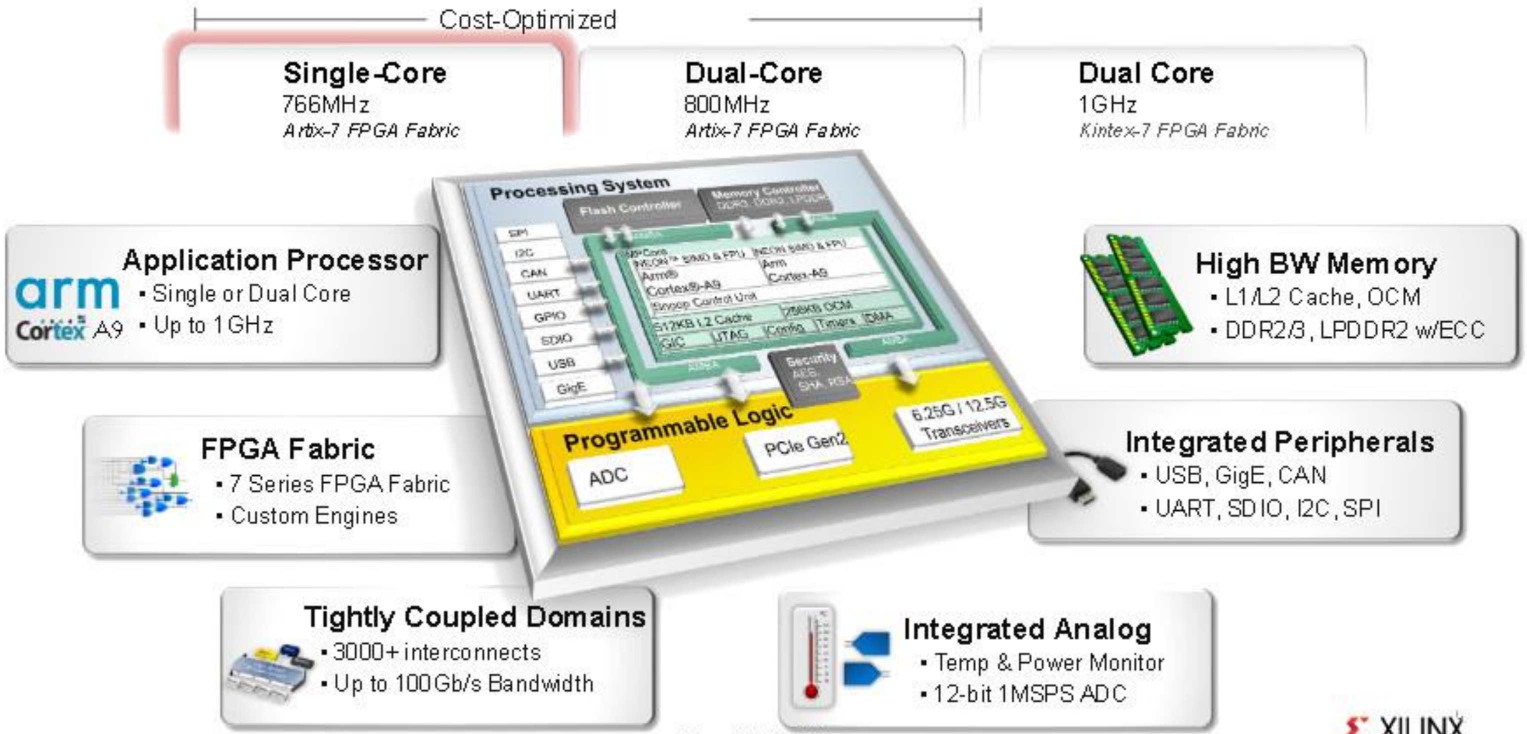
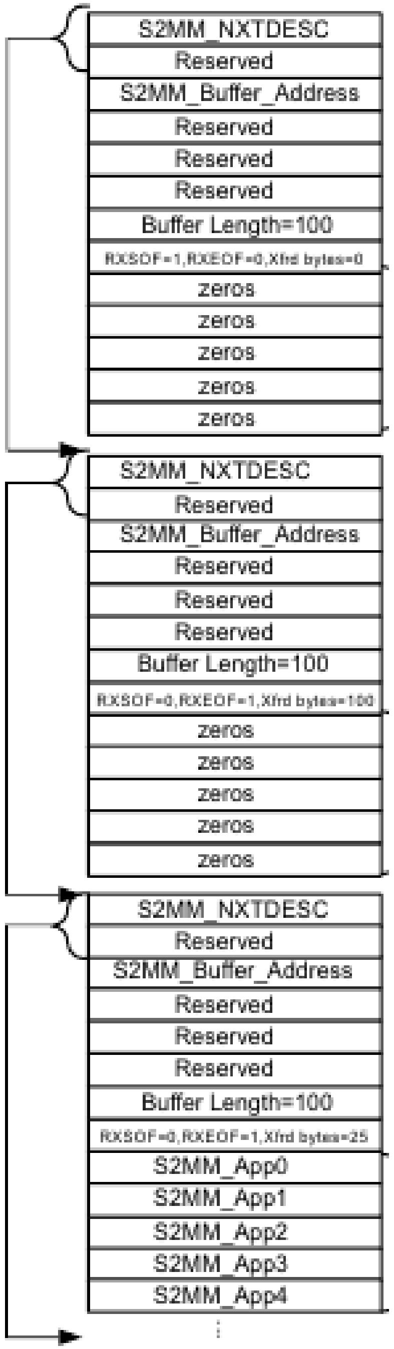
Zynq 7000: AXI DMA in Scatter Gather Mode



