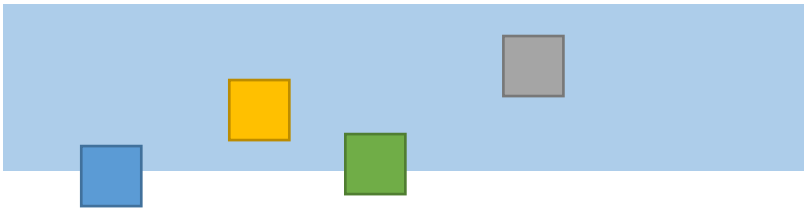

Hardware watchdog v1.0

1. Назначение устройства

Hardware watchdog (HWDG) предназначен для перезагрузки персональных настольных компьютеров стандарта ATX в случае зависания операционной системы.

2. Принцип работы

Под операционной системой запускается клиентское приложение, передающее определённые команды (см. п. 4) на HWDG. После запуска контроля за операционной системой (ОС) HWDG ждёт периодического сигнала от приложения, сигнализирующего о работоспособности ОС. Если периодический сигнал от клиентского приложения не поступает, HWDG подаёт сигнал перезагрузки на «RST» контакт (см. рис. 1). «RST» контакт необходимо соединить с выводом «RST+» на материнской плате.

Если в настройках материнской платы включено дежурное питание USB, то можно использовать режим жёсткой перезагрузки (пятисекундное удержание кнопки PWR с последующим включением). Для этого необходимо активировать соответствующий режим и соединить вывод «PWR» (см. рис. 1) с выводом «PWR+» на материнской плате.

3. Схема HWDG

HWDG представляет собой одностороннюю печатную плату размером 25x18 мм, предназначенную для установки в слот USB на материнской плате компьютера.

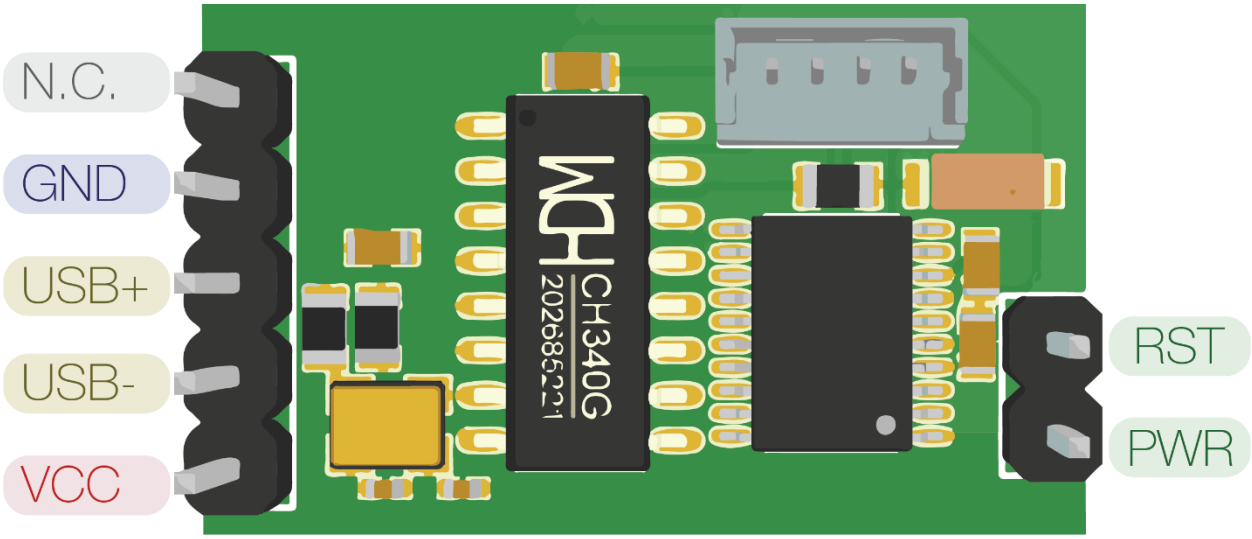

Рисунок 1. Схема hardware watchdog

4. HWDG API

API представляет собой однобайтовые команды, которые принимает HWDG по последовательному порту со скоростью 9600 бод. В ответ на каждую команду посылается статус обработки запрошенной команды.

