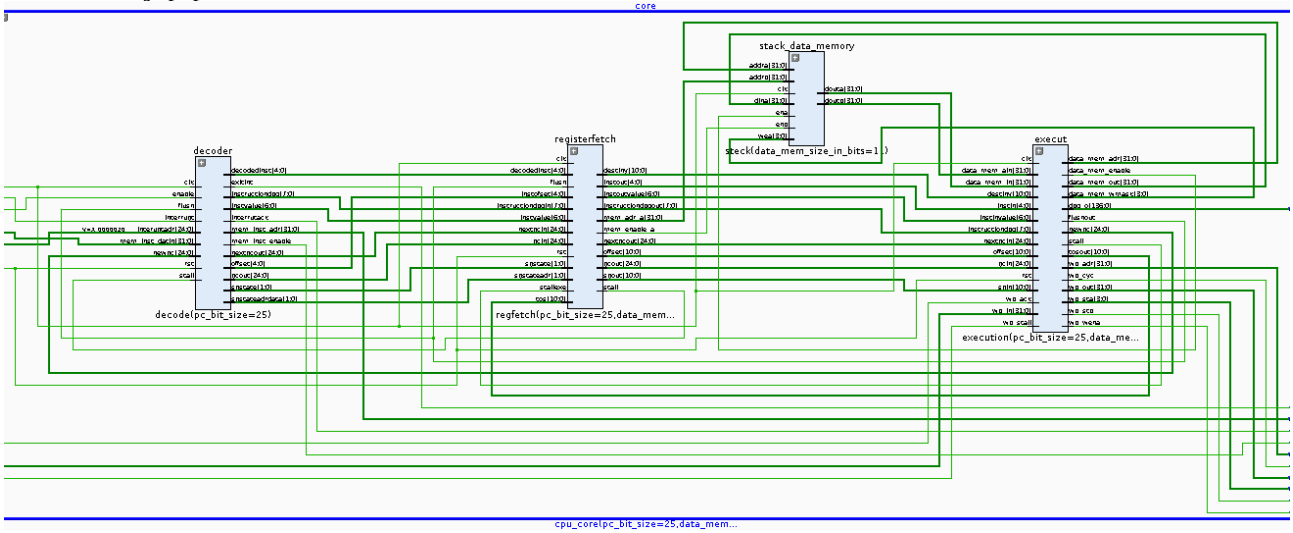
Core:

fully pipleined core with dedicated hardware stack.



Designed core

it has wisbone interface with a mux on ram so it can support slave wisbone bus to ram/rom not to stack for dma.

wisbone bus:
wb_in: 32 bit wisbone data in
wb_out: 32 wishbone data out
wb_sel: 4 bit wishbone select
wb_stb: wishbone strobe
wb_cyc: wishbone cycle
wb_ack: wishbone ack
wb_stall: If an other master is conected to this device it will bw 1
wb_we: wishbone write enabe
wb_adr: 32 bit wishbone address

Instrruction set

Mnemonic	Opcode binary	Description	Clocks pipelined	Clocks pipelined midle
BREAKPOINT	00000000	Show if executed on a signal cpu can use as nop	1	1
IM x	1xxxxxxx	Push or append 7-bit immediate	1	1

STORESP x	010xxxxx	Pop and store to stack with offset	1	1
LOADSP x	011xxxxx	Push value present on stack with offset	1	1
ADDSP x	0001xxxx	Add value present on stack with offset to TOS	1	1
EMULATE x	001xxxxx	Emulate instruction. See emulation table.	3	3
POPPC	00000100	Pop current value and set PC	2+2 because clear pipeline	2+2 because clear pipeline
LOAD	00001000	Pop current value and push memory value	3	3
STORE	00001100	Store NOS into memory pointed by TOS, Pop both	2	2
PUSHSP	00000010	Push current SP	1	1
POPSP	00001101	Pop and store into SP	3	3
ADD	00000101	Integer addition	1	1
AND	00000110	Bitwise AND	1	1
OR	00000111	Bitwise OR	1	1
NOT	00001001	Bitwise NOT	1	1
FLIP	00001010	Flip bits	1	1
NOP	00001011	NO-Operation	1	1
POPINT	00000011 if POPINT enabled	Pop current value and set PC and reenable interrupt controller	2+2 because clear pipeline	2+2 because clear pipeline
Optional instructions				
Mnemonic	Opcode decimal	Description	Clocks pipelined	Clocks pipelined midle
LOADH	34	Load short	3	emulated
STOREH	35	Store short	2	emulated
LESSTHAN	36	Signed less-than comparison	1	1
LESSTHANOREQUAL	37	Signed less-than-or-equal comparison	1	emulated
ULESSTHAN	38	Unsigned less-than comparison	1	1
ULESSTHANOREQUAL	39	Unsigned less-than-or-equal comparison	1	emulated
MULT	41	Signed 32-bit multiplication	3	3
LSHIFTRIGHT	42	Logical shift right	1	emulated
ASHIFTLEFT	43	Arithmetic shift left	1	emulated