© Copyright 2020 Xilinx

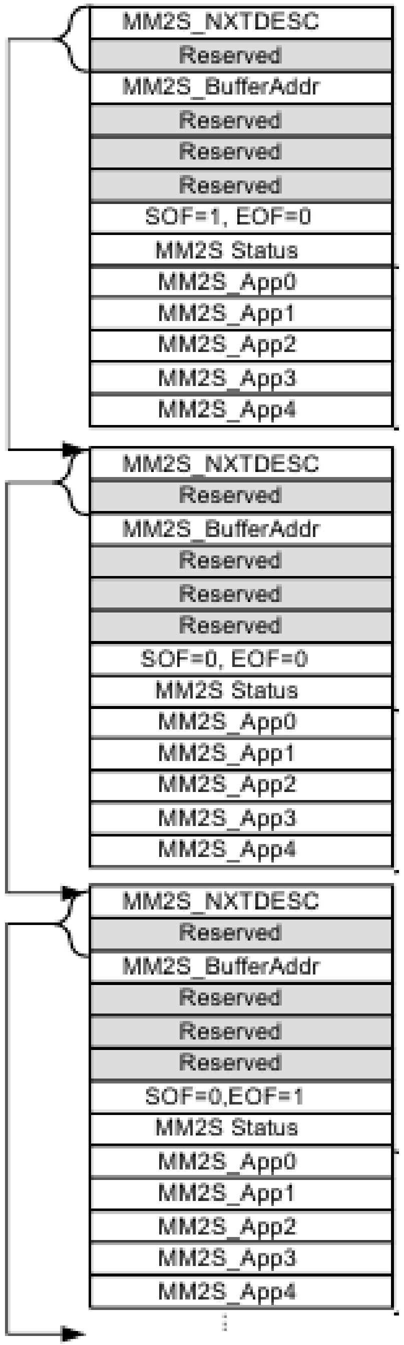
Aleksei Rostov, PhD,
Senior R&D Engineer,
FPGA/Embedded Linux Developer,
aleksei.rostov@protonmail.com



S2MM_NXTDESC
Reserved
S2MM_Buffer_Address
Reserved
Reserved
Reserved
Buffer Length=100
RXSOF=1,RXEOF=0,Xfrd bytes=0
zeros
zeros
zeros
zeros
zeros

S2MM_NXTDESC
Reserved
S2MM_Buffer_Address
Reserved
Reserved
Reserved
Buffer Length=100
RXSOF=0,RXEOF=1,Xfrd bytes=100
zeros
zeros
zeros
zeros
zeros

