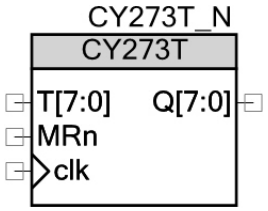
Octal T-Type Flip-Flop with Reset

CY273T v1.0

Features



- Common synchronous clock
- Common asynchronous reset
- Eight positive edge triggered T-type flip-flops
- Simple to deconstruct

General Description

This flip-flop is based on a component that was never built, but should have been. T flip-flops are very useful in the synthesis of spare matrix state machines. This component has eight edge-triggered, T-type flip-flops with individual T inputs and Q outputs. The common clock (clk) and master reset (MRn) inputs load and reset (clear) all flip-flops simultaneously. All outputs are forced LOW independently of clock or data inputs by an active LOW to the MRn input.

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 it was built through its deconstruction.

Pin Description

Pin	Type	Function
T[7:0]	inputs	data inputs
MRn	input	master reset input (active low)
clk	input	clock input
Q[7:0]	outputs	Flip-flop outputs

Function Table

Operating Modes	Inputs			Outputs
	MRn	clk	Tx	Qx
reset	0	x	x	0

Operating Modes	Inputs			Outputs
	MRn	clk	Tx	Qx
load	1	↑	0	Qx
	1	↑	1	~Qx

Deconstructing the Component

This example shows that with a couple simple changes of a single datapath instruction that a component can be changed from a octal D-type flip-flop to an octal T-type flip-flop. This component requires that the parallel input data be the permanent input to the left side of the ALU (ASRC). If you do not understand how this is done then first review the CY273 component. It also shows how to route a serial input into a register.

The parallel out (PO) interface is always available and connects to either A0 or A1. The parallel in (PI) interface must be routed into the left input of the ALU (ASRC). If the PI_SEL bit (CFG15.7) is set, the PI is the permanent input to ASRC.

The toggle operation is implements by taking the register (AO) and XOR with the parallel input data. The datapath instruction will implement this.

For this counter there are two different operations.

- Reset
- Load

While the reset operation will be handled with datapath reset circuitry, the load operation needs to be implemented with a datapath instruction.

Using PSoC Creator, open the CY273T example project to see the project schematic (*TopDesign.cysch*). It has a CY273T component connected to input switches, output LEDs, and a clock.

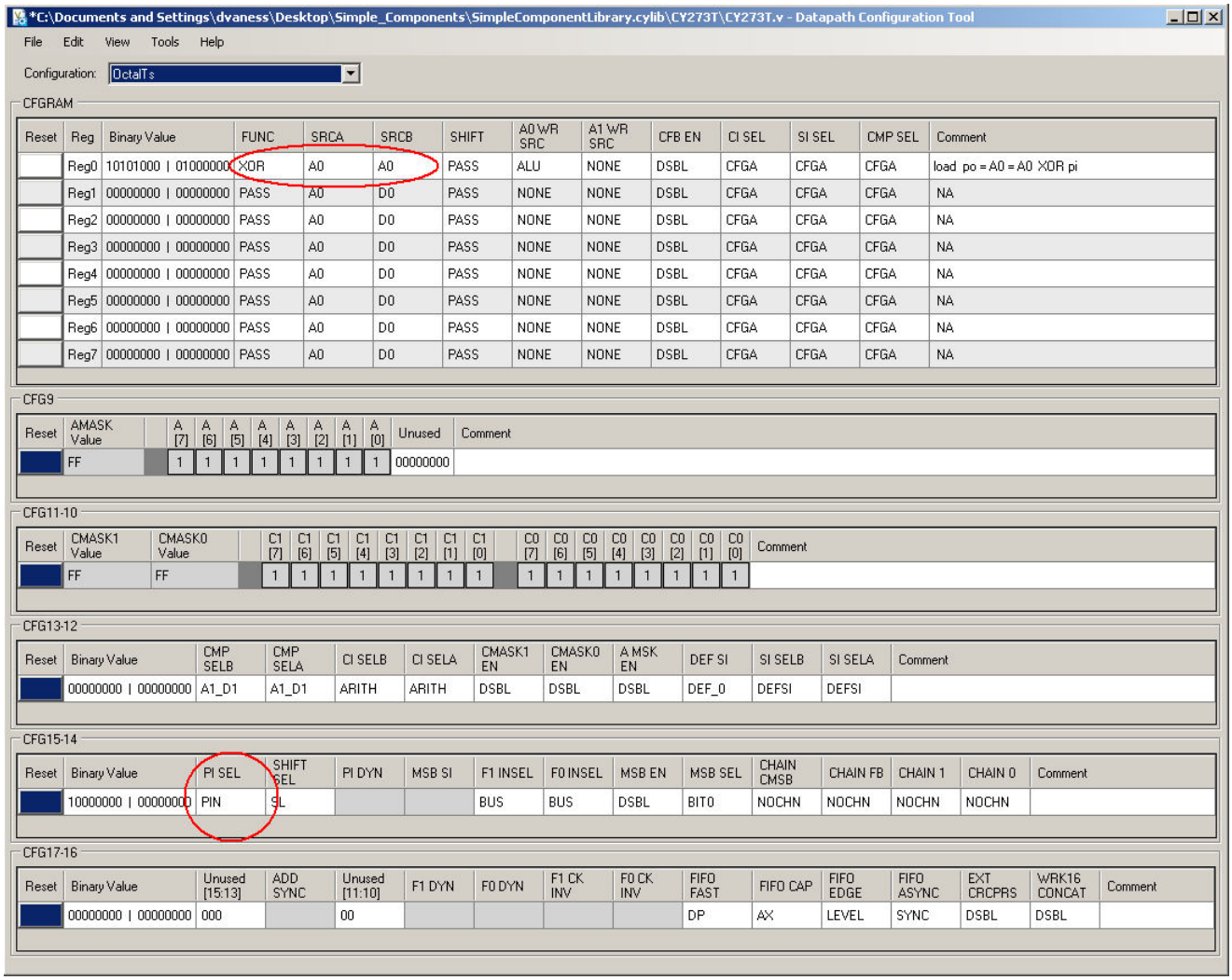
In the Workspace Explorer, click the **Components** tab. Then, right-click on the project and select **Import Component**. Navigate to where the CYCC_SimpleComps library project is, and select the CYCC_CY273T_v1_0 component. Click **OK** and the following files are shown for the component:

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

Open the symbol file to find a symbol with three inputs and one output. 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 24 – 27, these definitions were passed to this Verilog file when it was created. The first 25 lines of this header list register usage and the datapath instruction definitions.

There is a need for intermediate signals and this is handled in lines 31 - 32. 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 is only a single instruction and it requires 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 XORed with A0 and placed back in A0.

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(reset),   // Connect reset to datapath
/*  input             */  .clk(clk),       // Connect clock to datapath
/*  input   [02:00]   */  .cs_addr(3'b0),  // Only one instruction (REG0) is used
/* input [07:00]      */  .pi(T[7:0]),     // Connect parallel data 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
- Allows only a single instruction to be continuously processed
- 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.

© Cypress Semiconductor Corporation, 2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control, or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

This work is licensed under a Creative Commons Attribution 3.0 Unported License.

http://creativecommons.org/licenses/by/3.0/deed.en_US

You are free to:

- Share — to copy, distribute and transmit the work
- Remix — to adapt the work
- Make commercial use of the work