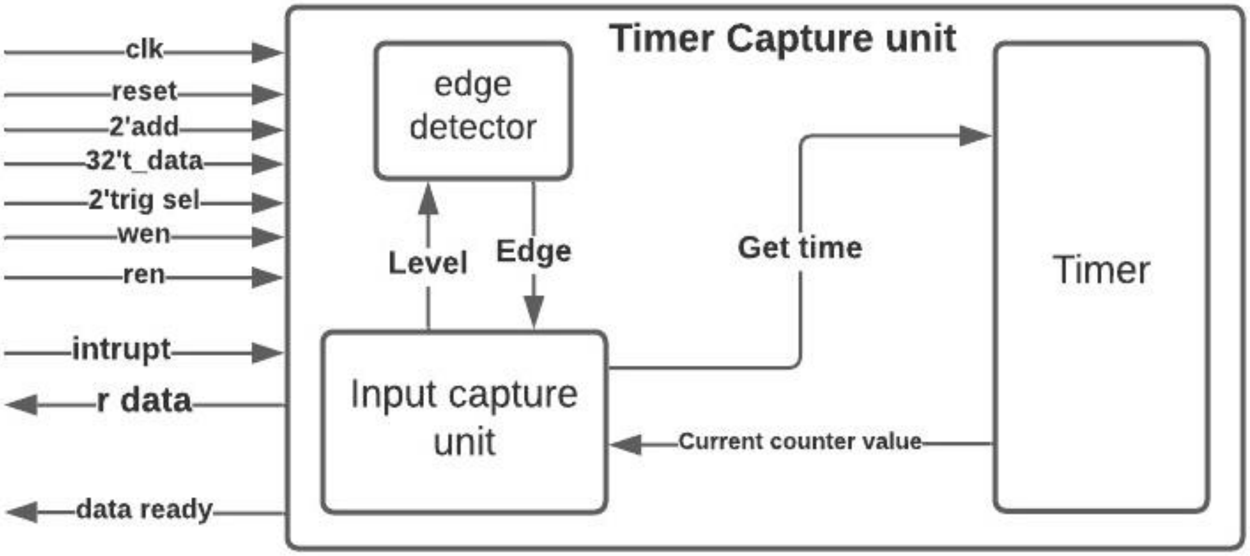
TIMER X CAPTURE UNIT

Features:

- 32-bit timer with 12-bit prescaler and 32-bit step register
- Can be used with APB (Advanced Peripheral Bus) or TL-UL (Tile link).

Block Diagram:



Hardware interface

Signal Name	Width	Direction	Description
Clk	1 bit	input	Primary clock
Reset	1 bit	input	Active low asynchronous reset
Add	2 bit	input	Address for timer write and capture unit enable.
t_data	32 bit	input	Write data for address specified.
Ren	1 bit	input	Read enable for capture unit.
Wen	1 bit	input	Write enable for capture unit.
Intrupt	1 bit	input	Input intrupt signal for ICU.
trig_tel	2 bit	input	Capture units trigger type select.
data_ready	1 bit	output	Output ready enable.
r_data	32 bit	output	Output read data.

Programmers Guide:

INITIALIZATION

The software should configure the step and prescaler bits before enabling the timer control to correctly increment the timer value if the default settings aren't required. The **t_data** register should be modified to the desired **prescaler** and **step** value with respect to their address value in **add** register for setting delay in primary **clk** signal and increment value respectively. Now the timer compare value should be written to the **mtimecmp** to set the limit of the timer counter. Moving forward now the **reset** register needs to be written 0x1 in order to activate the timer to start the counter. Now if the programmer needs to reset the timer can do it by setting the hardware reset pin 0x0.

REGISTER ACCESS

This timer is capable of handling a 32 bit timer compare and 32 bit timer register value

CONFIGURATION STEPS

1. Wake up the ICU: If the capture unit is attached with timer, then every intrupt's time comes in capture unit but ICU will not store this value until the user tell to do so by making wen enable. This is to provide more control to user to handle things as per the need.
2. Generating the request signal(get_time): The intrupt convert in request signal from timer after passing through edge detector if the user wants to capture the input as level, then the edges coming from edge detector will not be translated in real_intrupt (See details in register map section) else the request will be generated as edge.
3. Storing the time: The current counter value coming from timer will store then in intrupt_time on the positive edge of clk checking the wen signal to be enable.
4. Display: To see the last intrupt time ren signal should be enable which will let the r_data register to be filled with the intrupt time value stored in intrupt_time register and a indicating signal data_ready.

Register Map:

Timer.delay
Delay primary clock reset default
= 0x1

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
delay															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
delay															

Bits	**Type**	**Reset**	**Name**	**Description**
31:0	wo	0x1	delay	To delay the primary clock

Timer.mtime
counter reset
default = 0x0

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="16">mtime</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">mtime</td></tr>
</table>

Bits	**Type**	**Reset**	**Name**	**Description**
31:0	rw	0x0	mtime	The counter register with positive edge of secondary clock

Timer.t_data @0x2

Comparison register

reset default = 0xffffffff

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="16">mtimecmp</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">mtimecmp</td></tr>
</table>

Bits	**Type**	**Reset**	**Name**	**Description**
31:0	wo	0xffffffff	mtimecmp	The comparison register with mtime.

Timer.t_data @0x1
Increment value
reset default = 0x1

<table>
<tr><td>31</td><td>30</td><td>29</td><td>28</td><td>27</td><td>26</td><td>25</td><td>24</td><td>23</td><td>22</td><td>21</td><td>20</td><td>19</td><td>18</td><td>17</td><td>16</td></tr>
<tr><td colspan="16">step</td></tr>
<tr><td>15</td><td>14</td><td>13</td><td>12</td><td>11</td><td>10</td><td>9</td><td>8</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<tr><td colspan="16">step</td></tr>
</table>

Bits	**Type**	**Reset**	**Name**	**Description**
31:0	wo	0x1	step	The counter’s increment value/level register.

Timer.t_data @0x0

Delay setter reset

default = 0x9

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
prescaler															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
prescaler															

Bits	Type	Reset	Name	Description
31:0	wo	0x9	prescaler	The delay set according to this signal.

Timer.over_flow

Limit reached indicator reset

default = 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
														over_flow	

Bits	Type	Reset	Name	Description
1:0	wo	0x0	over_flow	When counter reached the limit it increments.

Input_capture_unit.w_data

Time write reset

default = 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
w_data															
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
w_data															

Bits	Type	Reset	Name	Description
31:0	wo	0x0	w_data	Intrupt time from timer.

Input_capture_unit.get_time @0x3

Fetch the time reset

default = 0x0

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
															Get_time

Bits	Type	Reset	Name	Description
1	wo	0x0	get_time	Signal to timer to write the current counter value in w_data.