Wykład 8

Układy odmierzania czasu

Źródła sygnału zegarowego

- W mikrokontrolerze LM3S6965 są cztery źródła sygnałów zegarowych możliwe do wykorzystania:
  - oscylator wewnętrzny (Internal oscillator – IOSC),
  - oscylator główny, (Main Oscillator),
  - oscylator wewnętrzny 30kHz,
  - zewnętrzny oscylator czasu rzeczywistego.

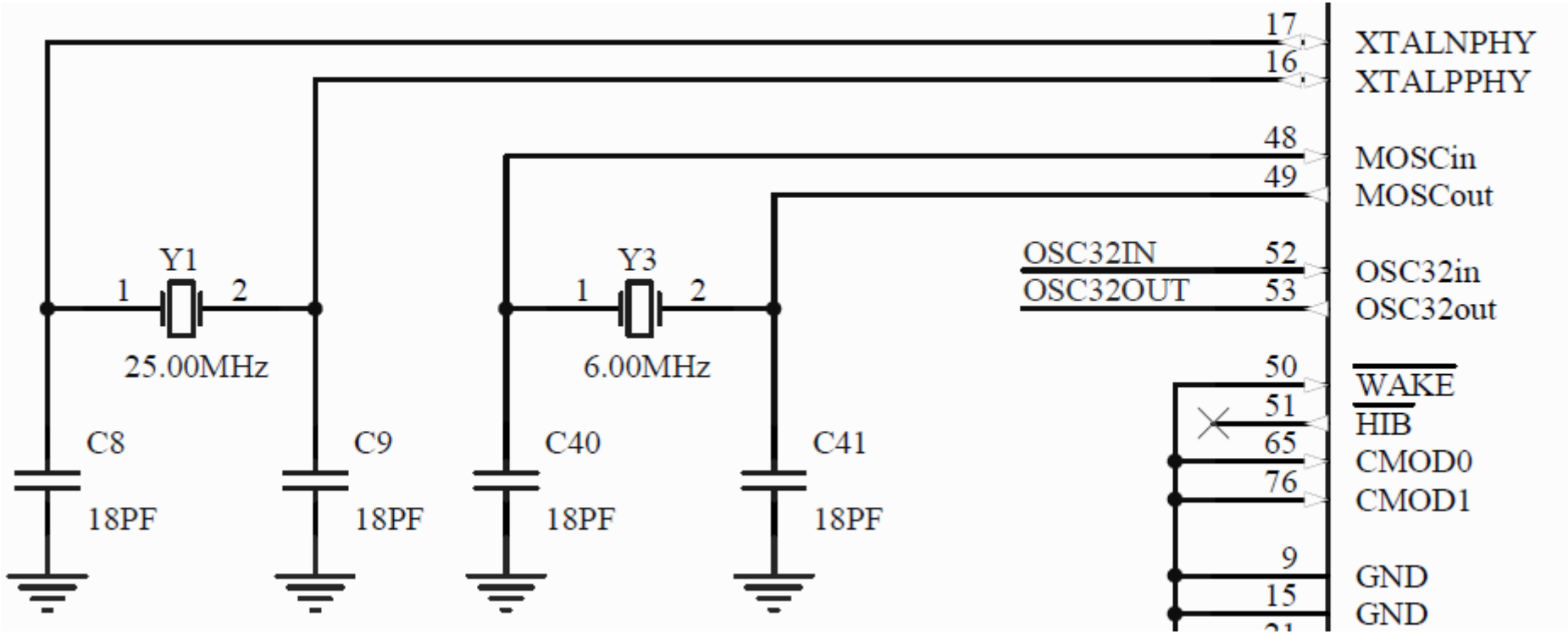
Oscylator wewnętrzny (Internal oscillator, IOSC)

- Oscylator wewnętrzny jest źródłem sygnału zegarowego wbudowanym w układ scalony.
- Nie wymaga on żadnych zewnętrznych komponentów. Częstotliwość oscylatora wynosi 12 MHz ± 30%.
- Aplikacje które nie wymagają dokładnego odmierzania czasu, mogą wykorzystać ten oscylator w celu redukcji kosztów systemu.
- Oscylator wewnętrzny jest jest domyślnie uaktywniany po wystąpieniu sygnału zerującego (reset).
- Jeżeli wymagane jest użycie innego oscylatora należy go uaktywnić po resecie i pozwolić na ustabilizowanie jego częstotliwości przed przełączeniem źródła sygnału taktującego.