Таблица 1. Команды API HWDG

Команда	Имя	Ответ	Описание
0b1xxxxxxx	t_h	0x20	Таймаут для жёсткой перезагрузки системы (0-127). Таймаут вычисляется как 10 + t_h * 5 секунд. Пример: t_h=0x33, таймаут равен 10 + 51 * 5 = 265 секунд. По умолчанию после подачи питания на чип таймаут t_h равен 120 секундам.
0b01xxxxxx	t_s	0x21	Таймаут для перезагрузки системы (0-63). Таймаут вычисляется как (t_s + 1) * 1 секунд. Пример: t_s=0x11, таймаут равен (17 + 1) * 1 = 18 секунд. По умолчанию после подачи питания на чип таймаут t_s равен 10 сек.
0b001xxxxx		0x29	Зарезервировано
0b00010xxx	aSft	0x22	Число попыток перезагрузки aSft+1 (0-7)
0b00011xxx	aHrd	0x23	Число попыток жёсткой перезагрузки. Число перезагрузок равно aHrd + 1 (0-7).
0b00001xxx		0x29	Зарезервировано
0b00000001	Start	0x24	Разрешить перезагрузку и запустить монитор состояния ПК. После получения данной команды запускается счётчик времени до перезагрузки. Если счётчик не сброшен командой Stop или Ping, то на пин RST подаётся сигнал к перезагрузке.
0b00000010	Stop	0x25	Выключить монитор ОС.
0b00000011	HRon	0x26	Включить опцию жёсткой перезагрузки при неудачной попытке перезагрузки. Если по прошествии времени t_h компьютер не начинает передавать сигнал Ping, на пин PWR подаётся сигнал к жёсткой перезагрузке.
0b00000100	HRoff	0x27	Отключить опцию жёсткой перезагрузки.
0b00000101	Ping	0x28	Эту команду необходимо передавать не реже чем раз в t_s как сигнал исправности ОС.
0b00000110		0x29	Зарезервировано
0b00000111		0x29	Зарезервировано

5. Реализация софтверной части

Система классов спроектирована таким образом, чтобы обеспечить максимальную унификацию и абстракцию от платформы. В полученной модели (см. рис. 2) файлы, зависящие от платформы выделены жёлтой рамкой. Остальные же файлы универсальны и без изменений могут быть использованы на любой платформе.

В проекте присутствует простенькая событийная модель, позволяющая классам подписываться и отписываться от событий, генерируемых таймерами и блоком UART.

