INAF - Osservatorio Astrofisico di Arcetri
INAF - I.R.A.

Progetto Fasti

Manuale d'uso del Software - II
Il programma di plot e le interfacce grafiche

E. Giani, C. Baffa

Rapporto Interno di Arcetri N° **2/2004**
Firenze, Settembre 2004

Sommario.

Per lo sviluppo e l'uso dell'elettronica di controllo per rivelatori infrarossi bidimensionali Fasti *sono stati sviluppati diversi strumenti software. Questo rapporto ne descrive l'uso e fornisce le ulteriori informazioni necessarie per il test e l'utilizzo di* Fasti. *Il presente rapporto interno è il secondo di una serie in cui vengono descritti tutti gli strumenti software di* Fasti. *Nel seguito verranno descritti il programma di visualizzazione delle forme d'onda,* **GsvbPlot**, *il programma di visualizzazione di un flusso ininterrotto di dati* **FreeRun**, *il programma di controllo remoto* **GClient**. *In un precedente rapporto[2] sono stati descritti sia il linguaggio che il compilatore (***svb1asm***) dello assembler per il sistema di generazione di sequenze per* Fasti, *lo* SVB1, *che l'uso dell'emulatore software dello* SVB1, **EmuSvb**. *In un rapporto interno successivo verranno descritti il programma di controllo generale,* **Ftest**, *e lo standard di scrittura delle forme d'onda per* NICS.

Capitolo 1

Introduzione

Il gruppo infrarosso di Arcetri ha realizzato una elettronica *leggera* per l'acquisizione dati con rivelatori bidimensionali infrarossi, Fasti. La caratteristica di Fasti è l'appoggiarsi più su standard industriali e *disegni concettuali* che su devices specifici, per la facilità di sviluppo e per evitare una prematura obsolescenza. La struttura dettagliata di Fasti si può trovare in diversi documenti[1, 4, 5].

Capitolo 2

Il programma GClient

Il programma **GClient** è stato sviluppato inizialmente con il preciso scopo di verificare il funzionamento del protocollo di comunicazione con l'applicazione di controllo **Ftest** quando questa è eseguita in modalità demone.

Nella prima fase di sviluppo, il programma era una semplice applicazione interattiva: al prompt veniva inserito il comando del protocollo e si verificava che questo fosse stato ricevuto correttamente.
Successivamente si è pensato di creare un programma più articolato, con un'interfaccia grafica semplice ma completa che consentisse di effettuare tutte le operazioni di acquisizione con l'elettronica di Fasti.

GClient non è perciò un software destinato all'utente-astronomo, ma un'applicazione che può essere ampiamente utilizzata in laboratorio per l'analisi della catena di acquisizione di Fasti.

2.1 Interazione Ftest GClient

L' applicazione **Ftest** nella sua versione finale per il telescopio, è una classica applicazione *server* eseguita sulla macchina *embedded* `fasti1`[6]. L'applicazione **Ftest** esegue all'avvio l' inizializzazione dei vari sottosistemi ed infine crea la parte server per la connessione da parte dell'applicazione client.
In particolare vengono aperte due socket (una per i comandi l'altra per i dati) in corrispondenza della porta 8081 e su tale porta il server si pone in ascolto in attesa di una connessione da parte di un'applicazione cliente. Quest'ultima, a sua volta, viene costruita in modo da aprire due socket ed eseguire una connessione al server specificandone l'indirizzo IP e la porta (8081) a cui intende connettersi.
Se la precedura descritta non ha originato errori, la connessione tra le due applicazioni di rete è attiva e l'utente è in grado di eseguire il test dell'elettronica inviando i comandi eseguibili dall'interfaccia di **GClient**.

