



Hercules™ Safety Microcontrollers

1 Day Safety MCU Workshop



Agenda:

- Introduction
- What is Functional Safety & Safety Standards Overview
- IEC 61508 & ISO 26262 Safety Standards
- Safety System Architectures
- SafeTI™
- Hercules Safety Concept
- Development Tools: Hardware kits, Software tools
- Printed Circuit Board Design Considerations
- **Lab 1: Hercules™ Safety MCU Demos**
- Hercules™ Architecture
- Embedded Flash Memory tools
- Real Time Interrupt (RTI)
- Vectored Interrupt Manager (VIM)
- Direct Memory Access (DMA)
- General-purpose I/O (GIO) & NHET Timer Co-processor
- **Lab 2: PWM Generation using the NHET & Clock Monitor**
- Communication Interfaces: Multi-Buffered Serial Peripheral Interface (MibSPI), CAN, FlexRay, EMAC, USB, UART, LIN
- External Memory Interface (EMIF) / Parameter Overlay (POM)
- Multi-buffered Analog-to-Digital Converter (MibADC)
- **Lab 3: MibADC Light Sensor & SCI Communication**
- Additional Hercules Information: Web, Forum, WIKI & Training



Hercules™ Software Install Instructions

Required Software

Three software titles will be used during the lab exercises of this workshop: Hercules Safety MCU Demos, Code Composer Studio v5.x, HALCoGen

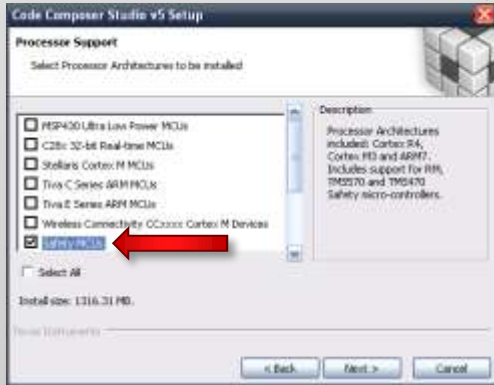
Software Download & Install Location:

• Hercules Safety MCU Demos

- The Demo software can be downloaded here:
[Hercules Safety MCU Demos LINK](#)
- A standard install of the software is required

• Code Composer Studio

- CCS can be downloaded here:
[CCSv5 Download LINK](#)
- A full install (Complete Feature Set) is acceptable, but at a minimum a custom install where the 'Safety MCUs' processor support is selected will be necessary.



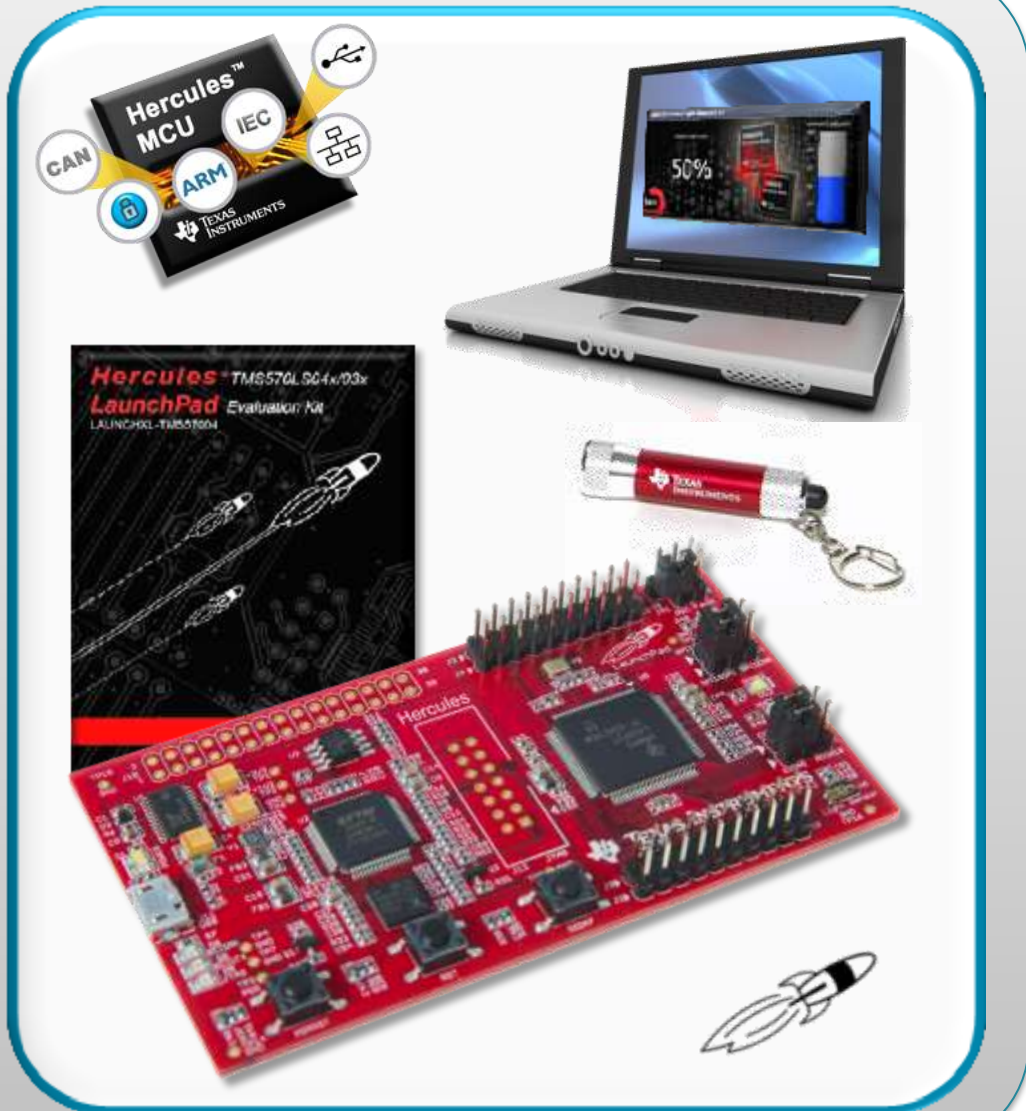
- Select the FREE LICENSE option the first time CCS is run:



• HALCoGen




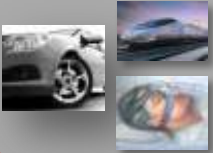





- HALCoGen can be downloaded here:
[HALCoGen Download LINK](#)
- A standard install of the software is required



Hercules™ Safety MCU: Introduction

TI Embedded Processing Portfolio

Microcontrollers				Applications Processors / DSP		
16-bit	32-bit Real-time	32-bit M4F ARM	32-bit R4F ARM [®]	32-bit ARM+	32-bit ARM+DSP	DSP
ARM Core Offerings						
<p>MSP430</p> <hr/> <p>Ultra-low Power</p> <hr/> <p>Up to 25 MHz</p> <hr/> <p>Flash 1 KB to 256 KB</p> <p>Analog I/O, ADC LCD, USB, RF</p> <hr/> <p>Measurement, Sensing, General Purpose</p> <hr/> 	<p>C2000</p> <hr/> <p>Fixed & Floating Point</p> <hr/> <p>Up to 300 MHz</p> <hr/> <p>Flash 32 KB to 512 KB</p> <p>PWM, ADC, CAN, SPI, I²C</p> <hr/> <p>Motor Control, Digital Power, Lighting</p> <hr/> 	<p>Tiva™ C Series</p> <hr/> <p>Industry Std Low Power</p> <hr/> <p><100 MHz</p> <hr/> <p>Flash Up to 256 KB</p> <p>USB, ENET, ADC, PWM, CAN</p> <hr/> <p>Host Control</p> <hr/> 	<p>Hercules™ TMS570/RM4</p> <hr/> <p>Floating Point</p> <hr/> <p>Over 350 DMIPS</p> <hr/> <p>Flash Up to 3 MB</p> <hr/> <p>Timer co-proc ENET, ADC, CAN</p> <hr/> <p>Safety Critical Transportation, Industrial & Medical</p> <hr/> 	<p>ARM9 ARM Cortex-A8</p> <hr/> <p>Industry-Std Core, High-Perf GPP</p> <hr/> <p>Accelerators</p> <hr/> <p>MMU</p> <hr/> <p>USB, LCD, MMC, EMAC</p> <hr/> <p>Linux/WinCE User Apps</p> <hr/> 	<p>ARM9/Cortex-A8 plus C64x+</p> <hr/> <p>Industry-Std Core + DSP for Signal Proc.</p> <hr/> <p>4800 MMACS/ 1.07 DMIPS/MHz</p> <hr/> <p>MMU, Cache</p> <hr/> <p>VPSS, USB, EMAC, MMC</p> <hr/> <p>Lin/Win O/S + Video, Imag, MM</p> <hr/> 	<p>C55x, C64x+ C647x</p> <hr/> <p>Leadership DSP Performance</p> <hr/> <p>24,000 MMACS</p> <hr/> <p>Up to 3 MB L2 Cache</p> <hr/> <p>1G EMAC, SRIO, DDR2, PCI-66</p> <hr/> <p>Comm, WiMAX, Industrial/ Medical Imaging</p> <hr/> 



TI Hercules™ MCU Platform

ARM® Cortex™ Based Microcontrollers



Hercules™
Safety MCU
Platform



RM

Industrial and Medical Safety MCUs



- Industrial Applications
- Medical Applications
- -40 to 105°C Operation
- ENET, USB, CAN & UART
- Developed to Safety Standards
 - IEC 61508 SIL-3
- Cortex-R – over 350 DMIPs

TMS570

Transportation and Safety MCUs



- Transportation Applications
- Automotive Q100 Qualification
- -40 to 125°C Operation
- FlexRay, ENET, CAN, LIN/UART
- Developed to Safety Standards
 - ISO 26262 ASIL-D
 - IEC 61508 SIL-3
- Cortex-R – over 280 DMIPs

TMS470M

Value Line Transportation & Safety MCUs



- Transportation Applications
- Automotive Q100 Qualification
- -40 to 125°C Operation
- CAN, LIN/UART Connectivity
- Supports Safety for
 - IEC 61508 Systems
- Cortex-M – to 100 DMIPs

Hercules™ Safety MCU Applications



Aerospace & Railway

Avionics / Autopilot

Flight Control

Anti-Skid Control

Motor Control

Communications Gateway

Industrial

Industrial Motor Control

Wind Power

Manufacturing / Robotics

Elevator Escalator

Industrial Automation / PLC

Sensor & Communications Gateway

Solar Power

Automotive

Braking / Stability Control

Airbag

Hybrid & Electric Vehicles

Radar / Collision Avoidance (ADAS)

Active Suspension

Electric Power Steering

Chassis / Domain Control



Infusion Pumps

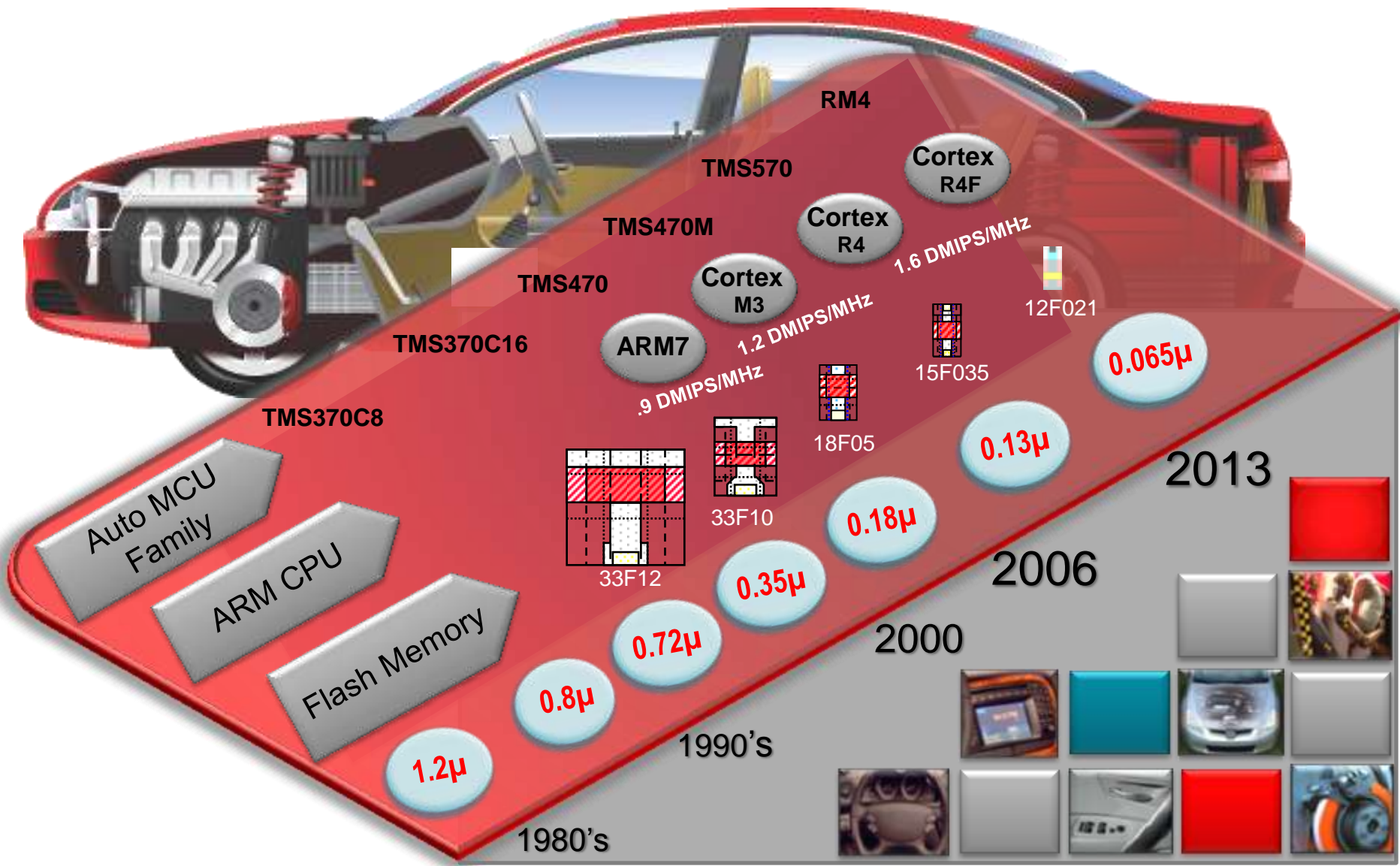
Oxygen Concentrators

Respirators

Anesthesia

Medical

TI Transportation & Safety MCU Technology



Hercules™ RM Cortex™-R Roadmap

2012

2013

2014

High

RM48L9x – 220MHz R4F
3MB Flash, 256kB RAM
SafeTI ISO & IEC

RM48L5x – 200MHz R4F
2MB Flash, 192kB RAM
SafeTI ISO & IEC

Features:

Lock Step Architecture QEP/PWM Ethernet

CAN CAN USB

ISO ISO 13849 IEC IEC 61508 SafeTI

Next Gen High
SafeTI ISO & IEC

Mid

RM46L8x – 220MHz R4F
1.25MB Flash, 192kB RAM
SafeTI ISO & IEC

RM46L4x – 200MHz R4F
1MB Flash, 128kB RAM
SafeTI ISO & IEC

Next Gen Mid
SafeTI ISO & IEC

Low

Production

Sampling

Development

RM42x – 100MHz R4
384kB Flash, 32kB RAM
SafeTI ISO & IEC

Next Gen Low
SafeTI ISO & IEC

RM46x Block Diagram

Dual Core Lockstep ARM Cortex-R4F w/ Floating Point

Features



Performance / Memory

- Up to 220 MHz ARM Cortex-R4F w/ Floating Point
- Up to 1.25MB Flash and 192KB Data SRAM w/ECC
- Dedicated 64KB Data Flash (EEPROM Emulation)
- 16 Channel DMA

Safety

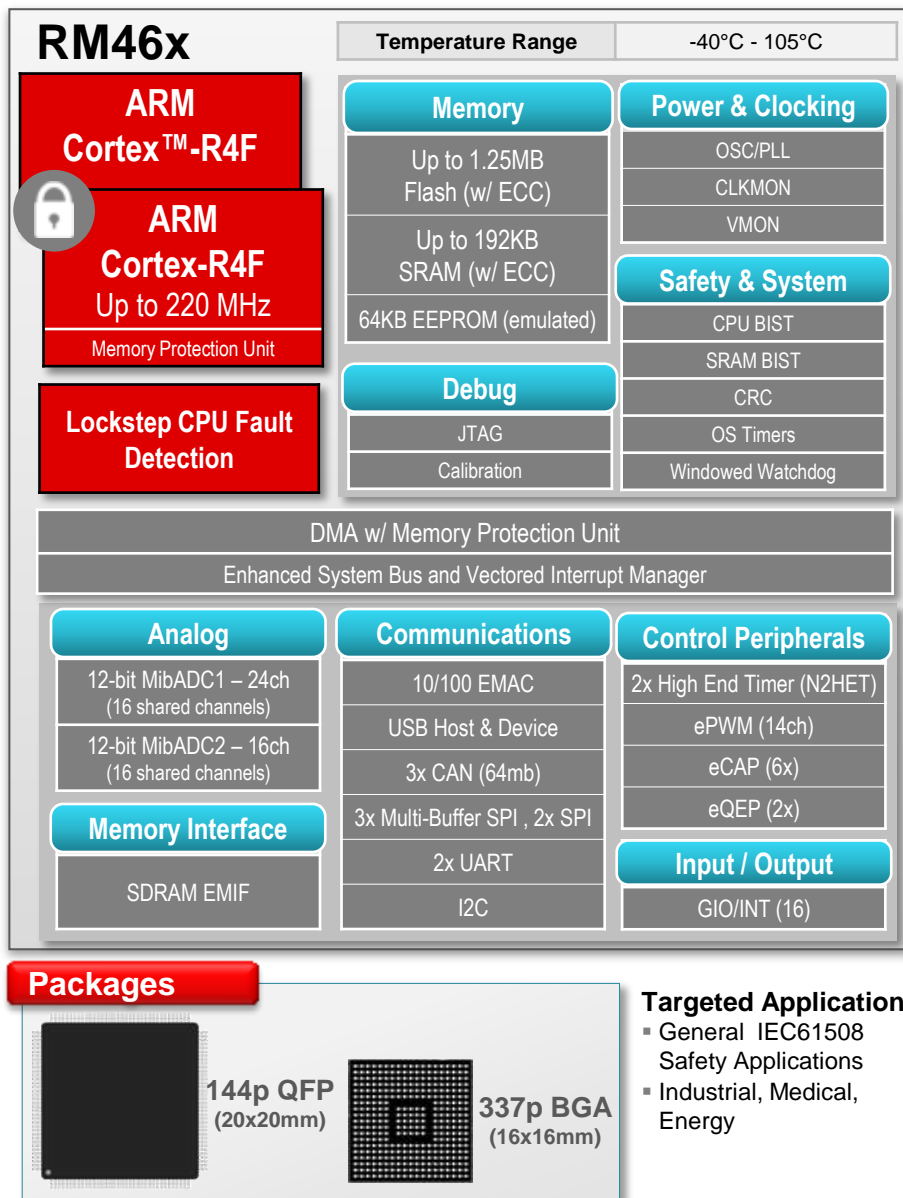
- Dual CPUs in Lockstep
- CPU Logic Built in Self Test (LBIST)
- Up to 12 CPU MPU regions
- Flash & RAM w/ ECC (w/ bus protection)
- Memory Built-in Self Test (PBIST)
- Cyclic redundancy checker module (CRC)
- Select peripheral RAMs protected by Parity

Communication Networks

- 10/100 MAC
- USB: Host and Device
- 3 CAN Interfaces
- 5 SPI (3 Multi-Buffered)
- 2 UART, 1 I2C

Enhanced I/O Control

- 2x High End Timer Coprocessor (N2HET) w/DMA
 - Up to 44 pins plus 6 monitor channels
 - Pins can be used as Hi-Res PWM or Input Capture
- Motor Control Timers
 - 7x ePWM (14 ch), 6x eCAP, 2x eQEP
- 2 x12-bit Multi-Buffered ADC
 - 24 total input channels (16 shared)
 - Calibration and Self Test
- Up to 101 GPIO pins (16 dedicated)



Hercules™ TMS570 Roadmap

2012

2013

2014

High

TMS570LS31x – 180MHz R4F
 3MB Flash, 256kB RAM
 SafeTI ISO & IEC

TMS570LS21x – 180MHz R4F
 2MB Flash, 192kB RAM
 SafeTI ISO & IEC

Features:

Lock Step Architecture QEP/ePWM Ethernet

CAN CAN FlexRay

ISO ISO 26262 IEC IEC 61508 SafeTI

Next Gen High
 SafeTI ISO & IEC

Mid

TMS570LS12x – 180MHz R4F
 1.25MB Flash, 192kB RAM
 SafeTI ISO & IEC

TMS570LS11x – 180MHz R4F
 1MB Flash, 128kB RAM
 SafeTI ISO & IEC

Next Gen Mid
 SafeTI ISO & IEC

Low

Production

Sampling

Development

TMS570LS04x – 80MHz R4
 384kB Flash, 32kB RAM
 SafeTI ISO & IEC

TMS570LS03x – 80MHz R4
 256kB Flash, 24kB RAM
 SafeTI ISO & IEC

Next Gen Low
 SafeTI ISO & IEC

TMS570LS31x/21x Block Diagram

Dual Core Lockstep ARM Cortex-R4F w/ Floating Point

Features



Performance / Memory

- Up to 180 MHz ARM Cortex-R4F w/ Floating Point
- Up to 3MB Flash and 256KB Data SRAM
- Dedicated 64KB Data Flash (EEPROM Emulation)
- 16 Channel DMA

Safety

- Dual CPUs in Lockstep
- CPU Logic Built in Self Test (LBIST)
- Up to 12 CPU MPU regions
- Flash & RAM w/ ECC (w/ bus protection)
- Memory Built-in Self Test (PBIST)
- Cyclic redundancy checker module (CRC)
- Select peripheral RAMs protected by Parity

Communication Networks

- 10/100 MAC
- FlexRay w/DMA
- 3 CAN Interfaces
- 5 SPI (3 Multi-Buffered)
- 2 UART (1 LIN capable), 1 I2C

Enhanced I/O Control

- 2x High End Timer Coprocessor (N2HET) w/DMA
 - Up to 44 pins plus 6 monitor channels
 - Pins can be used as Hi-Res PWM or Input Capture
- 2 x12-bit Multi-Buffered ADC
 - 24 total input channels (16 shared)
 - Calibration and Self Test
- Up to 120 GPIO pins (16 dedicated)

TMS570LS31x

Temperature -40°C - 125°C AEC Q100

ARM Cortex™-R4F

ARM Cortex-R4F
Up to 180 MHz

Memory Protection Unit

Lockstep CPU Fault Detection

Memory

Up to 3MB
Flash (w/ ECC)

Up to 256KB
SRAM (w/ ECC)

64KB EEPROM (emulated)

Debug

JTAG

ETM, RTP, DMM

Power & Clocking

OSC/PLL

CLKMON

VMON

Safety & System

CPU BIST

SRAM BIST

CRC

OS Timers

Windowed Watchdog

DMA w/ Memory Protection Unit

Enhanced System Bus and Vectored Interrupt Manager

Analog

12-bit MibADC1 – 24ch
(16 shared channels)

12-bit MibADC2 – 16ch
(16 shared channels)

Memory Interface

SDRAM/ASYNCEMIF

Communications

10/100 EMAC

2ch FlexRay

3x CAN (64mb)

3x Multi-Buffer SPI, 2x SPI

2x UART (1 LIN capable)

I2C

Control Peripherals

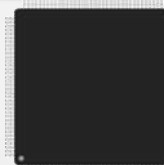
High End Timer 1
(N2HET1 = 32ch)

High End Timer 2
(N2HET2 = 14ch)

Input / Output

GIO/INT (16)

Packages



144p QFP
(20x20mm)



337p BGA
(16x16mm)

Targeted Applications

- IEC 61508 and ISO 26262 Safety Applications
- Automotive, Rail, Aerospace (COTS), Off Road

What is Functional Safety & Safety Standards Overview

International Functional Safety Standards

Automotive and Transportation



HEV/EV Cars



Radar / Collision Avoidance (ADAS)



Active suspension, ABS, electric power steering, airbag and more!



Railway Systems



Aerospace Systems



- Safety critical systems are everywhere
- Systems need to manage hazardous failures
- Many systems need to be safety-certified

Industrial and Medical



Sensor & communications gateway



Manufacturing, robotics, industrial automation, motor control



Wind & Solar Power



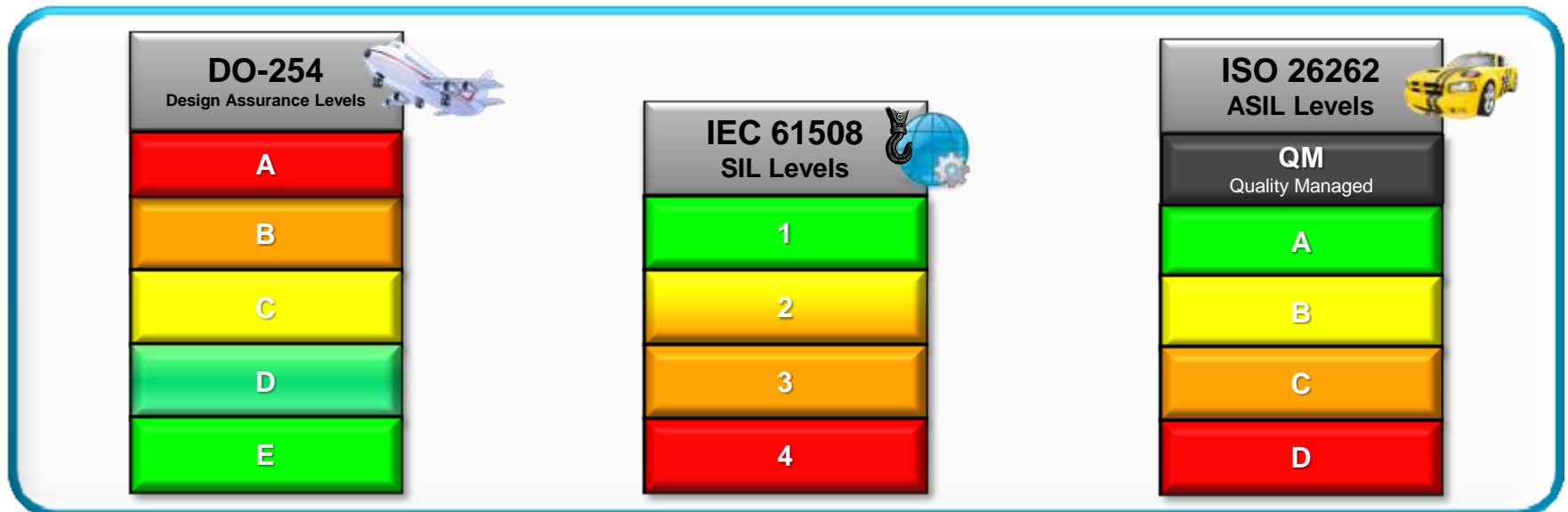
Anesthesia machines, respirators, ventilators, oxygen concentrators

What is Functional Safety?



- **Basic Functional Safety Concepts:**

- All systems will have some inherent, quantifiable failure rate. It is not possible to develop a system with zero failure rate.
- For each application, there is some tolerable failure rate which does not lead to unacceptable risk.
- Acceptable failure rates vary per application, based on the potential for direct or indirect physical injury in the event of system malfunction.
- Categories can be developed to quantify similar levels of risk. These are known as *Safety Integrity Levels*, or *SILs*.



Functional Safety Definitions



IEC 61508 Definition:

- *Safety* is the freedom from unacceptable risk of physical injury or of damage to the health of people, either directly, or indirectly as a result of damage to property or to the environment.
- *Functional Safety* is part of the overall safety that depends on a system or equipment operating correctly in response to its inputs.

ISO 26262 Definition:

- Absence of unacceptable risk due to hazards caused by malfunctioning behavior of electrical and/or electronic systems

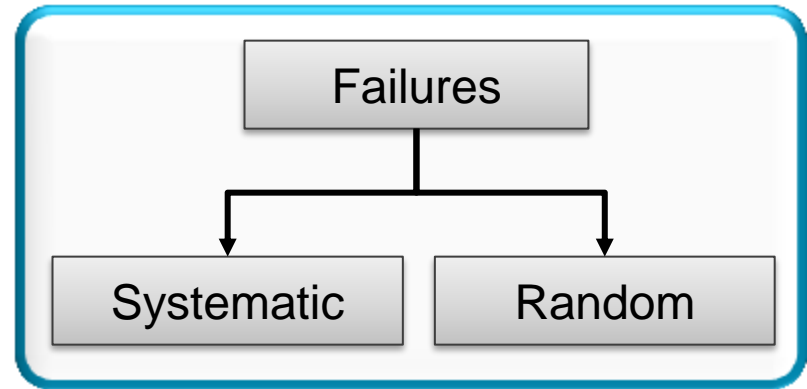
Other Safety Concepts & Definitions



- **Fault:**
 - Operational issue in a system which may lead to a failure.
- **Failure:**
 - Result of a fault which leads to an inability to execute safety critical functionality
- **Fault Tolerance :**
 - Ability to continue safe operation after a fault.
- **Fail Safe System:**
 - System where a fault which may lead to failures is detected and the system is put into a safe state such that faults may not propagate to other systems
- **Fail Functional/Operational System:**
 - System where a fault which may lead to failures is detected and the system can continue operation without loss of safety function.
- **Reliability**
 - Ability to execute operations in system without failure (*generally independent of consideration for a safety function*)
- **Availability**
 - Amount of time in which a safety function is available divided by total system operation time. Systems with high reliability and fail functional systems tend to have higher availability than fail safe systems
- **Security**
 - Ability to detect, resist, or prevent tampering with product functionality.

Safety Failures and their Causes

- Failures in a functional safety system can be broadly classified into two categories: Systematic and Random failures



- Systematic Failures**

- Result from a failure in design or manufacturing
- Often a result of failure to follow best practices
- Rate of systematic failures can be reduced through continual and rigorous process improvement

- Random Failures**

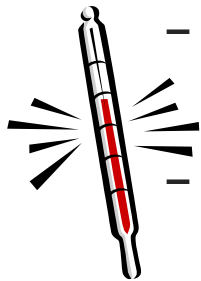
- Result from random defects inherent to process or usage condition
- Rate of random failures cannot generally be reduced; focus must be on the detection and handling of random failures in the application.

Note: *Software failures are considered to be systematic*

Functional Safety vs. Quality/Reliability

- High quality and reliability do not guarantee safety.
 - Methods to ensure quality and reliability have high overlap to methods used to manage systematic safety failures.
 - Requirements to manage safety of random hardware failures in applications typically do not overlap quality, reliability and security requirements.
-

When faced with a potential system over temperature fault that causes wiring to melt resulting in a system failure:



- A reliability engineer would approach the problem by designing the system with high temperature wiring
- A functional safety engineer would approach the problem by designing the system to detect the over temperature condition and placing the system into a safe state before the wiring could melt.

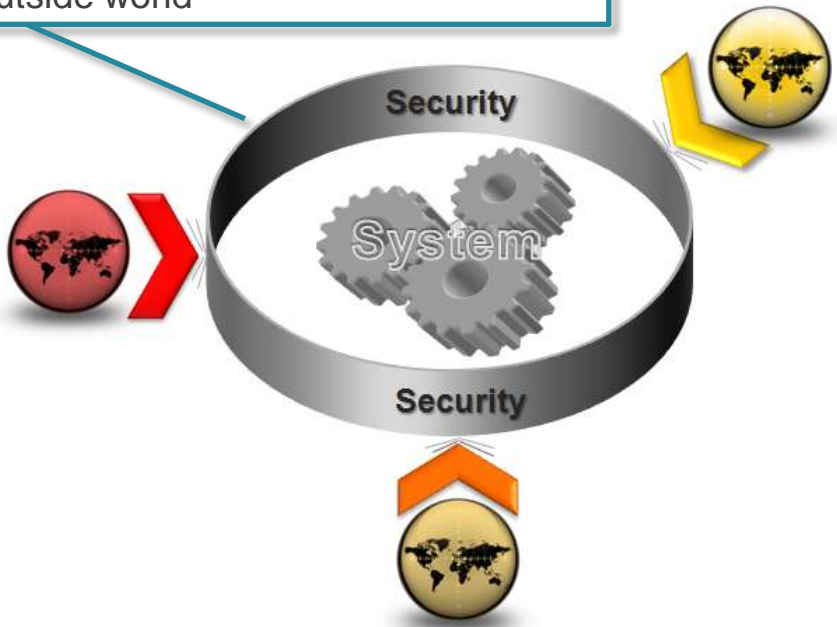
Functional Safety vs. Security



- A secure system is capable of detecting, resisting or preventing tampering from the outside world
- A functionally safe system is capable of detecting faults and preventing damage to the outside world

Secure System:
Avoids system manipulation from the outside world

Functional Safe System:
Does not create an unacceptable risk to the health of people, the system or the environment



Safety Goals



- Safety Goals must be defined for a safety system
 - Example: Automotive Air Bag System:
 - Air bag must deploy in the event of an accident
 - Air bag must not inadvertently deploy when there is not an accident.
- Safety goals are used to help determine the level of functional safety is necessary in a system.
- These safety levels are usually referred to as Safety Integrity Levels or SILs.
- Different safety standards have different metrics and naming classifications for SILs.

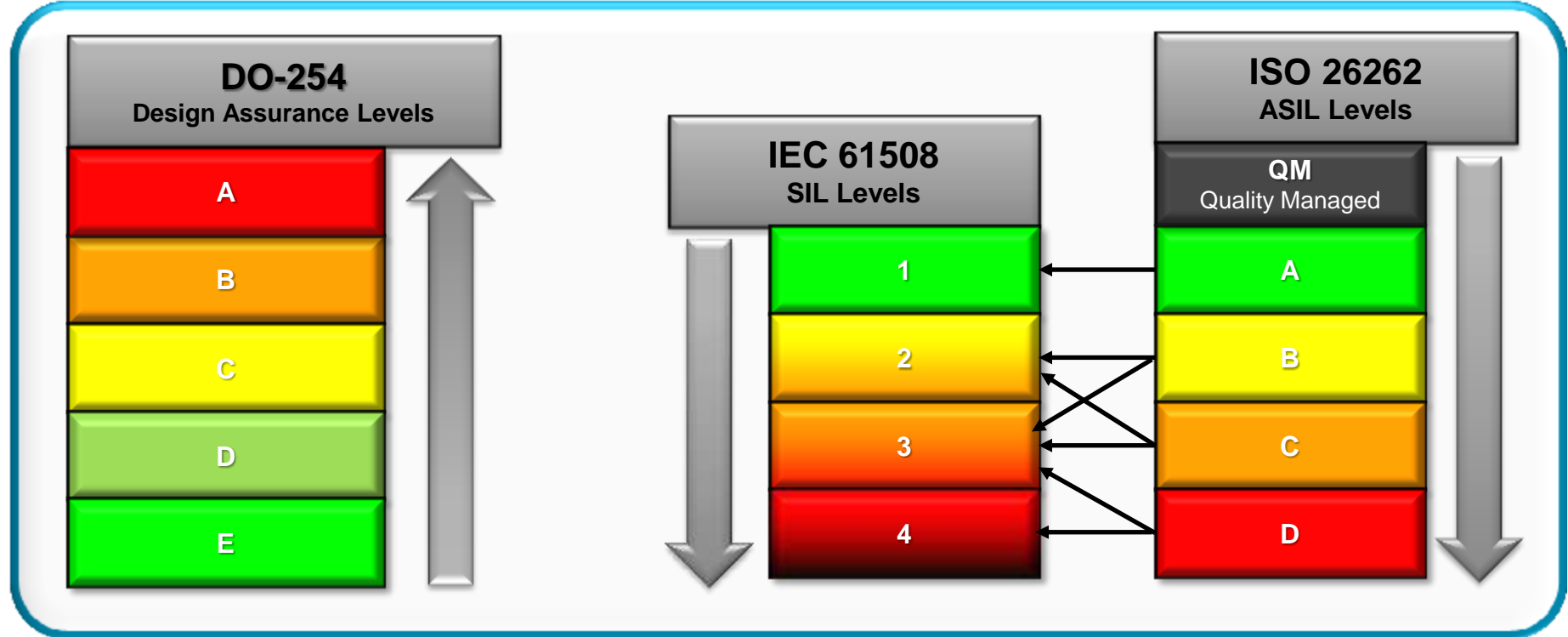
ISO 26262 vs IEC 61508 Safety Integrity Levels

- ISO 26262 was developed to meet automotive industry specific needs as replacement for IEC 61508.
- IEC 61508 defines 4 safety integrity levels (SIL-1,2,3,4)
- ISO26262 defines a Quality Managed level in addition to 4 safety integrity levels (ASIL-A,B,C,D)
- There is no direct correlation between IEC61508 SIL and ISO 26262 ASIL levels



Aerospace DO-254 Design Assurance Levels

- DO-254 Consists of 5 'Design Assurance Levels' (DALs)
 - DAL-A is the most stringent, DAL-E is the least
 - An FAA 'Designated Engineering Representation' (DER) must audit & approve



Determining ISO 26262 ASIL Level

- To determine the ASIL level of a system a Risk Assessment must be performed for all Hazards identified.
- Risk is comprised of three components: **Severity, Exposure & Controllability**

S = Severity

Class	Description
S0	No injuries
S1	Light and moderate injuries
S2	Severe and life-threatening injuries (survival probable)
S3	Life-threatening injuries (survival uncertain), fatal injuries

C = Controllability

Class	Description
C0	Controllable in general
C1	Simply controllable
C2	Normally controllable
C3	Difficult to control or uncontrollable

E = Exposure

Class	Description
E0	Incredible
E1	Very low probability
E2	Low probability
E3	Medium probability
E4	High probability

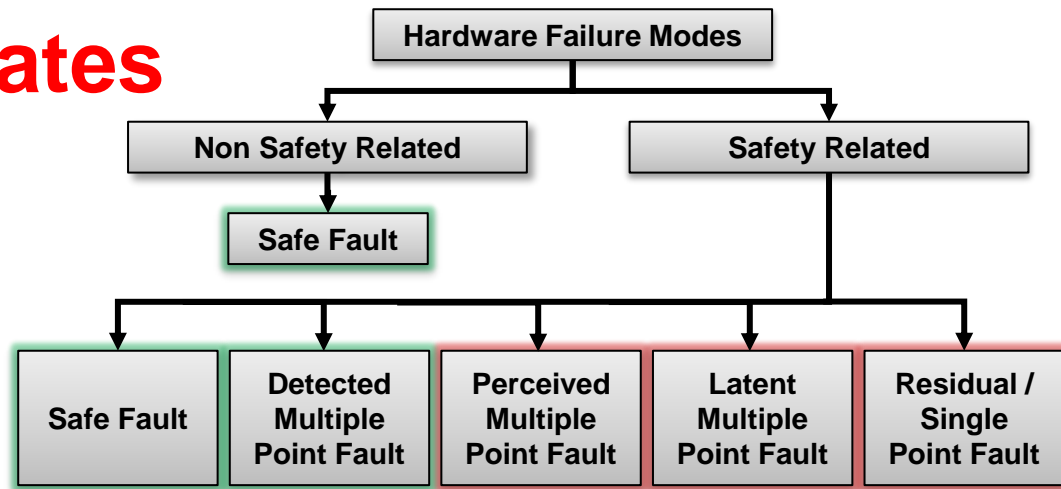


ASIL Determination Table

$$\text{Risk} = \text{Severity} \times (\text{Exposure} * \text{Controllability})$$

		Controllability		
Severity	Exposure	C1 Simply	C2 Normal	C3 Difficult
S1 Light and moderate injuries	E1 Very Low	QM	QM	QM
	E2 Low	QM	QM	QM
	E3 Medium	QM	QM	ASIL A
	E4 High	QM	ASIL A	ASIL B
S2 Severe and life-threatening injuries (survival probable)	E1 Very Low	QM	QM	QM
	E2 Low	QM	QM	ASIL A
	E3 Medium	QM	ASIL A	ASIL B
	E4 High	ASIL A	ASIL B	ASIL C
S3 Life-threatening injuries (survival uncertain), fatal injuries	E1 Very Low	QM	QM	ASIL A
	E2 Low	QM	ASIL A	ASIL B
	E3 Medium	ASIL A	ASIL B	ASIL C
	E4 High	ASIL B	ASIL C	ASIL D

ISO 26262 Failure Rates



Failure Rate λ



- λ_{SPF} - Single Point Faults
- λ_{RF} - Residual Faults
- λ_{MPFDP} - Detected/Perceived Multi Point Faults
- λ_{MPFL} - Latent Multi Point Faults
- λ_{MPF} - $\lambda_{MPFDP} + \lambda_{MPFL}$ = Multi Point Faults
- λ_S - Safe Faults

$$\lambda = \lambda_{SPF} + \lambda_{RF} + \lambda_{MPF} + \lambda_S$$

FIT = Failures In Time = 1 failure in 10^9 device hours

Fault Metrics



- Minimize single point and residual faults.
 - ✓ Detected and handled by system within system safety response time.

$$\text{single point fault metric} = 1 - \frac{\Sigma(\lambda_{\text{SPF}} + \lambda_{\text{RF}})}{\Sigma \lambda} = \frac{\Sigma(\lambda_{\text{MPF}} + \lambda_{\text{S}})}{\Sigma \lambda}$$

Metric	ASIL B	ASIL C	ASIL D
Single point fault metric	≥ 90%	≥ 97%	≥ 99%

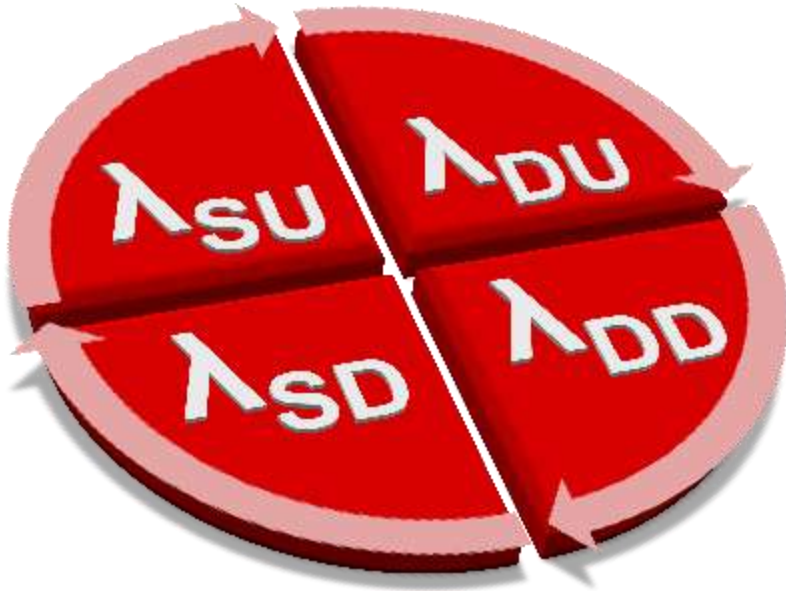
- Minimize latent multi point faults.
 - ✓ Detected and handled within hours through test algorithms.

$$\text{latent fault metric} = 1 - \frac{\Sigma(\lambda_{\text{MPFL}})}{\Sigma(\lambda - \lambda_{\text{SPF}} - \lambda_{\text{RF}})} = \frac{\Sigma(\lambda_{\text{MPFDP}} + \lambda_{\text{S}})}{\Sigma(\lambda - \lambda_{\text{SPF}} - \lambda_{\text{RF}})}$$

Metric	ASIL B	ASIL C	ASIL D
Latent fault metric	≥ 60%	≥ 80%	≥ 90%

IEC 61508 Failure Rates

Failure Rate λ



- λ_S – Safe failure rate
 - **No impact** on safety function
 - λ_{SD} – Safe detected failure rate
 - λ_{SU} – Safe undetected failure rate
- λ_D – Dangerous failure rate
 - **Impact** on safety function
 - λ_{DD} – Dangerous detected failure rate
 - λ_{DU} – Dangerous undetected failure rate

$$\lambda = \lambda_S + \lambda_D = (\lambda_{SD} + \lambda_{SU}) + (\lambda_{DD} + \lambda_{DU})$$

FIT = Failures In Time = 1 failure in 10^9 device hours

IEC 61508 Safe Failure Fraction & SIL Determination



$$\text{Safe Failure Fraction (SFF)} = 1 - \frac{\lambda_{DU}}{\lambda}$$

High Demand System

Hardware Fault Tolerance = 0 (single channel)

- 1 Fault may lead to loss of safety function.

Hardware Fault Tolerance = 1 (redundant)

- 2 or more faults needed to loss of safety function.

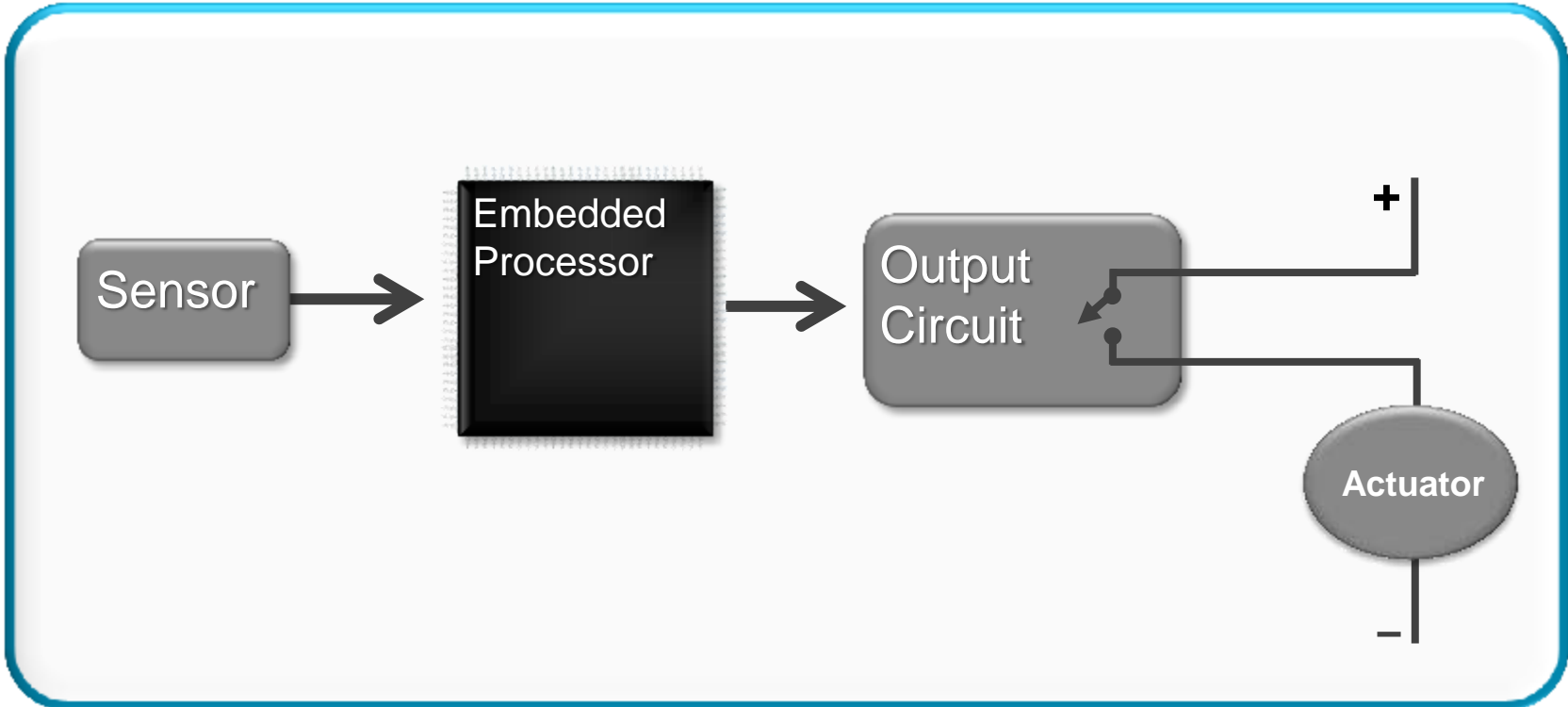
Safe Failure Fraction (High Demand System)	Hardware Fault Tolerance	
	HFT = 0	HFT = 1
0 ... < 60	-	SIL1
60 ... < 90	SIL1	SIL2
90 ... < 99	SIL2	SIL3
≥ 99	SIL3	SIL4

Safety System Architectures

Safety System Architectures

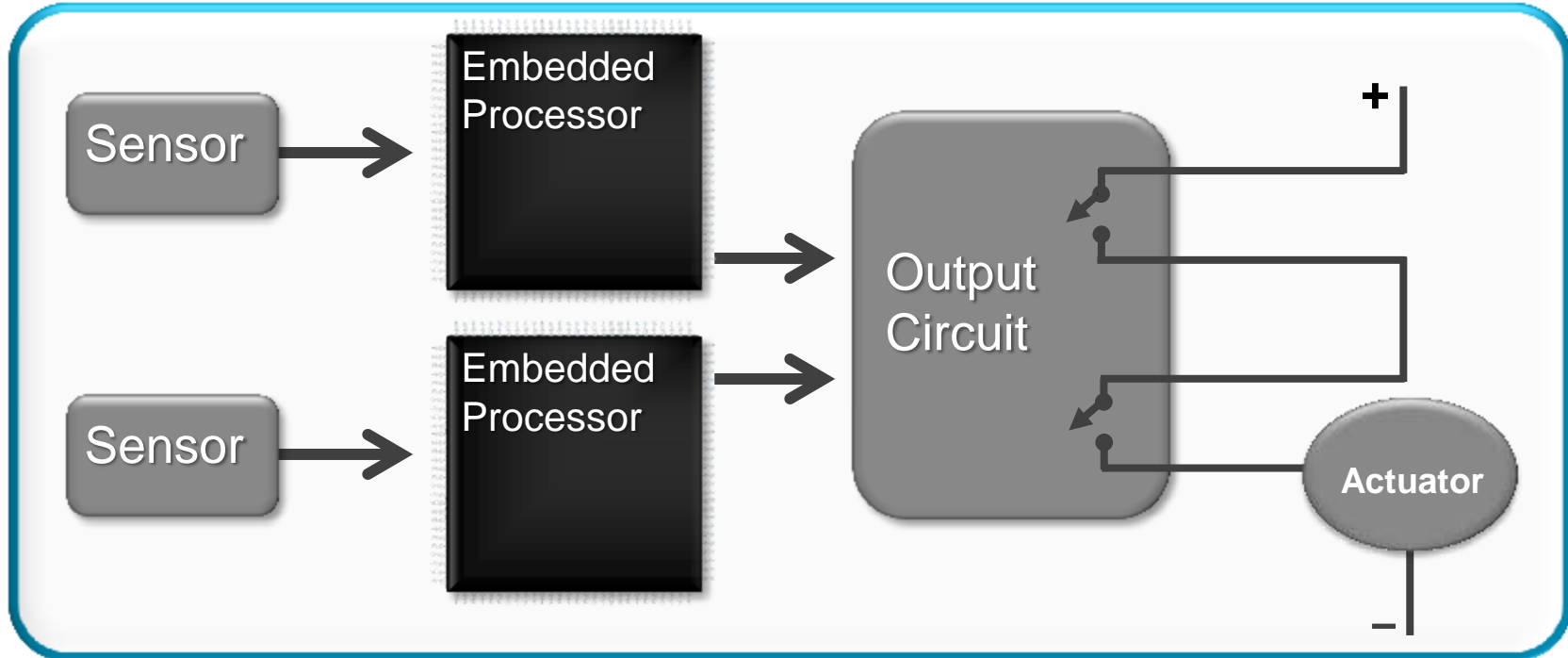
- There are numerous system architectures designed for functional safety. This section describes a few of the most common, but is not exhaustive.
- General terminology is “XooY”
 - X out of Y
 - X out of Y total units must fail for total system to fail

1001 Processing Architecture



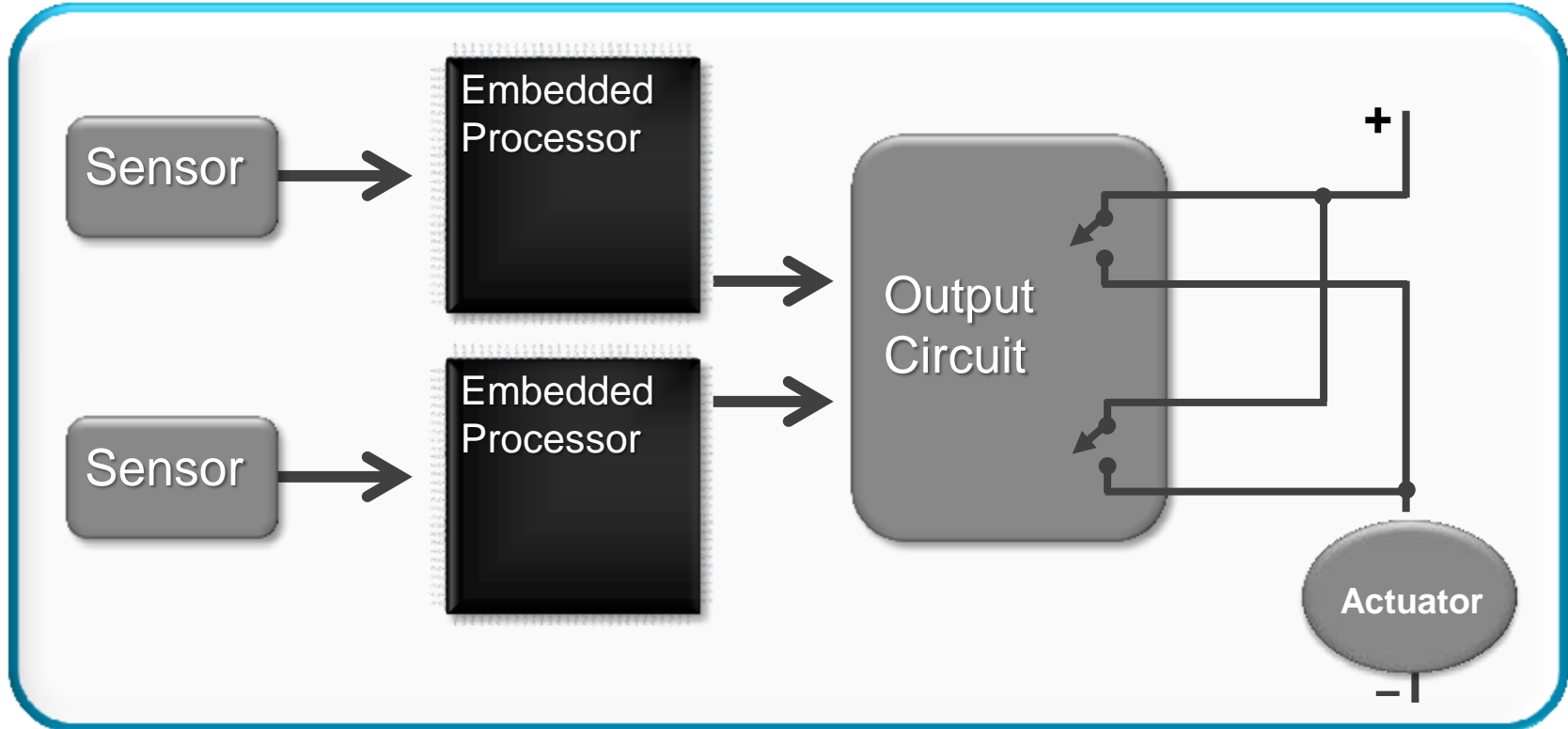
- “One out of One” subsystem failure creates system failure
- Most minimal system configuration possible
- No internal diagnostics
- No fault redundancy

