Customer training workshop

TRAVEO™ T2G Body Entry CPU Subsystem (CPUSS)
## 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™ T2G Automotive Body Controller Entry</td>
<td>CYT2B6</td>
<td>Up to 576 KB</td>
</tr>
<tr>
<td>TRAVEO™ T2G Automotive Body Controller Entry</td>
<td>CYT2B7</td>
<td>Up to 1088 KB</td>
</tr>
<tr>
<td>TRAVEO™ T2G Automotive Body Controller Entry</td>
<td>CYT2B9</td>
<td>Up to 2112 KB</td>
</tr>
<tr>
<td>TRAVEO™ T2G Automotive Body Controller Entry</td>
<td>CYT2BL</td>
<td>Up to 4160 KB</td>
</tr>
</tbody>
</table>
CPU subsystem overview

The CPU subsystem (CPUSS) is based on dual 32-bit Arm® Cortex® CPUs

**CPU subsystem components**

- **Executes application software as the Main CPU**
- **Implements security, safety, and protection features as the secondary CPU**
- **System RAM** mainly stores data
- **Data transfer functions such as peripherals to memory, memory to peripherals, and memory to memory**
- **System Interconnect (Multi Layer AHB, IPC, MPU/SMPU)**
  - **FLASH Controller**
  - **SRAM Controller**
  - **P-DMA**
  - **M-DMA**
  - **CRYPTO**
    - AES, SHA, CRC, TRNG, RSA, ECC
  - **SWJ/ETM/ITM/CTI**
  - **Arm Cortex M4**
    - 160 MHz
    - FPU, NVIC, MPU
  - **Arm Cortex M0+**
    - 100 MHz
    - MUL, NVIC, MPU
  - **ROM**
    - 32 KB
  - **Included Code/Work Flash** mainly stores programs and data
  - **eCT Flash**
    - 4160 KB Code-flash + 128 KB Work-flash
  - **SRAM0**
    - 256 KB
  - **SRAM1**
    - 256 KB
  - **eCT Flash 4160 KB Code-flash + 128 KB Work-flash**
  - **8 KB $ 8 KB $**
  - **SRAM Controller**
  - **Flash Controller**
  - **Hardware cryptographic functions**
  - **Stores the boot code executed after reset**

Review TRM section 4.1 for additional details specific to the CPUSS.
Cortex®-M4 features summary

› Main CPU
  – Execution of application software
  – Up to 160-MHz operation (CYT2BL)

› System Tick (SysTick) timer

› Floating-point unit (FPU)
  – Single precision
  – Compliant with the ANSI/IEEE Std 754-2008
  IEEE Standard for Binary Floating-Point Arithmetic

› Memory protection unit (MPU)
  – Eight protection regions
  – Privileged/unprivileged, read/write attributes

› Nested vector interrupt controller (NVIC)
  – Eight external system interrupts, eight internal software interrupts, eight interrupt levels, and one non-maskable interrupt
  – Wakeup interrupt controller (WIC) support
  – Vector table relocation (VTOR)

› Debug components
  – Supports SWD and JTAG interface (SWJ)
  – Tracing components (ETM, ITM, ETB)
  – Cross-triggering components (CTI)

1 See the device datasheet for operation frequency of target product.

Arm® provides additional reference material on their webpage at: infocenter.arm.com

Training section references:
- Interrupts
- Program and Debug Interface
- Protection Units
- Combined SWD/JTAG interface (SWJ)
- Embedded trace macrocell (ETM)
- Instrumentation trace macrocell (ITM)
- Cross-triggering interface (CTI)
- Embedded trace buffer (ETB)
Cortex®-M0+ features summary

- **Secondary CPU**
  - Execution of boot process
  - Secure master in secure system to establish a root of trust
  - Up to 100-MHz operation (CYT2BL) ¹, ²
- **SysTick timer**
- **Memory protection unit (MPU)**
  - Eight protection regions
  - Privileged/unprivileged access attributes
- **Debug components**
  - Supports SWD and JTAG interface (SWJ)
  - Tracing components (MTB)
  - Cross-triggering components (CTI)

- **Nested vector interrupt controller (NVIC)**
  - Eight external system interrupts, eight internal software interrupts, four interrupt levels, and one non-maskable interrupt
  - Wakeup interrupt controller (WIC) support
  - Vector table relocation (VTOR)

---

1. For CM0 to operate at 160 MHz, CM0+ is required to operate at a frequency of 80 MHz
2. See the device datasheet for operation frequency of target product
CPUSS dedicated master identifier

- Each bus master has a dedicated master identifier. This master identifier is used for:
  - Bus arbitration
  - IPC lock acquire functionality
  - Identification of access violated bus master detected by MPU, SMPU, and PPU

