TECHNOLOGY WHITE PAPER

Recursion C++ DSP Toolkit Libraries

DM6437 Cross Development

ABSTRACT

Developing complex applications, such as video, and keeping them adaptive is a real problem in the Digital Signal Processing area. Easy to use, high performance tools can alleviate this problem to a large extent. Recursion C++ Toolkit is a commercial off-the-shelf product that provides a wide range of features for application developers using the Texas Instruments DM6437 and other similar processors. These features provide both easy abstractions to the complex OS layer APIs and are also a high performance enhancement to the native compilers in the TI Code Composer Studio. This white paper provides a brief description of the Recursion C++ Toolkit, its advantages, and coding samples.

Sponsored by

RECURSION SOFTWARE, Inc.

DSP Development Challenges

Video applications are in major demand now. These applications enable key business areas such as video security, digital transmission of sporting events and media events, video collaboration, gaming and many others. Video applications are typically developed on an embedded operating system such as Texas Instruments DSP/BIOS using DSP Processors such as the DM6437.

Application development for embedded real-time systems has always been a tedious task. It involves a cross compiler on a host computer, downloading the file to a target machine, debugging the system on the target machine from a remote computer, fixing the issues, and repeating the steps. The amount of time involved in the development of the application can be dramatically reduced if the application can first be compiled and debugged on the host machine to verify most of the functionality. All applications involve a good deal of generic code like list manipulation, TCP/IP communications, reading/writing files, exchanging data between tasks, etc. Most of this generic functionality can be tested on the host computer if there is an API that can abstract the differences between the host operating system (for example MS Windows) and a target operating system (for example DSP/BIOS). Such APIs are rare in general and almost non-existent when it comes to DSP/BIOS.

DSP Solutions

The DSP/BIOS port of the Recursion's C++ Toolkit offers a great solution to this problem.

Figure 1 shows the architecture of the Recursion C++ Toolkit. The application was developed using the Recursion C++ Toolkit. The C++ Toolkit internally uses the features provided by the compiler and the operating system and provides a uniform API interface across all platforms and compilers. To evaluate the C++ Toolkit and follow the code samples, download a free evaluation version at recursionsw.com.

In cross development scenarios, such as DSP/BIOS, the developers can code and test a majority of their application on their host platform before moving their code to the target platform.

