

Stellaris[®] ARM[®] Cortex[™]-M4F Training

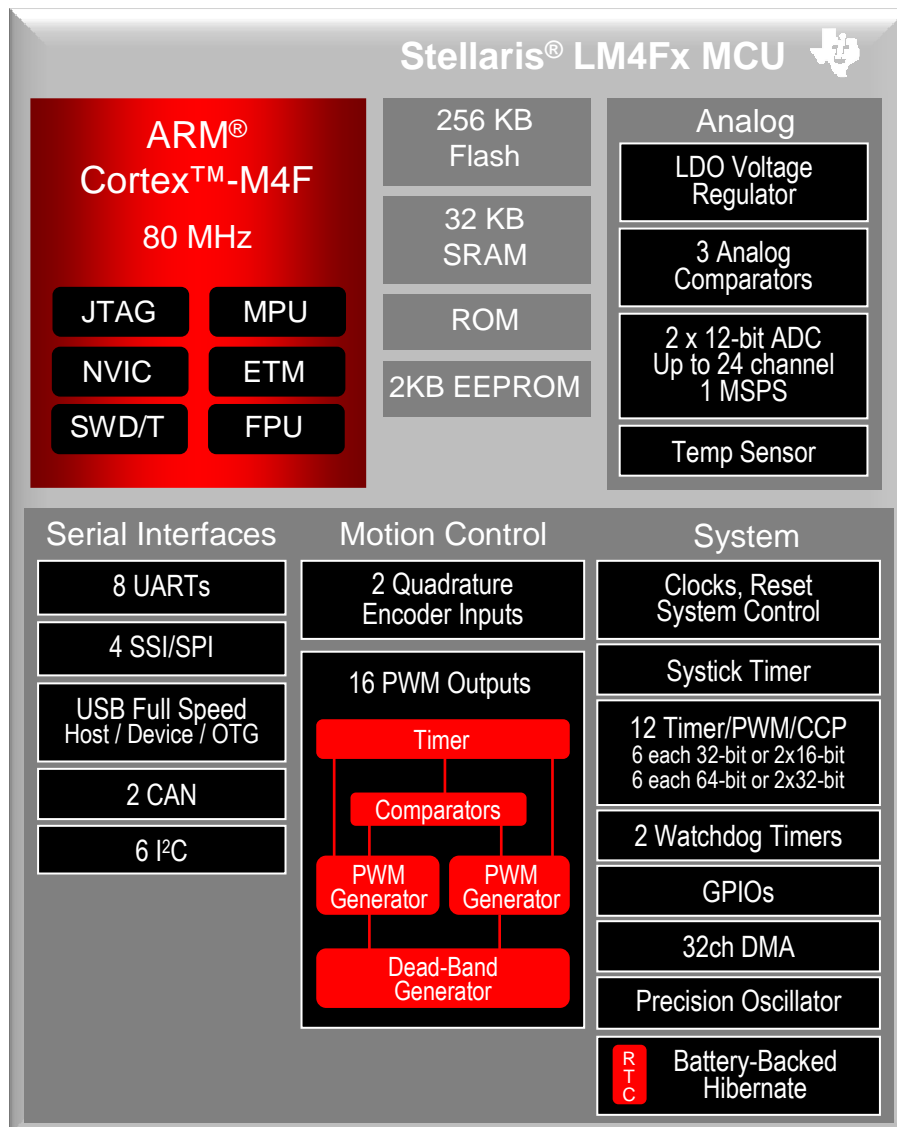
Peripheral Overview

Agenda

- Stellaris LM4F General Specifications
- Features of ARM[®] Cortex[™]-M4F
- Other System Features
 - Low Power Features
 - Watchdog Timers
- Timers and GPIOs
- Analog Peripherals
- Connectivity
- Motion Control Peripherals

Stellaris[®] LM4F Devices

Stellaris[®] ARM[®] Cortex[™]-M4F



Connectivity features:

- CAN, USB H/D/OTG, SPI, I2C, UARTs

High-performance analog integration

- Two 1 MSPS 12-bit ADCs
- Three analog comparators

Best-in-class power consumption

- As low as 370 uA/MHz
- 500µs wakeup from low-power modes
- RTC currents as low as 1.7uA

Solid roadmap

- Higher speeds
- Larger memory
- Ultra-low power

Stellaris® LM4F Technical Specifications

- ARM® Cortex™-M4F
 - IEEE 754 compliant single-precision floating-point unit
 - Embedded Trace Macrocell (ETM)
- System clock frequency up to 80 MHz
- Up to 24 Timers (twelve 16-bit & twelve 32-bit)
 - Six 16/32 bit timers (two 16 bit timers each)
 - Six 32/64 bit timers (two 32 bit timers each)
- 2 PWM modules, each containing 4 PWM generators
- 2 Quadrature Encoder Interface (QEI) modules
- Direct Memory Access support

Internal Memory

- 256 KB single-cycle Flash memory up to 40 MHz; a prefetch buffer improves performance above 40 MHz
- 32 KB single-cycle SRAM with bit-banding
- Internal ROM loaded with StellarisWare software:
 - Stellaris Peripheral Driver Library
 - Stellaris Boot Loader
 - Advanced Encryption Standard (AES) cryptography tables
 - Cyclic Redundancy Check (CRC) error detection functionality
- 2KB EEPROM

Features of ARM™ Cortex®-M4

ARM® Cortex™-M4F

- What's new
 - IEEE 754 compliant single-precision floating-point unit (FPU)
 - SIMD (Single Instruction Multiple Data) for 16-bit data types
 - 32 x 32 multiply accumulate (MAC) with 64-bit result
 - Saturation math
 - Embedded Trace Macrocell (ETM)
- Advantages
 - Higher precision in control loops (can save energy in motors)
 - Faster signal processing
 - Easier to integrate with tools such as MATLAB and LabVIEW
 - Easier debugging and code optimization with ETM

ARM[®] Cortex[™]-M4 Peripherals

- Floating Point Unit (FPU)
- Embedded Trace Macrocell (ETM)
- JTAG boundary scan and in-circuit programming
- System Timer (Systick)
- Nested Vectored Interrupt Controller (NVIC)
- Memory Protection Unit (MPU)

Floating Point Unit

- IEEE 754 compliant
- 32-bit instructions for single-precision data-processing operations
- Combined multiply and accumulate functions for increased precision
- Hardware support for conversion, addition, subtraction, multiplication with optional accumulate, division, and square-root
- Hardware support for denormals and all IEEE rounding modes
- 32 dedicated 32-bit single-precision registers, also addressable as 16 double-word registers
- FPU may be disabled to conserve power

Embedded Trace Macrocell

- ETM allows a debugger interface to record processor activity during execution
- Run code at full speed with all interrupts and timers while still connected to the debugger
 - Allows the programmer to see when and how interrupt service routines are entered
 - Observe how time-sensitive instructions are executed
- Supported by Keil and IAR with additional hardware
- Advantages:
 - See exactly where your processor time is spent
 - Selectively optimize functions that take the most time to execute
 - Easier debugging of interrupt-related problems

JTAG

- JTAG
 - Industry standard boundary scan for in-circuit testing
 - In-circuit flash programming
- Parallel JTAG TAP
 - Allows access to chip JTAG for boundary scan or Cortex-M4 JTAG for debug support
- Serial Wire Debug / Serial Wire Trace (SWD/SWT)
 - Provides debug access and control in two pins, with an optional pin for trace information

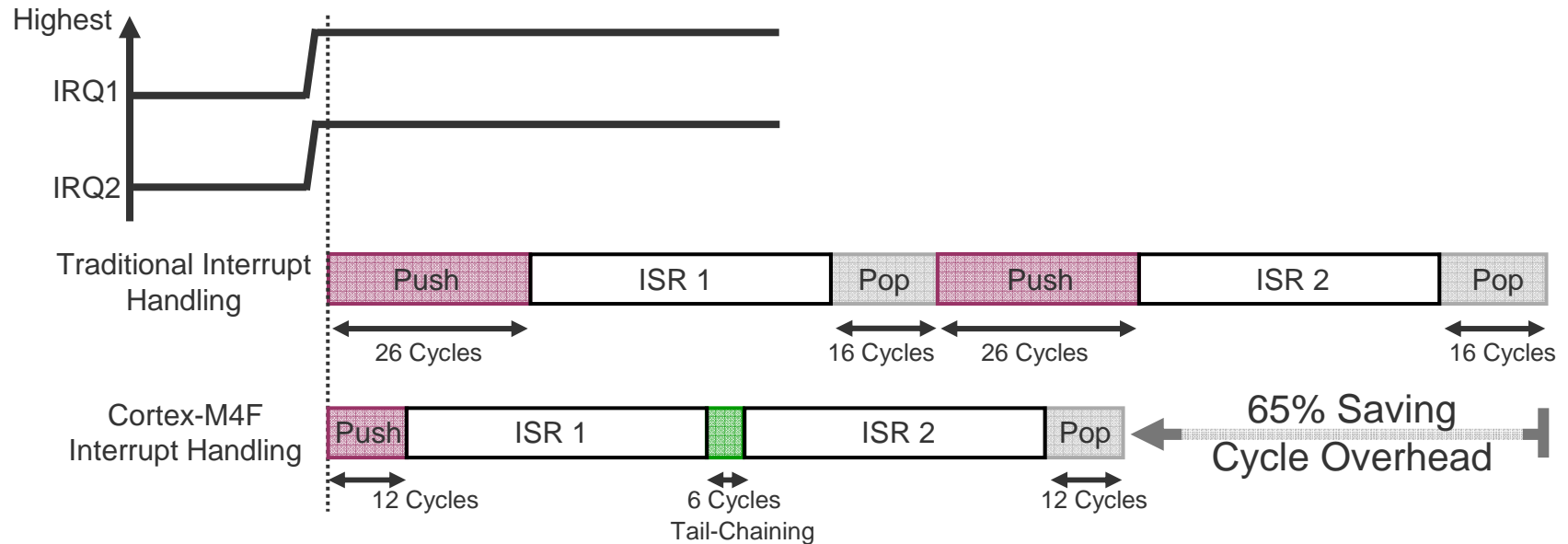
System Timer (SysTick)

- 24-bit clear-on-write, decrementing, wrap-on-zero counter
- New on LM4F devices: SysTick may run at PIOSC/4 instead of at the System Clock frequency
- Could be set as:
 - RTOS tick timer which fires at a programmable rate and invokes a SysTick routine
 - High-speed alarm timer using the system clock
 - A variable rate alarm or signal timer
 - A simple counter used to measure time to completion and time used
 - An internal clock source control based on missing/meeting durations

Nested Vectored Interrupt Controller (NVIC)

- Handles exceptions and interrupts
- 8 programmable priority levels, priority grouping
- Automatic state saving and restoring
- Automatic reading of the vector table entry
- Pre-emptive/Nested Interrupts
- Tail-chaining

Interrupt Response – Tail Chaining



- 12 cycles from IRQ1 to ISR1 (Interruptible/Continual LSM)
- 6 cycles from ISR1 exit to ISR2 entry
- 12 cycles to return from ISR2

Memory Protection Unit

- Features:
 - 8 Protection regions (no access, R/W, Read only) from 32B to 4GB range
 - Access permissions (privileged/user)
 - Overlapping protection regions with region priority
 - MPU mismatches and permission violations invoke a fault handler
- Benefits:
 - Enforce privilege rules
 - Separate processes
 - Enforce access rules

