GRAND CHALLENGE - 2025

SHAKTI SoC DEVICE REGISTER MANUAL

DEVELOPED BY :
SHAKTI DEVELOPMENT TEAM @ IITM
SHAKTI.ORG.IN

0.1 Proprietary Notice

The project was funded by Ministry of Electronics and Information Technology (MeitY), Government of India

0.2 Release Information

Version	Date	Updates
1.0	24/04/2025	Initial version

Table of Contents

YAMUNA SoC

1. SoC Details:

Yamuna is the 32 bit RISC-V SoC with 6 stage SHAKTI C class, compliance with RV32IMAC extension. Yamuna is supported on the Artix 7 100T board.

The peripherals used in this SoC are

→ 6 PWM
→ 2 SPI
→ 2 UART
→ 2 I2C
→ 4 TIMER
→ 32 GPIO
→ PLIC
→ CLINT
→ XADC and Ethernet IP from Xilinx.

In Yamuna SoC, primary signals are multiplexed to reduce the pin count and also to enable a variety of pin assignment in Arty 100t such as pin assignment compatible to Arduino UNO.

2. Peripheral Register Details:

2.1 PULSE WIDTH MODULATORS

Pulse Width Modulators (PWM) are used to produce pulses with variable duty cycle. The duty cycle and the period of the pulse can be varied through PWM DUTY and PWM PERIOD registers respectively. The duty cycle and the period can be changed using the PWMUPDATEEN bit in the PWMCONTROL register. Interrupts are provided at rising, falling and half-period instances of the signal.

The required dead band delay is provided through PWM DEADBAND register. The number of dead band cycles are given as the number of prescalled clock cycles (number of down converted clock cycles).

PWM Registers

Register name	Offset address	Accessible Size	Description
PWM CONTROL	‘h00	16 bits	PWM Clock control Register for cluster 0(16 bits)(Read and Write)
PWM CONTROL	‘h04	16 bits	PWM Control Register (16 bits)(Read and Write)
PWM PERIOD	‘h08	32 bits	PWM Period Register (32 bits)(Read and Write)
PWM DUTY	‘h0C	32 bits	PWM Duty Cycle Register (32 bits)(Read and Write)
PWM DEADBAND	‘h10	16 bits	PWM Dead Band Frequency Register (16 bits)(Read and Write)

PWMCLKCONTROL Register

15 ... 1	0
PWMPRESCALE	Reserved

Bits 15:1 - PWMPRESCALE : Prescale value to get required clock speed. Required clock frequency = Internal clock frequency / (Prescale + 1).

PWM Frequency = system clock / (2 * prescaler reg * period reg)
= 50000000/(2 * 61440 * 240) = 1.688Hz.

PWM Period = 1 / PWM frequency = 1 / 1.688Hz = 0.589824 seconds.

PWM On time = (PWM Duty reg * PWM period) / (PWM Period reg)
= 128 * 0.589824 / 240 = 0.3145728 seconds.

Bit 0 - Reserved

PWMCONTROL Register

15 ... 13	12	11	10	9	8	7	6
RESERVED	PWM UPDATEEN	PWM RISE STS	PWM FALL STS	PWM HP STS	PWM RISE INTREN	PWM FALL INTREN	PWM HP INTREN

5	4	3	2	1	0

PWMCOMPOUTEN	PWMCOUNTRESET	PWMOUTPOL	PWMOUTEN	PWMSTART	PWMEN

Bits 15:13 - Reserved
Bit 12 - PWMUPDATEN : This bit when enabled the duty and period of PWM signal is updated at rising edge or falling edge.
Bit 11 - PWMRISESTS : This is a read only bit which will be set during rising edge of PWM signal
Bit 10 - PWMFALLSTS : This is a read only bit which will be set during falling edge of PWM signal
Bit 9 - PWMHPSTS : This is a read only bit which will be set during half period of PWM signal
Bit 8 - PWMRISEINTREN : This bit when enabled, raises interrupt during rising edge of PWM signal
Bit 7 - PWMFALLINTREN : This bit when enabled, raises interrupt during falling edge of PWM signal
Bit 6 - PWMHPINTREN : This bit when enabled, raises interrupt during half period of PWM signal
Bit 5 - PWMCOMPOUTEN : PWM complementary output selection when set(1), complementary output with dead-band time will be available at the PWM COMP OUT pin.
Bit 4 - PWMCOUNTRESET : PWM Counter Reset. This when enabled(1), the PWM counter is reset to zero.
Bit 3 - PWMOUTPOL : PWM Output Polarity. This when set(1), PWM output will be high(1) when the counter value is less than or equal to the duty cycle value and when reset(0), PWM output will be low(0) when the counter value is less than or equal to the duty cycle.
Bit 2 - PWMOUTEN : PWM Output Enable. This when enabled(1), the PWM output can be obtained from PWM OUT pin.
Bit 1 - PWMSTART : PWM Start Command. This when enabled(1), the PWM generation will be started.
Bit 0 - PWMENABLE : PWM Enable Command. This when enabled(1), the PWM operation will be enabled

The controller has sixteen PWM which gives utmost sixteen independent PWM outputs or at most eight complementary outputs. Each PWM produces both the required and its complementary outputs and the PWM outputs are multiplexed at the cluster level through PWMOUTPUTCTRL register. PWMs are controlled pair wise to get the complementary output from one PWM or independent outputs from both the PWMs.

