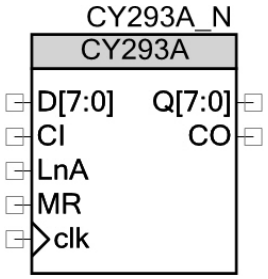
8-Bit Accumulator with Carry In and Out

CY293A v1.0



Features

- Common synchronous clock
- Synchronous load or accumulator control with single pin
- Asynchronous reset
- Simple to deconstruct

General Description

This counter is based on the 74HC293 4-bit binary counter. However it has been converted to an 8-bit accumulator. The load enable input (LnA) is used to either load the data to the accumulator or to allow the component to accumulate. If LnA is held HIGH, this component functions as an accumulator only, and it must be initialized with an asynchronous master reset (MR). Carry in (CI) and carry out (CO) sections of these accumulators to be combined from wider data lengths.

This component was built as a teaching tool. This classic component's operation is well understood and this datasheet's function is to help understand how the component was built through its deconstruction.

Pin Description

In	Type	Function
D[7:0]	inputs	data inputs
CI	input	carry input
LnA	input	load enable (active low)
MR	input	master reset
Clk	input	clock input
Q[7:0]	outputs	accumulator outputs
CO	output	carry output

Function Table

Operating Modes	Inputs			Outputs	
	MR	clk	LnA	Q	CO
reset	1	x	x	0	0
load	1	↑	0	D	0
accumulate	1	↑	1	mod(256,Q+D+CI)	256 <= Q+D+CI

Deconstructing the Component

All the datapath instructions used for this component have the parallel interface as the permanent input path for the left side of the ALU (ASRC). If you do not understand how this is done then first review the CY273 component.

For this counter there are three different operations.

- Reset
- Load
- Accumulate

While the reset operation will be handled with datapath reset circuitry, the other two operations need to be implemented with datapath instructions.

Using Creator, open the CY293A example project to see the project schematic (*TopDesign.cysch*). It has a CY293A component connected to input switches, output LEDs, and a clock.

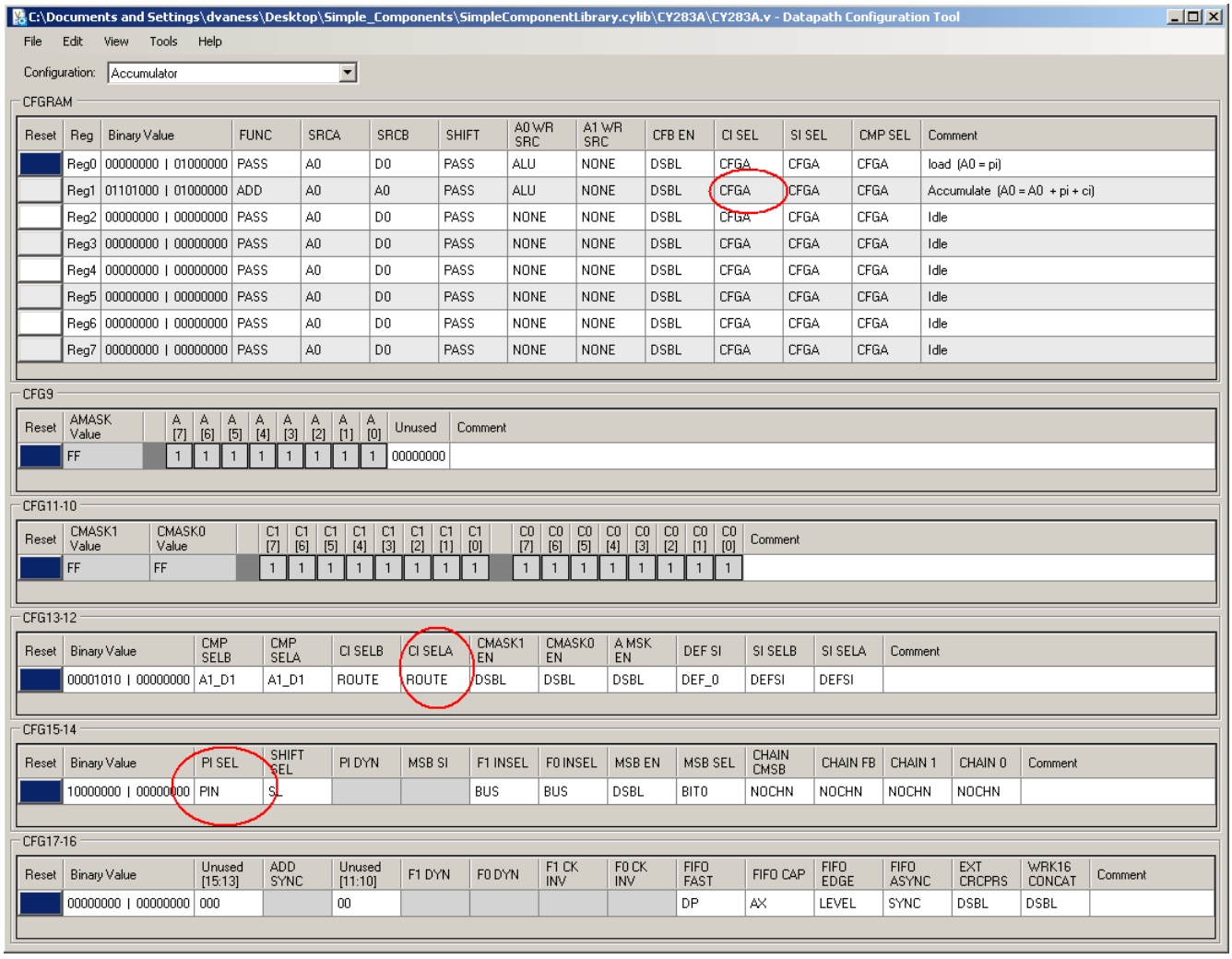
In the Workspace Explorer, click the **Components** tab. Then, right-click on the project and select **Import Component**. Navigate to where the CYCC_SimpleComponentLibrary project is, and select the CYCC_CY293A_v1_0 component. Click **OK** and the following files are shown for the component:

