ADS Lab 3 Report

Eva Maria Deltor Cortés and Pau Jordan Oliveras Cejas

# VHDL Code

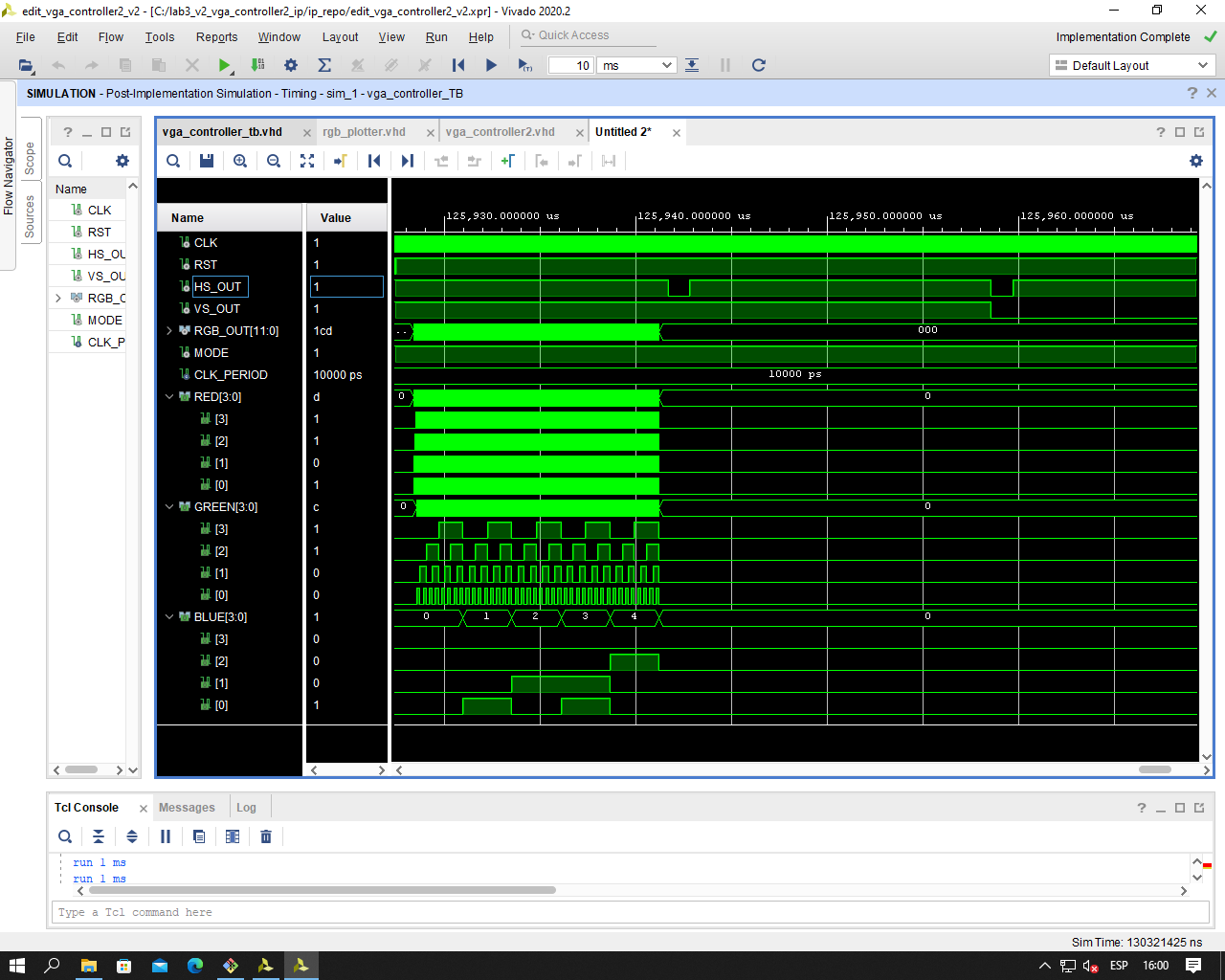
## Vga\_controller entity

|  |
| --- |
| ----------------------------------------------------------------------------------  -- Company: UPC-ADS  -- Engineer: Pau Jordan Oliveras, Eva Maria Deltor  --  -- Create Date: 10/14/2021 05:48:35 PM  -- Design Name:  -- Module Name: vga\_controller - vga\_controller\_arc  -- Revision:  -- Revision 0.01 - File Created  ----------------------------------------------------------------------------------  **library** IEEE**;**  **use** IEEE**.**STD\_LOGIC\_1164**.ALL;**  **use** IEEE**.**NUMERIC\_STD**.ALL;**  **entity** vga\_controller **is**  **generic** **(** -- We define the constants of the component  h\_pixels **:** integer **:=** 1688**;** -- Total number of hortizontal pixels (visible and non visible) - total number of pixel clock cycles  h\_sync **:** integer **:=** 112**;** -- Sync pulse width in clock cycles  h\_start\_pixel **:** integer **:=** 360**;** --pixel in which the visible part starts  h\_end\_pixel **:** integer **:=** 1640**;** -- first pixel out of the visible region  v\_lines **:** integer **:=** 1066**;** --Total number of lines in a frame (visible and non visible)  v\_sync **:** integer **:=** 3**;** --Sync vertical pulse width in lines  v\_start\_line **:** integer **:=** 41**;** -- line in which the visible part starts  v\_end\_line **:** integer **:=** 1065**;** -- first line out of the visible region  h\_bits **:** integer **:=** 11**;** -- N bits horizontal counter (max util 1688)  v\_bits **:** integer **:=** 11**);** -- N bits vertical counter (max util 1066)    **Port** **(**  mode **:** **in** std\_logic**;**  heartbeat **:** **out** STD\_LOGIC**;** -- this signal is not in the assigment but has been added to know if the clock is running.  clk **:** **in** STD\_LOGIC**;**  resetn **:** **in** STD\_LOGIC**;**  vsync **:** **out** STD\_LOGIC**;**  hsync **:** **out** STD\_LOGIC**;**  PIXEL\_X **:** **out** unsigned**(**h\_bits **downto** 0**);** -- Numero de Pixel X  PIXEL\_Y **:** **out** unsigned**(**v\_bits **downto** 0**);** -- Numero de Pixel Y  red **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**  green **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**  blue **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**));**  **end** vga\_controller**;**  **architecture** vga\_controller\_arc **of** vga\_controller **is**  **signal** c\_x **:** integer **range** **(**2**\*\*(**h\_bits**)-**1**)** **downto** 0**;** -- signal that counts the hortizontal pixels  **signal** c\_y **:** integer **range** **(**2**\*\*(**v\_bits**)-**1**)** **downto** 0**;** -- signal that counts the lines  **signal** HS\_s**,** VS\_s**,** DISPLAY\_E\_s**:** std\_logic**;** -- signal for the vga sync signals, and signal to indicate current pixel is active.  **signal** heartbeat\_counter **:** integer **range** **(**2**\*\***26**)** **downto** 0**;** --counter that overflows at 2 Hz (as was used by the led in the board, the frequency should be seen by the human eye)  **signal** pixel\_x\_s **:** unsigned**(**h\_bits **downto** 0**);** --signal that coutns the visible x axis pixels  **signal** pixel\_y\_s **:** unsigned**(**v\_bits **downto** 0**);** -- signal that counts the visible y axis lines  **component** rgb\_plotter -- component defined in the rgb\_plotter.vhd file, generates the color patterns given the pixel counters and the enable signal (generated in this current vhd)  **generic** **(**  --colors definitions  --horizontal: if we know that we have a horitzontal display area of 1024 - it will be divided by 3 in 342  -- Here we define all the constants of the component RGB plotter  horizontal\_red\_fin**:** integer **:=** 342**;** --ends red line  horizontal\_green\_fin**:** integer **:=** 682**;**-- ends the green line  horizontal\_blue\_fin**:** integer **:=** 1023**);**-- ends the blue line    **Port** **(** ENABLE **:** **in** STD\_LOGIC**;**  MODE **:** **in** STD\_LOGIC**;**  PIXEL\_X **:** **in** STD\_LOGIC\_VECTOR **(**11 **downto** 0**);**  PIXEL\_Y **:** **in** STD\_LOGIC\_VECTOR **(**11 **downto** 0**);**  RED **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**  GREEN **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**  BLUE **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**));**  **end** **component;**    **begin**  rgb\_plotter1 **:** rgb\_plotter --we include an instance of the rgb plotter component.  **Port** **map(**  ENABLE **=>** DISPLAY\_E\_s**,**  MODE **=>** MODE**,**  PIXEL\_X **=>** std\_logic\_vector**(**pixel\_x\_s**),**  PIXEL\_Y **=>** std\_logic\_vector**(**pixel\_y\_s**),**  RED **=>** red**,**  GREEN **=>** green**,**  BLUE **=>** blue  **);**  **process(**CLK**)**  **begin**  **if(**CLK **=** '1' **and** CLK'**event)** **then** -- CLK  **if(**resetn **=** '0'**)** **then** --we define all the counters to 0 once the reset is activated  heartbeat\_counter **<=** 0**;**  c\_x **<=** 0**;**  c\_y **<=** 0**;**  HS\_s **<=** '0'**;**  VS\_s **<=** '0'**;**  **else** -- the counter is incremented each clock cycle, when it reaches 33554432 the led is tuned off, else it is turned on.  -- When the counter overflows it is set to zero.  -- We have defined this procedure as we needed to have a visual clue that the system is working, without the screen.  **if(**heartbeat\_counter **>** 2**\*\***25**)** **then**  heartbeat **<=** '0'**;**  **else**  heartbeat **<=** '1'**;**  **end** **if;**    **if(**heartbeat\_counter **>=** **(**2**\*\***26 **-** 1**))** **then**  heartbeat\_counter **<=** 0**;**  **else**  heartbeat\_counter **<=** heartbeat\_counter **+** 1**;**  **end** **if;**      **if(**c\_x **>=** **(**h\_pixels**-**1**))** **then** -- If we have finished the line  c\_x **<=** 0**;** -- Reset the horiztonal counter and check the vertical one  HS\_s **<=** '0'**;** -- Hortizonal sync pulse  **if(**c\_y **>=** **(**v\_lines**-**1**))** **then** -- If the frame has ended  c\_y **<=** 0**;** -- Reset the vertical counter  VS\_s **<=** '0'**;** -- Vertical sync pulse  **else**  **if** **(**c\_y **>=** **(**v\_sync **-** 1**))** **then** -- If it is no longer a vertical pulse line, pulse ends.  