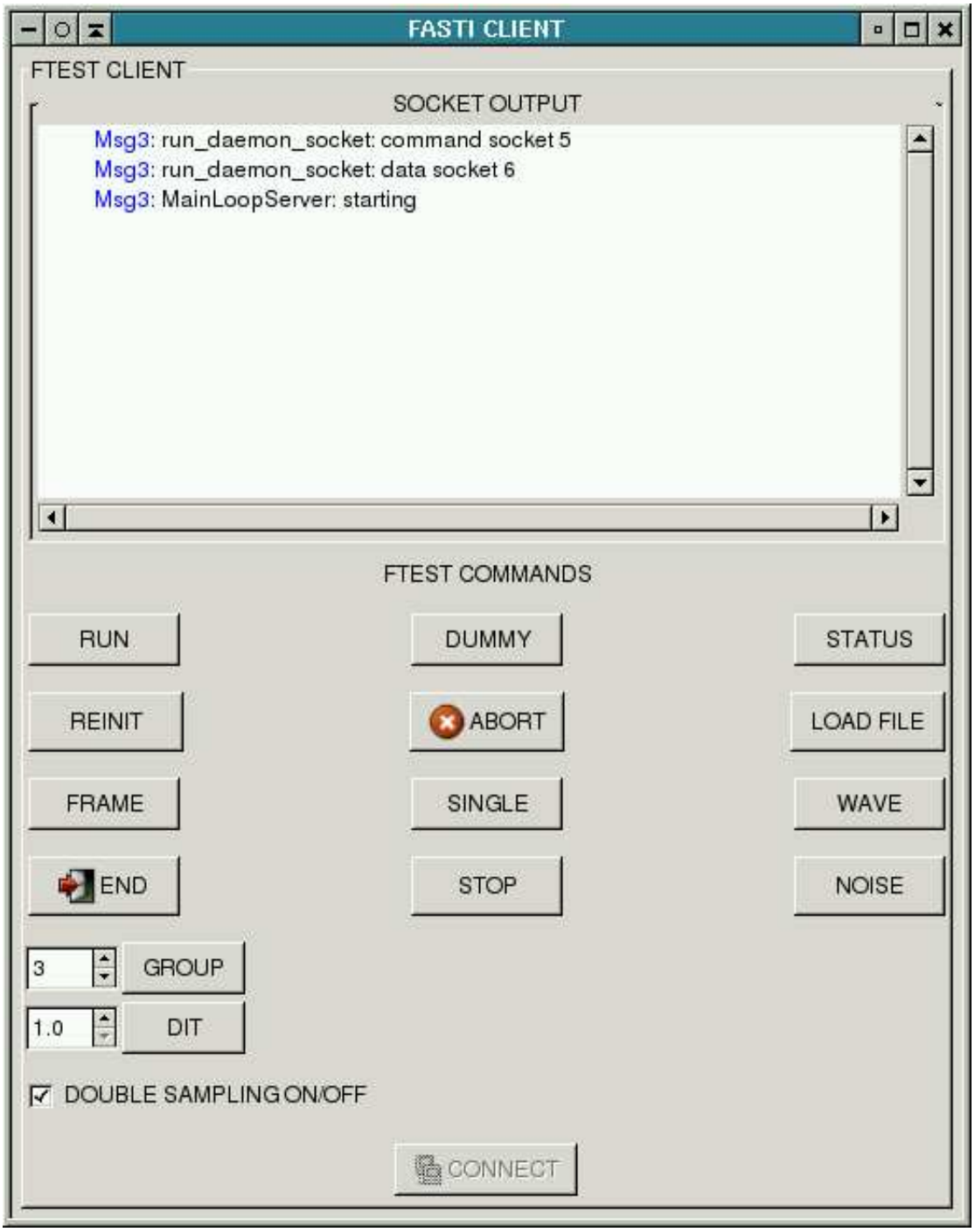

Figura 2.1: Finestra principale

Figura 2.2: Messaggio iniziale

Figura 2.3: Messaggio d'errore

2.2 Esecuzione di GClient

Il programma **GClient** viene eseguito lanciando il seguente comando:

```
gclient  <hostname>
```

dove `hostname` è il nome della macchina su cui viene eseguito il programma di controllo **Ftest** in modalità demone. Se l'utente non specifica alcun argomento, viene assunto come valore di default per *hostname* `fasti1`.
Quando il demone **Ftest** è regolarmente in esecuzione sul sistema embedded, sul monitor si apre la finestra principale del programma **GClient** (vedi Fig. 2.1) ed insieme ad essa appare una finestra di messaggio con il nome del file su cui verranno salvati i dati della successiva acquisizione (vedi Fig. 2.2).
Nel caso in cui il demone **Ftest** non sia attivo, l' utente viene avvisato da un messaggio del tipo di Fig. 2.3, dove 8081 è, come specificato in precedenza, la porta sulla quale il server **Ftest** aspetta le connessioni da parte del programma cliente.
In quest'ultimo caso, la finestra principale viene comunque visualizzata, ma, ovviamente, nessuno dei comandi attivati dai tasti viene eseguito.
I primi messaggi ricevuti dall' applicazione **GClient** dopo l'avvenuta connessione al demone **Ftest** sono i seguenti:

```
Msg3: run_daemon_socket: command socket 5
Msg3: run_daemon_socket: data socket 6
```

```
Msg3: MainLoopServer: starting
```

L' ultimo messaggio indica che il programma demone è pronto a ricevere ed eseguire i comandi inviati da **GClient**. Per una descrizione dei messaggi rimandiamo a § 2.3.

2.3 Formato dei messaggi

I messaggi mostrati a schermo sono composti di tre parti:

- la prima parte indica se si tratta di un messaggio di avvertimento (`Msg`) o di un errore (`Err`). Per distinguere meglio i due casi, le intestazioni vengono evidenziate con colori diversi: in blu il messaggio, in rosso l'errore
- la seconda parte specifica la routine del programma che ha originato il messaggio o l' errore
- la terza parte è il testo del messaggio.

In particolare, gli avvisi con intestazione `Msg` sono seguiti da un numero che rappresenta il livello di severità del messaggio.
Il programma **GClient** essendo essenzialmente un'applicazione destinata al debug, non tiene conto di tale valore, e stampa nella finestra di scroll ogni messaggio ricevuto, indipendentemente dal livello di attenzione specificato. Questo parametro aggiuntivo del messaggio è stato specificatamente introdotto per l'applicazione utente Xnics che utilizza il livello di severità per effettuare uno screening dei messaggi in modo che l'utente riceva solo quelli utili a comprendere la situazione in atto.
I livelli di severità sono i seguenti:

- 0 : il messaggio rappresenta solo un avviso. Non viene spedito all' applicazione cliente ma viene registrato nel file di log creato dall' applicazione **Ftest**
- 1 : il messaggio viene spedito all'applicazione cliente che lo mostra all'utente
- 2 : il messaggio viene spedito all'applicazione cliente che lo mostra all'utente e lo registra nel proprio file di log,
- 3 : il messaggio viene spedito all'applicazione cliente che lo trascrive nel proprio log file senza mostrarlo all'utente.