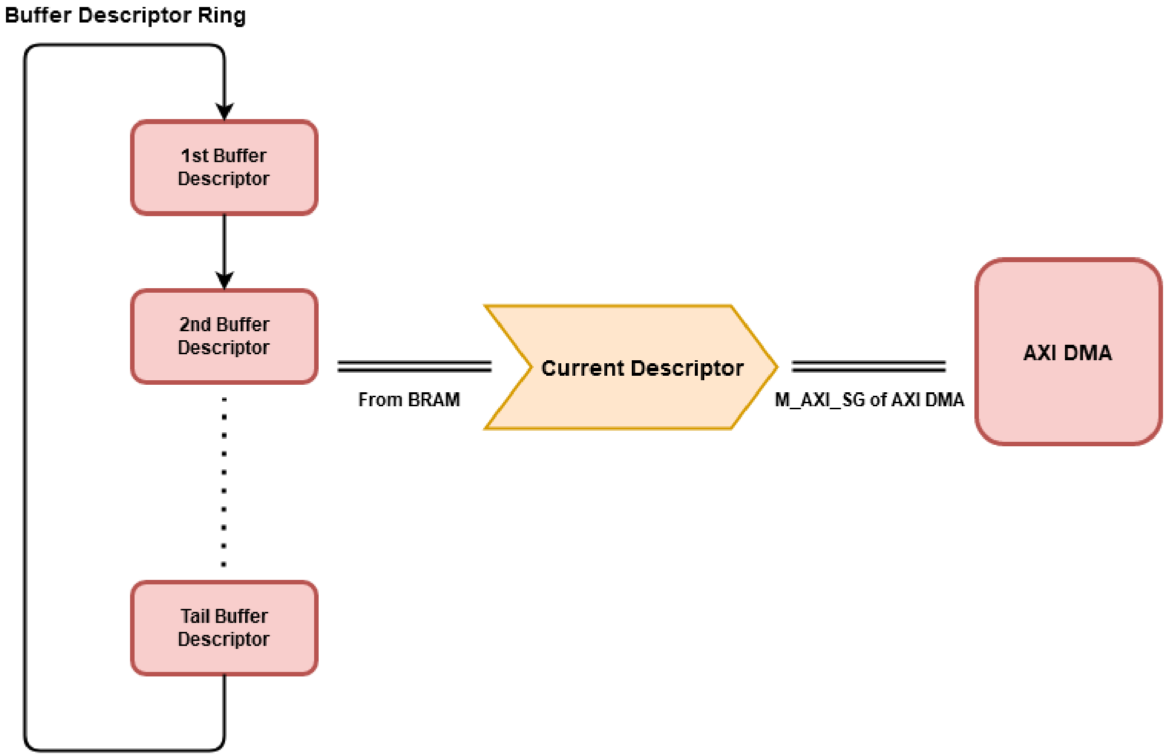
S2MM_NXTDESC
Reserved
S2MM_Buffer_Address
Reserved
Reserved
Reserved
Buffer Length=100
RXSOF=0,RXEOF=1,Xfrd bytes=25
S2MM_App0
S2MM_App1
S2MM_App2
S2MM_App3
S2MM_App4

