AN2358

Manchester Decoder Using PSoC® 1

Author: Philippe Larcher

Associated Project: Yes

Associated Part Family: CY8C29x66, CY8C27x43, CY8C24X94, CY8C24x23A, CY8C23x33, CY8C21x34, CY8C21x23

Software Version: PSoC® Designer™ 5.4

Related Application Notes: None

This application note describes how to build a Manchester Decoder using two digital blocks and some combinatorial logic of the PSoC® 1 device. After it is initialized, the decoder requires no firmware intervention. Clock and serial data recovered by the receiver can serve as inputs for a number of serial data communication methods including SPI and pattern recognition circuits.

Contents

Introduction

Manchester code is widely used in communication systems because of its simplicity: a single signal conveys data and clock information without the need for high-level protocol. Additional benefits include self-synchronization, zero DC components, and independence from transmission media. A Manchester link consists of a transmitter (Manchester encoder) and a receiver (Manchester decoder).

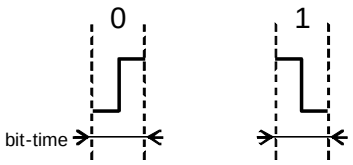
Manchester decoding is more complex because it requires extracting clock and data information from a single signal. This application note describes a Manchester decoder based on digital reconstruction; implemented with only two PSoC 1 digital blocks; it is portable to a PSoC 1 device family which contains at least two digital blocks, such as CY8C29x66, CY8C27x43, CY8C24x94, CY8C24x23A, CY8C23x33, CY8C21X34, and CY8C21x23.

Decoding speed is programmable and can easily achieve 200 kbps or more.

Manchester Code Principle

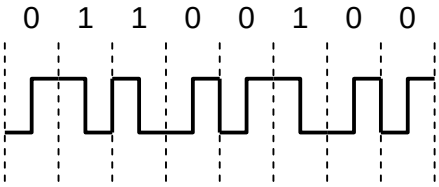
Manchester code embeds clock information with data in a very simple way: each bit is transmitted with a transition in the middle of bit time. For a ‘0’, transition is 0 to 1, for a ‘1’, transition is 1 to 0 (see Figure 1).

Figure 1. Manchester Coding for Bit Values 0 and 1



When transmitting successive bits, additional transitions are inserted between bits to satisfy the mid-bit transition rule, as represented in Figure 2.

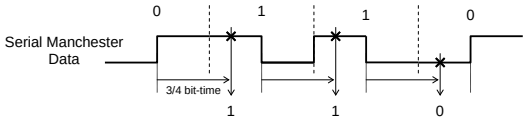
Figure 2. Transmitting Multiple Bits



Manchester Decoder - The Digital Way

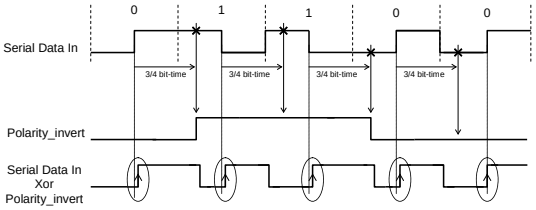
The method described here is not based on the direction of mid-bit edge. It is based on the fact that the bit value is present during the first half of bit time, prior to the transition edge. If a delay of three-fourths bit time is triggered by the incoming mid-bit transition, the value captured at the end of the delay tells the next bit value (see Figure 3).

Figure 3. Capturing the Next Bit Value



Note If the next bit value is '1', the receiver sets a signal to invert the input bit stream polarity. Therefore, the next signal transition appears as a low-to-high transition as shown in Figure 4. If the next bit value is '0', the receiver resets the inverted signal.

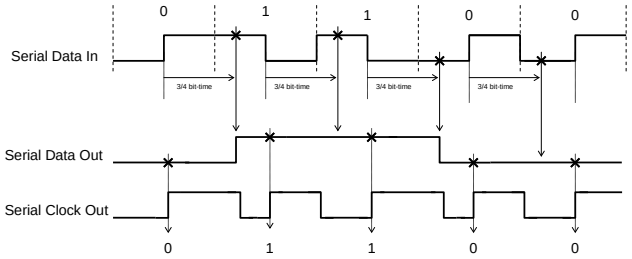
Figure 4. Inverting Mid-Bit Transition



The next step is to simply rename "Polarity_invert" to "Serial Data Out" and "Serial Data In XOR Polarity_invert" to "Serial Clock Out", following which the Manchester receiver is complete (see Figure 5).