VS\_s **<=** '1'**;**  **end** **if;**  c\_y **<=** c\_y **+** 1**;** -- Increase vertical counter  **end** **if;**  **else**  **if(**c\_x **>=** **(**h\_sync **-** 1**))** **then** -- If it is no longer a hortizontal pulse line, pulse ends.  HS\_s **<=** '1'**;**  **end** **if;**  **if(**c\_x **>=** h\_start\_pixel**-**1 **and** c\_x **<** h\_end\_pixel**-**1 **and** c\_y **>=** v\_start\_line **and** c\_y **<** v\_end\_line**)** **then**  DISPLAY\_E\_s **<=** '1'**;** -- If the selected pixel is inside the display area  **else**  DISPLAY\_E\_s **<=** '0'**;** -- If the selected pixel is not inside the display area  **end** **if;**  c\_x **<=** c\_x **+** 1**;** --Increment the horizontal counter  **end** **if;**  **end** **if;**  **end** **if;**  **end** **process;**      -- We assign the internal signals to the ports  hsync **<=** HS\_s**;**  vsync **<=** VS\_s**;**  PIXEL\_X **<=** pixel\_x\_s**;**  PIXEL\_Y **<=** pixel\_y\_s**;**  pixel\_x\_s **<=** **to\_unsigned(**c\_x **-** h\_start\_pixel**,** pixel\_x\_s'**length);** -- n pixel X es comptador horitzontal - primer pixel  pixel\_y\_s **<=** **to\_unsigned(**c\_y **-** v\_start\_line**,** pixel\_y\_s'**length);** -- n linia Y es comptador vertical - primera linia    **end** vga\_controller\_arc**;** |

## Rgb\_plotter entity

|  |
| --- |
| ----------------------------------------------------------------------------------  -- Company: ADS  -- Engineer: Pau Jordan Oliveres, Eva María Deltor  --  -- Create Date: 11/03/2021 06:05:29 PM  -- Design Name:  -- Module Name: rgb\_plotter - rgb\_plotter\_arc  --This component generates two color patterns based on the mode. The enable signal enables or disables the output and the pixel\_x and pixel\_y inputs are used to generate the pattern.  --It is a fully combinational component.  -- Revision:  -- Revision 0.01 - File Created  -- Additional Comments:  --  ----------------------------------------------------------------------------------  **library** IEEE**;**  **use** IEEE**.**STD\_LOGIC\_1164**.ALL;**  **use** IEEE**.**NUMERIC\_STD**.ALL;**  **entity** rgb\_plotter **is**  **generic** **(**  -- Here we define all the constants of the component RGB plotter  horizontal\_red\_fin**:** integer **:=** 342**;** --ends red line  horizontal\_green\_fin**:** integer **:=** 683**;**-- ends the green line  horizontal\_blue\_fin**:** integer **:=** 1023**);**-- ends the blue line      **Port** **(** ENABLE **:** **in** STD\_LOGIC**;**  MODE **:** **in** STD\_LOGIC**;**  PIXEL\_X **:** **in** STD\_LOGIC\_VECTOR **(**11 **downto** 0**);**  PIXEL\_Y **:** **in** STD\_LOGIC\_VECTOR **(**11 **downto** 0**);**  RED **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**  GREEN **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**);**  BLUE **:** **out** STD\_LOGIC\_VECTOR **(**3 **downto** 0**));**  **end** rgb\_plotter**;**  **architecture** rgb\_plotter\_arc **of** rgb\_plotter **is**  **signal** pixel\_x\_s **:** integer**;** -- this signal counts which pixel in the x axis is currently in we only take into account the visible ones  **signal** pixel\_y\_s **:** integer**;** -- this signal counts which pixel in the y axis is currently in we only take into account the visible ones  **begin**  pixel\_x\_s **<=** **TO\_INTEGER(**unsigned**(**PIXEL\_X**));** -- the input port is an unsigned type, our signal is an integrer type  pixel\_y\_s **<=** **TO\_INTEGER(**unsigned**(**PIXEL\_Y**));** -- the input port is an unsigned type, our signal is an integrer type    **process(**ENABLE**,** MODE**,** PIXEL\_X**,** PIXEL\_Y**)** -- if any of this signals changes the process needs to be run  **begin**  **if** **(**ENABLE **=** '0'**)** **then** -- when the enable is zero; the output (color of the screen) is black.  RED **<=** x"0"**;**  GREEN **<=** x"0"**;**  BLUE **<=** x"0"**;**  **else**  **if(** MODE **=** '0'**)then** -- hortizontal stripes when mode = 0  -- we compare pixel y against the constants defined at the beining of the file, which define the bounds of each color stripe  **if** **(**pixel\_y\_s **<=** horizontal\_red\_fin**)** **then**  RED **<=** x"F"**;** -- red all in 1  GREEN **<=** x"0"**;** -- green all in 0  BLUE **<=** x"0"**;** -- blue all in 0  -- so in this case will be red.  --Same procedure in the following situations.  **elsif** **(**pixel\_y\_s **<=** horizontal\_green\_fin**)** **then**  RED **<=** x"0"**;**  GREEN **<=** x"F"**;**  BLUE **<=** x"0"**;**  **elsif** **(**pixel\_y\_s **<=** horizontal\_blue\_fin**)** **then**  RED **<=** x"0"**;**  GREEN **<=** x"0"**;**  BLUE **<=** x"F"**;**  **else** -- In case of not being in any of the previous situations (error situation) the screen will be in white.  RED **<=** x"F"**;**  GREEN **<=** x"F"**;**  BLUE **<=** x"F"**;**  **end** **if;**  **else** -- if we are in mode 1; we simply assign the different color values some bits of the PIXEL\_X counter, resulting in a color pattern.  RED **<=** PIXEL\_X**(**3 **downto** 0**);** --changes fast as represents the LSB  GREEN **<=** PIXEL\_X**(**7 **downto** 4**);**--changes slower than red, as we can have more than one color cycle  BLUE **<=** '0' **&** PIXEL\_X**(**10 **downto** 8**);** -- the slowest change; MSB    **end** **if;**  **end** **if;**  **end** **process;**  **end** rgb\_plotter\_arc**;** |

# Post-implementation simulation



# Timing report

|  |
| --- |
| Copyright 1986-2020 Xilinx, Inc. All Rights Reserved.  -----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------  | Tool Version : Vivado v.2020.2 (win64) Build 3064766 Wed Nov 18 09:12:45 MST 2020  | Date : Mon Dec 13 16:23:09 2021  | Host : c5b1 running 64-bit major release (build 9200)  | Command : report\_timing\_summary -max\_paths 10 -file vga\_controller2\_v2\_timing\_summary\_routed.rpt -pb vga\_controller2\_v2\_timing\_summary\_routed.pb -rpx vga\_controller2\_v2\_timing\_summary\_routed.rpx -warn\_on\_violation  | Design : vga\_controller2\_v2  | Device : 7z020-clg484  | Speed File : -1 PRODUCTION 1.12 2019-11-22  -----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------  Timing Summary Report  ------------------------------------------------------------------------------------------------  | Timer Settings  | --------------  ------------------------------------------------------------------------------------------------  Enable Multi Corner Analysis : Yes  Enable Pessimism Removal : Yes  Pessimism Removal Resolution : Nearest Common Node  Enable Input Delay Default Clock : No  Enable Preset / Clear Arcs : No  Disable Flight Delays : No  Ignore I/O Paths : No  Timing Early Launch at Borrowing Latches : No  Borrow Time for Max Delay Exceptions : Yes  Merge Timing Exceptions : Yes  Corner Analyze Analyze  Name Max Paths Min Paths  ------ --------- ---------  Slow Yes Yes  Fast Yes Yes  check\_timing report  Table of Contents  -----------------  1. checking no\_clock (0)  2. checking constant\_clock (0)  3. checking pulse\_width\_clock (0)  4. checking unconstrained\_internal\_endpoints (0)  5. checking no\_input\_delay (46)  6. checking no\_output\_delay (52)  7. checking multiple\_clock (0)  8. checking generated\_clocks (0)  9. checking loops (0)  10. checking partial\_input\_delay (0)  11. checking partial\_output\_delay (0)  12. checking latch\_loops (0)  1. checking no\_clock (0)  ------------------------  There are 0 register/latch pins with no clock.  2. checking constant\_clock (0)  ------------------------------  There are 0 register/latch pins with constant\_clock.  3. checking pulse\_width\_clock (0)  ---------------------------------  There are 0 register/latch pins which need pulse\_width check  4. checking unconstrained\_internal\_endpoints (0)  ------------------------------------------------  There are 0 pins that are not constrained for maximum delay.  There are 0 pins that are not constrained for maximum delay due to constant clock.  5. checking no\_input\_delay (46)  -------------------------------  There are 46 input ports with no input delay specified. (HIGH)  There are 0 input ports with no input delay but user has a false path constraint.  6. checking no\_output\_delay (52)  --------------------------------  There are 52 ports with no output delay specified. (HIGH)  There are 0 ports with no output delay but user has a false path constraint  There are 0 ports with no output delay but with a timing clock defined on it or propagating through it  7. checking multiple\_clock (0)  ------------------------------  There are 0 register/latch pins with multiple clocks.  8. checking generated\_clocks (0)  --------------------------------  There are 0 generated clocks that are not connected to a clock source.  9. checking loops (0)  ---------------------  There are 0 combinational loops in the design.  10. checking partial\_input\_delay (0)  ------------------------------------  There are 0 input ports with partial input delay specified.  11. checking partial\_output\_delay (0)  -------------------------------------  There are 0 ports with partial output delay specified.  