RF-MPSoC の加速器制御への適用検討

APPLICATION STUDY OF RF-MPSOC TO ACCELERATOR CONTROL

漁師雅次#, 岩城孝志, 林和孝 , 張替豊旗
Masatsugu Ryoshi #, Takashi Iwaki, Kazutaka Hayashi, Toyoki Harigae
Mitsubishi Electric TOKKI Systems

Abstract

In LLRF (Low Level RF system) and BPM (Beam Position Monitor), the picked-up high frequency signal is frequency-converted and detected and A/D (Analog/Digital) converted and used for digital control and monitoring. It is difficult to correct variations in characteristics because circuits with nonlinear characteristics are used for these. Therefore, using the latest FPGA (Field Programmable Gate Array) RF-MPSoC (Radio Frequency-Multi Processor System on Chip), the A/D conversion of the RF signal is performed directly, or the D/A (Digital/Analog) conversion is performed. We evaluated the performance of a configuration that eliminates nonlinear circuits as much as possible by directly using RF signals. In RF-MPSoC, high-speed digital signal processing can be performed by FPGA, and flexible software processing can be done in 1 chip by built-in APU (Application Processor Unit) and RPU (Real-time Processor Unit). We examined the possibility of making very small LLRF and BPM systems by eliminating the need for DSP and CPU modules.

1. はじめに

LLRF は、空胴内電磁界をピックアップして得た高周波信号をモニタして、フィードバック・フィードフォワード制御することで、空胴加速電圧強度と位相を安定化させている。BPM は、加速ビームバンチによる誘起電磁界（高周波信号）をピックアップ電極で取り出し、演算により得られるビームの位置情報をモニタしている。加速器内のビーム軌道の監視制御には欠かせない。

従来の LLRF や BPM では、検出された高周波信号を、アナログ回路の周波数コンバータを使い、回路を構成しやすい中間周波数信号に変換して処理している。最近では、高精度 ADC のサンプリング高速化（数百 MSPS）が進み、それを使って、ナイキスト周波数を超える周波数の高周波信号をアンダーサンプリングして、FPGA を使った高速デジタル信号処理をしているシステムもある。これにより周波数変換器に使うミキサなど非線形アナログデバイスを使わなくてもいい回路構成をとることができるようになった。

他方では、高周波信号をオーバーサンプリングできるほどの高速サンプリング（数 GSPS）できる ADC が高分解能になってきた。そこで、高周波信号を直接 A/D 変換してフィードバック・フィードフォワード制御などをして、D/A 変換して高周波信号を直接生成する回路構成をとる検討ができるようになってきた。また、2 種類のプロセッサを内蔵した大規模 FPGA に、高速サンプリングができる ADC および DAC を内蔵した RF-MPSoC というデバイスが出てきた。このデバイスを利用することで、従来の LLRF や BPM と比べ非線形デバイスを使うことなく、システムを構成できる可能性がでてきた。

今回は、この RF-MPSoC の評価ボードを使い LLRF や BPM への適用可能性を検討した。

ma-ryoshi@west.melos.co.jp

2. 評価ユニットの構成

2.1 RF-MPSoC の特徴

RF-MPSoC は、Xilinx からリリースされている。GHz でサンプリング可能な ADC および DAC を内蔵しており、ARM Cortex-A53 プロセッサ（以降、APU）と ARM Cortex-R5 プロセッサ（以降、RPU）の二種類のプロセッサを内蔵した FPGA である。[1]

ADC は、最大 4GHz のアナログ入力帯域幅で、12bit の分解能で、4.096GSPS でサンプリングが可能である。この ADC が 8ch 内蔵されている。DAC は、最大 4GHz のアナログ出力帯域幅で、14bit の分解能で、6.554GSPS でサンプリングが可能である。この DAC が 8ch 内蔵されている。アナログ信号側の入出力インタフェイスは差動信号となっている。デジタル信号側のインタフェイスは内部で高速シリアルに変換されて FPGA 部分に接続されている。Cortex-A53[2]は、64bit アドレス命令に対応したアプリケーションプロセッサである。低消費電力でありながら様々な用途に使え、リッチな OS（例えば、Linux）を使うことが可能である。Cortex-R5[3]は 32bit のリアルタイムプロセッサであり、時間制約を守る必要がある処理に適用可能である。2 種類のプロセッサは 128bit の AXI(Advanced eXtensible Interface)で FPGA 部分に内部で接続されており、高速なデータ伝送が可能となっている。FPGA の外部インタフェイスは、一般的な UART・USB2.0・I2C などと、高速伝送可能な Ethernet・USB3.0・PCIe Gen3 x16 などが内蔵されている。外付けのメモリは DDR4 までの SDRAM が接続可能となっている。

