SIMPLIFIED INSTRUMENT/APPLICATION DEVELOPMENT AND SYSTEM INTEGRATION USING LIBERA BASE SOFTWARE FRAMEWORK

Matej Kenda, Tomaž Beltram, Tomaž Juretič, Borut Repič, Damijan Škvarč, Črt Valentinčič,
Instrumentation Technologies, Solkan, Slovenia

Abstract

Development of many appliances used in scientific environment forces us to face similar challenges, often executed repeatedly. One has to design or integrate hardware components. Support for network and other communications standards needs to be established. Data and signals are processed and dispatched. Interfaces are required to monitor and control the behaviour of the appliances. At Instrumentation Technologies we identified and addressed these issues by creating Libera BASE, which is a framework composed of several reusable building blocks. Libera BASE simplifies some of the tedious tasks and leaves more time to concentrate on real issues of the application. Further more, the end product quality benefits from larger common base of this framework. We present the benefits on examples of instrument implemented on MTCA platform accessible over graphical user interface.

DEVELOPMENT OF ELECTRONIC DEVICES

The need or a vision for a new or improved electronic devices usually emerges from a real-world challenge, issue or a question that needs to be answered.

The device gets defined through the development process in the means of functional (functionality, performance) and non-functional (user experience, scalability, interoperability) requirements. It is then realised through analysis, design implementation, validation and production. This is a day-to-day job in development enterprises all over the world.

For the purpose of accelerator community, the end result of the development is often a measurement instrument, comprising:

- selected hardware architecture (for example Libera hardware architecture A/B, MicroTCA)
- instrument specific electronics, RF
- input signals (analog, digital, timing)
- platform management
- imposed communication protocols
    - PCI, IPMI, Ethernet, SATA, RapidIO
    - Control System protocols, ...
- generic FPGA cores and application specific processing
- application-specific algorithms, parameters, ...

THE ROLE OF SOFTWARE IN RECONFIGURABLE ELECTRONIC DEVICES

The importance of software in electronic devices is increasing. Software is not just an add-on, it became an essential part of electronic devices.

Many of the chips are becoming increasingly integrated and programmable or configurable to some extent. For example FPGA by its nature, ADCs, VCXOs that are controlled over SPI, I2C buses. As a consequence, more software needs to be written to control them.

Software usually acts as the integration layer to make hardware components, application logic and user interfaces to work together. Issues not thought of before the integration need to be resolved then.

Software interfaces are also the points where people communicate with the device. They largely define user experience through graphical, programming and other interfaces. Human behaviour also doesn't comply to standards: any kind of usage of the electronic device needs to be handled in software [1].

REUSABLE SOFTWARE IN MEASUREMENT INSTRUMENTS: SEEING THE PATTERNS

There is a set of concepts that is occurring repeatedly in measurement instruments:

- hardware detection, platform management
- access to functionality implemented in FPGA
- configuration parameters
- notification of changes
- signal acquisition, processing and dispatching
- scaffold for running instrument applications
- supporting standard control system interfaces

Figure 1: Libera BASE logo.

LIBERA BASE

Libera BASE (Figure 1) is a software framework, developed at Instrumentation Technologies. The design

started in the beginning of 2010 based on many years of previous experience. A common framework was internally needed to develop multiple instruments/applications on the same hardware architecture.

The project's delivery was software framework Libera BASE, which proved itself to be useful widely as initially anticipated [10].

Libera BASE narrows the gap between customer's hardware and the machine's control system. It helps to focus on the application, for which the instrument is designed for with

- software framework for application development and
- intuitive structure and programming interfaces.

Libera BASE simplifies integration into various control systems, however it does not aim to act as a replacement for them.

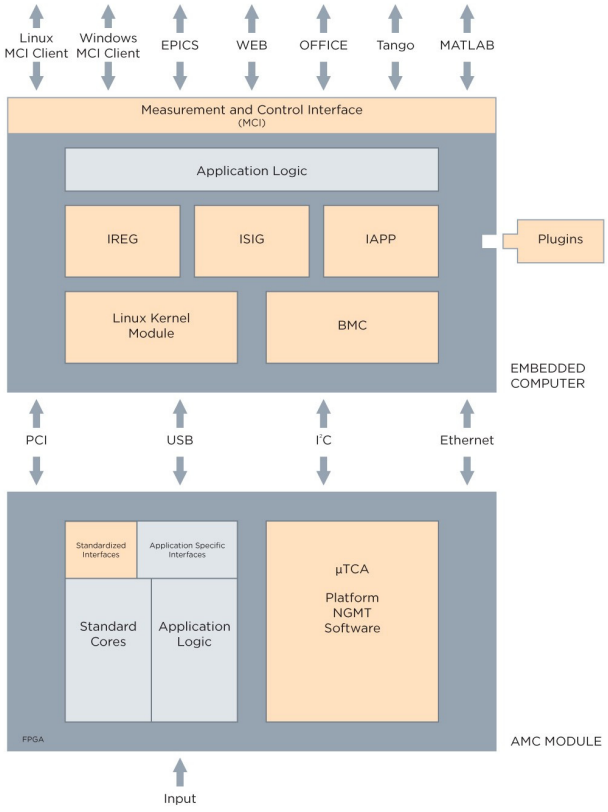

Figure 2: Libera BASE software framework.

