Customer training workshop
TRAVEO™ T2G Direct Memory Access (DMA)

Q4 2022

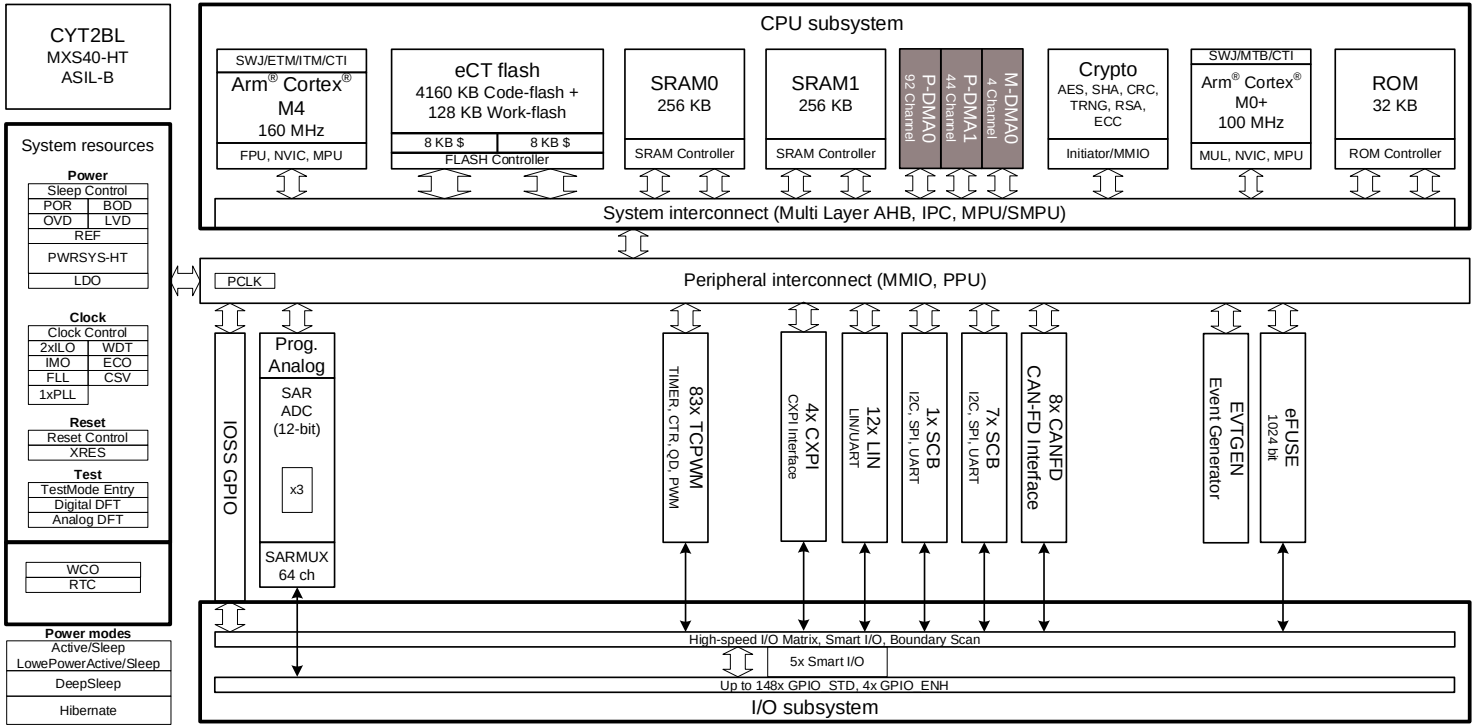
Target products

› Target product list for this training material

Family category	Series	Code flash memory size
TRAVEO™ T2G Automotive Body Controller Entry	CYT2B6	Up to 576KB
TRAVEO™ T2G Automotive Body Controller Entry	CYT2B7	Up to 1088KB
TRAVEO™ T2G Automotive Body Controller Entry	CYT2B9	Up to 2112KB
TRAVEO™ T2G Automotive Body Controller Entry	CYT2BL	Up to 4160KB
TRAVEO™ T2G Automotive Body Controller High	CYT3BB/CYT4BB	Up to 4160KB
TRAVEO™ T2G Automotive Body Controller High	CYT4BF	Up to 8384KB
TRAVEO™ T2G Automotive Cluster Entry	CYT2CL	Up to 4160 KB
TRAVEO™ T2G Automotive Cluster 2D	CYT3DL	Up to 4160KB
TRAVEO™ T2G Automotive Cluster 2D	CYT4DN	Up to 6336KB
TRAVEO™ T2G Automotive Cluster 2D	CYT4EN	Up to 6336 KB

Introduction to TRAVEO™ T2G Body Controller Entry

› DMA is part of the CPU subsystem

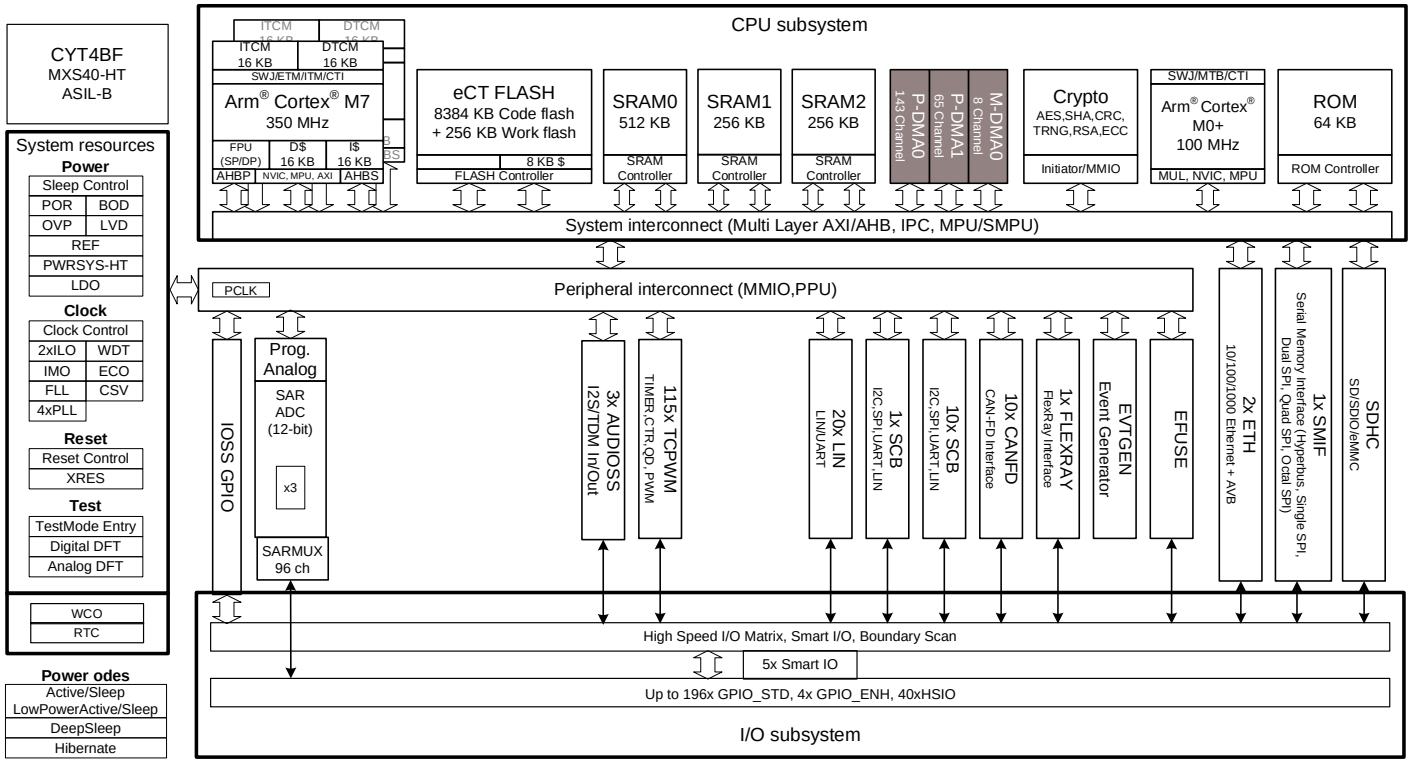


Hint Bar

Review TRM chapter 7 for additional details

Introduction to TRAVEO™ T2G Body Controller High

› DMA is part of the CPU subsystem

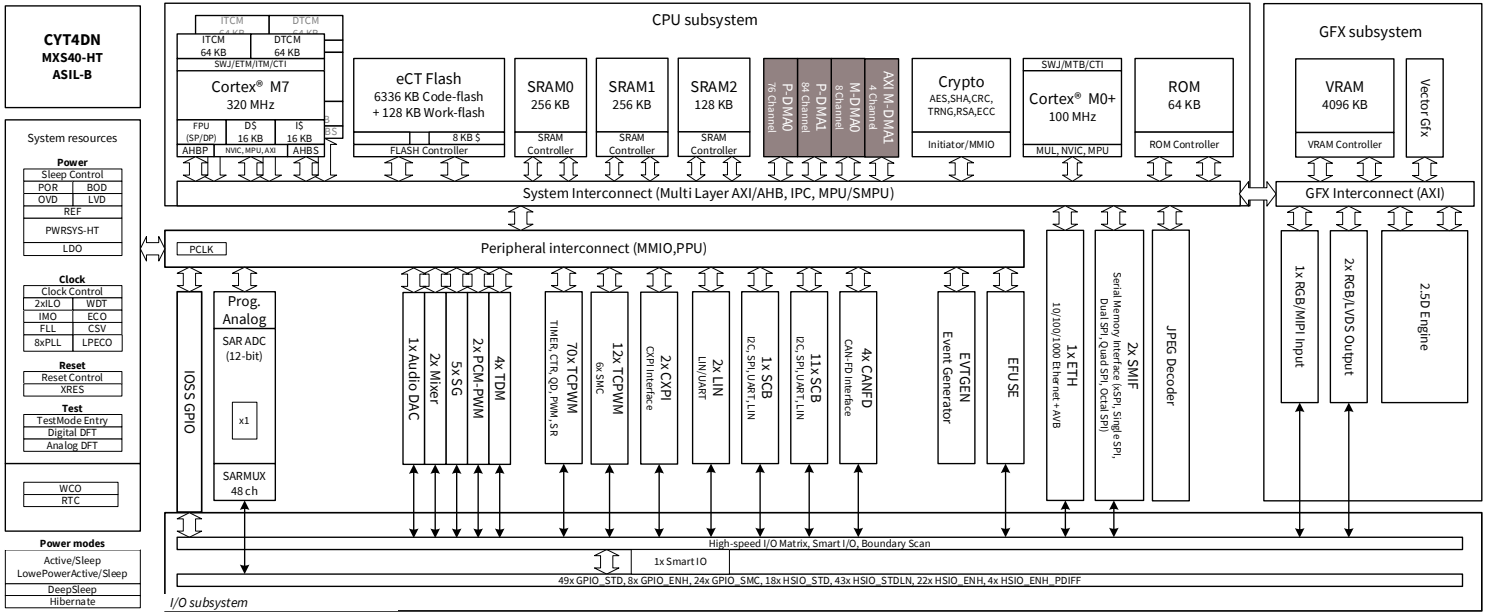


Hint Bar

Review TRM chapter 7 for additional details

Introduction to TRAVEO™ T2G Cluster

› DMA is part of the CPU subsystem



Hint Bar

Review TRM chapter 7 for additional details

P-DMA / M-DMA

DMA overview

- TRAVEO™ T2G has two types of DMA
  - Peripheral DMA (P-DMA)
  - Memory DMA (M-DMA)

