**reg.v Document – Julie Swift**

**Interfaces**

|  |  |  |  |
| --- | --- | --- | --- |
| Signal Name | Direction | Bits | Comment |
| clk | in | 1 | Master Clock |
| rst | in | 1 | Reset |
| i2c\_op | in | 1 | 1: write 0: read |
| i2c\_addr | in | 11 | register address |
| i2c\_wdata | in | 8 | data to be written for a write op |
| i2c\_xfc\_write | in | 1 | write data transfer complete |
| i2c\_rdata | out | 8 | data returned to i2c |
| i2c\_xfc\_read | out | 1 | read data transfer complete |
| i2s\_inoverrun | out | 1 | will turn into sticky bit |

**Functional Requirements:**

* Data Plane Requirements:
  + The register block will never be in a waiting state because the amount of clock cycle it takes deserialize in the i2c will be much longer than it will take to store the data and be ready to take in more data.
  + Reading from I2C’s address, the reg.v will produce an output from whatever is stored in that address
  + When an overrun or underrun occurs from I2S’s FIFO, the status register field will input or output the audio FIFO overrun/underrun
* Control Plane Requirements:
  + When xfc goes high then all the data has been either written or read
  + There will be 256 register fields for the 512 filter coefficients
  + i2c\_op configures the corresponding I/O port pin as either an input or an output to respectively read or write data.
* Control and Status Interface Bit Descriptions:
  + Status Bit
    - i2s\_inoverrun = 1 then there was an overflow in FIFO, if it is = 0 then there is not overflow
  + Control Bits being stored
    - rf\_i2so\_dec\_fact
      * sample and hold audio values
    - rf\_i2si\_dec\_factor
      * sample and hold audio values
    - rf\_filter\_clip\_en
      * 1- performs clipping
      * 0- no clipping
    - rf\_filter\_shift
      * number of bit positions to shift after filter accumulator
    - rf\_i2si\_bist\_en
      * 0- audio source is BIST
      * 1- audio source is i2si
    - rf\_soft\_reset
      * 0- normal operation
      * 1- assert soft reset
  + Status Bits being stored
    - ro\_fifo\_underrun- fifo overrun clear
    - trig\_fifo\_underrun- input audio fifo overrun
    - ro\_fifo\_overrun- fifo underrun clear
    - trig\_fifo\_overrun- output audio fifo underrun

**Micro-Architecture:**

* Sub-blocks
  + trig\_generator – when triggered, this sub-block generates a signal to clear any of the status bits (such as the overrun and underrun)
* Block Diagram  
  + Interfaces:

RESPONSE

CLK to I2C

addr 12

wdata 8

xfc\_write 1

w\_enable 1

i2s\_inoverrun 1

rdata 8

xfc\_read 1

RO Signals

clk

rst

RF Signals

0

1

D

Q

CLK

Write Enable

Write Data

* + Internally­­­­­­­­­

**Design:**

* What the register block is storing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Register Name | Signal Name | RO/WO/RW | Bits | Comment |
| CHIP\_INFO |  |  |  |  |
|  | ro\_chip\_id | RO | 7:0 | Fixed Chip ID |
|  | ro\_revision\_id | RO | 15:8 | Fixed Revisions ID |
| CONTROL |  |  |  |  |
|  | rf\_soft\_reset | RW | 1 | 0- normal operation  1- assert soft reset |
|  | rf\_i2si\_bist\_en | RW | 1 | 0- audio source is i2si  1- audio source is BIST |
|  | rf\_filter\_shift | RW | 4 | number of bit positions to shift after filter accumulator |
|  | rf\_filter\_clip\_en | RW | 1 | 1- performs clipping  0- no clipping |
|  | rf\_i2si\_dec\_factor | RW | 4 | sample and hold audio values |
|  | rf\_i2s0\_dec\_fact | RW | 4 | sample and hold audio values |
| I2S\_CLOCK\_CONTROL |  |  |  |  |
|  | rf\_i2so\_clk2sck\_div | RW | 16 | half of the clock frequency divided by this # |
| STATUS |  |  |  |  |
|  | trig\_fifo\_overrun | WO | 0:0 | fifo overrun clear |
|  | ro\_fifo\_overrun | RO | 1:1 | input audio fifo overrun |
|  | trig\_fifo\_underrun | WO | 2:2 | fifo underrun clear |
|  | ro\_fifo\_underrun | RO | 3:3 | output audio fifo underrun |
| BIST |  |  |  |  |
|  | rf\_i2si\_bist\_start\_val | RW | 12 | start value of sawtooth wave |
|  | rf\_i2si\_bist\_incr | RW | 8 | increment of sawtooth wave |
|  | rf\_i2si\_bist\_upper\_limit | RW | 12 | upperlimit of the sawtooth wave |
| I2C\_REG\_INDIR\_ADDR |  |  |  |  |
|  | rf\_i2c\_reg\_indir\_addr | RW | 11 | address register used for indirect addressing via 12c |
| I2C\_REG\_INDIR\_DATA |  |  |  |  |
|  | ro\_i2c\_reg\_indir\_data | RO | 8 | data access register |
| FILT\_COEFFS\_0\_1 |  |  |  |  |
|  | rf\_filter\_coeff0 | RW | 15:0 | Filter Coefficient 0 |
|  | rf\_filter\_coeff1 | RW | 31:16 | Filter Coefficient 1 |
| FILT\_COEFFS\_2\_3 |  |  |  |  |
|  | rf\_filter\_coeff2 | RW | 15:0 | Filter Coefficient 2 |
|  | rf\_filter\_coeff3 | RW | 31:16 | Filter Coefficient 3 |
| FILT\_COEFS\_510\_511 |  |  |  |  |
|  | rf\_filter\_coeff510 | RW | 15:0 | Filter Coefficient 510 |
|  | rf\_fiter\_coeff511 | RW | 31:16 | Filter Coefficient 511 |

* Number of Gates and Flip-Flops
* Are all controls/statuses correct?

**Verification**

* Testbench
* Test Plan
  + To test I will be reading and writing to every register bit in order to check to make sure the outcome is as expected.
    - Physically check every address to make sure it is outputting the right data.
    - Physically check the default values of every address
  + Make sure the trig\_generater is functioning correctly
    - Making sure it is being triggered at the correct time
    - Making sure it is clearing the status bit every time it is executed