sono stati trovati *bin* colpiti.

Ovviamente, durante l'analisi di un evento è possibile che un *pattern* scatti in un istante qualsiasi, in quanto ogni *pattern* memorizzato viene confrontato con tutti i *bin* colpiti presenti in ogni evento che scorrono continuamente sul bus di dati. Per ottimizzare i tempi di analisi, vengono estratti immediatamente i *pattern* per i quali tutti i *layer* sono scattati, anche se questo accade prima del termine dell'analisi di tutti i *bin* colpiti presenti in un evento. Per
tutti gli altri *pattern*, che possono invece avere meno di 8 *layer* colpiti, si deve aspettare la fine dell'analisi di tutta la lista di *bin* prima di poterli eventualmente raccogliere e utilizzare (Figura 4.3).



Figura 4.3: Nella Figura a sinistra è mostrato un *pattern* con 4 *layer* colpiti su 4. Nella Figura a destra è illustrato un *pattern* con 3 *layer* colpiti su 4. Nel primo caso il pattern trovato può essere estratto dal *chip* di memoria associativa prima che sia terminata l'analisi dei dati dell'evento, nel secondo caso si deve aspettare che l'analisi dei dati dell'evento sia stata del tutto terminata.

Le road trovate vengono lette in ordine di numero di *layer* mancanti partendo con quelle dove risultano colpiti tutti e 8 i *bin*, che permettono di ricostruire tracce di miglior qualità. La lettura prosegue con le *road* che hanno 1 *layer* mancante. Ovviamente, sarebbe possibile abbassare ulteriormente la soglia, cioè la richiesta di *layer* colpiti nella *road*, ma questo comporterebbe un peggioramento della qualità delle tracce ricostruite, che risulterebbe insufficiente per il funzionamento del *trigger* di Livello 2 di ATLAS. Il protocollo di comunicazione tra il *chip* di memoria associativa ed i *chip* programmabili montati sulla LAMB-FTK utilizzati per raccogliere le *road* trovate sarà descritto nel Capitolo 6.

# 4.2 ARCHITETTURA DELLA SCHEDA LAMB-FTK

Come mostrato in Figura 4.4, la scheda LAMB-FTK è stata resa il più possibile simmetrica. Essa ospita 32 *chip* di memoria associativa, 16 per lato, organizzati in due matrici 4x4. I *pattern* sono memorizzati nei *chip* di memoria associativa, utilizzando il protocollo JTAG e il VME.



Figura 4.4: Immagine della LAMB-FTK. A sinistra è visibile il lato *bottom* della scheda ed è messo in evidenza il connettore attraverso il quale transitano i dati. A destra è visibile il lato *top* e sono evidenziati i *chip* che raccolgono e distribuiscono i dati (INDI, SCAN, GLUE) e una locazione in cui è installato un *chip* di memoria associativa.

Nella colonna centrale della Figura 4.5 sono stati evidenziati i chip (FPGA

e CPLD) che effettuano la distribuzione dei *bin* colpiti ai *chip* di memoria associativa. Li abbiamo denominati "INDI" per il fatto che essi svolgono la funzione di *INput DIstributor*. Tutti i dati in uscita dai *chip* di memoria associativa vengono raccolti da 2 Glue-FPGA<sup>1</sup>, che si occupano di serializzare i dati e di spedirli verso la *Auxiliary Board*, passando per i *chip* Roadout-FPGA sulla AMB-FTK. Questi *chip* verranno descritti in maggior dettaglio nei prossimi Capitoli, in quanto la loro programmazione (parziale o totale) è stata il mio lavoro di Tesi.

### 4.2.1 ALIMENTAZIONE DEL CHIP DI MEMORIA ASSOCIATIVA

Il *chip* di memoria associativa è un dispositivo *standard cell* 65 nm ed è caratterizzato da un package LQ208, di spessore di 1.4 mm. Il *chip* utilizza due tipi di alimentazioni, 1.2 V per il core e 3.3 V per le interfacce di I/O. La prima alimentazione è fornita alla scheda LAMB-FTK attraverso i connettori evidenziati in giallo nella Figura 4.5, mentre la seconda alimentazione è fornita dai connettori SMD evidenziati in viola. Il connettore evidenziato in viola viene anche utilizzato per la trasmissione dei dati fra le schede LAMB-FTK e AMB-FTK.

<sup>&</sup>lt;sup>1</sup>Il nome Glue-FPGA o *chip* di "colla" è stato scelto in riferimento alla funzione svolta. Infatti, il compito principale del *chip* è di raccogliere i dati in uscita da tutti *chip* di memoria associativa e di "incollarli" in un unico bus di uscita. Per ogni altro dettaglio, si rimanda al Capitolo 6.



Figura 4.5: Schema della scheda LAMB-FTK. Sono messe in evidenza i connettori per i due tipi di alimentazioni necessarie del *chip* di memoria associativa. In particolare, in viola è evidenziato il connettore per l'I/O Voltage (3.3 V), necessaria per alimentare l'interfaccia input/output del *chip*. In giallo è mostrata la posizione del connettore per il *Core* Voltage (1.2 V), necessaria per il funzionamento del *chip*.

## 4.2.2 DISTRIBUZIONE DEI DATI AI CHIP DI ME-MORIA ASSOCIATIVA

La lista dei *bin* colpiti è trasmessa alle schede LAMB-FTK attraverso i connettori SMD. Sul prototipo attualmente a disposizione in laboratorio la trasmissione dei dati è effettuata da parte dei *chip* denominati *INput DIstributor* (*INDI*) in parte con un protocollo parallelo ed in parte seriale. Questi dispositivi sono CPLD ed FPGA. In particolare, i 4 bus distribuiti in parallelo dalla AMB-FTK sono ricevuti da 3 CPLD per ogni metà LAMB-FTK, per un totale di 6 CPLD per ogni LAMB-FTK, mostrati nel rettangolo blu in Figura 4.6.

Invece, i 4 bus distribuiti serialmente dalla AMB-FTK sono ricevuti da due FPGA Spartan-6 nella parte inferiore della LAMB-FTK. Questi *chip* utilizzano i loro GTP<sup>2</sup> di ingresso per convertire in parallelo i dati ricevuti e inviarli ai *chip* di memoria associativa (Figura 4.7).



Figura 4.6: I CPLD, posti al centro della LAMB-FTK, distribuiscono su un bus parallelo una parte della lista di *bin* colpiti ai *chip* di memoria associativa.

 $<sup>^2\</sup>mathrm{Nel}$  Capitolo 5 saranno descritti i GTP, Intellectual Property delle Xilinx Spartan 6 FPGA.



Figura 4.7: Gli FPGA Spartan-6 posti al centro della LAMB-FTK distribuiscono su link seriali una parte della lista di *bin* colpiti ai *chip* di memoria associativa.

## 4.2.3 ESTRAZIONE DELLE ROAD DAI CHIP DI ME-MORIA ASSOCIATIVA

I Glue-FPGA (Figura 4.8) raccolgono le *road* trovate dai *chip* di memoria associativa, ne fanno la serializzazione e ne gestiscono la trasmissione verso la scheda AMB-FTK. Si deve notare che i dati in uscita da un *chip* di memoria associativa posto in alto vengono trasmessi al *chip* di memoria associativa collocato immediatamente sotto e sono successivamente ritrasmessi al *chip* a valle della catena, fino ad arrivare alla Glue-FPGA. I 32 *chip* di memoria associativa sono collegati in 8 *pipeline* di 4 *chip* di memoria associativa ciascuna su ogni LAMB-FTK, 4 *pipeline* sul lato *Top* e 4 sul lato *Bottom*. Le *road*  vengono instradate da queste *pipeline* verso 4 bus di uscita tramite 2 *chip* Glue-FPGA, mostrati nella finestra verde in Figura 4.8. I segnali si propagano da *chip* di memoria associativa a *chip* di memoria associativa attraverso piccoli collegamenti, senza l'utilizzo di vie. Descriveremo nel Capitolo 6 il protocollo di comunicazione usato per trasmettere le road in uscita dal *chip* di memoria associativa al Glue-FPGA.



Figura 4.8: Schema di collegamento delle uscite dei *chip* di memoria associativa alle 2 Glue-FPGA, evidenziate nei quadrati verdi.

# 4.3 BREVE STORIA DEL CHIP DI MEMO-RIA ASSOCIATIVA

Il *chip* di memoria associativa attualmente utilizzato sulla scheda LAMB-FTK è la versione AM-chip04, ed è il prodotto di una evoluzione e sviluppo continui dei prototipi precedenti. I miglioramenti principali sono stati l'incremento della densità di memoria e la riduzione dei consumi.

Il primo prototipo di AMchip è stato sviluppato nell'anno 1992, per il processore SVT utilizzato nell'esperimento CDF a Fermilab [25, 26]. Si trattava di un *chip full custom* realizzato con tecnologica 0.7  $\mu m$ , che poteva contenere 128 pattern [27] (Figura 4.9a). Nell'anno 1998, è stato realizzato un secondo prototipo, mappato su FPGA. Anche in questo caso, il *chip* poteva contenere 128 pattern [28] (Figura 4.9b). Nell'anno 2006 è stato realizzato un terzo prototipo *standard cell*, con tecnologia 0.18  $\mu m$ , che poteva contenere 5000 pattern in un *die* di 1 cm<sup>2</sup>[29]. Questo nuovo dispositivo è stato usato per l'*upgrade* di SVT a CDF.

Nell'anno 2012 è stato progettato e realizzato il *chip* AM-chip04, con tecnologia 65 nm, anche questo a standard cell. Per memorizzare i *pattern* si utilizza una cella *full custom* progettata interamente dal gruppo di ricerca di FTK. Il *chip* è stato realizzato su un *die* di silicio di 14 mm<sup>2</sup>, assorbe 0.25 W e può contenere circa 8000 *pattern*.

Nell'anno 2013 è stato avviato il progetto di un nuovo *chip* di memoria associativa, con l'obiettivo di utilizzarlo nella presa dati di ATLAS programmata per l'anno 2015. Questo *chip* può memorizzare ben 32000 *pattern*, ed assorbe circa 1.5 W di potenza.



Figura 4.9: (a) Primo prototipo del *chip* di memoria associativa realizzato nell'anno 1992 per il processore SVT utilizzato nell'esperimento CDF di Fermilab. (b) Secondo prototipo mappato su FPGA della casa produttrice Xilinx realizzato nell'anno 1998. (c) Terzo prototipo *standard cell* realizzato nell'anno 2003 per l'*upgrade* del processore SVT. (d) Quarto prototipo *standard cell* realizzato nell'anno 2012 per il processore FTK all'esperimento ATLAS del CERN.

# Capitolo 5

# LA FAMIGLIA SPARTAN-6 DELLA XILINX

In questo Capitolo descriveremo brevemente le caratteristiche principali di alcuni FPGA della casa produttrice Xilinx che appartengono alla famiglia denominata Spartan-6 e sono ampiamente utilizzati in FTK.

Questa famiglia di FPGA [30] rappresenta un ottimo insieme di dispositivi che può essere utilizzato per numerose applicazioni e fornisce un'elevata programmabilità a basso costo. I dispositivi sono realizzati con una tecnologia *low power* 45 nm, che offre un ottimo bilanciamento tra costi, consumo di potenza e prestazioni. Essi dispongono internamente di vari blocchi logici programmabili, come manager di clock, blocchi di memoria RAM, blocchi di memoria FIFO, e *Low Power Gigabit Transceivers* per trasmissioni veloci di dati, tutti elementi logici che ho ampiamente utilizzato nel mio lavoro. Nel seguito del Capitolo descriveremo brevemente le caratteristiche principali di questi componenti.

### 5.1 MANAGER DEL CLOCK

Ogni FPGA della famiglia Spartan-6 dispone fino a 6 *Clock Management Tile* (CMT), ognuno composto da due *Digital Clock Manager* (DCM) e un *Phase Loocked Loop* (PLL), che possono essere usati anche individualmente.

Il DCM fornisce quattro fasi del segnale di clock in ingresso, cioè uno sfasamento di 0°, 90°, 180° o 270°. Esso può fornire, inoltre, un segnale di *clock* a frequenza doppia, uno a frequenza doppia e sfasato di 180°, uno a frequenza dimezzata e, tramite un *Phase Shifter*, fornire uno sfasamento a scelta al segnale di *clock* in ingresso.

