

# PicoTerm

By Vikram & Chethan

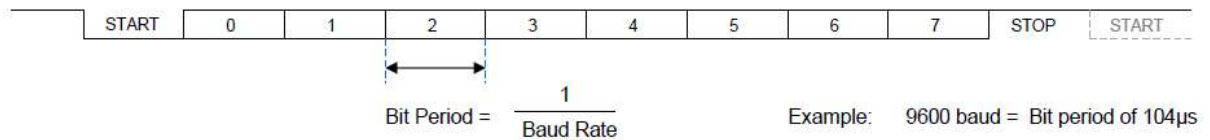
Advisor: Prof. Gandhi Puvvada

# **CONTENTS**

- Introduction to PicoTerm
- Information on the files provided
- Getting Started with PicoTerm
- Device Control Strings for Communication
- Significant Features Supported
- Closing PicoTerm

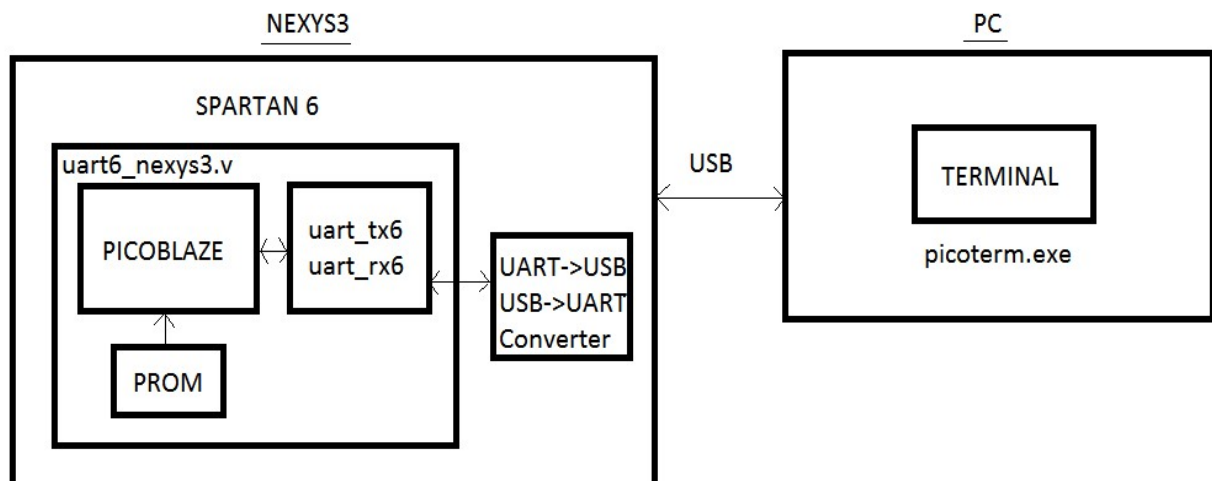
## ➤ Introduction to PicoTerm

PicoTerm is primarily a simple PC based terminal ideally suited for communication with PicoBlaze based designs. It utilizes the UART macros connected to a USB/UART port on a development board or evaluation Kit. PicoTerm has the communication fixed to 1 Start Bit, 8 character bits, 1 Stop Bit, No Parity and No Handshake which is compatible with the UART macros provided with PicoBlaze. This means that the only two variables are the number of the COM port and the BAUD rate.



However, PicoTerm has been pre-configured to match with the parameters required for a PicoBlaze/UART designs and has a default BAUD rate of 115200. So only the COM port needs to be specified to communicate with the design. Additionally, since it can be used to communicate with any 'COM' port, it can be connected to virtually any hardware and design. The KCPSM6 processor interfaces to the UART6 receiver and transmitter macros using the same input and output ports. In short PicoTerm provides quick and reliable way to establish a working connection with a PicoBlaze based UART design.

The application (e.g. PicoBlaze) can control the main PicoTerm terminal window and to open and control a variety of independent features such as virtual LEDs and Switches. A selection of 'Escape Sequences' and 'Device Control Strings' (DCS) enable the application to exert control over PicoTerm features.