Oscylator główny

- Oscylator główny pozwala na uzyskanie dokładnego sygnału taktującego mikrokontroler.
- Pozwala on na podłączenie zewnętrznego sygnału taktującego do wejścia OSC0, lub zewnętrznego kwarcu podłączonego pomiędzy wejścia MOSC0 i MOSC1.
- Zakres możliwych częstotliwości kwarcu zależy od tego, czy oscylator jest wykorzystywany jako źródło sygnału dla pętli PLL. Jeżeli tak, to zakres częstotliwości zawiera się pomiędzy 3.579545 MHz, a 8.192 MHz.
- Jeżeli pętla PLL nie jest wykorzystywana zakres ten zawiera się pomiędzy 1 MHz, a 8.192 MHz.
- Częstotliwość kwarcu jest ustawiana w bitach XTAL w rejestrze RCC.

Schemat połączeń oscylatora



Oscylator wewnętrzny 30 kHz

- Wewnętrzny oscylator o częstotliwości 30 kHz ± 30% jest wykorzystywany w stanie głębokiego uśpienia (Deep-Sleep), w trybach oszczędzania energii.
- Wykorzystanie tego oscylatora pozwala na wyłączenie oscylatora głównego i zmniejszenie ilości wewnętrznych przełączeń.

Zewnętrzny oscylator czasu rzeczywistego

- Zewnętrzny oscylator czasu rzeczywistego (External Real-Time Oscillator) udostępnia dokładny sygnał zegarowy o niskiej częstotliwości wynoszącej 32.768kHz.
- Jest on przewidziany do wykorzystania w systemach wymagających precyzyjnego odmierzania czasu rzeczywistego.
- Sygnał ten można uzyskać z zewnętrznego oscylatora o wymaganej częstotliwości, podłączanego do wejścia XOSC0 lub kwarcu o częstotliwości 4.194304 MHz.
- Kwarc podłączany jest pomiędzy wejścia XOSC0 i XOSC1. Częstotliwość z kwarcu jest dzielona przez 128, w celu uzyskania wymaganej częstotliwości.
- Oscylator ten może także być źródłem sygnałów taktujących dla trybów oszczędzania energii jest on częścią modułu hibernacji (Hibernation Module).

Wewnętrzny sygnał zegarowy

- Wewnętrzny sygnał zegarowy (sysclk) może być pobierany z wyjścia jednego z czterech oscylatorów lub z wyjścia pętli PLL albo wewnętrznego oscylatora (30 kHz) podzielonego przez cztery.
- Do ustawienia żądanego źródła sygnałów i jego parametrów wykorzystywane sa rejestry RCC (Run-Mode Clock Configuration) i RCC2 (Run-Mode Clock Configuration 2). Rejestr RCC2 jest poszerzoną wersją rejestru RCC i posiada więcej opcji. Nadpisuje ustawienia z rejestru RCC.

Ustawienia rejestru RCC

- Możliwe ustawienia w rejestrze RCC:
  - SYSDIV(26:23) – wartość dzielnika zegara,
  - USESYSDIV(22) – uaktywnienie dzielnika sygnału zegarowego,
  - USEPWMDIV(20) – uaktywnienie dzielnika dla układów PWM,
  - PWMDIV(19:17) – wartość dzielnika dla układów PWM,
  - PWRDN(13) – wyłączenie pętli PLL,
  - BYPASS(11) – pominięcie pętli PLL, użycie OSC,
  - XTAL(9:6) – częstotliwość kwarcu,
  - OSCSRC(5:4) – ustawienie źródła sygnału zegarowego,
  - IOSCDIS(1) – wyłączenie wewnętrznego oscylatora,
  - MOSCDIS(0) – wyłączenie głównego oscylatora.

Ustawienia rejestru RCC

Bit/Field	Name	Type	Reset	Description
26:23	SYSDIV	R/W	0xF	System Clock Divisor Specifies which divisor is used to generate the system clock from the PLL output. The PLL VCO frequency is 400 MHz.

Value	Divisor (BYPASS=1)	Frequency (BYPASS=0)
0x0	reserved	reserved
0x1	/2	reserved
0x2	/3	reserved
0x3	/4	50 MHz
0x4	/5	40 MHz
0x5	/6	33.33 MHz
0x6	/7	28.57 MHz
0x7	/8	25 MHz
0x8	/9	22.22 MHz
0x9	/10	20 MHz
0xA	/11	18.18 MHz
0xB	/12	16.67 MHz
0xC	/13	15.38 MHz
0xD	/14	14.29 MHz
0xE	/15	13.33 MHz
0xF	/16	12.5 MHz (default)

Ustawienia rejestru RCC

Bit/Field	Name	Type	Reset	Description
9:6	XTAL	R/W	0xB	Crystal Value

This field specifies the crystal value attached to the main oscillator. The encoding for this field is provided below.

