Customer Training Workshop
Traveo™ II Watchdog Timer

Q4 2020
## Target Products

### Target product list for this training material

<table>
<thead>
<tr>
<th>Family Category</th>
<th>Series</th>
<th>Code Flash Memory Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traveo™ II Automotive Body Controller Entry</td>
<td>CYT2B6</td>
<td>Up to 576 KB</td>
</tr>
<tr>
<td>Traveo II Automotive Body Controller Entry</td>
<td>CYT2B7</td>
<td>Up to 1088 KB</td>
</tr>
<tr>
<td>Traveo II Automotive Body Controller Entry</td>
<td>CYT2B9</td>
<td>Up to 2112 KB</td>
</tr>
<tr>
<td>Traveo II Automotive Body Controller Entry</td>
<td>CYT2BL</td>
<td>Up to 4160 KB</td>
</tr>
<tr>
<td>Traveo II Automotive Body Controller High</td>
<td>CYT3BB/CYT4BB</td>
<td>Up to 4160 KB</td>
</tr>
<tr>
<td>Traveo II Automotive Body Controller High</td>
<td>CYT4BF</td>
<td>Up to 8384 KB</td>
</tr>
<tr>
<td>Traveo II Automotive Cluster</td>
<td>CYT3DL</td>
<td>Up to 4160 KB</td>
</tr>
<tr>
<td>Traveo II Automotive Cluster</td>
<td>CYT4DN</td>
<td>Up to 6336 KB</td>
</tr>
</tbody>
</table>
Introduction to Traveo II Body Controller Entry

Watchdog Timer (WDT) is part of System Resources

Review TRM chapter 20 for additional details
Introduction to Traveo II Body Controller High

Watchdog Timer (WDT) is part of System Resources

- CYT4BF
- MXS40-HT
- ASL-B

System Resources

- Power
  - Sleep Control
  - POR
  - RDO
  - GVP
  - LVD
  - PWRSYS-HT
  - LDO

- Clock
  - Clock Control
  - 2xPLL
  - IMO
  - ECO
  - 4xPLL

- Reset
  - Reset Control
  - XRES

- Test
  - TestMode Entry
  - Digital DFT
  - Analog DFT

- WE0
- RTC

- Power Modes
  - Active/Sleep
  - LowPower/Active/Sleep
  - DeepSleep
  - Hibernate

CPU Subsystem

- Arm Cortex-M7
  - 350 MHz
- eCT FLASH
  - 8384 KB Code flash
  - 256 KB Work flash
- SRAM0
  - 512 KB
- SRAM1
  - 256 KB
- SRAM2
  - 256 KB
- NVM0
- CRYPto
  - AES, SHA, CRC, TRNG, RSA, ECC
- MDMA
- DDR Controller
- Flash Controller

System Interconnect (Multi Layer AXI/AHB, IPC, MPU/SMPU)

Peripheral Interconnect (MMIO, PPU)

High-Speed I/O Matrix, Smart I/O, Boundary Scan

I/O Subsystem

- Up to 196x GPIO_STD, 4x GPIO_ENH, 40x HSIO

- 5x Smart I/O

- 10x SCB
- 10x CANFD
- 1x FLEXRAY
- 1x FlexRay Interface
- 1x EBYTE
- 1x RAY
- 1x EXP
- 2x ETH
- 10/100/1000 Ethernet + AVB
- 1x SMIF

- Serial Memory Interface (Hyperbus, Single SPI, Dual SPI, Quad SPI, Octal SPI)
- 2x ETH
- SDHC
- SD/SDIO/eMMC

- 2x ETH
- 1x SMIF
- 2x ETH
- 2x ETH

- 8384 KB Code flash
- 256 KB Work flash

- 350 MHz
- 196x GPIO_STD
- 4x GPIO_ENH
- 40x HSIO
Introduction to Traveo II Cluster

- Watchdog Timer (WDT) is part of System Resources

Hint Bar
Review TRM chapter 20 for additional details
Watchdog Timer (WDT) Overview