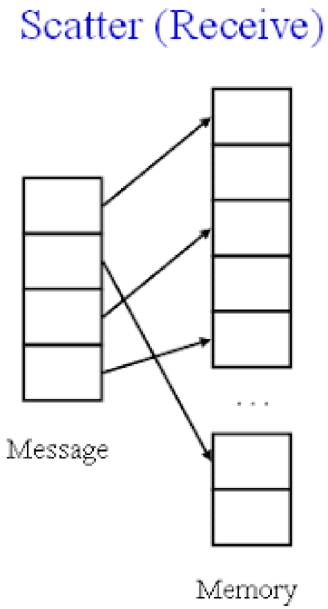


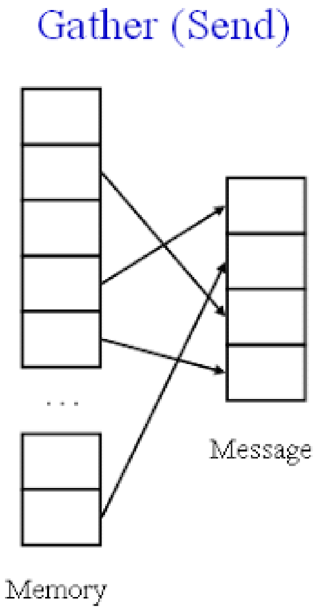
MM2S_NXTDESC
Reserved
MM2S_BufferAddr
Reserved
Reserved
Reserved
SOF=1, EOF=0
MM2S Status
MM2S_App0
MM2S_App1
MM2S_App2
MM2S_App3
MM2S_App4

MM2S_NXTDESC
Reserved
MM2S_BufferAddr
Reserved
Reserved
Reserved
SOF=0, EOF=0
MM2S Status
MM2S_App0
MM2S_App1
MM2S_App2
MM2S_App3
MM2S_App4

MM2S_NXTDESC
Reserved
MM2S_BufferAddr
Reserved
Reserved
Reserved
SOF=0,EOF=1
MM2S Status
MM2S_App0
MM2S_App1
MM2S_App2
MM2S_App3
MM2S_App4







