DOI: 10.58168/CISMP2024_539-544
УДК 004.9

ЦИФРОВАЯ ОБРАБОТКА СИГНАЛОВ В PHP С ИСПОЛЬЗОВАНИЕМ PCNTL_SIGNAL

В.А. Головин, А.А. Андрюшин
5vladislasv5@gmail.com

ФГБОУ ВО «Воронежский государственный лесотехнический университет имени Г.Ф. Морозова»

Аннотация. В данной работе рассмотрены возможности применения PHP для обработки цифровых сигналов с использованием функции pcntl_signal. Исследованы методы управления сигналами и возможности их обработки, что позволяет создавать многозадачные приложения. Статья демонстрирует, что PHP может успешно использоваться для цифровой обработки сигналов в задачах, требующих простоты в реализации и способности реагировать на сигналы в реальном времени.

Ключевые слова: цифровая обработка сигналов, PHP, pcntl_signal, параллельные процессы, реальное время.

DIGITAL SIGNAL PROCESSING IN PHP USING PCNTL_SIGNAL

V.A. Golovin, A.A. Andryushin
5vladislasv5@gmail.com

Voronezh State University of Forestry and Technologies named after G.F. Morozov

Abstract. This paper explores the possibilities of using PHP for digital signal processing through the pcntl_signal function. It examines signal management methods and their processing capabilities, enabling the creation of multitasking applications. The article demonstrates that PHP can be effectively used for digital signal processing in tasks requiring simplicity in implementation and the ability to respond to real-time signals.

Keywords: digital signal processing, PHP, pcntl_signal, parallel processes, real-time.

Введение

Цифровая обработка сигналов (или просто ЦОС) — это способ, как можно работать с сигналами. Например, когда мы слушаем музыку через интернет или смотрим видео, это тоже обработка сигналов. Она важна и в более сложных штуках, например, в мед. устройствах и на заводах. Смысл ЦОС — брать некий сигнал, менять его, если нужно, и получать что-то полезное. Для этого часто делают всякие вычисления, например, измеряют шум или применяют математические приёмы, чтобы достать из сигнала то, что нужно.

Работа с сигналами часто требует быстроты, буквально в реальном времени. Это важно в местах, где нужна точность, например, в медицине или

автоматике. Ведь если задержка будет даже на долю секунды, результат может отличаться и, возможно, не понравится.

Почему обычно для ЦОС берут другие языки

Часто для работы с сигналами программисты выбирают такие языки, как C и C++. Это потому, что они дают возможность делать операции быстро и точно управлять ресурсами. Например, в роботах любая небольшая задержка может сбить точность движений. Конечно, такие языки требуют больше времени на программирование, но зато они дают то, что нужно для задач, где важна точность и скорость.

PHP и работа с сигналами

Сейчас начали обращать внимание и на другие языки, например, на PHP. Хотя PHP часто используется для создания сайтов, у него есть функции, которые помогают работать с сигналами. Одна из таких функций — это pcntl_signal. Она даёт возможность PHP принимать и обрабатывать сигналы, которые посылает операционная система.

Например, с помощью pcntl_signal PHP может реагировать на сигналы SIGTERM (сигнал, чтобы завершить работу) и SIGHUP (сигнал для перезагрузки). Это значит, что программы на PHP могут адаптироваться под разные ситуации, например, корректно завершить работу или обновить настройки.

Основные этапы в обработке сигнала

Для работы с цифровыми сигналами есть несколько шагов, которые нужны, чтобы получить нужный результат:

Перевод аналогового сигнала в цифровой (АЦП). Это нужно, чтобы работать с сигналом как с набором чисел.

Предварительная обработка. Здесь убирают шумы и исправляют искажения.

Основная обработка. Тут уже делают вычисления, например, преобразование Фурье, чтобы выделить частоты.

Финальная обработка и вывод. Тут либо выводят данные на экран, либо сохраняют для следующего шага.

Обратный перевод в аналоговый вид (ЦАП), если, например, нужно вывести звук на колонки.

Пример фильтрации шума с помощью PHP

Чтобы убрать шум в простых случаях, можно усреднять значения.

```php
function filter_noise($signal) {
    $filtered_signal = [];
    for ($i = 1; $i < count($signal) - 1; $i++) {
        $filtered_signal[] = ($signal[$i - 1] + $signal[$i] + $signal[$i + 1]) / 3;
    }
    return $filtered_signal;
}

$raw_signal = [10, 15, 10, 12, 14, 13];
$clean_signal = filter_noise($raw_signal);
print_r($clean_signal);
```

Рисунок 1 – Пример усреднения значений

Этот метод позволяет сгладить шум, делая сигнал более "чистым".