Concepts and Building Blocks

Libera BASE consists of several components (Figure 2) that work together in synergy to support applications based on it ([2], [3], [4]).

- **fw**: MicroTCA-compliant platform management
- **bmc**: Hardware abstraction layer (uses IPMI, USB, OpenHPI)
- **lkm**: PCI Linux kernel module relies on a set of standardised FPGA registers
- **ireg**: Application parameters as hierarchical tree
- **isig**: Signal acquisition, processing and dispatching
- **iapp**: Application development framework, plugins
- **mci**: Client programming interface (API) for GNU/Linux and Windows: exposes registry and access to signals
- **tools**: Simple command line tools for automation and scripting
- **adapters**: Matlab and LabView scripts, web, EPICS (EDM, pyEpics) [8], Tango CS [7], FESA [6]

Relation to Libera Instruments

Using Libera BASE to develop instrument application software accelerates development. Measurements of source code size reveal that the share of application-specific software is around 10% of total software; remaining 90% is Libera BASE.

Application software defines the set of user settable parameters, signals, processing and algorithms (Figure 3, Figure 4).

In September 2011, the following instruments use Libera BASE: Libera Brilliance+, Libera Single Pass H, Libera Spectra, Libera LLRF.

Common MCI API is used on all of the instruments which simplifies integration of different instruments into the control system.

Libera BASE is improved incrementally with each new instrument or its new version. Occasionally there are dedicated projects scheduled to develop common functionality.

Improvements of Libera BASE during development of one instrument get incorporated into other instruments on regular basis.

This synergy between the instruments results in better stability and quality of many instruments over time.

```
$ ./libera-ireg dump -h 10.0.3.106 boards.tim2
info
  revision=11650
  health_status=16
clock_info
  adc_frequency=117418690
pll
  locked=true
  clk_good=true
  unlock_thr=100
  vcxo_offset=0
  compensate_offset=false
  pi_coarse
  pi_fine
events
  trigger=61346545890899
  current_time=61346625145926
signals
  pll
    access_type=Stream
    atom_size=64
    group_size=1
    components_names=Err,Dac,Lock,ClkGood,Freq,P,I,IsCoarse
    components_number=8
    components_type=Double
  event
    access_type=Stream
    atom_size=32
    group_size=1
    components_names=id,count,timestamp,reserved
    components_number=4
    components_type=uint64
sc_source=Internal
```

Figure 3: Example of parameters in the registry from Libera Brilliance+.

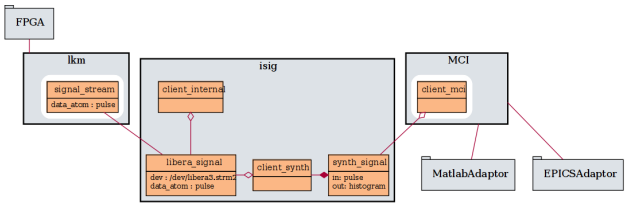

Figure 4: Processing and dispatching a signal.

Libera BASE and Hardware Architectures

Software framework was initially designed to support the modular Libera Hardware Architecture B. Since the coupling between the modules of Libera BASE is low, it is relatively easy to support different hardware architectures with the same software framework. The framework currently supports:

- Libera Hardware Architecture B
- Micro TCA
- Micro TCA for Physics (MTCA.4) [5]

PICMG (PCI Industrial Computer Manufacturers Group [9]) is working on a reference software implementation for applications on MTCA.4. Libera BASE was created sooner and independently from the MTCA.4 reference implementation, because of the market needs.

MTCA.4 reference implementation and Libera BASE share the same mission: help to focus on the application, not the platform.

Libera BASE was presented to PICMG to share ideas and will adapt to the MTCA.4 reference implementation when available.

Libera BASE and Libera Platform A

Instruments based on Libera Platform A provide CSPI API for integration with control systems.

CSPI is only an API, however Libera BASE is a complete framework for application development.

CSPI was designed for a single hardware architecture and instrument. Libera BASE can be adapted to support Libera hardware architecture A, but the opposite would be harder.

