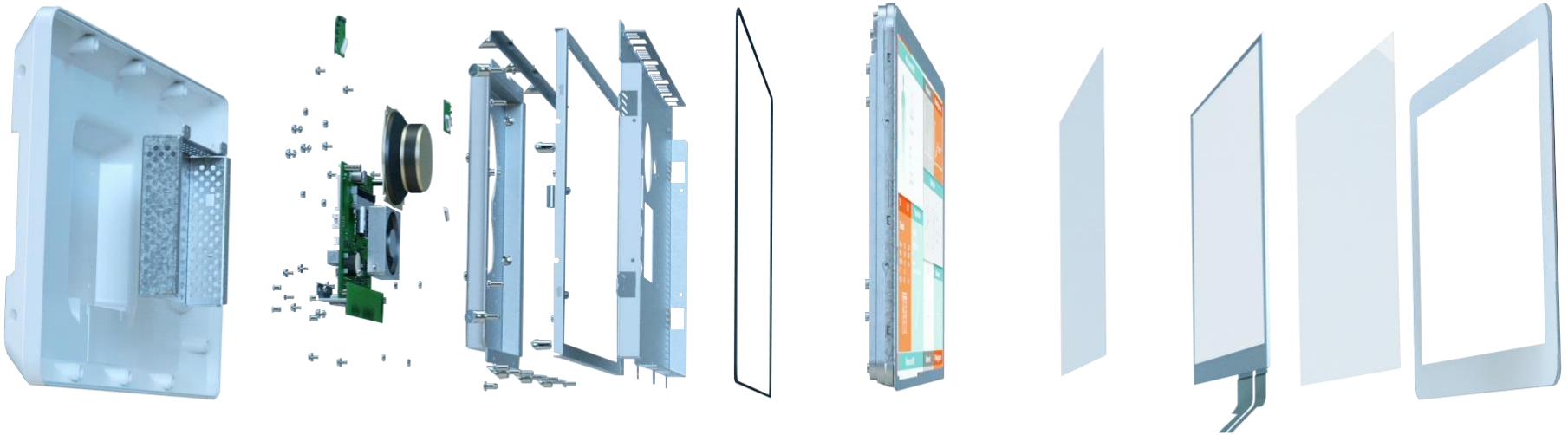
DATA MODUL

All Technologies. All Competencies. One Specialist.

