H265 Encoder Codec

API Specification

05/01/2017
Revision 2.0

Revision History

The following table shows the revision history for this document.

Date	Version	Revision	Author
2017.1.20	SOC initial Release	1.0	Blake
2017.5.01	Changed the name of Reg 0x65 Added Version control Reg 0x66	2.0	Blake

Introduction

The API interface has 256 register (16 bit for writing and 32 bits for reading). Each register has an address to allow user-access. The registers accept command code for controlling the operation of the features within the FPGA. The registers may be read to acquire status and debugging information.

Some registers are allocated for user configurations and the rest are reserved for system functions such as debugging. The following table(s) specifies the address, definition, and access right (read/write) for the user-accessible registers. Note: registers of any API_TYPE that are NOT in the table are not designed for user access. If such a register is written to, the decoder may stop running. To recover, the core must be re-downloaded into the FPGA.

Control the API inside the FPGA

If it is necessary to control the API via the modules inside the FPGA, the ports of the API are available. Details, including the clock frequency, are provided in the corresponding integration manual.

Control the API outside the FPGA

If it is necessary to control the Encoder from outside the FPGA, the address, date, and R/W pins can be routed to the FPGA I/O pins. The desired FPGA I/O pins for the API may be selected or the SOC default pins may be used. An UART interface is available, upon request. The interface provides access to the API via uart_tx and uart_rx pins.

H265 CODEC Registers

Addr (HEX)	Information	Access
92	Bit[2:0]: Multi-Channel/Time Sharing API Core Selection - 0 – Selects all cores for writing - 1-7 = Core # - Note only 1 core can be read at a time. A value of 0 should not be used for reading registers.	R/W
The following registers are unique to each time-shared core (when used)		
0	Bit[0]: Enable H265 Encoder (0=Pause, 1=Run)	R/W

3	Bit[1:0]: Chroma Format - "01" = 4:2:0 - "10" = 4:2:2 (Requires Special Licence)	R/W
4	Bit[1:0]: Bit Depth - "00" = 8 bit - "10" = 10 bit (Requires Special Licence) This is supported in only version higher than 10-bit version but not in 8-bit version. Please check your order information.	R/W
17	Bit[0]: Constrained Intra Prediction - '1' = Intra prediction formed from Intra Neighbour blocks only - '0' = Intra prediction formed form any block type With '1', the refresh speed is faster but bit-rate higher.	R/W
23	Bit[0]: Disable Deblocking Filter - '1' = Disabled - '0' = Enabled - * Some build may have the deblocking feature permanently disabled *This is just a filtering, so there is no big difference in most case compared to without the filtering.	R/W
35-3F	See Note 1	R/W
The following registers are shared between all cores even when muli/time-sharing cores are used		
52	Bit[15:8]: Random Intra Block Size X Bit[7:0]: Random Intra Block Size Y - 1 Block size is 16x16 pixels - This register only has an effect when 'Random Intra Mode 0x5B' is enabled.	R/W

5B	Bit[0]: Random Intra Mode Enable - '1' = Enabled - '0' = Disabled - When Enabled and not Intra frames exist, 'Constrained Intra Prediction 0x17' must be enabled for this to function as intended.	R/W
5F	Bit[7:0]: Disable Video Channel - Can Disable(1)/Enable(0) each video encoding channel - *For Time-Sharing Encoder Core only* - Example - "00001010" = Cores 1 and 3 are disabled - Note bit 0 has no effect	R/W
60	Bit[31:24]: Encoder Pipeline Number Bit[5]: P frame is supported Bit[4]: Deblocking filter is supported Bit[2]: 4K encoding is supported	R
61	Bit[31:0]: H265 Build Date	R
62	Bit[31:0]: H265 Build Time Stamp	R
64	H265 Encoder core production number(used by SOC only) Value: 0x53000200 for H265 encoder	R
65	H265 git date (used by SOC only) Example: 20170501	R
66	H265 core git version: Bit[15:8]: major version Bit [7:0]: minor version The start git version is 2.0 with value 0x0200.	
70	Bit[15]: use_I_frame_balace, Only when use_I_frame_balace ='1', the bit[2:0], Intra_frame_scale takes effect. Bit[14]: Reserved, don't change, W1RB* Bit[13]: Reserved, don't change, W1RB* Bit[12]: control_i_slice: control bit-rate for I slice. When '1', the I slice is not in countered in the bit-rate control.	R/W*