HARDWARE DESIGN

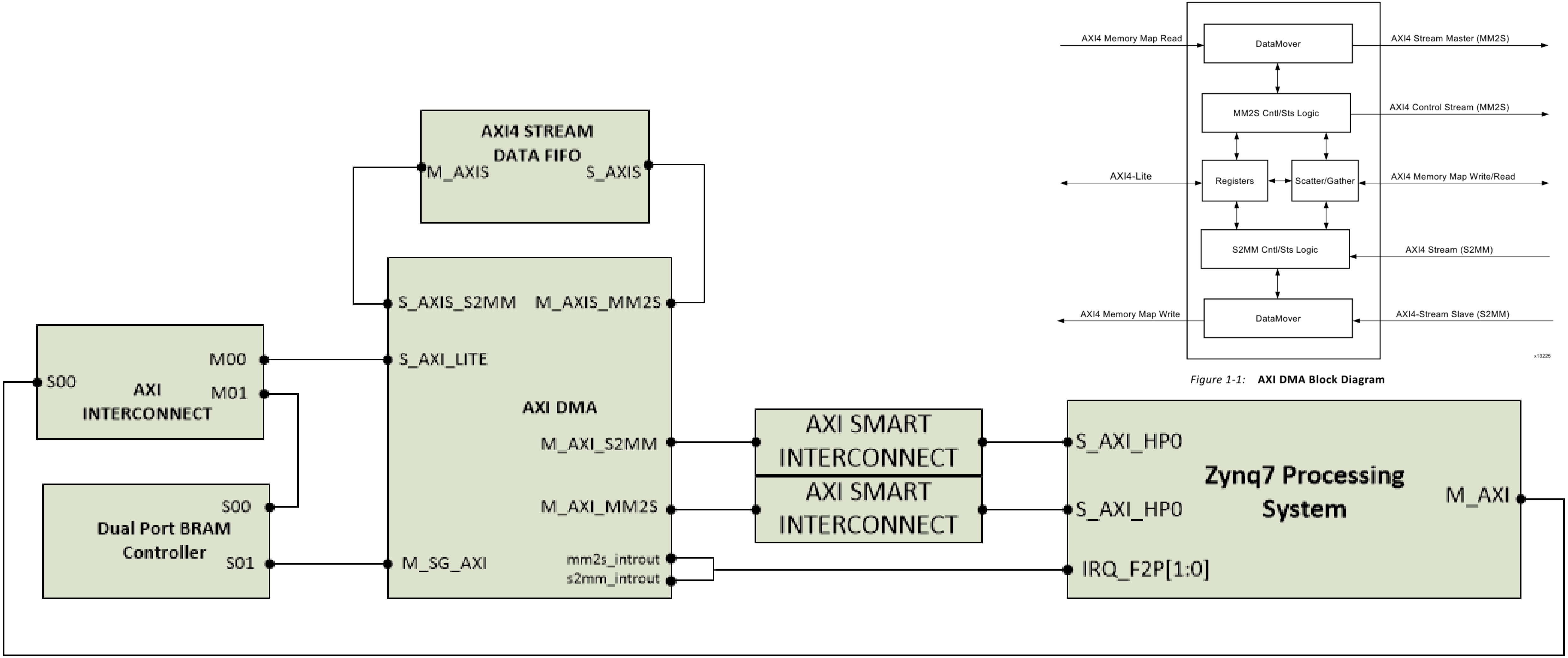


Figure 1-1: **AXI DMA Block Diagram**

aleksei.rostov@protonmail.com

AXI DMA SCATTER GATHER MODE

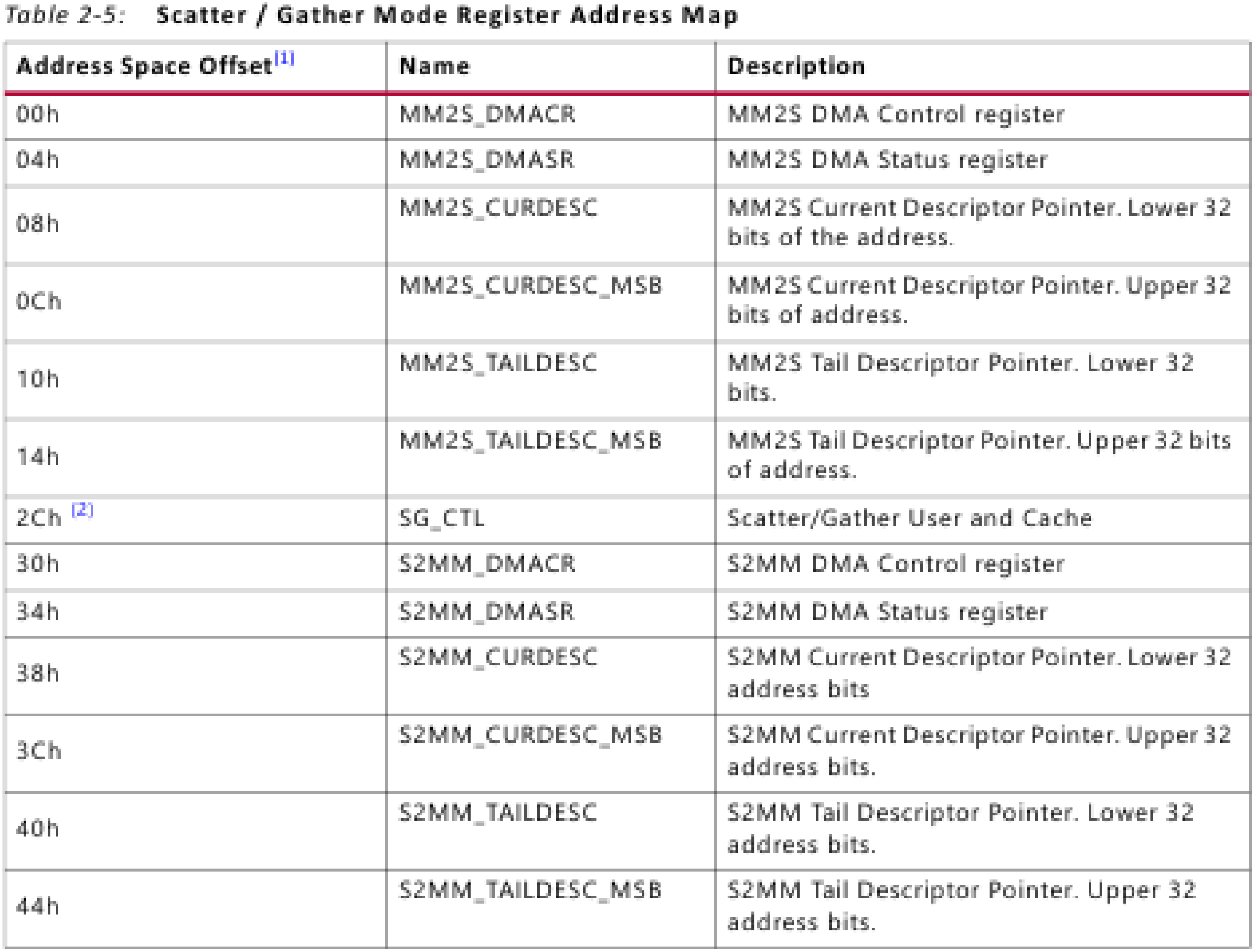


Table 2-5: **Scatter / Gather Mode Register Address Map**

Address Space Offset[1]	Name	Description
00h	MM2S_DMACR	MM2S DMA Control register
04h	MM2S_DMASR	MM2S DMA Status register
08h	MM2S_CURDESC	MM2S Current Descriptor Pointer. Lower 32 bits of the address.
0Ch	MM2S_CURDESC_MSB	MM2S Current Descriptor Pointer. Upper 32 bits of address.
10h	MM2S_TAILDESC	MM2S Tail Descriptor Pointer. Lower 32 bits.
14h	MM2S_TAILDESC_MSB	MM2S Tail Descriptor Pointer. Upper 32 bits of address.
2Ch [2]	SG_CTL	Scatter/Gather User and Cache
30h	S2MM_DMACR	S2MM DMA Control register
34h	S2MM_DMASR	S2MM DMA Status register
38h	S2MM_CURDESC	S2MM Current Descriptor Pointer. Lower 32 address bits
3Ch	S2MM_CURDESC_MSB	S2MM Current Descriptor Pointer. Upper 32 address bits.
40h	S2MM_TAILDESC	S2MM Tail Descriptor Pointer. Lower 32 address bits.
44h	S2MM_TAILDESC_MSB	S2MM Tail Descriptor Pointer. Upper 32 address bits.

