Research in Astronomy and Astrophysics, 24:015025 (11pp), 2024 January https://doi.org/10.1088/1674-4527/ad0e99


# PSRDP: A Parallel Processing Method for Pulsar Baseband Data

Ya-Zhou Zhang[1,2], Hai-Long Zhang[1,2,3,4], Jie Wang[1,4], Xin-Chen Ye[1,2,4], Shuang-Qiang Wang[1], Xu Du[1,2], Han Wu[1,2], Ting Zhang[1,2], Shao-Cong Guo[5], and Meng Zhang[6]
[1] Xinjiang Astronomical Observatory, Chinese Academy of Sciences, Urumqi 830011, China; zhanghailong@xao.ac.cn
[2] University of Chinese Academy of Sciences, Beijing 100049, China
[3] Key Laboratory of Radio Astronomy, Chinese Academy of Sciences, Nanjing 210008, China
[4] National Astronomical Data Center, Beijing 100101, China
[5] Southeast University, Nanjing 211189, China
[6] School of Cyber Science and Engineering, Qufu Normal University, Qufu 273165, China
*Received 2023 October 20; revised 2023 November 13; accepted 2023 November 20; published 2024 January 16*

## Abstract

To address the problem of real-time processing of ultra-wide bandwidth pulsar baseband data, we designed and implemented a pulsar baseband data processing algorithm (PSRDP) based on GPU parallel computing technology. PSRDP can perform operations such as baseband data unpacking, channel separation, coherent dedispersion, Stokes detection, phase and folding period prediction, and folding integration in GPU clusters. We tested the algorithm using the J0437-4715 pulsar baseband data generated by the CASPSR and Medusa backends of the Parkes, and the J0332+5434 pulsar baseband data generated by the self-developed backend of the NanShan Radio Telescope. We obtained the pulse profiles of each baseband data. Through experimental analysis, we have found that the pulse profiles generated by the PSRDP algorithm in this paper are essentially consistent with the processing results of Digital Signal Processing Software for Pulsar Astronomy (DSPSR), which verified the effectiveness of the PSRDP algorithm. Furthermore, using the same baseband data, we compared the processing speed of PSRDP with DSPSR, and the results showed that PSRDP was not slower than DSPSR in terms of speed. The theoretical and technical experience gained from the PSRDP algorithm research in this article lays a technical foundation for the real-time processing of QTT (Qi Tai radio Telescope) ultra-wide bandwidth pulsar baseband data.

*Key words:* (stars:) pulsars: general – methods: data analysis – techniques: miscellaneous

## 1. Introduction

With the rapid development of manufacturing technology and information technology, radio telescope receiving system has developed toward ultra-wide bandwidth and PAF (phased array feed). Multi-beam and ultra-wide bandwidth receiving systems make the amount of astronomical data obtained by observation increase sharply (Zhang & Duan 2022). For example, low-frequency array needs to process about 14 Tb raw astronomical signals per second at a sampling rate of 200 MHz (van Haarlem et al. 2013). The multi-beam backend of FAST (Five-hundred-meter Aperture Spherical Telescope; Nan et al. 2011) has a data sampling rate of 12.8 GB $s^{-1}$ under the condition of 100 $\mu$s time resolution, 4000 channels and dual polarization (Li et al. 2018). The QTT (Qi Tai radio Telescope; Wang et al. 2023), which is being built in Xinjiang, China, its ultra-wide bandwidth and PAF (van Cappellen et al. 2022) receiving system will generate an annual archive data volume that is expected to exceed 10 PB. The real-time processing of ultra-wide bandwidth pulsar data is one of the urgent problems to be solved and also the starting point of this study.

With the continuous advancement of astronomical research and the rapid development of digital technology, the performance requirements of digital backend systems and related equipment for astronomical observation are constantly increasing. Backend systems need to achieve high-speed sampling, real-time analysis, and data preprocessing at wider bandwidths, higher time resolution, and higher frequency resolution. Handling the high-speed data streams generated by PAF and multi-beam receiving systems, as well as the real-time processing of massive astronomical data on heterogeneous platforms, are currently urgent challenges in the operation of radio observation facilities. Due to limitations in storage device performance, massive astronomical signals can only be processed and analyzed in real-time, posing significant challenges to the performance of data processing equipment (Wei 2019).

Current mainstream digital backend systems often employ a hybrid architecture consisting of FPGA, CPU, and GPU. Heterogeneous systems require data to be exchanged between different devices. One of the technical challenges in data processing systems based on heterogeneous platforms is how to

achieve high-speed data flow between the CPU and GPU while avoiding issues such as high error rates and packet loss during the transmission of massive data. For the high-speed transmission and distribution of data, one feasible approach is to create a ringbuffer in memory to perform real-time high-speed caching of data and transfer the buffer data to the GPU memory, ultimately enabling real-time data processing on the GPU.

The use of GPU can significantly enhance the running speed of applications in scientific analysis, simulations, and other fields. Internationally, real-time processing of pulsar signals using CPU+GPU heterogeneous computing platforms has become mainstream. Utilizing the computational units of GPUs to process observational data can reduce the CPU workload, greatly enhancing the data flow processing efficiency of the system. As radio astronomy digital backend technology continues to advance, real-time signal processing presents a serious challenge to traditional computing techniques. The powerful computational capabilities of GPU clusters provide a feasible solution for handling the massive data streams generated during radio astronomy observations.

The Australian Parkes ultra-wide bandwidth receiver system uses an FPGA+GPU architecture for preprocessing and recording observational data. It divides the 704–4032 MHz signal into three analog RF bands for sampling. In the FPGA preprocessing stage, it further divides the data into 26 continuous subbands, each with a bandwidth of 128 MHz, using PFB (polyphase filter bank) technology (Zhang et al. 2023b). The FPGA packages the data into VDIF format[7] and transfers it to GPU nodes using the UDP protocol. To handle high-speed network flows, each GPU node uses PSRDADA[8] to establish high-speed ringbuffers to temporarily store the data. Subsequently, the data is copied from the buffer to the GPU memory for further processing (Hobbs et al. 2020). The GBT (Green Bank Telescope) in the United States uses the ROACH2[9]+GPU architecture for processing pulsar signals in its VEGAS (Versatile GBT Astronomical Spectrometer) backend. VEGAS's pipeline software consists of multiple concurrent threads. The network threads transmit data from ROACH2 to a shared buffer in the computing node via a 10 GbE network link. Then, the reading threads copy the data to the GPU memory. After data processing, the results are sent to the disk-writing threads for storage on the disk (Chennamangalam et al. 2014).

