WAVEFORM GENERATOR PROJECT

MICRO-ARCHITECTURE DOCUMENT

Table of Content

1 Intended Use of This Document 5

2 Overview 5

3 Functionality 7

3.1 Waveform Generator Module (wave\_gen\_top) 7

3.1.1 Waveform Generator Package (wave\_gen\_pkg) 8

3.1.2 UART Top Module (uart\_top) 8

3.1.3 Command Handler Module (cmd\_handler) 12

3.1.4 Enable Generator Module (en\_gen) 14

3.1.5 Waveform Generator Module (wave\_gen) 14

3.1.6 SCLK Generator Module (sclk\_gen) 15

3.1.7 Synchronous Data Generator Module (sync\_data\_gen) 15

4 Communication between FPGA and PMOD DA1 16

Table of Figures

[Figure 1 : Top level block diagram 5](#_Toc527580251)

[Figure 2: Design Schematic 7](#_Toc527580252)

[Figure 3: UART transmit module state machine 10](#_Toc527580253)

[Figure 4: UART receive module state machine 12](#_Toc527580254)

[Figure 5: Command handler module state machine 13](#_Toc527580255)

[Figure 6: Timing Diagram of PMOD DA1 Interface 16](#_Toc527580256)

[Figure 7: DA1 16 Bits Data Structure 17](#_Toc527580257)

Table of Tables

[Table 1: Frequency Selection 6](#_Toc527580260)

[Table 2: Waveform Selection 6](#_Toc527580261)

[Table 3: Information Write Selection 6](#_Toc527580262)

[Table 4: Wave\_gen\_top design port list 7](#_Toc527580263)

[Table 5: UART top module port list 8](#_Toc527580264)

[Table 6: UART baudrate generator module port list 9](#_Toc527580265)

[Table 7: UART transmit module port list 9](#_Toc527580266)

[Table 8: UART receiver module port list 11](#_Toc527580267)

[Table 9: Command hander module port list 13](#_Toc527580268)

[Table 10: Enable generator module port list 14](#_Toc527580269)

[Table 11: Waveform generator module port list 14](#_Toc527580270)

[Table 12:SCLK generator module port list 15](#_Toc527580271)

[Table 13: Synchronous data generator module port list 15](#_Toc527580272)

[Table 14: Control Bit Values for AD7303 17](#_Toc527580273)

# Intended Use of This Document

This document provides the details of the digital design of waveform generator block.

# Overview

Waveform Generator is supposed to generate the following waveform types with ten different frequencies.

* Saw-tooth wave
* Square wave
* Triangular wave

The design will be placed on the FPGA on a Nexys 4 DDR board. The board will be connected to a computer and the commands will be sent using the keyboard ASCII characters over the UART interface to select the “Waveform Types” and the “Frequencies” of the waveforms.

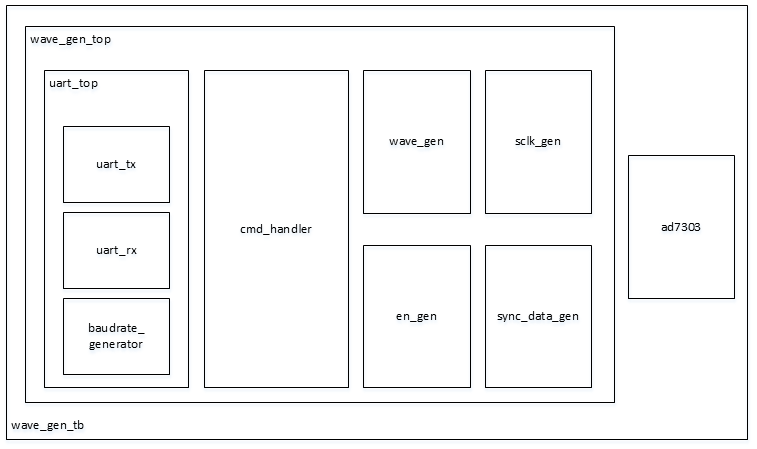


Figure 1 : Top level block diagram

The frequencies of the waveforms can be programmed to the following frequencies over a UART interface. Limit of the counter in en\_gen module will be changed with given commands.

