.

Thread Tracking Toolkit Show/Hide Account Look-Up Tool

Thread ID: 1225175 **Account:** TI EDUCATION TECHNOLOGY **Region:** FR7 **Company:** Texas Instruments

Thread Status (Internal only) Closed

Priority Normal

Assign j-gundavarapu@ti.cc

Email addresses entered above will receive a one-time email notifying them of assignment to this thread **and** will be automatically email subscribed to all subsequent replies.

Notify

Email addresses/lists entered above will receive a one-time email notifying them of this thread.

Responsible Organization --[PROCESSORS] SITARA MCU

Notes:

Submit Click "Submit" button to save any changes above.

This thread has been locked.
If you have a related question, please click the "Ask a related question" button in the top right corner. The newly created question will be automatically linked to this question.

TMS570LC4357: What is the correct mechanism to use when receiving a data stream of unknown size bursts on an SCI link?

Geoffrey Ficara *192.91.60.14* *Intellectual 1790 points* Texas Instruments

Part Number: TMS570LC4357

Hi Team,

I am contacting you because my customers have a problem with the use of the DMA on the TMS570LC4357 processor.

In our use case, they want to receive a serial data stream using the SCI component and transfer the data to RAM in masked time with DMA.

To do this, they program a transfer to a buffer in RAM with auto-initiation activated, and on the software side they empty the buffer like a circular buffer.
To know where the DMA is, they consult the Current Destination Address Register (CDADDR), but the value of the register does not seem to be updated systematically.

Highlight of the bug :

To highlight this, they used 2 buffers, the DMA switches from one buffer to the other when it reaches the end (via a BTC interrupt, they write a new control packet to point to the next buffer). they set a frame size of one byte and activate the FTC interrupt to know if a transfer has taken place or not.

What they observe is that when the DMA changes buffer and a transfer has taken place in the new buffer (validated via the FTC interrupt and the new data is in RAM), the CDADDR register always points to the end of the previous buffer. Worse, it can point to an old address during 2 or 3 FTC interrupts.

back to their use case :

In their operational case (with a buffer and auto-initiation), they have data bursts arriving on the serial. What they observe is that at the end of the burst, CDADDR points to the second last byte received and that is still the case, even several tens of milliseconds later. The CDADDR register will only be updated when the next burst starts to arrive.

About this register, the documentation (SPNU563A - March 2018) says: "These bits are only updated after a channel is arbitrated out of the priority queue." .
This is not extremely clear and suggests that it is not possible to use the CDADDR register to track reception as it happens, since it is only updated on non-predictable events.

So my question is, what is the correct mechanism to use when receiving a data stream of unknown size bursts on an SCI link?

Regards,

Geoffrey

9 months ago

jagadish gundavarapu *192.163.5.9* *9 months ago*

TI__Mastermind 35305 points

Hi Geoffrey,

> *Geoffrey Ficara said:*
>
> *So my question is, what is the correct mechanism to use when receiving a data stream of unknown size bursts on an SCI link?*

.

The SCI doesn't support any character time out interrupt. So, SCI with DMA could not change the packet size dynamically.

So, you have to implement some protocol in application level, like the transmitter should needs to tell the receiver about the packet size it is going to send(this packet length is fixed), and receiver can change the DMA packet settings for the new packet size. In this way you can receive variable size of data using UART with DMA. Once it receives unknown length packet again it should need to change DMA packet settings to the known length packet to receive next unknown length of the packet and should continue the process.

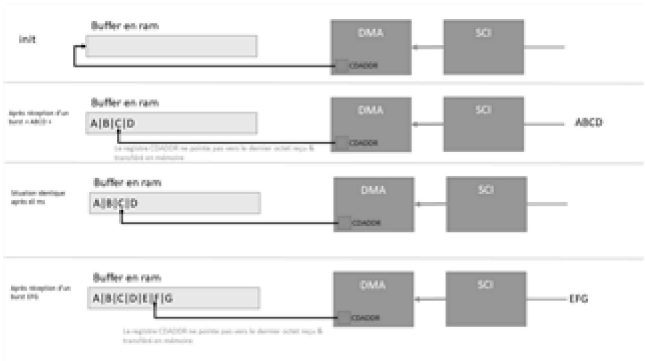
--

Thanks & regards,
Jagadish.

Geoffrey Ficara *192.91.60.14* *9 months ago* *in reply to* *jagadish gundavarapu* TI__Intellectual 1790 points

Hi Jagadish,

I don't think I understood your way, can you elaborate ?



