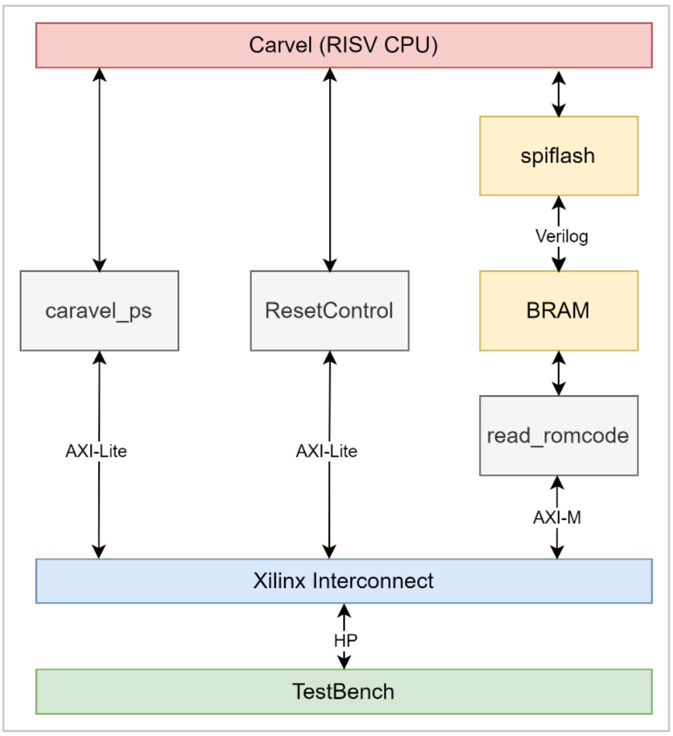
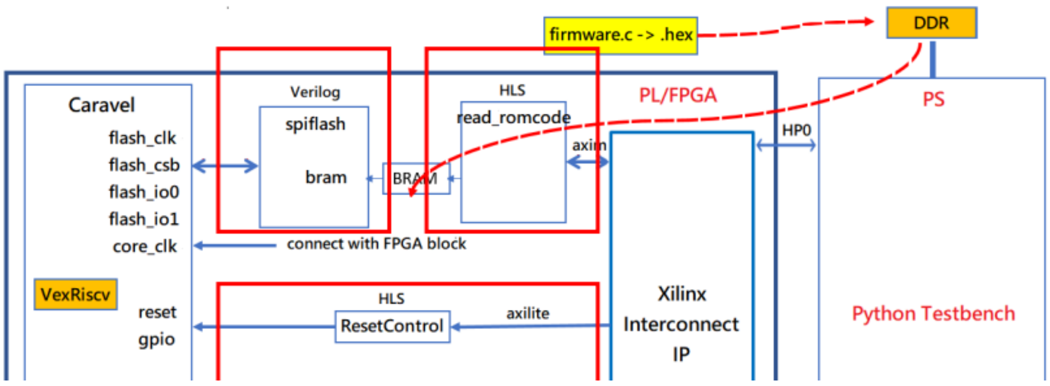
Block diagram

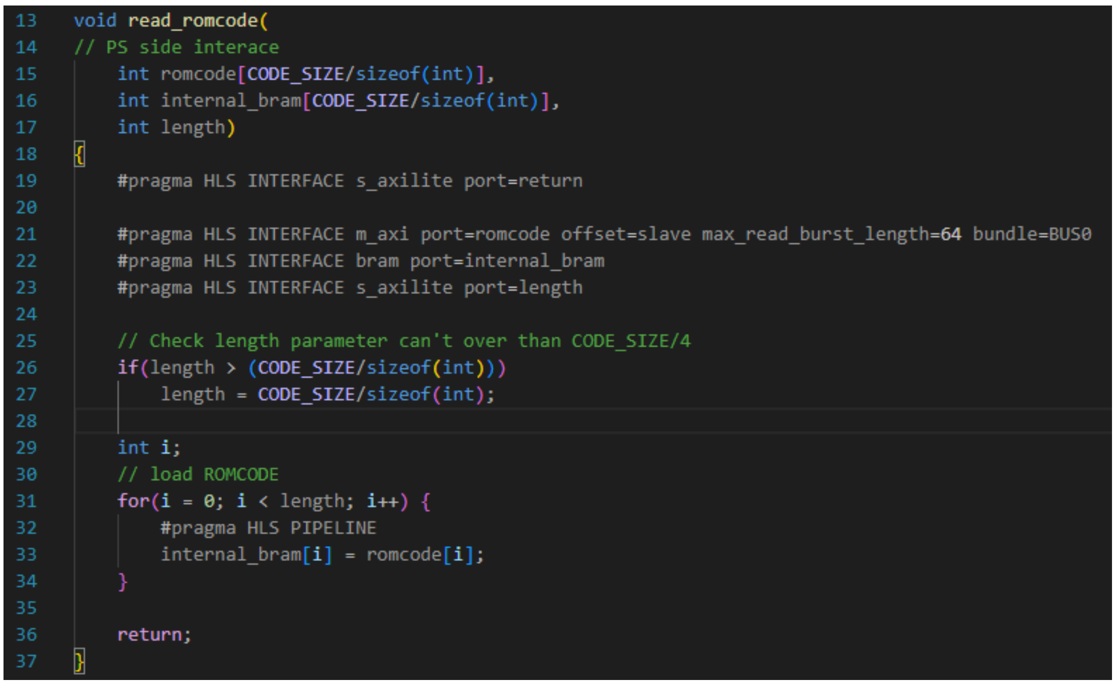


compile 後的 firmware code 是存在 PS side 的 DDR 裡，它會透過 read_romcode 這個 IP，利用 AXI master 的形式將其 load 進硬體的 BRAM 裡。而當 firmware load 進來後，則有另一個 IP（ResetControl），會把 RESET release 掉，RISCV CPU 就開始跑，開始從 spiflash fetch firmware code。跑完之後，firmware 會將值丟到 mprj pin 上。但現在沒有 simulator 可以直接從 pin 上看值，所以需要再透過一個 IP（carvel_ps），把這些值反應在 AXI-Lite 上，PS side 上的 CPU 才能做 MMIO 的 read。

IP 用處：

1. read_romcode & spiflash: 因為兩個是連在一起的所以一起講



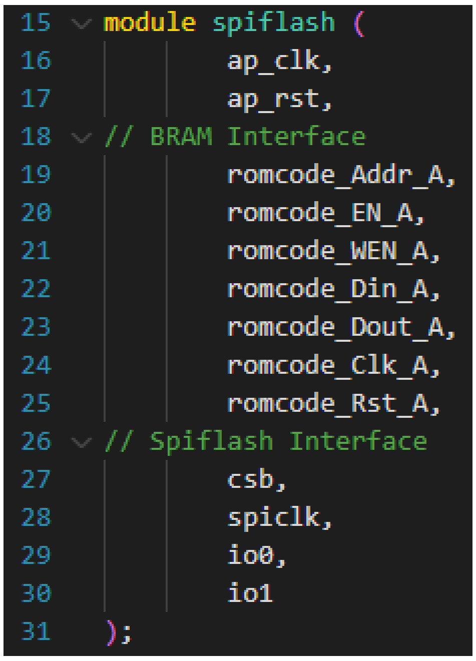


```
13  void read_romcode(
14  // PS side interace
15      int romcode[CODE_SIZE/sizeof(int)],
16      int internal_bram[CODE_SIZE/sizeof(int)],
17      int length)
18  {
19      #pragma HLS INTERFACE s_axilite port=return
20
21      #pragma HLS INTERFACE m_axi port=romcode offset=slave max_read_burst_length=64 bundle=BUS0
22      #pragma HLS INTERFACE bram port=internal_bram
23      #pragma HLS INTERFACE s_axilite port=length
24
25      // Check length parameter can't over than CODE_SIZE/4
26      if(length > (CODE_SIZE/sizeof(int)))
27          length = CODE_SIZE/sizeof(int);
28
29      int i;
30      // load ROMCODE
31      for(i = 0; i < length; i++) {
32          #pragma HLS PIPELINE
33          internal_bram[i] = romcode[i];
34      }
35
36      return;
37  }
```

Read_romcode 的用處其實註解講得挺清楚的，是想利用 axi_m 去讀取 system memory 內的 romcode，得到的結果再存進 bram 裡面，CODE_SIZE/sizeof(int)就是 system 總共能讀取的 data 總量。

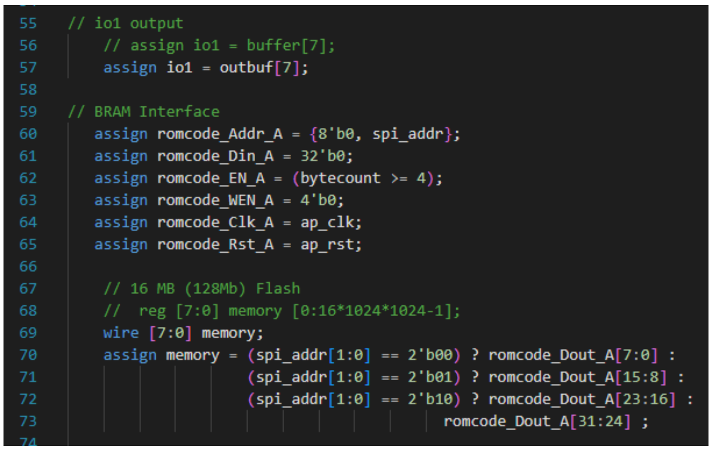
26 行開始的 if 的用處：如果這個 parameter length(想要讀取的 data 數量)比能讀取的 data 數還要多，就會把讀取次數限制在上限。

31 行開始的 for 迴圈就是將讀出來的 romcode 寫進 bram，總共要讀 length 次。



```
module spiflash (
        ap_clk,
        ap_rst,
// BRAM Interface
        romcode_Addr_A,
        romcode_EN_A,
        romcode_WEN_A,
        romcode_Din_A,
        romcode_Dout_A,
        romcode_Clk_A,
        romcode_Rst_A,
// Spiflash Interface
        csb,
        spiclk,
        io0,
        io1
);
```

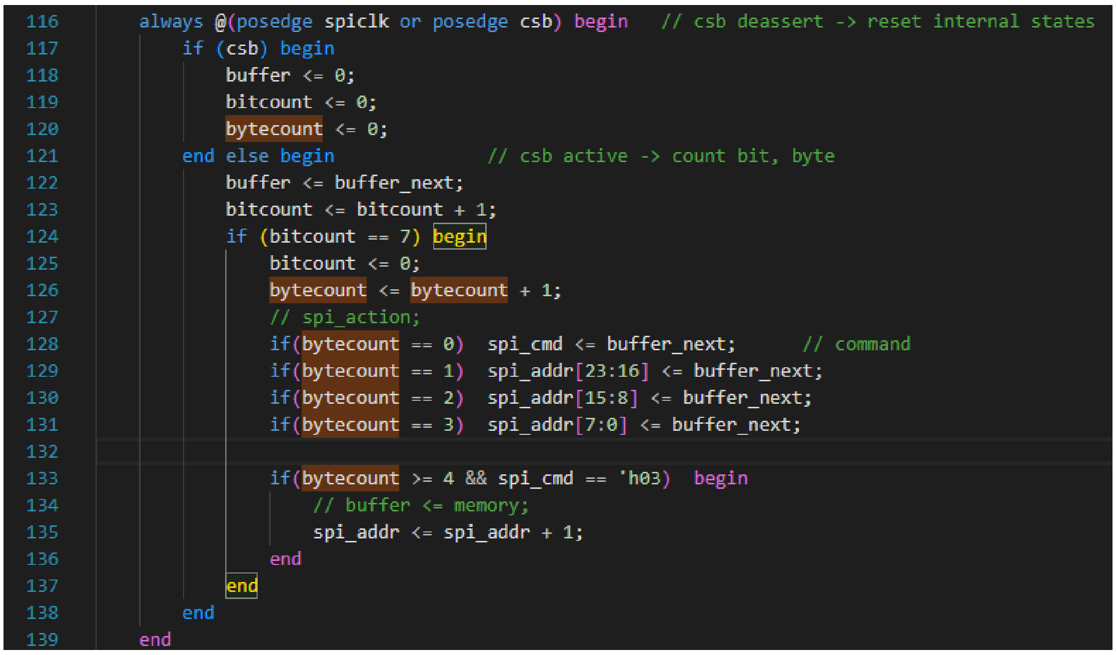
Spifalsh 依照 block diagram 宣告對應的 interface 訊號，io0 相當於 input，io1 相當於 output。



```
// io1 output
    // assign io1 = buffer[7];
    assign io1 = outbuf[7];

// BRAM Interface
   assign romcode_Addr_A = {8'b0, spi_addr};
   assign romcode_Din_A = 32'b0;
   assign romcode_EN_A = (bytecount >= 4);
   assign romcode_WEN_A = 4'b0;
   assign romcode_Clk_A = ap_clk;
   assign romcode_Rst_A = ap_rst;

    // 16 MB (128Mb) Flash
    //  reg [7:0] memory [0:16*1024*1024-1];
    wire [7:0] memory;
    assign memory = (spi_addr[1:0] == 2'b00) ? romcode_Dout_A[7:0] :
                    (spi_addr[1:0] == 2'b01) ? romcode_Dout_A[15:8] :
                    (spi_addr[1:0] == 2'b10) ? romcode_Dout_A[23:16] :
                                          romcode_Dout_A[31:24] ;
```

