

УДК 004.62

НЕКОТОРЫЕ ВОПРОСЫ ОТЛАДКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ СБОРА И ПРЕДВАРИТЕЛЬНОЙ ОБРАБОТКИ ДАННЫХ НЕЙТРОННОГО ДЕТЕКТОРА НА ОСНОВЕ ДВУХ КОЛЬЦЕВЫХ БУФЕРОВ

© 2023 г. М. А. Голубев^a, *, А. О. Поллошкин^a, В. А. Соловей^a

^aНИИЦ “Курчатовский институт” – ПИЯФ, Гатчина, 188300 Россия

*e-mail: golubev_ma@pnpi.nrcki.ru

Поступила в редакцию 22.08.2022 г.

После доработки 28.10.2022 г.

Принята к публикации 28.10.2022 г.

Рассмотрены вопросы отладки программного обеспечения устройства сбора и предварительной обработки данных с сетевым интерфейсом для двумерного позиционно-чувствительного детектора тепловых нейтронов на основе линий задержки. Оригинальное программное обеспечение использует способ передачи данных через два кольцевых буфера. Предложены изменения в коде программного обеспечения для проверки возможности переполнения кольцевых буферов. Показано, что при частоте входных импульсов до 1 МГц потери данных в кольцевых буферах нет. Соответствующее мертвое время регистрации составляет порядка 1 мкс.

Ключевые слова: детектор нейтронов, линия задержки, программируемая логическая интегральная схема, кольцевой буфер, веб-интерфейс, локальная сеть, обработка данных, программное обеспечение, электроника, передача данных.

DOI: 10.31857/S1028096023040076, EDN: KBUZBR

ВВЕДЕНИЕ

Ранее [1] было опубликовано описание оригинального устройства для сбора и предварительной обработки данных с сетевым интерфейсом, которое впоследствии должно стать частью двумерных детекторов нейтронов на основе линий задержки. Линии задержки позволяют уменьшить число каналов обработки сигналов двумерного детектора с сотен до пяти, разнося импульсы с соответствующих участков детектора по времени [2], а далее задачу восстановления картинки решает такое устройство. В [1] речь идет о многопроводных пропорциональных камерах, однако устройство можно использовать и для иных видов двумерных детекторов с линиями задержки, предварительно преобразовав сигналы к логическому виду соответствующим образом. Новшеством среди подобных устройств стало создание пользовательского веб-интерфейса для визуализации и получения данных на основе сетевого протокола HTTP и передача данных по локальной сети через разъем Ethernet. Аналогичные устройства для передачи данных на пользовательский компьютер использовали шину VME [3], PCI [4–7] или разъем USB [8, 9].

Способ передачи данных через два кольцевых буфера является наиболее простым решением для использованного в устройстве одноплатного

компьютера с программируемой логической интегральной схемой (ПЛИС). Благодаря уже существующей и интегрированной в среду разработки ПЛИС шине Avalon между ПЛИС и процессором одноплатного компьютера разбить кольцевую память на два буфера – в ПЛИС и в оперативной памяти – оказалось легче, чем делить доступ напрямую к оперативной памяти из ПЛИС с операционной системой одноплатного компьютера.

Кольцевой буфер представляет из себя массив переменных (или соответствующую кольцевую структуру данных), для которого запись и чтение происходят по кругу: по завершении – сначала. Кольцевые буферы используют при регистрации и обработке сигналов в режиме реального времени. В цифровой обработке сигналов их применяют для создания фильтров с конечной импульсной характеристикой [10, 11]. Процесс-производитель, осуществляющий запись в кольцевой буфер, и процесс-потребитель, осуществляющий последующее чтение из кольцевого буфера, могут работать либо в режиме перезаписи, когда непрочитанные данные будут заново перезаписаны в случае переполнения, либо процесс-производитель должен блокироваться по заполнении кольцевого буфера, чтобы не произошло потери уже полученных данных [12]. В случае кольцевой структуры данных с указателями страниц памяти

возможно заменять уже заполненные страницы памяти кольцевого буфера пустыми с помощью соответствующих манипуляций с указателями [12]. Однако такая подмена в работе не применима из-за разной арифметики указателей адресного пространства шины Avalon и постраничной передачи данных по сети, что, возможно, будет исправлено в будущих совместимых версиях прошивки и программы.

