

### **Dual-Mode Configurable RISC-V Processor IP**

### 芯来科技 Nuclei System Technology



# **Dual-Mode Configurable**

Configurable feature is to meet different application scenarios:

- Real-Time System Mode:
  - ✓ Instruction and Data Local Memory (ILM, DLM)
  - ✓ Physical Memory Protection (PMP)
  - ✓ Fast Interrupt Handling and Rich Interrupt Features (ECLIC)
- Application System Mode:
  - ✓ Instruction and Data Cache (I-Cache, D-Cache)
  - ✓ Memory Management Unit (MMU)
  - ✓ Private Timer
  - ✓ Platform Level Interrupt Controller (PLIC)
  - ✓ High Speed System Bus (AXI)



## **Dual-Mode Programmable**

So here Dual-Mode configurable can be:

- Either Real-Time mode OR Application mode related features are configured
- Both the Real-Time mode AND Application mode related features are configured
  - ✓ Programmable: After Reset, SW can Enable/Disable features for the required MODE



## **Nuclei 600 Series Processor**

**600** Series is configurable processor to meet different application scenarios.

- RISC-V RV32/64-I/M/A/C/F/D/P ISA supported
- 5-7 pipeline stages
- Configurable ILM (Instruction Local Memory) & DLM0/DLM1 (Data Local Memory) with ECC
- Configurable I-Cache & D-Cache with ECC
- **64-bit AXI** system bus, configurable 64-bit AXI slave port
- Besides Machine mode & User mode, Supervisor mode is suppored for MMU (RISC-V SV39 Mode)
- Configurable **NICE** interface for **user-defined extesnions**
- Configurable **ECLIC** (enhanced core level interrupt controller) or **PLIC** (platform level interrupt controller)
- 4-wire JTAG debug ports supported



# 600 Configurable ISA

DSP (P-Extension)						
Supervisor Supported for TEE or MMU	RV <i>32/64</i> I/M/A/C	NICE - User Defined Extensions				
SP & DP FPU (F/D Extension)						

- Configurable & Extensible
- FPU options with both Single Precision & Double Precision
- DSP option with SIMD, partical SIMD, 64bit, Non-SIMD instructions
- Supervisor mode supported for Hardware-Software Codesign Penglai TEE in N/NX class cores
- Supervisor mode supported for MMU in **UX** class cores
- Instruction & Data closely coupled Icoal memories
- NICE interface for user-defined instruction extensions



# **NICE (Nuclei Instruction Co-unit Extension)**

**NICE** allows customers to add user-defined instructions to customize their processor implementation, also including the extension of tightly coupled register and memory access instructions.



NUCLE

## Low-Power Micro-Architecture Design

- 5-7 Pipeline Stages
- Various Low-Power Design (Clocking gating, Logic gating, etc.)



# **Sleep Modes**

- Two Sleep Modes
  - Sleep mode & deep sleep mode, controlled by SoC MPU

Power Consumption

- Entering Sleep Mode
  - WFI (wait for interrupt)
  - WFE (wait for event)
- Wake Up
  - NMI
  - Interrupt
  - Event
  - Debug Request





### **Memory Resources**

#### • ILM (Instruction Local Memory)

- Configurable an independent SRAM interface & address space
- DLM (Data Local Memory)
  - Configurable an independent SRAM interface & address space

Configuration	Comment
600_CFG_HAS_ILM	■This Macro configures to have ILM.
600_CFG_ILM_BASE_ADDR	This Macro to configure the base address of the ILM.
600_CFG_ILM_ADDR_WIDTH	This Macro to configure the address space of ILM.
600_CFG_HAS_DLM	■This Macro configures to have DLM.
600_CFG_DLM_BASE_ADDR	This Macro to configure the base address of the DLM.
600_CFG_DLM_ADDR_WIDTH	This Macro to configure the address space of DLM.
600_CFG_HAS_LM_SLAVE	This Macro configures to have slave port for ILM/DLM access
600_CFG_HAS_LM_ECC	This Macro configures to have ECC protections for ILM and DLM

