



*Toolbox*  
**Configuration Utilities  
User Guide**

Document Number 500-00409

Revision 1.01

September 1998

*Copyright © 1998 Spectrum Signal Processing Inc.*

*All rights reserved, including those to reproduce this document or parts thereof in any form without permission in writing from Spectrum Signal Processing Inc.*

*All trademarks are registered trademarks of their respective owners.*

*Spectrum Signal Processing reserves the right to change any of the information contained herein without notice.*

---

# Customer Feedback

At Spectrum, we recognize that product documentation that is both accurate and easy to use is important in aiding you in your new product development. We appreciate hearing your comments on how our product's documentation could be improved.

If you wish to comment on any Spectrum documentation then please fax or e-mail a completed copy of this page to us.

Full Name of Document: \_\_\_\_\_

Document Number: \_\_\_\_\_ Version Number: \_\_\_\_\_

If you have found a technical inaccuracy please describe it here:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

If you particularly liked or disliked an aspect of the manual then please describe it here:

\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_  
\_\_\_\_\_

It may be helpful for us to call you to discuss your comments. If this would be acceptable please provide the following details:

Name: \_\_\_\_\_ Telephone #: \_\_\_\_\_

Organization: \_\_\_\_\_

Thank you for your time,

Spectrum Signal Processing Documentation Group

Fax: (604) 421-1764

Email: [documentation@spectrumsignal.com](mailto:documentation@spectrumsignal.com)

---

# Contacting Spectrum...

Spectrum's team of dedicated Applications Engineers are available to provide technical support to you for this product. Our office hours are Monday to Friday, 8:00 AM to 5:00 PM, Pacific Standard Time.

Telephone 1-800-663-8986 or (604) 421-5422

Fax (604) 421-1764

Email [support@spectrumsignal.com](mailto:support@spectrumsignal.com)

Internet <http://www.spectrumsignal.com>

When you contact us, please have the following information on hand:

- A concise description of the problem
- The name of all Spectrum hardware components
- The name and version number of all Spectrum software components
- The minimum amount of code that demonstrates the problem
- The version number of all software packages, including compilers and operating systems

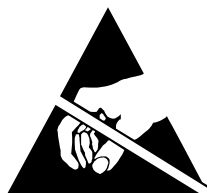
---

# Preface

Spectrum Signal Processing offers a complete line of DSP hardware, software and I/O products for the DSP Systems market based on the latest DSP microprocessors, bus interface standards, I/O standards and software development environments. By delivering quality products, and DSP expertise tailored to specific application requirements, Spectrum can consistently exceed the expectations of our customers. We pride ourselves in providing unrivaled pre and post sales support from our team of application engineers. Spectrum has excellent relationships with third party vendors which allows us to provide our customers with a more diverse and top quality product offering.



Spectrum achieved ISO 9001 quality certification in 1994.



As Spectrum's hardware products are static sensitive, please take precautions when handling and make sure they are protected against static discharge.



---

# Table of Contents

---

1 Introduction.....	1
1.1. Purpose of this Manual .....	1
1.2. Reference Documents .....	1
<hr/>	
2 What Is ToolBox .....	3
<hr/>	
3 Installation .....	5
<hr/>	
4 ToolBox Supported Hardware .....	7
<hr/>	
5 Running ToolBox .....	9
5.1. Notes on TIM Modules.....	9
5.1.1. Dual TIMs.....	9
5.1.2. Speed Conflicts .....	10
<hr/>	
6 Example ToolBox Sessions .....	11
6.1. C4x Example.....	11
6.2. C6x Example.....	20
6.3. Example - Using the Generated Files in ALIB Code.....	23
6.3.1. RDF use for Maranello (Quad TIM Carrier).....	23
6.3.2. SDF/LDF use for Detroit (Single C6x Carrier).....	24
6.3.3. RDF use for Morocco (Octal Sharc Board) .....	24
<hr/>	
7 Appendix A - Modifying ToolBox Configuration Files.....	27



# 1 Introduction

## 1.1. Purpose of this Manual

This manual provides the information you need to use the ToolBox utility. ToolBox allows for the configuration of single and multi-board systems of boards and processors.