Feature	P-DMA	M-DMA
Focus	Low latency	**High memory bandwidth**
Used for	Transfer between peripheral and memory[1]	**Transfer between memories[2]**
Transfer engine	Shared between all channels	**Dedicated for each channel**
Transfer size	8-bit/16-bit/32-bit	8-bit/16-bit/32-bit
Channel priority	Four levels Preemptable	Four levels
Transfer mode	- Single - 1D/2D - CRC transfer	- Single - 1D/2D **- Memory copy** **- Scatter**
Descriptor	- Source and destination address - Transfer size - Channel action - Data transfer mode - Activation trigger type (4 types) - Output trigger type (4 types) - Interrupt type (4 types) - Descriptor chaining	- Source and destination address - Transfer size - Channel action - Data transfer mode - Activation trigger type (4 types) - Output trigger type (4 types) - Interrupt type (4 types) - Descriptor chaining
Access-control attributes[3]	- Privileged/Unprivileged - Secure/Non-secure - Protection contexts	- Privileged/Unprivileged - Secure/Non-secure - Protection contexts

Hint Bar

Review TRM chapter 7 for additional details

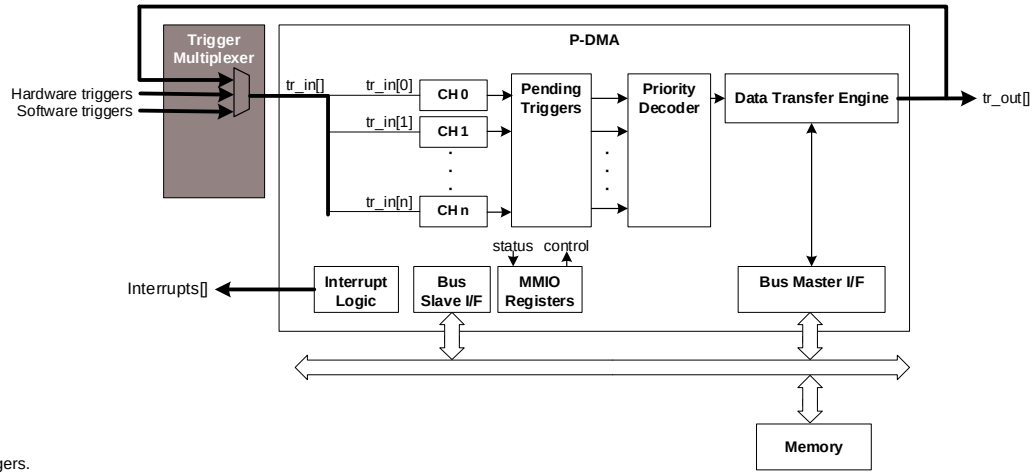
Features highlighted in bold are the main difference between P-DMA and M-DMA.

Refer to the Protection Units training section for additional details

[1] P-DMA can also be used to transfer data between memories.
[2] M-DMA can also be used to transfer data between peripheral and memory.
[3] P-DMA and M-DMA channels inherit the access attributes of the bus transfer that programmed the channel.

P-DMA block diagram

- Trigger multiplexers
  - Connect each channel to one specific system trigger
  - Include hardware[1] (HW) and software (SW) system triggers
  - Trigger output (tr_out)
    - Each trigger output can be used as its own active trigger using trigger multiplexers
    - They can be used to trigger different transfers as input triggers for other channels[2]



Hint Bar

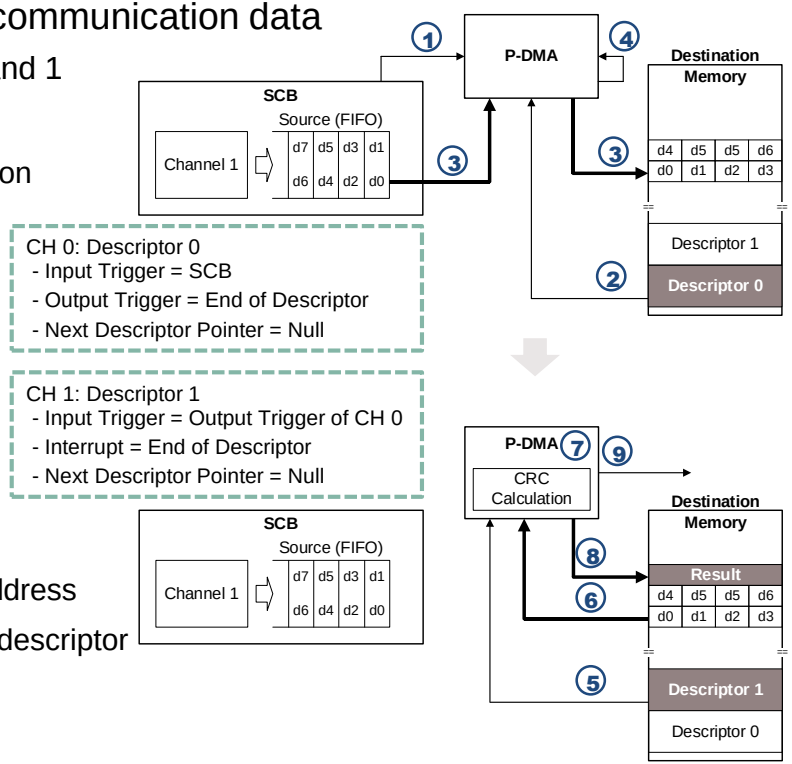
Review TRM section 7.1.6 and chapter 29 for additional details

Refer to Descriptor Chaining for Chain Transfer details

[1] Refer to the device datasheet for available hardware triggers.
[2] tr_out can execute a chain transfer. Descriptor chaining can also execute a chain transfer.

Trigger output (tr_out)

- Use case: CRC calculation of serial communication data
  - Combined operation of Descriptors 0 and 1
  - P-DMA:
    1. Is triggered by SCB when data reception is complete
    2. Loads Descriptor 0 from memory
    3. Transfers from FIFO to memory
    4. Outputs tr_out after completing Descriptor 0 transfer
    5. Loads Descriptor 1 by tr_out
    6. Activates CRC Transfer mode
    7. Runs CRC calculation
    8. Transfers CRC result to destination address
    9. Generates interrupt to CPU by end of descriptor



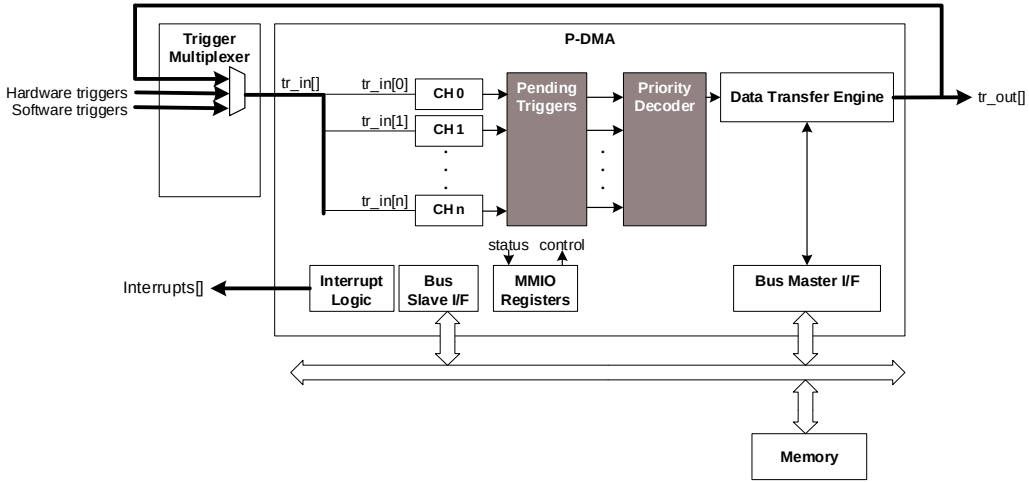
Hint Bar

Refer to CRC Transfer for CRC calculation

- Advantage
  - Multiple data transfers can be executed continuously by one trigger

P-DMA block diagram

- Pending triggers
  - Keeps track of activated triggers by storing channel triggers
  - Manages multiple pending channel triggers
- Available pre-emptive setting
  - If set, the higher-priority pending channel can pre-empt the current channel between single transfers
- Priority decoder
  - Determines the highest-priority channel of pending triggers
  - Has four priority levels
  - Performs round-robin arbitration within the same priority group

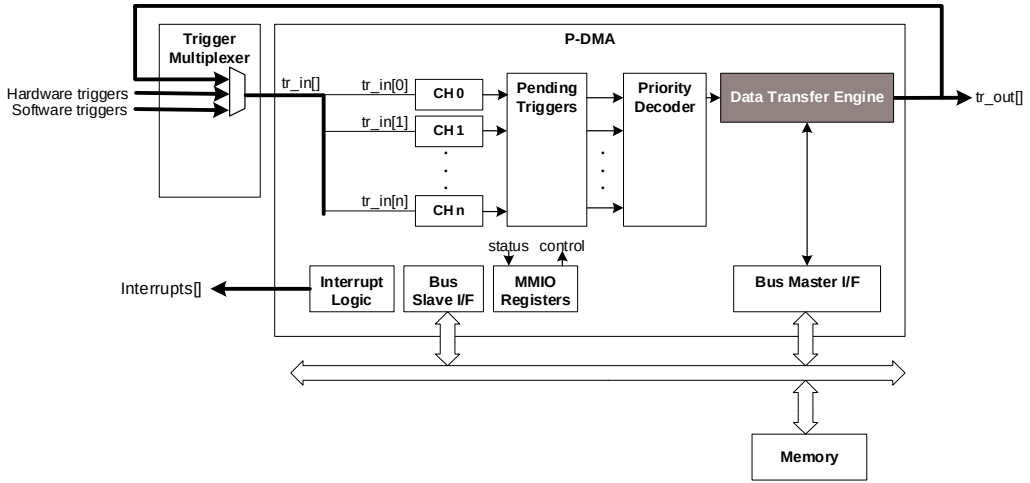


Hint Bar

Review TRM section 7.1.6 for additional details

P-DMA block diagram

- Data transfer engine
  - Shared by each channel
  - Transfers data from source to destination according to descriptor
  - Reads channel descriptor from memory
  - Generates the trigger out to trigger multiplexers

