

Wersja (major_minor) 0x3_0xDF

Mapa rejestrów FPGA:

<u>Nazwa</u>	<u>Opis</u>	<u>Start</u>	<u>Koniec</u>	<u>Rozmiar</u>
<u>reg_axi_0</u>	<u>Rejestry kontrolne</u>	<u>0x4000_0000</u>	<u>0x4000_0FFF</u>	<u>4K</u>
<u>axi_bram_ctrl_1</u>	<u>Kontroler bram</u>	<u>0x4000_1000</u>	<u>0x4000_1FFF</u>	<u>4K</u>
<u>reg_axi_1</u>	<u>Rejestr axi_1</u>	<u>0x4000_2000</u>	<u>0x4000_2FFF</u>	<u>4K</u>
<u>reg_axi_meas</u>	<u>Rejestr measure</u>	<u>0x4000_3000</u>	<u>0x4000_3FFF</u>	<u>4K</u>
<u>reg_axi_pll</u>	<u>Rejestr pll</u>	<u>0x4000_4000</u>	<u>0x4000_4FFF</u>	<u>4K</u>
<u>reg_chan_0</u>	<u>Rejestr chan_0</u>	<u>0x4001_0000</u>	<u>0x4001_0FFF</u>	<u>4K</u>
<u>reg_chan_1</u>	<u>Rejestr chan_1</u>	<u>0x4001_1000</u>	<u>0x4001_1FFF</u>	<u>4K</u>
<u>reg_chan_2</u>	<u>Rejestr chan_2</u>	<u>0x4001_2000</u>	<u>0x4001_2FFF</u>	<u>4K</u>
<u>reg_chan_3</u>	<u>Rejestr chan_3</u>	<u>0x4001_3000</u>	<u>0x4001_3FFF</u>	<u>4K</u>
<u>reg_chan_4</u>	<u>Rejestr chan_4</u>	<u>0x4001_4000</u>	<u>0x4001_4FFF</u>	<u>4K</u>
<u>reg_chan_5</u>	<u>Rejestr chan_5</u>	<u>0x4001_5000</u>	<u>0x4001_5FFF</u>	<u>4K</u>
<u>reg_chan_6</u>	<u>Rejestr chan_6</u>	<u>0x4001_6000</u>	<u>0x4001_6FFF</u>	<u>4K</u>
<u>reg_chan_7</u>	<u>Rejestr chan_7</u>	<u>0x4001_7000</u>	<u>0x4001_7FFF</u>	<u>4K</u>
<u>reg_chan_8</u>	<u>Rejestr chan_8</u>	<u>0x4001_8000</u>	<u>0x4001_8FFF</u>	<u>4K</u>
<u>reg_chan_9</u>	<u>Rejestr chan_9</u>	<u>0x4001_9000</u>	<u>0x4001_9FFF</u>	<u>4K</u>
<u>reg_chan_10</u>	<u>Rejestr chan_10</u>	<u>0x4001_A000</u>	<u>0x4001_AFFF</u>	<u>4K</u>
<u>reg_chan_11</u>	<u>Rejestr chan_11</u>	<u>0x4001_B000</u>	<u>0x4001_BFFF</u>	<u>4K</u>
<u>reg_chan_12</u>	<u>Rejestr chan_12</u>	<u>0x4001_C000</u>	<u>0x4001_CFFF</u>	<u>4K</u>
<u>reg_chan_13</u>	<u>Rejestr chan_13</u>	<u>0x4001_D000</u>	<u>0x4001_DFFF</u>	<u>4K</u>
<u>reg_chan_14</u>	<u>Rejestr chan_14</u>	<u>0x4001_E000</u>	<u>0x4001_EFFF</u>	<u>4K</u>
<u>reg_chan_15</u>	<u>Rejestr chan_15</u>	<u>0x4001_F000</u>	<u>0x4001_FFFF</u>	<u>4K</u>
<u>reg_chan_16</u>	<u>Rejestr chan_16</u>	<u>0x4002_0000</u>	<u>0x4002_0FFF</u>	<u>4K</u>
<u>reg_chan_17</u>	<u>Rejestr chan_17</u>	<u>0x4002_1000</u>	<u>0x4002_1FFF</u>	<u>4K</u>
<u>reg_chan_18</u>	<u>Rejestr chan_18</u>	<u>0x4002_2000</u>	<u>0x4002_2FFF</u>	<u>4K</u>
<u>reg_chan_19</u>	<u>Rejestr chan_19</u>	<u>0x4002_3000</u>	<u>0x4002_3FFF</u>	<u>4K</u>
<u>reg_chan_20</u>	<u>Rejestr chan_20</u>	<u>0x4002_4000</u>	<u>0x4002_4FFF</u>	<u>4K</u>
<u>reg_chan_21</u>	<u>Rejestr chan_21</u>	<u>0x4002_5000</u>	<u>0x4002_5FFF</u>	<u>4K</u>
<u>reg_chan_22</u>	<u>Rejestr chan_22</u>	<u>0x4002_6000</u>	<u>0x4002_6FFF</u>	<u>4K</u>
<u>reg_chan_23</u>	<u>Rejestr chan_23</u>	<u>0x4002_7000</u>	<u>0x4002_7FFF</u>	<u>4K</u>
<u>reg_chan_24</u>	<u>Rejestr chan_24</u>	<u>0x4002_8000</u>	<u>0x4002_8FFF</u>	<u>4K</u>
<u>reg_chan_25</u>	<u>Rejestr chan_25</u>	<u>0x4002_9000</u>	<u>0x4002_9FFF</u>	<u>4K</u>
<u>reg_chan_26</u>	<u>Rejestr chan_26</u>	<u>0x4002_A000</u>	<u>0x4002_AFFF</u>	<u>4K</u>
<u>reg_chan_27</u>	<u>Rejestr chan_27</u>	<u>0x4002_B000</u>	<u>0x4002_BFFF</u>	<u>4K</u>
<u>reg_chan_28</u>	<u>Rejestr chan_28</u>	<u>0x4002_C000</u>	<u>0x4002_CFFF</u>	<u>4K</u>
<u>reg_chan_29</u>	<u>Rejestr chan_29</u>	<u>0x4002_D000</u>	<u>0x4002_DFFF</u>	<u>4K</u>
<u>reg_chan_30</u>	<u>Rejestr chan_30</u>	<u>0x4002_E000</u>	<u>0x4002_EFFF</u>	<u>4K</u>
<u>reg_chan_31</u>	<u>Rejestr chan_31</u>	<u>0x4002_F000</u>	<u>0x4002_FFFF</u>	<u>4K</u>
<u>axi_bram_ctrl_0</u>	<u>Kontroler bram</u>	<u>0x4004_0000</u>	<u>0x4005_FFFF</u>	<u>128K</u>
<u>axi_gpio_0</u>	<u>Gpio led oraz switch</u>	<u>0x4120_0000</u>	<u>0x4120_FFFF</u>	<u>64K</u>
<u>axi_bram_ctrl_2</u>	<u>Kontroler bram</u>	<u>0x4200_0000</u>	<u>0x4200_3FFF</u>	<u>16K</u>

