ELEN6350 VLSI Design Lab

NPU---MNIST Classifier

Datasheet (V7.1)

Tianchen Yu

Tianrui Wang

Yufei Jin

Contents

1 Introduction

This NPU (Neural Processing Unit) is designed as an auxiliary device for hardware acceleration purposes, particularly for image processing purposes. “Auxiliary” means this is not a “standalone” system. There must be a master system to control this NPU and receive output from it to achieve hardware acceleration. The main acceleration from this is MAC operation, which stands for Multiply-and-Accumulate. This operation is one of the most common operations in any AI neural network inference process. As the name “NPU-MNIST Classifier” implies, the chip is initially designed for hand-written number classification task based on a MLP (Multi-Layer-Perceptron) network. However the computation section of the device is structured to be generic, thus it is applicable to any neural networks with MAC operations.

2 Features

- Operation frequency: 6.25 MHz
- 4 8-bit wide input channels, 1 8-bit wide output channel (parallel input and parallel output)
- 2 parallel MAC processing unit
- 6 different output choices
- FIFO for automatic output recording
- Comparator for automatic neuron value comparison
- Adder with automatic saturation control

3 Top Level Architecture

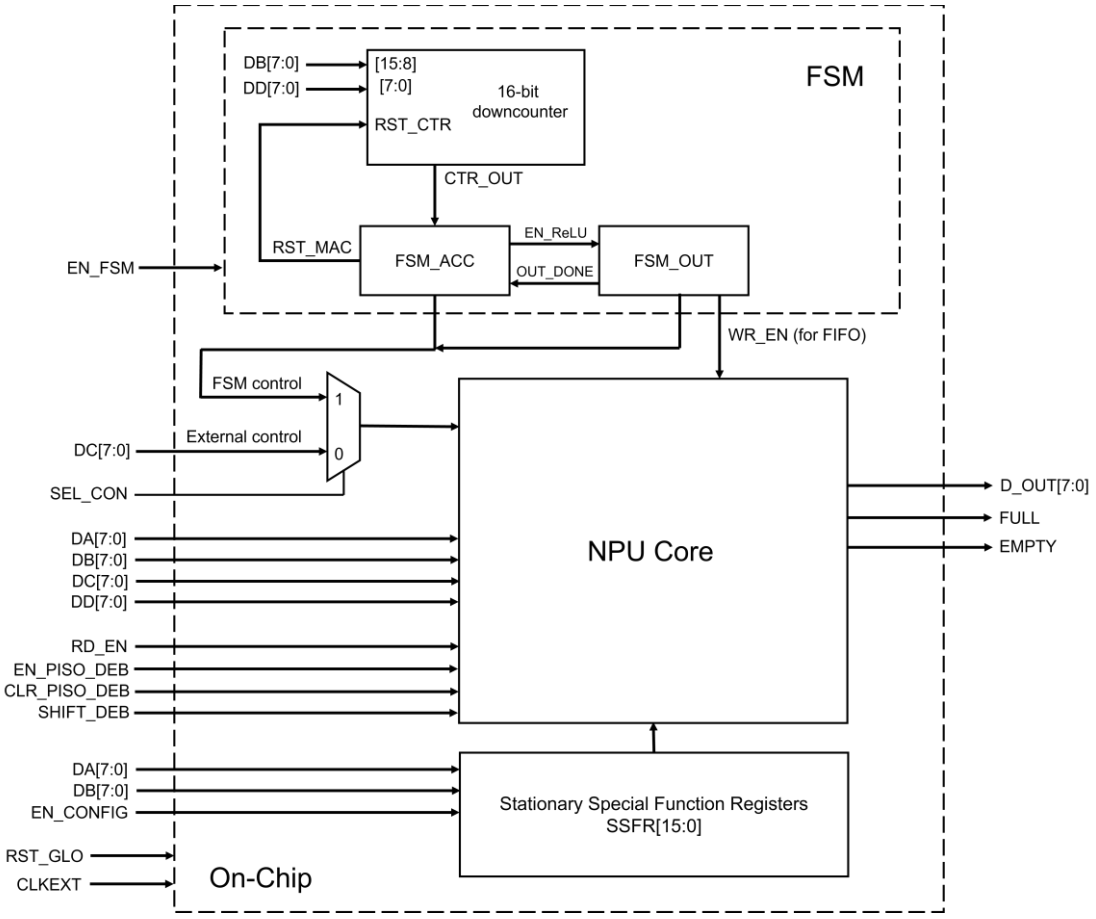

Figure 1. Top level architecture diagram of NPU

There are three main blocks in top level, namely FSM, NPU Core and SSFR (Stationary Special Function Register). NPU core includes all the datapath for computation. FSM includes two programmable finite state machines for providing controls signals to NPU Core during computation. SSFR includes 16 bits of control signals for configurating output options of NPU. It is called "Stationary" because its content remains stationary (unchanged) for most of the time, and it can only be toggled at particular timing windows. User can pull SEL_CON pin high to use the internal FSM to generate all control signals for NPU Core automatically, or it can be pulled low for NPU Core to accept external manual control signals through DC port. This will be explained in detail in **Manual Mode and Auto Mode** section.