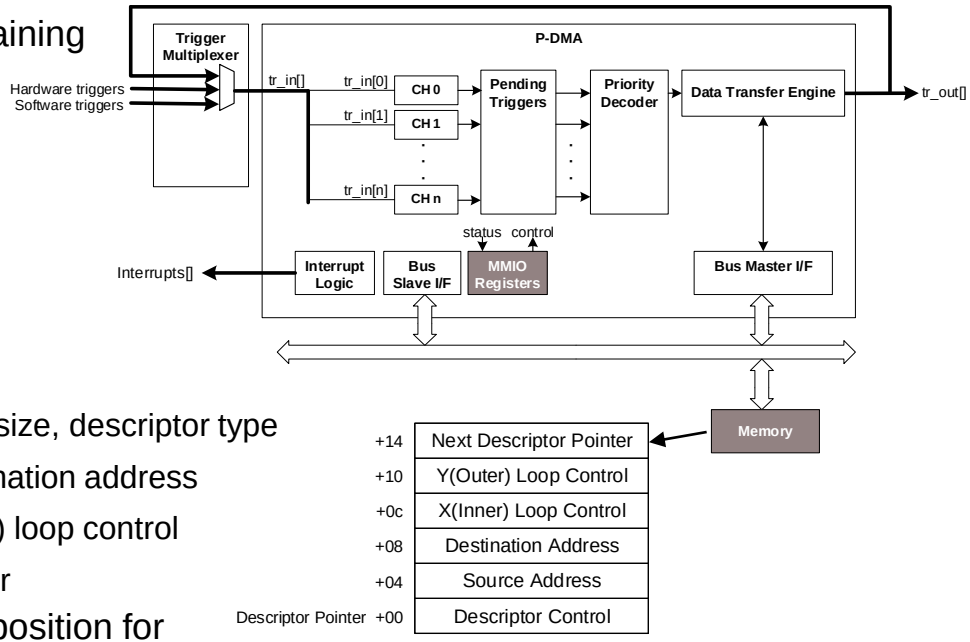


Hint Bar

Review TRM section 7.1.6 for additional details

P-DMA block diagram

- Descriptor
  - Stored in memory
  - Supports descriptor chaining
  - Descriptor types
    - Single transfer
    - 1D transfer
    - 2D transfer
    - CRC transfer
  - Descriptor structure
    - Descriptor control:
      Trigger type, transfer size, descriptor type
    - Source address/destination address
    - X(Inner) loop/Y(Outer) loop control
    - Next descriptor pointer
  - The descriptor pointer position for each channel is stored in the register



Hint Bar

Review TRM chapter 7.1.3 and Register TRM for additional details

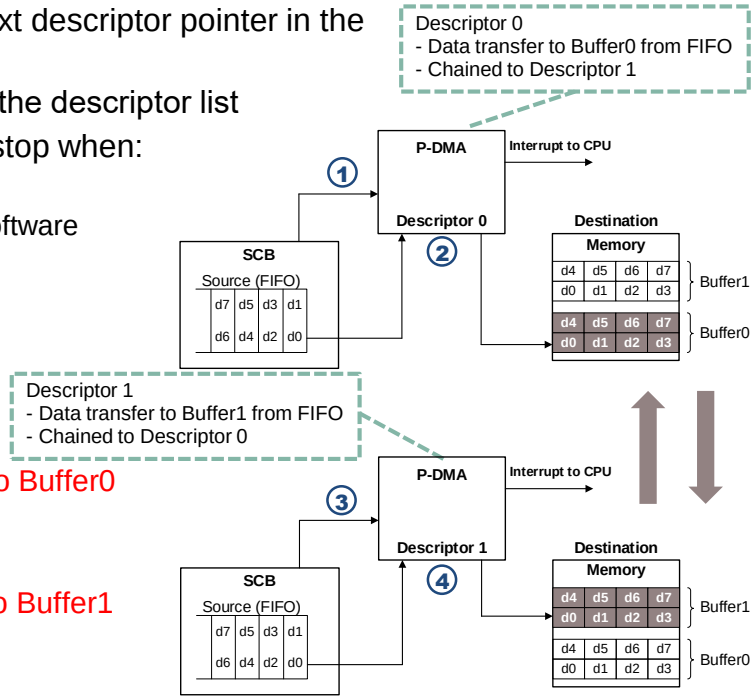
System RAM is used store the descriptor

Descriptor chaining

- Descriptor chaining
  - Descriptors are chained by storing the next descriptor pointer in the current descriptor
  - “0” (NULL pointer) is stored at the end of the descriptor list
  - It is possible to have a circular list. It will stop when:
    - A transfer error occurs
    - The controller or channel is disabled by software

- Use case: Double buffer storing using two descriptors
  - Descriptors 0 and 1 are chained together

1. P-DMA is triggered by SCB
2. Descriptor 0: Data transfer from SCB to Buffer0
   ⇒ Descriptor chaining to Descriptor 1
3. P-DMA is triggered by SCB
4. Descriptor 1: Data transfer from SCB to Buffer1
   ⇒ Descriptor chaining to Descriptor 0



Hint Bar

A descriptor chaining group is also referred to as a descriptor list

The address of the current descriptor is indicated in the CH_CURR_PTR register

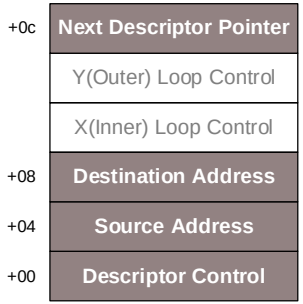
The next descriptor address is updated after the execution of the current descriptor

M-DMA also has the same register

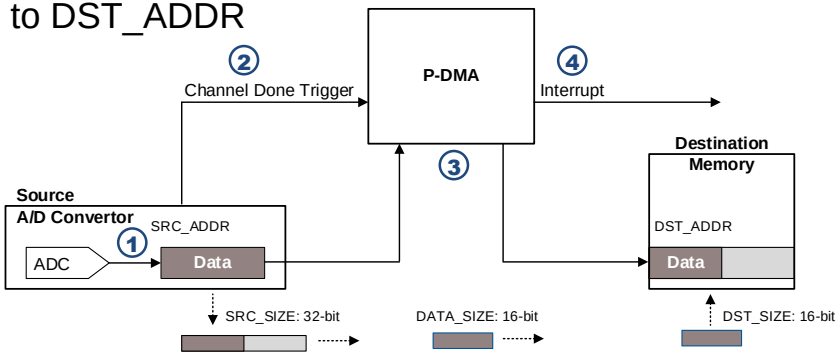
Descriptor type (1/4)

- Single transfer
  - Single data element (8-bit, 16-bit, or 32-bit) transfer
  - Uses four-words descriptor
  - Transfer example:

    DST_ADDR = (DATA_SIZE) SRC_ADDR



- Use case: Transfer A/D conversion results with A/D trigger
  1. Move stored conversion data to data register
  2. Activate P-DMA with Channel Done trigger
  3. Transfer data from SRC_ADDR to DST_ADDR
  4. Generate interrupt to CPU



Hint Bar

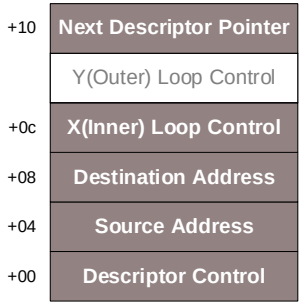
Review TRM chapter 7.1.3 and Register TRM for additional details

Descriptor type (2/4)

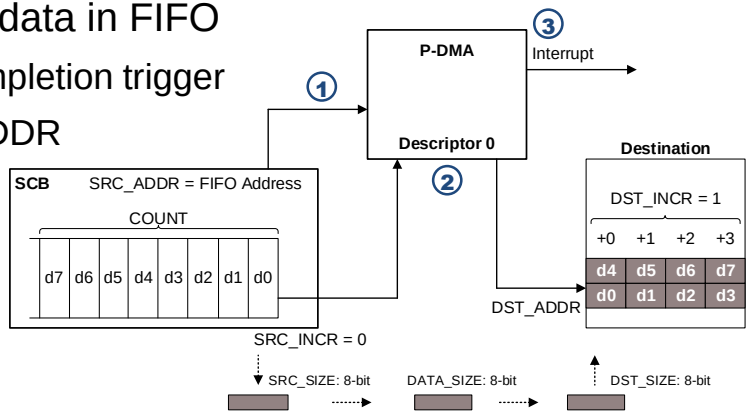
- 1D transfer
  - One-dimensional “for loop” transfer
  - Uses five-words descriptor
  - Transfer example:

```
for (X_IDX =0; X_IDX <= COUNT; X_IDX++){
   DST_ADDR[DST_INCR] = (DATA_SIZE) SRC_ADDR[SRC_INCR]
 }
```

*DST_INCR/SRC_INCR depend on X_INCR



- Use case: Transfer SCB reception data in FIFO
  1. Activate P-DMA by receiving a completion trigger
  2. Repeat data transfer from SRC_ADDR to DST_ADDR by COUNT
  3. Generate interrupt to CPU



Hint Bar

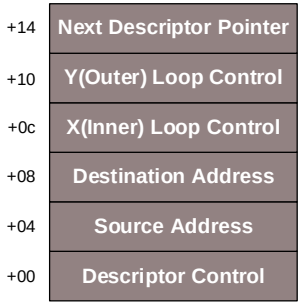
Review TRM section 7.1.3 and Register TRM for additional details

Descriptor type (3/4)

- 2D transfer
  - Two-dimensional “for loop” transfer
  - Uses six-words descriptor
  - Transfer example:

```
for (Y_IDX =0; Y_IDX <= Y_COUNT; Y_IDX++){
    for (X_IDX =0; X_IDX <= X_COUNT; X_IDX++){
        DST_ADDR[DST_INCR] = (DATA_SIZE) SRC_ADDR[SRC_INCR]
    }
}
```

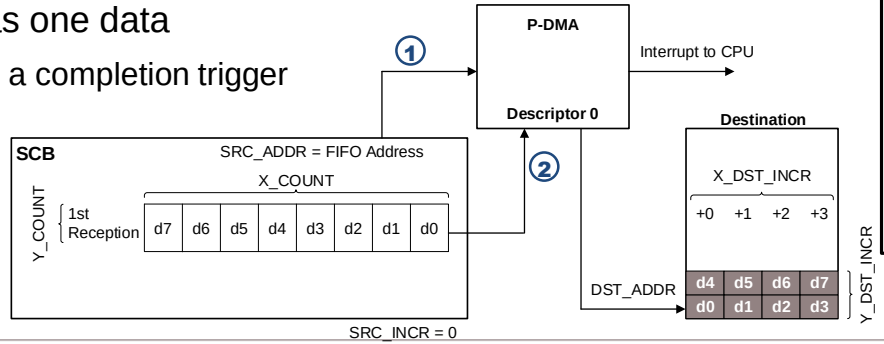
*DST_INCR/SRC_INCR depend on X/Y_INCR



Hint Bar

Review TRM section 7.1.3 and Register TRM for additional details

- Use case: Transfer SCB reception data in FIFO (1/2)
  - Handling multiple FIFO data as one data
    1. Activate P-DMA by receiving a completion trigger
    2. Repeat data transfer from SRC_ADDR to DST_ADDR by X_COUNT

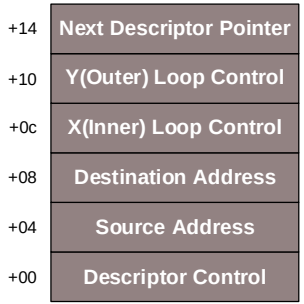


Descriptor type (3/4)