## 1.2. Reference Documents

Installation Guides, Technical Reference Manuals, Programmer's Guides for boards being configured.



---

## 2 What Is ToolBox

ToolBox is used to generate configuration files for DSP systems. The current configuration files generated are:

- System Definition Files (SDF) - used in calls to **SystemOpen** functions in Host Application Libraries (ALIBs). For example if a system is called *MySystem* then ToolBox generates the SDF file *mssystem.sdf*.
- Load Definition Files (LDF) - used in calls to **SystemLoad** functions in Host ALIBs. For example if a system is called *MySystem* then ToolBox generates the LDF file *mssystem.ldf*.
- Resource Definition Files (RDF) - used in calls to **SystemOpen** functions in Host Application Libraries (ALIBs). For example if a system is called *MySystem* then ToolBox generates the RDF file *mssystem.rdf*.

These configuration files are used to configure and operate one or more boards in a DSP system. An example code segment is given later in the manual.

Running under Solaris and Win95/NT, ToolBox provides a set of text based menus to generate these configuration files which are then used by your host application library to open devices and load processors.



---

## 3 Installation

Installation of Toolbox consists of two steps:

- 1) Copying the contents to a hard drive
- 2) Setting the `ssp_path` environment variable

### Step 1:

Copy ToolBox using a Windows Explorer or a command line utility to your hard drive. A recommended directory is:

`c:\spectrum\toolbox` (Win32)

`~/spectrum/toolbox` (Solaris)

The '~' for a Solaris system represents your user root directory. Toolbox can be placed in a common directory if this is more appropriate for your Solaris system.

### Step 2:

Set the `ssp_path` environment variable to point at the `toolbox.ini` file by adding

`c:\spectrum\toolbox\gsl` (Win32)

`~/spectrum/toolbox/gsl` (Solaris)

to the `ssp_path` setting.

For Windows95 `ssp_path` can be set by adding it in your `autoexec.bat` file.

For WindowsNT `ssp_path` can be set through Control Panel under the System\Environment option.

For Solaris add `ssp_path` to your environment in a `.cshrc` or equivalent file.



## 4 ToolBox Supported Hardware

The following are the supported hardware products which ToolBox supports:

Carriers:                   Dakar  
                                  Detroit  
                                  Lemans  
                                  Maranello  
                                  Morocco  
                                  Monaco

TIMs  
(for C4x carriers):       DAB  
  
                                  DDC  
                                  SS/DSP1  
                                  ST/DSP2  
                                  MD  
                                  SDDR  
                                  TA

Additional custom hardware can be added by modifying existing configuration files as described in a later section of the manual.

The following table lists all the carriers and the configuration files output for each of the carrier types. Note that certain configuration files are not required for all carriers:

**Configuration files Output by Carrier Type**

Carrier	SDF	LDF	RDF
Dakar	✓	✓	✓
Detroit	✓	✓	
Lemans	✓	✓	
Maranello	✓	✓	✓
Morocco	✓	✓	✓
Monaco	✓	✓	✓



---

## 5 Running ToolBox

ToolBox is a Win32 Console application or a Solaris text based application. To begin run

```
c:\spectrum\toolbox\toolbox.exe(Win32)
```

```
~/spectrum/toolbox/toolbox (Solaris).
```

Examples of running ToolBox are shown in the next sections, generally the sequence of operations is as follows:

- 1) Select Define a system of boards from the Main Menu
- 2) Choose a System name
- 3) Enter from menus the hardware in your System
- 4) Generate a System Definition File (SDF)
- 5) Generate a Resource Definition File (RDF), if required
- 6) Exit

The files generated by ToolBox (SDF, LDF, RDF, etc.) are placed in the directory:

```
c:\spectrum\toolbox\system\ (Win32)
```

```
~/spectrum/toolbox/system/ (Solaris)
```

To use these files with the Host application libraries, copy the required files to the directory where you are developing your host code.

For some host ALIBs which use SDF files, the `ssp_path` environment variable can be set to search the system directory (as well as the toolbox directory) to automatically find the generated SDF/LDF files. By setting the `ssp_path` environment variable it is not necessary to move the files after they have been generated.