Future Plans

Feature set of Libera BASE is becoming rich enough to be able to implement many applications on different hardware architectures.

In the next period it is intended to improve out-of-the box connectivity, abilities to customise/extend the applications and usability.

USAGE SCENARIOS OF LIBERA BASE

Development of New Instruments

Libera BASE software framework provides a productive environment for development of a new instrument.

Integration of Instruments into their Working Environment

The MCI programming interface is designed to be simple, intuitive and powerful. Different instruments expose the same networked API, which can be used from GNU/Linux and Windows clients (Figure 6).

The API is used to implement different client programs (command line, graphical) and adaptors for integration with applications and control systems.

Different instruments have already been accessed from the following clients and adaptors:

- graphical user interface
- command line interface
- Matlab, SCILab
- OpenOffice.org spreadsheet
- mobile devices: iPhone, iPad, Android-based phones (Figure 5)
- web browsers
- EPICS EDM

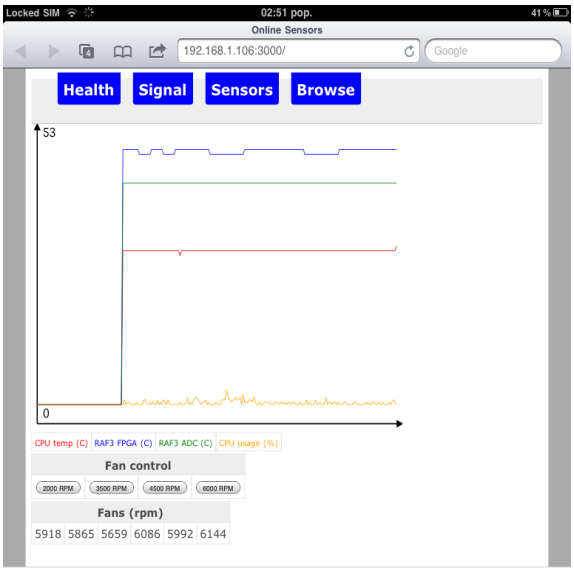

Figure 5: Controlling an instrument from a mobile device.

Customising and Extending an Instrument

Several tools are provided to customise Libera instruments:

- FPGA development kit (FDK)
- FPGA to registry map (no programming needed)
- Software plugins

New parameters and signals implemented when extending the instrument are exported through MCI API in the same way as those originally provided by the instrument.

Source Code Example

```cpp
#include <iostream>
#include "mci/mci.h"
#include "mci/node.h"

int main(int a_argc, char* a_argv[])
{
    mci::Init(a_argc, a_argv);

    auto root = mci::Connect("192.168.1.100", mci::Root::Application);

    //
    // Dump complete tree of registry parameters
    //
    auto nodes = mci::SubTree(root);
    for (auto i(nodes.begin()); i != nodes.end(); ++i) {
        std::cout
            << i->GetFullPath() << " = "
            << i->ToString() << std::endl;
    }

    //
    // Access and modify a parameter
    //
    mci::Path path = mci::Tokenize(
        "boards.raf3.tbt.spike_removal.averaging_window");

    auto n = root.GetNode(path);

    // Read a numeric parameter
    int32_t aw = n;
    std::cout << "Averaging window: " << aw << std::endl;

    // Modify a numeric parameter
    aw = 16;
    n = aw;

    mci::Shutdown();
}
```

Figure 6: Example of C++ source code.

SUMMARY

Libera BASE narrows the gap between your hardware and the machine control system. It simplifies development of instruments and integration into control systems.

Libera BASE provides MCI programming interface that is powerful, easy to learn and use.

The software framework is designed for various hardware technologies.

High level of re-usability increases reliability and quality of instruments.

Supports reconfigurable instruments with its modular structure and extendibility.

REFERENCES

[1] Matej Kenda, Hinko Kočevar, Tomaž Beltram, Aleš Bardorfer , “Programming Interfaces for Reconfigurable Instruments ”, PCaPAC 2010, Saskatoon, WEPL032, October 2010.

[2] Gamma, Helm, Johnson, Vlissides, “Design Patterns”, Addison-Wesley, 1995.

[3] http://en.wikipedia.org/wiki/Software_design

[4] http://en.wikipedia.org/wiki/Software_framework

[5] “Specification MTCA.0”, PICMG, July 2006

[6] T. Hoffmann, GSI, “FESA - The Front-end Software Architecture at FAIR”, PCaPAC 2008, Ljubljana, WEP007, October 2008.

[7] www.tango-controls.org

[8] www.aps.anl.gov/epics/

[9] www.picmg.org

[10] Instrumentation Technologies, “Libera BASE home page”, www.i-tech.si/accelerators-instrumentation/technologies/libera-base.