## ➤ Information on the files provided

```
uart6_nexys3.v
Nexys3_master.ucf
kcpsm6.v
nexys_picoterm.psm
    PicoTerm_routines.psm
    soft_delays_100mhz.psm
uart_rx6.v
uart_tx6.v
```

### ➔ 'uart6\_nexys.v' and 'uart6\_nexys.ucf'

Hardware design is defined here. These includes additional ports which interface with 8 switches and 8 LEDs on the NEXYS board.

### ➔ 'nexys\_picoterm.psm'

This is the top level KCPSM6 Processor Program code for 'uart6\_nexys.v'. This provides multiple functionalities. It also have INCLUDE directives to include 'PicoTerm\_routines.psm' and 'soft\_delays\_100mhz.psm'.

#### ○ 'PicoTerm\_routines.psm'

This file provided with PicoTerm contains a set of KCPSM6 routines, UART interface definition and UART routines including routines written specifically to use PicoTerm's special features.

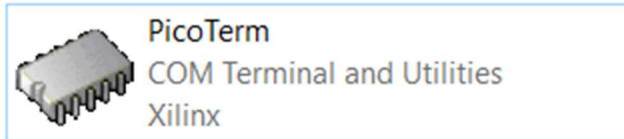
#### ○ 'soft\_delays\_100mhz.psm'

This provides Routines implementing delays based on a 100MHz clock which is used by all logic. Note that the baud rate and software delays in PSM code are defined relative to 100MHz.

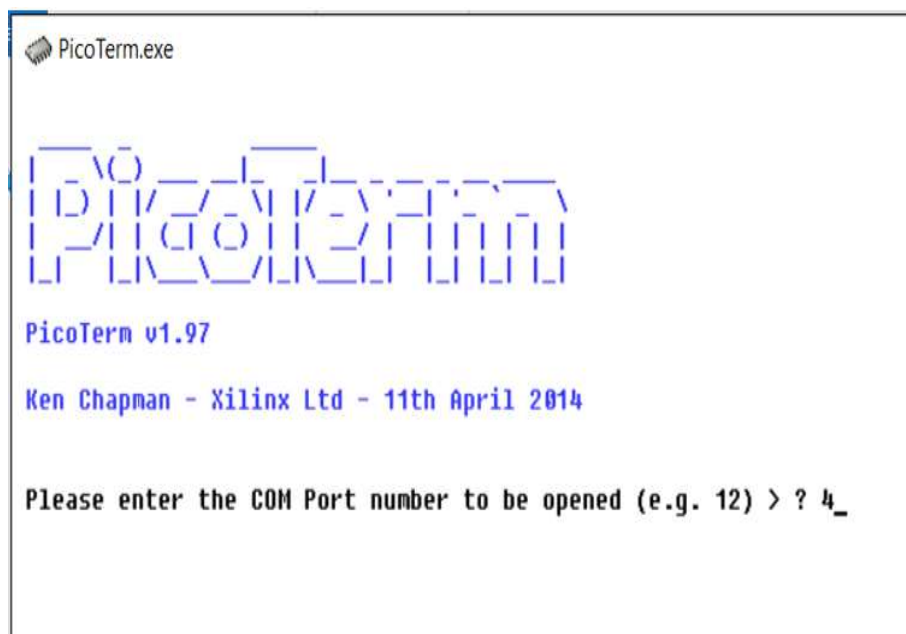
The PSM codes (KCPSM6 programs) illustrate UART macro interaction and communication with terminal (keyboard and display). You must assemble the top level PSM file 'nexys\_real\_time\_clock.psm' to generate the program definition files and then implement the design to configure the device.