Here is the description of the problem, at the end of a received burst, the CDADDR register doesn't point to the last value received, but to the one before. So I don't know the last value of a burst until the next burst arrives.

How does you wau solve this issue ?

Regards

Geoffrey

Geoffrey Ficara *192.91.60.14* *9 months ago* *in reply to* *Geoffrey Ficara* TI__Intellectual 1790 points

Hi,

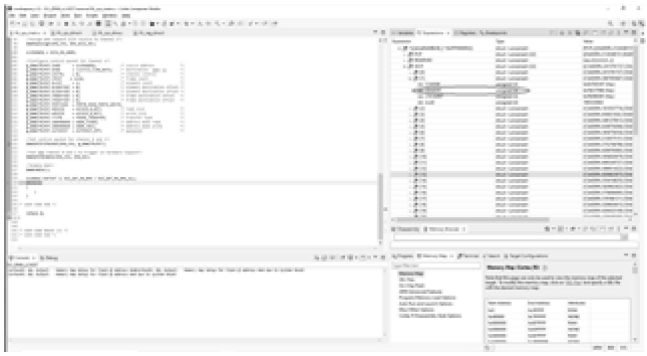
Any info ?

Geoffrey

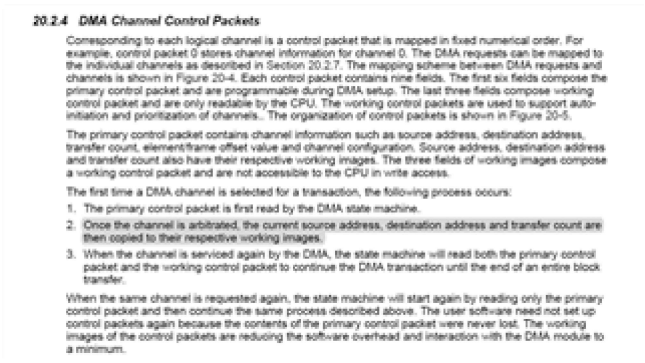
jagadish gundavarapu *192.163.5.9* *8 months ago* *in reply to* *Geoffrey Ficara* TI__Mastermind 35305 points

Hi Geoffrey,

> *Geoffrey Ficara said:*
>
> *Here is the description of the problem, at the end of a received burst, the CDADDR register doesn't point to the last value received, but to the one before*



The CDADDR register will not provide valid address of the last received value, this is because the CDADDR register value will get updated only when arbitration condition occurs.



20.2.4 DMA Channel Control Packets

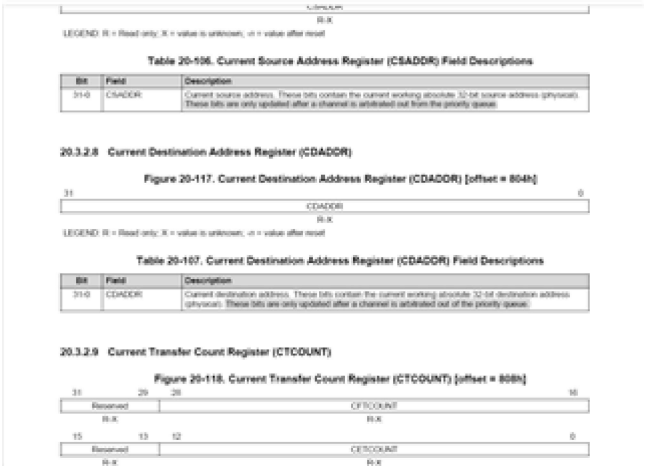
Corresponding to each logical channel is a control packet that is mapped in fixed numerical order. For example, control packet 0 stores channel information for channel 0. The DMA requests can be mapped to the individual channels as described in Section 20.2.7. The mapping scheme between DMA requests and channels is shown in Figure 20-4. Each control packet contains nine fields. The first six fields compose the primary control packet and are programmable during DMA setup. The last three fields compose working control packet and are only readable by the CPU. The working control packets are used to support auto-initiation and prioritization of channels.. The organization of control packets is shown in Figure 20-5.

The primary control packet contains channel information such as source address, destination address, transfer count, element/frame offset value and channel configuration. Source address, destination address and transfer count also have their respective working images. The three fields of working images compose a working control packet and are not accessible to the CPU in write access.