The FAST backend employs an architecture combining ROACH2 and GPUs along with high-speed networking to fulfill observational requirements such as pulsar searches and timing (Nan & Li 2014; You et al. 2021). The Digital Backend System of the Shanghai Tian Ma Radio Telescope uses a hybrid FPGA+GPU architecture. Data output by the FPGA is transmitted through a 10 GbE network link to eight high-performance computers equipped with GPUs for coherent dedispersion and incoherent dedispersion (Yan et al. 2017). The Yunnan Astronomical Observatory utilizes a pulsar observation system with a bandwidth of 512 MHz, 8 bit sampling precision, and dual-polarization input, using ROACH2 as the baseband data acquisition backend and Digital Signal Processing Software for Pulsar Astronomy (DSPSR)[10] as the data processing core (Xu et al. 2015). This system offers 128 channels (each with a 4 MHz bandwidth) for processing, including data decoding, coherent dedispersion, polarization computation, and folding. The results of data processing are stored in the PSRFITS format (Hotan et al. 2004). The pulsar observation system for the HRT (Haoping Radio Telescope) employs the ROACH2+GPU architecture. Data output by ROACH2 is transmitted via a 10 GbE link to eight GPU nodes, supporting both incoherent dedispersion and coherent dedispersion observational modes (Luo et al. 2020).

## 2. Parallel Processing Algorithm for Pulsar Baseband Data Based on GPU

To meet the real-time or near-real-time processing requirements of ultra-wide bandwidth pulsar baseband data, we have designed and implemented the PSRDP (PulSaR baseband Data Processing) algorithm using GPU parallel computing technology. With the powerful computational capabilities of GPUs, the algorithm's efficiency has been greatly improved. GPUs have more cores than CPUs and can simultaneously execute more threads, making them suitable for handling massive computational tasks involving ultra-wide bandwidth pulsar baseband data.

The main process of the PSRDP algorithm is illustrated in Figure 1. It starts with reading the pulsar baseband data on the HOST, copying the data to the DEVICE, and then performing channelization, coherent dedispersion, stokes detection, folding integration, and other operations. Once data processing is completed, the results are copied back to the HOST for storage.

### 2.1. Unpack Baseband Data

After copying the baseband data to the GPU memory, the baseband data needs to be unpacked first. For the baseband data recorded by the CASPSR backend, dual polarization 8 bit sampling, two polarizations every four data (real numbers) are stored alternately. As shown in Figure 2, the dual polarization data are unpacked, the data are recombined, and data with the same polarization are put together to facilitate subsequent processing. In order to realize GPU parallel unpacking, Equation (1) is used to establish a relationship between the data index after unpacking and the data index before

[7] https://vlbi.org/vlbi-standards/vdif
[8] https://psrdada.sourceforge.net
[9] https://casper.astro.berkeley.edu/wiki/ROACH2
[10] https://dspsr.sourceforge.net

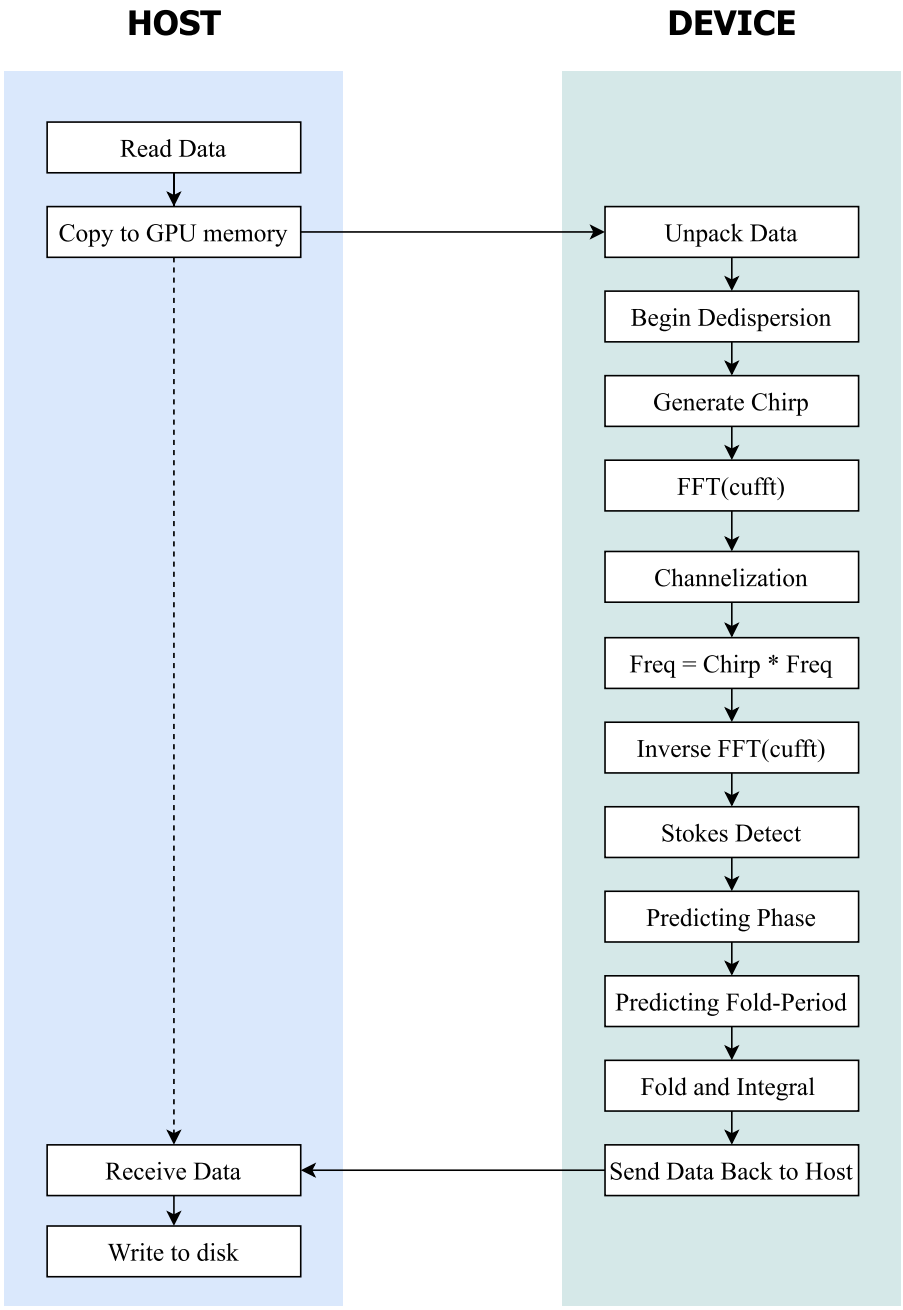


**Figure 1.** PSRDP data processing flow.

unpacking.

$$j = \left(\frac{i}{npad}\%2\right) \times polsize + i\%npad + \left(\frac{i}{2 \times npad}\right) * npad, \quad (1)$$

where $i$ is the data index before unpacking, $j$ is the data index after unpacking, $npad = 4$ (because dual polarizations are stored alternately every four data), if the data block length is $N$, then $polsize = N/2$. For the dual polarization baseband data recorded by the Medusa backend, dual polarization 32 bit sampling, the dual polarizations are stored alternately every 2048 data (1024 complex numbers are also equal to 2048 real numbers), as shown in Figure 3. It is similar to the CASPSR unpacking process, $npad = 2048$, and the data needs to be Perform offset binary decoding. Each datum is independent, GPU parallel technology can be fully utilized to accelerate unpacking.