- A hardware timer that automatically resets the device in the event of an unexpected software execution path and provides warning interrupts and faults for multi-counter WDT (MCWDT)
- Two types of WDT
  - Basic WDT
  - MCWDT
- Features
  - Basic WDT: One 32-bit free-running counter
  - MCWDT: Two 16-bit counters and one 32-bit counter
  - Both WDT types support:
    - Window mode
    - Running and freezing timers during DeepSleep mode
    - Debug mode
    - Interrupt generation

Review TRM section 20.1 for additional details
Features
- 32-bit free-running counter supporting Window mode
- Device reset and interrupt
- Input clock source: ILO0
- Enabled after power-on
- Programmable LOWER LIMIT, WARN LIMIT, and UPPER LIMIT
- Registers that are protected and require an unlock sequence
- WARN LIMIT that generates an interrupt to request servicing
- Device reset that occurs when UPPER LIMIT is reached in the Window mode or when the counter is cleared before LOWER LIMIT
- Four power modes
  - Active
  - Sleep
  - DeepSleep
  - Hibernate

1 In Hibernate mode, any interrupt to wake up the device results in reset

Review TRM section 20.3.1 for additional details
Internal Low-speed Oscillator (ILO)
Refer to the Device Power Modes training section for additional power modes details
Basic WDT Operation

1. Counter is cleared between LOWER_LIMIT and WARN_LIMIT
   No WARN interrupt is issued and no RESET is done

2. Counter is cleared between WARN_LIMIT and UPPER_LIMIT
   A WARN interrupt is issued but no RESET is done

3. When the counter reaches the UPPER_LIMIT, a reset is executed

4. A reset is issued if the counter is cleared before the LOWER_LIMIT is reached

Use Case
- LOWER_LIMIT and UPPER_LIMIT are used to activate the Window function
- By using WARN_LIMIT, the interrupt can be used to debug before reset

Hint Bar
Review Register TRM and TRM section 20.3.2 for additional details
Multi-Counter WDT Block Diagram

Hint Bar

Review TRM section 20.4 for additional details

TVII-B-E devices support two units of MCWDT

TVII-B-H devices support three units of MCWDT

Depending on the device, up to four units are supported
Multi-Counter WDT Overview

› Features
  - Three independent counters
    - Two 16-bit counters that support:
      - Window mode
      - Interrupt mode with enabled/disabled AUTO_SERVICE
    - One 32-bit counter that supports:
      - Interrupt generation
      - Input clock source: LFCLK (ILO0, ILO1, ECO, and WCO)
  - Can operate in three power modes
    - Active
    - Sleep
    - DeepSleep
  - Disabled after power-on
  - Registers that are protected and require an unlock sequence
  - Device reset that occurs if the FAULT is not handled within two LFCLK cycles

Review TRM section 20.4 for additional details
Refer to the Clock System training section for additional ILO, ECO, and WCO details
Subcounter 0/1 Operation (1/5)

Subcounter 0 (MCWDTx_CNT0) and Subcounter 1 (MCWDTx_CNT1) are independent 16-bit up-counters.

1 Prefix “x” in MCWDTx represents the number of MCWDT. The number of MCWDT varies by device. It starts with 0.
Subcounter 0/1 Operation (2/5)

- MCWDT does not change limits for DeepSleep mode entry or exit automatically
- To get a behavior similar to DeepSleep, subcounters 0 and 1 can work together through SW configuration
- Use Case
  - Subcounter 0: Can be configured with a timeout threshold that protects running SW and configured to stop during DeepSleep
  - Subcounter 1: Can be configured with a longer timeout that continues to operate during DeepSleep

1 One MCWDT block can be associated to one CPU. This selects which CPU SLEEPDEEP signal is used for SLEEPDEEP_PAUSE

Review TRM section 20.4.5 for additional details
Subcounter 0/1 Operation (3/5)

