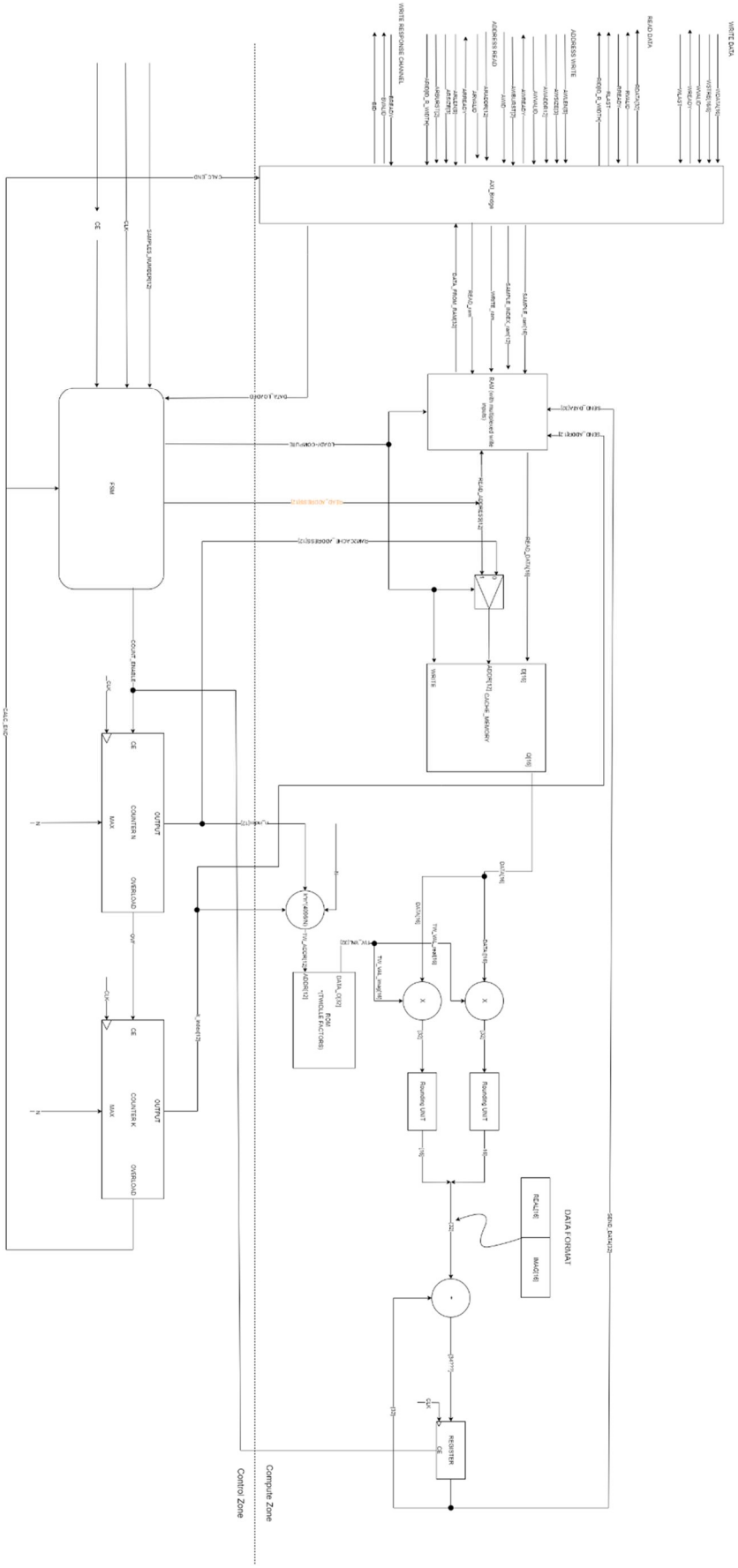
1. Diagram of MAC unit



2. Signal tables for every module.

*In case of need in other file, MAC UNIT.drawio.svg there are same diagram in better quality and possibility to easy zoom and maneuver in picture

