Современные 32-разрядные ARM-микроконтроллеры серии STM32: сторожевые таймеры

Олег Вальпа

В статье приведено описание сторожевых таймеров 32-разрядных ARM-микроконтроллеров серии STM32 от компании STMicroelectronics. Рассмотрена архитектура и состав регистров сторожевых таймеров, а также приведены практические примеры программ.

ВВЕДЕНИЕ

Все микропроцессорные устройства подвержены влиянию различных помех, которые могут нарушить их работу и даже привести к «зависанию» программы. Это в свою очередь приводит к останову системы управления и может повлечь за собой крайне опасные ситуации.

Источниками помех могут являться различные коммутационные устройства в питающей сети, воздействие электромагнитных полей, атмосферные явления в виде разрядов молний и тому подобное. Защититься от подобных помех не всегда возможно, поэтому в случае «зависания» процессора необходимо автоматически восстановить его работу, например, путем перезапуска. Это позволит восстановить работу системы и обеспечит высокий уровень безопасности систем.

В свое время разработчики процессоров решили эту задачу с помощью сторожевых таймеров, получивших название «сторожевая собака». Идея решения заключается в том, что сторожевой таймер инициализируется и запускается самим процессором в начале работы программы. После чего данный таймер самостоятельно отсчитывает заданное время, и если в течение этого времени процессор ни разу не перезагрузит его, то таймер обнулится и перезапустит процессор. Таким образом, при нормальной работе процессора таймер будет регулярно перезапускаться, а в случае «зависания» процессора таймер через заданное время сформирует сигнал сброса для процессора и перезапустит его.

Микроконтроллеры серии STM32 [1] имеют два таких таймера. Один из них называется «независимый сторожевой таймер» (Independent Watchdog, IWDG), а другой — «оконный сторожевой таймер» (Window Watchdog, WWDG). Рассмотрим каждый из них отдельно.

НЕЗАВИСИМЫЙ СТОРОЖЕВОЙ ТАЙМЕР

Таймер IWDG тактируется специализированным низкочастотным сигналом LSI, благодаря чему он продолжает работу, даже если пропадает системный тактовый сигнал. Таймер осуществляет обратный отсчет от заданного значения до нуля. Сброс процессора таймер производит при достижении счетчиком таймера нуля.

Этот таймер лучше всего подходит для тех приложений, которым необходимо, чтобы сторожевой таймер был запущен как полностью независимый процесс, вне основного приложения, но у которого были бы не слишком высокие требования к временны́м параметрам. Функциональная схема независимого сторожевого таймера IWDG приведена на рисунке 1.

Таймер IWDG запускается путем записи значения 0xCCCC в ключевой регистр IWDG_KR. Счетчик таймера начинает обратный отсчет от значения, заданного в регистре перезагрузки IWDG_RLR. По умолчанию этот регистр имеет значение 0xFFF. Когда счетчик досчитает до нуля, формируется сигнал сброса для процессора от таймера IWDG.

Каждый раз, когда в ключевой регистр IWDG_KR записывается значение

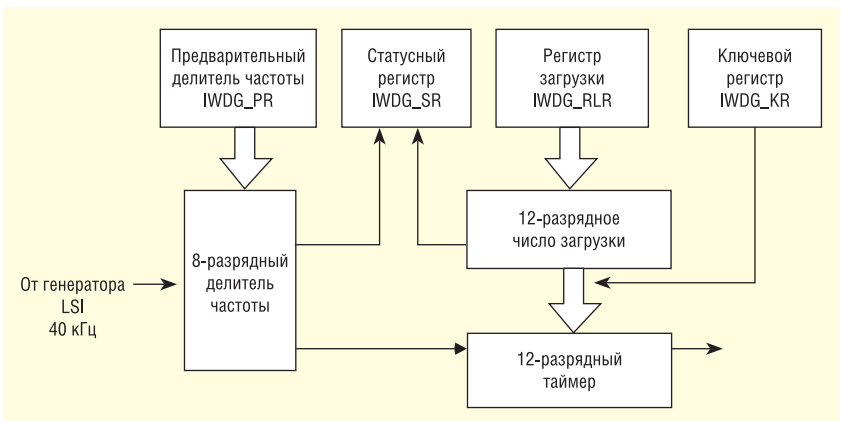

Рис. 1. Функциональная схема сторожевого таймера IWDG

Таблица 1. Минимальное и максимальное значения периода сторожевого таймера

Предварительный делитель	Значение регистра IWDG_ PR[2:0], биты	Минимальное значение для RL[11:0]=0x000, мс	Максимальное значение для RL[11:0]=0xFFF, мс
4	0	0.1	409.6
8	1	0.2	819.2
16	2	0.4	1638.4
32	3	0.8	3276.8
64	4	1.6	6553.6
128	5	3.2	13107.2
256	6 или 7	6.4	26214.4

0xAAAA, данные из регистра IWDG_RLR перегружаются в счетчик, что возобновляет отсчет и предотвращает сброс процессора от IWDG.