Sequence of scenarios in Window mode

1. Counter is cleared between LOWER_LIMIT and WARN_LIMIT. No WARN interrupt and no RESET

2. Counter is cleared between WARN_LIMIT and UPPER_LIMIT. WARN interrupt is issued but no RESET

3. When the counter reaches the UPPER_LIMIT a FAULT is issued. A RESET can be issued in case the FAULT is not handled in time by the software

4. Counter is cleared before the LOWER_LIMIT is reached. A FAULT is issued. A RESET can be issued if the FAULT is not handled in time by the software.
Subcounter 0/1 Operation (4/5)

16-bit subcounters with WARN interrupt only (AUTO_SERVICE = 0) operate as follows:

1. Counter continues to increment after the counter value matches the WARN_LIMIT
2. Counter continues to count up until the 16-bit maximum value is reached
3. Counter overruns and restarts at zero

Review TRM section 20.4.2.1 for additional details
Subcounter 0/1 Operation (5/5)

- 16-bit subcounters with WARN interrupt only with enabled automatic service (AUTO_SERVICE = 1) operate as follows:

1. When the counter matches the WARN_LIMIT, an interrupt is issued.
2. The AUTO_SERVICE function clears the counter on any match event.

Hint Bar

Review TRM section 20.4.2.1 for additional details.
32-bit Counter Operation (1/2)

- **Subcounter 2** (MCWDTx_CNT2) is a 32-bit free-running counter that can be configured to generate an interrupt when one of the counter bits toggle.
- **Does not support Window mode**

Review TRM section 20.4.2.2 for additional details.

Multi-Counter Watchdog Timer

- MCWDTx_CNT0 (16-bit Counter)
  - Subcounter 0
  - Count
  - Count < LOWER_LIMIT
  - Count == WARN_LIMIT
  - Count >= UPPER_LIMIT

- MCWDTx_CNT1 (16-bit Counter)
  - Subcounter 1
  - Count
  - Count < LOWER_LIMIT
  - Count == WARN_LIMIT
  - Count >= UPPER_LIMIT

- MCWDTx_CNT2 (32-bit Counter)
  - Subcounter 2
  - Count
  - Count < LOWER_LIMIT
  - Count == WARN_LIMIT
  - Count >= UPPER_LIMIT

- MCWDTx_CTR2_CONFIG
  - Configuration
  - MCWDTx_CTR2_CONFIG_ACTION
  - MCWDTx_INTR_CTR2_INT

- Interrupt
- Fault
- Timeout
- Reset
When BIT = 4 is selected on the register, Subcounter 2 operates as follows:

1. Counter counts up and interrupt\(^1\) is generated on each toggle of the corresponding bit
2. Counter continues to count up until the 32-bit maximum value is reached
3. Counter overflows and restarts

\(^1\) The interrupt generated is a warning interrupt
Enabling and Disabling WDT

› WDT counters can be enabled and disabled by setting the registers
› Basic WDT
  – Enabled by setting the ENABLE[31] bit in the WDT_CTL register
  – Disabled by clearing the setting
› MCWDT counters
  – Enabled by setting the ENABLE[31] bit in the MCWDTx_CTL and MCWDTx_CTR2_CTL registers
  – Disabled by clearing the setting
› Advantage
  – Settings can be configured according to your system requirements

Hint Bar
Review Register TRM and chapter 20 for additional details
WDT Lock Feature

› Basic WDT
   - Locked by default
   - When the WDT_LOCK bits are not equal to ‘0’, write access to the following registers is prohibited:
     - CTL, LOWER_LIMIT, WARN_LIMIT, UPPER_LIMIT, CNT, and SERVICE

› MCWDT counters
   - Unlocked by default
   - When the MCWDT2_LOCK bits are not equal to ‘0’, write access to some registers is prohibited¹

› When registers are locked, software writes are always ignored

› Advantage
   - Protection through lock feature is for the functional safety of the system

¹ Refer to the Appendix for the registers list
Both Basic WDT and MCWDT support debug mode

Depending on DEBUG_TRIGGER_EN and DEBUG_RUN bit configuration, there are three options in debug mode:

- Counter is stopped when a debugger is connected
- Counter is stopped only when a debugger is connected and the CPU is halted during a breakpoint
- Counter is running when a debugger is connected. No reset is issued when the CPU is halted during a breakpoint but the counter is not stopped

Advantage

- Even if you break for debugging, you can debug with the same counter operation as the actual operation

Review Register TRM and TRM chapter 20 for additional details

It may take up to two clk_hf cycles for the counter to pause and another two cycles to resume, due to the internal synchronization
Reset Cause Detection

- Reset generated by the WDT counters are indicated by the bit in the RES_CAUSE register
  - Basic WDT
    - RESET_WDT [0] bit
  - MCWDT counters
    - RESET_MCWDT0 [5], RESET_MCWDT1 [6], RESET_MCWDT2 [7], and RESET_MCWDT3 [8] bits
- The bits remain set until cleared or until a power-on reset (POR), brownout reset (BOD), or external reset (XRES) occurs
## Register List - Basic WDT

<table>
<thead>
<tr>
<th>Lockable</th>
<th>Register</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>WDT_CTL</td>
<td>Watchdog Control Register</td>
<td>Control register for the Basic WDT</td>
</tr>
<tr>
<td>✔️</td>
<td>WDT_LOWER_LIMIT</td>
<td>WDT Lower Limit Register</td>
<td>Lower limit for the Basic WDT</td>
</tr>
<tr>
<td>✔️</td>
<td>WDT_UPPER_LIMIT</td>
<td>WDT Upper Limit Register</td>
<td>Upper limit for the Basic WDT</td>
</tr>
<tr>
<td>✔️</td>
<td>WDT_WARN_LIMIT</td>
<td>WDT Warn Limit Register</td>
<td>Warn limit for the Basic WDT</td>
</tr>
<tr>
<td>✔️</td>
<td>WDT_CONFIG</td>
<td>WDT Configuration Register</td>
<td>Configuration for the Basic WDT. Includes the ACTION configuration for UPPER, LOWER, and WARN limits, auto-servicing and pause settings in low-power and debug modes</td>
</tr>
<tr>
<td>✔️</td>
<td>WDT_CNT</td>
<td>WDT Count Register</td>
<td>Count value for the Basic WDT</td>
</tr>
<tr>
<td></td>
<td>WDT_LOCK</td>
<td>WDT Lock Register</td>
<td>Lock or unlock the Basic WDT registers</td>
</tr>
<tr>
<td>✔️</td>
<td>WDT_SERVICE</td>
<td>WDT Service Register</td>
<td>Clears the Basic WDT counter</td>
</tr>
<tr>
<td></td>
<td>WDT_INTR</td>
<td>WDT Interrupt Register</td>
<td>Interrupt signal from Basic WDT</td>
</tr>
<tr>
<td></td>
<td>WDT_INTR_SET</td>
<td>WDT Interrupt Set Register</td>
<td>Can be used to set interrupts for firmware testing</td>
</tr>
<tr>
<td></td>
<td>WDT_INTR_MASK</td>
<td>WDT Interrupt Mask Register</td>
<td>Controls whether interrupt is forwarded to CPU. All masks block the interrupt when 0 and forward the interrupt when 1</td>
</tr>
<tr>
<td></td>
<td>WDT_INTR_MASKED</td>
<td>WDT Interrupt Masked Register</td>
<td>Bitwise AND between the interrupt request and mask registers so firmware can read the status of all mask enabled interrupt causes with a single load operation</td>
</tr>
</tbody>
</table>
## Register List - MCWDT (1/2)