Целью работы была проверка надежности передачи данных и работоспособности в целом при максимальных нагрузках. Программное обеспечение включает в себя прошивку, многопоточную программу и веб-клиент для работы по локальной сети в отличие от подобных решений [8], содержащих только прошивку и клиент.

ОСОБЕННОСТИ УСТРОЙСТВА

Устройство представляет собой плату собственного производства с преобразователем время—код (time-digital converter) фирмы Asam, соединенную с одноплатным компьютером с ПЛИС от Intel Altera. Эти компоненты на момент создания устройства имели наименьшую стоимость и подходящее качество. Для сигналов двумерного детектора нейтронов — сигнала “старт”, соответствующего моменту регистрации частицы, и четырех сигналов “стоп” от двух линий задержки для координат X и Y — есть пять логических входов. Через конфигурационные регистры разработанной прошивки ПЛИС задается окно времени после сигнала “старт”, соответствующее максимальной из двух линий задержки, в течение которого успевают прийти все сигналы “стоп”. Окно времени, сложенное с некоторой константой, определяемой временем сброса микросхемы-преобразователя время—код, равно мертвому времени регистрации частицы. Так максимальная

длина линии задержки для мертвого времени 1 мкс равна 60 нс, однако в разрабатываемых сейчас детекторах на основе многопроводных пропорциональных камер линии задержки будут по 500 нс, что соответствует мертвому времени 1.4 мкс.

По шине данных микросхемы-преобразователя время—код прошивка ПЛИС собирает 16-битные значения задержек, т.е. по 64 бита на событие, и записывает их в кольцевой буфер внутри ПЛИС на 282 события. Размер буфера ограничен размерами ПЛИС. Этот кольцевой буфер и текущее положение указателя доступно в адресном пространстве памяти операционной системы GNU/Linux одноплатного компьютера по шине данных Avalon. Программа, написанная на языке программирования C, использует отдельный поток исполнения, чтобы извлекать данные из этого кольцевого буфера и записывать их в кольцевой буфер в оперативной памяти размером 64 МБ (8388606 событий). В основном потоке исполнения программы работает простой HTTP-сервер, который управляет программой и прошивкой и передает данные из оперативной памяти клиенту, обрабатывая его запросы.

ИСПЫТАНИЯ

К входу “старт” устройства был подключен генератор с частотой следования импульсов 1 МГц. Была задана длина линий задержки 300 нс. В инструментах разработчика браузера можно видеть, что веб-клиент, запрашивая каждую секунду данные, получает примерно по 4 Мб данных (рис. 1). Это соответствует частоте регистрируемых импульсов 500 кГц. Каждый второй приходящий импульс не регистрируется, потому что приходится на мертвое время, которое больше 1 мкс. Если время линии задержки сократить до 60 нс, то получим, как и предполагалось, примерно по

Name	Status	Type	Initiator	Size	Time
☐ t	200	xhr	(index):5139	158 B	172 ms
☐ d	200	xhr	(index):4983	4.0 MB	348 ms
☐ d	200	xhr	(index):4983	4.0 MB	345 ms
☐ d	200	xhr	(index):4983	4.0 MB	345 ms
☐ d	200	xhr	(index):4983	4.0 MB	343 ms
☐ d	200	xhr	(index):4983	4.0 MB	345 ms
☐ d	200	xhr	(index):4983	4.0 MB	346 ms
☐ d	200	xhr	(index):4983	4.0 MB	346 ms
☐ d	200	xhr	(index):4983	4.0 MB	347 ms
☐ d	200	xhr	(index):4983	4.0 MB	364 ms
☐ d	200	xhr	(index):4983	4.0 MB	349 ms
☐ p	200	xhr	(index):5044	158 B	289 ms
☐ d	200	xhr	(index):4983	3.0 MB	275 ms

Рис. 1. Изображение инструмента разработчика веб-браузера, демонстрирующего скорость передачи данных программой клиенту раз в секунду при частоте регистрации событий 500 кГц: t — запуск измерений; d — запрос данных; p — остановка измерения.