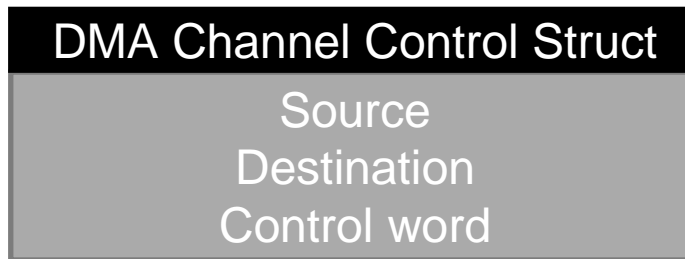
System Features

Direct Memory Access, Hibernate Module, Watchdog Timers

Direct Memory Access

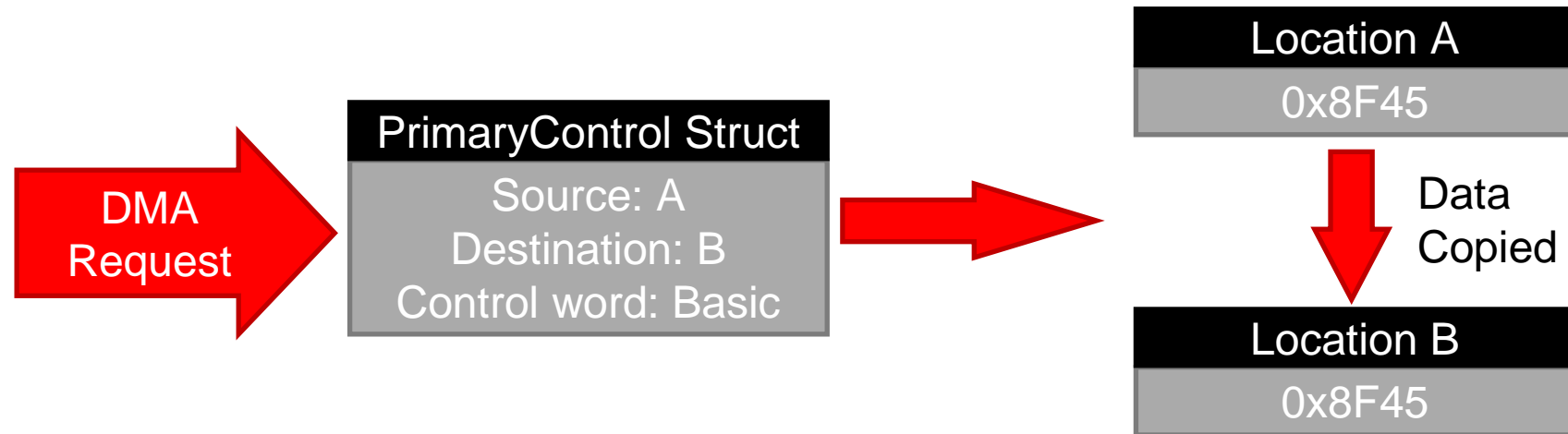
- 32 channel configurable μ DMA controller
- Dedicated channels for supported peripherals
 - One channel each for receive and transmit path for bidirectional peripherals
 - Multiple data sizes, two levels of priority, maskable device request
- Interrupt on transfer completion, with a separate interrupt per channel
- Main processor is always given priority for bus access
- Multiple transfer modes:
 - Basic
 - Auto
 - Ping-pong, for continuous data flow to/from peripherals
 - Scatter-gather, from a programmable list of arbitrary transfers initiated from a single request
- DMA requests supported by:
 - UART, Timer, USB, ADC, SSI, External Peripherals I/F, GPIO

DMA Requests



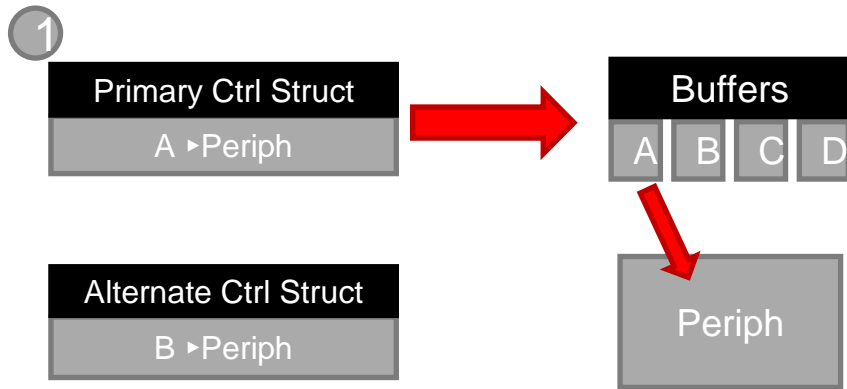
- DMA requests made by software or peripherals are sent to the DMA module, and executed based on a DMA channel control structure
- Control structure contains data on the source, destination, and how the transfer is to be carried out
- The control word consists of the transfer mode in addition to other information about the data to be sent
- Each channel has a primary control structure, and an alternate control structure

DMA Basic Transfer and Auto Transfer

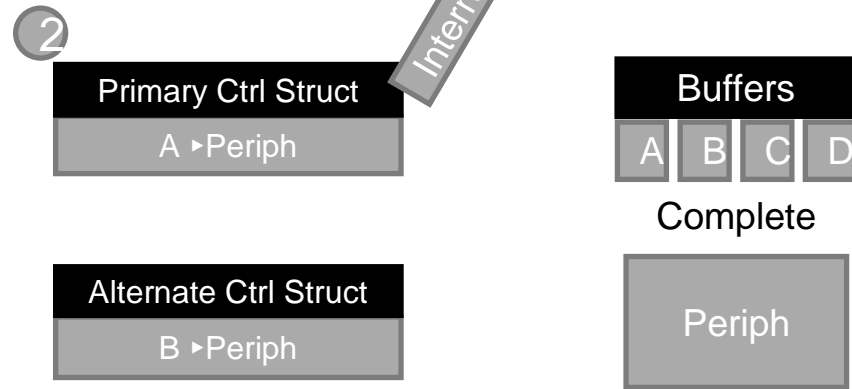


- Basic Transfers move data from one location to another as long as the request remains active, and there is still data to transfer. This works well for peripherals which normally keep their requests active for the duration of the transfer.
- Auto Transfers work similarly, but finish the transfer, even if the request is removed. This is typically used for software requests, so the request does not have to remain active for the entire duration.

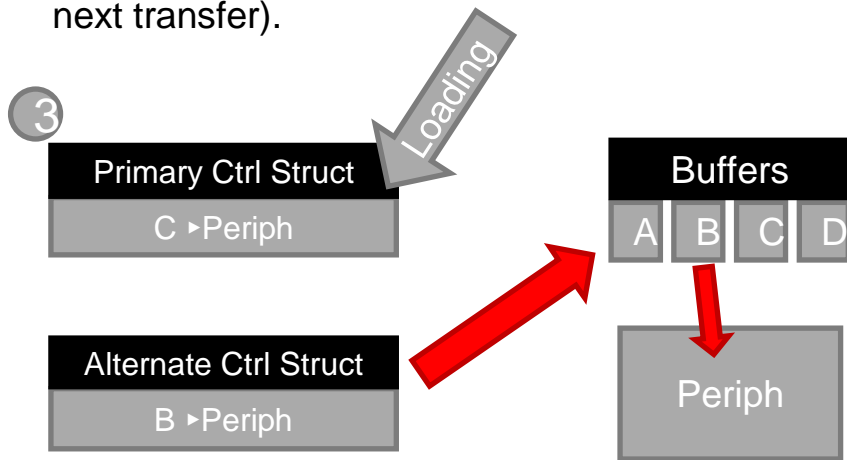
DMA Ping-Pong Transfers



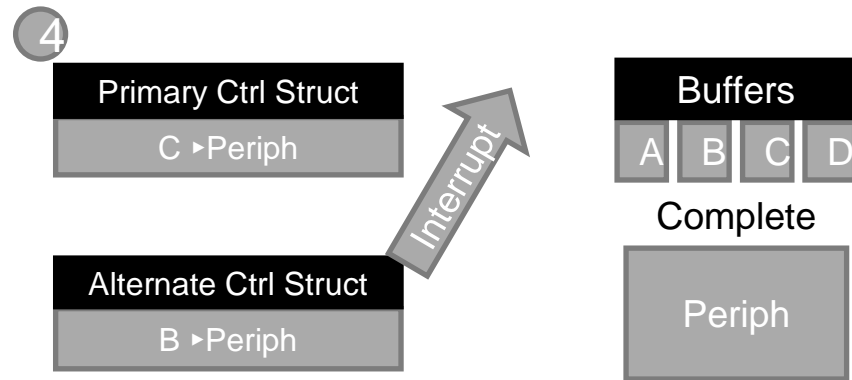
Primary control struct performs its transfer while alternate struct is idle (or being loaded with the next transfer).



Primary control struct sends an interrupt to the processor to declare its transfer complete.

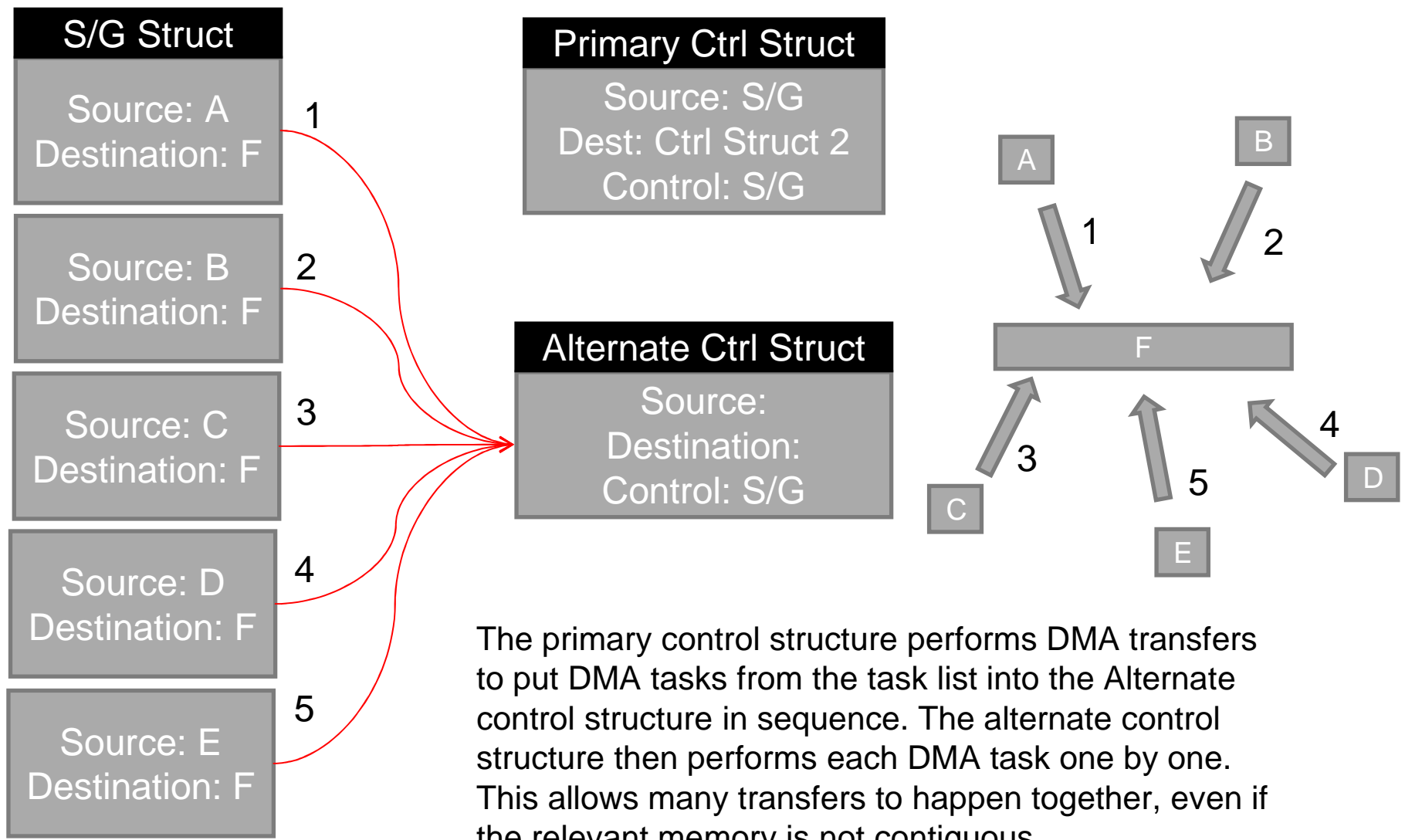


Alternate control struct performs its transfer, while primary control struct is loaded with the next transfer.



Alternate control struct sends an interrupt to the processor to declare its transfer complete, and the cycle starts over from step 1.