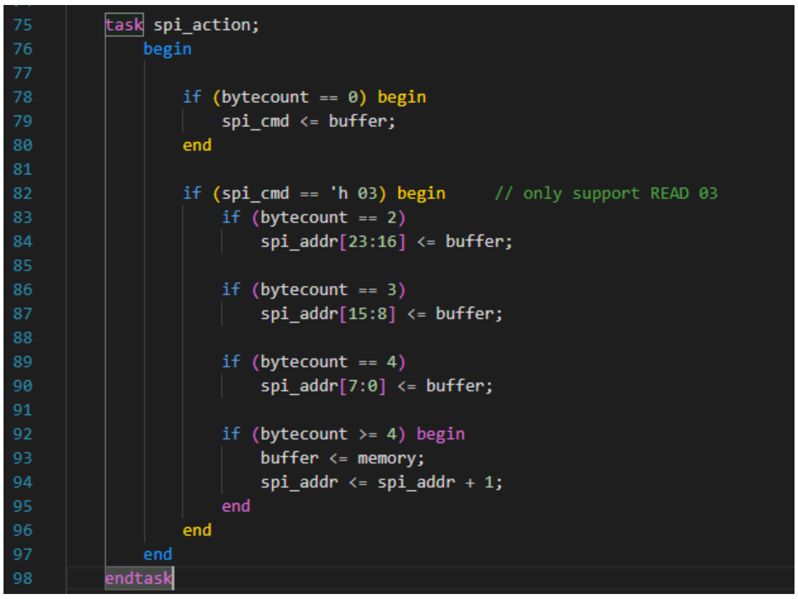
因為是 spiflash 當方面從 bram 讀取，所以 romcode_Din 和 WEN 永遠為 0，byte_count 大於等於 4 再進行 EN。

Memory 算是一個 rom_code_Dout 的 buffer，藉由 spi_addr 就能知道要拿的 romcode_Dout 是哪一部分。



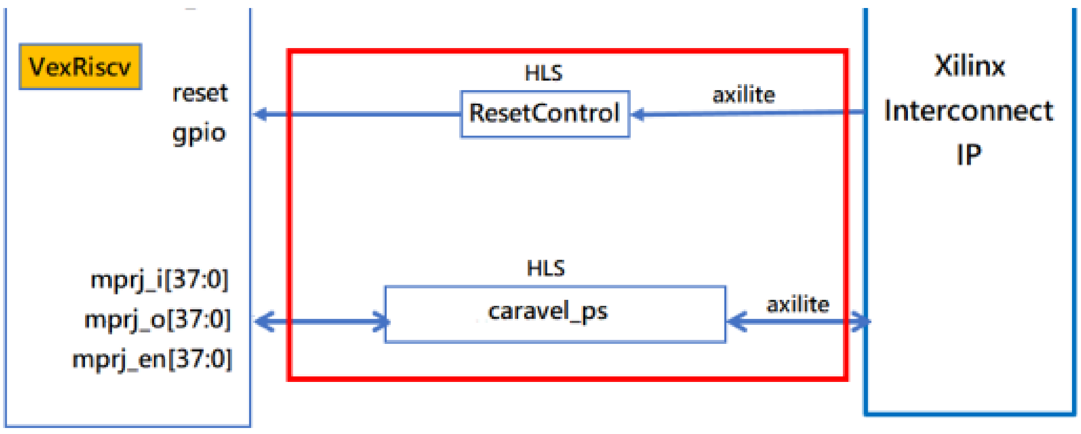
```
    always @(posedge spiclk or posedge csb) begin   // csb deassert -> reset internal states
        if (csb) begin
            buffer <= 0;
            bitcount <= 0;
            bytecount <= 0;
        end else begin              // csb active -> count bit, byte
            buffer <= buffer_next;
            bitcount <= bitcount + 1;
            if (bitcount == 7) begin
                bitcount <= 0;
                bytecount <= bytecount + 1;
                // spi_action;
                if(bytecount == 0)  spi_cmd <= buffer_next;      // command
                if(bytecount == 1)  spi_addr[23:16] <= buffer_next;
                if(bytecount == 2)  spi_addr[15:8] <= buffer_next;
                if(bytecount == 3)  spi_addr[7:0] <= buffer_next;

                if(bytecount >= 4 && spi_cmd == 'h03)  begin
                    // buffer <= memory;
                    spi_addr <= spi_addr + 1;
                end
            end
        end
    end
```

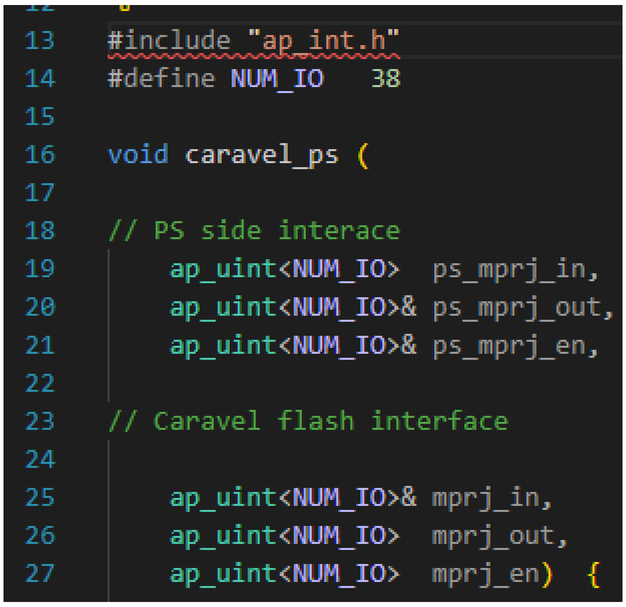
這邊可以看到 bitcount 和 bytecount 就是簡單的 counter，bytecount 總共有 12bit，可以數非常久，算是一個簡單的 FSM，在 bytecount =< 3 的時候，buffer 會幫所有 addr 和 cmd reset：



```
    task spi_action;
        begin

            if (bytecount == 0) begin
                spi_cmd <= buffer;
            end

            if (spi_cmd == 'h 03) begin     // only support READ 03
                if (bytecount == 2)
                    spi_addr[23:16] <= buffer;

                if (bytecount == 3)
                    spi_addr[15:8] <= buffer;

                if (bytecount == 4)
                    spi_addr[7:0] <= buffer;

                if (bytecount >= 4) begin
                    buffer <= memory;
                    spi_addr <= spi_addr + 1;
                end
            end
        end
    endtask
```

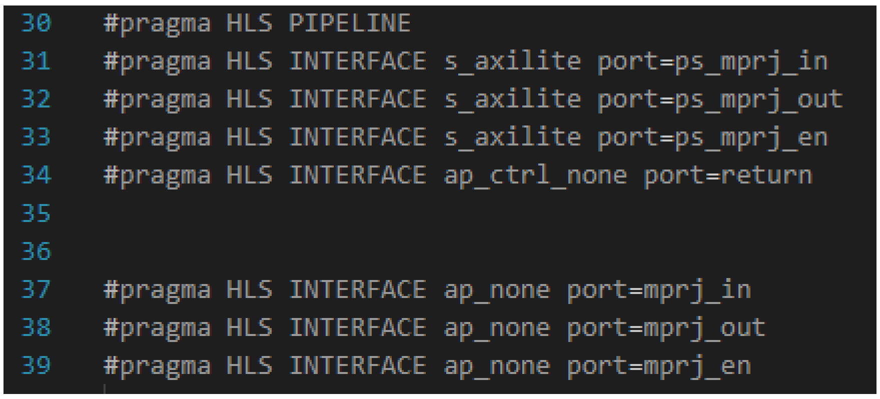
2. ResetControl & caravel_ps：
   這兩個因為都是使用 axi-lite 來和 system interface 進行溝通，所以可以併在一起討論。





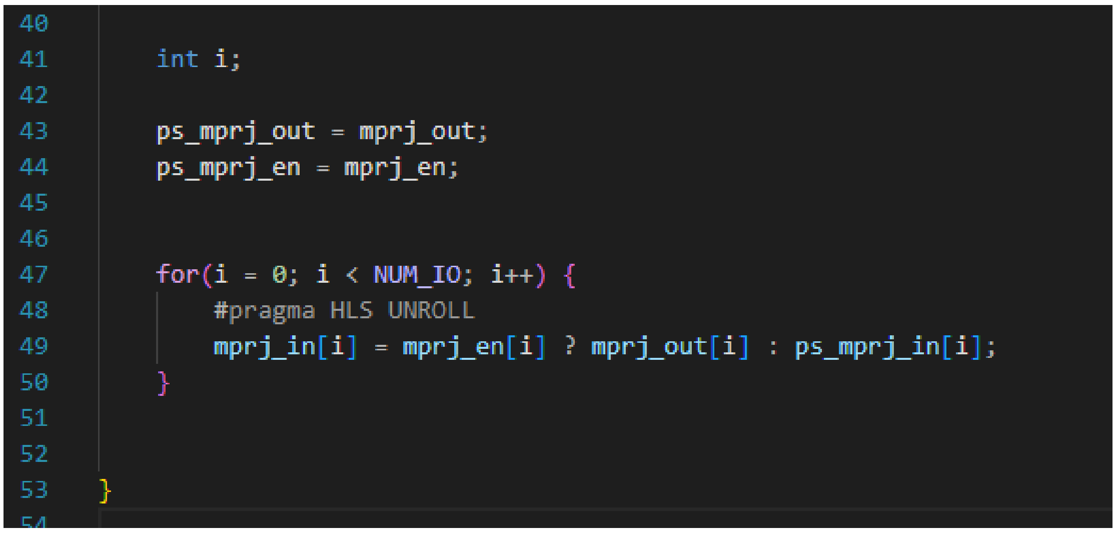
```
#include "ap_int.h"
#define NUM_IO   38

void caravel_ps (

// PS side interace
    ap_uint<NUM_IO>  ps_mprj_in,
    ap_uint<NUM_IO>& ps_mprj_out,
    ap_uint<NUM_IO>& ps_mprj_en,

// Caravel flash interface

    ap_uint<NUM_IO>& mprj_in,
    ap_uint<NUM_IO>  mprj_out,
    ap_uint<NUM_IO>  mprj_en)  {
```

和上面的 block diagram 敘述的一樣，將 NUM_IO 設定為 38。Ps 屬於 interconnect IP 的 mprj，沒有 PS 的就是 RiscV 端的 mprj。



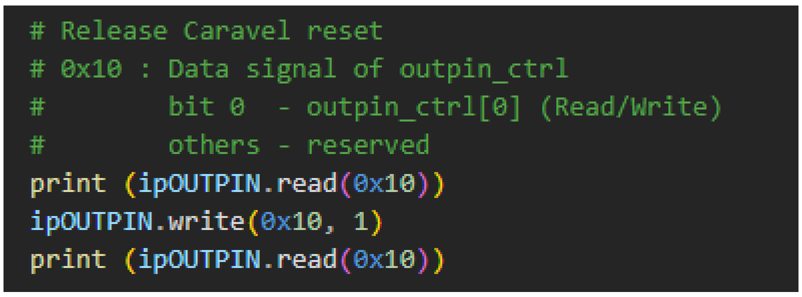
```
#pragma HLS PIPELINE
#pragma HLS INTERFACE s_axilite port=ps_mprj_in
#pragma HLS INTERFACE s_axilite port=ps_mprj_out
#pragma HLS INTERFACE s_axilite port=ps_mprj_en
#pragma HLS INTERFACE ap_ctrl_none port=return


#pragma HLS INTERFACE ap_none port=mprj_in
#pragma HLS INTERFACE ap_none port=mprj_out
#pragma HLS INTERFACE ap_none port=mprj_en
```

用 pragma 指定傳輸的 interface，也和 block diagram 一致。



```

    int i;

    ps_mprj_out = mprj_out;
    ps_mprj_en = mprj_en;


    for(i = 0; i < NUM_IO; i++) {
        #pragma HLS UNROLL
        mprj_in[i] = mprj_en[i] ? mprj_out[i] : ps_mprj_in[i];
    }


}
```

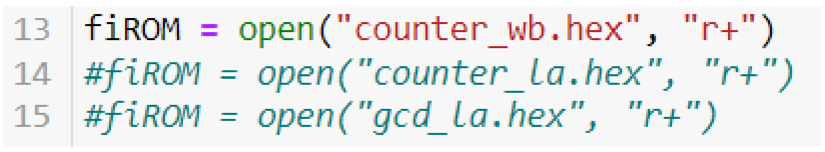
這塊表示 38 個 mprj_in 會根據對應的 mprj_en 去吃自己的 out 或者是從 bram 那邊去拿 data。



```python
# Release Caravel reset
# 0x10 : Data signal of outpin_ctrl
#         bit 0  - outpin_ctrl[0] (Read/Write)
#         others - reserved
print (ipOUTPIN.read(0x10))
ipOUTPIN.write(0x10, 1)
print (ipOUTPIN.read(0x10))
```

