CECS 460 - Transmit Engine

Marc Dominic Cabote

014938597

Spring 2018

TRANSMIT ENGINE

Prepared by:	Date:	Revision:
Marc Dominic Cabote	13 March 2018	1

TABLE OF CONTENTS

TABLE OF TABLES

TABLE OF FIGURES

TRANSMIT ENGINE

Prepared by:	Date:	Revision:
Marc Dominic Cabote	13 March 2018	1

1 INTRODUCTION

This project is intended for introducing Universal Asynchronous Receiver Transmitter (UART)—this will only be the transmit part. At the heart of the Tx engine is a shift register that shifts out data one bit at a time. The baud rate determines the speed at which data will be transferred. This document will also be modified once the full UART is designed.

1.1 PURPOSE

The purpose of this document is to specify the requirements for the transmit engine. This document will serve as the initial spec for the entire UART. This document will then be modified for the next part of the project which will have the receive engine as well.

2 REQUIREMENTS

The requirements for this project is pretty straightforward:

i) To be able to monitor TxRdy in the TxRdy SR Flip-Flop
ii) When the TxRdy is high, the transmit should be able to continuously transmit "CSULB CECS 460 – [LINE COUNT] <CR> <LF>"—where the characters are in hex and converted to ASCII.

3 TOP LEVEL DESIGN

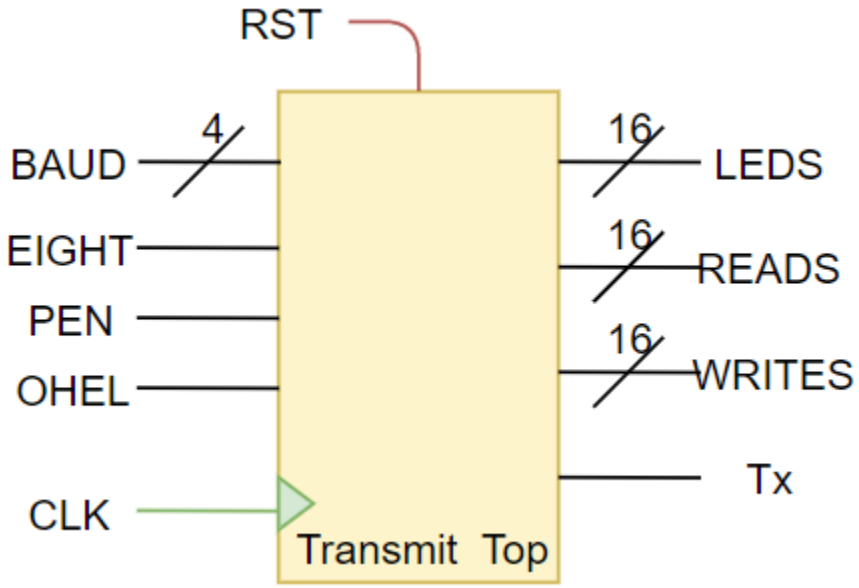


FIGURE 1 TOP LEVEL DESIGN

3.1 DESCRIPTION

The transmit engine has the logic needed to communicate with a computer using UART. For the terminal to have a stable connection in order to output the proper characters, the program—Realterm—should know what the setting is for the transmit engine—it should know the baud rate, odd or even parity and number of bits. If the program settings match the FPGA, it will then show “CSULB CECS 460 – [LINECOUNTER] <CR> <LF>”. If the settings do not match, different characters will show up.