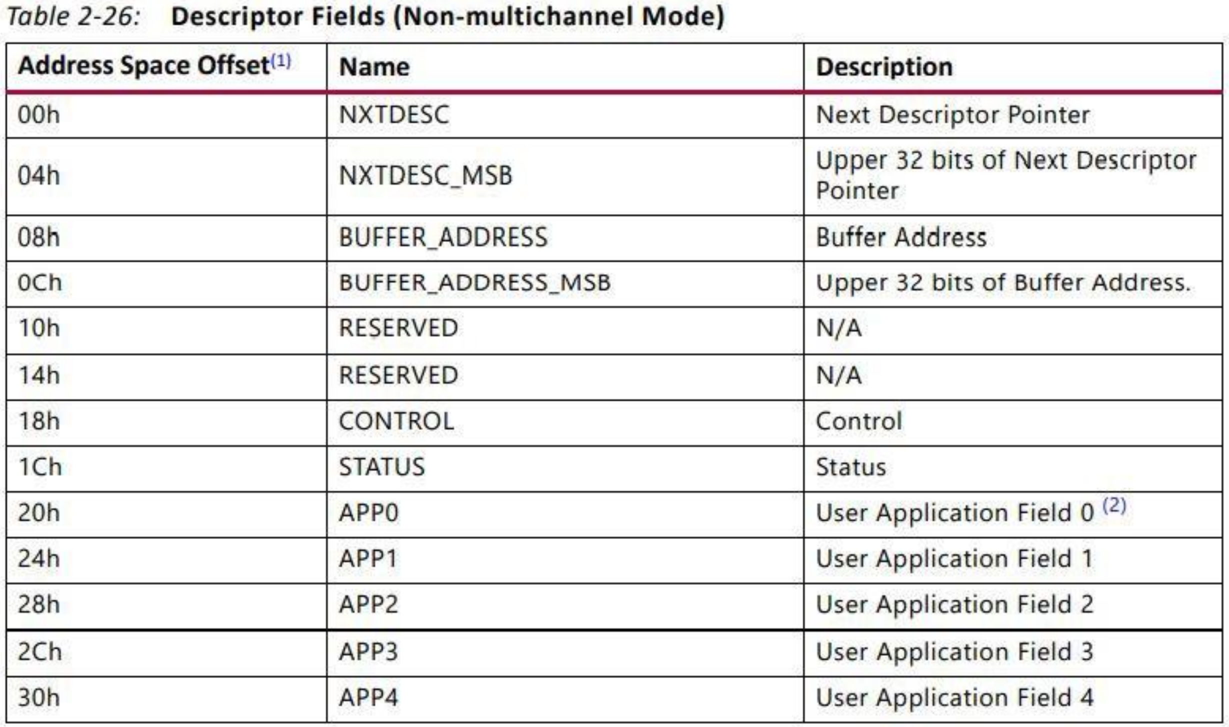
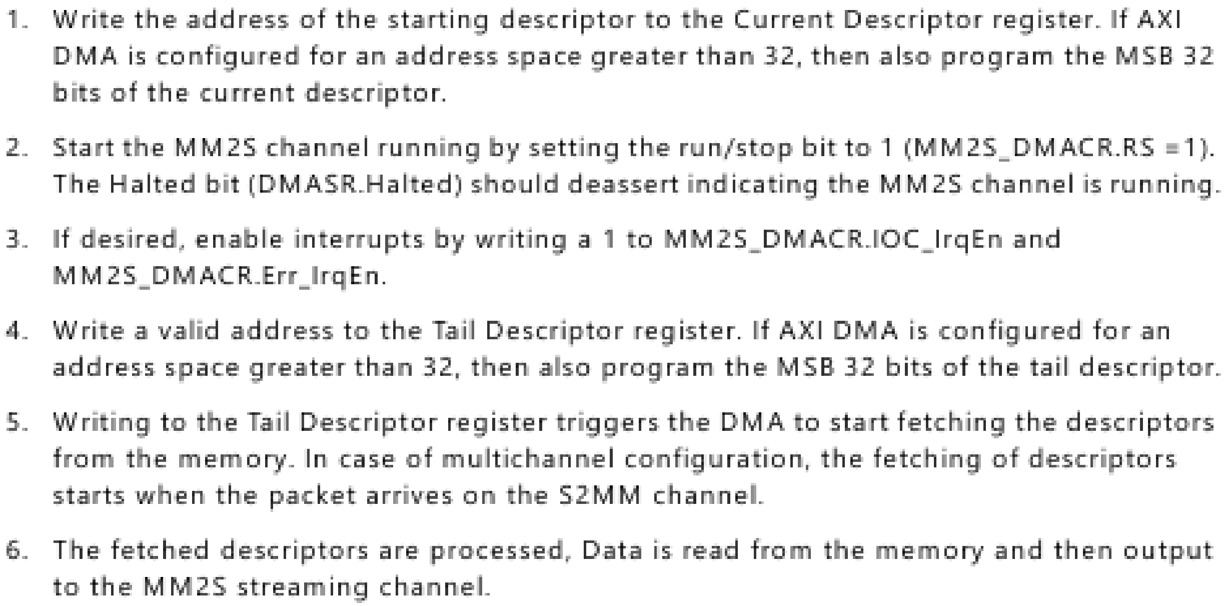