- Symbol file (cysym)
- Datasheet (pdf)
- Verilog File (v)

Open the symbol file to find a symbol with five inputs and two outputs. It looks like the symbol shown on the first page. There are no additional symbol parameters.

Open the Verilog file and notice that at lines 25 – 31, these definitions were passed to this Verilog file when it was created. The first 19 lines of this header list register usage and the datapath instruction definitions.

No intermediate signals are required. What follows is the datapath module definition. It was created and inserted by the Datapath Configuration Tool. This information is backward compatible so opening up the datapath configuration tool for this Verilog file results in the following.



Note that there are only two instructions and both require parallel data input. The PI_SEL bit has been enabled so the parallel input is the permanent ASRC input to the ALU.

- For the load instructions the parallel input is passed through the ALU and stored into A0.
- For the accumulate the parallel input is added to A0 and placed back in A0. The CI_SELA flag is set from the carry input to be routed to the ALU.

When saved, this tool inserts the updated configuration data back into the Verilog file along with an instance of the datapath interface. Just pass the correct signals in to and out of it and you are done.

This counter parameter list may look ominous but only a few parameters have to be entered as shown below.

```
/* input          */ .reset(MR),          // Connect to datapath reset
/* input          */ .clk(clk),           // Connect to datapath clock
/* input [02:00]  */ .cs_addr({2'b00, LnA}),  // Logic for instructions
/* input          */ .route_si(1'b0),
/* input          */ .route_ci(CI),       // Connect the carry in
/* output         */ .co_msb(CO),         // Connect to carry out
/* input [07:00]  */ .pi(D[7:0]),         // Connect parallel input to datapath
/* output [07:00] */ .po(Q[7:0])          // Connect datapath to parallel output
```

These parameters do the following:

- Connect a reset signal to the datapath reset circuitry
- Connect a clock signal to the datapath clock circuitry
- Controls which two instructions are performed
- Connects CI input to datapath
- Connects datapath to CO output
- Connect the parallel input to the datapath PI
- Connect the datapath PO to the parallel output

Support

PSoC Creator Community Components are developed and supported by the Cypress Developer Community. Go to www.cypress.com/CommunityComponents to discuss this and other Community Components.