15th International Particle Accelerator Conference,Nashville, TN JACoW Publishing
ISBN: 978-3-95450-247-9 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2024-TUPR83

PSI'S OPEN-SOURCE FPGA DSP LIBRARIES

B. Stef*, O. Bründler, J. Purtschert, R. Rybaniec, Paul Scherrer Institut (PSI), Villigen, Switzerland

Abstract

PSI has led significant advancements in accelerator electronics development, leveraging Field Programmable Gate Arrays (FPGA) based Digital Signal Processing (DSP) across various critical systems, including Low Level RF (LLRF), Longitudinal Beam Loss Monitoring (LBLM), charge particle measurement via Integrating Current Transformers (ICT), Timing, Filling Pattern Monitor (FPM), Beam Position Monitor (BPM) and other essential beam instruments. Over the past decade, PSI's approach to develop in-house control system platform (e.g. CPCI-S.0), has encouraged innovation. The strategic reorganization within PSI, fostering collaboration among FPGA firmware engineers, led to the inception of Open-Source FPGA DSP libraries hosted on GitHub. Serving as a comprehensive repository, these libraries empower developers by providing common FPGA IPs, fundamental DSP algorithms and Fixed-Point (FP) arithmetic units. Their presence advances prototype development by enabling rapid assembly of several measurement and or control concepts. In this contribution, we present the features and the transformative impact of the PSI Open-Source FPGA libraries with a focus on LLRF. This initiative has not only empowered our team to provide valuable insights, but has also streamlined the integration of new recruits and students, enabling the seamless continuation of FPGA design frameworks.

INTRODUCTION

At PSI, exploiting synergies across teams and projects enhances productivity and efficiency. Transitioning from a siloed structure to a collaborative, matrix-based approach optimizes solutions, reduces maintenance dependency, and speeds up project completion.

Despite numerous initiatives to create open-source FPGA resources [1], custom in-house development continues to offer better adaptation, standardization and support. Given the scarcity of FPGA design engineers, our group tackled this challenge by developing an open-source library under a modified LGPL license, hosted on GitHub [2]. This approach fosters collaboration, focusing on high-quality architecture while reducing risks of bugs and maintaining consistent VHDL entities. In this paper, we discuss the new electronic development at PSI, showcasing the ideal platform for reusing library elements across multiple particle accelerator projects. We also explore the library's features, its impact on the FPGA developer community, and the maintenance of the repositories.

* benoit.stef@psi.ch

CPCI-S ELECTRONIC DEVELOPMENT

The SLS 2.0 project at PSI [3] leverages a new control system electronic toolbox based on CompactPCI Serial (CPCI-S.0) [4], featuring a powerful FPGA board equipped with the Zynq Ultrascale+ (Zynq US+) Multi-Processor System on-Chip (MPSoC) [5]. The in-house platform supports other sections within PSI by providing the tools and resources needed for specific applications [6]. The modular and scalable design of the toolbox facilitates development and ensures compatibility across various projects.

CORE LIBRARIES AND FEATURES

PSI's Open-Source FPGA DSP libraries offer a comprehensive suite of core entities that facilitate efficient and flexible RTL designs for developers. The **psi common** library includes presently 54 components compatible with different FPGA vendors, featuring functions like memory management, clock domain crossing, and data stream handling. The **psi fix** library provides 30 components based on a package designed by *Enclustra GmbH* which includes FP arithmetic and DSP elements such as FIR, IIR, CIC filters, DDC, DDS, and complex arithmetic (see Table 1). In addition a Python library is also available to perform bittrue analysis.

Table 1: psi fix RTL Components

entity	component description
psi_fix_bin_div	binary division
psi_fix_cic_dec...	CIC decimation filters
psi_fix_cic_int...	CIC interpolation filters
psi_fix_comparator	simple comparator
psi_fix_complex...	abs, add, sub, mult func
psi_fix_cordic...	CORDIC rot. or vector mode
psi_fix_dds	sinewave RF generator
psi_fix_demod	Non IQ demodulator
psi_fix_mod	Non IQ modulator
psi_fix_fir...	diverse FIR filters
psi_fix_inv	1/X calculation
psi_fix_lin_approx...	Sin, SQRT, Gaussify
psi_fix_lowpass_iir	IIR first order settable
psi_fix_mov_avg	moving average filter
psi_fix_pol2cart_approx	high speed pol2cart func
psi_fix_sqrt	Square root func

Both libraries include extended documentation and testbenches for regression testing. This careful testing ensures that changes do not compromise compatibility or functionality, giving developers confidence in their work and seamless integration of the libraries into their projects.