reg_axi_0

<u>Nazwa</u>	<u>Adres</u>	<u>Bit</u>	<u>Opis</u>	<u>Default</u>	<u>Tryb</u>
<u>Major ver</u>	<u>0x00</u>	<u>31:0</u>	<u>Wersja buildu major</u>	<u>0x3</u>	<u>RO</u>
<u>Major ver</u>	<u>0x04</u>	<u>31:0</u>	<u>Wersja buildu minor</u>	<u>0xDF</u>	<u>RO</u>
<u>Test reg</u>	<u>0x08</u>	<u>31:29</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>

		<u>28:24</u>	<u>Ilość kanałów offline/online -1</u>	<u>0x0</u>	<u>RO</u>
		<u>23:21</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>20:16</u>	<u>Aktualny kanał offline, 0 – 31 wskazuje odpowiedni kanał</u>	<u>0x0</u>	<u>RW</u>
		<u>15:0</u>	<u>Rejestr testowy, wpisana wartość zostaje zanegowana</u>	<u>0x0</u>	<u>RW</u>
<u>Transfer size</u>	<u>0x0c</u>	<u>31:0</u>	<u>Wielkość transferu do DDR (w słowach 4B)</u>	<u>0x8000</u>	<u>RW</u>
<u>Start channel</u>	<u>0x10</u>	<u>31:28</u>	<u>Ilość kanałów fizycznych (ilość układów max)</u>	<u>0x3</u>	<u>RO</u>
		<u>27:25</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>24:19</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>18:16</u>	<u>Stop write online channel (max 0, max 1, max 2 data stream) – „001” start max 0 złącze JD, „010” start max 1 stała wartość, „100” start max 2 stała wartość</u>	<u>0x0</u>	<u>WO</u>
		<u>15:13</u>	<u>Start write online channel (max 0, max 1, max 2 data stream) – „001” stop max 0 złącze JD, „010” stop max 1 stała wartość, „100” stop max 2 stała wartość</u>	<u>0x0</u>	<u>WO</u>
		<u>12</u>	<u>Done memory read ddr</u>	<u>0</u>	<u>W1C</u>
		<u>11</u>	<u>Busy memory read ddr</u>	<u>0</u>	<u>RO</u>
		<u>10</u>	<u>Busy online channel</u>	<u>0</u>	<u>RO</u>
		<u>9</u>	<u>Reset NCO</u>	<u>0</u>	<u>WO</u>
		<u>8</u>	<u>Busy DDR</u>	<u>0</u>	<u>RO</u>
		<u>7</u>	<u>Busy bram</u>	<u>0</u>	<u>RO</u>
		<u>6</u>	<u>Flaga done DDR</u>	<u>0</u>	<u>W1C</u>
		<u>5</u>	<u>Flaga done bram</u>	<u>0</u>	<u>W1C</u>
		<u>4</u>	<u>Start read channel DDR</u>	<u>0</u>	<u>WO</u>
		<u>3:0</u>	<u>Start write channel</u>	<u>0x0</u>	<u>WO</u>
<u>Counter Latch MSB</u>	<u>0x14</u>	<u>31:0</u>	<u>63:32 bity globalnego licznika, zatraskiwane w momencie wpisu do bitu start channel</u>	<u>0x0</u>	<u>RO</u>
<u>Counter Latch LSB</u>	<u>0x18</u>	<u>31:0</u>	<u>31:0 bity globalnego licznika, zatraskiwane w momencie wpisu do bitu start channel</u>	<u>0x0</u>	<u>RO</u>
<u>Counter Live MSB</u>	<u>0x1c</u>	<u>31:0</u>	<u>63:32 bity globalnego licznika, wartość aktualna</u>	<u>-</u>	<u>RO</u>
<u>Counter Live LSB</u>	<u>0x20</u>	<u>31:0</u>	<u>31:0 bity globalnego licznika, wartość aktualna</u>	<u>-</u>	<u>RO</u>
<u>DDR start address</u>	<u>0x24</u>	<u>31:0</u>	<u>Adres ddr zapisu danych</u>	<u>0x200000</u>	<u>RW</u>
<u>FCW mixer</u>	<u>0x28</u>	<u>31:0</u>	<u>FCW dla NCO mixer</u>	<u>0x1FFFFFFF</u>	<u>R/WS</u>
<u>Mixer test</u>	<u>0x2C</u>	<u>31:24</u>	<u>-</u>	<u>0x00</u>	<u>RO</u>
		<u>23:16</u>	<u>Dane z wyjścia mixera (23:20 I) (19:16 Q)</u>	<u>0x00</u>	<u>RO</u>
		<u>15:11</u>	<u>-</u>	<u>-</u>	<u>RO</u>
		<u>10</u>	<u>Ustaw tryb mixera</u>	<u>0</u>	<u>W1C</u>
		<u>9:8</u>	<u>Tryb miksera: „00” - mode 0: $I = i * \cos - q * \sin;$ $Q = q * \cos + i * \sin;$</u>	<u>00</u>	<u>RW</u>