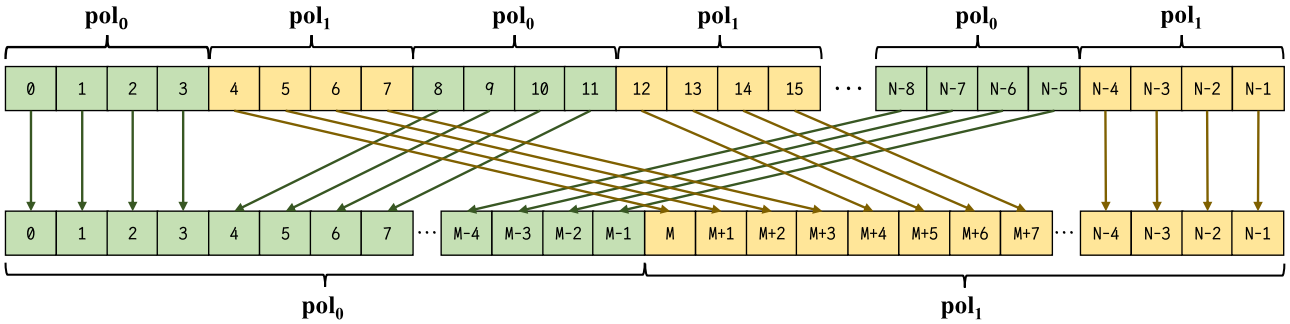


**Figure 2.** CASPSR backend baseband data unpacking (The data block length is $N$, $M = N/2$).

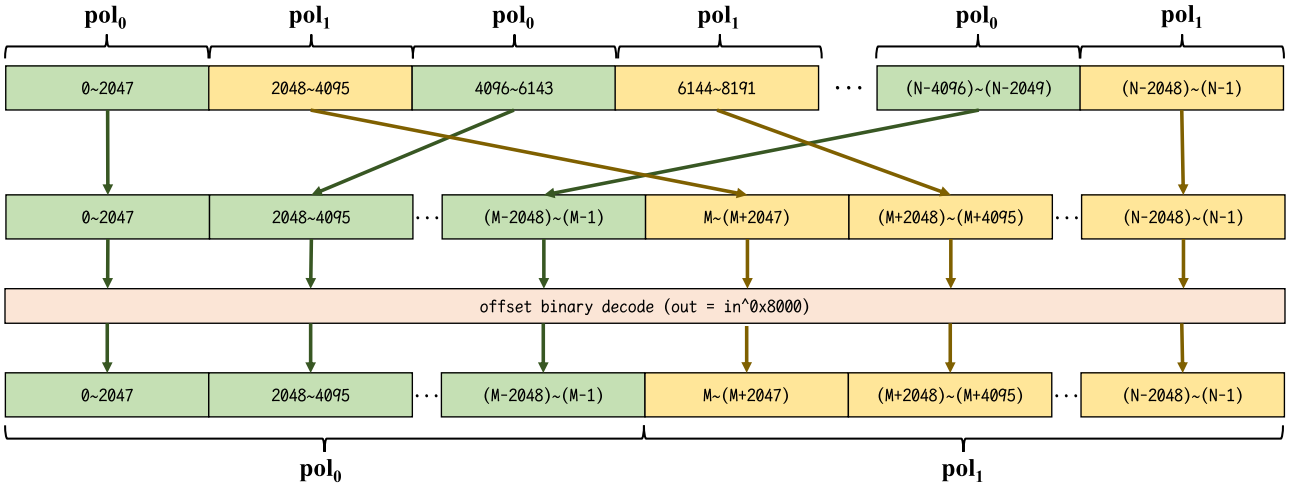


**Figure 3.** Medusa backend baseband data unpacking (The data block length is $N$, $M = N/2$).

### 2.2. Multi-channel Coherent Dedispersion

During the transmission of pulsar signals to Earth, they undergo dispersion due to the influence of the ISM (interstellar medium), causing high-frequency signals to arrive at Earth before low-frequency signals. The effect of the ISM on radio waves is akin to the action of a filter with an ISM transfer function. To restore the original characteristics of pulsar signals, it is necessary to perform dedispersion, where coherent dedispersion involves filtering the observed signal through the inverse of the transfer function of the ISM. In theory, coherent dedispersion can completely eliminate the dispersion effects caused by the ISM in the signal (Zhang et al. 2023a).

As shown in Figure 4, after performing FFT on the time data sequence, it is further channelized into multiple subbands. Based on the central frequency, bandwidth, and dispersion measure (DM) of each subband, chirp coefficients (the inverse function of the ISM transfer function) are generated (Lorimer & Kramer 2012), as shown in Equation (2).

$$H(f_{\mathrm{ref}} + \Delta f)^{-1} = \exp\left[-i\frac{2\pi D}{2(f_{\mathrm{ref}} + \Delta f)f_{\mathrm{ref}}^2}DM(\Delta f)^2\right], \tag{2}$$

In the equation, the reference frequency $f_{\mathrm{ref}}$ is set as the central frequency, $\Delta f$ represents the difference between a specific frequency point $f$ and $f_{\mathrm{ref}}$, and $D$ is the dispersion constant, which is equal to $4.15 \times 10^3\,\mathrm{MHz^2\,pc^{(-1)}\,cm^2 \cdot s}$. Subsequently, the chirp coefficients generated for each subband are multiplied point-to-point with the data from each subband to achieve coherent dedispersion. Since there is no interdependence between the data, this process can be efficiently parallelized on a GPU.

### 2.3. Stokes Detection

After coherent dedispersion, for dual-polarization data obtained from two orthogonal polarization channels, $pol_0$ and $pol_1$.