☐ x?m = 855789605	200	xhr	(index):5199	158 B	8 ms
☐ t	200	xhr	(index):5139	158 B	174 ms
☐ d	200	xhr	(index):4983	8.1 MB	691 ms
☐ d	200	xhr	(index):4983	8.0 MB	682 ms
☐ d	200	xhr	(index):4983	8.0 MB	696 ms
☐ d	200	xhr	(index):4983	8.0 MB	687 ms
☐ d	200	xhr	(index):4983	8.0 MB	684 ms
☐ d	200	xhr	(index):4983	8.0 MB	695 ms
☐ d	200	xhr	(index):4983	8.0 MB	716 ms
☐ d	200	xhr	(index):4983	8.0 MB	699 ms
☐ d	200	xhr	(index):4983	8.0 MB	703 ms
☐ d	200	xhr	(index):4983	8.0 MB	699 ms
☐ d	200	xhr	(index):4983	8.0 MB	715 ms
☐ d	200	xhr	(index):4983	8.0 MB	699 ms
☐ p	200	xhr	(index):5044	158 B	858 ms
☐ d	200	xhr	(index):4983	5.3 MB	475 ms

Рис. 2. Изображение инструмента разработчика веб-браузера, демонстрирующего скорость передачи данных программой клиенту раз в секунду при частоте регистрации событий 1 МГц: t — запуск измерений; d — запрос данных; p — остановка измерения.

8 Мб данных в секунду (рис. 2). Очевидно, что ежесекундные запросы в худшем случае в восемь раз меньше размера кольцевого буфера в оперативной памяти, а время такого запроса не превышает секунды, поэтому результаты испытаний кольцевого буфера в оперативной памяти не представляют интереса, и их приводить не будем.

Интерес представляет кольцевой буфер в ПЛИС, поскольку чтение данных из него создает нагрузку, с которой, предположительно, процессор встраиваемого компьютера может не справиться, что приведет к запаздыванию и потере данных. Указатель последнего записанного события, достигая конца буфера, возвращается в начало. Происходит перезапись и потеря предыдущих значений, если они не были прочитаны программой. Программа прочитывает данные из этого буфера по следующему алгоритму: 1) текущее значение указателя прочитывается и записывается в перемен-

ную “указатель”; 2) если “указатель” меньше “старого указателя” (по умолчанию указывающего на начало буфера), то значения между “старым указателем” и концом буфера прочитываются и записываются в кольцевой буфер в оперативной памяти, а “старому указателю” присваивается адрес начала памяти; 3) если “указатель” больше “старого указателя” (на предыдущем шаге “старый указатель” мог быть изменен), то значения между “старым указателем” и “указателем” прочитываются и записываются в кольцевой буфер в оперативной памяти, а “старому указателю” присваивается значение “указателя”; 4) ожидаем 10 мкс, чтобы разгрузить процессор, и повторяем сначала. Для того чтобы проверить отсутствие потери данных, на втором шаге перед изменением “старого указателя” перезапишем в кольцевой буфер в оперативной памяти вместо каждого записанного 64-битного события 32-битные значения “указателя” и “старого указателя”. Одного этого

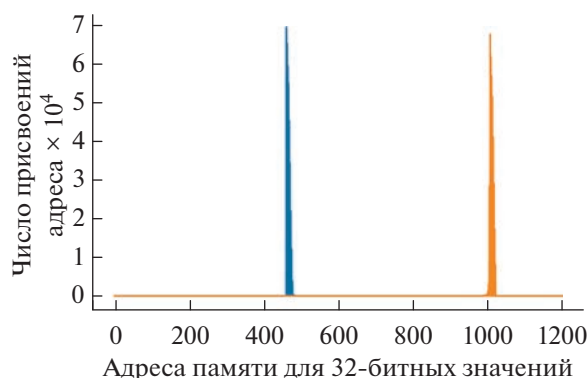


Рис. 3. Статистика адресов, присвоенных двум указателям потока, копирующего данные при переходе с конца в начало буфера ПЛИС. Частота регистрации событий 500 кГц.