2.4 Interfaccia grafica di GClient

L'interfaccia grafica del programma è molto semplice. È costituita da una finestra principale suddivisa in due parti (vedi Fig. 2.1).
La prima comprende una finestra di scroll dove sono stampati tutti i messaggi che **GClient** riceve dal programma demone **Ftest**.
La seconda è costituita da un lista di bottoni che attivano i comandi da inviare al sistema di acquisizione e configurano i parametri per la presa dati (numero di gruppi, tempo di integrazione, numero di integrazioni per gruppo, tipo di campionamento).
Ogni tasto esegue uno o più dei comandi previsti dal protocollo di comunicazione (vedi § 2.4.1).
Il protocollo di comunicazione stabilito, prevede la seguente lista di comandi: END, STOP, DATI (o DUMMYDATA), STATUS, TINT (o DIT), REINIT,GROUP, ABORT, DOUBLE, FRAME, RUN, WAVE, LOAD.
Alcuni di questi, come i comandi DIT, GROUP, DOUBLE, FRAME e LOAD, necessitano di paramentri aggiuntivi che sono:

- DIT : il tempo di integrazione espresso in secondi,
- GROUP: il numero di acquisizioni successive
- DOUBLE: una flag che identifica il tipo di campionamento: singolo (0) o doppio (1)
- FRAME: il numero dei gruppi ed il tempo di integrazione
- LOAD: il nome del file con la forma d'onda e la dimensione in bytes del file.

2.4.1 Comandi

I tasti ed i comandi eseguiti sono i seguenti:

- RUN: l' attivazione di questo tasto spedisce in successione all'applicazione **Ftest** i comandi GROUP, TINT, ITER, DOUBLE con i relativi parametri (i cui valori sono mostrati nelle finestre sottostanti) e il comando finale RUN che i avvia l'integrazione.
- DUMMY: vengono spediti due comandi in successione: DOUBLE con parametro 0 e DATI.
  Il primo disabilita automaticamente il doppio campionamento (visto che si tratta di un'acquisizione simulata), il secondo esegue un'acquisizione dummy. Il file dati generato viene spedito all'applicazione **GClient**.

- STATUS: questo tasto spedisce il comando STATUS al demone in esecuzione. Come risultato si ottiene la pubblicazione dello stato del sistema, sotto forma di messaggi che vengono spediti all'applicazione **GClient** e che questa visualizza nella finestra di scroll

- REINIT: il tasto spedisce il comando REINIT che genera una reinizializzazione completa dell'elettronica

- ABORT: provoca l'interruzione immediata delle acquisizioni in corso, mandando il comando ABORT all'applicazione server

- LOAD FILE: spedisce al programma di controllo il comando LOAD con i parametri necessari. Questo comando esegue la riprogrammazione del sequencer con la nuova forma d'onda, contenuta nel file spedito.

- FRAME: questo tasto spedisce il comando FRAME insieme ai parametri richiesti che l'applicazione **GClient** ricava dalle corrispndenti finestre di input.

- SINGLE: questo tasto esegue una singola acquisizione ed avvia il *viewer* `ds9` che visualizza il file fits risultante.

- WAVE: spedisce il comando WAVE che riprogramma il generatore di sequenze con il file di default `nics.obj`.

- END: viene spedito il comando END che termina l'applicazione di **GClient**

- STOP: viene spedito il comando STOP che causa la terminazione del demone **Ftest**

- NOISE: esegue una serie di integrazioni usate per valutare il rumore del sistema di acquisizione. In particolare viene usato il numero dei gruppi specificato dall'utente, mentre i tempi di integrazione sono letti da una tabella prestabilita (il default è 1, 2, 5, 10, 20, 50, 100, e 200 secondi).

- GROUP: il tasto spedisce il comando GROUP insieme al relativo parametro il cui valore è letto dalla finestra di testo adiacente al bottone. Il comando ha come risultato la configurazione del numero di gruppi. Se dopo l'esecuzione del comando GROUP, il valore del gruppo viene modificato e viene eseguito uno dei comandi RUN o FRAME, il valore del gruppo usato nell'acquisizione risulterà quello inserito per ultimo.

- DIT: questo tasto spedisce il comando TINT insieme al relativo parametro, il cui valore è specificato nella finestra di input accanto al bottone DIT.

Il comando TINT esegue la programmazione del tempo di integrazione usando il valore specificato.

- CONNECT: consente di avviare la connessione con il demone **Ftest** senza eseguire nuovamente il programma **GClient**.

2.4.2 Parametri

I parametri che configurano l'acquisizione sono i seguenti:

- GROUP: specifica il numero di acquisizione che devono essere eseguite. Il default è 3.
- DIT: specifica il tempo di integrazione di ogni singola acquisizione. Il minimo è 1 secondo ed il valore può essere modificato a passi di 0.1 secondi.
- DOUBLE SAMPLING ON/OFF: viene configurato il tipo di campionamento. Se la flag è attiva, vengono effettuate inegrazioni a campionamento multiplo.

Capitolo 3

Il programma viewer

Il programma *viewer* consente di visualizzare le immagini acquisite in *free-run* dal programma *ftest* e viene mandato in esecuzione una volta selezionata l'opzione *free-run* dal menu dell'applicazione **Ftest**.
L' applicazione *viewer* viene lanciata dal programma **Ftest** ricorrendo alla chiamata di sistema *system()*.
Durante questa operazione viene creata una connessione di rete tra le due applicazioni in modo che possano comunicare tra loro. In particolare il programma **Ftest** funziona da server per l'applicazione *viewer*.