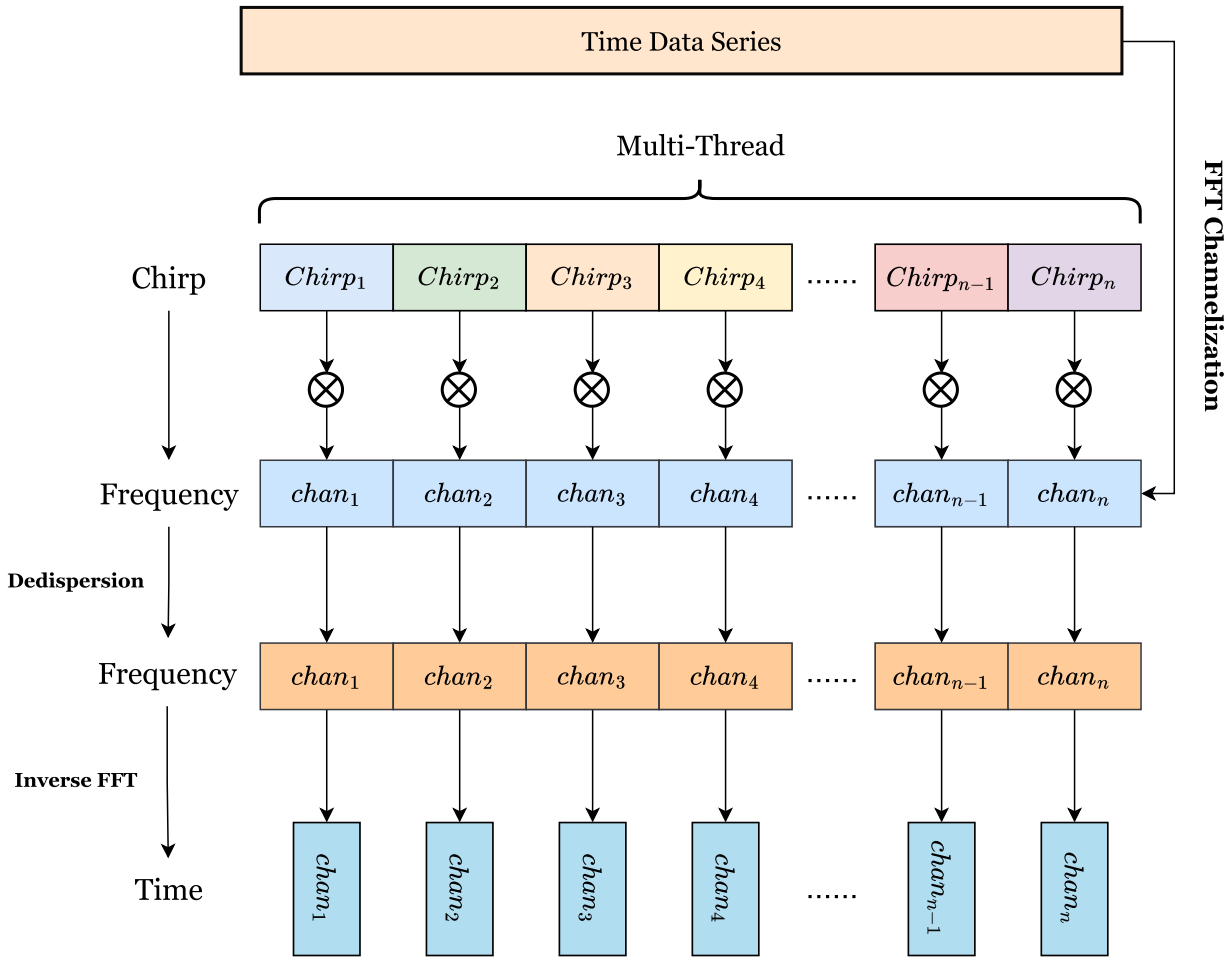


**Figure 4.** Multi-channel coherent dedispersion process.

Four Stokes parameters are $I$, $Q$, $U$ and $V$ (Sutinjo et al. 2022). For dual-linear feeds, the Stokes parameters can be calculated using Equation (3). For dual-circular feeds, Equation (4) can be used to calculate.

$$\begin{bmatrix} I \\ Q \\ U \\ V \end{bmatrix} = \begin{bmatrix} P_{AA} + P_{BB} \\ P_{AA} - P_{BB} \\ 2\,Re(P_{AB}) \\ 2\,Im(P_{AB}) \end{bmatrix}. \tag{3}$$

$$\begin{bmatrix} I \\ Q \\ U \\ V \end{bmatrix} = \begin{bmatrix} P_{AA} + P_{BB} \\ 2\,Im(P_{AB}) \\ 2\,Re(X * Y) \\ P_{BB} - P_{AA} \end{bmatrix}. \tag{4}$$

where $P_{AA}$ and $P_{BB}$ are the powers of two orthogonal polarization channels $pol_0$ and $pol_1$ respectively, $P_{AB}$ is the cross-correlation product of dual polarizations, $Re(P_{AB})$ and $Im(P_{AB})$ represent the real part and imaginary part of $P_{AB}$ respectively.

Stokes parameters are completely independent. As shown in Figure 5, each datum uses a GPU thread to implement calculations. GPU parallel technology can greatly reduce the resource loss caused by traditional CPU single thread and improve computing efficiency.

### 2.4. Folding and Integrating

Different pulsars have inconsistent periods, and coupled with different sampling rates, the number of sampling points in each period of the pulsar often cannot be represented by an integral power of 2 (in order to optimize the FFT execution, the number of input points is best represented by an integral power of 2) , which will cause the last remaining data to be discarded or processed specifically during the integral calculation process. The strategy adopted by this algorithm is to combine folding and integration operations and process them at the same time.

In the CUDA programming model, calculations are sorted according to a three-level hierarchy. Each call to the CUDA kernel creates a new Grid, which is composed of multiple