### 5.1. Notes on TIM Modules

#### 5.1.1. Dual TIMs

Dual or Double wide TIMs, those which take two side-by-side TIM sites, by default are dealt with through ToolBox as two single modules. For example,

when installing a DDC there are two models for each memory version. The first of the two models is the processor 0 of the dual wide TIM while second of the two models is processor 1 of the dual wide TIM. *You must specify the correct side-by-side TIM sites on the carrier board for the SDF to be correct.*

### **5.1.2. Speed Conflicts**

Toolbox does not check for processor speed conflicts between TIM modules, other TIM modules, and carrier boards. In many operating conditions it is possible to use different processor speeds by disabling global clocking or other means. If you are using different processor speeds on the same TIM carrier refer to the Technical Reference Manual for your carrier for setup information.

---

## 6 Example ToolBox Sessions

### 6.1. C4x Example

This example uses ToolBox to perform the following operations:

- Create system consisting of a single Maranello VME TIM Carrier Board
- Specify if the system is running under Solaris
- Enter the VME base address, interrupt vector/level
- Add 4 TIM SS/DSP1 modules
- Create a SDF/LDF
- Create a RDF

The first screen displayed prompts for the selection of an operation to perform:

```
ToolBox Configuration Tool
Spectrum Signal Processing Inc. (c)1998

MAIN MENU
[a] Define a new system of boards
[b] Quit the ToolBox

Enter Choice:
```

The first step is to define a new set of carrier boards by choosing *a*. After selecting *a*, enter a system name (in this case *MySystem*). If the system already exists you will be prompted whether to overwrite it:

```
ToolBox Configuration Tool
Spectrum Signal Processing Inc. (c)1998

MAIN MENU
[a] Define a new system of boards
[b] Quit the ToolBox

Enter Choice: a

Enter a System Definition File name: MySystem

BOARD MENU
[a] MARANELLO:      Quad C4x VME Carrier
[b] DAKAR:          Trio C4x PCI Carrier
[c] MOROCCO:        Octal Sharc VME Carrier
[d] LEMANS:         Octal C4x VXI Carrier
[e] DETROIT:        Single C6x PCI Carrier
[f] MONACO:         C6x VME Carrier
[g] Quit Menu

Enter Choice:
```

If a Maranello is selected by entering *a*, then a question is asked whether the host for the system is running Solaris. This is important as the configuration files change depending on the Host operating system. In this example, the Host is not running Solaris so *n* is entered:

```
BOARD MENU
[a] MARANELLO:      Quad C4x VME Carrier
[b] DAKAR:          Trio C4x PCI Carrier
[c] MOROCCO:        Octal Sharc VME Carrier
[d] LEMANS:         Octal C4x VXI Carrier
[e] DETROIT:        Single C6x PCI Carrier
[f] MONACO:         C6x VME Carrier
[g] Quit Menu

Enter Choice: a

Choose the Operating System for the System
[a] Win32(NT & '95)
[b] Solaris
[c] VxWorks

Enter Choice: a

MARANELLO MENU
[a] cv8_s1_60 Quad C4x VME Carrier 60 MHz (512 KB Global SRAM)
[b] cv8_s2_60 Quad C4x VME Carrier 60 MHz (1 MB Global SRAM)
[c] cv8_s3_60 Quad C4x VME Carrier 60 MHz (2 MB Global SRAM)
[d] Quit Menu

Enter Choice:
```

For this example, the Maranello model is *a*. After entering *a*, several questions regarding base address and interrupt level are asked. If a carrier other than Maranello was selected then similar questions, particular to the features of the carrier, would be asked. The defaults for many of these are shown in [] brackets. Defaults are chosen by pressing *Enter*. For this example the defaults were used. If not using the defaults then enter the actual values:

```
MARANELLO MENU
[a] cv8_s1_60 Quad C4x VME Carrier 60 MHz (512 KB Global SRAM)
[b] cv8_s2_60 Quad C4x VME Carrier 60 MHz (1 MB Global SRAM)
[c] cv8_s3_60 Quad C4x VME Carrier 60 MHz (2 MB Global SRAM)
[d] Quit Menu

Enter Choice: a

board base address: [0x10000000] 0x
board interrupt level: [0x00000004] 0x
board interrupt vector: [0x00000008] 0x

Install & remove TIMs
[a] Install TIM
[b] Remove TIM
[c] Done with board

Enter Choice
```