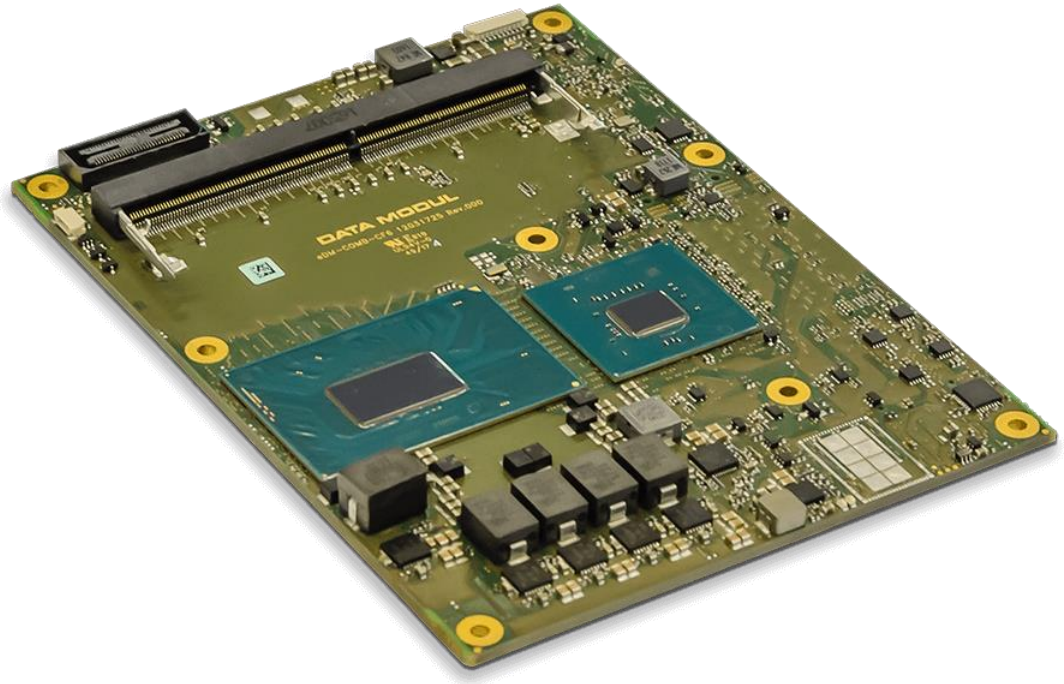


Wolfgang Meier – May 2019

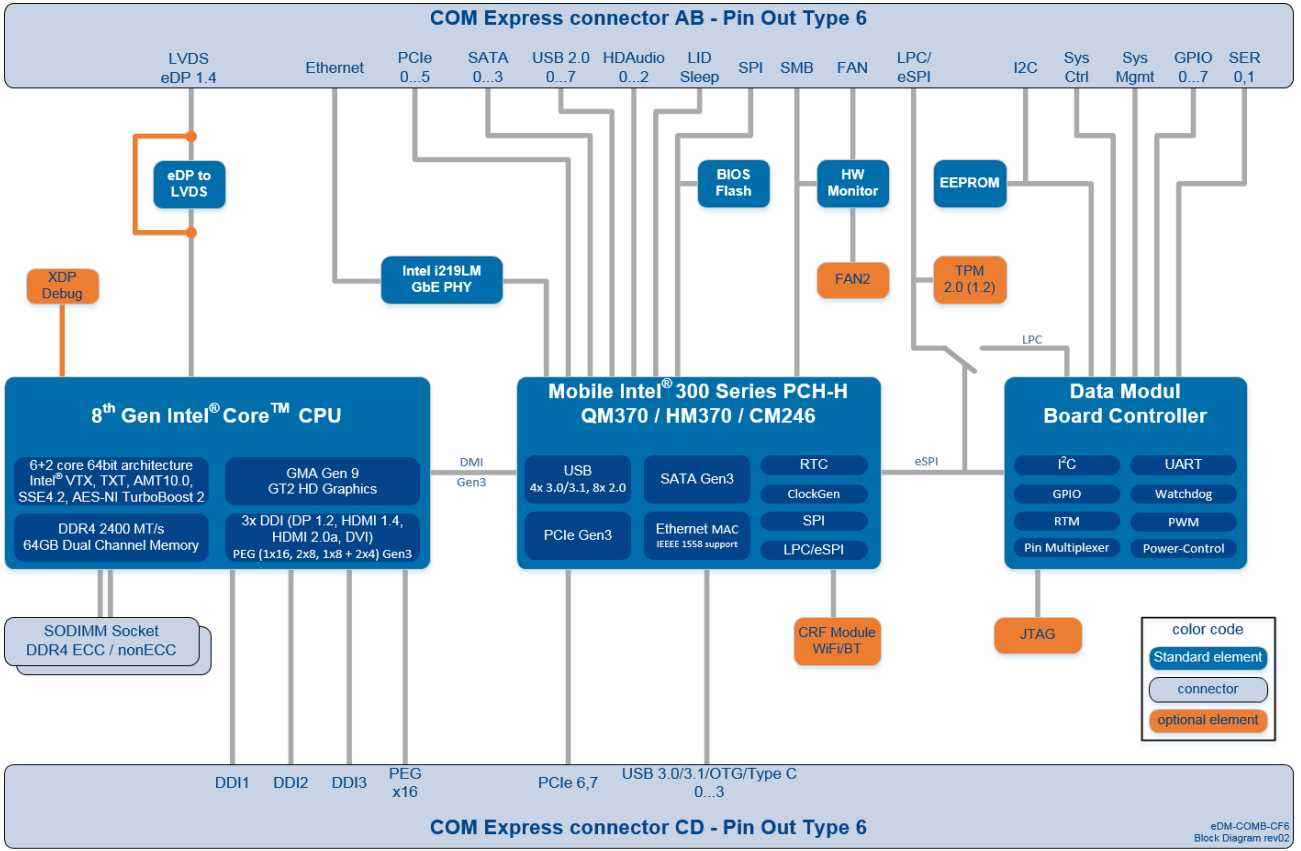
DATA MODUL

AGENDA

- Introduction & DMEC Overview
- DMEC Block Diagram
- Feature Details
- µC vs FPGA