4 Pinout

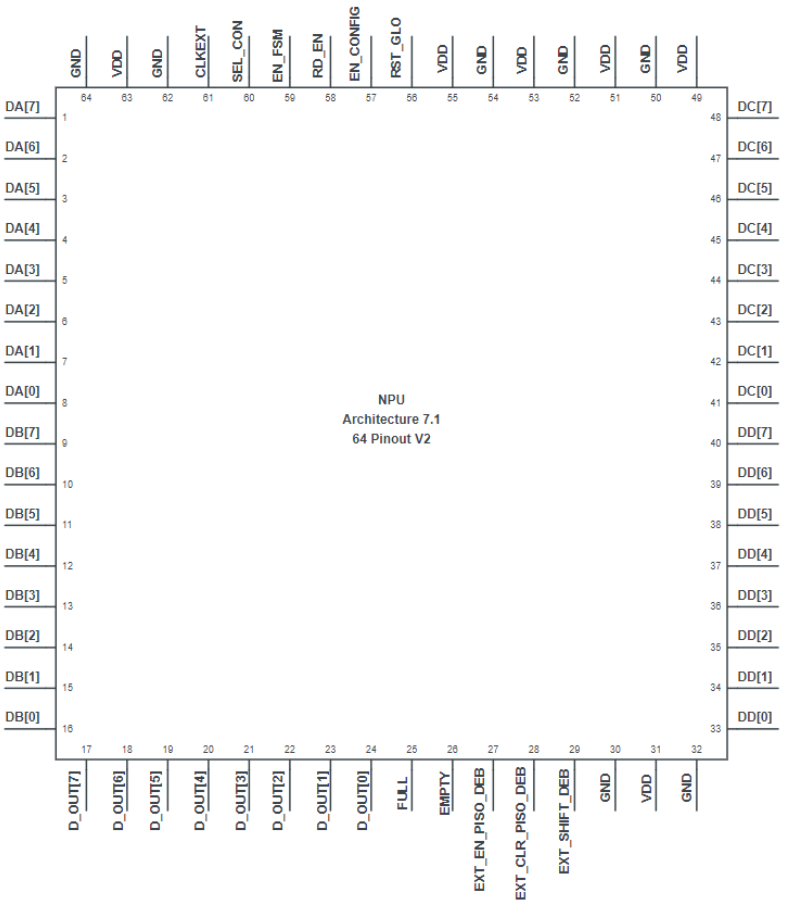

Figure 2. Pinout diagram of NPU

Regarding Figure 2, all power supply pins (VDD & GND) are placed at corners. Four input channels (32 pins) are on the left and right side. All output pins are on the bottom side.

Table 1. Pin list of NPU

Number	Type	Pin Name	Description
1-8	Input	DA[7:0]	Data parallel input channel A (8-bit wide)
9-16	Input	DB[7:0]	Data parallel input channel B (8-bit wide)
17-24	Output	D_OUT[7:0]	Data parallel output channel (8-bit wide)
25	Output	FULL	Full flag of FIFO (active high)
26	Output	EMPTY	Empty flag of FIFO (active high)

27	Input	EXT_EN_PISO_DEB	Debugging PISO enable pin
28	Input	EXT_CLR_PISO_DEB	Debugging PISO clear pin
29	Input	EXT_SHIFT_DEB	Switch of loading/shifting for debugging PISO
30	Power	GND	Ground pin
31	Power	VDD	VDD pin for IO buffers
32	Power	GND	Ground pin
33-40	Input	DD[0:7]	Data parallel input channel D (8-bit wide)
41-48	Input	DC[0:7]	Data parallel input channel C (8-bit wide)
49	Power	VDD	VDD pin for digital core
50	Power	GND	Ground pin
51	Power	VDD	VDD pin for digital core
52	Power	GND	Ground pin
53	Power	VDD	VDD pin for digital core
54	Power	GND	Ground pin
55	Power	VDD	VDD pin for digital core
56	Input	RST_GLO	Global reset pin (active high)
57	Input	EN_CONFIG	SSFR configuration enable pin (active high)
58	Input	RD_EN	Reading enable pin for output FIFO (active high)
59	Input	EN_FSM	Internal FSM triggering signal (active high)
60	Input	SEL_CON	Select pin for FSM control or manual control
61	Input	CLKEXT	External clock input
62	Power	GND	Ground pin
63	Power	VDD	VDD pin for IO buffers
64	Power	GND	Ground pin

5 Electrical Parameters

Table 2. NPU Electrical Parameters

Module	Parameter	Description	Typical Value	Unit
Power	IO_VDD	IO supply voltage	1.0	V
	CORE_VDD	Digital core supply voltage	1.0	V
	IDD_max	Maximum supply current	20	mA
Clock	CLKEXT_typ	Recommended Input clock frequency	6.25	MHz

Signal	RISE_typ	Recommended external signal rising time	<10	ns
	FALL_typ	Recommended external signal falling time	<10	ns
	EXT_EDGE	Recommended changing edge of external signal	Falling edge	/

6 NPU Core Specification

The complete architecture diagram of NPU Core is shown in Figure 3. The input data will be processed from left to right. All binary number representation assumes the decimal point is in the middle of the binary number, for both 8-bit and 16-bit (except the index output of comparator that will be explained separately). There is only one clock source called CLKEXT. Every module inside this system is synchronous to this clock source. Each module is explained as below:

6.1 Input Buffer

Four input channels (DA/DB/DC/DD) will first go through an input buffer, which is just a layer of flip-flop to control the input stream. The relevant control signals are:

EN_BUF_IN: Enable signal for input buffer
CLR_BUF_IN: Clear signal for input buffer

6.2 MAC Module

The output from the input buffer is connected to MAC modules. There are 2 MAC modules in parallel, each consisting of an 8-bit multiplier, a 16-bit 2's complement adder with saturation control (2CASC), a MUX and a flip-flop for storing previous value.

The 8-bit multiplier will take two 8-bit inputs and generate a 16-bit output, indicating that the fully multiplication precision is preserved. The multiplier is designed for 2's complement multiplication, which means it will automatically detect the sign of both inputs and generate a correct output based on the sign of two inputs.

