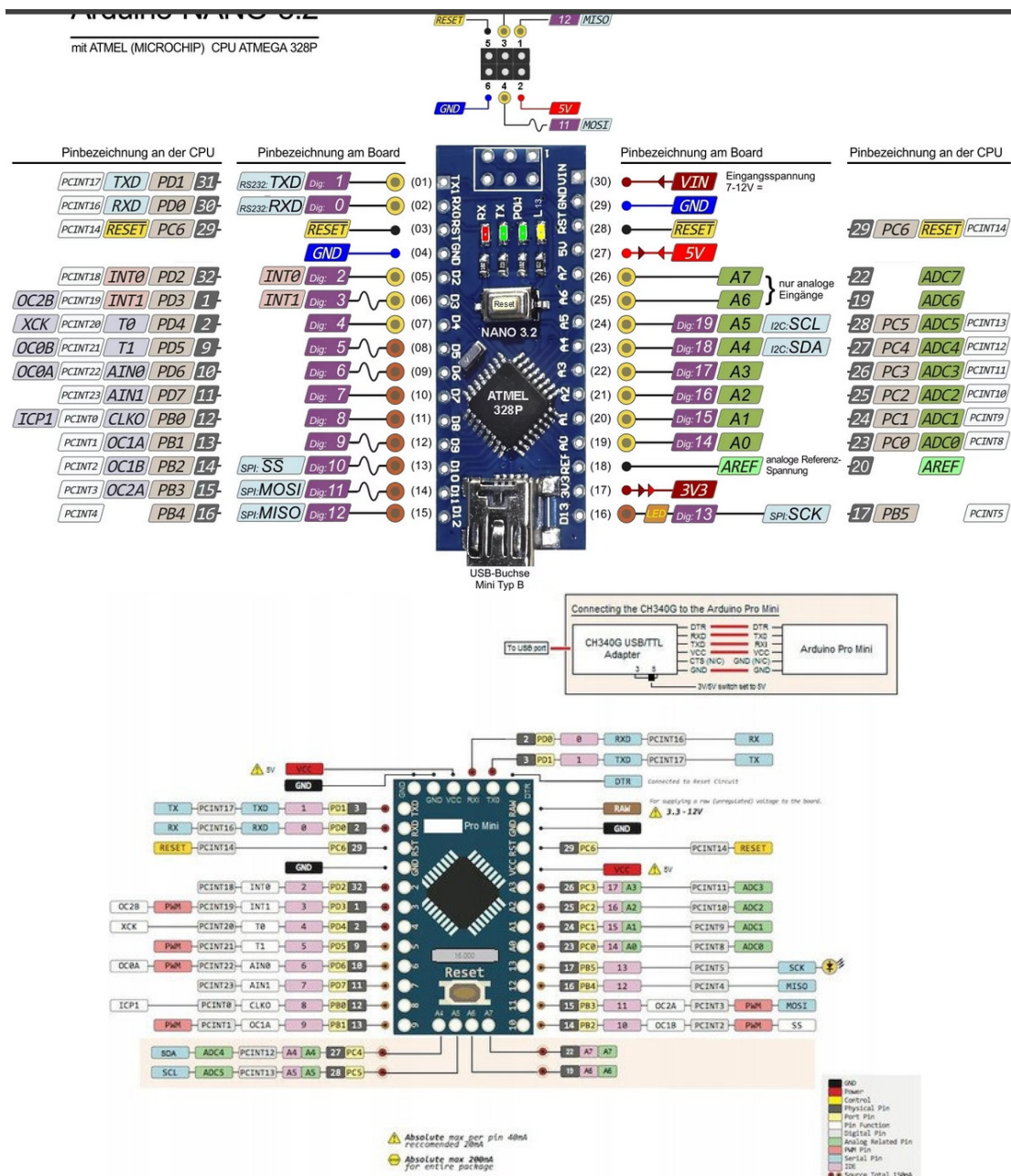


2. ФУОЗ НА ПЛАТФОРМЕ ARDUINO.

Представленный ниже ФУОЗ является примером описанного выше варианта на «медленном» МК. В отличие от других подобных устройств предлагается использовать готовые платы, на которых уже установлен МК и все необходимые для его работы элементы, включая и UART интерфейс – платформа Arduino совместимых плат с кварцевым резонатором 16 МГц (Рис. 2). Внутренний загрузчик можно при желании отключить.



