Sistema de Monitoreo de Estaciones de Medición

Trabajo Práctico Final de Circuitos Lógicos Programables

Transmisor y Receptor Serializador para intercambio de datos entre módulos

Roberto Oscar Axt

(roberto.axt@gmail.com)

04/11/2024

Historial de cambios

Versión	Fecha	Descripción	Autor	Revisores
A	14/04/2025	Versión Original	Roberto Axt	

Índice de contenido

1. Introducción

1.1. Contexto

El proyecto final de MIoT consiste en el desarrollo de un *Sistema de Monitoreo para Estaciones de Medición de Gas Natural* (SMEM). Estas estaciones están compuestas por shelters, en cuyo interior se alojan los sistemas de medición, energía y comunicaciones.

La mayoría de estas estaciones reciben energía eléctrica de la red, suministrada por cooperativas locales. Además, cuentan con un sistema básico de respaldo para casos de corte de suministro. Se ha identificado una necesidad clave: determinar el origen de las fallas eléctricas, es decir, si estas se deben a problemas en la red de suministro externa o a inconvenientes internos en la estación.

El sistema SMEM estará compuesto por dos módulos:

- **Módulo Central de Control**, en adelante denominado **MCC**, ubicado dentro del shelter.
- **Módulo de Supervisión Remota**, en delante denominado **MSR,** instalado en el pilar de energía.

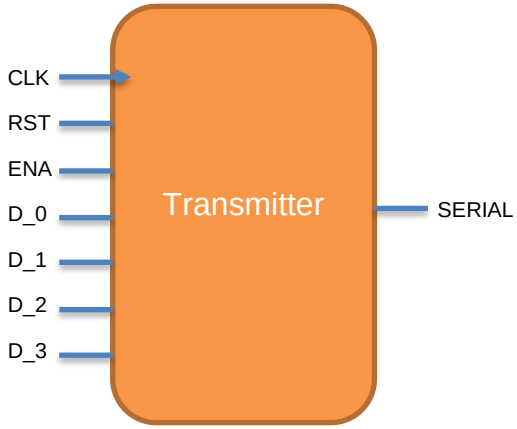
El módulo MSR tendrá dos sensores de presencia de 220 V: uno colocado antes y otro después del disyuntor o térmica principal de la estación. Además, contará con dos sensores tipo reed switch para la detección de manipulaciones (*tampering*). Estas cuatro señales serán serializadas y enviadas al módulo MCC a través de un submódulo de transmisión mediante un enlace de 433 MHz.

El módulo MCC recibirá los 4 bits de un submódulo de recepción, el cual se encarga de la decodificación de los datos seriales y su verificación.

EM&R El Chaja

1.2. Propuesta del Submódulo de Transmisión

El submódulo de transmisión recibe los 4 bits en paralelo de las señales mencionada y se encarga de serializar estos datos. Además, agrega una secuencia de 3 bits denominada preámbulo para mejorar las características de sincronización y otra secuencia de 3 bits de CRC-3 para poder validar que los datos fueron recibidos correctamente.

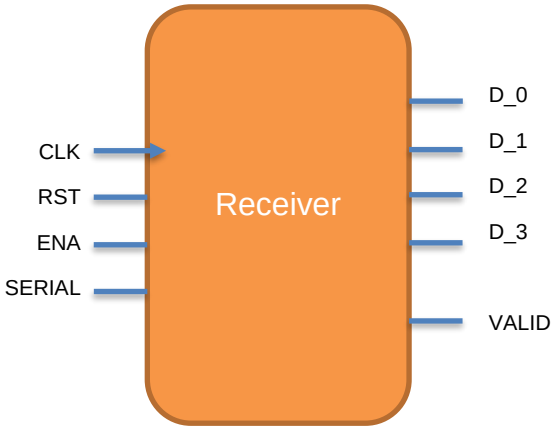


Donde los datos seriales de salida, tendrán la siguiente forma

Preámbulo			Datos				CRC			Idle
0	1	0	X	X	X	X	Y	Y	Y	1

1.3. Propuesta del Submódulo de Recepción

El submódulo de recepción recibe la secuencia mencionada. El mismo inicia esperando la secuencia del preámbulo para sincronizar el payload, que son los datos y el CRC-3. Una vez recibidos estos 7 bits verifica que la secuencia de CRC sea correcta y envía los datos a la salida junto con una señal de validación.