After entering the board type and its parameters it is now possible to enter the data for the TIM modules installed on the carrier board. Selecting *a* to Install TIM brings up the list of valid TIM sites. Any site name (i.e. SiteA, SiteB, etc.) which doesn't have a description beside it can have a TIM added to it:

```
Install & remove TIMs
[a] Install TIM
[b] Remove TIM
[c] Done with board

Enter Choice: a

Pick a site
[a] SiteA
[b] SiteB
[c] SiteC
[d] SiteD
[e] cancel

Enter Choice:
```

For this example, *a* is entered to select SiteA as the location for the TIM. After selecting a TIM site, the TIM types are displayed:

```
Pick a site
[a] SiteA
[b] SiteB
[c] SiteC
[d] SiteD
[e] cancel

Enter Choice: a

TIM MENU
[a] DAB:           Data Acquisition Board
[b] DDC:           Two Site C44 Digital Down Converter
[c] MD:           DRAM SIMM support C40
[d] SDDR:         Single C44 Digital Drop Receiver
[e] SS:           Super SRAM C40
[f] ST:           Super Twin Dual C44
[g] TA:           Dual C40
[h] Quit Menu

Enter Choice:
```

There are several models of each TIM type. Selecting *e* on the TIM menu to choose the SS/DSP1 then presents the SS Menu which presents each model of the SS/DSP1 TIM:

```
TIM MENU
[a] DAB:           Data Acquisition Board
[b] DDC:           Two Site C44 Digital Down Converter
[c] MD:            DRAM SIMM support C40
[d] SDDR:          Single C44 Digital Drop Receiver
[e] SS:            Super SRAM C40
[f] ST:            Super Twin Dual C44
[g] TA:            Dual C40
[h] Quit Menu

Enter Choice: e

SS MENU
[a] 40ss2_60 Super SRAM C40 60 MHz (1.5 MB SRAM)
[b] 40ss4_60 Super SRAM C40 60 MHz (2 MB SRAM)
[c] 40ss5_50 Super SRAM C40 50 MHz (4 MB SRAM)
[d] 40ss5_60 Super SRAM C40 60 MHz (8 MB SRAM)
[e] 40ss6_50 Super SRAM C40 50 MHz (8 MB SRAM)
[f] 40ss6_60 Super SRAM C40 60 MHz (8 MB SRAM)
[g] Quit Menu

Enter Choice:
```

Entering *a* selects the 60Mhz - 1.5 MB SRAM version of the TIM for the carrier board SiteA and returns again to the TIM menu:

```
SS MENU
[a] 40ss2_60 Super SRAM C40 60 MHz (1.5 MB SRAM)
[b] 40ss4_60 Super SRAM C40 60 MHz (2 MB SRAM)
[c] 40ss5_50 Super SRAM C40 50 MHz (4 MB SRAM)
[d] 40ss5_60 Super SRAM C40 60 MHz (8 MB SRAM)
[e] 40ss6_50 Super SRAM C40 50 MHz (8 MB SRAM)
[f] 40ss6_60 Super SRAM C40 60 MHz (8 MB SRAM)
[g] Quit Menu

Enter Choice: a

Install & remove TIMs
[a] Install TIM
[b] Remove TIM
[c] Done with board

Enter Choice:
```

When *a*, Install TIM, is selected again the TIM Site list is again shown, and the previously installed TIM is shown:

```
Install & remove TIMs
[a] Install TIM
[b] Remove TIM
[c] Done with board

Enter Choice: a

Pick a site
[a] SiteA: 40ss2_60 Super SRAM C40 60 MHz (1.5 MB SRAM)
[b] SiteB
[c] SiteC
[d] SiteD
[e] cancel

Enter Choice:
```