Il PLL è un sintetizzatore di frequenza<sup>1</sup> che può operare in un ampio intervallo di valori, ed è costituito principalmente da un Oscillatore Controllato in Tensione (*Voltage Controlled Oscillator*, o VCO), in un intervallo di frequenza che va da 400 MHz a 1.08 GHz. Ovviamente, variando opportuni parametri di controllo, è possibile ridurre o aumentare la frequenza del segnale di *clock* generato, a partire da un segnale in ingresso ad una frequenza appartenente all'intervallo indicato.

Ogni dispositivo della famiglia Spartan-6 fornisce un elevato numero di linee per la distribuzione del *clock*, con la possibilità di avere un elevato *fanout*,

<sup>&</sup>lt;sup>1</sup>Generatore di frequenza a partire da una frequenza fissa, fornita da un'oscillatore esterno.

un limitato ritardo di propagazione, e uno skew estremamente contenuto.

### 5.2 PIN DI INPUT/OUTPUT

Il numero di *pin* di I/O è elevato e dipende dal dispositivo e dal *package* utilizzato. Ogni *pin* di I/O è configurabile e si adatta ad ogni tipo di standard, fino a 3.3 V  $^2$ .

Tutti i *pin* di I/O sono organizzati in banchi, con il dispositivo più piccolo che contiene 4 banchi, mentre quello più grande contiene 6 banchi. Il numero di *pin* per banco è variabile e dipende dal dispositivo. Il Glue-FPGA ha 250 *pin* di I/O suddivisi in 4 banchi, mentre il Roadout-FPGA ha 348 *pin* suddivisi in 6 banchi. Inoltre, ogni banco contiene diversi *pin* di uscita di alimentazione che vengono utilizzati per pilotare alcuni *buffer* posti in ingresso su altri *pin*. Altri *pin*, invece, richiedono tensioni di riferimento particolari. I *pin* di uscita *single-ended* usano una struttura di uscita di tipo *push-pull* CMOS <sup>3</sup>, con la possibilità di riportare in uscita un livello logico alto, basso o di alta impedenza.

Poiché molte applicazioni combinano Input/Output seriali ad alta velocità con operazioni parallele più lente all'interno del dispositivo, è presente internamente una struttura *SerDes*, che serializza o deserializza i dati in uscita ed in ingresso. Ogni *input* ha accesso al suo deserializzatore, con la

<sup>&</sup>lt;sup>2</sup>Valore massimo di tensione associabile al livello logico 1.

<sup>&</sup>lt;sup>3</sup>Circuito elettronico in uscita costituito da una coppia di transistor che alternativamente forniscono o prelevano corrente dal carico.

possibilità di variare la larghezza del bus parallelo. Se l'ingresso è differenziale, allora entrambi i deserializzatori presenti in corrispondenza di quei *pin* sono utilizzati per ampliare il *bus* parallelo.

Ogni uscita ha accesso al suo serializzatore, con la possibilità anche in questo caso di determinare l'ampiezza del bus parallelo, e di impostarlo *single-ended* o differenziale.

### 5.3 BLOCCHI DI MEMORIA

Nei dispositivi della famiglia Spartan-6 è possibile realizzare dei blocchi di memoria, utilizzando le risorse integrate ed, in particolare, configurarle come blocchi RAM, RAM distribuite oppure come memorie di tipo FIFO.

#### 5.3.1 MEMORIA RAM

Ogni Spartan-6 ha tra le 12 e 268 RAM *dual-port*, ognuna della quali capace di memorizzare 18 Kb. Ogni RAM ha inoltre due porte completamente indipendenti, una per la scrittura ed una per la lettura, che condividono solo i dati memorizzati. Per questo, si hanno a disposizione due *pin* di ingresso per due eventuali *clock* diversi, uno per la fase di scrittura ed uno per la fase di lettura. Inoltre, durante una fase di scrittura, è possibile leggere i dati precedentemente memorizzati, quelli appena scritti, oppure nulla. Ogni RAM da 18 Kb può essere divisa in due RAM da 9 Kb completamente indipendenti fra di loro. La gran parte dei dispositivi della famiglia Spartan-6 include blocchi dedicati di controllo delle memorie, *Memory Control Block* (MCB). Il MCB gestisce della memoria ed, in pratica, svolge tutte le operazioni logiche necessarie sia in fase di scrittura sia in fase di lettura. Per ogni MCB sono a disposizione *pin* che, se non utilizzati, sono disponibili come generici pin di *Input/Output*.

#### 5.3.2 MEMORIA FIFO

Come è possibile notare in Figura 5.1, si possono utilizzare due segnali di *clock* distinti, uno per la fase di scrittura nella FIFO, e l'altro per la fase di lettura dalla memoria. La configurazione dei *clock* indipendenti permette quindi al programmatore di implementare segnali differenti per le due porte. Non è necessario, inoltre, programmare eventuali relazioni di fase o di frequenza fra i due *clock*. Le operazioni in fase di scrittura sono sincrone con il *clock* di scrittura, così come le operazioni di lettura sono sincrone con il *clock* di lettura. Al contrario, utilizzando un unico *clock*, si ottimizza il *buffering* e la trasmissione dei dati.

In entrambe le modalità, sono disponibili numerosi segnali che forniscono informazioni utili sullo stato della FIFO, come ad esempio *FIFO Full, FI-FO Almost Full, FIFO Empty, FIFO Almost Empty,* che sono determinati utilizzando delle soglie programmabili. In più, sono supportati anche *flag* di segnalazione errore o utili per implementare protocolli di *handshake* con altri dispositivi, come ad esempio *FIFO Overflow, FIFO Data Valid, FIFO Underflow,* un *Data Count* che fornisce in uscita il numero di parole scritte correttamente nella FIFO, ed un segnale di *Reset* della memoria, che può essere sincrono o asincrono.



Figura 5.1: Rappresentazione schematica di una memoria FIFO. Sono mostrati i due domini di clock indipendenti e i segnali di input ed output.

In fase di scrittura (Figura 5.2), il segnale di Write Enable (WR\_EN) abilita la porta di scrittura e i dati vengono forniti sul bus Data In (Din[N:0]), sincrono con il segnale di clock per la fase di scrittura (WR\_CLK). La flag FIFO Full indica il riempimento della FIFO, e se questo accade tutte le nuove richieste di scrittura in memoria vengono ignorate, per impedire di sovrascrivere i dati già memorizzati. Se la memoria è stata per errore sovrascritta, questo è segnalato dal flag di uscita Overflow. Il segnale di Almost Full indica che solo un'altra parola può essere scritta in memoria prima di riempire completamente la memoria. In alternativa, si può utilizzare Prog Full, un segnale che controlla una soglia programmabile mediante il segnale prog full *thresh* che indica l'avvenuto riempimento della memoria e di ignorare ogni altra richiesta di scrittura. *Data Count* [D:0] indica il numero di parole scritte nella FIFO. Il contatore permette al programmatore di controllare che la FIFO non venga sovrascritta. In Figura 5.2 è riportata la temporizzazione dei segnali in fase di scrittura.



Figura 5.2: Temporizzazione della fase di scrittura di una FIFO.

In fase di lettura, i dati sono forniti sul bus  $Data \ Out \ (Dout[M:0])$  sincrono con il segnale di clock per la lettura  $(RD\_CLK)$ . Anche in questo caso è presente un segnale di abilitazione *Read Enable*  $(RD\_EN)$ . Se la FIFO è vuota e, quindi, non è possibile leggere alcun dato, un flag di *FIFO Empty* è posto ad 1 e si ignora ogni richiesta di lettura dei dati. Anche in questa fase, è possibile utilizzare flag come *Almost Empty* o *Prog Empty* per arrestare la lettura ed evitare il completo svuotamento della memoria. In Figura 5.3 è riportata la temporizzazione in fase di lettura, con clock indipendenti.



Figura 5.3: Temporizzazione della fase di lettura di una FIFO.

Le FIFO sono definiti Intellectual Property Core  $^4$  e, quindi, la loro implementazione avviene tramite generatori appositi. In Figura 5.4 è riportata una schermata di un generatore di FIFO nella *suite* di programmazione ISE della Xilinx.

<sup>&</sup>lt;sup>4</sup>*Intellectual Property Core.* Unità di proprietà intellettuale della Xilinx. Per il loro utilizzo, la suite di programmazione della Xilinx, ISE, prevede dei generatori di Core.



Figura 5.4: Schermata generale di un generatore di FIFO Intellectual Property Core. Nella suite di programmazione della Xilinx è previsto un Core Generator per ogni Intellectual Property Core.

### 5.4 LOW POWER GIGABIT TRANSCEIVER

Per permettere connessioni e trasferimenti dati molto veloci, tutti i dispositivi Spartan-6 LXT forniscono dai 2 agli 8 *Gigabit Transceiver* (GTP). Ogni GTP *Transceiver* è una coppia di trasmettitore e ricevitore, capace di operare ad una frequenza fino a 3.2 Gb/s. Il trasmettitore ed il ricevitore sono circuiti indipendenti fra di loro. Il trasmettitore è in pratica un convertitore paralleloseriale, che nel nostro caso ci permette di convertire 20 bit paralleli a 100 MHz in una coppia differenziale seriale a 2GHz. I dati in parallelo in arrivo al GTP vengono prima memorizzati in una FIFO in uscita, e successivamente convertiti in seriale. Il ricevitore, invece, è un convertitore seriale-parallelo, che converte il segnale differenziale seriale in ingresso in un bus parallelo di 20 bit.

#### 5.4.1 GTP TRANSCEIVER

Il GTP *Transceiver* è un dispositivo di ricezione e trasmissione altamente configurabile e completamente integrato. In Figura 5.5, abbiamo riportato un diagramma a blocchi di un GTP *Transceiver*, in cui sono messi in evidenza i vari blocchi di cui è costituito. Nel seguito, descriveremo quelli utilizzati nel *firmware* da me sviluppato per la Tesi.



Figura 5.5: Diagramma a blocchi semplificato di un GTP *Transceiver* della famiglia Spartan-6.

I GTP transceiver sono disposti in gruppi di due in una  $Tile^5$ . Nei dispositivi Spartan-6 FPGA XC6SLX45T<sup>6</sup> sono presenti due Tile, e quindi 4 GTP in totale, mentre nei dispositivi Spartan-6 FPGA XC6SLX75T<sup>7</sup> sono

 $<sup>^5\</sup>mathrm{Struttura}$ schematica, per indicare il raggruppamento di due GTP che condividono più risorse.

<sup>&</sup>lt;sup>6</sup>Il chip Glue-FPGA sulla LAMB-FTK è un dispositivo della serie X45T.

<sup>&</sup>lt;sup>7</sup>Il chip Roadout-FPGA, in uscita dalla AMB-FTK, è della serie X75T.

presenti 4 Tile, con 8 GTP in totale. In Figura 5.6, si osserva la struttura di una  $GTP1 \ DUAL \ TILE^8$ , con 2 GTP transceiver ospitati.

<sup>&</sup>lt;sup>8</sup>La *GTP1 Dual Tile* è una struttura utilizzata dalla Xilinx, per indicare una coppia di GTP *Transceiver* che condividono alcune risorse, come il *clock*.



Figura 5.6: Due GTP Transceivers in una *GTPA1 DUAL Tile*. Al centro, il blocco del clock, risorsa comune tra i due *Transceiver*. In azzurro, sono evidenziati i bus paralleli in ingresso al *trasmitter* ed in uscita dal *receiver* e un particolare segnale di controllo, *txcharisk*. In giallo, sono messi in evidenza il clock in ingresso *clkin* e il PLL condiviso tra due GTP. In verde, sono mostrati i segnali di clock utilizzati dai GTP. In rosso, le uscite Tx e gli ingressi Rx differenziali.

#### 5.4.1.1 TRASMITTER

In Figura 5.7 è riportato lo schema di un trasmettitore implementato in un  $transceiver^9$ .



Figura 5.7: Diagramma a blocchi di un GTP Trasmitter.

L'interfaccia di trasmissione FPGA TX è il blocco interno che raccoglie i dati paralleli provenienti dalla logica della FPGA, e li trasferisce al trasmitter che serializza i dati e li trasmette in uscita dal dispositivo. I dati vengono serializzati, trasmettendo prima il bit 0 del bus txdata (evidenziato in blu in Figura 5.2), txdata(0), fino a txdata(n-1), con n che indica la dimensione del bus parallelo. In uscita, i dati sono trasmessi su una coppia differenziale, Txn e Txp <sup>10</sup> (evidenziato in rosso in Figura 5.6), che sono non solo i pin di

<sup>&</sup>lt;sup>9</sup>Coppia di Trasmitter e Receiver.