1002 Processing Architecture



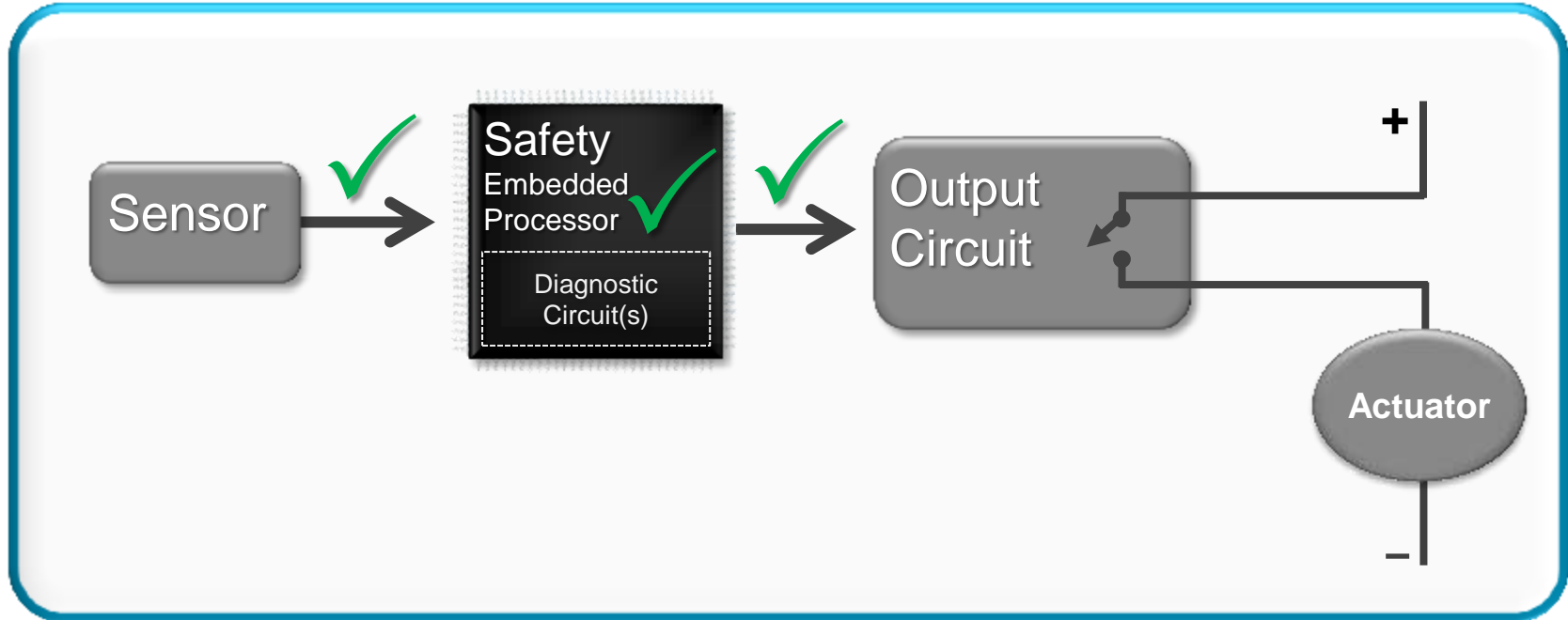
- “One out of Two” subsystems must fail for system to fail
- Two controllers with independent I/O
 - Both controllers must command an output for output to occur
 - Failure in both systems required for inadvertent activation
- Often implemented in airbag systems with a 32b main MCU and an 8b secondary MCU used to energize squib charges

2oo2 Processing Architecture



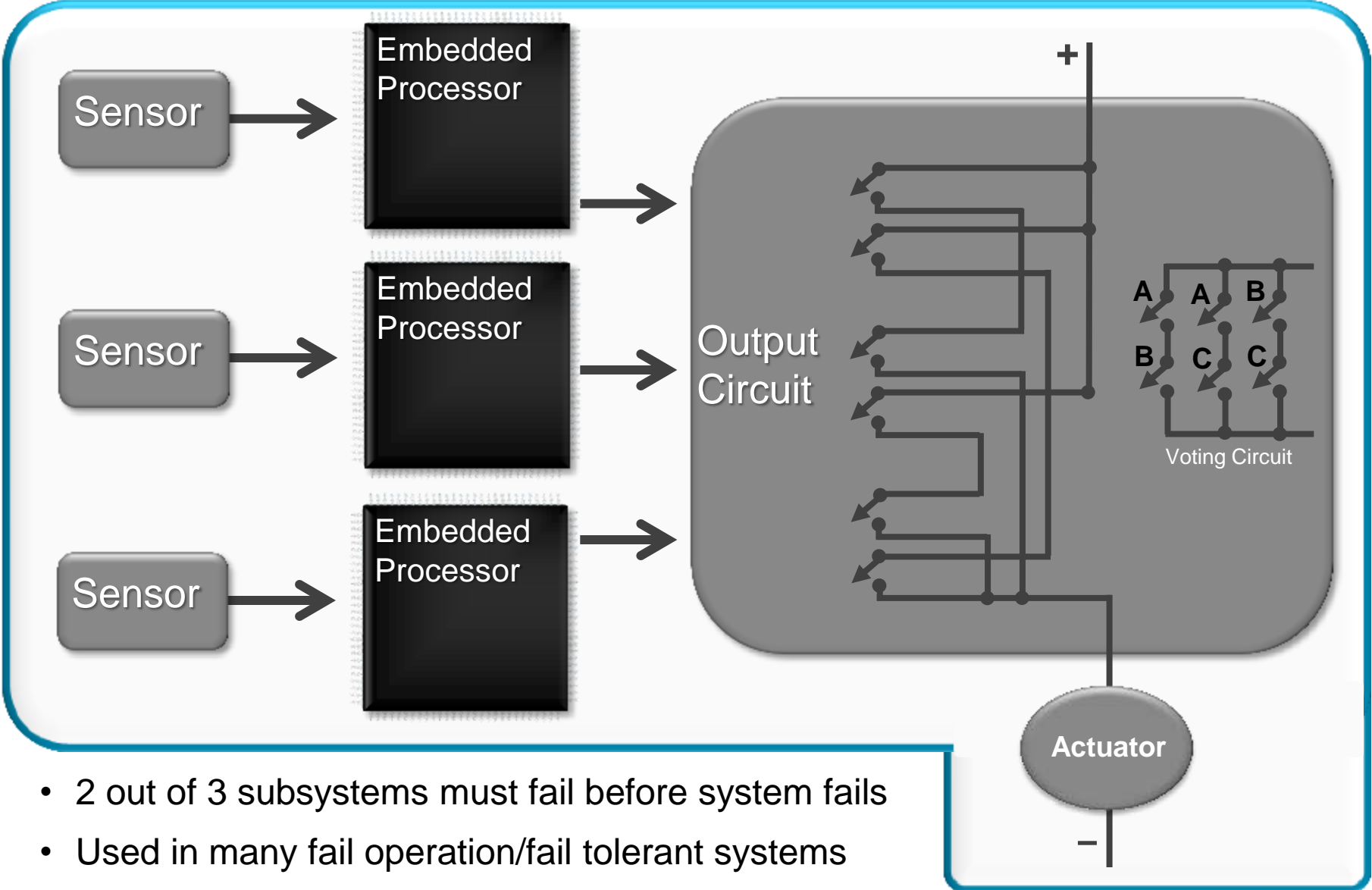
- “Two out of Two” subsystems must fail for system to fail
- Each system can energize the output
 - Used when it is undesirable for output to be de-energized
 - Fault tolerant of open circuit faults in either subsystem

1oo1D Processing Architecture



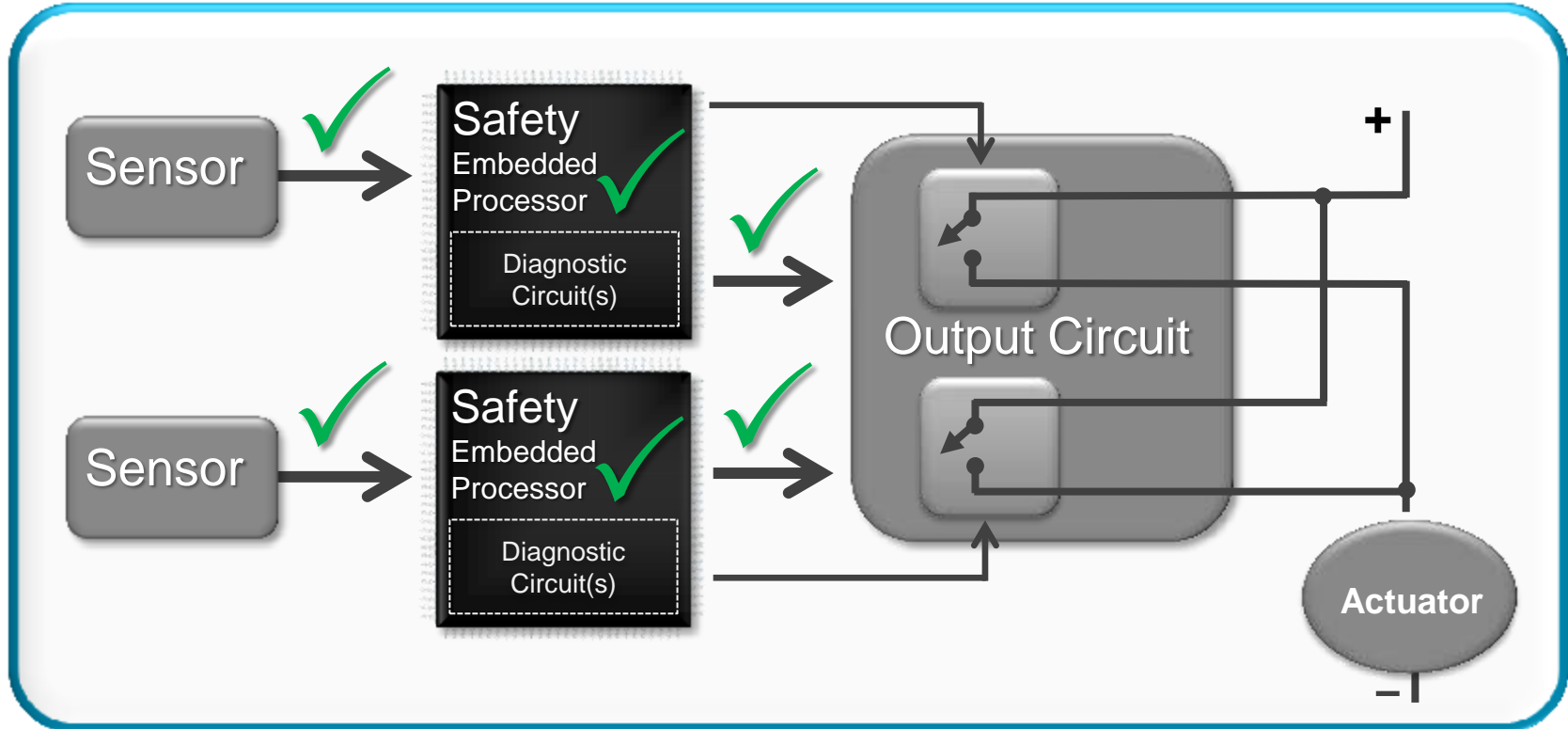
- Expansion of 1oo1 system to include a diagnostic channel
- Diagnostic channel can inhibit system output if a failure is detected in the functional system
- Additional failure rate potential due to failure in the diagnostic circuits (annunciation failure)
- TI Hercules “lockstep” processor implementations are a 1oo1D system

2003 Processing Architecture



- 2 out of 3 subsystems must fail before system fails
- Used in many fail operation/fail tolerant systems

2oo2D Processing Architecture



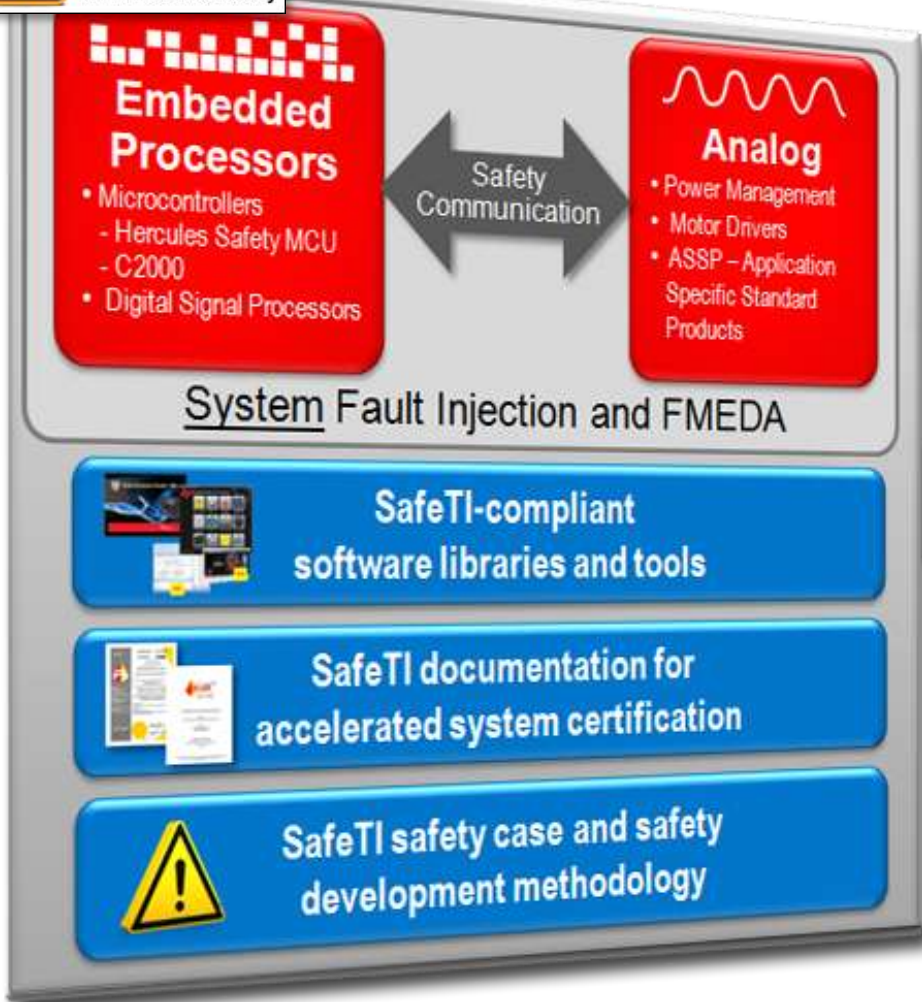
- Effectively a 2oo2 system where each channel is a 1oo1D system
- Provides a single level of fault tolerance
 - Upon single channel failure the system reverts to a 1oo1D system
 - Provides high fault detection rates

SafeTI™



SafeTI™
Design Packages
for Functional Safety

TI's SafeTI™ design packages for functional safety are robust and help speed certification



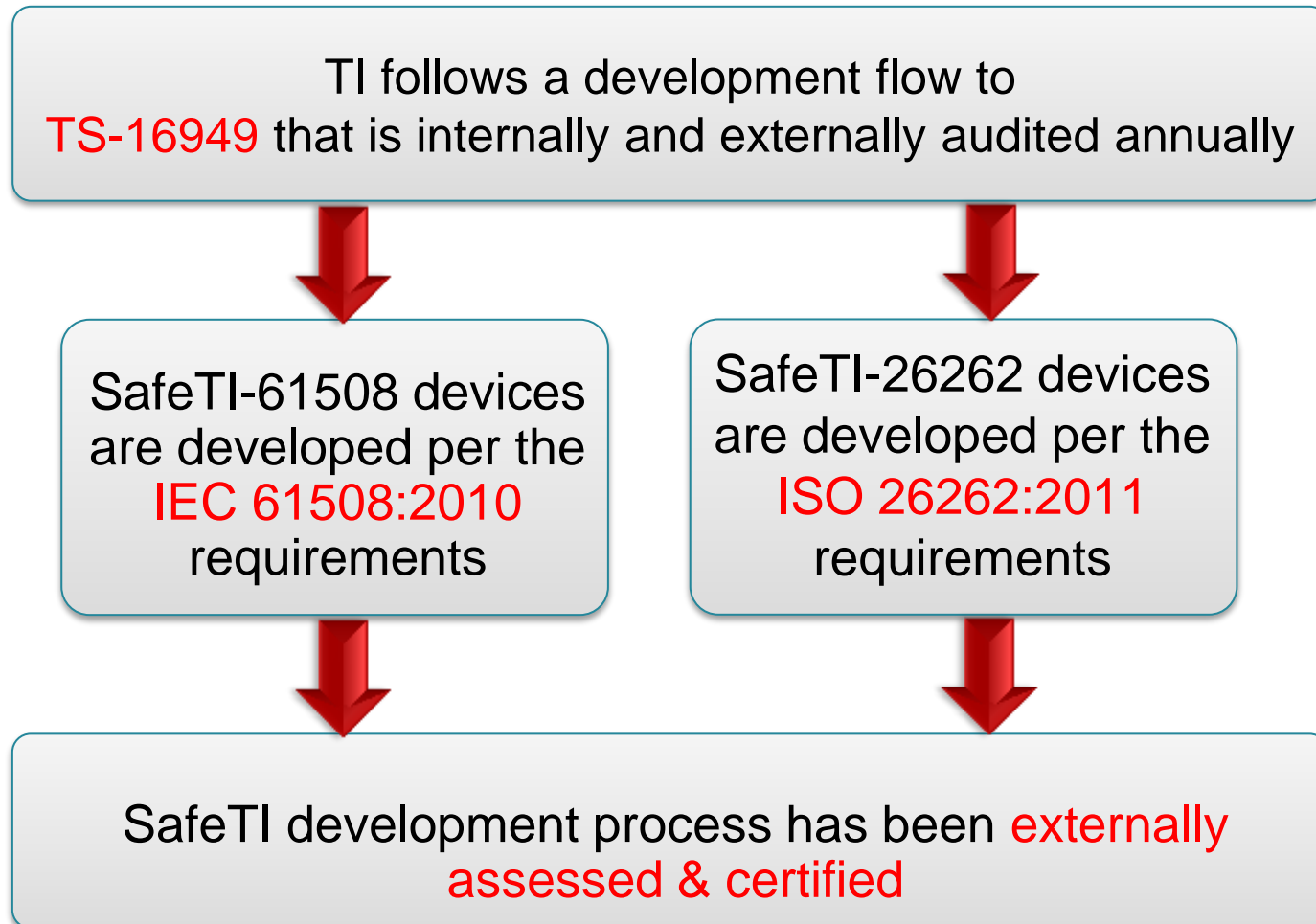
Standards-specific SafeTI solution bundles



- SafeTI - 61508 Industrial, Medical, Rail, Other
- SafeTI - 26262 Automotive
- SafeTI - 60730 Household appliance
- SafeTI - QM Quality Managed

www.ti.com/safeti

SafeTI™ - Safety Development Process



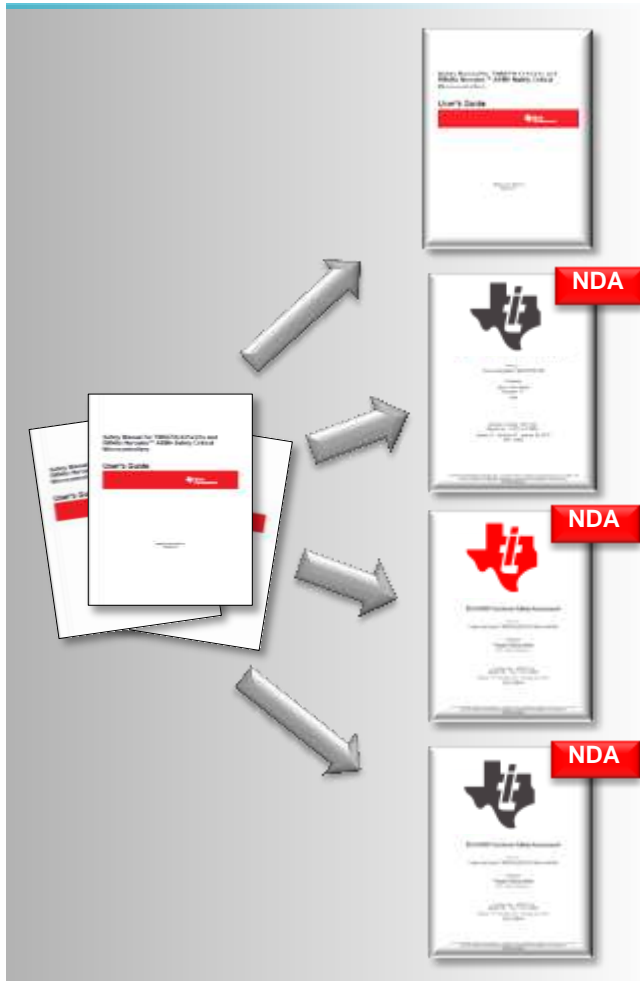
SafeTI™ Development Process Certification by TÜV SÜD



- TÜV SÜD is an internationally recognized and accredited independent assessor of compliance to quality, safety, and security standards.
- TÜV-SÜD has certified the functional safety development process for
 - SafeTI-61508
 - SafeTI-26262
- The certification demonstrates TI's commitment to have a process suitable for developing hardware components that are compliant to ISO 26262 and IEC 61508

Hercules Safety Documents

- Documents provided by TI *some under NDA* to assist in the safety certification process:



- **Hercules TMS570/RM Device Safety Manual (SM)**

- Details product safety architecture and recommended usage

- **Safety Analysis Report Summary (SAR1)**

- Summary of FIT rate and FMEDA at DEVICE level for IEC 61508 and ISO 26262

- **Detailed Safety Analysis Report (SAR2)**

- Full details of all safety analysis executed down to MODULE level for IEC 61508 and ISO 26262

- **Safety Case Report (SCR)**

- Summary of compliance to IEC 61508 and/or ISO 26262

SafeTI™ Compiler Qualification Kit



- Assists in qualifying the TI ARM C/C++ Compiler to functional safety standards
- Model-based tool qualification methodology developed by Validas
- Assessed by TÜV Nord to comply with both IEC 61508 and ISO 26262

Approved by

TÜV NORD

- Includes:
 - Qualification Support Tool
 - Documentation:
 - Tool Classification Report
 - Tool Qualification Plan
 - Tool Qualification Report
 - Tool Safety Manual
 - ACE SuperTest qualification suite
 - TI compiler validation test cases
 - Test Automation Unit (TAU)
 - 24hrs of Validas consulting services



IEC 61508



ISO 26262

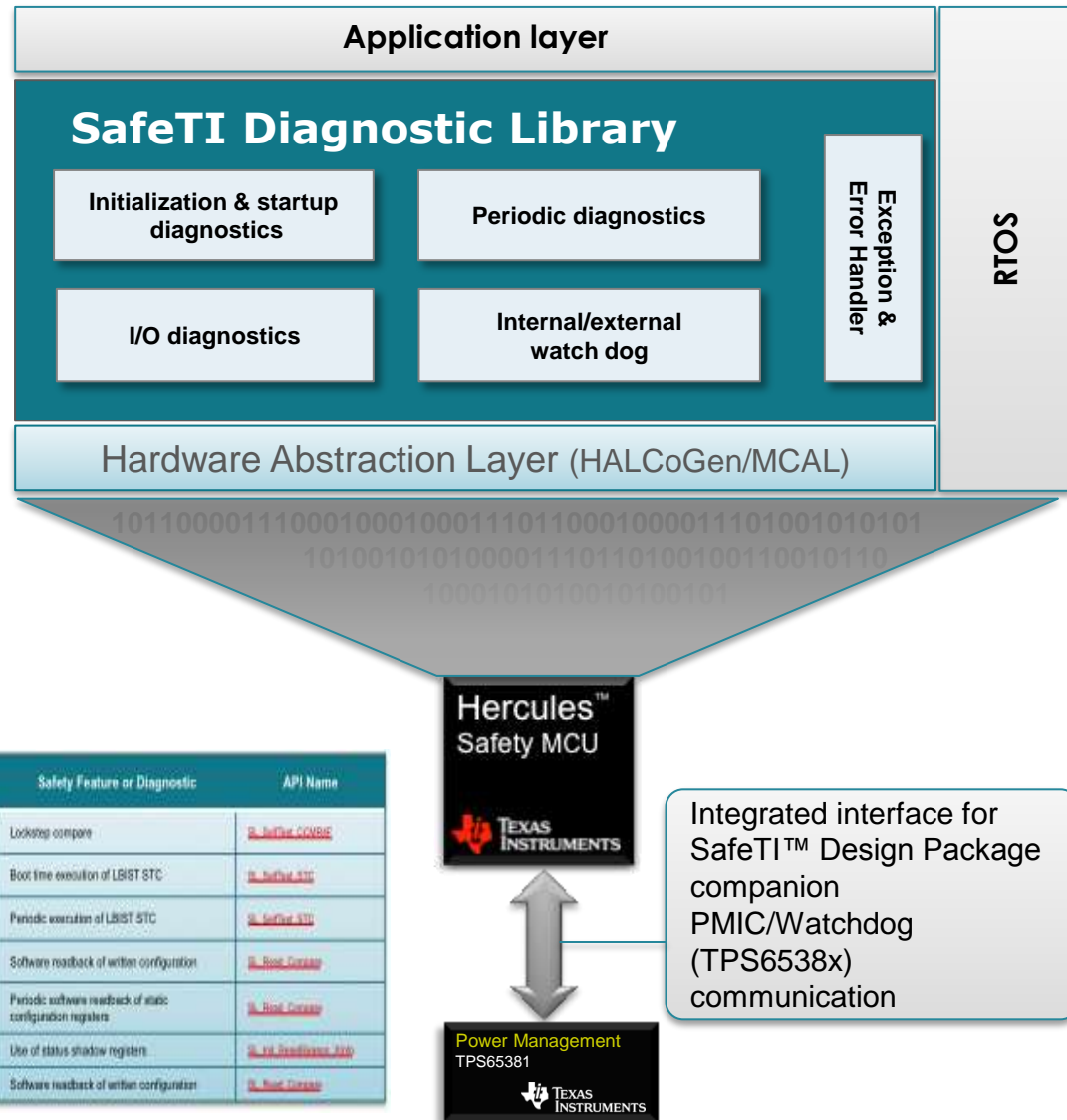
NOTE: There is a fee for this kit. See the web site listed below for more details.



The SafeTI™ Diagnostic Library: Hercules Safety MCUs

Provides simple interfaces and a framework for

- Initializing and Enabling Safety diagnostics/Features prescribed by the Hercules Safety Manual.
- Fault injection to allow testing of application fault handling
- Error Signaling Module (ESM) handler callback routine.
- Profiling for measuring time spent in diagnostic test/fault handling



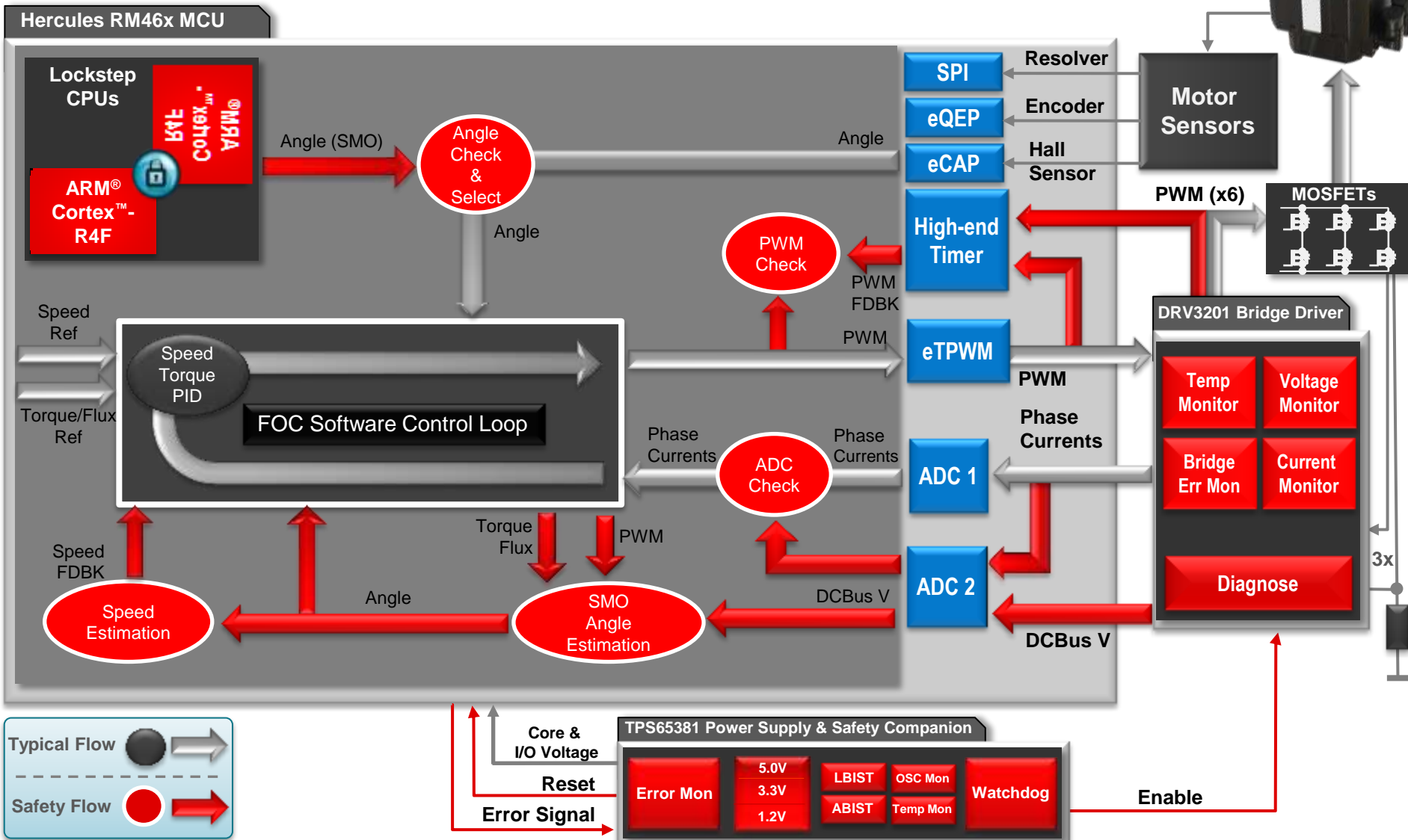
Functions map directly to the Hercules Safety Manual

Device Partition	Unique Identifier	Safety Feature or Diagnostic	API Name
Cortex-R4F CPU	CPU1	Locking compare	IS_InitLib_CNVBE
	CPU2A	Boot time evolution of LBIST STC	IS_InitLib_STC
	CPU2B	Periodic execution of LBIST STC	IS_InitLib_STC
	CPU7	Software readback of written configuration	IS_Read_Config
Error Signaling	ESM1	Periodic software readback of static configuration registers	IS_Read_Config
	ESM3	Use of status shadow registers	IS_Read_Status_Std
	ESM4	Software readback of written configuration	IS_Read_Config



Integrated interface for SafeTI™ Design Package companion PMIC/Watchdog (TPS6538x) communication

SafeTI™ Motor Control Example



[Hercules RM46x MCU](#)
[TPS65381 Power Supply & Safety Companion](#)
[DRV3201 Bridge Driver](#)



Links to Product Web Pages:



TI's 20+ years of Designing Solutions for Safety Markets help provide:

- Embedded Processing + Analog solutions that work together
- Functional Safety integrated in device hardware
- Tools and Software prepared for safety
- Comprehensive Safety Documentation

→ Enable faster and easier customer Safety Certification

Hercules™ Cortex™-R4F Safety Concept

Rationale of the Hercules™ Safety Concept

- “Safe Island” approach
- Region of device common to all safety functions is heavily protected by hardware diagnostic measures
 - CPU
 - CPU Interrupts
 - System control of power, reset, clock
 - OS critical IP: DMA, OS timer
- Once a known safe region can be guaranteed, logic in this region can be used to provide diagnostic coverage on other regions
- This partition has shown to give strong safety metrics while minimizing impact of safety on system BOM cost

Hercules Cortex-R4F MCU safety features

CPU Self Test Controller requires little S/W overhead

ECC for flash / RAM / interconnect evaluated inside the Cortex R4F

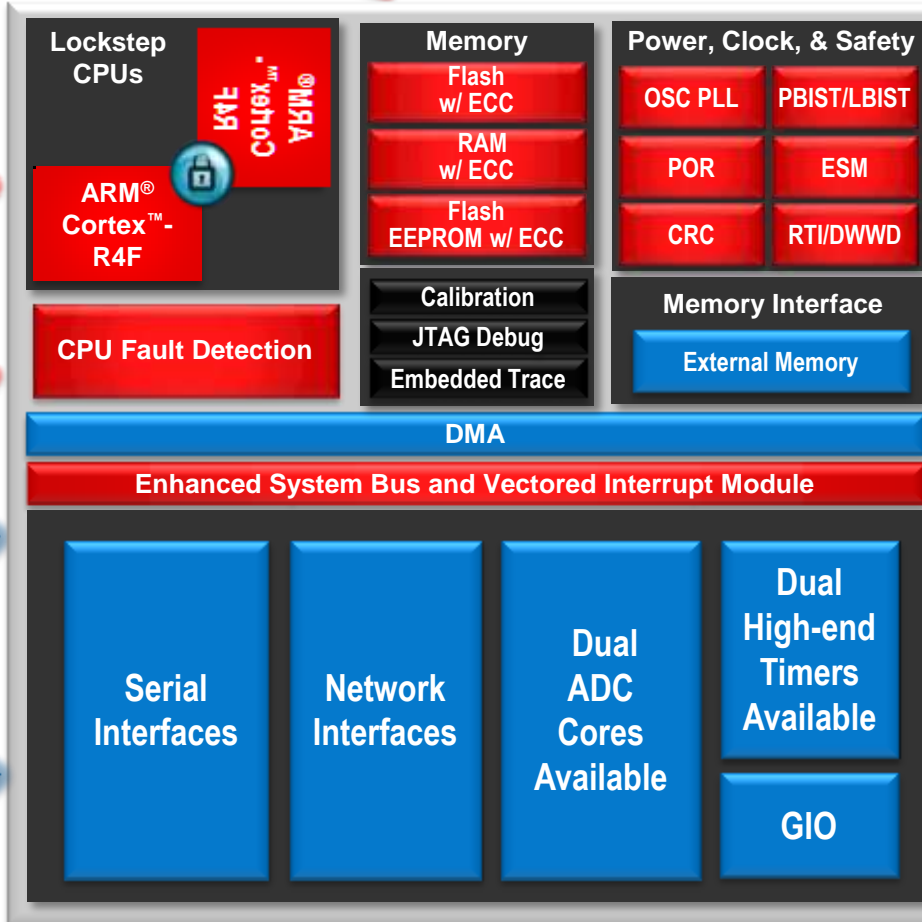
- Safe Island Hardware diagnostics (RED)
- Blended HW diagnostics (BLUE)
- Non Safety Critical Functions (BLACK)

Logical / physical design optimized to reduce probability of common cause failure

Dual Core Lockstep - Cycle by Cycle CPU Fault Detection

Parity on all Peripheral, DMA and Interrupt controller RAMS

Parity or CRC in Serial and Network Communication Peripherals



Memory BIST on all RAMS allows fast memory test at startup

On-Chip Clock and Voltage Monitoring

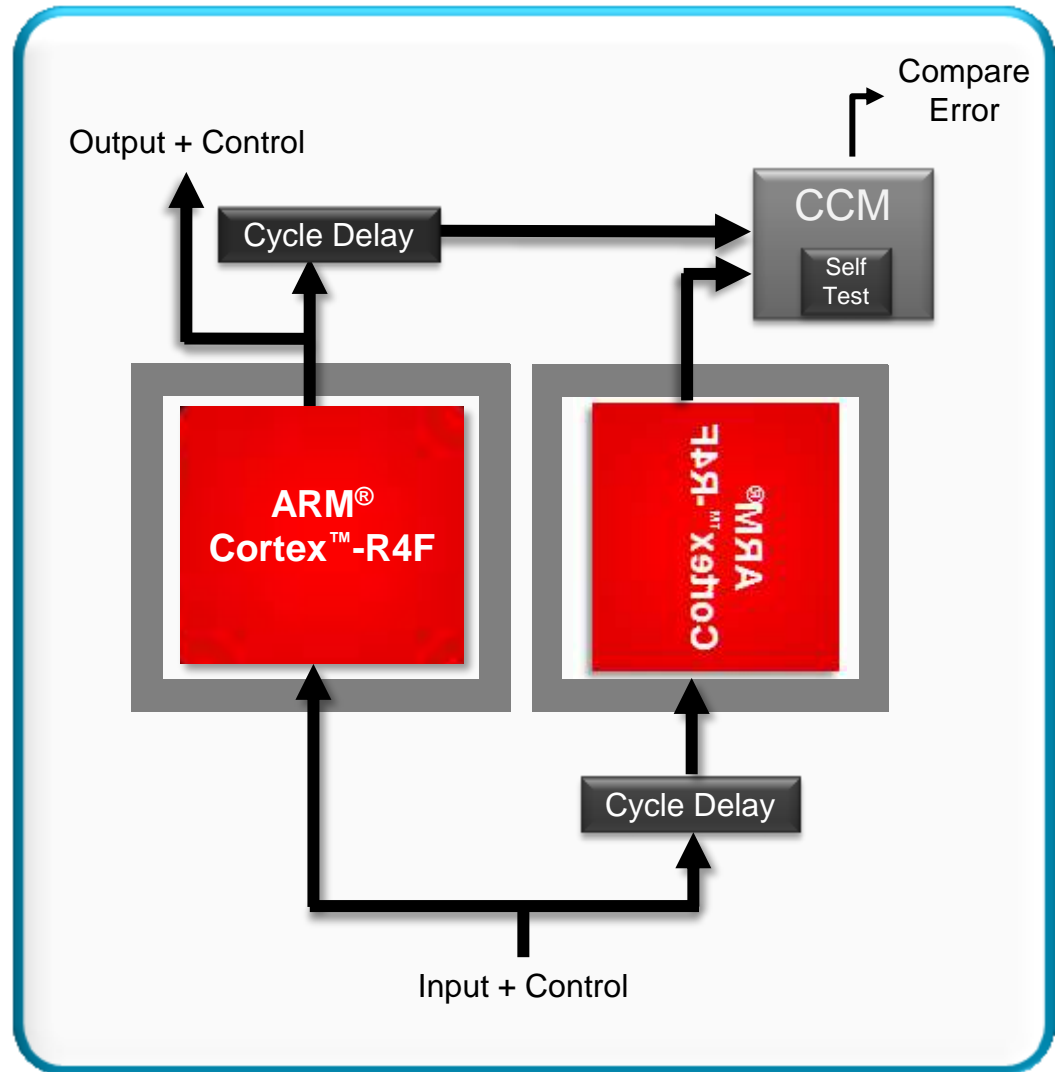
Error Signaling Module w/ External Error Pin

IO Loop Back, ADC Self Test, ...

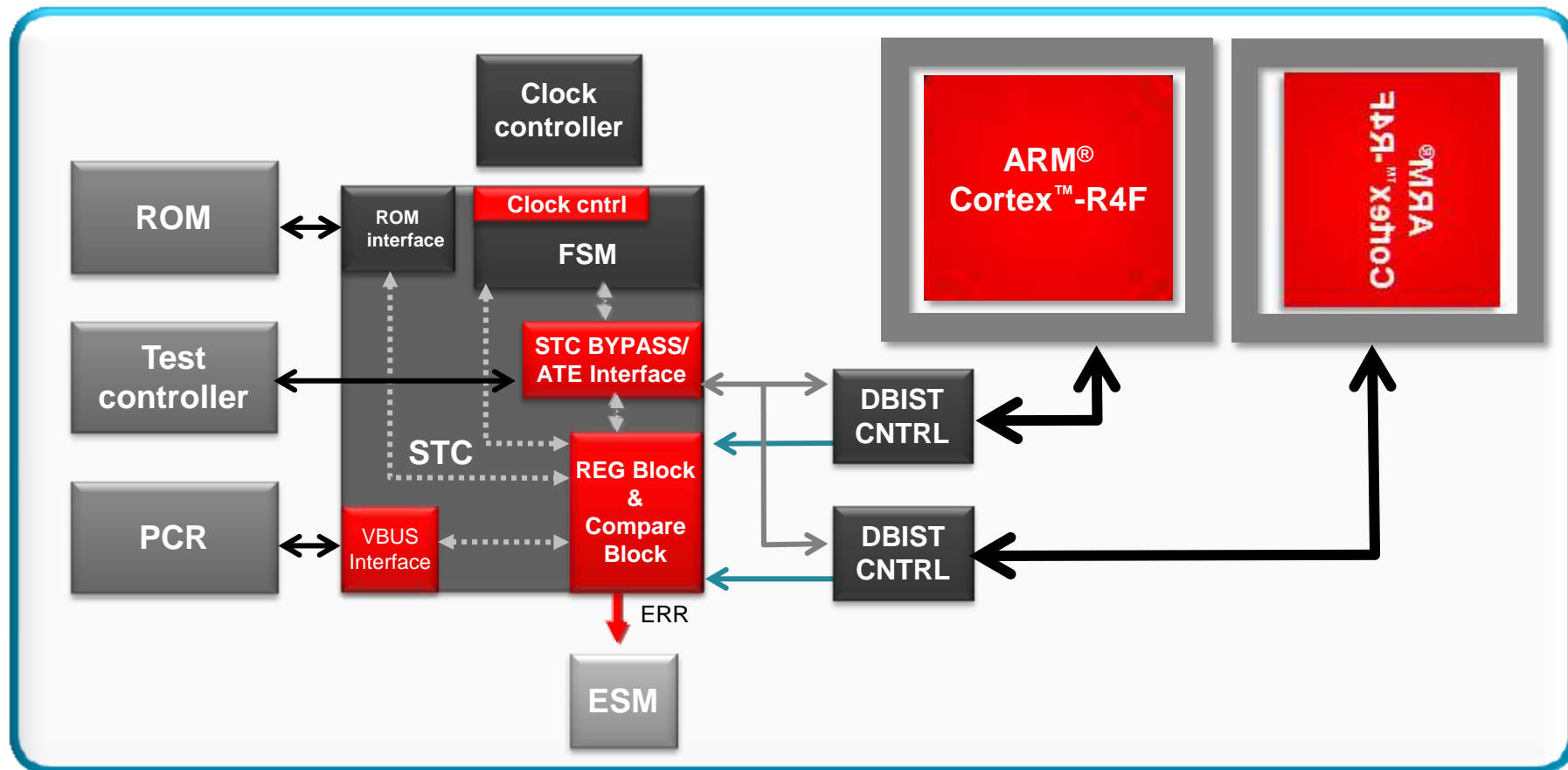
Dual ADC Cores with shared channels

1001D Dual Core Safety Concept

- Unique design to reduce common cause failures
 - Second CPU mirrored and rotated
 - Cycle delayed lockstep
 - Guard ring per CPU
 - Duplicated clock tree per CPU
- CPU Compare Module (CCM)
 - Self-test capability
 - Self-test error injection/error forcing
 - Output error injection



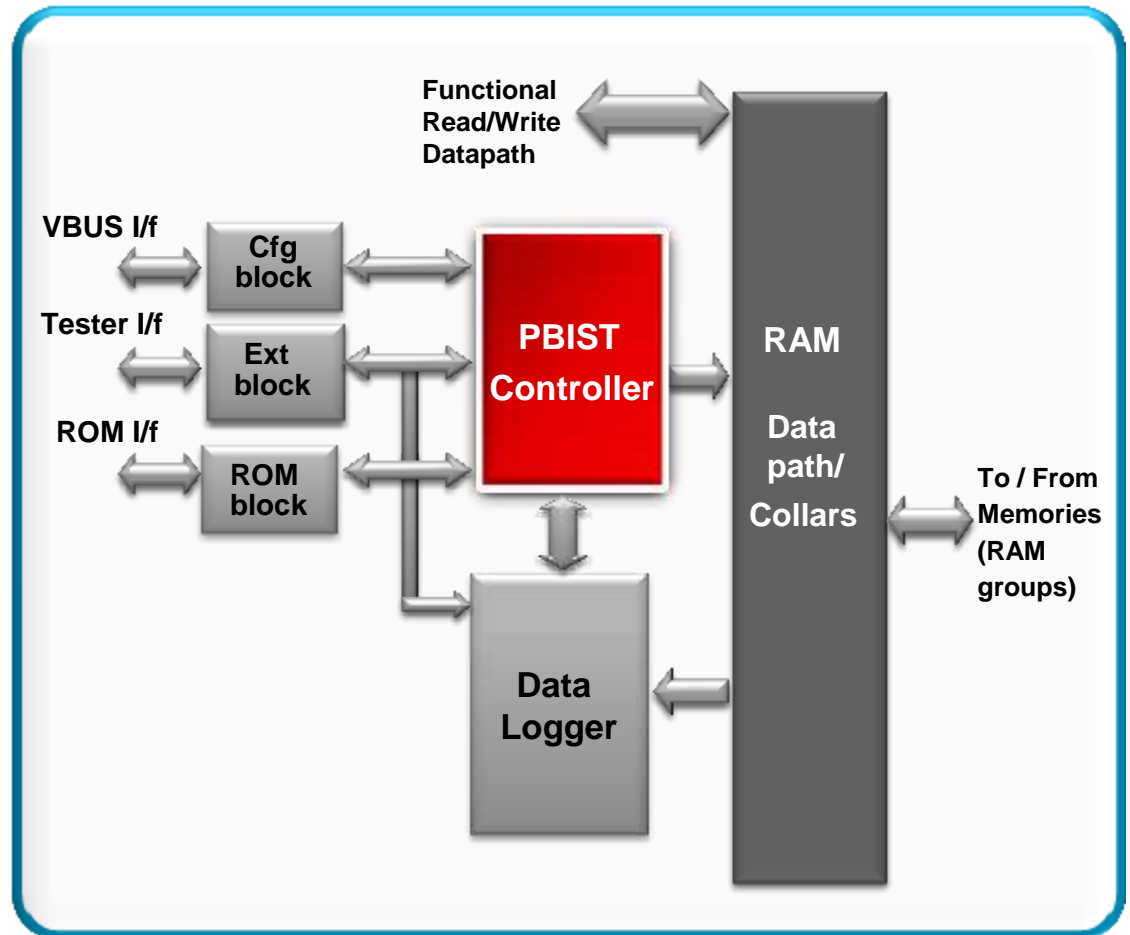
CPU Self Test Controller (STC/LBIST)



- Provides High Diagnostic Coverage
- Significantly Lowers S/W and Runtime Overhead
- No SW BIST (Built In Self Test) Code overhead in Flash
- Simple to configure and start BIST via register

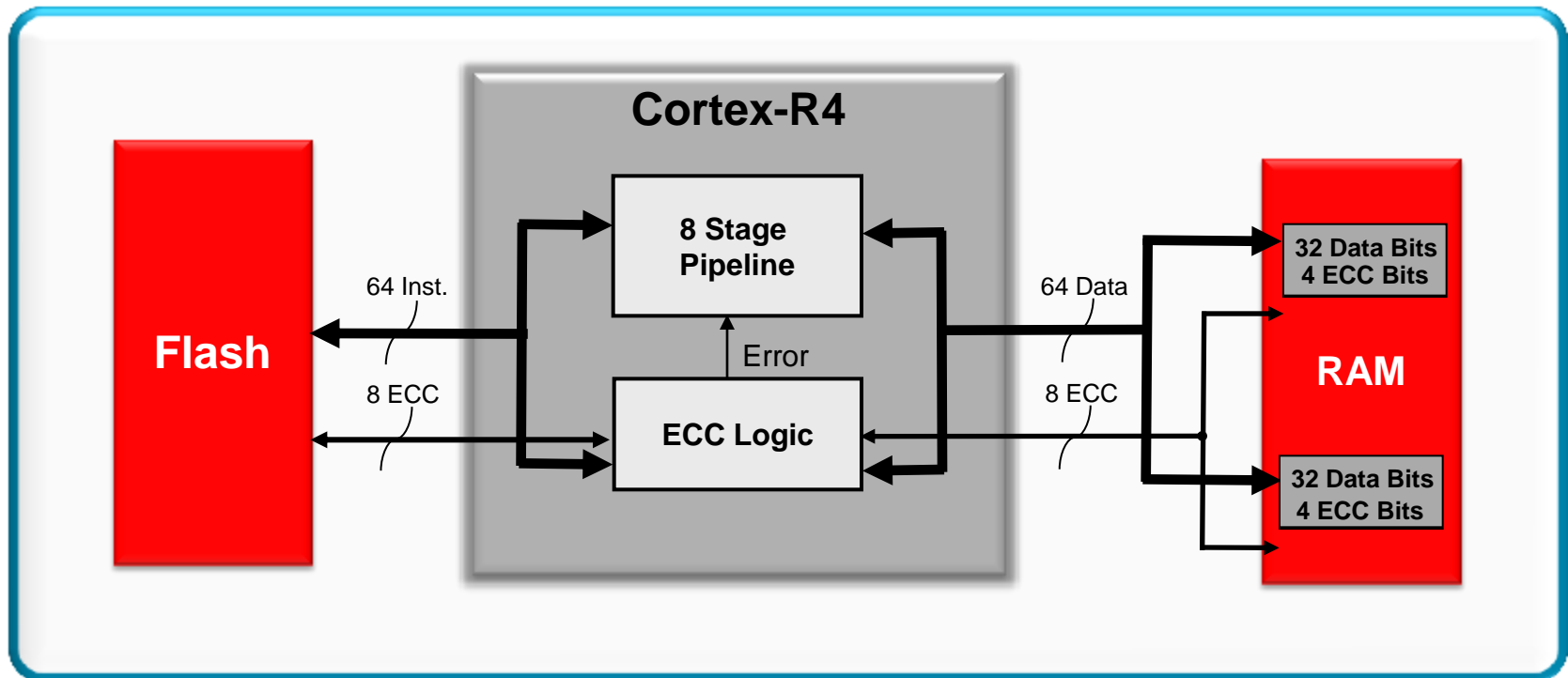
Programmable Memory BIST (PBIST)

- All on-chip RAMS can be tested
- Simple register setup and configuration
- Typically run at startup, but can be executed during the application
- Multiple Memory Test Algorithms
- Detects multiple failure modes



- Provides a mechanism to determine if runtime faults were caused by hard or soft error. This capability can be used to improve availability through inline recovery from soft error.