Если с помощью соответствующих битов конфигурации разрешена функция «аппаратный сторожевой таймер», то сторожевой таймер автоматически запускается после подачи питания.

Регистр предварительного делителя IWDG_PR позволяет программно задать коэффициент деления тактовой частоты для сторожевого таймера IWDG.

По умолчанию регистры IWDG_PR и IWDG_RLR защищены от записи. Прежде чем изменить их значения, необходимо записать код 0x5555 в регистр IWDG_KR. Запись в ключевой регистр любого другого значения снова заблокирует запись в регистры IWDG_PR и IWDG_RLR.

Период таймера вычисляется при помощи формулы:

```
T=((4*2^IWDG_PR)* IWDG_RLR)/40 [мс].
```

Определить период отсчета сторожевого таймера для тактовой частоты 40 кГц можно также с помощью таблицы 1.

В связи с нестабильностью внутреннего RC генератора микроконтроллера, указанные в таблице значения времени могут незначительно отличаться от реальных.

Регистр статуса IWDG_SR позволяет определить, какие операции выполняет таймер в текущий момент.

Когда микроконтроллер входит в режим отладки и его ядро останавливается, счетчик IWDG либо продолжает работать, либо останавливается, в зависимости от конфигурационного бита DBG_IWDG_STOP в модуле DBG.

Регистры независимого сторожевого таймера

Карта регистров сторожевого таймера IWDG приведена в таблице 2.

Ключевой регистр IWDG_KR имеет 16 бит для ключа KEY[15:0]. В эти биты программа должна регулярно записывать ключевое значение 0xAAAA, иначе сторожевой таймер произведет сброс, когда счетчик достигнет значения 0. Запись ключевого значения 0x5555 в этот регистр разрешает доступ к регистрам IWDG_PR и IWDG_RLR. Запись ключевого значения 0xCCCC запускает сторожевой таймер, кроме случая, когда он уже запущен в режиме «аппаратный сторожевой таймер».

Регистр предварительного делителя IWDG_PR содержит три бита PR[2:0], которые задают коэффициент деления частоты сторожевого таймера. Изменить значение IWDG_PR можно только тогда, когда бит PVU регистра IWDG_SR сброшен. Соответствие между значением бит PR[2:0] регистра IWDG_PR и коэффициент деления частоты сторожевого таймера приведены в таблице 1.

При чтении этого регистра его значение может быть некорректным, если операция чтения производится тогда, когда предыдущая операция записи в этот регистр еще не завершилась. По этой причине значение, прочитанное в этом регистре, действительно только тогда, когда бит PVU в регистре IWDG_SR сброшен.

Регистр перезагрузки IWDG_RLR имеет 12 бит RL[11:0], которые позволяют записать в него значение от 0 до 0xFFF. Это значение будет перезагружаться в счетчик сторожевого таймера каждый раз при записи значения 0xAAAA в регистр ключа IWDG_KR. Период таймера вычисляется на основе этого значения и тактовой частоты после предварительного делителя. Для того чтобы изменить или прочитать значение регистра перезагрузки, необходимо убедиться, что бит RVU регистра IWDG_SR сброшен.

Регистр статуса IWDG_SR имеет два информационных бита.

Бит 1 RVU устанавливается аппаратно и указывает, что идет процесс обновления значения перезагрузки. Очищается он аппаратно, когда операция обновления значения перезагрузки завершена. Данная операция занимает до 5 циклов RC генератора на частоте 40 кГц. Значение регистра перезагрузки можно обновлять только при сброшенном бите RVU.

Бит 0 PVU устанавливается аппаратно и указывает, что идет процесс обновления значения предварительного делителя. Очищается он аппаратно, когда операция обновления значения предварительного делителя закончена. Данная операция также занимает до 5 циклов RC генератора на частоте 40 кГц. Значение предварительного делителя можно обновлять только при сброшенном бите PVU.

Если приложение использует несколько значений перезагрузки или предварительного делителя, то, прежде чем изменить значение перезагрузки, необходимо ждать, пока не будет сброшен бит RVU. Также, прежде чем изменить значение предварительного делителя, следует ждать, пока не будет сброшен бит PVU.