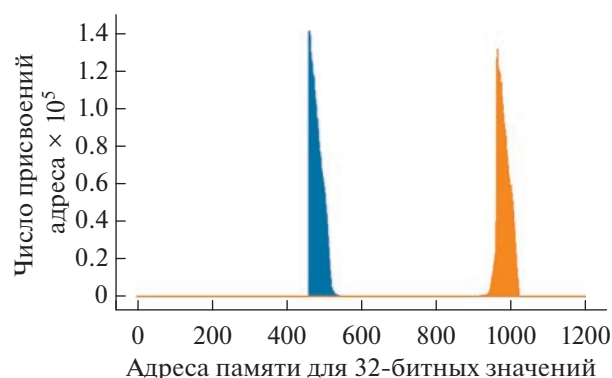


Рис. 4. Статистика адресов, присвоенных двум указателям потока, копирующего данные при переходе с конца в начало буфера ПЛИС. Частота регистрации событий 1 МГц.

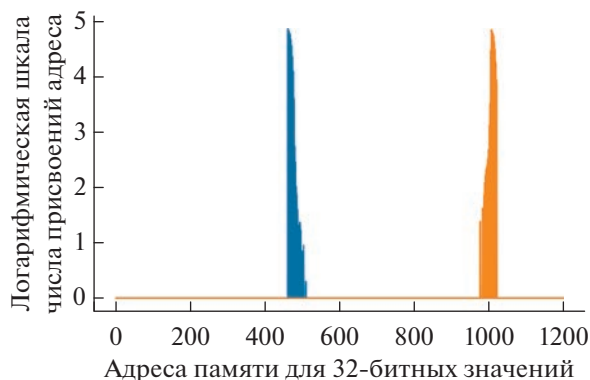


Рис. 5. Статистика адресов, присвоенных двум указателям потока, копирующего данные при переходе с конца в начало буфера ПЛИС (логарифмическая шкала). Частота регистрации событий 500 кГц.

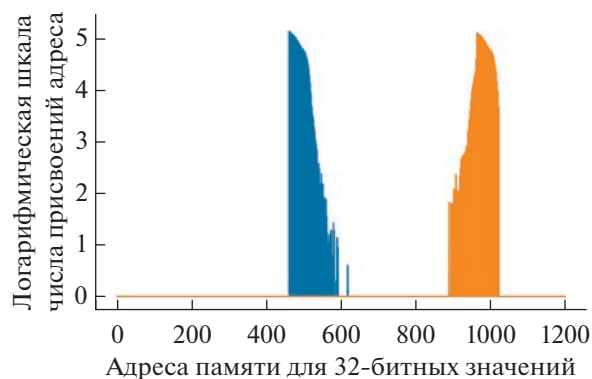


Рис. 6. Статистика адресов, присвоенных двум указателям потока, копирующего данные при переходе с конца в начало буфера ПЛИС (логарифмическая шкала). Частота регистрации событий 1 МГц.