	Bit[11]: control_Intra_mb_adj, W1RB*, default '1' Bit[10]: Reserved, don't change, W1RB* Bit[9]: control_Inter_mb_adj, W1RB*, default '1' Bit[8]: Reserved, don't change, W1RB* Bit[7]: Reserved, don't change, W1RB* Bit[6]: Reserved, don't change, W1RB* Bit[5]: Reserved, don't change, W1RB* Bit[4]: Reserved, don't change, W1RB* Bit[3]: Reserved, don't change, W1RB* Bit[2:0]: Intra_frame_scale, the I frame size scale than the P frame when use_I_frame_balace='1'. *Basically, the default value should be the best configuration But the user is allowed to change it. *To change I-slice quality, use register 0x55.	
79	Bit[6:0]: current macroblock Qp value. Informative.	R
7B	Bit[5:0]: Minimum QP Value *Note: There is no register for Max-QP value because it is not necessary to have one.	R/W

7F	Bit[15]: Intra(I) Constant QP Mode Bit[14:8]: Intra(I) Constant QP Value-26 Bit[7]: Inter(P) Constant QP Mode Bit[6:0]: Inter(P) Constant QP Value-26 - Bit[15] / Bit[7]: Write '1' to the bit to clear the bit. When reading, if bit[15]='1', the intra block uses bit[14:8]+26 as QP vlue. When reading, if bit[15]='0', the encoder use dynamic QP controlled by register 0x97 to encode intra blocks. When reading, if bit[7]='1', the inter block uses bit[6:0]+26 as QP vlue. When reading, if bit[7]='0', the encoder use dynamic QP controlled by register 0x97 to encode inter blocks. For example: write 0515H to this register, the register read value will be: 8595H. So, the encoder use QP=31(=26+5) to encode intra(I) blocks, and QP=47(=26+0x15) to encode inter(P) blocks. To clear the constant QP, write '1' to bit[7] or bit[15].	R/W
	For example, write 8080H to this register, the register read value will be: 0000H, means the constant QP was cleared. - Bitrate control must be disabled for proper operation (0x97=0) - If failed to change this value, try to write again until succeed.	

Note:

(1) All the default values depend on the build version. So you may have different values for defaults.

(2) W1RB*: Write 1 Reverse Bit. The bit is reversed by writing logic '1'.

Note 1:

The registers 35~3F control the encoding frame type or the Group of Pictures (GOP). The encoder scans the 11 registers in order before encoding next frame/frames/unit.

For each register in the range of 35-3F:
Bit[15:8] indicates the repeat time of the frame type
Bit[7:0] indicates the unit/frame/sequence type as below.

0x00 – NULL. The encoder will skip to next register without encode any unit/frame.
0xE5 – Intra Picture (I-Frame)
0x65 – Temporary Intra Picture Request – Inserts only 1 Intra Picture. Will be cleared to 0x00 by the core after this Intra frame is encoded.
0x41 – Inter (P) Picture (P frame)

Example:

REG 0x35 = 0x00E5
REG 0x36 = 0x0F41
REG 0x37 = 0x0000
….
REG 0x3F = 0x0000

Encoder running sequence

Here is how the encoder runs. The encoder reads the current position and finishes it. For the example above:

1. The encoder reads the position 35 which is 00E5, and start encoding the next frame as I frame.
2. The encoder reads the position 36, and begins to encode 15 P frames.
3. The encoder reads position 37 which is 0000, so no frame for this position.
4. The encoder reads position 38 until 3F, all 0000, so no frame or unit encoded for these positions.
5. The encoder backs to 35 and repeats the steps 1-6.

While the encoder is encoding any frame, user could change any of the 11 registers. But until the encoder reads the changed position again, it does not take effect.

Examples

Example 1: The regular GOP:

	Address(hex)	Value(hex)
35		00E5
36		0F41

37-3F 0000

The GOP sequence will be like this: I-Frame + 15 P frames

Example 2: The GOP with I slice mode:

Address(hex) Value(hex)
35 05E5
36-3F 0000

The GOP sequence will be like this:
5 Intra frame only

Example 3: The long:

Address(hex) Value(hex)
35 00E5
36 C841
37-3F 0000

The GOP sequence will be like this:
One I-Frame + 200 P frames