Value	Crystal Frequency (MHz) Not Using the PLL	Crystal Frequency (MHz) Using the PLL
0x0	1.000	reserved
0x1	1.8432	reserved
0x2	2.000	reserved
0x3	2.4576	reserved
0x4	3.579545 MHz	
0x5	3.6864 MHz	
0x6	4 MHz	
0x7	4.096 MHz	
0x8	4.9152 MHz	
0x9	5 MHz	
0xA	5.12 MHz	
0xB	6 MHz (reset value)	
0xC	6.144 MHz	
0xD	7.3728 MHz	
0xE	8 MHz	
0xF	8.192 MHz	

Ustawienia rejestru RCC

5:4	OSCSRC	R/W	0x1	Oscillator Source Picks among the four input sources for the OSC. The values are:

Value	Input Source
0x0	Main oscillator (default)
0x1	Internal oscillator (default)
0x2	Internal oscillator / 4 (this is necessary if used as input to PLL)
0x3	reserved

Systemy odmierzania czasu

- W mikrokontrolerach CortexM3 możemy wydzielić trzy podsystemy odmierzania czasu:
  - zegar systemowy (System Timer),
  - zegary uniwersalne (General-Purpose Timers),
  - zegar nadzorujący (Watchdog Timer).

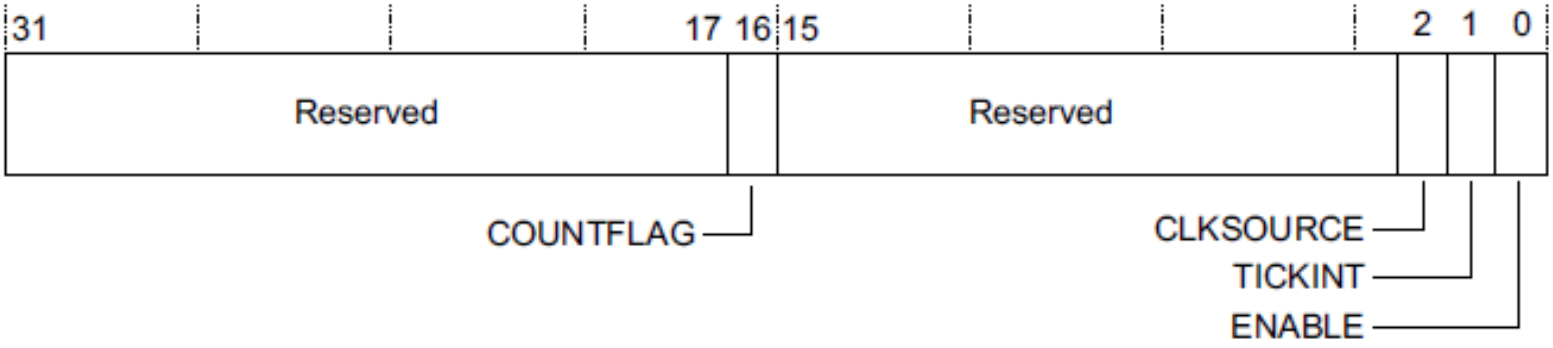
Zegar systemowy

- Zegar systemowy (System Timer) jest częścią kontrolera przerwań. Pozwala on na cykliczne generowanie przerwań. Konfiguracja zegara możliwa jest za pomocą następujących rejestrów:

Nazwa rejestru	Typ	Adres	Wartość po resecie
SysTick Control and Status Register	Read/write	0xE000E010	0x00000000
SysTick Reload Value Register	Read/write	0xE000E014	Unpredictable
SysTick Current Value Register	Read/write clear	0xE000E018	Unpredictable
SysTick Calibration Value Register	Read-only	0xE000E01C	STCALIB

SysTick Control and Status Register

- Rejestr wykorzystywany do konfigurowania zegara. Poszczególne bity tego rejestru przedstawione są na rysunku.



SysTick Control and Status Register

- Opis poszczególnych bitów jest następujący:
  - COUNTFLAG [16] - zwraca wartość 1 jeżeli licznik zliczył do 0 w czasie od ostatniego odczytu, kasowany w momencie odczytu przez aplikację
  - CLKSOURCE – 0=zewnętrzny zegar odniesienia, 1=zegar rdzenia,
  - TICKINT – 1=zliczanie w dół do zera z wygenerowaniem przerwania, 0=zliczanie w dół do zera bez generowania przerwania, możliwe jest programowe określenie czy licznik zliczył do 0 przez odczytanie COUNTFLAG,
  - ENABLE – 1= licznik pracuje w trybie wielokrotnego wyzwalania, oznacza to, że licznik jest ładowany ponownie po osiągnięciu 0 i ustawieniu bitu COUNTFLAG na 1, 0=licznik zablokowany.