eDM-COMB-CF6



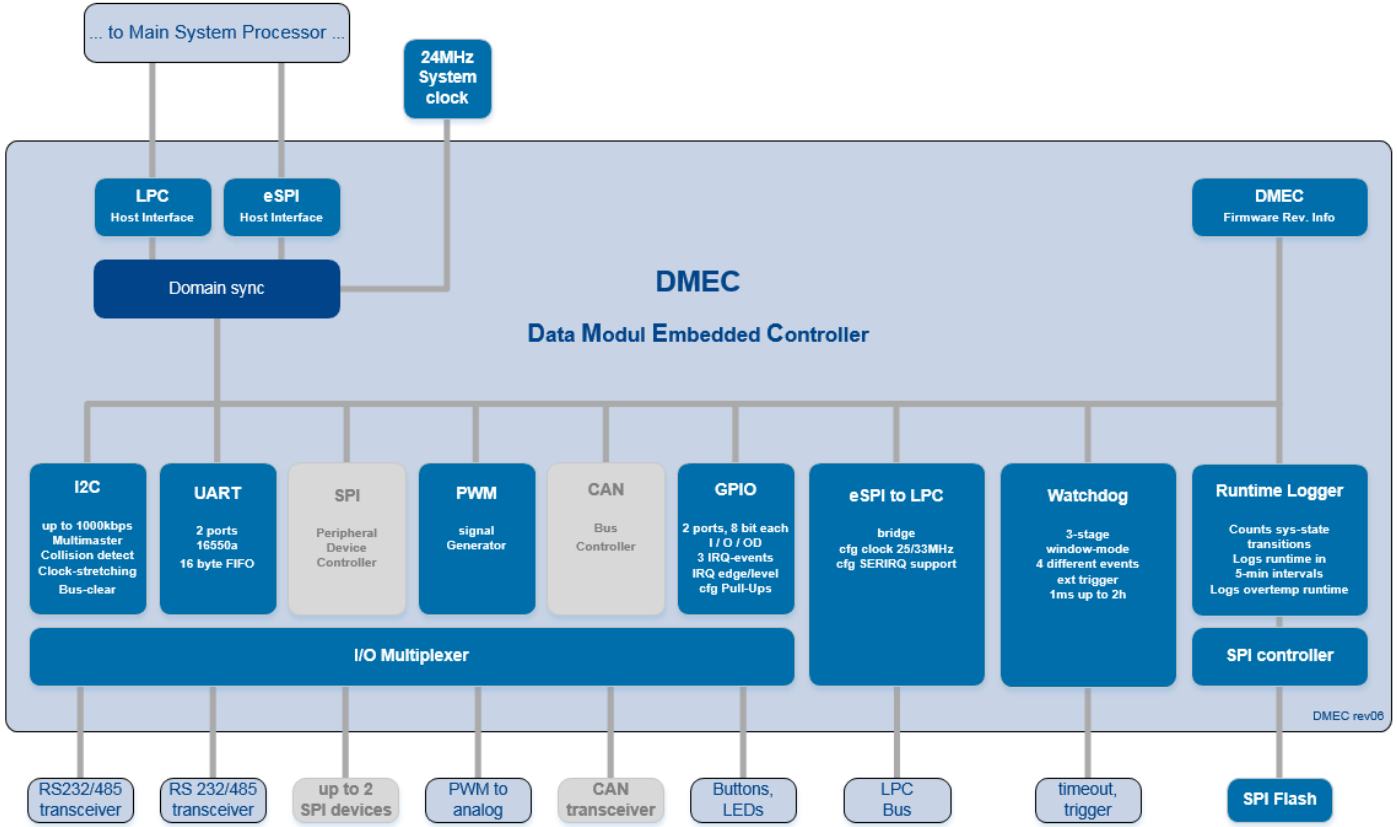
COMe Block Diagram

Data Modul Embedded Controller (DMEC)

- Provides functionality which may not be available in the chipset:
  - UART
  - Watchdog
  - I2C Controller
  - GPIOs
  - Running Time Logger
  - PWM Controller
  - CAN, SPI & other functions upon request
- Maintains HW and SW compatibility accross different platforms!!!

Implementation Overview

- Implemented in FPGA
  - Currently Intel MAX10, scalable @ 2k/4k/8k logical elements in identical package.
  - FPGA also manages power sequencing and glue logic
- Connected to the host either via LPC bus (past and current platforms) or eSPI bus (future platforms).
- Field-upgradable via Software from EFI shell.