The first time a DMA channel is selected for a transaction, the following process occurs:

1. The primary control packet is first read by the DMA state machine.
2. Once the channel is arbitrated, the current source address, destination address and transfer count are then copied to their respective working images.
3. When the channel is serviced again by the DMA, the state machine will read both the primary control packet and the working control packet to continue the DMA transaction until the end of an entire block transfer.

When the same channel is requested again, the state machine will start again by reading only the primary control packet and then continue the same process described above. The user software need not set up control packets again because the contents of the primary control packet were never lost. The working images of the control packets are reducing the software overhead and interaction with the DMA module to a minimum.



R-X

LEGEND: R = Read only; X = value is unknown; -n = value after reset

Table 20-106. Current Source Address Register (CSADDR) Field Descriptions

Bit	Field	Description
31-0	CSADDR	Current source address. These bits contain the current working absolute 32-bit source address (physical). These bits are only updated after a channel is arbitrated out from the priority queue.

20.3.2.8 Current Destination Address Register (CDADDR)

Figure 20-117. Current Destination Address Register (CDADDR) [offset = 804h]

31	0
CDADDR	
R-X	

LEGEND: R = Read only; X = value is unknown; -n = value after reset

Table 20-107. Current Destination Address Register (CDADDR) Field Descriptions

Bit	Field	Description
31-0	CDADDR	Current destination address. These bits contain the current working absolute 32-bit destination address (physical). These bits are only updated after a channel is arbitrated out of the priority queue.

20.3.2.9 Current Transfer Count Register (CTCOUNT)

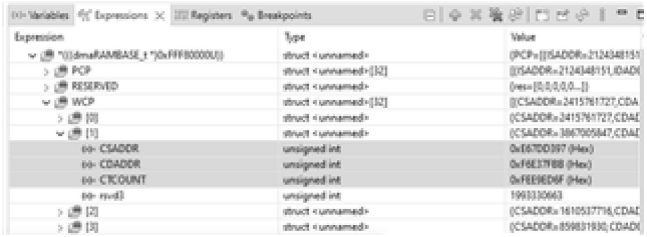
Figure 20-118. Current Transfer Count Register (CTCOUNT) [offset = 808h]

31	29	28	16
Reserved		CFTCOUNT	
R-X		R-X	
15	13	12	0
Reserved		CETCOUNT	
R-X		R-X	

For example:

If one channel trigger comes then DMA will initialize corresponding channel primary control packet information to the

either port A or port B registers, now DMA will move data and changes this port A registers for each element it shifts. If any high priority DMA channel triggered in between this process then this Port registers current configuration will get moved into the Working control packet at its arbitration, so that now DMA can move new channel primary control packet information to the Port registers.



Now after completion of this high priority channel execution by DMA, then DMA will again get the working control packet information of previous channel and it copies that into the port registers to resume the previous operation.

So, because of this reason i won't recommend customer to use the CDADDR register for to get unknown bytes shifted by DMA.

