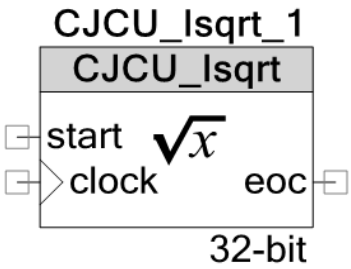
Integer Square Root Calculator (CJCU_Isqrt)

1.0

Features



- 8, 16, 24, and 32 bits
- CPU mode for easy API access
- DMA mode for HW-controlled access
- Supports PSoC 3 and PSoC 5 LP

General Description

The Integer Square Root Calculator (CJCU_Isqrt) uses hardware resources to compute the integer square root (isqrt) of a given number.

When to Use an Integer Square Root Calculator

Use an Integer Square Root Calculator to offload the CPU from having to perform the calculation. This can be to efficiently calculate several square roots in parallel (CPU mode), or to process data from a peripheral without assistance from the CPU (DMA mode).

Input/Output Connections

This section describes the various input and output connections for the Integer Square Root Calculator. An asterisk (*) in the list of I/Os indicates that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.

start – Input *

The start input signal triggers the start of a calculation. The start signal is ignored during calculation. This terminal is displayed if the **Mode** parameter is set to either **DMA-only** or **CPU/DMA**.

clock – Input

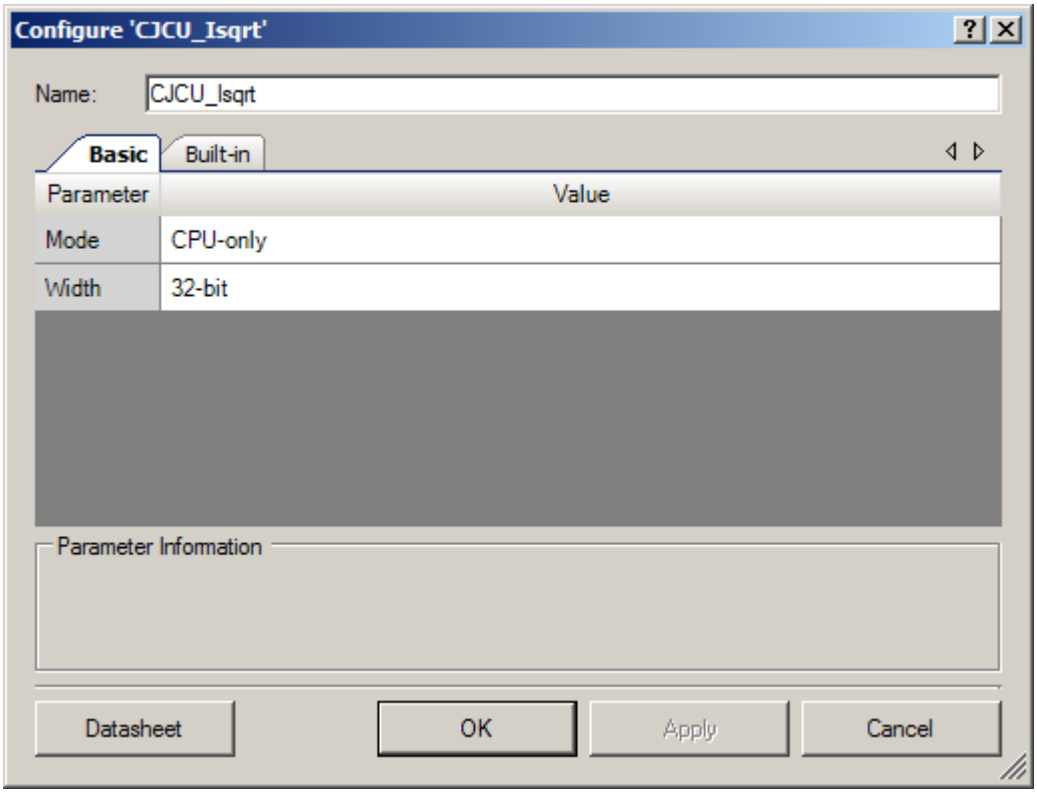
Clock source for the component.

eoc – Output *

A rising edge on the End of Calculation (eoc) output indicates that the previously requested calculation has completed. The signal will pulse high for one clock period. This terminal is displayed if the **Mode** parameter is set to either **DMA-only** or **CPU/DMA**.