SysTick Reload Value Register

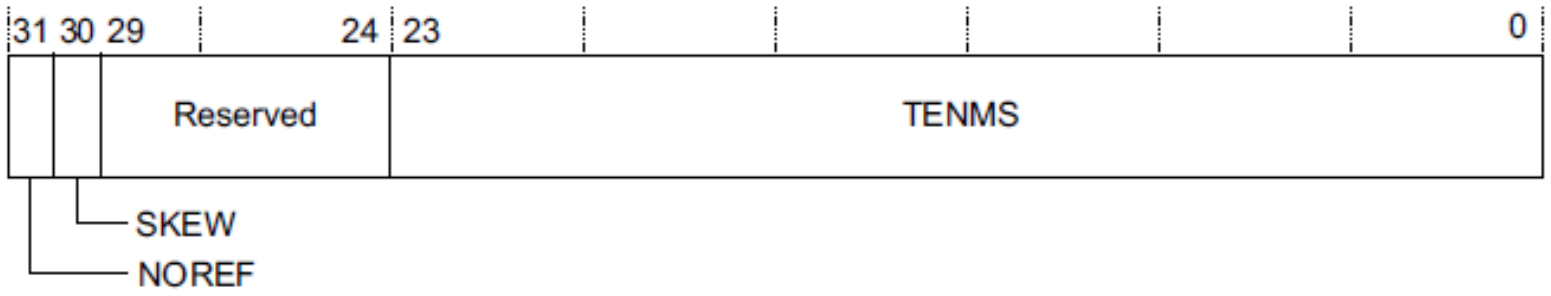
- Rejestr wykorzystywany do określenia początkowej wartości wpisywanej do licznika po osiągnięciu przez niego wartości 0.
- Rejestr może być zapisany dowolną wartością pomiędzy 1 i 0x00FFFFFF. Wpisanie jako wartości początkowej 0 jest możliwe, jednak nie daję żadnego efektu ponieważ COUNTFLAG i przerwanie są uaktywniane kiedy licznik zliczy z 1 na 0.
- Jeżeli zegar pracuje cyklicznie to uruchamia się co N+1 pulsów zegara, gdzie N jest wartością załadowana do licznika. W związku z tym, jeżeli przerwania są wymagane co 100 pulsów, do rejestru należy załadować wartość 99.
- Jeżeli nowa wartość jest wpisywana po każdym wystąpieniu przerwania, wtedy pełna wartość powinna być wpisana. Na przykład jeżeli koleiny impuls przerwania jest wymagany po 400 pulsach zegara, to należy do rejestru wpisać wartość 400.

SysTick Current Value Register

- Rejestr wykorzystywany do określenia aktualnej wartości licznika.
- Przy zapisie do tego licznika dowolnej wartości jest on kasowany do 0.
- Kasowanie tego rejestru kasuje także bit COUNTFLAG w rejestrze *SysTick Control and Status Register*.

SysTick Calibration Value Register

- Rejestr umożliwia na programowe skalowanie prędkości poprzez użycie dzielenia i mnożenia.
- Rejestr ten zawiera wartość jaką należy załadować do rejestru zegara , dla zapewnienia rozdzielczości 10ms.
- Poszczególne pola rejestru są przedstawione na rysunku.



SysTick Calibration Value Register

- NOREF – 1=zegar odniesienia jest niedostępny,
- SKEW – 1=wartość kalibrująca nie zapewni dokładnie wartości 10ms, co może wpływać na użyteczność zegara jako programowego zegara czasu rzeczywistego,
- TENMS – wartość jaką należy załadować rejestr, dla zapewnienia odmierzania 10ms odcinków czasu, zawiera wartość 0, kiedy wartość kalibracyjna jest niemożliwa do określenia.

API zegara systemowego

- Funkcje obsługi zegara systemowego zawarte są w bibliotece src/systick.c, z src/systick.h
- Dostępne funkcje
  - void SysTickDisable (void)
  - void SysTickEnable (void)
  - void SysTickIntDisable (void)
  - void SysTickIntEnable (void)
  - void SysTickIntRegister (void (#pfnHandler)(void))
  - void SysTickIntUnregister (void)
  - unsigned long SysTickPeriodGet (void)
  - void SysTickPeriodSet (unsigned long ulPeriod)
  - unsigned long SysTickValueGet (void)

Zegary uniwersalne

- Programowalne układy czasowo-licznikowe wykorzystywane są do zliczania lub odmierzania czasu zdarzeń zewnętrznych, które pobudzają wejścia układu.
- Procesory firmy Stellaris zawierają cztery bloki General-Purpose Timer Module (GPTM) oznaczone jako Timer0, Timer1, Timer2 i Timer3.
- Każdy z tych bloków posiada dwa 16-bitowe układy czasowo-licznikowe (TimerA i TimerB), które mogą ustawiane do pracy niezależnej jako układy czasowe lub liczniki lub konfigurowane do pracy 32-bitowej jako układ czasowy lub zegar czasu rzeczywistego RTC (Real Time Clock).
- Układy te mogą być także wykorzystywane do wyzwalania przetwornika analogowo-cyfrowego.