Figure 1: Recursion C++ Toolkit Stack

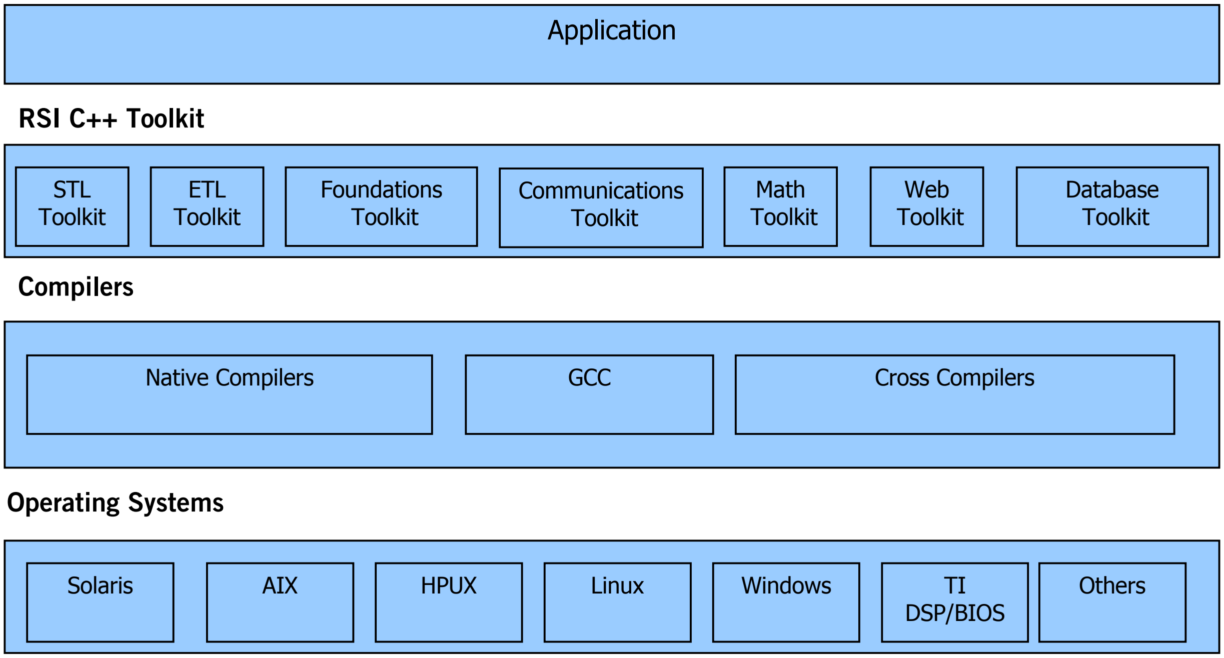


Figure 2 shows the internals of the Recursion C++ Toolkit. The Toolkit consists of:

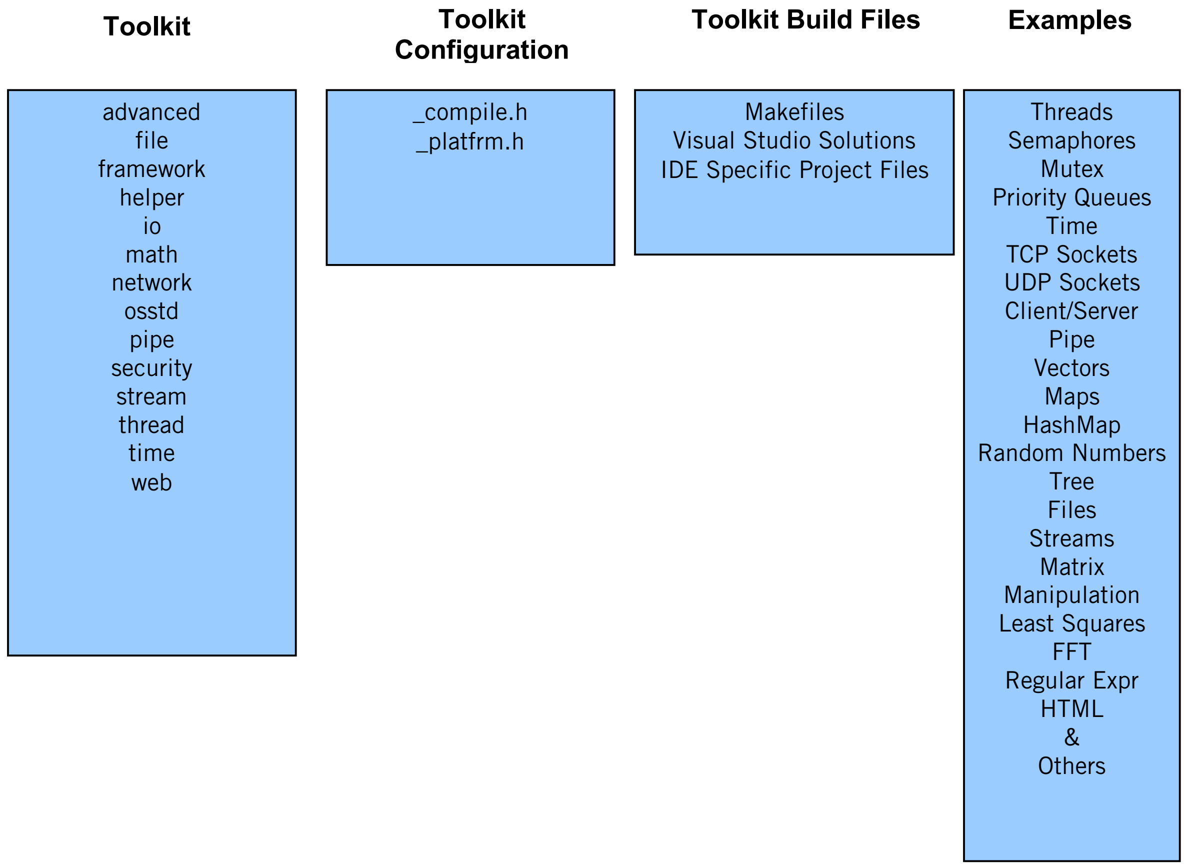
1. **Toolkit Source Code:** The source of the entire C++ Toolkit is provided to the developers. The most commonly used approach in the industry is to provide source code for templatized modules and provide the key intellectual property as libraries. Recursion does not make this differentiation. The developers can use this source code for debugging purposes when they hit complex issues.
2. **Toolkit Configuration:** As shown in Figure 1, the toolkit works on many different platforms. The configuration for the different platforms is stored in two key files (_compile.h & _platfrm.h). The configuration provides the various settings necessary to compile code that is applicable to various platforms.
3. **Toolkit Build Files:** The build files for various platforms such as Makefiles (for unix, linux, etc.), Visual Studio Solution Files and others.
4. **500+ Examples:** The toolkit provides a broad range of examples, more than any other product available. These examples enable the developer to self-train on the features and API of the toolkit. The examples can be used as-is in many development scenarios to enhance productivity.