ResetControl 的部分我是在 ipydb 上找到的，這邊利用對應的 axilite 位址去傳送 OUTPIN control 訊號，ap_ctrl 再把訊號丟到 RiscV 內部達到 control 的效果。

workload on caravel FPGA:



```python
fiROM = open("counter_wb.hex", "r+")
#fiROM = open("counter_la.hex", "r+")
#fiROM = open("gcd_la.hex", "r+")
```

```python
# 0x00 : Control signals
#        bit 0  - ap_start (Read/Write/COH)
#        bit 1  - ap_done (Read/COR)
#        bit 2  - ap_idle (Read)
#        bit 3  - ap_ready (Read)
#        bit 7  - auto_restart (Read/Write)
#        others - reserved
# 0x10 : Data signal of romcode
#        bit 31~0 - romcode[31:0] (Read/Write)
# 0x14 : Data signal of romcode
#        bit 31~0 - romcode[63:32] (Read/Write)
# 0x1c : Data signal of length_r
#        bit 31~0 - length_r[31:0] (Read/Write)

# Program physical address for the romcode base address
ipReadROMCODE.write(0x10, npROM.device_address)
ipReadROMCODE.write(0x14, 0)
# Program length of moving data
ipReadROMCODE.write(0x1C, rom_size_final)


# ipReadROMCODE start to move the data from rom_buffer to bram
ipReadROMCODE.write(0x00, 1) # IP Start
while (ipReadROMCODE.read(0x00) & 0x04) == 0x00: # wait for done
    continue

print("Write to bram done")

```

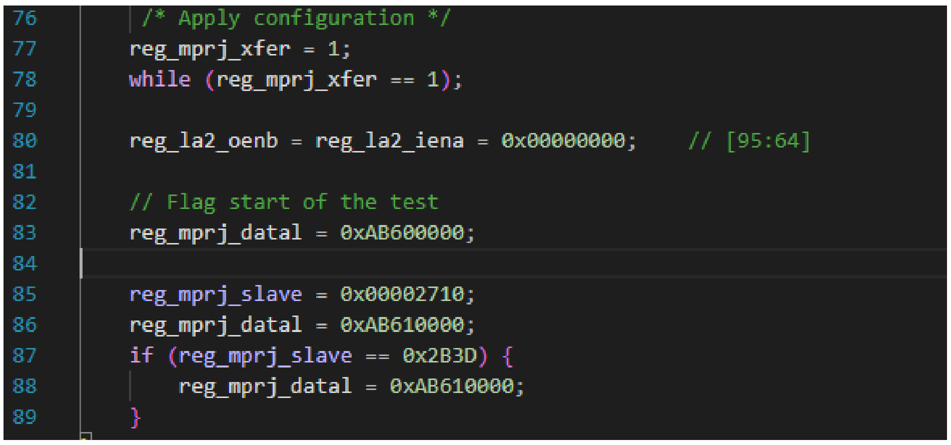
```
Write to bram done
```

```python
# Check MPRJ_IO input/out/en
# 0x10 : Data signal of ps_mprj_in
#        bit 31~0 - ps_mprj_in[31:0] (Read/Write)
# 0x14 : Data signal of ps_mprj_in
#        bit 5~0 - ps_mprj_in[37:32] (Read/Write)
#        others  - reserved
# 0x1c : Data signal of ps_mprj_out
#        bit 31~0 - ps_mprj_out[31:0] (Read)
# 0x20 : Data signal of ps_mprj_out
#        bit 5~0 - ps_mprj_out[37:32] (Read)
#        others  - reserved
# 0x34 : Data signal of ps_mprj_en
#        bit 31~0 - ps_mprj_en[31:0] (Read)
# 0x38 : Data signal of ps_mprj_en
#        bit 5~0 - ps_mprj_en[37:32] (Read)
#        others  - reserved

print ("0x10 = ", hex(ipPS.read(0x10)))
print ("0x14 = ", hex(ipPS.read(0x14)))
print ("0x1c = ", hex(ipPS.read(0x1c)))
print ("0x20 = ", hex(ipPS.read(0x20)))
print ("0x34 = ", hex(ipPS.read(0x34)))
print ("0x38 = ", hex(ipPS.read(0x38)))

```

```
0x10 =  0x0
0x14 =  0x0
0x1c =  0x8
0x20 =  0x0
0x34 =  0xfffffff7
0x38 =  0x3f
```

```python
# Check MPRJ_IO input/out/en
# 0x10 : Data signal of ps_mprj_in
#        bit 31~0 - ps_mprj_in[31:0] (Read/Write)
# 0x14 : Data signal of ps_mprj_in
#        bit 5~0 - ps_mprj_in[37:32] (Read/Write)
#        others  - reserved
# 0x1c : Data signal of ps_mprj_out
#        bit 31~0 - ps_mprj_out[31:0] (Read)
# 0x20 : Data signal of ps_mprj_out
#        bit 5~0 - ps_mprj_out[37:32] (Read)
#        others  - reserved
# 0x34 : Data signal of ps_mprj_en
#        bit 31~0 - ps_mprj_en[31:0] (Read)
# 0x38 : Data signal of ps_mprj_en
#        bit 5~0 - ps_mprj_en[37:32] (Read)
#        others  - reserved

print ("0x10 = ", hex(ipPS.read(0x10)))
print ("0x14 = ", hex(ipPS.read(0x14)))
print ("0x1c = ", hex(ipPS.read(0x1c)))
print ("0x20 = ", hex(ipPS.read(0x20)))
print ("0x34 = ", hex(ipPS.read(0x34)))
print ("0x38 = ", hex(ipPS.read(0x38)))
```

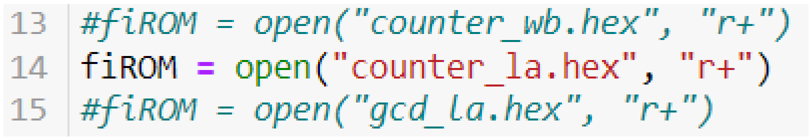
```
0x10 =  0x0
0x14 =  0x0
0x1c =  0xab610008
0x20 =  0x2
0x34 =  0xfff7
0x38 =  0x37
```



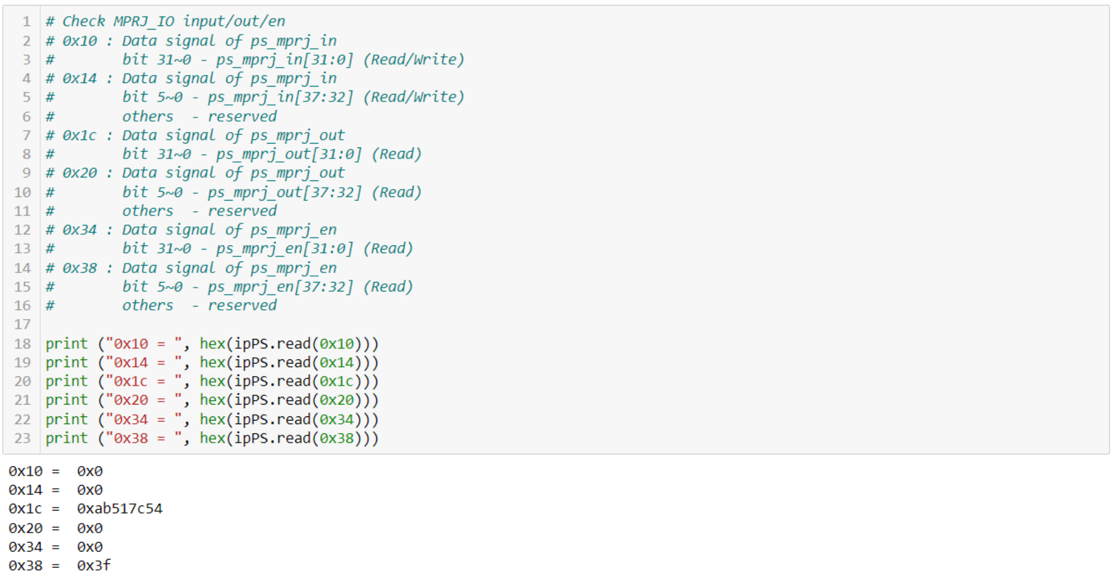
```
    /* Apply configuration */
    reg_mprj_xfer = 1;
    while (reg_mprj_xfer == 1);

    reg_la2_oenb = reg_la2_iena = 0x00000000;    // [95:64]

    // Flag start of the test
    reg_mprj_datal = 0xAB600000;

    reg_mprj_slave = 0x00002710;
    reg_mprj_datal = 0xAB610000;
    if (reg_mprj_slave == 0x2B3D) {
        reg_mprj_datal = 0xAB610000;
    }
```

配合 counter_wb.c 的內容，確實是從 0xAB610000 開始的。

Counter_la.hex:

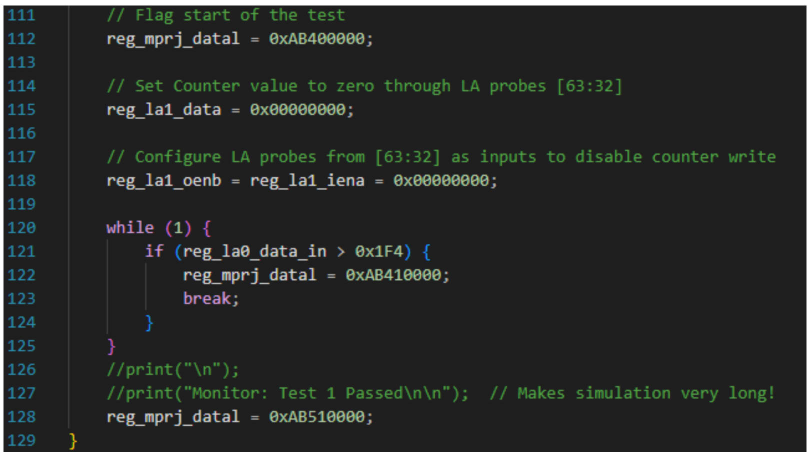


```
#fiROM = open("counter_wb.hex", "r+")
fiROM = open("counter_la.hex", "r+")
#fiROM = open("gcd_la.hex", "r+")
```



```
# Check MPRJ_IO input/out/en
# 0x10 : Data signal of ps_mprj_in
#        bit 31~0 - ps_mprj_in[31:0] (Read/Write)
# 0x14 : Data signal of ps_mprj_in
#        bit 5~0 - ps_mprj_in[37:32] (Read/Write)
#        others  - reserved
# 0x1c : Data signal of ps_mprj_out
#        bit 31~0 - ps_mprj_out[31:0] (Read)
# 0x20 : Data signal of ps_mprj_out
#        bit 5~0 - ps_mprj_out[37:32] (Read)
#        others  - reserved
# 0x34 : Data signal of ps_mprj_en
#        bit 31~0 - ps_mprj_en[31:0] (Read)
# 0x38 : Data signal of ps_mprj_en
#        bit 5~0 - ps_mprj_en[37:32] (Read)
#        others  - reserved

print ("0x10 = ", hex(ipPS.read(0x10)))
print ("0x14 = ", hex(ipPS.read(0x14)))
print ("0x1c = ", hex(ipPS.read(0x1c)))
print ("0x20 = ", hex(ipPS.read(0x20)))
print ("0x34 = ", hex(ipPS.read(0x34)))
print ("0x38 = ", hex(ipPS.read(0x38)))
```

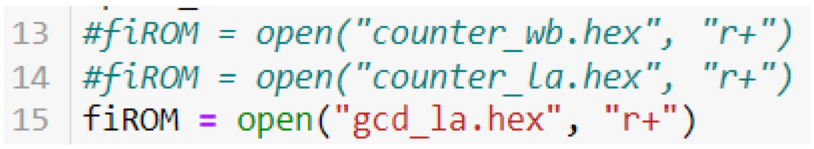
```
0x10 =  0x0
0x14 =  0x0
0x1c =  0xab517c54
0x20 =  0x0
0x34 =  0x0
0x38 =  0x3f
```