Arduino Nano

Arduino Pro mini

Рис. 2 Платформа

Отмечу, что вариант Arduino Pro mini мне кажется предпочтительнее из-за отключаемого (внешнего) преобразователя UART-USB, ибо в 99 процентах рабочего времени ФУОЗ он не нужен. Кроме того, сам преобразователь UART-USB тоже является устройством на основе микроконтроллера, и как показала практика, в неблагоприятных условиях подвержен «зависанию» при больших импульсных помехах.

Все программное обеспечение ФУОЗ содержит три уровня. Уровень Ядро (подключаемый при компиляции файл **CORE.INC**) - обеспечивает основные функции: контроль входных линий, измерение временных интервалов, расчетный модуль, работу четырех каналов АЦП, сервисные функции и прочее. Уровень Интерфейс - программа для персонального компьютера (ПК) **UOZ.EXE** (далее UOZ) для операционной системы Windows вместе интерфейсным модулем Ядра обеспечивает взаимодействие с пользователем (настройка параметров, индикация текущего

состояние, протоколирование работы Ядра). **Оба этих уровня носят завершённый характер и при переходе от двигателя к двигателю в коррекции не нуждаются.**

Главный уровень (файл **MAIN.ASM**) или Моторная часть - обеспечивает смысловую связь Ядра с сигналами датчиков, формирователями импульсов зажигания. **Именно эту часть программного обеспечения и надлежит изменять в процессе адаптации к конкретному двигателю и системе датчиков.** В этом блоке задаются все константы Ядра и устанавливаются режимы работы его отдельных частей.

Одним из предлагаемых решений является квазивиртуализация сигналов входных датчиков. Пользователь фактически сам определяет каким внешним физическим событиям соответствуют основные этапы функционирования Ядра, воздействуя на него изменением переменных и регистров Ядра, вызовом в нужные моменты времени макросов **StartDelta** и **StopDelta**, которые является отражением состояния виртуального входного датчика. В свою очередь, Ядро вызывает в нужные моменты времени соответствующие подпрограммы Моторной части. Эта квазивиртуализация стала следствием применения алгоритма защиты входных физических линий от импульсных помех, заключающегося в процедуре дополнительного чтения входных линий после их изменения на входах микроконтроллера, способных вызывать прерывание. Все вышеизложенное дало возможность использовать построенную систему и для двух-трех цилиндровых двигателей (последовательный запуск Ядра между цилиндрами), но и с разными типами датчиков и формирователей их сигналов.

3. ОСНОВЫ ФУНКЦИОНИРОВАНИЯ

Пусть есть виртуальный импульс от датчика известного углового размера Δ (метка **Delta**) перед верхней мертвой точкой цилиндра (**BMT**) - Рис.3. Угловое расстояние между меткой **Delta** и **BMT** - β определяется конструкцией датчика. Метка **Delta** возникает один раз в заданный интервал вращения **Base** (угол-база), значение которого следует выбирать по Таблице 1.

Таблица 1

Цилиндры	1	2 ⁽¹⁾	3 ⁽¹⁾
Base	360	180	120

Примечание: Ядро применяется последовательно от цилиндру к цилиндру.