#### • Instruction Cache

- n-way associative, 4KB/way, n is configurable
- Cache line size is 32 Byte

### • Data Cache

- 2-way associative, Cache line size is 32 Byte
- Cache Size is configurable

Configuration	Comment			
600_CFG_HAS_ICACHE	This Macro configures to have I-Cache			
600_CFG_ICACHE_WAY	This Macro to configure n-way associate, n=2,4,8			
Configuration600_CFG_HAS_ICACHE600_CFG_ICACHE_WAY600_CFG_HAS_DCACHE600_CFG_DCACHE_ADDR_WIDTH600_CFG_HAS_CACHE_ECC600_CFG_DEVICE_REGIONn_BASE600_CFG_DEVICE_REGIONn_MASK600_CFG_NONCACHEABLE_REGIONn_BASE600_CFG_NONCACHEABLE_REGIONn_MASK	This Macro configures to have D-Cache			
600_CFG_DCACHE_ADDR_WIDTH	■This Macro to configure the D-Cache size			
600_CFG_HAS_CACHE_ECC	This Macro configures to have ECC protections for I-Cache and D-Cache			
600_CFG _DEVICE_REGIONn_BASE	■This Macro to configure the base address of Device Region n, n=0~7			
600_CFG _DEVICE_REGIONn_MASK	■This Macro to configure the MASK value of Device Region n, n=0~7			
600_CFG _NONCACHEABLE_REGIONn_BASE	■This Macro to configure the base address of Non-Cacheable Region n, n=0~7			
600_CFG _ NONCACHEABLE _REGIONn_MASK	■This Macro to configure the MASK value of Non-Cacheable Region n, n=0~7			
600_CFG_HAS_CACHE_ECC       Inits Matrix Configures to have protections for I-Cache and D-C         600_CFG_DEVICE_REGIONn_BASE       Inits Matrix Configures to have protections for I-Cache and D-C         600_CFG_DEVICE_REGIONn_BASE       Inits Matrix Configures to have protections for I-Cache and D-C         600_CFG_DEVICE_REGIONn_MASK       Inits Matrix Configures to have protections for I-Cache and D-C         600_CFG_DEVICE_REGIONn_MASK       Inits Matrix Configures to have protections for I-Cache and D-C         600_CFG_NONCACHEABLE_REGIONn_BASE       Inits Matrix Configures the base of Device Region n, n=0~7         600_CFG_NONCACHEABLE_REGIONn_BASE       Inits Matrix Configures the base of Non-Cacheable Region n, n=0         600_CFG_NONCACHEABLE_REGIONn_MASK       Inits Matrix Configures the Matr				

### **ECC on SRAMs**

- ECC Mechanism: SECDED (Single Error Correction, Double Error Detection)
- ECC protection granularity:
  - ILM and I-Cache Data-Ram: 64-bit
  - DLM and D-Cache Data-Ram: 32-bit
  - I/D-Cache Tag-Ram and TLB: their Actual Size

### • ECC Full Write and Partial Write

- Full Write: data and corresponding ECC code will be updated simultaneously, high efficiency
- Partial Write: Read-Modify-Write sequency will be triggered when 8/16-bit writes, less efficiency

### • ECC Error Injection

- mecc\_code CSR is implemented for ECC error injection
- Can be configured to do ECC error injection on ILM, DLM, I-Cache/D-Cache/TLB Tag-Ram or Data-Ram
- ECC Lock
  - ECC related CSRs cannot be modified after ECC is locked unless Reset, for Security



## **CCM (Cache Control and Maintenance)**

- CCM is defined for SW to Control and Maintenance the internal I-Cache and D-Cache
- CCM Types:
  - by-ADDR and by-ALL
  - I-Cache: INVAL, INVAL\_ALL, LOCK and UNLOCK
  - **D-Cache**: INVAL, FLUSH, FLUSH&INVAL, INVAL\_ALL, FLUSH\_ALL, FLUSH&INVAL\_ALL, LOCK and UNLOCK