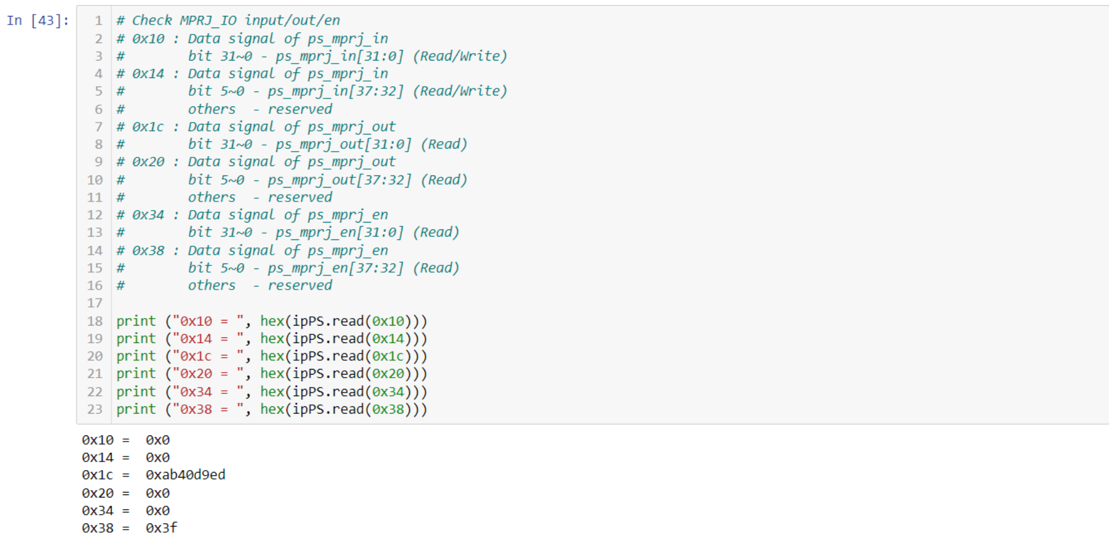
```
    // Flag start of the test
    reg_mprj_datal = 0xAB400000;

    // Set Counter value to zero through LA probes [63:32]
    reg_la1_data = 0x00000000;

    // Configure LA probes from [63:32] as inputs to disable counter write
    reg_la1_oenb = reg_la1_iena = 0x00000000;

    while (1) {
        if (reg_la0_data_in > 0x1F4) {
            reg_mprj_datal = 0xAB410000;
            break;
        }
    }
    //print("\n");
    //print("Monitor: Test 1 Passed\n\n");  // Makes simulation very long!
    reg_mprj_datal = 0xAB510000;
}
```

輸入 DATA 確實是從 AB510000 開始

GCD_la.hex：



```python
#fiROM = open("counter_wb.hex", "r+")
#fiROM = open("counter_la.hex", "r+")
fiROM = open("gcd_la.hex", "r+")
```



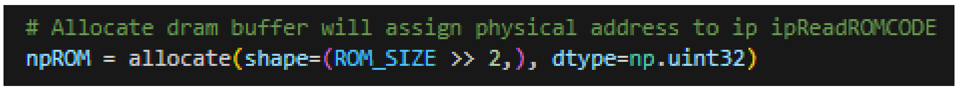
In [43]:

```python
# Check MPRJ_IO input/out/en
# 0x10 : Data signal of ps_mprj_in
#        bit 31~0 - ps_mprj_in[31:0] (Read/Write)
# 0x14 : Data signal of ps_mprj_in
#        bit 5~0 - ps_mprj_in[37:32] (Read/Write)
#        others  - reserved
# 0x1c : Data signal of ps_mprj_out
#        bit 31~0 - ps_mprj_out[31:0] (Read)
# 0x20 : Data signal of ps_mprj_out
#        bit 5~0 - ps_mprj_out[37:32] (Read)
#        others  - reserved
# 0x34 : Data signal of ps_mprj_en
#        bit 31~0 - ps_mprj_en[31:0] (Read)
# 0x38 : Data signal of ps_mprj_en
#        bit 5~0 - ps_mprj_en[37:32] (Read)
#        others  - reserved

print ("0x10 = ", hex(ipPS.read(0x10)))
print ("0x14 = ", hex(ipPS.read(0x14)))
print ("0x1c = ", hex(ipPS.read(0x1c)))
print ("0x20 = ", hex(ipPS.read(0x20)))
print ("0x34 = ", hex(ipPS.read(0x34)))
print ("0x38 = ", hex(ipPS.read(0x38)))
```

```
0x10 =  0x0
0x14 =  0x0
0x1c =  0xab40d9ed
0x20 =  0x0
0x34 =  0x0
0x38 =  0x3f
```

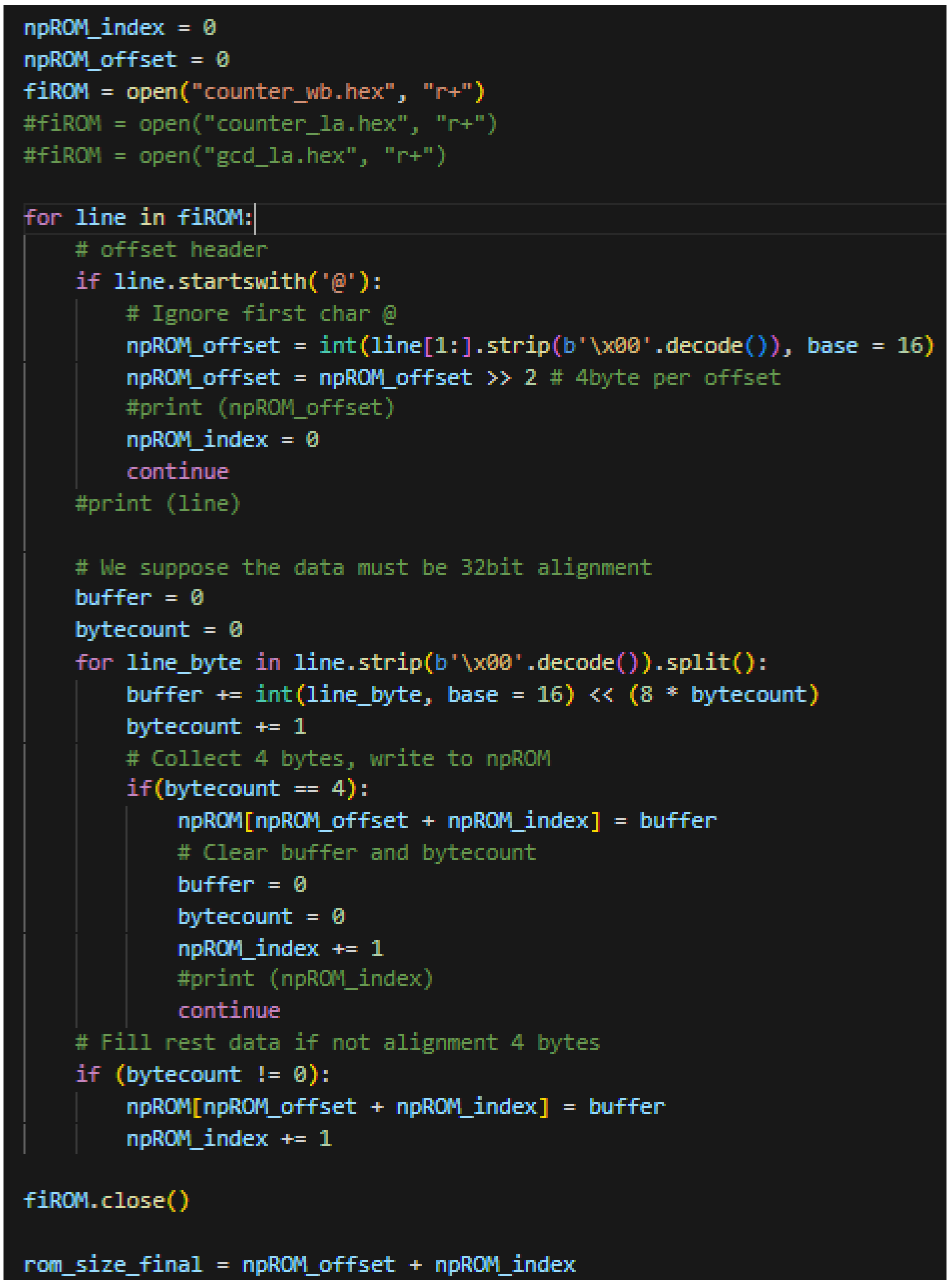
Study caravel_fpga.ipynb, and be familiar with caravel SoC control flow:

1. 先宣告一些 DDR 的 size(8K)給我們的.hex firmware code



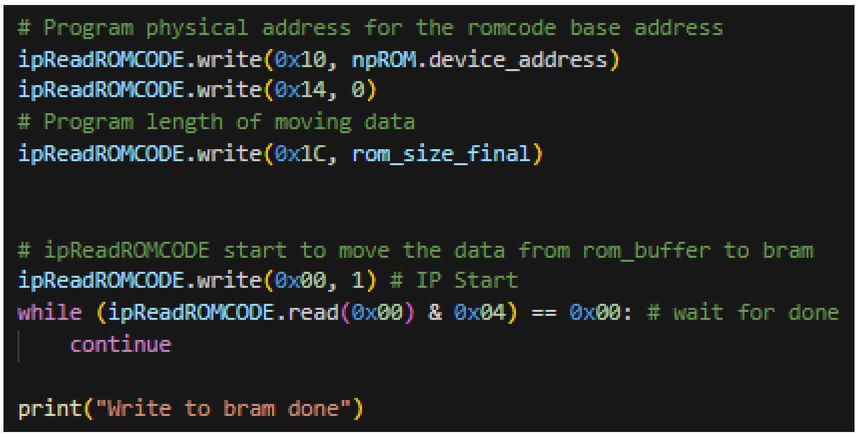
```python
# Allocate dram buffer will assign physical address to ip ipReadROMCODE
npROM = allocate(shape=(ROM_SIZE >> 2,), dtype=np.uint32)
```

2. 將.hex 檔案讀進 fiROM，並 parse 這個檔案將對應的資料放進 npROM 也就是真正的 dram 內部



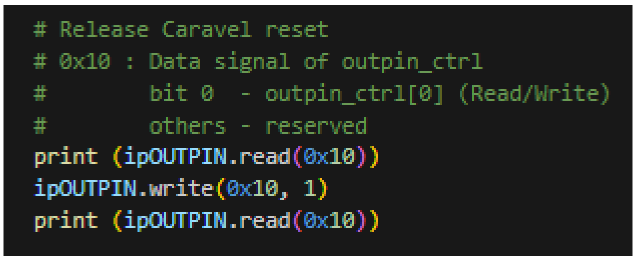
```python
npROM_index = 0
npROM_offset = 0
fiROM = open("counter_wb.hex", "r+")
#fiROM = open("counter_la.hex", "r+")
#fiROM = open("gcd_la.hex", "r+")

for line in fiROM:
    # offset header
    if line.startswith('@'):
        # Ignore first char @
        npROM_offset = int(line[1:].strip(b'\x00'.decode()), base = 16)
        npROM_offset = npROM_offset >> 2 # 4byte per offset
        #print (npROM_offset)
        npROM_index = 0
        continue
    #print (line)

    # We suppose the data must be 32bit alignment
    buffer = 0
    bytecount = 0
    for line_byte in line.strip(b'\x00'.decode()).split():
        buffer += int(line_byte, base = 16) << (8 * bytecount)
        bytecount += 1
        # Collect 4 bytes, write to npROM
        if(bytecount == 4):
            npROM[npROM_offset + npROM_index] = buffer
            # Clear buffer and bytecount
            buffer = 0
            bytecount = 0
            npROM_index += 1
            #print (npROM_index)
            continue
    # Fill rest data if not alignment 4 bytes
    if (bytecount != 0):
        npROM[npROM_offset + npROM_index] = buffer
        npROM_index += 1

fiROM.close()

rom_size_final = npROM_offset + npROM_index
```

3. 藉由 hls 產生出來的 IP 去將 DDR 內部 bram



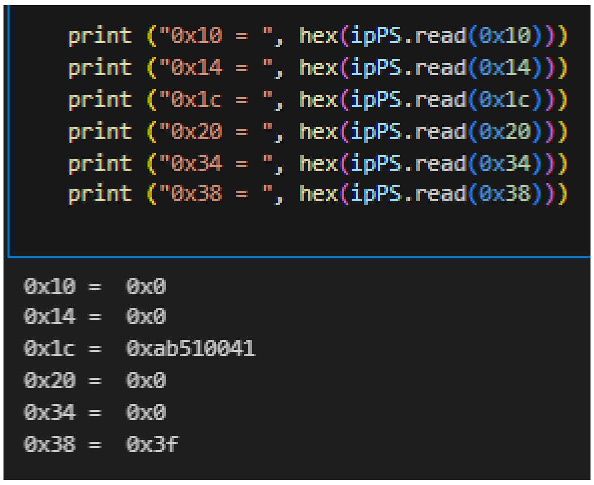
```
# Program physical address for the romcode base address
ipReadROMCODE.write(0x10, npROM.device_address)
ipReadROMCODE.write(0x14, 0)
# Program length of moving data
ipReadROMCODE.write(0x1C, rom_size_final)


# ipReadROMCODE start to move the data from rom_buffer to bram
ipReadROMCODE.write(0x00, 1) # IP Start
while (ipReadROMCODE.read(0x00) & 0x04) == 0x00: # wait for done
    continue

print("Write to bram done")
```

4. De-assert



```
# Release Caravel reset
# 0x10 : Data signal of outpin_ctrl
#        bit 0  - outpin_ctrl[0] (Read/Write)
#        others - reserved
print (ipOUTPIN.read(0x10))
ipOUTPIN.write(0x10, 1)
print (ipOUTPIN.read(0x10))
```