The structure of 2CASC is shown separately in the left corner of Figure 3. The 17-bit adder is a regular adder that takes two 17-bit inputs and generates one 17-bit output. The 17-bit input is created by duplicating the MSB of 16-bit input as the new MSB. Two MSBs of the 17-bit adder output are used as the select signals for the 4-input MUX. The configuration shown in Figure 3 will make sure that the output from the adder will saturate to "16'h 7FFF" (largest positive number) when the actual result is larger than this, and it will saturate to "16'h 8000" (smallest negative number) when the actual result is smaller than this.

The 2-input MUX in MAC module is used to select the value for storage. The MAC module will either store the output from 2CASC, or it can store the data from DA or DC as shown in Figure 3. The bias value for each computation will be pre-stored to each MAC using this MUX and proper timing. 8-bit bias from DA or DC will be automatically expanded to equivalent 16-bit value when stored in MAC. The relevant control signals for MAC module are:

RST_MAC: when RST_MAC is high, the flip-flop will store the value from DA or DC, when RST_MAC is low, the flip-flop will store the value from 2CASC
EN_MAC: enable signal for MAC module

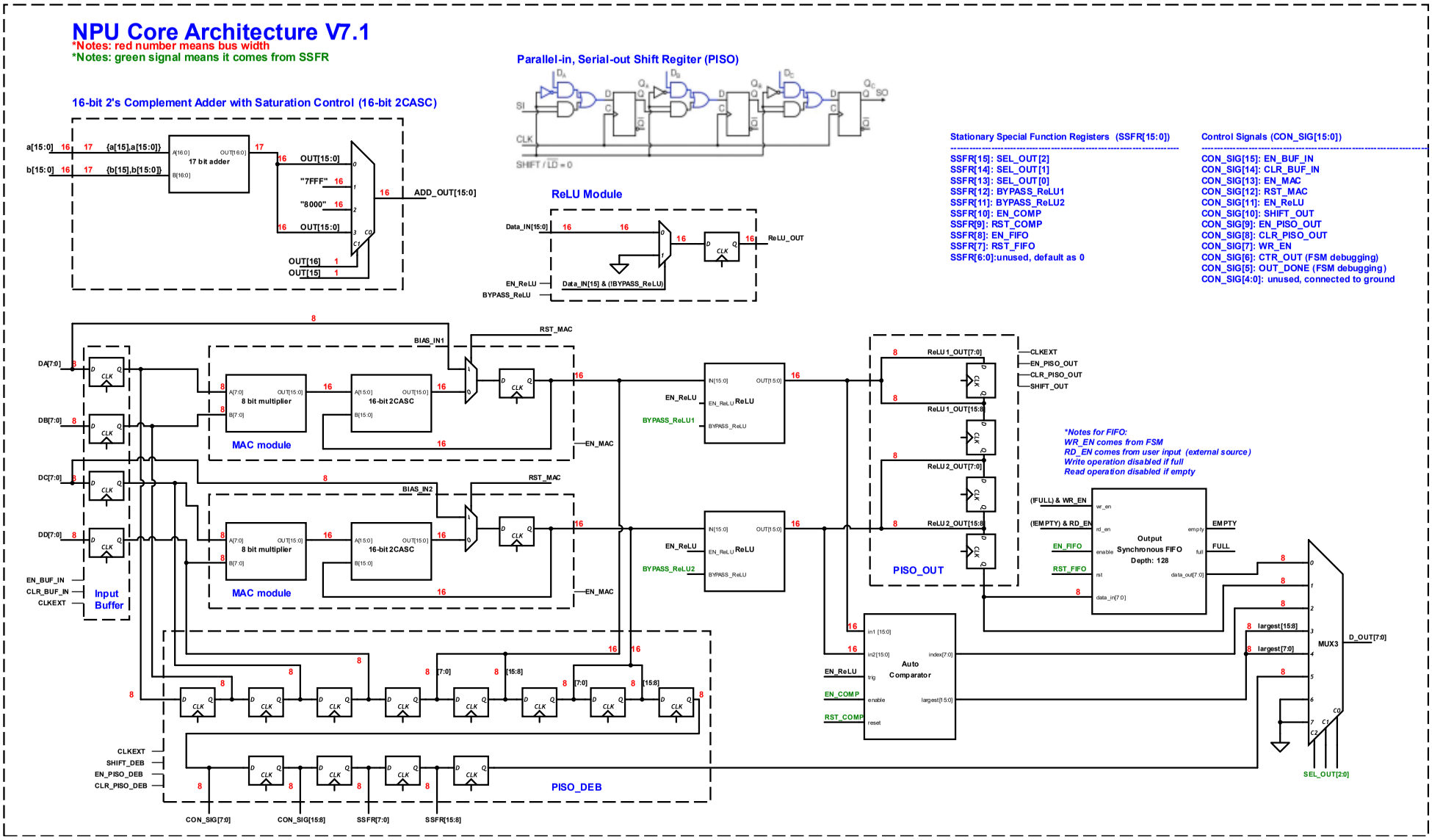

Figure 3. Architecture diagram of NPU Core (V7.1)

6.3 ReLU Module

The ReLU module provides the most common activation function to the results from MAC module. This function could be expressed as Eqn.1, which is essentially finding the maximum value between input value and 0.

$$ReLU(x) = \begin{cases} x, & x \geq 0 \\ 0, & x < 0 \end{cases} \quad \text{Eqn. 1}$$

The user has the freedom to bypass this function or not by configuring SSFR. It is recommended that the ReLU should be bypassed for the inference processing of neurons in the output layer (final layer). Allowing negative output neurons usually leads to higher overall accuracy. The relevant control signals are:

EN_ReLU: enabling signal for ReLU module. It must be high for the signal to pass ReLU module to the next block.