* Default counter limit of en\_gen module is 1000 and the default wave type for wave\_gen\_top is saw-tooth.
* The counter limit will be modified according to the following ASCII table. The frequency adjustment will be done from the keyboard.

Also, version and baudrate informations will be printed according to received command over UART.

Table 1: Frequency Selection

|  |  |  |  |
| --- | --- | --- | --- |
| **ASCII** | **Decimal** | **Counter Limit** | **Frequency** |
| 0 | 48 | 1000 | 390 Hz |
| 1 | 49 | 900 | 434 Hz |
| 2 | 50 | 800 | 488 Hz |
| 3 | 51 | 700 | 558 Hz |
| 4 | 52 | 600 | 651 Hz |
| 5 | 53 | 500 | 781 Hz |
| 6 | 54 | 400 | 976 Hz |
| 7 | 55 | 300 | 1.3 kHz |
| 8 | 56 | 200 | 1.953 kHz |
| 9 | 57 | 100 | 3.9 kHz |

Table 2: Waveform Selection

|  |  |  |
| --- | --- | --- |
| **ASCII** | **Decimal** | **Wave Type** |
| A | 48 | Saw-tooth |
| B | 49 | Square |
| C | 50 | Triangle |

Table 3: Information Write Selection

|  |  |  |
| --- | --- | --- |
| **ASCII** | **Decimal** | **Write Type** |
| a | 61 | Version |
| b | 62 | Baudrate |

Folder structure should be:

wave\_gen

* design
  + implement
    - wave\_gen.tcl
    - wave\_gen.xdc
  + rtl
    - command\_handler
      * cmd\_handler.vhd
    - dac\_controller
      * en\_gen.vhd
      * sclk\_gen.vhd
      * sync\_data\_gen.vhd
    - uart
      * baudrate\_gen.vhd
      * u\_rx.vhd
      * u\_tx.vhd
      * uart\_top.vhd
    - waveform\_generator
      * wave\_gen.vhd
      * wave\_gen\_pkg.vhd
      * wave\_gen\_top.vhd
  + run
    - run.do
  + verif
    - tb\_top\_level.vhd
    - DAC RTL simulation model.vhd
* docs
  + design
    - Waveform\_Generator\_MAD.docx

# Functionality

The detailed top-level schematic of the widtdesign is give in Figure 2.

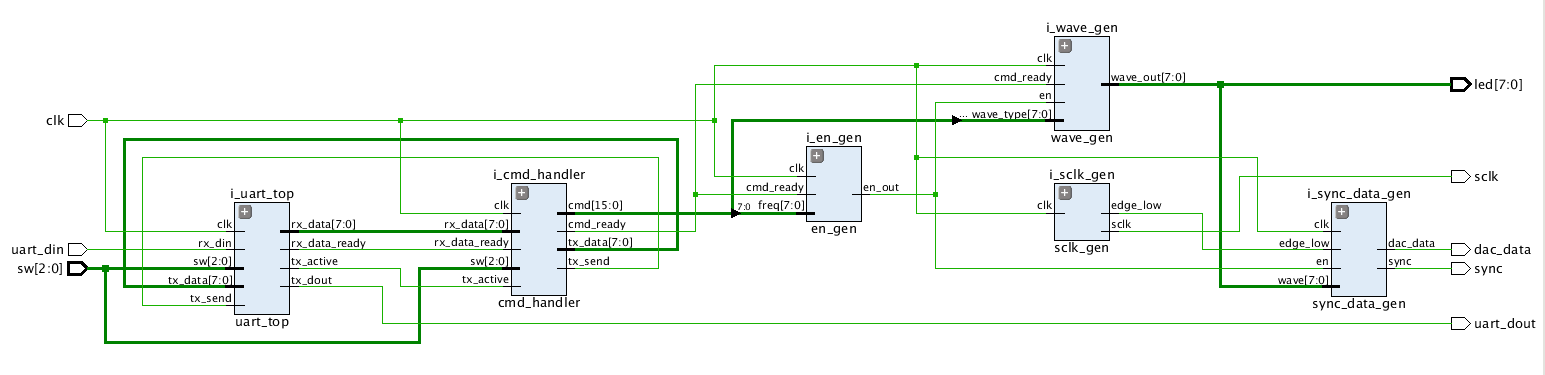


Figure 2: Design Schematic

## Waveform Generator Module (wave\_gen\_top)

This is the top level which instantiates and connects all the modules provided in the coming sections.