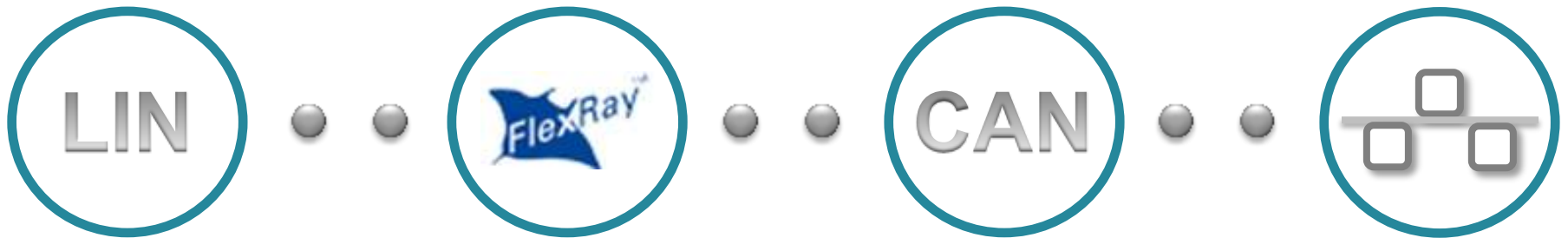
Flash / RAM ECC Protection



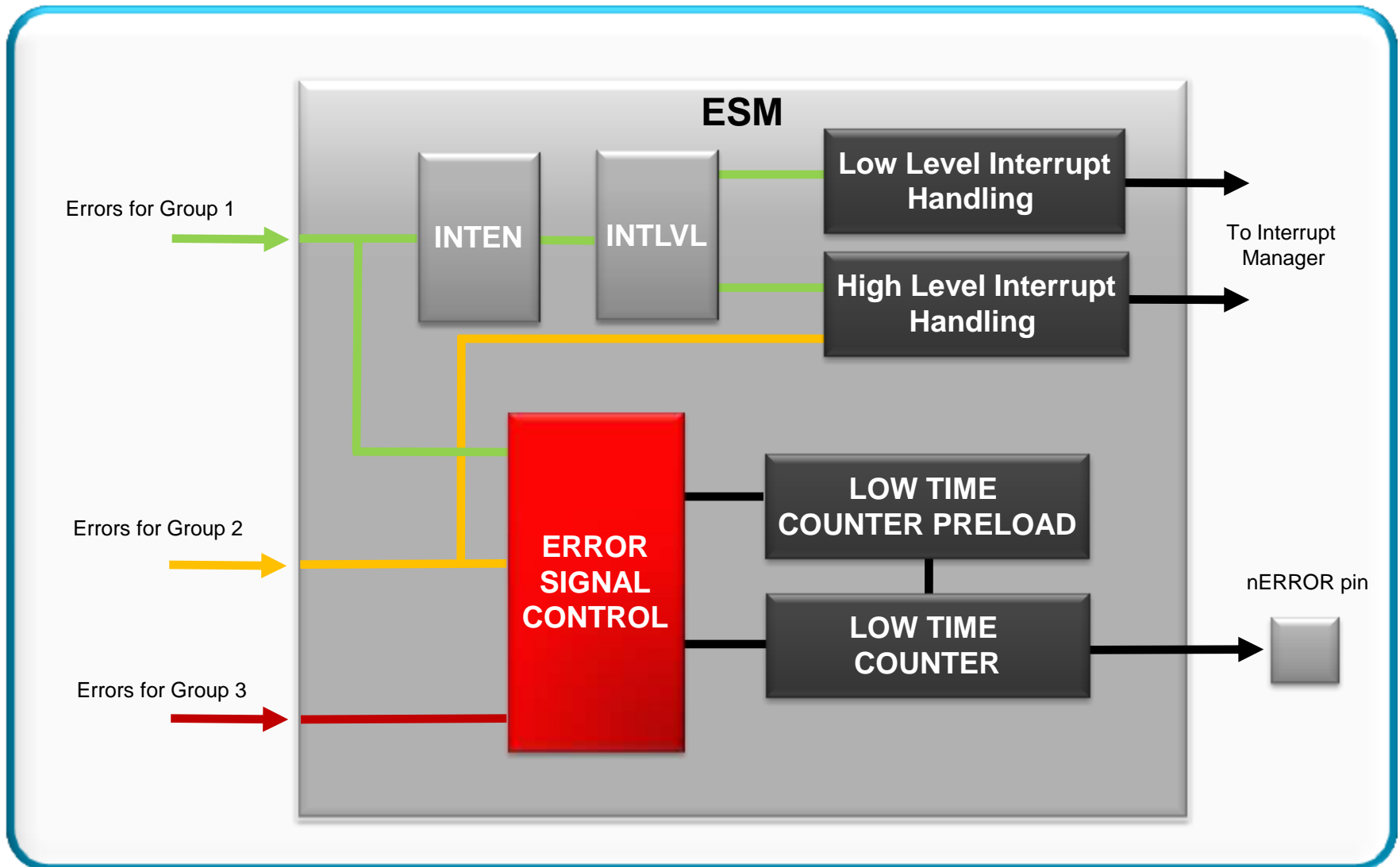
- ECC evaluated in the Cortex R4 CPU
 - Single Bit Error Correction and Double Bit Error Detection (SECEDED)
 - ECC evaluated in parallel to processing data/instructions
 - No latency or performance impact
 - Protects Busses from CPU to Flash and RAM

Safety Aspects of Network Interfaces

- Networked peripherals (Ethernet, FlexRay, DCAN, and SCI/LIN) are considered grey-channel / black-channel communications
- In such communications application level protocols (time redundancy, CRC in data packet, etc.) are necessary

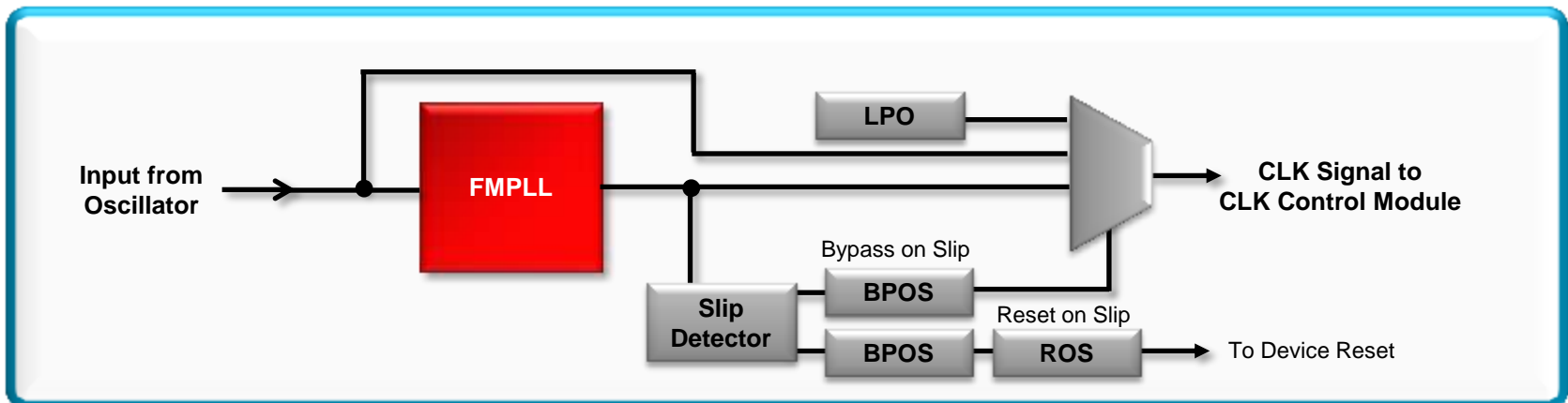


Error Signaling Module (ESM)



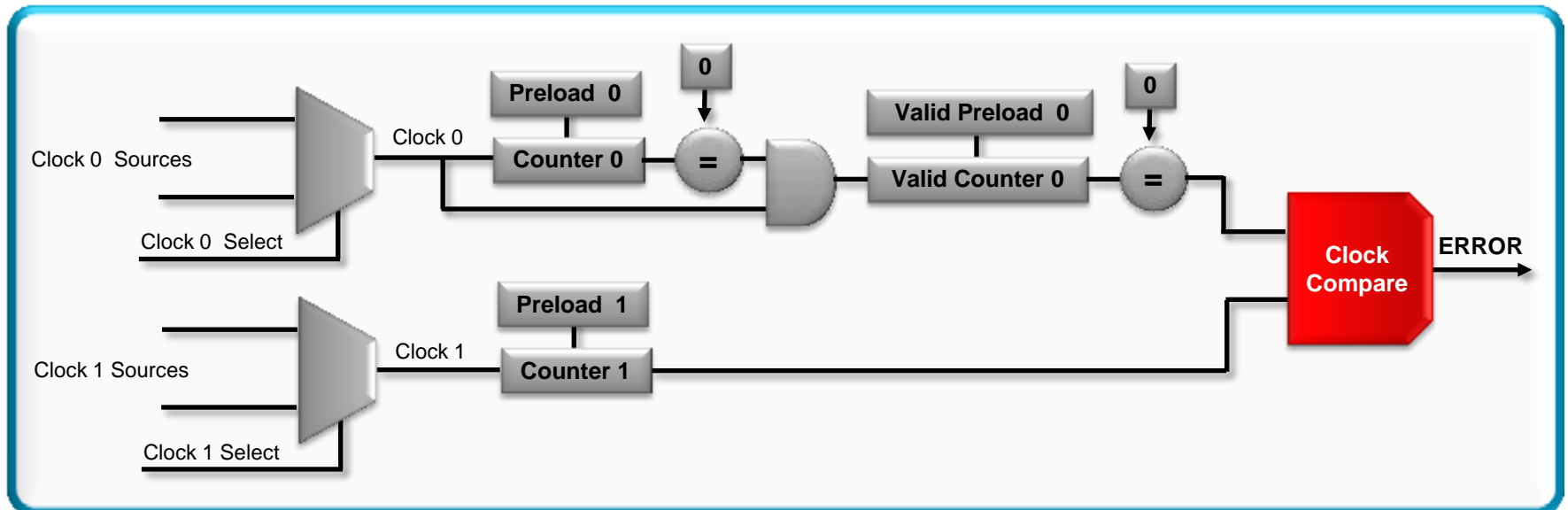
Clock Monitoring

- External clock prescaler (ECLK)
 - Allows external monitoring of CPU clock frequency
 - Configurable pin (GIO or ECLK)
- Oscillator monitor
 - Detects failure if oscillator frequency exceeds defined min/max thresholds
 - Selectable hardware response on oscillator fail
 - Reset device
 - Switch to internal 'low power oscillator' (LPO) clock source
- FMPLL slip detector
 - Indicates PLL slip if phase lock is lost
 - Selectable hardware response on PLL slip
 - Reset device
 - Switch to internal 'low power oscillator' (LPO) clock source
 - Switch to external oscillator clock source



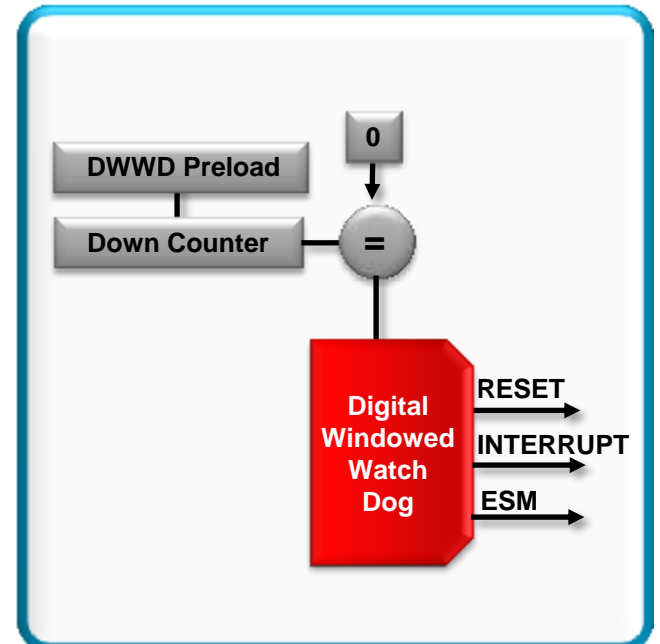
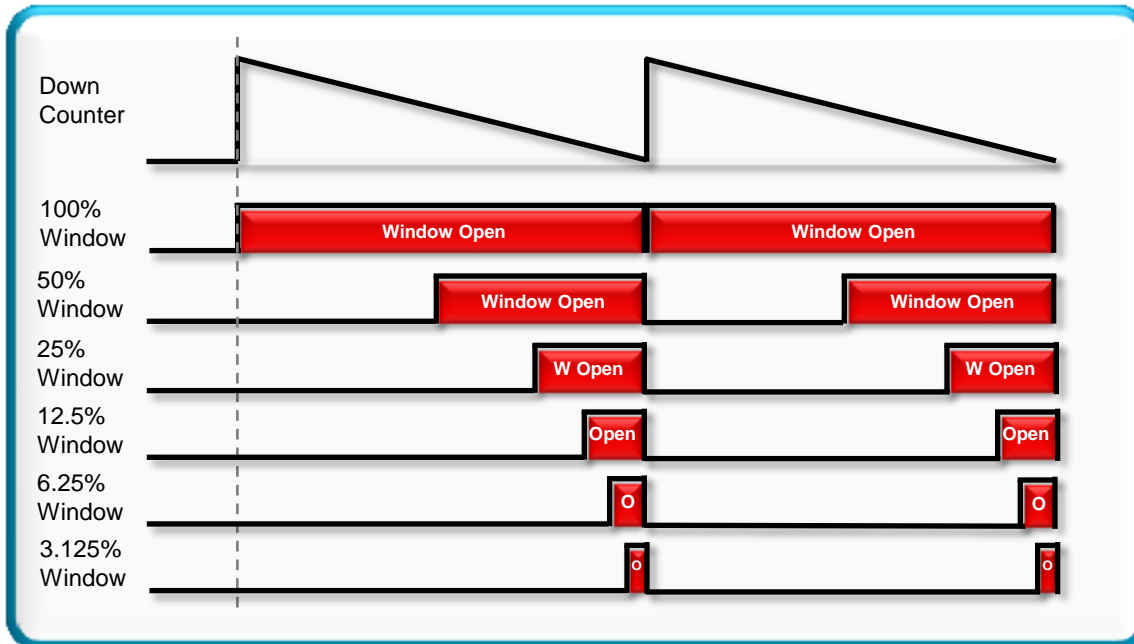
Dual Clock Comparator (DCC)

- The DCC module is used to measure the frequency of a clock signal using a second clock signal as a reference.
 - Allows application to ensure that a fixed frequency ratio is maintained between two clock signals
 - Supports the definition of a programmable tolerance window in terms of number of reference clock cycles
 - Supports continuous monitoring without requiring application intervention
 - Alternatively can be used in a single-sequence mode for spot measurements
 - Flexible clock source selection for Counter 0 and Counter 1 resulting in several specific use cases



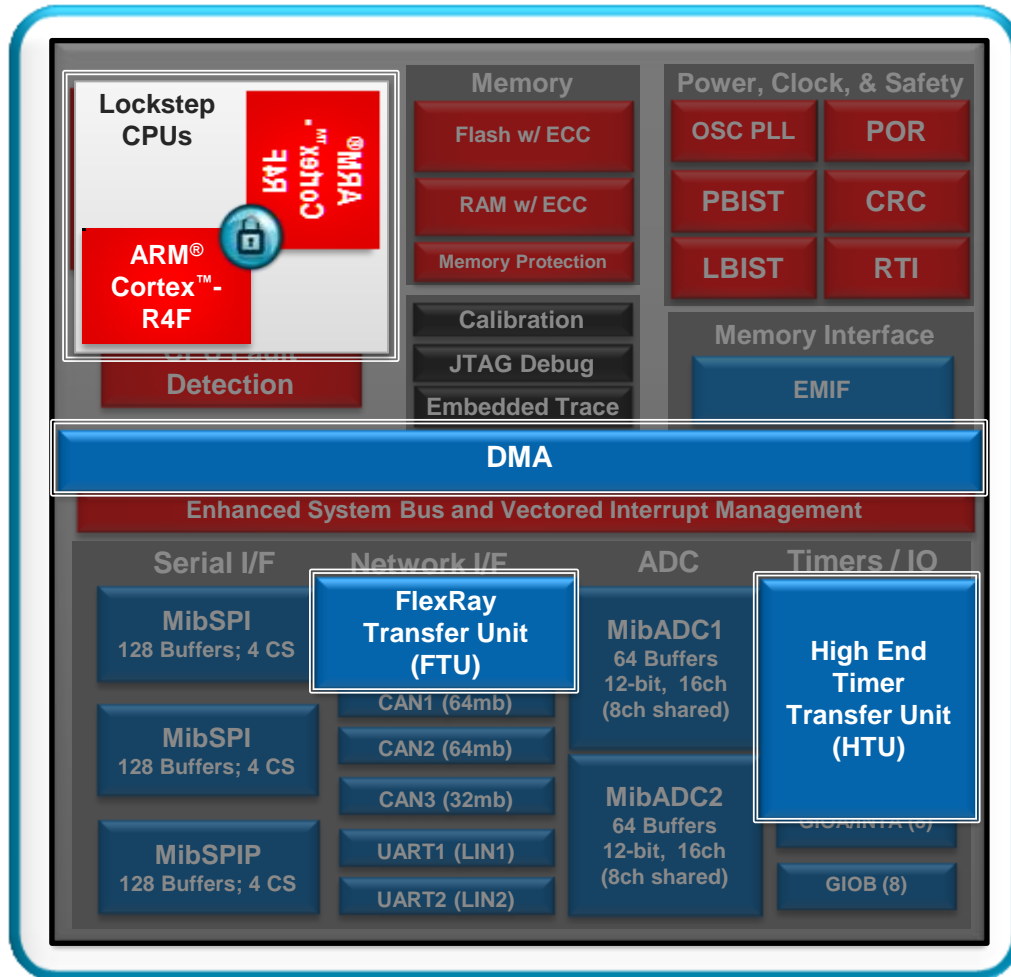
Digital Windowed Watch Dog (DWWD)

- The DWWD module will reset the MCU or generate a non maskable interrupt to the CPU if the application fails to service the watchdog to within the appropriate time window.
 - Optional safety diagnostic that can detect a runaway CPU
 - Includes a 25-bit down counter
 - Alerts the Error Signaling Module when a CPU interrupt is generated
 - Supports multiple service windows: 100%, 50%, 25%, 12.5%, 3.125%
 - Servicing requires a specific two part key sequence
 - Once enabled can only be disabled by a system or power on reset



Memory Protection Unit (MPU)

- A Dedicated Memory Protection Unit (MPU) is implemented for select bus masters

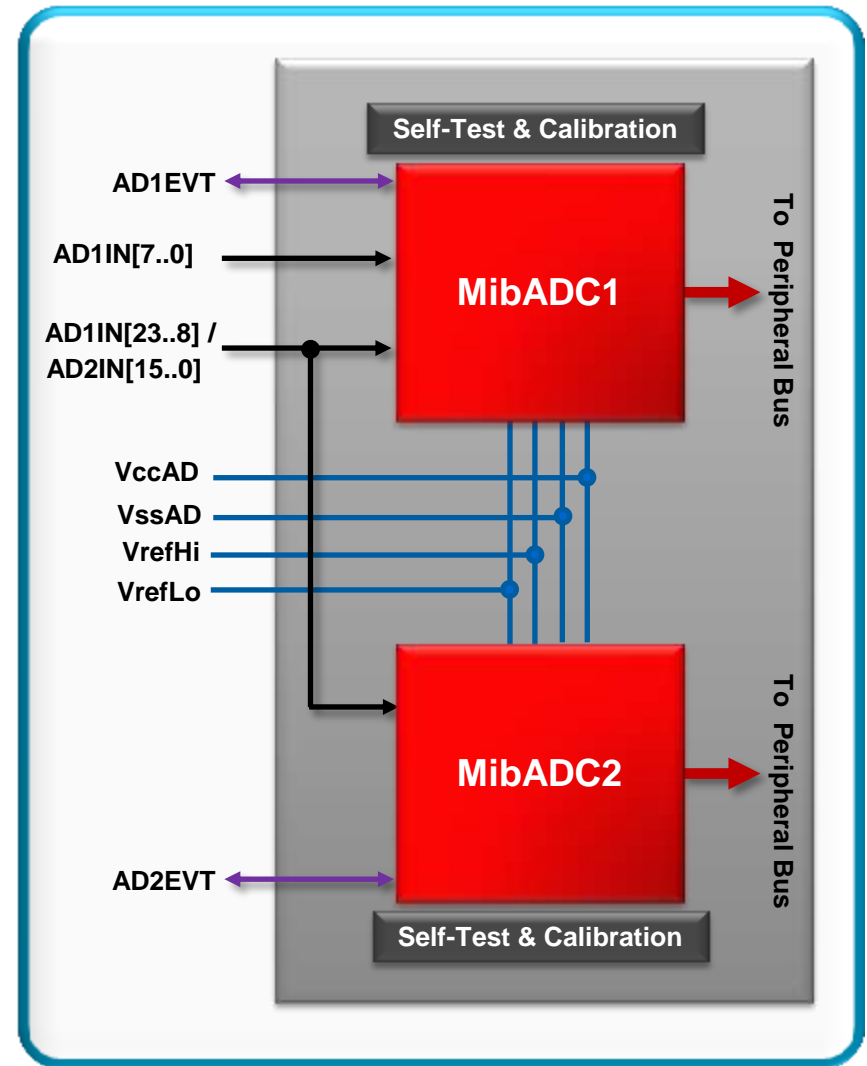


- Bus masters include the CPU, DMA, HTU and the FTU
- A memory region is defined which allows read and write access for the bus master
- Access outside the defined region can be any of the mode
 - **Read Only:** Read access allowed for the memory accesses outside the region. Write accesses are blocked
 - **No Access:** Read and write access is blocked.
- In the event of a memory protection violation an error is indicated

Dual Analog to Digital Converters

- Dual 12-bit ADC Cores:

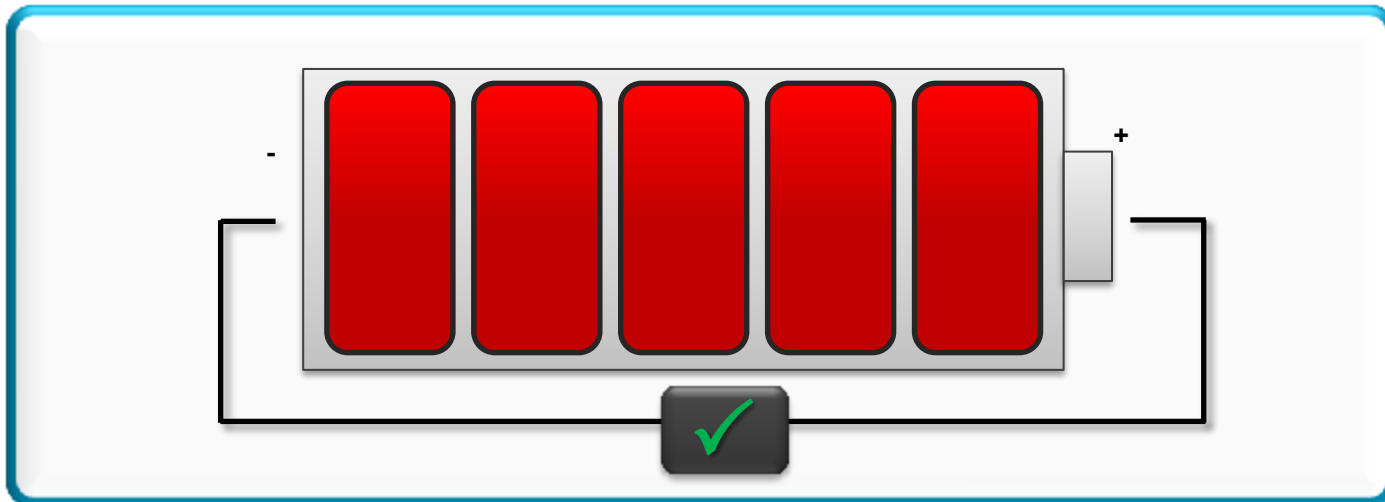
- MibADC 1 supports dedicated analog inputs & shared inputs with MibADC 2
- Up to 16 analog channels can be shared between the 2 cores for safety critical conversions/comparison
- Internal ADC reference voltages can be used to check converter functionality.
- Self Test Mode enables in application detection of opens/shorts on ADC inputs
- ADC calibration logic can improve accuracy or be used to detect drift between multiple test results.



Note: Not all Hercules MCUs are available with dual ADCs

Voltage Monitor

- Supply Voltage Monitor (VMON)
 - Holds reset until core and I/O rails in expected range (removes power sequencing requirements)
 - Asserts reset if core or I/O supply exceeds defined min/max thresholds
 - Asserts reset when core supply is below specified min voltage and asynchronously sets all I/O pins to high impedance mode



Exida Has Certified TMS570LS20216S IEC 61508 SIL 3 Capable



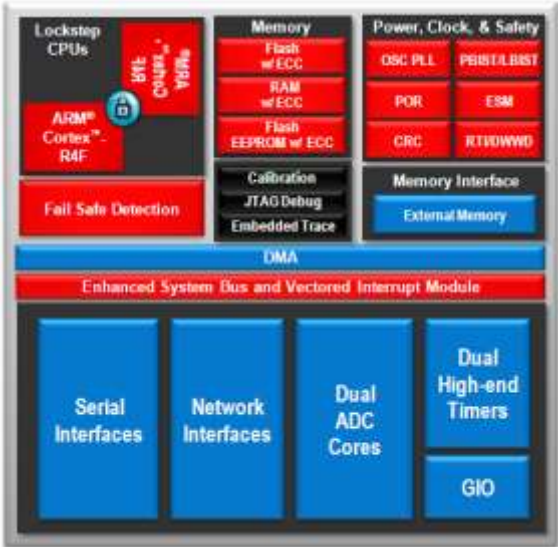
- Future Hercules TMS570 and RM products are planned to be assessed by Exida and/or TÜV-SÜD
- To get a Safety Certificate, Hercules MCUs are assessed for:
 - Development Process
 - Product Safety Architecture/Concept
 - Silicon
- TI has chosen to work with TÜV-SÜD (DAkkS accredited) and Exida (ANSI accredited) for both IEC 61508:2010 and ISO 26262:2011

Hercules™ Safety MCUs Provides Developers

Hercules MCUs provide developers of safety-critical applications:

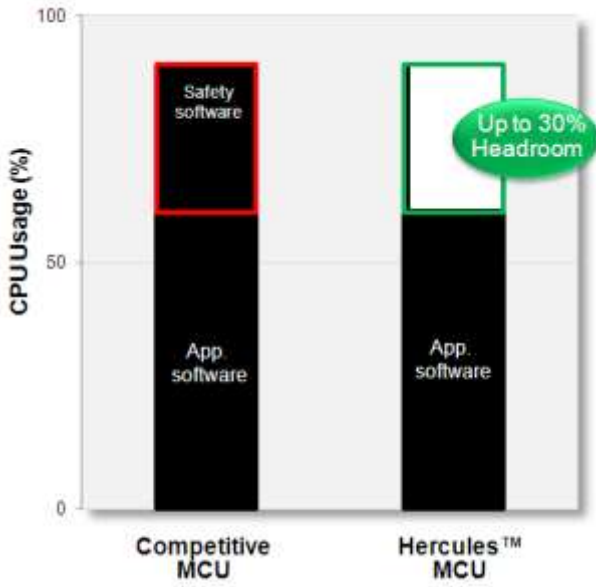
- ▶ Protection against random and systematic failures
- ▶ Headroom for application differentiation
- ▶ Simplified development and system certification

Hardware Safety Architecture



- Lockstep CPUs
- CPU & RAM Built-in Self Test
- Flash & RAM ECC
- Clock Monitoring
- Voltage Monitoring

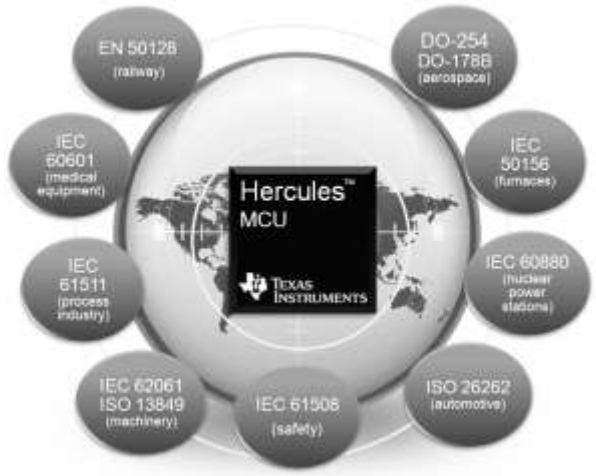
Application Headroom



Over 350 DMIPs of processing power



Safety Certification & Documentation



Hercules™ MCU Development Tools

Hercules™ IDEs, Compilers and RTOS Support

IDEs & Compilers



- TI Code Composer Studio – compiler qualification kit



- Embedded Workbench for ARM is certified by TÜV SÜD as suitable for use to IEC 61508 and ISO 26262



- MDK-ARM with uVision IDE and ARM C/C++ Compilation Tools



- MULTI IDE and Green Hills Compiler certified to ISO 26262 and IEC 61608



- Tantino-Cortex-R4 with professional HiTOP Debugger/IDE



- CoDeSys programming system and runtime system for IEC 61131-3 programmable logic controllers



- TargetLink code generation from MathWorks Simulink/Stateflow, certified for IEC 61508



- Processor-In-the-Loop (PIL) with MathWorks Simulink



- HET IDE with Synapticad WaveViewer or WaveFormer Pro

RTOS Support



- **FreeRTOS:** FreeRTOS.org
Portable, open source, royalty free, mini Real Time Kernel.



- **SafeRTOS:** High Integrity Systems Design assurance package for IEC61508, others



- **µC/OS:** Micrium
Certifiable design package for IEC61508, others



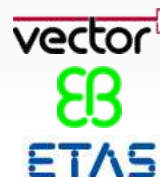
- **SCIOPTA:** SCIOPTA RTOS Kernel certified by TUV for IEC 61508 and EN50128 Hercules to SIL-3



- **CODESYS:** Smart Software Solutions Control and safety runtime system for Industrial PLCs



- **SMXRtos:** Micro Digital Modular RTOS that meets the needs of small to medium-size embedded systems



- **AUTOSAR OS/RTE:**
 - Vector MICROSAR Safe
 - ElektroBit tresos
 - ETAS RTA-OS & RTA-RTE
 - TI MCAL available for AUTOSAR v4.0.3

Hercules™ MCU Software Tools

Demos, Libraries & Example Code



Safety MCU Demos

- Safety Feature Highlight
- Ambient Light & Temperature Demo
- LED Light Show
- Maze Game
- Source Code Viewable via CCS



HALCoGen

- User Input on High Abstraction Level
- Graphical-based code generation
- Easy configuration
- Quick start for new projects
- Supports CCS, IAR, KEIL & GHS IDEs



Motor Software

- InstaSPIN-BLDC MotorWare™ Project
- Sensored FOC with redundant SMO MotorWare™ Project



Libraries

- DSP & Math Library: Optimized for ARM® Cortex-R4 & CMSIS Compliant
- SafeTI™ Diagnostic Library: Executable form of the safety manual



Example Code

- [Hercules Code Repository WIKI](#)

MiddleWare



- MISRA-compliant embedded TCP/IP stack that supports both IPv4 and IPv6 protocols.
- USB Host & Device, File systems, etc...



- MISRA-compliant CANopen real-time protocol and device driver used in medical automation and automotive equipment.



- Ethernet Driver and light weight IP Stack



- USB Device Driver & CDC Class

- Many MiddleWare options available from RTOS providers

Flash Programming



Automated offline Programmers:

- Data I/O
- BP Micro Systems



In Circuit JTAG Programmers:

- SMH Technologies
- Checksum
- XJTAG
- CCS UniFlash



Code Composer Studio



- **Based on Eclipse industry standard for embedded debug tools**
 - Modern window environment
 - Advanced source code editor
 - Scalable multi-core/processor environment
 - Program and Debug Application via JTAG
 - Test Automation via Scripting
 - Available for Windows & Linux

- **Support across TI's Embedded Processing Portfolio**
 - MSP430
 - Stellaris/Tiva
 - C2000
 - Hercules
 - C5000 & C6000 DSP

- **Hercules™ Debug Features**
 - 6 Hardware Breakpoints
 - Unlimited Software Breakpoints
 - Integrated Flash Programming

The screenshot shows the Code Composer Studio interface for a project named 'modemtx'. The main window displays C source code for a modem transmitter example. The code includes a main function with several variables and a loop. The memory window shows the current state of memory, with the 'main' function's stack frame visible. The console window shows the assembly code being executed, including instructions like 'B7H, D222', 'B9, *B--(2)', and 'M7H, B1'.



CCS Licensing Options

- **Evaluation:** Free, time limited licences for evaluation
- **Node Locked:** Tied to a specific computer
- **Floating:** Can be shared across multiple
- **Free/Dev Kit:** Can be used with Hercules kits w/ XDS100 emulators

<http://www.ti.com/tool/ccstudio>



Code Composer Studio Components:

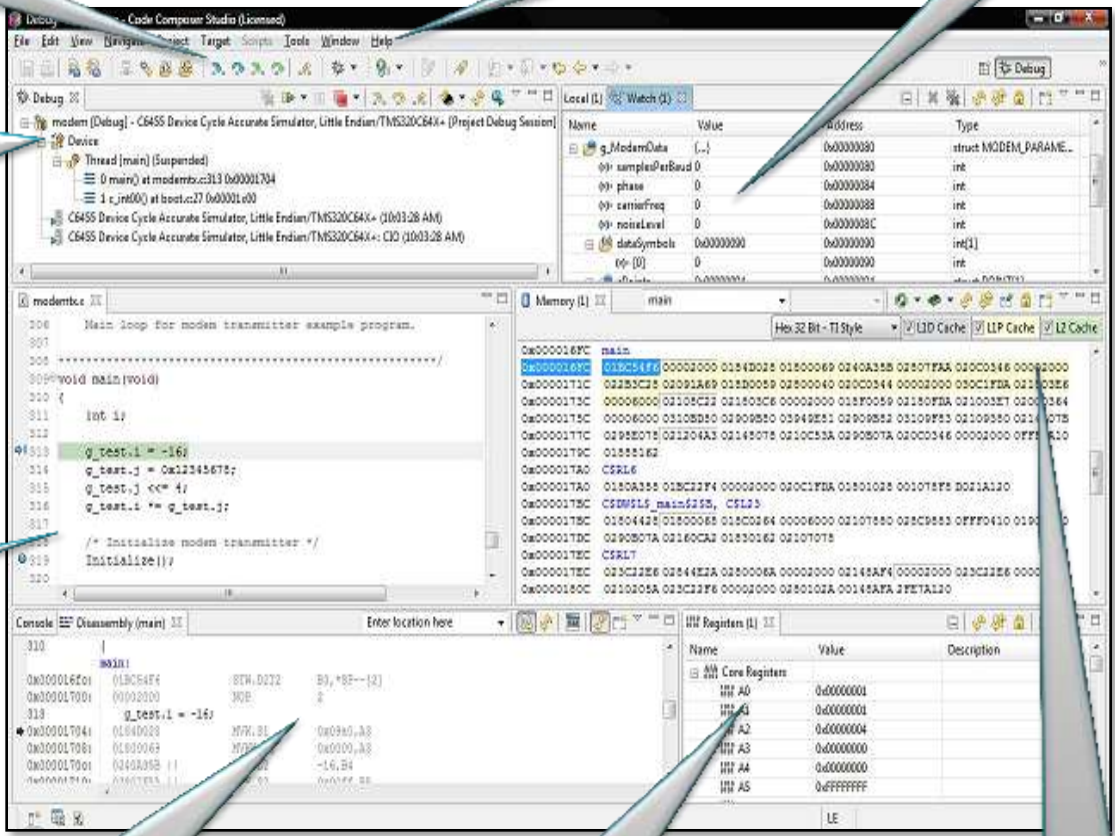
Menus and Icons

Help

Watch Window

Target Connection

- Source & object files
- File dependencies
- Compiler, assembler & linker build options



Source Code View

Disassembly Window

CPU Window

Memory Window



HALCoGen: Hardware Abstraction Layer Code Generator

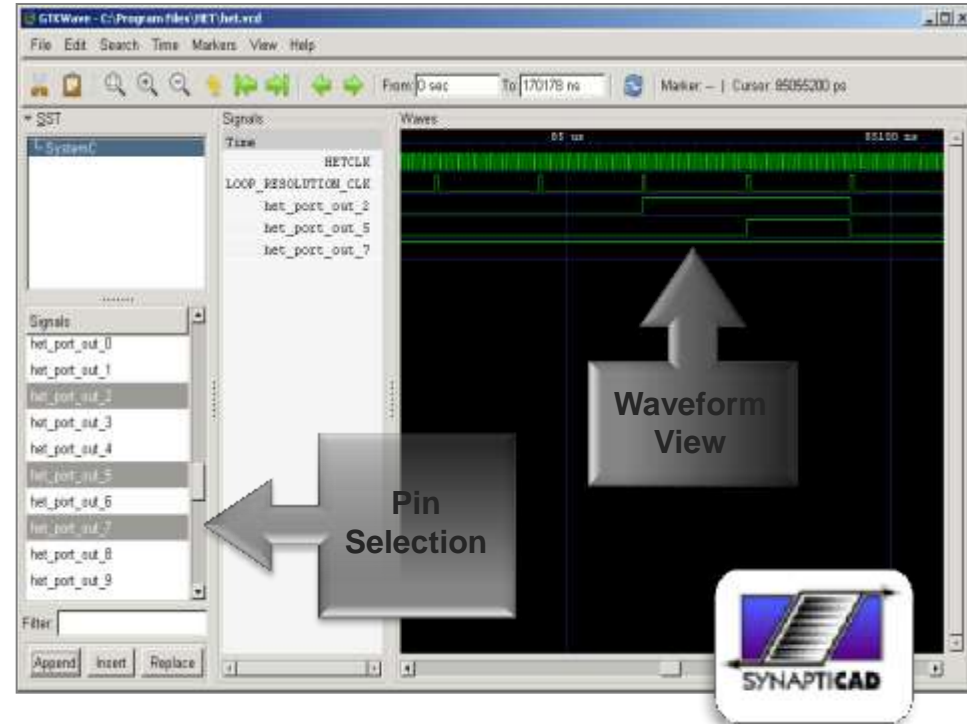
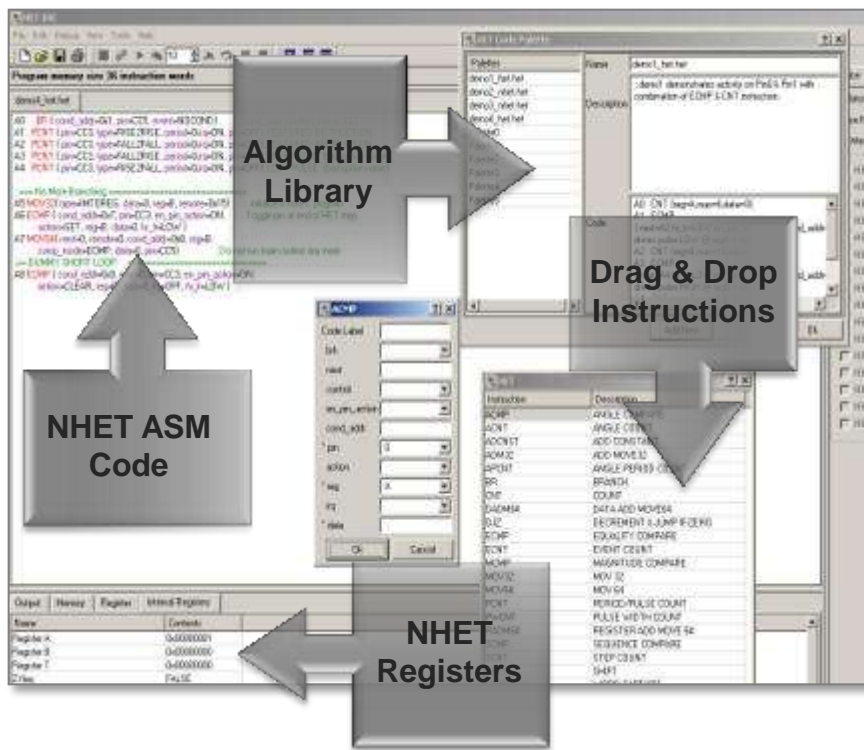
HAL
CoGen

The image displays three overlapping screenshots of the HALCoGen software interface. The top screenshot shows a C code editor with a function `canEnableErrorNotification` and a Device Explorer window showing a project structure for TMS570LS20216S2WT. The middle screenshot shows the HALCoGen Help window, displaying a file reference for `C:/HALCoGen/examples/example_rtiBlinky.c` and a dependency graph. The bottom screenshot shows the hardware configuration window for Port A and Port B, with a VIM (Virtual Input Module) configuration and an output window showing the generated code for LIN2 and HET.

Features

- **User Input on High Abstraction Level**
 - Graphical-based code generation
 - Easy configuration
 - Quick start for new projects
- **Generates C Source Code**
 - ANSI Conforming
 - Clear, structured, coding style
 - Customizable code for user maintenance
- **Supported Drivers**
 - **System Modules**
 - *Safety Init, MPU, PMU, PMM, PCR*
 - *LBIST, PBIST, VIM, ESM, CRC*
 - *EMIF, POM, DMA, PINMUX*
 - **Peripheral Modules**
 - *RTI, GIO, ADC*
 - *SCI/LIN, CAN, MIBSPI / SPI, I2C*
 - *USB, Ethernet*
 - *Timer Co-processor (NHET)*
 - *eCAP, eQEP, ePWM*
- **Interactive Help System**
 - Describes tool features and functions
 - Provides detailed dependency graphs
 - Provides useful example code
 - Tool tip help available
- **Native support for CCS, KEIL, IAR and GHS IDEs**

NHET Timer Co-Processor Development Tools



- Graphical Programming Environment
- Output Simulation Tool
- Generates CCS-ready software modules
- Includes functional examples from TI

- Graphical Waveform Viewer
- Input Generation Tool
- Seamless interface to coding tool
- Upgradable to Full SynapticAD

Hercules™ Development Kit Overview

Motor Control Kit



Spin 3 phase Brushless DC and Brushless AC Motors

Starting at \$499

[TMS570LS31](#), [LS12](#)

[RM48](#), [RM46](#)

SafeTI™-HSK



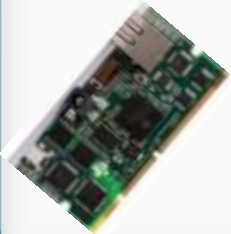
Evaluate Hercules MCU and TPS65381 Combination for Safety-critical Applications

\$499

[TMS570LS31](#)

[RM48](#)

controlCARD



Initial Software Development and Short-run Builds for System Prototypes

Starting at \$99

[TMS570LS31](#),

[LS12](#)
[RM48](#), [RM46](#)

HDK



Get Started on Development with Hercules MCU Platform

\$199

[LS31](#), [LS12](#), [LS04](#)

[RM48](#), [RM46](#), [RM42](#)

USB Stick



Low-cost Option to Evaluate Hercules MCU Platform

\$79

[TMS570LS31](#)

[RM48](#)

LaunchPad



Lowest cost Option to Evaluate Hercules MCU Platform

\$19.99

[TMS570LS04](#)

[RM42](#)

Hercules™ Development Kits

Evaluation

TMDXRM48USB – RM48 USB Stick Kit
TMDX570LS31USB – TMS570 USB Stick Kit
TMDX470MF066USB – TMS470M USB Stick Kit

- USB Powered
- On Board USB XDS100v2 JTAG Debug
- On Board SCI to PC Serial Communication
- Access to Select Signal Pin Test Points
- LEDs, Temp Sensor & Light Sensor
- Accelerometer
- CAN transceiver



Development

TMDXRM48HDK – RM48 Development Kit
TMDXRM46HDK – RM46 Development Kit
TMDXRM42HDK – RM42 Development Kit
TMDX570LS31HDK – TMS570 Development Kit
TMDX470MF066HDK – TMS470M Development Kit

- On Board USB XDS100v2 JTAG Debug
- External high speed emulation via JTAG
- CAN Transceivers
- LEDs, Temp Sensor & Light Sensor
- TRACE pads for ETM/RTP/DMM (TMS570 & RM48)
- RJ45 10/100 ENET (TMS570 & RM48 & RM46)
- USB-A Host Interface (RM48 & RM46)
- USB-B Device Interface (RM48 & RM46)



Software Included in Each Kit:

- CCStudio IDE: C/C++ Compiler/Linker/Debugger
- HALCoGen Peripheral Driver Generation Tool
- CCS and nowFlash Flash Programming Tools
- HET IDE: Simulator & Assembler
- GUI Demo with Project/Code Examples

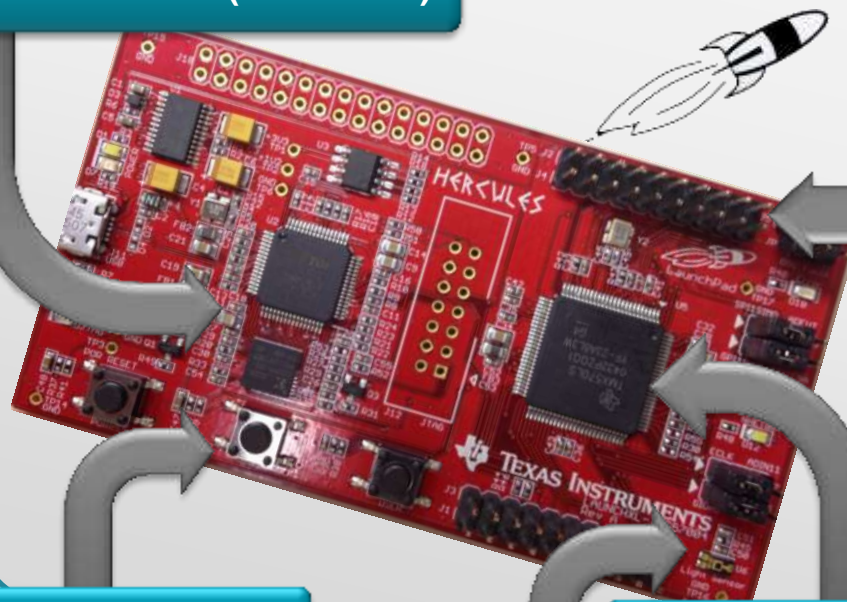


Hercules™ LaunchPad

- LAUNCHXL-RM42
- LAUNCHXL-TMS57004

BoosterPack XL Interface

On Board JTAG (XDS100v2)



GIO Push Button

Ambient Light Sensor

Hercules™ Safety MCU

\$19.99

LaunchPad Demos



Kit Overview

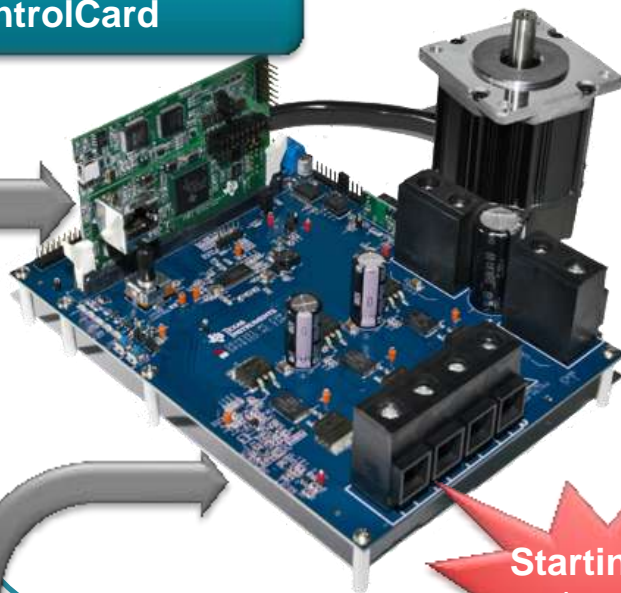
- USB powered
- On board USB XDS100v2 JTAG debug
- On board SCI to PC serial communication
- GIO & NHET LEDs
- Ambient Light sensor
- 40 pin BoosterPack XL Header
- Footprint for an Expansion header (not populated) to bring out all MCU Pins
- USB Cable
- Quick Start Guide

Hercules™ Motor Control Development Kits

- DRV8301-RM48-KIT
- DRV8301-RM46-KIT
- DRV8301-TMS570LS31-KIT
- DRV8301-TMS570LS21-KIT

Teknic 7 Amp, 6000 RPM Motor with encoder and hall sensors

Hercules Safety MCU ControlCard



DRV 8301 60V, 60A EVM with self protection and programmable gain amplifiers

Starting at \$499



Demo Software



Software Included In the Kit

- InstaSPIN-BLDC MotorWare™ Project
- Sensored FOC with redundant SMO MotorWare™ Project
- Demonstration Examples
- Code Composer Studio IDE
- ARM Cortex-R4 CMSIS DSP Library
- HALCoGen Driver Generation Tool
- High End Timer IDE

SafeTI™ Hitex Safety Kit

- **SAFETI-HSK-RM48**
- **SAFETI-HSK-570LS31**

TPS65381 Power Supply & Safety Monitor

On Board Display



ControlCard Interface

Hercules™ Safety MCU

\$499

Hitex Safety Kit Software



Kit Overview

- Cost effective entry into functional safety related to ISO26262 and IEC61508
- Evaluation board supporting all safety features according to the safety manual
- Error injection and reaction monitoring by second μC connected to GUI
- Full source code available for modification of the application or including the library in your own application
- Evaluation version of compiler and debugger included
- Evaluation version of SafeRTOS included
- User friendly documentation

-
-

Hercules™ LaunchPad

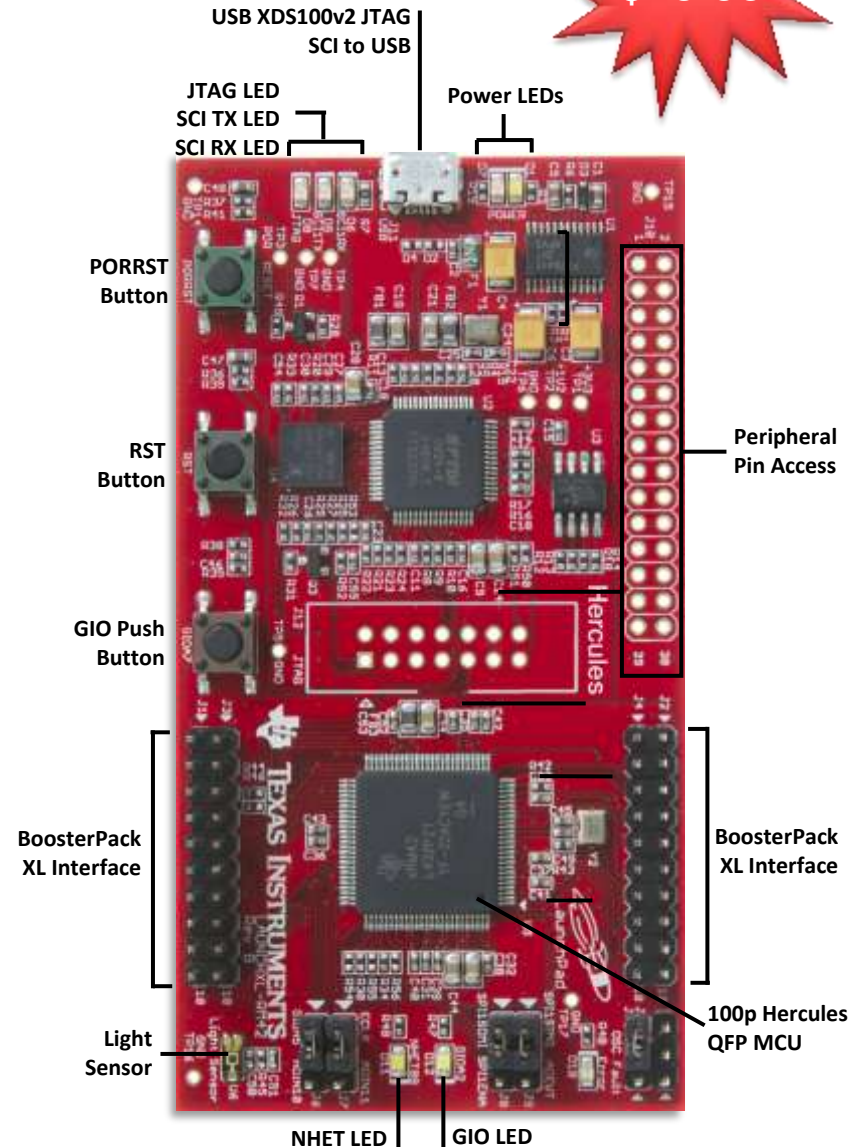
\$19.99

- **Main Features**

- USB Powered
- On Board USB XDS100v2 JTAG Debug
- On Board SCI to PC Serial Communication
- BoosterPack XL Interface
- Access to Select Signal Pin Test Points
- GIO Push Button
- 1 White GIO LED
- 1 White NHET LED
- 1 Red nERROR LED
- Ambient Light Sensor
- 100p QFP Packaged MPU

- **\$19.99 USD**

- **TMS570 Orderable Part #**
 - **LAUNCHXL-TMS57004**
- **RM42 Orderable Part #**
 - **LAUNCHXL-RM42**

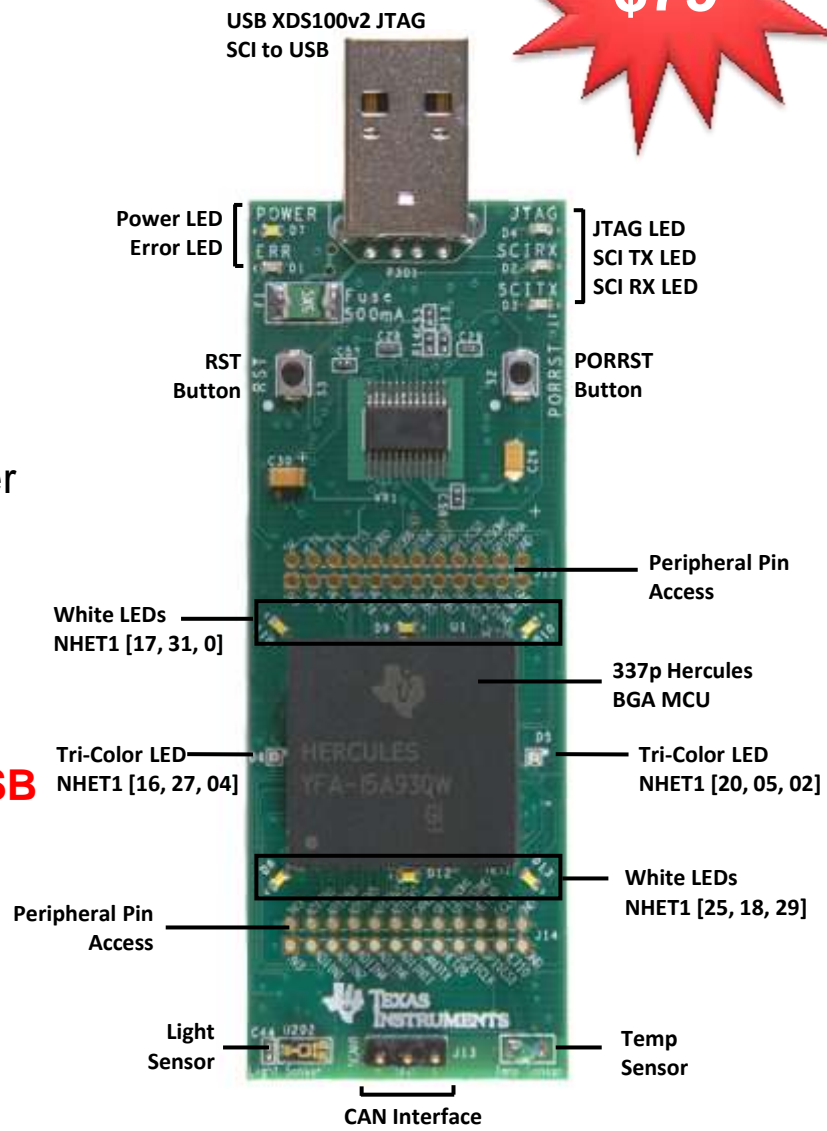


Microcontroller USB Stick Kit

\$79

- **Main Features**
 - USB Powered
 - On Board USB XDS100v2 JTAG Debug
 - On Board SCI to PC Serial Communication
 - Access to Select Signal Pin Test Points
 - 6 White NHET LEDs
 - 2 RGB Tri Color NHET LEDs
 - Temp Sensor, Light Sensor and Accelerometer
 - CAN Communication transceiver
 - 337p BGA Packaged MCU

- **\$79 USD**
 - **TMS570 Orderable Part # TMDX570LS31USB**
 - **RM48 Orderable Part # TMDXRM48USB**
 - Code Composer Studio IDE included



Hercules™ Development Kit (HDK)

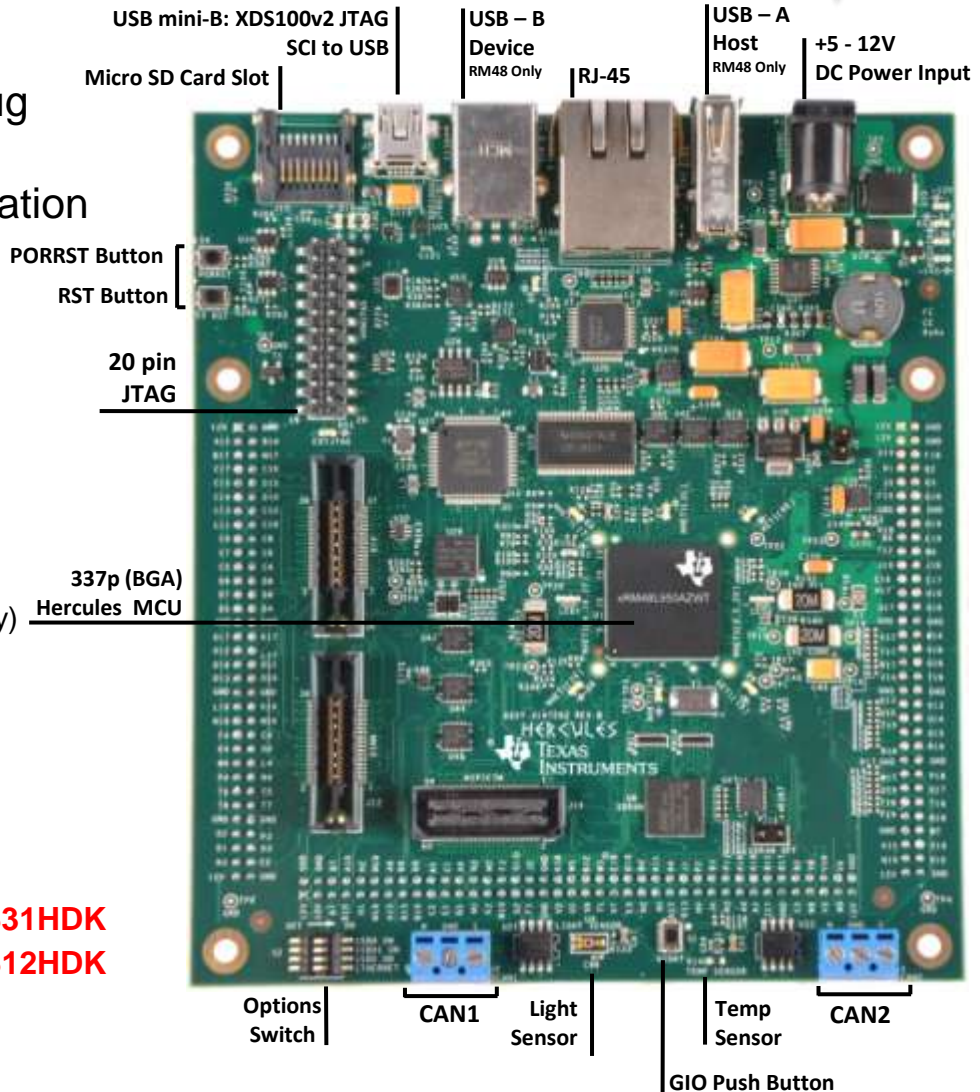
\$199

• Main Features

- On Board USB XDS100v2 JTAG Debug
- External High Speed JTAG Connector
- On Board SCI to PC Serial Communication
- Access Signal Pin Test Points
- Board Expansion Connectors
- 6 White NHET LEDs
- 2 RGB Tri Color NHET LEDs
- Temp Sensor and Light Sensor
- Micro SD Card Slot (SPI mode)
- RJ45 10/100 Ethernet Interface
- USB-B Device Interface (RM48 & RM46 Only)
- USB-A Host Interface (RM48 & RM46 Only)
- 2 CAN Communication Transceivers
- 337p BGA Packaged MCU

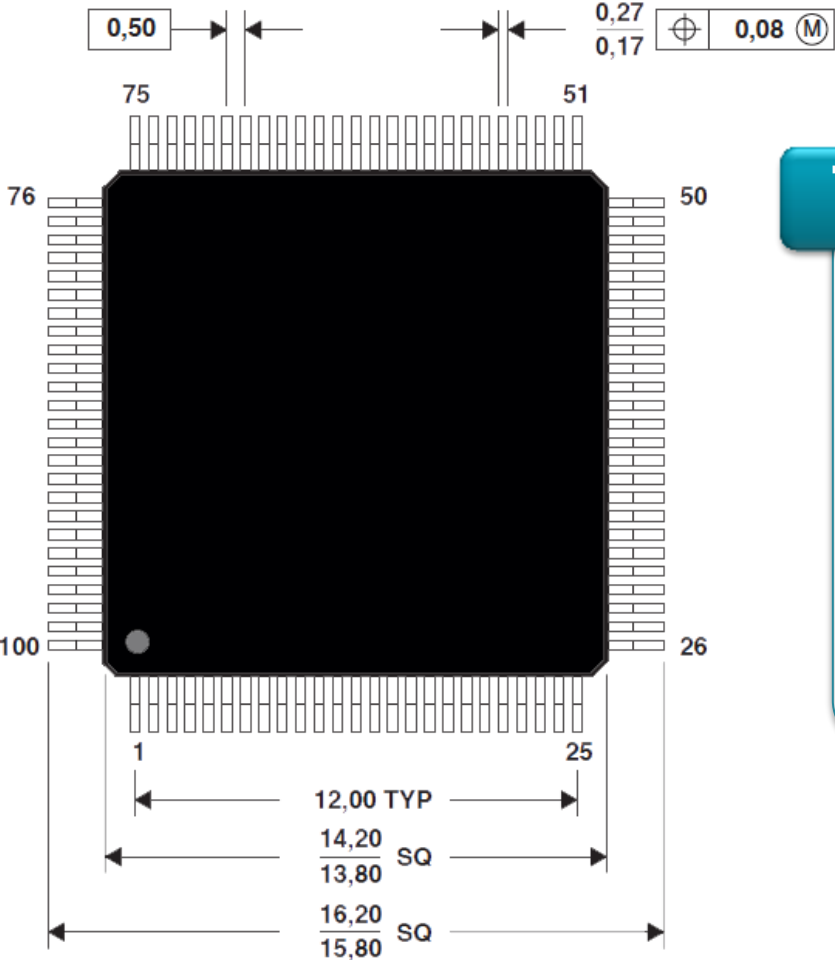
• \$199 USD

- **TMS570LS31 Orderable Part # TMDX570LS31HDK**
- **TMS570LS12 Orderable Part # TMDX570LS12HDK**
- **RM48 Orderable Part # TMDXRM48HDK**
- **RM46 Orderable Part # TMDXRM46HDK**



Printed Circuit Board Design Considerations

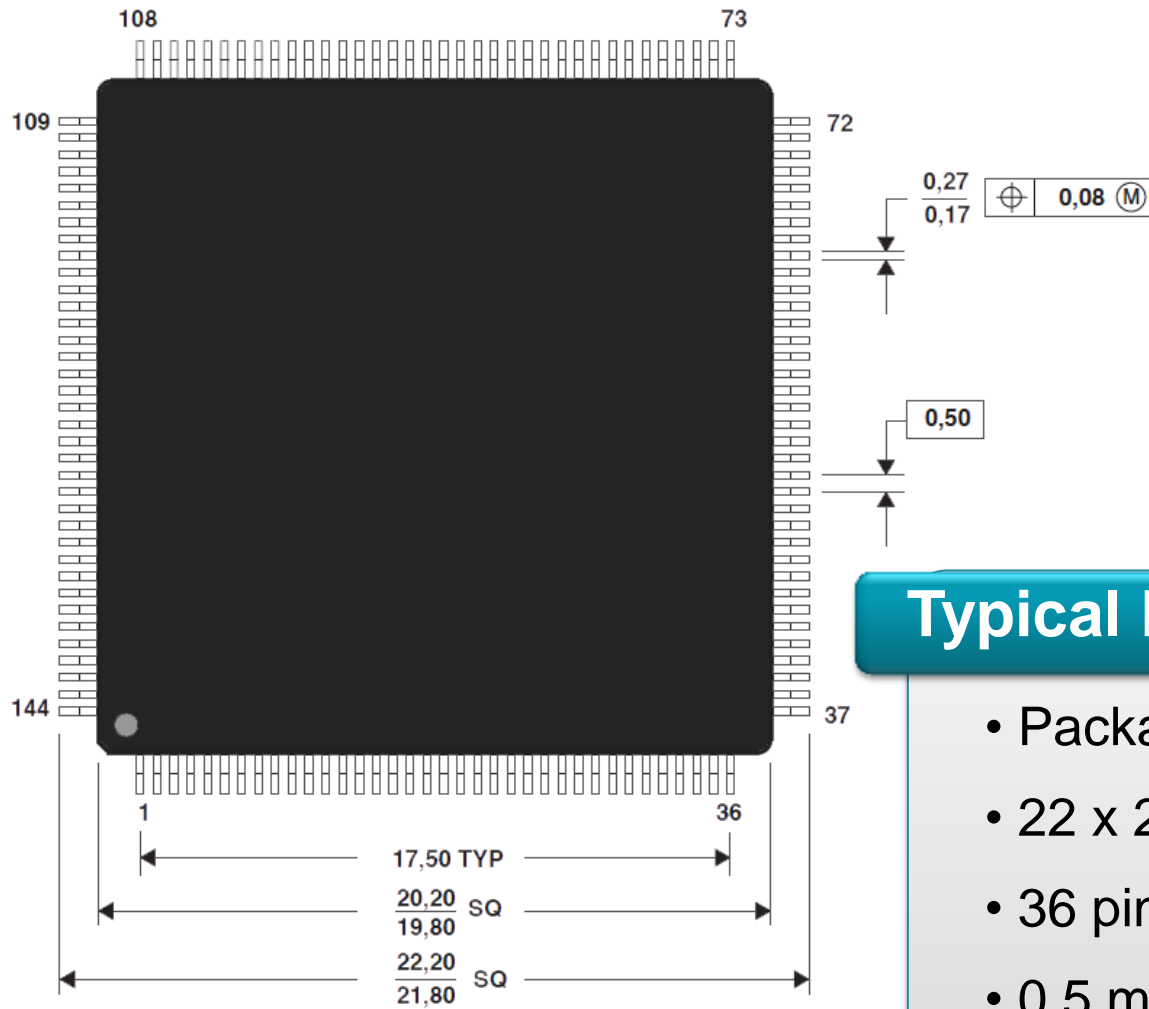
MCU Package Selection – 100p (PZ) QFP



Typical Package Information:

- Package Size = 14x14 mm
- 16 x16 mm including leads
- 25 pins on each side
- 0.5 mm pin pitch

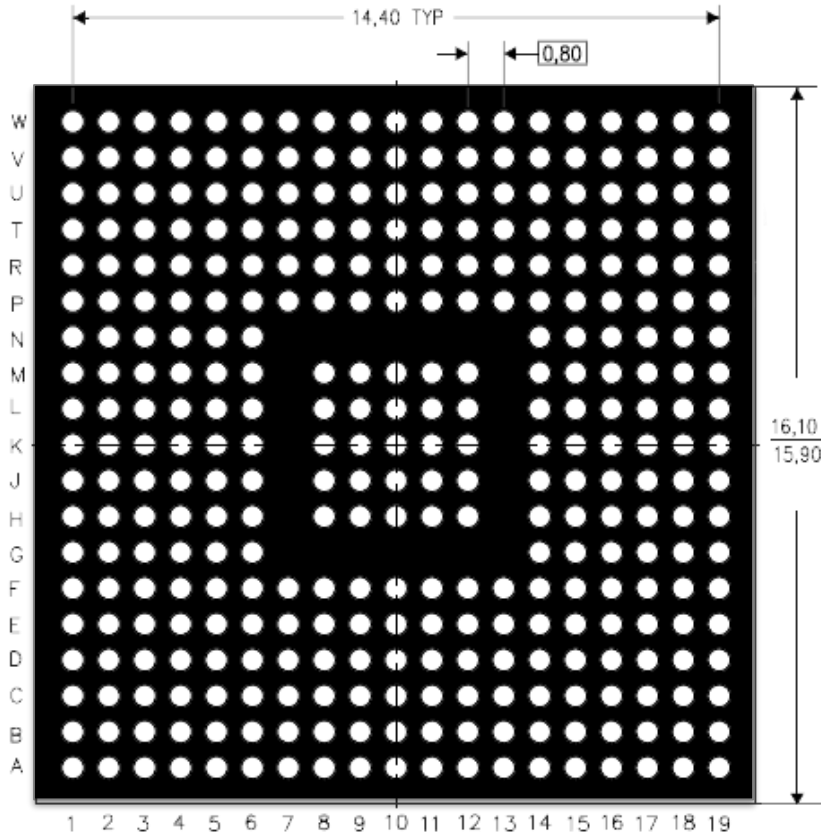
MCU Package Selection – 144p (PGE) QFP



Typical Package Information:

- Package Size = 20 x 20 mm
- 22 x 22 mm including leads
- 36 pins on each side
- 0.5 mm pin pitch

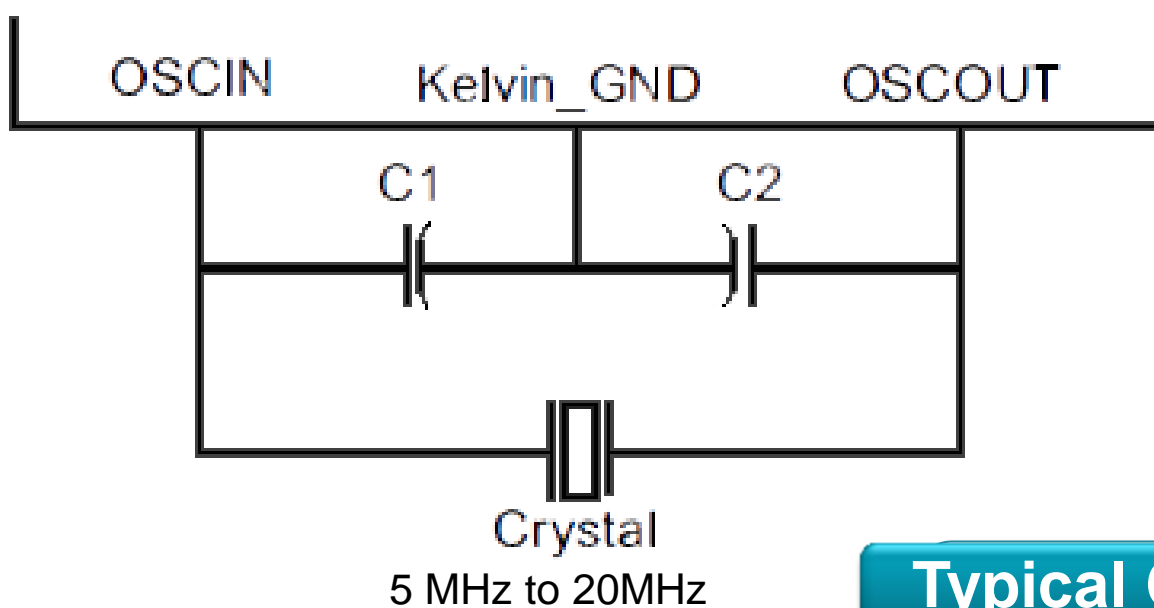
MCU Package Selection – 337p (ZWT) BGA



Typical Package Information:

- Package Size = 16 x 16 mm
- 337 ball grid array
- 0.8 mm ball pitch

MCU Crystal Selection – TMS570 & RM48



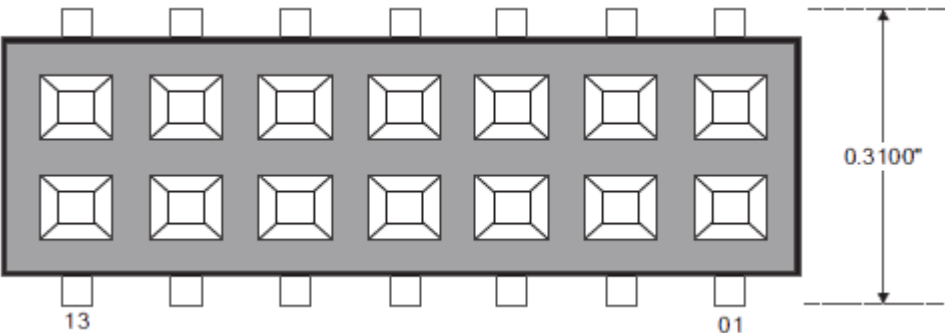
TI strongly encourages each customer to submit samples of the device to the resonator/crystal vendors for validation. The vendors are equipped to determine what load capacitors will best tune their resonator/crystal to the microcontroller device for optimum start-up and operation over temperature/voltage extremes.