Использование pcntl_signal для сигнала SIGTERM

pcntl_signal помогает программе на PHP срабатывать на сигналы от устройств. Вот пример работы с сигналом SIGTERM:

```php
declare(ticks=1);

function завершение() {
    echo "Программа завершает работу...\n";
    exit;
}

pcntl_signal(SIGTERM, "завершение");

while (true) {
    // Основной процесс программы
}
```

Рисунок 2 – Корректное завершение программы

Обработка разных сигналов

Иногда надо обрабатывать несколько сигналов, например, завершение и перезагрузку настроек:

```php
declare(ticks=1);

function завершение() {
    echo "Процесс завершает работу...\n";
    exit;
}

function перезагрузка() {
    echo "Настройки перезагружаются...\n";
}

pcntl_signal(SIGTERM, "завершение");
pcntl_signal(SIGHUP, "перезагрузка");

while (true) {
    // Основной процесс программы
}
```

Рисунок 3 – Завершение и перезагрузка настроек

В этом примере сигнал SIGTERM завершает процесс, а SIGHUP — перезагружает настройки, что удобно для серверных программ.

Параллельная работа с сигналами

Некоторые приложения требуют, чтобы программа не прерывалась, даже если поступают сигналы. Для этого можно создать несколько процессов с pcntl_fork:

```php
declare(ticks=1);

for ($i = 0; $i < 3; $i++) {
    $pid = pcntl_fork();

    if ($pid == -1) {
        die("Не удалось создать процесс");
    } elseif ($pid == 0) {
        echo "Дочерний процесс выполняет задачу $i\n";
        sleep(2);
        exit(0);
    }
}

while (pcntl_waitpid(0, $status) != -1);
```

Рисунок 4 – Создание процесса pcntl_fork

В этом примере основной процесс создает три дочерних, и каждый выполняет свою задачу.

Заключение

Функция pcntl_signal в PHP даёт программистам возможность работать с сигналами. Хотя PHP редко используется для обработки сигналов, его функции для выполнения нескольких задач и реакции на события позволяют использовать его для создания многозадачных приложений. Так что pcntl_signal делает PHP более гибким.

Список литературы

1. Оппенхайм, А.В. Цифровая обработка сигналов / А.В. Оппенхайм, Р.У. Шафер // Практическое руководство для инженеров. – М.: Пирсон, 2010. – С. 125-139.
2. Лайонс, Р.Г. Основы цифровой обработки сигналов / Р.Г. Лайонс. – М.: Вильямс, 2011. – 384 с.
3. Прокис, Дж.Г. Принципы цифровой обработки сигналов / Дж.Г. Прокис, Д.Г. Манолакис // Технология обработки сигналов: Теория и практика. – М.: Пирсон, 2006. – С. 72-89.
4. Миллер, Д. Паттерны проектирования в PHP / Д. Миллер, С. Досс // Программирование PHP. – М.: О’Рейли, 2013. – С. 47-59.
5. Зольников К.В., Гамзатов Н.Г., Евдокимова С.А., Потапов А.В., Допира Р.В., Кучеров Ю.С., Яночкин И.Е., Стоянов С.В., Плотников А.М. Моделирование процессов в полупроводниковых структурах при радиационном воздействии // Моделирование систем и процессов. – 2022. – Т. 15, № 3. – С. 106-127.
6. Кривобоков Н.В., Сазонова С.А., Акамсина Н.В. Численное моделирование технического состояния конструкций каркасного здания // Моделирование систем и процессов. – 2022. – Т. 15, № 1. – С. 52-66.
7. Журавлева И.В., Попова Е.А. Полупроводниковые технологии для реализации радиационно-стойких СБИС // Моделирование систем и процессов. – 2022. – Т. 15, № 1. – С. 44-52.

References

1. Oppenheim, A.V. Digital Signal Processing / A.V. Oppenheim, R.W. Schafer // Practical Guide for Engineers. – Moscow: Pearson, 2010. – pp. 125-139.
2. Lyons, R.G. Fundamentals of Digital Signal Processing / R.G. Lyons. – Moscow: Williams, 2011. – 384 p.
3. Proakis, J.G. Principles of Digital Signal Processing / J.G. Proakis, D.G. Manolakis // Signal Processing Technology: Theory and Practice. – Moscow: Pearson, 2006. – pp. 72-89.

4. Miller, D. Design Patterns in PHP / D. Miller, S. Doss // PHP Programming. – Moscow: O'Reilly, 2013. – pp. 47-59.
5. Zolnikov K.V., Gamzatov N.G., Evdokimova S.A., Potapov A.V., Dopira R.V., Kucherov Y.S., Yanochkin I.E., Stoyanov S.V., Plotnikov A.M. Modeling Processes in Semiconductor Structures under Radiation Exposure // Modeling Systems and Processes. – 2022. – Vol. 15, No. 3. – pp. 106-127.
6. Krivobokov N.V., Sazonova S.A., Akamsina N.V. Numerical Modeling of the Technical Condition of Frame Building Structures // Modeling Systems and Processes. – 2022. – Vol. 15, No. 1. – pp. 52-66.
7. Zhuravleva I.V., Popova E.A. Semiconductor Technologies for Implementing Radiation-Hardened Integrated Circuits // Modeling Systems and Processes. – 2022. – Vol. 15, No. 1. – pp. 44-52.