The clock speed can be varied by providing appropriate value to PWMCLKCONTROL register. There are two clusters of PWM instances in the controller and each instance is channelised to have eight PWMs making a total of sixteen PWMs in the controller. Clock Control register PWMCLKCONTROL is shared by each of the all PWMs in the instance.

PWMPERIOD Register

PWM PERIOD_REGISTER further divides the system clock by the (Prescaler value + 1).

31	0
PWM Period	

Bits 31:1 - PWM PERIOD : Period value to get required PWM period..
PWM Period = 1 / PWM frequency

PWMDUTY Register

PWM DUTY_REGISTER decides the ON time of PWM clock based on the duty value..

31	0
PWM Duty	

Bits 31:1 - PWM DUTY: ON Time of the PWM Period.

ON (HIGH) of PWM DUTY = (PWM Duty reg * PWM period) / (PWM Period reg)

Sequence of execution :

1. Write into the Period, Duty cycle, Clock control and Deadband registers
2. Write into PWM control register to start the PWM operation

ON (HIGH) of PWM DUTY = (PWM Duty reg * PWM period) / (PWM Period reg)

Note:
PWM Frequency = system clock / (2 * prescaler reg * period reg)
= 50000000/(2 * 61440 * 240)
= 1.688Hz
PWM Period = 1 / PWM frequency
= 1 / 1.688Hz = **0.589824 seconds**
PWM On time = (PWM Duty reg * PWM period) / (PWM Period reg)
= 128 * 0.589824 / 240 = **0.3145728 seconds**

2.2 SPI

SPI is a synchronous serial I/O port that allows a serial bit stream of programmed length to be shifted into and out of the device at programmable bit transfer rate. The length of the bit stream to be transferred or received can be programmed through the SPITOTALBITS TX and SPITOTALBITS RX bits in SPICOMCONTROL register and the required bit transfer rate is achieved by giving appropriate value to SPIPRESCALE bits in SPICLKCONTROL register.
32-level receive and transmit FIFO is provided to reduce servicing overhead. The module also provides simultaneous receive and transmit operation. Delayed transmit control is achieved by providing required delay through SPICS2TXDELAY bits in SPICLKCONTROL register and also the hold time for the slave can be through SPITX2CSDELAY bits in SPICLKCONTROL register.
Each SPI module has four external pins - Master in/Slave out(MISO) pin, Master out/Slave in(MOSI) pin, Synchronous clock(SCLK) and Chip Select(CS) pin. All the pins of the Standard SPI module have input qualification control and the input qualification cycle is configured through SPIQUAL register.

Following are the registers used in Standard SPI peripheral:

SPI Registers

Register name	Offset address	Accessible Size	Description
SPICOMMCONTROL	‘h00	32 bits	Standard SPI Communication Control Register (32 bits,Read and Write)
SPICLKCONTROL	‘h04	32 bits	Standard SPI Clock Control Register (32 bits, Read and Write)
SPIDATATX	‘h08	8/16/32 bits	Standard SPI Transmitter Data Register (32 bits, Write Only)
SPIDATARX	‘h0C	8/16/32 bits	Standard SPI Receiver Data Register (32 bits, Read Only)
SPIINTREN	‘h10	16 bits	Standard SPI Interrupt Enable Register (16 bits, Read and Write)
SPIFIFOSTATUS	‘h14	8 bits	Standard SPI FIFO Status Register (8 bits, Read Only)
SPICOMMSTATUS	‘h18	8 bits	Standard SPI Communication Status

			Register (8 bits, Read Only)
SPIQUAL	‘h1C	8 bits	Standard SPI Input Qualification Control Register (4 bits, Read and Write)

SPICOMMCONTROL Register

Address offset: 0x00

31	26	25	24
Reserved		SPIOUTENMOSI	SPIOUTENMISO

23	22	21	14	13	6
SPIOUTENNCS	SPIOUTENSCLK	SPITOTALBITSRX		SPITOTALBITSTX	

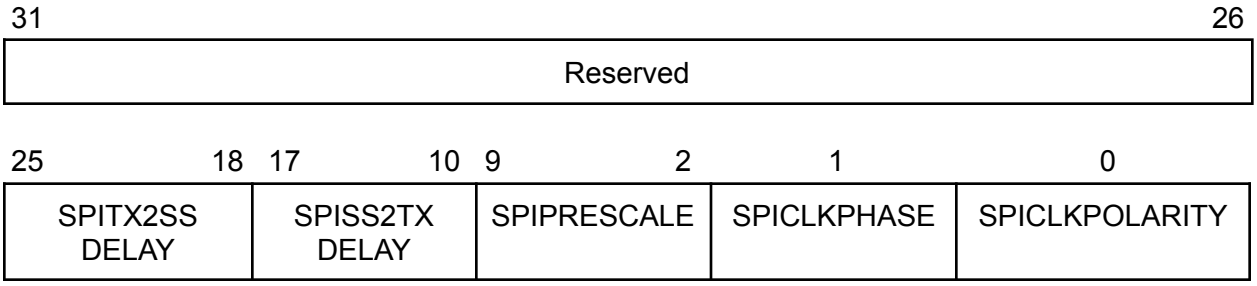
5	4	3	2	1	0
SPICOMMMODE	Reserved	SPILSBFIRST	SPIENABLE	SPIMASTER	