			<u>„01” - mode 1:</u> <u>$I = i * \cos + q * \sin;$</u> <u>$Q = q * \cos - i * \sin;$</u> <u>„10” - mode 2:</u> <u>$I = i * \cos + q * \sin;$</u> <u>$Q = -q * \cos + i * \sin;$</u> <u>„11” - mode 3:</u> <u>$I = i * \cos - q * \sin;$</u> <u>$Q = -q * \cos - i * \sin;$</u>		
		<u>7:5</u>	<u>-</u>	<u>-</u>	<u>RO</u>
		<u>4</u>	<u>Neguj wpisywanie danych IQ na wejście mixera (software mode)</u>	<u>0</u>	<u>W1C</u>
		<u>3:0</u>	<u>Dane I i Q na wejście mixera (3:2 I) (1:0 Q)</u>	<u>0x0</u>	<u>R/WS</u>
<u>FCW code</u>	<u>0x30</u>	<u>31:0</u>	<u>FCW dla NCO kodu</u>	<u>0x1FFFFFFF</u>	<u>R/WS</u>
<u>Code ctrl</u>	<u>0x34</u>	<u>31</u>	<u>-</u>	<u>-</u>	<u>RO</u>
		<u>30</u>	<u>Wymuś kod</u>	<u>'0'</u>	<u>W1C</u>
		<u>29</u>	<u>-</u>	<u>'0'</u>	<u>RO</u>
		<u>28:25</u>	<u>Wymuszony wskaźnik chip_div</u>	<u>0x0</u>	<u>RW</u>
		<u>24:20</u>	<u>Aktualny kod (MSB VE,E,P,L,VL LSB)</u>	<u>0x00</u>	<u>RO</u>
		<u>19:16</u>	<u>chip_div - ilość przepełnień nco kodu po którym zostanie wczytany nowy bit chip (0x0 - 1, 0x1 - 2, 0x2 - 3 itd...)</u>	<u>0x0</u>	<u>R/WS</u>
		<u>15:14</u>	<u>spacing_len - ilość przepełnień nco kodu po którym zostaje przesunięty wektor kodu veplv_code ("00" - 1, "01" - 2 "10" - 3 "11" - 4)</u>	<u>„10”</u>	
		<u>13:0</u>	<u>Długość ciągu kodowego (w bitach-1)</u>	<u>0x03FE</u>	
<u>Code data0</u>	<u>0x38</u>	<u>31:28</u>	<u>Dane z wyjścia VE I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>27:24</u>	<u>Dane z wyjścia VE Q (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>23:20</u>	<u>Dane z wyjścia E I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>19:16</u>	<u>Dane z wyjścia E Q (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>15:12</u>	<u>Dane z wyjścia P I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>11:8</u>	<u>Dane z wyjścia P Q (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>7:4</u>	<u>Dane z wyjścia L I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>3:0</u>	<u>Dane z wyjścia L Q (kod)</u>	<u>0x0</u>	<u>RO</u>
<u>Code data1</u>	<u>0x3C</u>	<u>31:28</u>	<u>Dane z wyjścia VL I (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>27:24</u>	<u>Dane z wyjścia VL Q (kod)</u>	<u>0x0</u>	<u>RO</u>
		<u>23:17</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>16</u>	<u>Wyczyść Integrate & Dump</u>	<u>0x0</u>	<u>W1C</u>
		<u>15</u>	<u>-</u>	<u>0</u>	<u>RO</u>
		<u>14:0</u>	<u>Wymuszony indeks chip</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 0</u>	<u>0x40</u>	<u>31:0</u>	<u>Dane z wyjścia VE I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 1</u>	<u>0x44</u>	<u>31:0</u>	<u>Dane z wyjścia VE Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 2</u>	<u>0x48</u>	<u>31:0</u>	<u>Dane z wyjścia E I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 3</u>	<u>0x4C</u>	<u>31:0</u>	<u>Dane z wyjścia E Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 4</u>	<u>0x50</u>	<u>31:0</u>	<u>Dane z wyjścia P I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 5</u>	<u>0x54</u>	<u>31:0</u>	<u>Dane z wyjścia P Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 6</u>	<u>0x58</u>	<u>31:0</u>	<u>Dane z wyjścia L I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 7</u>	<u>0x5C</u>	<u>31:0</u>	<u>Dane z wyjścia L Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>I&D data 8</u>	<u>0x60</u>	<u>31:0</u>	<u>Dane z wyjścia VL I (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>

<u>I&D data 9</u>	<u>0x64</u>	<u>31:0</u>	<u>Dane z wyjścia VL Q (Integrate & dump)</u>	<u>0x0</u>	<u>RO</u>
<u>Memory read len</u>	<u>0x68</u>	<u>31:27</u>	<u>=</u>	<u>0x0</u>	<u>RO</u>
		<u>26:0</u>	<u>Ilość danych do przetwarzania z DDR (w słowach 32B)</u>	<u>0x1000</u>	<u>RW</u>
<u>Ce ctrl</u>	<u>0x6C</u>	<u>31:7</u>	<u>=</u>	<u>0x0</u>	<u>RO</u>
		<u>6</u>	<u>Set ce ctrl</u>	<u>0x0</u>	<u>W1C</u>
		<u>5:4</u>	<u>Mode ce ctrl: „00” - software data „01” - max data „10” - ddr read data</u>	<u>0x0</u>	<u>RW</u>
		<u>3:0</u>	<u>Drop data (saples)</u>	<u>0x0</u>	<u>RW</u>
<u>Force CE code NCO</u>	<u>0x70</u>	<u>31:0</u>	<u>Wymuszona nastawa NCO kodu</u>	<u>0x0</u>	<u>RW</u>
<u>Spi mux ctrl</u>	<u>0x74</u>	<u>31:4</u>	<u>-</u>	<u>0x0</u>	<u>RO</u>
		<u>3</u>	<u>Ustaw nastawy channel mux select</u>	<u>0x0</u>	<u>W1C</u>
		<u>2:0</u>	<u>Multipleksacja spi, (bit 2 = SPI2, 1 = SPI1, 0 = SPI0) ‘1’ aby ustawić transmisję SPI na tym interfejsie</u>	<u>0x1</u>	<u>RW</u>
<u>Channel mux select</u>	<u>0x78</u>	<u>31:30</u>	<u>Ustaw odpowiedni kanał fizyczny (max) dla kanału 15 online („00” – max 0, „01” max 1, „10” max 2, „11” kanał wyłączony)</u>	<u>0x3</u>	<u>RW</u>
		<u>...</u>			
		<u>..</u>	<u>Analogiczna opisana wyżej funkcjonalność dla kanałów 14-2</u>		
		<u>...</u>			
		<u>3:2</u>	<u>Ustaw odpowiedni kanał fizyczny (max) dla kanału 1 online („00” – max 0, „01” max 1, „10” max 2, „11” kanał wyłączony)</u>	<u>0x3</u>	<u>RW</u>
		<u>1:0</u>	<u>Ustaw odpowiedni kanał fizyczny (max) dla kanału 0 online („00” – max 0, „01” max 1, „10” max 2, „11” kanał wyłączony)</u>	<u>0x3</u>	<u>RW</u>
<u>Memory channel read enable</u>	<u>0x7C</u>	<u>31:30</u>	<u>Ustaw odpowiedni kanał fizyczny (max) dla kanału 31 online („00” – max 0, „01” max 1, „10” max 2, „11” kanał wyłączony)</u>	<u>0x3</u>	<u>RW</u>
		<u>...</u>	<u>Analogiczna opisana wyżej funkcjonalność dla kanałów 30-18</u>		
		<u>3:2</u>	<u>Ustaw odpowiedni kanał fizyczny (max) dla kanału 17 online („00” – max 0, „01” max 1, „10” max 2, „11” kanał wyłączony)</u>	<u>0x3</u>	<u>RW</u>
		<u>1:0</u>	<u>Ustaw odpowiedni kanał fizyczny (max) dla kanału 16 online („00” – max 0, „01” max 1, „10” max 2, „11” kanał wyłączony)</u>	<u>0x3</u>	<u>RW</u>
		<u>15:0</u>	<u>Zezwól na wyczytywanie danych z pamięci dla odpowiedniego kanału (MSB 15 LSB 0)</u>	<u>0xFFFF</u>	<u>RW</u>