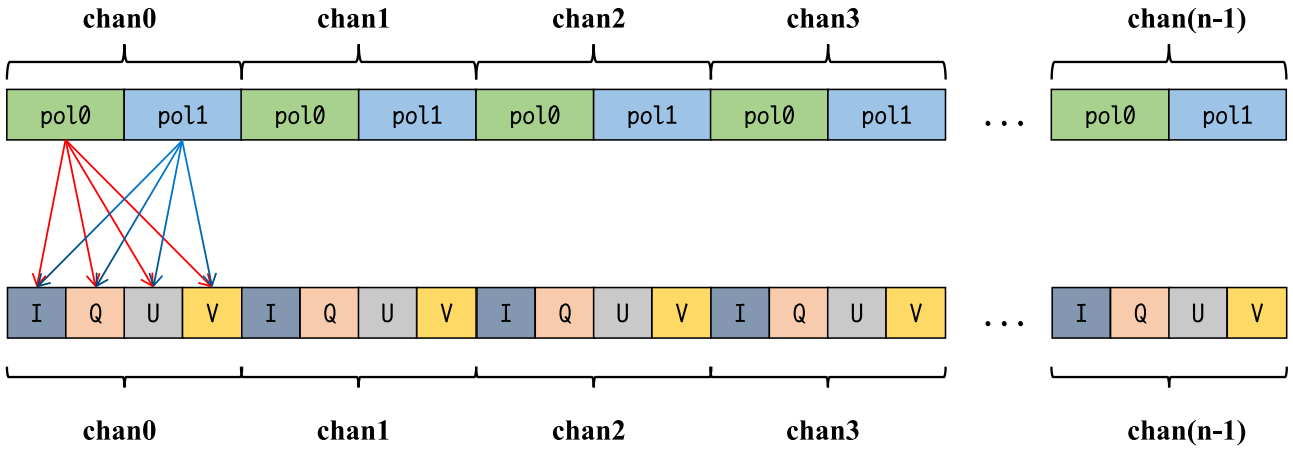


**Figure 5.** Parallel detection of Stokes parameters.

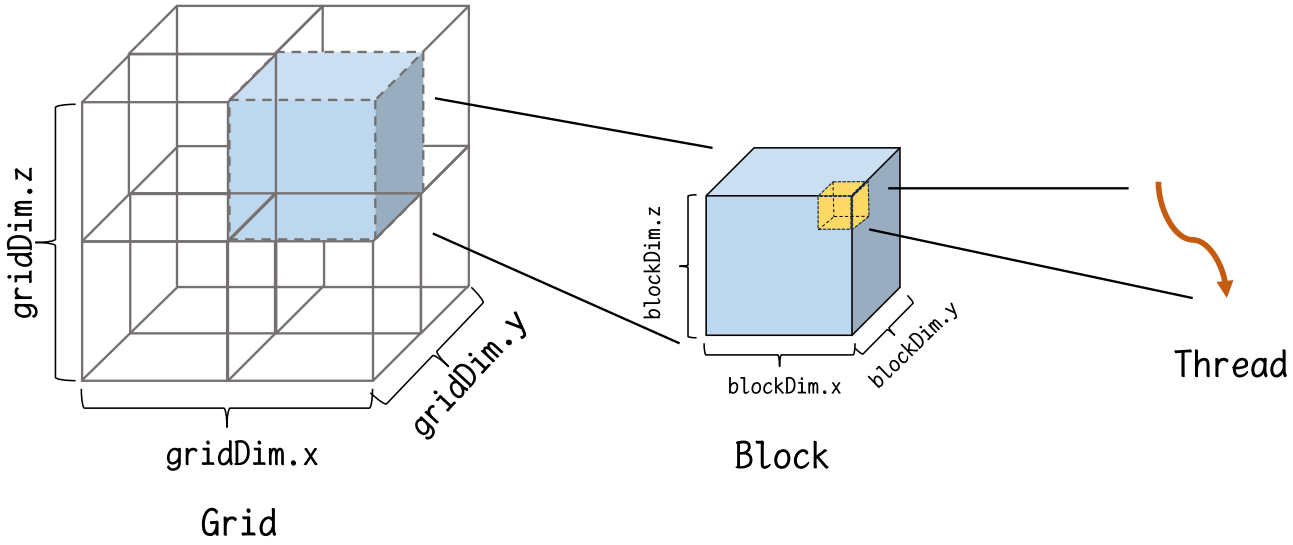


**Figure 6.** CUDA three-level thread model.

Blocks. Each Block is composed of up to 1024 separate Threads (Hijma et al. 2023). As shown in Figure 6, Grid can control the number of Blocks by setting three-dimensions: *gridDim.x*, *gridDim.y* and *gridDim.z*. Similarly, Block can also control the number of Threads through *blockDim.x*, *blockDim.y* and *blockDim.z*.

By default, the number of profile data after final folding integration is 1024. Therefore, for $n$ channels and four Stokes parameters, a space of size $4 * 1024 * n$ needs to be pre-allocated to store the final generated profile data. For multi-channel multi-polarization data, since the starting recording time is the same, the corresponding profile sequence index for each group is also the same. Only one set of index sequence needs to be generated, and its index sequence generation method is shown in Figure 7.

The period of pulsars is very stable, but the locations of observatories, pulse dispersion, orbit parameters of the solar system (Romer delay, Shapiro delay and Einstein delay) and binary systems can affect the times-of-arrival of pulses, which in turn affects the folding period of pulsar data (Pennucci 2015). In order to accurately fold the pulse profile and the phase of a certain timestamp, it is necessary to predict the folding period and the phase of the pulsar at that timestamp (Zhang et al. 2023a).

When processing the pulsar baseband data, it is necessary to divide the baseband data into blocks for further processing. In order to accurately fold the pulse profile, it is necessary to predict the folding period of the pulse and the relative phase value (between 0 and 1) of the first sample point in each block according to the starting time of each block. After determining

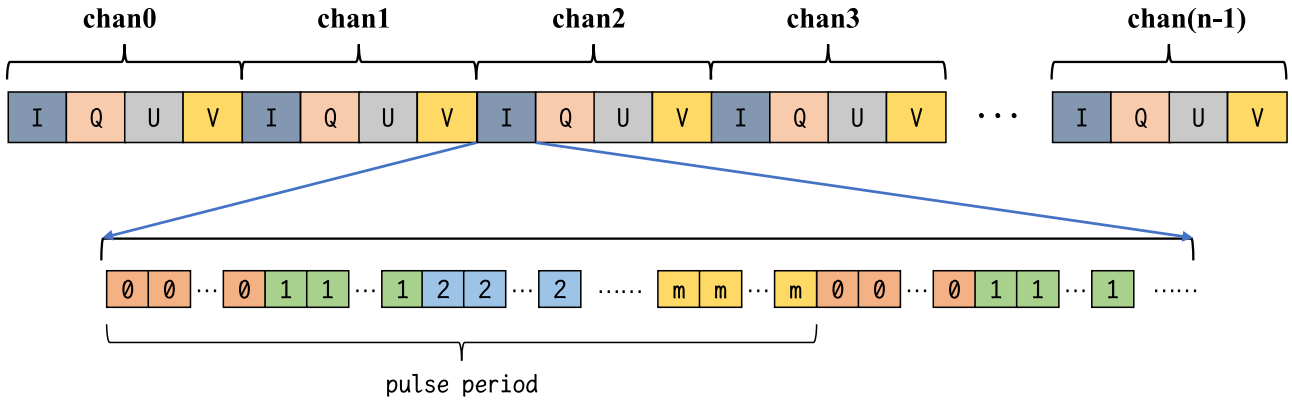


**Figure 7.** Index sequence generation procedure.

the phase value of the first sample point, the phase values of all sample points in this block after folding can be calculated. The phase value of the $i$th sample point is shown in Equation (5).

$$phase_i = (phase_1 + i * phase\_per\_sample) - floor(phase_1 + i * phase\_per\_sample). \quad (5)$$

Among them, $phase_1$ is the phase value of the first sampling point, which can be obtained through prediction. The *floor*() function represents rounding down, and *phase_per_sample* represents the phase size interval of each sampling point, which can be expressed as Equation (6).

$$ibin = phase_i \times nbin. \quad (6)$$

The same indexes in the index sequence may not be clustered together (a piece of data may be more than one period). For the folding operation, the corresponding points of each period are added and averaged, while the integral is the average of multiple consecutive points. The pulse profile index generation diagram is shown in Figure 7. We combine the folding and integration operations, that is, add the data corresponding to the same index and average it.

During the implementation of the algorithm, direct addition of GPU multi-threaded calculation results will lead to error results. In this case, atomic addition operations are often used. Because atomic addition uses a locking mechanism at the bottom layer for data protection, it seriously slows down the calculation speed. In order to deal with the above problems, we designed and implemented a multi-threaded folding integration algorithm based on GPU parallel technology based on the folding integration method proposed in Section 7.1 of the handbook of pulsar astronomy (Lorimer & Kramer 2012). The core idea is that for multi-channel multi-polarized data, assuming that for N-channel M-polarized data, the final contour data size generated by each polarization is *nbin*. Use GPU multi-threading technology to open $N * M * nbin$ threads, each *nbin* threads process one polarization channel, where each polarization performs a serial addition of data with the same index within a channel.

The core idea is that for multi-channel multi-polarized data, assuming that for N-channel M-polarized data, the final contour data size generated by each polarization is nbin. Use GPU multi-threading technology to open $N * M * nbin$ threads.

In order to implement the above algorithm, set the dimensions of Grid: $gridDim.x = 1$, $gridDim.y = nchan$(number of channels) and $gridDim.z = npol$ (number of polarizations), and similarly set the dimensions of Block: $blockDim.x = 1024$ (the number of pulse profiles included), $blockDim.y = 1$ and $blockDim.z = 1$, there are a total of $npol * nchan * 1024$ threads. As shown in Figure 8, the data with index 1 in a certain polarization of a certain channel uses a thread to perform serial loop addition, avoiding the resource overhead problem caused by atomic addition.