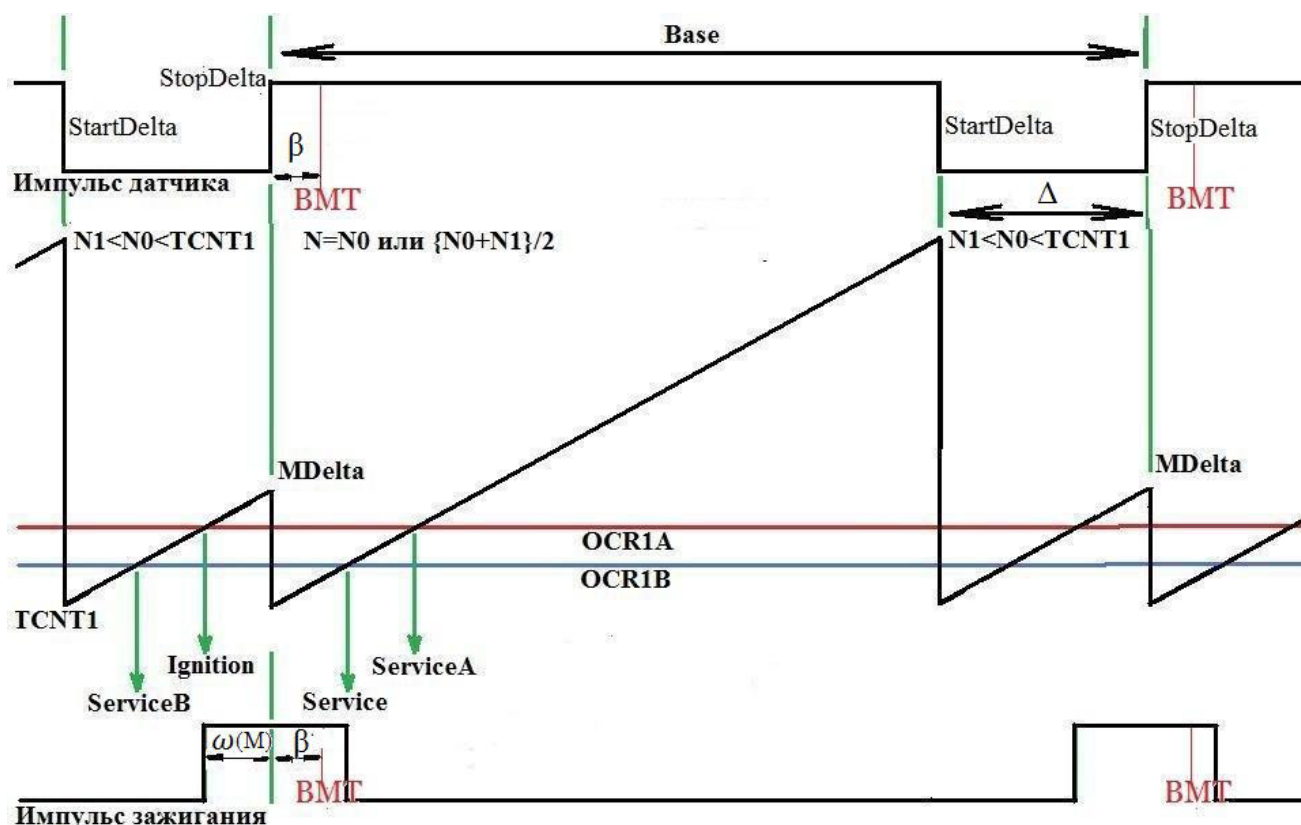


Рис. 3 Принцип работы системы

Центральным узлом Ядра является 16 разрядный счетчик T1 в нормальный режим, на вход которого подключен управляемый тактовый генератор – 8 разрядный таймер T0 в режиме СТС. Количественное значение тактовой частоты F определяется соответствующей константой в Моторной части - F_n . Формула для расчета тактовой частоты с кварцевым резонатором 16 МГц имеет вид:

$$F = \frac{16 \text{ МГц}}{2 \cdot (1 + F_n)} = \frac{8000000}{(1 + F_n)}, \text{ Гц}$$

Для обеспечения работы этой связки таймеров необходимо соединить линии D5 и D6 (5 и 6) используемой платформы с МК (линии PD5 и PD6 МК) – Рис.4.

Значения счетчика таймера T1 накопленное от события StopDelta до StartDelta и являющееся оценкой периода вращения маховика в отсчетах тактовой частоты, поступает на сдвигающую цепочку $\rightarrow N0 \rightarrow N1$. Данное решение обусловлено предоставлением возможности принятия за оценку скорости вращения маховика как значения счетчика таймера в последнем цикле, так и среднее значение двух последних циклов. Полученное значение используется для расчета времени задержки искрообразования. Значения счетчика таймера T1, накопленное от события StartDelta до StopDelta, является оценкой длины метки MDelta в отсчетах тактовой частоты на данной скорости вращения маховика. Значения MDelta используется для расчета времени отключения выходного сигнала.