Таблица 2. Карта регистров IWDG и их значения после сброса

Сдвиг	Регистр	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	IWDG_KR	Резерв																KEY[15:0]															
	Исх. значение																	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0x04	IWDG_PR	Резерв																													PR[2:0]		
	Исх. значение																														0	0	0
0x08	IWDG_RLR	Резерв																	RL[11:0]														
	Исх. значение																		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
0x0C	IWDG_SR	Резерв																														RVU	PVU
	Исх. значени																															0	0

Листинг 1

```
// Функция инициализации сторожевого таймера IWDG
void Init_IWDG(u16 tw) // Параметр tw от 7мс до 26200мс
{
  // Для IWDG_PR=7 Tmin=6,4мс RLR=Tмс*40/256
  IWDG->KR=0x5555; // Ключ для доступа к таймеру
  IWDG->PR=7; // Обновление IWDG_PR
  IWDG->RLR=tw*40/256; // Загрузить регистр перезагрузки
  IWDG->KR=0xAAAA; // Перезагрузка
  IWDG->KR=0xCCCC; // Пуск таймера
}

// Функция перезагрузки сторожевого таймера IWDG
void IWDG_res(void)
{
  IWDG->KR=0xAAAA; // Перезагрузка
}
```

Однако, после обновления значения предварительного делителя или перезагрузки, необходимости анализировать состояние бит RVU и PVU, прежде чем продолжить выполнение кода, нет. Операция записи будет принята к исполнению и завершена даже в случае перехода в режим низкого потребления.

Примеры программ для независимого сторожевого таймера

В листинге 1 приведен пример функций для работы с таймером IWDG. Комментарии позволяют понять назначение всех операций, выполняемых в этой программе. Используя данные функции, задействовать в программе сторожевой таймер IWDG и осуществлять управление им не составит труда.

ОКОННЫЙ СТОРОЖЕВОЙ ТАЙМЕР

Таймер WWDG лучше всего подходит для тех приложений, которые требуют, чтобы сторожевой таймер реагировал на перезапуск в пределах точного промежутка времени, то есть окна времени.

Таймер тактируется сигналом, полученным делением сигнала тактовой частоты PCLK1 с шины APB1. Он имеет конфигурируемое окно времени, которое служит для перезагрузки сторожевого таймера, и может быть задан так, чтобы обнаруживать неправильное поведение программы, в виде запаздывания или опережения в работе.

Обычно этот сторожевой таймер используется для того, чтобы обнаруживать возникновение некорректной работы программного обеспечения, вызванное внешним вмешательством или непредвиденными логическими условиями, которые заставляют прикладную программу отклоняться от своего нормального алгоритма.

Функциональная схема оконного сторожевого таймера WWDG приведена на рисунке 2. Фактически оконный сторожевой таймер является расширенной версией традиционного встраиваемого сторожевого таймера.

Оконный сторожевой таймер представляет собой 7-битный вычитающий счетчик, тактируемый сигналом PCLK1, поделенным на 4096 с помощью аппаратного 12-битного предварительного делителя и программно управляемого делителя от 1 до 8.

Оконный сторожевой таймер активируется путем установки бита WDGA в регистре WWDG_CR. С этого момента его нельзя отключить, за исключением общего сброса. После активизации сторожевой таймер начинает счет в обратном направлении и генерирует сброс при изменении состояния счетчика с 0x40 на 0x3F, то есть при обнулении шестого разряда таймера, условно обозначаемого как T6. Если программа перегружает счетчик, когда его значение больше, чем значение в регистре окна времени, это также инициирует сброс.

Чтобы предотвратить сброс процессора, прикладная программа, во время нормальной операции, должна регулярно производить запись в регистр WWDG_CR. Эта операция должна производиться только тогда, когда значение счетчика меньше, чем значение в регистре окна времени. Значение, которое можно записать в регистр WWDG_CR, должно быть в пределах 0xFF...0x40.

Счетчик таймера WWDG является автономным. Он всегда производит обратный отсчет. Даже тогда, когда сторожевой таймер отключен. При включении сторожевого таймера необходимо установить бит T6 счетчика, чтобы предотвратить немедленный сброс процессора. Биты T[5:0] содержат число, которое представляет собой временну́ю задержку до сброса от WWDG.

Регистр конфигурации WWDG_CFR содержит верхний предел окна времени. Чтобы предотвратить сброс, обратный счетчик должен быть перезагружен тогда, когда его значение меньше, чем значение регистра окна времени, но больше 0x3F.

На рисунке 3 представлена временна́я диаграмма, демонстрирующая оконный процесс работы сторожевого таймера.