Table 4: Wave\_gen\_top design port list

|  |  |
| --- | --- |
| File Name | wave\_gen\_top.vhd |
| Entity Name | wave\_gen\_top |
| Architecture Name | Struct |
| Generics |  |
| *osc\_freq* | Frequency integer constant (100\_000\_000) |
| *width* | UART width integer constant (8) |
| *no\_of\_sample* | Number of samples taken during one baud time integer constant (16) |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *rst* | Active high synchronous input |
| *uart\_din* | UART RX input |
| *sw(2:0)* | 3 bits switch bus connected to the switches on the board |
| Output Ports |  |
| *uart\_dout* | UART TX output |
| *sync* | DAC sync control |
| *dac\_data* | DAC data |
| *s\_clk* | DAC clock |
| *led(7:0)* | 8 bit led control bus |

### Waveform Generator Package (wave\_gen\_pkg)

Packages are useful to collect common datas used by many modules. Package usage will increase the readability of the code. When the types, constants, functions etc. defined in the package, just looking at the package file will be enough to find the related data.

In our project a package file will be created named “wave\_gen\_pkg”. Constant arrays of 8-bit ASCII values (for version and baudrate print), component declarations and function definitons will be added to the package. If the data in the package will be used, packages shall be defined in the related vhd file as a library source under the IEEE libraries. Then all the content of the package will be accesible for that vhd file.

### UART Top Module (uart\_top)

Uart\_top module instantiates the following modules inside:

* baudrate\_gen
* u\_tx
* u\_rx

Table 5: UART top module port list

|  |  |
| --- | --- |
| File Name | uart\_top.vhd |
| Entity Name | uart\_top |
| Architecture Name | struct |
| Generics |  |
| *osc\_freq* | Frequency integer constant (100\_000\_000) |
| *width* | UART width integer constant (8) |
| *no\_of\_sample* | Number of samples taken during one baud time integer constant (16) |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *rst* | Active high synchronous input |
| *sw(2:0)* | 3 bits switch bus connected to the switches on the board |
| *rx\_din* | UART RX input pin |
| *tx\_data(7:0)* | UART TX 8 bits sent data |
| *tx\_send* | UART transmitter send pulse |
| Output Ports |  |
| *tx\_dout* | UART TX output |
| *tx\_active* | UART is sending data when this signal is asserted as 1 |
| *rx\_data\_ready* | Informs cmd\_handler that rx\_data is ready |
| *rx\_data(7:0)* | UART RX output. To be connected to cmd\_handler |

#### UART Baudrate Generator Module (baudrate\_gen)

This module sets the baudrate depending on the switches set on the FPGA board. The module generates *no\_of\_sample* baud pulses (one clock cycle) during one baud time.

Table 6: UART baudrate generator module port list

|  |  |
| --- | --- |
| File Name | baudrate\_gen.vhd |
| Entity Name | baudrate\_gen |
| Architecture Name | behav |
| Generics |  |
| *osc\_freq* | Frequency integer constant (100\_000\_000) |
| *no\_of\_sample* | Number of samples taken during one baud time integer constant (16) |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *sw(2:0)* | 3 bits switch bus connected to the switches on the board |
| *rx\_active* | Receive active signal (Receiving data) |
| *tx\_active* | Transmit active signal (Transmitting data) |
| Output Ports |  |
| *baud\_en\_rx* | Baudrate ticks signal for uart receiver block |
| *baud\_en\_tx* | Baudrate tick signal for uart transmitter block |

#### UART Transmit Module (u\_tx)

Table 7: UART transmit module port list

|  |  |
| --- | --- |
| File Name | u\_tx.vhd |
| Entity Name | u\_tx |
| Architecture Name | behav |
| Generics |  |
| *width* | UART width integer constant (8) |
| *no\_of\_sample* | Number of samples taken during one baud time integer constant (16) |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *tx\_send* | Send pulse signal. Asserts 1, UART starts to transmit |
| *data\_in(width-1:0)* | TX (width) bits data input from cmd\_handler |
| *baud\_en\_tx* | The enable signal to take samples |
| Output Ports |  |
| *tx\_data\_out* | One bit serial UART TX data connected to top level output port (tx\_dout) |
| *tx\_active* | UART is sending data when this signal is asserted as 1 |