reg_axi_1

Nazwa	Adres	Bit	Opis	Default	Tryb
Component ver	0x00	31:0	ID danego komponentu axi	0x3	RO
Coeff 0	0x04	31:19	-	-	RO
		18	Ustaw wartości współczynnika 0 i 1	0x0	W1C
		17:0	Signed współczynnik 0	0x0	RW
Coeff 1	0x08	31:18	-	-	RO
		17:0	Signed współczynnik 1	0x0	RW
Filter data in	0x0C	31:26	-	-	RO
		25	Wpisz dane na wejście filtra	0x0	W1C
		24:0	Signed dane wejściowe filtra	0x0	RW
Filter data out0	0x10	31:0	Dane wyjściowe z filtra 31-0	0x0	RO
Reserved	0x14	31:0	-	-	RO
IRQ reg 0	0x18	31:16	IRQ type – 0 high lvl 1 pulse	0x0	RW
		15:0	IRQ status / IRQ clear	0x0	W1C
IRQ reg 1	0x1C	31:16	IRQ enable	0x0	RW
		15:0	IRQ force	0x0	RW
Filter state 0	0x20	31:0	Dane state filtra(31—0) in/out signed	0x0	RW
Filter state 1	0x24	31:12	-	-	RO
		11:0	Dane state filtra(43—32) in/out signed	0x0	RW
Measure counter	0x28	31:0	Ilość cykli oczekiwania do wykonania pomiaru	0x0	R/WS
Irq src	0x2C	31:0	Irq source – przypisania pod tabelami	0x0	W1C
Reserved	0x30 - 0x3C		-	-	RO

reg_axi_meas

Nazwa	Adres	Bit	Opis	Default	Tryb
Component ver	0x00	31:0	ID danego komponentu axi	0x4	RO
Current chan	0x04	31:5	-	-	RO
		4:0	Aktualny kanał (0 - 32)	0x0	RW
Mixer reg 0	0x08	31:0	Mixer nco latched	0x0	RO
Mixer reg 1	0x0C	31:24	-	-	RO
		23:0	Mixer nco overflow count latched	0x0	RO
Code data 0	0x10	31:26	-	0x0	RO
		25:21	Code sequence counter (licznik przepełnień kodu)	0x0	RO
		20:17	Code overflow	0x0	RO
		16:15	Chip div	0x0	RO
		14:0	Chip pointer (bram address)	0x0	RO
Code data 1	0x14	31:0	Code nco latched	0x0	RO
Dyskr dll I data early	0x18	31:27	-	0x0	RO
		26	Start processing data	0x0	W1C
		25:0	I data early	0x0	RW
Dyskr dll Q data early	0x1C	31:26	-	0x0	RO
		25:0	Q data early	0x0	RW
Dyskr dll I data late	0x20	31:26	-	0x0	RO
		25:0	I data late	0x0	RW

Dyskr dll Q data late	0x24	31:26	-	0x0	<u>RO</u>
		25:0	Q data late	0x0	RW
Dyskr dll Q data late 0	0x28	31:0	31:0 bit Q dyskryminator DLL	0x0	<u>RO</u>
Dyskr dll Q data late 1	0x2C	31:0	63:32 bit Q dyskryminator DLL	0x0	<u>RO</u>
Dyskr dll F data late 0	0x30	31	-	0x0	<u>RO</u>
		30	Sum zero kanał 0	0x0	<u>RO</u>
		29:0	29:0 bit LSB F dyskryminator DLL	0x0	<u>RO</u>
Counter Latch LSB	0x34	31:0	31:0 bity globalnego licznika, zatrzaskiwane w momencie pomaru	0x0	RO
Counter Latch MSB	0x38	31:0	63:32 bity globalnego licznika, zatrzaskiwane w momencie pomaru	0x0	RO
Debug	0x3C	31:1	-	0x0	<u>RO</u>
		0	Wyczyść wskaźnik snoop memory	0x0	<u>WIC</u>

reg_axi_pll

Nazwa	Adres	Bit	Opis	Default	Tryb
Component ver	0x00	31:0	ID danego komponentu axi	0x5	RO
Data pll I in	0x04	31:26	-	0x0	<u>RO</u>
		25	Start processing data	0x0	<u>WIC</u>
		24:0	Data PLL in I	0x0	RW
Data pll Q in	0x08	31:25	-	0x0	<u>RO</u>
		24:0	Data PLL in Q	0x0	RW
Result reg 0	0x0C	31:0	Quotient PLL data out 31:0	0x0	<u>RO</u>
Result reg 1	0x10	31:12	-	0x0	<u>RO</u>
		11	Sum zero flag	0x0	<u>RO</u>
		10:0	Quotient PLL data out 10:0	0x0	<u>RO</u>
Debug irq reg	0x14	31:8	-	0x0	RO
		7:0	Measure irq count	0x0	RO
<u>Reserved</u>	0x18-x3C		-	-	RO

reg_chan 0 - reg_chan 31

Nazwa	Adres	Bit	Opis	Default	Tryb
Component ver	0x00	31:5	ID danego komponentu axi	0x0000001	RO
		4:0	ID danego kanału (0x0 dla kanału 0, 0x1 dla kanału 1, 0x2 dla kanału 2... etc)	0x0	RO
N overflow	0x04	31	Pozwól na zapis kodu do block ramu dla tego kanału	0x0	RW
		30:11	-	-	RO
		10	Stop kanału ok (przy stop za pomocą licznika)	0x0	<u>WIC</u>
		9	Ustaw licznik stopu kanału online	0x0	WO
		8	Start kanału ok	0x0	<u>WIC</u>
		7	Stop kanału online	0x0	WO
		6	Ustaw licznik startu kanału online	0x0	WO
		5	Ustaw wartość licznika przepełnień	0x0	WO
		4:0	Liczba przepełnień N kodu po której	0x0	RW