Другой способ перезагрузить счетчик состоит в том, чтобы ис-

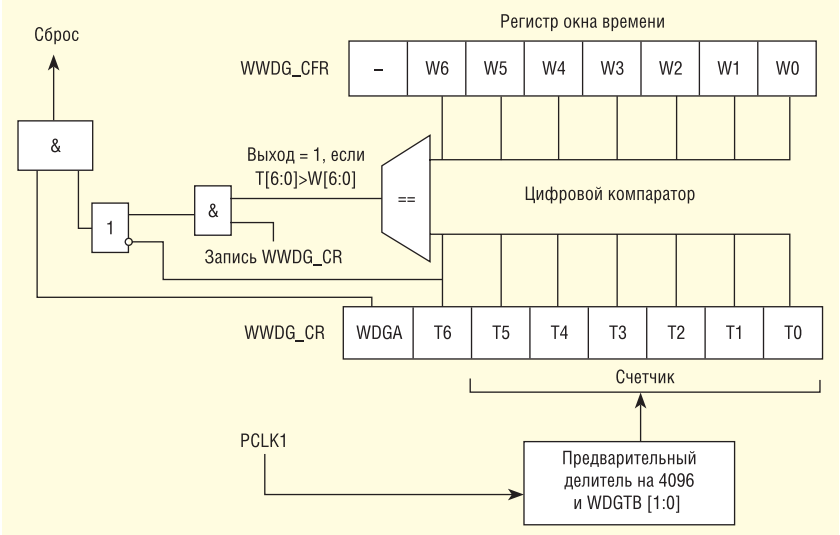

Рис. 2. Функциональная схема сторожевого таймера WWDG

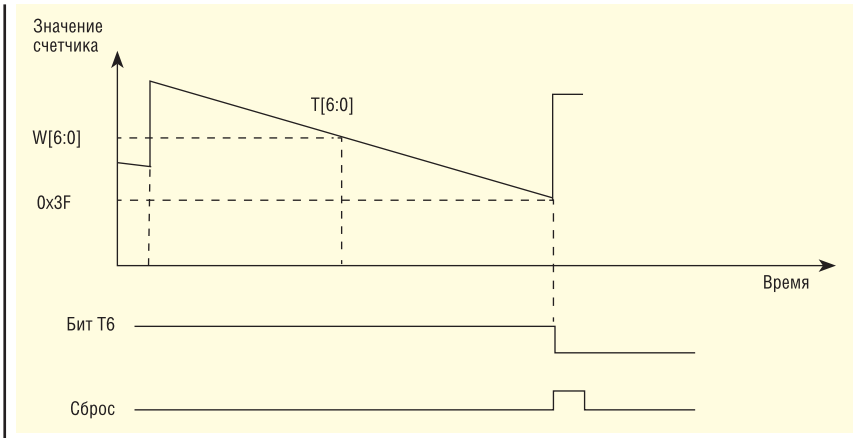

Рис. 3. **Временна́я диаграмма сторожевого таймера WWDG**

Таблица 3. Диапазон значения периода таймера WWDG

Предварительный делитель	Значение WDGTB[1:0], биты	Минимальный период для T[5:0]=0, мкс	Максимальный период для T[5:0]=0x3F, мкс
1	0	170	10920
2	1	341	21850
4	2	682	43690
8	3	1365	87380

пользовать прерывание от раннего пробуждения. Это прерывание разрешается установкой бита EWI в регистре WWDG_CFR. Когда обратный счетчик достигает значения 0x40, генерируется прерывание от раннего пробуждения, и можно использовать соответствующий обработчик, чтобы перезагрузить счетчик и предотвратить сброс от WWDG.

Запрос этого прерывания очищается записью нуля в разряд EWIF регистра WWDG_SR.

При необходимости можно сгенерировать программный сброс процессора путем установки в единичное состояние бита WDGA и сбросив бит T6.

В таблице 3 приведен диапазон значений периода таймера WWDG для тактовой частоты сигнала PCLK равной 24 МГц, в зависимости от установленных значений регистров таймера.

Для более точного расчета периода необходимо использовать следующую формулу:

```
Twwdg = Tpclk1 * 4096 * 2^WDGTB
* (T[5:0]+1).
```

Чтобы избежать немедленного сброса процессора, при записи данных в регистр WWDG_CR необходимо устанавливать бит T6 в единичное состояние.

Когда микроконтроллер входит в режим отладки, счетчик WWDG будет продолжать работать или останавливается, в зависимости от конфигурационного бита DBG_WWDG_STOP в модуле DBG.