<sup>&</sup>lt;sup>10</sup>Txp è il filo positivo della coppia differenziale. Txn è il filo negativo della coppia differenziale di trasmissione.

uscita del GTP ma corrispondono anche a due *pad* della Spartan-6 FPGA, e per questo si deve indicare opportunamente un *constraint*<sup>11</sup> (Fig 5.6). I dati vengono scritti sulla porta *txdata* sul fronte positivo di *txusrclk2*<sup>12</sup> (in verde in Figura 5.6). La larghezza del bus parallelo può essere configurato per essere di 1, 2, o 4 *byte*, e dipende dalla codifica 8B/10B<sup>13</sup> in trasmissione abilitata e dalla velocità della linea di trasmissione. Un ulteriore clock *txusrclk* deve essere fornito per la logica interna del trasmettitore. La frequenza di *txusrclk* è data dalla frequenza di linea diviso la larghezza del bus parallelo interno. La frequenza di *txusrclk2* è invece la stessa di *txusrclk* se il parametro *txdatawidth* è uguale a 0, dimezzata se il parametro vale 1, un quarto se il parametro vale 2. I due clock devono essere *positive-edge aligned* e con il minor *skew* possibile. Per questo, possono essere utilizzare alcune risorse interne, dette *Bufg*, per pilotarli. Inoltre, devono comunque derivare dallo stesso oscillatore, anche se possono operare a frequenze diverse. Ricevono quindi un clock esterno tramite il pin *clkin* ( evidenziato in giallo in Figura 5.6).

**TX 8B/10B Encoder** Questa codifica dei dati in uscita aggiunge due bit per ogni byte che viene trasmesso. Il GTP Trasmitter include un encoder 8B/10B per effettuare questa modifica sui dati in uscita, senza utilizzare ulteriori risorse della FPGA. Se l'encoding non è richiesto, può essere disabilitato

<sup>&</sup>lt;sup>11</sup>Nel Capitolo 7, descrivendo il test dei firmware, si accennerà al file dei constraint dei codici VHDL.

 $<sup>^{12}</sup>$ Il txusrclk2 è il clock di riferimento per la lettura e scrittura su GTP. E' generato da txusrclk dimezzando la frequenza.

 $<sup>^{13}\</sup>mathrm{Questa}$  codifica è descritta in un paragrafo successivo.

per ridurre la latenza per i dati in uscita. In Figura 5.8 è riportato un esempio di codifica 8B/10B. Se txdata è di 16 bit, al termine della trasmissione del primo byte viene trasmesso il primo bit aggiunto j0; successivamente si prosegue con la trasmissione del secondo byte, ed infine il secondo bit aggiunto j1.



Figura 5.8: Codifica 8B/10B.

Questo standard di codifica prevede anche l'utilizzo di parole speciali, indicate genericamente con K-words, che spesso vengono utilizzate per funzioni di controllo e diagnostica dei dati. Per trasmettere una *K-word*, bisogna prima porre ad 1 un opportuno bit, *txcharisk*, che indica appunto al dispositivo a valle del FPGA che i bit che sta per ricevere non sono dati validi ma bit corrispondenti ad una parola di controllo.

Per maggiori dettagli su questa codifica, si rimanda al datasheet dei dispositivi Spartan-6 della Xilinx [30].

#### 5.4.1.2 RECEIVER

In Figura 5.9 è riportato lo schema di *receiver* implementato in un *Transceiver*.



Figura 5.9: Diagramma a blocchi di un GTP Receiver.

A monte del ricevitore, è presente un blocco di Front-end analogico configurabile a seconda dello standard di comunicazione dei dati.

I dati giungono su una coppia differenziale seriale, Rxp ed Rxn, in rosso in Figura 5.6. Dopo una serie di controlli sulla integrità e validità dei dati giunti, il flusso giunge al decoder 8B/10B. Se la codifica/decodifica è abilitata, il decoder trasforma il flusso seriale in un flusso parallelo, secondo lo schema riportato nella Figura 5.9.



Figura 5.10: Decodifica 8B/10B dei dati seriali che giungono in ingresso al receiver.

Il decoder riceve per primo il bit 0, mentre il GTP è solito ricevere per primo il bit più significativo. Per questo, il decoder prevede una logica interna che inverte automaticamente l'ordine dei bit in arrivo. Una volta riordinati i bit in ingresso, il decoder scarta i due bit aggiunti ed associa il primo *byte* sui primi 8 elementi del vettore *rxdata*, e successivamente associa il secondo byte agli elementi *rxdata*[15:8]. Così come in trasmissione, anche in ricezione è possibile avere delle K-words, utilizzate prettamente per la diagnostica o in protocolli di *handshake* fra la Spartan-6 FPGA ed il *chip* che ha trasmesso i dati. Se il dato trasmesso è una parola di controllo, viene settato ad uno *rxcharisk*.

Nei dispositivi della famiglia Spartan-6 FPGA è presente l'*Elastic Buffer*, un elemento che automaticamente effettua l'allineamento fra le fasi del *clock* con cui sono trasmessi i dati e quello utilizzato in ricezione. Se non utilizzato, l'allineamento deve essere effettuato manualmente sfruttando vari bit o segnali di controllo, al fine di avere la validità ed integrità dei dati ricevuti.

Per quanto riguarda i *clock*, anche il ricevitore prevede due domini diversi di *clock*, ma legati fra di loro. I due *clock* sono *rxusrclk* e *rxusrclk2*, in verde in Figura 5.6. Come per la trasmissione, il primo è legato al rapporto tra frequenza della linea di trasmissione e la larghezza del bus parallelo che ospiterà i dati. Il secondo clock, invece, ha una frequenza uguale o ridotta a seconda delle necessità. Sono positive-edge aligned, con lo skew ridotto al minimo.

# Capitolo 6

# ARCHITETTURA LOGICA DEL GLUE-FPGA DELLA SCHEDA LAMB-FTK

In questo Capitolo descriveremo l'architettura del microcodice di programmazione da me sviluppato per il Glue-FPGA della scheda LAMB-FTK. Questo è uno dei miei contributi principali al progetto FTK nei mesi di lavoro che ho svolto presso la Sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare.

La Figura 6.1 mostra i *chip* programmabili utilizzati nella scheda LAMB-FTK. Al centro della scheda, sono evidenziate con i blocchi gialli e rossi, rispettivamente le Spartan-6 e i CPLD, utilizzati per distribuire la lista dei *bin* colpiti ai *chip* di memoria associativa. I Glue-FPGA, evidenziati con i quadrati verde-azzurri in basso, raccolgono dai *chip* di memoria associativa le *road* trovate e le inviano ad altri FPGA detti Roadout-FPGA<sup>1</sup> sulla scheda AMB-FTK (Figura 3.2). Sulla scheda AMB-FTK sono presenti 2 Roadout-FPGA ciascuno dei quali riceve le *road* da 2 LAMB-FTK. Per ogni *road* trovata, il *chip* di memoria associativa trasmette due parole, la prima contiene l'indirizzo della locazione di memoria associativa in cui è stata trovata la *road*, la seconda, chiamata *bitmap*, contiene la lista dei piani che hanno avuto un *bin* colpito per quella *road* (Figura 6.2). I Roadout-FPGA trasmettono le *road* alla *Auxiliary Board*, per il *fit* tridimensionale delle tracce.

 $<sup>^1\</sup>mathrm{Per}$  maggiori dettagli sui chip di Roadout-FPGA, si rimanda al Capitolo 7.



Figura 6.1: Schema della scheda LAMB-FTK. Nel blocco rosso, sono evidenziati i CPLD che distribuiscono i dati ai *chip* di memoria associativa su bus paralleli, nel blocco giallo le Spartan-6 che ricevono i dati su link seriale e li distribuiscono su bus paralleli ai *chip* di memoria associativa , nei quadrati verde-azzurro le due Glue-FPGA che raccolgono le *road* trovate dai *chip* di memoria associativa.



Figura 6.2: Schema semplificato di una scheda LAMB-FTK. Il *chip* denominato GLUE-FPGA ha la funzione di raccogliere le *road* trovate nei *chip* di memoria associativa e di trasmetterle alla Auxilary Board attraverso i *chip* Roadout-FPGA collocati sulla scheda AMB-FTK.

È importante soffermarsi sulla descrizione del protocollo di comunicazione tra i *chip* di memoria associativa ed il Glue-FPGA, per il fatto che questo ha avuto un impatto importante sul *firmware* che ho sviluppato per Glue-FPGA. Il *chip* di memoria associativa più vicino al Glue-FPGA raccoglie le *road* e le *bitmap* da ogni *chip* che lo precede nella colonna e trasmette questi dati al Glue-FPGA. (Figura 6.2 e 6.3).



Figura 6.3: Trasmissione dei dati (*road* e *bitmap*) dai *chip* di memoria associativa al Glue-FPGA. Ogni chip della catena raccoglie *road* e *bitmap* dal *chip* che lo precede, e trasmette i dati al *chip* successivo nella catena. L'ultimo *chip* della catena, il più vicino al Glue-FPGA, raccoglie tutti i dati e li invia al Glue-FPGA.

La logica utilizzata nel Glue-FPGA è stata schematizzata in Figura 6.4. Come spiegato in precedenza, i 32 *chip* di memoria associativa presenti sulla scheda LAMB-FTK sono suddivisi in 8 catene composte da 4 *chip* ciascuna. Ciascuna Glue-FPGA raccoglie i dati da 4 catene di *chip* di memoria associativa, 2 catene sul lato *top* e 2 catene sul lato *bottom*, per un totale di 4 bus paralleli, ciascuno da 14 bit. All'interno della Glue-FPGA questi dati sono immediatamente memorizzati in 4 FIFO, una per ogni catena di *chip* o bus di dati ricevuto in ingresso (Figura 6.4). Per ogni coppia di bus in ingresso è stato creato un blocco funzionale, detto *Merger*, utilizzato per gestire i dati memorizzati nelle 2 FIFO corrispondenti. Per ogni coppia di catene di *chip* di memoria associativa è disponibile un solo GTP ed un solo link seriale differenziale per trasmettere i dati al Roadout-FPGA. Quindi, è stato necessario sviluppare una logica capace di riunire i dati ricevuti da due colonne di *chip* di memoria associativa e trasmetterli sull'unico GTP disponibile.

Il GTP riceve i dati sul bus parallelo da 16 bit, effettua la conversione dei dati in seriale e ne effettua la trasmissione verso Roadout-FPGA. Da ogni Glue-FPGA si hanno in uscita 2 bus seriali e, di conseguenza, in uscita da ogni LAMB-FTK si hanno 4 link seriali, per un totale di 16 link seriali dalla scheda AMB-FTK. (Figura 6.4).



Figura 6.4: Rappresentazione schematica del percorso compiuto dai dati nella Glue-FPGA. Si può notare che il Glue-FPGA è diviso in due unità logiche identiche ed indipendenti, ciascuna delle quali riceve e trasmette i dati da 2 catene di *chip* di memoria associativa. Il *Merger* riunisce in un unico bus di dimensione 16 bit in ingresso al GTP 2 flussi di dati da 14 bit provenienti dalle FIFO in ingresso.

Il dispositivo programmabile utilizzato per realizzare il Glue-FPGA è un XC6SLX25T della famiglia Spartan-6 della Xilinx.

### 6.1 ELEMENTI LOGICI DEL GLUE - FPGA

**Reset - Init Event Recognizer** Qualsiasi funzione della Glue-FPGA inizia solo dopo aver ricevuto un segnale di *Init Event*. Il segnale di *Init Event* è generato dal Control-FPGA della scheda AMB-FTK, indica l'avvio di tutte le operazioni che devono essere svolte per elaborare i dati di un evento ed è trasmesso simultaneamente a tutti i dispositivi delle schede AMB-FTK e LAMB-FTK. Nella versione delle schede attualmente in uso, però, è disponibile un unico collegamento in uscita dal Control-FPGA sia per inviare il segnale di *Init Event* sia per inviare un segnale di *Reset* globale. Il *Reset* globale è un segnale che azzera ogni dispositivo, riportandolo al suo stato iniziale. In pratica, il Control-FPGA invia un unico segnale, genericamente denominato *Init*, che va discriminato logicamente per riconoscere se è un segnale di *Reset* globale o di *Init Event* (Figura 6.5).


Figura 6.5: Sulla sinistra è mostrata la scheda AMB-FTK con tutti i suoi FPGA. Il Control-FPGA genera un segnale di *Init* che viene inviato a basso fanout a tutti i chip. In base alla sua durata, il segnale di *Init* viene interpretato o come un segnale di *Init Event* oppure come un segnale di *Reset* globale. Il segnale di *Init* viene ricevuto dall'unità logica *Reset - Init Event Recognizer* (Figura a destra), presente in ogni FPGA della scheda. In uscita il segnale è duplicato: in blu è illustrato il segnale di *Init Event*, in nero è illustrato il segnale di *Reset* globale.