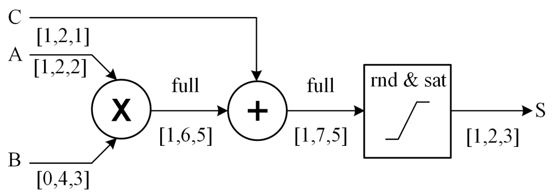

Figure 1: Multiplier and adder with FP format defined [sign bit,integer bits, fractional bits].

Fixed Point Arithmetic and psi fix Package

FP arithmetic is often the preferred approach for DSP in FPGA applications due to its reduced latency of processing and optimized resource usage compared to floating point calculations. Implementing FP arithmetic in VHDL can be challenging due to the need for careful selection of bit width and scaling factor to balance precision and logic resource usage. Handling overflow, underflow, truncating and rounding demands meticulous design to keep required precision. Additionally, debugging and testing FP operations in VHDL can be complex, as developers must account for the placement of the binary point and potential errors from truncation or rounding.

The following code shows a traditional VHDL code realizing the design from Fig. 1. It introduces bit manipulations that are not obvious and difficult to maintain.

```
--declaration
signal A   : slv( 4 downto 0); -- FP : (1,2,2)
signal B   : slv( 6 downto 0); -- FP : (0,4,3)
signal C   : slv( 3 downto 0); -- FP : (1,2,1)
signal m   : slv(12 downto 0); -- FP : (1,6,5)
signal sum : slv(12 downto 0); -- FP : (1,7,5)
signal rad : slv(12 downto 0); -- FP : (1,7,5)
signal rnd : slv(10 downto 0); -- FP : (1,7,3)
signal sat : slv( 5 downto 0); -- FP : (1,2,3)
begin
 m   <= slv(signed(A) * signed('0' & B));
 sum <= slv(signed(m) + signed( C &"0000"));
 rad <= slv(signed(m) + 4);
 rnd <= rad(12 downto 0);
 if signed(rnd) > 31 then
    sat <= "011111";
 else
    sat <= rnd(5 downto 0);
 end if;
end;
```

The **psi fix** library enables developers to simplify their algorithms, accelerate the simulation process, and improve code readability. Each block includes a Python model to facilitate rapid modeling, allowing for FP precision adjustments and enhanced verification. The subsequent code employs the **psi fix** package and new format type that ease FP handling.

```
use work.psi_fix_pkg.all;
...
--declaration
constant A_fmt : psi_fix_fmt_t := (1,2,2);
constant B_fmt : psi_fix_fmt_t := (0,4,3);
...
-- automnatic scaling of internal formats
constant m_fmt_t : psif_fix_fmt_t :=
    (max(A_fmt.s, B_fmt.s), A_fm_t.i+B_fmt.i,
    A_fmt.f+B_fmt.f);
signal A : slv(psi_fix_size(A_fmt)-1 downto 0);
signal B : slv(psi_fix_size(B_fmt)-1 downto 0);
signal m : slv(psi_fix_size(m_fmt)-1 downto 0);
...
begin
 m <= psi_fix_mult(A, A_fmt, B, B_fmt, m_fmt);
 S <= psi_fix_add(m, m_fmt, C, C_fmt, S_fmt,
     rnd, sat);
end;
```

Next is an example of the **psi fix** Python package that uses similar syntax.

```
from psi_fix_pkg import *
import numpy as np
...
 def model(A:np.array, B: np.array, C: np.array)
     -> np.array:
   m = psi_fix_mult(A, A_fmt, B, B_fmt, m_fmt);
   S = psi_fix_add(m, m_fmt, C, C_fmt, S_fmt,
       rnd, sat);
   return S
```

Library Use Case at PSI

The development of the libraries started with High Intensity Proton Accelerator Facility (HIPA) LLRF project renewal. By employing DSP techniques and control algorithms, LLRF systems adjust the amplitude and phase (A/Phi) of RF signals, ensuring optimal beam performance and stability. FPGA technology is often leveraged to enhance the speed and flexibility of digital LLRF systems, enabling precise adjustments and feedback control. These systems integrate seamlessly with other beam diagnostic tools, such as beam position and current monitors, to provide comprehensive monitoring and control.