TI DSP/BIOS developers can reap great benefits in terms of time by using the Recursion C++ Toolkit. The toolkit offers the same set of API for Windows, Linux, DSP/BIOS and other platforms (with exception for a few limitations).

The advantage of using the Recursion C++ Toolkit can be seen in the following simple thread creation example.

- **Figure 3** shows the code required to create a thread on Windows platform. It should be noted that all the required variable settings are not shown for brevity.

Figure 2 RSI C++ Toolkit Internals



Toolkit

advanced
file
framework
helper
io
math
network
osstd
pipe
security
stream
thread
time
web

Toolkit Configuration

_compile.h
_platfrm.h

Toolkit Build Files

Makefiles
Visual Studio Solutions
IDE Specific Project Files

Examples

Threads
Semaphores
Mutex
Priority Queues
Time
TCP Sockets
UDP Sockets
Client/Server
Pipe
Vectors
Maps
HashMap
Random Numbers
Tree
Files
Streams
Matrix
Manipulation
Least Squares
FFT
Regular Expr
HTML
&
Others

- **Figure 4** shows the code required to create a thread on DSP/BIOS.
- **Figure 5** shows an example of creating a thread using Runnable model in Recursion C++ Toolkit.
- **Figure 6** shows the code required to create a thread using Recursion C++ Toolkit.

The following advantages can be easily seen from the code samples.

1. The Recursion Toolkit code accomplishes the task in one line as compared to the many lines on Windows and DSP/BIOS
2. The Recursion Toolkit captures the errors and creates an easy to use exception.
3. For simple thread creation, the Recursion Toolkit internally assigns default values for all thread parameters.
4. The Recursion Toolkit does not require C functions to be used as wrappers for C++ thread classes.
5. Recursion Toolkit provides a runnable model to create threads.

Figure 3 Sample Thread Creation on Windows

```
ANDLE  thread_id;
 int tid;

          thread_id = (HANDLE),
          ::CreateThread (0, , stack_size, start_fxn,  _arg,
          0, // initflag (0 = running, 1 = suspended)
          (unsigned long*) &tid_);


if (GetLastError() != 0)
{
     throw “Error in thread creation ….”;
}
```

Figure 4 Sample Thread Creation on DSP/BIOS

```
TSK_Attrs attrs = TSK_ATTRS;
    TSK_Handle thread_id;

            if(stack_size != 0)
            {
            attrs.stacksize = stack_size;
            }
            else
            {
            attrs.stacksize = DEFAULT_STACK_SIZE;
            }
            attrs.priority = DEFAULT_THREAD_PRIO;
            attrs.initstackflag = TRUE;
```

```
   thread_id = TSK_create((Fxn)start_fxn, &attrs,  _arg);

   if(thread_id == NULL)
         {
         throw “Error in thread creation ….”;
         }
```