Tryby pracy układu czasowo-licznikowego

- tryby 32-bitowego układu czasowego
  - programowalny jednokrotny timer (One-Shot Timer),
  - programowalny okresowy timer (Periodic Timer),
  - zegar czasu rzeczywistego RTC wykorzystujący sygnał zegarowy 32768 kHz,
  - programowo sterowane wstrzymywanie zdarzeń (wyłączając RTC),

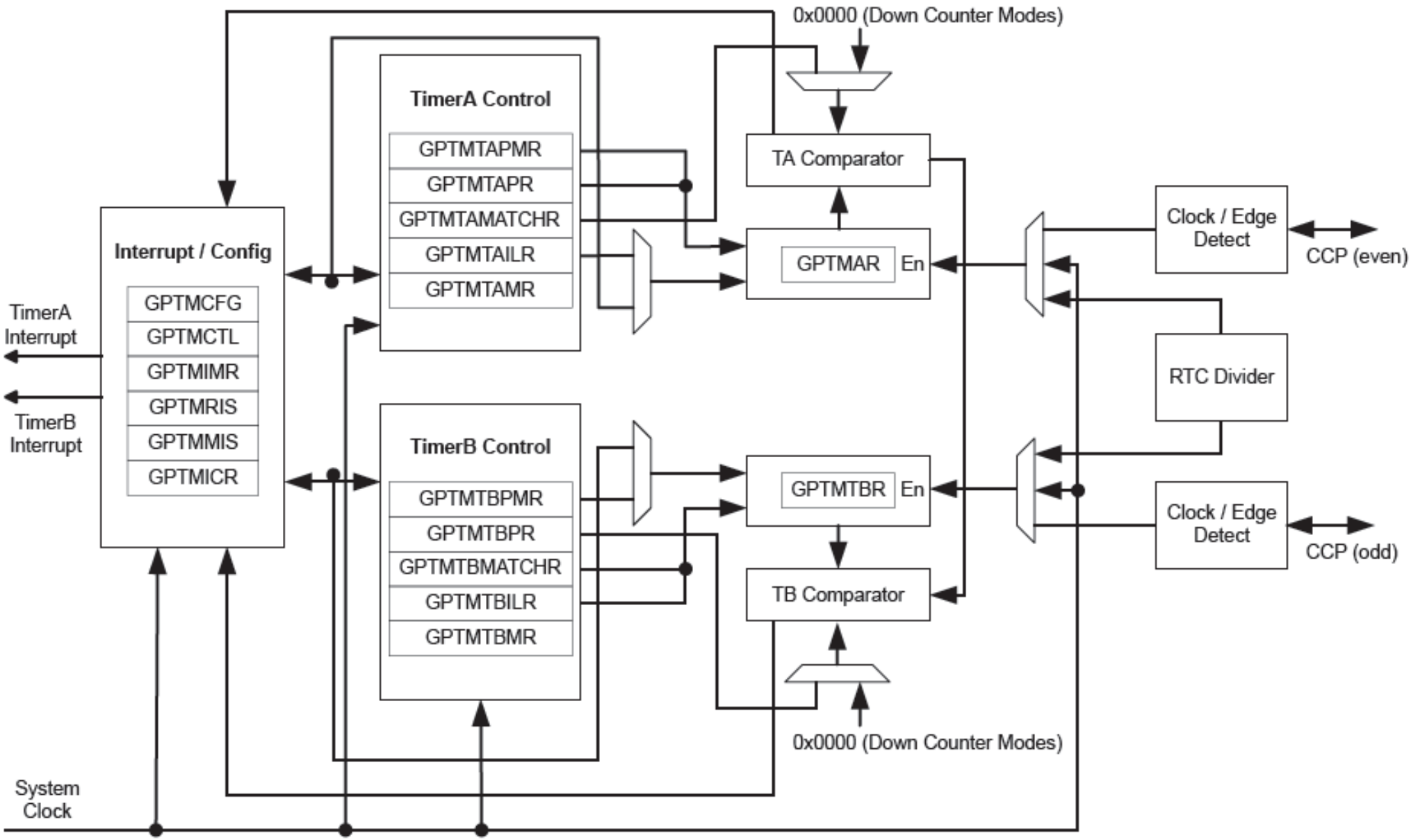
Tryby pracy układu czasowo-licznikowego

- tryby 16-bitowego układu czasowego
  - uniwersalny układ czasowy z 8-bitowym prescalerem,
  - programowalny jednokrotny timer (One-Shot Timer),
  - programowalny okresowy timer (Periodic Timer),
  - programowo sterowane wstrzymywanie zdarzeń (wyłączając RTC),

Tryby pracy układu czasowo-licznikowego

- 16-bitowy licznik
  - zliczanie zboczy na wejściu,
  - odmierzanie czasu zboczy na wejściu
- 16-bitowy PWM

Budowa bloku GPTM



Opis bloku GPTM

- Głównym elementem każdego z bloków GPTM są dwa 16-bitowe liczniki zliczające góra/dół (TimerA i TimerB), dwa 16-bitowe rejestry dopasowujące, dwa rejestry prescalera i dwa rejestry ładująco/inicjujące z powiązanymi z nimi rejestrami sterującymi.
- Pełna konfiguracja układów GPTM odbywa się programowo poprzez rejestry:
- GPTM Configuration (GPTMCFG),
- GPTM TimerA Mode (GPTMTAMR),
- GPTM TimerB Mode (GPTMTBMR) .