<table>
<thead>
<tr>
<th>Lockable</th>
<th>Register</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️</td>
<td>MCWDTx_CTRy_CTL</td>
<td>MCWDT Subcounter 0/1 Control Register</td>
<td>Control register for MCWDT subcounter</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTRy_LOWER_LIMIT</td>
<td>MCWDT Subcounter 0/1 Lower Limit Register</td>
<td>Lower limit for this MCWDT subcounter</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTRy_UPPER_LIMIT</td>
<td>MCWDT Subcounter 0/1 Upper Limit Register</td>
<td>Upper limit for this MCWDT subcounter</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTRy_WARN_LIMIT</td>
<td>MCWDT Subcounter 0/1 Warn Limit Register</td>
<td>Warn limit for this MCWDT subcounter</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTRy_CONFIG</td>
<td>MCWDT Subcounter 0/1 Configuration Register</td>
<td>Configuration for MCWDT subcounter. Includes the ACTION configuration for Upper, Lower, and Warn limits</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTRy_CNTy</td>
<td>MCWDT Subcounter 0/1 Count Register</td>
<td>Count value for this MCWDT subcounter</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTR2_CTL</td>
<td>MCWDT Subcounter 2 Control Register</td>
<td>Control register for MCWDT subcounter 2</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTR2_CONFIG</td>
<td>MCWDT Subcounter 2 Configuration Register</td>
<td>Configuration for MCWDT subcounter 2</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_CTR2_CNT</td>
<td>MCWDT Subcounter 2 Count Register</td>
<td>Count value for this MCWDT subcounter 2</td>
</tr>
<tr>
<td></td>
<td>MCWDTx_LOCK</td>
<td>MCWDT Lock Register</td>
<td>Lock or unlock the respective configuration registers of subcounters 0/1/2 of this MCWDT</td>
</tr>
<tr>
<td>✔️</td>
<td>MCWDTx_SERVICE</td>
<td>MCWDT Service Register</td>
<td>Includes service bits to clear subcounter 0/1 of this MCWDT</td>
</tr>
</tbody>
</table>
# Register List - MCWDT (2/2)

<table>
<thead>
<tr>
<th>Lockable</th>
<th>Register</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MCWDTx_INTR</td>
<td>MCWDT Interrupt Register</td>
<td>Interrupt status register for subcounters 0/1/2 for this MCWDT</td>
</tr>
<tr>
<td></td>
<td>MCWDTx_INTR_SET</td>
<td>MCWDT Interrupt Set Register</td>
<td>This register can be used to trigger an interrupt for firmware testing</td>
</tr>
<tr>
<td></td>
<td>MCWDTx_INTR_MASK</td>
<td>MCWDT Interrupt Mask Register</td>
<td>This register controls whether a subcounter interrupt is forwarded to the corresponding processor. All masks block the interrupt when 0 and forward the interrupt when 1</td>
</tr>
<tr>
<td></td>
<td>MCWDTx_INTR_MASKED</td>
<td>MCWDT Interrupt Masked Register</td>
<td>Bitwise AND between the interrupt request and mask registers so firmware can read the status of all mask enabled interrupt causes with a single load operation</td>
</tr>
<tr>
<td></td>
<td>CLK_SELECT</td>
<td>Clock Selection Register</td>
<td>Clock source selection register</td>
</tr>
<tr>
<td></td>
<td>CLK_ILO0_CONFIG</td>
<td>ILO0 Configuration</td>
<td>ILO0 configuration</td>
</tr>
<tr>
<td></td>
<td>RES_CAUSE</td>
<td>Reset Cause Observation Register</td>
<td>Reset cause observation register</td>
</tr>
</tbody>
</table>
Part of your life. Part of tomorrow.
## Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>ECN</th>
<th>Submission Date</th>
<th>Description of Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>**</td>
<td>6162538</td>
<td>05/03/2018</td>
<td>Initial release</td>
</tr>
<tr>
<td>*A</td>
<td>6344087</td>
<td>10/11/2018</td>
<td>Added page 2, 4, 5, 23, and the note descriptions of all pages. Updated page 3, 6, 9, 10 and 22.</td>
</tr>
<tr>
<td>*C</td>
<td>7042488</td>
<td>12/11/2020</td>
<td>Updated page 2, 3, 4, 13, 18, 26 and 27.</td>
</tr>
</tbody>
</table>