2. Proyecto

2.1. Formato de carpetas

El proyecto se encuentra en el siguiente repositorio

https://github.com/RobAxt/CLP_workspace/tree/main/TPfinal

Está compuesto por las carpetas

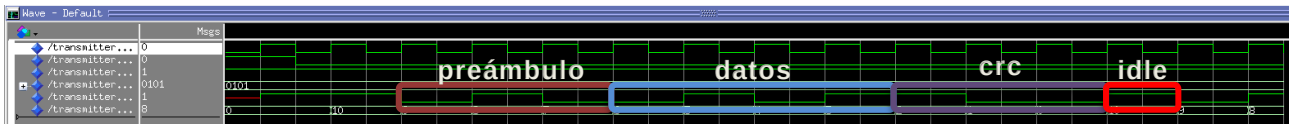
- **Documentos:** carpeta con este documento y reporte de utilización.
- **Fuentes**
  - **Implementación:** carpeta con los códigos fuentes de la implementación del transceiver para la placa ArtyZ7_10.
  - **Receptor:** carpeta con el código fuente del receptor y un testbench donde se simulan casos exitosos y de falla del conjunto transmisor receptor, denominado transceiver.
  - **Transmisor:** carpeta con el código fuente del transmisor y un testbench donde se valida la trama de salida.
- **Simulación:** simulación de cada paso de la implementación usando modelSIM
- **Síntesis:** síntesis del proyecto utilizando la herramienta Vivado, dentro de la carpeta Transceiver.runs/impl_1 se encuentra el archivo transceriver_impl.bit para la programación de la placa de desarrollo ArtyZ7_10.

2.2. Implementación del Transmisor

La lógica del transmisor se encuentra en el archivo *"transmitter.vhd",* el mismo fue programado por comportamiento. La idea principal de la programación esta definida por una variable llamada bit_count, la cual lleva el control de las acciones a tomar. Cuando esta variable está en cero se procede a la carga de los valores de la entrada, junto con su CRC y el preámbulo para formar el frame de salida. Así también se mantiene la salida en estado IDLE durante este estado y se actualiza el valor de bit_count a la cantidad de bits del frame a transmitir. Cuando bit_count es distinto de cero, se transmite cada bit del frame partiendo de los bits más significativos.

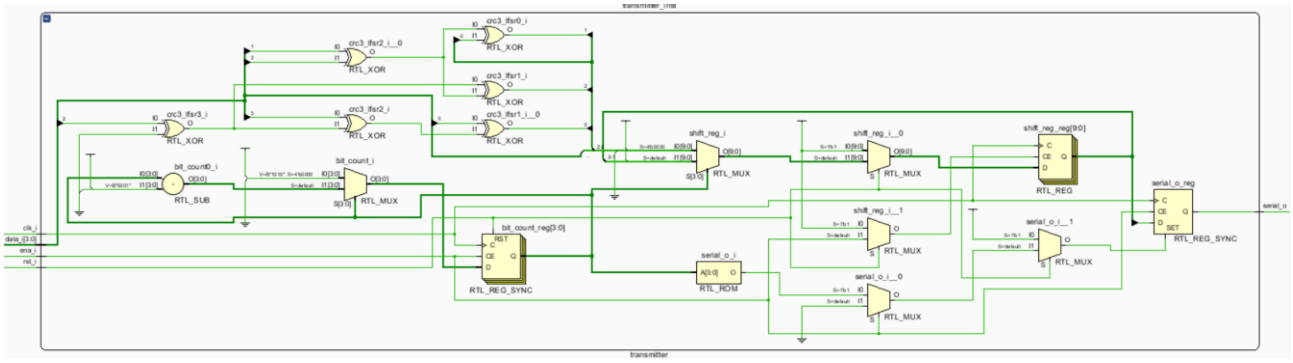
Lo mencionado ocurre cada flanco de subida del clock y si y solo sí el bit de reset se encuentra bajo y el bit de enable se encuentra alto.

Para probar esta lógica se realizo un testbench llamado *"transmitter_tb.vhd"* y se lo simuló utilizando modelSIM, con los siguientes resultados.