BYPASS_ReLU1/2: bypassing signal for ReLU module (encoded in SSFR)

6.4 Output PISO (PISO_OUT)

PISO stands for Parallel-Input Serial_Output, which is used to convert parallel inputs to serial output. The internal datapath of PISO is shown in the top region of Figure 3. The PISO_OUT is responsible for converting 2 16-bit outputs from 2 MAC modules to 4 8-bit outputs so that they are compatible with D_OUT[7:0] port. The sequence of output is shown in Figure 3. The relevant control signals are:

EN_PISO_OUT: enabling signal for PISO_OUT
CLR_PISO_OUT: clear signal for PISO_OUT
SHIFT_OUT: select signal to choose from shifting/loading mode (high for shifting)

6.5 Output MUX

The output MUX is used to select different output options for NPU. Totally there are six different output options for V7.1. These options are summarized in Table 3.

Table 3. Different output options of NPU

SEL_OUT[2:0]	D_OUT[7:0]
000	Output from FIFO
001	Output from PISO_OUT
010	Index from Comparator
011	8 MSBs of largest neuron from Comparator
100	8 LSBs of largest neuron from Comparator
101	Output from PISO_DEB
110/111	Ground (unused)

The SEL_OUT[2:0] signals are also encoded in SSFR.

6.6 Output FIFO (Depth: 128)

The idea of output FIFO is to store the results from PISO_OUT so that user has the flexibility to read the results afterwards and meanwhile the computation results will not vanish permanently after one cycle.

This FIFO follows a synchronous design, which means there is only one clock source for both reading and writing. Reading and writing can happen simultaneously within one cycle. The writing enable signal (WR_EN) can only be generated from internal FSM in automatic control mode, indicating user will not be able to use FIFO in manual control mode. The reading enable signal (RD_EN) is fully accessible as a pin so that the master system can decide when to receive data from the FIFO.

The FULL and EMPTY flags of FIFO are also accessible as dedicated pins. The operation logic is same as any regular FIFO, that is, never read from FIFO when EMPTY is high and never write to FIFO when FULL is high. To minimize the system failure risk, the FULL and EMPTY signal are used to hardcode the writing and reading enable signal. PISO_OUT will not be able to write to FIFO if FULL is high, and user will not be able to read from FIFO is EMPTY is high, regardless of the voltage level on RD_EN pin. The relevant control signals are:

EN_FIFO: enabling signal for FIFO
RST_FIFO: reset signal for FIFO. Both read pointer and write pointer will be reset if high.
WR_EN: writing enable signal, can only be generated by FSM
RD_EN: reading enable signal, accessible as a dedicated pin

The EN_FIFO and RST_FIFO are encoded in SSFR.

6.7 Auto Comparator

This comparator can compare 2 16-bit inputs with respect to the previous largest value stored in comparator and replace the stored value with current largest one. It can output the current largest value and its index based on the input sequence. The largest value stored in comparator by default is "16'h 8000", i.e., the smallest negative number. The index by default is 0, which is a reserved error code to indicate that the comparator has not worked at all. The valid index starts at 1. The number presentation of index has no decimal point (which differs from the others), indicating that the index "2" will be "8'b0000_0010". The purpose of this comparator is to accelerate the process of deciding the final recognition results. If the index of the largest neuron in the final output layer could be found, there is no need to process all output neuron values.

One example is shown here to demonstrate the operation of comparator. Assuming there are 10 output neurons and the NPU is operating in automatic control mode with 2 MAC modules enabled, the index change will be:

Table 4. Example of auto comparator operation

Inference Frame	ReLU1 output	ReLU2 output	EN_COMP	RST_COMP	Index[7:0]	Largest [15:0]
1	16'h0000	16'h0000	0	1	0	16'h8000
2	16'hFFF1	16'hFFF2	1	0	2	16'hFFF2
3	16'hFFF4	16'hFFF3	1	0	3	16'hFFF4
4	16'hFFFF	16'hFFFF	1	0	3	16'hFFF4
5	16'h0000	16'h0003	1	0	8	16'h0001
6	16'h0001	16'h0002	1	0	8	16'h0002

Usually this comparator should only be enabled for the inference of final output layer. Users may also use it for some maxpooling application as long as the timing is followed. The relevant control signals are:

EN_COMP: enabling signal for auto comparator

RST_COMP: reset signal for auto comparator, the index will be reset to 0 and the largest output will be “16’h8000”.
Trig: This signal is connected to EN_ReLU for timing alignment consideration. Users don’t need to care about it.

EN_COMP and RST_COMP are encoded in SSFR.

6.8 PISO_DEB

This is a dedicated debugging system installed in NPU Core for post-silicon debugging purposes. The debugging PISO can take samples from many different places in the system and stream them out one by one for debugging. There is a separate section for explaining the operation of debugging system. The relevant control signals are:

SHIFT_DEB: selecting signal to choose from shifting and loading mode (high for shifting)
EN_PISO_DEB: enabling signal for debugging module
CLR_PISO_DEB: clear signal for debugging module

7 SSFR Specification

SSFR (Stationary Special Function Registers) is a group of total 16 1-bit registers to control special functions of the NPU, majorly options related to the output. They are introduced to resolve the issue of limited pin numbers. All SSFR are connected to a certain module in NPU Core and the corresponding control signals are marked by green color. As shown in Figure 1, these registers are also connected to the input ports DA and DB, indicating that they can be toggled based on the values from DA and DB port. EN_CONFIG, a dedicated enable pin, is used to control the writing process to the SSFR. Data from DA[7:0] and DB[7:0] will be written to SSFR[15:0] if EN_CONFIG is high, regardless of any other operations. User should only pull EN_CONFIG high to modify the content of SSFR at appropriate timing windows according to the recommendation in **Recommended Operation Timing Diagram** section, otherwise unexpected failure could happen. All SSFR are summarized below:

Table 5. Summary of all SSFR allocation

Stationary Special Function Registers (SSFR[15:0])							
SSFR[15]	SSFR[14]	SSFR[13]	SSFR[12]	SSFR[11]	SSFR[10]	SSFR[9]	SSFR[8]
SEL_OUT[2]	SEL_OUT[1]	SEL_OUT[0]	BYPASS_ReLU1	BYPASS_ReLU2	EN_COMP	RST_COMP	EN_FIFO
SSFR[7]	SSFR[6]	SSFR[5]	SSFR[4]	SSFR[3]	SSFR[2]	SSFR[1]	SSFR[0]
RST_FIFO	Unused						
Default Values (if reset)							
SSFR[15:8]	SSFR[7:0]						
00100010	10000000						

By default, SSFR[15:0] is set to 0010_0010_1000_0000, which means the default output comes from PISO_OUT, no ReLU is bypassed, both comparator and FIFO are disabled.