Таймер T1 имеет два равнозначных канала сравнения с предустановленными значениями OCR1A и OCR1B. Канал OCR1A используется в Ядре для формирования временной задержки искрообразования в интервале между метками StartDelta до StopDelta. Основной математический аппарат, используемый в Ядре, направлен на расчет о программирование значения порога срабатывания OCR1A при заданных оборотах. С этим действием связаны две подпрограммы Моторной части. Во-первых, это подпрограмма Ignition, которая вызывается ядром при превышении значения таймера T1 кода в канале OCR1A. Это есть момент подачи искры в цилиндр. **Обращаю внимание**, что искра подается по зависимости $\omega(M)$, см. Рис.3, относительно не ВМТ, а конца метки Delta, которая смещена на угол β от неё, зависящий от конструкции двигателя и датчиков. Следовательно реальный угол опережения зажигания, не может быть меньше, чем β и не больше, чем $\Delta + \beta$. Во-вторых, это подпрограмма UOZMinError вызывается при попытке установить слишком низкий уровень OCR1A. В силу того, что математический аппарат требует времени для проведения вычислений, то установить порог срабатывания OCR1A ниже, чем значение таймера, которое уже накопилось во время работы математического блока, невозможно. При наступлении этого события активируется флаг Status.UOZMin и искрообразования через обычный механизм не происходит. Пользователь сам решает надо ли формировать сигнал зажигания по этому событию.

Очевидно, что при малых оборотах счетчик T1 может переполняться. Ядро помечает это событие как режим Низких оборотов (LowMode). В этом случае искрообразование через механизм Ignition и UOZMinError не происходит и следует позаботиться пользователю в Моторной части об искрообразовании во время детектирования события StopDelta. В зависимости от величины тактовой частоты, поступающего на таймер T1 сигнала нижний предел оборотов, при которых включится режим Низких оборотов, должен быть менее или равен характерным холостым оборотам двигателя.

Второй канал OCR1B используется Ядром в качестве средства отключения включенного сигнала зажигания в интервале между метками StopDelta до StartDelta. Расчет и программирование значения OCR1B осуществляется после получения оценки MDelta. Для обеспечения этого в Ядро встроены подпрограммы SetService и SetEOCR1B. При условии превышения текущего значения таймера значения OCR1B Ядром вызывается программа **Service**, в котором и следует включить сигнал управления зажигания цилиндра.

В случае чрезвычайно низких оборотов счетчик таймера T1 начнет переполняться и при оценке величины MDelta. Вплоть до этого момента, механизм отключения включенного сигнала зажигания функционирует за счет искусственного увеличения разрядности OCR1B до 24 бит (EOCR1B). Данное решение позволяет выдерживать корректные временные интервалы для отключения импульса зажигания при искрообразовании начиная с 50-200 оборотов в минуту, что очень важно для режима запуска. Дополнительно это позволяет получать оценку частоты вращения двигателя программой **UOZ** (кнопка «Т» стр. 33).

Подпрограммы Ignition и Service являются не отключаемыми. Дополнительно к ним существуют еще две отключаемые подпрограммы ServiceB и ServiceA. Подпрограмма ServiceB может вызываться ядром при равенстве величины счетчика с порогом OCR1B на интервале между метками StartDelta до StopDelta. Для использования этого функционала следует разрешить это действие при настройке Ядра и производить программирование порога после метки StartDelta. Аналогичная ситуация с функционалом ServiceA на интервале между метками StopDelta до StartDelta. Программировать порог OCR1A для этого следует программировать после метки StopDelta.

Еще одна функция, возложенная на таймер T1 это детектирование полной остановки двигателя (StopMode). Как было указано выше, в режиме Низких оборотов (LowMode) происходит переполнение таймера T1. Ядро проводит подсчет числа переполнения при условии отсутствия вызова событий StartDelta и/или StopDelta. Считаем, что двигатель остановлен, если этих событий не было заданной количество секунд.