3.1 Interfaccia grafica del viewer

Il programma *viewer* crea una finestra principale (vedi Fig. 3.1), organizzata nel seguente modo.
Nella parte inferiore viene mostrata una porzione di 256x256 pixels dell'immagine acquisita : spostanto le barre verticali ed orizzontali che appaiono lateralmente, possiamo esplorarne l'intero aspetto.
Nell'angolo in alto a sinistra di tale finestra compare un quadrato la cui posizione può essere modificata spostando il mouse e cliccando il bottone sinistro. La porzione di immagine racchiusa viene visualizzata nell'area che appare nella parte superiore destra della finestra principale e corrisponde alla regione dell'immagine che può essere ingrandita o ridotta con i tasti **ZoomOut** e **ZoomIn**.
Nella parte in alto a sinistra della finestra principale è presente una lista di tasti, organizzati verticalmente, tra cui, oltre ai bottoni **ZoomOut** e **ZoomIn**, appiaono i tasti **Colormap, Scale** e **Quit**.
Nella parte centrale della finestra principale compaiono tre finestre etichettate con i nomi **Value**, **X**, **Y**, dove sono mostrati il valore e le coordinate del punto dell'immagine su cui è posizionato il mouse.

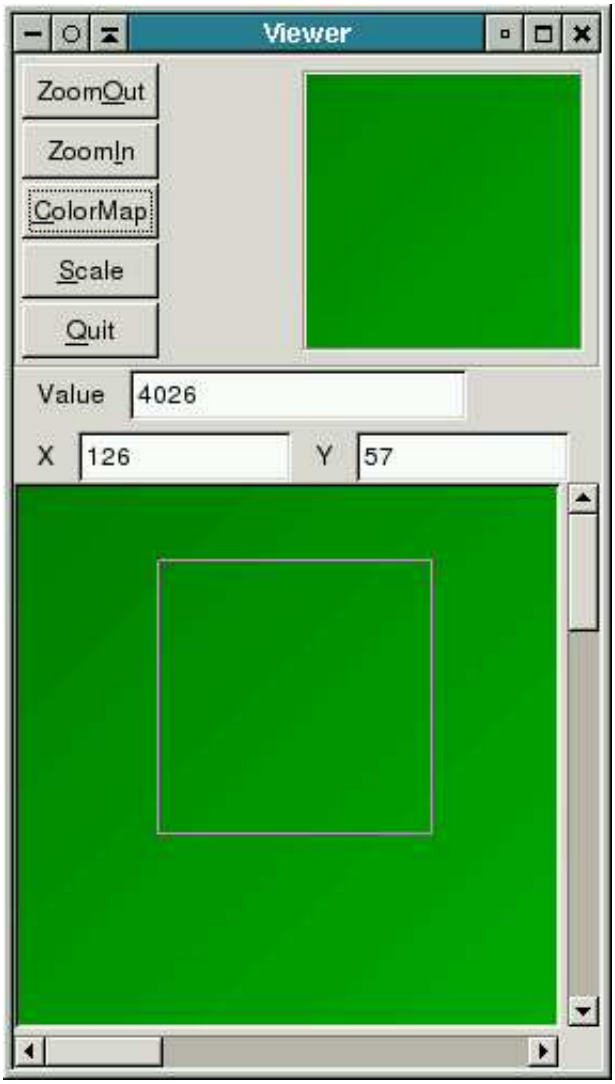

Figura 3.1: Finestra principale del programma viewer

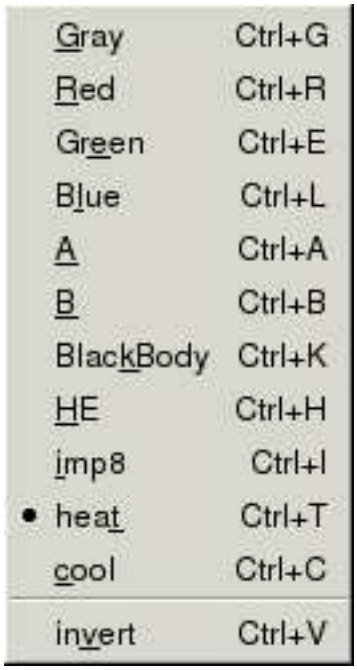

Figura 3.2: Colormap popup menu

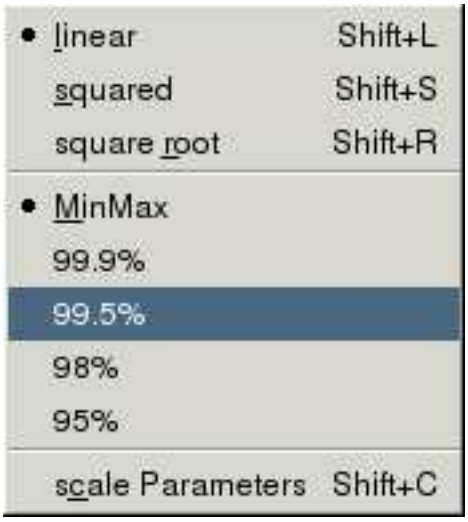

Figura 3.3: Scale popup menu

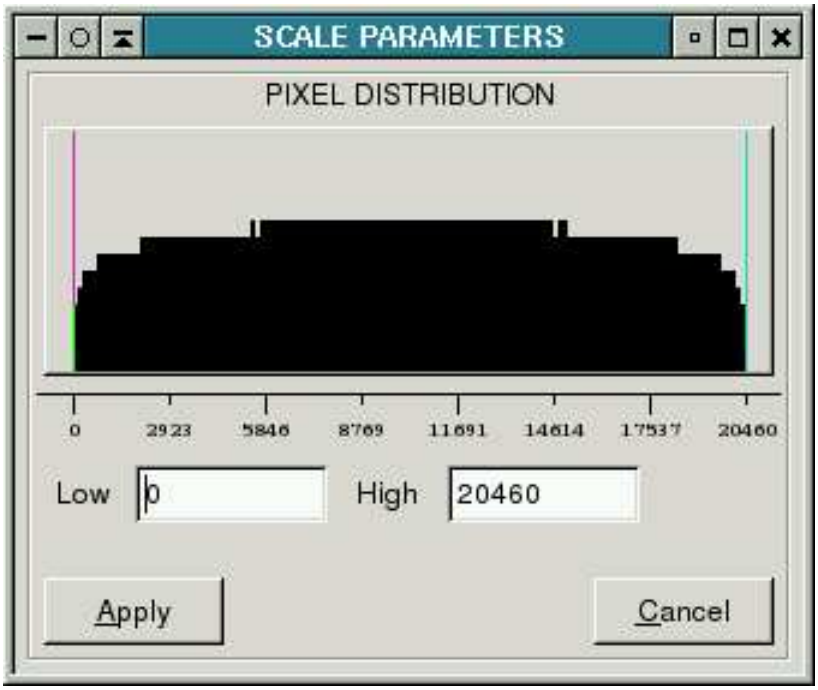

Figura 3.4: Colormap popup menu

3.2 Funzionalità dei comandi

Il comando **Colormap** apre un *popup menu* (vedi Fig. 3.2) con la lista delle mappe di colore che l' utente può selezionare per visualizzare l' immagine. Sotto la linea di separazione del menu compare l'opzione ***invert*** che opera l'inversione dei colori della mappa attiva in quel momento.
Il tasto **Scale** visualizza a sua volta un *popup menu* (vedi Fig. 3.3) le cui voci sono suddivise in tre categorie:

- nella prima sono presenti tre sottocomandi che eseguono operazioni di *scaling* sui dati.

- nella seconda sono presenti cinque opzioni che selezionano la percentuale dei dati da considerare nell'operazione di scaling.
  Per far ciò viene valutato l'istogramma dell'immagine e poi vengono determinati gli estremi dell'intervallo in modo che all'interno di esso cada la percentuale di dati selezionata dall' utente.

- nella terza è presente il comando che permette all'utente di selezionare liberamente i valori estremi da usare nell' operazione di *scaling*.
  In corrispondenza del comando ***scale Parameters*** viene aperta una finestra (vedi Fig. 3.4) in cui compare l'istogramma dell'immagine.
  Agli estremi di tale istogramma appaiono due linee di diverso colore,

la cui posizione può essere variata tenendo premuto il tasto sinistro del mouse, mentre questo viene trascinato.
La posizione delle due linee determina i valori minimo e massimo usati nell'operazione di *re-scaling* dei dati. I valori estremi dell' intervallo appaiono nelle due finestre **Low** e **High**.
