АВТОНОМНЫЕ ПЕРИФЕРИЙНЫЕ УСТРОЙСТВА: МАЛОЕ ПОТРЕБЛЕНИЕ В РЕАЛЬНОМ ВРЕМЕНИ

ГРЕГОР САНДЕРДИК (GREGOR SUNDERDIEK), менеджер по развитию бизнеса MCU8 EMEA, Microchip Technology

Независимая от ядра периферия (CIP) представляет собой автономные, связанные между собой интеллектуальные устройства. Благодаря этой периферии при выполнении ряда задач не требуется вмешательства центрального процессора (ЦП), что позволяет использовать его для других задач. В результате приложения получают несколько преимуществ. В статье рассматриваются преимущества CIP-периферии на практических примерах.

Во-первых, CIP-периферия позволяет исключить использование центрального процессора в установлении связи между внешними устройствами. В таких случаях, пока ядро находится в режиме сна, работа программного обеспечения не прерывается. Очевидно, если ядро «засыпает», а ПО не используется, потребление тока снижается. У центрального процессора, являющегося частью микроконтроллера, – самый высокий потребляемый ток. Следовательно, использование независимых от ядра периферийных устройств позволяет уменьшить энергопотребление.

Во-вторых, независимые от ядра периферийные устройства не вызывают прерываний, что намного ускоряет коммуникации. Ядру центрального процессора, которое управляет программным обеспечением, требуется достаточно много времени на генерацию прерывания по сигналу периферийного устройства для выполнения определенной операции. На это прерывание необходимы пять тактовых циклов, из которых два цикла затрачиваются на соответствующую операцию перехода. Кроме того, несколько циклов может понадобиться на переключение контекста, чтобы сохранить данные в регистрах стека, что зависит от особенностей приложения. Независимая от ядра периферия позволяет намного ускорить процесс прерывания.

В-третьих, использование независимых от ядра периферийных устройств позволяет быстрее вывести продукцию на рынок за счет меньшего объема кода программного обеспечения, возложив определенные задачи на оборудование. В результате снижается риск появления программных ошибок, и меньше времени требуется на проверку ПО. Таким образом, время разработки продукции становится меньше.

У микроконтроллеров серии AVR вся независимая от ядра периферия подключается с помощью системы обработки событий (Event System). В этой системе сигналы о внешних событиях поступают в генератор событий через мультиплексор. События бывают синхронными и асинхронными. Асинхронным событиям требуется меньше одного тактового цикла, а синхронным – два таких цикла.

Многие периферийные устройства, подключенные к системе обработки событий, становятся независимыми от ядра. К этим устройствам относятся таймеры, счетчики реального времени (RTC), таймеры периодических прерываний (PIT), конфигурируемая пользователем логика (CCL), аналоговые компараторы (AC), АЦП, универсальные синхронно-асинхронные приемопередатчики (USART) и интерфейсы ввода/вывода общего назначения (GPIO).

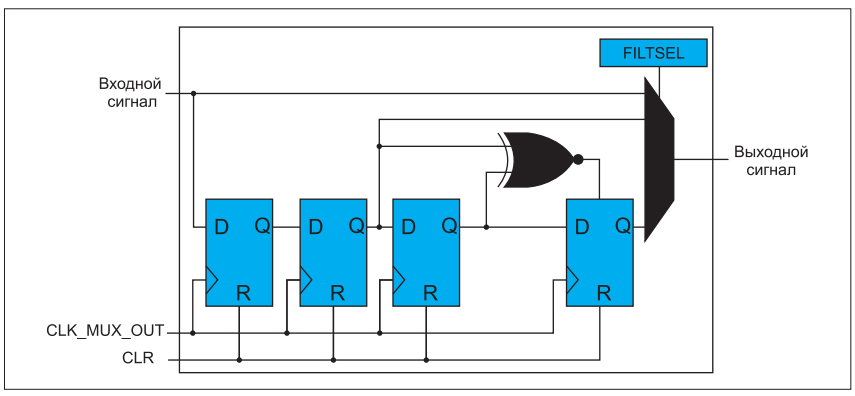

Рис. 1. **Фильтр блока CCL**

ПРИМЕНЕНИЕ CIP

Перед использованием независимые от ядра периферийные устройства конфигурируются. Центральный процессор выполняет команду по инициализации системы обработки событий и требуемых периферийных устройств.