2.2 評価ユニットの構成

Xilinx 製の RF-MPSoC の評価ボード ZCU111[4]を使い 19 インチラックマウントサイズユニットに実装した。前面および背面を Fig. 1 および Fig. 2 に示し、内部の前面機能ブロックを Fig. 3 に示した。にアナログ入出力用の SMA コネクタ、背面に電源入力および Ethernet・USB・

Display Port を実装した。ADC および DAC なども評価ボード上のクロックで動作させている。APU 上で Linux（Ubuntu）のデスクトップを起動させており、プログラミング言語 Python を使い ADC,DAC を制御およびデータ取得をおこない、結果をグラフ化した。

Figure 1: Evaluation module.（Front-side）

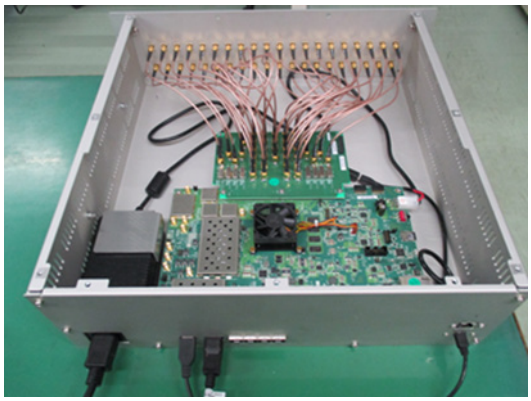

Figure 2: Evaluation module.（Rear-side）

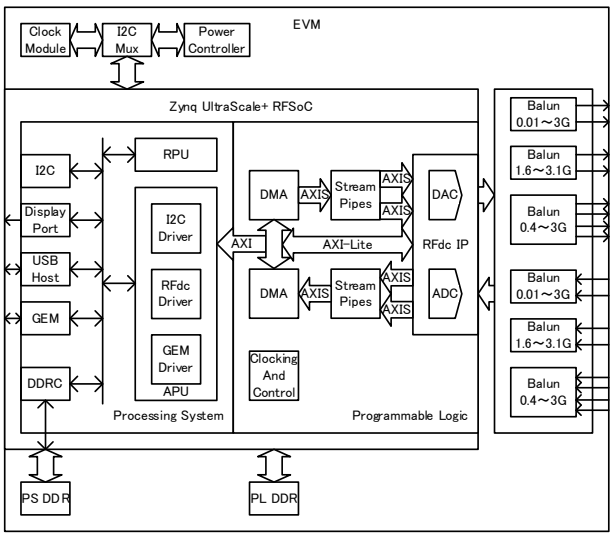

Figure 3: Evaluation module.（Block）

3. LLRF・BPM への適用検討

3.1 LLRF への適用検討

現行のデジタル LLRF では、空胴からのピックアップ高周波信号を 10MHz 程度の中間周波数にミキサなどを用いて周波数変換したあとに A/D 変換素子を使いデジタル信号を生成している。その後、FPGA を利用したフィードバック・フィードフォワード制御などのデジタル処理をして得た I/Q ベースバンド信号を D/A 変換素子を使いアナログ IQ 信号に変換し、アナログ IQ-MOD を用いて高周波信号に変換して出力している。フィードバック制御などのデジタルパラメータ設定およびデジタル信号のモニタは、VME-bus や PCI-express などのカード外部の CPU バスを用いているシステムや、FPGA 内蔵の CPU で EPICS-IOC を動作させて Ethernet 経由で行うシステムがある。

今回は、空胴からの高周波信号をダイレクトにサンプリングして、空胴へダイレクトに高周波信号に戻すことを考えてみる。RF-MPSoC の入出力周波数帯域幅から S バンドまでの高周波加速器には適用できると考えられる。また、デジタル LLRF において空胴の周波数帯域幅相当（1MHz 程度）までのフィードバック帯域の確保が必要といわれている。そのため、デジタル LLRF の処理遅延が小さい要望（1 μ sec 以下）がされることが多い。つまり、ADC から DAC 出力までの間の処理のスループット時間（レイテンシ）の評価が必要である。

また、A/D 変換の振幅精度および位相精度の評価が重要である。一般的にサンプリングシステムの精度を劣化させる要因として、サンプリングジッタがある。入力 RF 周波数が高くなるほど、サンプリングクロックのジッタが信号の純度を下げてしまう。そのため、SNR が劣化する。