## ➤ Getting Started with PicoTerm

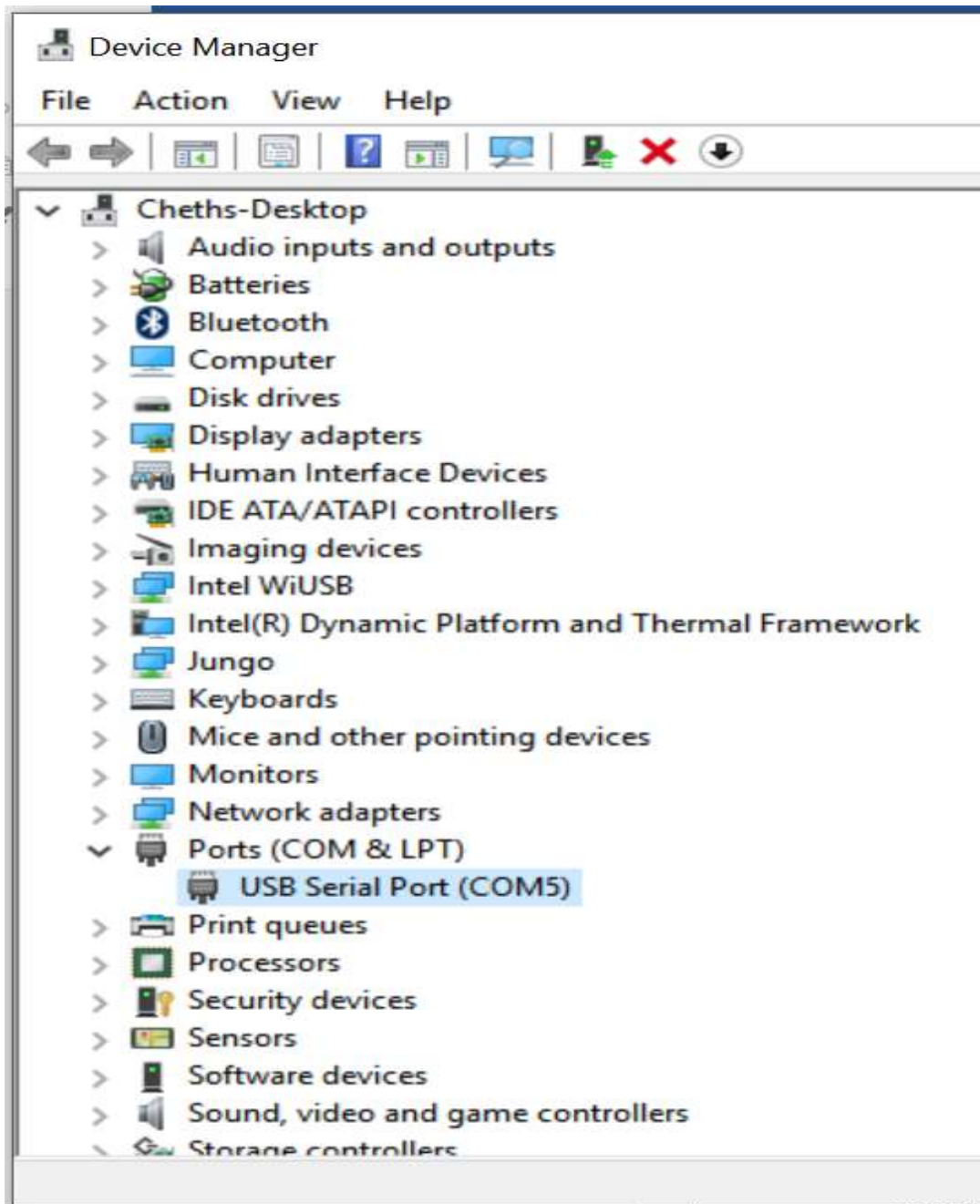
➔ Open the executable file provided in the folder.



➔ It opens the below Window. Enter the COM port and press enter to connect to the board and get started.



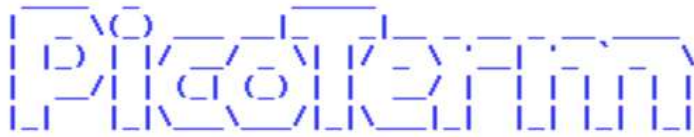
➔ How to know which COM port to connect:  
Open Device Manager  
Search for COM



COM Port number is 5

- ➔ Having entered a COM port number into PicoTerm, it will attempt to open that port. If it is unable to open that port then it will tell you. It may be that you specified an invalid port number but you should also remember to check that you have no other applications open that are already accessing the same port before trying again. When a COM port is opened successfully then a message like the following will be displayed.