Bits 31:26 - Reserved
Bit 25 - SPIOUTENMOSI : Output enable for MOSI pin. When set(1), MOSI will be an output and when reset(0), it will be an input.
Bit 24 - SPIOUTENMISO : Output enable for MISO pin. When set(1), MISO will be an output and when reset(0), MISO will be an input.
Bit 23 - SPIOUTENNCS : Output enable for NCS pin. When set(1), NCS will be an output and when reset(0), NCS will be an input..
Bit 22 - SPIOUTENSCLK : Output enable for SCLK pin. When set(1), SCLK will be an output and when reset(0), SCLK will be an input.
Bits 21:14 - SPITOTALBITSRX : Total number of bits to be received. The values vary from 0 to 255
Bits 13:6 - SPITOTALBITSTX : Total number of bits to be transmitted. The values vary from 0 to 255
Bits 5:4 - SPICOMMMODE : SPI Communication Mode
00 - Simplex Transmit Only
01 - Simplex Receive Only
10 - Half Duplex - Transmit and Receive but not at the same time (First Transmit and then receive option is only available now)
11 - Full Duplex - Transmit and Receive simultaneously
Bit 3 - Reserved

Bit 2 - SPILSBFIRST : When set (1), the data is transmitted or received with LSB first whereas when reset(0), the data is transmitted or received with MSB first
Bit 1 - SPIENABLE : When set (1), the SPI communication is enabled else disabled
Bit 0 - SPIMASTER : When set (1), SPI module acts as Master else acts as slave

SPICLKCONTROL Register

Address offset: 0x04



Bits 31:26 - Reserved
Bits 25:18 - SPITX2SSDELAY : It is Transmit end to Slave select disable delay. It is the hold time for the slave device. It takes values from 0 to 255 indicating 0 to 255 SPI clock cycles
Bits 17:10 - SPISS2TXDELAY : It is Slave select Active to Transmit begin delay. It is the setup time for the slave device. It takes values from 0 to 255 indicating 0 to 255 SPI clock cycles
Bits 9:2 - SPIPRESCALE : Internal clock frequency(f) is divided by the prescaler value to attain the required bit rate. It is an 8-bit value and hence can provide a maximum of 255 different data rates.

Required Bit Rate = Internal clock frequency/(Prescaler value+1)

Bit 1 - SPICLKPHASE : It gives the clock offset at which the data to be transmitted or received. When set(1), the SPI transmits or receives data in the second clock transition whereas when reset(0), the SPI transmits or receives data in the first clock transition
Bit 0 - SPICLKPOLARITY : It holds the value of SCLK in Idle state.

SPIDATATX Register

The value to be transmitted is written into the TX register.

Address offset: 0x08

31	0
Data to be transmitted	

SPIDATARX Register

The received value can be read from the SSPI Receive DATA register.

Address offset: 0x0C

31	0
sspi Received Data	

SPIINTREN Register

Address offset: 0x10

15	9	8	7	6
Reserved		SPIRXOVERRUN INTREN	SPIRXFULL INTREN	SPIRXHALF INTREN

5	4	3	2	1	0
SPIRXQUAD INTREN	SPIRXEMPTY INTREN	SPITXFULL INTREN	SPITXHALF INTREN	SPITXQUAD INTREN	SPITXEMPTY INTREN

Bits 15:9 - Reserved
Bit 8 - SPIRXOVERRUN INTREN : This bit should be set to get interrupt at overrun
Bit 7 - SPIRXFULL INTREN : This bit should be set to get interrupt when RX FIFO gets 32 entries

Bit 6 - SPIRXHALF INTREN : This bit should be set to get interrupt when RX FIFO gets 16 entries
Bit 5 - SPIRXQUAD INTREN : This bit should be set to get interrupt when RX FIFO gets 8 entries
Bit 4 - SPIRXEMPTY INTREN : This bit should be set to get interrupt when RX FIFO is empty
Bit 3 - SPITXFULL INTREN : This bit should be set to get interrupt when TX FIFO gets 32 entries
Bit 2 - SPITXHALF INTREN : This bit should be set to get interrupt when TX FIFO gets 16 entries
Bit 1 - SPITXQUAD INTREN : This bit should be set to get interrupt when TX FIFO gets 8 entries
Bit 0 - SPITXEMPTY INTREN : This bit should be set to get interrupt when TX FIFO is empty

SPICOMMSTATUS Register

Address offset: 0x18

7	6 5	4 3	2	1	0
SPIOVR	SPIRXFIFO	SPITXFIFO	SPIRXNE	SPITXE	SPIBUSY

Bit 7 - SPIOVR : SPI Overrun error. This flag is set when the receiver FIFO is full and there is a write to Receiver Data Register
Bits 6:5 - SPIRXFIFO : SPI Receiver FIFO Level
00 - FIFO empty to Quad-1
01 - ¼ FIFO to Half-1
10 - ½ FIFO to Full-1
11 - FIFO full
Bits 4:3 - SPITXFIFO : SPI Transmission FIFO Level
00 - FIFO empty to Quad-1
01 - ¼ FIFO to Half-1
10 - ½ FIFO to Full-1
11 - FIFO full
Bit 2 - SPIRXNE : SPI Receiver Not Enable. When this is reset(0), the data is being received
Bit 1 - SPITXE : SPI Transmitter Enable. When this is set(1), the data is being transmitted
Bit 0 - SPIBUSY : SPI Busy. When this is set(1), the SPI communication is being carried out and when this is reset(0), the SPI is in Idle state.