Durante lo spostamento delle linee, l'immagine della finestra principale viene automaticamente aggiornata senza la necessità di ricorrere al tasto **Apply**.

Il tasto ***Quit*** chiude il programma di visualizzazione e la connessione di rete con il server.

Capitolo 4

Il programma Gsvbplot

Il programma **GsvbPlot** è un'applicazione che riporta in grafico le forme d'onda usate per la programmazione del sequencer di Fasti.
Il programma **GsvbPlot** utilizza come input i file in formato `wvf` prodotti dal programma di emulazione software **EmuSvb**, come descritto nel rapporto [2].
L'applicazione viene lanciata con il comando:

```
gsvbplot <wavefile>
```

Nel caso in cui l'utente non specifichi alcuna opzione oppure il file specificato come argomento non esista, il programma parte caricando una forma d'onda di test che genera internamente.
Una volta eseguito il comando, sul display appare una finestra come quella riportata in Fig. 4.1.
Nella parte sinistra della finestra principale sono mostrate le *label* dei segnali, a destra sono riportati in grafico i valori dei corrispondenti segnali in funzione dei tempi: questi valori sono letti dal file `wvf` della forma d'onda.
Sopra e sotto la finestra di *plot*, sono presenti una serie di tasti: TIME, X, ViewLog, File, SVB e Quit. A questi si aggiunge, nella parte superiore, una barra orizzontale con un cursore mobile.
I comandi attivabili con gli elementi presenti nella regione superiore della finestra principale, variano la rappresentazione grafica della forma d'onda, sia attraverso la selezione del tempo base (TIME) sia variando la regione della forma d'onda che vogliamo visualizzare (X e barra).
Gli altri comandi presenti nella parte bassa della finestra, a parte il tasto tasto `Quit` la cui funzione è ovvia, attivano una serie di operazioni che facilitano il lavoro dell'utente.
Di seguito descriviamo le operazioni effettuate dai diversi elementi dell'interfaccia grafica.

4.1 Tasto TIME

Il tasto `TIME` attiva un `popup menu` con la lista delle basi dei tempi selezionabili per la rappresentazione della forma d'onda.
Il programma è stato strutturato in modo che, una volta selezionata la base dei tempi, venga sempre visualizzato una regione della forma d'onda pari a sei volte il tempo base. Questa regione è centrata sul *tempo corrente* mostrato come *label* del tasto X
Questo è evidenziato nel grafico, dalla riga orizzontale disegnata in alto che risulta suddivisa in sei intevalli, ciascuno di ampiezza pari al tempo base prescelto. Insieme a questa linea, ne viene riportata un'altra verticale che riflette la posizione centrale dell 'intervallo.
Quando l'utente seleziona un valore della scala dei tempi diverso da quello attivo, il grafico della forma d'onda varia di conseguenza, ma non la sua posizione. Questa rimane la stessa, come si può vedere sia dalla posizione del cursore della barra orizzontale, sia dal valore che appare nella label del bottone X.