Il blocco funzionale che fa parte della logica nella Glue-FPGA e riconosce se il segnale di *Init* inviato dal Control-FPGA è un segnale di *Init Event* oppure di *Reset* globale è stato denominato *Reset - Init Event Recognizer* (Figura 6.6).



Figura 6.6: Struttura logica del Reset-Init Event Recognizer.

Se il segnale *Init* in ingresso vale 1 per un periodo minore di 4 cicli di clock viene interpretato come *Init Event*, mentre se vale 1 per un periodo maggiore di 4 cicli di clock viene generato un *Reset* globale. Per distinguere i due casi, è stata realizzata la struttura logica mostrata in Figura 6.6. Supponiamo di voler dare un segnale di *Reset* globale, allora è necessario dare un segnale di *Init* di durata maggiore di 4 cicli di clock. In questa maniera, dopo 4 colpi di clock, il segnale *control* vale 1, il bit a0 vale ancora 1, e grazie al demultiplexer a0 viene associato al segnale di *Reset* globale. Se invece si vuol dare un segnale di *Init Event*, allora il segnale *Init* deve valere 1 per un periodo minore di 4 cicli di clock. Così, la variabile *control* vale 0 sempre, e  $a\theta$  viene subito associato al segnale in ingresso al registro a valle del demultiplexer, e dopo quattro cicli di clock in segnale è associato ad *Init Event*.

Scrittura dei dati nella FIFO di ingresso Si può verificare in Figura 6.4 come il Glue-FPGA sia costituito da due strutture logiche identiche che in parallelo ricevono i dati dai *chip* di memoria associativa. Per semplicità ne descriveremo solamente una (Figura 6.7).



Figura 6.7: Rappresentazione schematica della logica utilizzata per gestire il flusso dei dati ricevuto dai *chip* di memoria associativa all'interno della Glue-FPGA.

I dati provenienti dalle 2 catene di *chip* di memoria associativa *top* e *bottom* associate alla struttura logica rappresentata in Figura 6.7 giungono nel dispositivo tramite 2 bus a 14 bit, denominati *addin top* ed *addin bottom*. Questi dati sono trasmessi alla Glue-FPGA dall'ultimo elemento della catena di *chip* di memoria associativa secondo il protocollo di comunicazione descritto nel paragrafo 6.1.1. In particolare, non appena l'ultimo chip di memoria associativa ha disponibile almeno una road scattata, come mostrato in Figura 6.3, si determina la transizione ad un livello basso di un segnale attivo alto, e denominato dataready (Figura 6.8). Di conseguenza, la scrittura nelle FIFO inizia solo quando il dataready va basso. In realtà, il dataready ha un andamento particolare. Quando il chip di memoria associativa sta trasmettendo l'indirizzo della locazione di memoria dove è stata trovata una road, il dataready rimane basso e quando, al colpo di clock successivo, il chip di memoria associativa trasmette la bitmap associata a quella road, il dataready va alto. La trasmissione dall'ultimo chip di memoria associativa termina quando il dataready torna costante al valore logico 1 senza più transizioni tra i due livelli. Questo significa che non ci sono altre road da trasmettere. È compito della logica del Glue-FPGA saper interpretare le varie fasi.

Un segnale importante per l'handshake fra il chip di memoria associativa e il Glue-FPGA è il selup. Il segnale di selup, anch'esso attivo basso, è trasmesso dal Glue-FPGA al chip di memoria associativa ed è necessario per la sincronizzazione della fase di ricezione dati e scrittura delle FIFO. In pratica, quando il chip di memoria associativa è pronto ad inviare dati ed effettua la transizione del dataready, il selup transita al valore logico 0 ed indica che il Glue-FPGA è pronto per ricevere nuovi dati. Il trasferimento di coppie di parole avviene solo se entrambi i segnali dataready e selup sono attivi nello stesso ciclo di clock in cui è trasferita la prima parola. In Figura 6.8 vediamo un esempio di temporizzazione della fase di trasmissione dati dal chip di memoria associativa al Glue-FPGA.



Figura 6.8: Esempio di temporizzazione della fase di trasmissione dati dal *chip* di memoria associativa alla Glue-FPGA. Nella Figura, è illustrata la trasmissione di due *road* e della *bitmap* corrispondente.

Lettura dei dati dalle FIFO di ingresso La lettura dei dati dalle FIFO di ingresso è completamente gestita dal blocco funzionale *Merger*. Il *Merger* controlla le *flag* di stato delle FIFO, *full FIFO* ed *empty FIFO*, per capire se nelle FIFO sono presenti *road* da leggere. Descriveremo in dettaglio la logica del blocco funzionale *Merger* nella Sezione 6.1.3.

Scrittura e lettura dei dati dalla FIFO intermedia Fra il Merger ed il GTP è stato necessario inserire un'altra memoria FIFO in quanto il *clock* utilizzato dalle FIFO di ingresso e dal *Merger* è quello di sistema, cioè il clock da 100 MHz ricevuto dalla scheda LAMB-FTK, mentre il GTP utilizza un clock ricostruito a partire da un segnale generato da un quarzo di alta precisione ospitato sulla scheda LAMB-FTK, ed i due clock potrebbero non essere in fase. Utilizzando una FIFO intermedia, vengono sincronizzati i dati in uscita dal Merger con il clock del GTP. In questo modo i dati in uscita dal Merger sono scritti con il clock di sistema e sono letti con il clock del GTP.

Serializzazione nel GTP e trasmissione dati Una volta che i dati in uscita dal Merger arrivano al GTP, i 16 bit del bus vengono serializzati secondo la codifica 8B/10B descritta nel Capitolo 5. L'informazione è pronta per essere trasmessa su link seriale differenziale ai Roadout-FPGA della scheda AMB-FTK.

#### 6.1.1 LA LOGICA DEL MERGER

Abbiamo denominato *Merger* il blocco funzionale che gestisce la lettura dei dati registrati nelle FIFO di ingresso del Glue-FPGA. I compiti principali del *Merger* sono i seguenti:

• Il Merger gestisce i due bus di dati ricevuti dalle 2 catene di chip di memoria associativa e memorizzati nelle FIFO di ingresso e instradarli sull'unico bus parallelo di dati disponibile in ingresso al GTP (Figura 6.10). Per ogni coppia top-bottom di catene di chip di memoria associativa si utilizza un Merger. Inoltre, si è deciso di dare priorità alla lettura della catena top. Di conseguenza, se al Merger giungono contemporaneamente dati sia dalla catena top che da quella bottom, i primi ad esser letti e trasmessi sono i dati dalla catena *top*, seguiti dai dati provenienti dalla catena *bottom*.



Figura 6.9: Struttura interna del Merger. Sono visibili i due blocchi Controller, e il blocco indicante la Macchina a Stati Finiti, per la gestione dei dati verso il GTP.

I dati in ingresso, indicati con i bus *din top* e *din bottom* riportati in Figura 6.9, sono le uscite delle 2 FIFO e relativi rispettivamente alla catena sul lato *top* e alla catena sul lato *bottom* (Figura 6.7), viene attivato il segnale *read enable* alla FIFO e i dati vengono letti dalle FIFO. Per maggiori dettagli, si rimanda al Paragrafo 6.1.1.2, dove è descritta la macchina a stati finiti implementata per la trasmissione in uscita dei dati dal *Merger*.

• Il Merger aggiunge 2 bit ai dati trasmessi al GTP, uno di indirizzo ed uno di controllo, per accrescere le informazioni già fornite da road e bitmap. Come detto in precedenza, i 2 bus di dati che giungono al Merger in uscita dalle FIFO sono di 14 bit. Infatti, sia l'indirizzo della road sia la bitmap sono parole di 14 bit. In uscita dal Merger si ha invece un bus di 16 bit (Figura 6.10).



Figura 6.10: Flusso di dati in uscita dal *Merger*. Il *Merger* aggiunge in cima 2 bit ai 14 bit di una parola memorizzata nella FIFO.

In particolare bit 14 vale 1 se il dato che si sta trasmettendo in quell'istante proviene dal lato *top*, vale 0 se proviene dal lato *bottom* (Tabella

	Bit 15	Bit 14	Bit 130
da TOP	0	1	road / bitmap
da BOTTOM	0	0	road / bitmap
END EVENT	1	0	00

Tabella 6.1: Formato del dato a 16 bit in uscita dal Merger.

6.1). Il Most Significant Bit<sup>2</sup> (bit 15) è usato per segnalare l'End Event. Questo bit vale 1 solo sulla parola che conclude la trasmissione di tutte le parole dell'evento in corso sia dal lato top sia dal lato bottom (Tabella 6.1).

Il Merger trasmette parole di controllo. È stato scelto di trasmettere la parola di controllo quando non ci sono parole di un evento da dover trasmettere ed è inserita tra la lista di road e bitmap di due eventi successivi o di due letture da catene di chip di memoria associativa diverse. Una volta terminata la trasmissione dei dati relativi ad un evento, viene prima inviata la parola di End Event, che indica al chip di Roadout-FPGA e al Control-FPGA che non ci sono più road da trasmettere da quei chip di memoria associativa, e successivamente viene inviata una parola di controllo e di riallineamento delle fasi, chiamata K-word. Questa parola di controllo ha un valore prestabilito, e cioè "50BC" in esadecimale. Inoltre, ogni volta che sul bus in uscita dalla Glue-FPGA viene trasmessa la K-word, un segnale txcharisk su 2 bit viene settato a "01", dando ai chip a valle un'ulteriore indicazione della parola K che

<sup>&</sup>lt;sup>2</sup>In informatica il bit più significativo (in inglese MSB, *Most Significant Bit*) è in un numero binario la posizione del bit che ha il valore più grande.

stanno per ricevere. Sul bus in uscita dal *Merger* non è mai presente un dato che non sia o una *road* scattata o la relativa *bitmap* o qualche parola di controllo, tipo 50BC. Così è possibile individuare eventuali errori nella trasmissione dati.



Figura 6.11: Flusso dei dati sul bus che trasmette le informazioni dal Merger al GTP. Due eventi successivi sono separati dalla parola di End Event e dalla K-word "50BC".

La logica interna del *Merger* è costituita principalmente da due unità logiche distinte, una detta *Controller* e una *Finite State Machine*, che descriveremo nei paragrafi seguenti.

#### 6.1.1.1 CONTROLLER

In Figura 6.9 è mostrata la logica interna del *Merger* costituita da due blocchi funzionali denominati *Controller* e da una *Finite State Machine*. La funzione del *Controller* è verificare per quanti cicli di clock il *dataready* rimane al valore 1. Questo controllo è necessario in quanto l'assenza di una transizione del segnale *dataready* per un certo intervallo di tempo vuol dire che non sono più presenti parole da trasmettere e, di conseguenza, se si è in fase di lettura dei dati dal lato *top*, si può passare a leggere i dati dal lato *bottom*, mentre se si è in fase di lettura dei dati dal lato *bottom*, si può trasmettere la parola di *End Event*. L'intervallo di tempo da dover attendere può essere all'incirca 10 cicli di clock. Infatti, supponiamo che sia il chip più lontano dal Glue-FPGA a voler trasmettere indirizzo e *bitmap* di un certo numero di *road*. In questo caso si devono attendere due cicli di clock prima che i due dati siano in uscita da quel chip di memoria associativa stesso. Successivamente, si devono aspettare almeno altri 6 cicli di clock prima che i dati abbiano attraversato altri 3 *chip* di memoria associativa e giungano alla Glue-FPGA. Il totale è quindi di 8 cicli di clock, ma per mantenere un certo margine di sicurezza, si attendono 10 cicli di clock.

Vediamo quindi come funziona e come è stato realizzato il *Controller*. (Figura 6.12).



Figura 6.12: Architettura logica del Controller.

Come è possibile vedere in Figura 6.12, questa unità è caratterizzata da un contatore. L'enable del *counter* è la *and* logica fra due segnali. Il primo segnale è ad 1 ed è dovuto all'arrivo del segnale di *Init Event*. Infatti, il segnale di *Init Event* è un'impulso di breve durata. Controllando con questo segnale un *flip-flop*, si riesce ad avere in uscita un segnale bloccato ad 1 ogni volta che si riceve un segnale di *Init Event*. Il secondo segnale è il risultato dell'*and* logica fra il segnale *dataready* e il segnale *delayed dataready*, cioè il segnale *dataready* ritardato di un ciclo di *clock*.