SPIFIFOSTATUS Register

Address offset: 0x14

7	6	5	4	3	2	1	0
SPIRX FULL	SPIRX HALF	SPIRX QUAD	SPIRX EMPTY	SPITX FULL	SPITX HALF	SPITX QUAD	SPITX EMPTY

Bit 7 - SPIRXFULL : This bit will be set when RX FIFO gets 32 entries
Bit 6 - SPIRXHALF : This bit will be set when RX FIFO gets 16 entries
Bit 5 - SPIRXQUAD : This bit will be set when RX FIFO gets 8 entries
Bit 4 - SPIRXEMPTY : This bit will be set when RX FIFO is empty
Bit 3 - SPITXFULL : This bit will be set when TX FIFO gets 32 entries
Bit 2 - SPITXHALF : This bit will be set when TX FIFO gets 16 entries
Bit 1 - SPITXQUAD : This bit will be set when TX FIFO gets 8 entries
Bit 0 - SPITXEMPTY : This bit will be set when TX FIFO is empty

Sequence of execution:

For transmission,

1. write into tx data register (if the data to be transmitted is more than 32 bits, write multiple times into the tx data register, every write will push its contents to the TX FIFO)
2. Write into clock control register, interrupt enable register and input qualification register
3. Now enable the SPI transaction through communication control register

For reception,

1. Write into clock register, interrupt enable register and input qualification register
2. And now enable the SPI transaction through communication control register
3. The received data will be available in RX data register(If data to be received is more than 32 bits, perform multiple read RX data register, this will pop the contents from RX FIFO)

2.3 I2C

I2C is a serial protocol for a two-wire interface to connect low-speed devices like EEPROMs, A/D and D/A converters, I/O interfaces and other similar peripherals in embedded systems. Multiple slave devices can be connected to a single master with I2C. I2C only uses two wires to transmit data between devices

I2C Registers

Register name	Offset address	Accessible Size	Description
Prescale	‘h00	8 bits	I2C Prescaler Register (Read and Write)
Control	‘h08	8 bits	I2C Control Register (Read and Write)
Data	‘h10	8 bits	I2C Data Shift Register (Read and Write)
Status	‘h18	8 bits	I2C Status Register (Read)
SCL	‘h38	8 bits	I2C Clock Register (Read and Write)

I2C PRESCALE Register

I2C Prescale Register divides the System clock by (Prescale value + 1). This clock is used as clock input for I2C Serial Clock.
I2C Prescaler clock = System Clock / (Prescaler Value + 1)

Address offset: 0x0

7	0
PRESCALE VALUE	

I2C SCL Register

I2C SCL Register divides the I2C Prescaler clock by (SCL value + 1). This clock is used as
I2C SCL clock = I2C Prescaler Clock / (SCL COUNT + 1).

Address offset: 0x38

7	0
SCL COUNT	

I2C CLK FREQUENCY=core frequency/2*(PRESCALE)*(SCL COUNT)

Status Register

I2C Status Register has status of I2C data transfer. The bit level details are given below.

Address offset: 0x18

7	6	5	4	3	2	1	0
I2C_INI	Reserved	I2C_STS	I2C_BER	I2C_AD0/i I2C_LRB	I2C_AAS	I2C_LAB	I2C_BB

Signal Name	Bit Position	Comments
I2C_INI	7	High when I2C communication is in progress. Becomes low once I2C communication is complete.
Reserved	6	Set to 0
I2C_STS	5	When in slave receiver mode, this flag is asserted when an externally generated STOP condition is detected (used only in slave receiver mode).
I2C_BER	4	Bus error; a misplaced START or STOP condition has been detected

I2C_AD0/I2C_LRB	3	AD0(Address 0) - General Call bit used for Broadcast LRB - Last Received Bit through I2C Bus This status bit serves a dual function, and is valid only while PIN = 0: 1. LRB holds the value of the last received bit over the I2C-bus while AAS = 0 (not addressed as slave). Normally this will be the value of the slave acknowledgement; thus checking for slave acknowledgement is done via testing of the LRB. 2. AD0; when AAS = 1 (‘Addressed As Slave’ condition), the I2C-bus controller has been addressed as a slave. Under this condition, this bit becomes the ‘AD0’ bit and will be set to logic 1 if the slave address received was the ‘general call’ (00H) address, or logic 0 if it was the I2C-bus controller’s own slave address.
I2C_AAS	2	Addressed As Slave’ bit. Valid only when PIN = 0. When acting as slave receiver, this flag is set when an incoming address over the I2C-bus matches the value in own address register
I2C_LAB	1	Lost Arbitration Bit. This bit is set when, in multi-master operation, arbitration is lost to another master on the I2C-bus.
I2C_BB	0	Bus Busy bit. This is a read-only flag indicating when the I2C-bus is in use. A zero indicates that the bus is busy, and access is not possible

Control Register

I2C control register configures and controls the I2C data transfer.

Address offset: 0x08

7	6	5	4	3	2	1	0
I2C_PIN	I2C_ES0	Reserved	Reserved	I2C_ENI	I2C_STA	I2C_STO	I2C_ACK

Signal Name	Bit Position	Comments
I2C_PIN	7	Pending Interrupt Not, Used as a software reset. If set to 1, all

I2C_ESO	6	Enable Serial Output. 0 - Registers can be initialized. 1 - I2C Serial Transmission
Reserved	5	Reserved.
Reserved	4	Reserved.
I2C_ENI	3	Enables the external interrupt output, which is generated when the PIN is active (Low)
I2C_STA	2	Transmits Start condition + Slave address..
I2C_STO	1	Transmits the stop condition.
I2C_ACK	0	Acknowledgement bit. 1: I2C automatically sends an acknowledgement after a read/write transaction. 0: I2C Master sends Negative Acknowledge to stop the I2C transfer.