УСТРАНЕНИЕ ДРЕБЕЗГА МЕХАНИЧЕСКИХ КЛАВИШ

В ряде современных приложений для ввода по-прежнему используются кнопки. Для каждой из них требуется логический блок, позволяющий устранять возникающий при коммутации дребезг, или ПО, предотвращающее ложное срабатывание. Реализация этой функции программным способом при использовании микроконтроллеров AVR не представляет особого труда. Функция осуществляется с помощью задержек или логического блока в прикладной программе. Заметим, что в таких случаях используются ресурсы центрального процессора. Факт нажатия кнопки проверяется путем опроса или прерывания

от контроллера GPIO. И в том, и в другом случаях используется ЦП.

Задача по устранению дребезга механических клавиш решается с помощью независимой от ядра периферии. При этом нагрузка на ЦП не возрастает – необходимо только воспользоваться блоком CCL. Интерфейс GPIO там, где происходит подключение кнопки, конфигурируется как асинхронный генератор событий. Блок CCL фиксирует внешнее событие. Сигнал с вывода GPIO на вывод CCL передается без задержки. Таблица истинности в блоке CCL конфигурируется таким образом, чтобы выходной сигнал был равен входному. Выходной сигнал CCL поступает в фильтр (см. рис. 1). Ложные сигналы устраняются из входного сигнала. При этом имеется возможность для выходного сигнала установить задержку фильтра в диапазоне 2–5 тактовых циклов (периферийного или вспомогательного тактового генератора). Например, при использовании тактовой частоты 32 кГц задержка составит 1,5 мс. Время задержки можно увеличить, используя другую тактовую частоту, другой генератор или таймер.

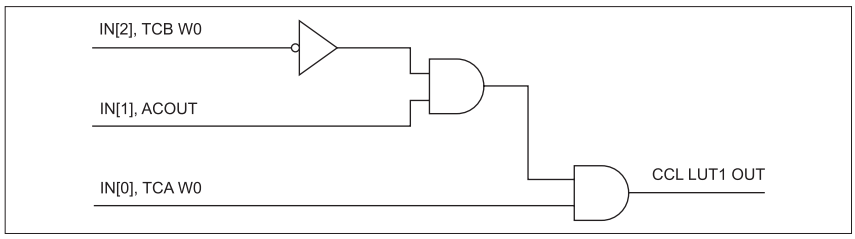

Рис. 2. **Генерация мертвого времени CCL LUT 0**

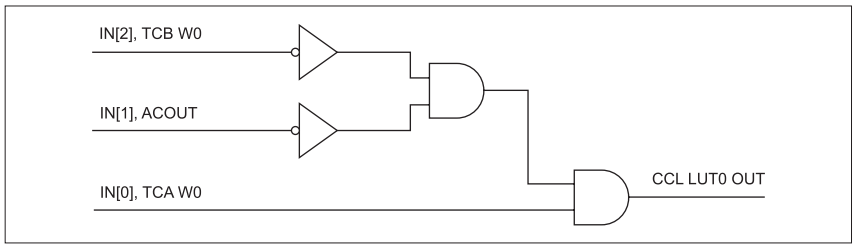

Рис. 3. **Генерация мертвого времени CCL LUT 1**

ЗАДЕРЖКА С ТАЙМЕРОМ

Например, таймер–счетчик (Timer/Counter B, TCB) настраивается для работы в однотактном режиме. При получении таймером сигнала о событии начинается отсчет, пока не будет достигнуто запрограммированное максимальное значение, после чего отсчет прекращается. Выходной сигнал с таймера TCB подается в блок CCL. В этом блоке создается требуемая комбинация сигнала задержки. Такой подход обеспечивает очень гибкую настройку задержки времени. Каждый новый сигнал о событии, поступающий в таймер TCB, инициирует новый отсчет.