4.2 Comando X

Il tasto `X` attiva la selezione del *tempo corrente* della forma d'onda, cioè del tempo che viene riportato al centro del grafico e rispetto al quale viene disegnata la forma d'onda nell'intervallo compreso tra ± 3 volte il valore del tempo base.
In particolare, il comando attivato dal tasto X crea un `popup menu` con le seguenti voci:

- BEGIN: la forma d'onda viene visualizzata a partire dall'inizio. Il tempo iniziale compare perciò al centro della finestra di plot.
  Questa opzione è quella di default all'avvio del programma.

- CENTER: la forma d'onda viene visualizzata con il centro corrispondente al tempo medio dell'intervallo complessivo della forma d'onda

- END: la forma d'onda viene visualizzata in modo che l'ultimo valore dei tempi appia al centro del grafico.

- TIME: l'attivazione di questa voce apre una finestra di testo in cui l'utente può inserire a mano il valore rispetto al quale vengono riportati i valori della forma d'onda nell'intervallo fissato dal tempo base.
  Una volta inserito il valore, si può variare l'unità di misura cliccando sul tasto adiacente alla finestra di testo.
  La pressione del tasto `Enter` o la chiusura della finestra, conferma la scelta ed aggiorna il valore che viene sempre riportato come label del tasto X.

4.3 Barra orizzontale

Subito sotto i due tasti TIME ed X è posizionata una barra orizzontale con un cursore la cui posizione varia premendo i tasti del mouse, oppure azionando i tasti [←] e [→], oppure [PgUp] e [PgDown].
La posizione del cursore determina quale tempo della forma d'onda, espresso come percentuale dell'intervallo totale, viene riportato al centro della finestra di plot.
La variazione della posizione del cursore aggiorna automaticamente il valore mostrato nel tasto X.
Abbiamo inoltre introdotto la possibilità di un controlo più fine da parte dell'utente, attraverso l'uso del puntatore.
Cliccando con il tasto 1 in un punto prescelto della forma d'onda, il grafico verrà aggiornato in modo che la posizione selezionata risulti quella corrente, cioè quella che viene mostrata al centro della finestra.
Se invece premiamo il tasto 2 o 3 del mouse all'interno dell'area di plot, otteniamo i valori della forma d'onda corrispondenti alla posizione del cursore.

4.4 File e SVB

Entrambi i comandi creano una finestra di dialogo per la selezione dei file, con una lista di directory e di files presenti nella directory di riferimento.
Nel primo caso vengono mostrati solo i files con estensione `wvf` (vedi Fig. 4.2), nel secondo solo quelli con estensione `asm` (vedi Fig. 4.3).
L'utente si può spostare attraverso il filesystem usando la lista delle directory oppure attraverso il popup menu che compare premendo il tasto presente sopra le due aree specificate.
Nel primo caso possiamo selezionare un file `wvf` diverso e caricarlo in modo da eseguirne il grafico.
Nell'altro caso possiamo invece selezionare un nuovo file `asm` e generare attraverso i programmi **svb1asm** e **EmuSvb** il corrispondente file in formato wvf.
Durante le operazione di caricamento o creazione del file con la forma d'onda, il tasto di comando della finestra di dialogo ed i tasti [File] o [SVB] dell' interfaccia principale, risultano inattivi. Il loro stato torna attivo quando termina l'esecuzione dell'operazione.
È comunque possibile interrempore l'esecuzione dell'operazione in corso in qualunque momento, chiudendo direttamente la finestra di dialogo.

4.5 ViewLog

Il tasto [ViewLog] apre una finestra dove vengono stampati i messaggi generati durante l'esecuzione dei programmi **svb1asm** ed **EmuSvb**.