Some considerations are still required for the first transition versus the idle state of Serial Data In, which are discussed at the end of this application note, after the description of implementation.

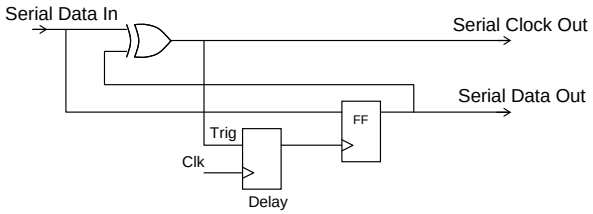
Figure 5. Manchester Receiver Outputs



PSoC 1 Implementation

The previous description translates into the following block diagram.

Figure 6. Manchester Receiver Block Diagram



As shown in the block diagram, the following three functions are required:

- A XOR gate to generate the Serial Clock Out
- A D Flip-Flop to register the state of Serial Data In when the delay expires
- A counter to generate a three-fourths bit time delay, triggered by the XOR-gate output (always a 0-to-1 transition).

The following sections discuss the PSoC 1 implementation for these functions.

XOR Gate

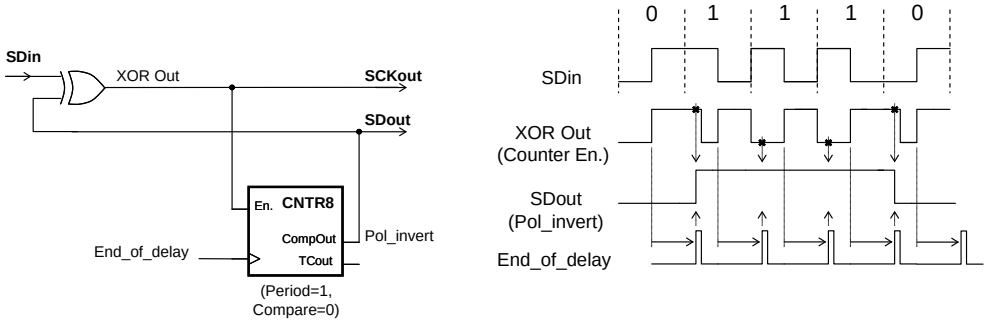
This function is easily implemented with one of the Look-Up Tables (LUT) placed on output rows and does not require further explanation.

D Flip-Flop

There is no D Flip-Flop function directly available in the PSoC 1 architecture. However, it is possible to implement a “conditional T Flip-Flop” with a digital block configured as a counter. If the counter period is set to ‘1’ and the compare value to ‘less than or equal to 0’, then the compare out signal toggles upon each clock cycle when enable is high.

With this arrangement, we just need to connect the enable signal to the XOR gate output, and the Polarity_invert signal only toggles when there is a change (0 to 1 or 1 to 0) in the input bit value, as seen in Figure 7.

Figure 7. Emulation of D Flip-Flop with Counter and Enable

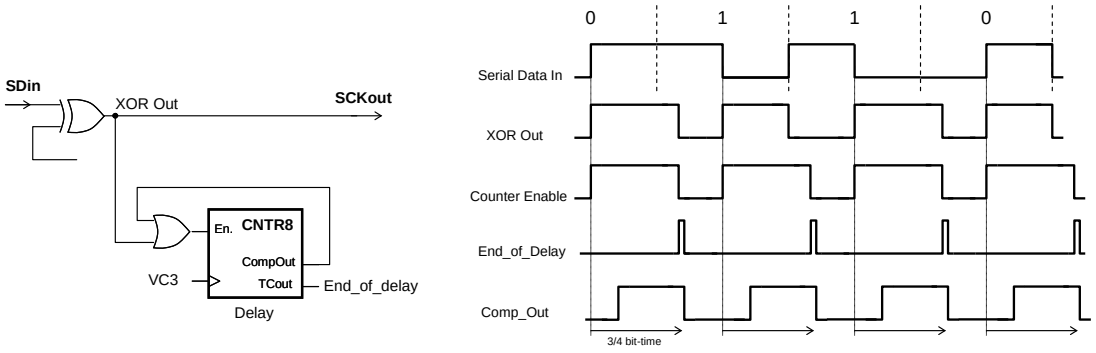


Delay Counter

The XOR gate output can be used as an enable signal to start counting. However, its duration may be only one-half bit time wide, which is insufficient to cover a three-fourths bit time counting period.