• I chip di memoria associativa non trasmettono dati al Glue-FPGA. La and fra dataready e delayed dataready vale 1 se e solo se non c'è alcun dato pronto per essere inviato alla Glue-FPGA, cioè quando il segnale *dataready* è sempre alto. Solo in questa maniera, infatti, *dataready* e *delayed dataready* rimangono entrambi alti, la seconda *and* fornisce così un valore alto, ed il contatore è abilitato (Figura 6.13a).

• I chip di memoria associativa trasmettono dati al Glue-FPGA. Se invece il chip di memoria associativa è disponibile ad inviare dati, allora il dataready va basso. In quest'istante il delayed dataready è ancora al suo valore iniziale di 1 e il risultato della prima and è 0. Il risultato di questa and rimane ancorato a 0 finché il dataready torna ad essere fisso ad 1 senza transizioni verso il livello logico basso. Dal colpo di clock successivo anche il delayed dataready torna fisso ad un valore logico alto e il contatore torna ad essere abilitato. In questa maniera si disabilita il contatore quando il chip di memoria associativa sta trasmettendo dati alla Glue-FPGA (Figura 6.13b).



Figura 6.13: La Figura (a) mostra il caso nel quale non ci sono parole da scrivere nella FIFO di ingresso. I segnali *dataready* e *delayed dataready* sono sempre al valore logico 1, quindi il contatore è abilitato sempre. Giunto al valore 10, si passa a leggere dalla catena *Bottom*, oppure si trasmette la parola di *End Event*. La Figura (b) mostra il caso nel quale ci sono parole da scrivere nelle FIFO di ingresso. Il contatore è disabilitato quando *dataready* e *delayed dataready* variano tra il valore logico 0 ed 1, cioè quando si trasmettono parole. Il conteggio riprende quando la *and* fra i due segnali vale 1, e quindi non ci sono più parole da trasmettere.

Perciò, quando non sono disponibili dati, oppure è terminata la trasmissione di un certo numero di dati, il *dataready* torna ad essere alto, il contatore è abilitato, e l'uscita del contatore è il numero di cicli di clock in cui il *da*- taready è rimasto alto dopo l'Init Event. Se questo valore supera la soglia prestabilita, ad esempio i 10 cicli di clock, il Merger considera conclusa la ricezione dati da quella catena, e può commutare su bottom, se prima eravamo su top, oppure mettere sul bus dati txdata la parola di End Event e, successivamente, la parola di controllo, se si stavano ricevendo dati dalla catena bottom.

#### 6.1.1.2 LOGICA DELLA FINITE STATE MACHINE

Per gestire la lettura dalle FIFO e la trasmissione dei dati dal Merger al GTP è stata realizzata una macchina a stati finiti. Questa è necessaria perché le fasi di scrittura e lettura dalle FIFO sono separate. Infatti, mentre in fase di scrittura, le due FIFO possono essere scritte anche contemporaneamente, in fase di lettura è stato necessario implementare il blocco logico, Merger, in quanto si ha solo un bus parallelo in ingresso al GTP ed è importante che sul bus dati in uscita dal Merger ci sia sempre un dato valido, sia esso l'indirizzo di una road o la bitmap corrispondente, oppure una parola di controllo. L'andamento tipico dei dati sul bus è mostrato in Figura 6.11. Dopo aver trasmesso tutti i dati relativi ad un evento, si invia una parola di *End Event* e, successivamente, la parola di controllo "50BC". Questa parola di controllo deve rimanere sul bus dati finché inizia la trasmissione di dati relativa all'evento successivo. In questa maniera siamo in grado di monitorare e valutare i dati trasmessi, in quanto ogni altra parola che non sia un indirizzo di una *road*, una bitmap, o la parola di controllo, indica che c'è stato un errore in trasmissione.

La macchina a stati finiti è di tipo Mealy, in quanto le uscite in ogni stato dipendono sia dallo stato corrente sia dal valore degli ingressi in quell'istante. In Figura 6.13 possiamo vedere un grafo che riassume l'evoluzione degli stati della macchina.



Figura 6.14: Grafo a palle della macchina a stati finiti. Per ogni stato, è indicato il dato in uscita che il *merger* pone su *txdata*.

Gli stati implementati sono 5: Reset-State, Wait-State, Top-State, Bottom-State, End Event-State. Gli ingressi alla macchina a stati finiti sono i segnali di fifo empty da due FIFO, il bus in uscita dout fifo top della FIFO della catena top, il bus in uscita dout fifo bottom della FIFO della catena bottom, il segnale di reset (rst), il clk, il flag bottom e il flag top. Vediamo qual'è il significato dei segnali in ingresso e quali sono le condizioni che permettono alla macchina di transitare da uno stato ad un altro.

- Dout fifo top, dout fifo bottom. Il segnale dout fifo top è assegnato al segnale din top a livello del Merger, così come il segnale dout fifo bot è associato al segnale din bot. Su questi due bus da 14 bit transitano le road trovate nelle due catene di chip di memoria associativa e memorizzate nelle FIFO di ingresso.
- *Empty.* Il segnale di empty vale 1 se la FIFO è vuota, vale 0 se nella FIFO c'è almeno un dato memorizzato.
- Flag top, Flag bottom. Flag top e flag bottom sono un'opportuna combinazione logica fra i bit rispettivamente di drcycle1 e drcycles0, uscite dei 2 contatori su dataready top e dataready bottom, in modo da riconoscere il valore 10. Questi flag sono necessari in quanto determinano le transizioni da Top State a Bottom State, o da Bottom State a End Event State.

Stato Reset e Stato Wait Facciamo riferimento alla Figura 6.10. La macchina si trova inizialmente nello stato di Reset, da cui esce non appena viene fornito un segnale di reset globale. Una volta resettata la macchina, la prima transizione è automatica e si passa nello stato di Wait. In questo stato la macchina attende l'Init Event dal Control-FPGA, cioè il segnale che indica l'avvio di tutte le operazioni su AMB-FTK e LAMB-FTK. Finché

non si ha il segnale dal Control-FPGA si rimane nello stato di *Wait*. In queste condizioni, non essendoci dati validi da dover trasmettere, il *Merger* trasmette sul bus di uscita *bus out* la parola di controllo "50BC".

Stato Top Una volta ricevuto il segnale di Init Event, la macchina transita nello stato denominato Top, in cui si leggono, se presenti, i dati dalla FIFO corrispondente alla colonna *top*, altrimenti si mette sul bus di uscita nuovamente la parola di controllo. In particolare, una volta nello stato Top, si controllano i segnali *fifo empty top* e *flaq top*. Il segnale *empty* di una FIFO viene posto a 0 quando è stato scritto almeno un dato in memoria, altrimenti rimane nel suo stato di attivo alto. Perciò, si controlla il valore del segnale *empty top*, e se questo vale 1 vuol dire che ancora non è stato scritto alcun dato, e quindi sul bus in uscita si pone la parola di controllo "50BC" e si attiva il contatore di cicli del *dataready* riferito alla colonna *top*. Se invece fifo empty top vale 0, allora viene prelevato il dato ed in uscita viene riportato "01" concatenato al primo dato, quindi "01" & din top, aggiungendo i due bit come spiegato precedentemente, e cioè il bit 15 che indica l' End Event, mentre il bit 14 bit indica, se 1, la trasmissione dalla catena top. Lo stato successivo è ancora Top, ed è così fino a che empty top vale 0, riportando sempre in uscita i due bit aggiuntivi concatenati al successivo dato. Quando fifo empty top torna ad 1, vuol dire che è stata letta completamente la FIFO associata alla catena top, e che quindi è terminata per ora la trasmissione dei dati, e si può porre nuovamente sul bus in uscita la parola di controllo "50BC". Questo però non vuol dire che la trasmissione dalla catena top sia finita. Può succedere, infatti, che nuovi dati siano in arrivo, e che quindi il dataready associato alla catena top può ancora variare per trasmettere nuovi dati. Per questo, si controlla flag top, cioè il valore del contatore. Il contatore associato alla catena top comincia a contare nel momento in cui si è entrati per la prima volta nello stato top, e ha fermato il conteggio durante la fase di lettura dei dati, per poi riprendere a contare una volta che il dataready è tornato costante al valore 1, cioè non appena empty top è tornato al valore 1 dopo la fine della lettura dei dati dalla FIFO. Quindi, finché il valore di flag top non vale 1, cioè finchè il dataready non raggiunge i 10 cicli di clock in cui rimane fisso al valore 1, si rimane nello stato di Top, ponendo in uscita la parola di controllo "50BC".

Stato Bottom Una volta che Flag top vale 1, e non ci sono più dati nella FIFO associata alla catena top ed è stato raggiunto il limite di tempo per la trasmissione dei dati dalla catena, allora si passa nello stato di Bottom, in cui si rieseguono in pratica le stesse operazioni e gli stessi controlli sui segnali equivalenti per la catena bottom, e cioè flag bottom e fifo empty bottom. Infatti, non appena si entra nello stato di Bottom, parte il contatore del dataready associato alla catena. Se fifo empty bottom vale 1, allora in uscita si ha la parola di controllo. Se fifo empty bottom vale 0, allora vuol dire che alcuni dati sono stati scritti nella FIFO associata alla catena e vengono letti dal Merger, con la solita convenzione sui due bit aggiuntivi: non essendoci ancora l'*End Event*, allora il bit 15 rimane a 0, così come anche il bit 14, poiché si sta trasmettendo dalla catena *bottom*.

Stato End Event Una volta che empty bottom vale 1 e non ci sono più dati nella FIFO associata alla catena bottom, flag bottom vale 1, cioè è stato raggiunto il limite di tempo per la trasmissione dei dati dalla catena e si passa nello stato di End Event. In questo stato si rimane per un solo ciclo di clock, il tempo necessario ad inviare la parola di End Event.

Dallo stato *End Event* si passa automaticamente nello stato di *Wait*, in cui si rimane finché un nuovo segnale di *Init Event* non è ricevuto. Questo segnale indica l'inizio dell'analisi dell'evento successivo.

Infine, indipendentemente dallo stato in cui si trova la macchina, ad ogni eventuale segnale di *reset* si transita immediatamente nello stato di *Reset*, per poi passare nello stato di *Wait* ed attendere il nuovo *Init Event*.

## Capitolo 7

# ARCHITETTURA LOGICA DEL ROADOUT-FPGA DELLA SCHEDA AMB-FTK

In questo Capitolo descriveremo l'architettura logica del microcodice di programmazione utilizzato nel Roadout-FPGA della scheda AMB-FTK. Questa parte del progetto è frutto del mio lavoro, a parte la logica standard di FTK, come gli *Spy Buffer*, il cui codice già esisteva.

### 7.1 FUNZIONI SVOLTE DAL ROADOUT-FPGA

Come descritto in precedenza, il Roadout-FPGA fa parte della scheda AMB-FTK, riceve le *road* dai Glue-FPGA e ne effettua la trasmissione alla Au*xiliary Board.* In particolare, si utilizzano 2 Roadout-FPGA su ogni scheda AMB-FTK, in quanto ogni Glue-FPGA trasmette in uscita due coppie differenziali. Dal momento che in totale si hanno 4 schede LAMB-FTK e di conseguenza 8 Glue-FPGA, il numero complessivo di coppie differenziali necessarie per la trasmissione dei dati alla *Auxiliary Board* è 16. Il massimo numero di GTP disponibili sui dispositivi utilizzati della famiglia Spartan-6 Xilinx è 8 e, per questo motivo, sulla scheda AMB-FTK sono necessari due Roadout-FPGA che svolgono la stessa funzione logica (Figura 7.1).



Figura 7.1: Schema semplificato che mostra i 4 link seriali in uscita da ogni LAMB-FTK e i due Roadout-FPGA che raccolgono 8 link seriali ciascuno.

In Figura 7.1 abbiamo riportato lo schema delle coppie differenziali in uscita da ogni Glue-FPGA e dei collegamenti seriali fino ai Roadout-FPGA. In ingresso ad ogni Roadout-FPGA si hanno 8 coppie differenziali ed ognuna di esse è in ingresso ad un *GTP Transceiver*. In uscita dal Roadout-FPGA si hanno 8 coppie differenziali connesse al P3 per la trasmissione alla *Auxiliary Board*.