12. checking latch\_loops (0)  ----------------------------  There are 0 combinational latch loops in the design through latch input  ------------------------------------------------------------------------------------------------  | Design Timing Summary  | ---------------------  ------------------------------------------------------------------------------------------------  WNS(ns) TNS(ns) TNS Failing Endpoints TNS Total Endpoints WHS(ns) THS(ns) THS Failing Endpoints THS Total Endpoints WPWS(ns) TPWS(ns) TPWS Failing Endpoints TPWS Total Endpoints  ------- ------- --------------------- ------------------- ------- ------- --------------------- ------------------- -------- -------- ---------------------- --------------------  4.437 0.000 0 319 0.106 0.000 0 319 4.129 0.000 0 224  All user specified timing constraints are met.  ------------------------------------------------------------------------------------------------  | Clock Summary  | -------------  ------------------------------------------------------------------------------------------------  Clock Waveform(ns) Period(ns) Frequency(MHz)  ----- ------------ ---------- --------------  axi\_clock {0.000 4.630} 9.259 108.003  ------------------------------------------------------------------------------------------------  | Intra Clock Table  | -----------------  ------------------------------------------------------------------------------------------------  Clock WNS(ns) TNS(ns) TNS Failing Endpoints TNS Total Endpoints WHS(ns) THS(ns) THS Failing Endpoints THS Total Endpoints WPWS(ns) TPWS(ns) TPWS Failing Endpoints TPWS Total Endpoints  ----- ------- ------- --------------------- ------------------- ------- ------- --------------------- ------------------- -------- -------- ---------------------- --------------------  axi\_clock 4.437 0.000 0 319 0.106 0.000 0 319 4.129 0.000 0 224  ------------------------------------------------------------------------------------------------  | Inter Clock Table  | -----------------  ------------------------------------------------------------------------------------------------  From Clock To Clock WNS(ns) TNS(ns) TNS Failing Endpoints TNS Total Endpoints WHS(ns) THS(ns) THS Failing Endpoints THS Total Endpoints  ---------- -------- ------- ------- --------------------- ------------------- ------- ------- --------------------- -------------------  ------------------------------------------------------------------------------------------------  | Other Path Groups Table  | -----------------------  ------------------------------------------------------------------------------------------------  Path Group From Clock To Clock WNS(ns) TNS(ns) TNS Failing Endpoints TNS Total Endpoints WHS(ns) THS(ns) THS Failing Endpoints THS Total Endpoints  ---------- ---------- -------- ------- ------- --------------------- ------------------- ------- ------- --------------------- ------------------- |

# Area report

|  |
| --- |