#### • M/S/U mode has its own CCM operations

- **S/U** can execute CCM operations without needing switching privilege mode, but need to pass permission checking
- 'Illegal instruction exception' will be triggered when lower privilege mode operates on higher privilege mode CCM CSRs
- INVAL will be upgrade to be FLUSH&INVAL in U-mode for **Security**

# • CCM operations can still work on the Disabled cache

Туре	Operation	Codes	Description
-Cache	INVAL	5b01_000	When Cache hit, Unlock and Invalid the specific cacheline; Ignored when cache miss.
peration	LOCK	5b01_011	When cache hit, Lock the specific cacheline; When cache miss, Refill the specific cacheline then Lock it. Check ccm_data to check the Lock succeed or not.
	UNLOCK	5b01_100	When cache hit, Unlock the specific cacheline; Ignored when cache miss.
	INVAL_ALL	5b01_101	Unlock and Invalid ALL the cacheline.
D-Cache	INVAL	5boo_000	When cache hit, Unlock and Invalid the specific cacheline; Ignored when cache miss.
peration	WB	5b 00_001	When cache hit and Dirty, Flush the specific cacheline; When cache miss or hit but not dirty, ignored. Lock bit is not affected.
	WBINVAL	5b 00_010	When cache hit, Unlock and Flush and Invalid the specific cacheline; Ignored when cache miss.
	LOCK	5b 00_011	When cache hit, Lock the specific cacheline; When cache miss, Refill the specific cacheline then Lock it. Check ccm_data to check the Lock succeed or not.
	UNLOCK	5b 00_100	When cache hit, Unlock the specific cacheline; Ignored when cache miss.
	INVAL_ALL	5b 10_111	Unlock and Invalid ALL the cacheline.
	WB_ALL	5b 00_111	Flush ALL the Valid and Dirty cachelines; Lock bit is not affected.
	WBINVAL_ALL	5b 00_110	Unlock and Flush and Invalid ALL the Valid and Dirty cachelines.



### **Bus Interfaces**

- **System Bus Interface** 64bit AXI with integer clock ratios
- ILM Bus Interface 64bit (configurable) SRAM interface, for accessing private instruction local memory
- DLM Bus Interfaces 2 32bit (configurable) SRAM interfaces, for accessing private data local memory (DLM0/DLM1)
- Private Peripheral Interface (PPI) 32bit, AHB-Lite interface protocol for accessing private peripherals
- **Slave Port** 64bit AXI interface for other masters to access ILM/DLM0/DLM1



## **Enhanced Core Local Interrupt Controller**

- ECLIC (Enhanced Core Local Interrupt Controller)
  - Optimized based on the RISC-V standard CLIC for fast interrupt handling scheme, **compatible with CLIC**
  - Private inside the core
  - Enabled by setting the LSB bits of CSR register mtvec as CLIC/ECLIC mode
  - Configurable number of interrupt levels and priorities
  - Support interrupt preemptions based on interrupt levels
  - Support vectored interrupt processing mode for extremely fast interrupt response (6 cycles)
  - Support fast interrupts tail-chaining mechanism (non-vectored)



CSR Registers	Comment					
mtvt	ECLIC Interrupt Vector Table Base Address					
mnxti	Used to enable taking the next interrupt and return the entry address of the next interrupt handler.					
mintstatus	Current Interrupt Levels					
mnvec	Customized register used to indicate the NMI handler entry address					
mmisc_ctl Customized register controlling the selection of the NMI Handler Entry Address.						
msavestatus Customized register storing the value of mstatus.						
mtvt2	Customized register used to indicate the common handler entry address of non-vectored interrupts.					
jalmnxti	Customized register used to enable the ECLIC interrupt. The read operation of this register will take the next interrupt, return the entry address of next interrupt handler, and jump to the corresponding handler at the same time.					
pushmcause	Customized register used to push the value of mcause into the stack memory.					
nushmenc	Customized register used to push the value of menc into the stack memory					