# 7.2 ELEMENTI LOGICI DEL ROADOUT -FPGA

La logica del Roadout-FPGA è abbastanza semplice. Infatti, questo FPGA riceve le road trovate dai chip di memoria associativa e le relative bitmap dal Glue-FPGA, trasmette questi dati alla Auxiliary Board e consente operazioni di diagnostica per monitorare il flusso dei dati che transitano sui link seriali. Non è possibile monitorare direttamente i link seriali, in quanto non è facile ricostruire i dati utilizzando le sequenze trasmesse sui collegamenti differenziali. Per questo, è necessario un dispositivo che permetta di parallelizzare i dati che transitano verso la Auxiliary Board, di copiarli in una memoria apposita, riserializzarli e ritrasmetterli verso l'uscita. In Figura 7.2 è mostrato proprio quanto descritto finora. I dati in ingresso arrivano su 8 link seriali differenziali, denominati Rx, ed ognuno di essi è trasmesso in ingresso ad un GTP del Roadout-FPGA. Questi dati vengono ricevuti e, tramite decodifica 10B/8B, convertiti in parallelo su un bus di 16 bit. Successivamente, vengono memorizzati in 16 memorie FIFO per sincronizzare il clock di sistema con il clock privato dei GTP. Una logica posta a valle gestisce la lettura delle FIFO. I dati vengono nuovamente spediti verso il corrispondente GTP di uscita. La novità in questa logica consiste nella presenza degli *Spy Buffer*, memorie utilizzate per effettuare una copia dei dati in uscita dalle FIFO e permettere così di effettuare operazioni di *monitor* mediante lettura degli *Spy Buffer* da VME.



Figura 7.2: Architettura logica del Roadout-FPGA.

I segnali di *Hold to Lamb* e *Hold from Auxiliary Board* sono utilizzati dalla logica interna al Roadout-FPGA nel protocollo di comunicazione con le schede LAMB-FTK ed Auxiliary Board. Se le FIFO in ingresso al chip di Roadout sono piene, viene inviato al chip di Glue-FPGA il segnale di Hold to Lamb per fermare momentaneamente la trasmissione dei dati. Analogamente, se le FIFO presenti in ingresso alla Auxiliary Board sono piene, il Roadout-FPGA riceve dalla scheda Auxilary Board il segnale di Hold from Auxilary Board e viene bloccata la trasmissione dei dati in uscita, fino a quando non sono disponibili nuove locazioni di memoria.

### 7.2.1 *MONITOR* DEL FLUSSO DEI DATI CON GLI *SPY BUFFER*

Il Roadout-FPGA utilizza un importante strumento di controllo dell'integrità e della validità dei dati. Questo strumento è lo *Spy Buffer* che, di fatto, è una spia del flusso di dati. In pratica, viene fatta una copia parassita dei dati su una memoria RAM, in modo da poter controllare la loro validità, integrità e allo stesso tempo non alterare il normale flusso dei dati. Qualora sia rilevato un errore, la logica di controllo provvede a fornire segnali di errore ed a *freezare* gli *Spy Buffer* con il contenuto già memorizzato. Nel nostro caso, gli *Spy Buffer* sono gestiti tramite il protocollo VME e l'utente controlla ed effetua la diagnostica dei dati con *software* appositamente realizzati. Lo *Spy Buffer* ha 2 modalità di funzionamento, la fase *spy*, in cui i dati sono continuamente copiati nella RAM e la fase di *freeze*, in cui il contenuto della RAM viene congelato e letto tramite l'interfaccia VME della scheda AMB- FTK.

Gli *Spy Buffer* sono implementati come dei banchi di memoria RAM circolari, dove un puntatore è incrementato ogni volta che una parola arriva in ingresso e viene scritta in memoria. Il puntatore è l'indirizzo di memoria dove il nuovo dato verrà scritto. Quando la memoria va in *overflow*, le locazioni di memoria vengono sovrascritte, proprio come avviene nelle memorie circolari. All'accensione, il puntatore viene resettato a 0, e il flag di *overflow* va ad 1 subito dopo che la memoria è stata completamente scritta, così da permettere al programmatore di sapere quanti e quali dati validi sono presenti nella RAM prima della sovrascrittura.

In Figura 7.3 possiamo vedere come per ogni bus da 16 bit in uscita dalle FIFO sia previsto uno *Spy Buffer*. Lo *Spy Buffer* è gestito direttamente tramite il VME ed il programmatore può scegliere quale *Spy Buffer* leggere e monitorare.



Figura 7.3: Schema che illustra la funzione degli *Spy Buffer*. I dati raccolti su ogni bus parallelo in uscita dalle FIFO vengono copiati negli *Spy Buffer*. I dati in memoria sono letti tramite l'interfaccia VME della scheda AMB-FTK.

## Capitolo 8

# DESCRIZIONE DEI TEST DEL SISTEMA

In questo Capitolo descriveremo i test effettuati sui prototipi di schede AMB-FTK e LAMB-FTK e ci soffermeremo in maggior dettaglio sui test del *firm*ware da me sviluppato. Per programmare tutte le schede abbiamo utilizzato IMPACT, il tool proprietario della Xilinx, il programmatore della Xilinx e la catena JTAG<sup>1</sup>. Per la diagnostica delle schede programmate, è stato utilizzato il software  $ChipScope^{TM}$  Pro tool [31] che permette di utilizzare ed inserire un logic analyzer, un system analyzer e Input/Output virtuali nel progetto del dispositivo programmabile, così da poter monitorare segnali in-

<sup>&</sup>lt;sup>1</sup>Lo standard JTAG, *Joint Test Action Group*, prende il nome dal consorzio di aziende che ha dato vita a questo sistema di test per circuiti integrati. Il JTAG è nato allo scopo di isolare i dispositivi dal circuito sul quale sono montati, prendere il controllo dei pin di ingresso e di uscita per poter eettuare test sull'elettronica interna al *chip*, oppure test sulla qualità delle connessioni sul PCB. Una catena JTAG connette tutti i dispositivi programmabili presenti sia sulla scheda AMB-FTK sia sulla scheda LAMB-FTK.

terni alla FPGA e il funzionamento della logica implementata. I test sono stati suddivisi in due fasi. Nella prima fase sono stati testati i *firmware* tramite opportune simulazioni *behavioural* e *post-fitting* sul simulatore della suite ISE, verificando il funzionamento della logica implementata. Successivamente, i test sono stati effettuati direttamente sulle schede. In Figura 8.1 è visibile la postazione di test utilizzata nel laboratorio FTK della Sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare.



Figura 8.1: Postazione di test disponibile nel laboratorio di FTK. Sulla destra, è possibile notare il crate VME con una scheda AMB-FTK e una scheda LAMB-FTK montata, collegate tramite programmatore Xilinx ad un pc, ed una scheda contenente la CPU, per comunicare con la scheda tramite protocollo VME.

Per effettuare i test del sistema, sono state eseguite le seguenti operazioni:

• Sono stati programmati gli FPGA e CPLD tramite le catene JTAG presenti sulla scheda LAMB-FTK e sulla scheda AMB-FTK.

- Sono stati generati e memorizzati i *pattern* nei *chip* di memoria associativa.
- Sono state generate liste di *bin* colpiti casuali e sono state trasmesse ai *chip* di memoria associativa per effettuare le operazioni di *pattern recognition*.

I nostri test di funzionamento del sistema ed, in particolare, della logica del Glue-FPGA e del Roadout-FPGA, si sono concentrati sul controllo dell'integrità dei dati trasmessi dai *chip* di memoria associativa. Il test controlla che la lista di *road* trovate e delle relative *bitmap* coincide con la lista ottenuta dalla simulazione (Figura 8.2).



Figura 8.2: Schema dei test effettuati. Tramite VME, la CPU scrive tutti i *pattern* nei *chip* di memoria associativa, genera un lista di *bin* colpiti casuali e li invia ai *chip* di memoria associativa. Le *road* in uscita giungono ai Glue-FPGA e, successivamente, ai Roadout-FPGA.

### 8.1 TEST DEL GLUE-FPGA

Un primo test del funzionamento del Glue-FPGA preso singolarmente ci ha permesso di valutare attentamente quali potessero essere i punti critici della programmazione ed apporre le necessarie correzioni.

In una prima fase, per testare il Glue-FPGA si è operato in questo modo. Abbiamo memorizzato i *pattern* sui *chip* di memoria associativa, abbiamo generato e trasmesso i *bin* colpiti sulla scheda LAMB-FTK e quindi ai *chip* di memoria associativa e, se c'era corrispondenza fra il dato precedentemente memorizzato e quello generato, si avevano *road* e *bitmap* in uscita dalla catena di *chip* di memoria associativa (Figura 8.2). Questi dati, come abbiamo visto nel Capitolo 6, sono trasmessi al Glue-FPGA dall'ultimo *chip* di memoria associativa della catena. Per monitorare il completo funzionamento del Glue-FPGA, abbiamo monitorato ogni punto possibile, inserendo il *ChipScope* nei punti indicati nella Figura 8.3 ed in alcuni punti interni al *Merger*. Solo grazie a *Chipscope* è stato possibile testare la logica interna.



Figura 8.3: Schema del test sul Glue-FPGA. Sono messi in evidenza i punti in cui è stato inserito il *logic analyzer ChipScope* per controllare l'integrità dei dati.

Con questi dati in ingresso, opportuni *testbench* e con l'ausilio dei *tool*, abbiamo quindi verificato:

- il funzionamento del blocco funzionale *Controller*, interno al *Merger*. In questa maniera abbiamo verificato che, in assenza di dati dai *chip* di memoria associativa, la macchina a stati finiti aspetta all'incirca 20 cicli di *clock*, dopo i quali spedisce la parola di *End Event*.
- la trasmissione dati effettuata da una singola catena di *chip* di memoria associativa, per verificare la trasmissione da parte dei *GTP Transceiver* sui link seriali dalla scheda LAMB-FTK alla scheda AMB-FTK.

- la funzione del *Merger*. Abbiamo controllato la capacità di convogliare in un unico bus parallelo in uscita i dati ricevuti in ingresso su due bus paralleli dando priorità alla catena *top*.
- la capacità della logica di accrescere le informazioni tramite l'aggiunta di 2 bit più siginificativi nella sequenza di 14 bit provenienti dal *chip* di memoria associativa.

In una seconda fase, nel momento in cui sarà disponibile una scheda LAMB-FTK con numerosi *chip* di memoria associativa, sarà testato completamente la logica del Glue-FPGA. Ogni Glue-FPGA, come abbiamo spiegato, deve gestire le informazioni di 16 *chip* di memoria associativa. Al momento però non sono disponibili i 32 *chip* e non è possibile valutare completamente il funzionamento del *firmware*.

### 8.2 TEST DEL ROADOUT-FPGA

Una volta testato il funzionamento del Glue-FPGA e della trasmissione dei dati alla scheda AMB-FTK, è stato effettuato un test sul Roadout-FPGA. Anche in questo caso, è stato utilizzato il tool *ChipScope*, inserendolo nei punti indicati in Figura 8.4.



Figura 8.4: Schema della logica interna al Roadout-FPGA con i punti in cui è stato inserito il *ChipScope*.

Il test su Roadout-FPGA ci ha permesso di controllare:

- il funzionamento dei *link* seriali a 2 Gb/s tra le schede LAMB-FTK e i Roadout-FPGA della scheda AMB-FTK. In particolare, abbiamo avuto la conferma di come la trasmissione dei dati funziona correttamente alla frequenza indicata, senza che alcun dato venga perso.
- la logica interna del Roadout-FPGA, la parallelizzazione dei dati ricevuti in seriale.
- l'integrità e validità dei dati pronti ad esser trasmessi alla Auxiliary Board dell'University of Chicago.

• il funzionamento degli Spy Buffer.

Abbiamo verificato, inoltre, il corretto funzionamento del connettore P3 con la trasmissione dei dati a 2 GHz, utilizzando un *logic analyzer* equivalente della Altera sulla *Auxiliary Board*. Abbiamo verificato che i dati memorizzati nello *Spy Buffer* e letti su file o tramite *ChipScope* coincidevano con i dati inviati dalla scheda LAMB-FTK, anche questi precedentemente monitorati, come detto nella sezione 8.1 ed 8.2. Al momento, i test effettuati, seppur intensi, hanno riguardato solo una piccola quantità di dati, non essendo attualmente disponibili un numero elevato di *chip* di memoria associativa.
## Capitolo 9

## CONCLUSIONI

Il lavoro descritto in questa Tesi è stato svolto tra i mesi Novembre 2012 e Giugno 2013 presso la Sezione di Pisa dell'Istituto Nazionale di Fisica Nucleare, dove è in fase di progettazione e realizzazione il nuovo processore *Fast Tracker* (FTK) che sarà utilizzato nella presa dati dell'esperimento ATLAS al CERN prevista per l'anno 2015.

