Design of turbo decoder for collision communication receiver

Mehdi Sadeghzadeh

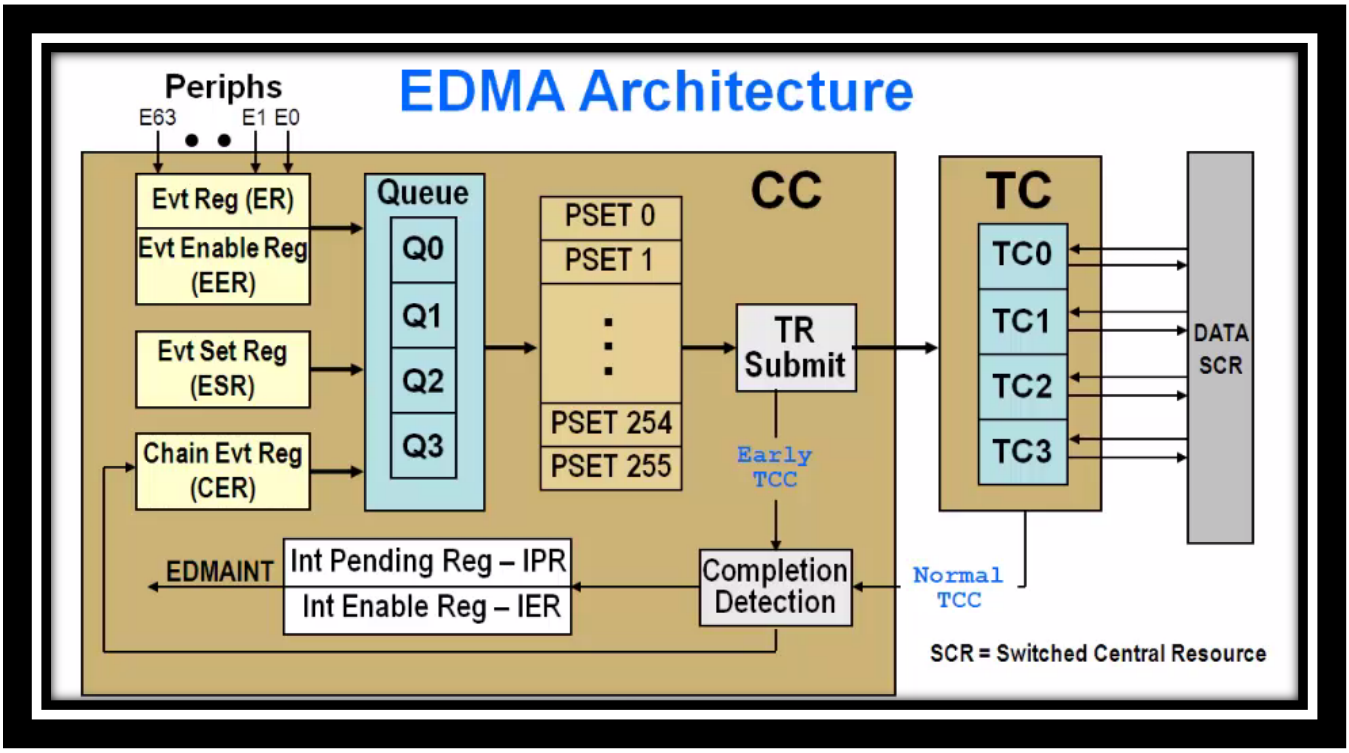
Some Important Facts

Problem:

Let's say we have some block which we want to decode them using turbo decoders (TCP3Ds) in C6670. We want to design a system to decode these blocks as efficient as possible.

EDMA Architecture:

In the following figure, we see the architecture for the EDMA with 4 queues. Each queue is associated with one transfer controller (TC):



Facts of EDMA:

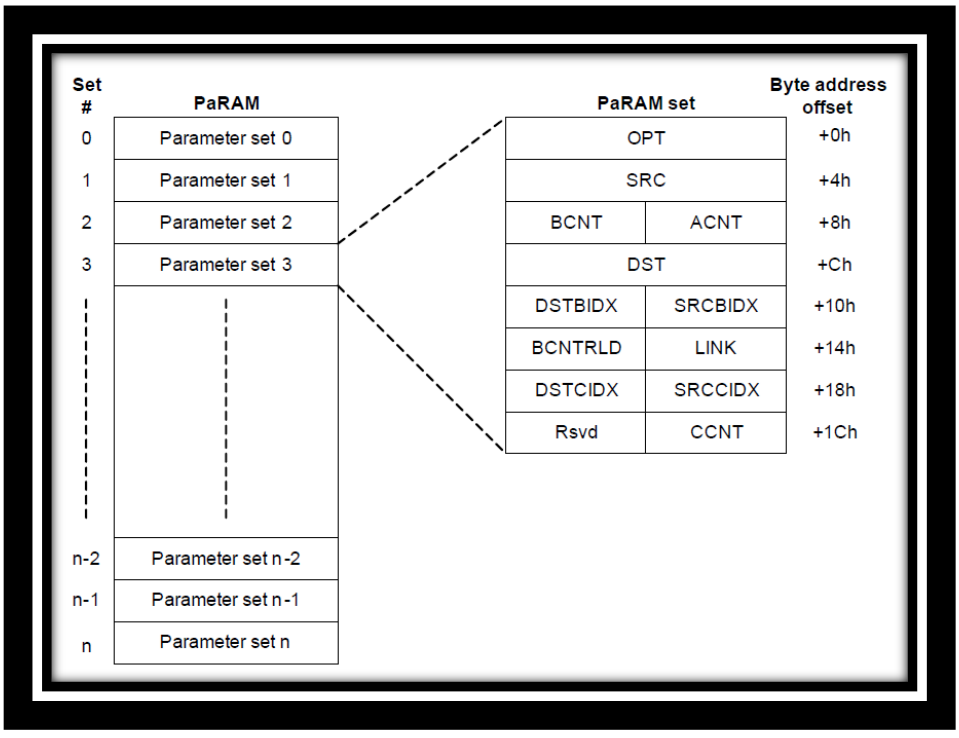
- Different TCs have different FIFO size and BUS width

Facts of C6670:

- There are 3 EDMA (EDMA3) and 3 TCPD (TCP3D) in C6670
- The three TCP3D are named as TCP3D A, TCP3D B, and TCP3D C
- The three EDMA3 are named as EDMA3 CC0, EDMA3 CC1, and EDMA3 CC2
- The EDMA3 CC0 has just two transfer controls (TCs) which each of them has FIFO size of 1024 bytes (The reason I mentioned FIFO size is that for the optimization purpose, we have to transfer bigger blocks with bigger FIFOs)
- The EDMA3 CC1 has four transfer controls (TCs) which TC0 and TC2 have FIFO size of 1024 bytes and TC1 and TC3 have FIFO size of 512 bytes.
- The EDMA3 CC1 has four transfer controls (TCs) which TC0 and TC3 have FIFO size of 1024 bytes and TC1 and TC2 have FIFO size of 512 bytes.

How to use EDMA3

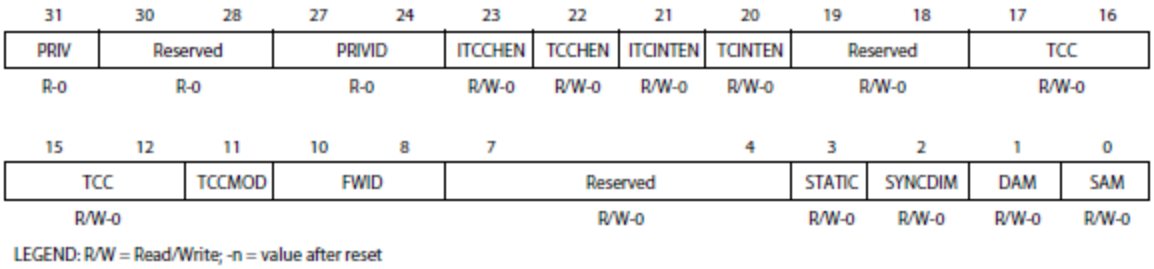
To use EDMA3, we need to assign related parameters to the parameter RAM (PaRAM) sets. In the following there is a picture shows the PaRAM sets:



There are n numbers of sets for each EDMA3 and n depends to the different EDMA3 varies. As shown in the figure, there are some parameters in PaRAM set needs to be explained before moving to the next part.