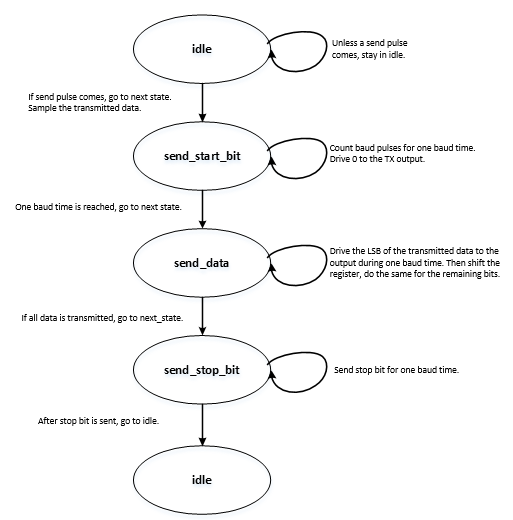


Figure 3: UART transmit module state machine

#### UART Receiver Module (u\_rx)

Table 8: UART receiver module port list

|  |  |
| --- | --- |
| File Name | u\_rx.vhd |
| Entity Name | u\_rx |
| Architecture Name | behav |
| Generics |  |
| *width* | UART width integer constant (8) |
| *no\_of\_sample* | Number of samples taken during one baud time integer constant (16) |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *data\_in* | UART RX input |
| *baud\_en\_rx* | The enable signal to take samples |
| Output Ports |  |
| *rx\_active* | When data is receiving, it is asserted. Connected to the baud generator |
| *data\_out(width-1:0)* | Received data (width bits) connected to cmd\_handler |
| *rx\_data\_ready* | Ready signal which shows data\_out is ready. Connected to cmd\_handler |

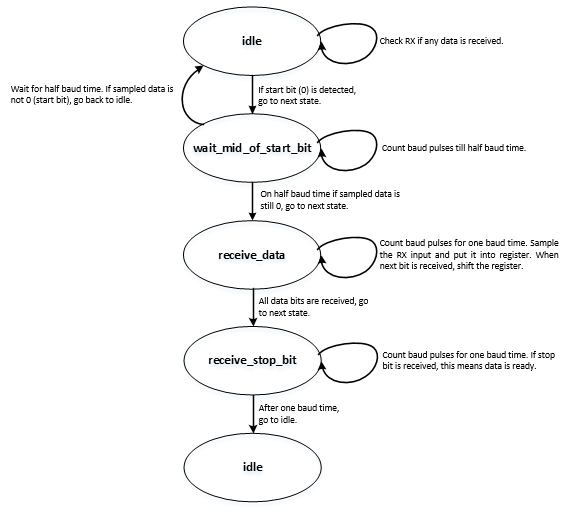


Figure 4: UART receive module state machine

### Command Handler Module (cmd\_handler)

This module decides: validity of the received data, the type of the wave, the frequency of the wave and version or baudrate print request.

Command handler is responsible for detecting the coming commands and giving orders to the other modules. As it is explained in the previous sections, waveform commands are type command and frequency command. A, B and C characters are for wave type. When wave type command is detected, second command shall be frequency command (0 to 9). Otherwise, received data won’t be used.

Also command handler will print a version message and operating baud rate. When a (version) or b (baud rate) char is detected, ASCII values shall be sent to the computer through the UART. The ASCII values can be stored in a constant array.