3.2 BPM への適用検討

現在 500MHz 未満の高周波信号をアンダーサンプリングして、デジタル信号処理を行っているシステムがある。この場合、SNR を改善して正確な周波数選択をするためには、高性能なアナログ BPF が必要となる。オーバーサンプリングできると、LPF とデジタルフィルタで必要な周波数の選択が可能となる。BPM と補正磁石間で高速なフィードバック制御が必要な場合を除いて、処理時間に余裕があることが多いため、LLRF よりも処理遅延（レイテンシ）に対する要望は少ない。

4. 性能評価

4.1 SNR

評価ボード上のサンプリングクロックを 4.096GHz に設定して、周波数 1GHz の入力のスペクトラムを評価した。内蔵 ADC には、複素 DDC（Digital Down Converter）が付属しており、そこで 8 デシメイションして 512MSPS の IQ 信号に変換した。その信号の FFT 結果を Fig. 4 に示した。SFDR は-72.6dBc で、ノイズフロアは-92.5dBc である。同じ入力条件で、4 デシメイションした場合、ノイズフロアは-88.8dBc（-3.7dB）となり、RBW（Resolution Band Width）が 1/2 となって約-3dB となるため概ね一致する。

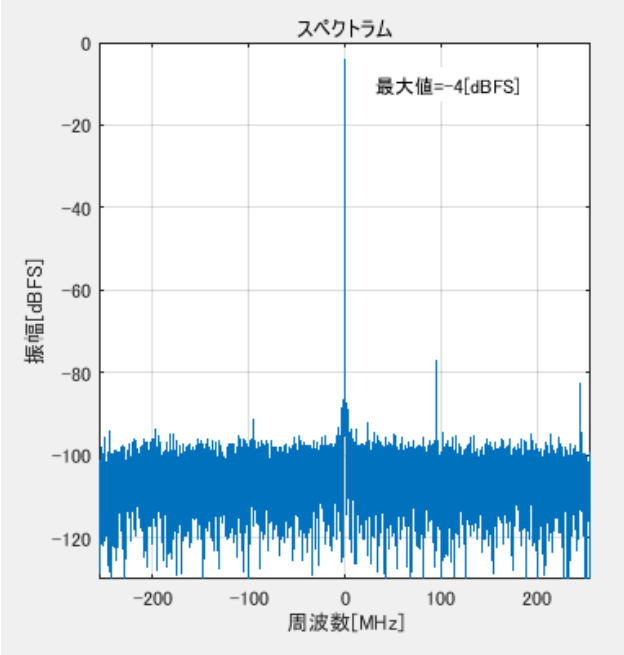

Figure 4: 1.0GHz input 8 decimation after DDC. (BW500MHz)

次に、内蔵 DAC で周波数 1GHz を出力し、内蔵 ADC の入力へ折り返し DDC および 8 デシメイションした結果を Fig. 5 に示す。DAC の出力レベルと ADC の入力レベルを調整しておらず、ピークレベルは-9dB となった。ノイズフロアは、-83.5dBc となり、ピークレベルの減少分を戻すと-92.5dBc となり、Fig. 4 と一致するため DAC 出力のノイズフロアは、ADC 評価時の SG のノイズフロアと同等以下と考えられる。

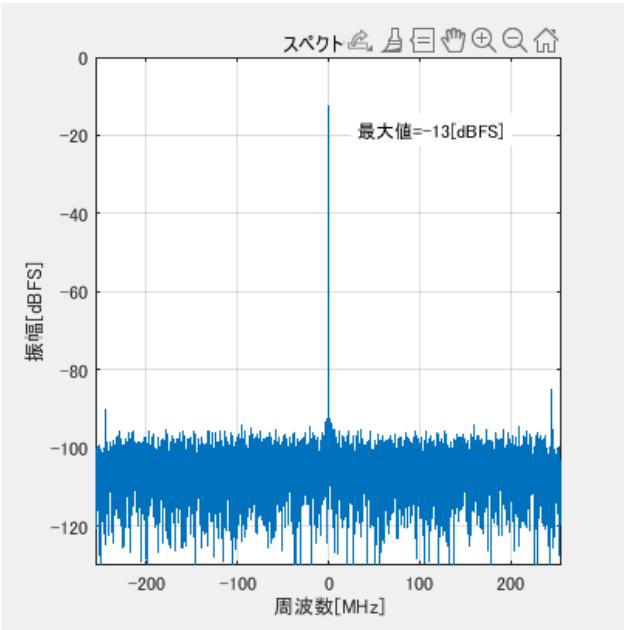

Figure 5: 1.0GHz input from DAC 8 decimation after DDC. (BW500MHz)