- OPT = Transfer option configuration

Figure 2-8 Channel Options Parameter (OPT)



31	30 – 28	27 – 24	23	22	21	20	19 – 18	17 – 16
PRIV	Reserved	PRIVID	ITCCHEN	TCCHEN	ITCINTEN	TCINTEN	Reserved	TCC
R-0	R-0	R-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

15 – 12	11	10 – 8	7 – 4	3	2	1	0
TCC	TCCMOD	FWID	Reserved	STATIC	SYNCDIM	DAM	SAM
R/W-0			R/W-0	R/W-0	R/W-0	R/W-0	R/W-0

LEGEND: R/W = Read/Write; -n = value after reset

Some of the important options are described here:

- TCCHEN = transfer complete chaining enable
  - We use this feature to chain some PaRAM sets after each other and run them after each other. This way DSP is not going to tell EDMA3, what the next PaRAM set should be run.
- TCINTEN = Transfer complete interrupt enable
  - We are going to use the feature in our design and we will interrupt the DSP(s) to run a function after completing the HD out PaRAM set. I will describe it later.
- DAM = Destination address mode (linking or chaining)
- FWID = FIFO Width
- PRIV = Privilege level

- SRC = Channel source address, it is the address which the EDMA3 will put its pointer there to read.
- ACNT = Count for the 1st Dimension
- BCNT = Count for the 2nd Dimension
- CCNT = Count for the 3rd Dimension

ACNT, BCNT, and CCNT are used to manage how much data to transfer. These conuts (ACNT, BCNT, and CCNT) are going to tell EDMA3 how much data to read from the place the EDMA3 puts the pointer (SRC).

- DST = Channel destination address, it is the address which the EDMA3 will put its pointer there to write.
- SRCBIDX = Source BCNT index, it is going to tell the EDMA3 where to puts the pointer after reading the ACNT of the data
- SRCCIDX = Source CCNT index, it is going to tell the EDMA3 where to puts the pointer after reading the BCNT of the data
- DSTBIDX = Destination BCNT index, it is going to tell the EDMA3 where to puts the pointer after writing the ACNT of the data

- SRCCIDX = Source BCNT index, it is going to tell the EDMA3 where to puts the pointer after writing the BCNT of the data
- Link = Link address, it is going to tell the EDMA3 where is the next PaRAM set has to run if the linking parameter is set to 1 in the OPT.