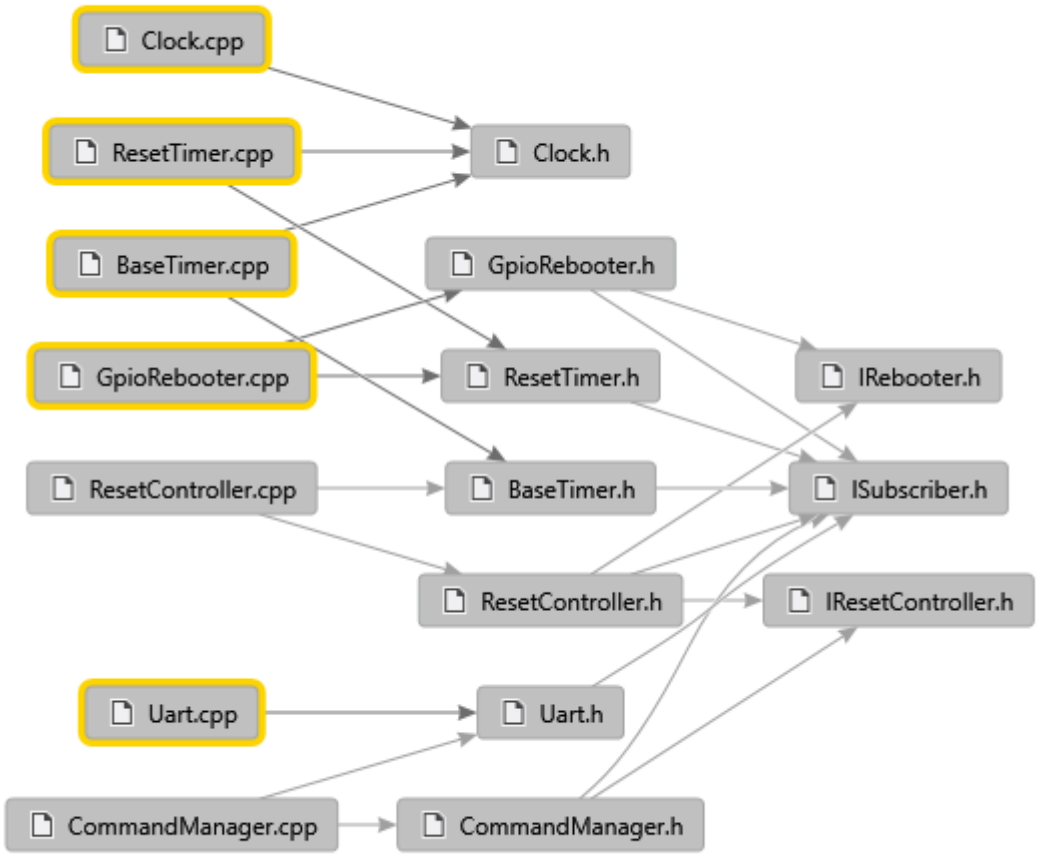

Рисунок 2. Зависимости файлов HWDG

ISubscriber.h – компонент событийной модели. Для того, чтобы объект мог подписываться на события и отписываться от событий, он должен реализовать данный интерфейс.

IRebooter.h – компонент логики перезагрузки компьютера. Содержит всего два метода: HardReset и SoftReset, вызываются они в момент, когда логика управления перезагрузкой решит сделать обычную перезагрузку и жёсткую перезагрузку.

IResetController.h – компонент логики перезагрузки компьютера. Отвечает за реализацию логики команд, поступивших от менеджера команд.

Clock.h – описывает доступные частоты микроконтроллера, позволяет задать и получить значение тактовой частоты микроконтроллера.

ResetTimer.h – описывает таймер, отвечающий за отсчёт таймингов для перезагрузки компьютера.

BaseTimer.h – описывает таймер, отвечающий за формирование прерывания каждую миллисекунду.

Uart.h – предоставляет API для взаимодействия с UART модулем микроконтроллера.

GpioRebooter.h – описывает класс, реализующий интерфейс IRebooter.

ResetController.h – описывает класс, реализующий интерфейс IResetController.

CommandManager.h – описывает класс, который занимается парсингом команд и отдаёт указания контроллеру перезагрузки (IResetController).

Clock.cpp – реализация управления тактированием микроконтроллера.

ResetTimer.cpp – реализация управления таймером, отвечающим за отсчёт таймингов для перезагрузки компьютера.

BaseTimer.h – реализация управления таймером, отвечающим за формирование прерывания каждую миллисекунду.

Uart.cpp – реализация управления UART модулем микроконтроллера.

GpioRebooter.cpp – реализация логики управления GPIO для корректной перезагрузки компьютера.

ResetController.cpp – реализация логики перезагрузки компьютера.

CommandManager.cpp – предоставляет внешний API HWDG.