1.1. AXI_Bidge

Signals	Width	Direction		Description	Clock domain
AWADDR	12	IN		Indicates the first memory cell in burst data write.	
AWVALID	1	IN		When set to '1', master has valid address to write.	
AWREADY	1	OUT		Set to '1' when address may be written to slave.	
AWBURST	2	IN		This signal indicates burst type. We use INCR type, it means signal ARBUST set to 2'b01.	
AWLEN	8	IN		Specifes number of tranfers in current transaction.	
AWSIZE	3	IN		This signal codes number of bytes of every transfer.	
ARID	ID_W_WIDTH	IN		This signal gives an unique ID for the transaction.	
ARADDR	12	IN		Indicates the first memory cell in burst data read.	
ARVALID	1	IN		When set to '1', master has valid address to write.	
ARREADY	1	OUT		Set to '1' when address may be written to slave.	
ARBURST	2	IN		This signal indicates burst type. We use INCR type, it means signal ARBUST set to 2'b01.	
ARLEN	8	IN		Specifes number of tranfers in current transaction.	
ARSIZE	3	IN		This signal codes number of bytes of every transfer.	
ARID	ID_R_WIDTH	IN		This signal indicates ID of transaction to be read.	
WDATA	16	IN		Data to be written do RAM.	
WSTRB	2	IN		Indicates valid bytes in transaction.	
WVALID	1	IN		Set to '1' when master has valid data to send.	
WREADY	1	OUT		Set to '1' when slave could	

				recieve data.	
WLAST	1	IN		When set to '1' last word is being sent.	
RDATA	32	OUT		Data to be read from RAM. (it's 32 bit sample in frequency domain)	
RVALID	1	OUT		Set to '1' when RAM has valid data.	
RREADY	1	IN		Set to '1' when master could reaad	
RLAST	1	OUT		Set to '1' when last word is being sent.	
RID	ID_R_WIDTH	OUT		ID of current transaction being read from RAM.	
BREADY	1	IN		Set to '1' when master could read ID of sent transaction	
BVALID	1	OUT		Set to '1' when slave could send ID of recieved transaction.	
BID	ID_W_WIDTH	OUT		ID of already recieved transaction, will be send back to master	
SAMPLE_ram	16	OUT		Current sample to be write to RAM.	
SAMPLE_INDEX_ram	12	OUT		Address of current sample to be read to RAM.	
WRITE_ram	1	OUT		When set to '1', means that sample is being write to memory cell addressed by SAMPLE_INDEX_ram.	
READ_ram	1	OUT		When set to '1' means that data is being read from RAM	
DATA_FROM_RAM	32	IN		Sample from RAM.	
DATA_LOADED	1	OUT		Set to '1' when last sample is being loaded.	
CALC_END	1	IN		Set to '1' when whole fft has been written to RAM.	

1.2. RAM

Signals	Width	Direction		Description
SAMPLE_ram	16	IN		Current sample to be read to RAM.
SAMPLE_INDEX_ram	12	IN		Address of current sample to be read to RAM.
WRITE_ram	1	IN		When set to '1', means that RAM writes SAMPLE to memory cell addressed by SAMPLE_INDEX.
READ_ram	1	IN		When set to '1' means that data is being read from RAM.
SEND_DATA	32	IN		Copmuted data is to be stored in RAM.
SEND_ADDRESS	12	IN		Address of memory cell which it will be written SEND_DATA
LOAD/~COMPUTE	1	IN		If '1' data is being loaded from RAM to CACHE. When '0', signalizes that SEND_DATA is being written to RAM
READ_DATA	16	OUT		Data to be stored in CACHE.
READ_ADDRESS	12	OUT		Address of READ_DATA in CACHE.

1.3. FSM (finite state machine).

Signals	Width	Direction		Description	Clock domain
CE	1	IN		Enables clock.	
SAMPLES_NUMBER	12	IN		Specifes number of input samples.	
END_SEND	1	IN		Seto to '1' when data is send to cache memory.	
START_SEND	1	OUT		Set to '1' when data is going to be sending to RAM	
DATA_NUMBER	12	OUT		Number of input samples	
START_READ	1	OUT		Set to '1' when ARDATA is to be read	
LOAD/~COMPUTE	1	OUT		This signal addresses MUX, when set to '1' there is load data to cache memory. When '0' data from cache is being read.	
CALC_END	1	IN		When set to '1' means that calculation of fft has been ended.	
DATA_LOADED	1	IN		When set to '1' means that data has been loaded to RAM.	

1.4. Cache memory.

Signals	Width	Direction	Description	Clock domain
READ_DATA	32	IN	Actually taken sample from RAM given to the output.	
READ_ADDRESS	12	IN	Address of actually read sample.	
WRITE_ADDRESS	12	IN	Address of actually written data.	
DATA	32	OUT	Data stored to cache memory.	

1.5. ROM

Signals	Width	Direction	Description	Clock domain
TW_ADDR	12	IN	Address of Twiddle Factor.	
TW_VAL	32	OUT	Value of Twiddle Factor.	

1.6. Rounding unit.

Signals	Width	Direction	Description	Clock domain
UNROUNDED_SAMPLE	52	IN	Sample multiplied by Twiddle Factor, frequency domain	
ROUNDED_SAMPLE	32	OUT	Rounded sample in frequency domain	

1.7. Register

Signals	Width	Direction	Description	Clock domain
ROUNDED_SAMPLE	32	IN	Rounded sample in frequency domain	
SEND_DATA	32	OUT	Just buffered ROUNDED_SAMPLE.	