Figure 5 Sample Thread Creation using Recursion C++ Toolkit

```
os_thread_toolkit initialize;

try
{
  os_thread t1( start_fxn ); // Spawn new thread.
}
catch(os_thread_toolkit_error err)
{
….
}
```

Figure 6 Sample Thread Code using Recursion C++ Toolkit

```
class os_printer
  {
  public:
    void* run();
  };


void*
os_printer::run()
  {
  for ( int i = 0; i < 7; i++ )
    {
    os_this_thread::sleep( 1 ); // Sleep for 1 second.
    }
  return 0; // Ignored in this example.
  }


int
test_thread ()
  {
  os_thread_toolkit initialize;
  // Create thread through runnable interface.
  os_printer runnable;
  os_thread t1( &runnable ); // Spawn new thread.

  os_this_thread::wait_for_thread( t1 );

  return 0;
  }
```

The advantages seen above in thread creation can be seen in all the different modules of the C++ Toolkit. A development team will have to invest several man-years to create their own set of classes equivalent to the C++ Toolkit classes before they can start development of critical application modules. Development managers can benefit by investing this effort in improving the functionality and quality of the video application they are developing.

In many situations, the final configuration of the target is not available in the initial stages of development. Developers have to compete (or timeshare) for access to the target platform. The common API of the Recursion C++ Toolkit across platforms adds more value to developers and development managers.

Developers can compile and test a majority of their application on the host platform. **Figure 7** shows the development of an application using the toolkit on a Windows Host using Microsoft Visual Studio. The application and the toolkit can be compiled using the Windows configuration files. The application can be run and tested on Windows.

The toolkit can be recompiled for DM6437 target processor by using the DM6437 configuration files, as shown in **Figure 8**. The application compiled for target processor can be loaded on to the TI board using the code composer studio. As mentioned before, the toolkit source code is also available to the user during debugging.