The problem is solved by OR-ing the initial enable signal with a counter compare out signal, thus extending the enable duration over the whole delay period, as shown in Figure 8. The combinatorial OR gate is implemented with a row LUT.

Figure 8. Three-Fourths Bit Time Delay Generation



Note that the polarity control is updated on the rising edge of TCout, causing the XOR Out enable signal to be reset one clock cycle prior to CompOut. This prevents any edge crossing.

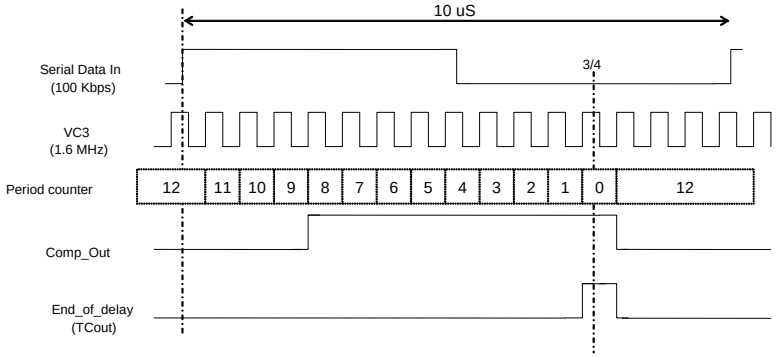
VC3 (counter clock frequency) counts three-fourths of bit time; this requires a VC3 clock at least four times the bit rate. However, tolerance must be added to cope with intrinsic precision and jitter of the transmitter and receiver.

Selecting an x16 oversampling rate gives more than 10 percent frequency tolerance on each side and is the retained value for the design.

For example, if the transmission bit rate is 100 kbps, then the clock frequency for the delay counter is 1.6 MHz, that is, VC3 = SysClk/15.

Figure 9 shows the adequate value of period and compare out for a three-fourths bit time delay. These values depend only on the selected oversampling rate and do not need to be changed when changing the transmission speed through VC3.

Figure 9. Period and Compare Out Values for Delay Counter



Place and Route

Figure 10 represents the whole Manchester receiver function. The PSoC 1 device offers several possibilities to internally route back block output signals to block input (as required, for example, for XOR Out). However, internal feedbacks may create placement or pinout constraints if the receiver is imported into an existing design. Therefore, as shown in Figure 11, external feedbacks are recommended, at the cost of three additional input pins. Figure 12 is a screenshot of the User Module placement and routing in the associated PSoC Designer™ project.