Диаграмма зависимостей классов приведена на рис. 3. Розовая стрелка показывает, что класс от которого направлена стрелка использует класс, к

которому направлена стрелка. Зелёная стрелка показывает, что объект от которого направлена стрелка реализует интерфейс, к которому эта стрелка направлена. Например, класс ResetController использует интерфейс IRebooter и класс Timer, в то же время ResetController реализует интерфейсы ISubscriber и IResetController.

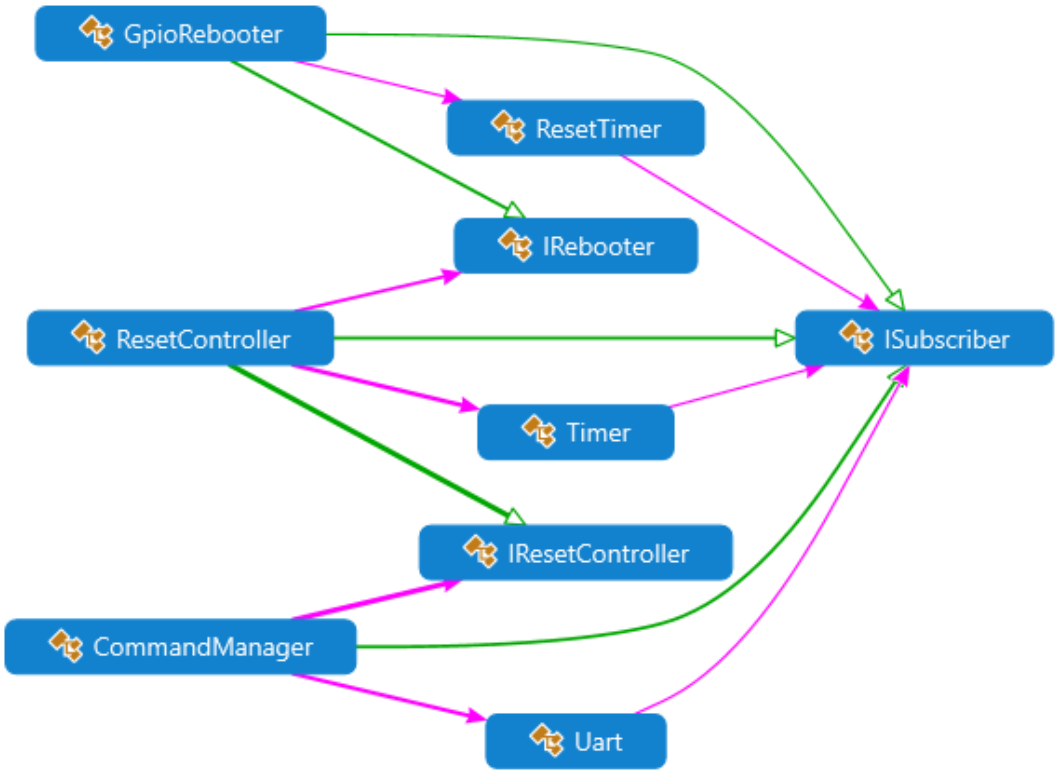

Рисунок 3. Диаграмма зависимостей классов HWDG

6. Конечный автомат перезагрузки

После подачи питания на микроконтроллер конечный автомат (RSM) переходит в состояние Idle (см. рис. 4). После получения команды на старт мониторинга RSM переходит в состояние ожидания команды Ping. Если по истечении t_s команда Ping не получена, RSM уменьшает число попыток перезагрузки, перезагружает компьютер и переходит в состояние ожидания загрузки ОС. После перезагрузки ОС клиентское приложение должно вывести HWDG в состояние Idle. Если по истечении t_h клиентское приложение не запущено, это означает что загрузка ОС прошла неудачно. Если количество попыток перезагрузки не исчерпано, RSM будет делать перезапуск системы до

тех пор, пока счётчик попыток не снизится до ноля. Далее, если режим жёсткой перезагрузки не активирован, RSM переходит в состояние Idle. Если же режим жёсткой перезагрузки активирован, RSM произведёт жёсткий перезапуск системы и перейдёт в режим ожидания загрузки ОС. Если по истечении t_h клиентское приложение не запущено, это означает что жёсткая перезагрузка прошла неудачно. Если количество попыток жёсткой перезагрузки не исчерпано, RSM будет делать жёсткий перезапуск системы до тех пор, пока счётчик попыток не снизится до ноля. Если счётчик попыток жёсткой перезагрузки стал равен нолю, RSM переходит в режим Idle.