| Copyright 1986-2020 Xilinx, Inc. All Rights Reserved.  -----------------------------------------------------------------------------------------------------------  | Tool Version : Vivado v.2020.2 (win64) Build 3064766 Wed Nov 18 09:12:45 MST 2020  | Date : Mon Dec 13 16:34:02 2021  | Host : c5b1 running 64-bit major release (build 9200)  | Command : report\_utilization -file C:/ADS\_do/lab3/report/utilization\_report.txt -name utilization\_1  | Design : vga\_controller2\_v2  | Device : 7z020clg484-1  | Design State : Routed  -----------------------------------------------------------------------------------------------------------  Utilization Design Information  Table of Contents  -----------------  1. Slice Logic  1.1 Summary of Registers by Type  2. Slice Logic Distribution  3. Memory  4. DSP  5. IO and GT Specific  6. Clocking  7. Specific Feature  8. Primitives  9. Black Boxes  10. Instantiated Netlists  1. Slice Logic  --------------  +-------------------------+------+-------+-----------+-------+  | Site Type | Used | Fixed | Available | Util% |  +-------------------------+------+-------+-----------+-------+  | Slice LUTs | 129 | 0 | 53200 | 0.24 |  | LUT as Logic | 129 | 0 | 53200 | 0.24 |  | LUT as Memory | 0 | 0 | 17400 | 0.00 |  | Slice Registers | 223 | 0 | 106400 | 0.21 |  | Register as Flip Flop | 223 | 0 | 106400 | 0.21 |  | Register as Latch | 0 | 0 | 106400 | 0.00 |  | F7 Muxes | 0 | 0 | 26600 | 0.00 |  | F8 Muxes | 0 | 0 | 13300 | 0.00 |  +-------------------------+------+-------+-----------+-------+  1.1 Summary of Registers by Type  --------------------------------  +-------+--------------+-------------+--------------+  | Total | Clock Enable | Synchronous | Asynchronous |  +-------+--------------+-------------+--------------+  | 0 | \_ | - | - |  | 0 | \_ | - | Set |  | 0 | \_ | - | Reset |  | 0 | \_ | Set | - |  | 0 | \_ | Reset | - |  | 0 | Yes | - | - |  | 0 | Yes | - | Set |  | 0 | Yes | - | Reset |  | 3 | Yes | Set | - |  | 220 | Yes | Reset | - |  +-------+--------------+-------------+--------------+  2. Slice Logic Distribution  ---------------------------  +--------------------------------------------+------+-------+-----------+-------+  | Site Type | Used | Fixed | Available | Util% |  +--------------------------------------------+------+-------+-----------+-------+  | Slice | 72 | 0 | 13300 | 0.54 |  | SLICEL | 53 | 0 | | |  | SLICEM | 19 | 0 | | |  | LUT as Logic | 129 | 0 | 53200 | 0.24 |  | using O5 output only | 0 | | | |  | using O6 output only | 92 | | | |  | using O5 and O6 | 37 | | | |  | LUT as Memory | 0 | 0 | 17400 | 0.00 |  | LUT as Distributed RAM | 0 | 0 | | |  | LUT as Shift Register | 0 | 0 | | |  | Slice Registers | 223 | 0 | 106400 | 0.21 |  | Register driven from within the Slice | 86 | | | |  | Register driven from outside the Slice | 137 | | | |  | LUT in front of the register is unused | 125 | | | |  | LUT in front of the register is used | 12 | | | |  | Unique Control Sets | 27 | | 13300 | 0.20 |  +--------------------------------------------+------+-------+-----------+-------+  \* \* Note: Available Control Sets calculated as Slice \* 1, Review the Control Sets Report for more information regarding control sets.  3. Memory  ---------  +----------------+------+-------+-----------+-------+  | Site Type | Used | Fixed | Available | Util% |  +----------------+------+-------+-----------+-------+  | Block RAM Tile | 0 | 0 | 140 | 0.00 |  | RAMB36/FIFO\* | 0 | 0 | 140 | 0.00 |  | RAMB18 | 0 | 0 | 280 | 