Figure 10. Manchester Receiver Final

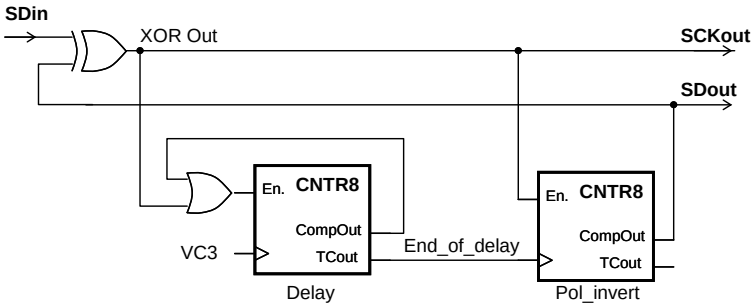


Figure 11. Manchester Receiver for Flexible Place and Route

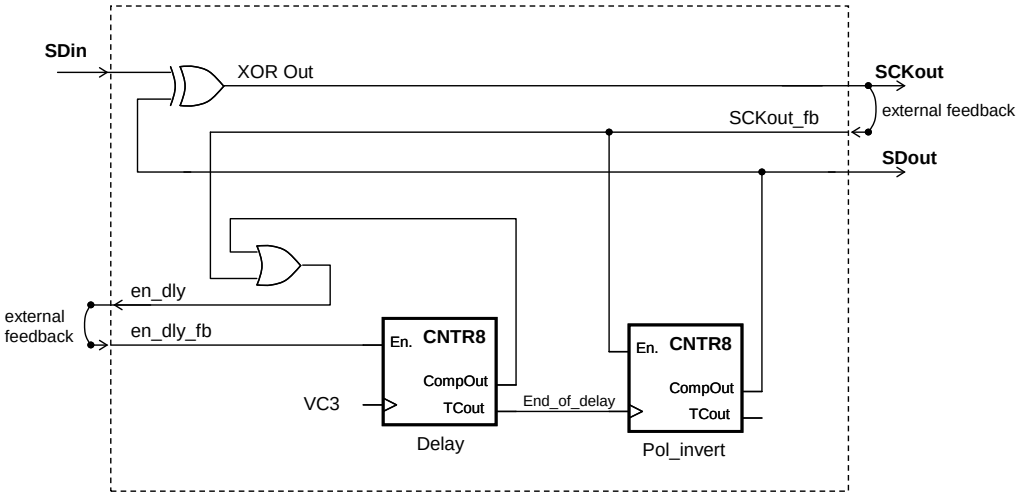
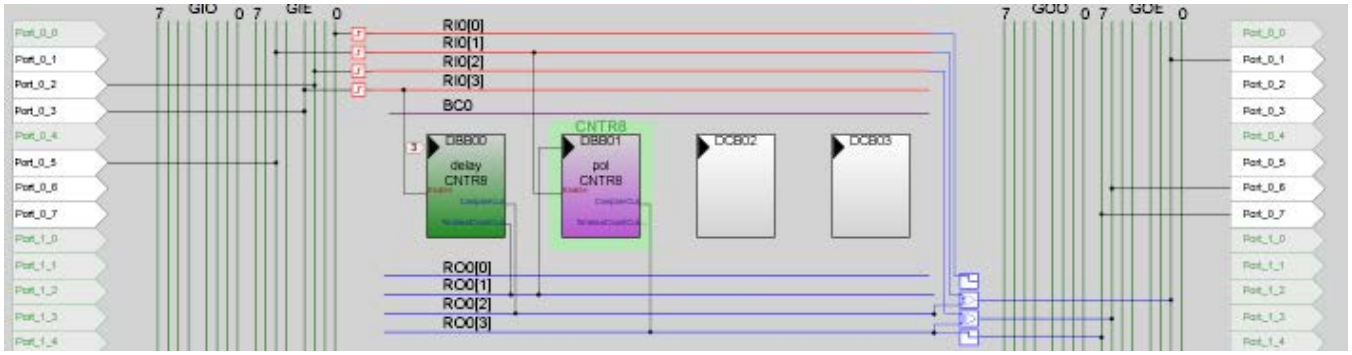


Figure 12. Manchester Receiver, Device Editor Interconnect View



Porting the Manchester Decoder

The project associated with this application note can be used independently or can be plugged into a larger application. The setup is simple.

Hardware

- Externally connect SCKout and SCKout_fb, en_dly and en_dly_fb.
- Set VC3 frequency to 16 times the serial bit rate.

Firmware Initialization

- Start the Delay counter and the Pol_invert counter.
- Readjust the Pol_invert counter period to '1' (correct startup requires initial period to be '0').

Synchronizing on the Right Edge

Until now, the Manchester receiver behavior analysis assumed that the first detected edge was at mid-bit transition. However, there are cases where the first transition seen by the receiver is an inter-bit transition.

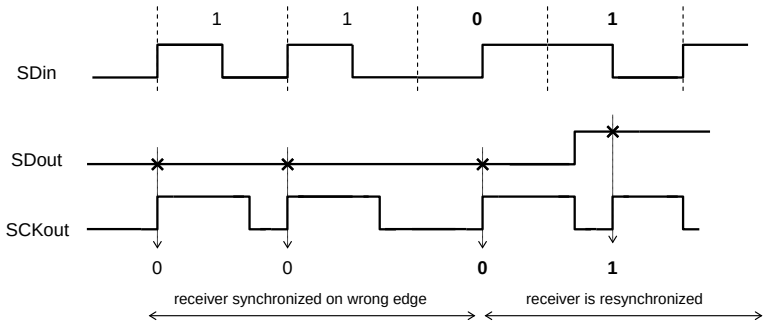
For example, this occurs if the idle state of the line is low and the first received bit is a '1' (or inversely), or, if the receiver is randomly enabled in the middle of a packet transmission. Such initial conditions translate into synchronization aliasing. However, a worthy receiver should be able to dynamically recover from this situation without manual intervention.