8 FSM Specification

FSM (Finite State Machine) is used to automatically generate control signals to drive NPU Core. The FSM is designed to be programmable based on external input so that it is compatible with any number of accumulations. The block diagram of FSM is shown in Figure 4.

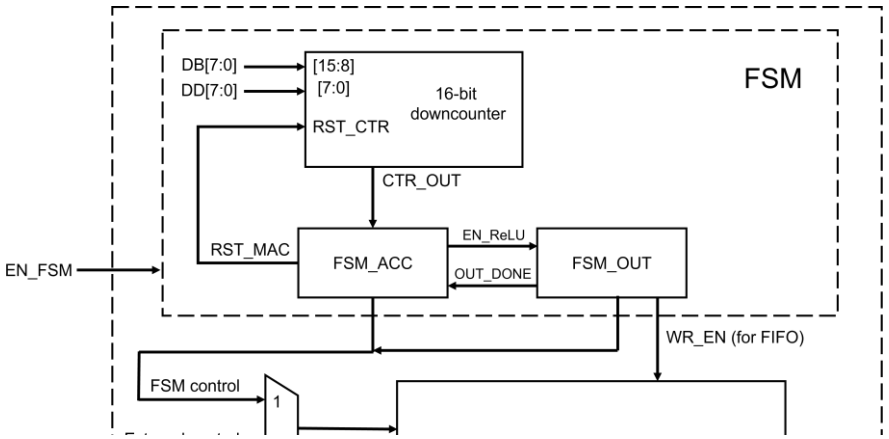

Figure 4. Block diagram of FSM

There are three main blocks in FSM, namely 16-bit downcounter, FSM_ACC and FSM_OUT. The 16-bit downcounter is a counter used to generate a flag signal to FSM_ACC for it to move to the next state. The starting point of this counter is the programmable part of FSM. Users can program this value according to the current requirement of number of accumulations in this inference frame. More details are covered in **Recommended Operation Timing Diagram** section.

The FSM_ACC and FSM_OUT are two separate FSM modules. FSM_ACC is mainly for controlling the MAC operation and FSM_OUT is for output operation. Users do not need to understand how both FSMs are structured as long as the recommended timing is followed. However if interested, the state transition diagram and table of FSM_ACC and FSM_OUT are shown below.