The design requires a DDC with CORDIC to provide A/Phi information to high-level software applications across 8 channels. Additionally, the system regulates A/phi within the cavity using independent PI controllers. The controlled signals are up-sampled and modulated before being forwarded to the DAC output at a sampling rate of 250 Msps. Other features include error signal monitoring and fast statistical processing integrated into the programmable logic.

As part of this project, significant efforts were invested in developing each basic block with a high level of genericity to enable reuse in other projects, depicted as grey colored boxes in Fig. 2.

The same DSP chain have been entirely reuse for MESTRA electronics [7] as well for DBPM3 [8] device with different parameters. Table 2 outlines the spread of PSI fix elements across PSI's electronic equipment following

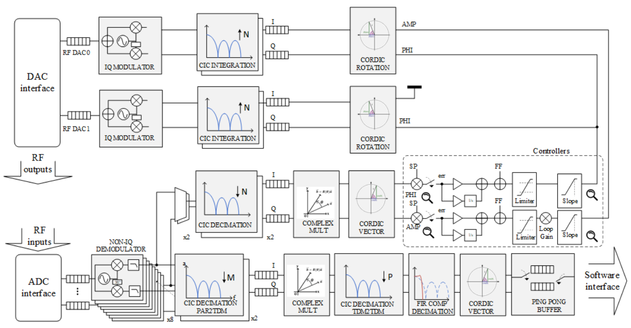

Figure 2: HIPA LLRF signal processing in FPGA.

Table 2: Library in Use at PSI GFA Department

Facility	Device	Platform	Target
HIPA	LLRF	VME + IFC1210	Virtex 6
HIPA	RF ILK	Custom	Spartan 6
HIPA	MESTRA	VME + IFC1210	Virtex 6
HIPA	ILK	VME + ILK	Artix 7
SLS 2.0	LBLM	CPSI-Serial	Zynq US+
SLS 2.0	LLRF	CPSI-Serial	Zynq US+
SLS 2.0	RF ILK	Custom	Spartan 6
SLS 2.0	PCT	CPSI-Serial	Zynq US+
SLS 2.0	BPM	DBPM3	Zynq US+
SLS 2.0	Timing	CPSI-Serial	Zynq US+
SwissFEL	RF ILK	Custom	Spartan 6

an internal workshop to promote the resources within the group.

IMPACT ON THE FPGA DEVELOPER COMMUNITY

Empowering Developers

The libraries empower developers at multiple levels, creating a comprehensive community that spans internal groups, the wider PSI organization, and the global professional community [9]. At the most immediate layer, members of the DSP group benefit from a shared repository of well-documented set of entities and DSP blocks, allowing for more efficient collaboration and rapid prototyping. It accelerates project development and promotes consistency and standardization across different efforts within the group and beyond.

The traffic data in Fig. 3, indicates daily activity on the repository; however, obtaining user feedback and contributions remains challenging despite active promotion. Maintaining the repository also places additional demands on maintainers to meet user expectations. Compared to the previous discussion of the library at LLRF22 [10], the viewership has more than doubled, reflecting ongoing interest and engagement, with 42 new clones of repository recorded over the same period.

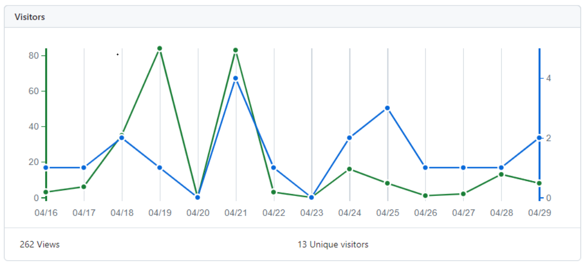

Figure 3: Repository traffic overview for **psi common**.