5. 讀出最終的 output pin



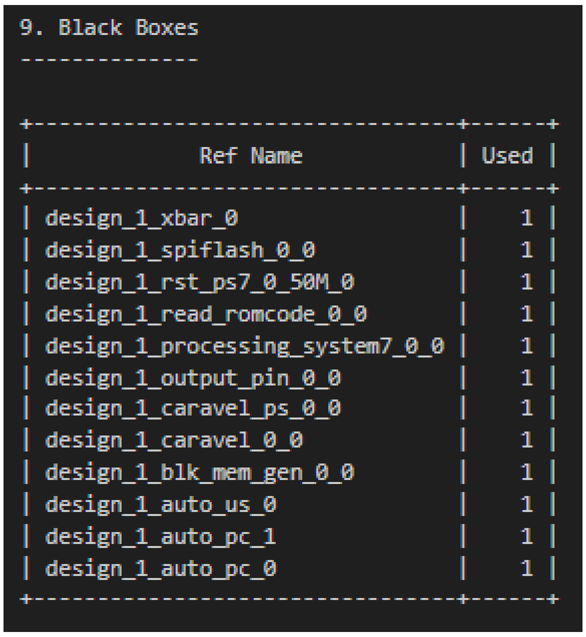
```
    print ("0x10 = ", hex(ipPS.read(0x10)))
    print ("0x14 = ", hex(ipPS.read(0x14)))
    print ("0x1c = ", hex(ipPS.read(0x1c)))
    print ("0x20 = ", hex(ipPS.read(0x20)))
    print ("0x34 = ", hex(ipPS.read(0x34)))
    print ("0x38 = ", hex(ipPS.read(0x38)))
```

```
0x10 =  0x0
0x14 =  0x0
0x1c =  0xab510041
0x20 =  0x0
0x34 =  0x0
0x38 =  0x3f
```

去看原本的.c firmware code 確實最終的 mprj data 就是 0xab51

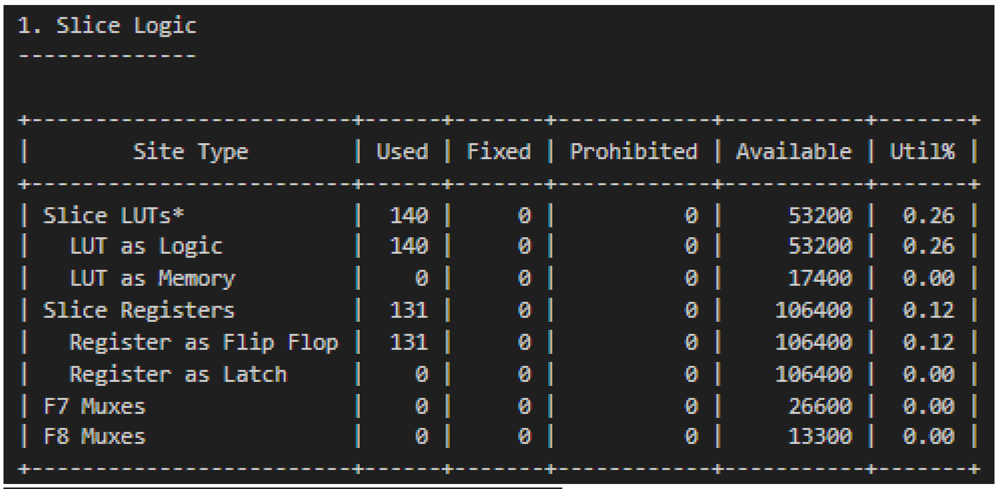
FPGA utilization:

Overall design wrapper utilization synthesis:

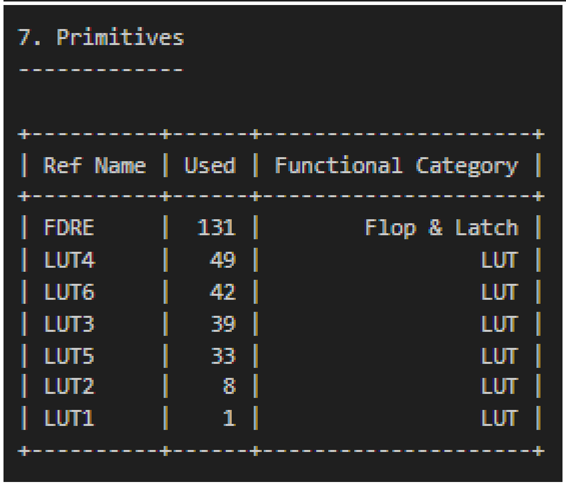


```
9. Black Boxes
--------------

+---------------------------------+------+
|             Ref Name            | Used |
+---------------------------------+------+
| design_1_xbar_0                 |    1 |
| design_1_spiflash_0_0           |    1 |
| design_1_rst_ps7_0_50M_0        |    1 |
| design_1_read_romcode_0_0       |    1 |
| design_1_processing_system7_0_0 |    1 |
| design_1_output_pin_0_0         |    1 |
| design_1_caravel_ps_0_0         |    1 |
| design_1_caravel_0_0            |    1 |
| design_1_blk_mem_gen_0_0        |    1 |
| design_1_auto_us_0              |    1 |
| design_1_auto_pc_1              |    1 |
| design_1_auto_pc_0              |    1 |
+---------------------------------+------+
```

Design_1_xbar

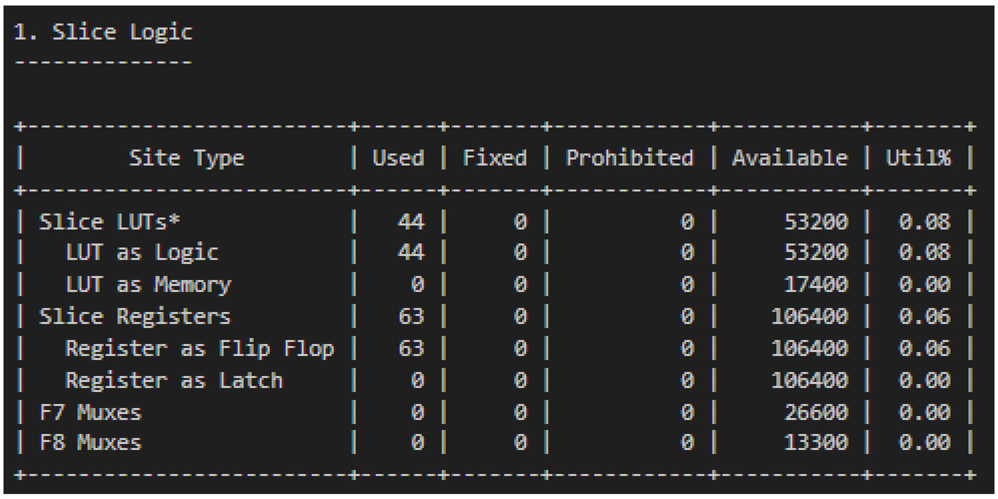


```
1. Slice Logic
--------------

+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |  140 |     0 |          0 |     53200 |  0.26 |
|   LUT as Logic          |  140 |     0 |          0 |     53200 |  0.26 |
|   LUT as Memory         |    0 |     0 |          0 |     17400 |  0.00 |
| Slice Registers         |  131 |     0 |          0 |    106400 |  0.12 |
|   Register as Flip Flop |  131 |     0 |          0 |    106400 |  0.12 |
|   Register as Latch     |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                |    0 |     0 |          0 |     13300 |  0.00 |
+-------------------------+------+-------+------------+-----------+-------+
```

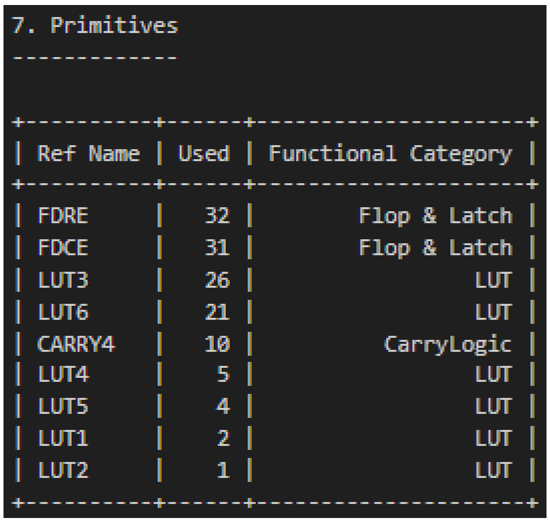


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |  131 |        Flop & Latch |
| LUT4     |   49 |                 LUT |
| LUT6     |   42 |                 LUT |
| LUT3     |   39 |                 LUT |
| LUT5     |   33 |                 LUT |
| LUT2     |    8 |                 LUT |
| LUT1     |    1 |                 LUT |
+----------+------+---------------------+
```

Design_1_splifash_0_0

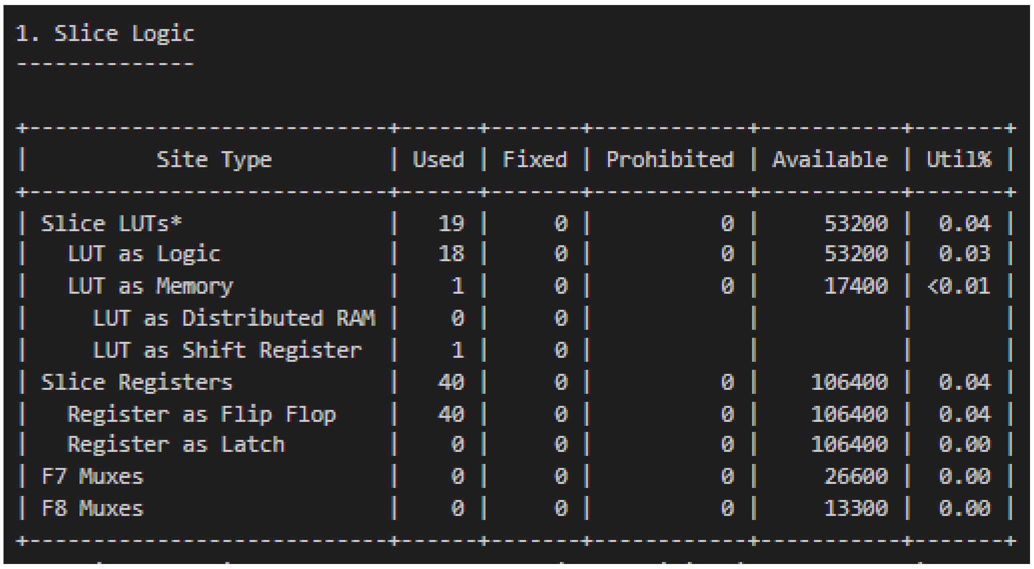


```
1. Slice Logic
--------------

+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |   44 |     0 |          0 |     53200 |  0.08 |
|   LUT as Logic          |   44 |     0 |          0 |     53200 |  0.08 |
|   LUT as Memory         |    0 |     0 |          0 |     17400 |  0.00 |
| Slice Registers         |   63 |     0 |          0 |    106400 |  0.06 |
|   Register as Flip Flop |   63 |     0 |          0 |    106400 |  0.06 |
|   Register as Latch     |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                |    0 |     0 |          0 |     13300 |  0.00 |
+-------------------------+------+-------+------------+-----------+-------+
```

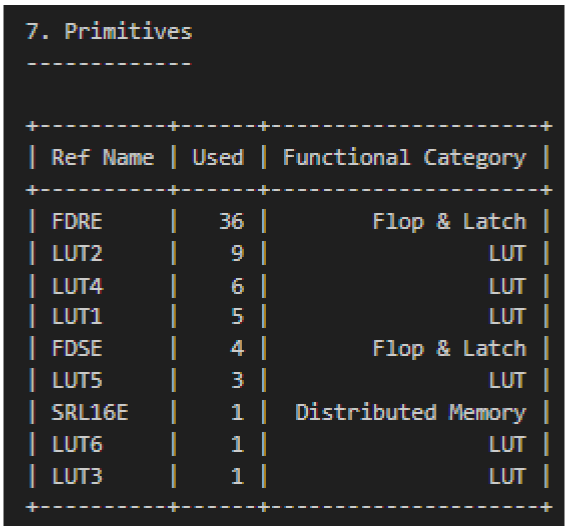


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |   32 |        Flop & Latch |
| FDCE     |   31 |        Flop & Latch |
| LUT3     |   26 |                 LUT |
| LUT6     |   21 |                 LUT |
| CARRY4   |   10 |          CarryLogic |
| LUT4     |    5 |                 LUT |
| LUT5     |    4 |                 LUT |
| LUT1     |    2 |                 LUT |
| LUT2     |    1 |                 LUT |
+----------+------+---------------------+
```