ASHIFTRIGHT	44	Arithmetic shift right	1	emulated
CALL	45	Call function	2+2 because clear pipeline	emulated
EQ	46	Comparison equal	1	1
NEQ	47	Comparison not equal	1	emulated
NEG	48	Negative	1	emulated
SUB	49	Subtract	1	1
XOR	50	Exclusive-OR	1	emulated
LOADB	51	Load byte	3	3
STOREB	52	Store byte	2	2
DIV	53	Signed 32-bit division	emulated	emulated
MOD	54	Signed 32-bit modulus	emulated	emulated
EQBRANCH	55	Branch if equal	2(+2 because clear pipeline if branch)	emulated
NEQBRANCH	56	Branch if not equal	2(+2 because clear pipeline if branch)	2(+2 because clear pipeline if branch)
POPPCREL	57	Pop PC relative	2+2 because clear pipeline	emulated
CONFIG	58	Internal configuration	emulated	emulated
PUSHPC	59	Push current PC	1	1
PUSHSPADD	61	Push SP + offset	1	1
CALLPCREL	63	Call relative function	2+2 because clear pipeline	2+2 because clear pipeline

Registers:

PC: program counter
SP: stack pointer
TOS: top of stack
NOS: next of stack

Interrupt controler:

it has 32 interrupt source 3 internal 29 external.

The interrupt vector is 0x20 at this address save R0 R1 R2 registers then read the interrupt number * 4 then add it to interrupt_vector_table base address and load the address from that address then call the address when end the subroutine it restore R0 R1 R2 the return with POPINT for this you have to use enable_POPINT and enable_vectored_int definitions. The code that execute it is places at 0x20 in modified Crt0.s.

If enable_POPINT, enable_priority_int and enable_vectored_int is undefined it will work as zpu 1 vector and 1 priority level and the core is compatible with zylin cpu.

If enable_POPINT defined 1 POPINT instruction is enabled and can use modified crt0.S and you can define enable_vectored_int as 1 and can use modified crt0.s and syscall.c it has 32 int vector and 1 priority level.

If define enable_POPINT, enable_priority_int and enable_vectored_int as 1 too, it has 32 vector and 2 priority level it support chained interrupt too.

Interrupt controller registers:

interrupt controller require enable_itc definition.

R/W: read/write

R: read

W: write

address	C name	bits	Hdl name
0x080a0020	Global_Interrupt_mask	[0] global interrupt mask (R/W) 0 enabled (default) 1 disabled if enable_priority_int defined [0] global high priority interrupt mask and interrupt mask (R/W) 0 enabled (default) 1 disabled [1] global low priority interrupt mask (R/W) 0 enabled (default) 1 disabled	sysgiereg
0x080a001c	Interrupt_priority_mask	If enable_priority_int defined [0-31] priority level (R/W) 0 high priority (default) 1 low priority	intpriobegin
0x080a0000	Interrupt_number_reg	If enable_vectored_int defined [6:2] the number of interrupt source 0-31 it have to read to service interrupt if not read cpu never leave interrupt if enable_vectored_int defined only modified ctr0.S use it to jump target vector	intnumberreg

64 bit timer:
64 bit timer require enable_64b_timer definition.

address	C name	bits	Hdl name
0x080A0014	Counter(1)	Bit [0] Reset counter (W) 0 N/A 1 Reset counter Bit [1] Sample counter (W) 0 N/A 1 Sample counter Bit [31:0] Counter bit (R) 31:0	counterl
0x080A0018	Counter(2)	Bit [31:0] Counter bit 63:32 (R)	counterh

32 bit down counter
32 bit down counter require enable_sys_timer definition.

address	C name	bits	Hdl name
0x080A002C	Timer_Interrupt_enable	Bit [0] Timer interrupt enable(R/W) 0 Interrupt disable 1 Interrupt enable	systimerinte
0x080A0030	Timer_interrupt	Bit [0] Timer interrupt pending (R) 0 No interrupt pending 1 Interrupt pending Bit [1] Reset Timer counter (W) 0 N/A 1 Timer counter reset Bit [0] Clear Timer interrupt (W) 0 N/A 1 Interrupt cleared	systimerintf
0x080A0034	Timer_Period	Bit [31:0] Interrupt period (R/W) Number of clock cycles between timer interrupts	systimermax
0x080A0038	Timer_Counter	Bit[31:0] timer curent value(R)	systimervalue

GPIO:
GPIO require enable_gpio definition.

address	C name	bits	Hdl name
0x080A0004	GPIO data	Bit [31:0] input data 31:0 (R/W) Bit [31:0] output data 31:0 (R/W)	gpiodata
0x080A0008	GPIO direction	Bit [31:0] data direction 31:0 (R/W) 0 output 1 input (default)	gpiodir

UART:
UART require enable_uart definition.

address	C name	bits	Hdl name
0x080A000C	UART_TX	Bit [8] TX buffer ready (R) 0 TX buffer not ready (full) 1 TX buffer ready Bit [7:0] TX byte (W)	uarttx
0x080A000C	UART_RX	Bit [8] RX buffer data valid (R) 0 RX buffer not valid 1 RX buffer valid Bit [7:0] RX byte (R)	uartrx
0x080A0024	UART_INTERRUPT_ ENABLE	Bit [0] UART RX interrupt enable (R/W) 0 Interrupt disable 1 Interrupt enable Bit [1] UART TX interrupt enable (R/W) 0 Interrupt disable 1 Interrupt enable	
0x080A0028	UART_interrupt	Bit [0] UART RX interrupt pending (R) 0 No interrupt pending 1 Interrupt pending Bit [1] UART TX interrupt pending when TX FIFO is empty (R) 0 No interrupt pending 1 Interrupt pending Bit [0] Clear UART RX interrupt (W) 0 N/A 1 Interrupt cleared Bit [1] Clear UART TX interrupt (W) 0 N/A 1 Interrupt cleared	