Opis bloku GPTM

- Po resecie moduł GPTM jest w stanie nieaktywnym i wszystkie sterujące rejestry są wykasowane do ich domyślnych wartości.

- Adresy bazowe układów czasowo-licznikowych
- Timer0: 0x4003.0000
- Timer1: 0x4003.1000
- Timer2: 0x4003.2000
- Timer3: 0x4003.3000

Adresy rejestrów

Offset	Nazwa	Typ	Reset	Opis
0x000	GPTMCFG	R/W	0x0000.0000	GPTM Configuration
0x004	GPTMTAMR	R/W	0x0000.0000	GPTM TimerA Mode
0x008	GPTMTBMR	R/W	0x0000.0000	GPTM TimerB Mode
0x00C	GPTMCTL	R/W	0x0000.0000	GPTM Control
0x018	GPTMIMR	R/W	0x0000.0000	GPTM Interrupt Mask
0x01C	GPTMRIS	RO	0x0000.0000	GPTM Raw Interrupt Status
0x020	GPTMMIS	RO	0x0000.0000	GPTM Masked Interrupt Status
0x024	GPTMICR	W1C	0x0000.0000	GPTM Interrupt Clear
0x028	GPTMTAILR	R/W	0x0000.FFFF (16-bit mode) 0xFFFF.FFFF (32-bit mode)	GPTM TimerA Interval Load
0x02C	GPTMTBILR	R/W	0x0000.FFFF	GPTM TimerB Interval Load
0x030	GPTMTAMATCHR	R/W	0x0000.FFFF (16-bit mode) 0xFFFF.FFFF (32-bit mode)	GPTM TimerA Match 232
0x034	GPTMTBMATCHR	R/W	0x0000.FFFF	GPTM TimerB Match 233
0x038	GPTMTAPR	R/W	0x0000.0000	GPTM TimerA Prescale 234
0x03C	GPTMTBPR	R/W	0x0000.0000	GPTM TimerB Prescale 235
0x040	GPTMTAPMR	R/W	0x0000.0000	GPTM TimerA Prescale Match 236
0x044	GPTMTBPMR	R/W	0x0000.0000	GPTM TimerB Prescale Match 237
0x048	GPTMTAR	RO	0x0000.FFFF (16-bit mode) 0xFFFF.FFFF (32-bit mode)	GPTM TimerA 238
0x04C	GPTMTBR	RO	0x0000.FFFF	GPTM TimerB 239

API układów czasowo-licznikowych

- Funkcje do obsługi układów czasowo-licznikowych zawarte są w bibliotece src/timer.c, z src/timer.h.
- Dostępne funkcje
  - void TimerConfigure (unsigned long ulBase, unsigned long ulConfig)
  - void TimerControlEvent (unsigned long ulBase, unsigned long ulTimer, unsigned long ulEvent)
  - void TimerControlLevel (unsigned long ulBase, unsigned long ulTimer, tBoolean bInvert)
  - void TimerControlStall (unsigned long ulBase, unsigned long ulTimer, tBoolean bStall)
  - void TimerControlTrigger (unsigned long ulBase, unsigned long ulTimer, tBoolean bEnable)
  - void TimerDisable (unsigned long ulBase, unsigned long ulTimer)

API układów czasowo-licznikowych

- Dostępne funkcje
  - void TimerEnable (unsigned long ulBase, unsigned long ulTimer)
  - void TimerIntClear (unsigned long ulBase, unsigned long ulIntFlags)
  - void TimerIntDisable (unsigned long ulBase, unsigned long ulIntFlags)
  - void TimerIntEnable (unsigned long ulBase, unsigned long ulIntFlags)
  - void TimerIntRegister (unsigned long ulBase, unsigned long ulTimer, void (#pfnHandler)(void))
  - unsigned long TimerIntStatus (unsigned long ulBase, tBoolean bMasked)
  - void TimerIntUnregister (unsigned long ulBase, unsigned long ulTimer)

API układów czasowo-licznikowych

- Dostępne funkcje
  - void TimerIntUnregister (unsigned long ulBase, unsigned long ulTimer)
  - unsigned long TimerLoadGet (unsigned long ulBase, unsigned long ulTimer)
  - void TimerLoadSet (unsigned long ulBase, unsigned long ulTimer, unsigned long ulValue)
  - unsigned long TimerMatchGet (unsigned long ulBase, unsigned long ulTimer)
  - void TimerMatchSet (unsigned long ulBase, unsigned long ulTimer, unsigned long ulValue)
  - unsigned long TimerPrescaleGet (unsigned long ulBase, unsigned long ulTimer)
  - unsigned long TimerPrescaleMatchGet (unsigned long ulBase, unsigned long ulTimer)