2.4 UART

UART module provides a two-wire asynchronous serial non-return-to-zero (NRZ) communication with RS-232 (RS-422/485) interface. Each UART module has transmit and receive buffers that can hold upto 16 entries. Data transfer rate can be modified by providing appropriate value to UARTBAUD register.

Required data rate = Internal clock frequency/(UARTBAUD * 16)

The data transmission and reception starts with a start bit followed by data words whose length can be varied through UARTTXBITS or UARTRXBITS in UARTCONTROL register which is followed by optional parity bit and stop bits. The parity can be Odd or Even and the parity can also be disabled through UARTPARITY bits in UARTCONTROL register. The number of stop bits can also be varied through UARTSTOPBITS in UARTCONTROL register. Delayed Transmit control is done by providing the required delay in UARTTXDELAY register.

The value to be transmitted is written into UARTTXDATA register and the value received is read from UARTRXDATA register. UARTSTATUS reg holds the UART status & UARTINTERRUPTEN reg holds the interrupt enable for the various interrupt events. UARTRXTH register holds the receiver FIFO threshold value, when the RX FIFO level increases beyond the threshold, corresponding status bit will be set and when interrupt is enabled, interrupt will be raised.UARTIQC holds the number of input qualification cycles for the receiver pin. UART can be operated in full-duplex or half-duplex mode. The inputs to UART receiver and transmitter are double-buffered and each has its own separate enable and interrupt bits.

UART has three external pins - Transmitter data pin, Receiver data pin and Transmitter output enable pin

UART Registers

Register name	Offset address	Accessible Size	Description
BAUD Register	‘h00	16 bits	UART Baud Rate Register (Read and write)
TX_REG Register	‘h04	32 bits	UART Transmitter Data Register (Write only)
RCV_REG Register	‘h08	32 bits	UART Receiver Data Register (Read Only)
Status Register	‘h0C	16 bits	UART Status Register (Read only)
Delay Register	‘h10	16 bits	UART Transmitter Delay Register(Read and write)

Control Register	‘h14	16 bits	UART Control Register (Read and write)
IEN Register	‘h18	8 bits	UART Interrupt Enable Register (Read and write)
IQCYCLES Register	‘h1C	8 bits	UART Input Qualification Control Register (Read and write)
RX_THRESHOLD Register	‘h20	8 bits	UART Receiver Threshold Register (Read and write)

UART Baud Register

The UART BAUD Register configures the Baud rate based on the Baud value set.

Address offset: 0x00

15	0
Baud	

Signal Name	Bit Position	Read/Write	Comments
Baud	15:0	Read/Write	Baud Rate is calculated by Baud_value=(Clock Frequency)/(16*Baudrate)

UART Status Register

The status register holds the status of various Transmit and receive status and errors.

Address offset: 0x0C

8	7	6	5	4	3	2	1	0
RXFIFOTHRE	BREAK ERROR	FRAME ERROR	OVERRUN ERROR	PARITY ERROR	RX FULL	RX NOT EMPTY	TX FULL	TX EMPTY

Bits 15:9 - Reserved
Bit 8 - RXFIFOTHRE : Will be set RX FIFO reached the value set in UARTRXTH register
Bits 7:4 - Corresponding bits will be set when the receiver error occurs
Bits 3:2 - Receive FIFO status bits
Bits 1:0 - Transmit FIFO status bits

Signal Name	Bit Position	Comments
BREAK_ERROR	7	Break Error (Sets when the data and stop are both zero)
FRAME_ERROR	6	Frame Error (Sets when the stopis zero)
OVERRUN	5	Overrun Error (A data overrun error occurred in the receive shift register. This happens when additional data arrives while the FIFO is full.)
PARITY-ERROR	4	Parity Error (Sets when The receive character does not have correct parity information and is suspect.)
STS_RX_FULL	3	Receiver Full (Sets when the Receive Buffer is Full)
STS_RX_NOT_FULL	2	Receiver Not Empty (Sets when there is some data in the Receive Buffer).
STS_TX_FULL	1	Transmitter Full (Sets when the transmit Buffer is full)
STS_TX_EMPTY	0	Transmitter Empty(Sets when the Transmit Buffer is empty).

UART Control Register

The UART and receive operation and protocol are configured using the UART control register.

Address offset: 0x14

15 11	10 5	4 3	2 1	0
Reserved	UARTCHARSIZE	UARTPARITY	UARTSTOPBITS	Reserved

Bits 15:11 - Reserved
Bits 10:5 - UARTCHARSIZE : Total number of data bits to be received. This value varied from 0 to 32
Bits 4:3 - UARTPARITY : This provides various options for parity. 00 - No parity bit, 01 - odd parity and 10 - even parity
Bits 2:1 - UARTSTOPBITS : This provides various options for number of stop bits. 00 - 1 stop bit, 01 - 1.5 stop bit and 10 - 2 stop bits
Bit 0 - Reserved

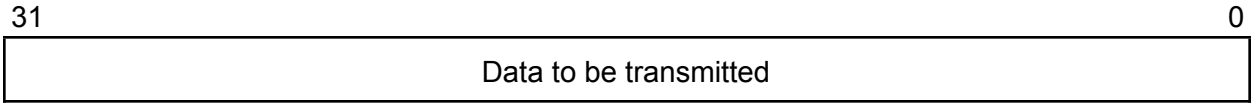
Signal Name	Position	Comments
Reserved	15:11	Reserved

UARTCHARSIZE	10:5	Character size of data. Maximum length is 32 bits.
UARTPARITY	4:3	Insert Parity bits 00 - None 01 - Odd 10- Even 11 - Unused or Undefined
UARTSTOPBITS	2:1	Stop bits 00 - 1 Stop bits 01 - 1.5 Stop bits 10 - 2 Stop bits
Reserved	0	Reserved

UART TX_REG Register

The value to be transmitted is written into the TX_REG register.