Scatter-Gather



The primary control structure performs DMA transfers to put DMA tasks from the task list into the Alternate control structure in sequence. The alternate control structure then performs each DMA task one by one. This allows many transfers to happen together, even if the relevant memory is not contiguous.

Hibernation and Low Power States

Stellaris family devices offer the following three low power states:

- Sleep Mode
- Deep-Sleep Mode
- Hibernation

Sleep Mode and Deep Sleep Mode

Sleep Mode

- Turns off the clock for the main processor
- Will “wake up” when a high enough priority interrupt is received

Deep Sleep

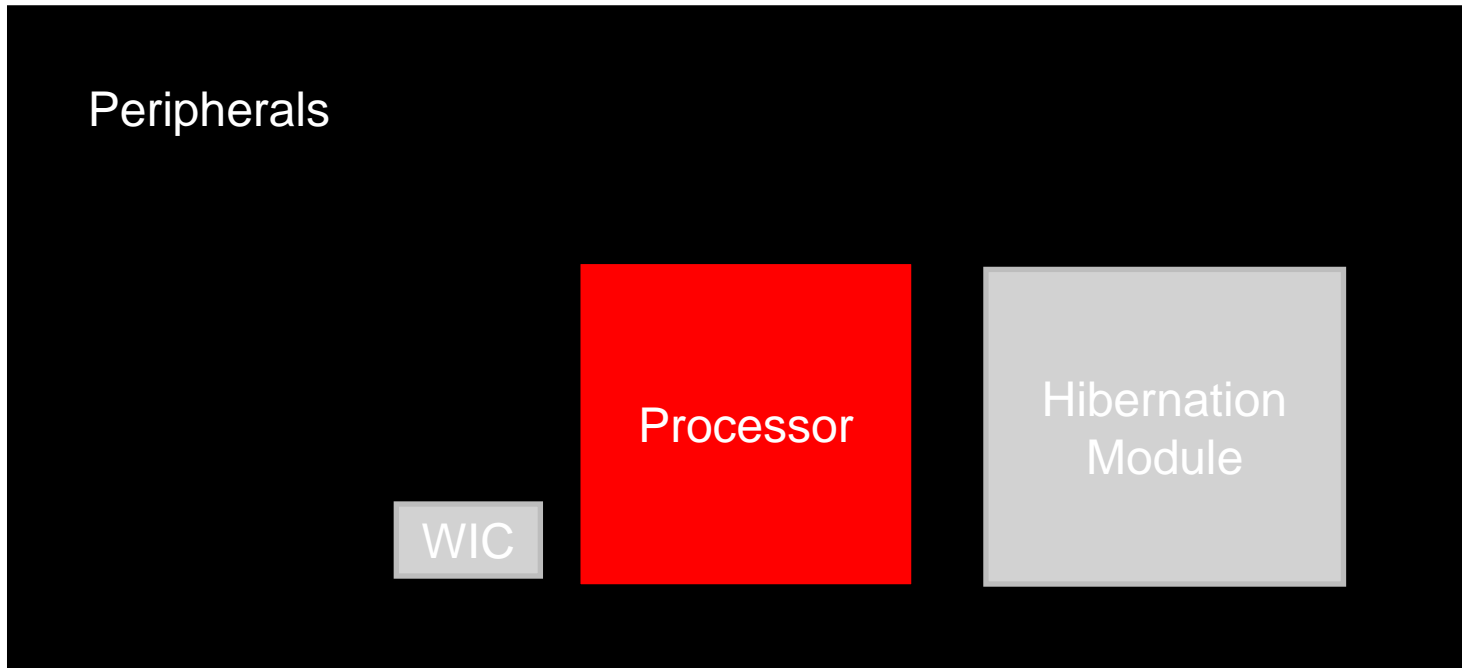
- Turns off the system clock, PLL, and flash memory
- Enables a Wake-up Interrupt Controller, which will bring the processor out of deep sleep if an interrupt is received

Hibernation Module

The Hibernation module manages removal and restoration of power as a means for reducing power consumption. It can completely remove power from all parts of the chip except itself.

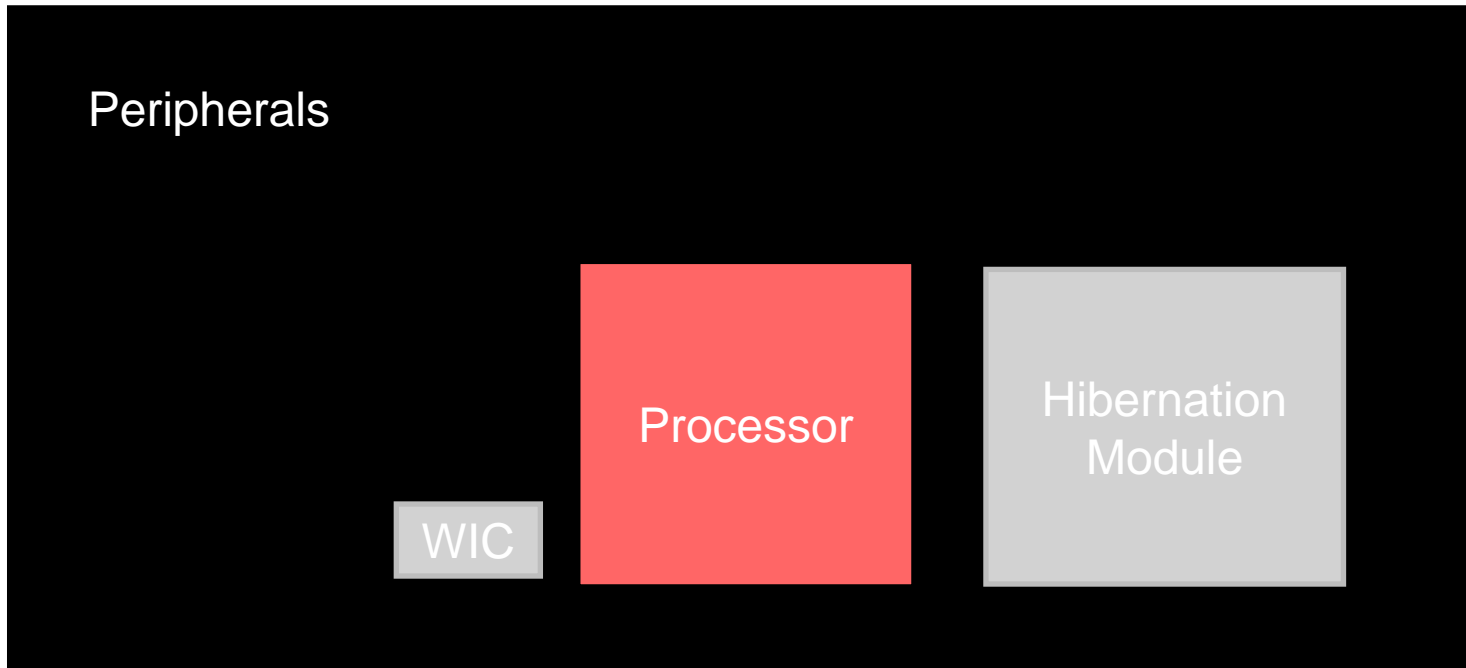
- Includes a 32-bit real-time seconds counter with 1/32,768 second resolution for timed wake up
- Dedicated pin for an external wakeup trigger
- GPIO pin state may be retained, and 16 32-bit words of non volatile memory may store the system state during hibernation
- Programmable interrupts for RTC match, external wake, and low battery events
- Runs on a separate clock and power source

Normal Operation



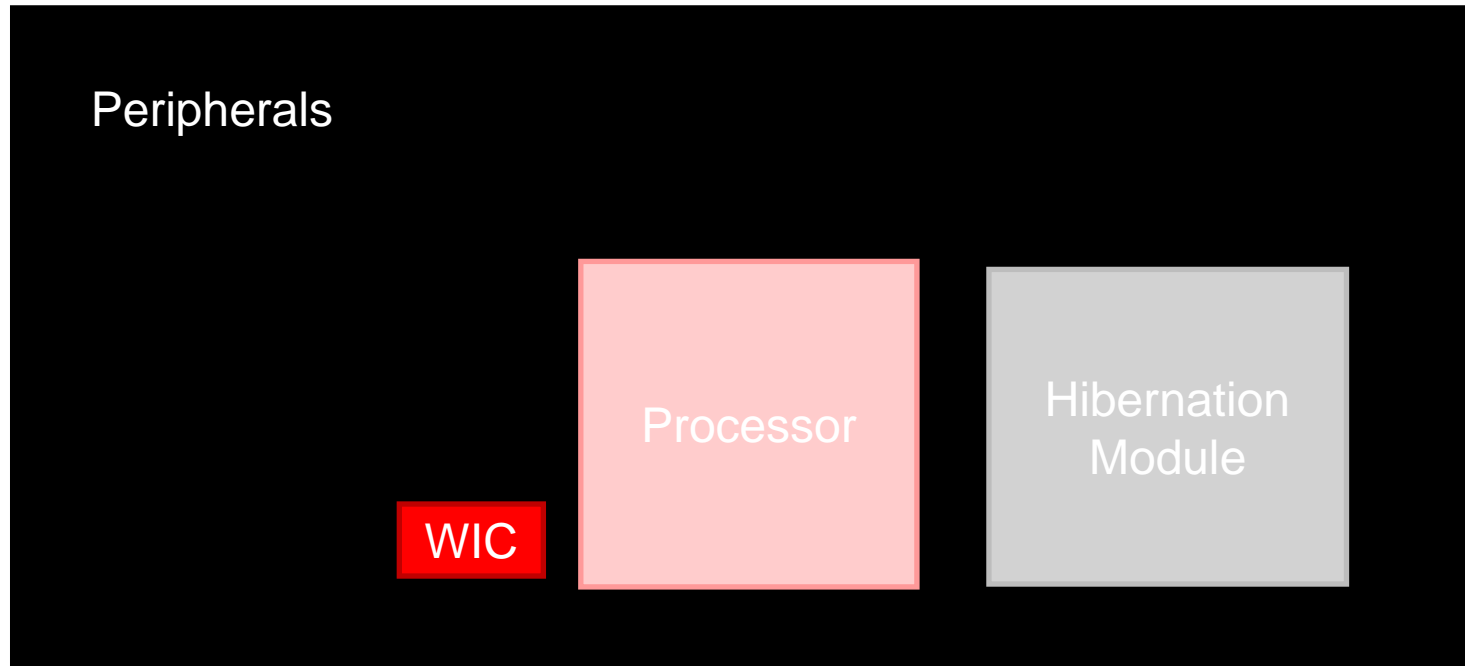
Processor and peripherals are powered; sleep, deep sleep, and hibernate-related features are turned off.