DMEC Block Diagram

Host Interface (LPC or eSPI)

- LPC up to 33MHz LPC input clock, eSPI up to 50MHz/Quad I/O
- eSPI slave supports up to 50MHz/Quad I/O, peripheral and virtual wire channels
- Supports byte/word/dword I/O access
- Supports Serial IRQ (either direct or via eSPI virtual wire channel)
- Supports I/O mapped Index/Data (fixed address) and linear addressing (configurable address) for configuration registers
- No support for memory and FWH cycles
- No DMA support

eSPI to LPC Bridge

- Provides basic LPC support for future chipsets w/o native LPC bus
- LPC Clock 25MHz or 33MHz, configurable via BIOS setup
- LPC Serial IRQ support, configurable via BIOS setup
- No support for LPC memory and FWH cycles
- No LPC DMA support

GPIOs

- Up to two banks with 8 pins each
- Individually configurable as input or output
- Alternate functions configurable via Port Integration Module (PIM)
- Supports event generation on falling/rising/both edges and low level
- Event signaling via IRQ/NMI/SCI, event signaling configurable per bank.
- Separate Clear/Set bits, allows clear/set with a single I/O access instead of read/modify/write
- Configurable via BIOS setup (usage, direction and initial level)

I2C Controller

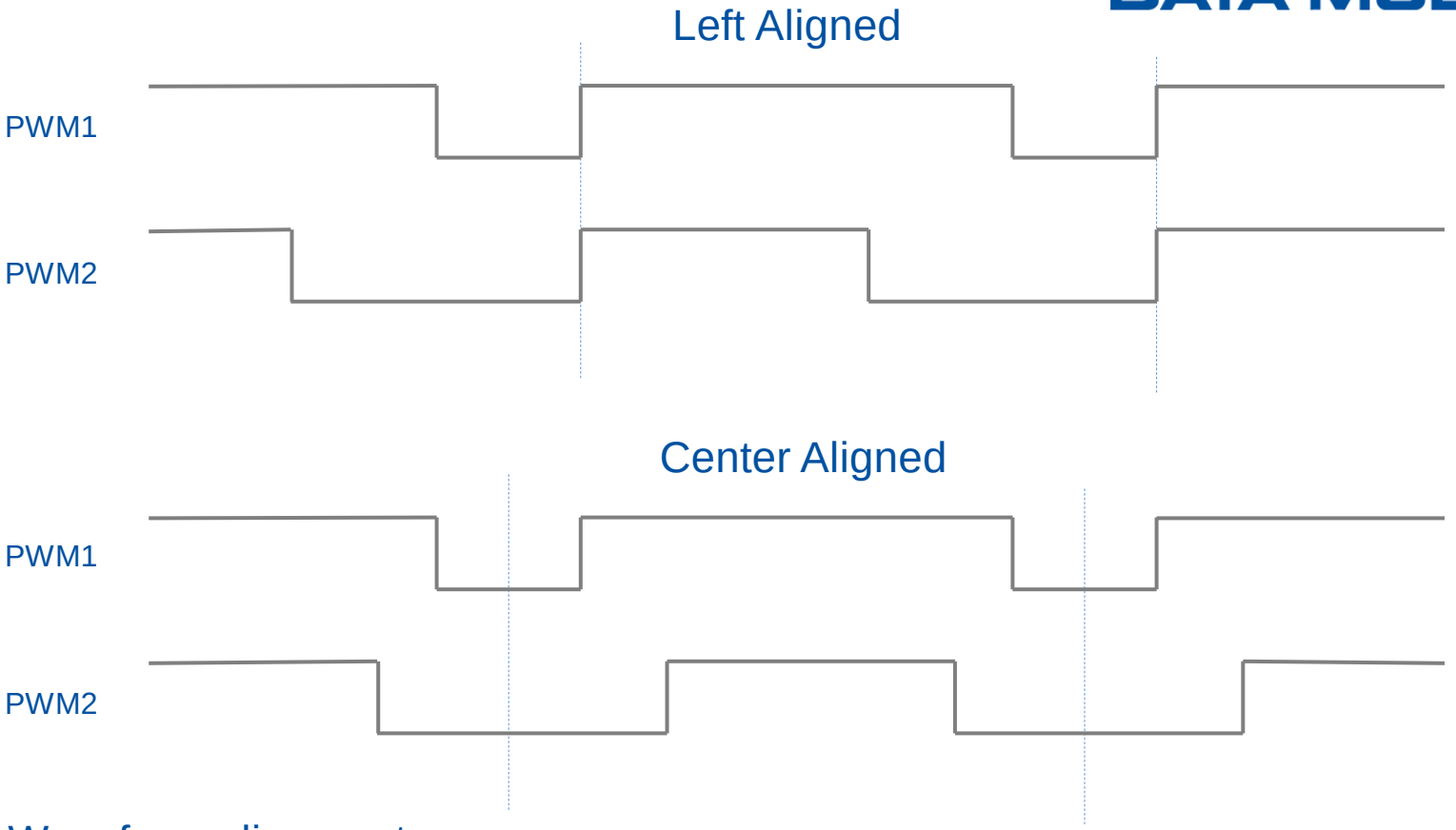
- I²C bus standard 3.0 compliant, Master only
- Interrupt supported, byte-wise transfer
- Multi-master capable
- Arbitration and bus collision and busy detection
- Supports clock stretching
- Bus-clear feature allows bus recovery if SDA held low by a slave
- Supports Fast Mode Plus
- I2C Clock (theoretically) scalable from 100bps up to 2Mbps
- Can control multiple I²C-busses through multiplexer
- Configurable via BIOS setup