Table 2-26: **Descriptor Fields (Non-multichannel Mode)**

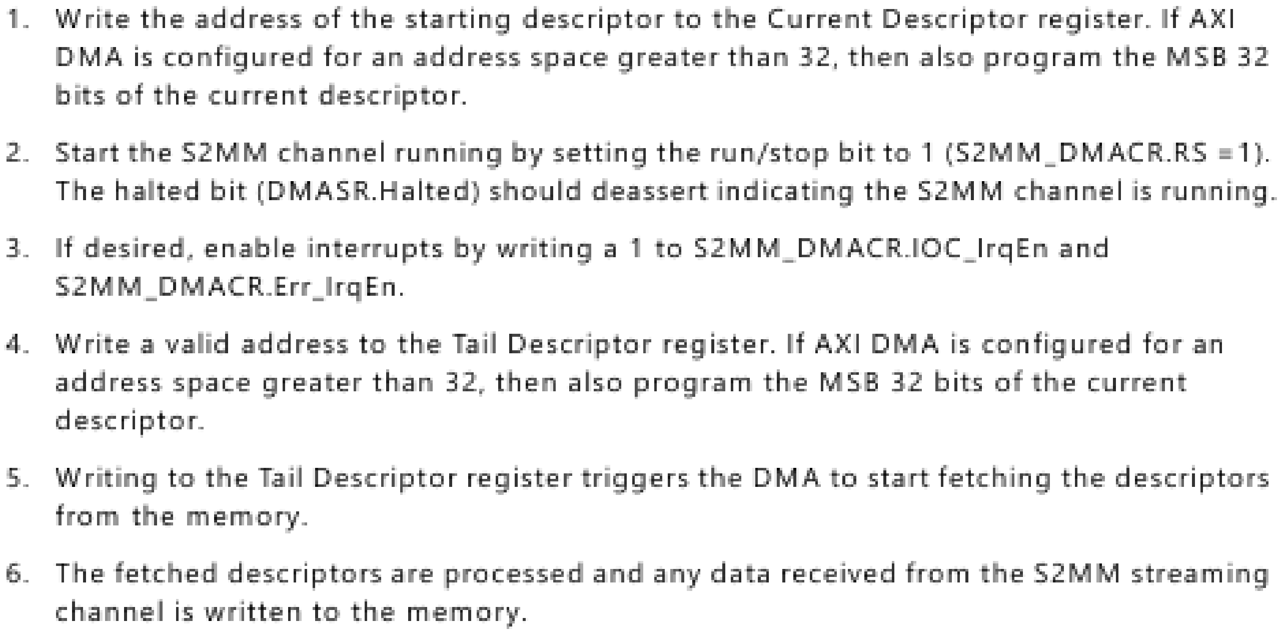
Address Space Offset(1)	Name	Description
00h	NXTDESC	Next Descriptor Pointer
04h	NXTDESC_MSB	Upper 32 bits of Next Descriptor Pointer
08h	BUFFER_ADDRESS	Buffer Address
0Ch	BUFFER_ADDRESS_MSB	Upper 32 bits of Buffer Address.
10h	RESERVED	N/A
14h	RESERVED	N/A
18h	CONTROL	Control
1Ch	STATUS	Status
20h	APP0	User Application Field 0 (2)
24h	APP1	User Application Field 1
28h	APP2	User Application Field 2
2Ch	APP3	User Application Field 3
30h	APP4	User Application Field 4

aleksei.rostov@protonmail.com

AXI DMA PROGRAMMING SEQUENCE



1. Write the address of the starting descriptor to the Current Descriptor register. If AXI DMA is configured for an address space greater than 32, then also program the MSB 32 bits of the current descriptor.
2. Start the MM2S channel running by setting the run/stop bit to 1 (MM2S_DMACR.RS =1). The Halted bit (DMASR.Halted) should deassert indicating the MM2S channel is running.
3. If desired, enable interrupts by writing a 1 to MM2S_DMACR.IOC_IrqEn and MM2S_DMACR.Err_IrqEn.
4. Write a valid address to the Tail Descriptor register. If AXI DMA is configured for an address space greater than 32, then also program the MSB 32 bits of the tail descriptor.
5. Writing to the Tail Descriptor register triggers the DMA to start fetching the descriptors from the memory. In case of multichannel configuration, the fetching of descriptors starts when the packet arrives on the S2MM channel.
6. The fetched descriptors are processed, Data is read from the memory and then output to the MM2S streaming channel.



1. Write the address of the starting descriptor to the Current Descriptor register. If AXI DMA is configured for an address space greater than 32, then also program the MSB 32 bits of the current descriptor.
2. Start the S2MM channel running by setting the run/stop bit to 1 (S2MM_DMACR.RS =1). The halted bit (DMASR.Halted) should deassert indicating the S2MM channel is running.
3. If desired, enable interrupts by writing a 1 to S2MM_DMACR.IOC_IrqEn and S2MM_DMACR.Err_IrqEn.
4. Write a valid address to the Tail Descriptor register. If AXI DMA is configured for an address space greater than 32, then also program the MSB 32 bits of the current descriptor.
5. Writing to the Tail Descriptor register triggers the DMA to start fetching the descriptors from the memory.
6. The fetched descriptors are processed and any data received from the S2MM streaming channel is written to the memory.