Typical Crystal Connections:

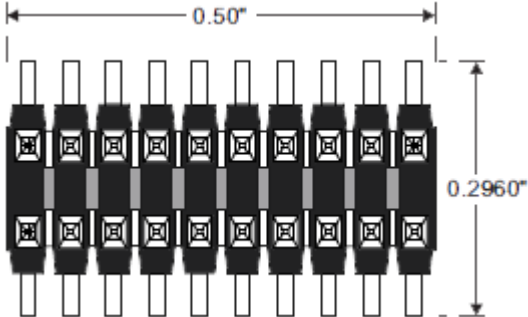
- OSCIN & OSCOUT pins
- BGA Packages: Kelvin GND
- Frequency Range: 5 to 25MHz

External JTAG Headers

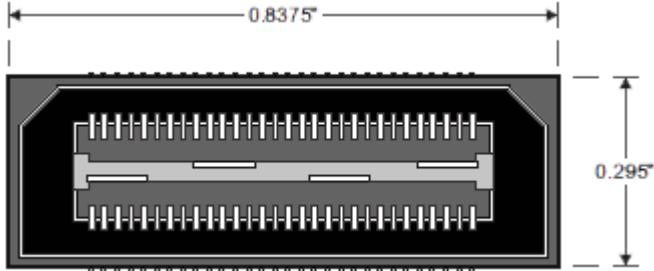
14 Pin TI



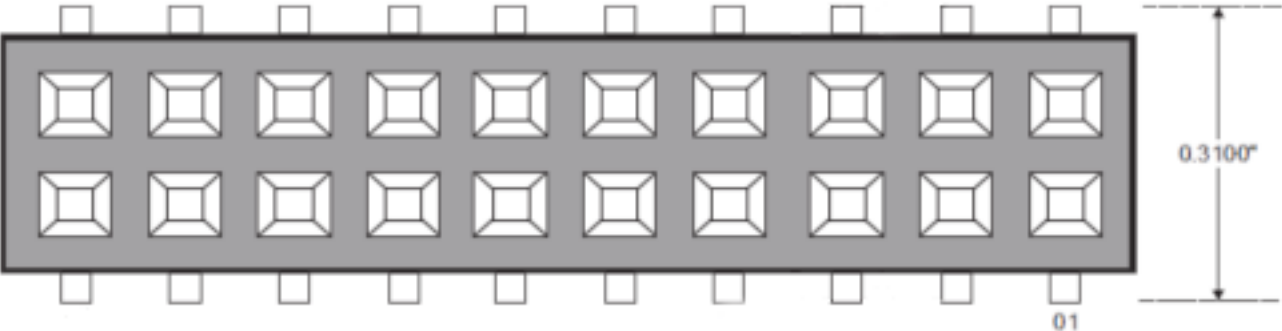
20 Pin CTI



60 Pin MIPI

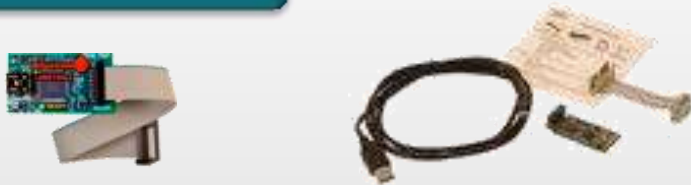


20 Pin ARM



External XDS JTAG Emulators

XDS100v2



- Entry level JTAG emulator
- USB interface
- 3 models based on JTAG headers (14pin TI, 20pin TI, 20/10pin ARM)
- Free CCS XDS100v2 license available
- \$79

XDS200



- Excellent balance of performance and cost
- USB interface (Ethernet version available)
- 20pin TI, 14pin TI, 20pin ARM and 10pin ARM connectors
- \$295

XDS560v2



- High performance JTAG emulator
- USB or USB + Ethernet interfaces
- Includes multiple JTAG adapters (14pin TI, 20pin TI, 20pin ARM, 60pin MIPI, some include 60pin TI)
- System Trace
- \$995 - \$1495

Pro Trace

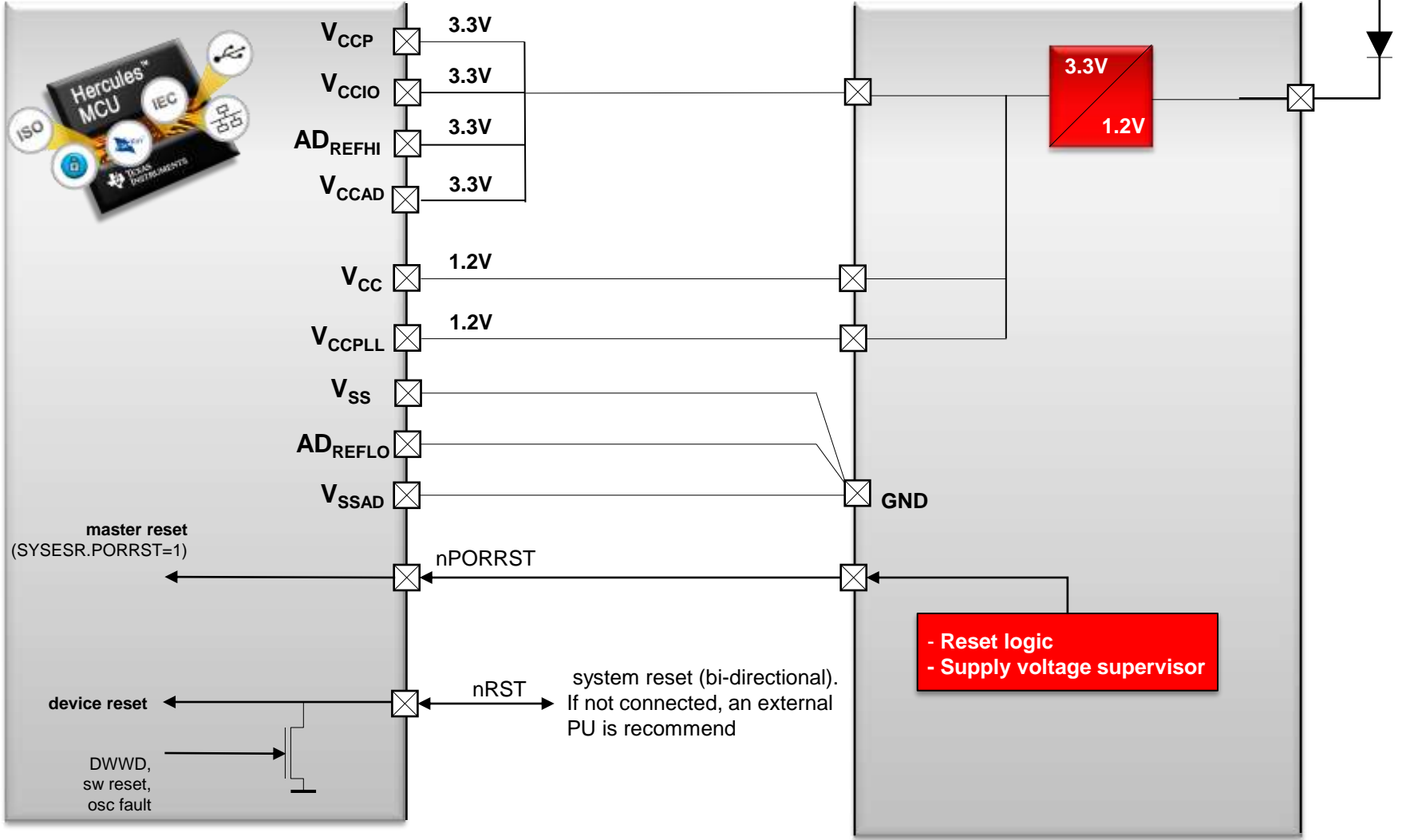


- Trace Receiver & XDS560v2 JTAG emulator
- USB + Ethernet interfaces
- MIPI60 and 60pin TI adapters
- DSP & ARM Trace to pins
- System Trace
- \$3495

Power Supply Requirements

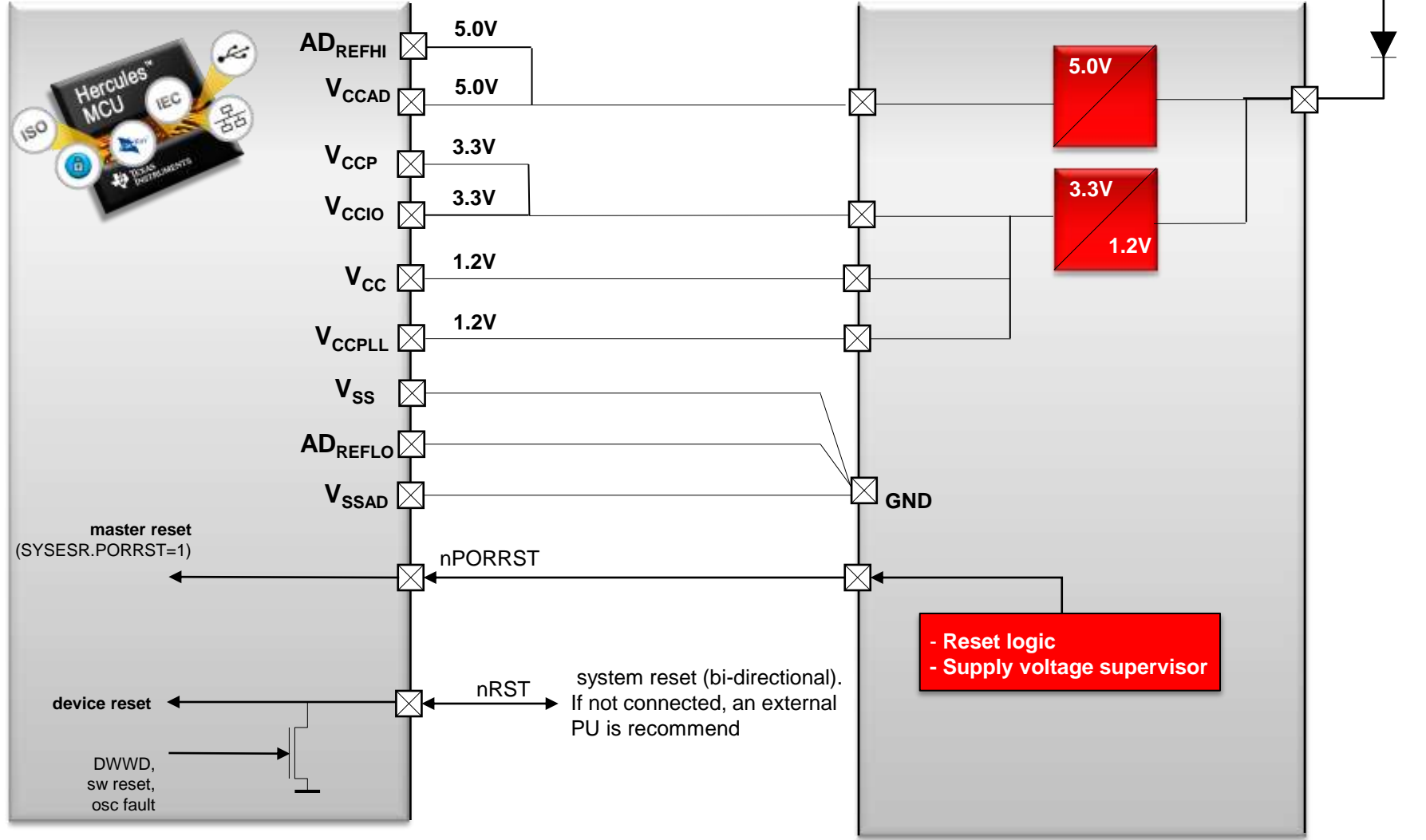
Hercules MCU:

TMS570LS31x/21x/12x/04x and RM48x/46x/42x



Power Supply Requirements

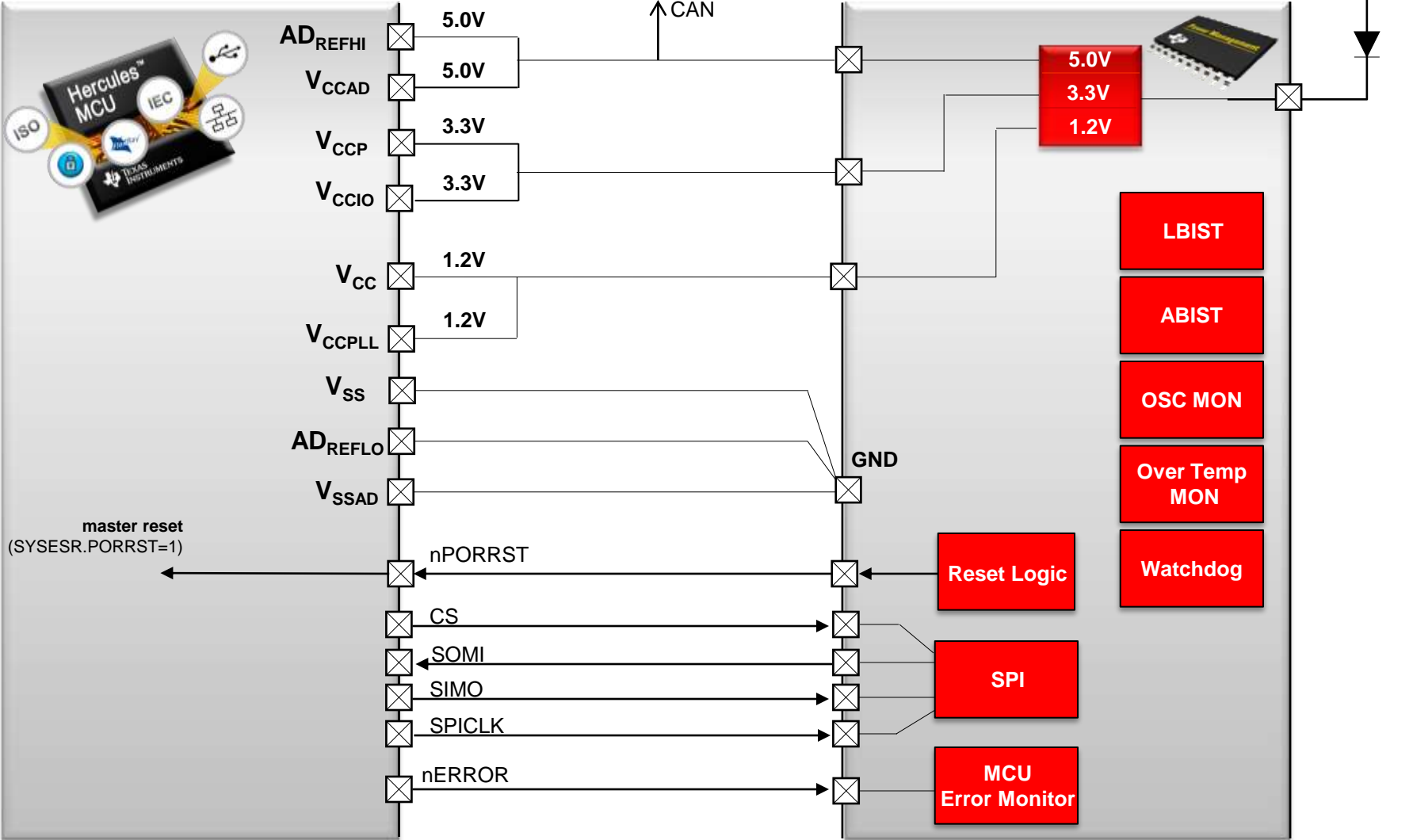
Hercules MCU:
TMS570LS31x/21x/12x/11x & RM48x/46x



Hercules & TPS65381 Safety Companion IC

Hercules MCU:

TMS570LS31x/21x/12x/11x & RM48x/46x



Note: Some Hercules MCU ADCs are only 3.3V capable

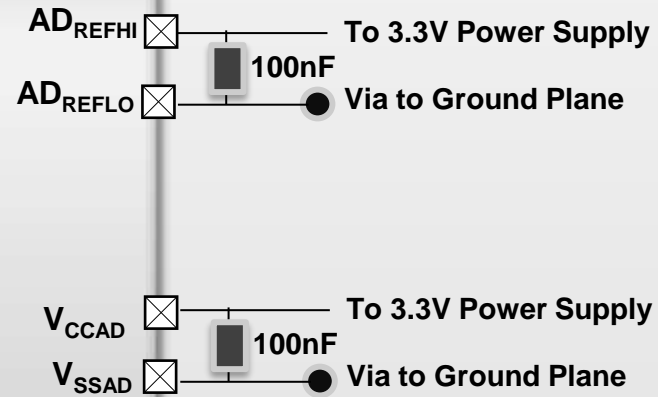
Package: 32-Pin HTSSOP PowerPAD™

For more information about the TPS65381 go to : <http://www.ti.com/product/tps65381-q1>



ADC Decoupling Capacitors

TMS570LSx or RM4x MCU



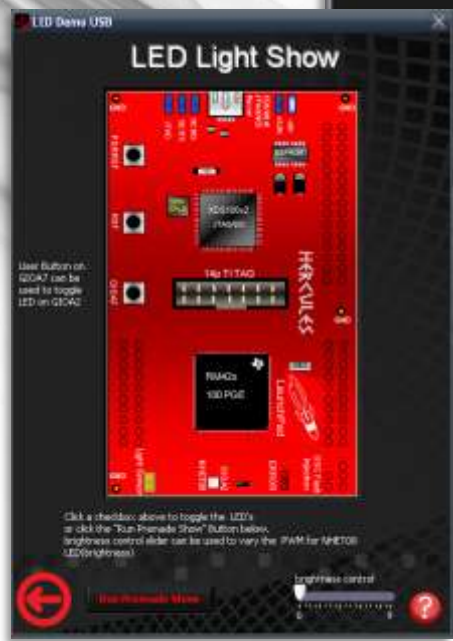
Exercise: Hercules Safety MCU Demos

Lab1: Hercules™ Safety MCU Demos

- To launch the demo software go to:



→ Programs → Texas Instruments → Hercules → Hercules Safety MCU Demos



Hercules Cortex™-R4F Architecture Overview: Memory Map, Clocking, Exceptions

High Performance Cortex-R4F floating-point CPU



ARM® v7R Cortex™ ISA fully backward Compatible to ARM7/9/11

Lockstep CPUs: Single core programming model – second core checks the first.

Supports ARM, Thumb and Thumb-2 instructions

Up to 220 MHz CPU Clock Speed

Single / double precision IEEE 754 floating-point

Fast MULT, DIV, and SQRT enables model-based control; simplifies algorithm implementation

Floating point and integer instructions operate in parallel

12 region memory protection

Superscalar, SIMD, 8 stage pipeline delivers 1.6 DMIPS/MHz

Broad IDE/Compiler Support: CCS, KIEL, IAR, GHS, etc...



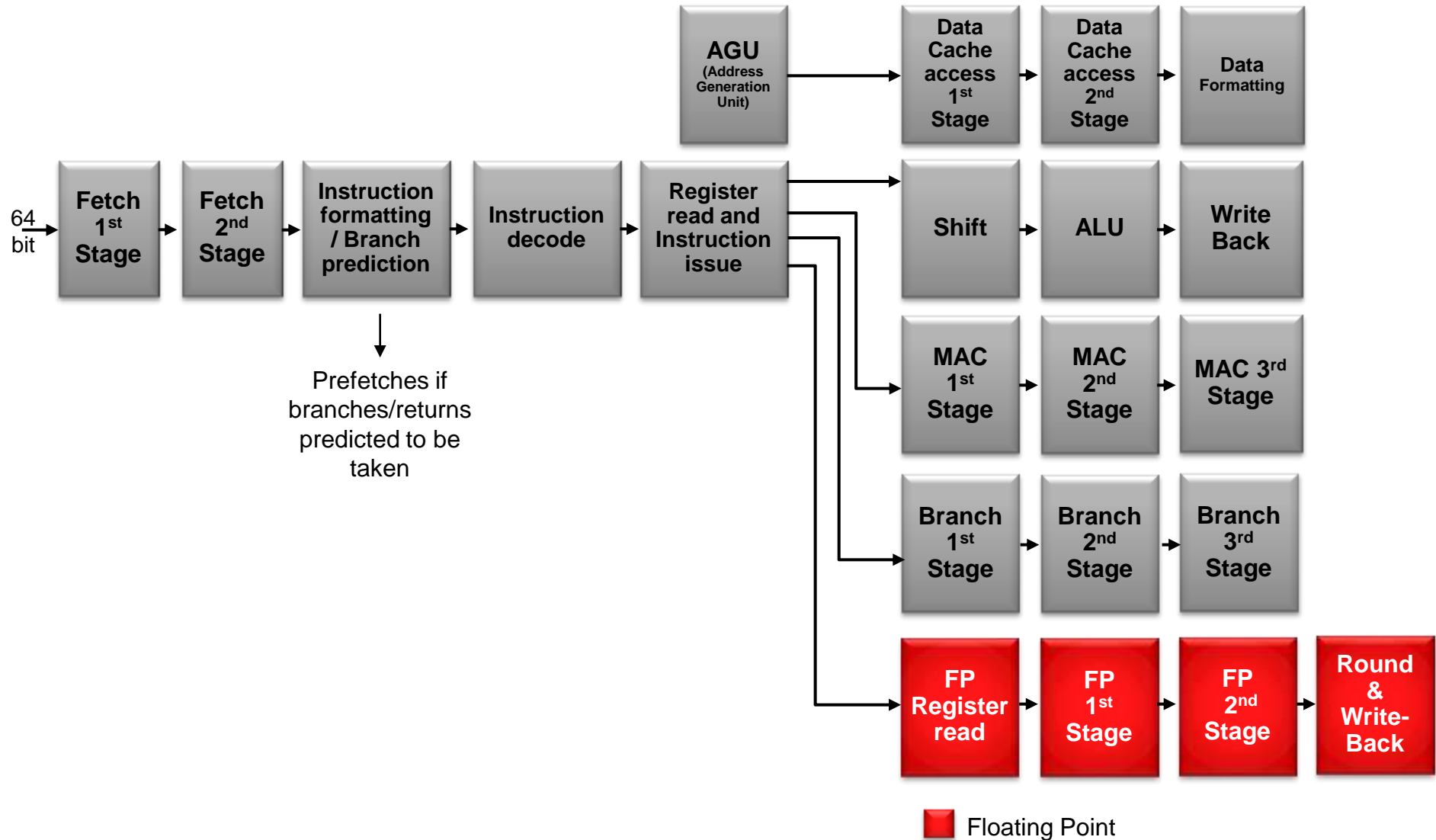
- Over 350 DMIPS of performance
- High performance floating point
- ARM-based: broad industry adoption

Scalable ARM Based Solutions from TI: Stellaris®, Concerto™ Hercules™ & Sitara™

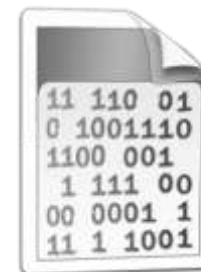
Cortex-R4F Features

- 32-bit ARM and 16/32-bit Thumb2 instruction set
- Integer unit with integral Embedded ICE-RT logic
- Dynamic branch prediction with a global history buffer and return stack
- Floating Point Unit
- Low interrupt latency
- Non-maskable interrupt
- Harvard level one memory system with:
 - *Tightly-Coupled Memories* (TCM) interfaces with support for error correction or parity checking memories
 - *Memory Protection Unit* (MPU)
- Level two memory interface:
 - Single 64-bit master interface
 - 64-bit slave interface, TCM RAM blocks and cache RAM blocks.
- Debug interface to a CoreSight® ETM-R4® or CoreSight DAP

Cortex-R4F Pipeline



Data Types

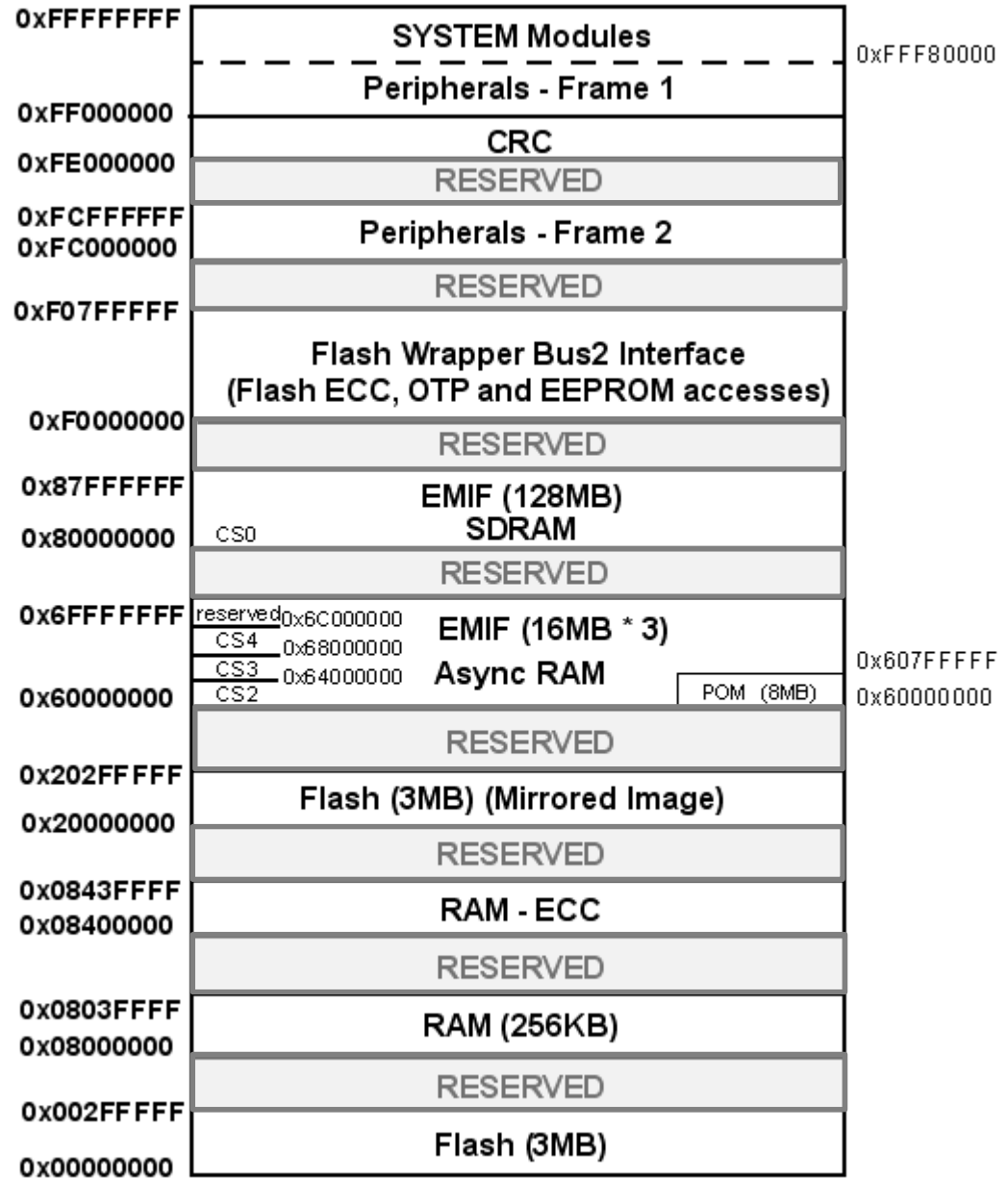


- The processor supports following data types
 - Double Word (64 bit)
 - Word (32 bit)
 - Half Word (16 bit)
 - Byte (8 bit)
- Although the processor supports unaligned accesses, TI does not recommend using unaligned accesses for bus performance
 - Above data types should be aligned at their respective size boundary
 - Most unaligned accesses are converted into multiple aligned accesses
- The TMS570 devices store their data in word invariant big endian format (BE32) due to a modification in the memory interface
- The RM4 devices store their data in little endian format (LE)

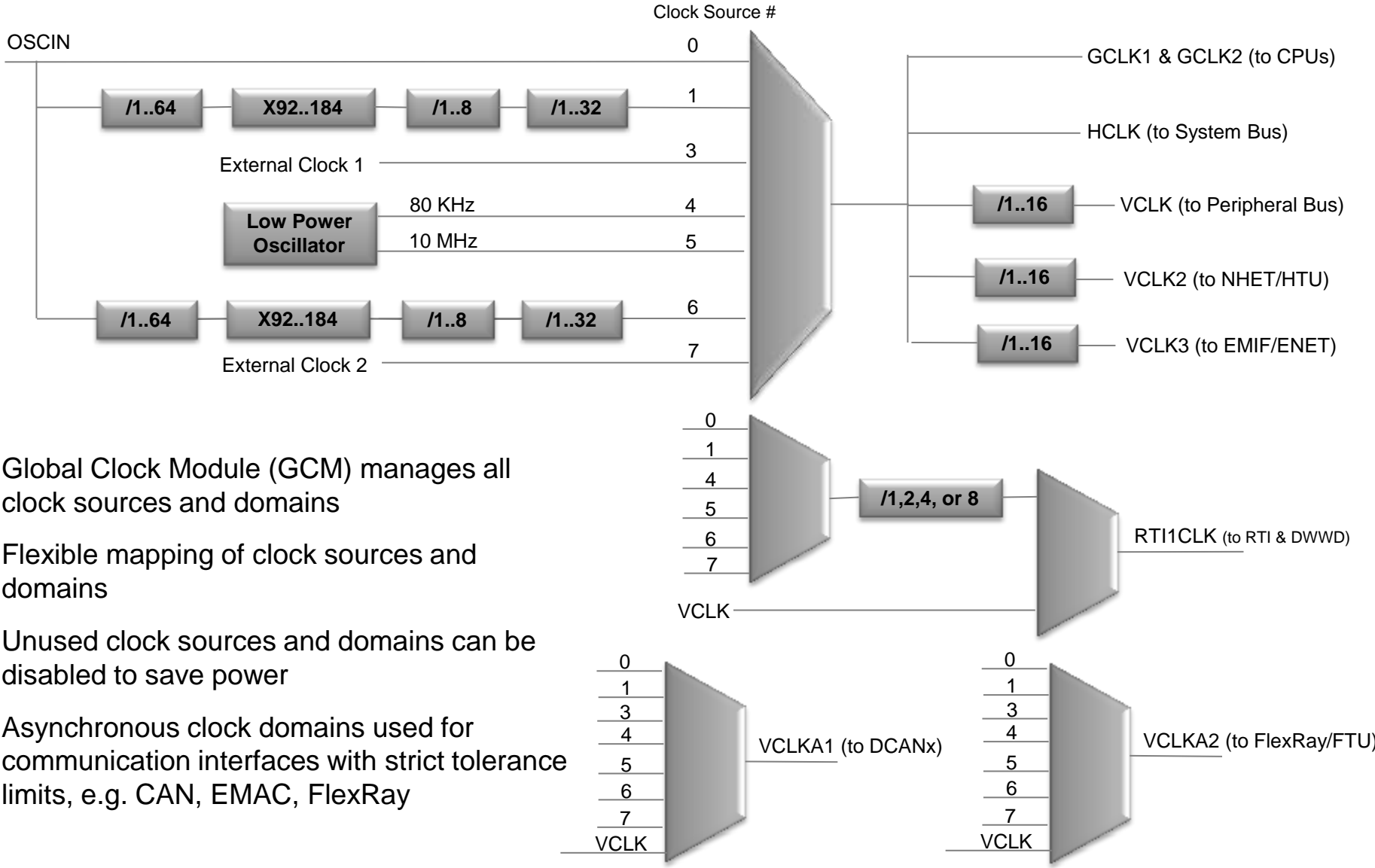
Memory Map

TMS570LS31x and RM48x

- Flash starts at 0x00000000 and CPU RAM starts at 0x08000000 by default
- Flash is mirrored outside of CPU TCM space for ECC diagnostics
- Code execution is only allowed from Flash, RAM and external asynch memory by default
- System and peripheral control registers' space is defined to be "strongly-ordered" by default
- CPU accesses to "Reserved" areas results in an Abort exception
- Behavior on accesses to reserved locations within defined frames specified in datasheet

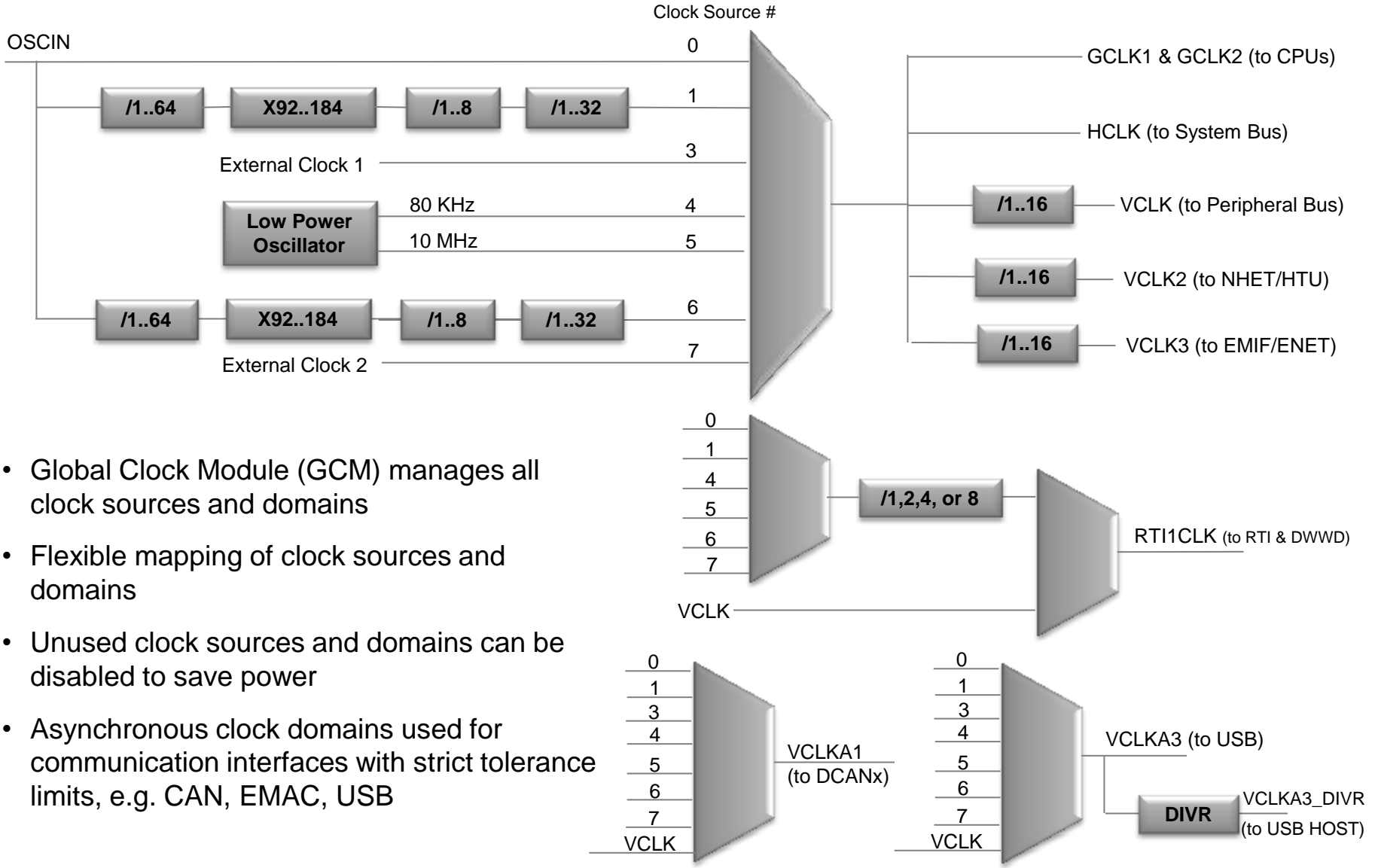


Clock Sources and Domains: TMS570LS31x/21x



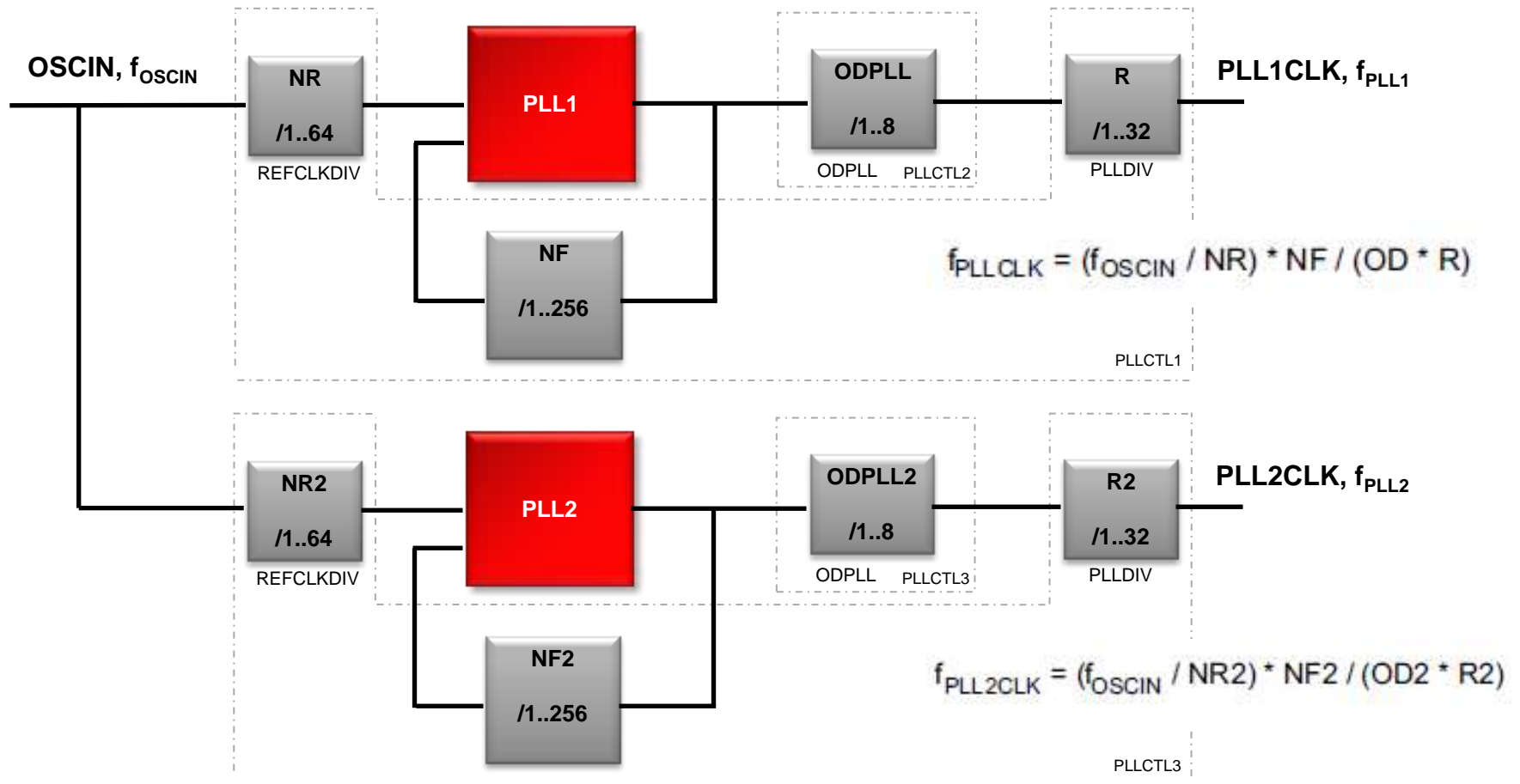
- Global Clock Module (GCM) manages all clock sources and domains
- Flexible mapping of clock sources and domains
- Unused clock sources and domains can be disabled to save power
- Asynchronous clock domains used for communication interfaces with strict tolerance limits, e.g. CAN, EMAC, FlexRay

Clock Sources and Domains: RM48x/RM46x



- Global Clock Module (GCM) manages all clock sources and domains
- Flexible mapping of clock sources and domains
- Unused clock sources and domains can be disabled to save power
- Asynchronous clock domains used for communication interfaces with strict tolerance limits, e.g. CAN, EMAC, USB

Phase-Locked-Loops (PLL1 and PLL2)



- PLL1 is configured using PLLCTL1 and PLLCTL2 registers
- PLL2 is configured using PLLCTL3 register
- TRM describes procedure for configuring frequency modulation settings for PLL1
- Frequency modulation not available for PLL2

Reset Sources

- Power-on Reset
 - Asserted by external voltage supervisor, or by internal voltage monitor
- Oscillator fail
 - Asserted by internal clock monitor when enabled by software
- CPU Reset
 - Asserted by CPU self-test controller after LBIST operation completes
- Software Reset
 - Asserted by software writing to the exception control register
- External Reset
 - Asserted by external circuitry driving the warm reset (nRST) signal LOW
- Debug Reset
 - Asserted by ICEPICK JTAG module

Hercules™: Flash Tools

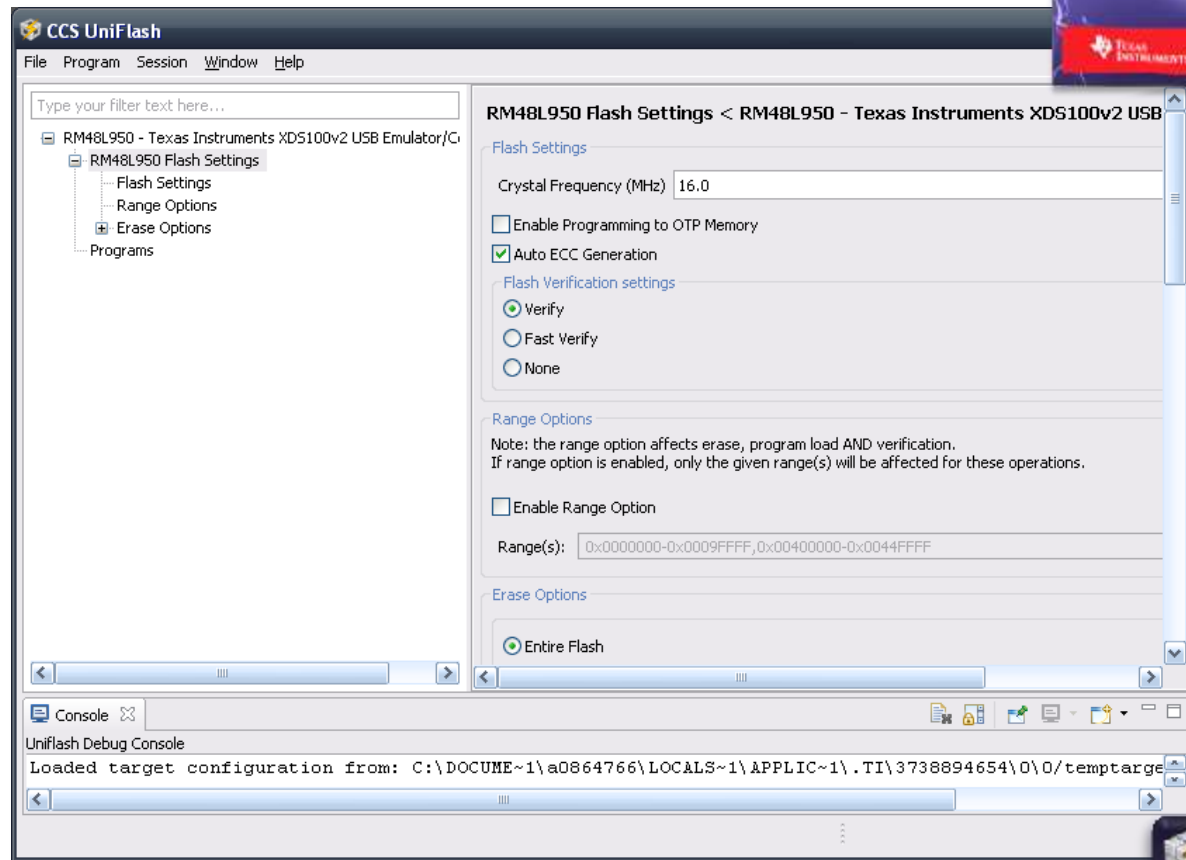
```
<return_value> nowECC [options] -i <input_file> [-o <output_file>]
```

- Generates ECC data for program flash
- Command-line executable
- Return value = 0 indicates no error during operation
 - Separate error codes to differentiate each type of error
- Input_file is only required parameter
 - Can be Extended Tektronix, Intel Hex, Motorola-S, COFF or ELF format
- Output_file specifies the name of the output file to be generated
 - If no name is specified, ECC is appended to input file specified

Options for Flash Programming

- On-board programming using Code Composer Studio v5.x/CCS UniFlash
 - Requires JTAG connection
 - Emulators Supported:
 - Blackhawk BHUSB560M
 - Spectrum Digital XDS200, XDS510USB, XDS560RUSB
 - Signum JTAGjet
 - Texas Instruments XDS100v2, XDS560
- On-board programming via customer boot-loader code
 - Must use Texas Instruments released API routines
 - Multiple communication interfaces can be used
 - Necessary to validate program and erase routines
- Off-board programming
 - Single-device or Concurrent programming
 - Supports high degree of automation

UniFlash Flash Programming Tool



- PC-based software tool
- Communicates with microcontroller via JTAG
- Can be used to program and erase flash memory
- Based on Eclipse – Supports Windows and Linux



Flash Application Programming Interface (API)

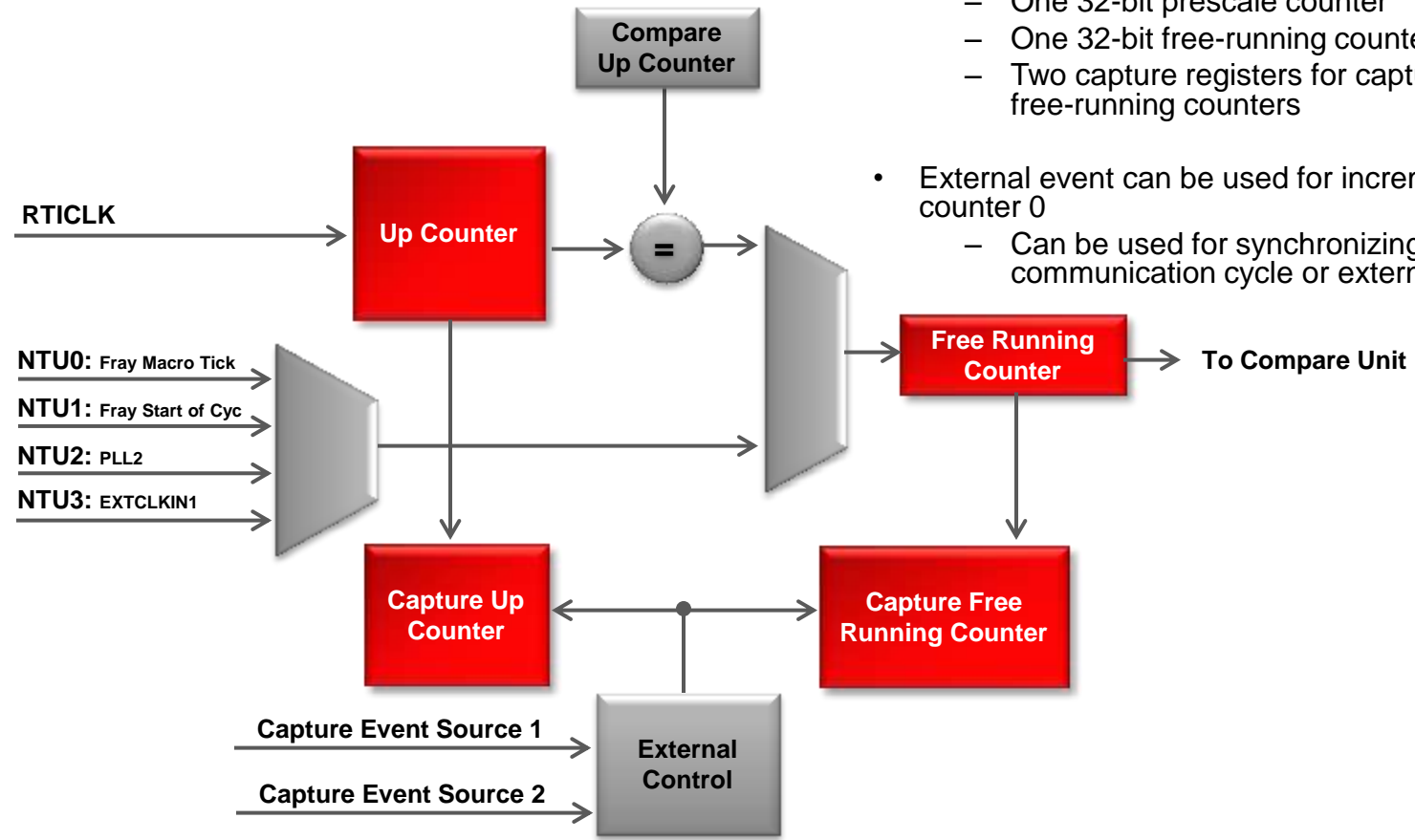
- Distributed only as an object library file
- Supports flash operations out of on-chip RAM
- Supports operations at max specified device clock frequency
- Library routines for
 - Blank check
 - Compaction
 - Erase
 - Program zeros
 - Program data
 - Calculate checksum
 - Verify
- Routines also manage ECC

Real-Time Interrupt Module (RTI)

RTI: Counter Block Diagram

RTI Counter Features:

- Two independent counter blocks for generating different time bases
- Each block consists of
 - One 32-bit prescale counter
 - One 32-bit free-running counter
 - Two capture registers for capturing the prescale and free-running counters
- External event can be used for incrementing free-running counter 0
 - Can be used for synchronizing with FlexRay bus communication cycle or external system clock input

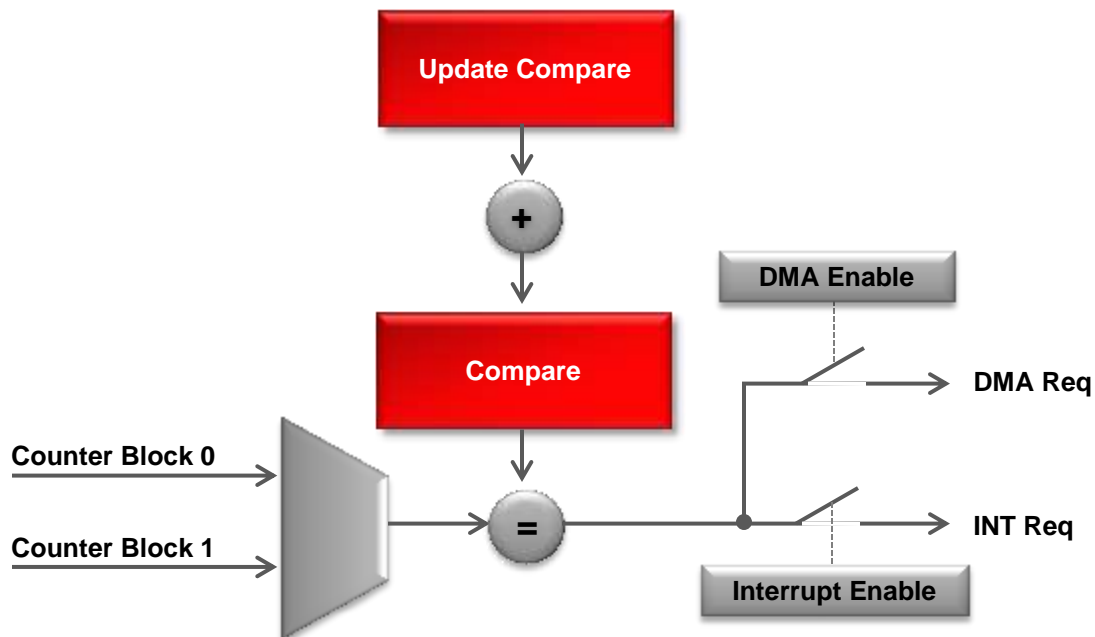


NOTE: Counter Block 1 does not contain external NTU inputs

RTI: Compare Block Diagram

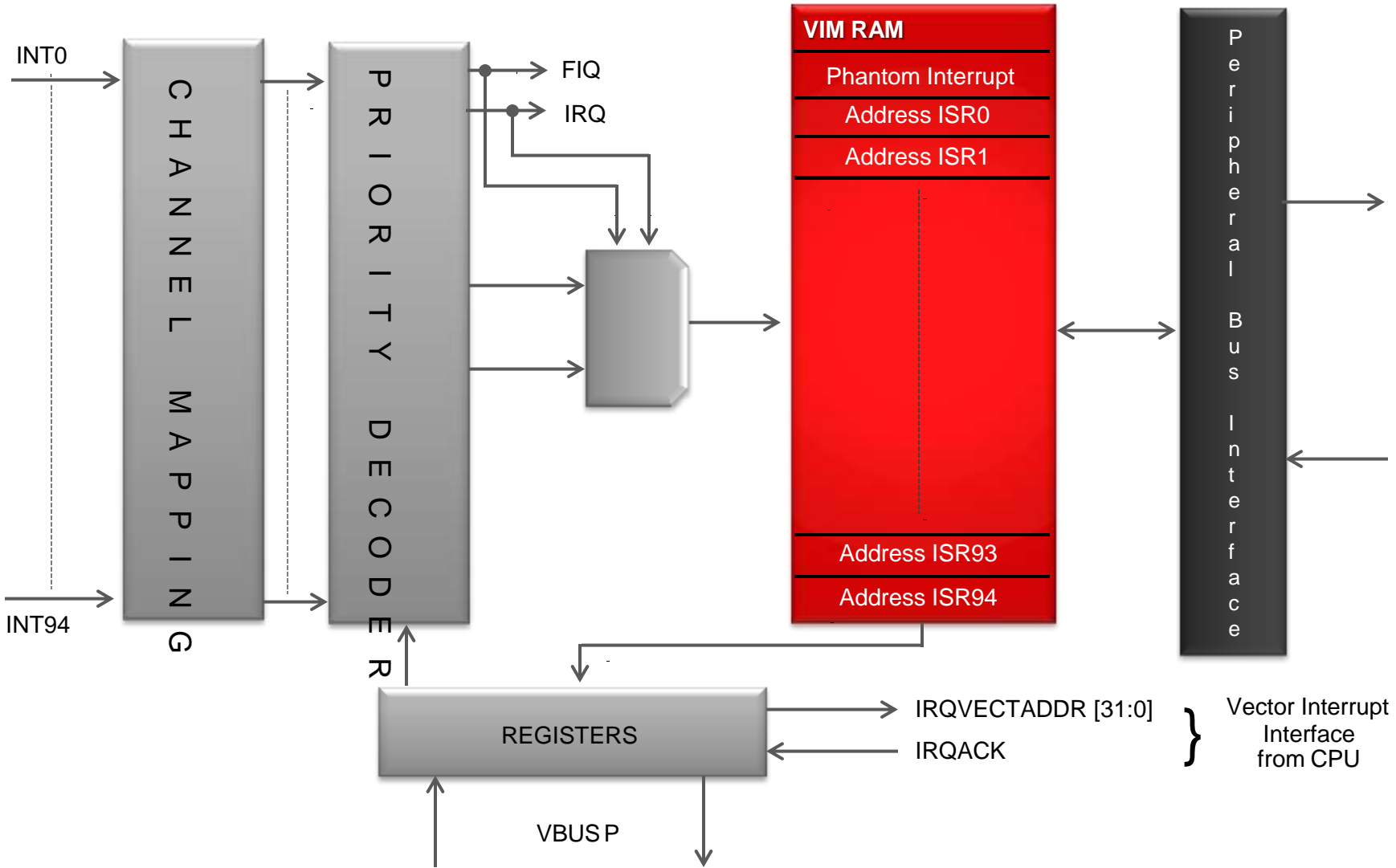
RTI Compare Features:

- Four compare interrupts and DMA requests
 - Each can use either of the two available free-running counters
 - Automatic update of compare values to minimize CPU intervention
 - Option to generate DMA request as well as the compare interrupt
- Two counter-overflow interrupts
 - Generated when a free-running counter overflows and goes to zero

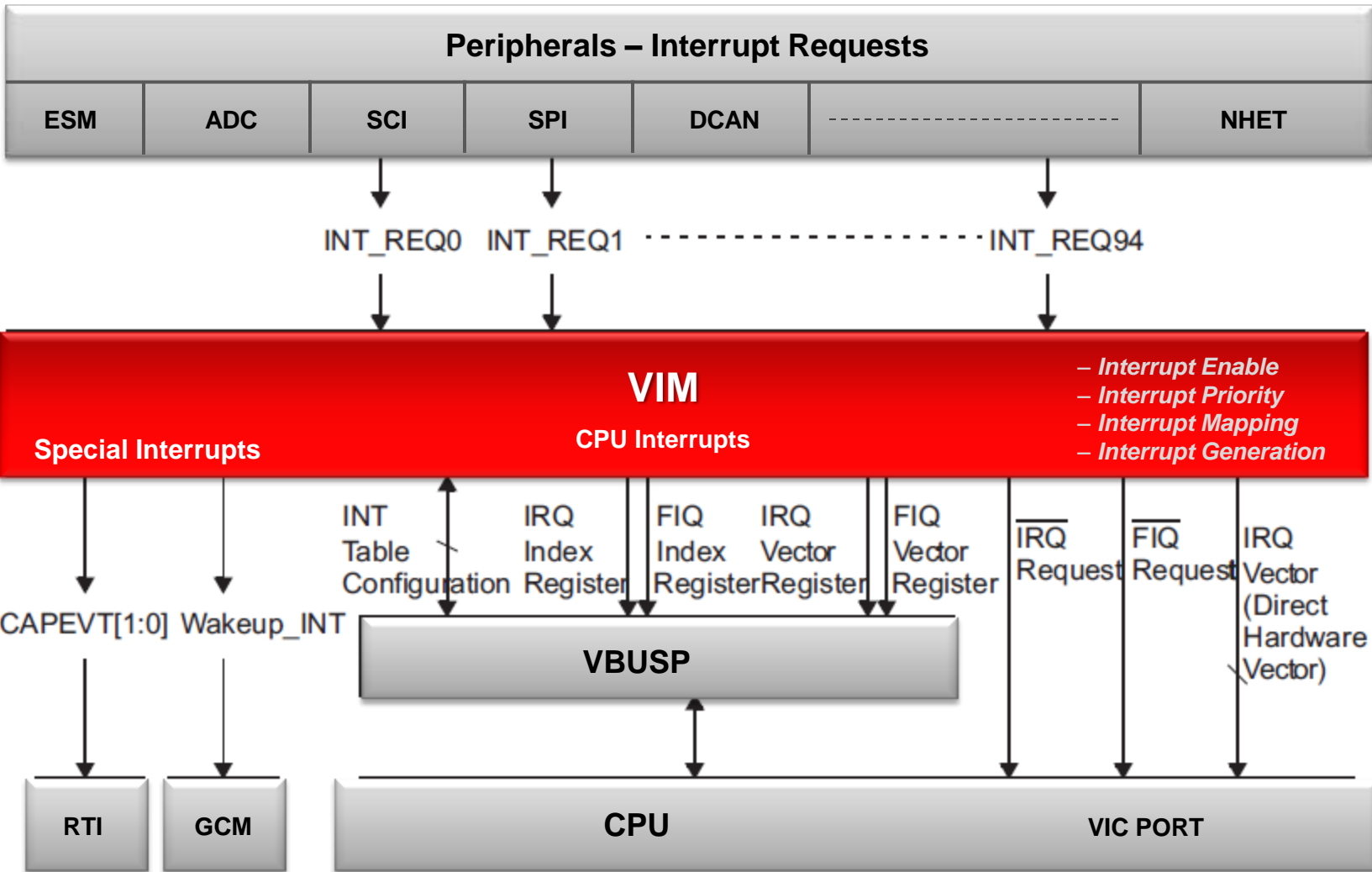


Vectored Interrupt Manager (VIM)

VIM: Block Diagram



VIM: Connection Block Diagram



VIM: Main Features

- **VIM Hardware**
 - Dedicated Vector Interrupt interface to ARM CPU
 - Hardware relocation of the IRQ vector address
 - Hardware assistance for prioritizing and controlling interrupt sources
- **VIM Functions**
 - 96 interrupt requests
 - Map interrupt request to interrupt channel via programming.
 - Provides programmable priority through interrupt request mapping
 - Prioritizes the interrupt channels to the CPU
 - Provides the CPU with the address of the interrupt service routine (ISR)
- **VIM Modes**
 - Legacy ARM7 Mode (FIQ/IRQ)
 - Vectored interrupt (FIQ/IRQ)
 - Hardware vectored interrupt (IRQ only)

Direct Memory Access (DMA)

DMA: Main Features

- 32 channels with individual enable
- 64 DMA requests
 - Software and hardware DMA requests (event synchronization)
- Supports 8, 16, 32 or 64 bit transactions
- Multiple addressing modes for source/destination
 - fixed, incrementing, indexed
- Auto Initiation
- Channel chaining capability
- 1 FIFO (First In First Out)
- One AHB master port (64 bit wide) to interface with the bus matrix
- One slave port to interface with VBUS for register interface
- Memory Protection for the address range DMA can access

Note: Not all Hercules MCUs are available with a DMA

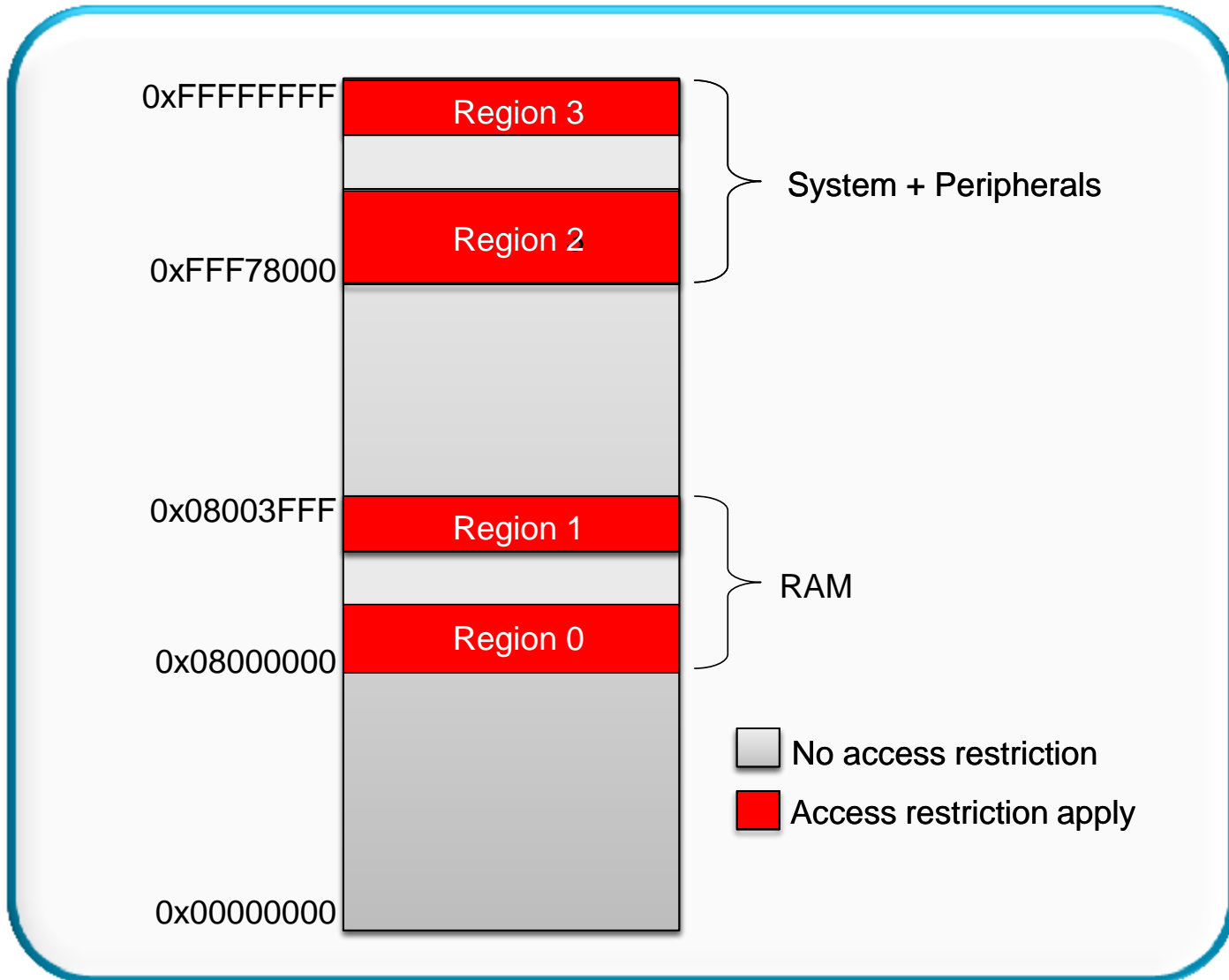
DMA: How to Start a Transfer?

- Software requests
 - By setting bit x in SWCHENAS [31:0] register transfer (channel x) will be triggered.
- Hardware requests
 - An active DMA request signal will trigger a DMA transaction.
 - Up to 64 DMAREQ lines can be handled.
 - Since DMA controller is clocked by HCLK, the duration of all DMA requests signals must be at least HCLK long.
- Triggered by other control packet
 - When a control packet finishes the programmed number of transfers it can trigger another channel to initiate its transfers.

DMA: Channel Interrupts

- Each channel can be configured to generate interrupts on several transfer conditions:
 - FTC (Frame Transfer Complete) interrupt
 - LFS (Last Frame Transfer Started) interrupt
 - HBC (First Half of Block Complete) interrupt
 - BTC (Block Transfer Complete) interrupt
 - BER (Bus Error) interrupt

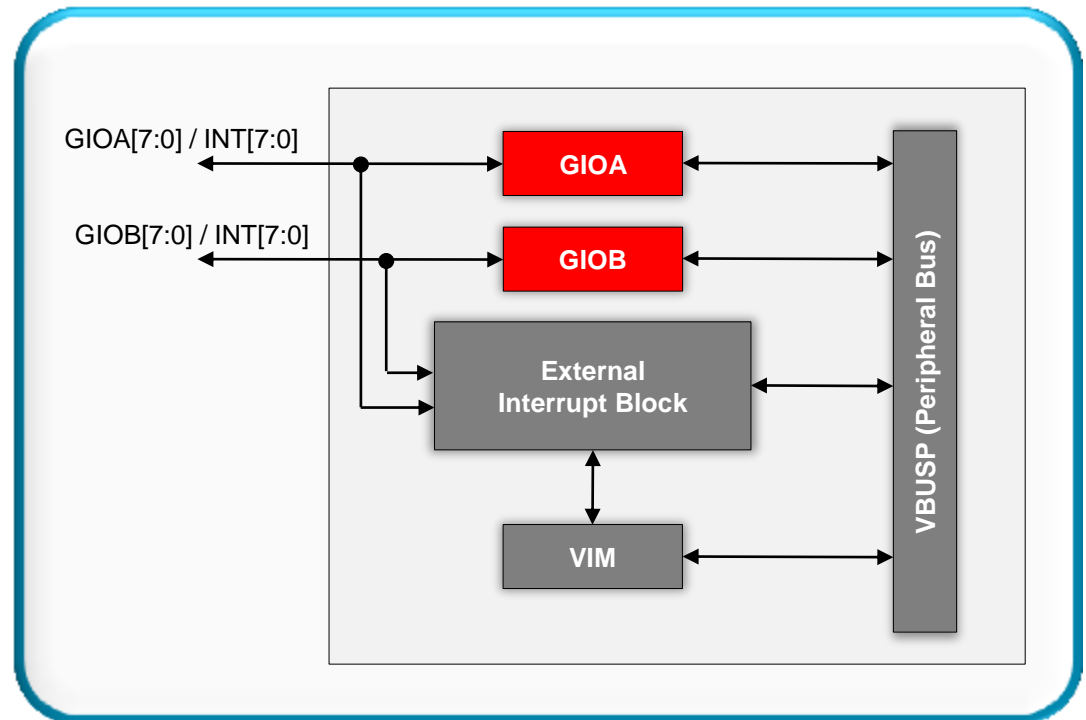
Memory Protection



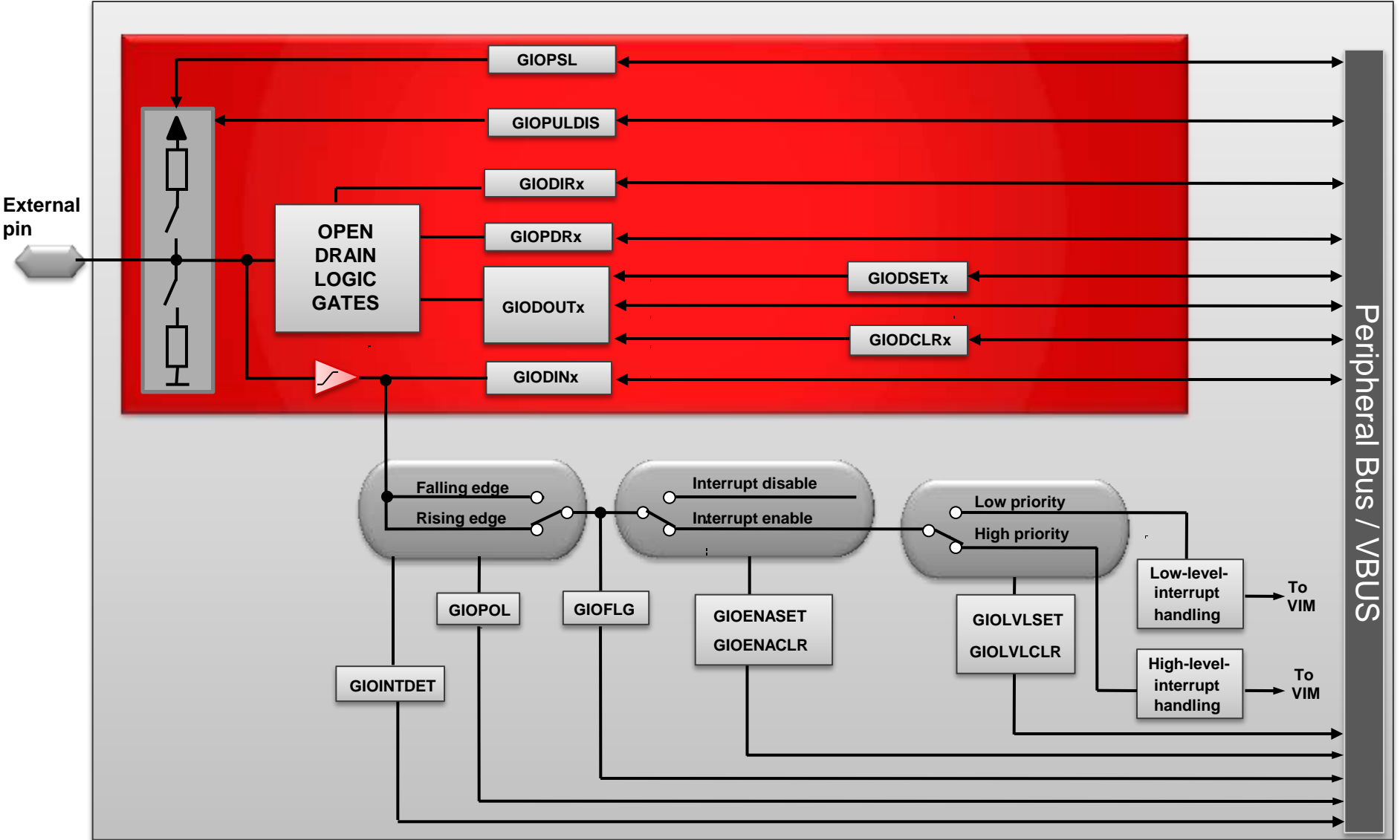
General-Purpose I/O (GPIO)

GIO Main Features

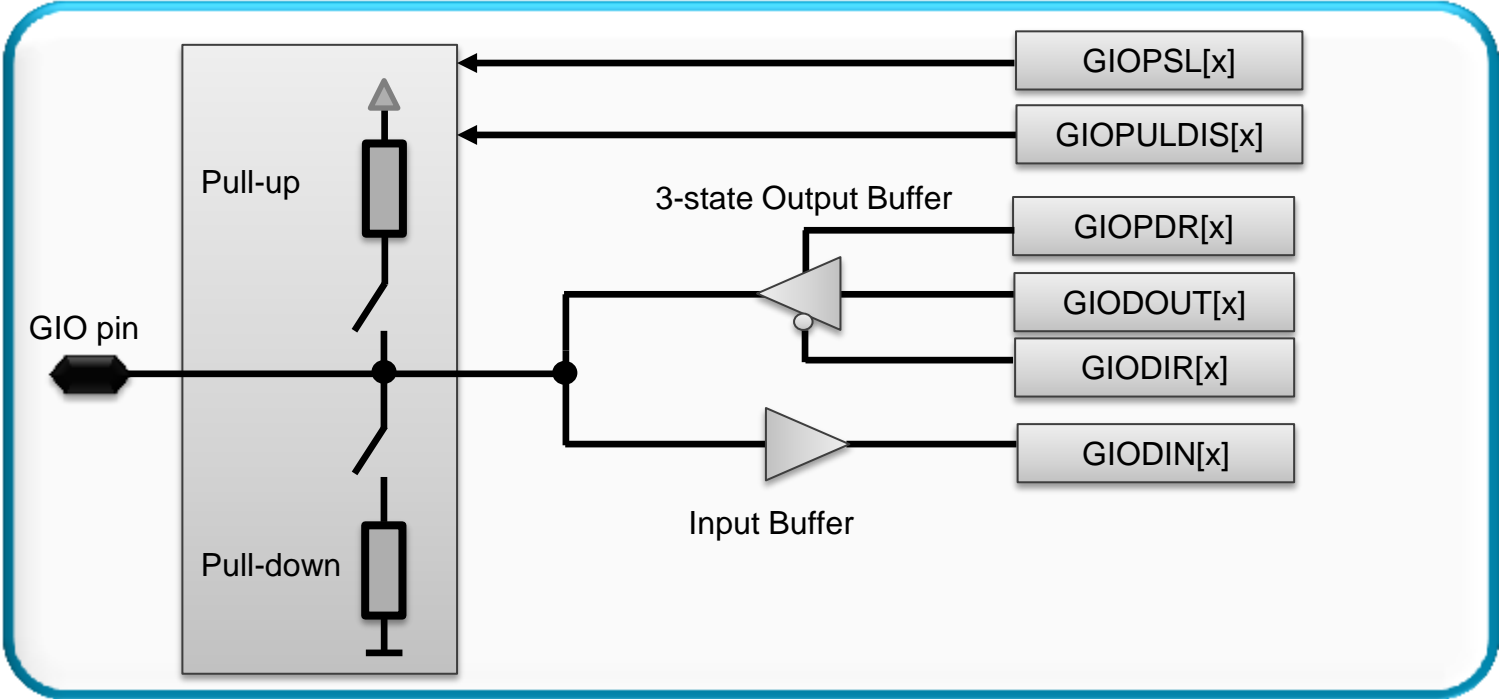
- Two ports (GIOA/B), each with 8 bidirectional and bit-programmable I/O pins
- External interrupt capability
 - Programmable interrupt detection on single or both edges
 - Programmable edge detection polarity
 - Programmable interrupt priority
- Possible pin configurations:
 - Data direction
 - Data input/output
 - Data set/clear
 - Open drain
 - Pull-up/Pull-down



GIO Block Diagram



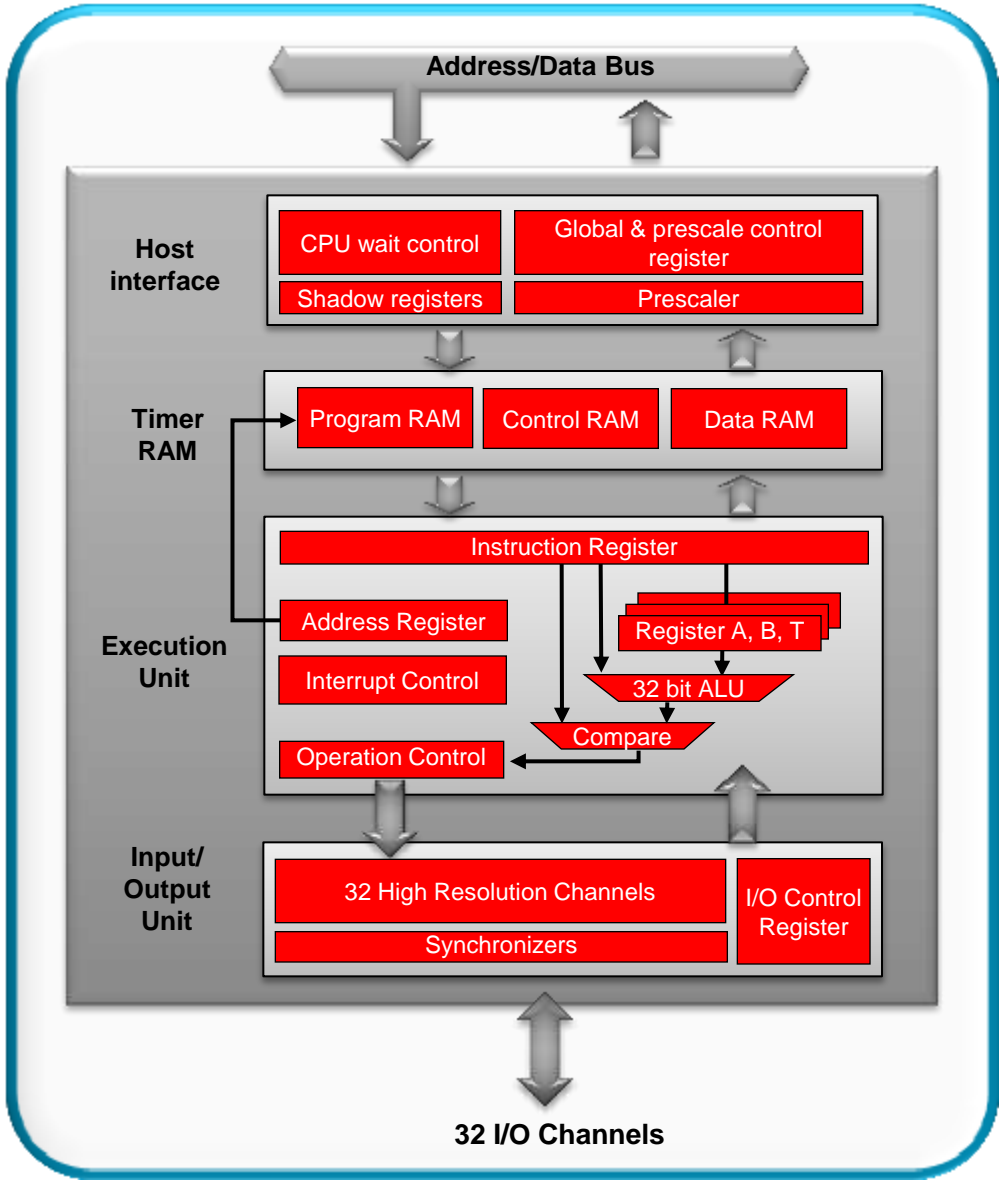
GPIO Cell Configuration



Register	Description
GIOPSL	Selects the pull type at pin (pull-up / pull-down)
GIOPULDIS	Disables the pull control capability at pin
GIOPDR	Controls the open drain configuration of the pin
GIODOUT	Controls what information is sent to external pin when configured as output
GIODIN	Receives information from external pin
GIODIR	Controls the direction of the pin (input / output)

High-End Timer (N2HET)

High End Timer (NHET)



- User-programmable Timing Co-Processor
- Provides high level and complex timing functions with low CPU overhead
- 128 word instruction RAM with Parity protection
- Dedicated DMA functionality (HTU) to transfer data from NHET to Data Ram w/o CPU
- Conditional program execution based on pin conditions and compares
- 32 input/output (I/O) channels (pins) for complex or classical timing functions such as capture, compare, PWM, GPIO
- Suppression filters eliminate undesired input frequencies
- Multiple 25-bit virtual counters for timers, event counters, and angle counters
- High Resolution I/Os and coarse resolutions implemented by sub loops for multiple resolution capability

NHET: Application Examples

Pulse Width Modulation

- Single / multi channel PWMs
- PWM with synchronous / asynchronous duty cycle update
- PWM with synchronous period update
- Phase shift PWM's using RADM64 instruction

Other Features

- Frequency Modulated Output
- Pulse width count (using PWCNT)
- Time stamp (using WCAP)
- Event counter (using ECNT)
- Pulse accumulator example (using ECNT)
- Multi-resolution scheme

Frequency and Pulse Measurements

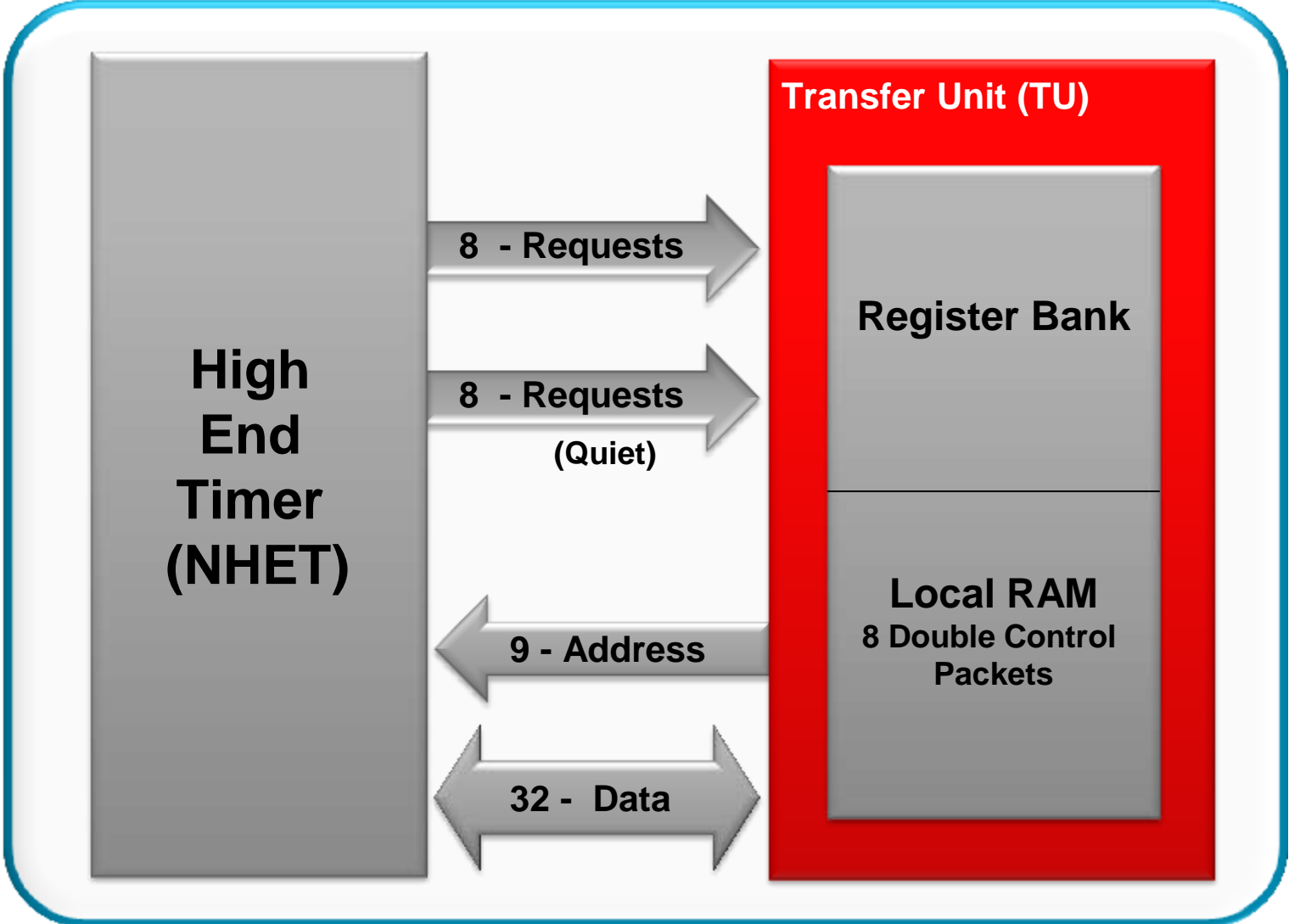
- Pulse width and period measurement (using PCNT)
- Period measurement using PCNT in HR mode, HRshare feature and 64 bit read access with “auto read/clear” bit set

NHET: Command Line Assembler

- Invoking the **NHET assembler (hetp.exe)**: **hetp** [options] input file
- Options:
 - -c32 produces an output file containing assembler directives for the TMS570 CodeGen Tools
 - -hc32 produces a **C file** and a **header file**. (used together with the -nx option)
 - -nx specifies the x-th HET module on the device (used together with -hc32 option)
 - -l (lowercase L) produces a listing file with the same name as the input file with a .lst extension.
 - -x produces a cross-reference table and appends it to the end of listing file.
- Example: **hetp -hc32 -n0 pwm.het**
- Input: pwm.het contains the assembly source of the HET program
- Output:
 - pwm.c provides a C array, which contains the HET program opcode
 - pwm.h provides a C structure, which allows a simple access to the NHET fields from other C code

High-End Timer Transfer Unit (HTU)

HTU: Block Diagram



HTU: Main Features

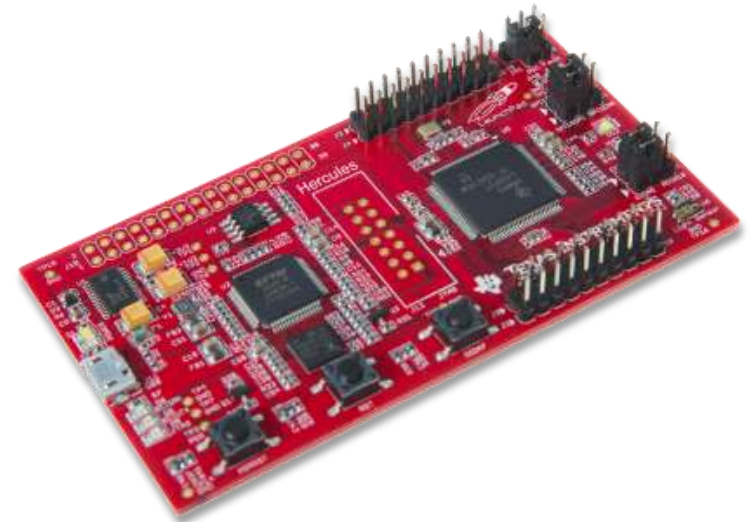
- CPU and DMA independent
- Master Port to access directly system memory
- HTU master accesses protected by dedicated Memory protection Unit
- One Slave port to interface with VBUS for register interface
- Maximum of 8 double control packets supporting dual buffer configuration
- Control packet information is stored in RAM protected by parity
- Event synchronization (HET transfer requests)
- Support 32 or 64 bits transaction
- Addressing modes for HET address (8 byte or 16 byte) and system memory address (fixed, 32 bit or 64bit)
- Each type of interrupt can be routed to either two different host CPUs
- One shot, circular and auto switch buffer transfer modes
- Request lost detection

Exercise: PWM Generation using the NHET

Overview



- **In this exercise we will:**
 - Create a new HALCoGen Project
 - Configure HALCoGen to generate
 - A basic PWM with a period of 1 second and a duty cycle of 75%
 - Use the PWM to toggle the NHET[08] LED on the board
 - Generate and Import code into Code Composer Studio
 - Build and Deploy our code to the microcontroller
- **Required Hardware:**
 - Windows Based PC (WinXP, Vista, 7)
 - **TMS570 LaunchPad** or **RM4 LaunchPad**
- **Required Software:**
 - [HALCoGen](#)
 - [Code Composer Studio](#)



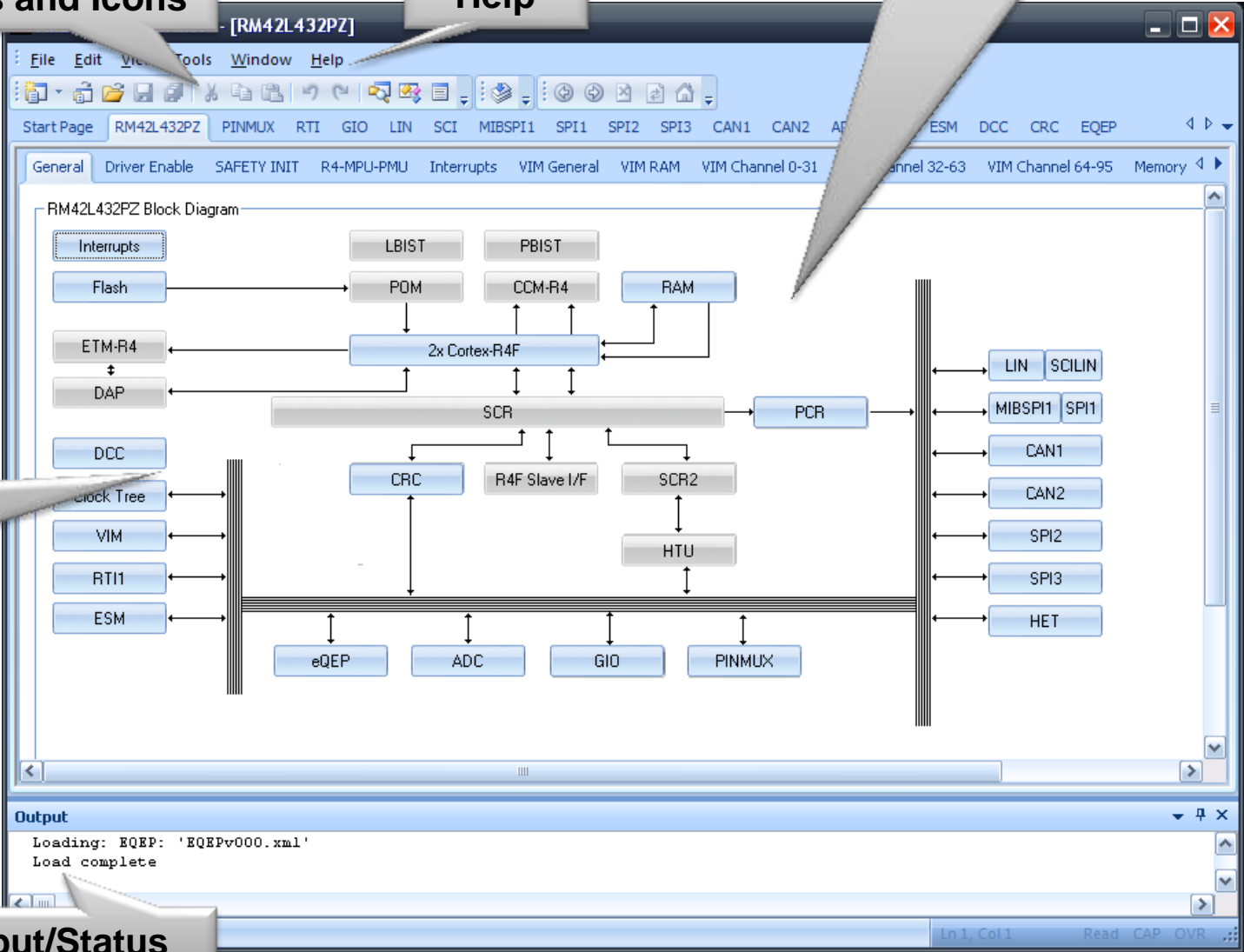
HALCoGen GUI Overview

Module Selection/Configuration

Menus and Icons

Help

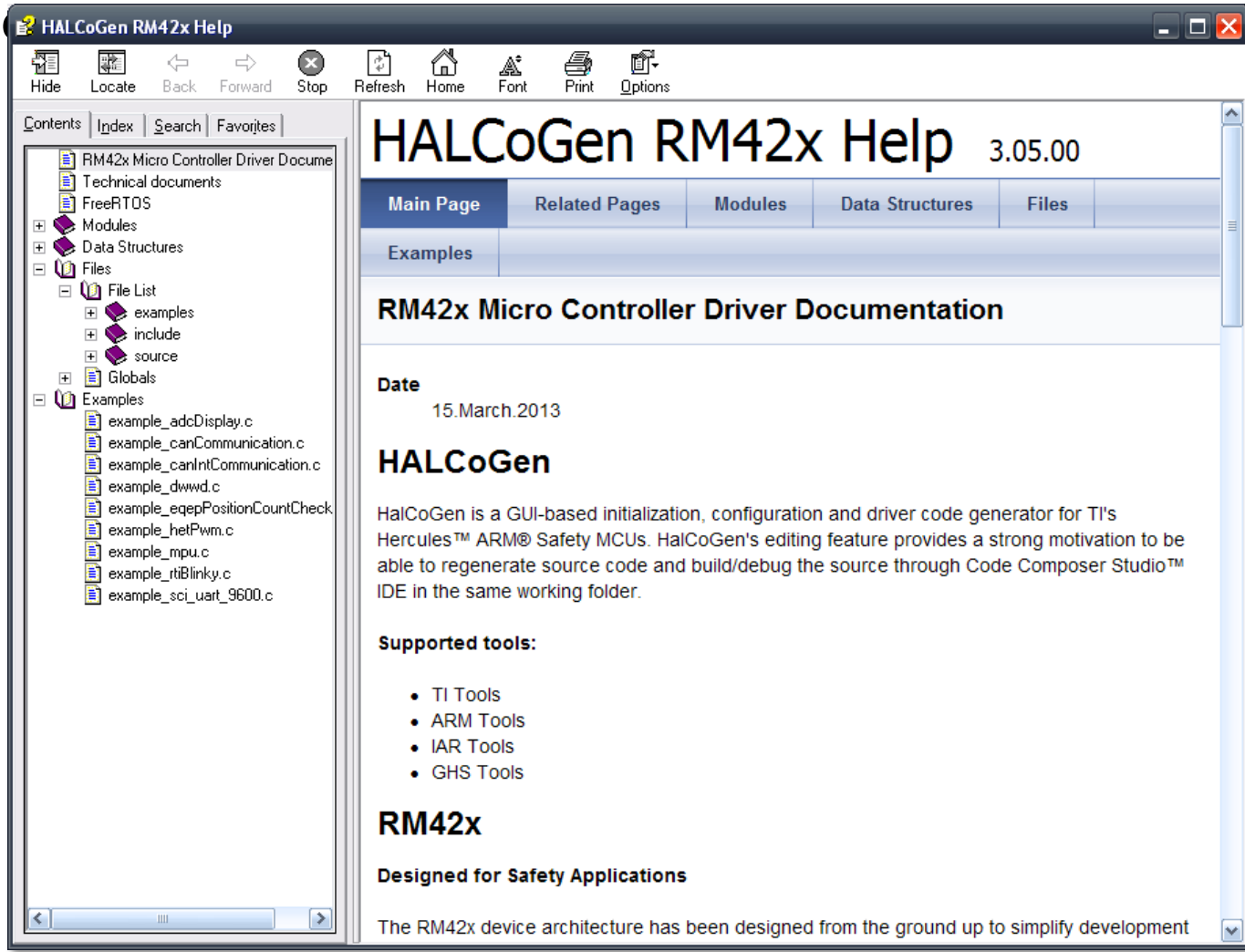
Device Block Diagram



Output/Status

HALCoGen Help

- HALCoGen's embedded help window provides full documentation of each communication drivers, implemented functions, file dependencies and examples



HALCoGen File Dependencies and Function Listing

The screenshot shows the HALCoGen Help application window. The 'Files' tab is active, displaying the 'C:/HALCoGen/gio.c File Reference'. The left pane shows a 'File List' with 'C:/HALCoGen/gio.c' selected. The main content area shows the file's dependencies, including 'gio.h', and a list of functions: `gioInit`, `gioSetDirection`, and `gioSetBit`.

File Explorer

File Information Control

File Dependency Diagram

Function Listing

Set up a New HALCoGen Project

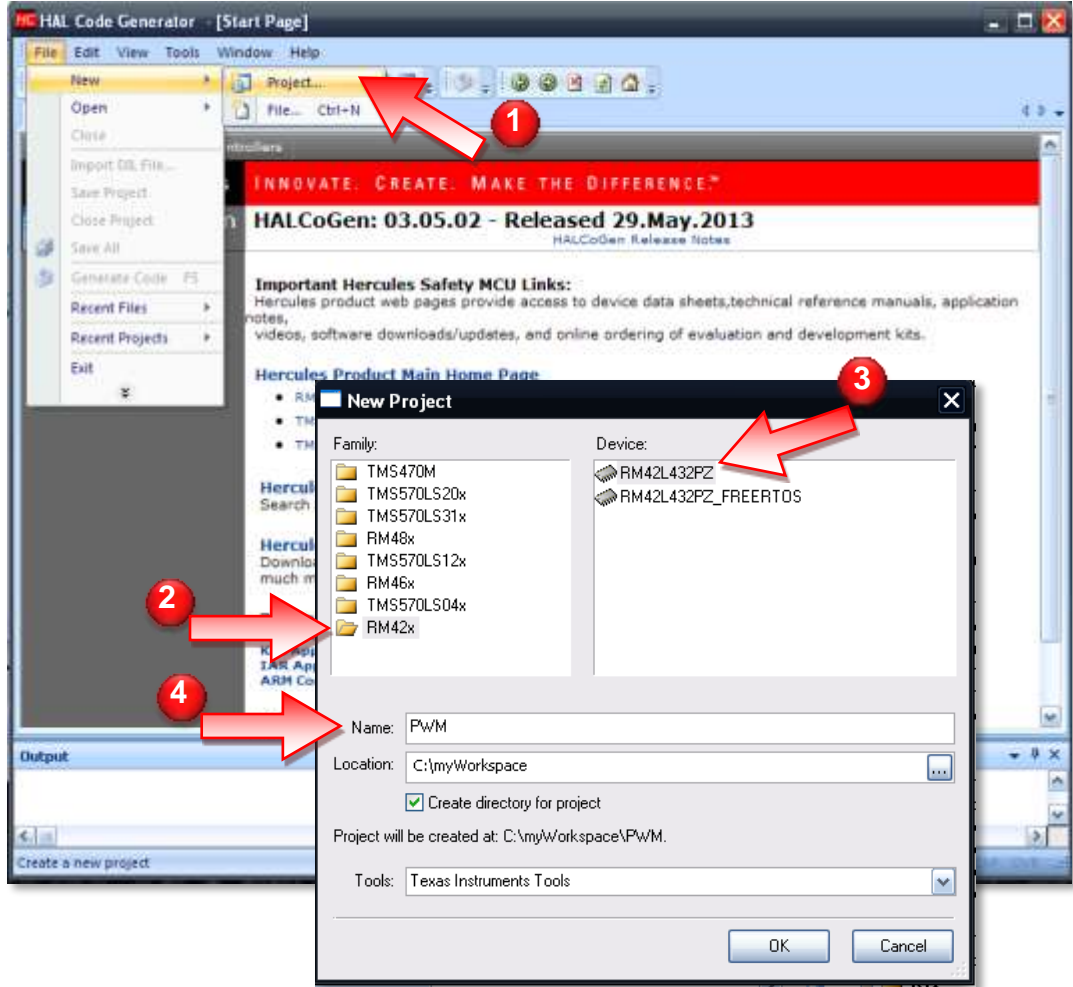
- To launch HALCoGen go to:
 → Programs → Texas Instruments → Hercules → HALCoGen

- Create a new project:
 - File → New → Project

- For the TMS570 Kit:**
 - Choose Family: TMS570LS04x
 - Device: TMS570LS0432PZ

- For the RM4 Kit:**
 - Choose Family: RM42x
 - Device: RM42L432PZ

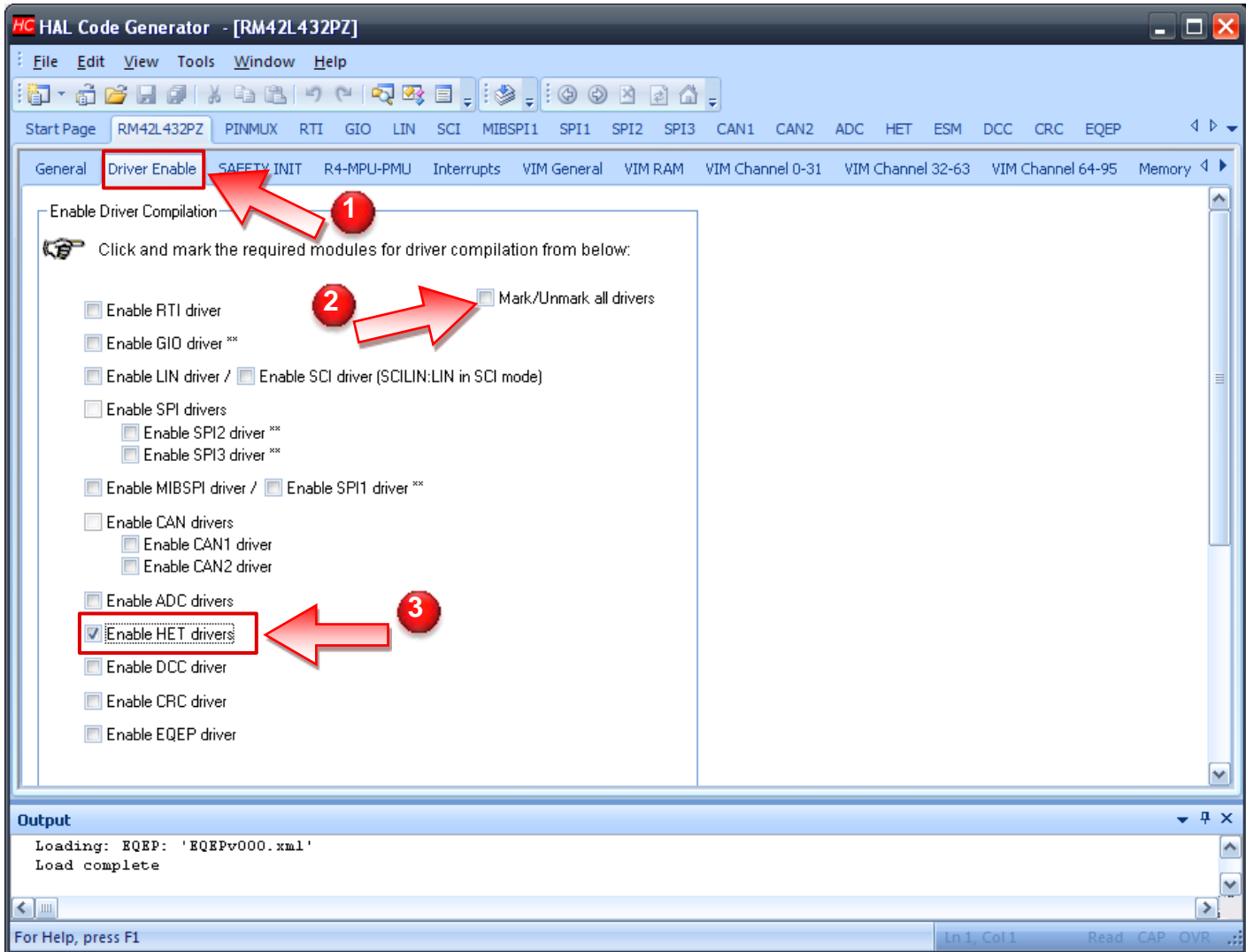
- Then define a name: **'PWM'**
- Location: "C:\myWorkspace"



Driver Enable



- In 'Driver Enable' tab enable the HET driver.



NHET PWM Configuration



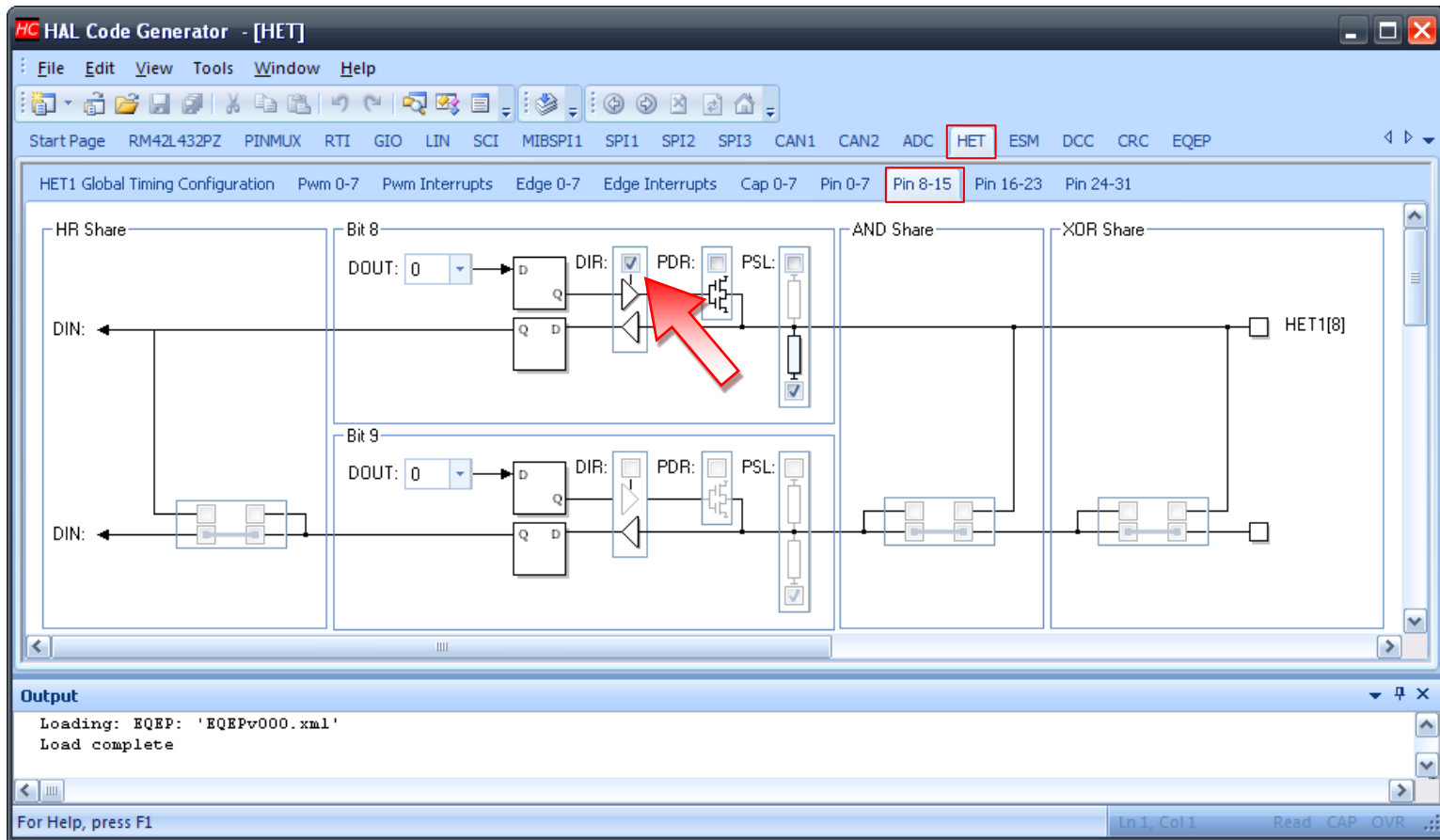
- In 'HET' → 'PWM 0-7' tab:
 - Configure PWM 0 to 75% Duty Cycle, with a Period 1000000.00uS on Pin 8

The screenshot shows the HAL Code Generator interface for configuring PWM 0. The 'Pwm 0-7' tab is selected, and the 'HET1 Global Timing Configuration' sub-tab is active. The PWM 0 configuration is shown with a duty cycle of 75% and a period of 1000000.000 us. The enable checkbox is checked, and the pin is set to 8. The PWM 1 configuration is also visible with a duty cycle of 50% and a period of 1000.000 us. The output window shows the loading of the EQEP configuration file.