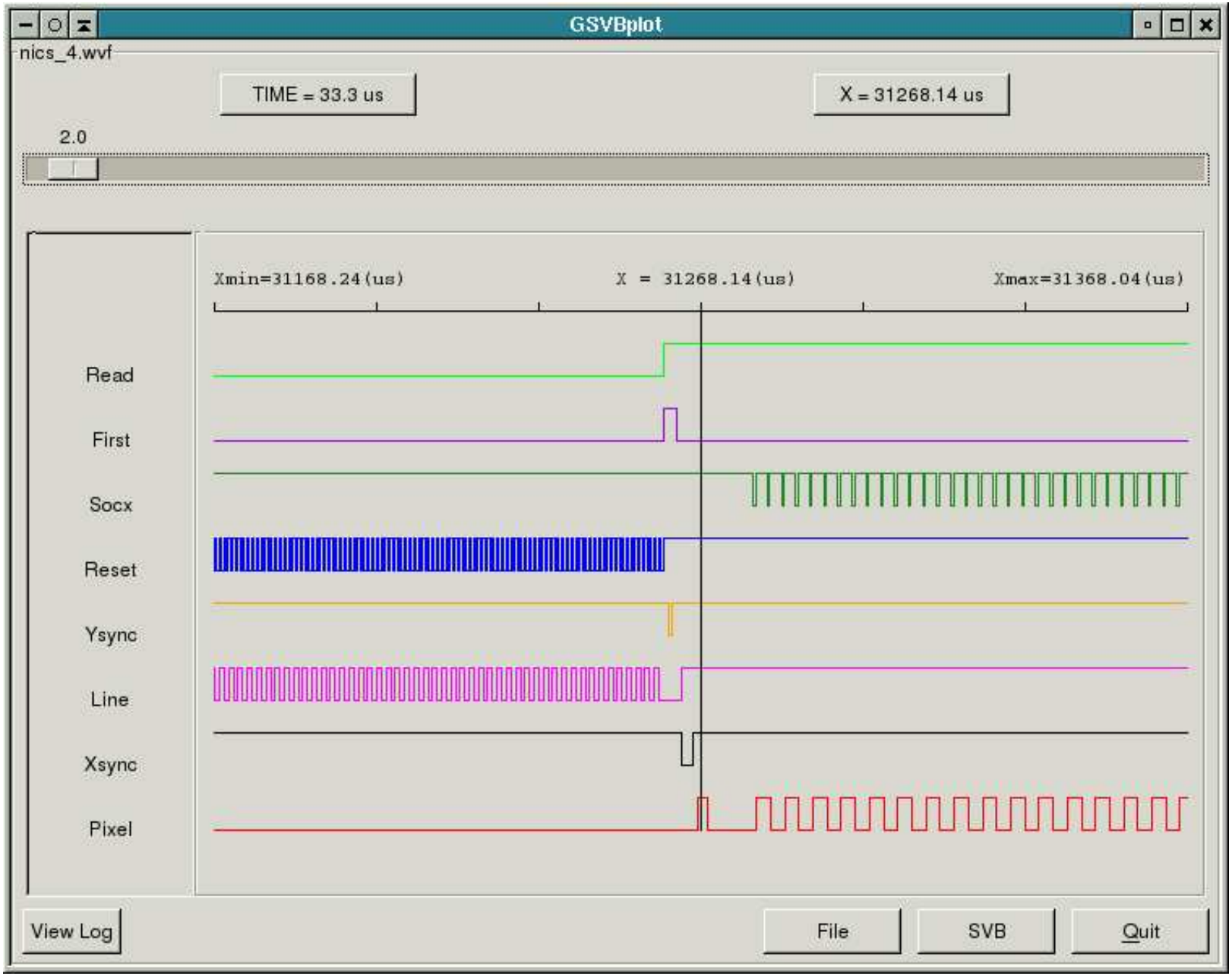

Figura 4.1: Finestra principale

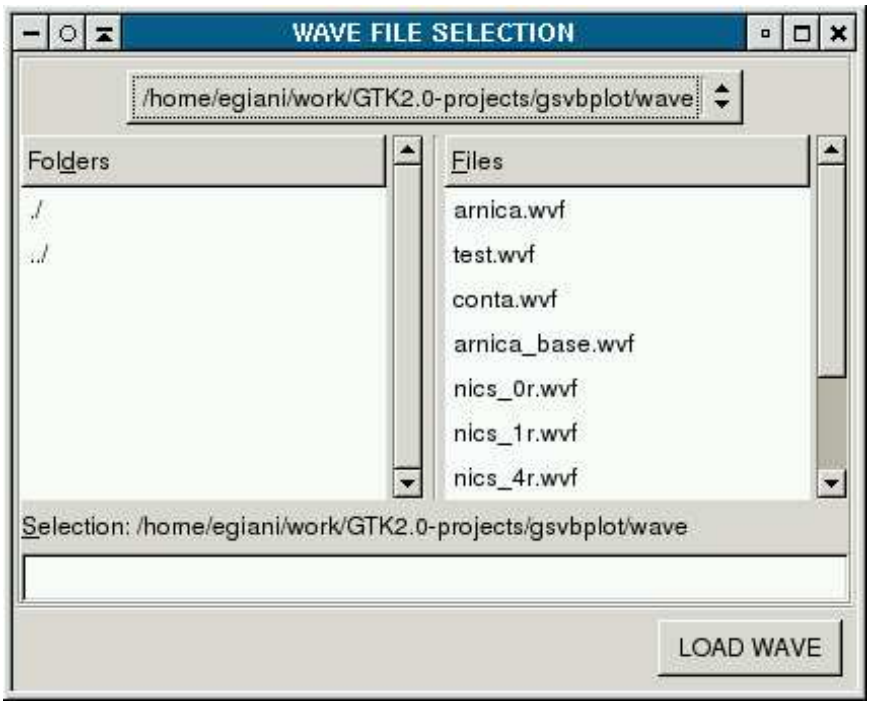

Figura 4.2: Finestra di dialogo

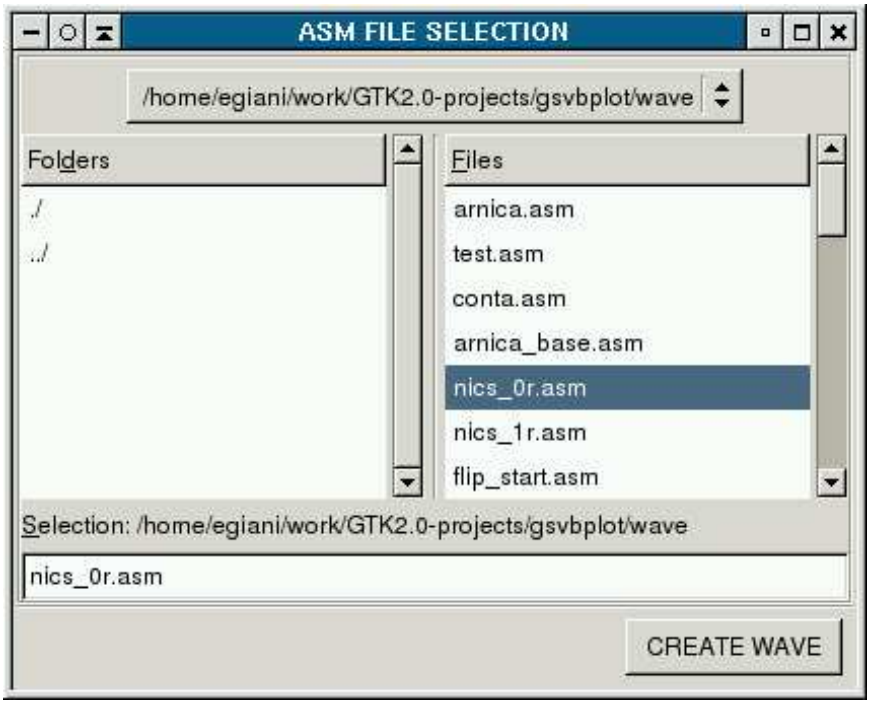

Figura 4.3: Finestra di dialogo

Capitolo 5

La struttura dettagliata dei programmi grafici

I programmi illustrati nei capitoli precedenti sono scritti in C e compilati sulle librerie GTK+ (versione 2), un toolkit multi-piattaforma usato nella creazione di interfacce grafiche.
L' interfaccia grafica delle applicazioni è stata generata con l'applicazione `Glade` un costruttore grafico di interfacce per GTK+ e GNOME.
`Glade` è in grado di produrre il codice sorgente del progetto in diversi linguaggi, tra cui il C.
Oltre ai sorgenti, i cui nomi possono differire se in fase di progetto sono state modificate alcune opzioni, il programma `Glade` crea nella directory principale i seguenti files:

- `autogen.sh`: è uno script che esegue `automake`, `autoconf` e le utility relative per costruire i Makefiles. A questo punto è sufficiente eseguire il comando `make` per costruire l'applicazione.
- `configure.in`: uno script standard passato ad `autoconf` per generare lo script `configure`
- `Makefile.am`: contiene le regole standard del make file passate ad `automake` per generare `Makefile.in` che lo script `configure` cambia in `Makefile`.
- `acconfig.h`: contiene alcune macro che sono settate durante lo script `configure` ed aggiunte al file header `config.h` (che deve essere il primo header file incluso in tutti i sorgenti).
- `stamp-h.in`: usato come timestamp da `automake` per la ricostruzione di alcuni file generati.
- `src/Makefile.am`: standard automake file

- `src/main.c` : contiene la funzione `main()` che crea la finestra principale.
- `src/interface.h`: include file con la dichiarazione delle funzioni che vengono chiamate per la creazione delle finestre e dei dialoghi costruiti con `Glade`.
- `src/interface.c`: il codice per creare le finestre, i dialoghi e tutte le widgets
- `src/callbacks.h`: contiene le dichiarazioni dei gestori dei segnali e delle funzioni di callback che lo sviluppatore deve scrivere
- `src/callbacks.c`: contiene le funzioni di callback e di gestione dei segnali. Queste sono essenzialmente dei contenitori vuoti. È compito del programmatore aggiungere il codice che implementa le funzionalità specifiche dell'applicazione.