ГЕНЕРАЦИЯ МЕРТВОГО ВРЕМЕНИ

Мертвое время используется в приложениях, в которых ключи (транзисторы, полевые транзисторы или IGBT) установлены последовательно между источником питания и заземлением. Если оба активны в одно и то же время, возникает короткое замыкание. Примером может служить мостовой силовой каскад, который обычно применяется для управления приводом. В зависимости от приложения мертвое время разделяет последовательные переключения или импульсы ШИМ. Мертвое время между этими импульсами требуется, например, при синусоидальном управлении и между переключениями в однополюсных бесщеточных двигателях по постоянному току (BLDC). Мертвое время между ШИМ-импульсами генерируется с помощью таймера TCD (Time Code Display – формирование временного кода).

Для генерации мертвого времени между переключениями требуются два таймера, логический блок CCL и сигнал переменного тока. На рисунках 2–3 показана таблица истинности блока CCL. Таймер TCA генерирует базовый ШИМ-сигнал управления приводом. Сигнал от внешнего источника переменного тока, поданный на датчик Холла двигателя, поступает через систему обработки событий в таймер TCB. Этот таймер генерирует сигнал мертвого времени при получении сигнала из источника переменного тока. Блок CCL комбинирует сигналы таймеров TCA (ШИМ), TCB (мертвое время) и сигнал из источника переменного тока. Входным сигналам, которые можно выбирать непосредственно в конфигурации CCL, не требуется использование системы обработки событий. Между этими модулями имеются фиксированные соединения. Затем блок CCL генерирует два ШИМ-сигнала (см. рис. 4), которые управляют ключами двигателя. Двигатель работает без использования ресурсов ЦП. Подробнее см. [1].

АВТОМАТИЧЕСКИЙ ШИМ-СИГНАЛ ОТКЛЮЧЕНИЯ

Поскольку во многих приложениях осуществляется контроль над потребляемым током, он не превышает максимального уровня. Эту функцию легко реализовать с помощью аналогового компаратора сигналов переменного тока. Компаратор измеряет напряжение (ток через резистор) с помощью шунтирующего резистора. При превышении заданного порога генерация ШИМ-сигнала мгновенно прекращается. В обоих приведенных ниже примерах используется независимая от ядра периферия. Подача выходного ШИМ-сигнала прекращается при обнаружении превышения тока; вмешательства ЦП не требуется.