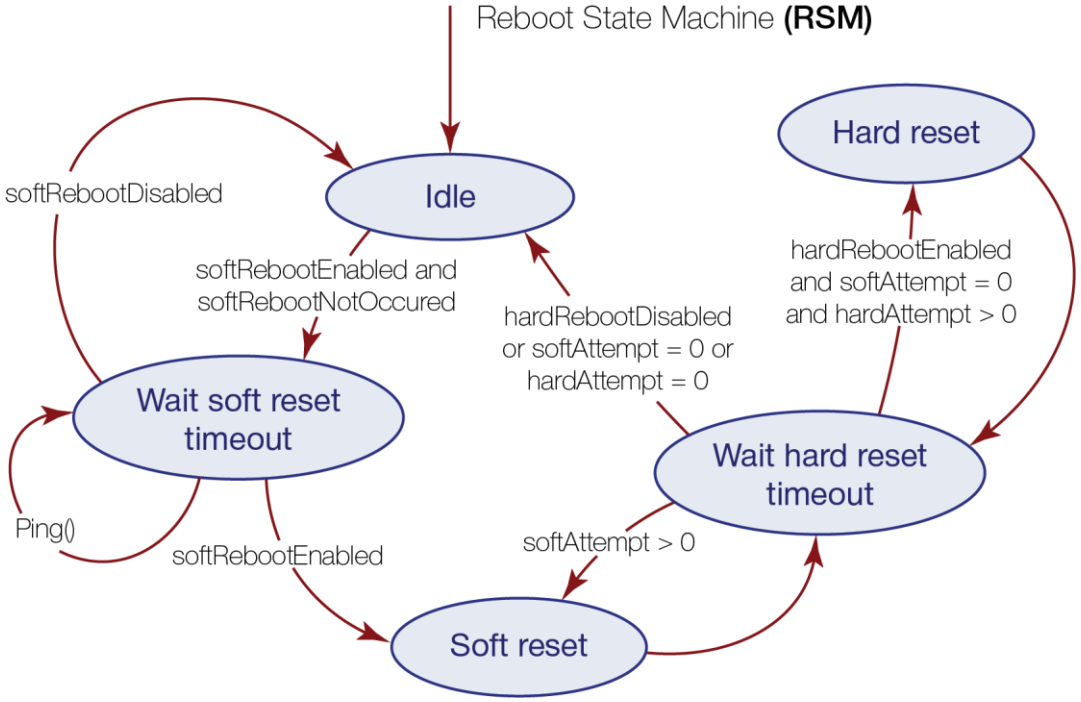

Рисунок 4. Конечный автомат перезагрузки

7. Тайминги перезагрузки

Материнская плата подтягивает выводы «PWR+» и «RST+» к дежурному питанию +5в. Если сорвать подтяжку и посадить «PWR+» или «RST+» на землю, происходит запуск компьютера и перезапуск компьютера соответственно. Для принудительного выключения компьютера необходимо держать «PWR+» прижатым к земле больше 5 секунд.

Для осуществления перезагрузки необходимо на вывод «RST+» на материнской плате подать импульс, подтягивающий «RST+» к земле. В реализации HWDG длина импульса обозначена как RST_TIM и равняется приблизительно 200 миллисекунд (см. рис. 5).