Design for C6670

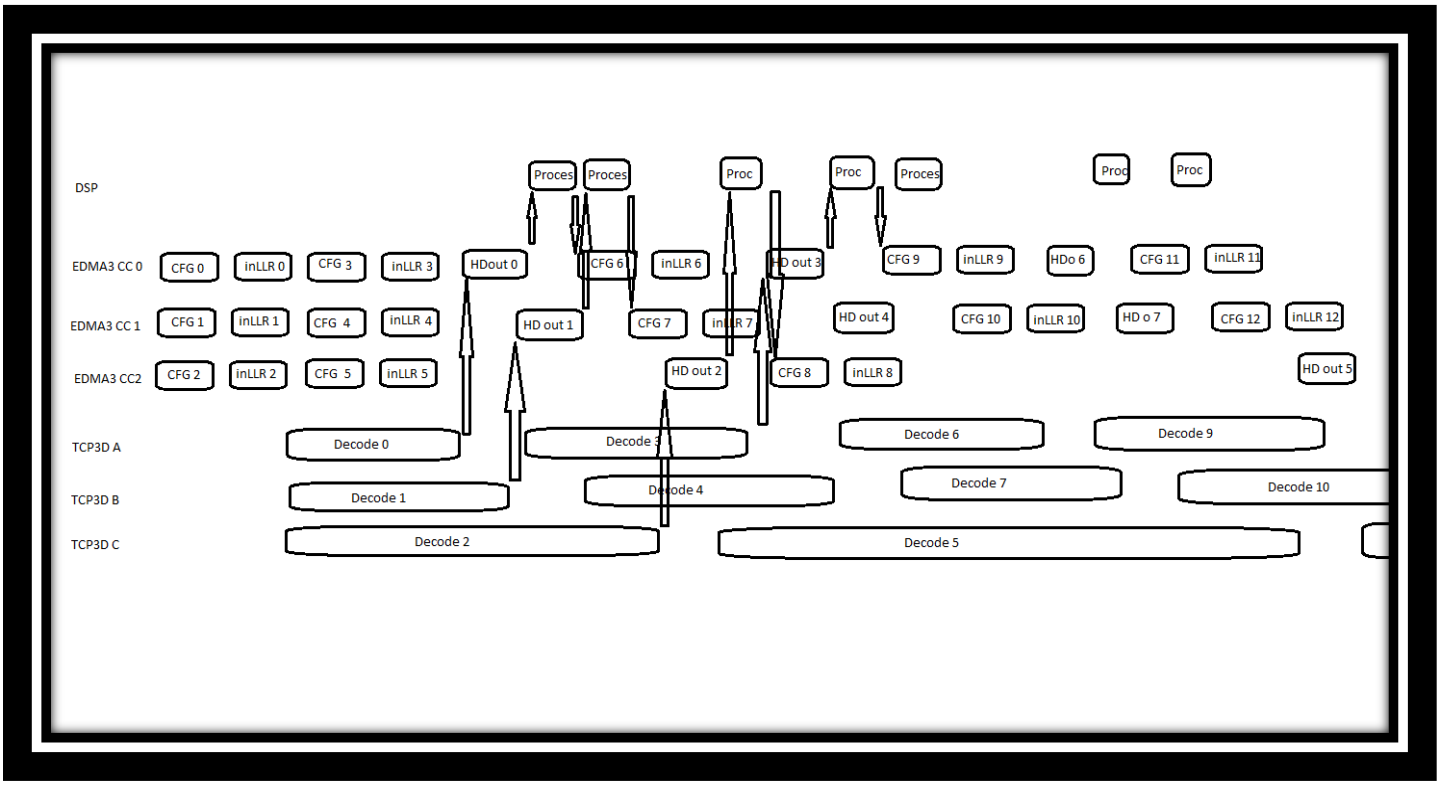
The whole idea is to use all three EDMA3, three TCP3D, and just one DSP core and use this core as less as possible. I assumed all the blocks for decoding are stored in Multicore Shared Memory (MSM) (2MB and the address starts from 0x0C00 0000 to 0x0C1F FFFF) and the decoded blocks will be stored in MSM. Each EDMA3 will use each CorePac L2 SRAMA as cache (CorePac0 L2 SRAM = 1MB starts from 0x1080 0000 to 0x108F FFFF, CorePac1 L2 SRAM = 1MB starts from 0x1180 0000 to 0x118F FFFF, CorePac2 L2 SRAM = 1MB starts from 0x1280 0000 to 0x128F FFFF, CorePac3 L2 SRAM = 1MB starts from 0x1380 0000 to 0x138F FFFF; we need just three cache). We are going to use double buffer to remove the delay of transferring the PaRAM sets to the EDMA3. As I mentioned earlier, to transfer data, DSP need to configure some PaRAM sets and initialize the EDMA3.For decoding any block; there are three PaRAM sets which should be configured:

1. CFG box = This box has the PaRAM set of the configuration parameters (tail bits, interleaver, size, SW0 length, max iteration, and stopping criterias) of the block is supposed to be decoded.
2. inLLR box = This box has the PaRAM set of the input LLRs (Systematic, Parity 0, and Parity 1 bits) of the block whose CFG has been transferred.
3. HD out box = This box has the PaRAM set of the output hard decision (HD) coming from TCP3D

What needs to be done in DSP:

- Process box = In the DSP we need to define a function which update the CFG, inLLR, HD out for the EDMA has a free buffer (ping or pong) which is going to be the output. The input will vary based on the developer! But, based on the history of the PaRAM sets has been set, we can configure the new PaRAM sets.

The proposed design is as follows:



Description of the Design:

- DSP initialize the PaRAM sets (CFGs, inLLRs, and HD out) for ping buffers of all three EDMAs
- These PaRAMs for each EDMA has been chained and they will be run after each other (We can configure chaining in the OPT)
- While EDMA is waiting for the output from the TCP3D (HD out is pending), DSP initialize the PaRAM sets for the pong buffers.
- When the HD out transfer for each EDMA has been completed, the interrupt to DSPs will be occurred (TCINTEN in OPT of all HD out PaRAM sets is set to 1)
- After Interrupting the DSP, DSP will run a function which should update the PaRAM sets needed for the next available block and decide these PaRAM sets is for ping buffer or pong buffer of which EDMA!
- At the last interrupt, the DSP decide not to transmit any PaRAM sets.

Optimization

- Spread out the transfers among all the Q’s
- Place larger transfers on the TCs with large FIFOs
- Match transfer sizes to bus widths
- Parallel coding in the DSP (I am not sure if we need it!)

References:

- EDMA3 controller user guide
- TMS320C6670 user guide