изменения в две строчки кода достаточно для проверки, поскольку весь код по отображению и скачиванию данных уже реализован. В результате в веб-клиенте отобразится статистика присвоения значений указателей на адреса кольцевой памяти ПЛИС, которые принимают значения от 462 до 1024 (мы не указываем базовый адрес, который, если вдаваться в подробности, в программе может быть непостоянным, и под указателями подразуме-

ваем смещение относительно базового адреса). Эти данные отобразятся на первом и третьем графиках в веб-клиенте, поскольку старшие 16 бит из 32 не заняты. Их можно скачать в виде CSV-файла и анализировать отдельно (рис. 3, 4). Воспользовавшись логарифмической шкалой как при анализе данных, так и в веб-клиенте, можно увидеть, что даже редкие присвоения не удаляются от краев буфера к середине на этом шаге (рис. 5–7), что было бы необ-

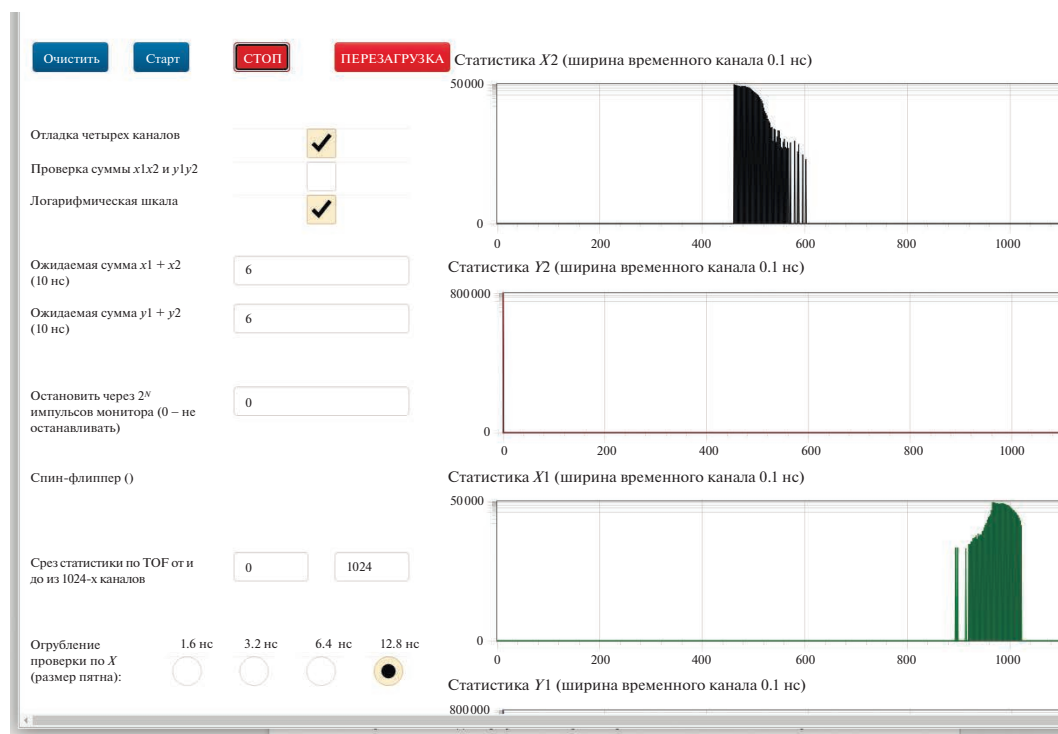


Рис. 7. Изображение пользовательского веб-интерфейса, демонстрирующего статистику адресов, присвоенных двум указателям потока, копирующего данные при переходе с конца в начало буфера ПЛИС с логарифмической шкалой с неизменными графиками задержек сигналов (вместо названия оси абсцисс следует подразумевать адрес памяти для 32-битного целого типа). Отсутствие переполнения. Частота регистрации событий 1 МГц.