*Hint: ASCII value for one char is one byte (std\_logic\_vector). There are multiple numbers of characters in a word will be printed. So an array should be defined to keep vectors. Even 3-D arrays can be used (baud rates).*

Table 9: Command hander module port list

|  |  |
| --- | --- |
| File Name | cmd\_handler.vhd |
| Entity Name | cmd\_handler |
| Architecture Name | Behave |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *rst* | Active high synchronous input |
| *rx\_data\_ready* | UART RX data is ready to be used by cmd\_handler |
| *rx\_data(7:0)* | UART RX one byte data |
| *tx\_active* | UART is sending TX data. This signal is active high when UART is transmitting |
| *sw(2:0)* | 3 bits switch bus connected to the switches on the board |
| Output Ports |  |
| *cmd\_ready* | Command received from the UART interface is ready to be used by other blocks |
| *cmd(15:0)* | 16 bits command received from the UART interface |
| *tx\_send* | When active high, informs u\_tx module to sample the data to be sent over the UART interface |
| *tx\_data(7:0)* | One byte TX data to be sent over the UART interface. Connect to u\_tx module |

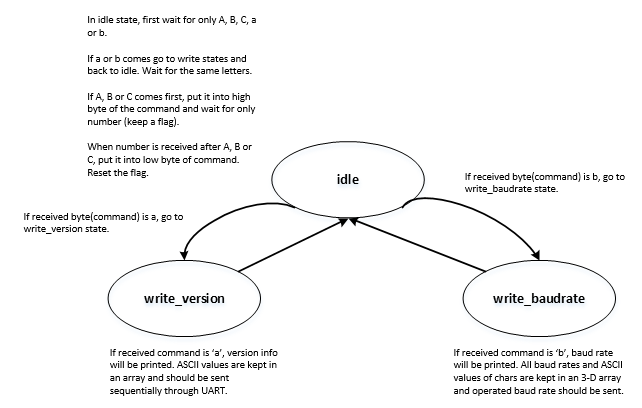


Figure 5: Command handler module state machine

### Enable Generator Module (en\_gen)

This module generates the enable signal according to the frequency setting received from the UART interface. This enable signal will make the counter in the wave\_gen module to count.

Table 10: Enable generator module port list

|  |  |
| --- | --- |
| File Name | en\_gen.vhd |
| Entity Name | en\_gen |
| Architecture Name | Behave |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *rst* | Active high synchronous input |
| *cmd\_rdy* | Command ready signal from cmd\_handler. If set to high, en\_out frequency is started according to the freq input |
| *freq(7:0)* | en\_out frequency value from the UART interface |
| Output Ports |  |
| *en\_out* | Enable output connected to wave\_gen and sync\_data\_gen blocks. Used to set the frequency of the waveform to be generated |

### Waveform Generator Module (wave\_gen)

This module generates the one byte digital waveform.

The DAC takes 8 bit values between 0 and 255. So the wave\_gen output can be controlled by an 8-bit counter. Saw-tooth, triangle and square wave will be obtained by help of counter.

*Hint: For triangle wave, during half period wave output wil be increasing values, the other half wave output will be decreasing values. To increase the amplitude double the output every time.*

Table 11: Waveform generator module port list

|  |  |
| --- | --- |
| File Name | wave\_gen.vhd |
| Entity Name | wave\_gen |
| Architecture Name | Behave |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *rst* | Active high synchronous input |
| *en* | Enable signal from en\_gen module. Used to set the frequency of the waveform |
| *cmd\_rdy* | Command ready signal from cmd\_handler. If set to high, waveform type (wave\_type) is sampled to an internal register |
| *wave\_type(7:0)* | Waveform type value from the UART interface |
| Output Ports |  |
| *wave\_out(7:0)* | One byte waveform output |

### SCLK Generator Module (sclk\_gen)

This module generates the sclk required for the PMOD DAC module external to the FPGA board.

Table 12:SCLK generator module port list

|  |  |
| --- | --- |
| File Name | sclk\_gen.vhd |
| Entity Name | sclk\_gen |
| Architecture Name | Behave |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *rst* | Active high synchronous input |
| Output Ports |  |
| *sclk* | 25 MHz clock output. Clk divided by 4 |
| *edge\_low* | Connected to sync\_data\_gen block. Needed to meet specifications of PMOD DAC |

### Synchronous Data Generator Module (sync\_data\_gen)

This module will generate SYNC (active low) signal. Addition to SYNC Signal, it will generate 16 Bit data (dac\_data) for the shift register of DA1 module. The structure of 16 Bit data is shown in Figure 6.