УПРАВЛЕНИЕ СВЕТОДИОДНЫМ ОСВЕЩЕНИЕМ

Таймер–счетчик A0 генерирует ШИМ-сигнал для светодиода. Поскольку для

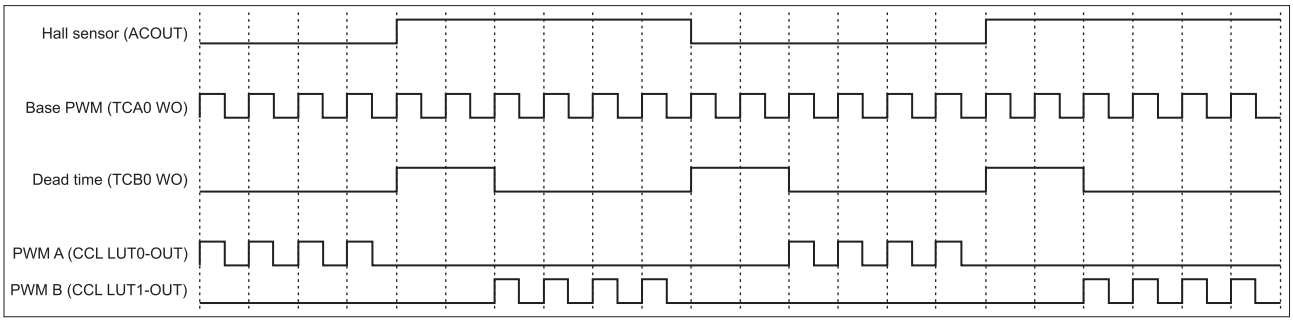

Рис. 4. **Временная диаграмма генерации мертвого времени**

обнаружения перегрузки по току используется источник импульсного тока, а блок CCL комбинирует эти сигналы, при обнаружении перегрузки по току автоматически прекращается подача ШИМ-сигнала. Сигналы переменного тока и TCA0 подаются через систему обработки событий в блок CCL. Выходной сигнал переменного тока и ШИМ-сигнал конфигурируются с помощью таблицы истинности блока CCL (см. табл.).

Таблица истинности для устранения отказов

AC	TCA0 – ШИМ	Выход
0	1	1
0	0	0
1	x	0

УПРАВЛЕНИЕ ПРИВОДОМ С ПОМОЩЬЮ TCD

Бесщеточный двигатель постоянного тока управляется с помощью таймера TCD, который генерирует ШИМ-сигналы двух каналов и двух вспомогательных каналов для управления четырьмя MOSFET силового мостового каскада. Величина тока двигателя определяется путем измерения напряжения шунта, который установлен между этим двигателем и землей. Сигнал импульсного тока поступает через систему обработки событий в таймер–счетчик D0 (TCD). Одна из функций этого таймера состоит в устранении отказов. При превышении порогового значения сигнала переменного тока (в случае перегрузки по току) поступает сообщение таймеру TCD, и подача ШИМ-сигналов автоматически прекращается.

ИЗМЕРЕНИЕ ВРЕМЕНИ ПРОХОЖДЕНИЯ СИГНАЛА

Измерение времени прохождения сигнала применяется для определения расстояния до конкретного объекта. Измерения начинаются в тот момент, когда сигнал покидает приемопередатчик, и завершаются при поступлении этого сигнала в приемник. Расстояние определяется с помощью времени прохождения сигнала и его скорости. В рассматриваемом примере мы измеряем расстояние, которое проходит ультразвуковой сигнал. В этом приложении используются независимые от ядра периферийные устройства TCA0, TCB0, TCS0, AC и 2×CCL. Время прохождения можно рассчитать без помощи ЦП.

На рисунке 5 показана справочная таблица LUT1, с помощью которой генерируется передаваемый сигнал (LUT1). Выход таймера TCA генерирует ШИМ-сигнал, а выход B таймера TCD – маску передачи. Инвертированная маска передачи и ШИМ-сигнал являются логическим И; с помощью таблицы истинности LUT1 генерируется сигнал передачи.

Таблица LUT0 генерирует отраженный сигнал. Выход AC Out обеспечивает активность приемной линии, а выход TCD Out A – приемную маску. Инвертированная приемная маска и линия приема являются логическим И, с помощью которого генерируется отраженный сигнал, что видно из таблицы истинности LUT0.

Защелка SR сбрасывается первым переданным сигналом, после чего активируется счетчик в TCD. Отсчет прекращается при появлении отраженного сигнала и установки защелки SR. Информация о времени прохождения хранится в счетчике таймера TCD; при этом ресурс ЦП не расходуется. Помощь центрального процессора необходима только при расчете расстояния, когда время прохождения умножается на скорость сигнала. Подробнее см. [4].