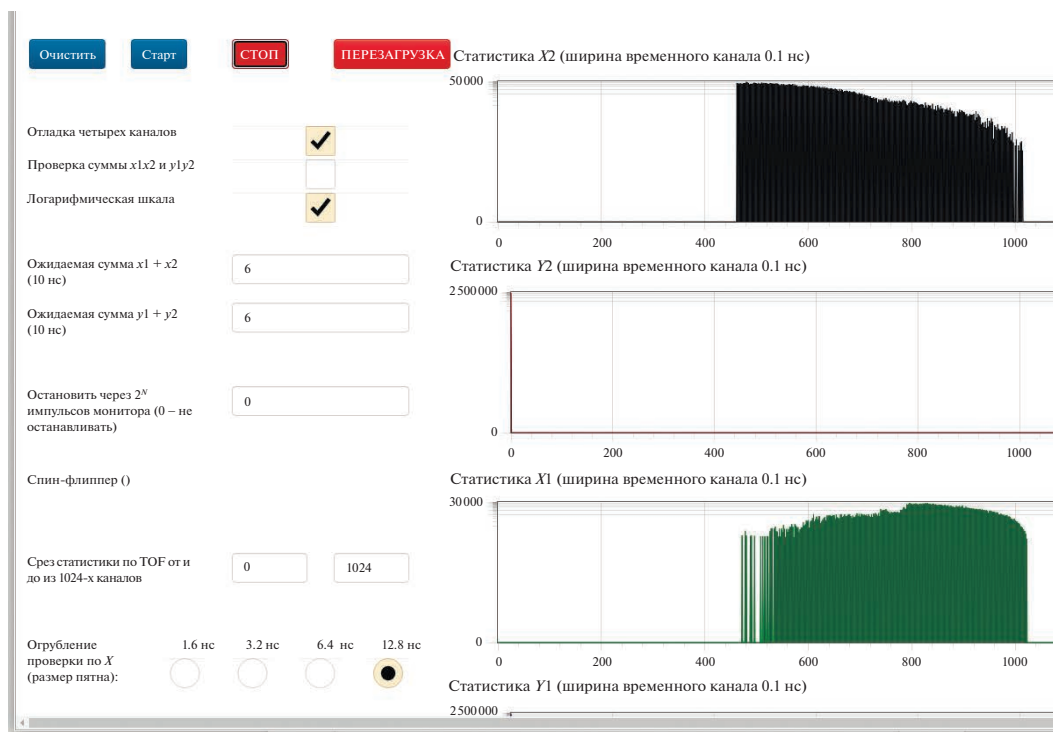


Рис. 8. Изображение пользовательского веб-интерфейса, демонстрирующего статистику адресов, присвоенных двум указателям потока, копирующего данные при переходе с конца в начало буфера ПЛИС с логарифмической шкалой с неизменными графиками задержек сигналов (вместо названия оси абсцисс следует подразумевать адрес памяти для 32-битного целого типа). Наличие переполнения. Частота регистрации событий 1 МГц.

ходимым признаком переполнения кольцевого буфера. В случае переполнения и потери данных (эту ситуацию можно инициировать, понизив приоритет программы в операционной системе), эти две гистограммы на каждом графике по мере набора статистики сблизятся и пересекутся (рис. 8).

ЗАКЛЮЧЕНИЕ

Решение на основе двух кольцевых буферов может показаться слишком сложным, однако это наиболее очевидное решение для этого одноплатного компьютера, не требующее использования низкоуровневых интерфейсов и подразумевающее работу с локальной сетью. Значительно более сложным и эффективным решением в данном случае могла быть модификация ядра операционной системы GNU/Linux и реализация одного кольцевого буфера в оперативной памяти, записываемого напрямую из ПЛИС. Однако программные интерфейсы ядра операционной системы GNU/Linux устаревают быстрее, чем стандарты POSIX, с использованием которого написана программа, а диапазон адресов памяти для обращения по шине Avalon уже зафиксирован в документации одноплатного компьютера. Кроме того, не следует преждевременно разрабатывать более сложное программное обеспечение, чем про-

шивка ПЛИС и принципиальная схема устройства. Возможны улучшения как программного обеспечения, так и устройства в будущем.

Поток исполнения программы, осуществляющий копирование значений из кольцевого буфера ПЛИС в кольцевой буфер в оперативной памяти, работает с достаточной скоростью, чтобы не происходило переполнения кольцевого буфера ПЛИС. Во втором кольцевом буфере переполнение тоже не происходит, и оно менее вероятно. Представленные данные наглядно демонстрируют работоспособность такой архитектуры.