3.2 BLOCK DIAGRAM

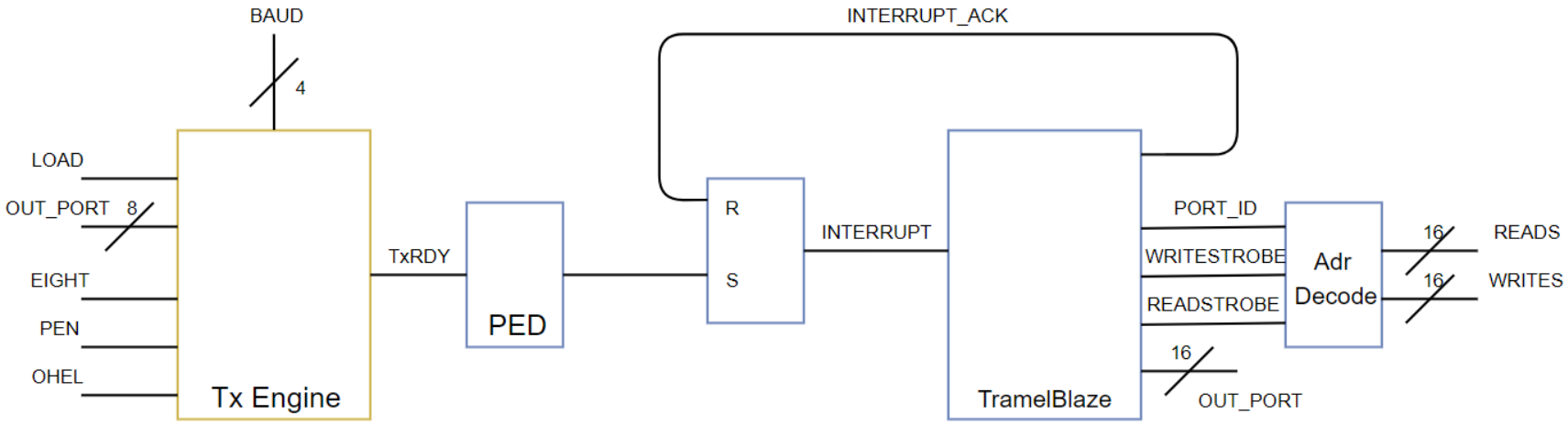


FIGURE 2 TOP LEVEL BLOCK

TRANSMIT ENGINE

Prepared by:	Date:	Revision:
Marc Dominic Cabote	13 March 2018	1

3.3 I/O

Inputs:

i) 4-bit Baud Rate

TABLE 1

CASE	COUNT	BITS PER SECOND
0000	333,333	300
0001	83,333	1200
0010	41,667	2400
0011	20,833	4800
0100	10,417	9600
0101	5,208	19200
0110	2,604	38400
0111	1,736	57600
1000	868	115200
1001	434	230400
1010	217	460800
1011	109	921600

ii) Eight

-If set high, 8 bits of data will be used instead of default 7

iii) Pen

-Enables parity

iv) OHEL

-If high checks odd parity, if low checks even parity

v) Clock

-100 MHz on-board clock

vi) Reset

-Reset is fed to the AISO which synchronizes the reset for all the registers

Outputs:

i) LEDS

-for debugging the tramelblaze

ii) Tx

-transmits the data

3.3.1 Signal Names

Inputs:
Clk
Rst
Eight
Pen
Ohel
[3:0] baud

Outputs:
Tx
[15:0] leds
[15:0] reads
[15:0] writes

3.3.2 Pin Assignments

TABLE 2 – Pin Assignments

Signal Name	Pin Location
clk	E3
ohel	L16
pen	M13
eight	R15
baud[0]	R17
baud[1]	T18
baud[2]	U18
baud[3]	R13
rst	N17
Tx	D4
leds[0]	H17
leds[1]	K15
leds[2]	J13
leds[3]	N14

Signal Name	Pin Location
leds[4]	R18
leds[5]	V17
leds[6]	U17
leds[7]	U16
leds[8]	V16
leds[9]	T15
leds[10]	U14
leds[11]	T16
leds[12]	V15
leds[13]	V14
leds[14]	V12
leds[15]	V11
----------	----------
----------	----------