PWM

- Two 8-bit channels, can be combined to one 16-Bit channel for better resolution
- Programmable period and double buffered duty cycle registers
- 50MHz source clock with programmable Pre-scalers allows for a wide granularity range from 20ns up to ~653µs

Pre-scaler	Granularity	Max. Period (center aligned)	Min. Period (left aligned, 8 steps)
1	20ns	2.621ms (20ns * 0xFFFF * 2)	160ns
128*65535	652.8µs	85.564s	5222µs

- Configurable polarity
- Programmable waveform alignment per channel: center or left aligned
- PWM pins configurable via BIOS setup



PWM: Waveform alignment

Running Time Logger

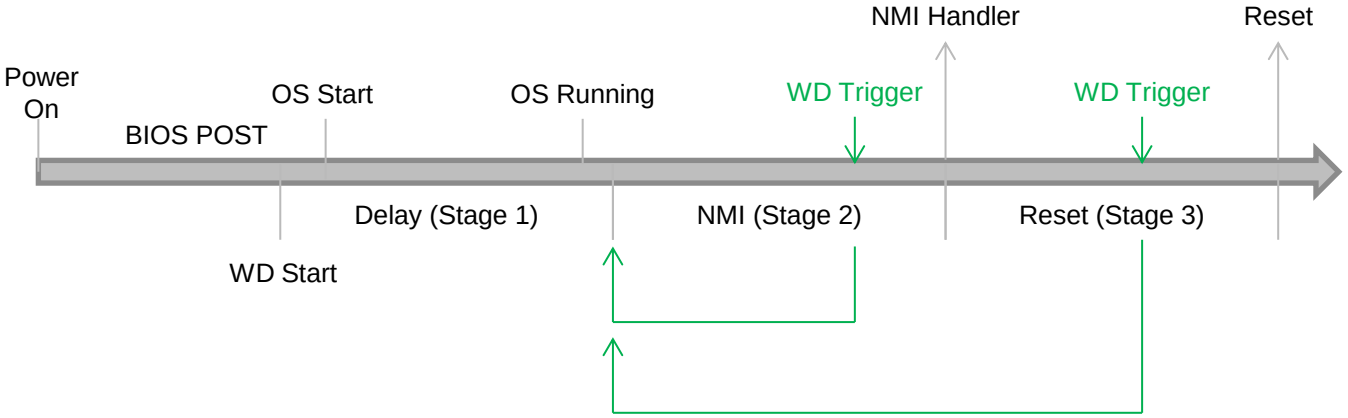
- Records running time and number of boot events
- 24-Bit counters for running time
  - Granularity 1 minute (~32years to overflow)
  - Saved to non-volatile storage every 5 minutes to prevent premature wear-out of external flash device.
- Optional second running time counter records out-of-specification (OOS) running time, i.e. time the board was running while one or more parameters were outside the specified range for normal operation
- 24-Bit counters for boot events (power-on/reset)
- Optional second boot counter can record custom boot events (e.g. GPIO toggle)
- Current values displayed in BIOS setup.