Sleep Mode



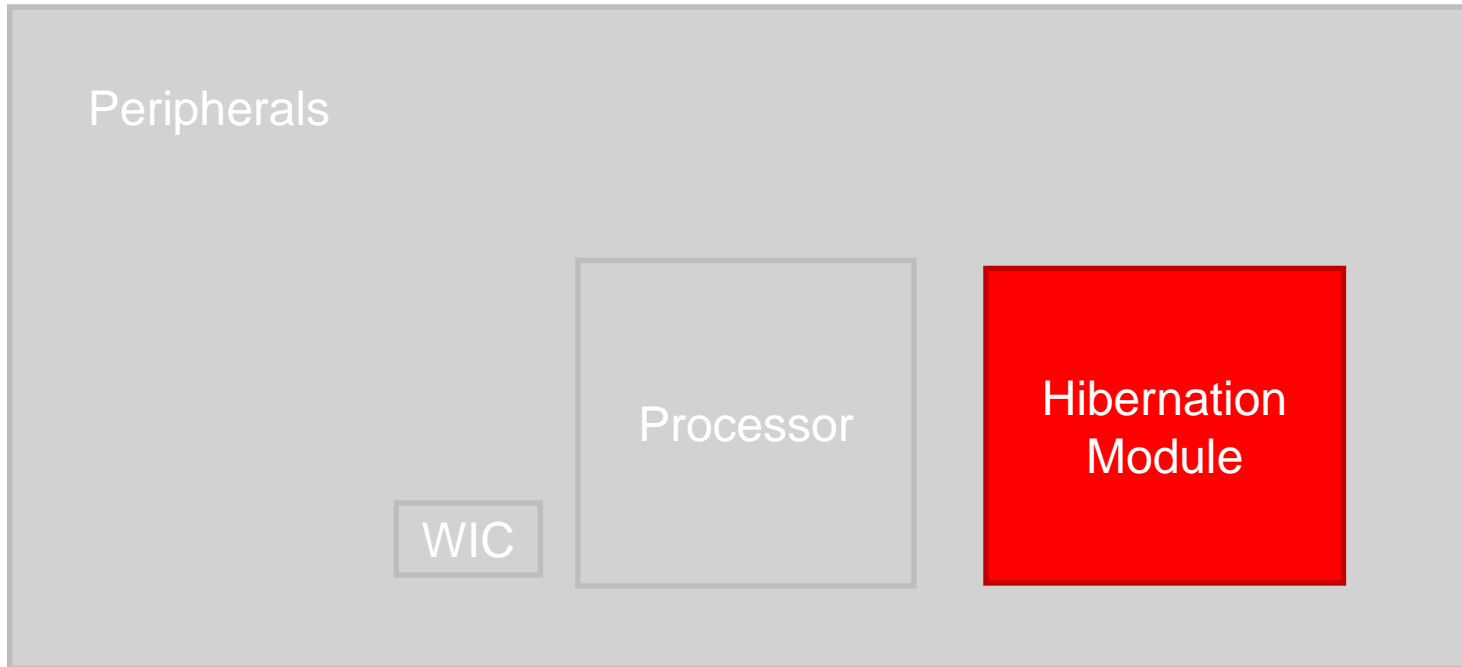
The processor enters a low-power state, and stops executing instructions until a sufficiently important interrupt is received.

Deep Sleep Mode



Processor enters a very low-power state, and some features (such as SysTick) are shut off completely. The Wakeup Interrupt Controller (WIC) intercepts important interrupts, and wakes the processor if necessary.

Hibernate



Power to the majority of the chip, including the processor and all peripherals, is shut off. The Hibernation module continues to operate using its own power source and clock until the wakeup condition is met.

Watchdog Timers

- Two 32-bit countdown timers, capable of generating maskable or non-maskable interrupts on roll-over.
- One Watchdog timer is clocked by the system clock. The other runs from the PIOSC
- Configuration may be locked to prevent inadvertent changes to the watchdog settings.
- User-enabled stalling for software debugging.
- May be used to generate a system reset when the processor fails to clear the interrupt
- Possible uses:
 - Recovering from software errors
 - Recover from external devices failing or not behaving as expected.
 - Reset and continue operation if the main crystal is damaged or removed.



Timers and GPIOs

One-Shot/Periodic Timer Mode

	Timer Use	
	Individual	Concatenated
16/32 bit GPTM Block		
32/64 bit GPTM Block		

- Count up or down
- Wait-for-Trigger mode
- μ DMA trigger
- One-Shot timer mode
 - The timer stops counting and clears the bit
- Periodic timer mode
 - Count up from 0x0 to loaded value & reloads with 0x0
 - Count down from its preloaded value & reloads with the preloaded value
 - Snap-shot mode- the actual free-running value of the timer at the time-out event is loaded and the free-running value of the prescaler is loaded

Real-Time Clock Timer Mode

	Timer Use	
	Individual	Concatenated
16/32 bit GPTM Block	N/A	
32/64 bit GPTM Block	N/A	

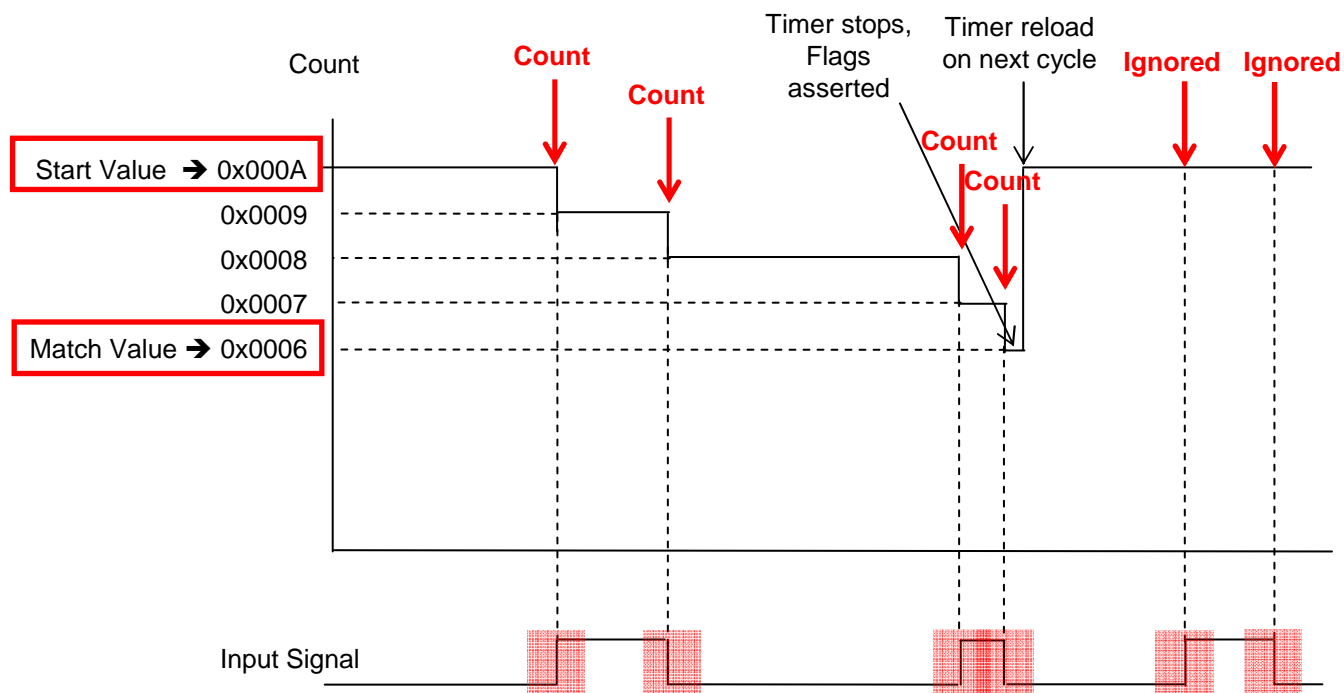
- Count up only
- After reset, the counter is loaded with a value of 0x1
- Input clock is required to be 32.768 KHz & it is then divided down to a 1-Hz
- When count value matches the preloaded value, GPTM asserts and continues counting until either a hardware reset, or it is disabled by software.
- When the timer value reaches the terminal count, the timer rolls over and continues counting up from 0x0.

Input Edge-Count Mode

	Timer Use	
	Individual	Concatenated
16/32 bit GPTM Block	<div style="border: 1px solid black; padding: 5px; display: flex; justify-content: space-around;"> <div style="background-color: red; color: white; padding: 2px 10px;">24-bit Timer</div> <div style="background-color: red; color: white; padding: 2px 10px;">Optional Prescaler</div> </div>	N/A
32/64 bit GPTM Block	<div style="border: 1px solid black; padding: 5px; display: flex; justify-content: space-around;"> <div style="background-color: red; color: white; padding: 2px 10px;">48-bit Timer</div> <div style="background-color: red; color: white; padding: 2px 10px;">Optional Prescaler</div> </div>	N/A

- The maximum input frequency is $\frac{1}{4}$ of the system frequency
- The timer is configured as a 24-bit or 48-bit up- or down-counting including the optional prescaler with upper count value
- Counting down: start value is initialized
- Counting up : start value is 0x0
- Capable of capturing three types of events
 - Rising edge
 - Falling edge
 - Both
- Interrupts, ADC and/or a μ DMA trigger can be generated

How Input Edge-Count mode works



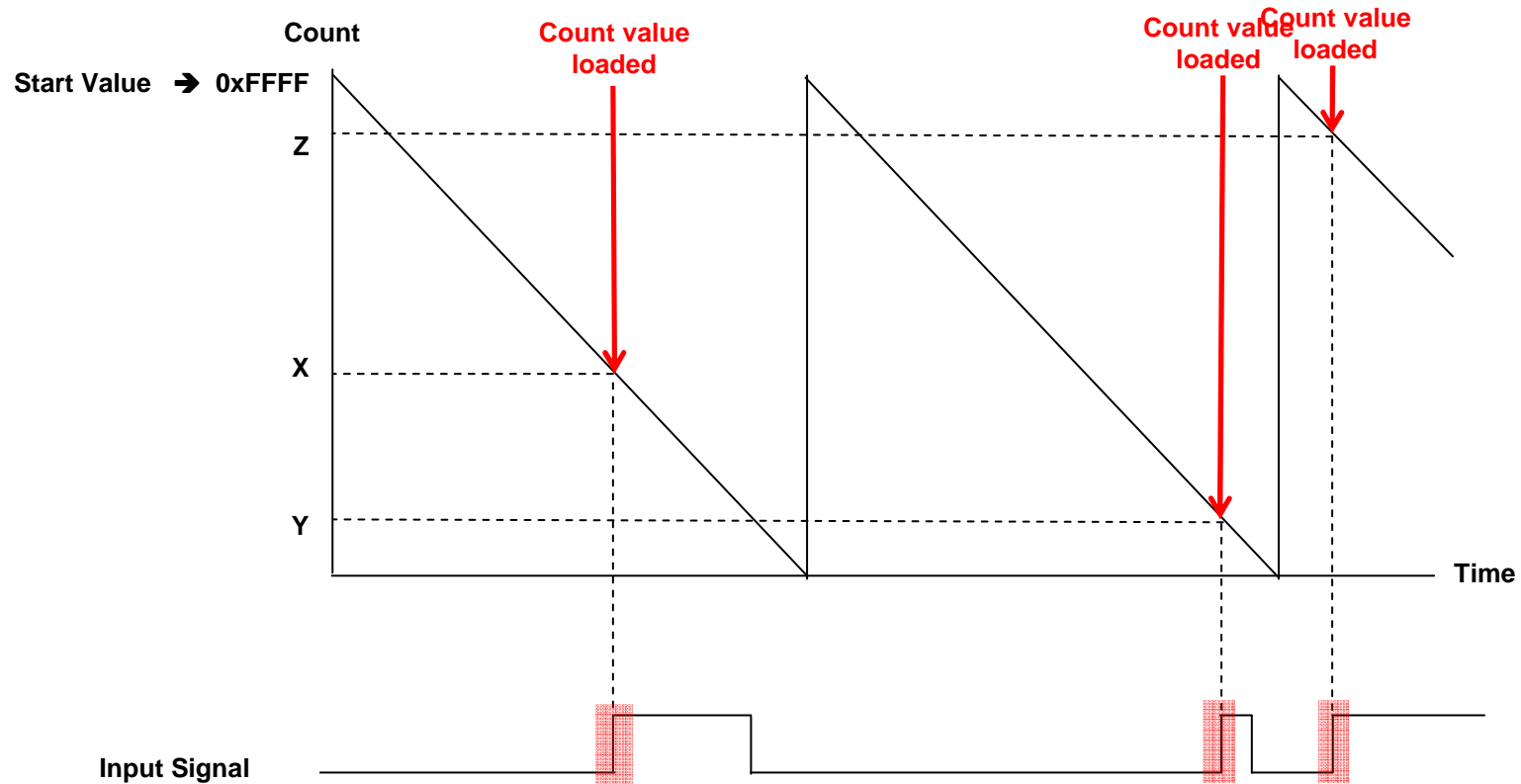
- After the match value is reached in down-count mode, the counter is then reloaded using the start value, and stopped

Input Edge-Time Mode

	Timer Use	
	Individual	Concatenated
16/32 bit GPTM Block	<div style="border: 1px solid black; padding: 5px; display: flex; justify-content: space-around;"> <div style="background-color: red; color: white; padding: 2px 10px;">24-bit Timer</div> <div style="background-color: red; color: white; padding: 2px 10px;">Optional Prescaler</div> </div>	N/A
32/64 bit GPTM Block	<div style="border: 1px solid black; padding: 5px; display: flex; justify-content: space-around;"> <div style="background-color: red; color: white; padding: 2px 10px;">48-bit Timer</div> <div style="background-color: red; color: white; padding: 2px 10px;">Optional Prescaler</div> </div>	N/A