4.2 入力間クロストーク

入力 CH4 に外部より周波数 1GHz の正弦波を入力して各 CH の ADC 後のスペクトラムを測定してクロストークを評価した。この結果を Table 1 に示した。CH4 は-4dBFS であり、最もクロストークが大きかったのは、Fig. 6 に示した CH5 の-77dB となった。ADC は 2ch ずつが一つのブロックとなっており CH4 と CH5 は同じブロックである。最小値は CH0 の-88dB となった。2 番目に大きなクロストークを測定したのは、CH7 の-78dB である。FPGA ピンアサイン・PWB パターン・コネクタピンアサインなどが近接していないため、他の要因でカップリングしていると考えられる。原因を特定して独自の基板を設計する場合に活かすようにしたいと考えている。

Table 1: Peak Level of Other CH when Input to CH4

CH	クロストーク[dB]	備考
CH0	-88	最小値
CH1	-85	
CH2	-87	
CH3	-87	
CH4	-	信号入力 CH
CH5	-77	最大値
CH6	-88	最小値
CH7	-78	

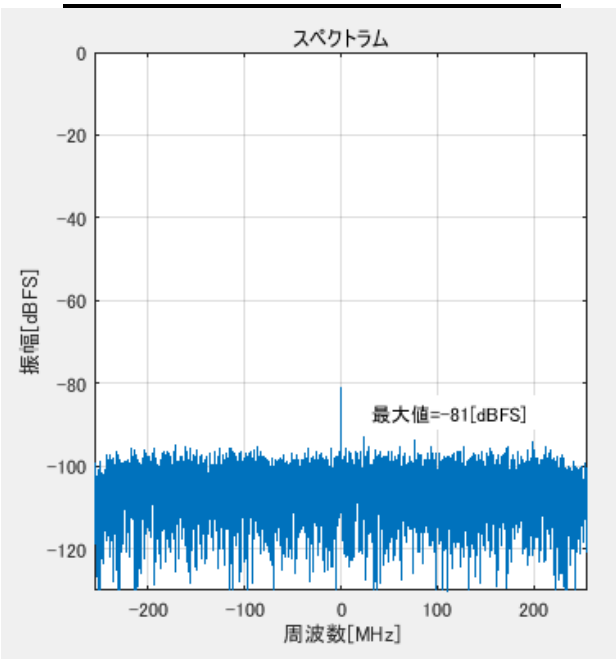

Figure 6: CH5 spectrum when inputting 1.0 GHz to CH4.

4.3 振幅・位相安定度

DAC から 10MHz を出力して、外部入力の信号発生器へ 10MHz リファレンス同期入力を使い周波数同期をとった。DDC 後 8 デシメイションした周波数帯域幅 500MHz の IQ データから、振幅と位相の安定度の計算

を行った。時間変動およびヒストグラムを Fig. 7 および Fig. 8 に示した。振幅は±0.15%rms 程度の位相は±0.15deg の安定度となった。それぞれの安定度を改善するためには、デジタルフィルタ等で SNR を改善する必要があると考えられる。位相の時間波形にうねりがあり、変動が大きく見える。これは、サンプリングクロックを外部から 10MHz を入力して内蔵 PLL を同期させたが、PLL の位相同期の挙動が安定していないと推定している。

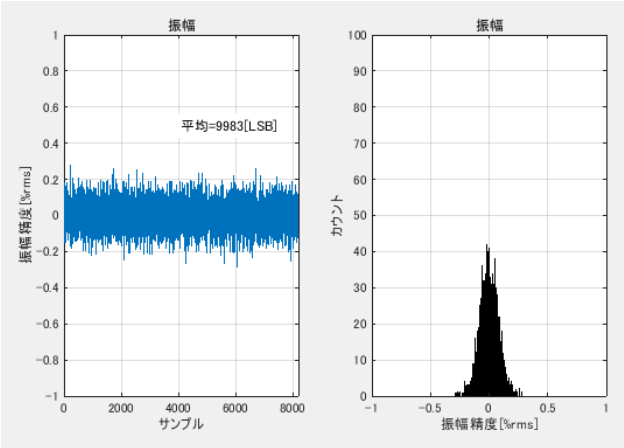

Figure 7: Amplitude stability (BW500MHz) when synchronized with external 10MHz.