# **Platform Level Interrupt Controller**

- PLIC (Platform Level Interrupt Controller)
  - RISC-V standard, for Linux Capable or SMP applications
  - Shared outside of the core
  - Enabled by setting the LSB bits of CSR register **mtvec** as CLINT mode
  - Up to 1024 Interrupts supported: level or edged
  - Configurable number of interrupt priorities
  - Support interrupt to M-mode or S-mode
  - PLIC Control Registers:
    - ✓ Interrupt Enable
    - ✓ Interrupt Pending
    - ✓ Interrupt Priority
    - ✓ Interrupt Threshold
    - ✓ Interrupt Claim/Complete



# **PMP (Physical Memory Protection)**

- Configurable PMP, providing per-hart machinemode control registers to allow physical memory access privileges (read, write, execute) to be specified for each physical memory region.
  - Configurable PMP entries, up to 16
  - The granularity of PMP is **4KB**
  - TOR mode is not supported
  - PMP checks are applied to all accesses when the hart is running in S or U modes; And for loads and stores when the MPRV bit is set in the mstatus register and the MPP field in the mstatus register contains S or U



# **TEE (Trusted Execution Environment)**

- RISC-V Privileged ISA based TEE Framework
- Smallest Trusted Code Base
  - RISC-V core (PMP/sPMP) + Verifiable security monitor (M-mode privilege) + TEEOS
- Secure Assurance
  - Strong isolation between enclave and other application or OS
  - Protect against a malicious or compromised OS
  - Secure boot and remote attestation for chain of trust
  - High performance and scalability



#### Penglai HEAVY.light architecture

- **light Zone:** A dedicated HW-isolated box for a single enclave
- **HEAVY Zone:** Multiple-Enclaves isolated through TEEOS

# **MMU (Memory Management Unit)**

- MMU in **UX** class cores can enable Linux capable applications
- RISC-V RV39 mode:
  - ✓ Page based 39-bit virtual memory system, mapping to 56-bit physical memory space
  - ✓ Permission checking (eXecutable, Writeable, Readable)
- Two level TLBs (Translation Lookaside Buffer) in MMU to cache page tables for fast accessing:
  - ✓ Main TLB: can be configured as 32, 64 or 128 entries
  - ✓ Micro TLB (I-TLB, D-TLB): each has 8 entries
- MMU supports 4KB, 2MB and 1GB page types
- Hardware Translation Table Walk mechanism when TLB missing without software handling



### **Nuclei Software Development Platform**

Board Application	Board Labs	TEE APP				
Nuclei SDK		TEE SDK				
NMSIS(Core/DSP/NN)	Third-party Library					
Nuclei Spec(ISA, DSP, TEE)						
Nuclei Processor Core Based Devices						



## Nuclei Studio IDE

- Eclipse based
- Integrated GCC and OpenOCD
- Libre and free
- Portable executables, without installation
- Easy-to-use project template
- Integrated editor
- In system debugging
- In system programming
- Integrated serialport tool
- Real time register display