## 3. Experimental Analysis and Results

### *3.1. Parkes Baseband Data Processing*

#### *3.1.1. CASPSR Baseband Data Processing*

The PSRDP was tested using the baseband data generated by the CASPSR backend observation of the Parkes. The baseband data observation source is J0437-4715, the bandwidth size is 400 MHz, the bandwidth range is 1182–1582 MHz, and the observation time is 8s and the data size is 12.8 GB. Use PSRDP to divide it into 1024 channels, and perform Stokes detection, coherent dedispersion, integration and folding on each channel. Extract the $I$ of the stokes parameters of each channel to obtain Figure 9. It can be clearly seen that there is dispersion delay between each channel. After performing incoherent dedispersion, as shown in Figure 10, the profiles of each subband can be correctly aligned. As shown in Figure 11, the phases and amplitudes of the two profiles are basically consistent, which

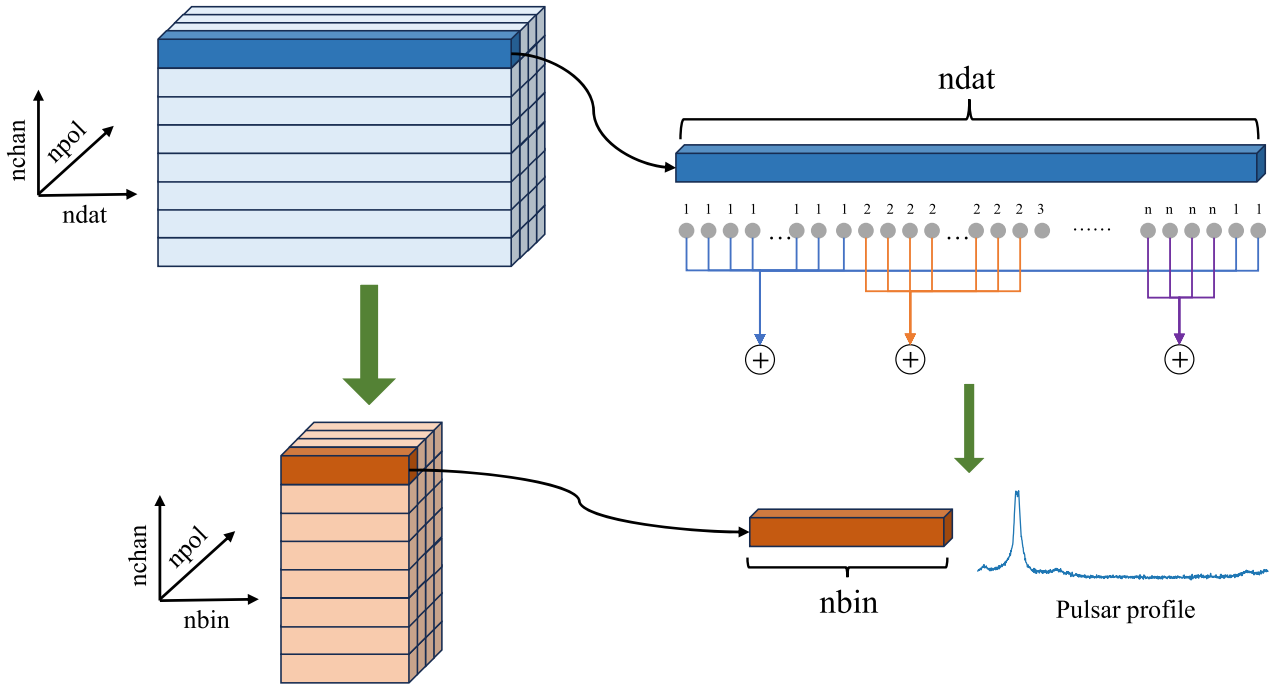


**Figure 8.** Multi-channel multi-polarization folding and integrating.

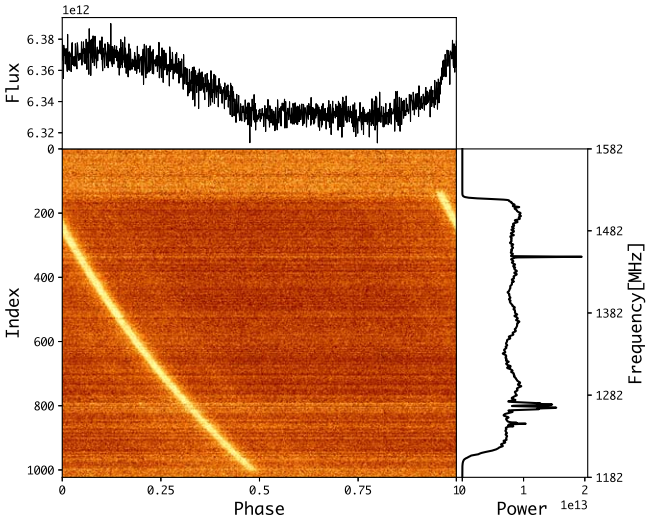


**Figure 9.** Dynamic spectrum of J0437-4715 (before dedispersion).

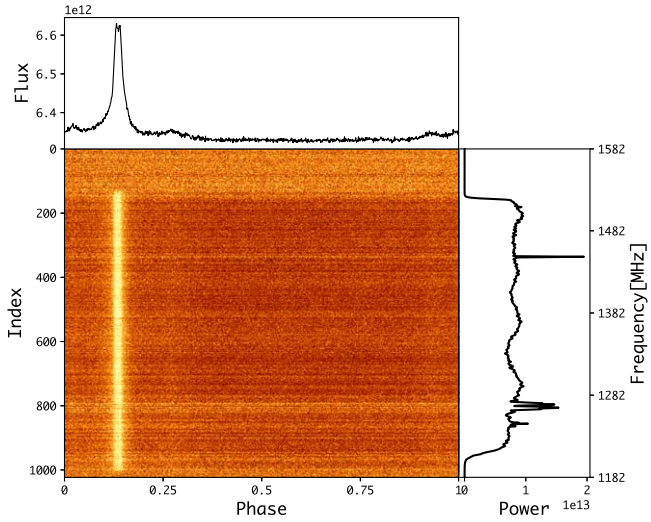


**Figure 10.** Dynamic spectrum of J0437-4715.

verifies the effectiveness of the PSRDP period and phase prediction methods.

We use DSPSR to process the baseband data and also output 1024 channels. The final profile generated is compared with the profile generated by PSRDP. As shown in Figure 10, the phases and amplitudes of the two profiles are basically consistent, which verifies the effectiveness of the PSRDP period and phase prediction methods.