 PicoTerm.exe



PicoTerm v1.97

Ken Chapman - Xilinx Ltd - 11th April 2014

Please enter the COM Port number to be opened (e.g. 12) > ? 5

COM5 is open for communication at 115200 baud.

➔ Baud Rate need not be changed as it is pre-configured to 115200.

### ➤ Device Control Strings (DCS) for Communication

PicoTerm implements 'Device Control Strings' (DCS) to enable communication with PicoBlaze applications. When PicoTerm receives one of the DCS sequences then it will perform a special operation as per the received request. Some DCS commands will result in PicoTerm responding with another Device Control String containing appropriate information such as the time on the PC whilst others are used to open and control separate windows such as the virtual 7-Segment digits display.

When a DCS is used to facilitate the transfer of information between PicoBlaze (or similar) and the PC (e.g. a request for time) then a 'PicoTerm DCS Transactions' window will automatically open and display a message confirming the request and information exchanged. The contents of a Device Control String may contain bytes of any value (i.e. data in the range 00 to FF hex). The following characters that begin and end all 'Device Control Strings' have codes that are also beyond the normal 7-bit ASCII range.

'DCS' = 'Device Control String' character (90 hex = 144).

'ST' = 'String Terminator' character (9C hex = 156).

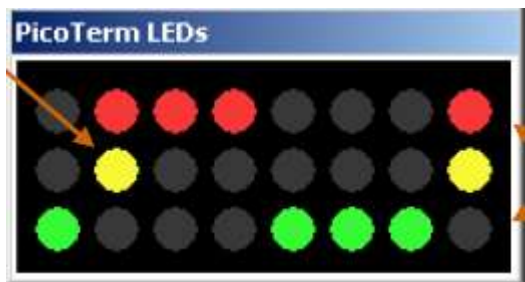
DCS and ST are predefined constants in the KCPSM6 assembler. When PicoTerm responds with a Device Control String it always starts with the same character that was used to make the request. For information on UART TX and RX refer to [UART6\\_User\\_Guide\\_and\\_Reference\\_Designs\\_30Sept14.pdf](#) provided in the folder.

## ➤ Significant Features supported by PicoTerm

The 'nexys\_real\_time\_clock.psm' program displays text and interacts with user keyboard entries. The hardware design includes additional ports which interface with 8 switches and 8 LEDs on the NEXYS3 board. The program reads the real switches on the NEXYS3 board and sets the virtual LEDs within PicoTerm and reads the virtual switches within PicoTerm and drives the real LEDs on the NEXYS board. The program also has an interrupt service routine (ISR) that responds to the interrupts generated at one second intervals to output the virtual switch data on the virtual 7 segment display to demonstrate communication between PicoTerm to Board and back to PicoTerm.

### ➔ Virtual LED Display

The PicoTerm Virtual LED Display is a pop-up window containing 24 virtual LEDs. There are 8 red, 8 yellow (amber) and 8 green LEDs arranged in 3 rows as shown below. The physical switches are read and a DCS transmitted to set the Red/Green virtual LEDs.



The virtual display is opened and updated using a 'Device Control String' (DCS) sequence. When PicoTerm receives the first virtual LED display DCS sequence it will open the virtual LED window with the specified LEDs 'turned on'. Subsequent virtual LED display DCS sequences will modify the LEDs to reflect the new control values provided. Note that PicoTerm does not transmit a DCS sequence back to the COM port.

The DCS sequence for the virtual LED display is as follows

'DCS' (90 hex = 144)

'L' (4C hex = 76 )