- The maximum input frequency is $\frac{1}{4}$ of the system frequency
- The timer is configured as a 24-bit or 48-bit up- or down-counting including the optional prescaler with upper count value
- Counting down: start value is initialized
- Counting up : start value is 0x0
- Capable of capturing three types of events
 - Rising edge
 - Falling edge
 - Both
- Interrupts, ADC and/or a μ DMA trigger can be generated
- After the event has been captured, the timer doesn't stop counting

How Input Edge-Time Mode works



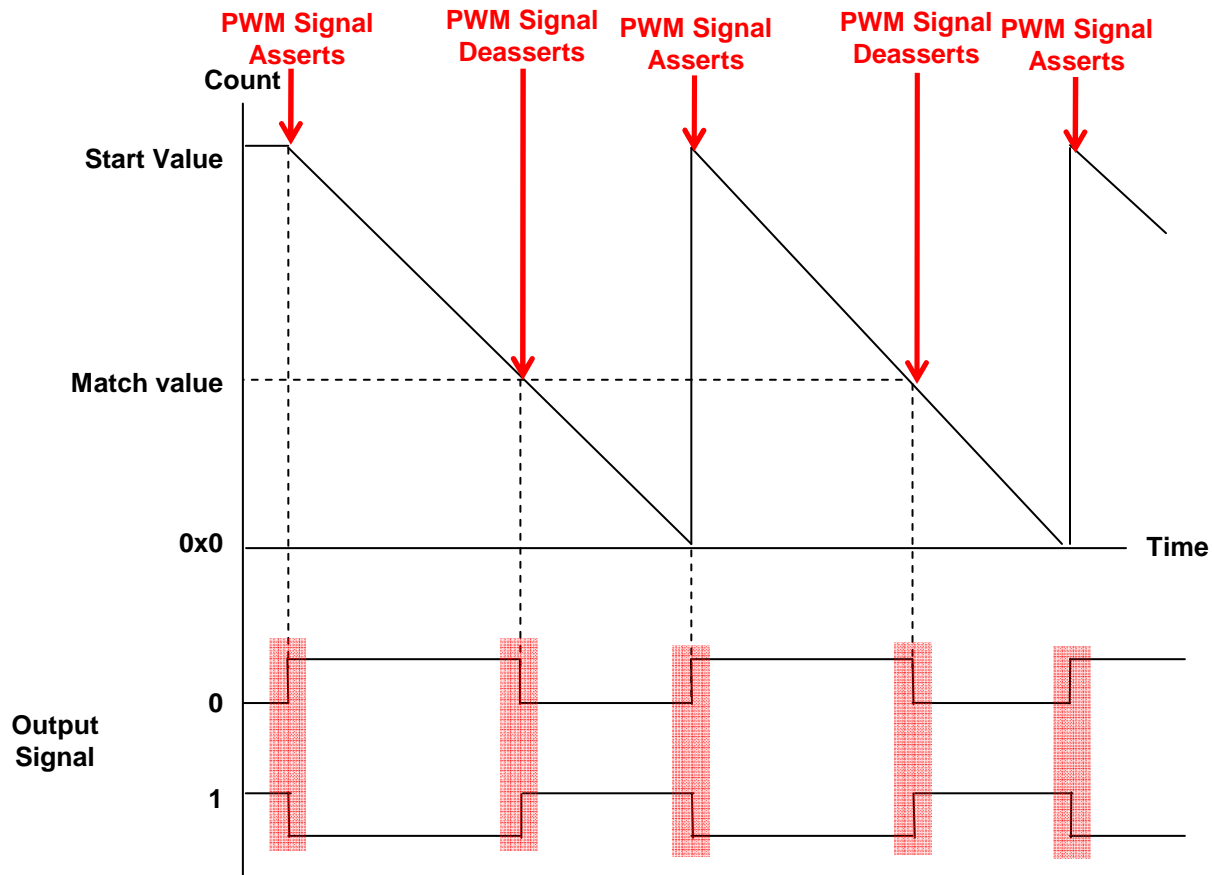
- Configured to capture rising edge events
- Each time a rising event is detected, the count value is loaded

PWM Mode

	Timer Use	
	Individual	Concatenated
16/32 bit GPTM Block	24-bit Timer	N/A
32/64 bit GPTM Block	48-bit Timer	N/A

- The timer is configured as a 34-bit or 48-bit down-counter with a start value (and thus period)
- PWM frequency and period are synchronous events and therefore guaranteed to be glitch free
- Counting down until it reaches the 0x0 state
- Wait-for-Trigger mode
- On the next counter cycle in periodic mode, the counter reloads its start value
- Capable of generating interrupts based on
 - Rising edge
 - Falling edge
 - Both

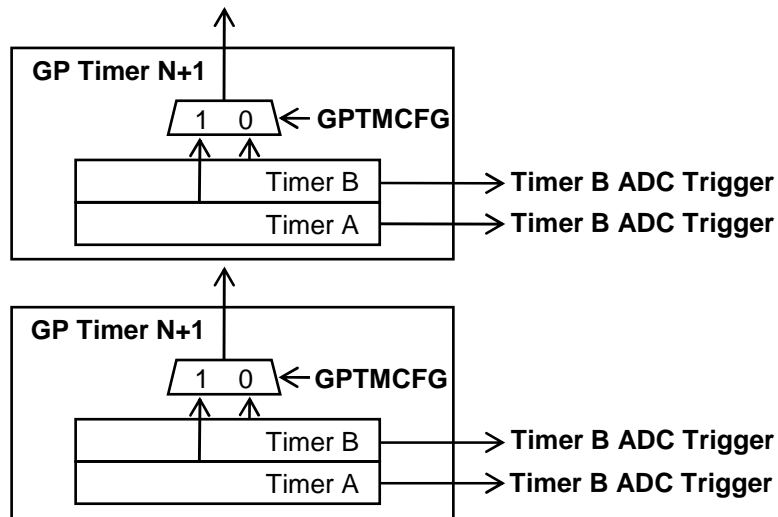
How PWM Mode works



- The output PWM signal asserts when the counter is at its start state
- The output PWM signal deasserts when the counter value equals the match value
- Capability of inverting the output PWM signal

Wait-for-Trigger Mode

- This allows daisy chaining of the timer modules
- If Timer A is in 32-bit mode, it triggers Timer A in the next module
- If Timer A is in 16-bit mode, it triggers Timer B in the same module
- Only for One-shot, periodic, and PWM modes



Synchronizing GP Timer Blocks

- Synchronize selected Timers to begin counting at the same time.
- No interruption when the Timers are synchronized.
- In concatenated mode, only the bit for Timer A must be set.

Mode	Count Dir	Time Out Action
32-bit One Shot	-	N/A
32-bit periodic	Down	Count value = ILR
	Up	Count value = 0
32-bit RTC	Up	Count value = 0
16-bit One Shot	-	N/A
16-bit Periodic	Down	Count value = ILR
	Up	Count value = 0
16-bit Edge-Count	Down	Count value = ILR
	Up	Count value = 0
16-bit Edge-Time	Down	Count value = ILR
	Up	Count value = 0
16-bit PWM	Down	Count value = ILR

DMA Operation

- The timers each have a dedicated μ DMA channel
- The request is a burst type and occurs whenever a timer raw interrupt condition occurs
- The arbitration size of the μ DMA transfer should be set to the amount of data that should be transferred whenever a timer event occurs.

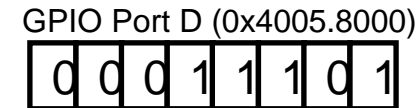
General Purpose IO

- Programmable pad configuration through GPIO module
- Any GPIO can be an external interrupt source
 - GPIO banks P and Q can have individual interrupts for each pin (larger packages)
- Toggle rate up to the CPU clock speed on the Advanced High-Performance Bus
- 5-V-tolerant input/outputs
- Programmable Drive Strength
 - 2, 4, 8 mA or 8 mA with slew rate control
- Programmable weak pull-up, pull-down, and open drain
- Digital input enables

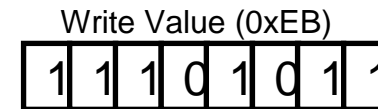
GPIO Address Masking

Bit-banded regions of memory have a base address (where the actual data is stored) and a range of possible offset addresses, which act as bit-masks for the main memory location during writes and reads.

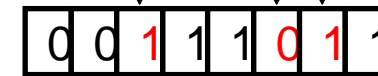
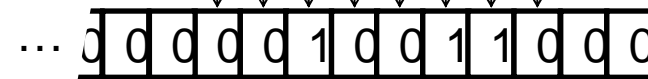
The register we want to change is GPIO Port D (0x4005.8000):



The value we will write is 0xEB:



Instead of writing to GPIO Port D directly, write to 0x4005.8098. Bits 9:2 (shown here) become a bit-mask for the value you write.



New value in GPIO Port D (note that only the red bits were written)

Analog Features

Comparators and ADCs

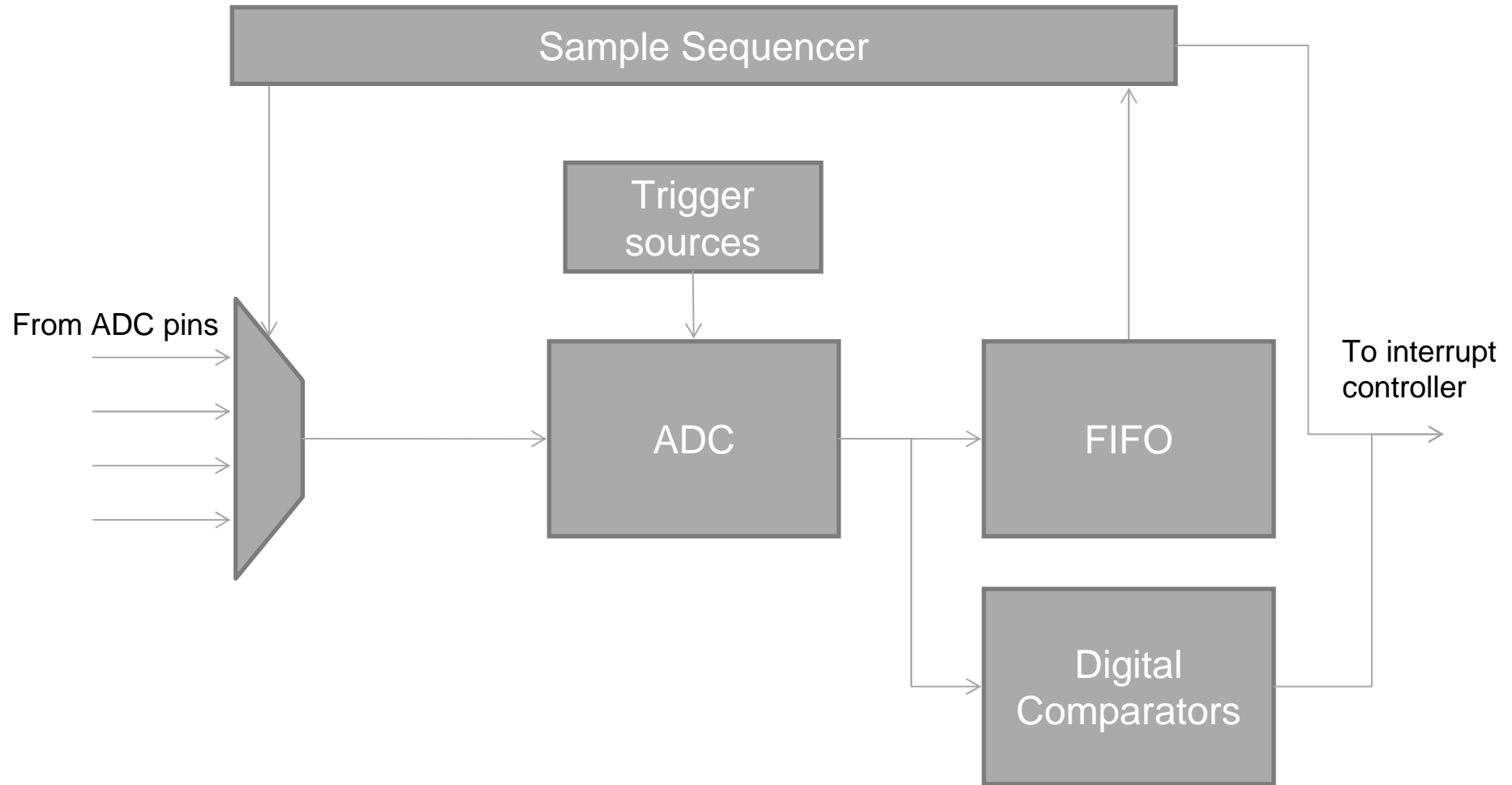
Analog to Digital Converter (ADC)