			zostanie wysłane przerwanie		
I&D data 0	0x8	31:0	Zatrzaśnięte dane z wyjścia VE I (Integrate & dump)	0x0	RO
I&D data 1	0xC	31:0	Zatrzaśnięte dane z wyjścia VE Q (Integrate & dump)	0x0	RO
I&D data 2	0x10	31:0	Zatrzaśnięte dane z wyjścia E I (Integrate & dump)	0x0	RO
I&D data 3	0x14	31:0	Zatrzaśnięte dane z wyjścia E Q (Integrate & dump)	0x0	RO
I&D data 4	0x18	31:0	Zatrzaśnięte dane z wyjścia P I (Integrate & dump)	0x0	RO
I&D data 5	0x1C	31:0	Zatrzaśnięte dane z wyjścia P Q (Integrate & dump)	0x0	RO
I&D data 6	0x20	31:0	Zatrzaśnięte dane z wyjścia L I (Integrate & dump)	0x0	RO
I&D data 7	0x24	31:0	Zatrzaśnięte dane z wyjścia L Q (Integrate & dump)	0x0	RO
I&D data 8	0x28	31:0	Zatrzaśnięte dane z wyjścia VL I (Integrate & dump)	0x0	RO
I&D data 9	0x2C	31:0	Zatrzaśnięte dane z wyjścia VL Q (Integrate & dump)	0x0	RO
Counter Latch LSB	0x30	31:0	31:0 bity globalnego licznika, zatrzaśkiwane w momencie N przepełnienia kodu	0x0	RO
Counter Latch MSB	0x34	31:0	63:32 bity globalnego licznika, zatrzaśkiwane w momencie N przepełnienia kodu	0x0	RO
Online start LSB	0x38	31:0	Bit 31:0 startu przetwarzania/zatrzymania danych online kanału	0x0	RW
Online start MSB	0x3C	31:0	Bit 63:32 startu przetwarzania/zatrzymania danych online kanału	0x0	RW

Oznaczenia trybu: RW – odczyt zapis, RO – tylko odczyt, WO – tylko zapis, W1C – po zapisie 1 wartość bitu jest czyszczona, R/WS – zapis ustawia wartości odpowiednich komponentów (nie jest wymagana dodatkowa flaga do ustawienia).

Mapowanie kanałów:

0 – złącze JD, Max (zapis do bram + DDR)

1 – stała wartość 0xFF na wejściu IQ (zapis do bram + DDR)

2 – stała wartość 0x66 na wejściu IQ (zapis do bram + DDR)

3 - licznik (zapis do DDR)

Irq source:

0 – dump irq chan 0

1 – dump irq chan 1

...

31 – dump irq chan 31

Stop przetwarzania danych w zależności od wartości globalnego licznika

W wersji 0x3_DF dodano możliwość zatrzymywania przetwarzania danych w zależności od zdefiniowanej wartości globalnego licznika dla każdego z kanałów online. W rejestrze reg_chan

dodano bit 9 rejestru N Overflow, który zatrzaśnie wartość globalnego licznika dla komponentu sterowania przepływem danych z MAX. W celu ustawienia wartości stop, należy wpisać zadaną nastawę globalnego licznika do rejestrów Online start LSB i Online start MSB (rejestry 0x38 i 0x3C są wykorzystywane dla start i stop w odpowiednim momencie globalnego licznika) oraz ustawić bit 9 rejestru N overflow na '1'. Jeżeli kanał zostanie poprawnie zatrzymany na zadanej wartości licznika, bit stop ok powinien zostać ustawiony na '1'.

Zatrząskiwanie przerwań przy przetwarzaniu przerwania irq dump

W wersji 0x3_DC została dodana funkcjonalność automatycznego zatrząskiwania przychodzących przerwań w celu usprawnienia softwerowej funkcji obsługi przerwań. Zmieniona funkcjonalność objawia się „zatrzaśnięciem” zawartości rejestru Irq src reg_axi_1 w momencie jego odczytu, oraz buforowania przychodzących przerwań do momentu zapisu wartości do ww. rejestru. Poprawna obsługa zaimplementowanego kodu powinna być następująca: odczytanie IRQ reg 0, w przypadku przerwania od dump należy wyczytać rejestr Irq src (spowoduje to zatrzaśnięcie rejestru do momentu clearu tego rejestru), wyczyszczenie rejestru IRQ reg 0, wyczyszczenie rejestru Irq src. Jeżeli pomiędzy wyczytaniem rejestru Irq src a zapisem wystąpi następne przerwania od innego kanału niż otrzymany przy odczycie, zostanie on automatycznie wpisany do Irq src oraz zostanie wysłane przerwania od dump. Przerwanie zostaje wysłane z 1 cyklem przerwania stanu wysokiego, ważne jest aby przetestować scenariusz automatycznego generowania przerwania w przypadku przyjscia przerwania z innego źródła niż aktualnie przetwarzane w funkcji obsługi przerwań.

Anulowanie wpisywania danych na wejście miksera

Dane na wejście miksera zostają automatycznie wpisywane w momencie zapisu do rejestru Mixer test reg_axi_0. W celu zanegowania tej funkcjonalności została dodana flaga. Negowanie wpisywania danych trybu software na wejście miksera odbywa się za pomocą dodanej flagi - bit 4 Mixer test rejestru reg_axi_0. Jeżeli wystąpi '1' w momencie wpisywania wartości do tego rejestru, dane na wejście miksera nie zostaną przekazane.

Ustawianie mixer mode

W wersji 0x3_D6 dodano funkcjonalność stałego przypisywania mixer mode do danego kanału za pomocą bitu 10 rejestru Mixer test dla reg_axi_0. W celu ustawienia danego trybu, wybrać odpowiedni kanał za pomocą Test reg bity 19:16 (reg_axi_0), następnie wpisać nastawę trybu miksera bity 9:8 Mixer test (reg_axi_0). Wpisanie '1' do bitu 10 Mixer test powinno ustawić odpowiedni tryb komponentu miksera.