RED\_control\_byte

YELLOW\_control\_byte



GREEN\_control\_byte

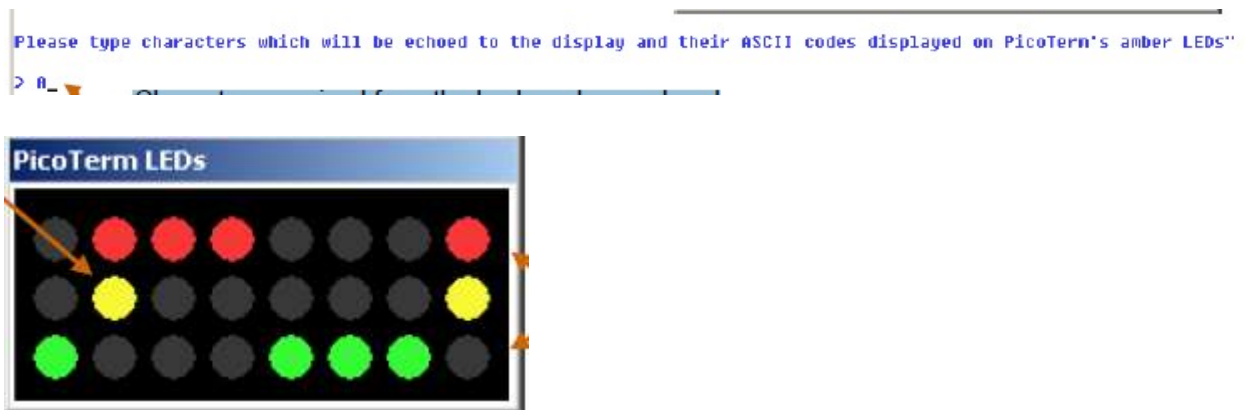
'ST' (9C hex = 156)

The virtual LEDs of each colour are controlled by the corresponding bits contained in each of control bytes. For example the least significant bit of 'GREEN\_control\_byte' will control the virtual LED in the lower right hand corner of the display.

Switches on NEXYS board -> Input Port -> KCPSM6 -> UART -> PicoTerm virtual LEDs (Red/Green)

### → Echoing of Keyboard Characters

Characters received from the keyboard are echoed back to the main screen. The ASCII code is also displayed on the amber virtual LEDs by transmitting a Device Control String (DCS)

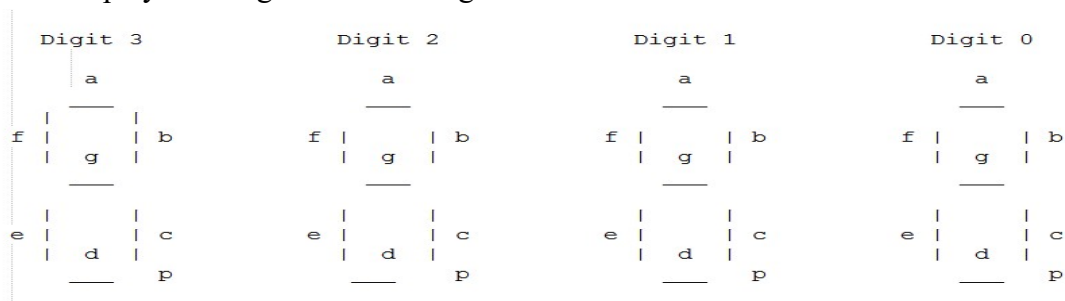


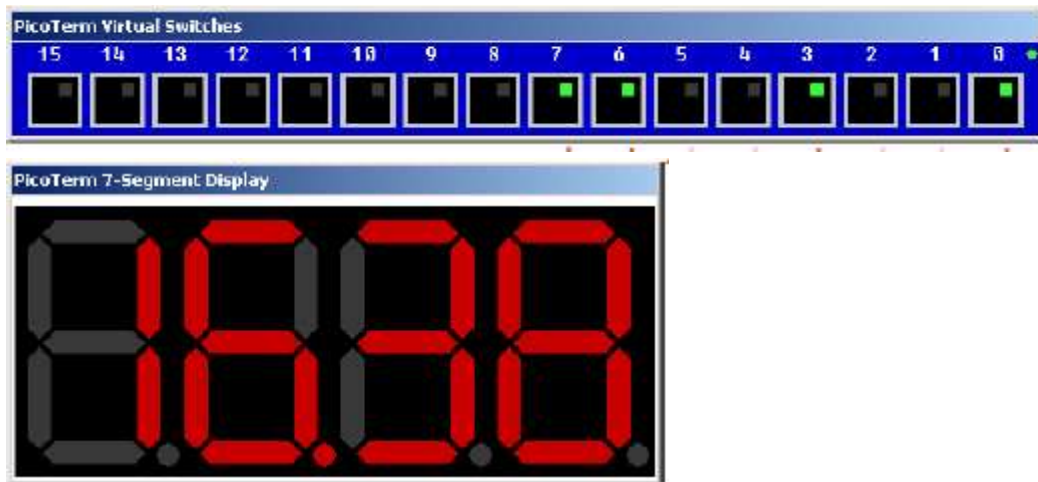
PicoTerm Keyboard -> UART -> KCPSM6 -> UART -> PicoTerm main display