Рассмотрим механизм обработки входных сигналов – рис. 4. Сигналы от двух входных датчиков поступают на входы микроконтроллера, способные вызывать асинхронные прерывания int0 и int1, вектора обработки прерываний которых указывают на одну и ту же программу обработки прерывания. При возникновении любых изменений на входных линиях происходит чтение всех линий и размещение их логического состояния в соответствующих сдвиговых регистрах. Так начинается выполнения протокола защиты от пульсаций, причем на время его выполнения прерывания int0 и int1 игнорируются. Одновременно с этим активизируется таймер T2, настроенный на вызов программного прерывания с периодом от нескольких микросекунд (ProtectClk). С этим периодом (интервалом дискретизации) продолжается чтение состояния входных линий и запись их состояния в сдвиговые регистры DH1 и DH2. Процесс останавливается при повторении на всех каналах одного (своего) уровня в течении ProtectCount тактов таймера T2. На рис.4 и рис.5 представлен случай ProtectCount=2. На этом процесс выполнения протокола защиты от пульсаций завершается. Результирующие логические уровни размещаются в младшие биты нибблов сдвигового регистра Событий (Events). Таким образом, содержание этого регистра изменяется в результате изменения любого входного сигнала и отражает их состояние в одни и те же моменты времени (моменты изменения любого из сигналов), причем текущем и трех предыдущих. Данное решение, вместе подпрограммой Моторной части **Actions**, является частью квазивиртуализации сигналов входных датчиков. Подпрограмма Action должна строить реакцию программной части на регистр Событий и выполнять макросы StartDelta и StopDelta при двух разных значения Events.

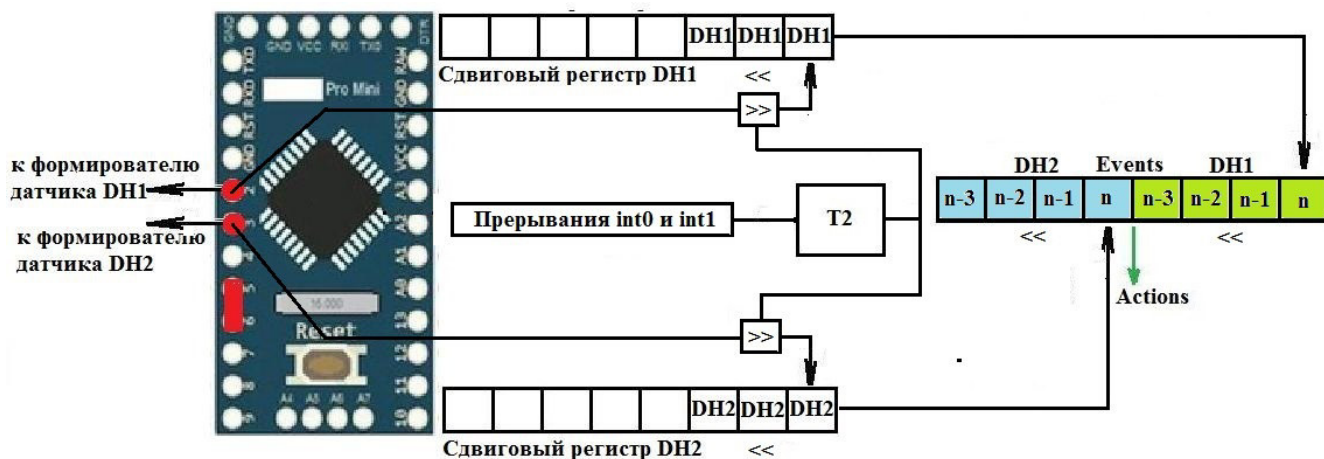


Рис. 4 Механизм обработки входных сигналов

Следует отметить следующие моменты. Во-первых, канал датчика DH2 является отключаемым. В случае его отключения изменения на линии D3(3) используемой платформы с МК не приводят к возникновению прерывания int1 и сдвиговая цепочка DH2 заполняется логическими нулями. Во-вторых, выполнение протокола защиты от пульсаций «отодвигает» момент реакции всего программного комплекса на реальные изменения входных линий. В лучшем случае этот процесс будет длиться $\text{ProtectCount} * \text{ProtectClk}$ микросекунд плюс характерные временные интервалы вызова и обработки прерываний микроконтроллера - порядка 6 мкс. (Рис.5) Суммарные размеры этих временных потерь от 10 до 100 мкс. **Эти временные потери компенсируются при работе математического аппарата Ядра** аддитивным слагаемым со знаком минус при расчете значения OCR1A.