ВЫВОДЫ

Микроконтроллеры новых серий ATtiny1617/1616/1614/817/816/814/417 от компании Microchip привнесли независимые от ядра периферийные устройства в семейство МК tinyAVR. С помощью этой периферии приложение может работать в реальном времени с меньшими программными издержками и меньшим потребляемым током, чем без этих устройств. Приведенные примеры показали, что независимая от ядра периферия легко настраивается, а производительность системы становится выше при меньшем энергопотреблении по сравнению с программно реализованными решениями. Такой уровень производительности в реальном времени не достигается даже при использовании намного более эффективных микроконтроллеров, но если это и происходит, то за счет большей потребляемой мощности.

Литература

1. AVR42778: Core Independent Brushless DC Fan Control Using Configurable Custom Logic on ATtiny817. Application Note//www.atmel.com.

2. AVR ATtiny817//www.microchip.com.

3. AVR42778 Application Note: Core Independent Brushless DC Fan Control Using Configurable Custom Logic on ATtiny817//ww1.microchip.com.

4. AVR42779 Application Note: Core Independent Ultrasonic Distance Measurement with ATtiny817//ww1.microchip.com.

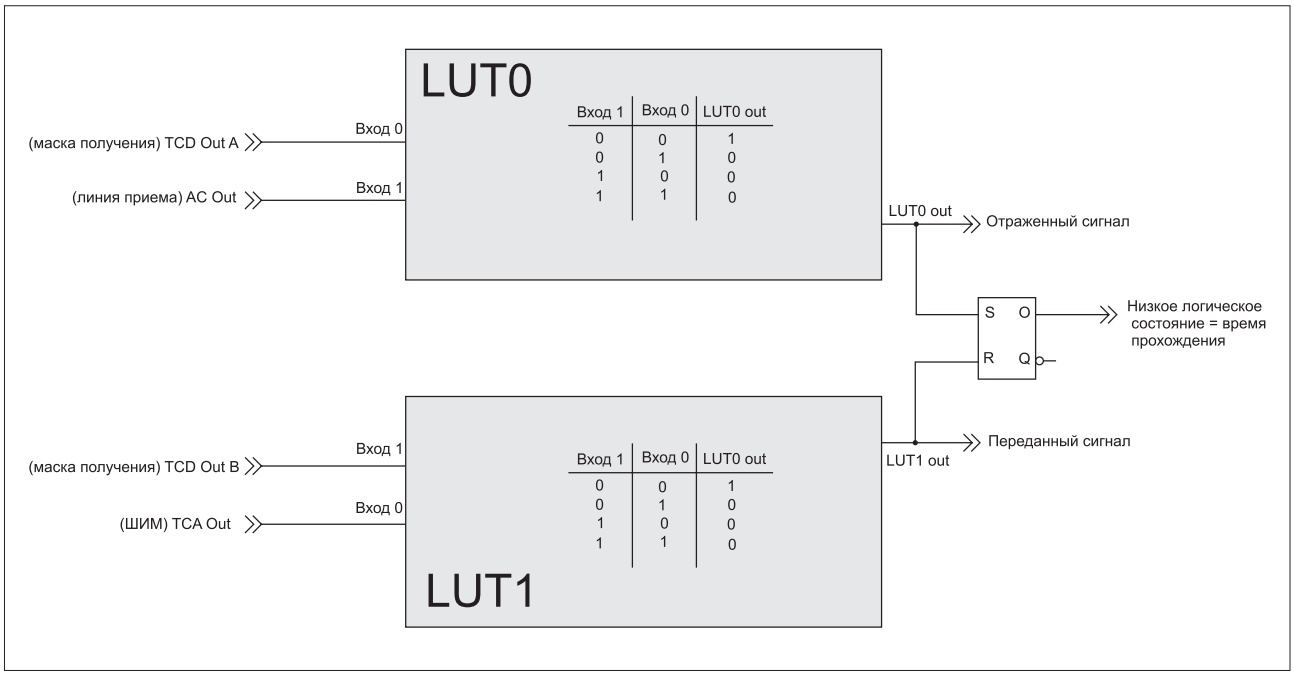

Рис. 5. **Измерение времени прохождения ультразвукового сигнала**