- 2 ADC modules
- 24 shared analog input channels
- 12-bit precision ADC
- Single-ended and differential-input configurations
- On-chip internal temperature sensor
- Max. sample rate of one million samples per second
- Flexible triggers (timers, analog comparators, PWM, GPIO, software)
- Hardware averaging of up to 64 samples for improved accuracy
- Efficient transfers using Direct Memory Access (μ DMA)
 - Dedicated channel for each sample sequencer
 - ADC module uses burst requests for DMA

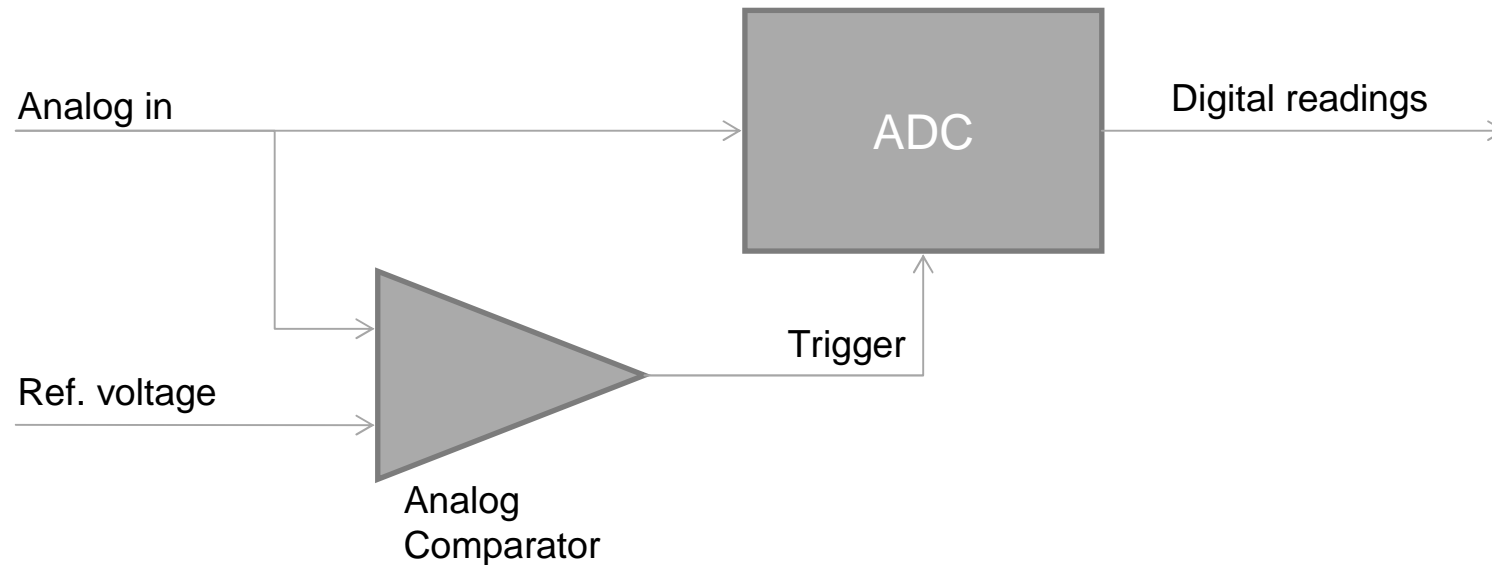
Analog Comparators

- 3 Analog comparators
- Multiple sources for reference voltages
 - Independent external pins
 - Shared external pin
 - Shared internal voltage source
- Can be used to generate processor interrupts.
- Can be used to trigger the ADC
 - Useful for limit-checking, or infrequent out-of-range events
 - Save processor time spent on the ADC
 - Faster notification when an input signal crosses a threshold
 - ADC triggers are independent of any interrupt generated

ADC Block Diagram



ADC Block Diagram



The analog comparators may also be used as ADC trigger sources:

- Allows the user to monitor a sensor value after it passes some threshold voltage.
- Conversions will only be triggered when the voltage is inside the range of interest

Analog measurements

- Shared inputs
 - Both ADC modules are connected to all 24 analog pins, and can take measurements independently.
- Sequencers:
 - ADC can be configured to take multiple different samples back to back from a single trigger, using a sequencer.
 - Sequencers come in 1,4, and 8 entry sizes.
 - The data is collected and put in a register, where it can be retrieved using DMA, reducing the CPU's time spent accessing the ADC
- The combination of sequencers, shared inputs, and direct memory access allows for complex analog measurement patterns without spending much processor interference.

Communications

SSI, I²C, UART, USB, CAN

Synchronous Serial Interface (SSI)

- Up to 4 SSI modules
- Programmable interface operation for Freescale SPI, MICROWIRE, and Texas Instruments synchronous serial interfaces
- Programmable clock bit rate and prescaler with SSI slave clock frequencies up to 1/6th of the system clock
- Separate transmit and receive FIFOs, 16 bits wide, 8 locations deep
- Programmable data frame size from 4 to 16 bits
- Internal loopback test mode for diagnostic/debug testing
- Efficient transfers using Micro Direct Memory Access Controller (μ DMA)
 - Separate channels for transmit and receive
 - Receive single request asserted when data is in the FIFO; burst request asserted when FIFO contains 4 entries
 - Transmit single request asserted when there is space in the FIFO; burst request asserted when FIFO contains 4 entries

Inter-Integrated Circuit (I²C)

- Up to 6 I²C Modules
- Master and slave modes supported
- Simultaneous master and slave operation
- Master arbitration, clock synchronization, multi-master support, and 7-bit addressing modes
- There are a total of four I²C modes:
 - Master Transmit, Master Receive
 - Slave Transmit, Slave Receive
- Standard (100 Kbps), Fast (400 Kbps), and High (3.4 Mbps) speeds supported
- Master and slave interrupts support
 - I²C master generates interrupts when a transmit or receive operation completes (or aborts).
 - I²C slave generates interrupts when data has been sent or requested by a master.

Universal Asynchronous Receiver/Transmitter (UART)

- Up to 8 UARTs
- Each UART has:
 - Separate transmit and receive FIFOs
 - Programmable FIFO length
 - FIFO trigger levels of 1/8, 1/4, 1/2, 3/4, and 7/8
 - Programmable baud-rate generator allowing rates up to speeds up to 10Mbps
 - Standard asynchronous communication bits for start, stop and parity
 - False start bit detection
 - Line-break generation and detection
- Fully programmable serial interface characteristics:
 - 5, 6, 7, or 8 data bits
 - Even, odd, stick, or no-parity bit generation/detection
 - 1 or 2 stop bit generation
- IrDA serial-IR (SIR) encoder/decoder providing:
 - Programmable use of IrDA Serial InfraRed (SIR) or UART input/output
 - Support of IrDA SIR encoder/decoder functions for data rates up to 115.2 Kbps half-duplex
 - Support of normal 3/16 and low-power (1.41-2.23 μ s) bit durations
 - Programmable internal clock generator enabling division of reference clock by 1 to 256 for low-power mode bit duration
- ISO 7816 Support

Universal Asynchronous Receiver/Transmitter (UART)

- LIN protocol support
- EIA-485 9-bit support
- Standard FIFO-level and End-of-Transmission interrupts
- Efficient transfers using Micro Direct Memory Access Controller (μ DMA)
 - Separate channels for transmit and receive
 - Receive single request asserted when data is in the FIFO; burst request asserted at programmed FIFO level
 - Transmit single request asserted when there is space in the FIFO; burst request asserted at programmed FIFO level

Universal Serial Bus (USB)

Integrated controller and PHY

- USB 2.0 Full Speed (12 Mbps) operation
- Devices with **OTG/Host/Device**
- Transfer: Control, Interrupt, Bulk and Isochronous
- 16 Endpoints
 - 0 and 1 hardwired for control transfers (one in, one out)
 - Remaining 14 may be configured by software.
- 4 KB Dedicated Endpoint Memory
 - Direct Memory Access
 - One endpoint may be defined for double-buffered 1023-byte isochronous packet size.

USB-IF Compliance

- TI is a member of the USB Implementers Forum.
- Complies with USB-IF certification standards
- TI's Stellaris VID available for sublicense (with assigned PIDs).

Controller Area Network (CAN)

- 2CAN controllers
- Each supports CAN protocol version 2.0 part A/B
- Bit rates up to 1Mb/s
- 32 message objects, each with own identifier mask
- Maskable interrupt
- Disable automatic retransmission mode for TTCAN
- Programmable loop-back mode for self test operation

Motion Control Features

PWM and Quadrature Encoder Interface

Pulse Width Modulation (PWM)

2 PWM modules with 4 generators each. Each generator contains:

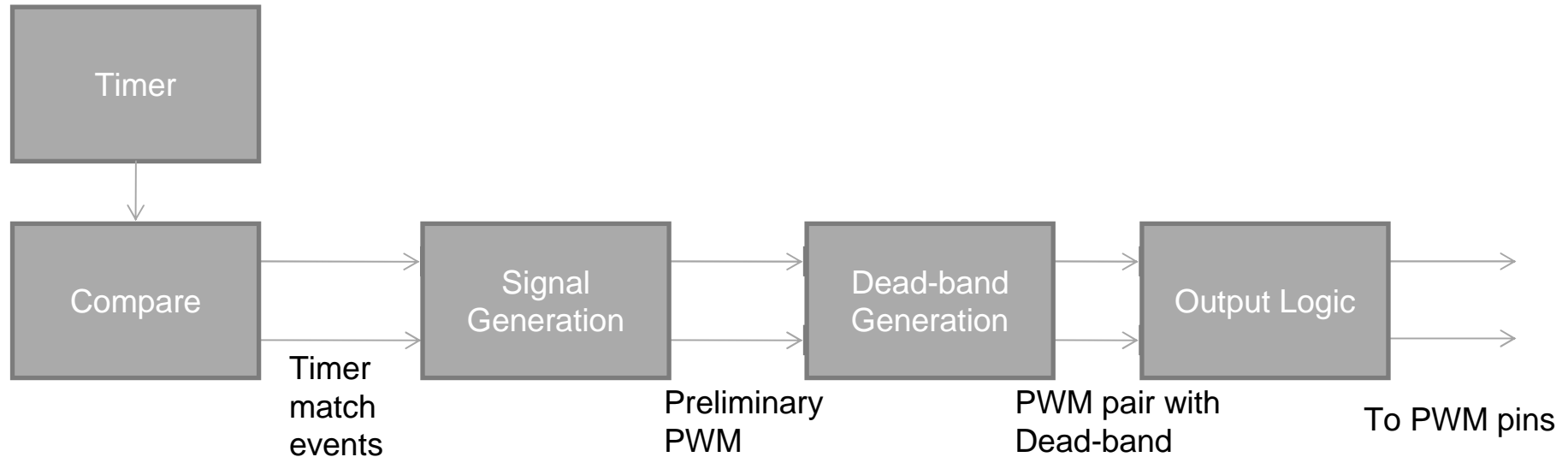
- One 16-bit counter
 - Runs in down or up/down mode
 - Output frequency controlled by a 16-bit load value
 - Load value updates can be synchronized
 - Produces output signals at zero and load value
- Two comparators
 - Comparator value updates can be synchronized
 - Produces output signals on match
- PWM generator
 - Output constructed based on actions taken as a result of the counter and comparator output signals
 - Produces two independent PWM signals



Pulse Width Modulation (PWM) cont.