0.00 |  +----------------+------+-------+-----------+-------+  \* Note: Each Block RAM Tile only has one FIFO logic available and therefore can accommodate only one FIFO36E1 or one FIFO18E1. However, if a FIFO18E1 occupies a Block RAM Tile, that tile can still accommodate a RAMB18E1  4. DSP  ------  +-----------+------+-------+-----------+-------+  | Site Type | Used | Fixed | Available | Util% |  +-----------+------+-------+-----------+-------+  | DSPs | 0 | 0 | 220 | 0.00 |  +-----------+------+-------+-----------+-------+  5. IO and GT Specific  ---------------------  +-----------------------------+------+-------+-----------+-------+  | Site Type | Used | Fixed | Available | Util% |  +-----------------------------+------+-------+-----------+-------+  | Bonded IOB | 104 | 0 | 200 | 52.00 |  | IOB Master Pads | 49 | | | |  | IOB Slave Pads | 52 | | | |  | Bonded IPADs | 0 | 0 | 2 | 0.00 |  | Bonded IOPADs | 0 | 0 | 130 | 0.00 |  | PHY\_CONTROL | 0 | 0 | 4 | 0.00 |  | PHASER\_REF | 0 | 0 | 4 | 0.00 |  | OUT\_FIFO | 0 | 0 | 16 | 0.00 |  | IN\_FIFO | 0 | 0 | 16 | 0.00 |  | IDELAYCTRL | 0 | 0 | 4 | 0.00 |  | IBUFDS | 0 | 0 | 192 | 0.00 |  | PHASER\_OUT/PHASER\_OUT\_PHY | 0 | 0 | 16 | 0.00 |  | PHASER\_IN/PHASER\_IN\_PHY | 0 | 0 | 16 | 0.00 |  | IDELAYE2/IDELAYE2\_FINEDELAY | 0 | 0 | 200 | 0.00 |  | ILOGIC | 0 | 0 | 200 | 0.00 |  | OLOGIC | 0 | 0 | 200 | 0.00 |  +-----------------------------+------+-------+-----------+-------+  6. Clocking  -----------  +------------+------+-------+-----------+-------+  | Site Type | Used | Fixed | Available | Util% |  +------------+------+-------+-----------+-------+  | BUFGCTRL | 1 | 0 | 32 | 3.13 |  | BUFIO | 0 | 0 | 16 | 0.00 |  | MMCME2\_ADV | 0 | 0 | 4 | 0.00 |  | PLLE2\_ADV | 0 | 0 | 4 | 0.00 |  | BUFMRCE | 0 | 0 | 8 | 0.00 |  | BUFHCE | 0 | 0 | 72 | 0.00 |  | BUFR | 0 | 0 | 16 | 0.00 |  +------------+------+-------+-----------+-------+  7. Specific Feature  -------------------  +-------------+------+-------+-----------+-------+  | Site Type | Used | Fixed | Available | Util% |  +-------------+------+-------+-----------+-------+  | BSCANE2 | 0 | 0 | 4 | 0.00 |  | CAPTUREE2 | 0 | 0 | 1 | 0.00 |  | DNA\_PORT | 0 | 0 | 1 | 0.00 |  | EFUSE\_USR | 0 | 0 | 1 | 0.00 |  | FRAME\_ECCE2 | 0 | 0 | 1 | 0.00 |  | ICAPE2 | 0 | 0 | 2 | 0.00 |  | STARTUPE2 | 0 | 0 | 1 | 0.00 |  | XADC | 0 | 0 | 1 | 0.00 |  +-------------+------+-------+-----------+-------+  8. Primitives  -------------  +----------+------+---------------------+  | Ref Name | Used | Functional Category |  +----------+------+---------------------+  | FDRE | 220 | Flop & Latch |  | LUT6 | 76 | LUT |  | OBUF | 56 | IO |  | IBUF | 48 | IO |  | LUT4 | 43 | LUT |  | LUT5 | 19 | LUT |  | LUT2 | 13 | LUT |  | CARRY4 | 13 | CarryLogic |  | LUT3 | 11 | LUT |  | LUT1 | 4 | LUT |  | FDSE | 3 | Flop & Latch |  | BUFG | 1 | Clock |  +----------+------+---------------------+  9. Black Boxes  --------------  +----------+------+  | Ref Name | Used |  +----------+------+  10. Instantiated Netlists  -------------------------  +----------+------+  | Ref Name | Used |  +----------+------+ |