Component Parameters

Drag an Integer Square Root Calculator component onto your design and double-click it to open the **Configure** dialog.



Mode

This parameter determines the access mode of the component. Valid values are **DMA-only**, **CPU-only**, and **CPU/DMA**. The default is **CPU-only**.

Width

This parameter determines the bit width of the component. Valid values are **8-bit**, **16-bit**, **24-bit**, and **32-bit**. The default is **32-bit**.

Clock Selection

There is no internal clock in this component. You must attach a clock source. This component operates from a single clock connected to the component.

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name “CJCU_Isqrt_1” to the first instance of a component in a given design. You can rename it to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance name used in the following table is “CJCU_Isqrt.”

Function	Description
CJCU_Isqrt_ComputeIsqrt()	Computes an integer square root and returns the computed value.
CJCU_Isqrt_ComputeIsqrtAsync()	Initiates the computation of an integer square root.
CJCU_Isqrt_ReadIsqrtAsync()	Reads the result of the previously requested calculation.
CJCU_Isqrt_AsyncComplete()	Reads the status of an asynchronously requested calculation.

uint8/16 CJCU_Isqrt_ComputeIsqrt(uint8/16/32 square)

Description: Computes the integer square root (isqrt) of a given number.

Parameters: uint8/16/32 square: The radicand to compute the square root of. Data type is determined by the “Width” component parameter (uint8 for 8-bit, uint16 for 16-bit, uint32 for 24-bit and 32-bit).

Return Value: uint8/16 Computed integer square root. Data type is determined by the “Width” component parameter (uint8 for 8-bit and 16-bit, uint16 for 24-bit and 32-bit).

Side Effects: None

void CJCU_Isqrt_ComputeIsqrtAsync(uint8/16/32 square)

Description: Initiates the computation of an integer square root.

Parameters: uint8/16/32 square: The radicand of which to compute the root. Data type is determined by the “Width” component parameter (uint8 for 8-bit, uint16 for 16-bit, uint32 for 24-bit and 32-bit).

Return Value: None

Side Effects: None

uint8/16 CJCU_Isqrt_ReadIsqrtAsync (void)

Description: Reads the result of the previously requested calculation.

Parameters: None

Return Value: uint8/16 Computed integer square root. Data type is determined by the “Width” component parameter (uint8 for 8-bit and 16-bit, uint16 for 24-bit and 32-bit).

Side Effects: None

uint8 CJCU_Isqrt_AsyncComplete (void)

Description: Reads the status of an asynchronously requested calculation.

Parameters: None

Return Value: uint8 calculation status.
0x00 indicates that the component is busy or that no calculation has been requested.
0x01 indicates that the requested calculation has completed.
The value is clear-on-read, so successive calls after 0x01 is returned will return 0x00.

Side Effects: None

Defines

- CJCU_Isqrt_WIDTH – Defines Integer Square Root Calculator width in bits.
- CJCU_Isqrt _CPU_MODE – Defines whether the component has CPU access enabled. A value of 1 indicates that the access mode is CPU-only or CPU/DMA, and APIs are available.
- CJCU_Isqrt_F1 – Defines the address of this component’s FIFO1.
- CJCU_Isqrt_CTL – Defines the address of this component’s Control Register. Only valid if the component is in CPU-only or CPU/DMA mode.
- CJCU_Isqrt_STS – Defines the address of this component’s Status Register. Only valid if the component is in CPU-only or CPU/DMA mode.

Sample Firmware Source Code

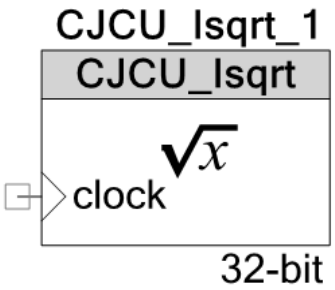
An example project and a unit test for the Integer Square Root Calculator component can be found in CJCU_IsqrtTest/CJCU_Isqrt_Tests.cywrk.

Functional Description

The Integer Square Root Calculator can interface with the CPU, with DMA, or with both.