PicoTerm Keyboard -> UART -> KCPSM6-> UART -> PicoTerm virtual LEDs (Amber)

### → Virtual 7-Segment Display to output Virtual Switch Data

The PicoTerm Virtual 7-Segment Display is a pop-up window containing a virtual 4-digit, 7-segment display. The digits and their segments are identified below.





The virtual display is opened and updated using a 'Device Control String' (DCS) sequence. When PicoTerm receives the first virtual display DCS sequence it will open the window and display the digits with the specified segments 'turned on'. Subsequent virtual display DCS sequences will modify the display to reflect the new control values provided. Note that PicoTerm does not transmit a DCS sequence back to the COM port.

The DCS sequence for the virtual 7-Segment display is as follows

'DCS' (90 hex = 144)

'7' (37 hex = 55 )

digit0\_segment\_control\_byte

digit1\_segment\_control\_byte

digit2\_segment\_control\_byte

digit3\_segment\_control\_byte

'ST' (9C hex = 156)

The segments of each digit are controlled by the bits contained in the control bytes. Each digit has 7 segments and a decimal point and a segment will be 'turned on' when the corresponding bit of the control byte is High (1).

Segment Bit

- a 0
- b 1
- c 2
- d 3
- e 4
- f 5
- g 6
- p 7 decimal point

When PicoTerm sends DCS containing the states of the virtual switches the UART\_RX routine will intercept the string and store it in scratch pad memory starting at location 'PicoTerm\_Response0'. So to display it on 7 segment we fetch the data that is stored in scratch pad memory and send it to virtual 7 segment on PicoTerm.

1-second interrupt -> KCPSM6 -> UART -> PicoTerm virtual 7-Segment display

## ➔ Virtual Switches

PicoTerm Virtual Switches is a pop-up window containing 16 virtual switches. Each switch has the appearance of a square black button with an embedded green LED. Clicking on a virtual button will toggle the state of the switch and the virtual LED will indicate the current state, i.e. LED on means switch is turned on ('1').



There are two 'Device Control String' (DCS) sequences that can be used in conjunction with the virtual switches. The first time either sequence is received by PicoTerm the Virtual Switches display will be opened. The first and most useful DCS sequence is used to read the current states of switches. If this is also used to open the window, then all 16 switches will initially be off ('0').

Request to PicoTerm to read virtual switches

'DCS' (90 hex = 144)

'S' (53 hex = 83 ) upper case 'S'

'ST' (9C hex = 156)

In response to each use of the above DCS sequence, PicoTerm will respond with the following DCS sequence reporting the current states of all 16 switches.

PicoTerm response

'DCS'

'S'

switches(0) (current states of switches[7:0])

switches(1) (current states of switches[15:8])

'ST'

Note that whilst the effect of clicking on a switch is immediately reflected by its indicator LED, PicoTerm will only generate a DCS sequence in response to a DCS request. Hence, PicoBlaze must issue a DCS request in order to determine the current states of the switches (this is similar to the way in which PicoBlaze would need to read an input port to determine the current states of physical switches). KCPSM6 transmits a DCS requesting the status of the virtual switches and then receives a DCS response with which it drives the physical LEDs.

PicoTerm Virtual Switches -> UART -> KCPSM6 -> Output port -> LEDs on Nexys board

For Graphic Display and other supporting features refer to PicoTerm\_README.txt provided in the folder.

#### ➤ Closing PicoTerm

To close PicoTerm press the 'Esc' key or close the window. Although no issues have been encountered when simply closing the PicoTerm window, using the 'Esc' key is preferred as it does result in a definitive closing of the COM port at the end of the session.