FTK è stato ideato per ricostruire in tempo reale la traiettoria delle particelle cariche che attraversano il rivelatore in silicio di ATLAS e migliorare così le prestazioni del sistema di *trigger* e di acquisizione dati dell'esperimento. Il componente fondamentale per il funzionamento di FTK è il *chip* di memoria associativa. Questa è una memoria particolare, differente da una normale memoria. Infatti, a differenza di un normale *chip* di memoria in cui si fornisce un'indirizzo e si legge il dato registrato in quella locazione, nel *chip* di memoria associativa si fornisce in ingresso un dato, e se questo coincide con un dato precedentemente memorizzato, si ottiene in uscita l'indirizzo di memoria corrispondente. In FTK questo dispositivo è utilizzato per svolgere la funzione di *pattern recognition*. I dati in ingresso al *chip* di memoria associativa sono la lista di *bin* colpiti nei vari strati del rivelatore, i dati in uscita sono la lista delle candidate tracce ricostruite a risoluzione ridotta, dette *road*.

In questo lavoro di Tesi, ho dato un contributo significativo allo sviluppo del microcodice di programmazione di alcuni FPGA utilizzati sulla scheda di memoria associativa (AMB-FTK) e sulla scheda *Little Associative Memory Board* (LAMB-FTK), che ospita i *chip* di memoria associativa. Ogni scheda di memoria associativa ospita 4 schede LAMB-FTK.

In particolare, ho sviluppato il microcodice di programmazione per il Glue-FPGA, utilizzato sulla scheda LAMB-FTK per raccogliere le *road* trovate in ogni evento dai *chip* di memoria associativa e trasmetterle ai Roadout-FPGA, che si trovano sulla scheda di memoria associativa.

In seguito, ho contribuito allo sviluppo del *firmware* del Roadout-FPGA utilizzato sulla scheda AMB-FTK per ricevere la lista delle *road* trovate dai *chip* di memoria associativa e trasmetterle alla *Auxilary Board*, per effettuare la ricostruzione delle tracce. I test hanno dimostrato che sappiamo usare i link seriali e trasferire i dati ad alta frequenza senza problemi.

Per quanto riguarda gli sviluppi futuri del mio lavoro, possiamo dire che sono in fase di progetto e realizzazione due nuove versioni della scheda AMB-FTK e della scheda LAMB-FTK, con novità sia per quanto riguarda l'*hardware* sia per quanto riguarda il *firmware*, quali l'introduzione di nuovi e potenti FPGA che permetteranno di migliorare la gestione del flusso di dati. Dovremo quindi rivedere i codici VHDL attualmente esistenti, per adattarli alle nuove versioni delle due schede.

## Elenco delle figure

1.1	Veduta aerea del CERN di Ginevra. Sono stati messi in evi-	
	denza il tracciato del Large Hadron Collider, i punti in cui si	
	intersecano e collidono i fasci e sono collocati gli esperimenti	
	ATLAS, ALICE, CMS, LHCb, TOTEM ed LHCf	12
1.2	Geometria dei fasci in corrispondenza di una regione di inte-	
	razione	13
1.3	Ricostruzione delle traiettorie delle particelle prodotte in un	
	bunch crossing ad LHC.	14
1.4	Rappresentazione schematica di LHC e della catena completa	
	di acceleratori. Sono inoltre indicate le posizioni di ATLAS,	
	CMS, ALICE ed LHCb.	15
1.5	Veduta schematica di ATLAS. Sono indicati i componenti prin-	
	cipali del rivelatore	17

1.6	ATLAS utilizza un sistema di coordinate destrorso. L'asse $\boldsymbol{x}$	
	punta al centro del tunnel di LHC, l'asse $\boldsymbol{y}$ punta verso l'alto,	
	l'asse $\boldsymbol{z}$ indica la direzione del fascio. Il verso positivo dell'asse	
	z è definito dal fascio che circola nel tunnel in senso antiorario.	19
1.7	Suddivisione convenzionale del rivelatore in quattro regioni,	
	barrel, transition, end-cap e forward, in funzione della pseudo-	
	rapidità	20
1.8	A sinistra, geometria degli avvolgimenti dei magneti: sono	

	visibili il solenoide (in viola), le 8 bobine del magnete toroi-
	dale del $barrel$ alternate alle bobine della regione di $end$ - $cap$
	(in rosso). A destra, veduta dei toroidi superconduttori del-
	la regione del barrel. La scala dimensionale è indicata dalla
	persona fotografata tra le due bobine in basso
1.9	Il sistema di tracciatura di ATLAS. Sono indicati il rivelatore
	a <i>pixel</i> , a <i>microstrip</i> centrale e in avanti ed il tracciatore a ra-
	diazione di transizione, nella regione del <i>barrel</i> e nella regione

	di <i>end-cap</i>	21
1.10	Sezione longitudinale del sistema di tracciatura. Sono indicati	
	il rivelatore in silicio a <i>pixel</i> e ad <i>microstrip</i> , ed il rivelatore a	
	radiazione di transizione	22
1.11	Rappresentazione schematica del rivelatore a $pixel$ di silicio.	

 rappiesentatione senematica del interactica a pineto di sinero.	
Si notano gli strati concentrici della regione del $barrel$ ed i	
dischi della regione di <i>end-cap</i>	23

1.12	Lo schema mostra i rivelatori e gli elementi strutturali attra-	
	versati da una traccia carica di p $_T=10~{\rm GeV/c}$ nel barrel. La	
	traccia attraversa la <i>beam-pipe</i> di berillio, i 3 strati cilindrici	
	di <i>pixel</i> , i 4 strati cilindrici di <i>microstrip</i> in silicio, ed appros-	
	simativamente 36 tubi assiali del tracciatore a radiazione di	
	transizione.	26
1.13	Nell'immagine sono mostrate due tracce cariche di $p_T = 10$	
	${\rm GeV/c}$ che attraversano la zona del $barrel$ e quella di $end\mathcharce cap.$	26
1.14	Il calorimetro di ATLAS. Sono indicati i principali componenti	
	della sezione elettromagnetica e di quella adronica	28
1.15	Struttura ad organetto del calorimetro elettromagnetico	29
1.16	Sezione trasversale dei vari rivelatori utilizzati in ATLAS ed	
	illustrazione della interazione con i diversi tipi di particel-	
	le. La linea tratteggiata indica che la particella attraversa	
	il rivelatore senza interagire ed è, di fatto, invisibile	32
1.17	Rappresentazione schematica dell'architettura in 3 livelli del	
	trigger di ATLAS, e del sistema di acquisizione dati. So-	
	no riportate le frequenze massime di acquisizione degli eventi	
	consentite per ogni livello di <i>trigger</i>	36
21	Bappresentazione schematica del sistema di acquisizione dati	
<i>-</i> .+	e del <i>triager</i> di ATLAS suddiviso in tre livelli FTK si colloca	
	tra gli attuali Livalla 1 a Livalla 2	20
		99

2.2	Configurazione degli strati nel rivelatore in silicio di ATLAS.
	Si possono riconoscere i 3 strati a $pixel$ nella regione interna
	(azzurro, verde, blu) e i 4 strati a $\mathit{microstrip}$ nella regione più
	esterna

2.3 (a) Veduta in sezione di una porzione di rivelatore a microstrip.
Si può notare il campo elettrico interno, che trascina la carica in prossimità di un canale di lettura, che raccoglie la carica e genera un segnale elettrico, letto da opportuna elettronica.

(b) Altra veduta in sezione di un rivelatore a *microstrip*. . . . 41

2.6	(a) Ogni strato del rivelatore è suddiviso in $bin$ (in verde). (b)	
	La presenza di un $hit$ (blu) attiva l'intero $bin$ nel quale è con-	
	tenuto. (c) La combinazione di più <i>bin</i> , uno per ogni <i>layer</i> , de-	
	termina una $road$ o $pattern$ riconducibile alla traiettoria reale	
	della particella.	43
2.7	Struttura di FTK, composta da 8 <i>Core Processor</i> ospitati in 8	
	Core Crate. Ogni Core Crate contiene 16 Processing Unit, per	
	un totale di 128 Processing Unit. I dati di una singola torre	
	sono elaborati da una coppia di <i>Processing Unit.</i>	46
2.8	Schema di un Core Crate. Ogni Core Crate ospita 16 Proces-	
	$sing\ Unit.$ I dati provenienti da una singola torre del rivelatore	
	sono elaborati da una coppia di <i>Processing Unit.</i>	46
2.9	Immagine di un $Core \ Crate$ VME che può ospitare un $Co$ -	
	re Processor. Nel crate sono disponibili le locazioni per il	
	montaggio della CPU e delle 16 <i>Processing Unit.</i>	47
2.10	Schema semplificato di un Core Processor di FTK. Ogni Core	
	Processor è costituito da 16 Processing Unit identiche. Ogni	
	coppia di Processing Unit elabora i dati provenienti da una	
	torre del rivelatore. Nello schema sono indicati i diversi blocchi	
	funzionali che compongono una Processing Unit.	48

- 2.11 Suddivisione del rivelatore in regioni. In rosso è messa in evidenza la regione di sovrapposizione tra due regioni adiacenti.
  I dati della regione di sovrapposizione sono spediti simultaneamente alle *Processing Unit* corrispondenti alle due regioni adiacenti.
  49
- 2.12 Schema delle funzioni svolte da FTK. Il Data Organizer riceve la lista di *hit* trovati nel rivelatore in un evento e trasmette la lista dei *bin* colpiti. Questa lista viene confrontata con i *pattern* memorizzati nei *chip* di memoria associativa. Il Data Organizer riceve le *road* trovate nell'evento, ed invia *hit* e *road* al *Track Fitter* che esegue il *fit* delle tracce a piena risoluzione. 51

3.3	Schema della scheda AMB-FTK, in cui sono messi in eviden-	
	za la posizione e le connessioni di Control-FPGA, che è in	
	pratica il "cervello" della scheda AMB-FTK. In verde è in-	
	dicato il percorso dei dati verso il connettore utilizzato per la	
	comunicazione tra la scheda LAMB-FTK e AMB-FTK. In ros-	
	so sono evidenziati le connessioni tra Control-FPGA e Pixel-	
	FPGA e SCT-FPGA. In blu sono mostrati i collegamenti con i	
	Roadout-FPGA. In arancio è evidenziata la connessione verso	
	la Auxiliary Board.	59
3.4	Schema della scheda AMB-FTK: in rosso, è indicato il percorso	
	dei dati verso le LAMB-FTK a partire da SCT-FPGA e Pixel-	
	FPGA, che ricevono le liste dei $bin$ colpiti sul rivelatore dalla	
	Auxiliary Board	60
3.5	In blu sono messi in evidenza i Roadout-FPGA sulla scheda	
	AMB-FTK. Ogni Roadout-FPGA riceve i dati da 2 schede	
	LAMB-FTK	62
3.6	Schema delle connessioni seriali e dei bus paralleli in uscita	
	da SCT-FPGA e Pixel-FPGA verso i connettori con le schede	
	LAMB-FTK. In rosso sono illustrati i collegamenti su link se-	
	riale, in verde sono illustrati i collegamenti su bus paralleli. In	
	blu sono illustrati i dispositivi Micrel, utilizzati per rigenerare	
	il segnale.	64

- 3.7 Schema dei generatori e rete di distribuzione del segnale di clock sulla scheda AMB-FTK. In giallo, abbiamo messo in evidenza l'oscillatore a 100 MHz; in verde, i generatori di clock per i GTP delle FPGA; in viola, il *RoboClock*; in blu, la rete di collegamento ed i segnali di clock *single ended*; in rosso, la rete di collegamento ed i segnali di clock differenziali. . . . . 66
- 4.1 La struttura del chip di memoria associativa può essere ricondotta ad una matrice bidimensionale, nella quale ogni riga rappresenta un pattern di memoria ed è composta da 8 celle CAM, una per ogni strato del rivelatore. Alle colonne della matrice corrispondono i bus sui quali scorrono i dati. . . . . . 71