UART

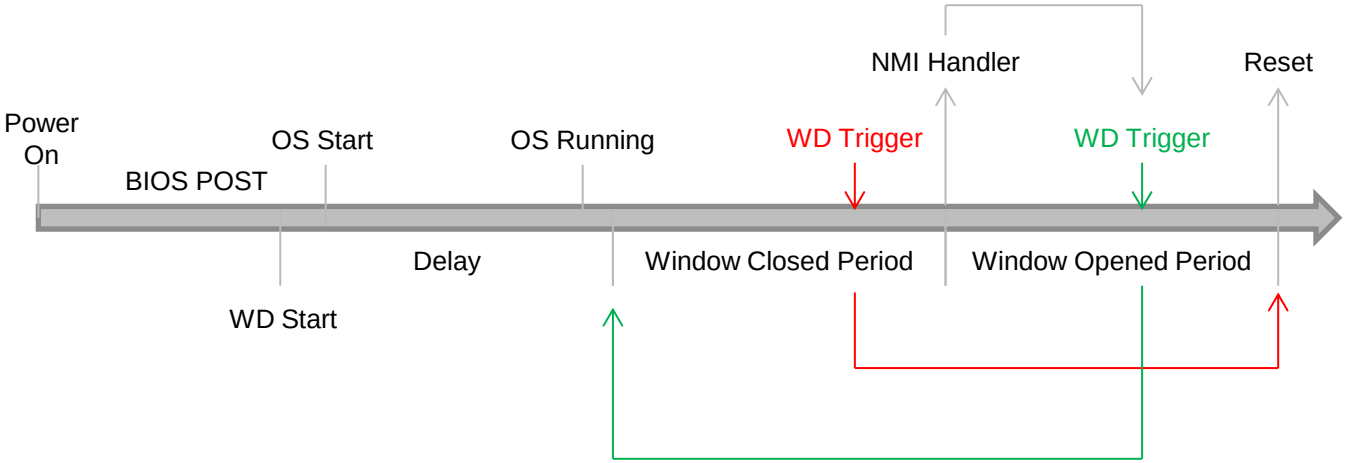
- Up to two UARTs supported, RX/TX only
- Handshake signals for UART0 optionally available on GPIO pins (configurable in BIOS setup)
- fully 16550A compliant register set, including 16 Byte FIFO
- Supports legacy resources 0x2F8/0x2E8/0x3F8/0x3E8 @ IRQ3/4/5/7/10/11
- Supports standard Baud rates with standard dividers
- Reported to OS via ACPI and detected as standard devices, no extra drivers required
- Configurable via BIOS setup

Watchdog

- Up to three configurable stages
- Supports standard timeout mode and Window mode for safety critical applications
- Different events (e.g. SMI, SCI, NMI, RESET, WDOUT) configurable per stage
- Supports auto-reload
- Supports register-lock
- Granularity 1ms (timeout 1ms - 65s) or 128ms (timeout 128ms - 140min)
- Configurable via BIOS setup



Watchdog: Standard mode example flow



Watchdog: Window mode example flow

Port Integration Module

- Allows multiplexing of DMEC pins with alternate functions
- Configuration handled by BIOS