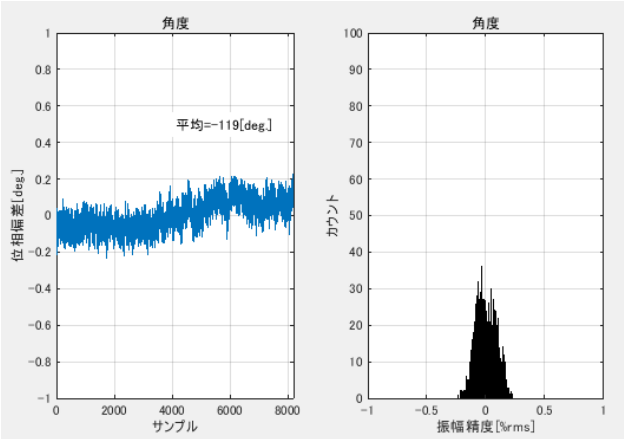

Figure 8: Phase stability (BW500MHz) at external 10MHz synchronization.

4.4 入出力レイテンシ

LLRF のフィードバック制御の性能に影響がある処理遅延は 1 μ sec 以下の要求が多い。Figure 9 に示したように、ADC 入力から DAC 出力へ信号をスルーした際の処理レイテンシを測定した。4.096GSPS でサンプリングした際の結果を Fig. 10 に示す。2 つのサンプリング周波数により測定して、固定遅延時間とサンプリング周波数依存の遅延時間を算出した。サンプリング周波数と遅延時間は下の通りとなった。

- 4.096GSPS : 130nsec
- 2.048GSPS : 200nsec

結果より遅延時間の内訳は、

- クロック依存遅延 : 287 クロック相当
- 固定遅延 : 80nsec

ADC および DAC と PL 間は、高速シリアルインタフェイスで接続されており、シリアルとパラレルの変換回路がある。この回路で（遅延 287 クロック中）多くのクロック数が占めているとみられる。

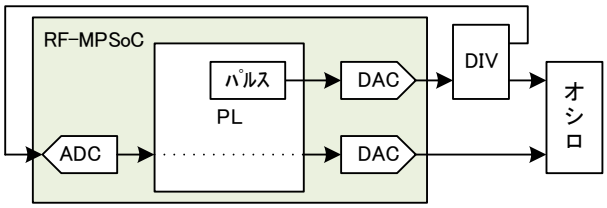

Figure 9: Delay time measurement between input and output.

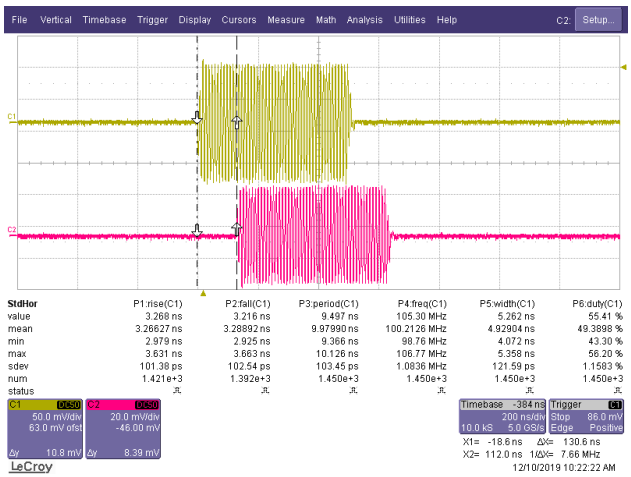

Figure 10: Delay time measurement between input and output @4.096GSPS.

4.5 APU

APU では、Linux（ubuntu）のデスクトップ環境を動かし、USB からキーボード・マウスで操作し、Display port からディスプレイへ表示をさせた。そのうえで python を使って ADC,DAC を制御して、PL からデータを収集した。入力信号や出力信号のモニタをするために、グラフ表示をして性能を評価した。AD 変換後 DDC した結果を 4ch 分収集して、FFT 演算・IQ 極座標・位相・振幅のグラフを動画で表示させた。1 秒以上の繰り返しで表示ができるため、十分に入力信号の状態をモニタすることができる。EPICS IOC をこの APU の上で動作が可能である事も確認した。