<table>
<thead>
<tr>
<th>Master identifier¹</th>
<th>Bus master (CYT2BL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Cortex®-M0+</td>
</tr>
<tr>
<td>1</td>
<td>Crypto</td>
</tr>
<tr>
<td>2</td>
<td>P-DMA 0</td>
</tr>
<tr>
<td>3</td>
<td>P-DMA 1</td>
</tr>
<tr>
<td>4</td>
<td>M-DMA</td>
</tr>
<tr>
<td>14</td>
<td>Cortex-M4</td>
</tr>
<tr>
<td>15</td>
<td>Test Controller</td>
</tr>
</tbody>
</table>

¹ See the device datasheet for master identifier number and regarding peripherals of target product
CPUSS bus infrastructure

- AHB-Lite bus infrastructure
- Two clock domains
  - Fast and slow clock domains
  - Each memory interface has both clock domains
  - Each domain has a dedicated bus infrastructure
- Bus competition
  - CPUSS has multiple bus masters
  - Design a system that considers bus competition by simultaneous access

Clock Domains

Fast Clock
- CM4 CPU
- CM0+ CPU
- Test Controller (TC)
- M-DMA
- Crypto
- P-DMA0
- P-DMA1
- System Interconnect (Fast Clock)
- Flash
- ROM
- SRAM

Slow Clock
- System Interconnect (Slow Clock)
- CM4 Peripheral Interface
- CM0+/TC Peripheral Interface
- P-DMA0 Peripheral Interface
- P-DMA1 Peripheral Interface

Hint Bar
Sys Interface in CM4 can only access peripherals
Code Interface in CM4 can only access memory controllers
Training section references:
- Clock system
Both CPUs support two operating modes and two privilege levels

- **Operating mode**
  - Thread mode executes application software and Handler mode handles exceptions

- **Privilege levels**
  - **Unprivileged**: Software has limited access to MSR/MRS instructions (uses CPS instructions), system timer, NVIC, system control block, and memory/peripherals
  - **Privileged**: Software can use all instructions and has access to all resources
  - Transition from Thread/Privileged to Thread/Unprivileged by CONTROL register
  - Transition from Thread/Unprivileged to Thread/Privileged via SVC

In Handler mode, software is always privileged. The thread privilege level after return from Handler mode depends on CONTROL.

Controlled by CONTROL register. CONTROL register can be written from privileged software only.

1 The CONTROL register is a CPU-specific register. It defines privileged/unprivileged, stack pointer, and FPU extension.
2 Supervisor calls are used to request privileged operations
Operation modes transition

Use case: Changing Privileged/Unprivileged mode

Handler/Privileged \[\rightarrow\] Thread/Privileged \[\rightarrow\] Thread/Unprivileged

Reset Release

Operation

Exception

Operation

Change CONTROL = Unprivileged

Operation

Change Privileged to Unprivileged

Change CONTROL = Privilege

Operation

SVC

Change Unprivileged to Privileged

Arm provides additional reference material on their webpage at: infocenter.arm.com
Memory protection

- Both CPUs have a memory protection unit (MPU)\(^1\) that:
  - Recognizes software separation
  - Includes address range, read/write, and privileged/unprivileged attributes
  - Features eight protection regions
- Use case
  - Software partitioning of ASIL and QM in functional safety

\(^1\) MPU registers are written by Privileged software and are only one part of the protection concept. For details, see the Protection Units training section.

Arm provides additional reference material on their webpage at: infocenter.arm.com
Vector table relocation

- Vector table offset register (VTOR)\(^1\)
  - Defines the location of the vector table of each core. There is a VTOR for CM4 and another one for CM0.
  - Can be used to relocate the vector table from Flash to SRAM, allowing the interrupt handlers to change dynamically
  - VTOR is written from Privileged software only
  - Boot ROM uses the sets the VTOR of CM0+, and CM0+ sets the VTOR of CM4:
    - CM0_VECTOR_TABLE_BASE: By default, set to beginning of Flash
    - CM4_VECTOR_TABLE_BASE: Set by CM0+ before releasing CM4 from reset
  - After boot, each core copies the vector table to SRAM and updates VTOR with the address of the new location

- Use cases
  - Execute the program with RAM only, for Flash programming or performance improvement
  - Use different vectors depending on the software level or system mode, such as normal and reprogramming

1 The VTOR register is a CPU-specific register.
System tick generation

› Both CPUs support a SysTick timer to measure time duration, which provides:
  – A 24-bit down counter
  – A selectable internal CPU clock or external clock
  – Active and Sleep mode operation
  – SysTick interrupt generation
› SysTick registers can be written from Privileged software only
› Use cases
  – RTOS tick timer
  – Alarm timer to alert when an action is not completed within a particular duration
  – Software completion time measurement

Arm provides additional supporting material on their webpage at: infocenter.arm.com
CPU subsystem memory feature summary

- **Flash**
  - Code and work flash
    - Code flash: Up to 4MB (CYT2BL)\(^1\)
    - Work flash: Up to 128KB (CYT2BL) \(^1\)
  - Error-correction code (ECC) function (SECDED)
  - Instruction cache for both CPUs
  - APIs for flash programming

- **SRAM**
  - Data storage and code execution
    - Up to 512KB (CYT2BL) \(^1\)
    - Dual CPU sharing
    - ECC function