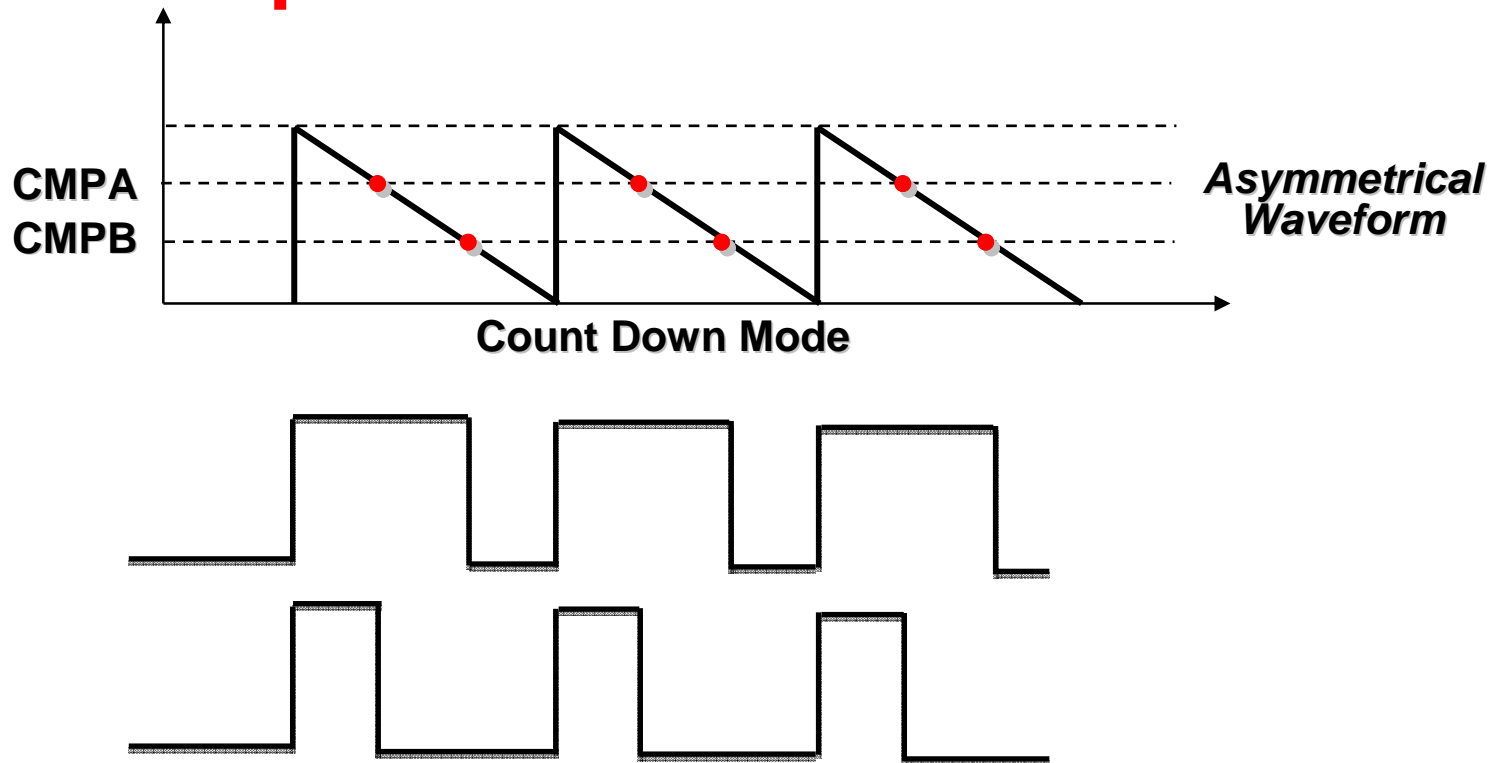
- Dead band generator
 - Produces two PWM signals with programmable dead band delays suitable for driving a half-H bridge
 - Can be bypassed, leaving input PWM signals unmodified
- Output control block
 - PWM output enable of each PWM signal
 - Optional output inversion of each PWM signal
 - Optional fault handling for each PWM signal
 - Synchronization of timers in the PWM generator blocks
 - Synchronization of timer/comparator updates across the PWM generator blocks
 - Interrupt status summary of the PWM generator blocks

PWM Block Diagram



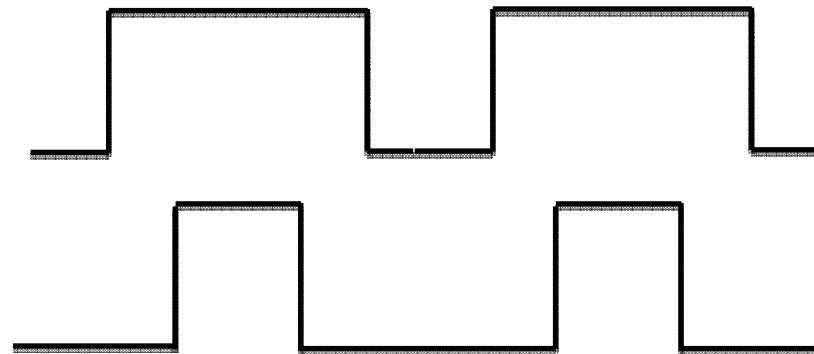
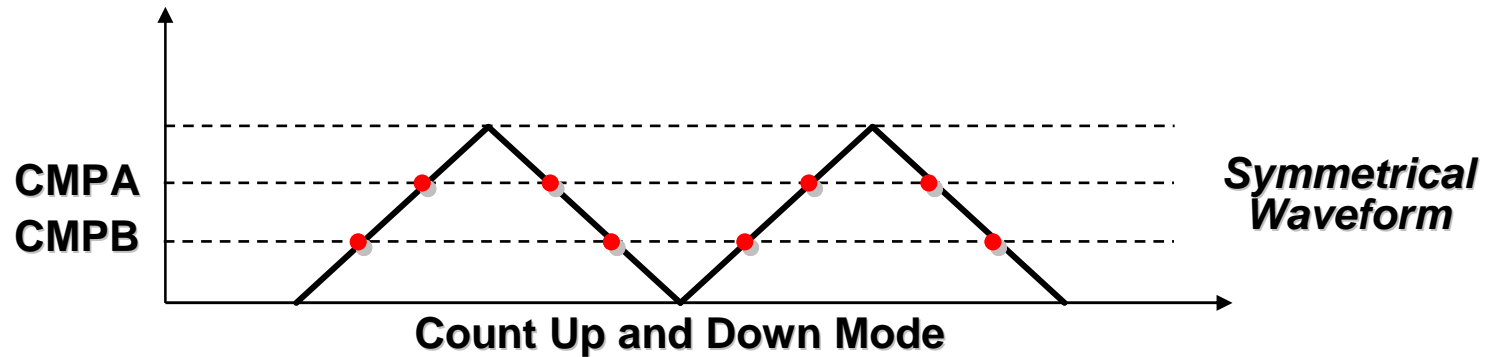
The PWM module essentially consists of a timer, a configurable comparison module, and three stages of signal generation and conditioning logic.

PWM Compare Event Waveforms



Match events produce signals, which can be used to drive positive or negative edges of a PWM. Each counter may generate up to two PWMs using different combinations of compare events.

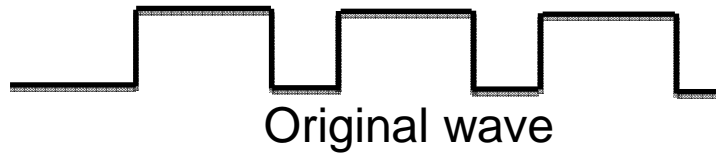
PWM Compare Event Waveforms



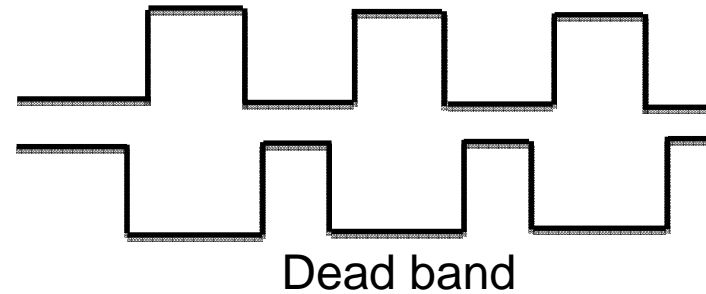
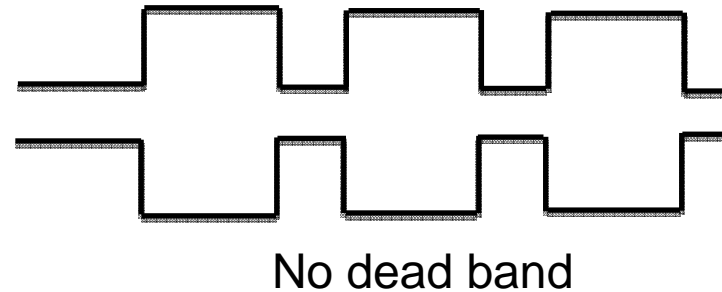
Up-down mode is well suited to symmetrical, center-aligned waveforms.

PWM Dead bands and Output Logic

PWM waveforms from the counters may be used as is, or the dead band and output logic can produce a complement waveform to one of them.



The complement waveform may optionally have dead bands inserted, which are convenient for some motor control applications.



Fault-condition Interrupts

- A fault condition is one in which the controller must be signaled to stop normal PWM function and then sets the outputs to a safe state.
- There are two basic situations where this becomes necessary:
 - The controller is stalled and cannot perform the necessary computation in the time required for motion control (Stall signal from internal debugger)
 - An external error or event is detected (FAULTn pin)



Quadrature Encoder Interface (QEI)

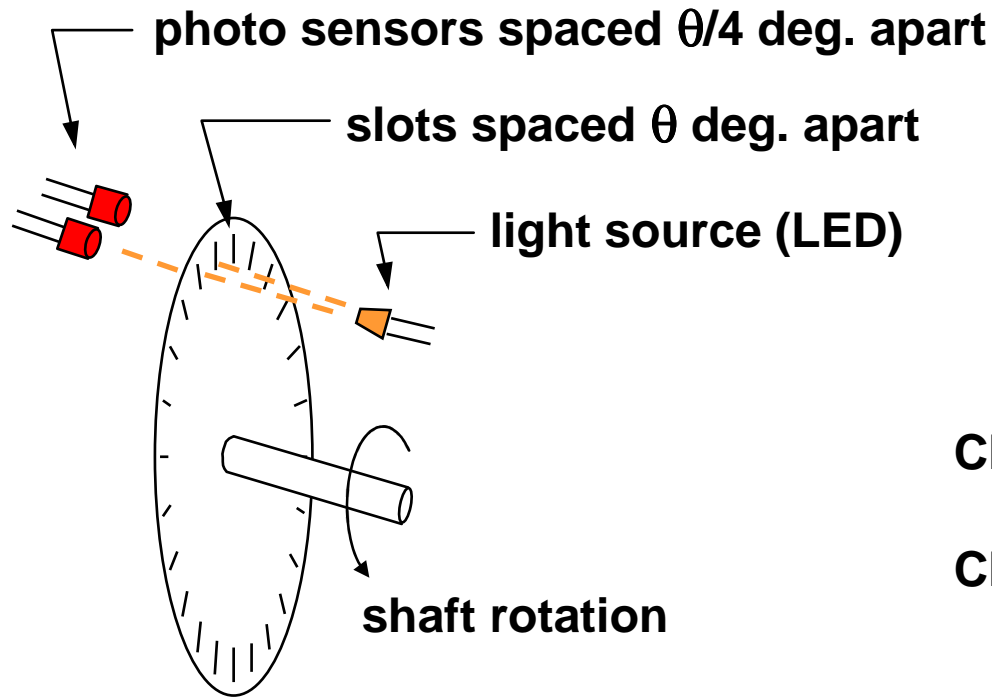
Also known as a 2-channel incremental encoder, a quadrature encoder interface converts angular displacement into a pulse signal. It is typically used in motion control applications, and can track the position, velocity, and direction of a motor