Figure 7 Application Development on Host

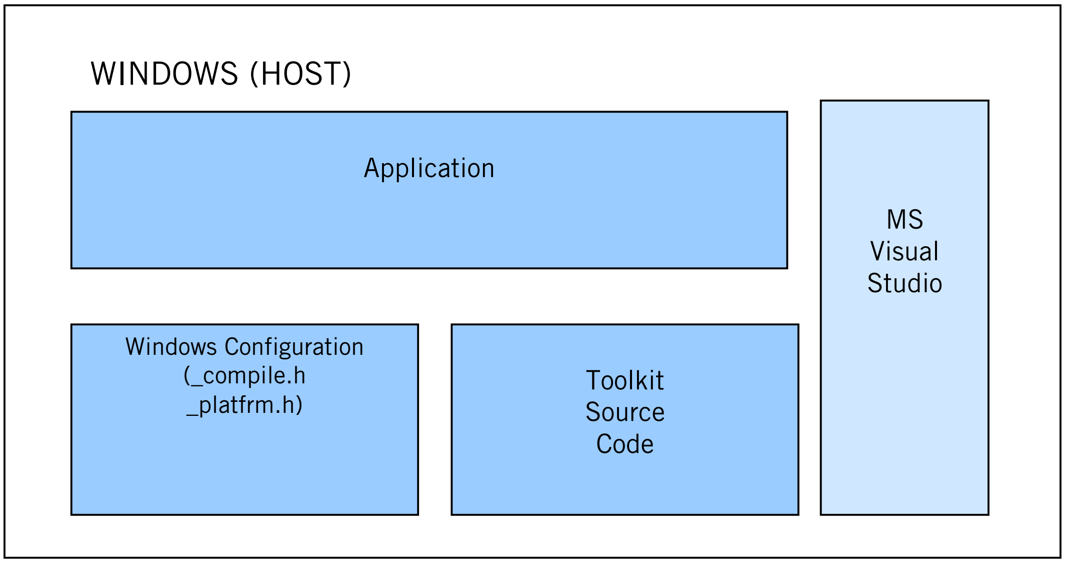
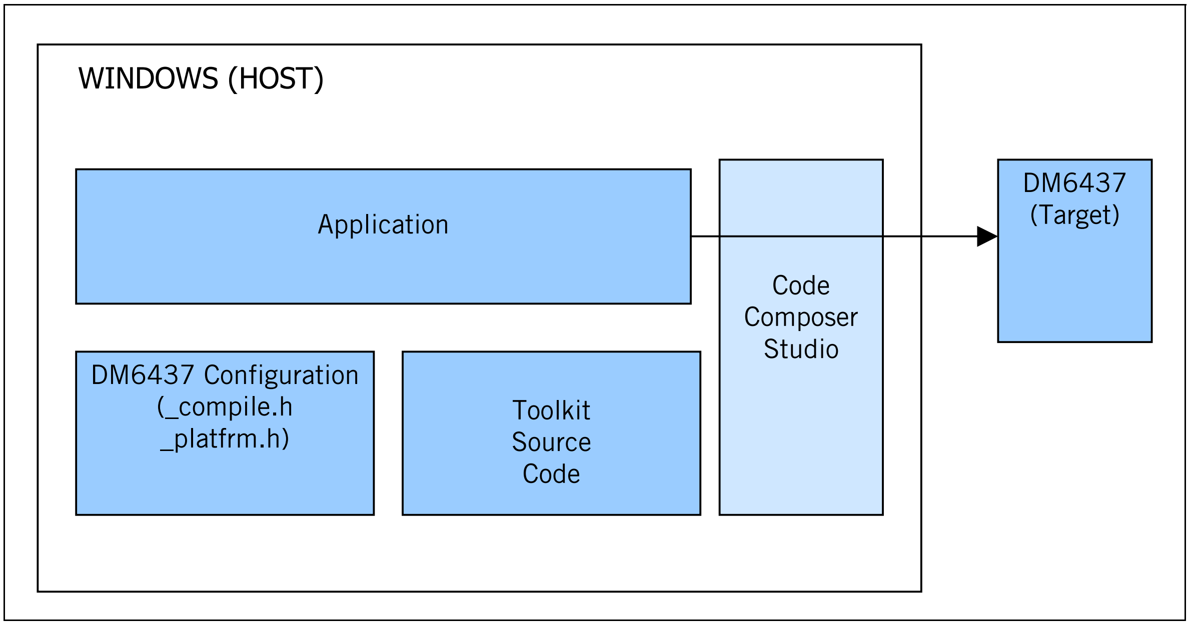


Figure 8 Application Debugging on Target (DM6437)



Summary

Recursion's C++ Toolkit is a multi-platform toolkit that has been ported over the 15 years onto a variety of operating systems. Recently, the Toolkit was ported to the TI DSP/BIOS on the DM 6437 platforms.

The Recursion Toolkit comes with complete source code and can be easily compiled to other variants of the processor. The Recursion Toolkit comes with many examples (almost 500). These examples are based on real world scenarios. They are both (a) excellent sources for training a new user of the Toolkit and (b) excellent references for real world scenarios that occur during programming. ■

About Recursion Software, Inc.

Recursion Software is an innovative provider of intelligent middleware and distributed computing solutions based on Service Oriented Architecture (SOA) principles and interoperability standards. Since 1993, our products have enabled enterprises to extend their current application architecture while providing the tools developers need to build the next-generation of intelligent, mobile applications. The company is a small, privately held corporation, located in the Dallas-Fort Worth area.

Recursion Software is regarded for its Voyager ONE platform, a powerful agent-based interoperable platform that supports a total range of edge devices, including handheld devices, PDAs, sensors, cameras, and other wireless devices. The company remains the leading proponent and preferred platform for intelligent mobile agent and agent community technology and has been issued more than 18 patents related to distributed computing, with 30 patents in various states of pending and filing.

For more information, visit our website at recursionsw.com