Signal	Native Function	ALT1	ALT2	ALT3	Usage
GPIO Port B					
GPIO PB0	PB0	-	-	-	M
GPIO PB1	PB1	-	-	-	M
GPIO PB2	PB2	-	-	-	M
GPIO PB3	PB3	-	-	-	M
GPIO PB4	PB4	-	-	-	M
GPIO PB5	PB5	-	-	-	M
GPIO PB6	PB6	-	-	-	M
GPIO PB7	PB7	-	-	-	M
GPIO Port A					
GPIO PA0	EXT_GPIO0	SPI0_MOSI	-	-	C
GPIO PA1	EXT_GPIO1	SPI0_MISO	-	-	C
GPIO PA2	EXT_GPIO2	SPI0_CLK	-	UART0_DSR	C
GPIO PA3	EXT_GPIO3	SPI0_CS0#	-	-	C
GPIO PA4	EXT_GPIO4	SPI0_CS1#	PWM0	UART0_CTS	C
GPIO PA5	EXT_GPIO5	WDTRIG #	PWM1	UART0_RTS	C
GPIO PA6	EXT_GPIO6	I2C2_SCL	-	UART0_DTR	C
GPIO PA7	EXT_GPIO7	I2C2_SDA	-	-	C
UART 0/1					
TxD0	TxD0	-	-	-	C
RxD0	RxD0	-	-	-	C
TxD1	TxD1	CAN0_TxD	-	-	C
RxD1	RxD1	CAN0_RxD	-	-	C
SPI Controller					
SPI1_MOSI	SPI1_MOSI	-	-	-	M
SPI1_MISO	SPI1_MISO	-	-	-	M
SPI1_CLK	SPI1_CLK	-	-	-	M
SPI1_SC0#	SPI1_SC0#	-	-	-	M
Watchdog					
WDT	WDT	-	-	-	C
WDTRIG#	WDTRIG#	-	-	-	Q7
I2C Controller					
I2C0_SDA	I2C0_SDA		-	-	C
I2C0_SCL	I2C0_SCL		-	-	C
I2C1_SDA	I2C1_SDA	-	-	-	M
I2C1_SCL	I2C1_SCL	-	-	-	M

DMEC PIM native and alternate functions

DMEC General Registers

- Feature support can be probed via register flags, allows generic SW implementation
- Versioning information for DMEC core and FPGA instance as well as individual devices
- Centralized configuration of DMEC devices (I/O addresses and IRQs) in a conflict-free manner
- Centralized identification of interrupt sources

uC vs FPGA

- uC Pros:
  - May easily implement complex functionality.
  - Can be maintained by any software developer
  - May support host CPU offloading.

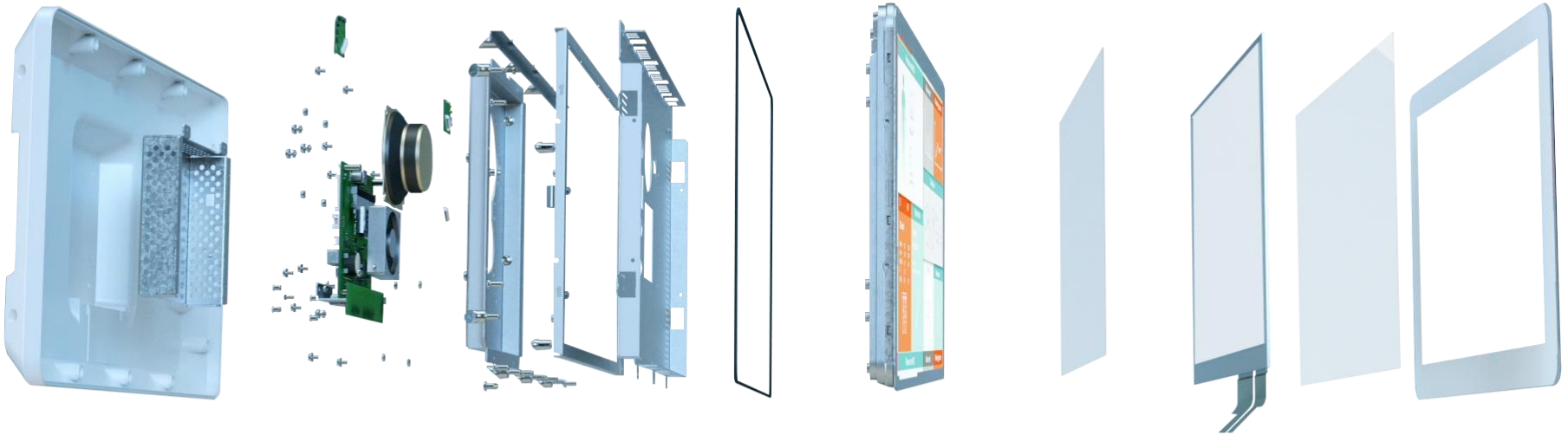
- FPGA Pros:
  - Supports implementation of legacy interfaces (e.g. 16550 UART @ legacy Addresses).
  - Functionality is always deterministic (incl. control of power timings)
  - Adding new functionality does not impact existing functions

OS Support

- EAPI drivers available for Windows and Linux

DATA MODUL

All Technologies. All Competencies. One Specialist.



Thank you!