Address offset: 0x04



UART RCV_REG Register

The received value can be read from the UART Receive DATA register.

Address offset: 0x08

UART IEN Register

Enables interrupts based on bit value set in the Interrupt Enable Register.

Address offset:0x18

8	7	6	5	4	3	2	1	0
RXFIFOTH_EN	BREAK ERROR_EN	FRAME ERROR_EN	OVERRUN ERROR_EN	PARITY ERROR_EN	RX FULL_EN	RX NOT EMPTY_EN	TX FULL_EN	TX EMPTY_EN

Bits 15:9 - Reserved
Bits 8:0 - When corresponding enable bits are set, interrupt will be raised at those events

Signal Name	Bit Position	Comments
Reserved	15:9	Reserved
ENABLE_RX_THRESHOLD	8	RX Threshold Interrupt Enable
ENABLE_BREAK_ERROR_EN	7	Break Error Interrupt Enable
ENABLE_FRAME_ERROR_EN	6	Frame Error Interrupt Enable
ENABLE_OVERRUN_ERROR_EN	5	Overrun Interrupt Enable
ENABLE_PARITY_ERROR_EN	4	Parity Error Interrupt Enable
ENABLE_RX_FULL	3	Receiver Full Interrupt Enable
ENABLE_RX_NOT_EMPTY	2	Receiver Not Empty Interrupt Enable
ENABLE_TX_FULL	1	Transmitter Full Interrupt Enable
ENABLE_TX_EMPTY	0	Transmission Empty Interrupt Enable

UART DELAY Register

Delayed Transmit control is done by providing the required delay in the UART DELAY register.

Address offset: 0x10

15	0
Transmit Delay Count	

UART IQCYCLES Register

UART IQCYCLES Register holds the number of input qualification cycles for the receiver pin.

7	0
Input Qualification Cycles Count	

UART RX_THRESHOLD Register

UART RX_THRESHOLD register holds the receiver FIFO threshold value, when the RX FIFO level increases beyond the threshold, corresponding status bit will be set and when interrupt is enabled, interrupt will be raised.

Address offset: 0x20

7	0
FIFO Receive Threshold Value	

Sequence of execution:

For transmission,

1. Write into TX data register
2. Write into baud, txdelay, interrupt enable, input qualification and receiver threshold registers
3. At the end write into control register to start the uart transmission

For reception,

1. Write into baud, txdelay, interrupt enable, input qualification and receiver threshold registers
2. Write into control register to start the uart reception

The received data will be available in RX data register

2.5 GENERAL PURPOSE INPUT / OUTPUT (GPIO)

The General Purpose Input/output peripheral can be used to generate custom waveforms, enable signals, interrupts, etc. The GPIO has a GPIODIR register which configures the GPIO pin as an input or output, GPIODATA register which holds the input data to GPIO or output data from GPIO and GPIOQUAL register holds the number of input qualification cycles needed to filter the unwanted noise glitches.

GPIOSET register is used to set(to 1) certain bits in GPIODATA register. The required bits to be set are set(made ‘1’) in GPIOSET register. This is done by OR operation between GPIODATA register and GPIOSET register.

GPIOCLEAR register is used to clear(to 0) certain bits in GPIODATA register. The required bits to be cleared are set (made ‘1’) in GPIOCLEAR register. Clearing is done by AND operation between GPIODATA register and NOT of GPIOCLEAR register.

GPIOTOGGLE register is used to toggle certain bits in GPIODATA register. The required bits to be toggled are set(made ‘1’) in GPIOTOGGLE register. Toggling is done by XOR operation between GPIODATA register and GPIOTOGGLE register.

GPIOINTRCONFIG register is used to set the polarity of the interrupt which are raised using GPIOs. By default, the interrupts are active high, if active low interrupts are required that particular GPIO bit in GPIOINTCONFIG register should be set.

The GPIO pins 0 - 31 can accept External events as interrupts. To use a GPIO pin (0 - 15) as an interrupt, that particular GPIO pin(s) should be configured as input. The GPIO data register is 1 byte, 2 byte and 4 byte accessible.

GPIO Registers

Register name	Offset address	Accessible size	Description
GPIODIR	‘h00	32 bits	GPIO Direction Control Register (32 bits)(Read and Write)
GPIODATA	`h08	32 bits	GPIO Data Register (32 bits)(Read and Write)
GPIOSET	‘h10	32 bits	GPIO Set Register (32 bits)(Write only)
GPIOCLEAR	‘h18	32 bits	GPIO Clear Register (32 bits)(Write only)
GPIOTOGGLE	‘h20	32 bits	GPIO Toggle Register (32 bits)(Write

			only)
GPIOQUAL	‘h28	8 bits	GPIO Input Qualification Control Register (4 bits)(Read and Write)
GPIOINTRCONFIG	‘h30	32 bits	GPIO Interrupts configuration Register(32 bits)(Read and Write)

Sequence of execution:

1. Write into the Direction register to configure GPIO pin as an input or output
2. Write appropriate values to data register
3. If any of GPIO pin is an input, set an appropriate input qualification cycles in Qual register
4. Configure Set, Clear, Toggle or Interrupt config registers appropriately if needed

2.6 GENERAL PURPOSE TIMERS

General Purpose timers (GP Timer) have four modes - a simple PWM mode, upcounter, down counter and up-down counter, the required mode is selected through GPTMODE bits in GPTCONTROL register.