Wyczytywanie z pamięci dla wielu kanałów

~~W wersji 0x3BE dodano funkcjonalność powielania danych wyczytywanych z pamięci kanału 0 dla kanałów 0-15. Dodano rejestr 0x7C (reg_axi_0) regulującymi na które z kanałów mają być propagowane dane. Domyślnie dane wyczytywane z kanału 0 są propagowane na kanały 0-15 (domyślne ustawienie 0xFFFF). Funkcjonalność wyczytywania danych z pamięci dla kanałów 1-15 została wyłączona ze względu na propagowanie danych z jednego rzędu wyczytania kanału 0 na wiele kanałów.~~

Obsługa dyskryminatora dll v2

W wersji 0x3_BA wprowadzono możliwość testowania komponentu dyskryminatora DLL ver 2 wraz z dzielnikiem. Rejestry przeznaczone do obsługi tego komponentu znajdują się w [reg_axi_meas](#), 0x18-0x24. Po wpisaniu wartości rejestrów wejściowych należy wpisać '1' do bitu 26 rejestru Dyskr dll I data early [reg_axi_meas](#). Wynik podany jest w postaci I64.Q0 (DLL_DISC_AMP_EXP = 18) wraz z dzieleniem przez 2. Zostaje również zatrzaśnięta wartość division zero.

0x3_B9

Dodano możliwość zapisywania wartości danych pomiarowych w dodanej pamięci BRAM `axi_bram_ctrl_2`. Zapis odbywa się automatycznie przy wygenerowaniu przerwania `measure`. Dane zapisywane w strukturze 64 bitowych słów:

```
BRAM_PORTB_0_din(7 downto 0) <= std_logic_vector(u_BRAM_PORTB_0_addr(7 downto 0));
```

```
BRAM_PORTB_0_din(12 downto 8) <= v_seq_cnt_latch(4 downto 0);
```

```
BRAM_PORTB_0_din(16 downto 13) <= v_code_of_cnt_latch_out(3 downto 0);
```

```
BRAM_PORTB_0_din(48 downto 17) <= v_code_nco_latch_out(31 downto 0);
```

```
BRAM_PORTB_0_din(63 downto 49) <= v_code_bram_addr_latch_out(14 downto 0);
```

Po zapisie 2048 słów po przerwaniach, aby ponownie zapisać dane, należy wyczyścić wskaźnik `bram_snoop` za pomocą rejestru Debug rejestru `reg_axi_meas`. BRAM zostanie nadpisany ponownie 2048 kolejnymi wartościami.

Konfiguracja kanałów fizycznych/online

W wersji 0x3_B3 została dodana możliwość wyboru źródła danych przychodzących z wybranego układu MAX. W aktualnej implementacji dostępny jest jedynie `max0`. Domyślnie wszystkie kanały są wyłączone, należy ustawić poprawny kanał na „00” w rejestrze `Channel mux select` aby przesyłać dane z `max0`.

Wybór kontrolera spi

W wersji 0x3_B3 dodano możliwość multiplexacji/kontroli obsługiwanego interfejsu SPI. Fragment kodu został przygotowany z myślą o przyszłych konfiguracjach płyty zcybo oraz 1 lub 3ch układów max. Aktualna wersja jest przygotowana do obsługi 1 układu max. Aby wysyłać sygnał spi cs do układu `max0`, bit 0 rejestru `Spi mux ctrl` musi mieć wartość '1' (domyślne ustawienia). Aby wyłączyć linię cs do tego układu należy wpisać '0' do rejestru `Spi mux ctrl bit 0`.

Obsługa dyskryminatora PLL

W wersji 0x3_A5 wprowadzono możliwość testowania komponentu dyskryminatora PLL wraz z dzielnikiem. Rejestry przeznaczone do obsługi komponentu znajdują się w `reg_axi_pll`. Po wpisaniu wartości rejestrów wejściowych (0x4 0x8) należy wpisać '1' do bity 25 rejestru `Data pll I in`. Wynik podawany jest jako wartość integer. Zostaje również zatrzaśnięta wartość `division zero`.

Wersja 0x3_9E

Doprowadzono zegar 100MHz do części dzielnika dyskryminatora dll (w poprzedniej wersji 40MHz).

Obsługa dyskryminatora dll

W wersji 0x3_9D wprowadzono możliwość testowania komponentu dyskryminatora DLL wraz z dzielnikiem. Rejestry przeznaczone do obsługi tego komponentu znajdują się w `reg_axi_meas`, 0x18-0x24. Po wpisaniu wartości rejestrów wejściowych należy wpisać '1' do bity 26 rejestru `Dyskr dll I data early` `reg_axi_meas`. Wynik podany jest w postaci I52.Q30 bez przesunięcia bitowego (czysty wynik `diff/sum`). Zostaje również zatrzaśnięta wartość `division zero`.

Obsługa reg_axi_meas

W wersji 0x3_8D wprowadzono możliwość zatraskiwania pomiarów ze strony fpga w określonych odstępach czasu. Przerwanie od pomiaru zostało ustawione na bit 0 wektora przerwania wraz z przerwaniem dump kanału 0. Aby uruchomić periodyczne pomiary, należy ustawić wartość licznika Measure counter rejestru `axi_reg1`. Po wpisaniu wartości licznika (odmierzanego w 25 ns cyklach) po odpowiednim czasie zostaną zatrzaśnięte wartości pomiarowe, które następnie zostaną umieszczone w rejestrze `reg_axi_meas`, oraz zostanie wygenerowane przerwanie numer '0'. W celu potwierdzenia źródła przewania, należy sprawdzić

rejestr Irq src, bit '1'. Należy wpisać '1' do odpowiedniego bitu aby wyczyścić irq src.

Global cntr start channel 0

W wersji 0x3_80 dodano możliwość startu kanału online 0 w momencie zdefiniowanym przez rejestr globalnego licznika. Aby wykorzystać tę funkcjonalność, należy zdefiniować moment startu kanału przez rejestry Online start LSB/MSB, wpis do bitu 6 rejestru N overflow spowoduje zatrzaśnięcie nastaw rejestru startu. Channel 0 zostanie uruchomiony jeżeli globalny licznik wewnątrz FPGA osiągnie zdefiniowaną wartość. Jeżeli to nastąpi, bit 8 rejestru N overflow zostanie ustawiony na '1'. W celu zatrzymania kanału, należy wpisać '1' do 7 bitu rejestru N overflow. W aktualnej wersji, start kanału nie jest automatyczna – co oznacza że po Start write online channel dane nie zostaną automatycznie przetworzone w kanale dla modu max data ce ctrl mode, dopiero po ustawieniu poprawnego licznika startu kanału.

Obsługa przerwania DUMP

W opisywanej wersji 0x3_7F został dodany dodatkowy rejestr reg_chan_0 odpowiedzialny za kontrolę oraz zbiór danych z akumulatorów wykonywanych w momencie N przepełnienia licznika kodu. W aktualnej wersji jedynie kanał 0 jest rejestrowany. Przerwanie generowane w momencie dojścia do końca kodu N razy jest podłączone do wejścia 0 IRQ. Aby napisana część kodu działała poprawnie, należy skonfigurować przerwanie IRQ 0 rejestru reg_axi_1. W celu ustawienia N liczby przepełnień, należy wpisać wartość do rejestru N overflow bity 4:0, wpisanie '1' bity 5, spowoduje wysłanie nastawy do odpowiedniego komponentu wewnątrz FPGA. Istnieje możliwość jednoczesnego zapisu nastawy i ustawienia komponentu.