Repeating the above process for the 3 other TIM sites with different 60Mhz versions of the SS/DSP1 TIM module results in the following TIM/Site definitions:

```
Install & remove TIMs
[a] Install TIM
[b] Remove TIM
[c] Done with board

Enter Choice: a

Pick a site
[a] SiteA: 40ss2_60 Super SRAM C40 60 MHz (1.5 MB SRAM)
[b] SiteB: 40ss4_60 Super SRAM C40 60 MHz (2 MB SRAM)
[c] SiteC: 40ss5_60 Super SRAM C40 60 MHz (8 MB SRAM)
[d] SiteD: 40ss6_60 Super SRAM C40 60 MHz (8 MB SRAM)
[e] cancel

Enter Choice:
```

When the TIMs have been completely installed select *e* if at the “Pick a site” menu to return to “Install & remove TIMs” and then *c* to select “Done with board”

The next question asked is whether Node A (equivalent to SiteA) on the Maranello is a boot processor. One or more boot processors are required to communicate between the host and the DSP board. For a Maranello each Node A is a boot processor so *y* is entered:

```
Pick a site
[a] SiteA: 40ss2_60 Super SRAM C40 60 MHz (1.5 MB SRAM)
[b] SiteB: 40ss4_60 Super SRAM C40 60 MHz (2 MB SRAM)
[c] SiteC: 40ss5_60 Super SRAM C40 60 MHz (8 MB SRAM)
[d] SiteD: 40ss6_60 Super SRAM C40 60 MHz (8 MB SRAM)
[e] cancel

Enter Choice: e

Install & remove TIMs
[a] Install TIM
[b] Remove TIM
[c] Done with board

Enter Choice: c

Make Node A a boot processor? (y/n) y
another board? (y/n)
```

If using a multi-board system, the process can then be repeated for additional boards by answering *y* for “another board (y/n)”. In this example *n* is entered as there is only one board.

### Front Panel Connections

It is possible to enter the inter-board or front panel communication port connections in Toolbox as well. This is useful if:

- booting a second board in a system through the front panel connections
- there are not a connection between all processors and a boot processor via the on-board communications links.

```
Install & remove TIMs
[a] Install TIM
[b] Remove TIM
[c] Done with board

Enter Choice: c

Make Node A a boot processor? (y/n) y
another board? (y/n) n

Modify Inter-Board Connections
[a] make connection
[b] break connection
[c] Done with system

Enter Choice:
```

In this case, *c* is entered as there are no front panel connections.

Now that the system has been defined, the Main Menu now presents the following options:

```
Modify Inter-Board Connections
[a] make connection
[b] break connection
[c] Done with system

Enter Choice: c

MAIN MENU
[a] Create a system definition file
[b] Create script file
[c] Quit the ToolBox

Enter Choice:
```

“Creating a system definition file” creates a text file that describes the system and the connections for the system. This text file can then be used with the Host application libraries that use SDF files. This option also generates an LDF file. Both of these files are output to:

- |                            |           |
|----------------------------|-----------|
| c:\spectrum\toolbox\system | (Win32)   |
| ~/spectrum/toolbox/system  | (Solaris) |

Creating a SDF file also checks that there is a connection for each processor back to a root processor in the system.

```
MAIN MENU
[a] Create a system definition file
[b] Create script file
[c] Quit the ToolBox

Enter Choice: a

System definition file successfully parsed.

MAIN MENU
[a] Create a resource definition file
[b] Create script file
[c] Quit the ToolBox

Enter Choice:
```

After creating a SDF file the Main Menu then allows for a Resource Definition File (RDF) to be created by selecting *a*. A RDF file is then created and can be used with Host application libraries that use this type of configuration file. Refer to the Programmer's Guide for the board to determine if you need an SDF or RDF.

A script file, created by entering *b*, is a listing of the options entered at each stage of the configuration. This file is not used by any of the application libraries and simply outputs a simple system description to a *SystemName.tbs* file.

After creating the system definition, SDF and/or RDF exit by selecting *c* from the Main Menu.