- 2D transfer
  - Two-dimensional “for loop” transfer
  - Uses six-words descriptor
  - Transfer example:

```
for (Y_IDX =0; Y_IDX <= Y_COUNT; Y_IDX++){
    for (X_IDX =0; X_IDX <= X_COUNT; X_IDX++){
        DST_ADDR[DST_INCR] = (DATA_SIZE) SRC_ADDR[SRC_INCR]
    }
}
```

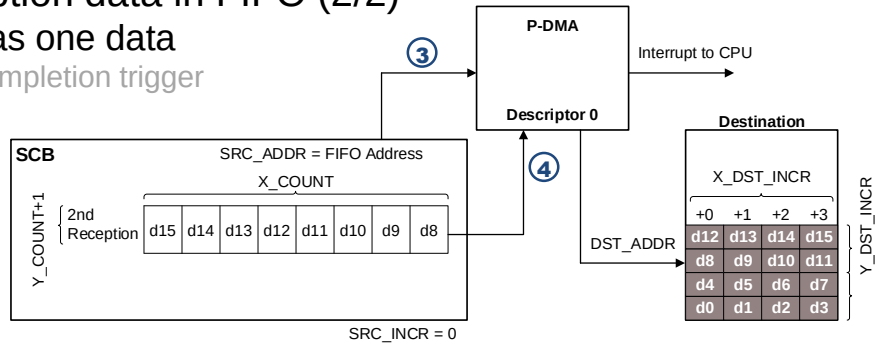
*DST_INCR/SRC_INCR depend on X/Y_INCR



Hint Bar

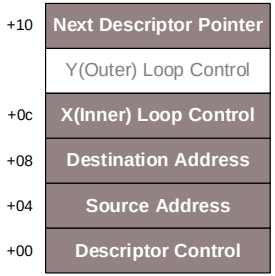
Review TRM section 7.1.3 and Register TRM for additional details

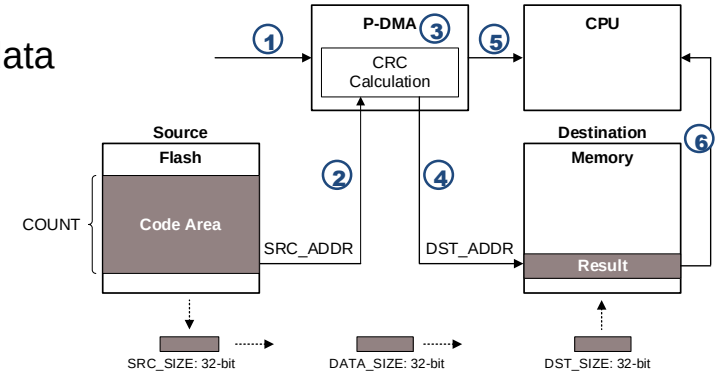
- Use case: Transfer SCB reception data in FIFO (2/2)
  - Handling multiple FIFO data as one data
    1. Activate P-DMA by receiving a completion trigger
    2. Repeat data transfer from SRC_ADDR to DST_ADDR by X_COUNT
    3. Activate P-DMA by receiving a completion trigger, when the second data is received
    4. Repeat data transfer from SRC_ADDR to DST_ADDR by X_COUNT



Descriptor type (4/4)

- CRC transfer
  - Execute the CRC calculation of the specified area
    - CRC-32, CRC-16, CRC-16-CCITT
    - Byte ordering/remainder bit reverse
    - Set CRC seed value
    - CRC result is stored to the destination address
  - This mode does not transfer data
- Use case: Initial CRC check for program data in Flash
  1. Activate P-DMA with software trigger
  2. Transfer data from SRC_ADDR
  3. CRC calculation (code area)
  4. Transfer CRC result to DST_ADDR
  5. Generate interrupt to CPU
  6. The CPU checks the CRC result with the expected value
- Advantage: Performs high-speed CRC calculation without a CPU





Hint Bar

Review TRM section 7.1.3 and Register TRM for additional details

Descriptor trigger types (1/5)

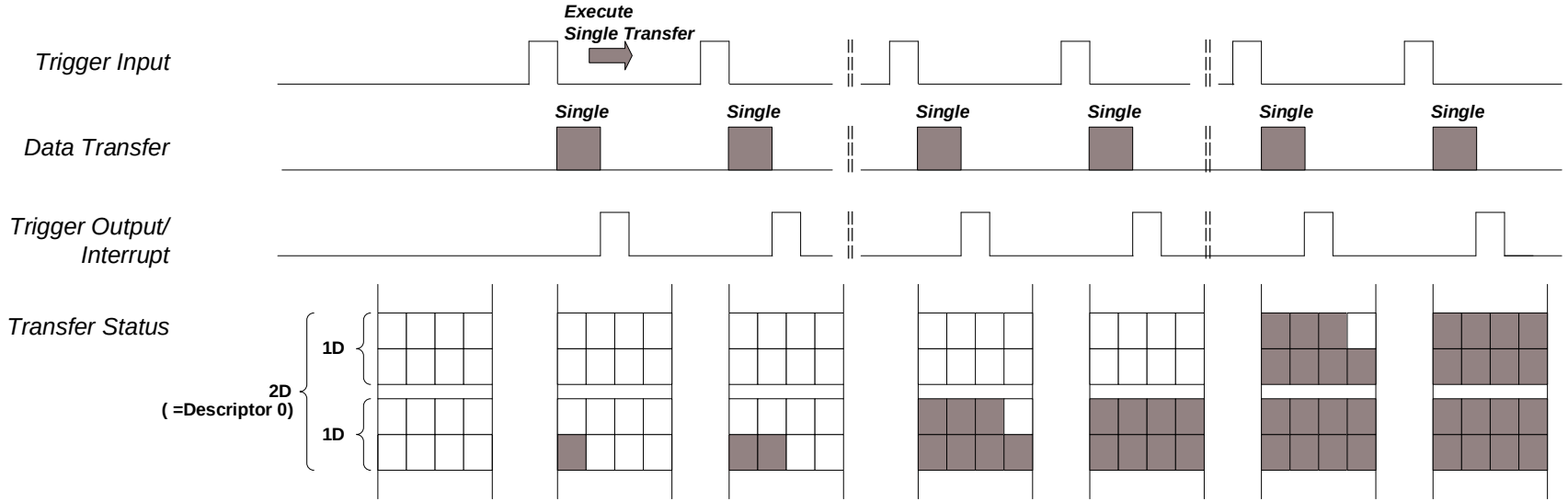
- Input trigger
  - Four types of input trigger action
    - Type 0: Executed by a single transfer
    - Type 1: Executed by a single 1D transfer
    - Type 2: Executed by the current descriptor
    - Type 3: Executed by a descriptor list
- Output trigger/interrupt
  - Four types of output trigger/interrupt generation
    - Type 0: Generated by a single transfer
    - Type 1: Generated by single 1D transfer
    - Type 2: Generated by the current descriptor
    - Type 3: Generated by descriptor list
- Input trigger, output trigger, and interrupt can be set individually

Hint Bar

Review TRM sections 7.1.3 and 7.2.3 and Register TRM for additional details

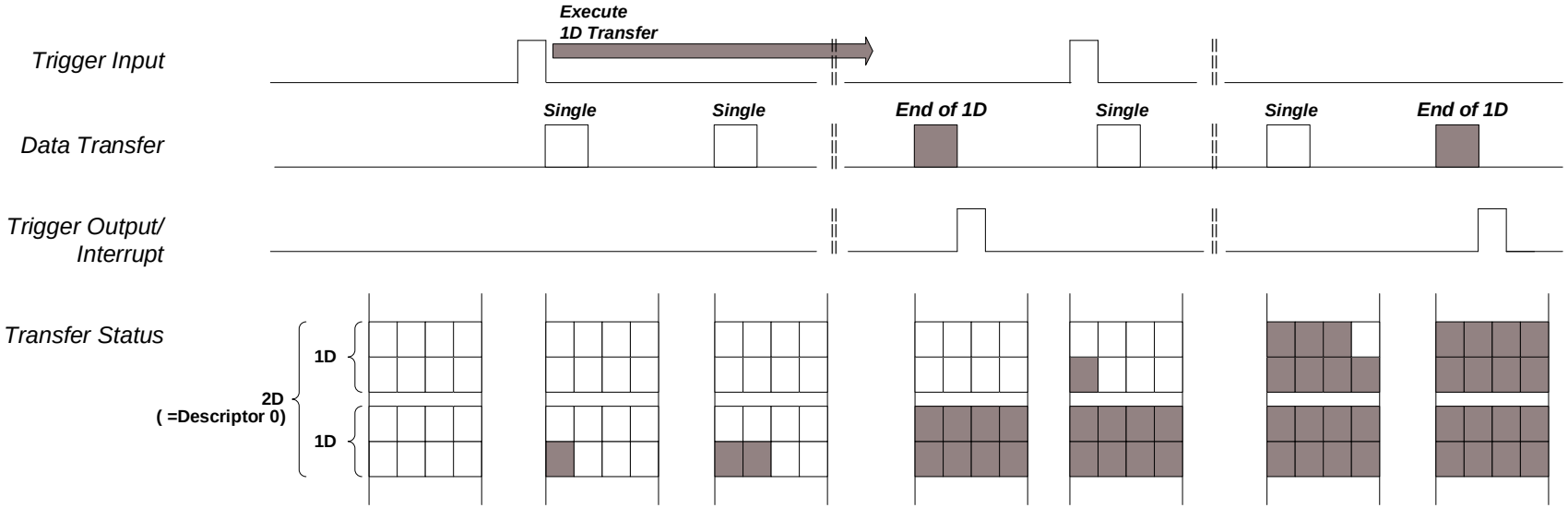
Descriptor trigger types (2/5)

- Example of trigger action
  - Descriptor input trigger Type 0: Executes a single transfer
  - Descriptor output trigger/interrupt Type 0: Generated by a single transfer