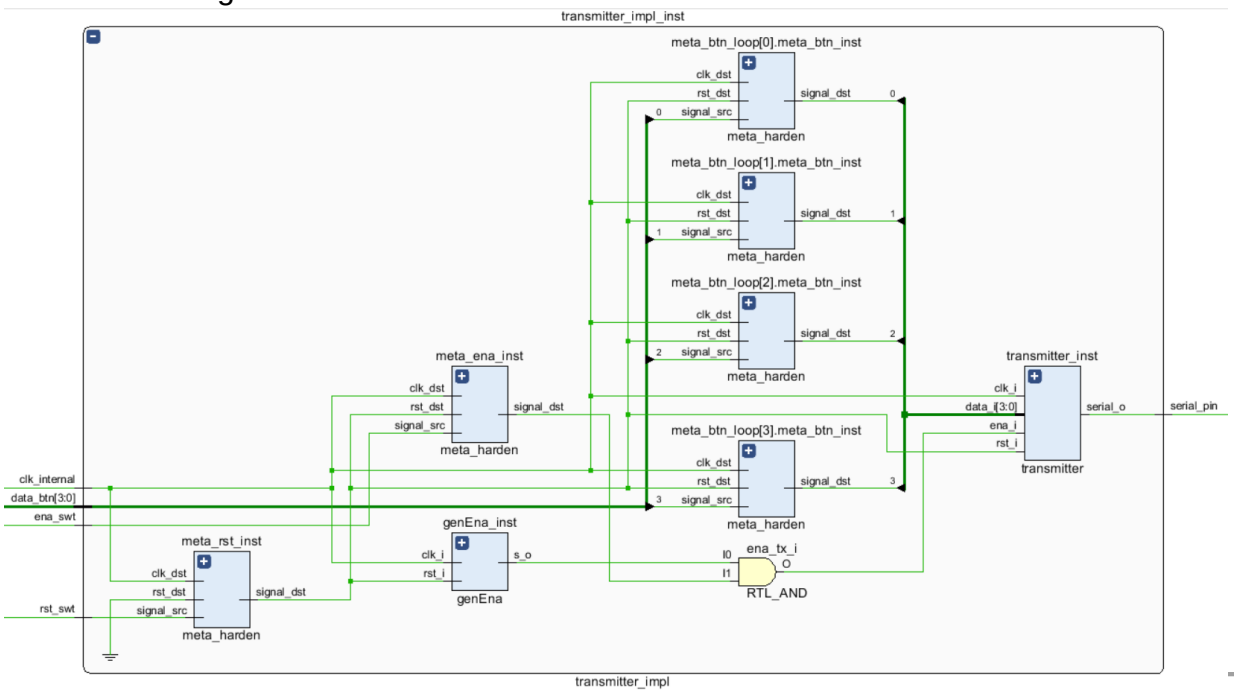


En esta simulación se observa el preámbulo "010", los datos "0101", el CRC de los datos "100" y su variación en cada flanco de subida del reloj. También se observa el bit de idle '1' que es cuando internamente se está haciendo la carga de los datos cuando bit_count es cero.

De esta programación por comportamiento el proceso RTL obtiene el siguiente esquemático



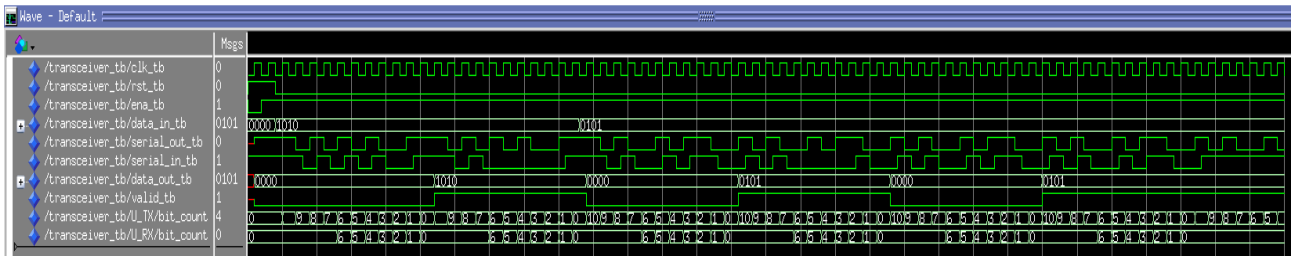
Antes de llevar esta lógica a la implementación es necesario tener en cuenta dos cosas. Primero el endurecimiento de las señales de entrada para evitar estados meta-estables y que los datos de salida deben salir con una frecuencia muy inferior a la frecuencia del reloj de la placa de desarrollo. En función de esto se crea otro archivo *"transmitter_impl.vhd"* y se observan las siguientes conexiones.