Obsługa filtra (w wersji 0x3 – rejestr Component ver 0x3 reg_axi_1)

W odróżnieniu do wersji 0x1, dodano rejestry stanu filtra w celu analogicznej implementacji jak w kodzie C. Dodano rejestry Filter state 0/1. Nastawa stanu jest wpisywana wraz z nastawą współczynników „Ustaw wartości współczynnika 0 i 1” bit 18 rejestru 0x04.

Wpisz nastawę stanu filtra do rejestru Filter state 0/1. Wpisz nastawę pierwszego i drugiego współczynnika pod adresy Coeff 0 Coeff 1 0x4000_2004 oraz 0x4000_2008, wpisz '1' pod bit 18 rejestru Coeff 0 aby wpisać nastawy współczynników oraz wpisać stan do komponentu filtra. Wpisz wartość danych wejściowych filtra do rejestru Filter data in 0x4000_200C, wpisanie bitu 25 rejestru Filter data in spowoduje przeliczenie danych przez filter oraz wpisanie do rejestrów Filter data out0 oraz Filter data out1. Przeliczony stan filtra nadpisze wartości rejestrów 0x20 i 0x24 Filter state 0/1.

Obsługa przerwania

W wersji 0x3_7B istnieje możliwość obsługi przerwania z poziomu rejestrów, za pomocą IRQ force z rejestru IRQ reg 1. 16 zaimplementowanych rejestrów zostało podpiętych pod zynq wejścia IRQ PL-PS 15-0. Istnieje możliwość konfiguracji rodzaju przerwania poprzez rejestr IRQ type, 0 aby ustawić poziom wysoki na linii do momentu wyczyszczenia przerwania za pomocą IRQ clear; lub IRQ type 1 w którym przerwanie jest wysyłane za pomocą pojedynczego pulsu trwającego 25 ns i nie wymagającego wyczyszczenia rejestru statusowego. Aby móc wysłać przerwanie należy ustawić rejestr dla odpowiedniego przerwania IRQ enable na '1' – w przeciwnym wypadku przerwanie będą maskowane. Następnie przy pomocy rejestru IRQ force zostanie wysłane przerwanie na odpowiednich bitach. Warto jest wspomnieć że irq controller jest rising edge sensitive, więc aby wywołać kolejne przerwanie należy wpisać IRQ force '0' następnie '1'.

Obsługa filtra (w wersji 0x1 – rejestr Component ver 0x1 reg_axi_1)

Wpisz nastawę pierwszego i drugiego współczynnika pod adresy Coeff 0 Coeff 1 0x4000_2004 oraz 0x4000_2008, wpisz '1' pod bit 18 rejestru Coeff 0 aby wpisać nastawy współczynników do komponentu filtra. Wpisz wartość danych wejściowych filtra do rejestru Filter data in 0x4000_200C, wpisanie bitu 25 rejestru Filter data in spowoduje przeliczenie danych przez filter oraz wpisanie do rejestrów Filter data out0 oraz Filter data out1.

Obsługa różnych kanałów offline:

Dodane zostały 7 kanały offline. Ilość kanałów offline w aktualnej wersji znajduje się pod adresem 0x8 bity 27:24 rejestr „Ilość kanałów offline”. W celu minimalizacji wykorzystanej przestrzeni rejestrów, sterowanie oraz wyniki obserwowane w niektórych rejestrach są mapowane w zależności od aktualnie wybranego kanału: 0x8 bity 18:16 rejestr „Aktualny kanał offline”. W powyższej tabeli zostały zaznaczone (kolorem czerwonym) rejestry które są mapowane w zależności od wybranego kanału offline. Dla kolejnych kanałów należy odpowiednio wgrać wartości w rejestrach w celu poprawnego działania. Niektóre rejestry są wykorzystywane przez wszystkie kanały, jedynie ustawiane w odpowiednich momentach (np. przy starcie kanału, lub przy ustawieniu odpowiedniej flagi). Reszta funkcjonowania kanałów offline pozostaje analogiczna jak w przypadku jednego kanału.

Wymuszenie nastaw wskaźnika chipa kodu, indeksu chip_div'a oraz NCO kodu:

Dodana została możliwość ustawienia chipa kodu, indeksu chip_div'a oraz wartości NCO kodu. Aby ustawić powyższe wartości należy wpisać '1' do rejestru Code ctrl bit 30 „Wymuś kod”.

Zostaną ustawione wartości z następujących rejestrów:

- wartość wskaźnika chipa kodu z adresu Code data1 0x3C bity 14:0 rejestr „Wymuszony indeks chip”
- wartość indeksu chip_div'a z adresu Code ctrl 0x34 bity 28:25 rejestr „Wymuszony wskaźnik chip_div”
- wartość NCO kodu z adresu Force CE code NCO 0x70 bity 31:0 rejestr „Wymuszona nastawa NCO kodu”.

Obsługa kanału z testowym mieszaczem:

Dodane zostały 2 funkcjonalności pracy z mieszaczem, generatorem kodu rozpraszającego oraz integratorem. Pierwsza opcja pozwala na wprowadzanie danych bezpośrednio z układu zewnętrznego MAX.

Przypadek 1: przetwarzanie danych przychodzących z układu MAX

Zapis 0x50 → 0x4000_006C – ustawi ścieżkę przetwarzania na max

Zapis 0x2000 → 0x40000010 – wystartuje pobieranie danych z max (kanał 0)

Odczyt bitu 10 rejestru 0x4000_0010 pozwoli potwierdzić to że aktualnie dane są przetwarzane.

Zapis 0x10000 → 0x40000010 – zatrzyma wysyłanie danych

Przypadek 2: przetwarzanie danych z pamięci DDR

Zapis 0x60 → 0x4000_006C – ustawi ścieżkę przetwarzania z pamięci DDR

Zapis 0x1000 → 0x4000_0068 - ustawić ilość wyczytywanej pamięci z DDR w rejestrze

Memory read len (0x68) w słowach 32B

Zapis 0x10 → 0x4000_0010 – start przetwarzania danych z pamięci DDR

Po udanej operacji powinna zostać podniesiona flaga Done memory read ddr (bit 12 rejestr 0x4000_0010).