Design_1_rst_ps7_0_50M_0

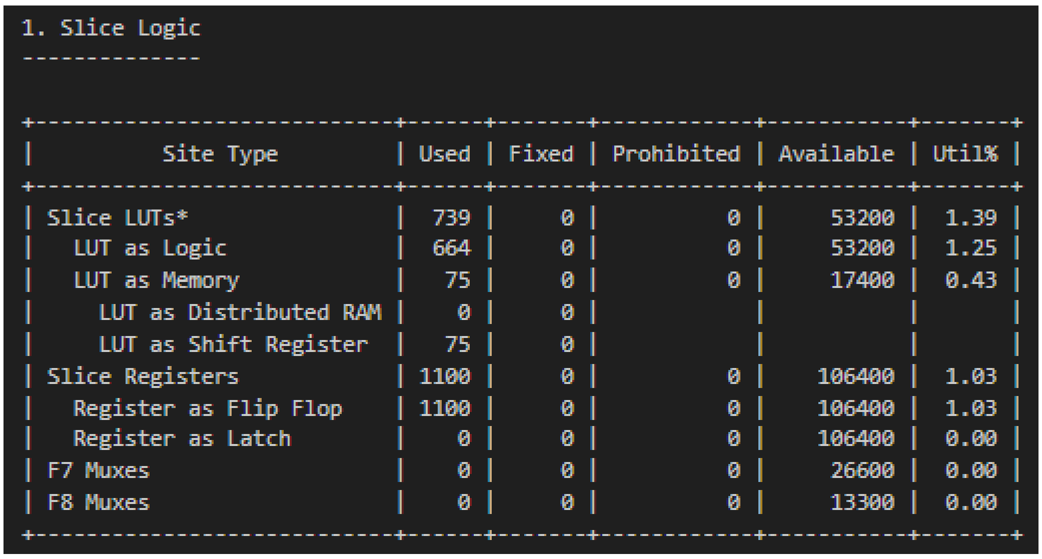


```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*                |   19 |     0 |          0 |     53200 |  0.04 |
|   LUT as Logic             |   18 |     0 |          0 |     53200 |  0.03 |
|   LUT as Memory            |    1 |     0 |          0 |     17400 | <0.01 |
|     LUT as Distributed RAM |    0 |     0 |            |           |       |
|     LUT as Shift Register  |    1 |     0 |            |           |       |
| Slice Registers            |   40 |     0 |          0 |    106400 |  0.04 |
|   Register as Flip Flop    |   40 |     0 |          0 |    106400 |  0.04 |
|   Register as Latch        |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                   |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                   |    0 |     0 |          0 |     13300 |  0.00 |
+----------------------------+------+-------+------------+-----------+-------+
```

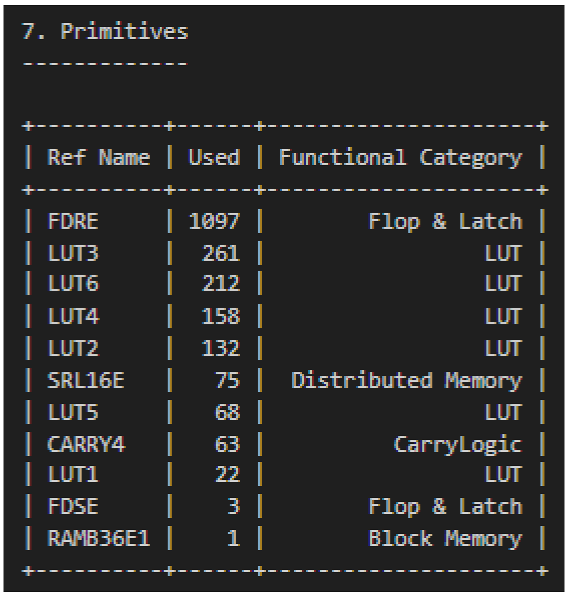


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |   36 |        Flop & Latch |
| LUT2     |    9 |                 LUT |
| LUT4     |    6 |                 LUT |
| LUT1     |    5 |                 LUT |
| FDSE     |    4 |        Flop & Latch |
| LUT5     |    3 |                 LUT |
| SRL16E   |    1 |  Distributed Memory |
| LUT6     |    1 |                 LUT |
| LUT3     |    1 |                 LUT |
+----------+------+---------------------+
```

Design_1_read_romcode_0_0

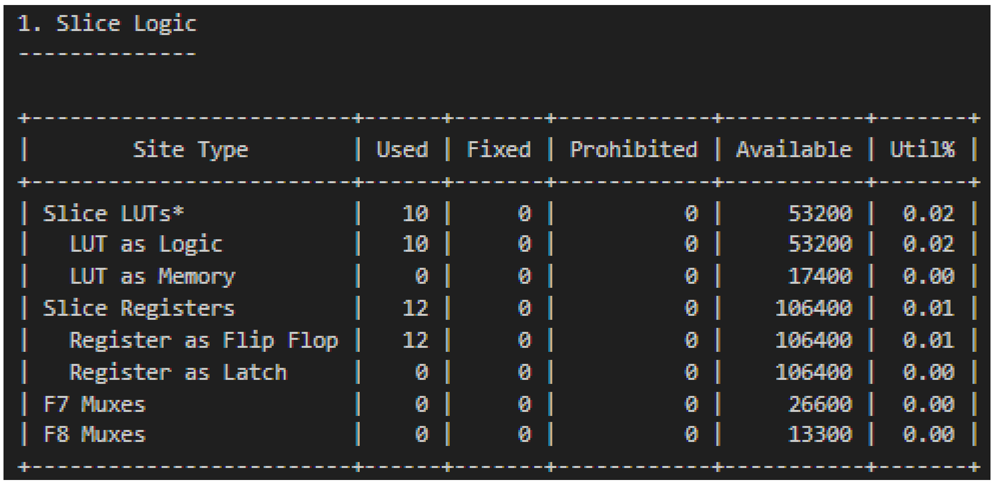


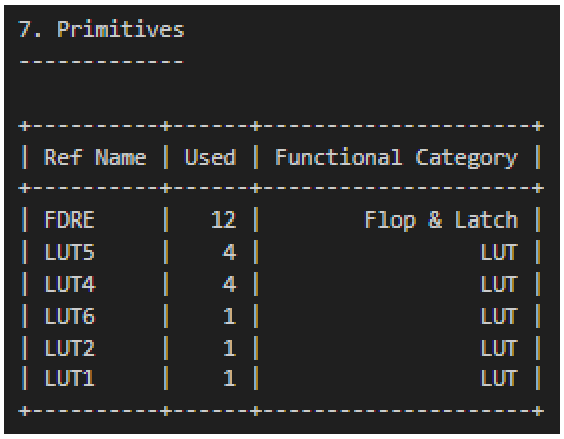
```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*                |  739 |     0 |          0 |     53200 |  1.39 |
|   LUT as Logic             |  664 |     0 |          0 |     53200 |  1.25 |
|   LUT as Memory            |   75 |     0 |          0 |     17400 |  0.43 |
|     LUT as Distributed RAM |    0 |     0 |            |           |       |
|     LUT as Shift Register  |   75 |     0 |            |           |       |
| Slice Registers            | 1100 |     0 |          0 |    106400 |  1.03 |
|   Register as Flip Flop    | 1100 |     0 |          0 |    106400 |  1.03 |
|   Register as Latch        |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                   |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                   |    0 |     0 |          0 |     13300 |  0.00 |
+----------------------------+------+-------+------------+-----------+-------+
```



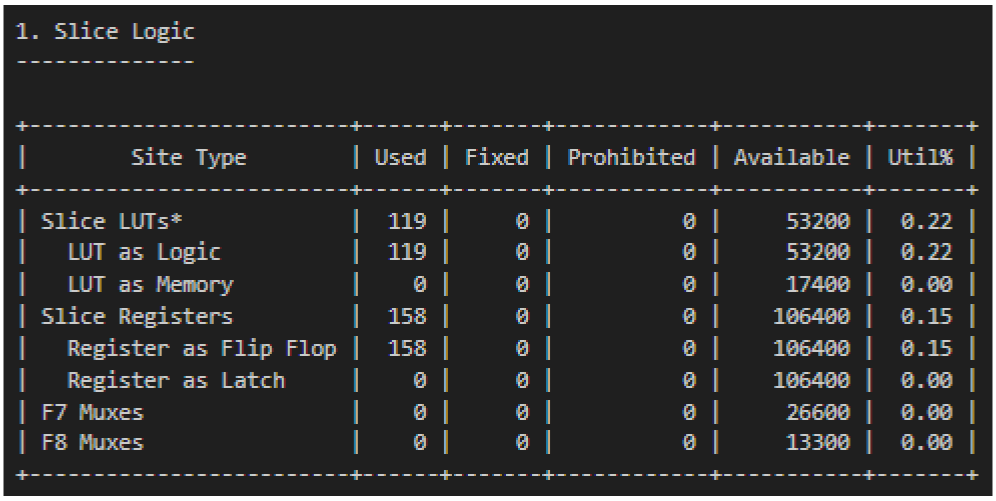
```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     | 1097 |        Flop & Latch |
| LUT3     |  261 |                 LUT |
| LUT6     |  212 |                 LUT |
| LUT4     |  158 |                 LUT |
| LUT2     |  132 |                 LUT |
| SRL16E   |   75 |  Distributed Memory |
| LUT5     |   68 |                 LUT |
| CARRY4   |   63 |          CarryLogic |
| LUT1     |   22 |                 LUT |
| FDSE     |    3 |        Flop & Latch |
| RAMB36E1 |    1 |        Block Memory |
+----------+------+---------------------+
```

Design_1_Output_pint_0_0

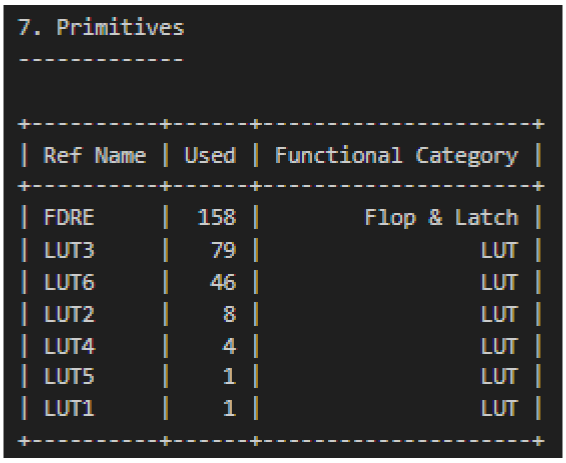

```
1. Slice Logic
--------------

+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |   10 |     0 |          0 |     53200 |  0.02 |
|   LUT as Logic          |   10 |     0 |          0 |     53200 |  0.02 |
|   LUT as Memory         |    0 |     0 |          0 |     17400 |  0.00 |
| Slice Registers         |   12 |     0 |          0 |    106400 |  0.01 |
|   Register as Flip Flop |   12 |     0 |          0 |    106400 |  0.01 |
|   Register as Latch     |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                |    0 |     0 |          0 |     13300 |  0.00 |
+-------------------------+------+-------+------------+-----------+-------+
```


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |   12 |        Flop & Latch |
| LUT5     |    4 |                 LUT |
| LUT4     |    4 |                 LUT |
| LUT6     |    1 |                 LUT |
| LUT2     |    1 |                 LUT |
| LUT1     |    1 |                 LUT |
+----------+------+---------------------+
```

Design_1_caravel_ps_0_0

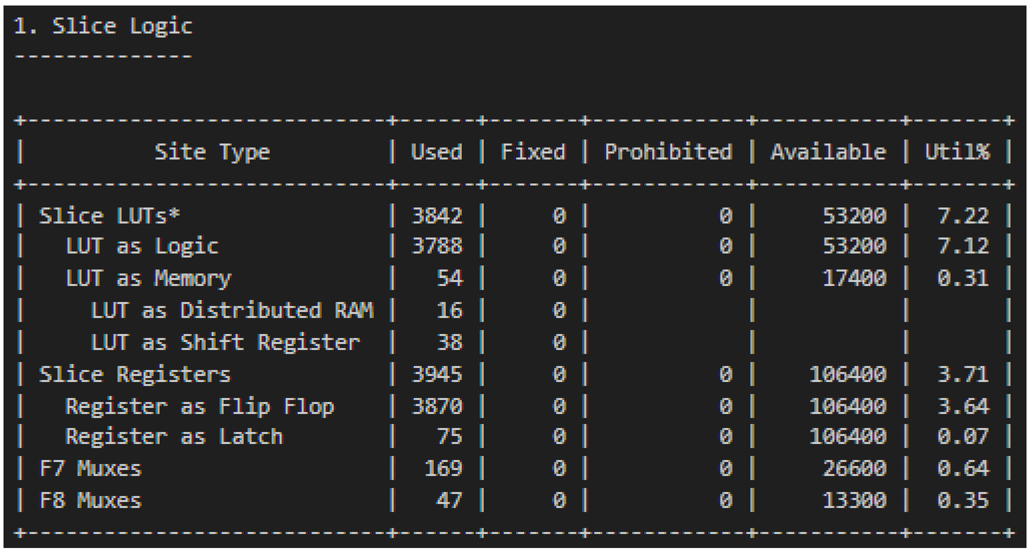


```
1. Slice Logic
--------------

+-------------------------+------+-------+------------+-----------+-------+
|        Site Type        | Used | Fixed | Prohibited | Available | Util% |
+-------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*             |  119 |     0 |          0 |     53200 |  0.22 |
|   LUT as Logic          |  119 |     0 |          0 |     53200 |  0.22 |
|   LUT as Memory         |    0 |     0 |          0 |     17400 |  0.00 |
| Slice Registers         |  158 |     0 |          0 |    106400 |  0.15 |
|   Register as Flip Flop |  158 |     0 |          0 |    106400 |  0.15 |
|   Register as Latch     |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                |    0 |     0 |          0 |     13300 |  0.00 |
+-------------------------+------+-------+------------+-----------+-------+
```