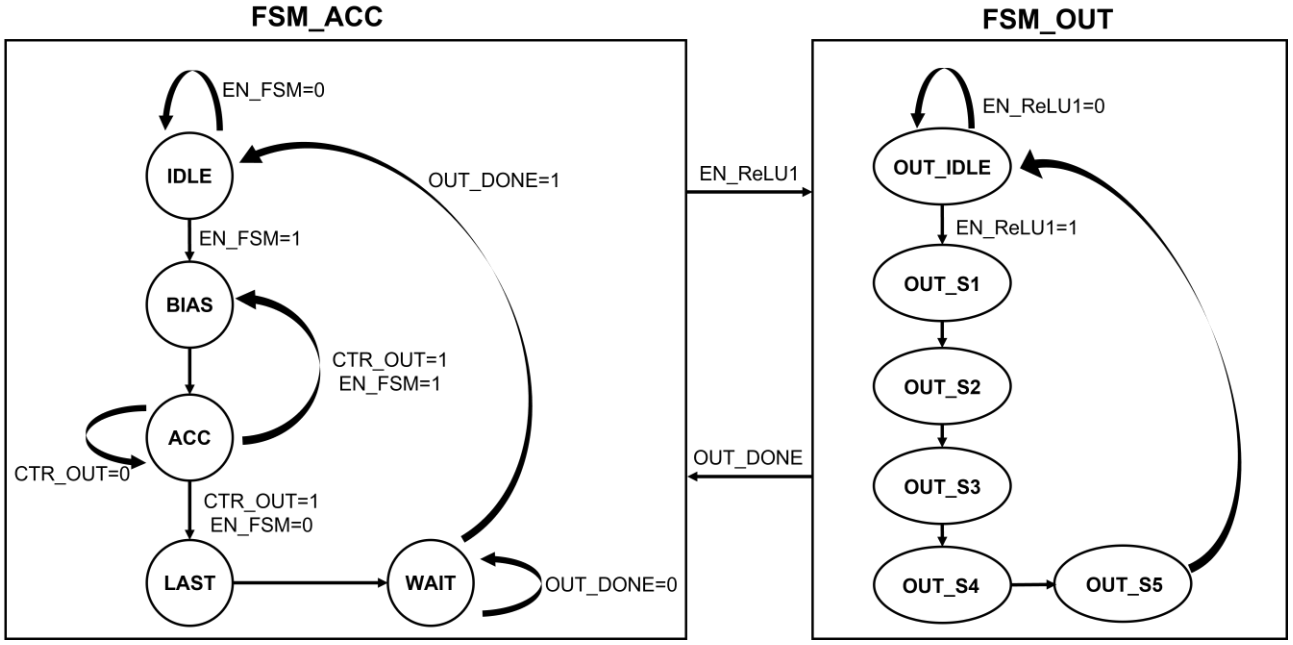

Figure 5. State transition diagram of FSM_ACC and FSM_OUT

Table 6. State transition table of FSM_ACC

FSM_ACC State Transition Table (Architecture V7.1)													
Current State	**Next State**				**Output**								
	EN_FSM=0 CTR_OUT=0	**EN_FSM=0 CTR_OUT=1**	**EN_FSM=1 CTR_OUT=0**	**EN_FSM=1 CTR_OUT=1**	**EN_BUF_IN**	**CLR_BUF_IN**	**EN_MAC**	**RST_MAC**	**CLR_PISO_OUT**	**EN_ReLU**		**ACC_FLAG**
										ACC_FLAG =0	**ACC_FLAG =1**	
IDLE	**IDLE**	**IDLE**	**BIAS**	**BIAS**	0	1	0	0	1	0	0	0
BIAS	**ACC**	**ACC**	**ACC**	**ACC**	0	1	1	1	0	0	1	No change
ACC	**ACC**	**LAST**	**ACC**	**BIAS**	1	0	1	0	0	0	0	1
LAST	**WAIT**	**WAIT**	**WAIT**	**WAIT**	0	0	1	0	0	1	1	No change
	OUT_DONE=0		**OUT_DONE=1**									
WAIT	**WAIT**		**IDLE**		0	0	0	0	0	0	0	No change

Table 7. State transition table of FSM_OUT

FSM_OUT State Transition Table (Architecture V7.1)							
Current State	**Next State**		**Output**				
	EN_ReLU=0	**EN_ReLU=1**	**Shift/~LD_OUT**	**EN_PISO_OUT**	**OUT_DONE**	**WR_EN**	
OUT_IDLE	**OUT_IDLE**	**OUT_S1**	1	0	0	0	
OUT_S1	**OUT_S2**	**OUT_S2**	0	1	0	0	
OUT_S2	**OUT_S3**	**OUT_S3**	1	1	0	1	
OUT_S3	**OUT_S4**	**OUT_S4**	1	1	0	1	
OUT_S4	**OUT_S5**	**OUT_S5**	1	1	0	1	
OUT_S5	**OUT_IDLE**	**OUT_IDLE**	1	0	1	1	