Parameter	PWM 0	PWM 1
Duty [%]	75	50
Period [us]	1000000.000	1000.000
Pin	8	10

N2HET Output Configuration

- In the 'HET' → 'Pin 8-15' tab:
 - Enable the output on Pin 8



- Generate Code: File → Generate Code

Setting up Code Composer Studio

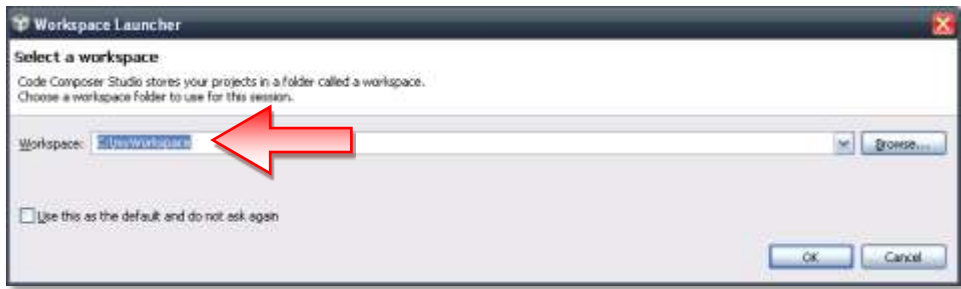


- Launch Code Composer Studio (CCS)

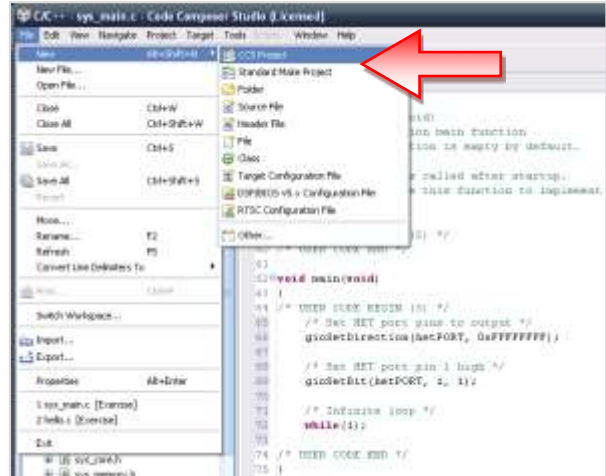


→ Programs → Texas Instruments → Code Composer Studio v5
→ Code Composer Studio v5

- When it launches, CCS will ask you to select a workspace, we will chose “C:\myWorkspace”



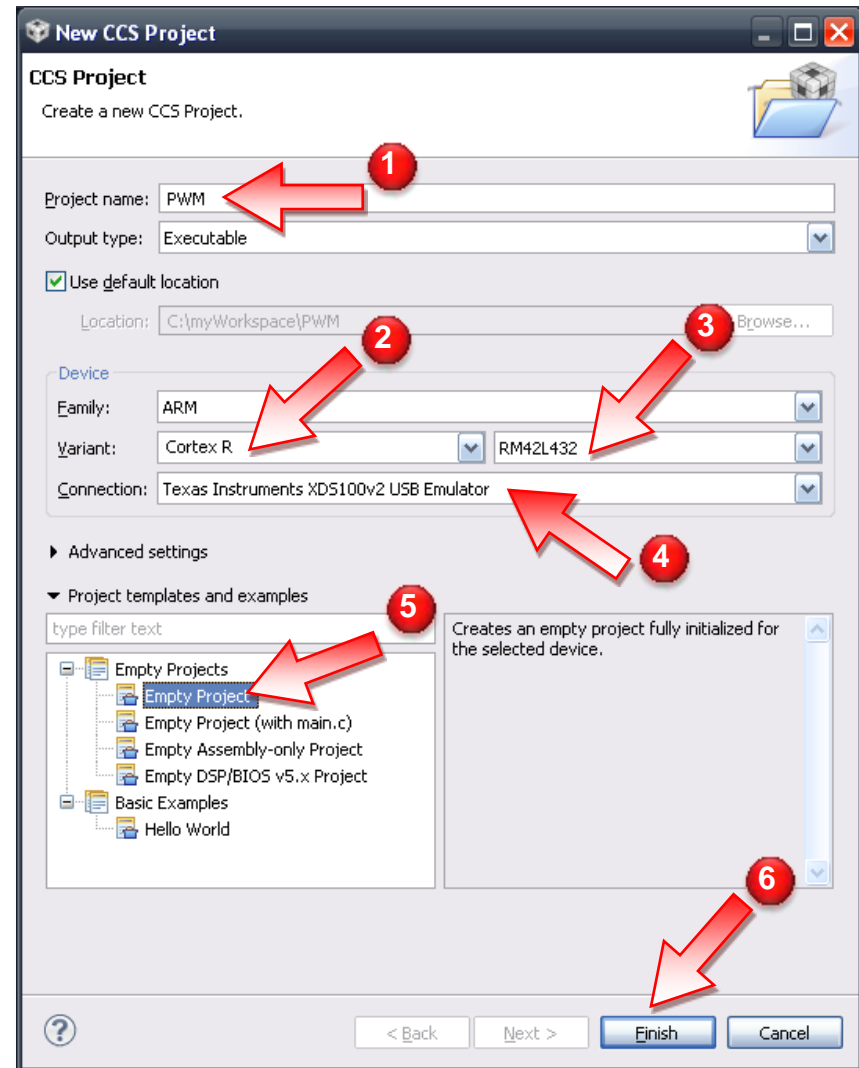
- Once CCS loads, go to File → New → CCS Project



Setting up our Project



- Our project name needs to match the name of our HALCoGen Project: **'PWM'**
- Make sure that your project **'Family'** is set to ARM
- Next, set the Variant to "Cortex R"
- **For the TMS570 kits:**
 - Choose: TMS570LS0432
- **For the RM4 kits:**
 - Choose : RM42L432
- Then set the 'Connection' to the Texas Instruments XDS100v2
- Then select 'Empty Project'
- Then click 'Finish'



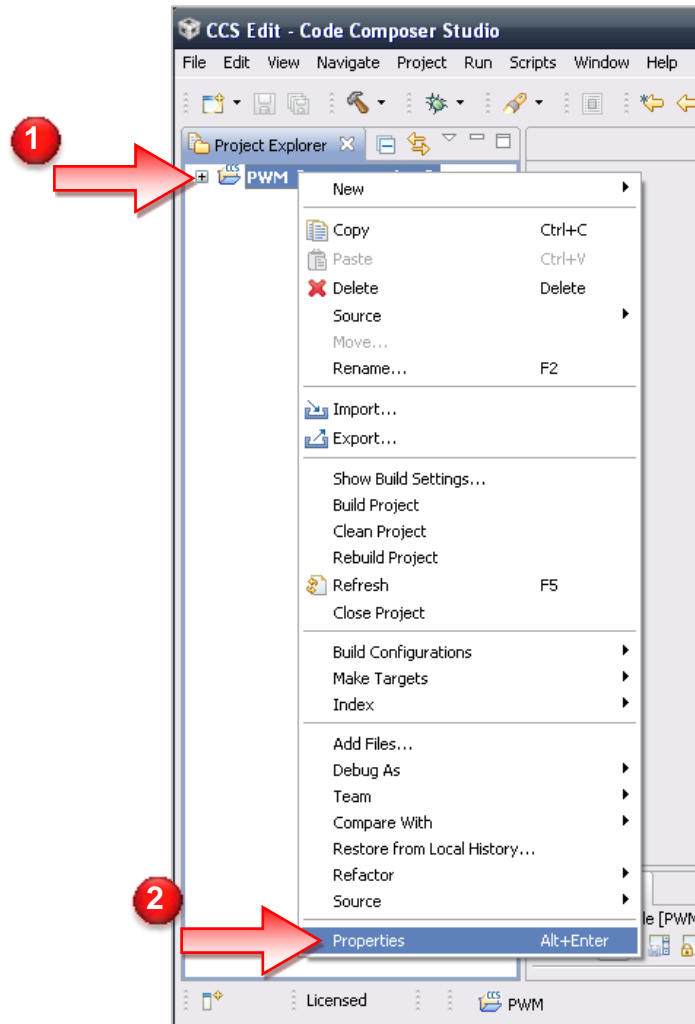
Setting up our Project



- Next we need to add our 'include' directory to the project from the CCS "Project Explorer"

- Right click on the 'PWM' project in the Project Explorer

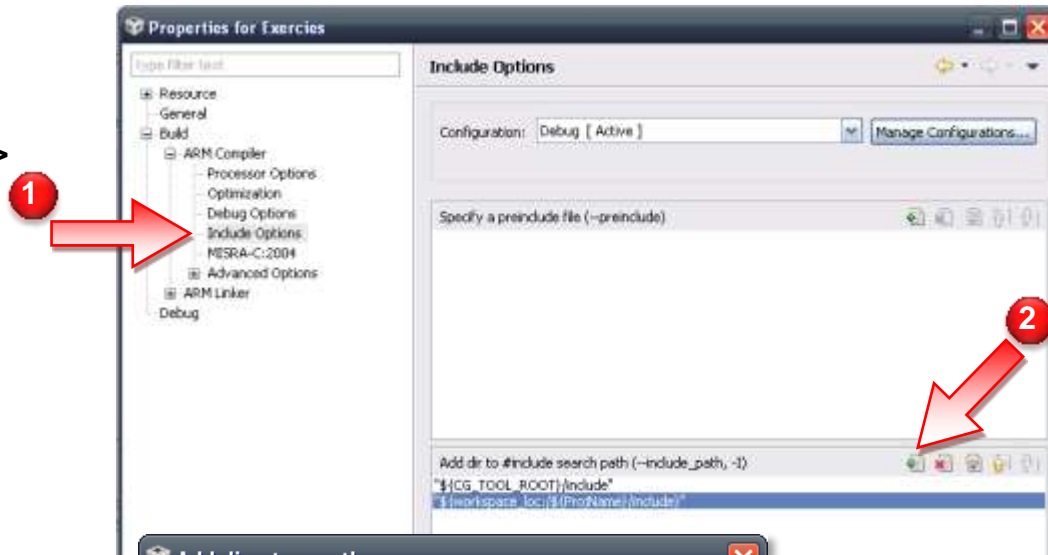
- Then choose 'Properties'




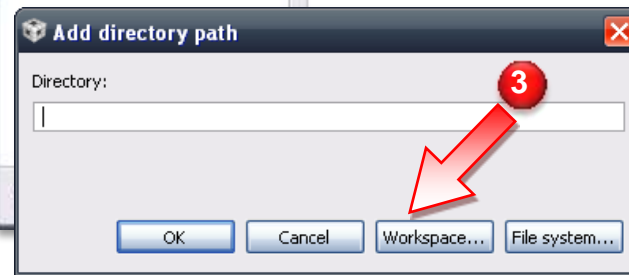
Setting up our Project



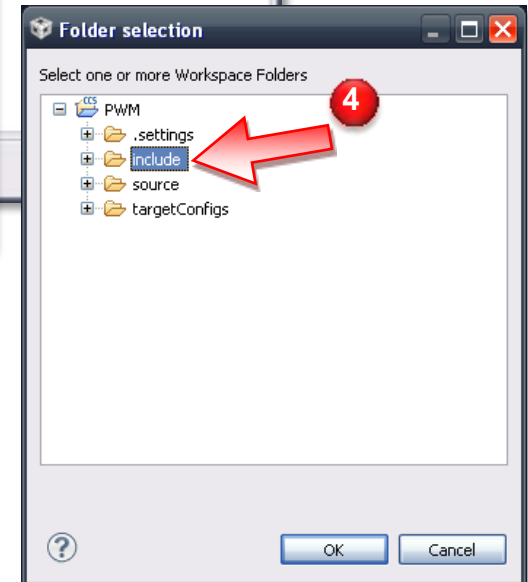
- Then in the 'Properties' window expand the 'Build -> ARM Compiler' category and select 'Include Options'



- Then select the  button to add the directory with our '.h' header files



- In the 'Add directory path' window, click the 'Workspace...' button

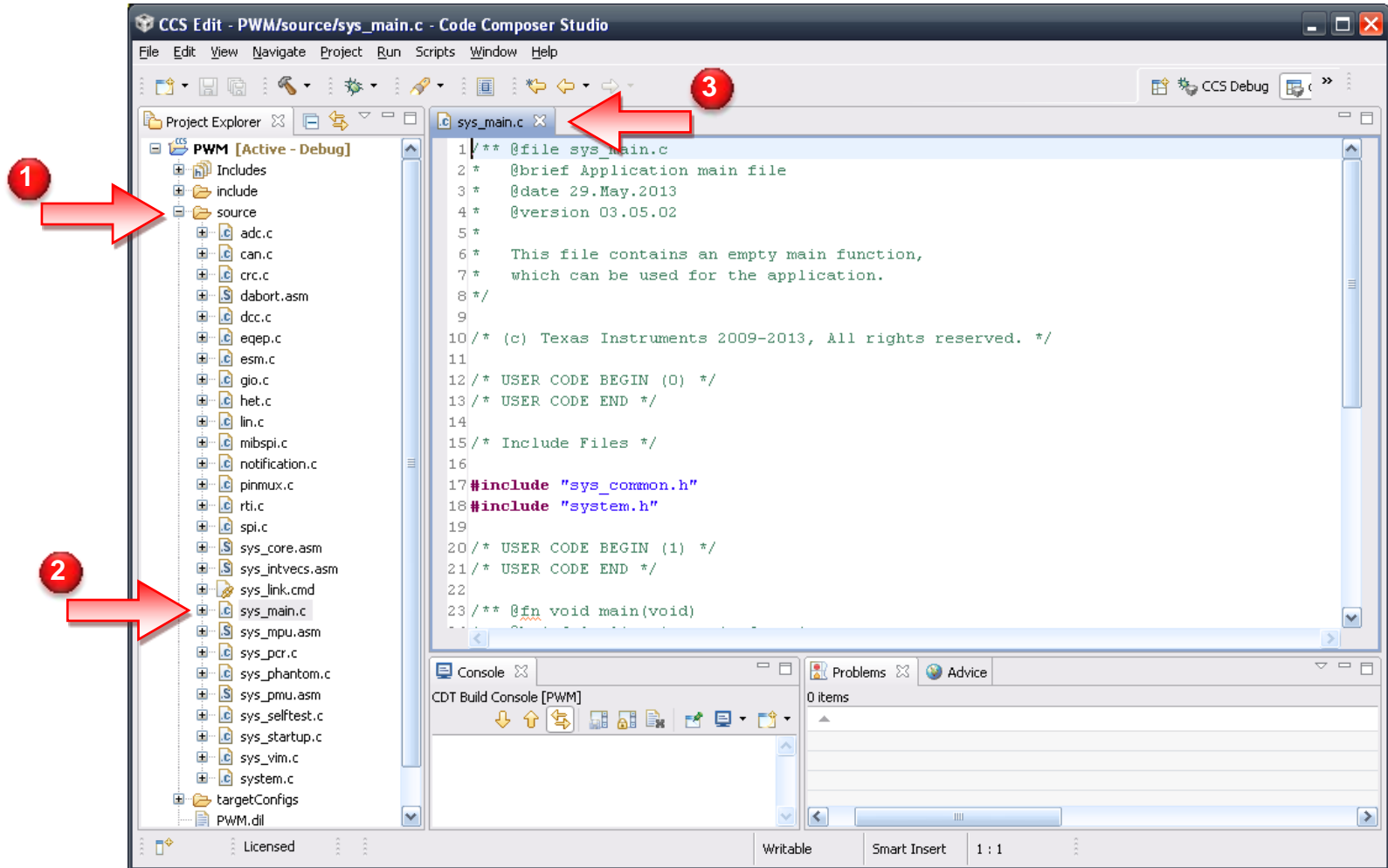


- Finally, select the 'include' folder that HALCoGen created for us that contains all our '.h' header files

Enter Code into the CCS Project



- Expand the project and open the “sys_main.c” file from the ‘source’ folder in the CCS “Project Explorer”





Code Composer Studio

- In the Code Composer Project and enter the following code:
 - Inside User Code 1, insert the code below.

```
/* USER CODE BEGIN (1) */  
#include "het.h"  
/* USER CODE END */
```

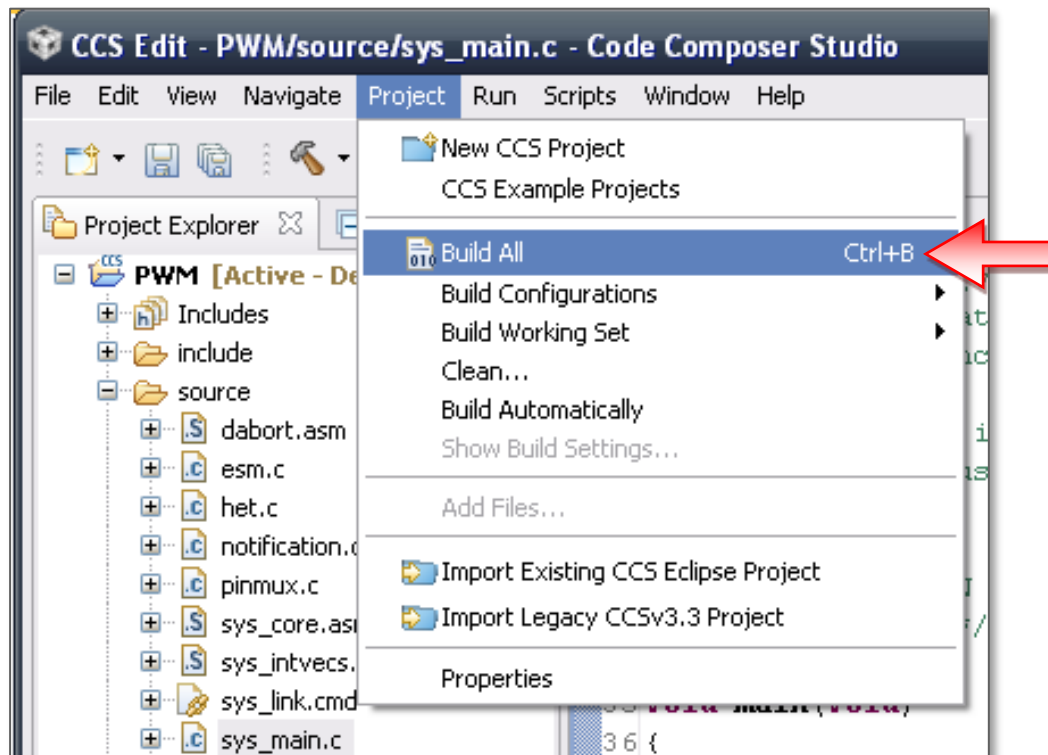
- Then in User Code 3, insert the code below.

```
/* USER CODE BEGIN (3) */  
hetInit();  
while(1);  
/* USER CODE END */
```




Compiling the Project

- The code is now complete and we are ready to build our project.
 - Go to Project → Build All

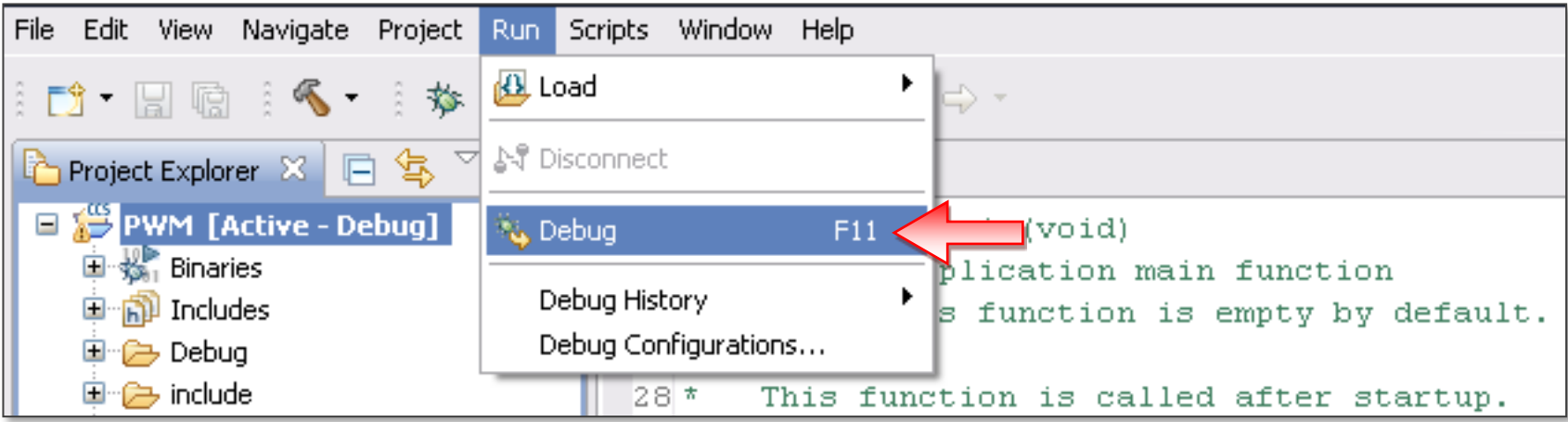


NOTE: It may take a 3 to 5 minutes to compile the RTS (Run Time Support Library) the first time a project is built.



Programming the Flash

- We are now ready to program the flash.
 - Go to Run → Debug
 - A new window should appear as it programs the flash memory.
 - This may take a few moments.



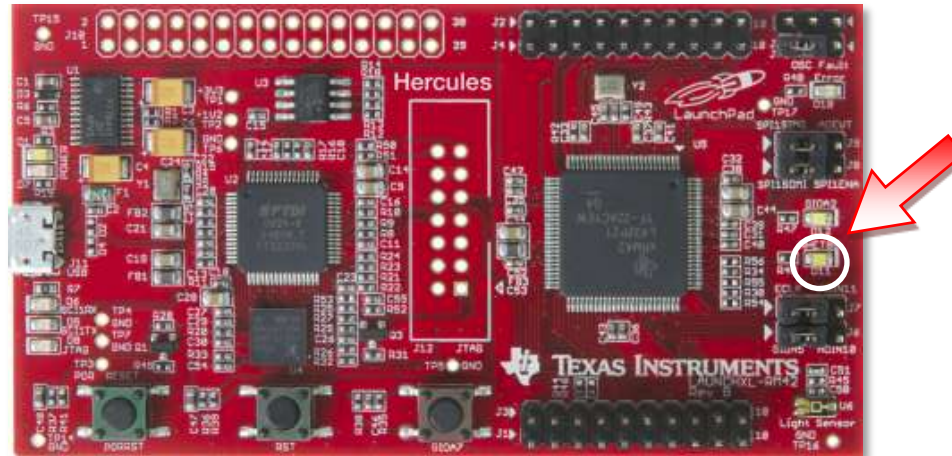
Testing our Program



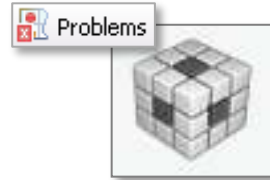
- Click the green arrow on the debug tab to run our program



- Alternatively the program can be run without the debugger connected by pressing the PORRST button on the LaunchPad
- Clicking the red square on the debug tab to terminate the debugger's connection
- Hitting the reset button on the board and observe the behavior of the NHET LED



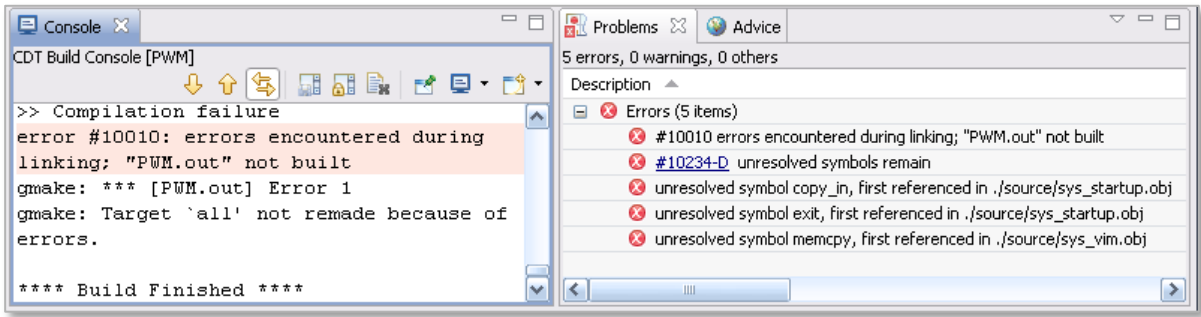
- Congratulations! You have completed the exercise.



Possible Errors

- **RM42x kits:**

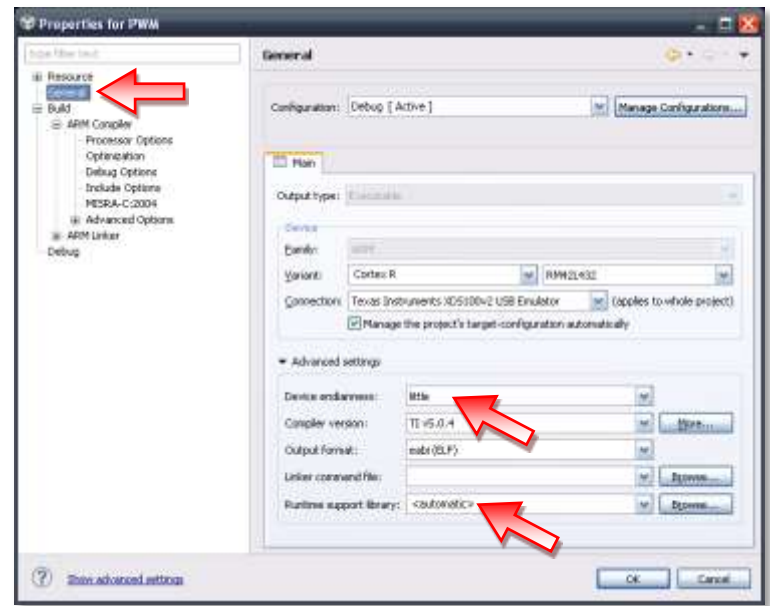
A build error may occur when using some versions of Code Composer Studio before v5.5



This error occurs because certain versions of CCS do not include the RTS (Run Time Support Library) for the little endian non floating point Cortex-R4 (rtsv7R4_T_le_eabi.lib) by default.

- To resolve this issue:
- 1) Open the “Properties” for the CCS project
 - 1) In the “General” Settings:
 - Set the “Device endianness:” to “little”
 - Set the “Runtime support library:” to “<automatic>”
 - 2) Re-Build the CCS project

NOTE: It may take 3 to 5 min to compile the RTS Library after this configuration change is made.

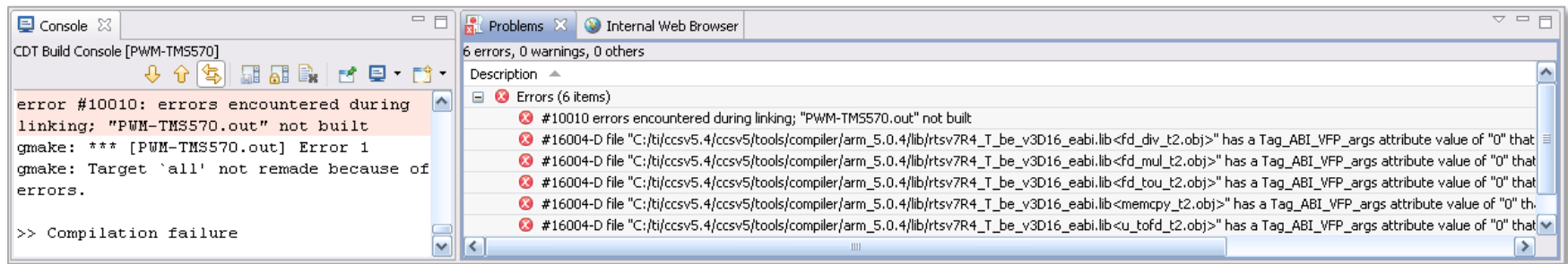




Possible Errors

- TMS570 kits:**

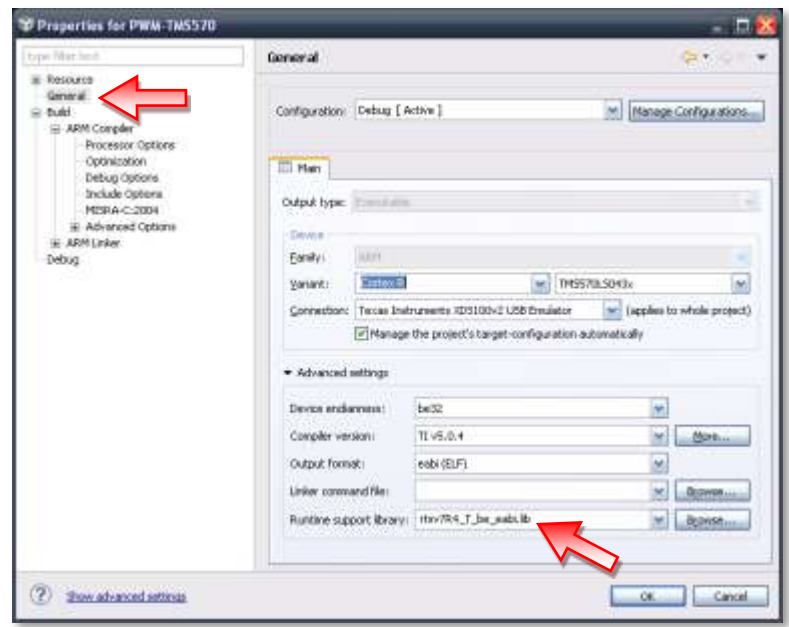
A build error may occur when using some versions of CCS before v5.5



This error occurs because certain versions of CCS set the floating point RTS (Run Time Support Library) for the big endian for non floating point Cortex-R4 MCUs by default.

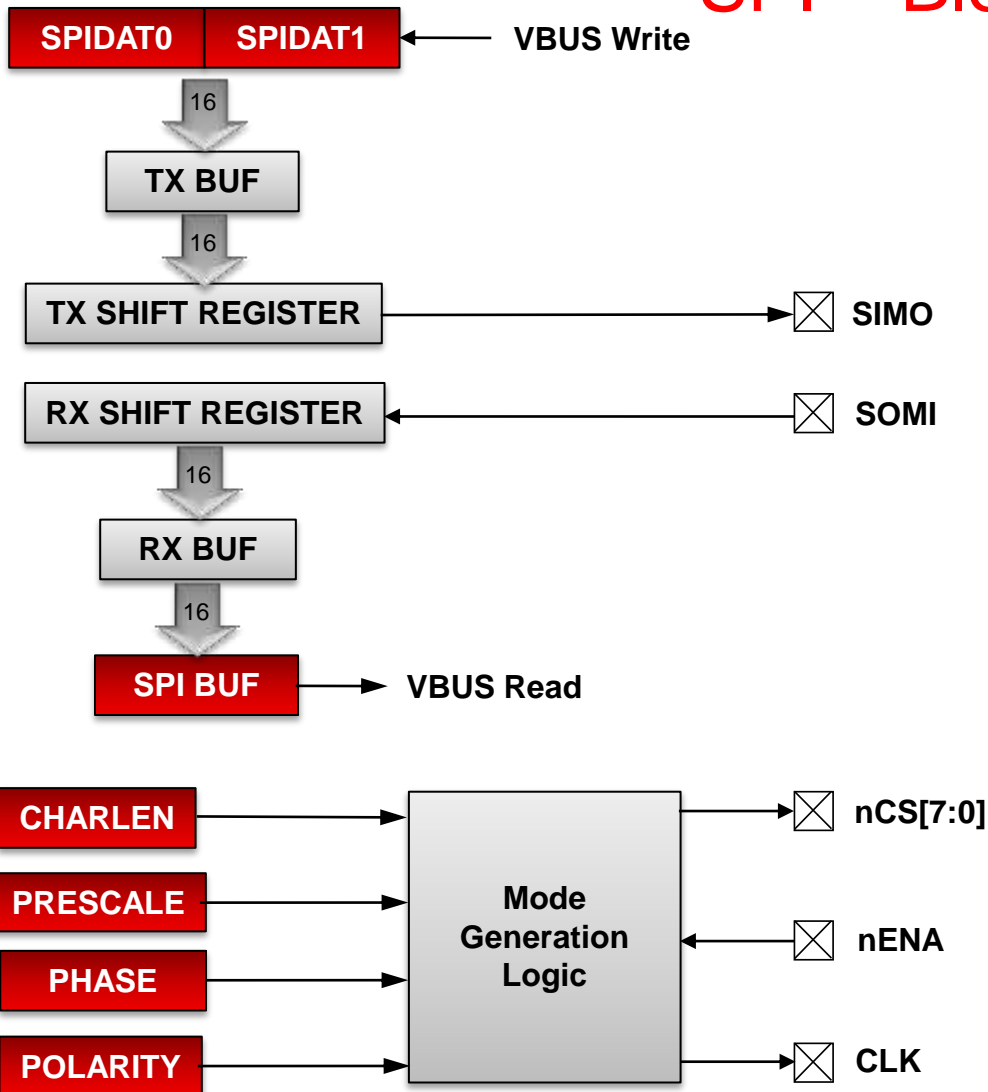
- To resolve this issue:
- 1) Open the “Properties” for the CCS project
 - 1) In the “General” Settings:
 - Set the “Runtime support library:” to “<rtsv7R4_T_be_eabi.lib>”

NOTE: It may take 3 to 5 min to compile the RTS Library after this configuration change is made.



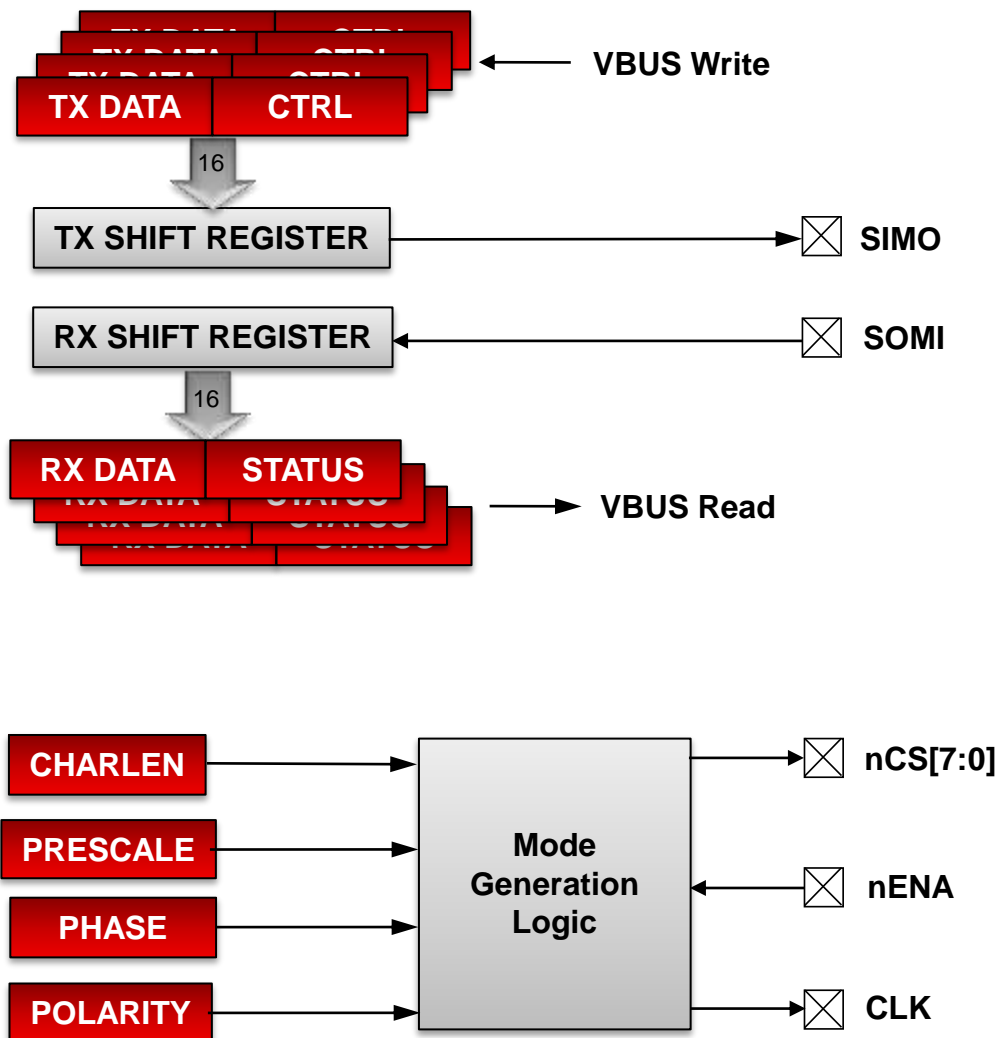
Multi-Buffered Serial Peripheral Interface (MibSPI)

SPI – Block Diagram & Features



- 16-bit Shift Registers
- Double-buffered TX and RX
- Master or Slave Mode
- Up to 4 SIMO / SOMI in parallel
- Selectable MSbit or LSbit first transfer
- Unused pins available as GP I/O
- CLK frequency VCLK/2 to VCLK/256
- 2- to 16-bit character length
- Selectable CLK phase and polarity
- Interrupt / DMA requests when
 - TX buffer empty
 - RX buffer full

MibSPI – Block Diagram, Features



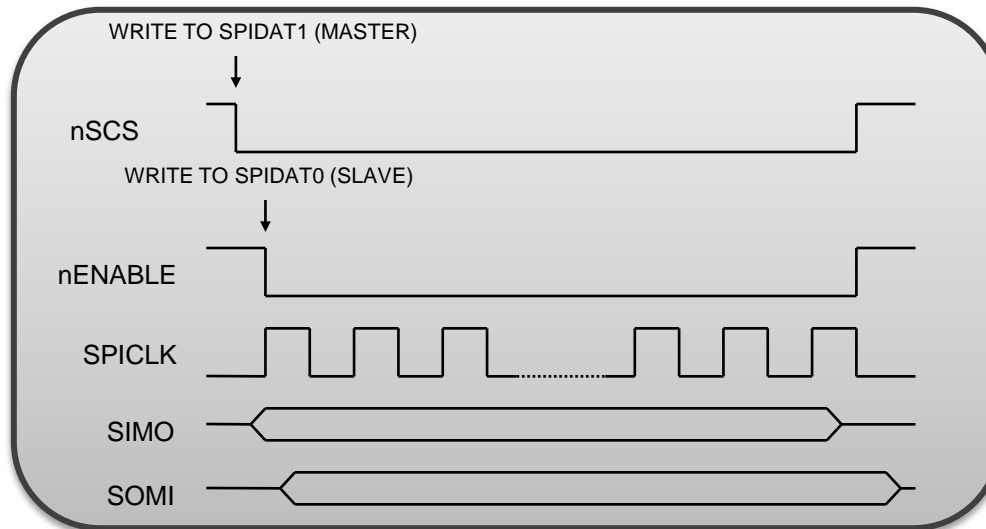
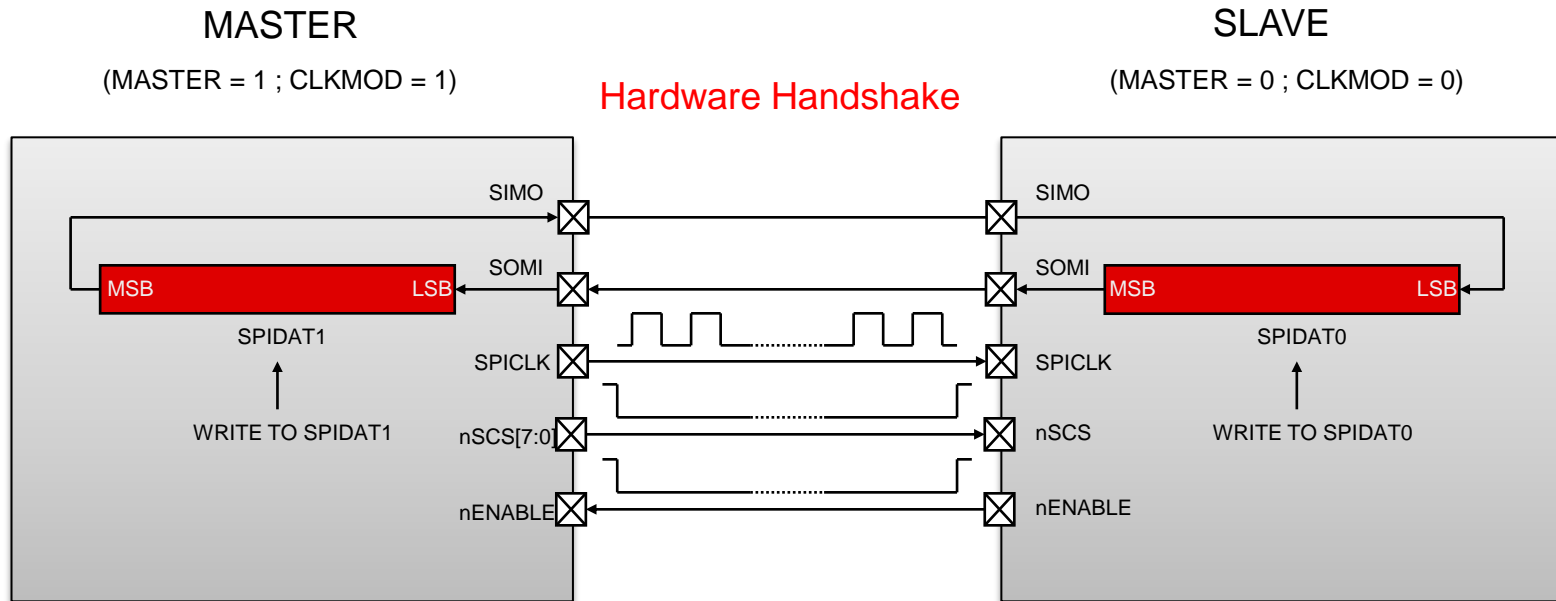
- 16-bit Shift Registers
- Up to 128 buffers for TX and RX
- Up to 8 transfer groups
- 15 sources to trigger transfers
- Memory protected by parity
- Master or Slave Mode
- Up to 4 SIMO / SOMI in parallel
- Selectable MSbit or LSbit first transfer
- Unused pins available as GP I/O
- CLK frequency VCLK/2 to VCLK/256
- 2- to 16-bit character length
- Selectable CLK phase and polarity
- Programmable interrupt and DMA request generation conditions
- Up to 16 DMA requests

SPI / MibSPI Safety Features



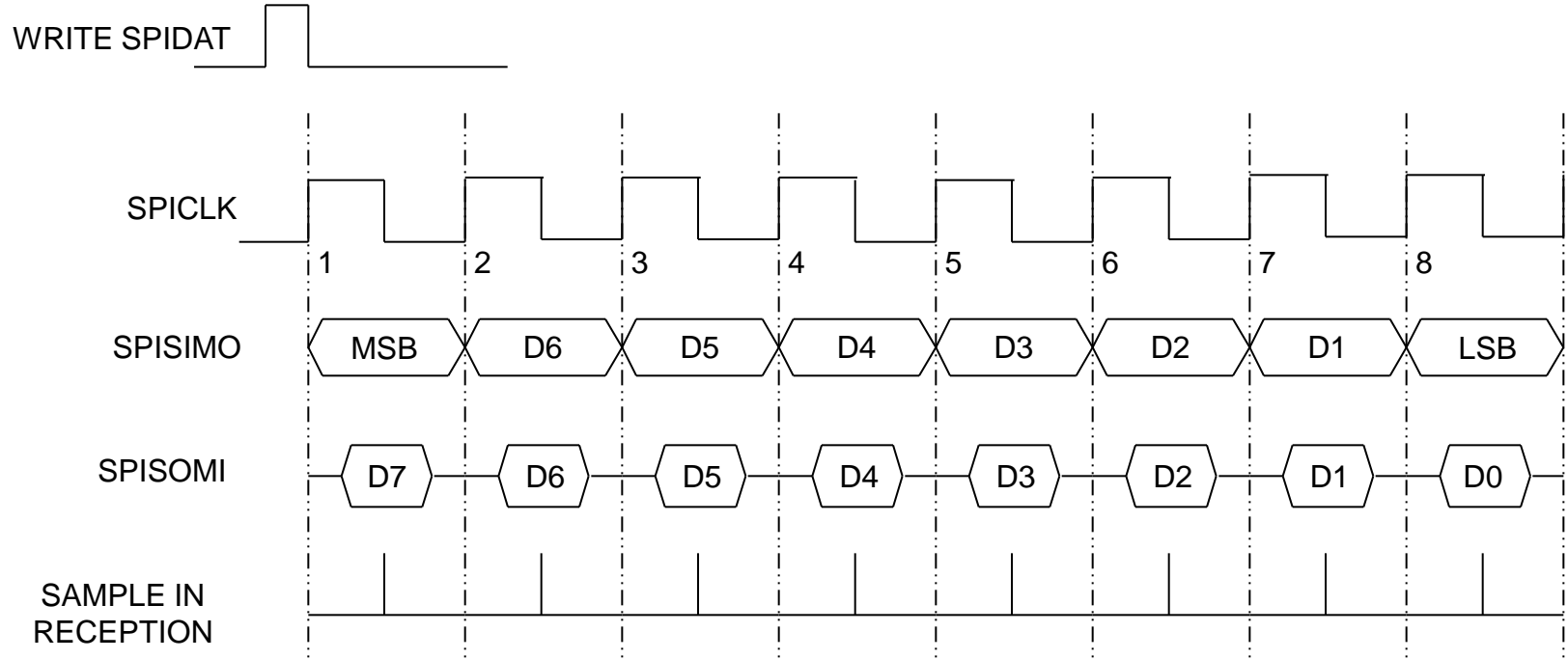
- Parity Error detection for all reads from MibSPI RAM
- Continuous monitoring of transmitted data in master and slave modes
- Detection of slave de-synchronization (master mode only)
- Timeout for a non-responsive slave (master mode only)
- Receiver overrun interrupt condition to prevent data loss
- Detection of a mismatch in data length

Transfer Mode – Five Pin Option



Clock Options

CLOCK POLARITY = 0, CLOCK PHASE = 0

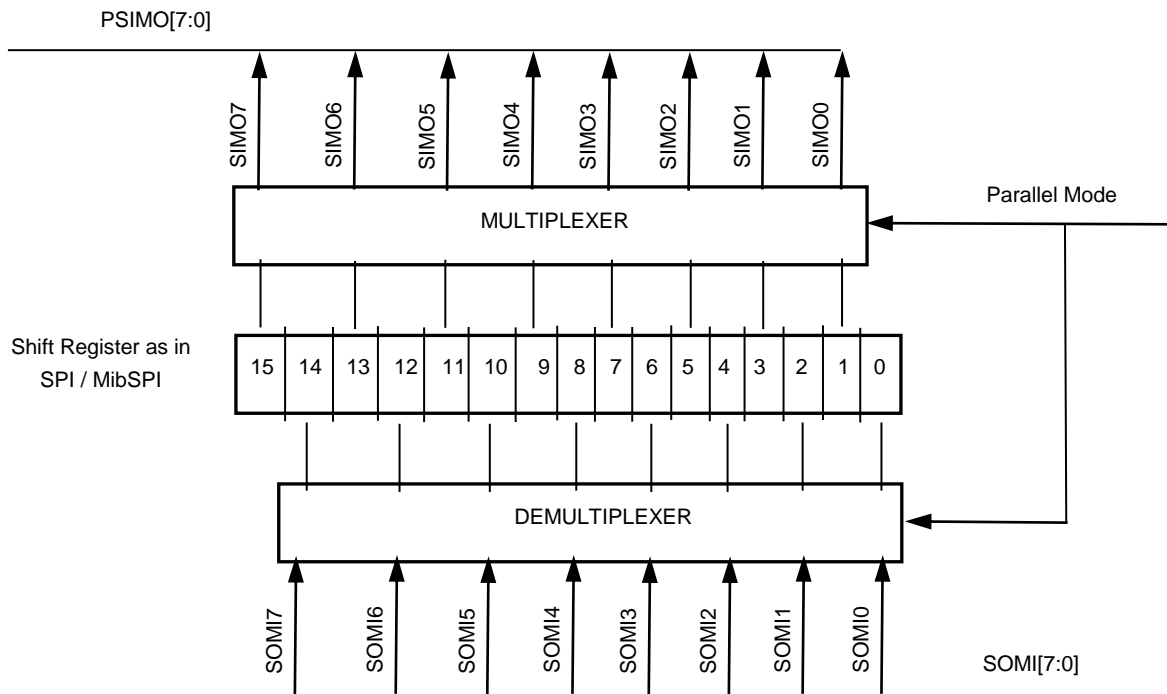


CLOCK PHASE = 0 (SPICLK WITHOUT DELAY)

- DATA IS OUTPUT ON THE RISING EDGE OF SPICLK
- INPUT DATA IS LATCHED ON THE FALLING EDGE OF SPICLK
- A WRITE TO THE SPIDAT REGISTER STARTS SPICLK

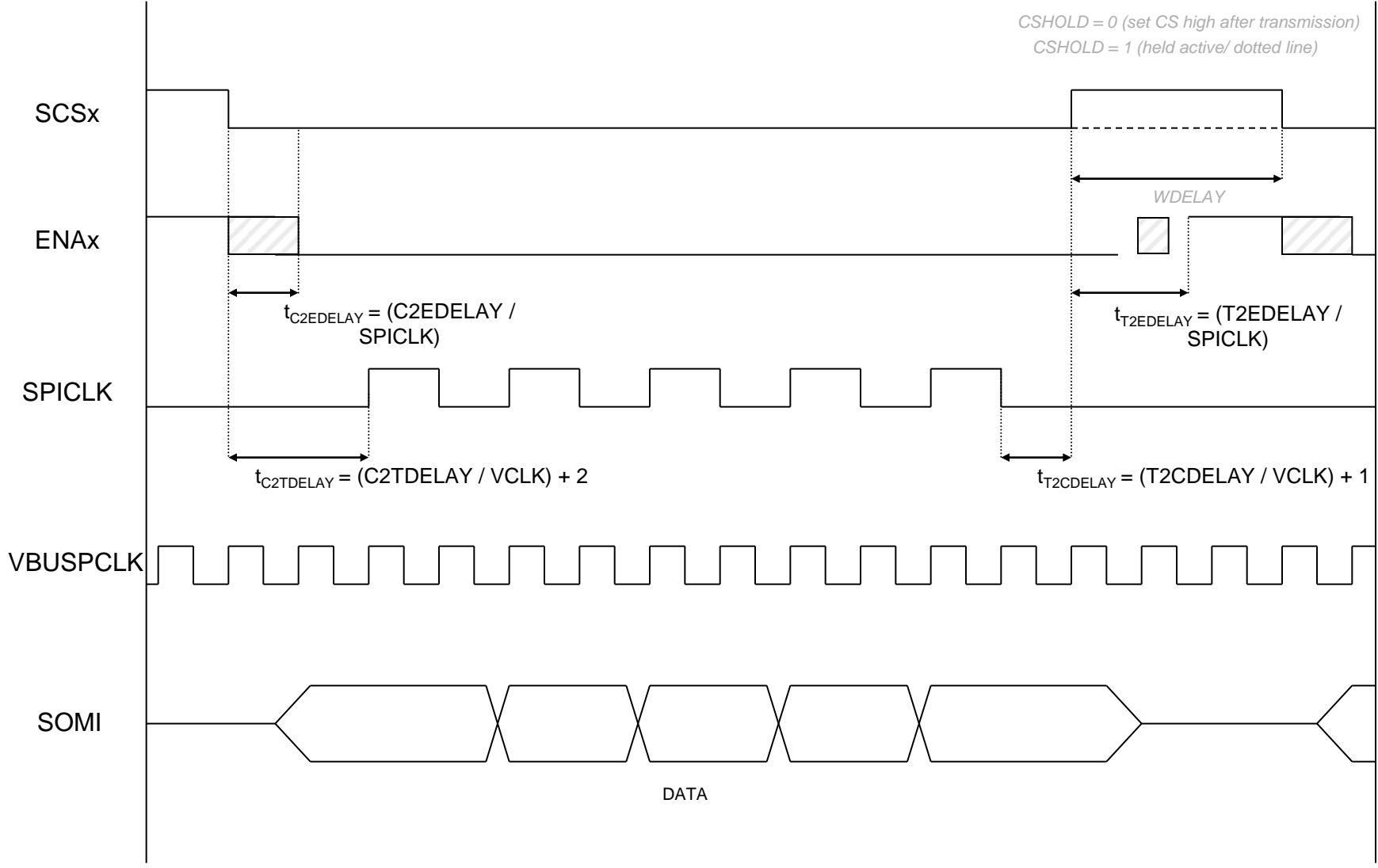
SPI / MibSPI Parallel Mode

- In order to achieve higher data flow, the parallel mode of the SPI / MibSPI enables the module to send data over more than one data line (Parallel 2, or 4).
- Figure of Parallel Mode with Shift register MSB first:



- Notes:
 - When parallel mode is used, the data length must be set as 16 bits
 - If parity is enabled one additional SPICLK will trigger the parity bit transfer

Timing Setup – Delay Register (SPIDELAY)



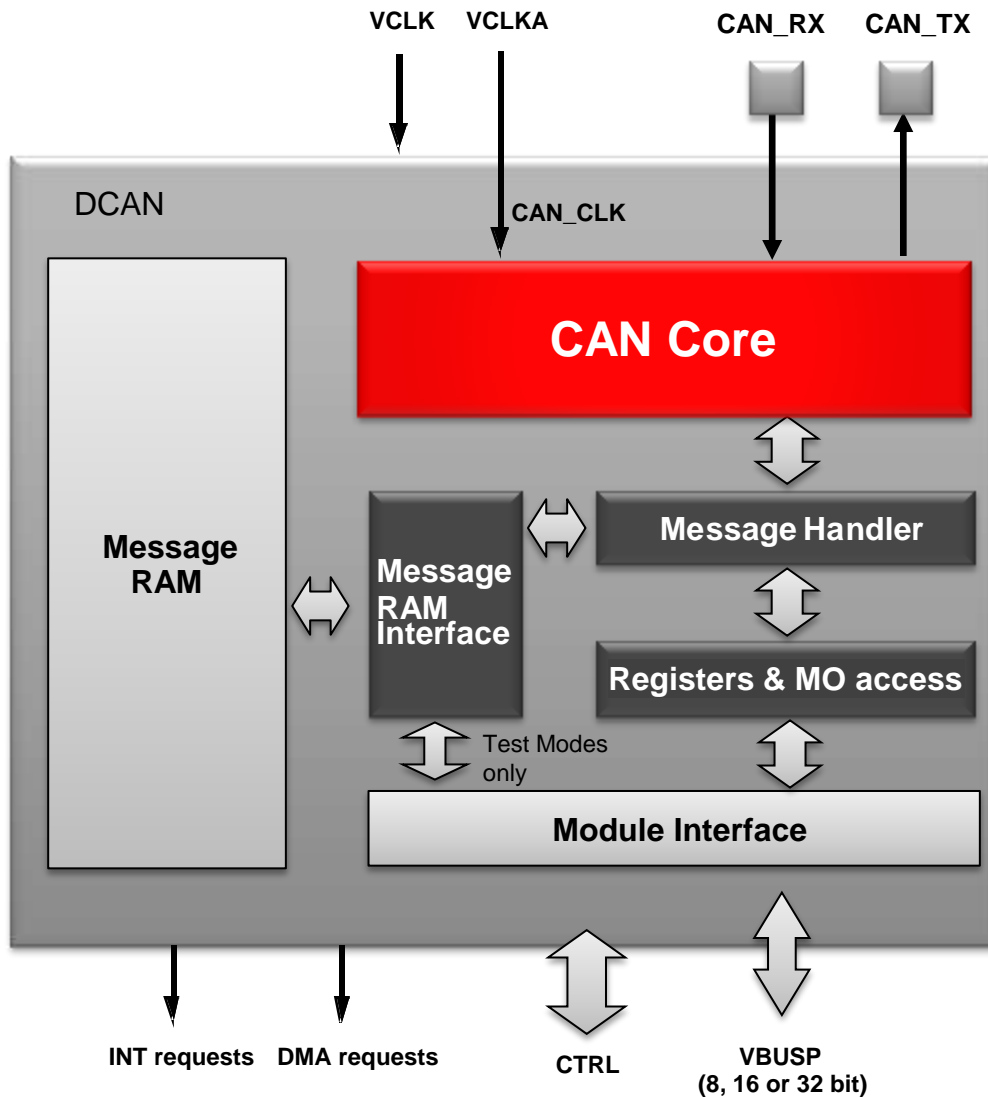
Controller Area Network (DCAN)