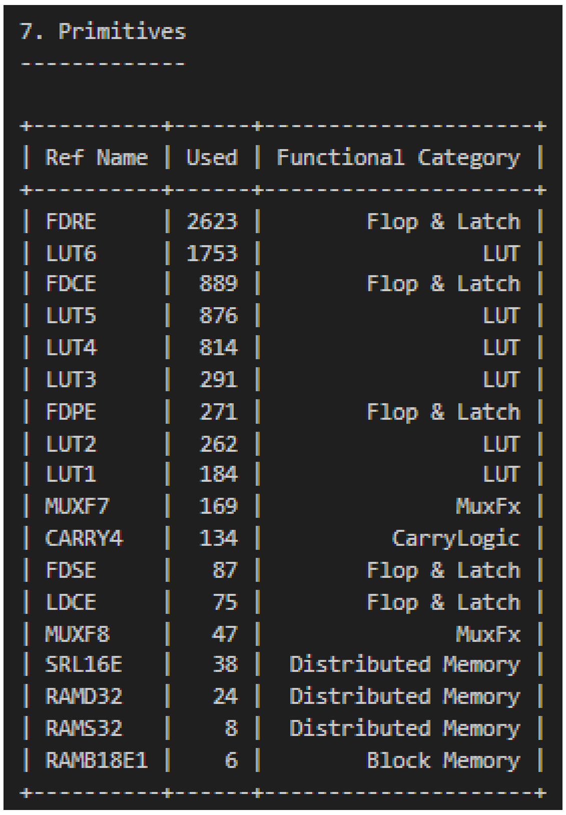


```
7. Primitives
------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |  158 |        Flop & Latch |
| LUT3     |   79 |                 LUT |
| LUT6     |   46 |                 LUT |
| LUT2     |    8 |                 LUT |
| LUT4     |    4 |                 LUT |
| LUT5     |    1 |                 LUT |
| LUT1     |    1 |                 LUT |
+----------+------+---------------------+
```

Design_1_caravel_0_0

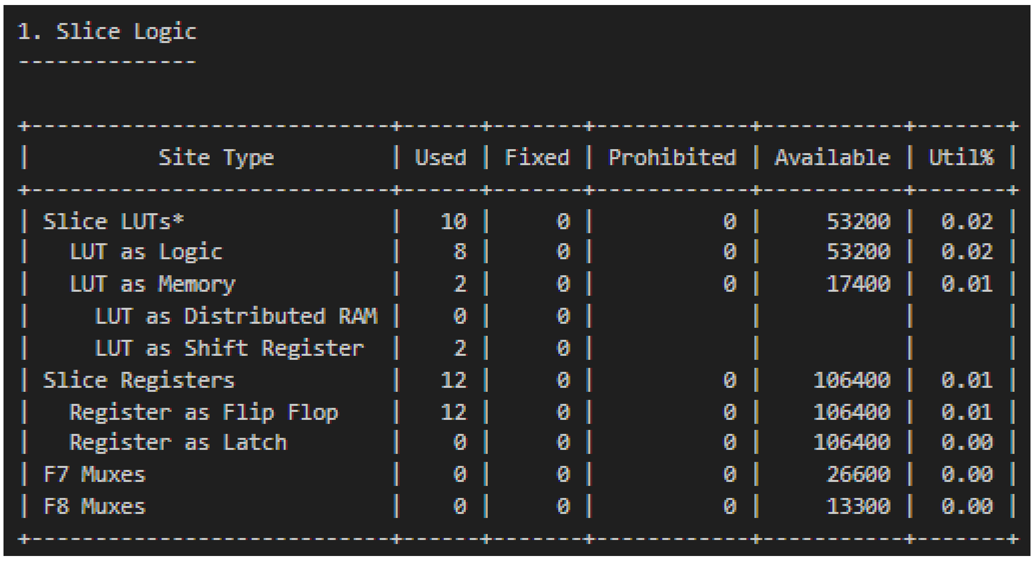


```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*                | 3842 |     0 |          0 |     53200 |  7.22 |
|   LUT as Logic             | 3788 |     0 |          0 |     53200 |  7.12 |
|   LUT as Memory            |   54 |     0 |          0 |     17400 |  0.31 |
|     LUT as Distributed RAM |   16 |     0 |            |           |       |
|     LUT as Shift Register  |   38 |     0 |            |           |       |
| Slice Registers            | 3945 |     0 |          0 |    106400 |  3.71 |
|   Register as Flip Flop    | 3870 |     0 |          0 |    106400 |  3.64 |
|   Register as Latch        |   75 |     0 |          0 |    106400 |  0.07 |
| F7 Muxes                   |  169 |     0 |          0 |     26600 |  0.64 |
| F8 Muxes                   |   47 |     0 |          0 |     13300 |  0.35 |
+----------------------------+------+-------+------------+-----------+-------+
```

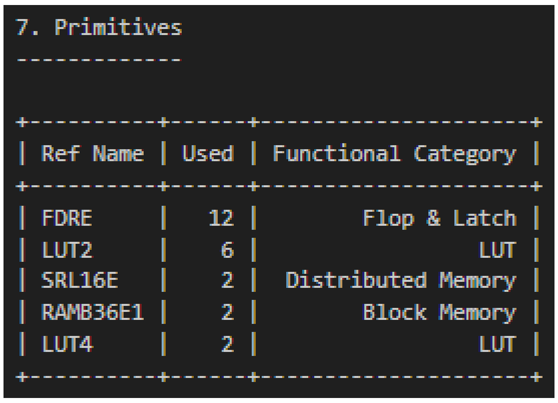


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     | 2623 |        Flop & Latch |
| LUT6     | 1753 |                 LUT |
| FDCE     |  889 |        Flop & Latch |
| LUT5     |  876 |                 LUT |
| LUT4     |  814 |                 LUT |
| LUT3     |  291 |                 LUT |
| FDPE     |  271 |        Flop & Latch |
| LUT2     |  262 |                 LUT |
| LUT1     |  184 |                 LUT |
| MUXF7    |  169 |               MuxFx |
| CARRY4   |  134 |          CarryLogic |
| FDSE     |   87 |        Flop & Latch |
| LDCE     |   75 |        Flop & Latch |
| MUXF8    |   47 |               MuxFx |
| SRL16E   |   38 |  Distributed Memory |
| RAMD32   |   24 |  Distributed Memory |
| RAMS32   |    8 |  Distributed Memory |
| RAMB18E1 |    6 |        Block Memory |
+----------+------+---------------------+
```

Design_1_blk_mem_gen_0_0

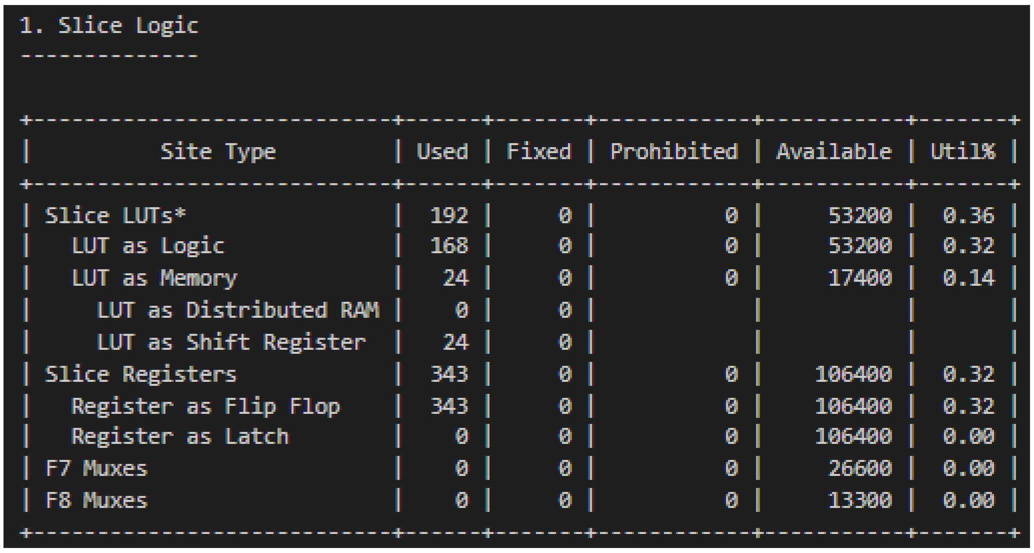


```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*                |   10 |     0 |          0 |     53200 |  0.02 |
|   LUT as Logic             |    8 |     0 |          0 |     53200 |  0.02 |
|   LUT as Memory            |    2 |     0 |          0 |     17400 |  0.01 |
|     LUT as Distributed RAM |    0 |     0 |            |           |       |
|     LUT as Shift Register  |    2 |     0 |            |           |       |
| Slice Registers            |   12 |     0 |          0 |    106400 |  0.01 |
|   Register as Flip Flop    |   12 |     0 |          0 |    106400 |  0.01 |
|   Register as Latch        |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                   |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                   |    0 |     0 |          0 |     13300 |  0.00 |
+----------------------------+------+-------+------------+-----------+-------+
```

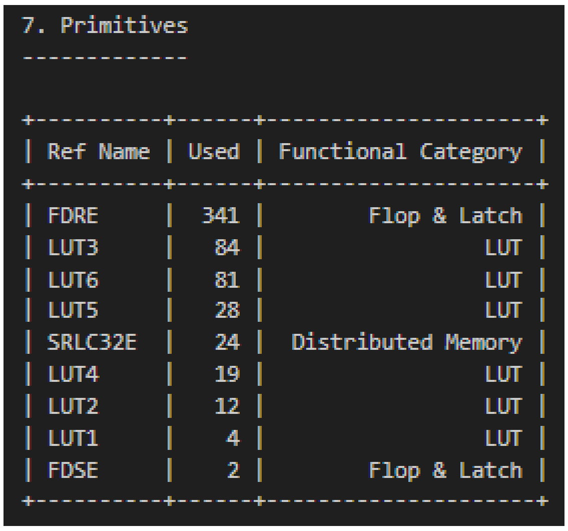


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |   12 |        Flop & Latch |
| LUT2     |    6 |                 LUT |
| SRL16E   |    2 |  Distributed Memory |
| RAMB36E1 |    2 |        Block Memory |
| LUT4     |    2 |                 LUT |
+----------+------+---------------------+
```

Design_1_auto_us_0

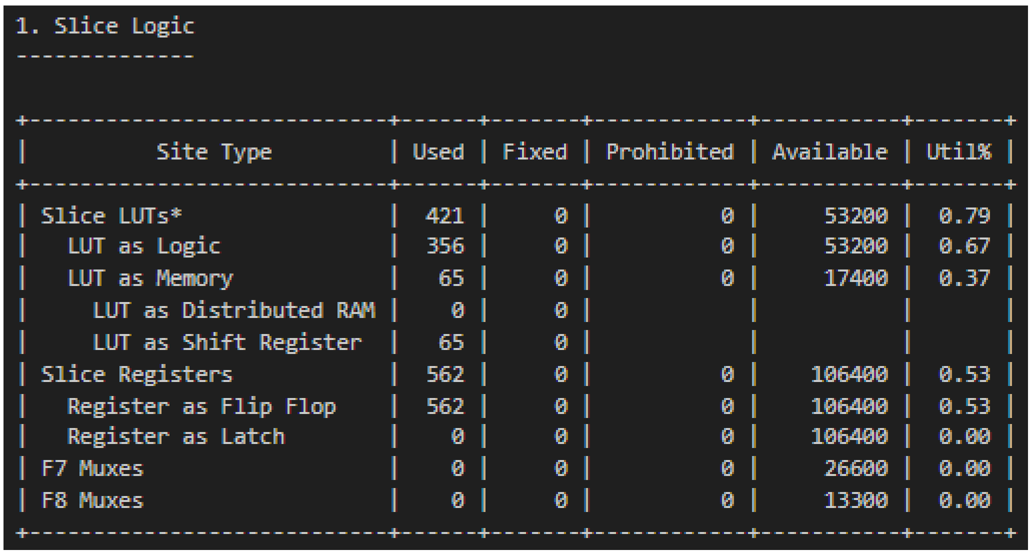


```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*                |  192 |     0 |          0 |     53200 |  0.36 |
|   LUT as Logic             |  168 |     0 |          0 |     53200 |  0.32 |
|   LUT as Memory            |   24 |     0 |          0 |     17400 |  0.14 |
|     LUT as Distributed RAM |    0 |     0 |            |           |       |
|     LUT as Shift Register  |   24 |     0 |            |           |       |
| Slice Registers            |  343 |     0 |          0 |    106400 |  0.32 |
|   Register as Flip Flop    |  343 |     0 |          0 |    106400 |  0.32 |
|   Register as Latch        |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                   |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                   |    0 |     0 |          0 |     13300 |  0.00 |
+----------------------------+------+-------+------------+-----------+-------+
```