4.1 BLOCK DIAGRAM OF THE TRANSMIT ENGINE

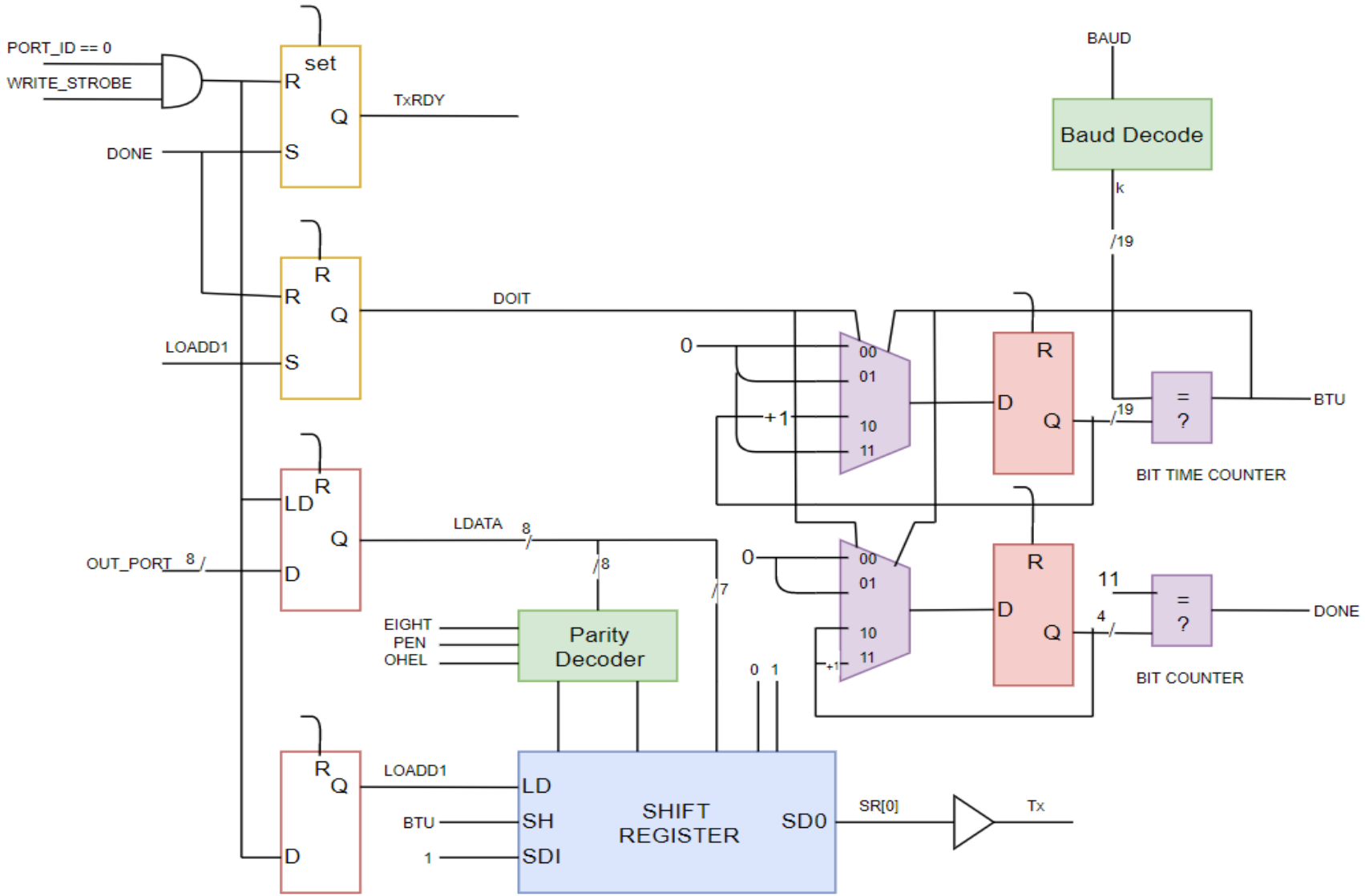


FIGURE 3 DETAILED TRANSMIT ENGINE

The highlighted harts are the important blocks for the transmit engine

4.1.1 Shift Register

The Shift Register with the parity decoder is basically the heart of the transmit engine. The Shift Register shifts out data one bit at a time, depending on the parity decoder settings. The shift register is 11'b11_1111111_11 at default—this tells us that this is the mark, and it is ready to transmit data. Once all the bits of data are shifted out, done and txrdy will go high. This is then recognized by the tramelblaze which converts the data into hex values to be converted to ASCII.

4.1.2 Bit Time Counter

The bit time counter tells us how fast the shift register should shift out data—which we can then say that it determines how fast data should be transmitted. The bit time counter is tied with the baud decoder. This determines when bit time up will go high. Bit time up then goes to the shift register—when set high, it tells the shift register to shift data out.

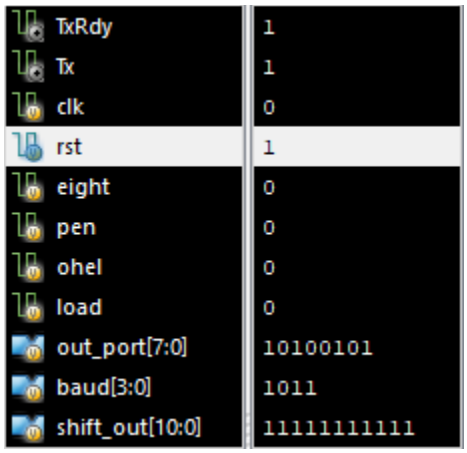
4.1.3 Bit Counter

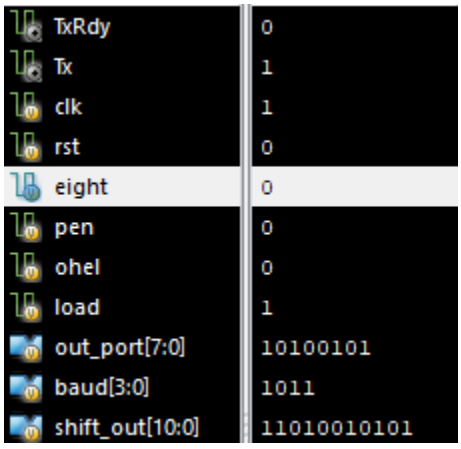
The bit counter has a very similar code with the bit time counter. While the bit time counter takes care of the how fast each data should be transmitted—baud rate—the bit counter counts up to 11 in decimal. It will tell us that once all 12 bits of data have been shifted out, "done" will go high telling us that the data has been transmitted.