DCAN Features Overview



- Full CAN according to protocol version 2.0 part A, B
- Standard and Extended Identifiers
- Programmable Bit Timing, Bit rates up to 1 MBit/s
- Up to 128 Message Objects (MO)
- Identifier Masks for each Message Object
- Programmable FIFO mode for Message Objects
- Dual clock feature
- Possible automatic retransmission of a frame in case of lost arbitration or error
- Bus diagnostic: Bus off, Bus error passive, Bus error warning, Bus stuck dominant
- Frame error report: CRC, Stuff, Form, Bit and Acknowledgement errors
- Programmable loop-back modes for self-test operation
- Suspend modes for debug support
- Parity check mechanism for all RAM modules

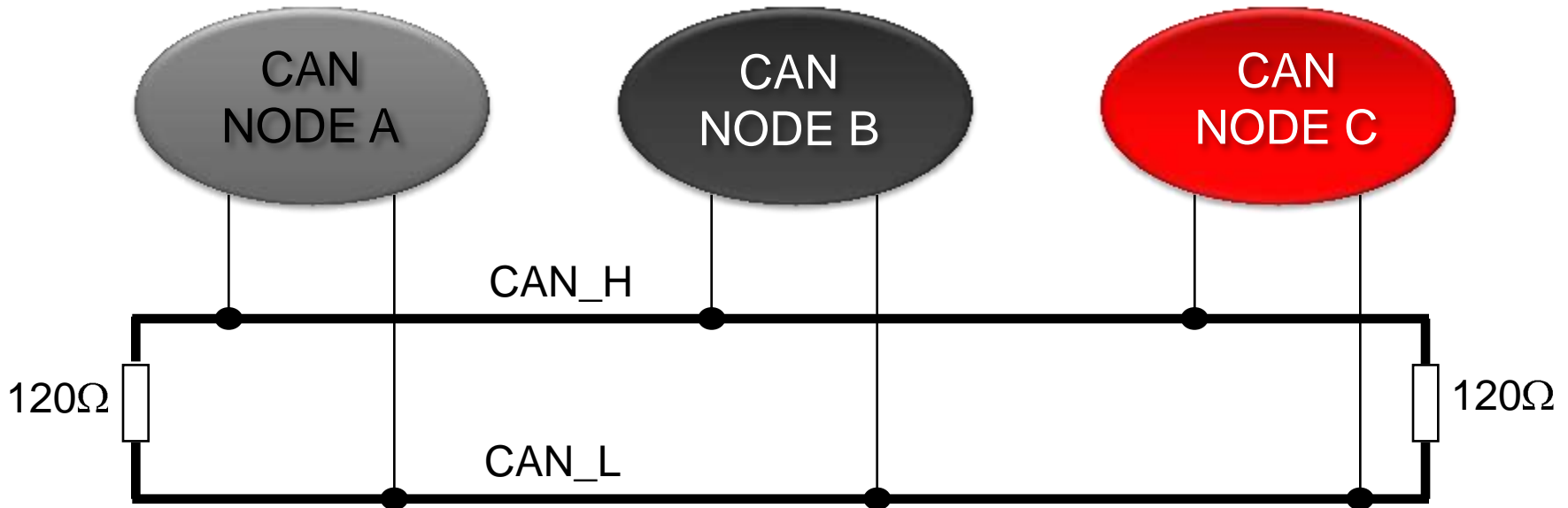
DCAN Block Diagram & Features



- Full CAN (protocol version 2.0 A, B)
- CAN Core
 - Handles all CAN protocol functions
- Message Handler
 - Controls data transfer between CAN core, message interface registers and RAM
 - Handles acceptance filtering and interrupt/DMA requests
- Message RAM
 - Up to 64 Message Objects
- Registers & Message Object access (IFx)
 - Status and configuration registers for module setup and indirect Message Object access through interface registers (IFx)
- Module Interface
 - 2-bit interface to VBUS peripheral clock domain

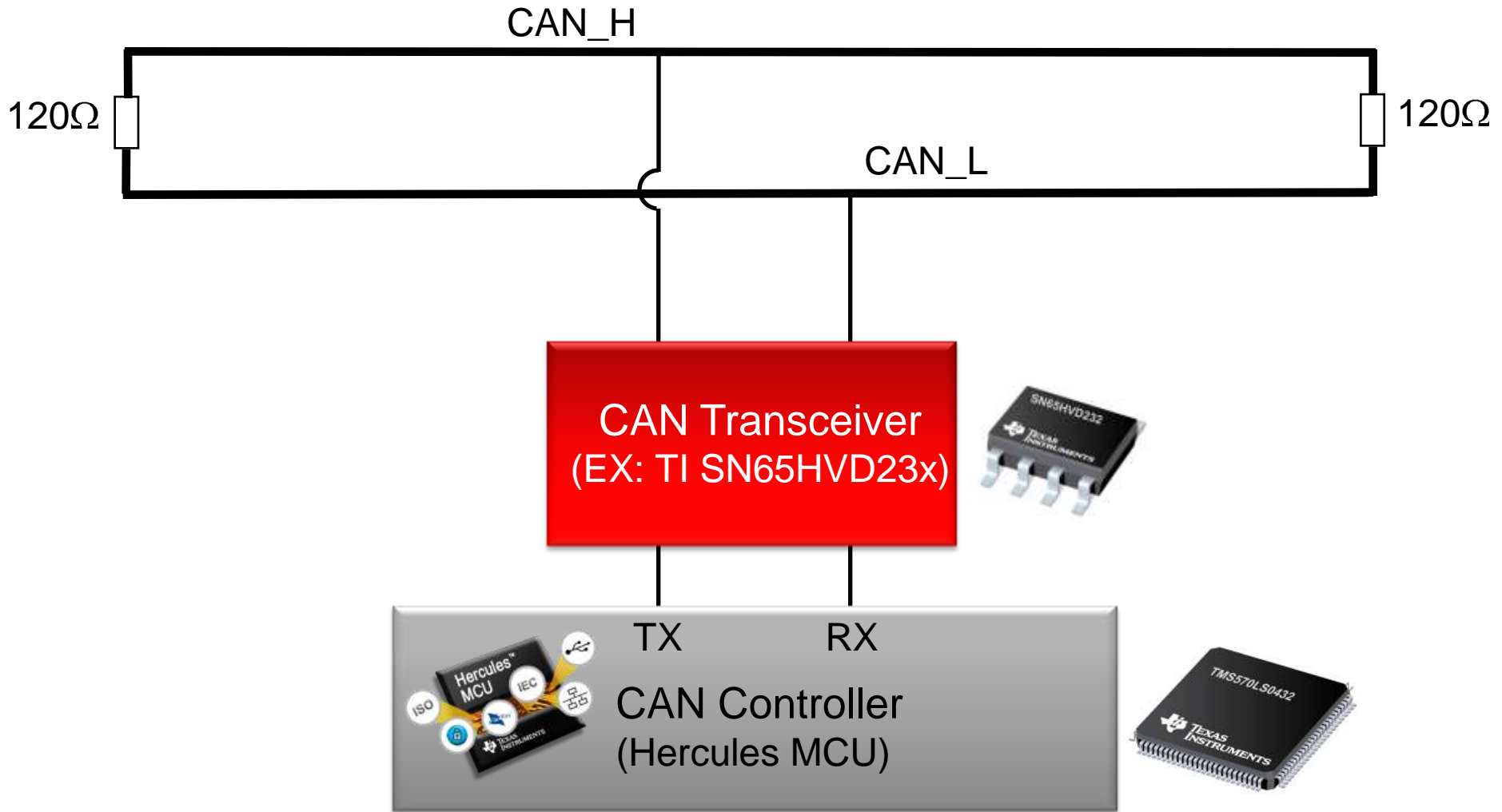
CAN Bus

- Two wire differential bus (usually twisted pair)
- Max. bus length depend on transmission rate
 - 40 meters @ 1 Mbps



CAN Node

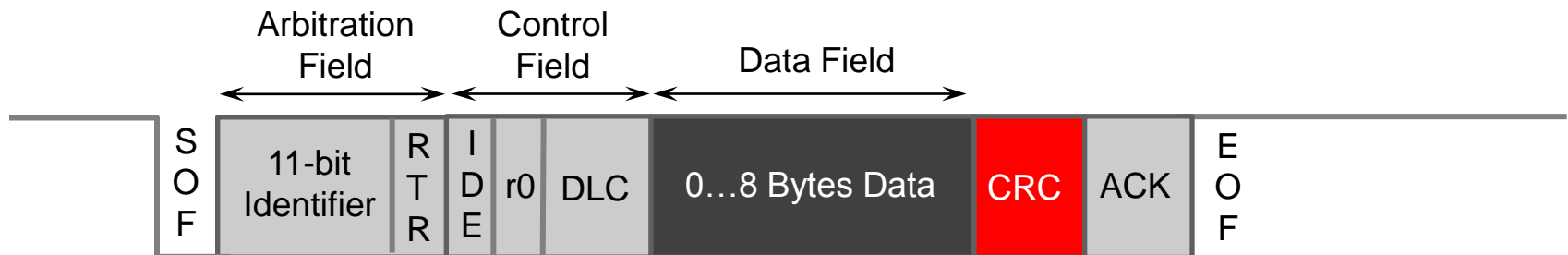
Wired-AND Bus Connection



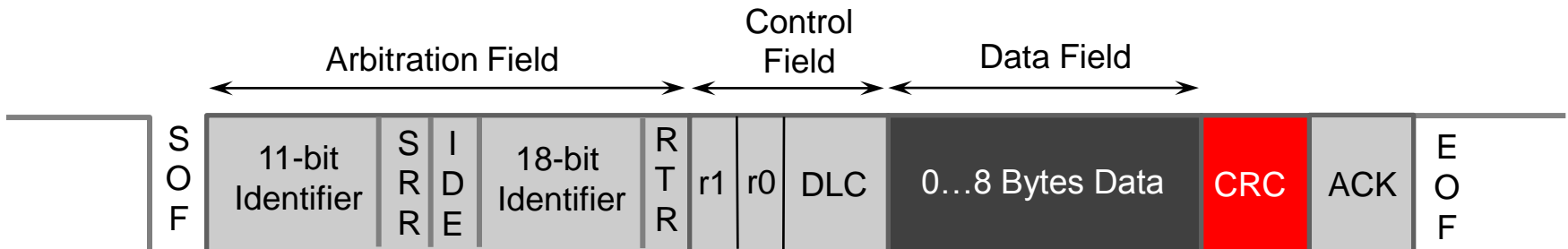
CAN Message Format

- Data is transmitted and received using Message Frames
- 8 byte data payload per message
- Standard and Extended identifier formats

Standard Frame: 11-bit Identifier (CAN v2.0A)



Extended Frame: 29-bit Identifier (CAN v2.0B)



FlexRay / Transfer Unit

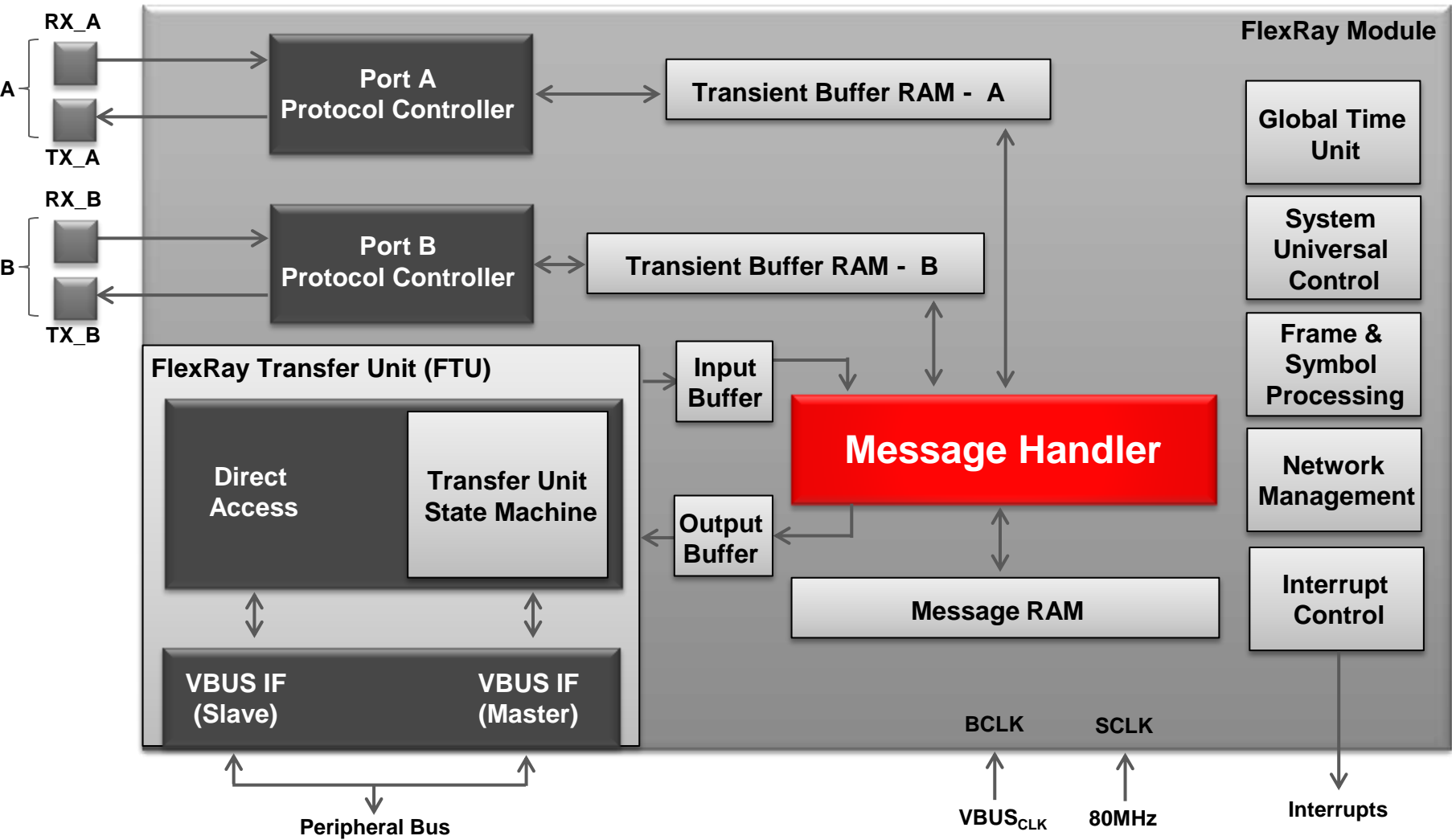
(Available on select TMS570 MCUs Only)

FlexRay Feature Overview

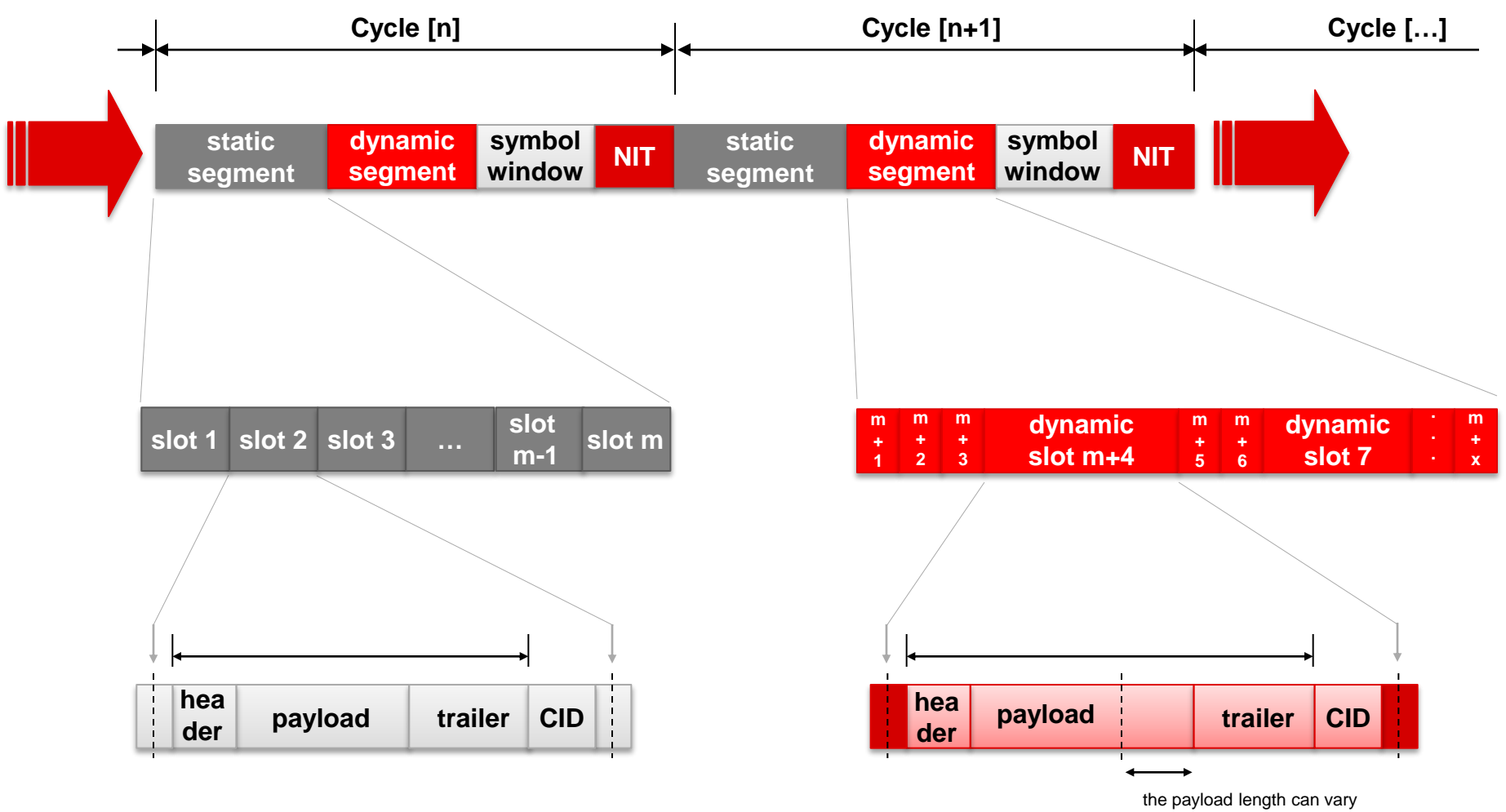


- Open Bus System
- Support of redundant transmission channels
- Data rate of 20 Mbit/sec (10Mbit/sec per channel)
- Support of a fault tolerant synchronized global time base
- Static and dynamic data transmission (scalable)
 - Deterministic data transmission
 - Arbitration free transmission
- Fault tolerant and time triggered services implemented in hardware
- Support of optical and electrical physical layers

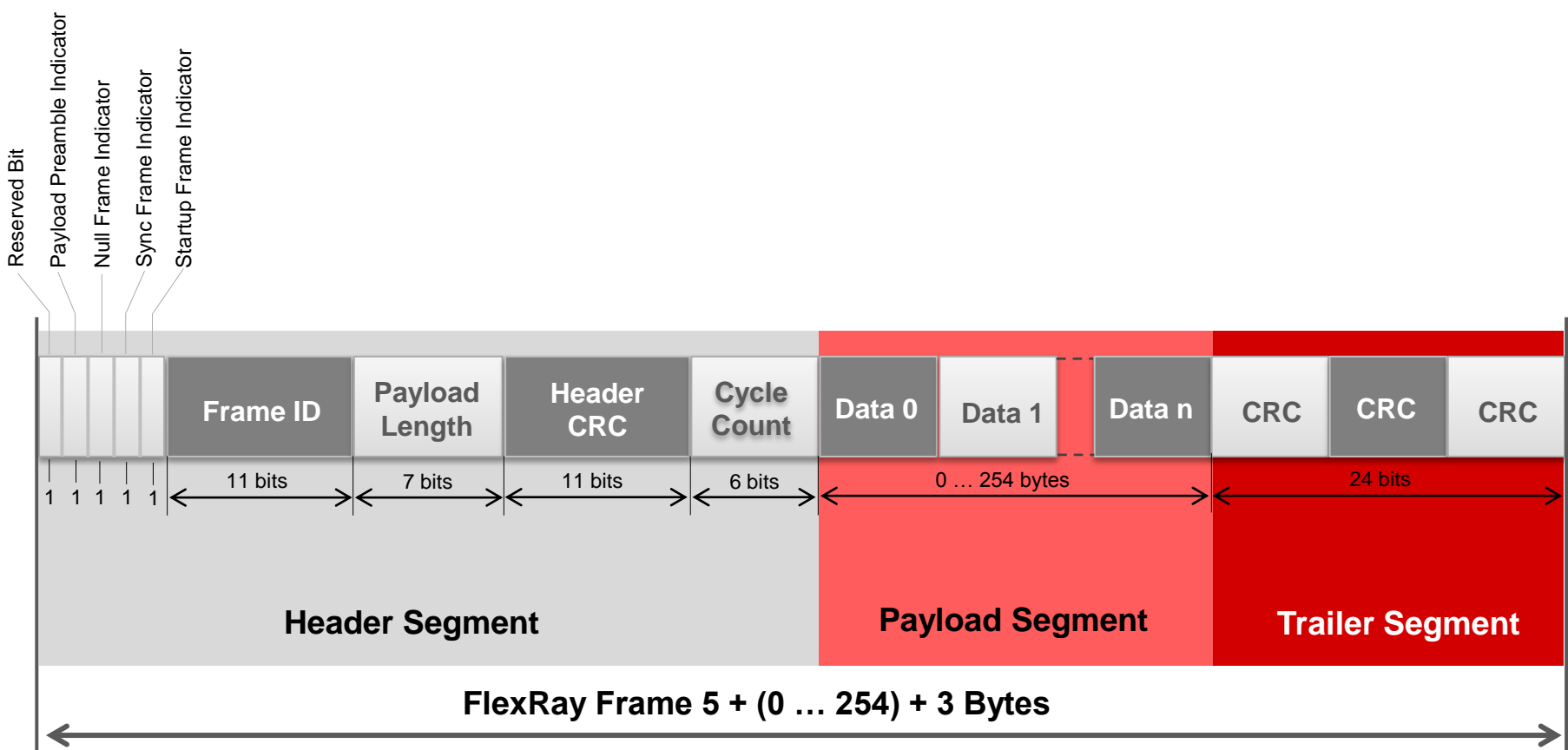
FlexRay Block Diagram



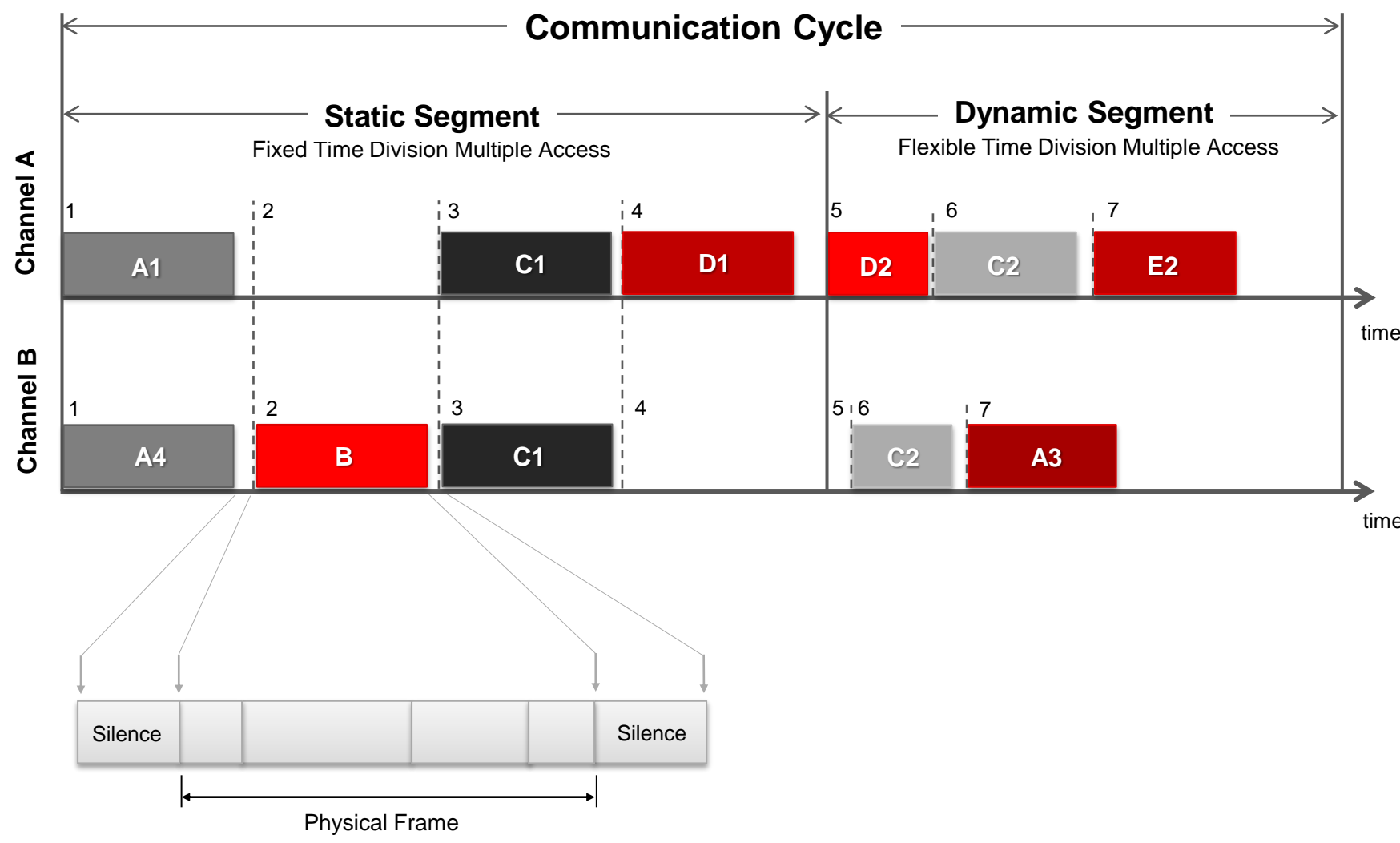
FlexRay Communication Structure



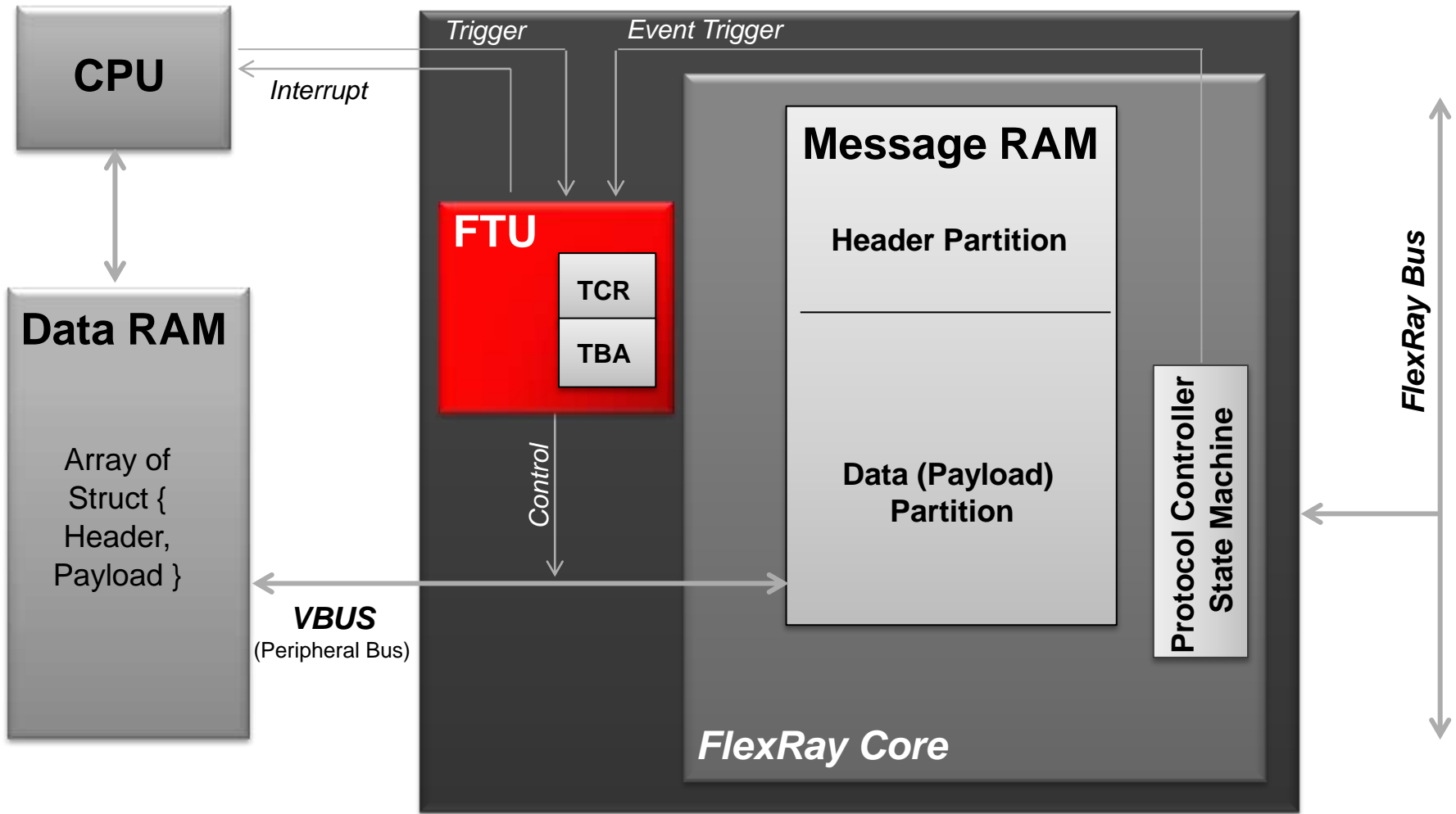
FlexRay Message Frame Format



FlexRay Communication Cycle



FTU Data Transfer Scheme



FlexRay Transfer Unit Key Features

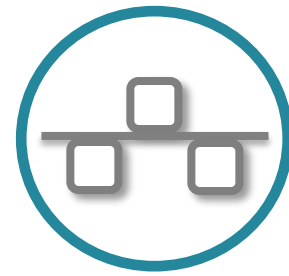
- Data Transfer without CPU interaction
 - From FlexRay Message RAM to Data RAM (Read)
 - From Data RAM to FlexRay Message RAM (Write)
- Transfer Types
 - data and header section
 - header section only
 - data section only
- Transfer Configuration RAM (with Parity)
 - Configures the transfer sequence
 - Parity protection
- Triggers to Start a Transfer
 - CPU driven (single transfer sequence)
 - Event driven (single or continuous transfer sequence)

FlexRay Transfer Unit Key Features...

- Different Transfer Conditions
 - If the status flags (header section) of the respective message buffer has been updated
 - If the data section of the respective message buffer has been updated
 - Always
- Maskable interrupt generation when Message Buffer transfer is finished
- Memory Protection Unit
 - One memory section (start- and end address) can be defined
 - No memory section is setup after reset

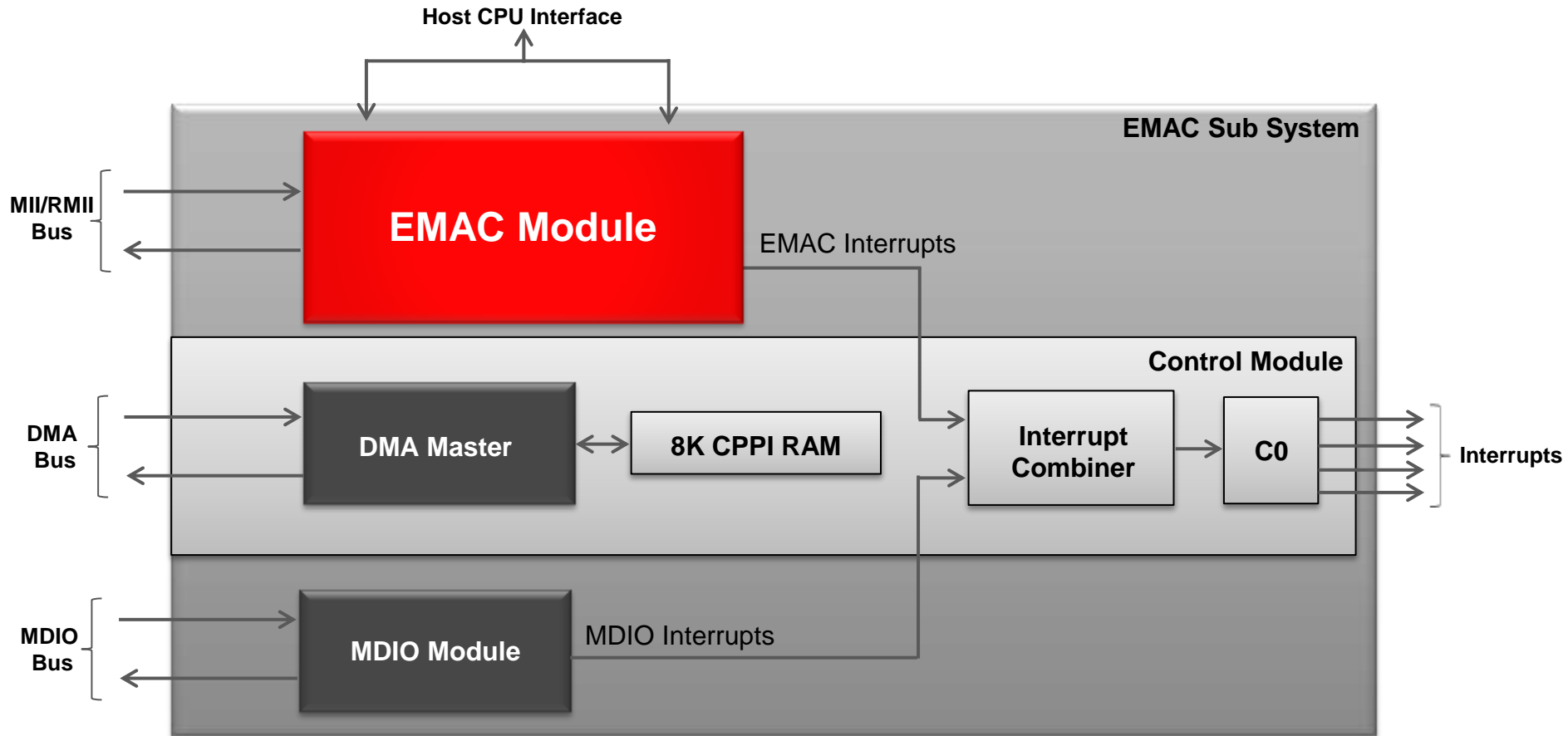
Ethernet Media Access Controller (EMAC)

EMAC Sub System: Features

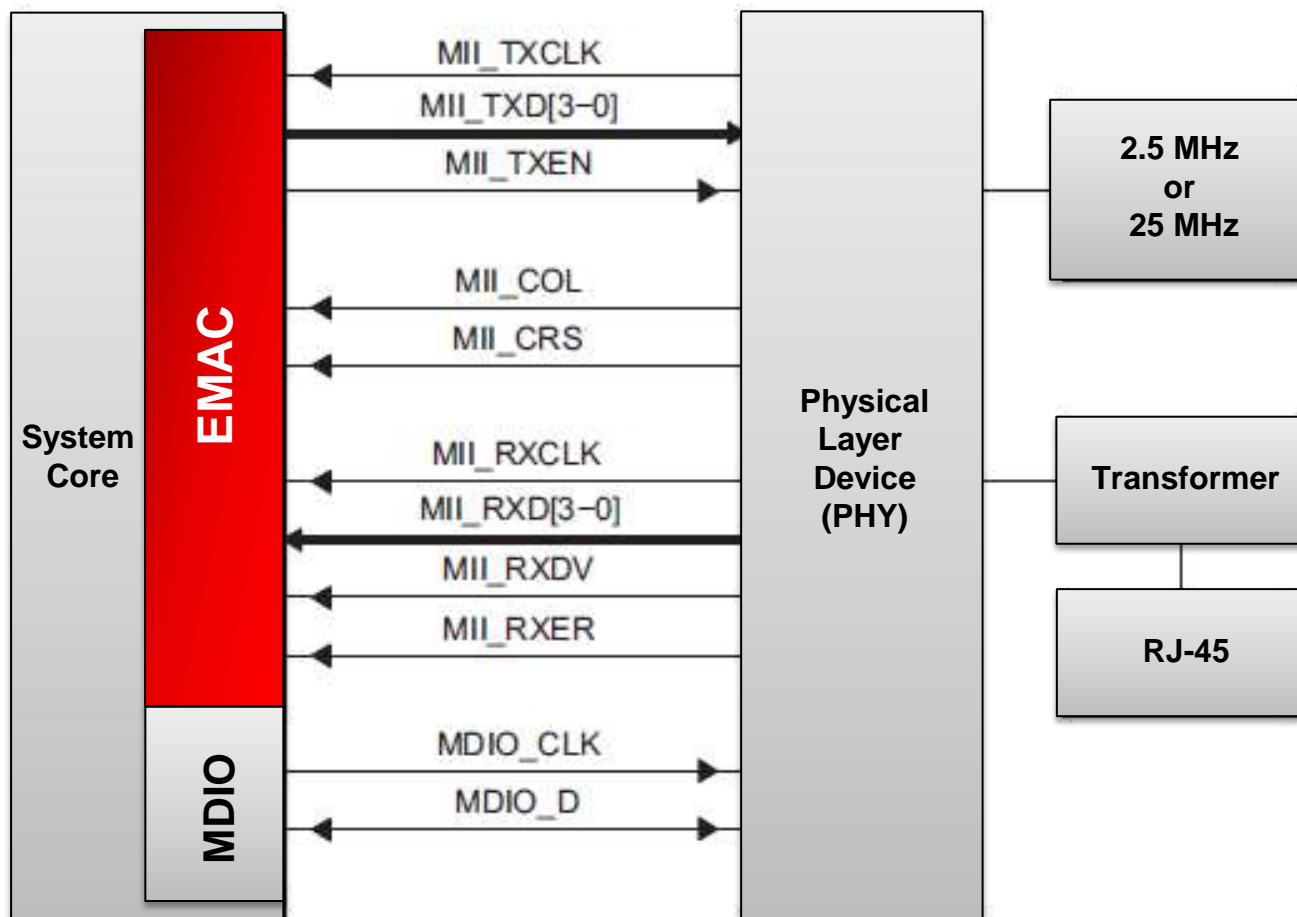


- EMAC Module
 - Synchronous 10/100 Mbps operation
 - Standard MII or RMII to external physical layer device (PHY)
 - Master port for transfers to/from internal and external RAM
 - Transmit and Receive Quality-of-Service (QoS) support
- EMAC Control Module
 - Ether-Stats and 802.3-Stats statistics gathering
 - 8 kB local EMAC descriptor memory (CPPI RAM)
 - Enough to transfer up to 512 Ethernet packets without CPU intervention
 - Programmable interrupt logic
 - Allows restriction of back-to-back interrupt generation
- MDIO Module
 - Implements the 802.3 serial management interface
 - Can control up to 32 Ethernet PHYs using a shared 2-wire bus
 - Used to configure each PHY connected to the EMAC

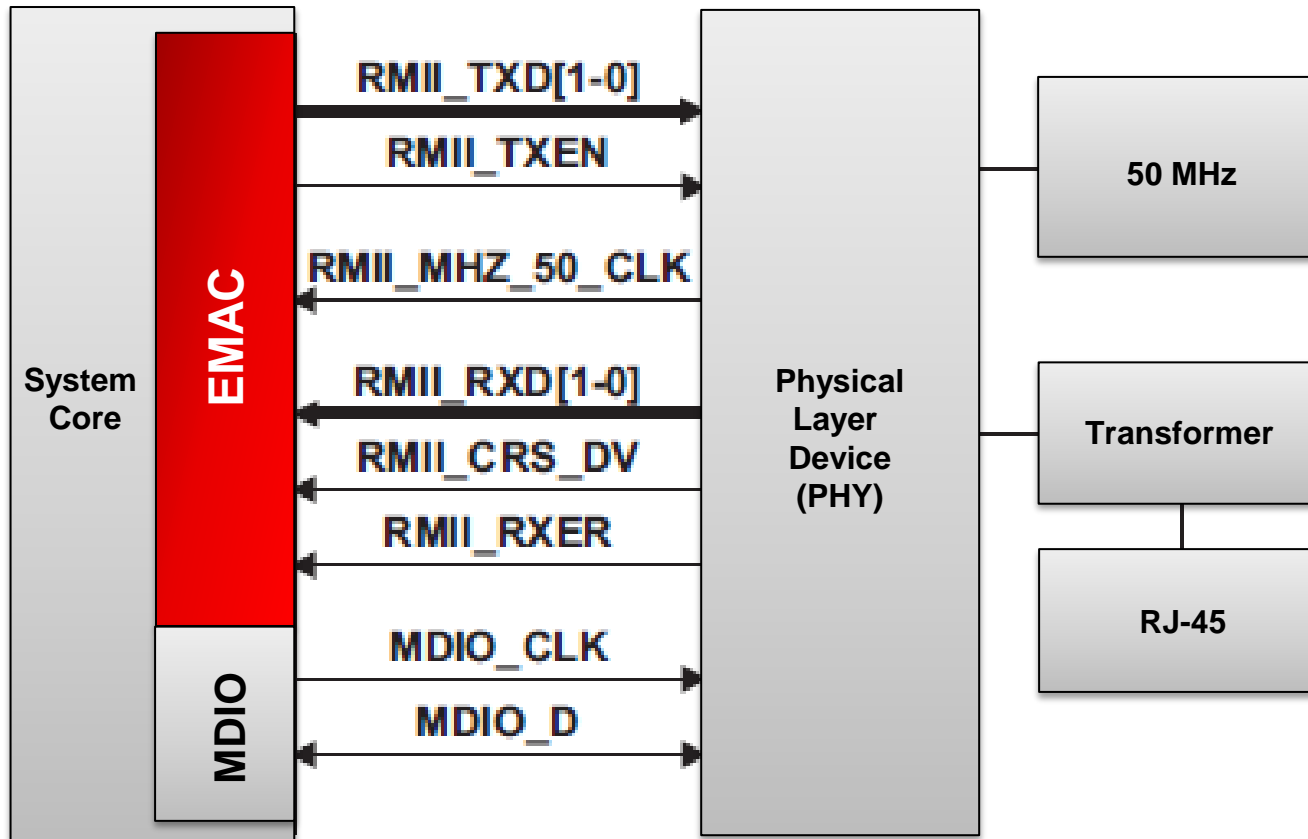
EMAC Block Diagram



MII Connections



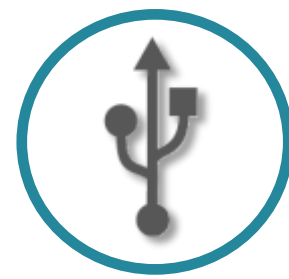
RMII Connections



Universal Serial Bus (USB)

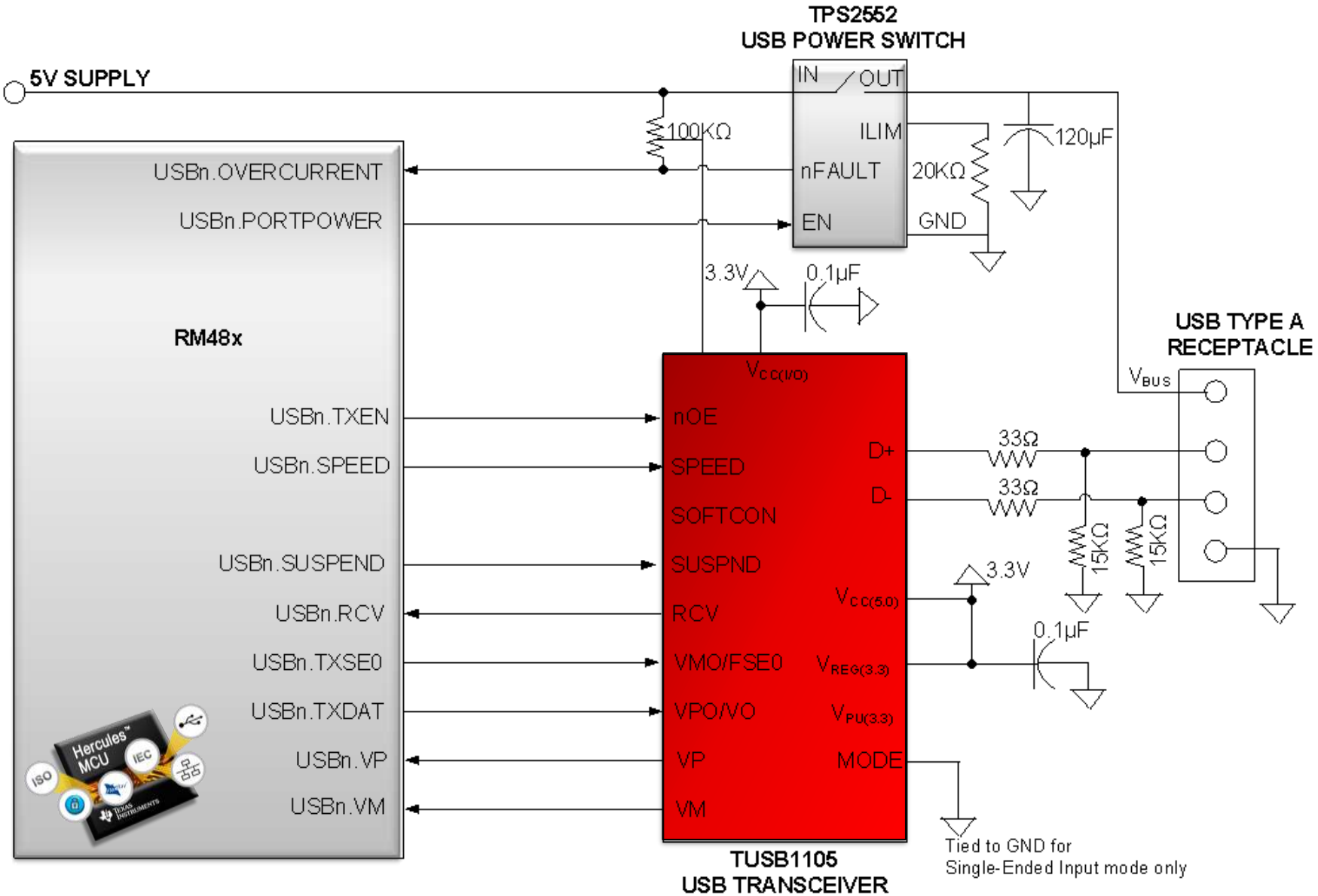
(Available on select RM MCUs Only)

USB Controller: Features



- **One full-speed USB device port**
 - Compliant to USB Specification Rev 2.0 and Rev 1.1
 - Interfaces host processor and the external USB transceiver (PHY)
- **Two USB host ports**
 - Compliant to USB Specification Rev 2.0
 - Based on Open Host Controller Interface (OHCI), Release 1.0a
 - Support for Overcurrent protection and automatic power switching
- **Second host port terminals are shared with device port terminals**
 - Can use 2 USB host ports, or 1 USB host port and 1 USB device port

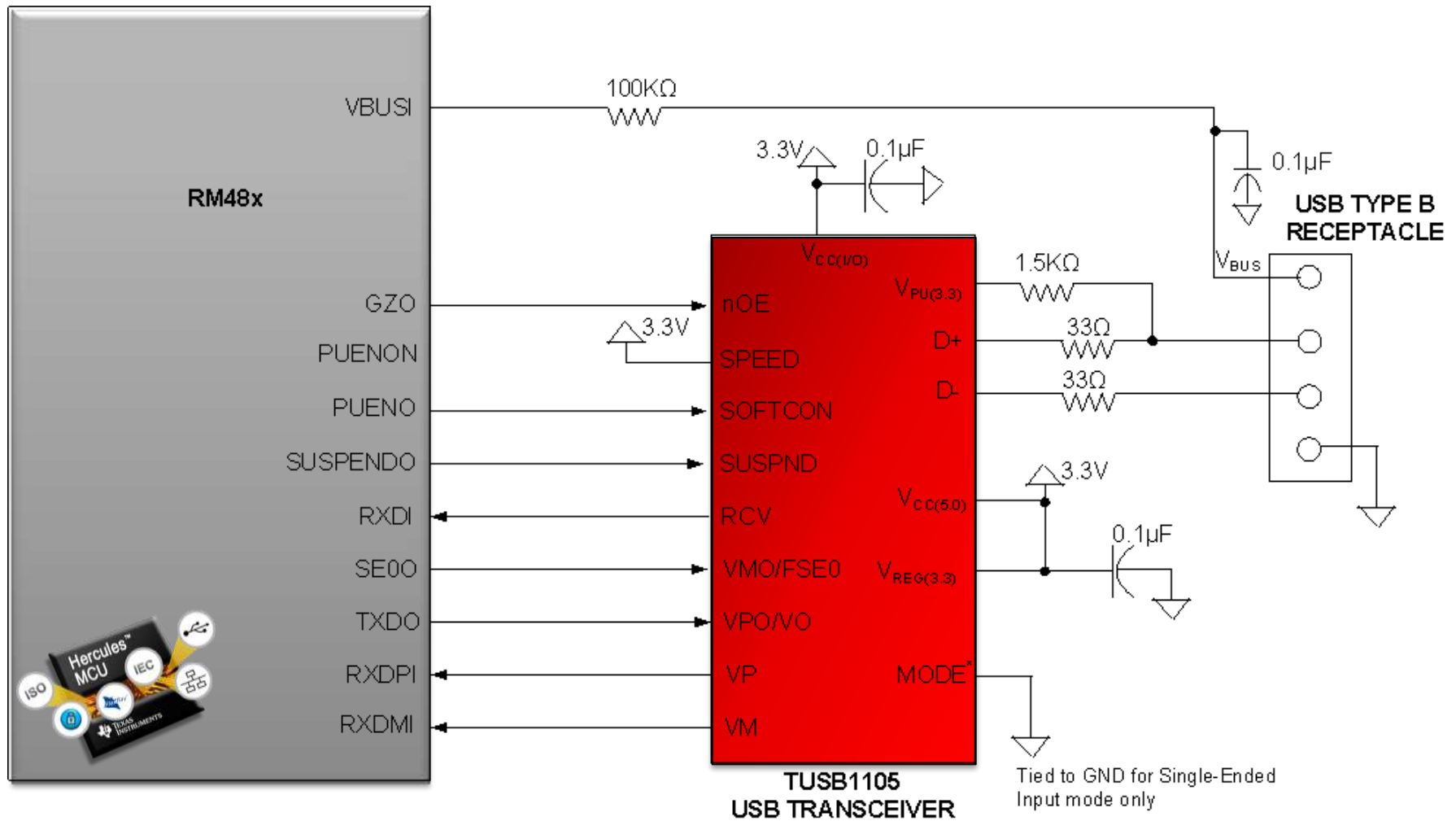
USB Host Port Connections



<http://www.ti.com/product/tusb1105>



USB Device Port Connections

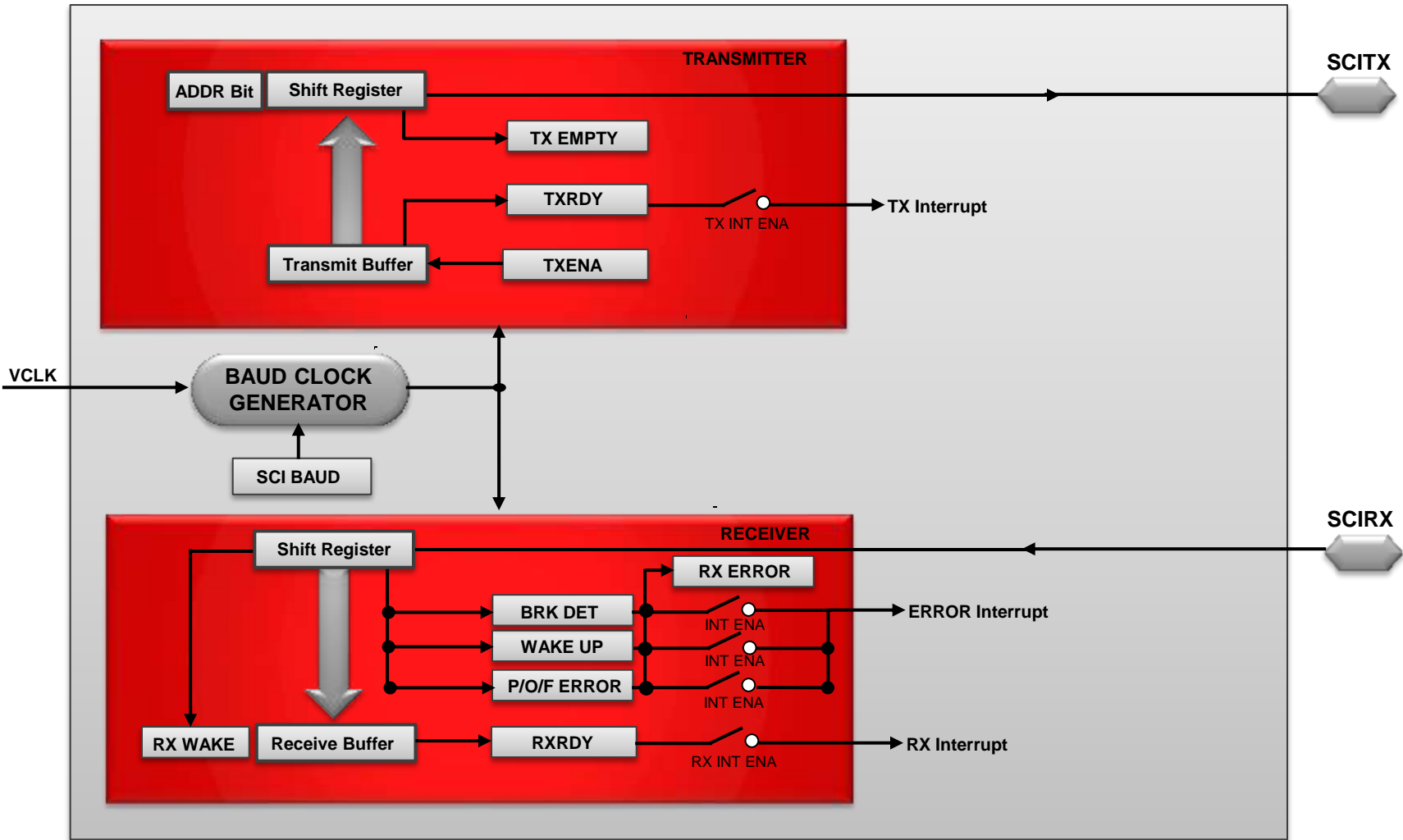


Serial Communication Interface (SCI/UART/LIN)

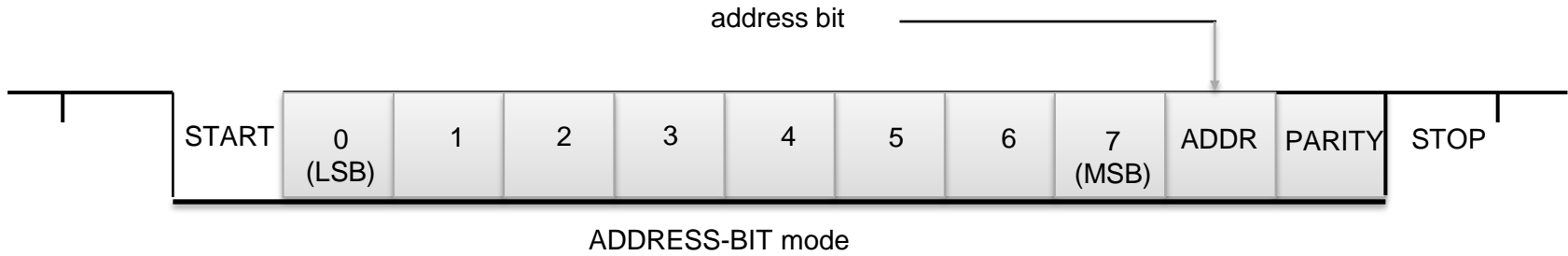
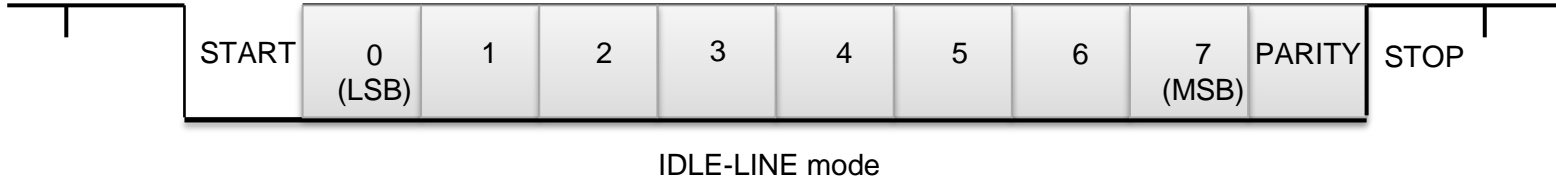
SCI Features