Initial State and First Bit Mismatch

Figure 13 shows a situation where the idle state of the serial input is low, and the first bit sent by the transmitter is a '1'. In good faith, the receiver considers the first transition to be a mid-bit transition, and starts extracting the serial data from this point.

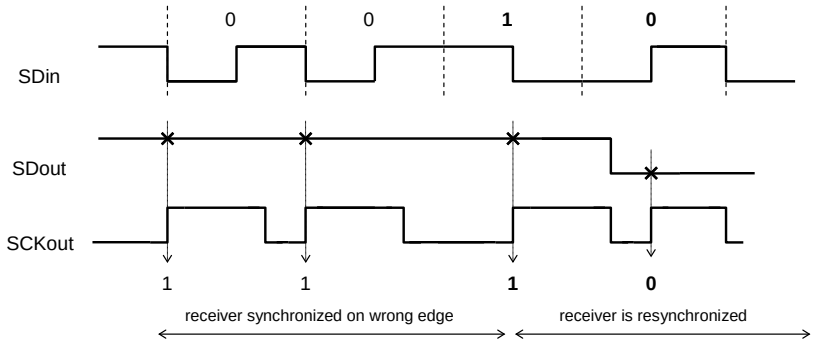
Figure 14 shows that faulty synchronization stops as soon as a '0' is received. This causes the receiver to resynchronize correctly and definitely, because a '1-then-0' bit sequence (and '0-then-1') only exhibits mid-bit transitions and no inter-bit transitions. This forces receiver realignment on the right edge.

Figure 13. Idle State Low and First Received Bit is '1'



The same sequence is repeated when the idle state of the line is high and the first received bit is a ‘0’ (see Figure 14). In this case, the correct synchronization starts with the first ‘1’ that is received.

Figure 14. Idle State High and First Received Bit is ‘0’



Synchronization errors due to hazardous enabling of the receiver during packet transmission create the same situations as described above, and are auto-recovered in the same way.

Auto Synchronization

As discussed in the previous section, the condition for correct synchronization is simple. The transmitter must ensure either one of the following:

- The first bit sent complies with the idle state of the line; for example, always start messages with a ‘0’ bit when the idle state is low.
- Or, send a 0 to 1 (or 1 to 0) preamble to guarantee a correct synchronization in any situation. The preamble can extend beyond these two bits to add other benefits such as speed synchronization, pattern recognition, and so on.

Summary

In this application note, a robust Manchester Decoder has been implemented using two PSoC 1 digital blocks. The recovered serial clock and serial data can feed a number of serial data communication methods including SPI and pattern recognition circuits.

About the Author

Name:	Philippe Larcher
Title:	Field Application Engineer, Cypress France
Background:	25 years of activity in computer and electronic system design. Authored several application notes, articles, and the book “VHDL, Introduction à la Synthèse Logique.”

Document History

Document Title: Manchester Decoder Using PSoC® 1 - AN2358

Document Number: 001-17375

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	1351984	HMT	08/06/2007	Entering existing application note in the spec system.
*A	1629363	JVY	10/15/2007	Updated to new template.
*B	2818666	XKJ	12/01/2009	Changed title to Manchester Decoder Using PSoC® 1. Updated content to be compatible with PSoC Designer™ 5.0.
*C	3102584	DASG	12/06/2010	The project is updated to PSoC Designer 5.1 In the project, the Counter (pol) period is set to 1. In the project's main SCKout_fb value is corrected to P0_5 Pin (Line 53).
*D	4221763	RJVB	12/16/2013	Updated to new template. Completing Sunset Review.
*E	4622492	ASRI	01/13/2015	Updated Software Version as “PSoC® Designer™ 5.4” in page 1. Updated Related Application Notes (Removed references of AN2281, AN2325 as these ANs are obsolete). Updated attached associated project to PSoC Designer 5.4.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc cypress.com/go/plc
Memory	cypress.com/go/memory
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/usb
Wireless/RF	cypress.com/go/wireless

PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com