API układów czasowo-licznikowych

- Dostępne funkcje
  - void TimerPrescaleMatchSet (unsigned long ulBase, unsigned long ulTimer, unsigned long
  - ulValue)
  - void TimerPrescaleSet (unsigned long ulBase, unsigned long ulTimer, unsigned long ulValue)
  - void TimerQuiesce (unsigned long ulBase)
  - void TimerRTCDisable (unsigned long ulBase)
  - void TimerRTCEnable (unsigned long ulBase)
  - unsigned long TimerValueGet (unsigned long ulBase, unsigned long ulTimer)

Konfiguracja zegara

- void TimerConfigure(unsigned long ulBase, unsigned long ulConfig)
- Parametry:
  - ulBase adres bazowy modułu.
  - ulConfig konfiguracja zegara.
- Funkcja konfiguruje tryby pracy zegara. Moduł czasowy jest blokowany przed konfiguracją.
- Konfiguracja określona jest parametrem ulConfig w następujący sposób:
  - TIMER_CFG_32_BIT_OS - 32-bit one shot timer
  - TIMER_CFG_32_BIT_PER - 32-bit periodic timer
  - TIMER_CFG_32_RTC - 32-bit real time clock timer
  - TIMER_CFG_16_BIT_PAIR - Two 16-bit timers

Konfiguracja zegara

- Jeżeli konfigurowana jest para 16-bitowych zegarów, każdy z nich jest niezależnie konfigurowany. Pierwszy zegar jest ustawiany jako logiczne OR podanych wartości i ulConfig:
  - TIMER_CFG_A_ONE_SHOT - 16-bit one shot timer
  - TIMER_CFG_A_PERIODIC - 16-bit periodic timer
  - TIMER_CFG_A_CAP_COUNT - 16-bit edge count capture
  - TIMER_CFG_A_CAP_TIME - 16-bit edge time capture
  - TIMER_CFG_A_PWM - 16-bit PWM output
- Podobnie dla drugiego zegara, tylko zmienione są nazwy na TIMER_CFG_B_#.
- Przykład
  - TimerConfigure(TIMER0_BASE, (TIMER_CFG_16_BIT_PAIR | TIMER_CFG_A_ONE_SHOT | TIMER_CFG_B_CAP_COUNT));