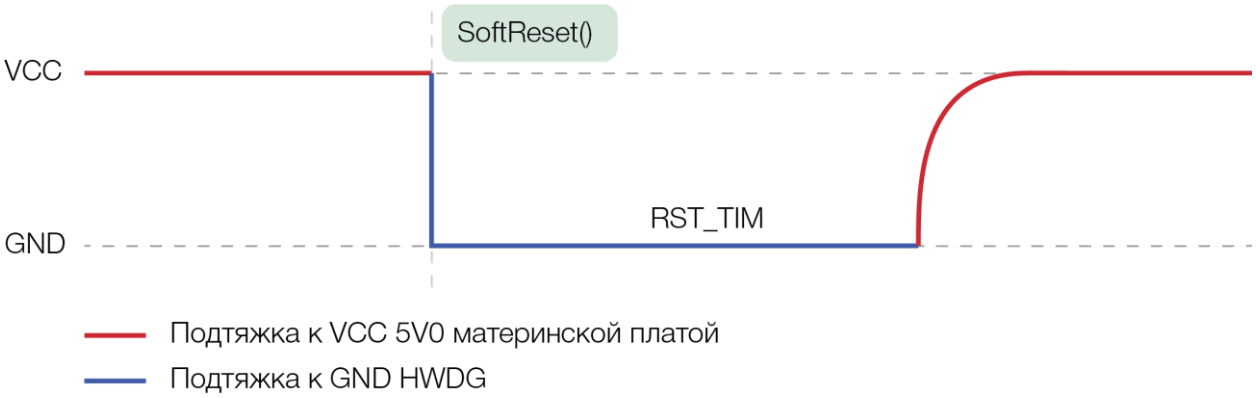

Рисунок 5. Последовательность перезагрузки

Для осуществления жёсткой перезагрузки (см. рис. 6) необходимо сначала принудительно выключить компьютер, подав низкий сигнал на вывод «PWR+» длиной HR_LO_TIM более 5 секунд. Затем материнская плата должна восстановить высокий уровень на выводе «PWR+». Для этого HWDG ждёт HR_HI_TIM (около 2 секунд) и подаёт импульс на «PWR+» для включения компьютера длиной RST_TIM.

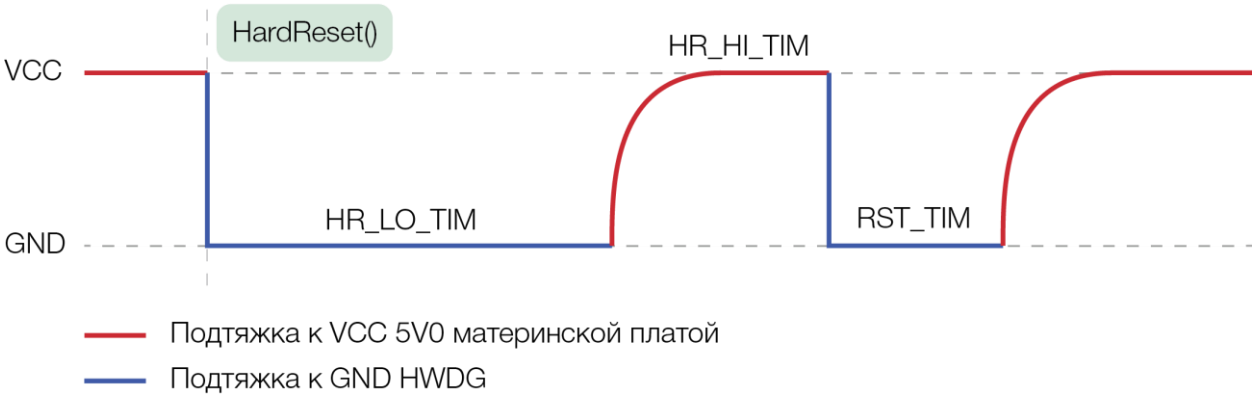

Рисунок 6. Последовательность жёсткой перезагрузки