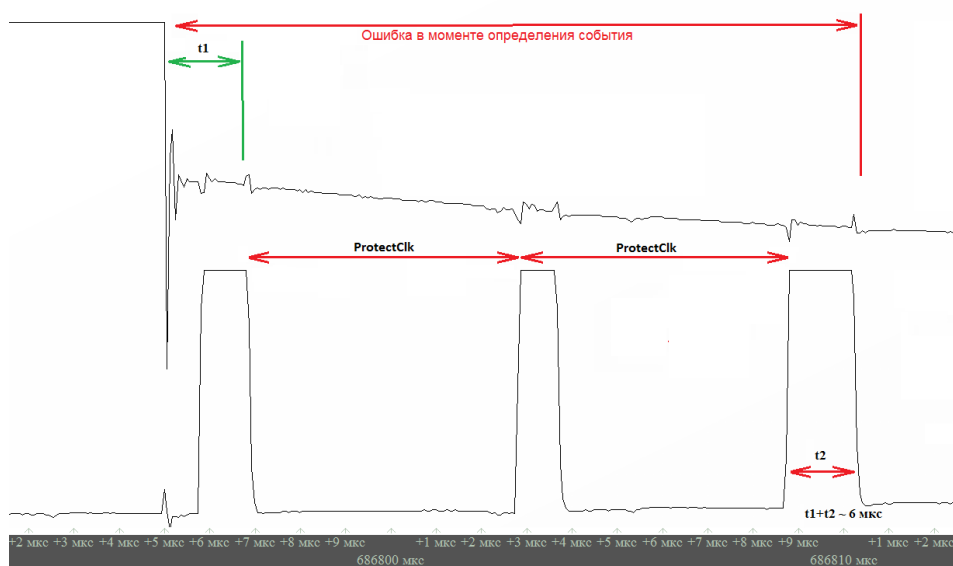
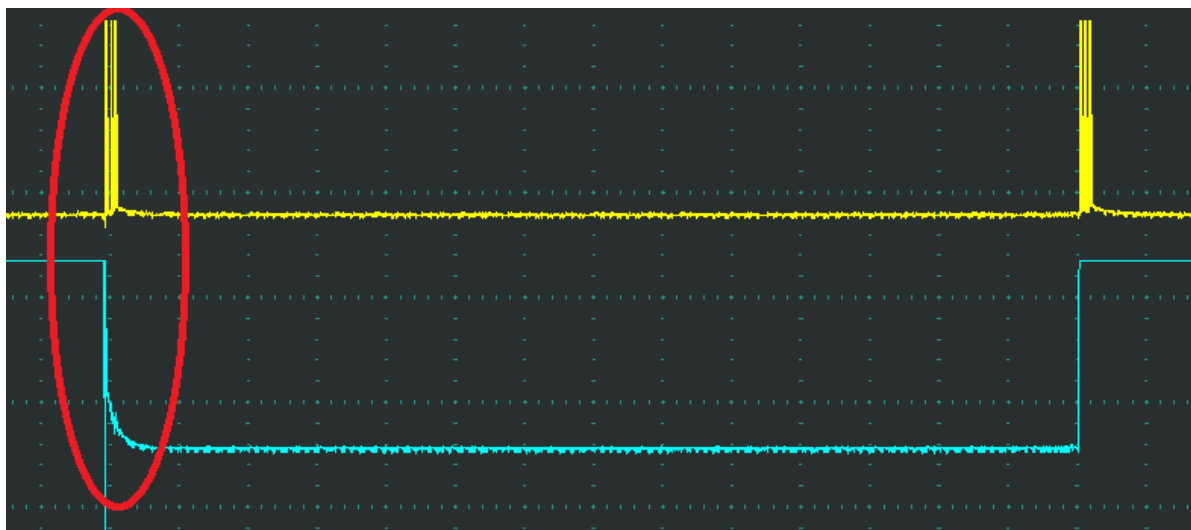


Рис. 5 Иллюстрация временных потерь при повторном чтении входной линии

На рис.6 приведена иллюстрация квазивиртуализации. Приведены три характерных примера. На рис. 6а представлен самый простой и очевидный случай - одноцилиндровый двигатель с одним оптическим датчиком или датчиком Холла (Base=360). Кроме того, ниже приведен пример подпрограммы Actions для этого случая:

Actions:

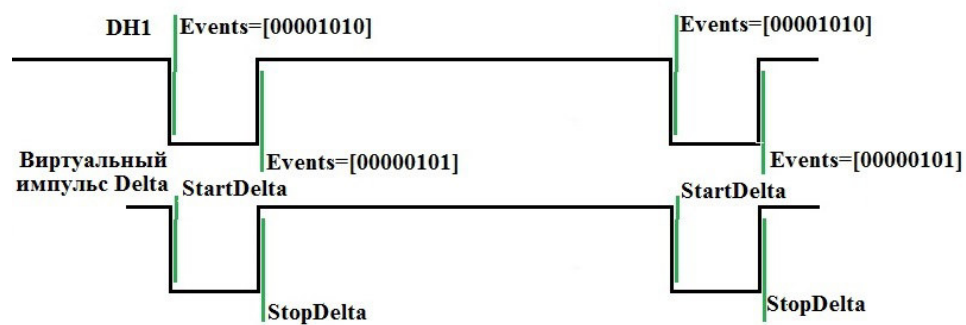
C_Ign: cpi	Events, 0b00001010	- Сравнение регистра Событий с признаком начала метки на DH1
brne	C_VMT	- Если не равны, то проверка следующего кода
StartDelta		- Макрос отметки времени (начало виртуального импульса Delta)
.....		
exit: ret		- Выход из подпрограммы
C_VMT: cpi	Events, 0b00000101	- Сравнение регистра Событий с признаком окончания метки на DH1
brne	exit	- Если не равны, то выход из подпрограммы
StopDelta		- Макрос отметки времени (окончание виртуального импульса Delta)
.....		
ret		- Выход из подпрограммы

На рис. 6 б) представлен другой пример - одноцилиндровый двигатель с индуктивным датчиком (Base=360). Предполагается, что положительная полуволна сигнала индуктивного датчика через формирователь поступает на канал DH1, отрицательная полуволна – на вход DH2. Характерных событий уже не 2, а четыре: начало положительного импульса, завершение положительного импульса, начало отрицательного импульса и момент его завершения. Красной меткой отмечены значения регистра Событий, соответствующих началу и завершению виртуального импульса Delta. Внешний вид подпрограммы Action будет аналогичным предыдущему случаю с учетом иных характерных значений кодов сравнения - вместо пары 00001010/00000101 будет пара 10111110/11101011.

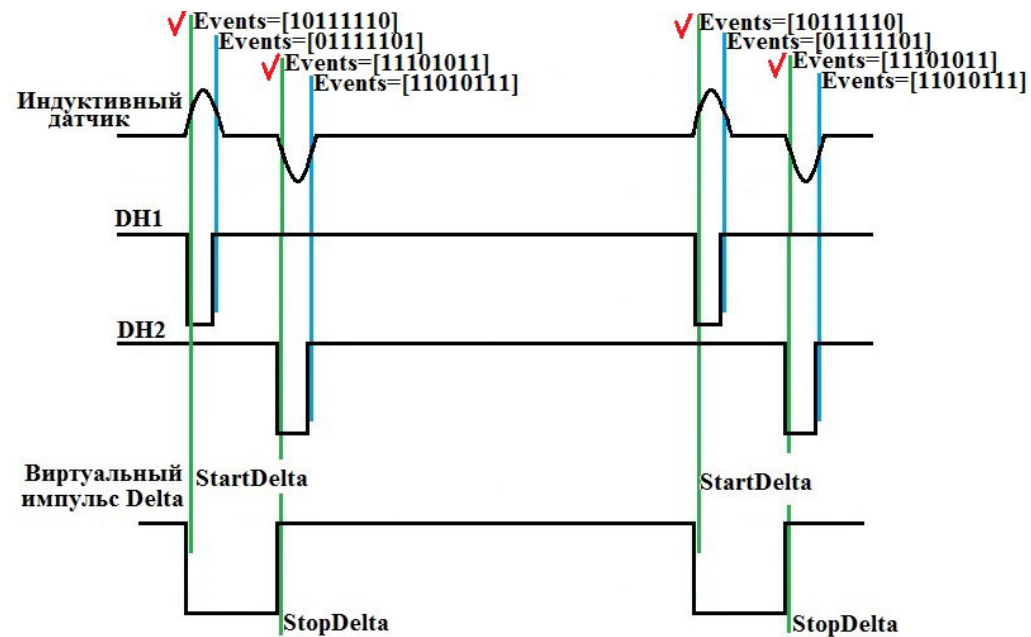
Третий пример более сложный - двухцилиндровый двигатель Ветерок с двумя униполярными датчиками Холла с реакцией от двух магнитных «башмаков» маховика (Base=180). Два других «башмака» маховика имеют другое направления магнитного поля. Виртуальный импульс Delta локализуется при переходе от цилиндра к цилиндру. Искрообразование раздельное благодаря применению дополнительного свободного флага Пользователя, которому придадим статус флага первого цилиндра (Расширенный регистр состояния EStatus, таблица 3 Продолжение).