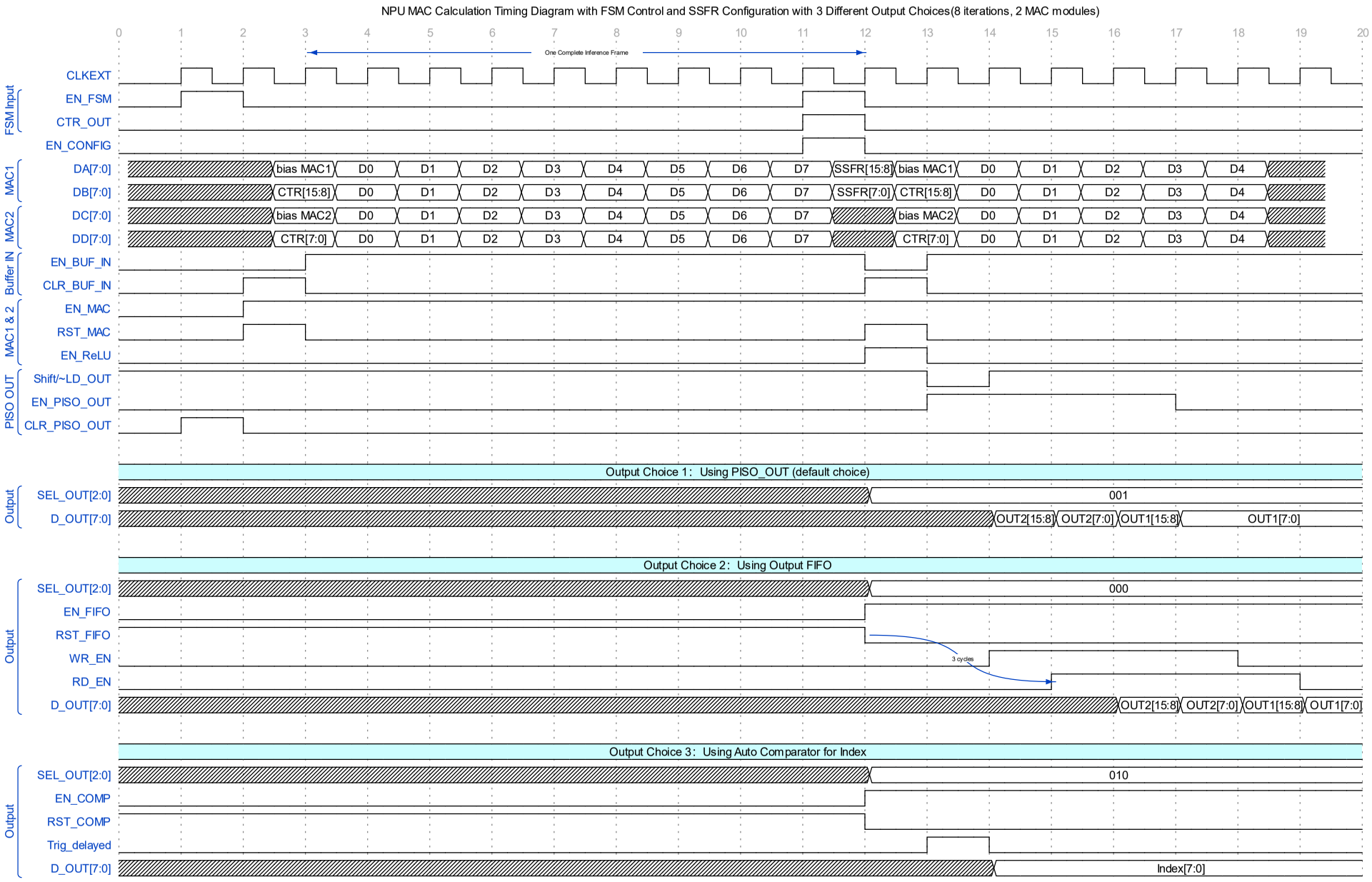

Figure 6. Recommended operation timing diagram

9 Recommended Operation Timing Diagram

The recommended operation timing diagram is shown in Figure 6. This diagram is applicable to both FSM auto control and manual control. The diagram is structured to be a generic reference for any application as it includes almost all output options. It needs to be emphasized that the timing requirement is very strict for this NPU, meaning that one single mismatch in timing could cause severe deviation from correct computation result. It is highly recommended that users always follow the timing in Figure 6.

9.1 Inference frame

The complete computation process of two neuron values is called a "inference frame", as marked by a blue arrow in Figure 6 (from cycle 3 to cycle 12). At the end of each inference frame, two neuron values are ready to be read and users can configure SSFR to choose different output options. Each inference frame is consisting of three phases:

Phase 1: one clock cycle at the beginning for inputting two biases and one starting point (N) for FSM downcounter. This starting point will be applied for both MACs. Bias for the first channel (MAC1) comes from DA[7:0] and bias for second channel (MAC2) comes from DC[7:0]. DB[7:0] together with DD[7:0] will be the starting point for the downcounter. It should be noted that the number representation of this starting point does not have decimal point. For example if the concatenated result CTR[15:0] is 16'h0100, it means the starting point is 256. The biasing value from DA and DC still follows the rule of placing the decimal point in the middle of the 8-bit number.

Phase 2: N cycles of multiply-and-accumulate process. For example if CTR[15:0] is 16'h0100, the NPU Core will assume there are 256 pairs of data to be multiplied and accumulated. In Figure 6 it will be D0, D1 …. D255. For each MAC, assuming the ReLU is not bypassed, the computation result will be:

$$MAC\ Result = ReLU(Bias + D_0 \times D_0 + D_1 \times D_1 + \cdots D_{255} \times D_{255}) \quad \text{Eqn. 2}$$

The activation of ReLU, bias value and data (D0,D1…) can be different for each MAC module, however the starting point of downcounter, i.e., the number of data pairs for accumulate, will be same for both MACs.

Phase 3: one clock cycle at the end for configuring output options by changing the values of SSFR. EN_CONFIG must be pulled high for changing the values of SSFR, though it is optional to change them or not. SSFR[15:8] is connected to DA[7:0] and SSFR[7:0] is connected to DB[7:0]. The definition of each bit in SSFR can be found in Table 5.

9.2 Alignment of two inference frames

No pause is needed between two frames as shown in Figure 6. The SSFR window of previous frame can be directly followed by the bias/counter window of the next frame for maximum throughput. For FSM auto control mode, users always need to control EN_FSM and EN_CONFIG manually by following the timing in Figure 6. EN_FSM is essentially a triggering flag for the FSM to begin, it must be high for the timing windows described in Figure 6 for continuous computation, but it can remain high between two pulses (assuming continuous operation) if easier for user to control. EN_CONFIG, however, can only be high for the particular timing windows in Figure 6 for SSFR to work correctly.

If all computation is completed, users should stop pulling EN_FSM high for the next frame. The FSM will automatically enter a "finishing sequence" to complete the final output sequence and reset itself to idle mode.

9.3 Output Choice 1: Using PISO_OUT

PISO_OUT is a basic option for receiving the accumulation results. SEL_OUT[2:0] should be set to 3'b001. 2 16-bit accumulation results will come out as 4 8-bit results in a sequence of OUT2[15:8], OUT2[7:0], OUT1[15:8], OUT1[7:0]. It should be noted that every 8-bit result must be read immediately otherwise it will vanish permanently after one cycle.

9.4 Output Choice 2: Using Output FIFO

FIFO will be able to store the output results from PISO_OUT for a maximum depth of 128 8-bit data, that is, 64 neuron values. SEL_OUT[2:0] should be set to 3'b000. EN_FIFO should be high and RST_FIFO should

be low. FIFO is only available in FSM auto control mode. Users will not be able to control WR_EN manually in manual mode. The data will be stored to FIFO in the same sequence as PISO_OUT. Reading from FIFO should follow regular manner for any other FIFO, that is, never read from FIFO when the EMPTY flag is high.

9.5 Output Choice 3: Using Auto Comparator for Index

The index from comparator can directly tell users the final recognition results without needing to compare the neuron values from the final output layer manually. SEL_OUT[2:0] should be set to 3'b010. EN_COMP should be high and RST_COMP should be low. The index[7:0] will be valid 3 cycles after the SSFR is changed and it will remain valid unless the index is updated or the output option is changed.

10 Manual Mode and Auto Mode

It is always recommended to use FSM auto mode for regular operation of NPU because it achieves highest throughput and reliable control. Manual mode should only be used if FSM control is not working properly or users want to debug the NPU. SEL_CON should be high for FSM auto control and be low for manual control.

By manual, it means users need to supply 8 control signals to NPU Core manually. They are:

- EN_BUF_IN
- CLR_BUF_IN
- EN_MAC
- RST_MAC
- EN_ReLU
- SHIFT_OUT
- EN_PISO_OUT
- CLR_PISO_OUT

If SEL_CON is set to low, the control signals from FSM will be disconnected and these 8 signals will be mapped to DC[7:0]. Meanwhile as DC port is used for control signals, the second MAC channel will be disabled automatically, which means only MAC1 is available to use thus the performance is halved. For normal operation of MAC1, users should still follow the timing in Figure 6. The timing for MAC2 should be ignored in manual mode.