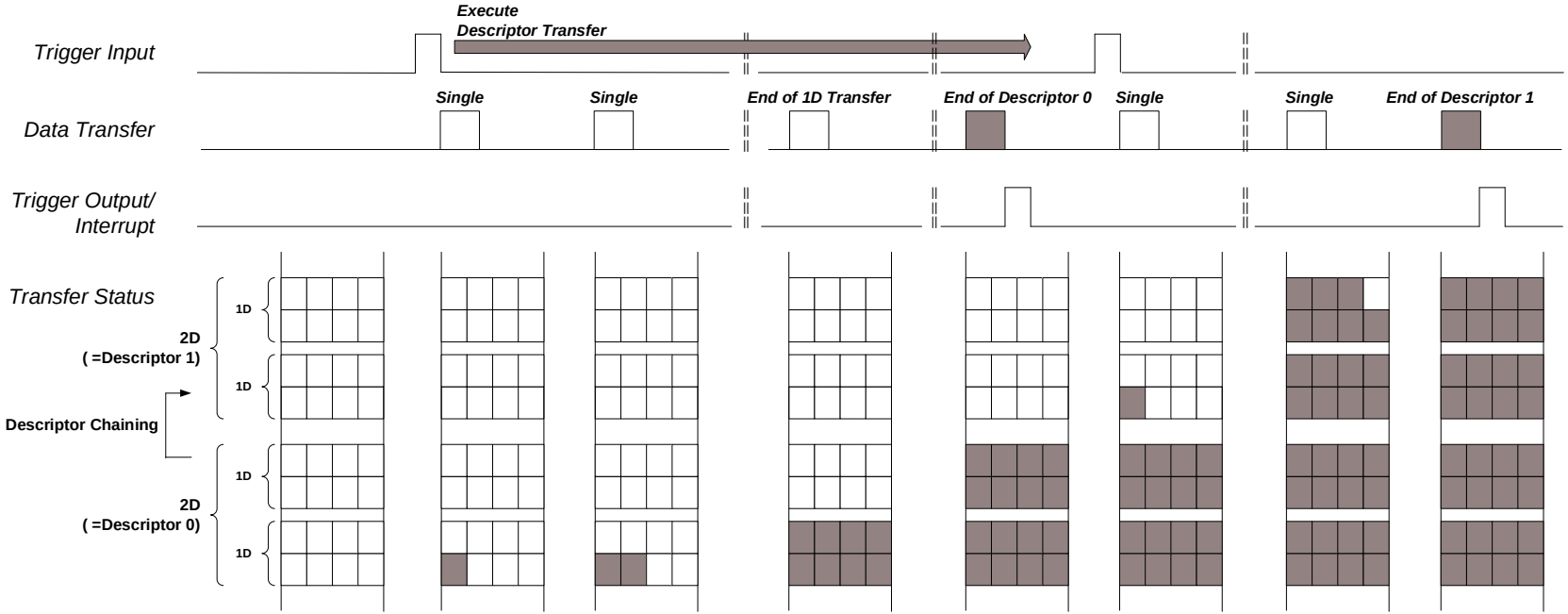
Descriptor trigger types (3/5)

- Example of trigger action
  - Descriptor input trigger Type 1: Executed by a single 1D transfer
  - Descriptor output trigger/interrupt Type 1: Generated by a single 1D transfer



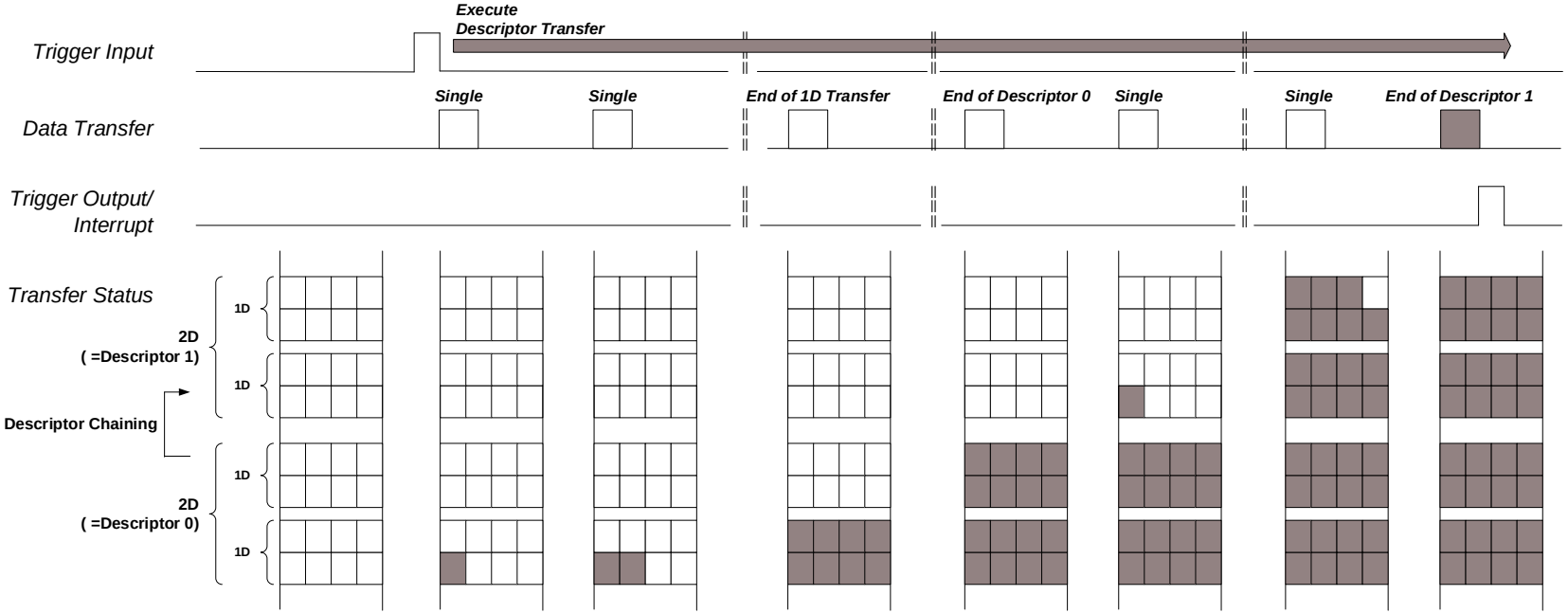
Descriptor trigger types (4/5)

- Example of trigger action
  - Descriptor input trigger Type 2: Executed by the current descriptor
  - Descriptor output trigger/interrupt Type 2: Generated by the current descriptor



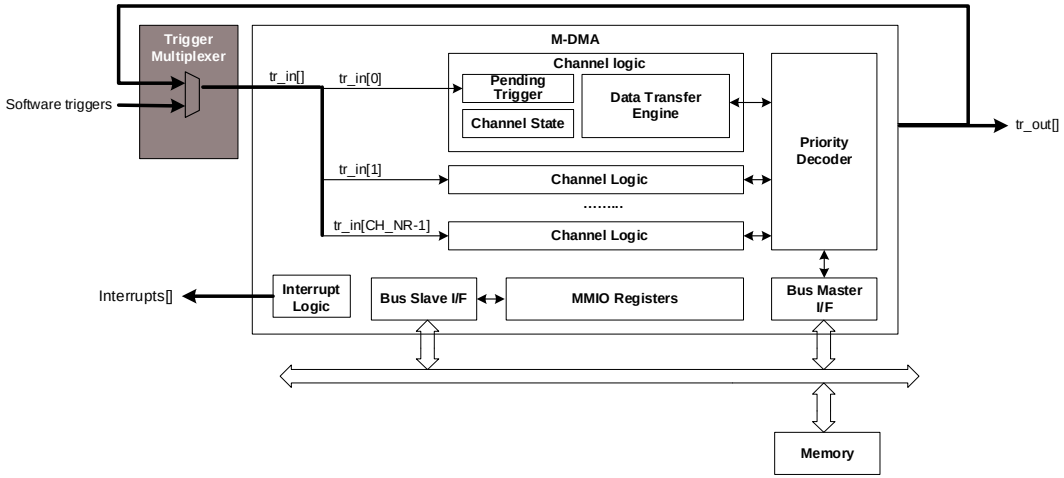
Descriptor trigger types (5/5)

- Example of trigger action
  - Descriptor input trigger Type 3: Executed by a descriptor list
  - Descriptor output trigger/interrupt Type 3: Generated by a descriptor list



M-DMA block diagram

- Trigger multiplexers
  - Connect each channel to one specific system trigger
  - Software (SW) system triggers[1]
  - Trigger output (tr_out)
    - Each trigger output can be used as its own active trigger using trigger multiplexers
    - They can be used to trigger different transfers as input triggers for other channels



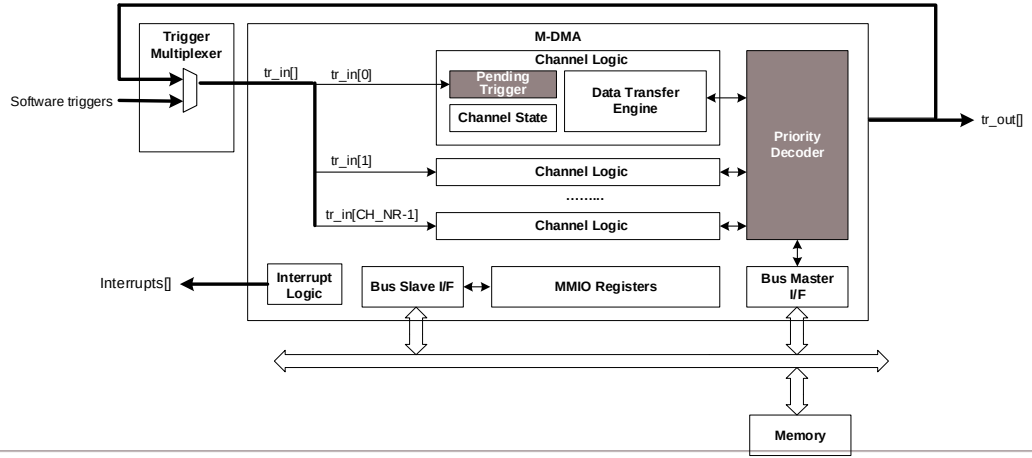
Hint Bar

Review TRM section 7.2.6 and chapter 29 for additional details

[1] This is different from P-DMA.

M-DMA block diagram

- Pending triggers
  - Keeps track of activated triggers by storing channel triggers
  - Manages multiple pending channel triggers
- Priority decoder
  - Determines the highest priority channel of transfer request from each transfer engine[1]
  - Has four priority levels
  - Performs round-robin arbitration within the same priority group



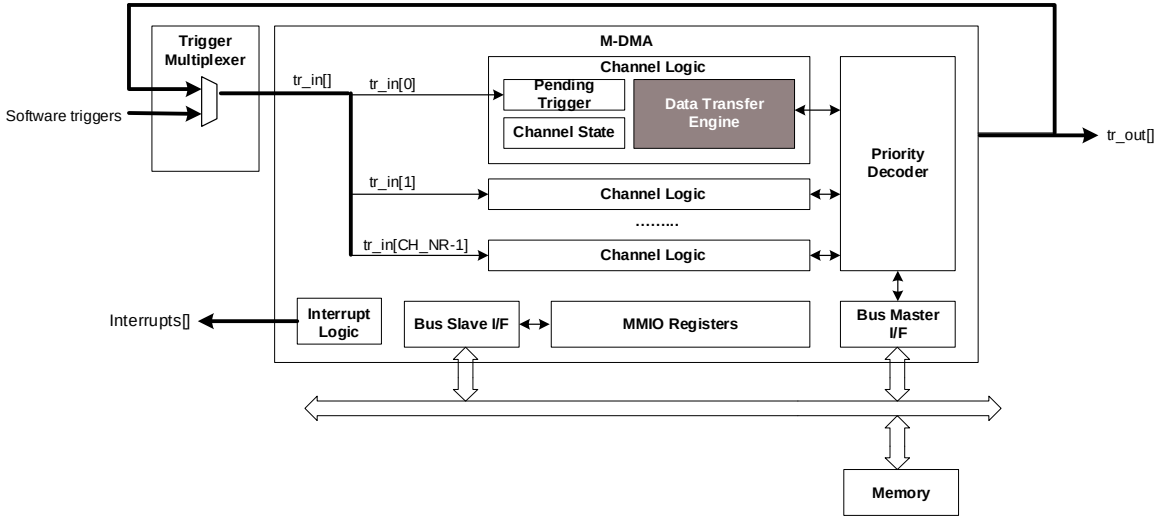
Hint Bar

Review TRM section 7.2.6 for additional details

[1] This is different from P-DMA.