СПИСОК ЛИТЕРАТУРЫ

1. Голубев М.А., Савельева Т.В., Гапон О.Н., Колхидашвили М.Р., Полюшкин А.О., Соловей В.А. // Поверхность. Рентген., синхротр. и нейтрон. исслед. 2022. № 11. С. 1.
2. Charpak G., Bouclier R., Bressani T., Favier J., Zupančič Č. // Nucl. Instrum. Methods. 1968. V. 65. P. 217. [https://doi.org/10.1016/0029-554X\(68\)90568-5](https://doi.org/10.1016/0029-554X(68)90568-5)
3. Kano H., Fukunaga C., Ikeno M., Sasaki O., Sato K., Matsuura S. // IEEE Trans. Nucl. Sci. 2001. V. 48. Iss. 3. P. 509. <https://doi.org/10.1109/23.940108>
4. Соловей В.А., Савельева Т.В., Колхидашвили М.Р., Гапон О.Н. // Приборы и техника эксперимента.

2019. T. 5. C. 145.
<https://doi.org/10.1134/S0032816219050112>
5. *Levchanovsky F.V., Litvinenko E.I., Nikiforov A.S., Gebauer B., Schulz Ch., Wilpert Th.* // Nucl. Instrum. Methods Phys. Res. A. 2006. V. 569. Iss. 3. P. 900.
<https://doi.org/10.1016/j.nima.2006.09.091>
 6. *Toledo J., Beltrán D., Bordas J., Ramos-Lerate I., Martínez J.C., Fernández F.* // IEEE Trans. Nucl. Sci. 2004. V. 51. Iss. 4. P. 1488.
<https://doi.org/10.1109/TNS.2004.832617>
 7. *López-Robles J.M., Alfaro R., Belmont-Moreno E., Grabski V., Martínez-Dávalos A., Menchaca-Rocha A.* // IEEE Trans. Nucl. Sci. 2005. V. 52. Iss. 6. P. 2841.
<https://doi.org/10.1109/TNS.2005.862789>
 8. *Hanu A.R., Prestwich W.V., Byun S.H.* // Nucl. Instrum. Methods Phys. Res. A. 2015. V. 780. P. 33.
<https://doi.org/10.1016/j.nima.2015.01.053>
 9. *Parsakordasiabi M., Vornicu I., Rodríguez-Vázquez Á., Carmona-Galán R.* // Sensors. 2021. V. 21. P. 308.
<https://doi.org/10.3390/s21010308>
 10. *Zhang F., Sun G.* // 2012 Int. Conf. on Control Engineering and Communication Technology. Washington, December 7–9, 2012. P. 974.
<https://doi.org/10.1109/ICCECT.2012.102>
 11. The Scientist and Engineer's Guide to Digital Signal Processing. 2011. <http://www.dspguide.com/pdfbook.htm>.
 12. Lockless Ring Buffer Design. 2009. <https://docs.kernel.org/trace/ring-buffer-design.html>.

Debugging Software for Data Acquisition and Processing for a Neutron Detector with an Architecture of Two Circular-Buffers

M. A. Golubev^{1, *}, A. O. Polyushkin¹, V. A. Solovei¹

¹NRC "Kurchatov Institute" – PNPI, Gatchina, 188300 Russia

*e-mail: golubev_ma@pnpi.nrcki.ru

A technique for debugging the software of a data acquisition and processing device with a network interface for a 2D position-sensitive thermal neutron detector with a delay line readout is described. The original software transfers data through two ring buffers. Changes are proposed in the software code to check the possibility of ring buffer overflow. It is shown that there is no data loss in the ring buffers at input pulse frequencies up to 1 MHz. The corresponding registration dead time is on the order of 1 μ s.

Keywords: neutron detector, delay line readout, field-programmable gate array, circular buffer, web interface, local area network, data processing, software, hardware, data transfer.