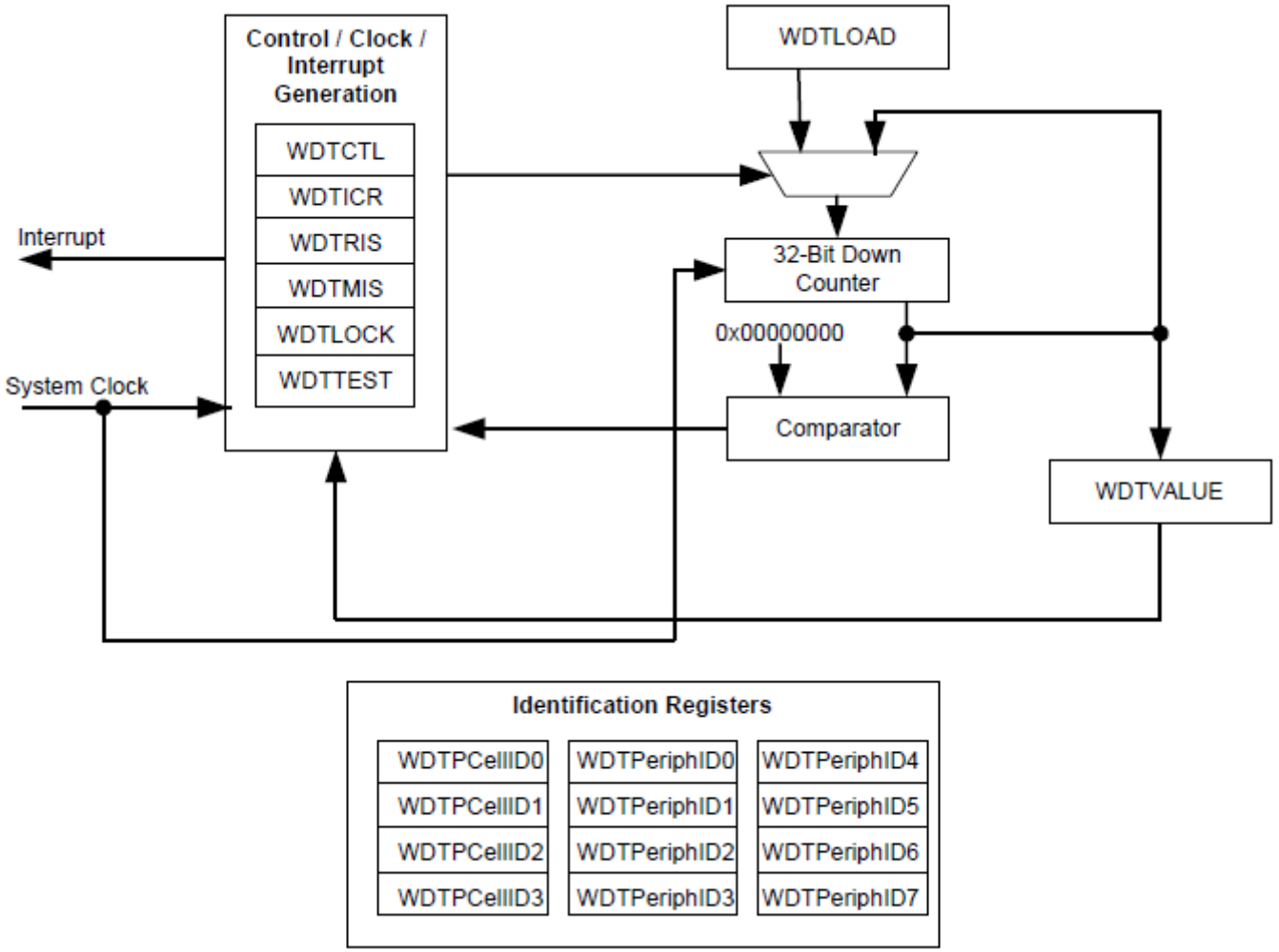
Zegar nadzorujący

- Układ zegara nadzorującego (Watchdog Timer) generuje przerwanie niemaskowalne (NMI) lub sygnał resetu, kiedy licznik wykryje przekroczenie zadanego czasu.
- Układ jest stosowany do odzyskania kontroli nad systemem w przypadku błędu oprogramowania lub błędu zewnętrznego urządzenia.
- W procesorach LM3S6965 moduł składa się z:
  - 32-bitowego licznika,
  - programowalnego rejestru ładującego,
  - układów generujących przerwanie,
  - rejestru blokujacego,
  - układu wstrzymujacego pracę liczników.

Zegar nadzorujący

- Układ nadzorujący może być skonfigurowany do generowania przerwania po pierwszym wystąpieniu przekroczenia czasu i do generowania sygnału resetującego po drugim wystąpieniu przekroczenia czasu.
- Rejestr blokujący Watchdog Timer Lock (WDTLOCK) pozwala zablokować możliwość przypadkowych zmian, po dokonaniu konfiguracji.

Zegar nadzorujący



Zegar nadzorujący

- Układ nadzorujący generuje pierwszy sygnał przekroczenia czasu, kiedy 32-bitowy licznik osiągnie wartość 0 po jego uaktywnieniu. Uaktywnienie licznika uaktywnia także przerwanie.
- Po wystąpieniu pierwszego przekroczenia licznik jest ponownie ładowany zawartością rejestru Watchdog Timer Load (WDTLOAD) i licznik ponownie zlicza w dół. Jeżeli zliczy do zera zanim wcześniejsze przerwanie zostanie skasowane to układ generuje sygnał resetu systemu.
- Zapisanie WDTLOAD nową wartością nie powoduje skasowania aktywnego przerwania. Przerwanie musi być specjalnie skasowane poprzez zapisanie rejestru Watchdog Interrupt Clear (WDTICR).
- W dowolnym czasie przerwania układu nadzorującego mogą być włączane i wyłączane. Po ponownym uaktywnieniu licznik nie startuje od ostatniego stanu, lecz jest ładowany od nowa.

API układu nadzorującego

- Funkcje obsługi zawarte są w pliku src/watchdog.c, z src/watchdog.h.
- Dostępne funkcje:
  - void WatchdogEnable (unsigned long ulBase)
  - void WatchdogIntClear (unsigned long ulBase)
  - void WatchdogIntEnable (unsigned long ulBase)
  - void WatchdogIntRegister (unsigned long ulBase, void (#pfnHandler)(void))
  - unsigned long WatchdogIntStatus (unsigned long ulBase, tBoolean bMasked)
  - void WatchdogIntUnregister (unsigned long ulBase)

API układu nadzorującego

- Dostępne funkcje:
  - void WatchdogLock (unsigned long ulBase)
  - tBoolean WatchdogLockState (unsigned long ulBase)
  - unsigned long WatchdogReloadGet (unsigned long ulBase)
  - void WatchdogReloadSet (unsigned long ulBase, unsigned long ulLoadVal)
  - void WatchdogResetDisable (unsigned long ulBase)
  - void WatchdogResetEnable (unsigned long ulBase)
  - tBoolean WatchdogRunning (unsigned long ulBase)
  - void WatchdogStallDisable (unsigned long ulBase)
  - void WatchdogStallEnable (unsigned long ulBase)
  - void WatchdogUnlock (unsigned long ulBase)
  - unsigned long WatchdogValueGet (unsigned long ulBase)