M-DMA block diagram

- Data transfer engine
  - Dedicated engine for each channel[1]
  - Transfers data from source to destination according to descriptor
  - Reads channel descriptor from memory
  - Generates the trigger out to trigger multiplexers



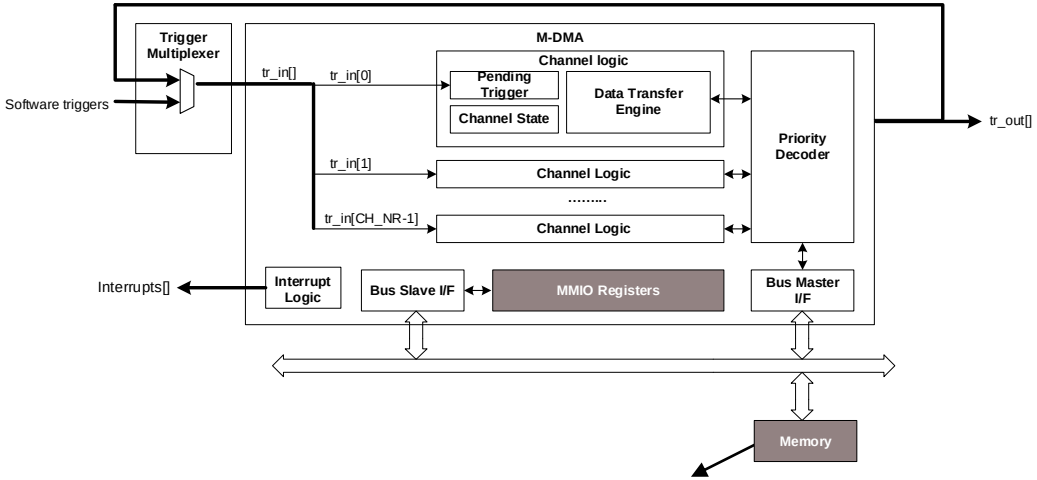
Hint Bar

Review TRM section 7.2.6 for additional details

[1] This is different from P-DMA.

M-DMA block diagram

- Descriptor
  - Stored in memory
  - Descriptor chaining
  - Descriptor types
    - Single transfer
    - 1D transfer
    - 2D transfer
    - Memory copy[1]
    - Scatter[1]
  - Descriptor structure
    - Descriptor control:
      Trigger type, transfer size, descriptor type
    - Source address/destination address
    - X(Inner) loop/Y(Outer) loop size and Increment
    - Next descriptor pointer
  - The descriptor pointer position for each channel is stored in the register



	+1c	Next Descriptor Pointer
	+18	Y(Outer) Loop Increment
	+14	Y(Outer) Loop Size
	+10	X(Inner) Loop Increment
	+0c	X(Inner) Loop Size
	+08	Destination Address
	+04	Source Address
Descriptor Pointer	+00	Descriptor Control

Hint Bar

Review TRM section 7.2.6 and Register TRM for additional details

Descriptor size is different from P-DMA, but the setting items are the same

System RAM is used as a memory to store the descriptor

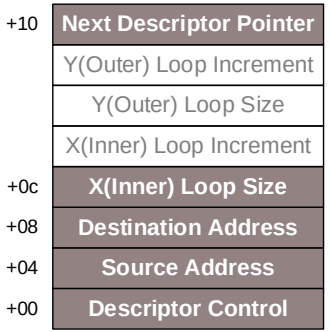
[1] This is different from P-DMA.

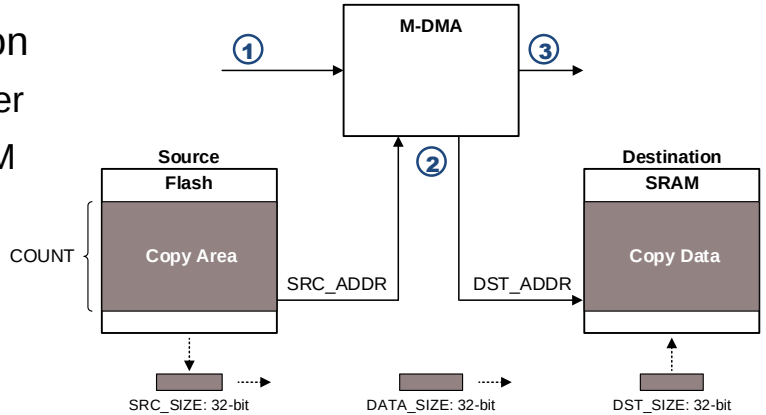
Descriptor type (1/3)

- Memory copy
  - One-dimensional “for loop” transfer
  - Uses five-word descriptor[1]
  - Transfer example:

```
for (X_IDX =0; X_IDX <= X_COUNT; X_IDX+){
        DST_ADDR[IDX] = SRC_ADDR[IDX]
}
```

- Use case: Interrupt vector copy or program copy for RAM execution
  1. Activate M-DMA by software trigger
  2. Copy the data from Flash to SRAM
  3. Generate interrupt to CPU





Hint Bar

Review TRM section 7.2.3 and Register TRM for additional details

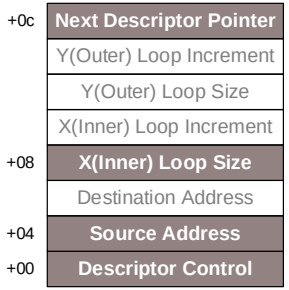
[1] SRC_SIZE, DST_SIZE, and DATA_SIZE fields are not used for memory copy. Memory copy transfers the number of bytes specified by X Loop Count in the optimum size. For example, when “Source Address” = 0x08000001, “X Loop Count” = 10, first transfer size is 8-bit, second transfer size is 16-bit, and third transfer size is 32-bit.

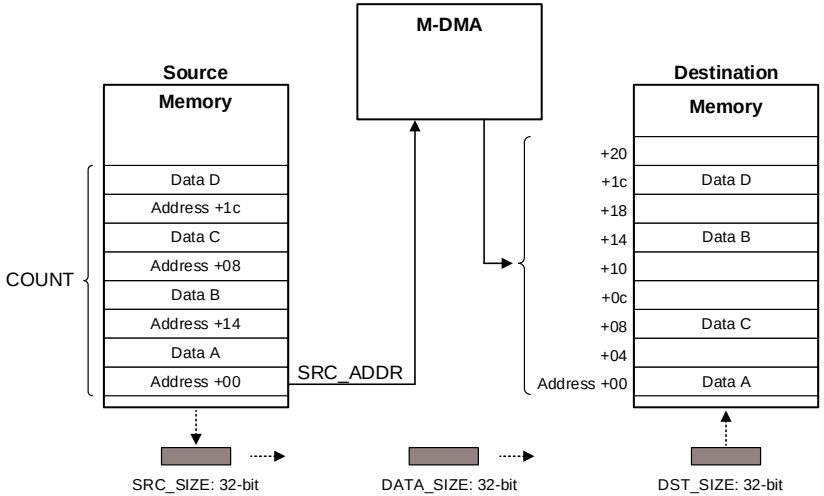
Descriptor type (2/3)

- Scatter
  - Writes a set of 32-bit data elements, which has addresses “scattered” around
  - Writes the specified data to the specified address
  - Uses four-word descriptor[1]
  - Transfer example:

```
for (X_IDX =0; X_IDX <= X_COUNT; X_IDX +=2){
    address  = SRC_ADDR[IDX]
    data     = SRC_ADDR[IDX+1]

    *address = data
}
```





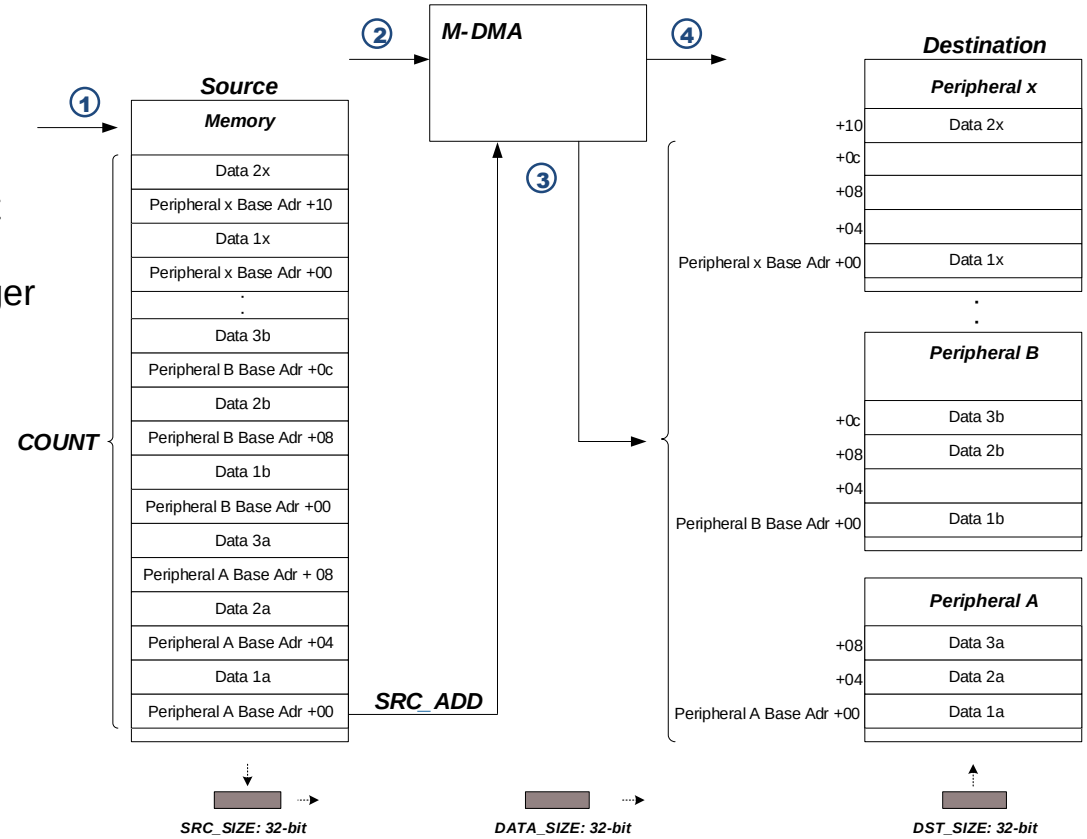
Hint Bar

Review TRM section 7.2.3 and Register TRM for additional details

[1] SRC_SIZE, DST_SIZE must be set to word (32-bit) for scatter.