Таблица 4. Карта регистров WWDG и их значения после сброса

Сдвиг	Регистр	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x00	WWDG_CR	Резерв																								WDGA	T[6:0]						
	Исх. значение																									0	1	1	1	1	1	1	1
0x0	WWDG_CFR	Резерв																						EWI	WDGTB1	WDGTB0	W[6:0]						
	Исх. значение																							0	0	0	1	1	1	1	1	1	1
0x0	WWDG_SR	Резерв																															EWIF
	Исх. значение																																0

Листинг 2

```
// Функция инициализации сторожевого таймера WWDG
void WWDG_Init(void)
{
// Инициализация прерывания
NVIC->ISER[0] |= NVIC_ISER_SETENA_0;
// Инициализация таймера WWDG
RCC->APB1ENR |= RCC_APB1ENR_WWDGEN; // Включить тактирование WWDG
WWDG->CFR |= WWDG_CFR_WDGTB0; // Задать частоту тактирования WWDG
WWDG->CFR |= 0x41; // Задать окно
WWDG->CR |= 0x7F; // Разрешить работу WWDG и установить начальное
значение
WWDG->SR &=~WWDG_SR_EWIF; // Очистить флаг EWI
WWDG->CFR |=WWDG_CFR_EWI; // Разрешить прерывания
}

// Функция обработки прерывания сторожевого таймера WWDG
void WWDG_IRQHandler(void)
{
WWDG->CR |= 0x7F; // Обновить значение таймера WWDG
WWDG->SR &=~WWDG_SR_EWIF; // Очистить флаг EWI
// Другие действия …
}

// Функция запуска сторожевого таймера WWDG
void WWDT_ON(void)
{
RCC->APB1ENR |= RCC_APB1ENR_WWDGEN; // Включить тактирование WWDG
WWDG->CR |= 0x7F; // Обновить значение таймера WWDG
WWDG->SR &=~WWDG_SR_EWIF; // Очистить флаг EWI
}

// Функция останова сторожевого таймера WWDG
void WWDT_      OFF(void)
{
RCC->APB1ENR |= RCC_APB1ENR_WWDGRST; // Отключить тактирование WWDG
}
```

Регистры оконного сторожевого таймера

Сторожевой таймер WWDG имеет три регистра. Карта этих регистров приведена в таблице 4.

Регистр управления WWDG_CR содержит бит WDGA для активации таймера и 7-разрядный счетчик T[6:0]. Бит WDGA устанавливается программно, а очищается аппаратно, только путем общего сброса. Когда WDGA равен 1, сторожевой таймер включен и может производить сброс. С появлением каждого тактового импульса счетчик T[6:0] уменьшает свое значение на единицу. Сброс генерируется, когда счетчик меняет значение с 0x40 на 0x3F, то есть при обнулении разряда T6.

Регистр конфигурации WWDG_CFR содержит 3 бита для настройки таймера и 7 разрядов W[6:0] для установки окна времени.

Разряд 9 EWI служит для разрешения прерывания раннего пробуждения процессора. Если он установлен, то при достижении счетчиком значения 0x40 генерируется прерывание. Обнуляется этот бит аппаратно, путем общего сброса микроконтроллера.

Биты 8 и 7 группы WDGTB[1:0] назначают коэффициент дополнительного деления частоты входного сигнала счетчика. Они могут иметь следующие значения:

- 00 — деление на 1;
- 01 — деление на 2;
- 10 — деление на 4;
- 11 — деление на 8.

Разряды W[6:0] организуют 7-битное значение окна времени, с которым сравнивается значение обратного счетчика.

Регистр статуса WWDG_SR содержит всего один бит статуса EWIF, представляющий собой флаг прерывания пробуждения процессора. Этот бит принимает единичное значение аппаратно, когда счетчик достигает значения 0x40 и тогда, когда прерывание выключено.

Очищается он программно, записью в него нулевого значения. Запись в этот разряд значения 1 эффекта не имеет.

Примеры программ для оконного сторожевого таймера

В листинге 2 приведены функции для работы с таймером WWDG с комментариями, позволяющими понять назначение всех операций, выполняемых в этой программе. С помощью таких несложных функций можно организовать контроль работы программы, обеспечивая тем самым защиту от сбоев.

Подробнее ознакомиться с таймерами микроконтроллера STM32 можно на сайте STMicroelectronics [2].

Литература:

1. www.st.com
2. www.st.com/web/en/resource/technical/document/reference_manual/CD00246267.pdf

CNY

* Статья перепечатана из журнала «Современная электроника», № 5, 2014 г., с разрешения редакции, тел. +7 (495) 232-00-87, www.soel.ru