## 6.2. C6x Example

This example uses ToolBox to perform the following operations:

- Create system consisting of a single Detroit C6x board with 2Mbytes of shared memory
- Set the BoardId for the Detroit
- Create an SDF/LDF

The first screen displayed prompts for the selection of an operation to perform:

```
ToolBox Configuration Tool
Spectrum Signal Processing Inc. (c)1998

MAIN MENU
[a] Define a new system of boards
[b] Quit the ToolBox

Enter Choice:
```

The first step is to define a new set of carrier boards by choosing *a*. After selecting *a*, enter a system name (in this case *MySystem*). If the system already exists you will be prompted whether to overwrite it:

```
ToolBox Configuration Tool
Spectrum Signal Processing Inc. (c)1998

MAIN MENU
[a] Define a new system of boards
[b] Quit the ToolBox

Enter Choice: a

Enter a System Definition File name: MySystem

BOARD MENU
[a] MARANELLO:      Quad C4x VME Carrier
[b] DAKAR:          Trio C4x PCI Carrier
[c] MOROCCO:        Octal Sharc VME Carrier
[d] LEMANS:         Octal C4x VXI Carrier
[e] DETROIT:        Single C6x PCI Carrier
[f] MONACO:         C6x VME Carrier
[g] Quit Menu

Enter Choice:
```

The next step is to select the board type. Entering *e* selects the Detroit carrier board:

```
BOARD MENU
[a] MARANELLO:      Quad C4x VME Carrier
[b] DAKAR:          Trio C4x PCI Carrier
[c] MOROCCO:        Octal Sharc VME Carrier
[d] LEMANS:         Octal C4x VXI Carrier
[e] DETROIT:        Single C6x PCI Carrier
[f] MONACO:         C6x VME Carrier
[g] Quit Menu

Enter Choice: e

DETROIT MENU
[a] de62_s1 Single C6x PCI Carrier 200 MHz (512 KB Global/256 KB Local
SRAM)
[b] de62_s2 Single C6x PCI Carrier 200 MHz (2 MB   Global/256 KB Local
SRAM)
[c] de62_s4 Single C6x PCI Carrier 200 MHz (2 MB   Global/512 KB Local
SRAM)
[d] Quit Menu

Enter Choice:
```

The next step is to select the 2Mbyte Global SRAM version by selecting *b* and entering a BoardId of 0x10 (see the Detroit Programmer's Guide for an explanation of BoardId):

```
BOARD MENU
[a] MARANELLO:      Quad C4x VME Carrier
[b] DAKAR:          Trio C4x PCI Carrier
[c] MOROCCO:        Octal Sharc VME Carrier
[d] LEMANS:         Octal C4x VXI Carrier
[e] DETROIT:        Single C6x PCI Carrier
[f] MONACO:         C6x VME Carrier
[g] Quit Menu

Enter Choice: e

DETROIT MENU
[a] de62_s1 Single C6x PCI Carrier 200 MHz (512 KB Global/256 KB Local
SRAM)
[b] de62_s2 Single C6x PCI Carrier 200 MHz (2 MB   Global/256 KB Local
SRAM)
[c] de62_s4 Single C6x PCI Carrier 200 MHz (2 MB   Global/512 KB Local
SRAM)
[d] Quit Menu

Enter Choice: b

board ID: [0x00000001] 0x10
another board? (y/n)
```

If using a multi-board system the process can then be repeated for additional boards by answering *y* for “another board (y/n)”. In this example *n* is entered as

there is only one board. The remaining operations are similar to that of the C4x example:

Now that the system has been defined, the Main Menu now presents the following options:

```
DETROIT MENU
[a] de62_s1 Single C6x PCI Carrier 200 MHz (512 KB Global/256 KB Local
SRAM)
[b] de62_s2 Single C6x PCI Carrier 200 MHz (2 MB Global/256 KB Local
SRAM)
[c] de62_s4 Single C6x PCI Carrier 200 MHz (2 MB Global/512 KB Local
SRAM)
[d] Quit Menu

Enter Choice: b

board ID: [0x00000001] 0x10
another board? (y/n) n

MAIN MENU
[a] Create a system definition file
[b] Create script file
[c] Quit the ToolBox

Enter Choice:
```