RISCV Workspace - GD32VF103_DEMO/Application/main.c - Nucl File Edit Source Refactor Navigate Search Project Run Winds	eiStudio				1.000	Contract of the second second	1000	10 C	A REPORT OF THE	a x
	nvineip Nielaria in state da sie sie			t a set lis a st	- the char				Quick Assess	1
S C DEDUG COSTONO_ CONTON				1.00 4121 - 01			-		Quick Access	1 80 1 100 (34
<ul> <li>CB32VF103 DEMO_Debug_OpenOCD [GDB OpenOCD Debuggi</li> <li>CD32VF103 DEMO_Debug_OpenOCD [GDB OpenOCD Debuggi</li> <li>CD32VF103 DEMO_deff</li> <li>Thread #1 (Suspended : Breakpoint)</li> <li>main() at main.c:55 0x8000e9a</li> <li>copenced exe</li> </ul>	ng] 36 #include "gd32vf103. ng] 36 #include "gd32vf103. 37 #include "systick.h" 38 #include <stdio.h> 39 400 /*! 41 \brief main fun</stdio.h>	2 @ main.c #: @ gd32/fl03.h @ [gdb[0].proc[42000].thr. 36 #include "gd12/fl03/_evel.h" 37 #include "systick.h" 38 #include <stdio.h> 39 40+/*! 41 \brief main function</stdio.h>		08000e82: 08000e84: 49 08000e88: 08000e88: 08000e8a: 50	li jal gd_eva li jal gd_eva	Enter location h + 1 4 0 (1998) 12 5 * a0,0 i ra,0x3000400 <gd_eval_led_init> d_eval_led_init(1DD); a0,1 1 ra,0x30040400 <gd_eval_led_init> d_eval_led_init(1DD); }</gd_eval_led_init></gd_eval_led_init>		<ul> <li>✓ ariables Steakpoints</li> <li>✓ e [function: main] [ty</li> <li>✓ e main.c [line: 55]</li> <li>✓ e main.c [line: 59]</li> <li>✓ e main.c [line: 63]</li> </ul>	▲ "Lexpressions ■ Modules 業務委会系  (pe: Temporary]	•••
I riscv-none-embed-gdb	42 \param[in] none 43 \param[out] none 44 \retval none			08000e90: 51	jal gd_eva	a0,2 ra,0x8000400 <gd_eval_led_init> l_led_init(LED4);</gd_eval_led_init>		No details to display for th	ne current selection.	
	45 */ 46=int main(void) 47 {			08000e94: 08000e96: 	jal gd	a0,3 ra,0x8000400 <gd_eval_led_init> _eval_led_on(LED1);</gd_eval_led_init>		Im Registers 22	2000 DI 11	8 0
a Console 🛛 👘 🖉 🖉 🖉 🖉 🖉 🖉 🖉 🖉	<pre>B • P D 48 gd_eval_led_init(LED</pre>	1);		• 08000e9a:	li	a0,0		Name	Value	
D32VF103_DEMO_Debug_OpenOCD [GDB OpenOCD Debugging] (	ppenocd.ex 49 gd_eval_led_init(LED	2); 3):		56	jai gd	eval led off(LED4);	-	* M General Registers		10
nto : ciscening on por c same for ceiner connections nfo : accepting 'gdb' connection on tcp/3333	51 gd_eval_led_init(LED	t);		08000ea0:	li	a0,3		111 zero	0	
nfo : device id = 0x19060410	52 53 sebt1a(1)/			08000ea2:	jal de	ra,0x80004c0 <gd_eval_led_off> lav_lms(1000):</gd_eval_led_off>		in ra	0x8000e9a <main+33< td=""><td>2&gt;</td></main+33<>	2>
nfo : flash_size_in_kb = 0x000000000 nfo : flash_size_= 120kbutes	54 /* turn on led1,	turn off led4 */		08000ea6:	li	a0,1000		IIII sp	0x20007ff0	
nfo : cmsis-dap JTAG TLR_RESET	P55 gd_eval_led_on(L	ED1);		08000eaa:	jal	0x8000ee4 <delay_1ms></delay_1ms>		IIIT gp	0x20000890	
nfo : cmsis-dap JTAG TLR_RESET	57 gd_eval_ied_off( 57 delay 1ms(1000);	.EU4);		08000eac:	li	a0,1		im tp	0x0	