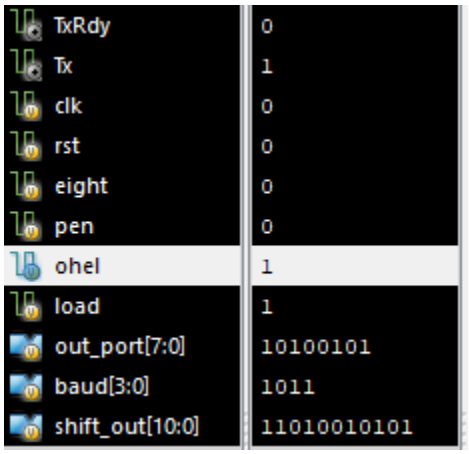
4.2 TRANSMIT ENGINE VERIFICATION

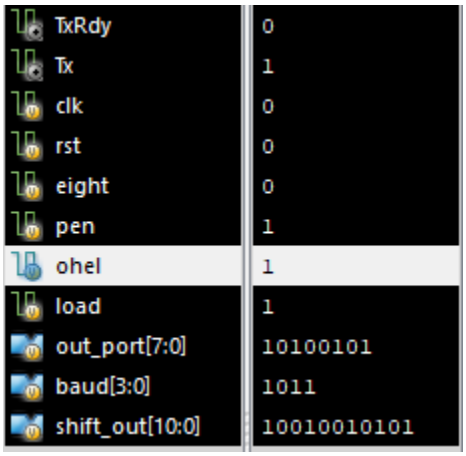
The transmit verification is pretty straightforward out_port is set to be 8'b1010_0101 or 8'hA5. Reset is exercised initially and baud rate is set to the fastest setting for easier verification. All the cases for {eight, pen, ohel} are then verified if they are properly working. Expected outputs are commented in the test fixture to verify shift register outputs on the waveforms. If the expected outputs are the same as the waveform outputs. This then verifies that the transmit engine works. It will also be seen that the TxRdy is high at reset.

4.2.1 Transmit Engine Verification Waveform









These are just some of the verified results. The entire waveform will be shown in the next page

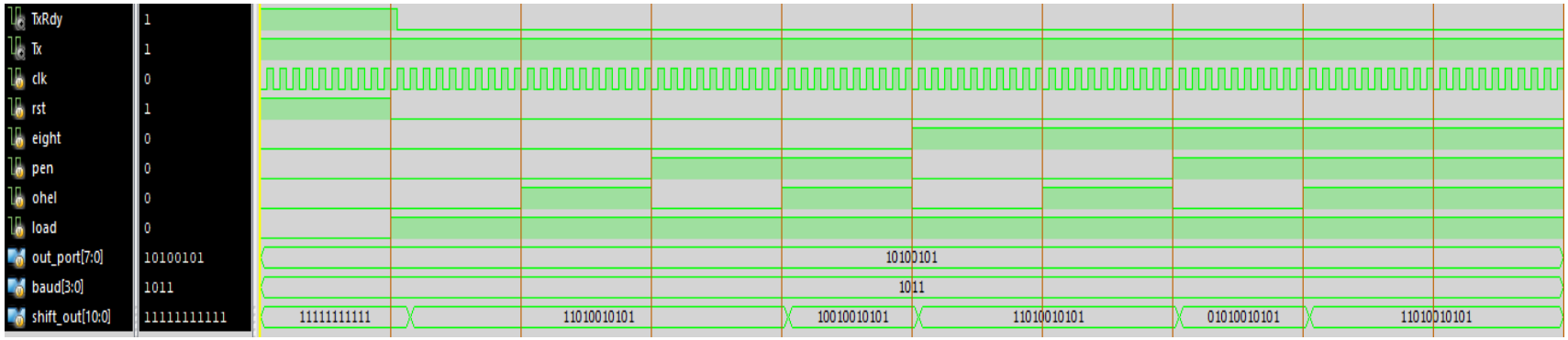


FIGURE 3 DETAILED TRANSMIT ENGINE

5 THE TRAMELBLAZE

The Tramelblaze's purpose for this project is an external block which is used to output the hex values needed. The hex values will then be converted by Realterm as ASCII characters. This will then display the desired output which is "CSULB CECS 460 – [COUNT] <CR> <LF>". It also outputs values at a different port which is tied to the LEDs to for debugging purposes.

A SOURCE CODES

The codes will all be presented starting in the next page onwards