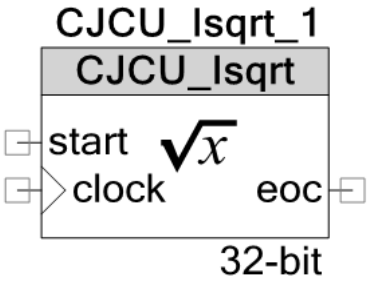
CPU-only

The default configuration of the Integer Square Root Calculator is CPU-only. This configuration provides APIs that coordinate the movement of data in and out of the component's internal FIFOs and the timing and state of the other component internals.



DMA-only

DMA-only mode allows the user to define the data flow and timing of the component. This allows the component to calculate without any CPU intervention. An example where this is useful would use a DMA component to transfer data into the Integer Square Root Calculator. This DMA's nrq terminal would hook up to the Isqrt's start terminal. Another DMA component could be used to transfer data out. Its drq would be driven by the Isqrt's eoc.



CPU/DMA

CPU/DMA mode provides the most flexibility to the user. APIs are available to use the prescribed CPU-driven data flow. Control and status signals are also exposed for the user to interface the component with hardware. An example where this is useful might use the APIs to have the CPU initiate a calculation, followed by a DMA component automatically responding to the completion of the calculation with no further CPU intervention. The symbol for CPU/DMA mode appears the same as for DMA-only mode.

It is important to ensure that the FIFO data is written and read correctly when accessed without the APIs provide in CPU mode. Before the start signal is asserted, F1 should contain N, where N

is the square of which to compute the root. After the component asserts the eoc signal, F1 contains the root. F0 is not available for CPU/DMA access. Reads and writes to F0 will be ignored.

Registers

CJCU_Isqrt_CTL

Note: This register does not exist if the component is in DMA-only mode. The register uses pulse mode: written values are present for one clock cycle, and then reset to '0'.

Bits	7	6	5	4	3	2	1	0
Value	Reserved							start

- start: Initiate a calculation.

CJCU_Isqrt_STS

Note: This register does not exist if the component is in DMA-only mode. The register uses sticky mode: A read will reset the register to '0'.

Bits	7	6	5	4	3	2	1	0
Value	Reserved							complete

- complete: Requested calculation has completed.

Resources

The Integer Square Root Calculator component is placed throughout the UDB array. The component utilizes the following resources.

Configuration	Resource Type					
	Datapath Cells	Macrocells	Status Cells	Control Cells	DMA Channels	Interrupts
8-bit	1	8	1*	1*	–	–
16-bit	2	8	1*	1*	–	–
24-bit	3	8	1*	1*	–	–
32-bit	4	8	1*	1*	–	–

*Reported Status/Control usage is for CPU mode. In DMA-only mode, the component uses 0 Status Cells and 0 Control Cells.

API Memory Usage

The component memory usage varies significantly, depending on the compiler, device, number of APIs used and component configuration. The following table provides the memory usage for all APIs available in the given component configuration.

The measurements have been done with the associated compiler configured in Release mode with optimization set for Size. For a specific design the map file generated by the compiler can be analyzed to determine the memory usage.

Configuration	PSoC 3 (Keil_PK51)		PSoC 5LP (GCC)	
	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes
8-bit	40	0	76	0
16-bit	52	0	76	0
24-bit	48	0	88	0
32-bit	48	0	76	0

DC and AC Electrical Characteristics

Specifications are valid for –40 °C ≤ T_A ≤ 85 °C and T_J ≤ 100 °C, except where noted. Specifications are valid for 1.71 V to 5.5 V, except where noted.

AC Characteristics

Parameter	Description	Min	Typ	Max[1]	Units
f_{CLOCK}	Component clock frequency				
	8-bit			47	MHz
	16-bit			39	MHz
	24-bit			35	MHz
	32-bit			31	MHz
$t_{COMPUTE}$	Maximum computation time from start pulse to eoc pulse (Using Max f_{CLOCK})				
	8-bit (30 cycles max)	0.64			µs
	16-bit (58 cycles max)	1.5			µs
	24-bit (86 cycles max)	2.5			µs
	32-bit (114 cycles max)	3.7			µs

[1]. The values provide a maximum safe operating frequency of the component. The component may run at higher clock frequencies, at which point you will need to validate the timing requirements with STA results.

Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes	Reason for Changes / Impact
1.0	Initial component version	