4.6 RPU

RPU の処理速度および処理時間のばらつきを測定して信号処理に使えるかどうかの評価を行った。結果を Table 2 および Table 3 にまとめた。今回は信号処理としてよく使う FFT（バタフライ演算）を、演算ポイント数毎の処理時間を測定し平均値を算出した。比較するために、APU で 2048 ポイントの FFT 演算した結果 20.113[msec] だった。RPU は FreeRTOS を実装して、演算した結果 60.960[msec]だった。RPU は APU に対して約 3 倍の処理時間がかかっている。これは、CPU の動作クロック周波数が APU の 1.2GHz に対して RPU が 0.5GHz であるためと考えられる。

次に、約 25[sec]程度の演算時間になる同じ処理を複数回実施し、実際にかかった処理時間の分布を測定した。APU でかかった処理時間が平均 25.190[sec]に対して-0.012[sec]+0.024[sec]の範囲となり、標準偏差が0.011[sec]ある。対して、RPU が平均24.458[sec]に対して、-0.000[sec]+0.001[sec]の範囲となり、標準偏差が0.000[sec]となった。結果、RPU の処理時間は、1[msec]以上の揺らぎがほとんどないと考えられる。ビームパルス毎の LLRF 制御等、実時間処理(タイミング)が重要な制御には、RPU が適している。

Table 2: Comparison of FFT Calculation Processing Time

CPU/OS	FFT (256)	FFT (512)	FFT (1024)	FFT (2048)
Cortex-R5 0.5GHz FreeRTOS	4.64	11.14	26.20	60.96
Cortex-A53 1.2GHz Ubuntsu16.04	1.702	3.907	8.897	20.113

[msec]

Table 3: Comparison of Fluctuations in Processing Time

CPU/OS	平均値	最小値	最大値	標準偏差
Cortex-R5 0.5GHz FreeRTOS	24.458	24.458	24.459	0.000
Cortex-A53 1.2GHz Ubuntsu16.04	25.190	25.178	25.214	0.011

[sec]

4.7 OpenAMP

従来は、2 種類のプロセッサ(APU と RPU)を使い連携して演算をする場合、別々のメモリ領域を使って演算するためメモリ間のデータ伝送が必要となり、処理時間のオーバーヘッドやばらつきの原因の一つとなっていた。そこで、Fig. 11 に示した複数種のプロセッサ間のデータ共有手法の「OpenAMP(Open Asymmetric Multi Processing)」を使った処理の評価をした。これにより、同一物理メモリ上に共有メモリ空間を確保するため、データ伝送が不要となり処理時間の短縮が可能となった。

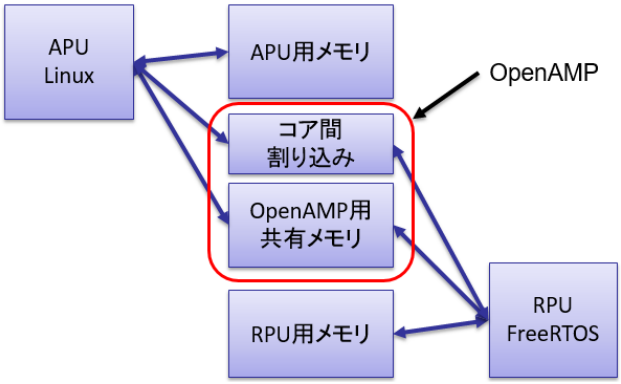

Figure 11: Configuration when using OpenAMP.

5. まとめ

RF-MPSoC が加速器制御に適応できるか検討した。今回は、入力信号の中心周波数が 1GHz で周波数帯域 500MHz の広帯域データを使って性能を評価した。ADC の振幅・位相精度は物足りないように見られたが、内部ノイズ密度は十分に低いため、デジタルフィルタで狭帯域化すれば、LLRF および BPM へ適用できると考える。ダイレクトサンプリングおよび内蔵 APU・RPU によるソフトウェア処理を組み合わせることで、従来ではできなかった高度な信号処理ができる見通しも得られたことから、小型・高性能な LLRF や BPM システム試作に向けて具体的な検討を進める。

参考文献

[1] Xilinx ZYNQ UltraScale+ RFSoC; https://japan.xilinx.com/products/silicon-devices/soc/rfsoc.html
[2] ARM CPU Cortex-A53; https://www.arm.com/ja/products/silicon-ip-cpu/cortex-a/cortex-a53
[3] ARM CPU Cortex-R5; https://www.arm.com/ja/products/silicon-ip-cpu/cortex-r/cortex-r5
[4] Xilinx Zynq UltraScale+ RFSoC ZCU111 EVM; https://japan.xilinx.com/products/boards-and-kits/zcu111.html