Descriptor type (3/3)

- Use case: Quick initial setting or setting change of peripheral registers without CPU
  1. Write the address and data to be set in the peripheral to the memory
  2. Activate M-DMA using software trigger
  3. Write the data from the memory to peripheral registers
  4. Generate an interrupt to CPU when the transfer is complete



AXI M-DMA

AXI M-DMA overview

- TRAVEO™ T2G cluster 2D series has AXI M-DMA[1]
  - AXI master interface
  - Data transfer between AXI slaves
  - Uses a memory copy transfer as the primitive[2]
  - Support 2D/3D memory copy
  - Cannot access the peripheral bus infrastructure
  - Buffer size between 64 bytes and 288 bytes per channel[3]
  - Interrupt, input trigger, and output trigger per channel
  - A channel is assigned a priority between 0 (high) and 3 (low)
  - Round-robin arbitration is applied within same priority group

Feature	AXI M-DMA
Transfer engine	Dedicated for each channel
Channel priority	Four levels
Transfer mode	- Memory copy - 2D Memory copy - 3D Memory copy
Descriptor	- Source and destination address - Channel action - Data transfer mode - Activation trigger type (4 types) - Output trigger type (4 types) - Interrupt type (4 types) - Descriptor chaining
Access-control attributes[3]	- Privileged/Unprivileged - Secure/Non-secure - Protection contexts

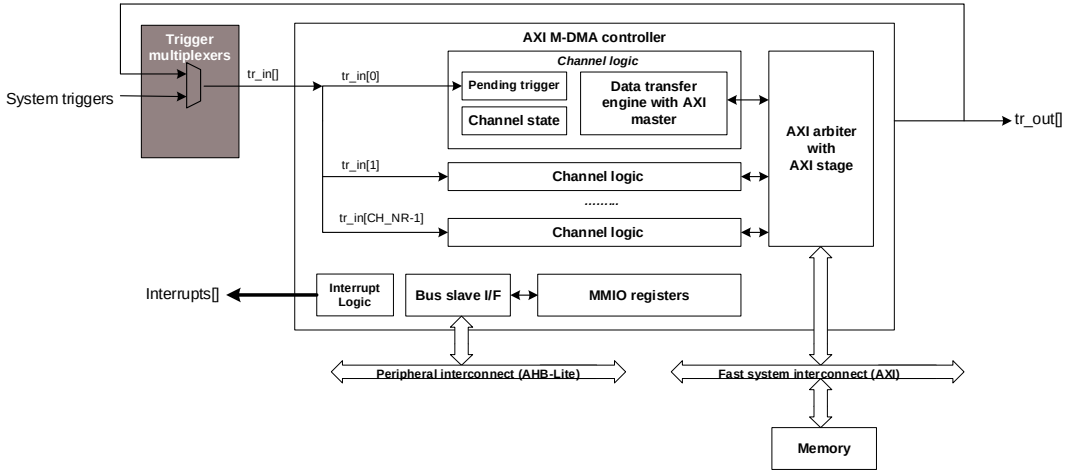
Hint Bar

Review TRM chapter 7.3 for additional details

[1, 3] See the device specific datasheet to see if the feature is supported.
[2] Copying of M_COUNT+1 bytes from a source address to a destination address using AXI bursts
[4] AXI M-DMA channels inherit the access attributes of the bus transfer that programmed the channel.

AXI M-DMA block diagram (1/3)

- A data transfer is initiated by an input trigger via trigger multiplexer
  - System trigger
  - Software (SW) trigger
  - Trigger output (tr_out)
    - Each trigger output can be used as its own active trigger using trigger multiplexers
    - They can be used to trigger different transfers as input triggers for other channels

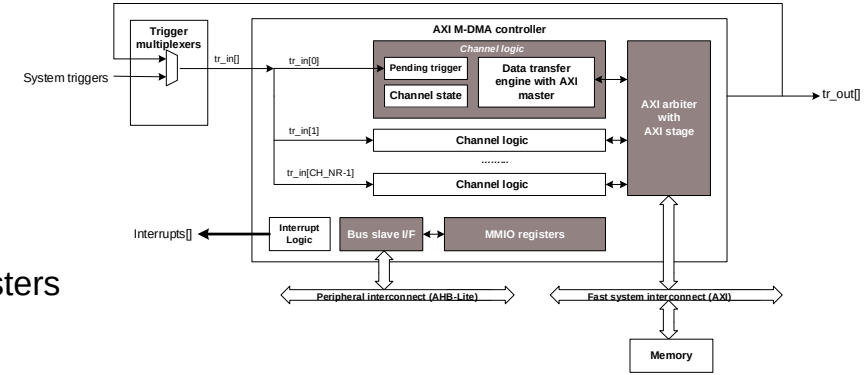


Hint Bar

Review TRM section 7.3.2 and chapter 29 for additional details

AXI M-DMA block diagram (2/3)

- Channel logic
  - Keeps track of the channel's input trigger
  - Maintains the channel state and a data transfer engine
  - Reads channel descriptor from memory
- AXI arbiter
  - Performs arbitration between the channels
  - Applies round-robin arbitration within the same priority group
- MMIO registers
  - AXI M-DMA control / status registers
- Bus slave I/F
  - AHB-Lite slave I/F for register access

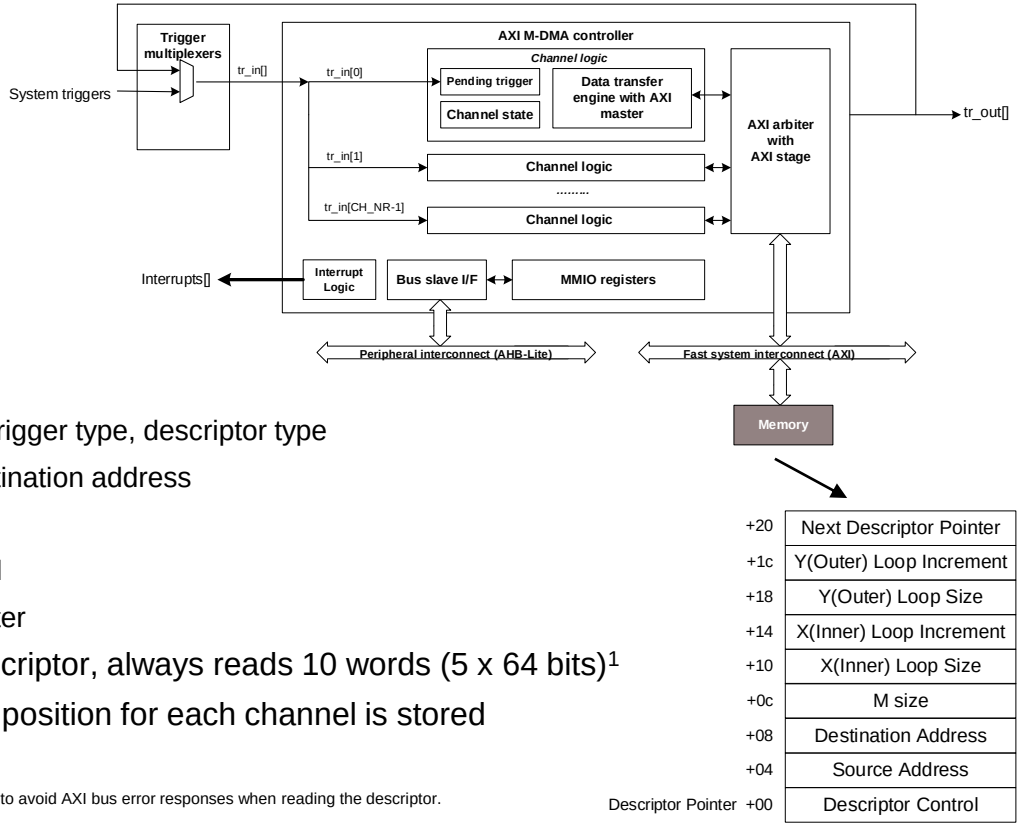


Hint Bar

Review TRM section 7.3.7 for additional details

AXI M-DMA block diagram (3/3)

- Descriptor
  - Stored in memory
  - Descriptor chaining
  - Descriptor types
    - Memory copy
    - 2D Memory copy
    - 3D Memory copy
  - Descriptor structure
    - Descriptor control: Trigger type, descriptor type
    - Source address/destination address
    - Memory copy size
    - X loop/Y loop control
    - Next descriptor pointer
  - When fetching the descriptor, always reads 10 words (5 x 64 bits)[1]
  - The descriptor pointer position for each channel is stored in the register



Hint Bar

Review TRM section 7.3.3 and Register TRM for additional details

System RAM is used as a memory to store the descriptor.

[1] Needs to be considered when setting up descriptors, to avoid AXI bus error responses when reading the descriptor. See the TRM section 7.3.3 for more details.

AXI transaction

- Rules for generating AXI transactions
  - Only incrementing burst is used; wrapping and fixed burst are not supported.
  - The data size of AXI transactions is always 64 bits
  - AXI transactions never cross a 32-byte boundary
  - Maximum burst length is four beats
  - Different memory copy operations are never combined to one AXI transaction
  - Within one iteration of a memory copy operation, the transfers within the same aligned 32-byte region are always performed as one AXI transaction
  - First AXI transaction is an unaligned transaction unless the start address is a multiple of 8
  - Last AXI transaction of each memory copy operation starts at an address that is a multiple of 32, and has the minimum burst length for end of the copy address range
  - AXI transactions between the first and the last are full 32-byte bursts
  - For unaligned write transactions at the start and incomplete write transactions at the end of a memory copy operation, only the correct bytes are written
  - For unaligned read transactions at the start and incomplete read transactions at the end of a memory copy operation, reading is always performed in multiples of 8 bytes.

Hint Bar

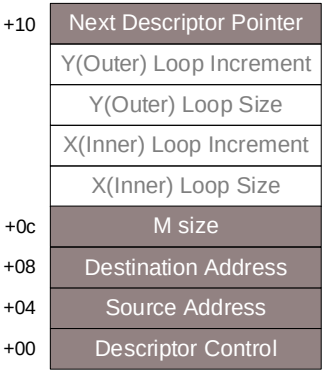
Review TRM 7.3.6 for additional details

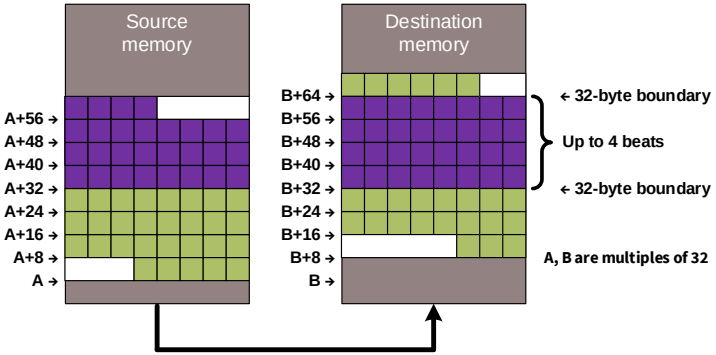
Descriptor type (1/3)

- Memory copy
  - One-dimensional “for loop” transfer
  - Uses five-words descriptor
  - Transfer example:

```
for (M_IDX =0; M_IDX <= M_COUNT; M_IDX++){
        DST_ADDR[M_IDX] = SRC_ADDR[M_IDX]
}
```

- Use case:
  - Transfer conditions
    - Source address: A+3
    - Destination address: B+13
    - Transfer count: 57 bytes
  - Read from source address using 2 burst transfers (4 beats + 4 beats)
  - Write to destination address using 3 burst transfers (3 beats + 4 beats + 1 beats)





Hint Bar

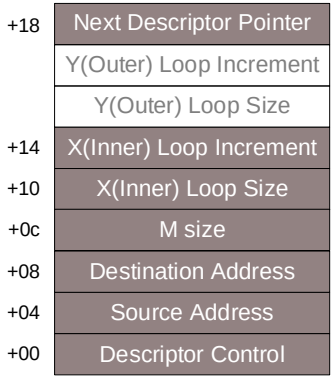
Review TRM section 7.3.1, 7.3.3 and 7.3.8 for additional details