In Counter mode, either up, down or up-down counting is done and the interrupts are raised during overflow and underflow according to the interrupt enable bits in GPTCONTROL register. The Counter reference value is given through the GPTCOUNTREF register and the present value of the counter can be obtained through GPTCOUNTER register. The input to the counter can be an internal clock or an external clock with prescaler. GPTREPEATCOUNT register hold the number of times the counter has repeated the count if it is set to continuous count in GPTCONTROL register.

GPTCAPTURE register holds the timestamp of input. The register is updated with GPTCOUNTER value when an input is received or whenever there is a change in input.

In PWM mode, according to the duty cycle and the period, PWM-waveform is generated and the interrupts are provided at compare (falling edge) and period (rising edge) according to the interrupt enable bits in GPTCONTROL register. Duty cycle of the pulse is set in GPTCOMPARE register and the period is set in GPTCOUNTREF register. The input to PWM mode is internal or external clock with prescaler. PWM mode here will not be used to produce complementary outputs hence dead-band frequency features are not included.

Internal or external clock speed can be decreased by providing appropriate value to the GPTPRESCALE bits in GPTCLKCONTROL register.

GPTCOUNTREF and GPTCOMPARE registers are double-buffered. GPTINPUTQUAL register holds the number of input qualification cycles for the input capture.

GPT Registers:

Register name	Offset address	Accessible Size	Description
GPTCONTROL	‘h00	16 bits	General Purpose Timer Control Register (16 bits)(Read and Write)
GPTCLKCONTROL	‘h04	32 bits	General Purpose Timer Clock Control Register (32 bits)(Read and Write)
GPTCOUNTER	‘h08	32 bits	General Purpose Timer Counter Register

			(32 bits)(Read Only)
GPTREPEATCOUNT	‘h0C	32 bits	General Purpose Timer Repeat Count Register(32 bits)(Read Only)
GPTCOMPARE	‘h10	32 bits	General Purpose Timer Compare Register (32 bits)(Read and Write)
GPTCOUNTREF	‘h14	32 bits	General Purpose Timer Counter Reference Register (32 bits)(Read and Write)
GPTCAPTURE	‘h18	32 bits	General Purpose Timer Capture Register(32 bits)(Read Only)
GPTINPUTQUAL	‘h1C	8 bits	General Purpose Timer Input Qualification Register(4 bits)(Read and Write)

GPTCONTROL Register

This register controls the operation and status of the timer.

Address offset: 0x00

15	14	13
GPTINPUTCAPTURE	GPTCNTR UNDERFLOW	GPTCNTR OVERFLOW

12	11	10	9	8	7
GPT PWM RISE	GPT PWM FALL	GPTCNTR UNDERFLOW INTREN	GPTCNTR OVERFLOW INTREN	GPT PWMRISE INTREN	GPT PWMFALL INTREN

6	5	4	3 2	1	0
GPTCONT COUNT	GPTCOUNT RESET	GPTOUTEN	GPTMODE	GPTRESET	GPTENABLE

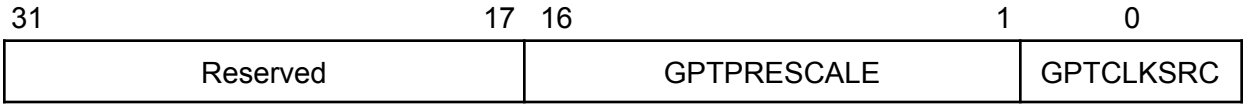
Bits 15 - GPTINPUTCATURE : Polarity of the input to be captured. When set to 1, the capture input when gone high will be captures else input when gone low will be captured
Bit 14 - GPTCNTR UNDERFLOW : read-only bit. This is set when there is an underflow from down-counter and up-down counter.
Bit 13 - GPTCNTR OVERFLOW : read-only bit. This is set when there is an overflow from up-counter and up-down counter

Bit 12 - GPTPWM RISE : read-only bit. This is set when the PWM period completes - Rising edge of the PWM mode
Bit 11 - GPTPWM FALL : read-only bit. This is set when the count value is greater than compare in PWM mode. This is falling edge of PWM signal
Bit 10 - GPTCNTR UNDERFLOW INTREN : This bit should be set if interrupt is needed during counter underflow
Bit 9 - GPTCNTR OVERFLOW INTREN : This bits should be set if interrupt is needed during counter overflow
Bit 8 - GPT PWMRISE INTREN : This bit should be set if interrupt is needed at PWM Rising edge
Bit 7 - GPT PWMFALL INTREN : This bit should be set if interrupt is needed at PWM Falling edge
Bit 6 - GPTCONT COUNT : When this bit is set, the counter will be in continuous count mode.
Bit 5 - GPTCOUNT RESET : This bit can be set to reset the counter register in the GPTimer
Bit 4 - GPTOUTEN : When this bit is set, PWM output will be made available at the GPTimer output
Bit 3:2 - GPTMODE : These bits define the mode of GPTimer. 00 - PWM; 01 - Up counter; 10 - Down counter; 11 - Up-Down counter
Bit 1 - GPTRESET : This bit is to reset the GP Timer. This when set(1), resets the clocks and registers which are used for counting or PWM operation
Bit 0 - GPTENABLE : This bit is set(1) to enable GP Timer and reset(0) to disable

GPTCLKCONTROL Register

Controls the clock source and speed of the GPT.