Onboarding and Training

Incorporating the libraries also greatly benefits onboarding and training processes for new recruits and students. By providing a comprehensive repository of common VHDL entities, fundamental DSP algorithms, and FP arithmetic units, the libraries offer a clear and structured learning pathway for beginners. This accelerates the integration of newcomers, enabling them to quickly grasp the principles of FPGA development at PSI. Moreover, the availability of well-documented, open-source codes serve as a valuable educational resource, allowing trainees to study and experiment with established designs. This hands-on experience promotes a deeper understanding of complex concepts.

CONCLUSION

The PSI's Open-Source FPGA DSP libraries hosted on GitHub provide developers with a comprehensive suite of resources that enhance efficiency in FPGA design and DSP applications. The integration of the new CPCI-S.0 platform with the Zynq US+ MPSoC in the SLS 2.0 project has streamlined control system design, offering powerful, flexible, and compatible solutions across various PSI projects.

Key findings from this work demonstrate the value of adopting a unified FPGA design framework and of open-source collaboration. The pros include accessibility, standardization, efficiency, quality, community feedback, and improved training and onboarding. However, there are also challenges such as potential compatibility issues, varying quality of contributions, dependency management, and maintenance requirements. The use of the new FP package helps considerably in FW maintenance and changes.

Future directions include continued development and expansion of the libraries, incorporating feedback from the broader community to drive improvements. PSI's focus on creating a shared resource base and promoting knowledge transfer will pave the way for further progress and transformative applications in accelerator and control system technology.

ACKNOWLEDGEMENTS

Many thanks goes to the support from PSI GFA/AEK department and contributors, the transfer technology group to make the library licensing process and last but not least *Enclustra GmbH* for the development and use of the libraries to make it robust, documented and reliable.

ISBN: 978-3-95450-247-9 ISSN: 2673-5490 doi: 10.18429/JACoW-IPAC2024-TUPR83

REFERENCES

[1] T. B. Preußer, M. Zabel, P. Lehmann, and R. G. Spallek, ”The portable open-source IP core and utility library PoC,” 2016 International Conference on ReConFigurable Computing and FPGAs (ReConFig), Cancun, Mexico, 2016, pp. 1-6, doi:10.1109/ReConFig.2016.7857191

[2] psi_fpga_all library repository, https://github.com/paulscherrerinstitute/psi_fpga_all

[3] SLS 2.0 project page at PSI, https://www.psi.ch/en/sls2-0

[4] PCI standard, https://www.picmg.org/openstandards/ compactpci-serial

[5] P. Pollet, B. Kalantari, W. Koprek, D. Maier-Manojlovic, and G. Theidel, ”Development of a CompactPCI-Serial Hardware Toolbox for SLS-2.0”, presented at TWEPP2022, Bergen, Norway, Sep. 2022, https://indico.cern.ch/event/1127562/contributions/4904878

[6] I. Johnson, *et al.*, ”Application development on CPSI-S0 Hardware at PSI”, *in Proc. ICALEPCS'23*, Cape Town, South Africa, 2024, paper THPDP071, pp 1508-1511, doi:10.18429/JACoW-ICALEPCS2023-THPDP071

[7] D. Kiselev *et al.*, ”Status and Future Projects of the PSI High Intensity Proton Accelerator”, *Proceedings of the 3rd J-PARC Symposium (J-PARC2019)*, Tsukuba, Japan, Sep. 2019, doi:10.7566/JPSCP.33.011004

[8] B. Keil, R. Ditter, M. Gloor, G. Marinkovic, and J. Purtschert, “First Application of a Multiprocessing System-on-Chip BPM Electronics Platform at SwissFEL”, in *Proc. 11th Int. Beam Instrum. Conf. (IBIC'22)*, Kraków, Poland, Sep. 2022, pp. 245-248, doi:10.18429/JACoW-IBIC2022-TUP12

[9] H. Commin, ”Generic High Performance DSP library for FPGAs”, presented at Embedded Computing Conference 2021, Virtual, Jun. 2021, unpublished, https://www.enclustra.com/assets/files/download/Enclustra_Generic_high-performance_DSP_Library_for_FPGAs.pdf

[10] B. Stef, ”Signal Processing Development Methodologies for FPGA Platforms using Reusable and Generic PSI Library components”, presented at LLRF Workshop'22, Brugg, Switzerland, Oct. 2022, http://indico.psi.ch/event/12911/contributions/38369