“Creating a system definition file” creates a text file that describes the system and its connections previously defined. This text file can then be used with the Host application libraries that use SDF files. This option also generates an LDF file. Both of these files are output to:

c:\spectrum\toolbox\system (Win32)

~/spectrum/toolbox/system (Solaris)

Creating an SDF file also checks that there is a connection for each processor back to a root processor in the system.

```
MAIN MENU
[a] Create a system definition file
[b] Create script file
[c] Quit the ToolBox

Enter Choice: a

System definition file successfully parsed.

MAIN MENU
[a] Create script file
[b] Quit the ToolBox

Enter Choice:
```

A script file is a listing of the options entered at each stage of the configuration. This file is not used by any of the application libraries.

### 6.3. Example - Using the Generated Files in ALIB Code

The files output from ToolBox are used in developing C, VisualBasic, or other applications. The Programmer's Guides for your hardware outlines the applications that the ALIBs can be used for in development and have specific examples of using the configuration files. The following are some code excerpts of using the files in a C programming environment:

#### 6.3.1. RDF use for Maranello (Quad TIM Carrier)

For this example, note that the Maranello is called the CV8 for the API calls:

```
#include "mysystem.rdf"    /* Resource Definition File
*/

/* The following table lists the files to be loaded it
is output as part
of the RDF file and is copied into this file and
the "MyCode.ldr"
filenames added */
S_PROC_FILE_LIST CV8_SystemFiles[] =
{
    {"Board1:NodeA:Proc0", "blink.out"},
    {"Board1:NodeB:Proc0", "blink.out"},
    {"Board1:NodeC:Proc0", "blink.out"},
    {"Board1:NodeD:Proc0", "blink.out"},
    {"", ""}
};

int main(void)
{
    RESULT result;

    /* Step 1: open system */
    if (!result)
        result = CV8_SystemOpen( &hSystem,
                                (char *)MySystem,
                                0);

    /* Step 2: reset all processors */
    if (!result)
        result = CV8_Control( (HCV8_RESOURCE)hSystem,
                              CV8_CNTRL_RESET,
                              0,
                              NULL);

    /* Step 3: load DSP code onto target system */
    if (!result)
        result = CV8_SystemLoad(hSystem,
                                (char*)CV8_SystemFiles);

    etc...
```

### 6.3.2. SDF/LDF use for Detroit (Single C6x Carrier)

For this example, note that the Detroit is called the DE62 for the API calls:

```
main()
{
    RESULT rv = OK;
    DE62_SYSTEM hSystem;

    if (!rv)
        rv = DE62_SystemOpen(    &hSystem,
                                "mysystem.sdf",
                                NO_FLAGS);

    if (!rv)
        rv = DE62_Control( (DE62_RESOURCE)hSystem,
                            DE62_CONTROL_RESET,
                            NO_FLAGS,
                            NO_VALUE);

    if (!rv)
        rv = DE62_SystemLoad(   hSystem,
                                "mysystem.ldr",
                                NO_FLAGS);

    etc..
}
```

### 6.3.3. RDF use for Morocco (Octal Sharc Board)

For this example, note that the Morocco is called the V8 for the API calls:

```
#include "mysystem.rdf"    /* Contains resource
definitions */

/* The following table lists the files to be loaded it
is output as part
of the RDF file and is copied into this file and
the "MyCode.ldr"
filenames added */
S_PROC_FILE_LIST DevelopmentFiles[] =
{ /* BEGIN FILES: */
    {"Board1:Cluster0:Proc0", "MyCode.ldr"},
    {"Board1:Cluster0:Proc1", "MyCode.ldr"},
    {"Board1:Cluster1:Proc0", "MyCode.ldr"},
    {"Board1:Cluster1:Proc1", "MyCode.ldr"},
    {"Board1:Cluster2:Proc0", "MyCode.ldr"},
    {"Board1:Cluster2:Proc1", "MyCode.ldr"},
    {"Board1:Cluster3:Proc0", "MyCode.ldr"},
    {"Board1:Cluster3:Proc1", "MyCode.ldr"},
    {"", ""} /* End of List Indicator */
}; /* END FILES */

int main(void)
{
    V8_SYSTEM hSystem;
    V8_RESULT rv = OK;
}
```

```
/* Open the board for access */
if (!rv)
    rv = V8_SystemOpen( &hSystem,
                      (char *)MySystem,
                      V8_RESOURCE_TABLE);

/* reset */
if (!rv)
    rv = V8_Control((V8_RESOURCE)hSystem,
                  V8_CONTROL_RESET,
                  NO_FLAGS, NO_VALUE);

/* down load code to the 8 processors */
if ( OK == rv )
    rv = V8_SystemLoad( hSystem,
                      (char *)DevelopmentFiles,
                      V8_FILE_TABLE);
```