Actions:

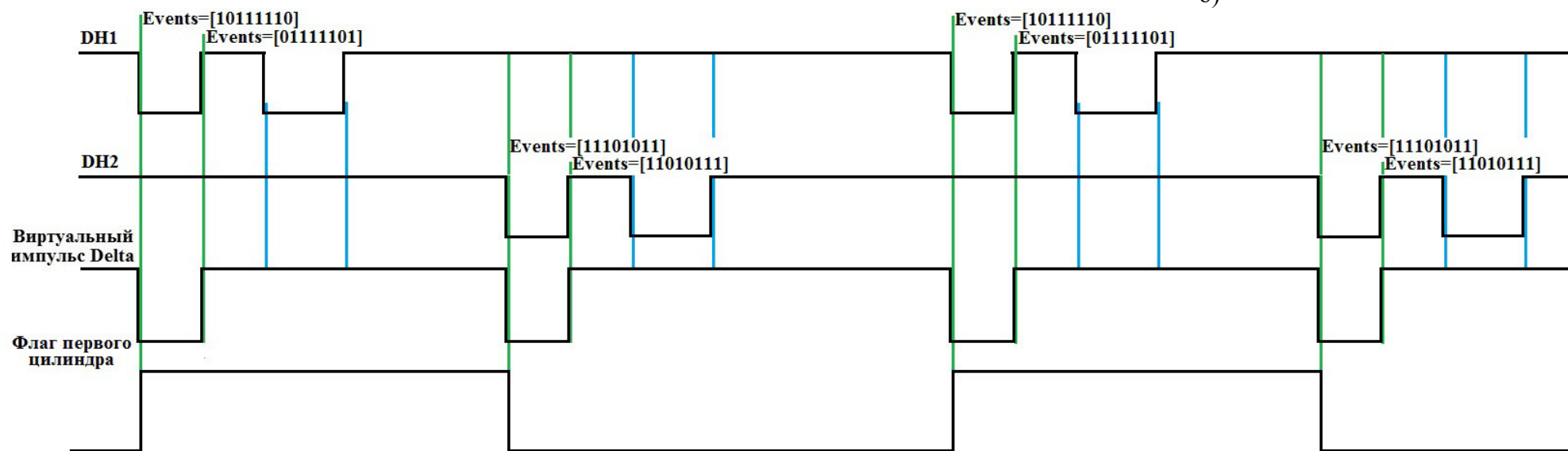
C_Ign1:	cpi	Events, 0b10111110	- Код начала метки Delta на первом цилиндре
	brne	C_Ign2	- Если не равны, то проверка следующего кода
	StartDelta		- Макрос отметки времени (начало виртуального импульса Delta)
	sbr	EStatus,Ign1	- Включить флаг первого цилиндра
			
C_Ign2:	ret		- Выход из подпрограммы
	cpi	Events, 0b11101011	- Код начала метки Delta на втором цилиндре
	brne	C_VMT1	- Если не равны, то проверка следующего кода
	StartDelta		- Макрос отметки времени (начало виртуального импульса Delta)
	cbr	EStatus,Ign1	- Выключить флаг первого цилиндра
C_VMT1:			
	ret		- Выход из подпрограммы
	cpi	Events, 0b01111101	- Код завершения метки Delta на первом цилиндре
	brne	C_VMT2	- Если не равны, то проверка следующего кода
	StopDelta		- Макрос отметки времени (окончание виртуального импульса Delta)
C_VMT2:	sbr	EStatus,Ign1	- Включить флаг первого цилиндра
			
	ret		- Выход из подпрограммы
	cpi	Events, 0b11010111	- Код завершения метки Delta на втором цилиндре
	brne	exit	- Если не равны, то выход
C_VMT2:	StopDelta		- Макрос отметки времени (окончание виртуального импульса Delta)
	cbr	EStatus,Ign1	- Выключить флаг первого цилиндра
			
exit: ret			- Выход из подпрограммы



a)



б)



в)

Рис. 6 Изменение регистра Событий (Events) во времени