1.8. Counter_n.

Signals	Width	Direction	Description	Clock

				domain
N	12	IN	Number of samples, specifes max. value of counter	
COUNT_ENABLE	1	IN	If '1', counter counts.	
n_index	12	OUT	Current state of counter used as sample index.	
OVF	1	OUT	When counter reaches max. value, OVF is set to'1' for one CLK edge.	

1.9. Counter_k.

Signals	Width	Direction	Description	Clock domain
N	12	IN	Number of samples, specifes max. value of counter	
OVF	1	IN	If '1', counter counts.	
k_index	12	OUT	Current state of counter used as frequency index.	
CALC_END	1	OUT	When counter reaches max. value, CALC_END is set to '1', it means that last sample is computed	

3. Short description of each module.

3.1. AXI_Bridge

This module allows communication between RAM in MAC and external peripherals using AXI magistral. Includes all necessary canals.

3.2. RAM

Random access memory that stores samples of signal. It may be input samples in time domain or ready samples in frequency domain.

3.3. FSM (finite state machine)

This module controls behaviour of whole MAC circuit. Controls individual modules depends of current state of MAC.

3.4. Cache memory

This memory gets input samples and replaces them with new samples in frequency domain.

3.5. ROM

This memory keeps ready twiddle factors to calculate fft, only to read.

3.6. Rounding unit

Rounds a sample after multiplication.

3.7. Register

Keeps accumulated sample.

3.8. Counter_n

Counts number of accumulation in each sample.

3.9. Counter_k

Counts number of samples.

4. Test Cases:

TC Tag	Block Name	Brief Description	Detailed Description
TC_0010	AXI	Verify correct communication with memory via the AXI interface, using MM and burst transfers.	1. Initialize the AXI interface 2. Send a sequence of data to the RAM 3. Read data from the memory and compare with the sent values
TC_0020	AXI	Verify that the AXI interface correctly handles at least 2 simultaneous transactions.	1. Send two independent write transactions to different addresses 2. Simultaneously send two read transactions 3. Verify that all transactions have been correctly processed and the data are consistent
TC_0030	RAM	Ensure that the RAM operates correctly in the 1-port, 1-memory block configuration.	1. Write data to the RAM. 2. Read the same data. 3. Compare the written and read values.
TC_0040	RAM	Verify correct access to RAM split into 2 memory blocks.	1. Write data to the first memory block. 2. Write different data to the second block. 3. Read data from both blocks and compare with the data written.
TC_0050	RAM	Verify that the 2-port RAM allows simultaneous access.	1. Simultaneously write and read data on different ports.

			2. Verify data integrity during simultaneous operations.
TC_0060	RAM	Ensure that communication with RAM in a different clock domain is correct.	1. Simulate different clock frequencies for the main block and RAM. 2. Write and read data at various clock frequency relations.
TC_0070	MAC	Confirm the correctness of multiplication and accumulation operations.	1. Perform a series of MAC operations on known input values. 2. Compare results with expected values calculated manually or using a reference tool.
TC_0080	MAC	Verify the correctness of DFT calculations using the MAC unit.	1. Load input data into RAM. 2. Invoke the command to compute DFT using MAC. 3. Read results and compare with results from script in python 4. Comprare operation time of two modules for same amount of input samples
TC_0090	FFT Radix-4	Verify the correctness of FFT Radix-4 calculations.	1. Load input data (e.g., sine wave) into RAM. 2. Invoke the command to compute DFT using FFT Radix-4. 3. Read results and compare with results from script in python 4. Comprare operation time of two modules for same amount of input samples
TC_0100	FFT Radix-4	Ensure that data in 16-bit 2's complement format are correctly processed.	1. Input data with boundary values (e.g., maximum and minimum values). 2. Verify correctness of arithmetic operations on these data.
TC_0110	FFT Radix-4	Assess the impact of internal data width on result accuracy.	1. Set different precision values (e.g., 16-bit, 24-bit, 32-bit). 2. Perform DFT/FFT calculations with these settings. 3. Compare results with reference data and analyze errors.
TC_0120	AXI RAM MAC FFT Radix-4	Verify the correct operation of the entire system as a whole.	1. Load a set of test data into RAM. 2. Perform the full computation process (DFT/FFT). 3. Read and analyze results.
TC_0130	FFT Radix-4MAC	Compare execution times of FFT Radix-4 and MAC modules when performing DFT on the same input samples.	1. Perform TC_0080 and record the operation time. 2. Perform TC_0090 and record the operation time. 3. Compare the operation times of both modules.