--

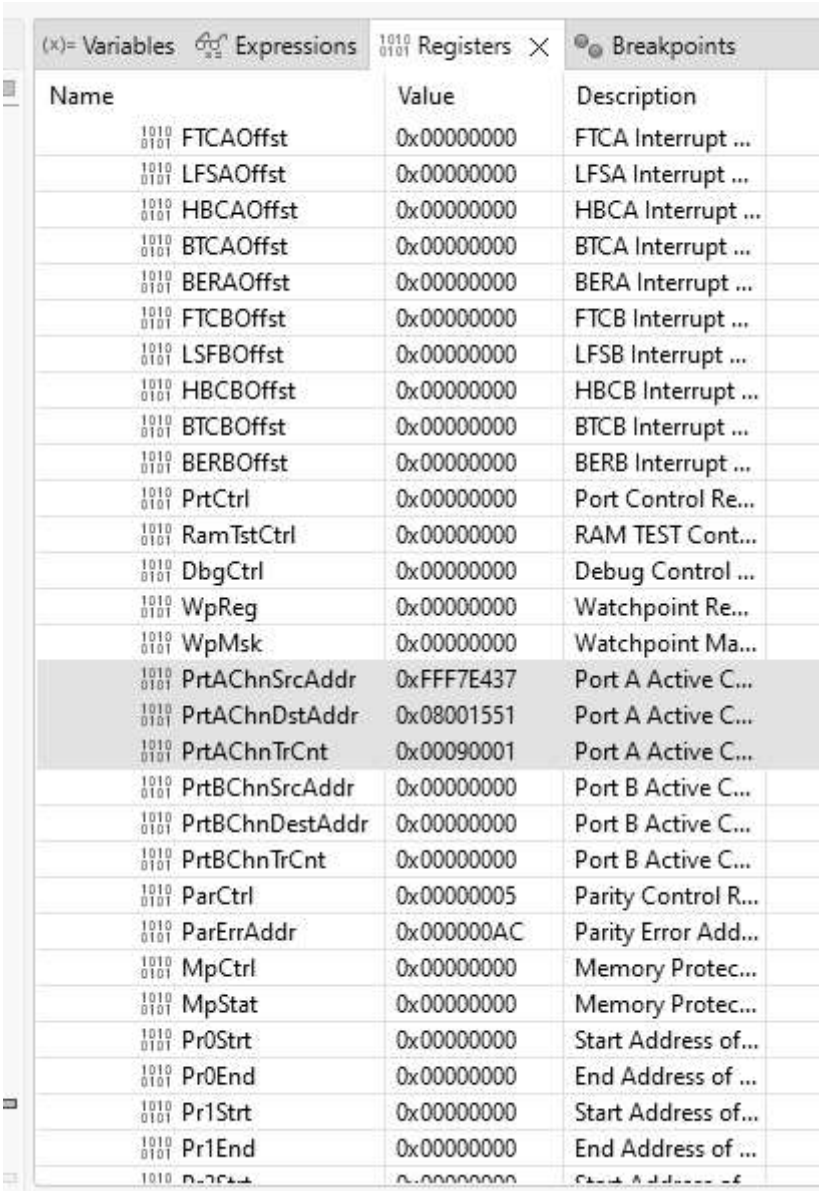
Thanks & regards,
Jagadish.

jagadish gundavarapu *192.163.5.9* *8 months ago* *in reply to* *jagadish gundavarapu* TI__Mastermind 35305 points

Hi Geoffrey,

So, my suggestion here is that

Ask customer to use Active port registers or FIFO registers i mentioned.



--

Thanks & regards,
Jagadish.

Geoffrey Ficara *192.91.60.15* *8 months ago* *in reply to* *jagadish gundavarapu* TI__Intellectual 1790 points

Hi Jagadish,

Thank you for the help. Can you just elaborate, how will active port registers or FIFO registers solve the problem ?

Regards

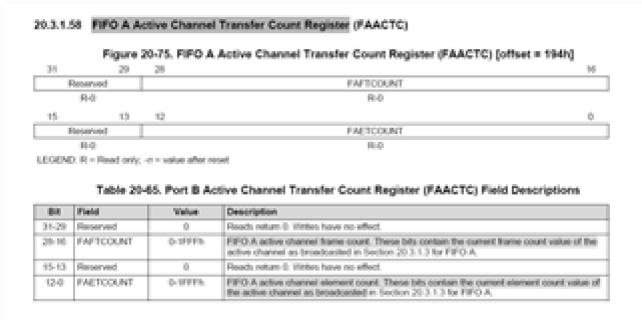
Geoffrey

jagadish gundavarapu *192.163.5.9* *8 months ago in reply to Geoffrey Ficara* TI__Mastermind 35305 points

Hi Geoffrey,

FIFO A Active Channel Transfer Count Register:

This register value get decrement each time a element/frame transferred by DMA, so i guess the customer can use this register value to calculate number of elements/frames transferred by DMA as he already has data of how many elements/frames he willing to transfer.



20.3.1.58 **FIFO A Active Channel Transfer Count Register** (FAACTC)

Figure 20-75. FIFO A Active Channel Transfer Count Register (FAACTC) [offset = 194h]

31 29	28 16
Reserved	FAFTCOUNT
R-0	R-0
15 13	**12 0**
Reserved	FAETCOUNT
R-0	R-0

LEGEND: R = Read only; -n = value after reset

Table 20-65. Port B Active Channel Transfer Count Register (FAACTC) Field Descriptions

Bit	Field	Value	Description
31-29	Reserved	0	Reads return 0. Writes have no effect.
28-16	FAFTCOUNT	0-1FFFh	FIFO A active channel frame count. These bits contain the current frame count value of the active channel as broadcasted in Section 20.3.1.3 for FIFO A.
15-13	Reserved	0	Reads return 0. Writes have no effect.
12-0	FAETCOUNT	0-1FFFh	FIFO A active channel element count. These bits contain the current element count value of the active channel as broadcasted in Section 20.3.1.3 for FIFO A.

--

Thanks & regards,
Jagadish.

.