2.3. Implementación del Receptor

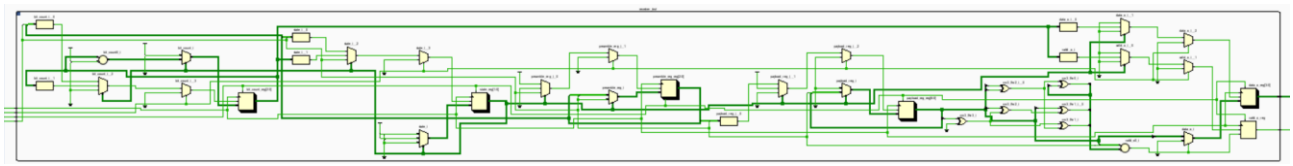
La lógica del receptor se encuentra en el archivo *“receiver.vhd”,* el mismo fue programado por comportamiento. El receptor esta implementado por una maquina de estados que tiene 3 estados PREAMBLE_SEACH, RECEIVING y WAIT_IDLE. En el primer estado, con el cual arranca el sistema, se encarga de la detección de la secuencia del preámbulo (“010”). Cuando la misma es detectada se pasa al siguiente estado que es la lógica de recepción. En este estado se implemento una idea similar al transmisor donde hay una variable de bit_count que va marcando que acción tomar. Cuando bit_count es distinto de cero se van cargando los bits recibidos desde el bit menos significativo del payload_reg y empujando los bits hacia los más significativos. Cuando el bit_count llega a cero, implica que ya se han cargado todos los bits del payload y se procede a verificar que el CRC. De validarse correctamente se cargan los bits a la salida, junto con la señal de datos válidos, de lo contrario se pasa todo a cero.

Se realiza un testbench denominado *“tranceiver_tb.vhd”* en donde se integran tanto el receptor como el transmisor y se crean casos de transmisión válidos y con fallas.

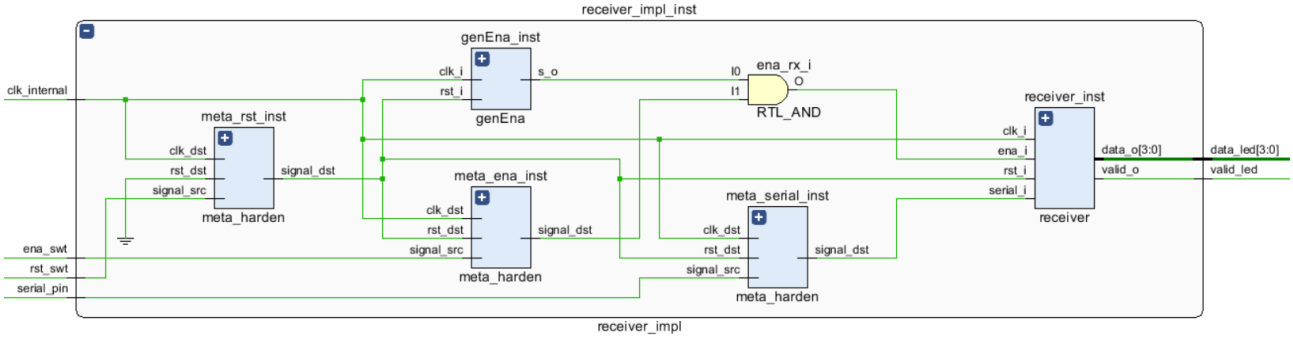


En esta simulación se observa la transmisión de los datos “1010” y que la salida del receptor se actualiza al terminar la trasmisión de los mismos. La siguiente secuencia es invalida, ya que se transmite la secuencia “0000” con un CRC invalido, por lo que se observa la señal de validación en cero. La tercera secuencia de transmisión es válida con datos “0101” y se observan los mismos a la salida. La cuarta secuencia vuelve a ser invalida y se reestablece en la quinta y final secuencia.

De esta programación por comportamiento el proceso RTL obtiene el siguiente esquemático de la máquina de estados implementada del receptor.