### *3.1.2. Medusa Baseband Data Processing*

The PSRDP was tested using the baseband data generated by the Medusa observation of the Parkes. The baseband data observation source is J0437-4715, the bandwidth size is 128 MHz, the bandwidth range is 704–832 MHz, the observation time is about 9.6 s, and the data size is 9.6 GB. The baseband data is divided into 128 MHz subbands. The processing steps are similar to those in Section 3.1.1. After

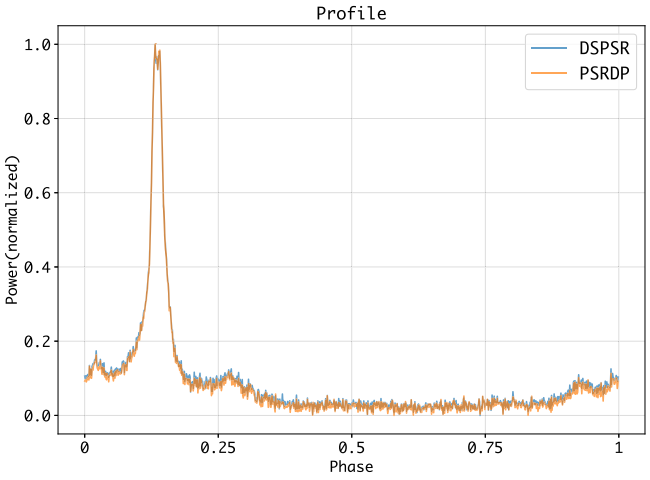


**Figure 11.** Comparison of profiles processed by PSRDP and DSPSR (CASPSR baseband data).

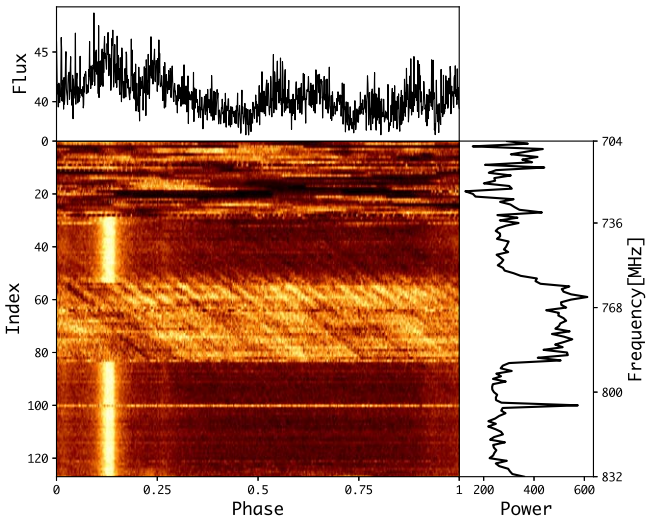


**Figure 12.** Dynamic spectrum of J0437-4715 (with RFI channel).

performing incoherent dedispersion, Figure 12 is obtained. It can be clearly seen that there is RFI (Radio Frequency Interference) in some channels. After removing the interfered channels, as shown in Figure 13, a clear pulse profile can be seen.

We use DSPSR to process the baseband data and output 128 channels. After removing the interference at the same position, the final profile generated is compared with the contour generated by PSRDP. As shown in Figure 14, the phase and amplitude of the two profiles are basically the same.

### *3.2. NSRT Baseband Data Processing*

We are based on the self-developed pulsar backend (the backend uses three self-developed FPGA development boards

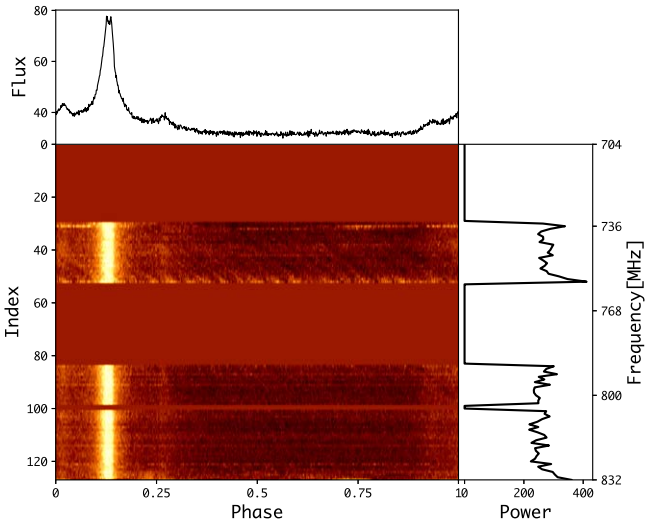


**Figure 13.** Dynamic spectrum of J0437-4715 (without RFI channel).

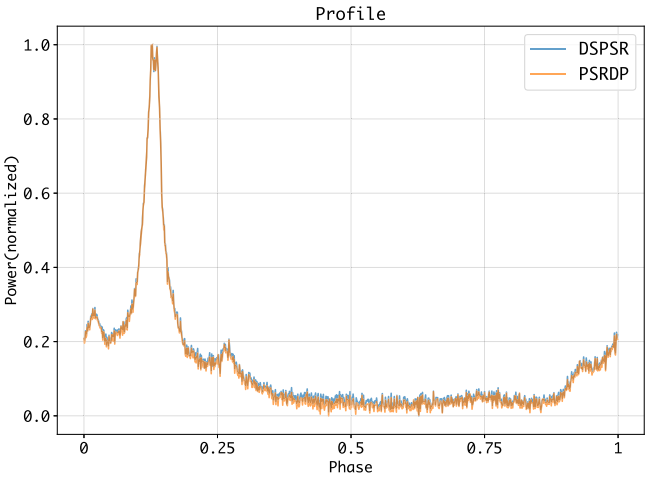


**Figure 14.** Comparison of profiles processed by PSRDP and DSPSR (Medusa baseband data).

and uses PFB technology to divide the 964–1732 MHz bandwidth signal into six 128 MHz bandwidth subbands, and each subband is transmitted to the GPU server for processing.), using the NSRT (NanShan Radio Telescope) *L*-band 1 GHz bandwidth receiver observation to generate baseband data. The observation source is J0332+5434, and the output single-channel 128 MHz bandwidth baseband data is dual-polarized with 16 bit accuracy, the spectrum range is 1220–1348 MHz, and the observation duration is 5 minutes. Using the same data processing method as in Section 3.1.1/3.1.2, Figure 15 is obtained. After deleting some of the interfered channels, the dynamic spectrum obtained is shown in Figure 16.