Since DA/DB/DC/DD have multiple functions, a table below is created to show all alternative functions that they have.

Table 8. DA/DB/DC/DD pin reuse summary

Pin Group	Reused as	Remark
DA[7:0]	MAC1 Bias[7:0]	When RST_MAC=1
	SSFR[15:8]	When EN_CONFIG=1
DB[7:0]	CTR[15:8]	Top 8-bit of FSM counter reset value
	SSFR[7:0]	When EN_CONFIG=1
DC[7:0]	MAC2 Bias[7:0]	When RST_MAC=1
DC[7]	EXT_EN_BUF_IN	When SEL_CON=0 (MAC2 and ReLU2 will be disabled)
DC[6]	EXT_CLR_BUF_IN	

DC[5]	EXT_EN_MAC	
DC[4]	EXT_RST_MAC	
DC[3]	EXT_EN_ReLU	
DC[2]	EXT_SHIFT_OUT	
DC[1]	EXT_EN_PISO_OUT	
DC[0]	EXT_CLR_PISO_OUT	
DD[7:0]	CTR[7:0]	Bottom 8-bit of FSM counter reset value

11 Debugging Mode Specification

The debugging module is a dedicated module for post-silicon debugging. It can take samples across multiple regions within the NPU and output them in a serial manner through D_OUT[7:0]. Itself is also a PISO, similar to PISO_OUT. Users should control SHIFT_DEB, EN_PISO_DEB, CLR_PISO_DEB according to the timing diagram below for receiving the correct debugging values.

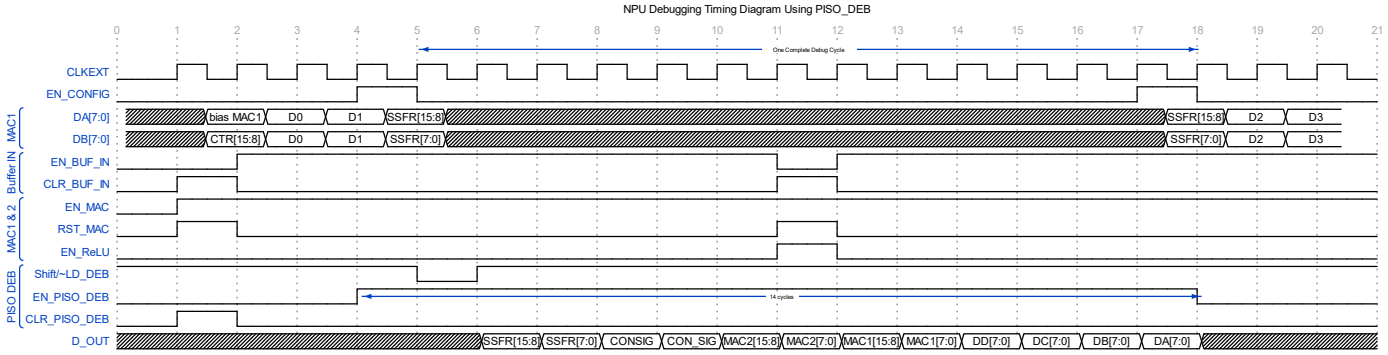

Figure 7. Timing diagram for debugging mode

In Figure 7, CLR_PISO_DEB is pulled high for one cycle to reset the PISO_DEB. During a regular operation, for example the data in cycle 4 needs to be debugged, users can pull EN_PISO_DEB high at the beginning of cycle 4, and keep it high for 14 cycles. SHIFT_DEB needs to be pulled low for one cycle after pulling EN_PISO_DEB high. EN_CONFIG should also be pulled high for one cycle to change SSFR values, primarily to change the output option to PISO_DEB. By setting EN_PISO_DEB high, the idea is that many other modules will freeze and remain unchanged for debugging. The following modules will be disabled when EN_PISO_DEB is high:

- Input buffer
- MAC1
- MAC2
- ReLU1
- ReLU2
- PISO_OUT
- Comparator
- FIFO
- FSMs

To switch output options, the following modules will still be enabled:

- PISO_DEB
- SSFR

The regular enabling signals (e.g. EN_BUF_IN, EN_MAC etc.) for the above modules will remain

unchanged during the debugging period, that is, these regular signals will be sampled by PISO_DEB during debugging, despite the module themselves are disabled. When the debugging is completed, EN_PISO_DEB should be pulled low, and EN_CONFIG should be pulled high again to switch back original SSFR values. The NPU will be able to resume its original computation from where it was stopped.

If correct timing is followed, users should receive the debugging data from D_OUT port in the following sequence:

- SSFR[15:8]
- SSFR[7:0]
- CON_SIG[15:8]
- CON_SIG[7:0]
- MAC2[15:8]
- MAC2[7:0]
- MAC1[15:8]
- MAC1[7:0]
- DD[7:0]
- DC[7:0]
- DB[7:0]
- DA[7:0]

Where CON_SIG is a unified notation for a group of control signals. They are not new signals. The specification for each bit is as follows:

Table 9. Specification of CON_SIG

CON_SIG[15]	CON_SIG[14]	CON_SIG[13]	CON_SIG[12]	CON_SIG[11]	CON_SIG[10]	CON_SIG[9]	CON_SIG[8]
EN_BUF_IN	CLR_BUF_IN	EN_MAC	RST_MAC	EN_ReLU	SHIFT_OUT	EN_PISO_OUT	CLR_PISO_OUT
CON_SIG[7]	CON_SIG[6]	CON_SIG[5]	CON_SIG[4]	CON_SIG[3]	CON_SIG[2]	CON_SIG[1]	CON_SIG[0]
WR_EN	CTR_OUT	OUT_DONE	Unused, connected to ground				