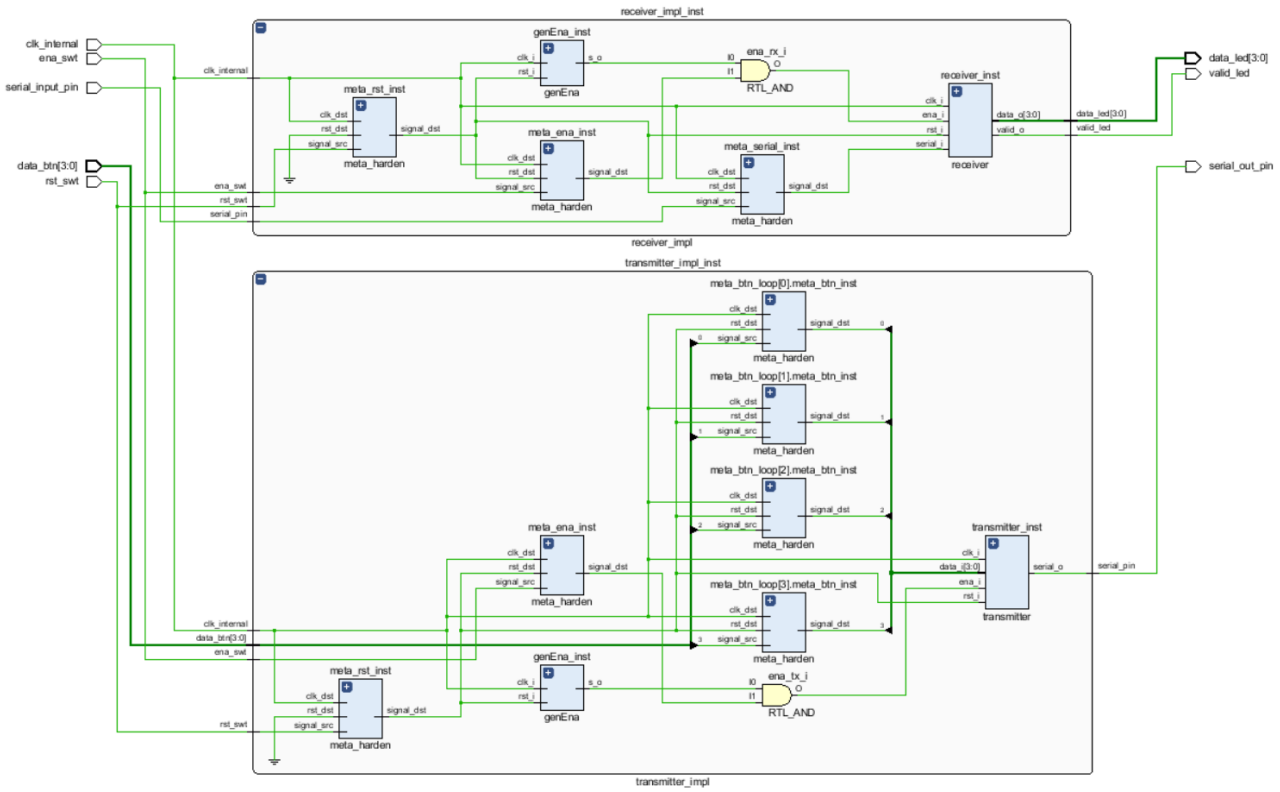


Antes de llevar esta lógica a la implementación se realiza el endurecimiento de las señales de entrada y la creación del bloque genEna para disminuir la frecuencia de operación del receptor a la misma del transmisor.



2.4. Implementación completa

La implementación completa del sistema se realiza en el archivo *"transceiver_impl.vhd"*. En este módulo se integran ambas implementaciones del transmisor y del receptor para su síntesis.



En la implementación con la placa de desarrollo se utilizan los switches para las señales de reset y enable. Los botones corresponden a los datos de entrada al transmisor y los leds asociados para la visualización de los datos de salida de receptor. Así se utiliza el color verde de uno de los leds RGB como visualizador del bit de validación de los datos de salida del receptor. Esta asignación esta en el archivo *"ArtyZ7_10.xdc".*

Con respecto al temporizado de la señal de salida, el componente de genEna cuenta 1.250.000 ciclos, por lo que cada bit en la implementación final es de 10 ms.

La utilización de la FPGA

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	55	17600	0.31
FF	92	35200	0.26
IO	14	100	14.00
BUFG	1	32	3.13

El resto del reporte se encuentra en el anexo I.