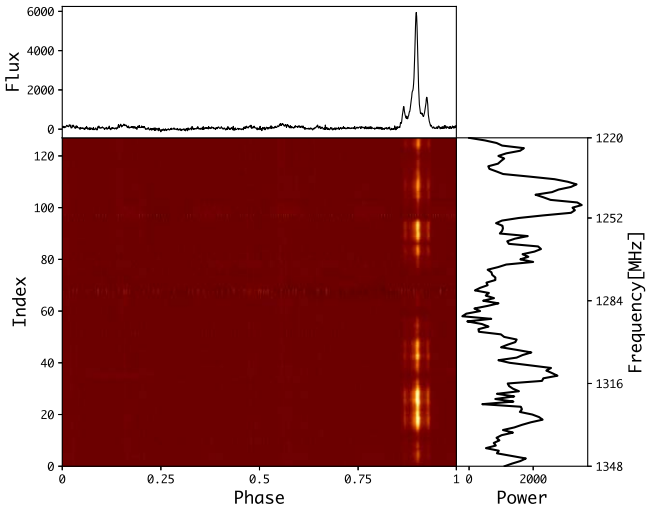


**Figure 15.** Dynamic spectrum of J0332+5434 (with RFI channel).

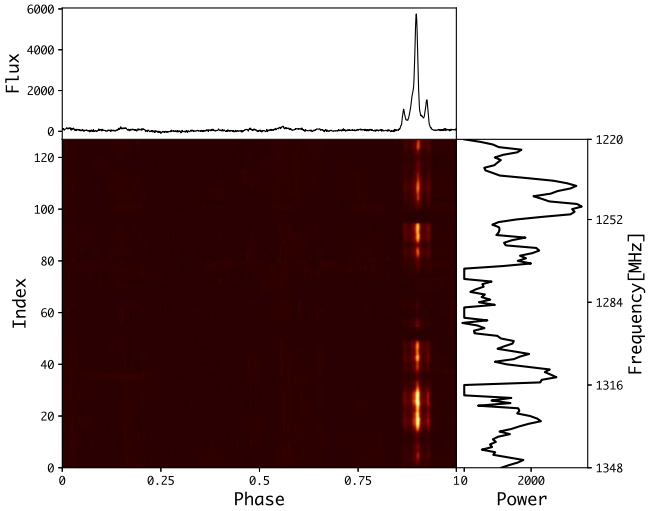


**Figure 16.** Dynamic spectrum of J0332+5434 (without RFI channel).

### *3.3. PSRDP and DSPSR Speed Comparison*

Regarding the execution time of the algorithm, we used the same data to conduct speed tests on PSRDP and DSPSR. The number of channels was set to 2, 4, 8, 16, 32, 64, 128, 256, 512 and 1024 respectively. The test file was CASPSR baseband data, after multiple rounds of testing, calculate the average processing time. The test results are shown in Figure 17. The execution time of PSRDP is not higher than DSPSR.

## 4. Conclusion

In response to the QTT ultra-wide bandwidth pulsar data processing requirements, we designed and implemented PSRDP based on GPU parallel technology. After the PSRDP reads the baseband data at the HOST, it transmits to the DEVICE. The wideband signal is further subdivided using FFT, and each subband after subdivision is subjected to coherent dedispersion, stokes detection, periodic phase prediction and folding integration operations. Finally, it is transmitted back to the HOST for scientific data storage.

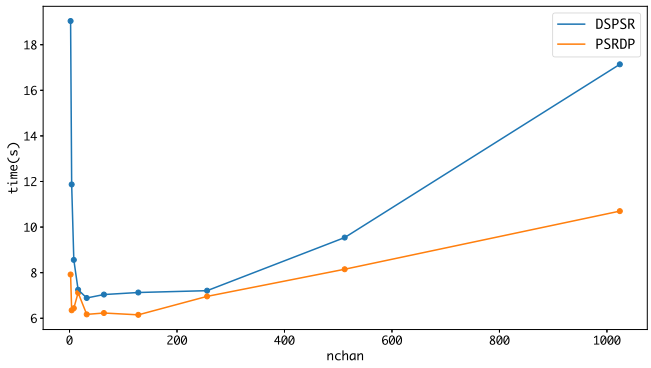


**Figure 17.** PSRDP and DSPSR speed comparison.

We systematically tested the PSRDP using the J0437-4715 pulsar baseband data generated by the CASPSR and Medusa backends and the J0332+5434 pulsar baseband data generated by the self-developed backend of the NSRT. The PSRDP is used to output multiple subbands, and profiles between multiple subbands are aligned through incoherent dedispersion, and the final profile is generated. Compared with DSPSR in phase and amplitude, the pulse profile generated by PSRDP has no obvious difference and is slightly better than DSPSR in terms of algorithm processing speed. By comparing the PSRDP experimental results with the DSPSR processing results, the effectiveness of the PSRDP algorithm is verified.

## Acknowledgments

This work is supported by the National Key R&D Program of China Nos. 2021YFC2203502 and 2022YFF0711502; the National Natural Science Foundation of China (NSFC) (12173077 and 12003062); the Tianshan Innovation Team Plan of Xinjiang Uygur Autonomous Region (2022D14020); the Tianshan Talent Project of Xinjiang Uygur Autonomous Region (2022TSYCCX0095); the Scientific Instrument Developing Project of the Chinese Academy of Sciences, grant No. PTYQ2022YZZD01; China National Astronomical Data Center (NADC); the Operation, Maintenance and Upgrading Fund for Astronomical Telescopes and Facility Instruments, budgeted from the Ministry of Finance of China (MOF) and administrated by the Chinese Academy of Sciences (CAS); Natural Science Foundation of Xinjiang Uygur Autonomous Region (2022D01A360).

## ORCID iDs

Ya-zhou Zhang https://orcid.org/0000-0001-6046-2950
Jie Wang https://orcid.org/0000-0003-0380-6395
Xu Du https://orcid.org/0000-0001-6448-0822

## References

Chennamangalam, J., Scott, S., Jones, G., et al. 2014, PASA, 31, e048
Hijma, P., Heldens, S., Sclocco, A., van Werkhoven, B., & Bal, H. E. 2023, ACM Comput. Surv., 55, 239
Hobbs, G., Manchester, R. N., Dunning, A., et al. 2020, PASA, 37, e012
Hotan, A. W., van Straten, W., & Manchester, R. N. 2004, PASA, 21, 302
Li, D., Wang, P., Qian, L., et al. 2018, IMMag, 19, 112
Lorimer, D. R., & Kramer, M. 2012, Handbook of Pulsar Astronomy (Cambridge: Cambridge Univ. Press)
Luo, J.-T., Gao, Y.-P., Yang, T.-G., et al. 2020, RAA, 20, 111
Nan, R., Li, D., Jin, C., et al. 2011, IJMPD, 20, 989
Nan, R., & Li, H. 2014, SSPMA, 44, 1063
Pennucci, T. T. 2015, PhD thesis, Univ. Virginia
Sutinjo, A. T., Ung, D. C. X., & Sokolowski, M. 2022, A&A, 664, A102
van Cappellen, W. A., Oosterloo, T. A., Verheijen, M. A. W., et al. 2022, A&A, 658, A146
van Haarlem, M. P., Wise, M. W., Gunst, A. W., et al. 2013, A&A, 556, A2
Wang, N., Xu, Q., Ma, J., et al. 2023, SCPMA, 66, 289512
Wei, D. 2019, PhD thesis, Univ. Chinese Academy of Sciences
Xu, Y., Li, J., Zhang, Y., et al. 2015, AR&T, 12, 480
Yan, Z., Shen, Z.-Q., Wu, Y.-J., Zhao, R.-B., & Liu, Q.-H. 2017, in 2017 XXXIInd General Assembly and Scientific Symp. of the Int. Union of Radio Science (URSI GASS), 1 (Piscataway, NJ: IEEE)
You, S.-P., Wang, P., Yu, X.-H., et al. 2021, RAA, 21, 314
Zhang, H.-L., Zhang, Y.-Z., Zhang, M., et al. 2023a, RAA, 23, 015023
Zhang, M., Zhang, H.-L., Zhang, Y.-Z., et al. 2023b, RAA, 23, 085012
Zhang, X., & Duan, R. 2022, Proc. SPIE, 12190, 1219032