4.4	Immagine della LAMB-FTK. A sinistra è visibile il lato $bottom$	
	della scheda ed è messo in evidenza il connettore attraverso il	
	quale transitano i dati. A destra è visibile il lato $top$ e sono	
	evidenziati i <i>chip</i> che raccolgono e distribuiscono i dati (INDI,	
	SCAN, GLUE) e una locazione in cui è installato un $\mathit{chip}$ di	
	memoria associativa.	74
4.5	Schema della scheda LAMB-FTK. Sono messe in evidenza i	
	connettori per i due tipi di alimentazioni necessarie del $chip$	
	di memoria associativa. In particolare, in viola è evidenziato il	
	${\rm connettore\ per\ l'I/O\ Voltage\ (3.3\ V),\ necessaria\ per\ alimentare}$	
	l'interfaccia input/output del $\mathit{chip.}$ In giallo è mostrata la	
	posizione del connettore per il $Core$ Voltage (1.2 V), necessaria	
	per il funzionamento del <i>chip</i>	76
4.6	I CPLD, posti al centro della LAMB-FTK, distribuiscono su	
	un bus parallelo una parte della lista di $bin$ colpiti ai $chip$ di	
	memoria associativa.	77
4.7	Gli FPGA Spartan-6 posti al centro della LAMB-FTK distri-	
	buiscono su link seriali una parte della lista di $bin$ colpiti ai	
	chip di memoria associativa	78
4.8	Schema di collegamento delle uscite dei $chip$ di memoria asso-	
	ciativa alle 2 Glue-FPGA, evidenziate nei quadrati verdi	79

4.9	(a) Primo prototipo del $chip$ di memoria associativa realizza-	
	to nell'anno 1992 per il processore SVT utilizzato nell'espe-	
	rimento CDF di Fermilab. (b) Secondo prototipo mappato	
	su FPGA della casa produttrice Xilinx realizzato nell'anno	
	1998. (c) Terzo prototipo <i>standard cell</i> realizzato nell'anno	
	2003 per l'upgrade del processore SVT. (d) Quarto prototipo	
	$standard\ cell\ realizzato\ nell'anno\ 2012\ per\ il\ processore\ FTK$	
	all'esperimento ATLAS del CERN.	81
5.1	Rappresentazione schematica di una memoria FIFO. Sono mo- strati i due domini di clock indipendenti e i segnali di input	
	ed output.	87
5.2	Temporizzazione della fase di scrittura di una FIFO	88
5.3	Temporizzazione della fase di lettura di una FIFO	89
5.4	Schermata generale di un generatore di FIFO Intellectual Pro-	
	perty Core. Nella suite di programmazione della Xilinx è	
	previsto un Core Generator per ogni Intellectual Property Core.	90
5.5	Diagramma a blocchi semplificato di un GTP Transceiver del-	
	la famiglia Spartan-6	92

5.6	Due GTP Transceivers in una GTPA1 DUAL Tile. Al centro,	
	il blocco del clock, risorsa comune tra i due Transceiver. In	
	azzurro, sono evidenziati i bus paralleli in ingresso al $\mathit{trasmitter}$	
	ed in uscita dal <i>receiver</i> e un particolare segnale di controllo,	
	txcharisk. In giallo, sono messi in evidenza il clock in ingresso	
	clkine il PLL condiviso tra due GTP. In verde, sono mostrati	
	i segnali di clock utilizzati dai GTP. In rosso, le uscite $Tx$ e	
	gli ingressi $Rx$ differenziali	94
5.7	Diagramma a blocchi di un GTP Trasmitter.	95
5.8	Codifica 8B/10B	97
5.9	Diagramma a blocchi di un GTP Receiver	98
5.10	Decodifica $8\mathrm{B}/10\mathrm{B}$ dei dati seriali che giungono in ingresso al	
	receiver.	99
6.1	Schema della scheda LAMB-FTK. Nel blocco rosso, sono evi-	
	denziati i CPLD che distribuiscono i dati a i $\mathit{chip}$ di memoria	
	associativa su bus paralleli, nel blocco giallo le Spartan-6 che	
	ricevono i dati su link seriale e li distribuiscono su bus paralle-	
	li a i $\mathit{chip}$ di memoria associativa , nei quadrati verde-azz urro	
	le due Glue-FPGA che raccolgono le $road$ trovate dai $chip$ di	
	memoria associativa.	03

6.2	Schema semplificato di una scheda LAMB-FTK. Il chip de-
	nominato GLUE-FPGA ha la funzione di raccogliere le $road$
	trovate nei $chip$ di memoria associativa e di trasmetter le al-
	la Auxilary Board attraverso i chip Roadout-FPGA collocati
	sulla scheda AMB-FTK

6.5	Sulla sinistra è mostrata la scheda AMB-FTK con tutti i suoi
	FPGA. Il Control-FPGA genera un segnale di <i>Init</i> che viene
	inviato a basso fanout a tutti i chip. In base alla sua durata,
	il segnale di $Init$ viene interpretato o come un segnale di $Init$
	Event oppure come un segnale di $Reset$ globale. Il segnale di
	Init viene ricevuto dall'unità logica Reset - Init Event Reco-
	gnizer (Figura a destra), presente in ogni FPGA della scheda.
	In uscita il segnale è duplicato: in blu è illustrato il segnale di
	Init Event, in nero è illustrato il segnale di Reset globale 109
6.6	Struttura logica del Reset-Init Event Recognizer
6.7	Rappresentazione schematica della logica utilizzata per gesti-
	re il flusso dei dati ricevuto dai <i>chip</i> di memoria associativa
	all'interno della Glue-FPGA
6.8	Esempio di temporizzazione della fase di trasmissione dati
	dal <i>chip</i> di memoria associativa alla Glue-FPGA. Nella Fi-
	gura, è illustrata la trasmissione di due $road$ e della $bitmap$
	corrispondente
6.9	Struttura interna del Merger. Sono visibili i due blocchi Con-
	troller, e il blocco indicante la Macchina a Stati Finiti, per la
	gestione dei dati verso il GTP
6.10	Flusso di dati in uscita dal Merger. Il Merger aggiunge in cima
	2 bit ai 14 bit di una parola memorizzata nella FIFO 117

6.11	Flusso dei dati sul bus che trasmette le informazioni dal $Mer$ -
	geral GTP. Due eventi successivi sono separati dalla parola
	di <i>End Event</i> e dalla <i>K-word</i> "50BC"119

- indicato il dato in uscita che il merger pone su txdata. . . . . . 125
- 7.1 Schema semplificato che mostra i 4 link seriali in uscita da ogni LAMB-FTK e i due Roadout-FPGA che raccolgono 8 link seriali ciascuno.
  7.2 Architettura logica del Roadout-FPGA.

7.3	Schema che illustra la funzione degli $Spy Buffer$ . I dati raccolti
	su ogni bus parallelo in uscita dalle FIFO vengono copiati negli
	$Spy \ Buffer.$ I dati in memoria sono letti tramite l'interfaccia
	VME della scheda AMB-FTK

- 8.1 Postazione di test disponibile nel laboratorio di FTK. Sulla destra, è possibile notare il crate VME con una scheda AMB-FTK e una scheda LAMB-FTK montata, collegate tramite programmatore Xilinx ad un pc, ed una scheda contenente la CPU, per comunicare con la scheda tramite protocollo VME. 138

## Bibliografia

- Lyndon Evans, (ed.), Philip Bryant, (ed.) (CERN), (2008) "LHC Machine", Published in JINST 3 (2008) S08001 164 pp. Lyndon Evans and Philip Bryant (2008) "THE CERN LARGE HADRON COLLIDER: ACCE-LERATOR AND EXPERIMENTS", published by Institute of Physics Publishing and SISSA
- [2] http://press.web.cern.ch/press-releases/2012/12/first-lhc-protons-runends-new-milestone
- [3] aa. vv. LHC Commissioning with Beam, Web page: http://lhccommissioning.web.cern.ch/lhc-commissioning/
- [4] ATLAS Collaboration, G. Aad (Marseille, CPPM) et al. (2008), "The ATLAS Experiment at the CERN Large Hadron Collider", Published in JINST 3 (2008) S08003 437 pp.
- [5] CMS Collaboration, S. Chatrchyan et al. (Aug 2008) "The CMS experiment at the CERN LHC", Published in JINST 3 (2008) S08004 361 pp.

- [6] ALICE Collaboration, K. Aamodt (Oslo U.) et al.. (2008) "The ALICE experiment at the CERN LHC", Published in JINST 3 (2008) S08002 259 pp.
- [7] LHCb Collaboration, A.Augusto Alves, Jr. (Rio de Janeiro, CBPF) et al. (2008) "The LHCb Detector at the LHC ", Published in JINST 3 S08005 217 pp.
- [8] LHCf Collaboration, O. Adriani (Florence U. & INFN, Florence) et al. (2008) "The LHCf detector at the CERN Large Hadron Collider" Published in JINST 3 (2008) S08006 39 pp.
- [9] TOTEM Collaboration, G. Anelli (CERN) et al. (2008) "The TOTEM experiment at the CERN Large Hadron Collider", Published in JINST 3 (2008) S08007 112 pp.
- (1999)"ATLAS: [10] ATLAS Collaboration Detecphysics performance torand technical designre-I" port, Vol. CERN-LHCC-99-014, ATLAS-TDR-14, http://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/TDR/access.html
- [11] The ATLAS Collaboration (1997) "ATLAS Inner Detector Technical Design Report vol.1" CERN/LHCC/97-1
- [12] P. Pralavorio (2002) "The ATLAS Liquid Argon electromagnetic calorimeter", AIP Conf. Proc. 549 pp. 872-874

- [13] The ATLAS Calorimeter Community (1996) "ATLAS Tile Calorimeter Technical Design Report" ATLAS TDR-2 CERN/LHCC/96-42
- [14] The ATLAS Muon Collaboration (1997) "ATLAS Muon Spectrometer Technical Design Report", ATLAS TDR-10 CERN/LHCC/97-22
- [15] ATLAS Collaboration (2003) "The ATLAS high-level trigger, data acquisition and controls: Technical design report," LHCC 2003-022, CERN
- [16] ATLAS Collaboration (1998) "The ATLAS Level-1 Trigger: Technical Design Report" LHCC 1998-14, ATLAS-TDR-12, CERN
- [17] A.Annovi et al. (2011) "FTK: a hardware track finder for the ATLAS trigger Technical Proposal"
- [18] A.Andreani et al. (2010) "The FastTracker Real Time Processor and Its Impact on Muon Isolation, Tau and b-Jet Online Selections at ATLAS" Conference Record 2010 17th IEEE-NPSS Real Time Conference, s.l, IEEE
- [19] J. Anderson et al.(2012) "FTK: A fast track trigger for ATLAS", Published in JINST 7 (2012) C10002
- [20] H.C. van der Bij et al. "S-LINK: A Prototype of the ATLAS Read-out Link" Fourth Workshop on Electronics for LHC Experiments, Rome, 21-25 September 1998. http://hsi.web.cern.ch/HSI/slink/introduc/introduc.htm

- [21] M.Dell'Orso, L.Ristori (1990) "A HIGHLY PARALLEL ALGORITHM FOR TRACK FINDING" Nuclear Instruments and Methods A287, 436 http://www.sciencedirect.com/science/article/pii/016890029091559T
- [22] M.Dell'Orso, L.Ristori (1989) "VLSI STRUCTURES FOR TRACK FINDING" Nuclear Instruments and Methods A278,436 http://www.sciencedirect.com/science/article/pii/0168900289908620
- [23] User's Manual, Series 6000 VME, -64x, -64xC, -64xP, VXI, W-Ie-Ne-R, November 2006
- [24] S.Citraro (2012) "Progetto di una scheda di Memoria Associativa basata su link seriali per il processore Fast Tracker all'esperimento ATLAS del CERN"
- [25] CDF Collaboration, Bill Ashmanskas et al. (Jun 2003), "The CDF silicon vertex trigger", Published in Nucl.Instrum.Meth. A518 (2004)
   532-536 FERMILAB-CONF-03-168-E
- [26] J. Adelman, A. Annovi et al. (2006) "The Silicon Vertex Trigger upgrade at CDF" Pisa meeting on Advanced Detectors, May 21 - 27, Isola d'Elba, Italy
- [27] F. Morsani et al. (1992) "The AMchip: a Full-custom MOS VLSI Associative memory for Pattern Recognition", IEEE Trans. on Nucl. Sci., vol. 39, pp. 795-797.

- [28] P. Giannetti et al. (1998) "A Programmable Associative Memory for Track Finding", Nucl. Intsr. and Meth., vol. A413/2-3, pp. 367-373
- [29] L. Sartori, A. Annovi et al. (Aug. 2006) "A VLSI Processor for Fast Track Finding Based on Content Addressable Memories", IEEE Transactions on Nuclear Science, Volume 53, Issue 4, Part 2 Page(s):2428 -2433
- [30] http://www.xilinx.com/products/silicon-devices/fpga/spartan-6/index.htm
- [31] http://www.xilinx.com/tools/cspro.htm