Address offset: 0x04

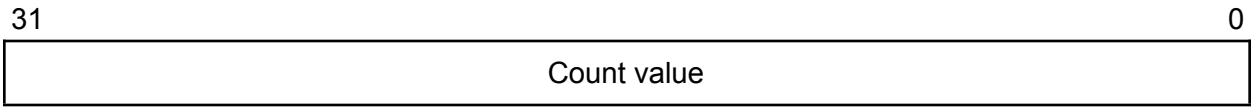


Bits 31:17 - Reserved
Bits 16:1 - GPTPRESCALE : Prescale value to get required clock speed. Required clock frequency = Internal clock frequency / (Prescale + 1).
Bit 0 - GPTCLKSRC : GP Timer Clock source. This bit when set(1), external clock is used and when reset(0), internal clock is used.

GPTCOUNTER Register

Shows the current count value of the timer.

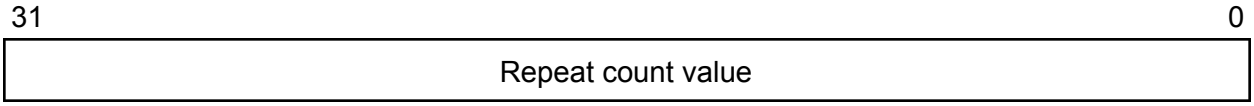
Address offset: 0x08



GPT REPEATCOUNT Register

Specifies how many times to **repeat the count cycle**.

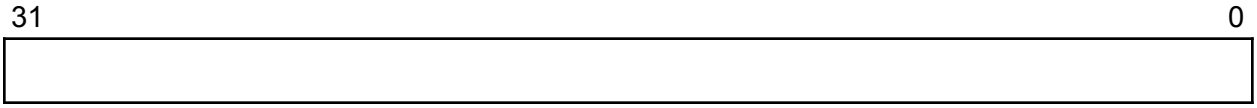
Address offset: 0x0c



GPTCOMPARE Register

Sets the **compare value**.

Address offset: 0x10



GPT COUNTREF Register

This is the **reference value** loaded into the counter when the timer is reset.

Address offset: 0x14

Count reference value

GPTCAPTURE Register

Holds the **timestamp value** of the counter when an input edge (rising/falling) is captured.

Address offset: 0x18

31	0
Gpt capture value	

Sequence of execution:

1. Write into compare, counterref, input qualification register and clock control register
2. Write into the control register to start the counter operation
3. Read corresponding registers such as counter register for up/down count values, capture register to get the timestamp of the input and repeat count register to get the number of times the counter has repeated the counting

3. Interrupt ID mapping to Device number

INT ID	Source	INT ID	Source
0	Unused	26	GPIO19
1	PWM0	27	GPIO20
2	PWM1	28	GPIO21
3	PWM2	29	GPIO22
4	PWM3	30	GPIO23
5	PWM4	31	GPIO24
6	PWM5	32	GPIO25
7	GPIO0	33	GPIO26
8	GPIO1	34	GPIO27
9	GPIO2	35	GPIO28
10	GPIO3	36	GPIO29
11	GPIO4	37	GPIO30
12	GPIO5	38	GPIO31
13	GPIO6	39	GPT0
14	GPIO7	40	GPT1
15	GPIO8	41	GPT2
16	GPIO9	42	GPT3
17	GPIO10	43	I2C0
18	GPIO11	44	I2C1
19	GPIO12	45	UART0
20	GPIO13	46	UART1
21	GPIO14	47	UART2
22	GPIO15	48	XADC
23	GPIO16	49	ETHERNET
24	GPIO17	50	SPI0
25	GPIO18	51	SPI1

4. Core Local Interrupter (CLINT)

The purpose of CLINT is to configure timer interrupts.

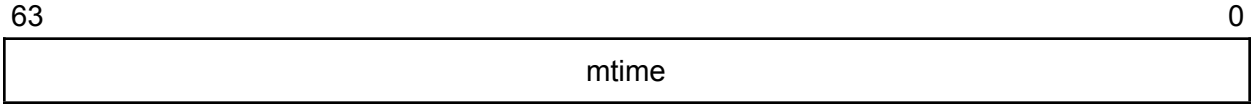
CLINT Registers

Register name	Offset address	Accessible Size	Description
mtime	‘hBFF8	64 bits	mtime register (Read and write)
mtimecmp	‘h4000	64 bits	mtime compare register (Read and write)

mtime Register

A memory mapped register of a real time counter.

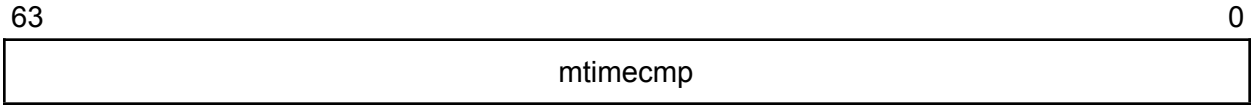
Address offset: 0xBFF8



mtimecmp Register

Machine mode timer compare register which causes a timer interrupt to be posted when the mtime register contains a value greater than or equal to the value in the mtimecmp register.

Address offset: 0x4000



mtimecmp = mtime + N * (Clock frequency / mtime divisor)

N - Number of seconds after which the interrupt is generated.
Clock Frequency - Frequency at which device clock in running
mtime divisor - Value of this divisor varies as per board
Yamuna - 256