- `src/support.h`: header file con la dichiarazione delle funzioni di supporto
- `src/support.c`: codice con le funzioni di supporto

Gli eventuali sorgenti ed include files aggiunti al progetto e non creati da `Glade` devono essere inseriti nel file `src/Makefile.am` affinchè vengano introdotti nel nuova versione del Makefile.
Quando l'interfaccia grafica viene modificata, `Glade` sovrascrive i seguenti files: `interface.h`, `interface.c`, `support.h`, e `support.c`. È importante perciò, che questi sorgenti non siano **mai** editati a mano.
Questa cautela non è richiesta per i files `callbacks.c` e `callbacks.h` perchè le eventuali funzioni di gestione dei segnali aggiunte, vengono semplicemente appese al codice contenuto nei files precedenti: il codice già scritto risulta perciò sicuro.

5.1 Generazione delle applicazioni

Per generare l'eseguibile di una applicazione grafica, deve essere prima generato il `Makefile`. Questo dipende dalla configurazione del sistema su cui lavoriamo, perche' la posizione delle librerie di sviluppo, può variare da distribuzione all'altra.
L'esecuzione di `autogen.sh` genera il `Makefile` relativo alla propria configurazione. Se non si verificano errori, nella directory `src` del progetto troveremo il Makefile per l'applicazione e lanciando la compilazione con `make` verrà prodotto il file eseguibile del progetto.

Indice

Bibliografia

[1] Baffa, C., Fasti un controller veloce per l'Infrarosso, 1998, Memo del Gruppo Infrarosso di Arcetri.

[2] Baffa, C., E. Giani, Progetto Fasti - L'assembler e l'emulatore, Rapporto interno dell'Osservatorio Astrofisico di Arcetri, **N°1/2004.**

[3] Baffa, C., C., Biliotti, V., Progetto Fasti - Un assembler per lo SVB1, Rapporto interno dell'Osservatorio Astrofisico di Arcetri, **N°1/2000.**

[4] Baffa, C., The Fasti Project, Memorie della Società Astronomica Italiana, 2003, **74**, p165.

[5] Baffa, C., Biliotti, V., Checcucci, A., Gavrioussev, V., Gennari, S., Giani, E., Lisi, F., Marcucci, G., Sozzi., M., "The Fasti Project", ADASS XII ASP Conference Series, Vol. **295**, 2003, Payne, H., Jedrzejewski, R., Hook, R. Eds., p.355

[6] Checcucci, A., Baffa, C., Giani, E., "Progetto Fasti – Fasti Global Controller, Rapporto interno dell'Osservatorio Astrofisico di Arcetri, **N°3/2001**.