- Programmable Frame Format
 - 1 Start Bit
 - 1 to 8 Data Bits
 - 0 or 1 Address Bit
 - 0 or 1 Parity Bit
 - 1 or 2 Stop Bits
- Asynchronous Communications Format
- 2 Multiprocessor Modes with Wake-up Capability
 - Idle-Line Mode; Address-Bit Mode
- Programmable Baud Rate
 - More than 16 700 000 different Baud Rates
 - Max 3.125Mbps with 100MHz VCLK
- Error Detection
 - Parity, Overrun and Framing Error
 - Break Detect
- Noise Protection Capability
- Double-buffered Receive and Transmit Function

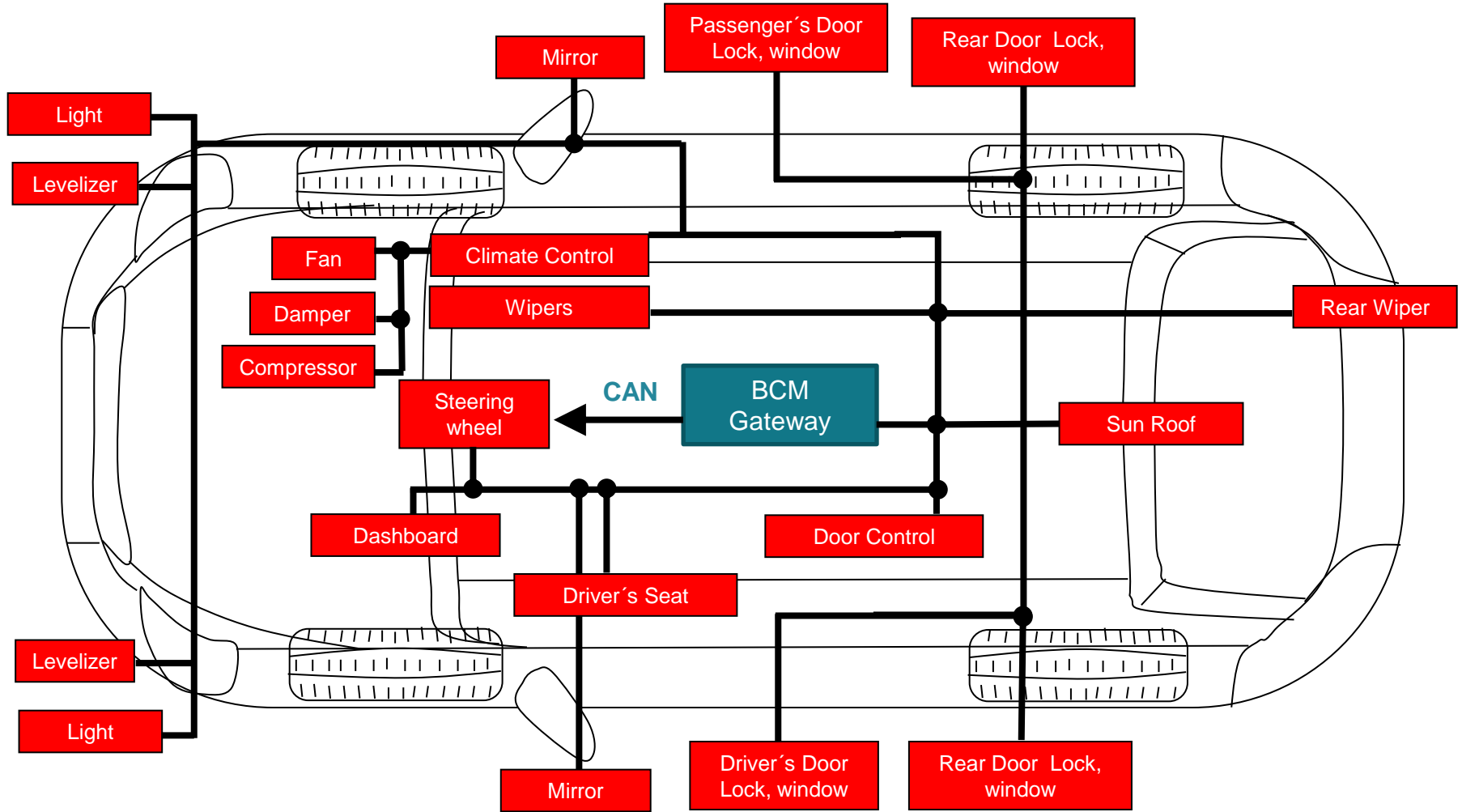
SCI Block Diagram



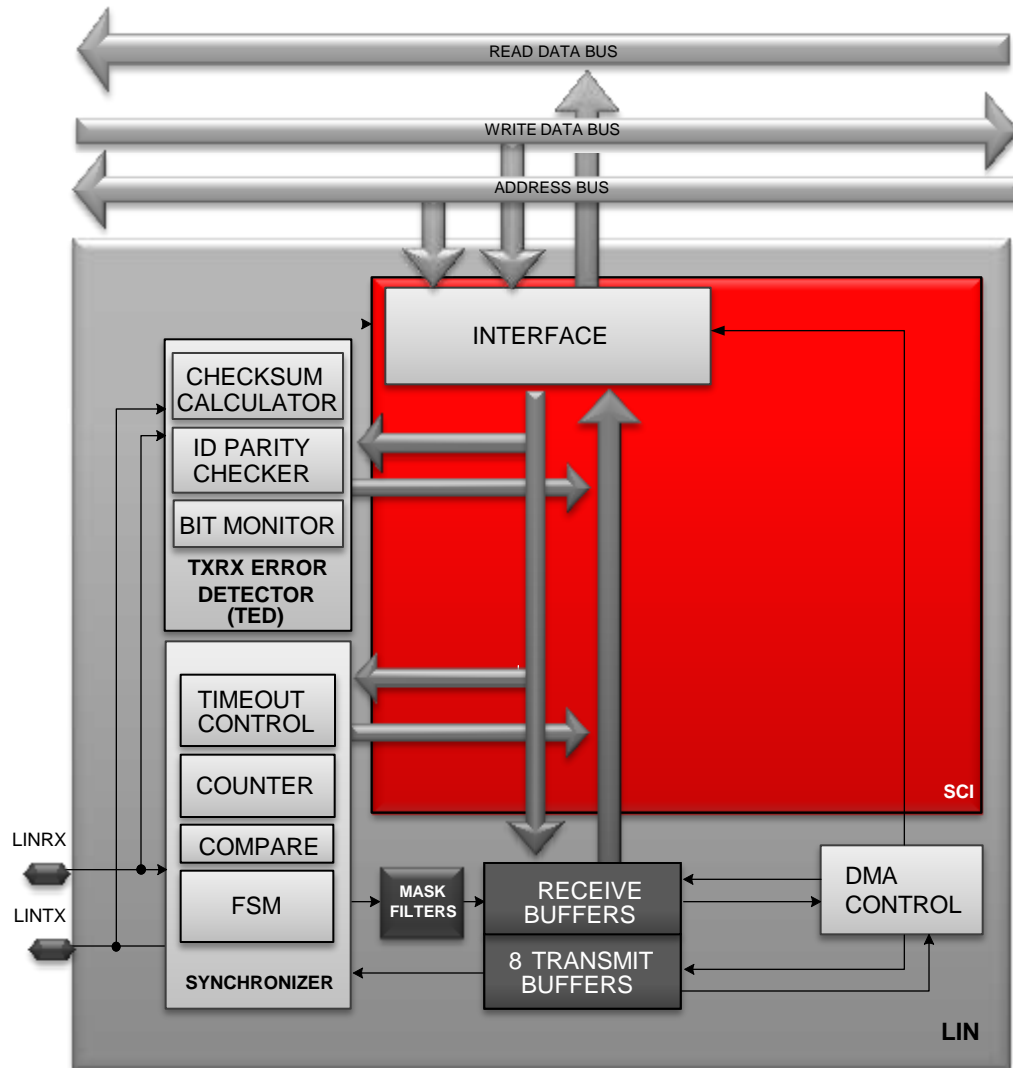
SCI Frame Format



Typical LIN Applications (TMS570)



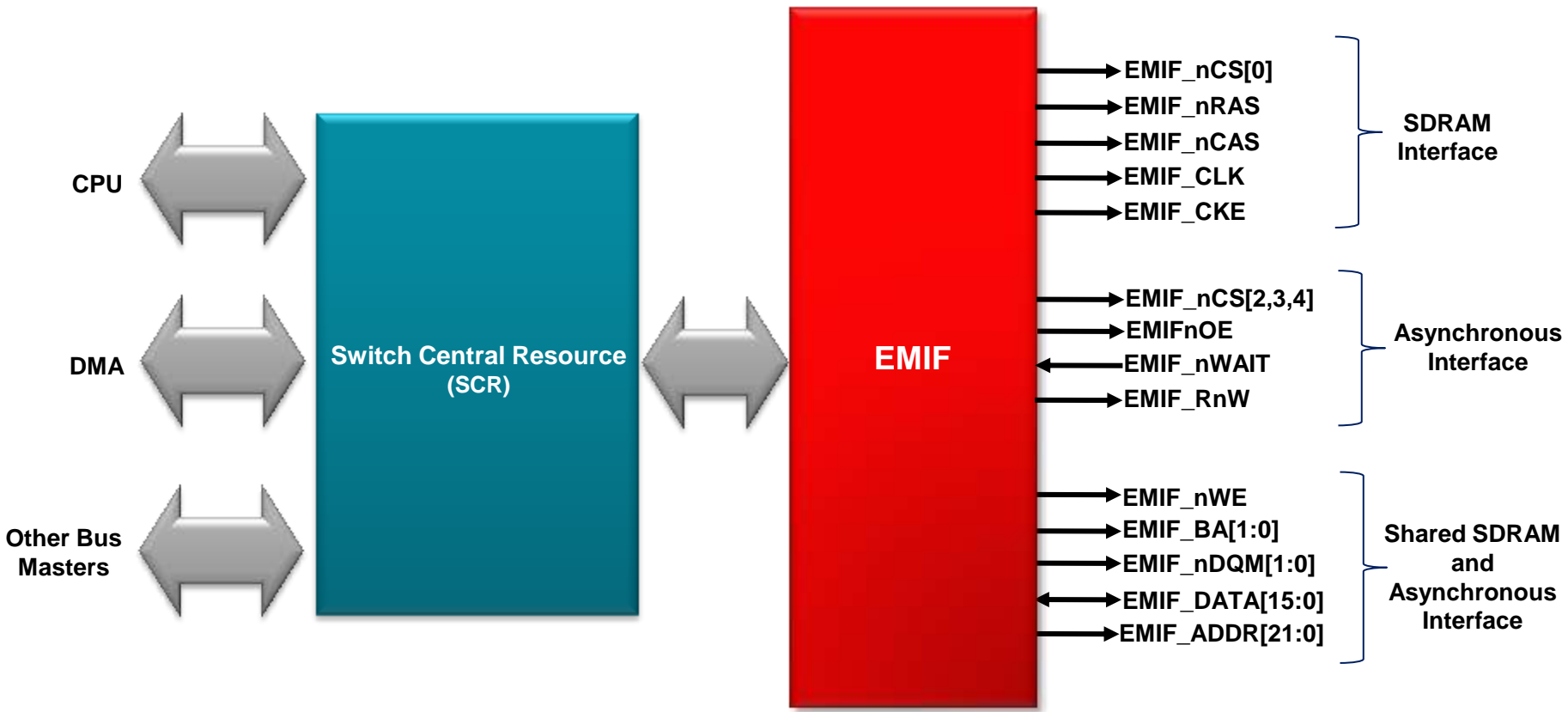
LIN Key Features



- Compatible with LIN 1.3 or 2.0
- LIN 2.0 Master Compliant
- HW LIN protocol handler
 - Multi-buffered receive and transmit units
 - Automatic checksum generation and validation
 - ID masks for message filtering
 - DMA capability
- Synch break detection
- Slave automatic synchronization
- Optional baud rate update
- Synchronization validation
- Automatic bit monitoring
- Automatic error detection

External Memory Interface (EMIF) / Parameter Overlay Module (POM)

EMIF: Block Diagram

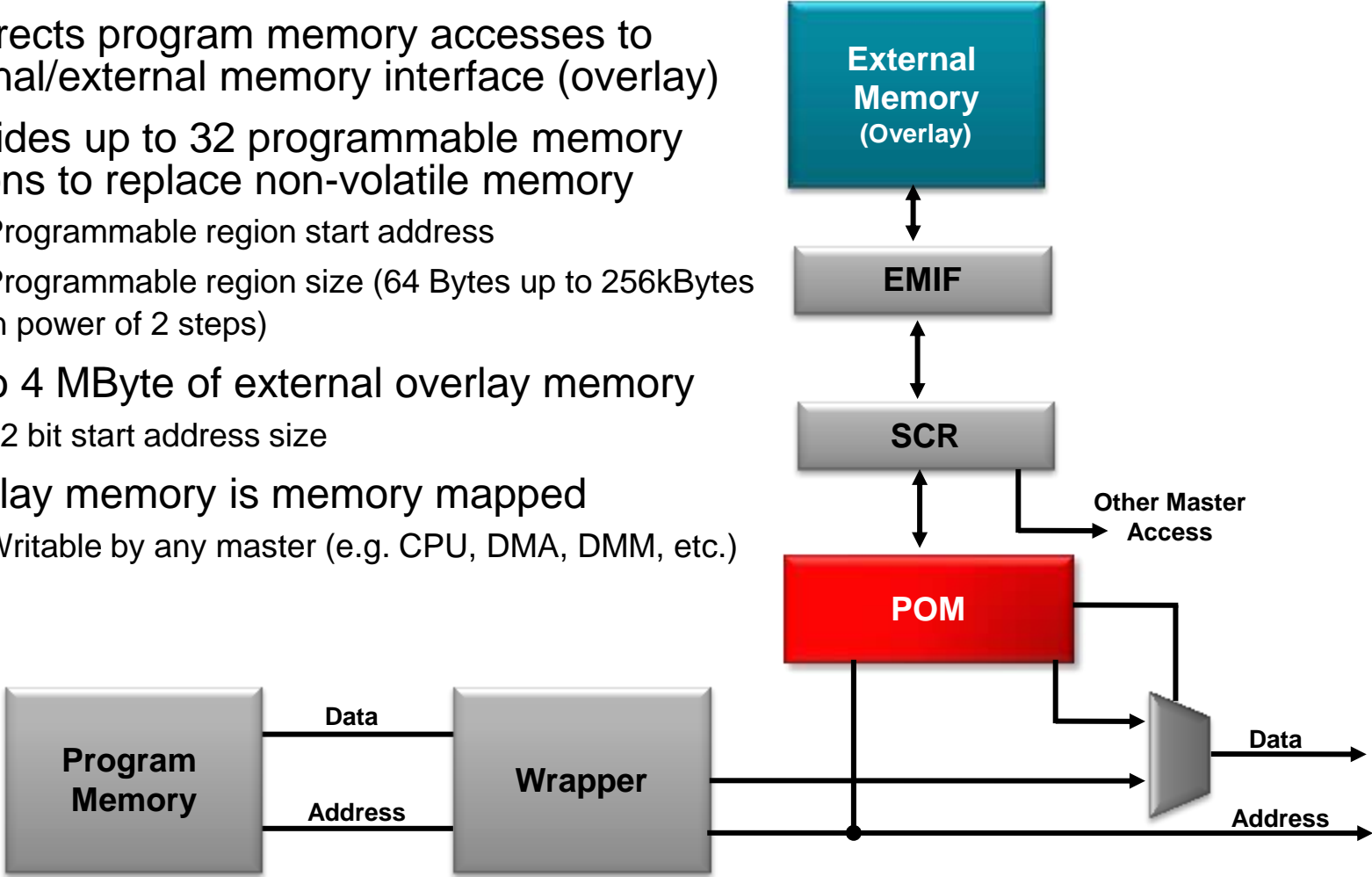


EMIF: Main Features

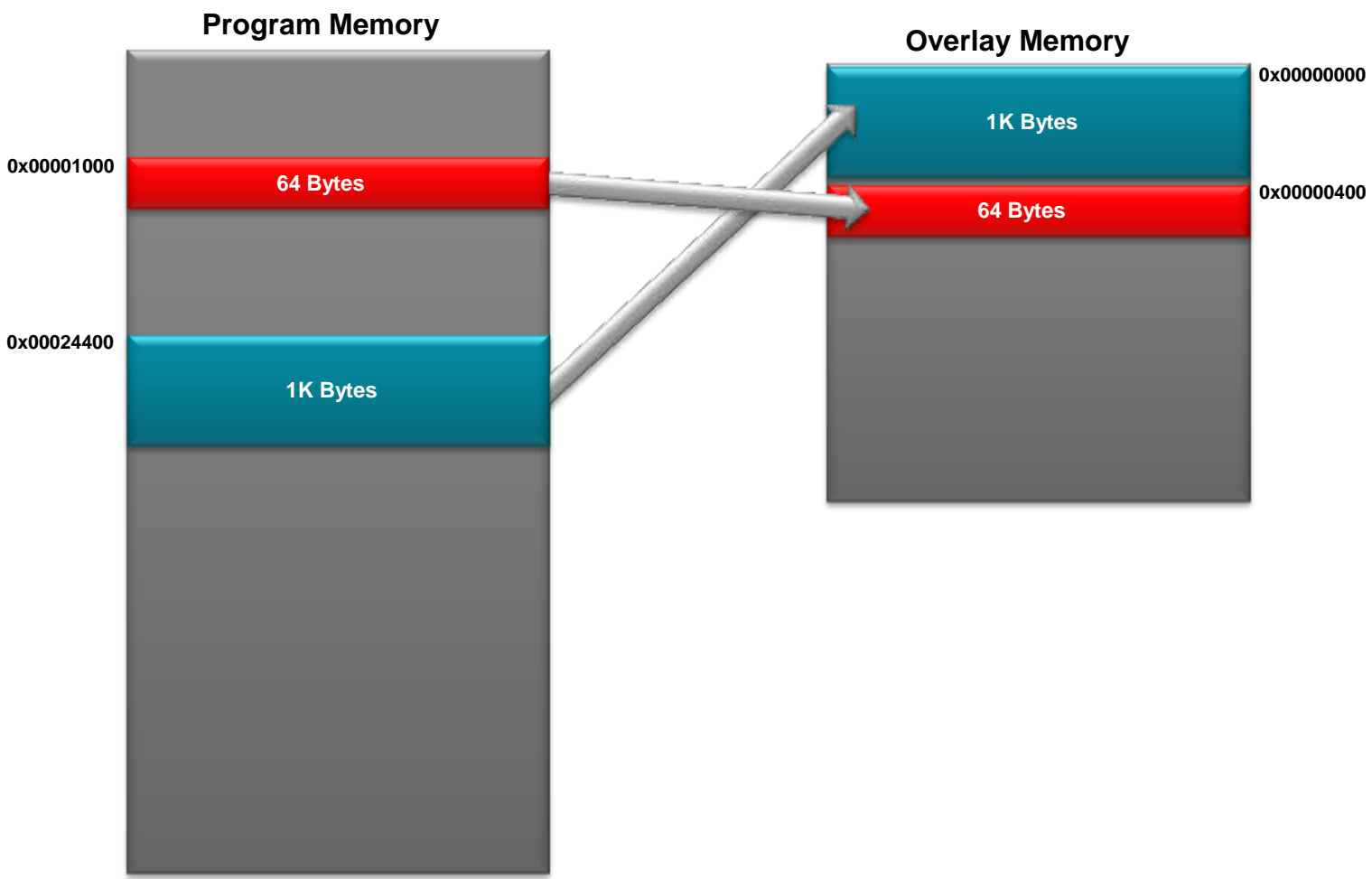
- Asynchronous Memory Support
 - Interfaces to SRAM memories as well as NOR Flash memories
 - 22 address lines, 3 chip selects of up to 16MB each
 - 16-bit data bus width
 - Programmable cycle timings
 - Select strobe mode option
 - Extended wait mode with programmable timeout period
 - Data bus parking
- Synchronous DRAM Memory Support
 - One, Two and Four Bank SDRAM devices
 - 22 address lines, 1 chip select
 - Devices with Eight, Nine, Ten, and Eleven Column Addresses
 - CAS latency of two or three clock cycles
 - 16-bit data bus width
 - 3.3V LVCMOS Interface
 - Support for SDRAM Self-Refresh and Powerdown modes

Parameter Overlay Module

- Redirects program memory accesses to internal/external memory interface (overlay)
- Provides up to 32 programmable memory regions to replace non-volatile memory
 - Programmable region start address
 - Programmable region size (64 Bytes up to 256kBytes in power of 2 steps)
- Up to 4 MByte of external overlay memory
 - 22 bit start address size
- Overlay memory is memory mapped
 - Writable by any master (e.g. CPU, DMA, DMM, etc.)

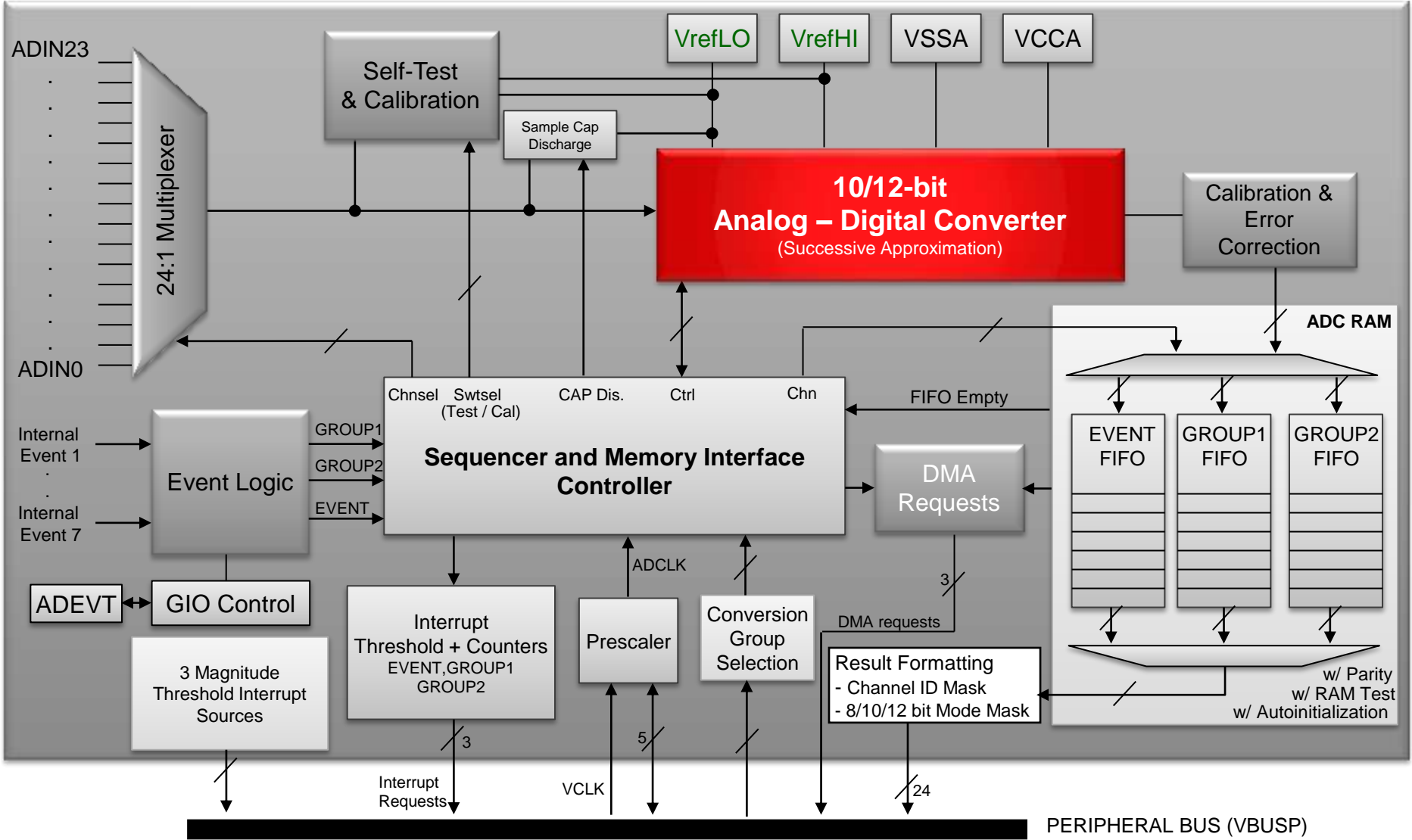


POM - Overlay Region Example



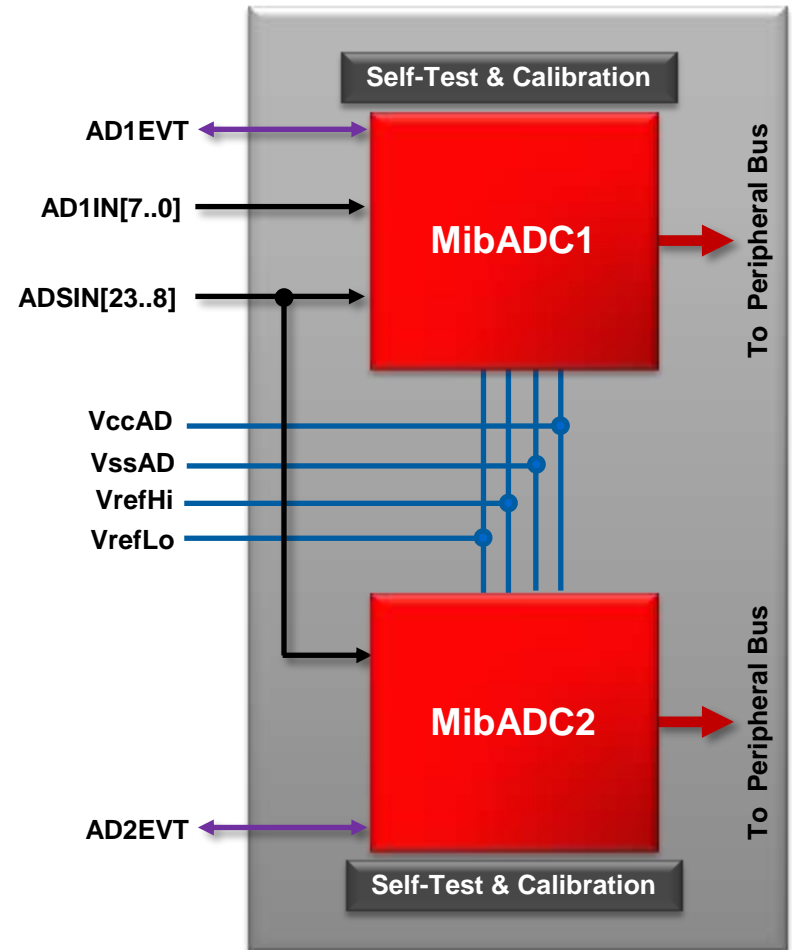
Multi-Buffered Analog to Digital Converter (MibADC)

MibADC Block Diagram



MibADC ADC Implementation

- Available Dual12-bit ADC cores:
 - MibADC1 (AD1IN + ADSIN = 24 ch)
MibADC2 (ADSIN = 16 ch)
 - 16 analog channels shared between the 2 cores for safety critical conversions/comparison
 - Internal ADC reference voltages can be used to check converter functionality
 - Self Test Mode enables application to detect opens/shorts on ADC inputs
 - ADC calibration logic can improve accuracy or be used to detect drift between multiple test results
 - offset error correction



Note: Not all Hercules MCUs are available with dual ADCs

MibADC Operation Modes

Conversion Mode

- Normal active mode for converting the selected external input voltage

Sample Capacitor Discharge Mode

- Active mode that grounds the ADC sampling capacitor

Calibration Mode

- Special active mode for calibration using internal reference voltages

Self-Test Mode

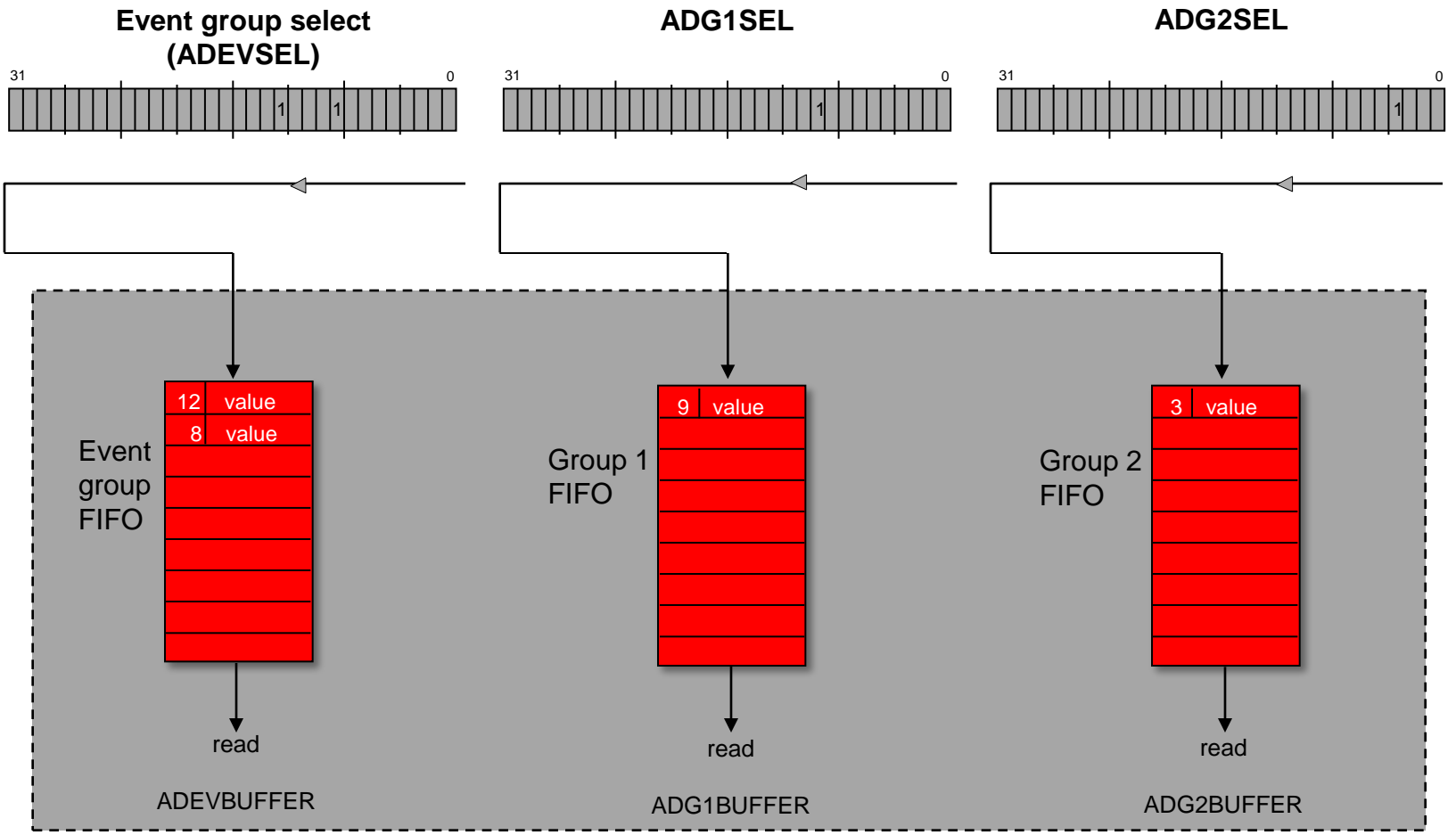
- Active mode for failure-detection using internal reference voltages

Power-Down Mode

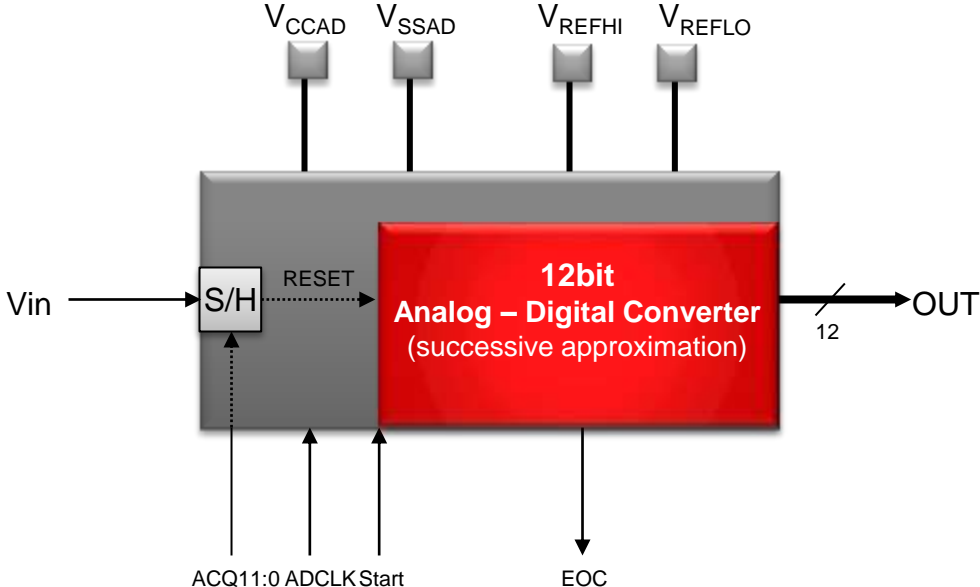
- Inactive mode in which the ADC internal clock is stopped

MibADC Conversion Groups

Input Channel Select Registers



MibADC Conversion Result



12 bit ADC: **$DIGITAL_RESULT = 4096 * (V_{in} - V_{REFLO}) / (V_{REFHI} - V_{REFLO})$**

MibADC Interrupts

Group Conversion End

- All channels that are assigned to a particular group are converted

Group Memory Threshold

- Number of conversion results exceed threshold register value

Group Memory Overrun

- Number of ADC conversions exceed the number of buffers allocated for that conversion group

Magnitude Threshold

- Magnitude comparison of conversion result on up to three channels. Programmable compare between two channels' conversion results or a channel's conversion result with a threshold value

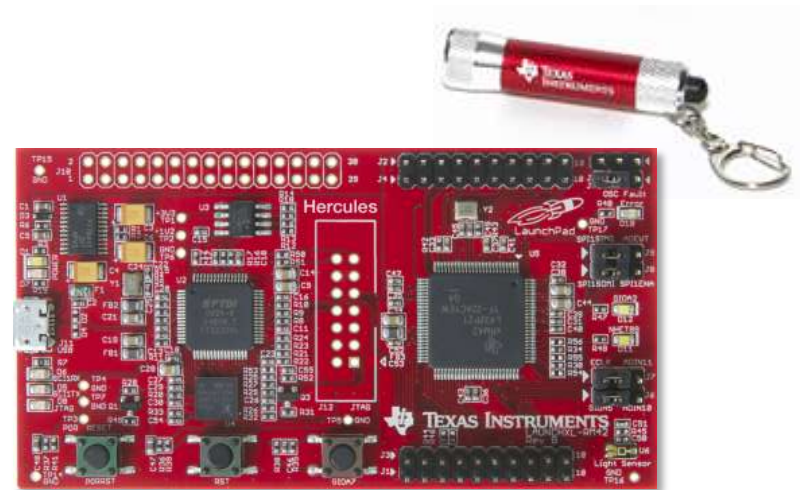
Parity Error

- On parity error the ADC module sends a parity error signal to the System module

Exercise:
Using the MibADC to collect Ambient Light Sensor data

ADC Exercise Overview

- In this exercise we will:
 - Acquire data from the ambient light sensor using the ADC module
 - Send the converted ADC value back through the SCI/UART module to the PC.



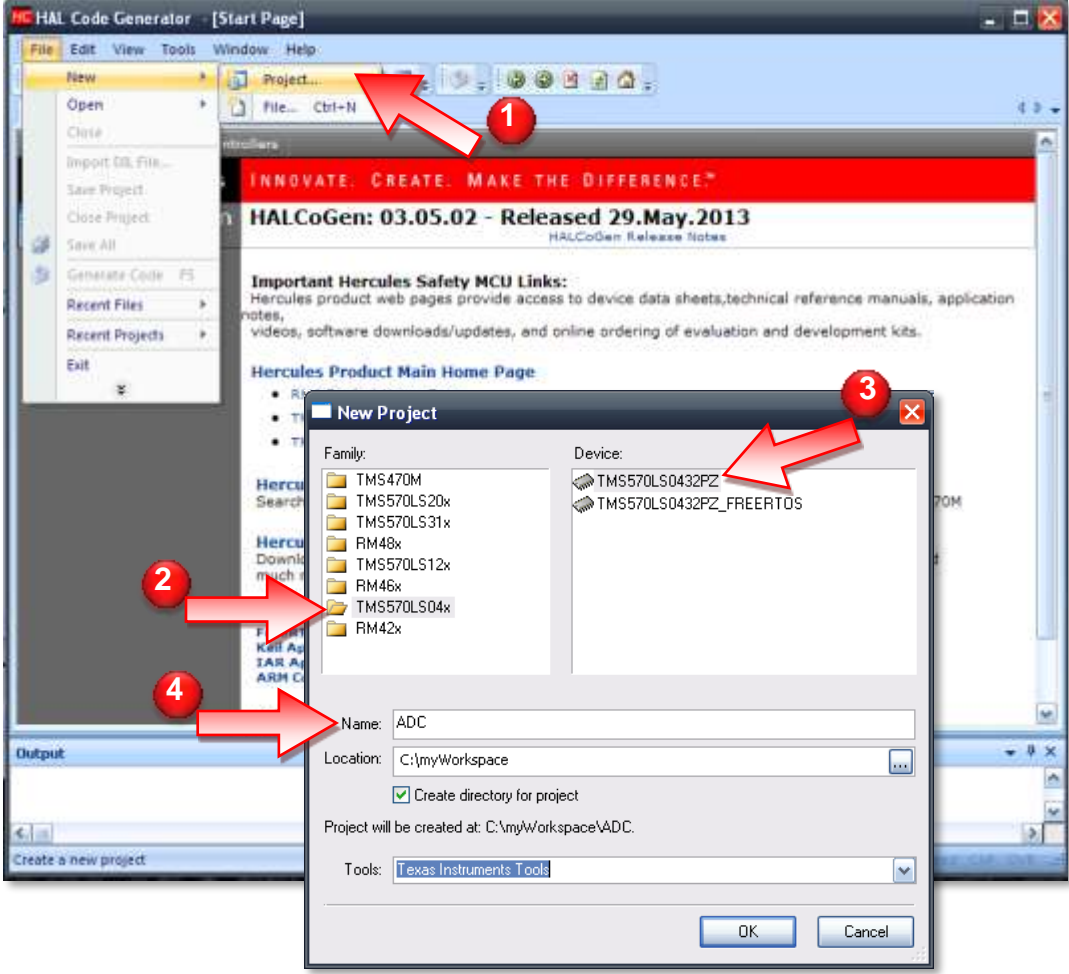
Set up a New HALCoGen Project

- To launch HALCoGen go to:



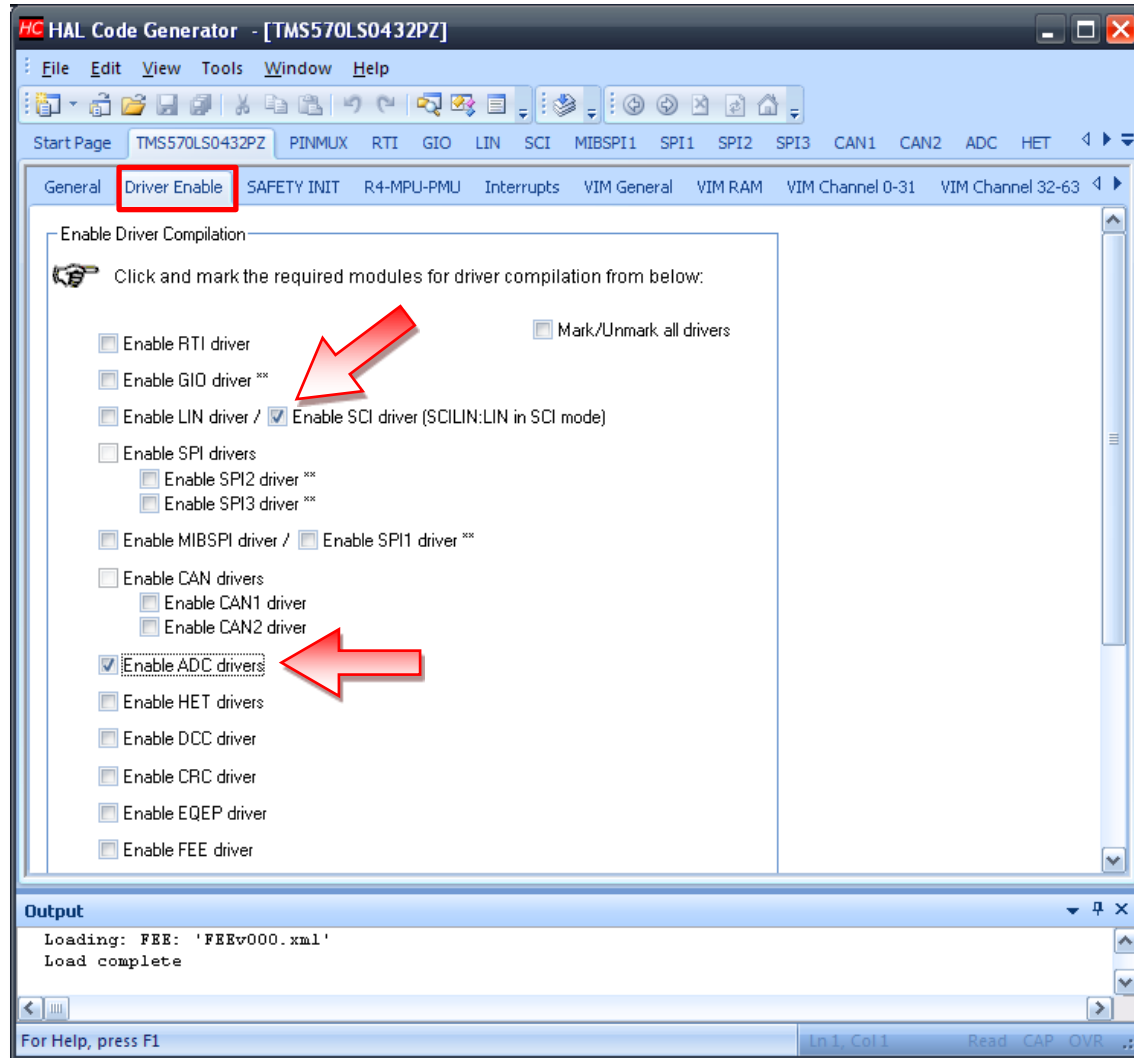
→ Programs → Texas Instruments → Hercules → HALCoGen

- Create a new project:
 - File → New → Project
- For the TMS570 Kit:**
 - Choose Family: TMS570LS04x
 - Device: TMS570LS0432PZ
- For the RM4 Kit:**
 - Choose Family: RM42x
 - Device: RM42L432PZ
- Then define a name: **'ADC'**
- Location: "C:\myWorkspace"



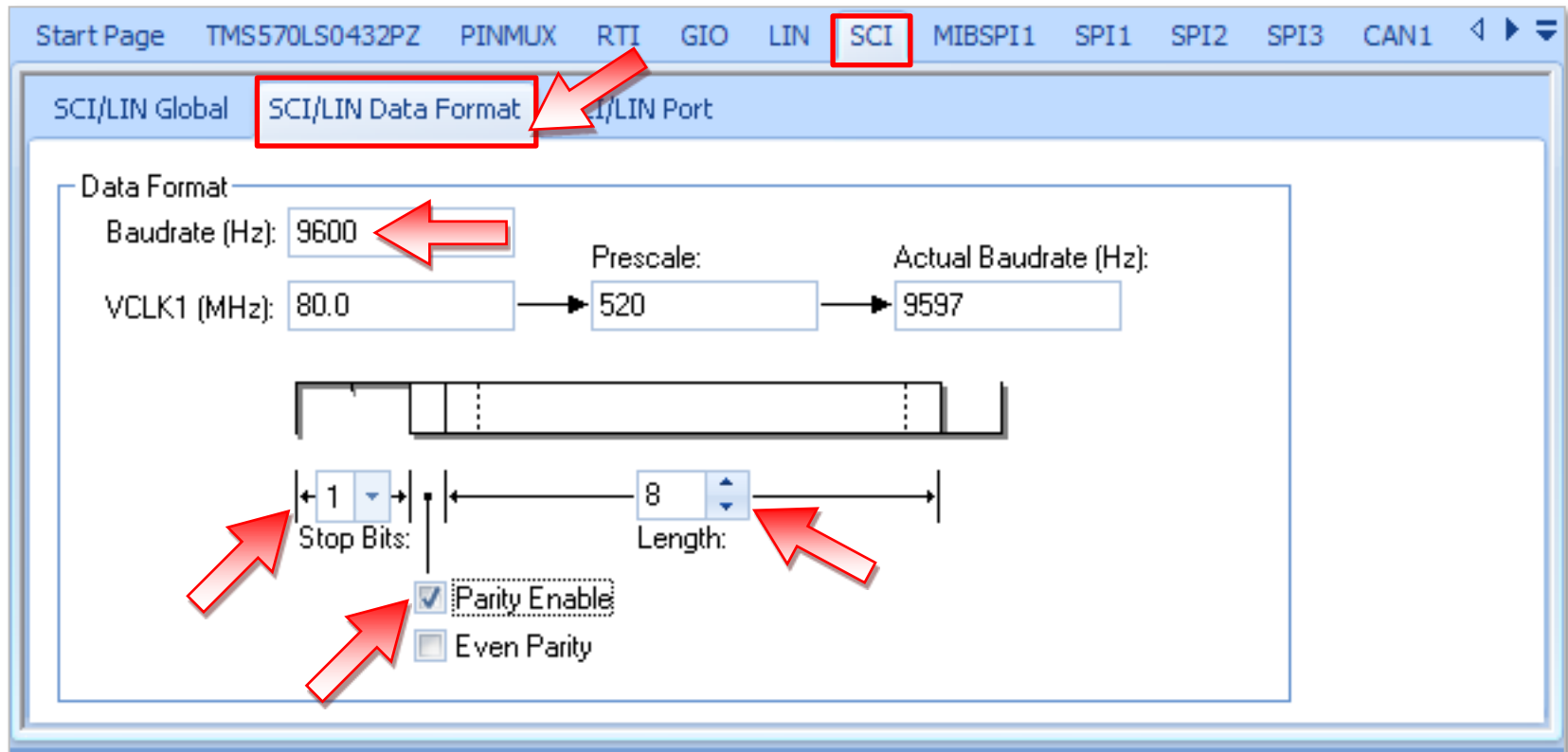
Driver Enable

- Enable the SCI and ADC drivers in the 'Driver Enable' sub tab of the device



SCI Configuration

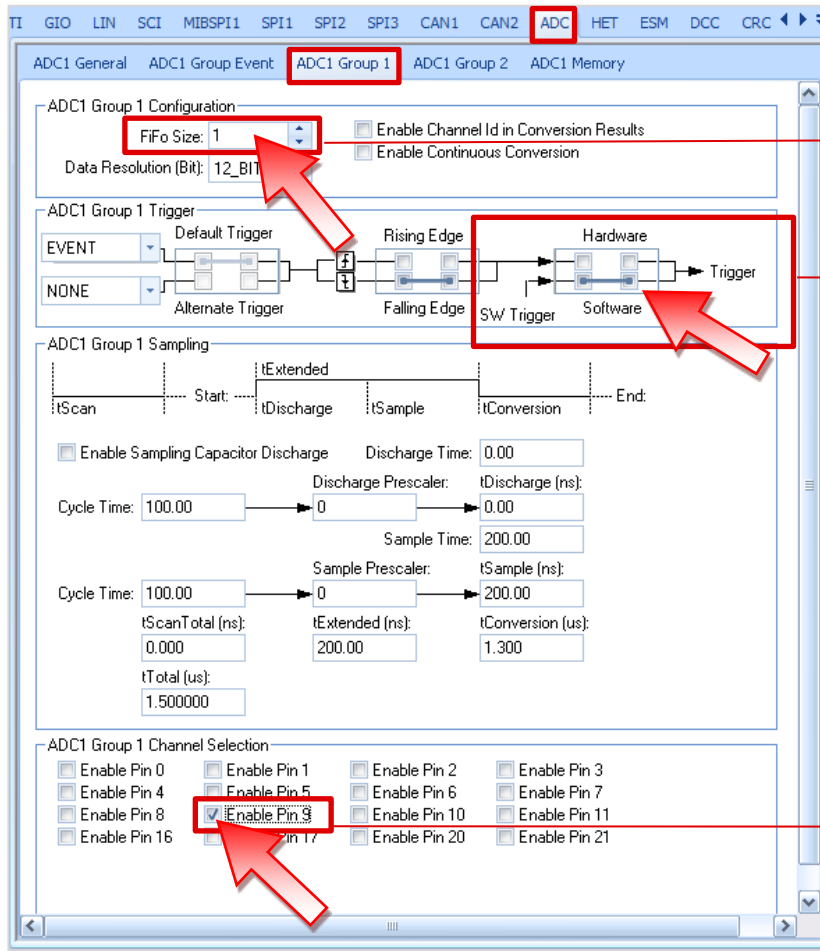
- Select the SCI tab and then the “SCI/LIN Data Format” subtab
- Ensure that the SCI module is setup with the following parameters:
 - Baud rate: 9600
 - Data bits: 8
 - Enable Parity (odd), 1 Stop bit



ADC Group Configuration



- In the 'ADC' tab and 'ADC1 Group 1' Subtab, Configure FIFO Size, Trigger Source and ADC channel as shown below



Only one conversion per software trigger is done

SW Trigger as Trigger Source

Ambient light sensor is connected to MibADC1 Pin9

- Generate code: File → Generate Code or click



MibADC Exercise



- Insert the following code in the corresponding sections within the 'sys_main.c' file
 - **USER CODE BEGIN (0) - #include header section**

```
/* USER CODE BEGIN (0) */  
#include "sci.h"  
#include "adc.h"  
#include "stdlib.h"  
unsigned char command[8];  
/* USER CODE END */
```

MibADC Exercise



- USER CODE BEGIN(3) - Main() section

```
void main(void)
{
    /* USER CODE BEGIN (3) */
    adcData_t adc_data; //ADC Data Structure
    adcData_t *adc_data_ptr = &adc_data; //ADC Data Pointer
    unsigned int NumberOfChars, value; //Declare variables

    sciInit(); //Initializes the SCI (UART) module
    adcInit(); //Initializes the ADC module

    while(1) // Loop to acquire and send ADC sample data via the SCI (UART)
    {
        adcStartConversion(adcREG1, 1U); //Start ADC conversion
        while(!adcIsConversionComplete(adcREG1, 1U)); //Wait for ADC conversion
        adcGetData(adcREG1, 1U, adc_data_ptr); //Store conversion into ADC pointer
        value = (unsigned int)adc_data_ptr->value;
        NumberOfChars = ltoa(value, (char *)command);
        sciSend(scilinREG, 2, (unsigned char *)"0x"); //Sends '0x' hex designation chars
        sciSend(scilinREG, NumberOfChars, command); //Sends the ambient light sensor data
        sciSend(scilinREG, 2, (unsigned char *)"\r\n"); //Sends new line character
    }
    /* USER CODE END */
}
```

- Now build and load your program on the microcontroller

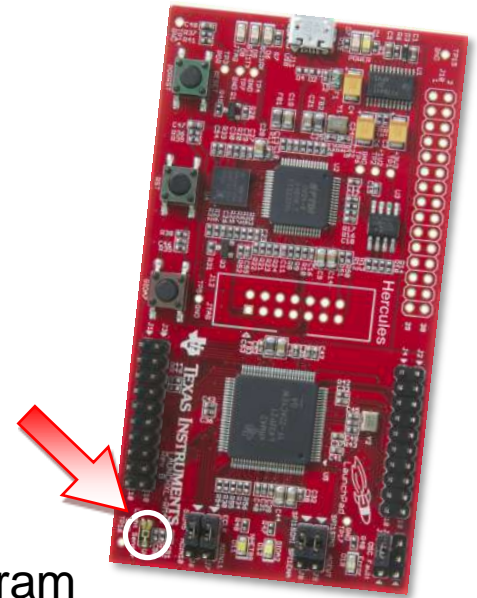
Testing your code



- Upon completion open your preferred terminal program.
 - Note: A terminal program is included in CCS. To enable it go to 'View' -> Other and select 'Terminal' from the 'Show View' menu. If the 'Terminal' option is not available it can be added as an eclipse plug-in by following these instructions:

http://processors.wiki.ti.com/index.php/How_to_install_the_terminal_plugin_in_CCSv5

- Setup the terminal program with the following properties:
 - Baud rate: 9600
 - Data bits: 8
 - Odd parity, 1 Stop bit
- Click the 'Run' button to run the program



Now you should see the ADC results in the terminal program

- Use a flashlight to change the light level supplied to the ambient light sensor on the board and notice the output values change in the terminal program.

Enabling the CCS Terminal

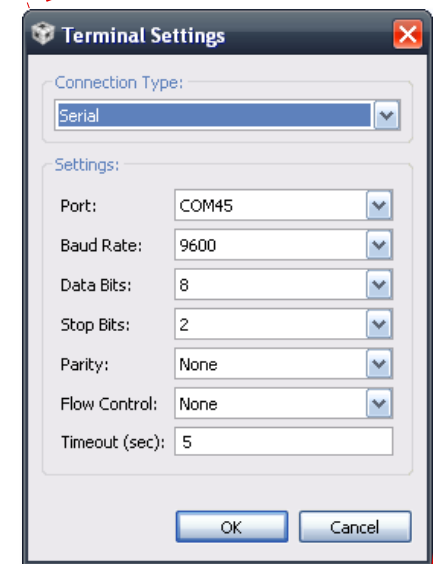
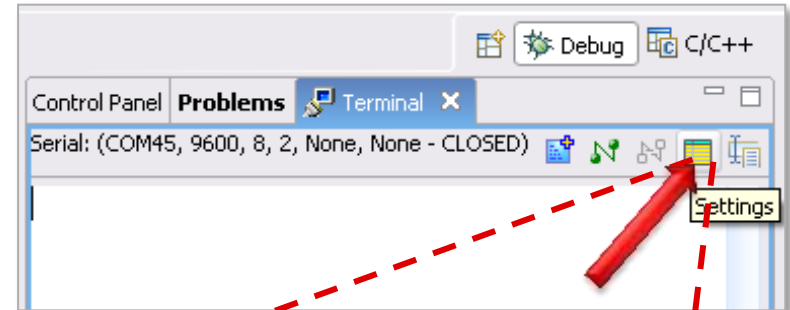
1) Select View → Other



2) Then select 'Terminal' from the 'Show View' menu.



3) Enter the proper communication settings for the 'Terminal Tab'



Additional Hercules™ Information

Additional Information

Hercules Web Page: www.ti.com/hercules

RM4 Web Page: www.ti.com/rm4

TMS570 Web Page: www.ti.com/tms570

TMS470M Web Page: www.ti.com/tms470m

- Data Sheets
- Technical Reference Manual
- Application Notes
- Software & Tools Downloads and Updates
- Order Evaluation and Development Kits

Engineer 2 Engineer Support Forum:

www.ti.com/hercules-support

- News and Announcements
- Useful Links
- Ask Technical Questions
- Search for Technical Content



Hercules WIKIs:

RM4 WIKI: www.ti.com/hercules-rm4-wiki

TMS570 WIKI: www.ti.com/hercules-tms570-wiki

TMS470M WIKI: www.ti.com/hercules-tms470m-wiki

- How to guides
- Intro Videos
- General Information





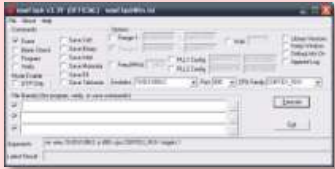
Want Additional Training?

Hercules™ Training Videos: www.ti.com/herculestraining

Safety Critical Design and Programming with Hercules™ Microcontrollers:

Day 1

- Welcome and Intro
- Hercules™ Product Overview / MCU Roadmap
- Safety Standards and Hercules Safety Features / Exercise
- HALCoGen / Exercise
- Code Composer Studio / Demonstration
- Compiler
- Flash Overview
- Flash Tools: nowFlash, nowECC / Exercise



Day 2

- Summary / Questions
- ARM® Cortex™ -R4F CPU
- System Module Overview
- Device setup/startup, Real Time Interrupt Module, Vectored Interrupt Manager
- CRC Controller, CPU Compare Module, Error Signaling Module, Dual Clock Compare, JTAG Security Module
- General Purpose I/Os / Exercise
- Direct Memory Access Controller (DMA)
- Serial Communication Interface (SCI/UART) / Exercise



Day 3

- Summary / Questions
- Multi-Buffer ADC (MIBADC) / Exercise
- Multi-Buffer Serial Peripheral Interface (SPI / MIBSPI-P)
- DCAN
- FlexRay
- External Memory Interface (EMIF) / Parameter Overlay Module (POM)
- Ethernet
- USB Host / Device
- HET (High End Timer) IDE
- N2HET & Transfer Unit / Exercise
- Summary / Questions / Survey



Thank You!

Please fill out the Training Class Survey