3. Anexo I: Reporte de utilización

```
 Copyright 1986-2018 Xilinx, Inc. All Rights Reserved.
----------------------------------------------------------------------------------------------------
| Tool Version : Vivado v.2018.1 (win64) Build 2188600 Wed Apr  4 18:40:38 MDT
2018
| Date         : Wed Apr  9 21:31:04 2025
| Host         : NB459408 running 64-bit major release  (build 9200)
| Command      : report_utilization -file
C:/Xilinx/TPfinal/Fuente/Tranceiver_UtilizationReport.txt -name utilization_1
| Design       : transceriver_impl
| Device       : 7z010clg400-1
| 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              |   55 |     0 |     17600 |  0.31 |
|   LUT as Logic          |   55 |     0 |     17600 |  0.31 |
|   LUT as Memory         |    0 |     0 |      6000 |  0.00 |
| Slice Registers         |   92 |     0 |     35200 |  0.26 |
|   Register as Flip Flop |   92 |     0 |     35200 |  0.26 |
|   Register as Latch     |    0 |     0 |     35200 |  0.00 |
| F7 Muxes                |    0 |     0 |      8800 |  0.00 |
| F8 Muxes                |    0 |     0 |      4400 |  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 |
| 14    |            Yes |         Set |             - |
| 78    |            Yes |       Reset |             - |
+-------+----------------+-------------+---------------+


2. Slice Logic Distribution
---------------------------

+--------------------------------------------+------+-------+-----------+-------+
|                  Site Type                 | Used | Fixed | Available | Util% |
+--------------------------------------------+------+-------+-----------+-------+
| Slice                                      |   44 |     0 |      4400 |  1.00 |
|   SLICEL                                   |   32 |     0 |           |       |
|   SLICEM                                   |   12 |     0 |           |       |
| LUT as Logic                               |   55 |     0 |     17600 |  0.31 |
|   using O5 output only                     |    0 |       |           |       |
|   using O6 output only                     |   49 |       |           |       |
|   using O5 and O6                          |    6 |       |           |       |
| LUT as Memory                              |    0 |     0 |      6000 |  0.00 |
|   LUT as Distributed RAM                   |    0 |     0 |           |       |
|   LUT as Shift Register                    |    0 |     0 |           |       |
| LUT Flip Flop Pairs                        |   32 |     0 |     17600 |  0.18 |
|   fully used LUT-FF pairs                  |    4 |       |           |       |
|   LUT-FF pairs with one unused LUT output  |   28 |       |           |       |
|   LUT-FF pairs with one unused Flip Flop   |   27 |       |           |       |
| Unique Control Sets                        |    9 |       |           |       |
+--------------------------------------------+------+-------+-----------+-------+
* Note: Review the Control Sets Report for more information regarding control
sets.


3. Memory
---------

+----------------+------+-------+-----------+-------+
|    Site Type   | Used | Fixed | Available | Util% |
+----------------+------+-------+-----------+-------+
| Block RAM Tile |    0 |     0 |        60 |  0.00 |
|   RAMB36/FIFO* |    0 |     0 |        60 |  0.00 |
|   RAMB18       |    0 |     0 |       120 |  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 |        80 |  0.00 |
+-----------+------+-------+-----------+-------+


5. IO and GT Specific
---------------------
```

Site Type	Used	Fixed	Available	Util%
Bonded IOB	14	14	100	14.00
IOB Master Pads	9			
IOB Slave Pads	5			
Bonded IPADs	0	0	2	0.00
Bonded IOPADs	0	0	130	0.00
PHY_CONTROL	0	0	2	0.00
PHASER_REF	0	0	2	0.00
OUT_FIFO	0	0	8	0.00
IN_FIFO	0	0	8	0.00
IDELAYCTRL	0	0	2	0.00
IBUFDS	0	0	96	0.00
PHASER_OUT/PHASER_OUT_PHY	0	0	8	0.00
PHASER_IN/PHASER_IN_PHY	0	0	8	0.00
IDELAYE2/IDELAYE2_FINEDELAY	0	0	100	0.00
ILOGIC	0	0	100	0.00
OLOGIC	0	0	100	0.00

6. Clocking

Site Type	Used	Fixed	Available	Util%
BUFGCTRL	1	0	32	3.13
BUFIO	0	0	8	0.00
MMCME2_ADV	0	0	2	0.00
PLLE2_ADV	0	0	2	0.00
BUFMRCE	0	0	4	0.00
BUFHCE	0	0	48	0.00
BUFR	0	0	8	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	78	Flop & Latch
CARRY4	22	CarryLogic

```
| LUT5      |   19 |                  LUT |
| LUT6      |   16 |                  LUT |
| FDSE      |   14 |        Flop & Latch |
| LUT2      |   11 |                  LUT |
| LUT4      |    8 |                  LUT |
| IBUF      |    8 |                   IO |
| OBUF      |    6 |                   IO |
| LUT3      |    5 |                  LUT |
| LUT1      |    2 |                  LUT |
| BUFG      |    1 |               Clock |
+-----------+------+----------------------+


9. Black Boxes
--------------

+----------+------+
| Ref Name | Used |
+----------+------+


10. Instantiated Netlists
-------------------------

+----------+------+
| Ref Name | Used |
+----------+------+
```