Stellaris QEI features:

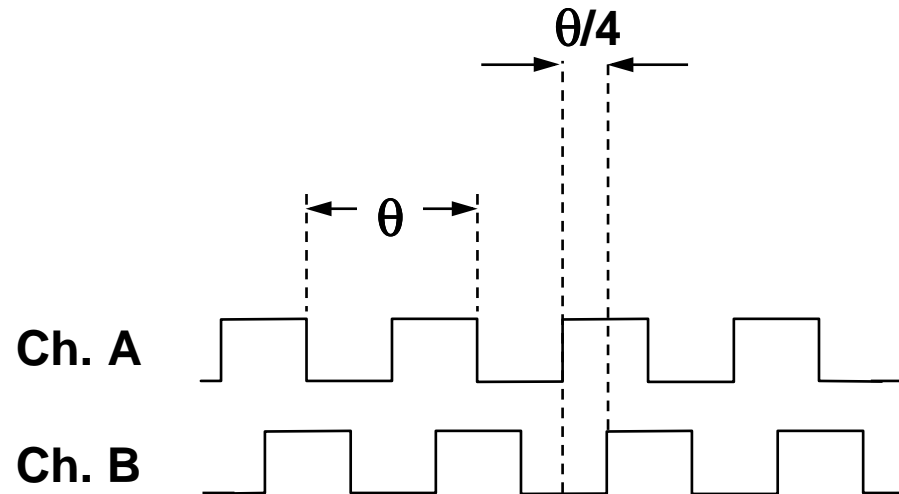
- Includes two quadrature encoder interface (QEI) modules
- Position integrator that tracks the encoder position
- Programmable noise filter on the inputs
- Velocity capture using built-in timer
- Interrupt generation on:
 - Index pulse
 - Velocity-timer expiration
 - Direction change
 - Quadrature error detection

Quadrature Encoder Interface (QEI)

A digital (angular) position sensor



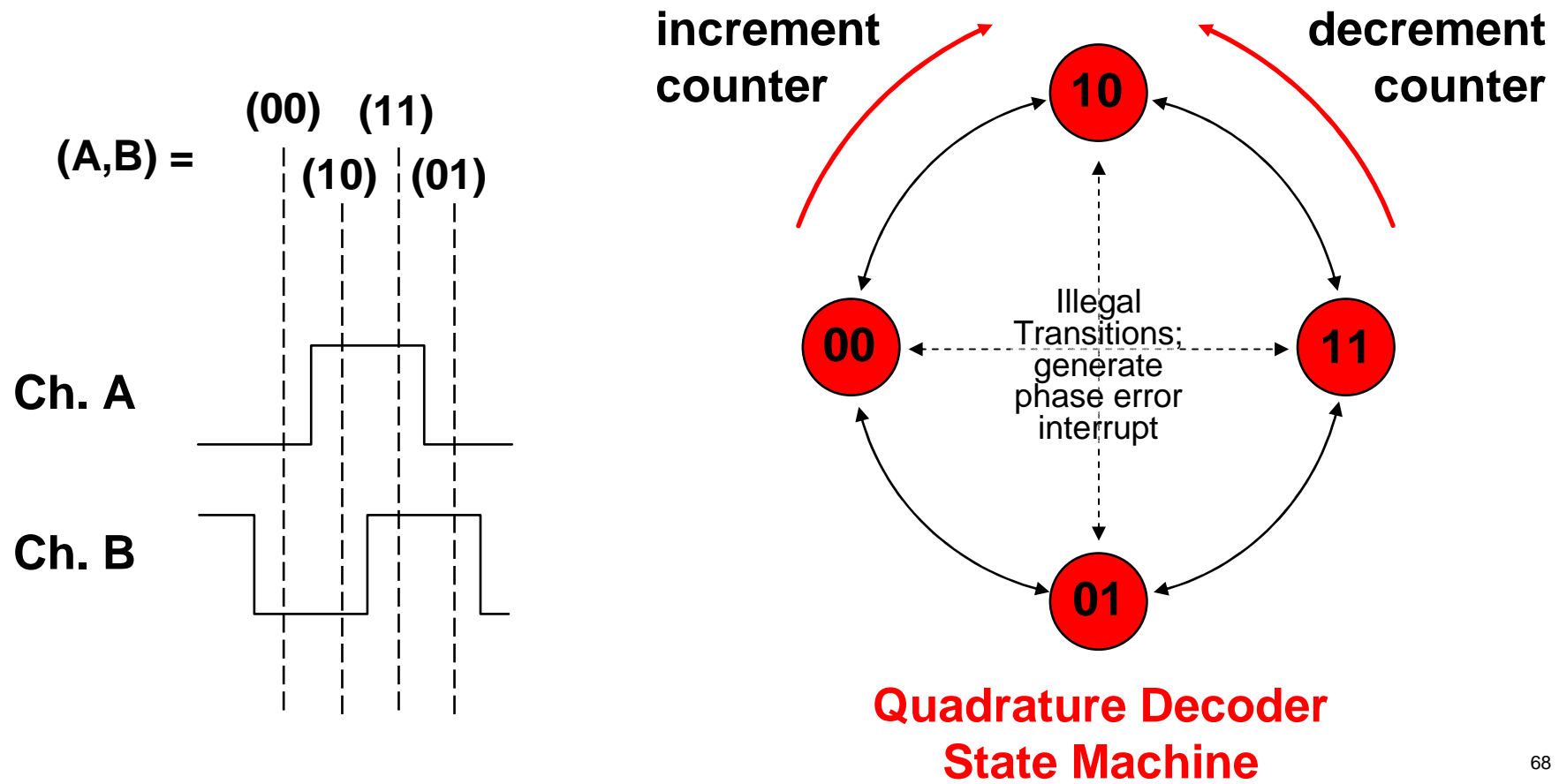
Incremental Optical Encoder



Quadrature Output from Photo Sensors

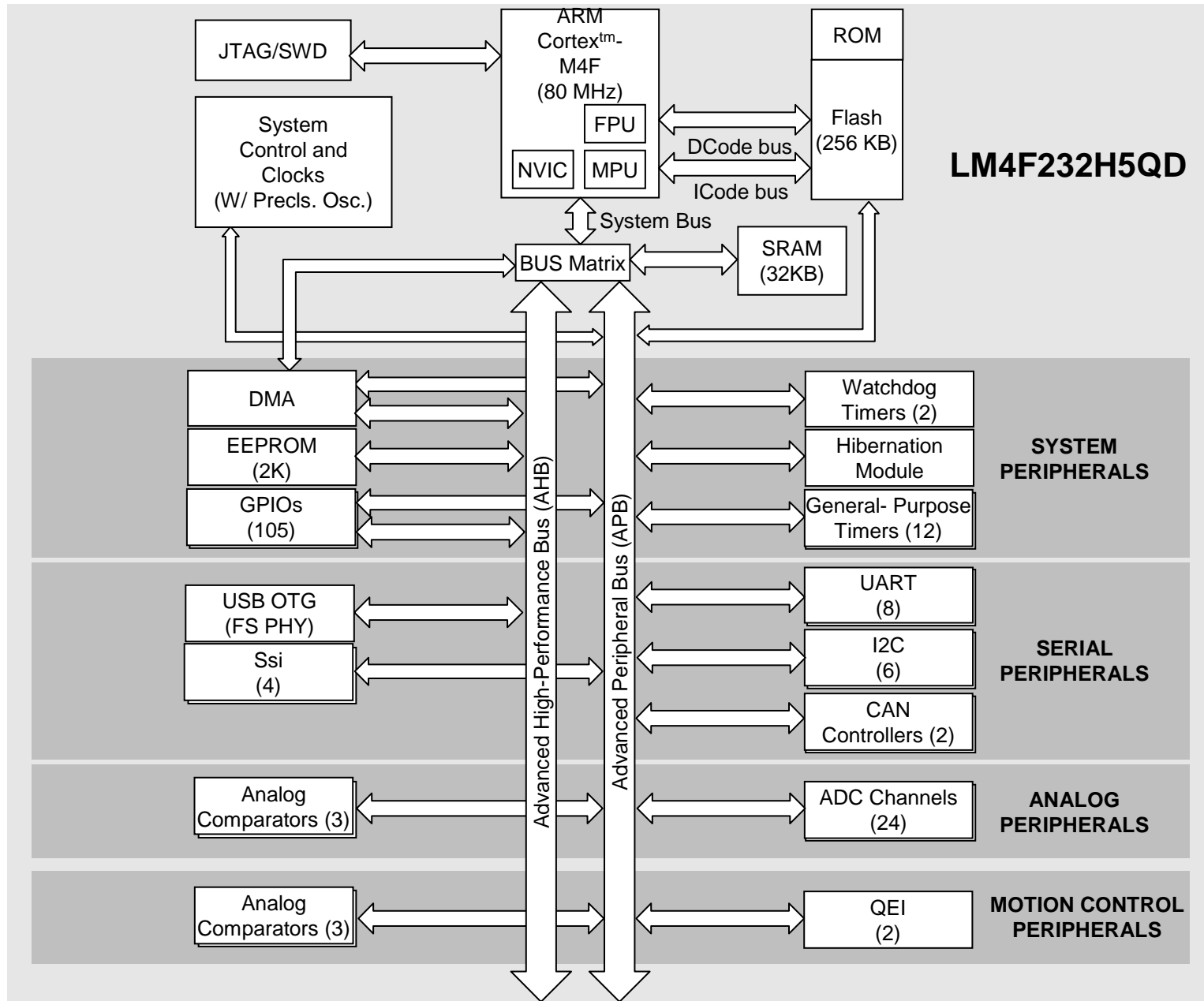
Quadrature Encoder Interface (QEI)

Position resolution is $\theta/4$ degrees



Reserve Slides

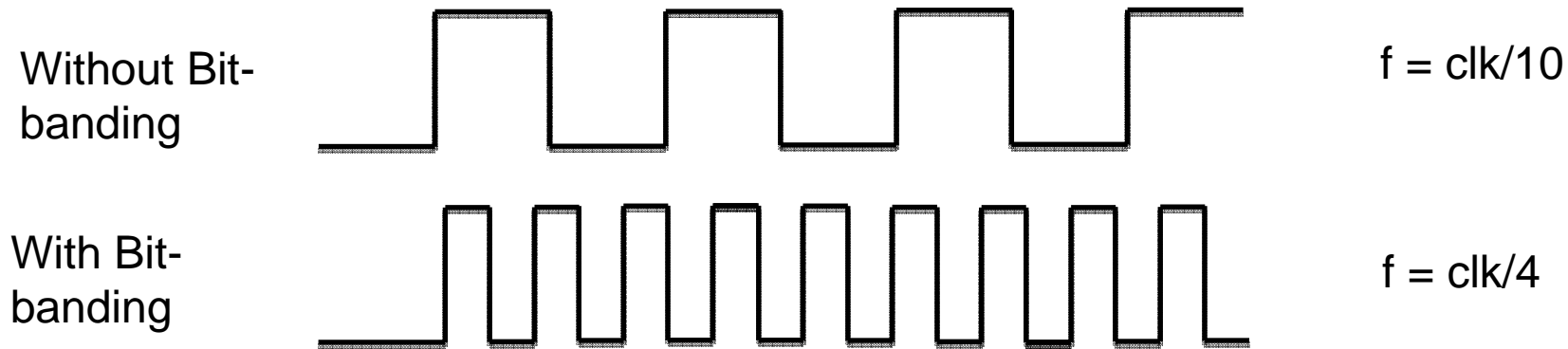
Stellaris architecture



Bit-banding

- Memory aliasing system that provides a more efficient way to change single bits in memory. (Atomic operations vs. read-modify-write)
- Bit-Banding regions exist for SRAM and some peripherals as well.
- Stellaris bit-banding is implemented with bit-masks, meaning that arbitrary groups of bits in a single memory location may be changed without penalty.
- Bit-banded reads are possible as well: only the selected bits will be read, and all others will appear as zeroes.

Bit-Banding Example: Writing to GPIOs



Without bit-banding, a read-modify-write sequence is required to toggle a single pin without affecting others. Bit-banding provides the same end behavior in only one instruction, yielding higher max toggle speeds.