nfo : JTAG tap: auto0.tap tap/device found: 0x790007a3 (#fg: 0x3d1 (	GigaDevic 58 /* turn on led2,	turn off led1 */		08000eae:	jal	ra,0x8000484 <gd_eval_led_on></gd_eval_led_on>		IIII tO	536870960	
nfo : Padding image section 0 at 0x08000238 with 8 bytes	259 gd_eval_led_on(L 68 ed_eval_led_off(	502); F01):		08000eb2:	li	_eval_led_ott(LED1); a0.0		un ti	0	
nfo : Padding image section 1 at 0x08001fae with 2 bytes nfo : cmsis-dap JTAG TLR RESET	61 delay_1ms(1000);			08000eb4:	jal	ra,0x80004c0 <gd_eval_led_off></gd_eval_led_off>		## t2	0	
nfo : cmsis-dap JTAG TLR_RESET	62 /* turn on led3,	turn off led2 */	=	61 08000ab8+	de	lay_1ms(1000);	1.00	III fp	0×20008000	
nfo : cmsis-dap JTAG TLR_RESET	GigeDenic 64 gd eval led off(	ED2);		08000ebc:	jal	0x8000ee4 <delay_1ms></delay_1ms>		IT S1	0	
nfo : cwsis-dap JTAG TLR_RESET	65 delay_1ms(1000);		-	263 00000-h	gd	_eval_led_on(LED3);		a0	1073813504	
nfo : cmsis-dap JTAG TLR_RESET	67 gd eval led on()	turn off led3 */		08000ec0:	jal	a0,2 ra.0x8000484 <gd eval="" led="" on=""></gd>		<sup>221</sup> al	16	
nto : cmsis-dap JIAG ILE_NESEI nfo : JTAG tap: auto0.tap tap/device found: 0x790007a3 (mfe: 0x3d1 (	GigaDevic 68 gd_eval_led_off(	ED3);		64	gd	_eval_led_off(LED2);		um a2	3	
==== RISC-V Registers	69 delay_1ms(1000);			08000ec4:	ial	a0,1 ra 0x80004c0 kgd eval led offs		an a3	1073813524	
(0) zero (/32) (1) na (/32)	71 }			65	de	lay_1ms(1000);		III a4	536870928	, '
*	, 72		-	08000eca:	li	a0,1000				-
A Peripherals 21	a	Debugger Conserve B Manager								
Peripheral Address Description	Monitors	🔶 🗶 💥 🚘	DAC: 0x4	40007400	New R	Renderings			**	
ADC0 0x4001 Analog to digital converter	DAC	R	Register				Add	iress	Value	
ADC1 0x4001 Analog to digital converter	10		5 DAC				0x4	0007400		
AFIO 0x4001 Alternate-function I/Os			HI CT	L			0x4	0007400	0x00000000	
BKP 0x4000 Backup registers			> 101 SW	л			0x4	0007404	0x0000000	
CANO 0x4000 Controller area network			III DA	CO R12DH			0×4	0007408	0x00000000	
CAN1 0x4000 Controller area network			> III DA	C0 L12DH			0x4	000740C	0x00000000	
CRC 0x4002 Cyclic redundancy check calculation unit			I III DA	CO REDH			0x4	0007410	0x00000000	
Z DAC 0x4000 Digital-to-analog converter			I III DA	C1 R12DH			0x4	0007414	0x00000000	
DBG 0xE004 Debug support			I III DA	C1 L12DH			0x4	0007418	0x00000000	
DMA0 0x4002 DMA controller			IN DA	C1_R8DH			0x4	000741C	0x00000000	
DMA1 0x4002 Direct memory access controller			DA	CC_R12DH			0x4	0007420	0x00000000	
ECLIC UXD20 Enhanced Core Local Interrupt Controller			I III DA	CC_L12DH			0x4	0007424	0x00000000	
EXMC 0xA000 External memory controller			> IN DA	CC R8DH			0x4	0007428	0x00000000	
EXTI 0x4001 External interrupt/event controller			ant DA	CO_DO			0x4	000742C	0x00000000	
			- Int DA	C1_DO			0x4	0007430	0x00000000	
				Mark Breech						



# **Supported 3rd Party Tools**



**SEGGER** 

### LAUTERBACH

IAR







## **THANK YOU**