etc...



## 7 Appendix A - Modifying ToolBox Configuration Files

There are several files which are used to create the output configuration files from ToolBox. These files are found in the Global System Library (GSL) directory tree \gsl under the toolbox directory.

This directory structure is as follows:

Directory	Contents
gsl\	Base of the Global System Library
gsl\hardware\	Hardware configuration information
gsl\hardware\board\	Carrier board information
gsl\hardware\board\barcelona\	Barcelona carrier information
gsl\hardware\board\barcelona\ba62_v1\	Barcelona carrier Version 1 information
gsl\hardware\board\barcelona\ba62_v2\	Barcelona carrier Version 1 information
gsl\hardware\board\dakar\f5_m1s1\	Dakar carrier memory version 1 information
gsl\hardware\board\dakar\f5_m2s2\	Dakar carrier memory version 2 information
etc...	
gsl\hardware\tim\	Tim module information
gsl\hardware\tim\dab\	DAB tim module information
gsl\hardware\tim\dab\44dab_60\	DAB tim module information 60Mhz
gsl\hardware\tim\ss\	DSP1 tim module information
gsl\hardware\tim\ss\40ss2_60\	DSP1 tim module 2Mbyte 60Mhz information
gsl\hardware\tim\ss\40ss4_60\	DSP1 tim module 4Mbyte 60Mhz information
gsl\hardware\tim\ss\40ss5_50\	DSP1 tim module 5Mbyte 50Mhz information
gsl\hardware\tim\ss\40ss5_60\	DSP1 tim module 5Mbyte 60Mhz information
gsl\hardware\tim\ss\40ss6_50\	DSP1 tim module 6Mbyte 50Mhz information

gsl\hardware\tim\ss\40ss6\_60\ DSP1 tim module 6Mbyte 60Mhz information  
 etc...

There are several files in this structure:

File	Contents
gsl\toolbox.ini	Specifies to toolbox.exe all the available hardware that can be configured. Adding to this file will result in additional items being added to ToolBox menus
gsl\hardware\board\*\sdfdef.ini	If present this file is added without modification to the SDF file being output when a particular board or tim is selected from one of the menus and the SDF file is selected for output.
gsl\hardware\tim\*\sdfdef.ini	
gsl\hardware\board\*\sdf.ini	This file contains the carrier/tim specific SDF information for a particular model. Elements within this file are replaced with calculated values (those with 'UNINIT', 'BOARD_' in the name) to create the SDF file.
gsl\hardware\tim\*\sdf.ini	
gsl\hardware\tim\*\tim.cfg	This required file for each tim contains the information used to program PEROMs and to calculate the LMCR/GMCR for the SDF.
gsl\hardware\board\*\board.cfg	This required file for each carrier contains the information used for PEROMs and the calculation of the LMCR/GMCR for the SDF. If not a C4x carrier then this file can be empty, but still must exist.

There are two ways to create the configuration files for custom carriers: First, a set of files in the existing subdirectory can be modified, or a new set of subdirectories can be created and their information added to toolbox.ini. This

would allow ToolBox to still be used by calling the carrier/TIM model version that has been modified.

Second, an SDF file can be generated for a carrier/tim that has similar properties to that of the custom hardware and then the file edited using a text editor and then an RDF generated (if required).