Adres od którego zostaje rozpoczęte wyczytywanie pamięci DDR jest zdefiniowany rejestrem DDR start address (0x4000_0024).

Dodatkowo istnieje możliwość ominięcia pewnej ilości danych za pomocą rejestru Ce ctrl bit 3:0 (0x4000_006C).

Obsługa testowego mieszacza:

Należy ustawić rejestr Ce ctrl aby działał w trybie software data (domyślna wartość):

Zapis 0x40 → 0x4000_006C

Zapis do rejestru FCW automatycznie ustawia wartość wpisaną do rejestru.

Zapis do rejestru Mixer test (3:0) automatycznie uruchamia mixer. Po 2ch taktach zegara dane znajdują się w rejestrze 23:16 (23:20 I) (19:16 Q). Domyślny tryb pracy miksera to mode 3 (wartość tryb miksera „00”). W celu wybrania innego trybu do testów, przy zapisie danych wejściowych miksera, należy również wybrać odpowiedni mod (bity 9:8).

Przypadek 1:

Zapis 0x30F → 0x4000_002C, I=11 Q=11 na wejściu miksera w mode = 3 $I = i * \cos - q * \sin$; $Q = -q * \cos - i * \sin$;

Odczyt → 0x4000_002C – wyjście miksera znajduje się pod bitami 23:16

Przypadek 2:

Zapis 0x106 → 0x4000_002C, I=01 Q=10 na wejściu miksera w mode = 1 $I = i * \cos + q * \sin$; $Q = q * \cos - i * \sin$;

Odczyt → 0x4000_002C – wyjście miksera znajduje się pod bitami 23:16

Obsługa testowego mieszacza wraz z kodem + integrate & dump

Zapis do rejestru FCW miksera automatycznie resetuje licznik NCO miksera i ustawia wartość zapisaną.

mwr 0x40000028 0x1FFFFFFF

Zapis do rejestru FCW kodu automatycznie resetuje licznik NCO kodu i ustawia wartość zapisaną.

mwr 0x40000030 0x7FFFFFFF

Aby ustawić współczynniki kodu, wykorzystaj rejestr Code ctrl (po wpisaniu do tego rejestru komponent kodu (przy zapisie rejestru wartości automatycznie zostają zapisane do IP)

mwr 0x40000034 0x3FE

spacing_len ustawiony na 0x0, wektor kodu zostaje przesunięty o każde przepełnienie nco kodu, długość kodu została ustawiona na 1023 bity (0x3FE+1).

Współczynniki kodu znajdują się pod adresem 0x4000_1000 (dodany zakres adresu) w postaci 32bitowych słów.

»Bity kodu zostają wyczytane od LSB do MSB. Aby zapisać kod do odpowiedniego kanału, należy ustawić bit rejestru reg_chan_0 0x4 bit 31 na '1'. Istnieje możliwość równoczesnego zapisu współczynników dla kilku kanałów naraz.

mwr 0x40010004 0x80000000 – pozwól na zapis współczynników kodu kanału 0 (dodane w wersji 0x3_85«

mwr 0x40001000 0x12345678

mwr 0x40001004 0xF0EDCBA9

...

Po zapisie danych do miksera, rejestry mixer test, code data 0-1, I&D 0-9 powinny zostać wypełnione danymi.

W celu obserwacji działania kodu zostały dodane następujące rejestry:

-aktualny kod z rejestru Code ctrl – wskazuje aktualny stan linii przesuwnej kodu (MSB to VE, LSB to VL),

-Bit kodu z pamięci oraz Adres kodu (w 1bit słowach) z adresu Code data1 – wskazuje na aktualnie wyczytywany bit kodu oraz adres spod którego jest wyczytywany.

Istnieje możliwość czyszczenia danych zapisanych w rejestrach Integrate and Dump za pomocą bitu Wyczyść Integrate & Dump rejestru Code data1,

mwr 0x4000003C 0x10000

Istnieje możliwość narzucenia wartości linii przesuwnej kodu za pomocą bitu Wymuś kod rejestru Code ctrl, po następnym wpisaniu danych do miksera, linia przesuwna kodu zostaje ustawiona jako rejestr Wymuszony kod, oraz dane wyczytywane z BRAM nie są brane pod uwagę do momentu aż bit Wymuś kod nie zostanie zresetowany.

mwr 0x40000034 0x7E0003FE

Linie przesuwne ustawione na „11111” (MSB VE LSB VL).

Obsługa zapisu bloku danych (DDR i BRAM):

Po wpisaniu wartości bitu Start channel następuje zapisanie 80000B (4ms) danych z wybranego kanału do BRAMu (Adres 0x4004_0000) oraz rozmiar transfer size (w 4B słowach) danych do pamięci DDR (począwszy od adresu rejestr DDR start address). Ilość danych DDR (Transfer size) musi być wielokrotnością 8 (ze względu na zapis burst 8 słów). W przypadku

wystartowania kanału 0-2, dane zostaną zapisane pod bram i pod ddr (tymczasowe rozwiązanie).

Przypadek 1:

0x1 → 0x4000_0010 aby zapisać dane z MAX:

- flaga Busy bram zostanie podniesiona, 80000 bajtów zostaje zapisane pod adres 0x4004_0000, flaga Busy bram zostanie zresetowana, flaga done bram zostanie podniesiona,

- flaga Busy DDR zostanie podniesiona, Transfer size słów 4bajtowych zostaje zapisane pod adres DDR start address (zakres z 0x20_0000 do 0x3FFF_FFFF), flaga Busy DDR zostanie zresetowana, flaga done DDR zostanie podniesiona.

Obydwa zapisy powinny zawierać te same dane. Działanie analogiczne w przypadku kanałów 1 i 2 (zapis 0xFF i 0x66).

Przypadek 2:

0x8 → 0x4000_0010 aby zapisać licznik do DDR.

- flaga Busy DDR zostanie podniesiona, Transfer size licznika 4 bajtowego zostaje zapisana pod adres DDR start address (zakres z 0x20_0000 do 0x3FFF_FFFF), flaga Busy DDR zostanie zresetowana, flaga done DDR zostanie podniesiona.

Kanał testowy poprawnego zapisu do DDR. Zapis jedynie się odbędzie pod DDR.

Przypadek 3:

0x10 → 0x4000_0010 aby odczytać dane burst 8 słów z DDR:

- 8 słów 4 bajtowych zostanie wyczytane spod adresu DDR start address i umieszczona w rejestrach Read DDR data. Funkcjonalność stworzona na potrzeby debugu.

W przypadkach 1 – 2, należy wpisać '1' aby wyczyścić rejestr Flaga done (W1C).

ILA:

brak