Descriptor type (2/3)

- 2D memory copy
  - Two-dimensional “for loop” transfer
  - Uses seven-words descriptor
  - Transfer example:

```
for (X_IDX = 0; X_IDX <= X_COUNT; X_IDX++) {
    for (M_IDX = 0; M_IDX <= M_COUNT; M_IDX++) {
        DST_ADDR[M_IDX + X_IDX * DST_X_INCR] =
                        SRC_ADDR[M_IDX + X_IDX * SRC_X_INCR];
    }
}
```

- Use case:
  - 2D memory copy can be used to transfer bitmaps



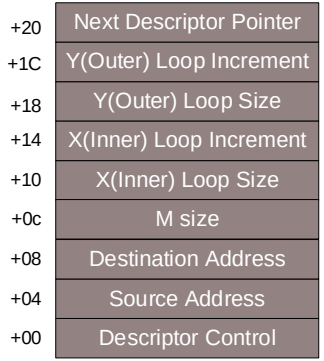
Hint Bar

Review TRM section 7.3.1 and 7.3.3 for additional details

Descriptor type (3/3)

- 3D memory copy
  - Three-dimensional “for loop” transfer
  - Uses nine-word descriptor
  - Transfer example:

```
for (Y_IDX = 0; Y_IDX <= Y_COUNT; Y_IDX++) {
    for (X_IDX = 0; X_IDX <= X_COUNT; X_IDX++) {
        for (M_IDX = 0; M_IDX <= M_COUNT; M_IDX++) {
            DST_ADDR[M_IDX + X_IDX * DST_X_INCR + Y_IDX * DST_Y_INCR] =
                SRC_ADDR[M_IDX + X_IDX * SRC_X_INCR + Y_IDX * SRC_X_INCR];
        }
    }
}
```

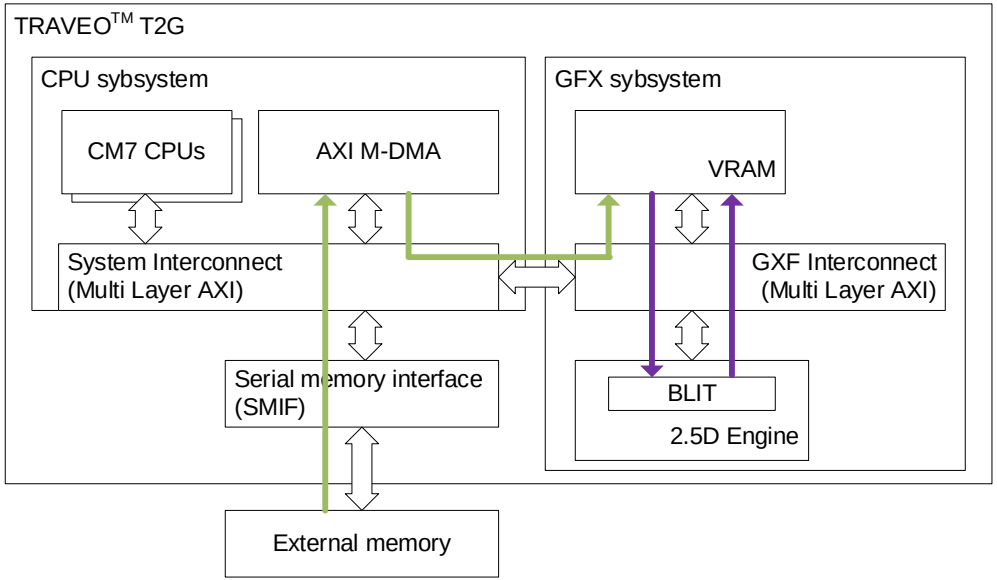


Hint Bar

Review TRM section 7.3.3 and Register TRM for additional details

Use case (1)

- Load the scene data to the GFX subsystem from external memory to VRAM[1]
  - The BLIT engine processes scene data in parallel during AXI M-DMA transfers
  - AXI M-DMA transfers data between external memory and GFX memory without CPU involvement

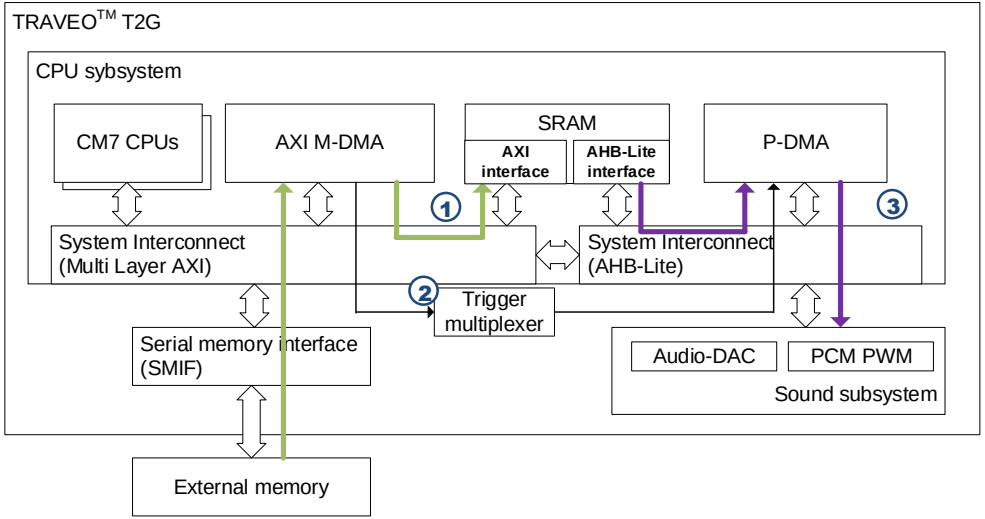


[1] CYT4EN series has LPDDR4 memory interface instead of VRAM. See the specific datasheet for details.

Use case (2)

- Transfer PCM stream from external memory to Sound subsystem
  1. AXI M-DMA transfers data from external memory to SRAM
  2. AXI M-DMA can initiate P-DMA[1] via trigger multiplexer
  3. P-DMA transfers stream data from SRAM to Sound subsystem, such as Audio-DAC and PCM-PWM

  - AXI M-DMA cannot access to peripheral registers directly



[1] See the specific datasheet for connection between DMAs by trigger multiplexer..

Descriptor trigger types

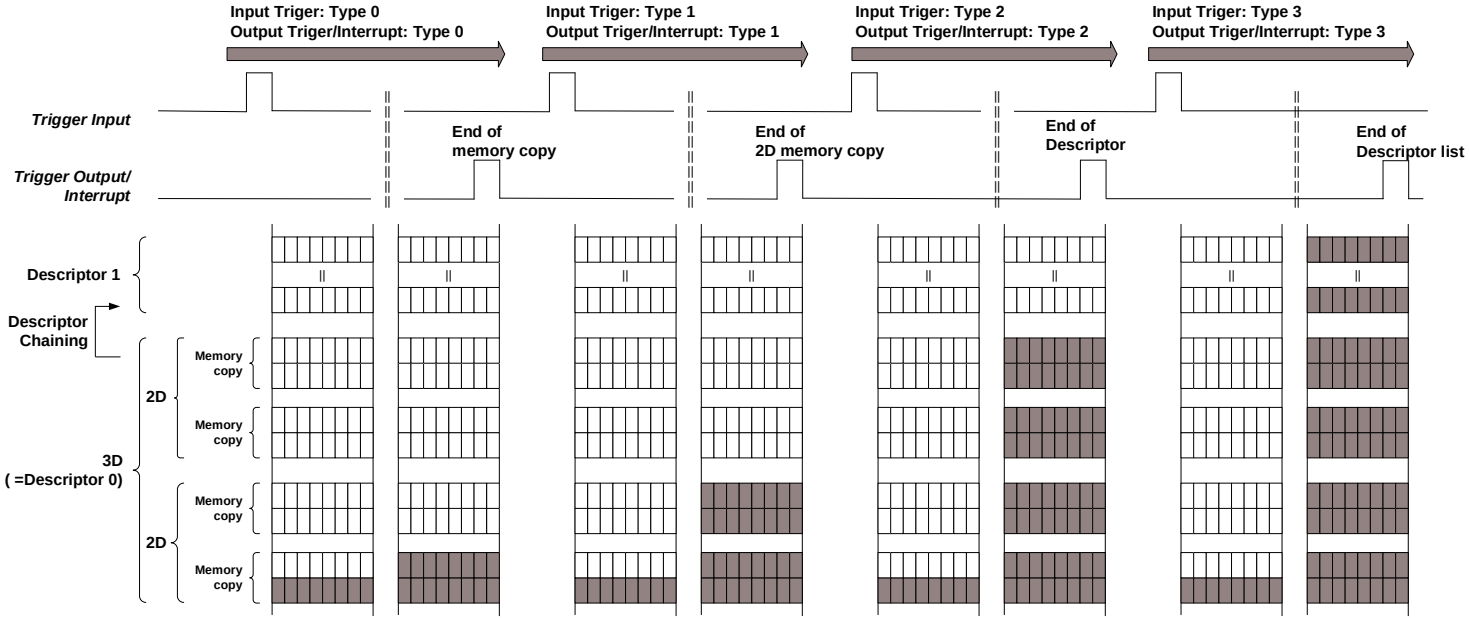
- Input trigger
  - Four types of input trigger action
    - Type 0: Executed a memory copy
    - Type 1: Executed a 2D memory copy
      - When the descriptor type is memory copy, this type behaves similar to type 0.
    - Type 2: Executed by the current descriptor
    - Type 3: Executed by a descriptor list
- Output trigger/interrupt
  - Four types of output trigger/interrupt generation
    - Type 0: Generated by a memory copy transfer
    - Type 1: Generated by a 2D memory copy transfer
      - When the descriptor type is memory copy, this type behaves similar to type 0.
    - Type 2: Generated by the current descriptor transfer
    - Type 3: Generated by descriptor list transfer
- Input trigger, output trigger, and interrupt can be set individually

Hint Bar

Review TRM sections 7.3.9 and Register TRM for additional details

Descriptor trigger types

- Example of each trigger action
  - Descriptor 0 is 3D memory copy and has next descriptor pointer address (Descriptor 1)
  - Descriptor 1 does not have next descriptor pointer address (= “0”)



Hint Bar

Review TRM sections 7.3.9 and Register TRM for additional details

Revision history

Revision	ECN	Submission Date	Description of Change
**	6136162	04/17/2018	Initial release
*A	6351891	16/10/2018	Added slide 2. Updated slides 3, 4, and 5. Added slide 29: Descriptor Type (3/3) Updated figures on slides 7, 9-11, and 23-26
*B	6633414	7/22/2019	Updated slide 2-4, 9. Added slide 5.
*C	6825576	03/06/2020	Updated slide 12, 15,16
*D	6952165	08/20/2020	Update slide 8, 27, 28
*E	7039675	11/26/2020	Updated page 2. Merged page 3 for TRAVEO™ T2G Body Controller Entry.
*F	7807792	09/06/2022	Updated slide 2,5 Added slide 31 to 43

Part of your life. Part of tomorrow.