- **ROM**
  - Boot code for both cores
  - API\(^2\) function implementation

---

**CPU subsystem components**

- Arm Cortex M4 160 MHz
  - FPU, NVIC, MPU
  - eCT Flash 4160 KB Code-flash + 128 KB Work-flash
  - 8 KB $ ^1 $ FLASH Controller
  - SRAM0 256 KB
    - SRAM Controller
    - 8 Channel P-DMA0
  - SRAM1 256 KB
    - SRAM Controller
    - 4 Channel P-DMA1
  - M-DMA0
  - M-DMA1
  - CRYPTO AES, SHA, CRC, TRNG, RSA, ECC
  - Initiator/MMIO
  - SWJ/ETM/ITM/CTI
  - SWJ/MTB/CTI
  - ROM 32 KB
    - Arm Cortex M0+ 100 MHz
      - MUL, NVIC, MPU
      - ROM Controller

---

1. See the device datasheet for memory size of target product.
2. No user access to read or modify SROM code.
P-DMA/M-DMA feature summary

› Peripheral DMA (P-DMA)
  – Single transfer engine shared for all channels
  – Focuses on low-latency transfer
  – Transfer modes:
    – Single, 1D, 2D, and CRC

› Memory DMA (M-DMA)
  – Dedicated transfer engine for each channel
  – Focuses on high-memory bandwidth
  – Transfer modes:
    – Single, 1D, 2D, Memory Copy, and Scatter
Cryptographic (Crypto) feature summary

- **Hardware crypto functions**
  - Symmetric key encryption and decryption
  - Hashing
  - Message authentication
  - Random number generation
  - Cyclic redundancy checking
  - Asymmetric key cryptography

1 Access limited to the secure master (CM0+)
Appendix
Comparison between families

<table>
<thead>
<tr>
<th>Features</th>
<th>Body Controller Entry (CYT2BL)</th>
<th>Body Controller High (CYT4BF)</th>
<th>Cluster (CYT4DN)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Main CPU</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPU</td>
<td>Cortex-M4 CPU</td>
<td>Two Cortex-M7 CPUs</td>
<td></td>
</tr>
<tr>
<td>Operating Frequency</td>
<td>Up to 160 MHz</td>
<td>Up to 350 MHz</td>
<td>Up to 320 MHz</td>
</tr>
<tr>
<td>FPU</td>
<td>Single-precision</td>
<td>Single/double-precision</td>
<td></td>
</tr>
<tr>
<td>Cache</td>
<td>N/A</td>
<td>16KB instruction, 16KB data</td>
<td></td>
</tr>
<tr>
<td>MPU</td>
<td>Cortex-M4: 8 regions</td>
<td>Cortex-M7: 16 regions</td>
<td></td>
</tr>
<tr>
<td>Interrupt Structure</td>
<td></td>
<td>NVIC+WIC</td>
<td></td>
</tr>
<tr>
<td>System Tick Timer</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Secondary CPU</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPU</td>
<td>Cortex-M0+ CPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Operating Frequency</td>
<td>Same (Up to 100 MHz)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MPU</td>
<td>Cortex-M0+: 8 regions</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt Structure</td>
<td></td>
<td>NVIC+WIC</td>
<td></td>
</tr>
<tr>
<td>System Tick Timer</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Flash</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bus Interface</td>
<td>AHB-Lite</td>
<td>AXI, AHB-Lite</td>
<td></td>
</tr>
<tr>
<td>ECC (SEC/DED)</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bank Modes</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Size (Code/Work)</td>
<td>4MB / 128KB</td>
<td>8MB / 256KB</td>
<td>6MB / 128KB</td>
</tr>
<tr>
<td><strong>SRAM</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bus Interface</td>
<td>AHB-Lite</td>
<td>AXI, AHB-Lite</td>
<td></td>
</tr>
<tr>
<td>ECC (SEC/DED)</td>
<td>Support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>TCM Size</td>
<td>N/A</td>
<td>16KB ITCM, 16KB DTCM</td>
<td>64KB ITCM, 64KB DTCM</td>
</tr>
<tr>
<td>SRAM Size</td>
<td>512KB</td>
<td>1024KB</td>
<td>640KB</td>
</tr>
<tr>
<td><strong>Boot</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Device Security with Crypto</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Direct Memory Access</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-DMA</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
<tr>
<td>M-DMA</td>
<td>Supported</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Part of your life. Part of tomorrow.
<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>6084432</td>
<td>03/12/2018</td>
<td>Initial release</td>
</tr>
<tr>
<td>*A</td>
<td>6390381</td>
<td>11/21/2018</td>
<td>Added slide 2, Updated slides 2-4, 6, 12-14. Deleted inside cover, Deleted slides 16,17,18</td>
</tr>
<tr>
<td>*B</td>
<td>7060645</td>
<td>01/06/2021</td>
<td>Updated slides 2-6, 13-15, 17.</td>
</tr>
<tr>
<td>*C</td>
<td>7450302</td>
<td>11/16/2021</td>
<td>Update slides 11</td>
</tr>
</tbody>
</table>