Table 13: Synchronous data generator module port list

|  |  |
| --- | --- |
| File Name | sync\_data\_gen.vhd |
| Entity Name | sync\_data\_gen |
| Architecture Name | Behave |
| Input Ports |  |
| *clk* | 100 MHz Clock |
| *rst* | Active high synchronous input |
| *en* | Enable signal from en\_gen module. Used to set the frequency of the waveform |
| *edge\_low* | Needed to meet specifications of PMOD DAC |
| *wave(7:0)* | One byte waveform data from the wave\_gen module |
| Output Ports |  |
| *sync* | Sync output to be connected to external PMOD DA1 module |
| *dac\_data* | Data output to be connected to external PMOD DA1 module |

The Pmod DA1 is an 8-bit Digital-to-Analog Converter powered by two [Analog Devices AD7303](http://www.analog.com/AD7303). With simultaneous dual outputs per single input, users can perform up to four isolated tests on the analog outputs. Users may provide data to the module through an interface like SPI.

* Two 8-bit DACs
* Four D/A conversion channels
* Maximum conversion rate of 1.875 MSa
* Small PCB size for flexible designs 1.0“ × 0.8” (2.5 cm × 2.0 cm)
* 6-pin Pmod connector with GPIO interface
* Library and example code available in [resource center](https://reference.digilentinc.com/reference/pmod/pmodda1/start#example_projects)

# Communication between FPGA and PMOD DA1

Three signals (SCLK, SYNC Active Low, DIN) are used to implement communication between FPGA and PMOD DA1.

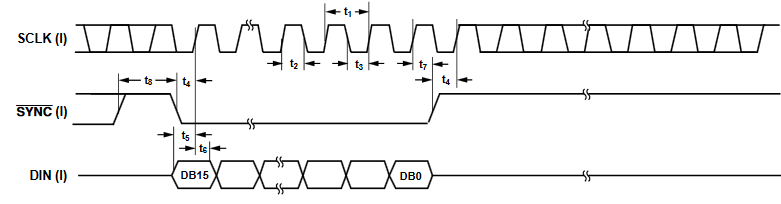


Figure 6: Timing Diagram of PMOD DA1 Interface

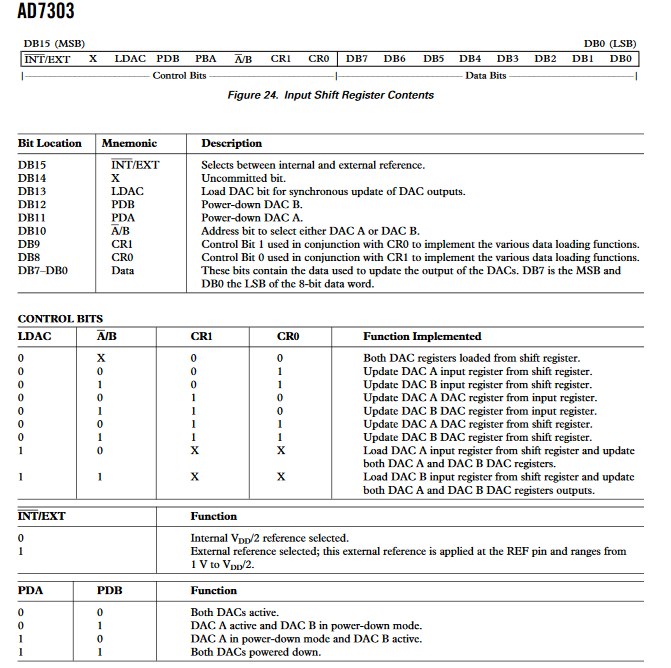


Figure 7: DA1 16 Bits Data Structure

Control bits (DB15 to DB8) shown in Figure 7, will be taken as constants during the VHDL coding as given in Table 14.

Table 14: Control Bit Values for AD7303

|  |  |  |
| --- | --- | --- |
| Control Bit # | Control Bit Field | Contant Value |
| DB15 | INT/EXT | 0 |
| DB14 | X | 0 |
| DB13 | LDAC | 1 |
| DB12 | PDB | 1 |
| DB11 | PDA | 0 |
| DB10 | A\_B | 0 |
| DB9 | CR1 | 0 |
| DB8 | CR0 | 0 |