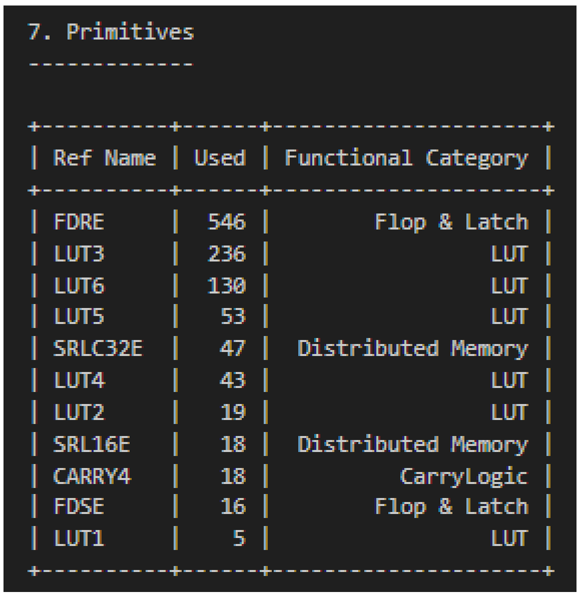


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |  341 |        Flop & Latch |
| LUT3     |   84 |                 LUT |
| LUT6     |   81 |                 LUT |
| LUT5     |   28 |                 LUT |
| SRLC32E  |   24 |  Distributed Memory |
| LUT4     |   19 |                 LUT |
| LUT2     |   12 |                 LUT |
| LUT1     |    4 |                 LUT |
| FDSE     |    2 |        Flop & Latch |
+----------+------+---------------------+
```

Design_1_auto_pc_1

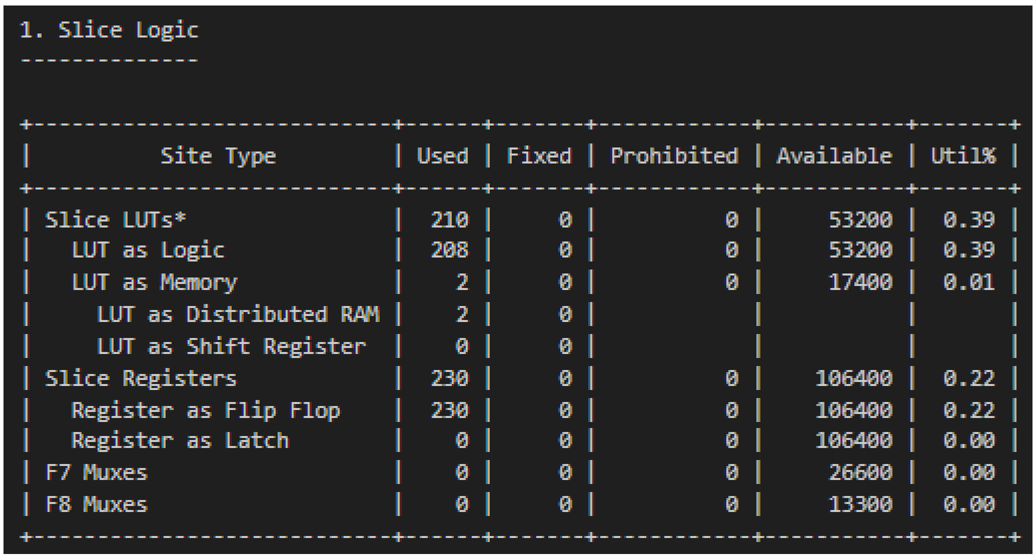


```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*                |  421 |     0 |          0 |     53200 |  0.79 |
|   LUT as Logic             |  356 |     0 |          0 |     53200 |  0.67 |
|   LUT as Memory            |   65 |     0 |          0 |     17400 |  0.37 |
|     LUT as Distributed RAM |    0 |     0 |            |           |       |
|     LUT as Shift Register  |   65 |     0 |            |           |       |
| Slice Registers            |  562 |     0 |          0 |    106400 |  0.53 |
|   Register as Flip Flop    |  562 |     0 |          0 |    106400 |  0.53 |
|   Register as Latch        |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                   |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                   |    0 |     0 |          0 |     13300 |  0.00 |
+----------------------------+------+-------+------------+-----------+-------+
```

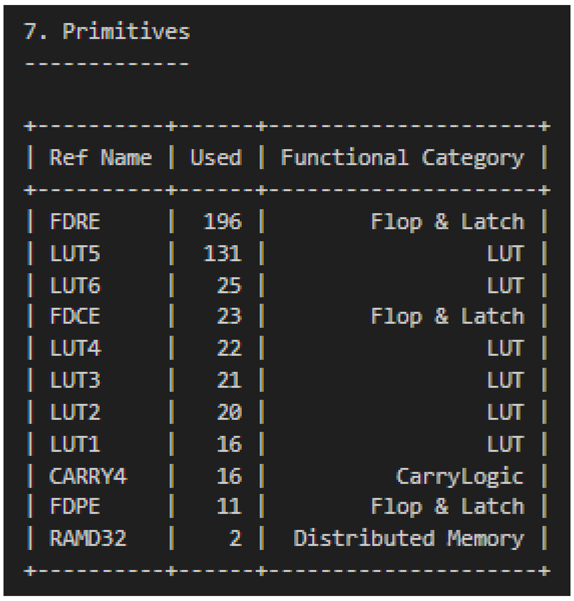


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |  546 |        Flop & Latch |
| LUT3     |  236 |                 LUT |
| LUT6     |  130 |                 LUT |
| LUT5     |   53 |                 LUT |
| SRLC32E  |   47 |  Distributed Memory |
| LUT4     |   43 |                 LUT |
| LUT2     |   19 |                 LUT |
| SRL16E   |   18 |  Distributed Memory |
| CARRY4   |   18 |          CarryLogic |
| FDSE     |   16 |        Flop & Latch |
| LUT1     |    5 |                 LUT |
+----------+------+---------------------+
```

Design_1_auto_pc_0

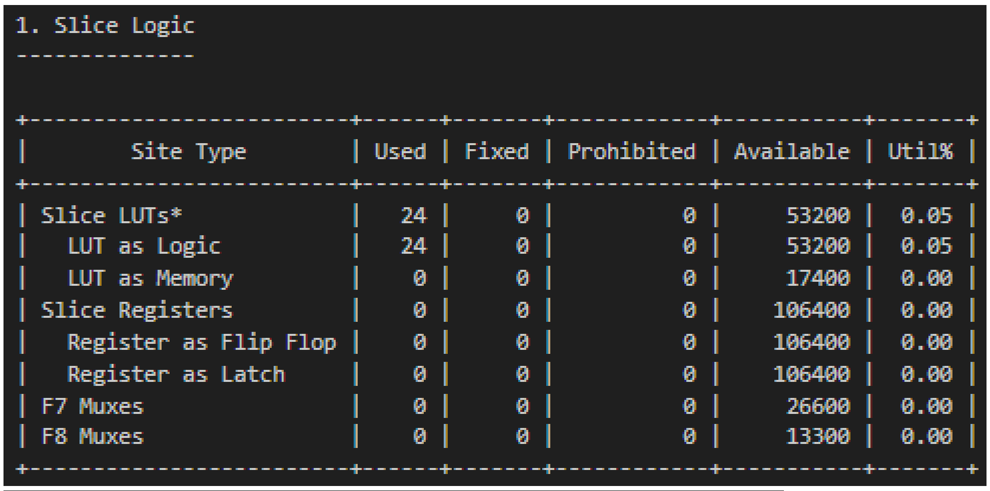


```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*                |  210 |     0 |          0 |     53200 |  0.39 |
|   LUT as Logic             |  208 |     0 |          0 |     53200 |  0.39 |
|   LUT as Memory            |    2 |     0 |          0 |     17400 |  0.01 |
|     LUT as Distributed RAM |    2 |     0 |            |           |       |
|     LUT as Shift Register  |    0 |     0 |            |           |       |
| Slice Registers            |  230 |     0 |          0 |    106400 |  0.22 |
|   Register as Flip Flop    |  230 |     0 |          0 |    106400 |  0.22 |
|   Register as Latch        |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                   |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                   |    0 |     0 |          0 |     13300 |  0.00 |
+----------------------------+------+-------+------------+-----------+-------+
```

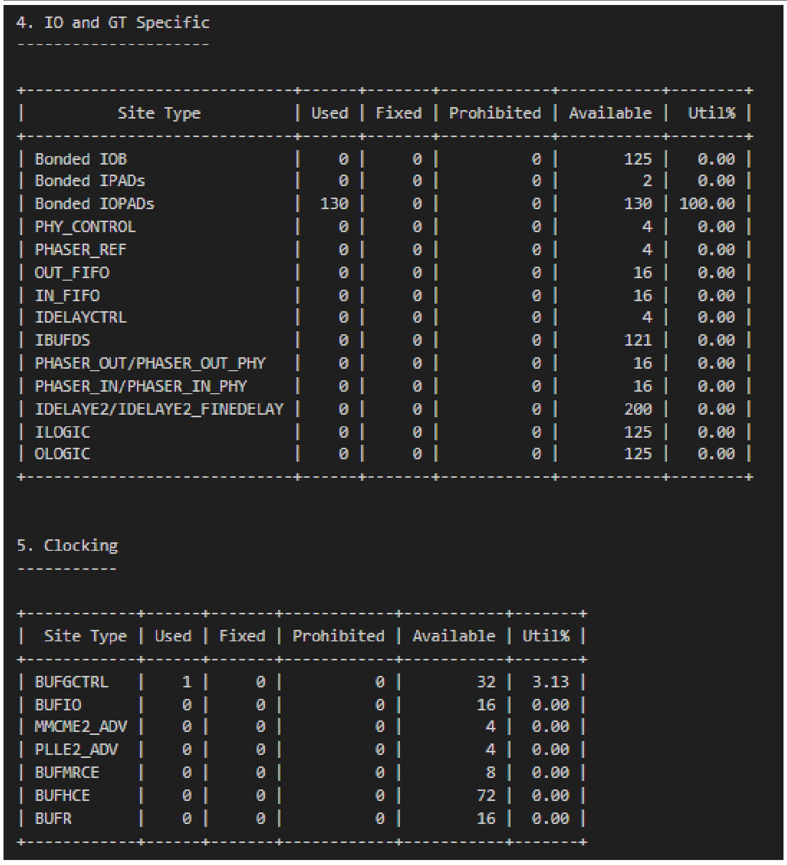


```
7. Primitives
-------------

+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     |  196 |        Flop & Latch |
| LUT5     |  131 |                 LUT |
| LUT6     |   25 |                 LUT |
| FDCE     |   23 |        Flop & Latch |
| LUT4     |   22 |                 LUT |
| LUT3     |   21 |                 LUT |
| LUT2     |   20 |                 LUT |
| LUT1     |   16 |                 LUT |
| CARRY4   |   16 |          CarryLogic |
| FDPE     |   11 |        Flop & Latch |
| RAMD32   |    2 |  Distributed Memory |
+----------+------+---------------------+
```

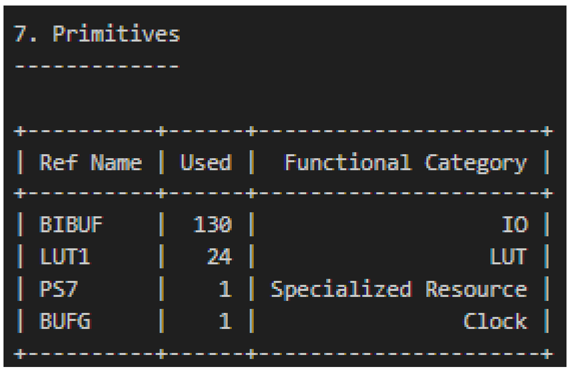
Design_1_processing_system



```
1. Slice Logic
--------------

+--------------------------+------+-------+------------+-----------+-------+
|        Site Type         | Used | Fixed | Prohibited | Available | Util% |
+--------------------------+------+-------+------------+-----------+-------+
| Slice LUTs*              |   24 |     0 |          0 |     53200 |  0.05 |
|   LUT as Logic           |   24 |     0 |          0 |     53200 |  0.05 |
|   LUT as Memory          |    0 |     0 |          0 |     17400 |  0.00 |
| Slice Registers          |    0 |     0 |          0 |    106400 |  0.00 |
|   Register as Flip Flop  |    0 |     0 |          0 |    106400 |  0.00 |
|   Register as Latch      |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                 |    0 |     0 |          0 |     26600 |  0.00 |
| F8 Muxes                 |    0 |     0 |          0 |     13300 |  0.00 |
+--------------------------+------+-------+------------+-----------+-------+
```

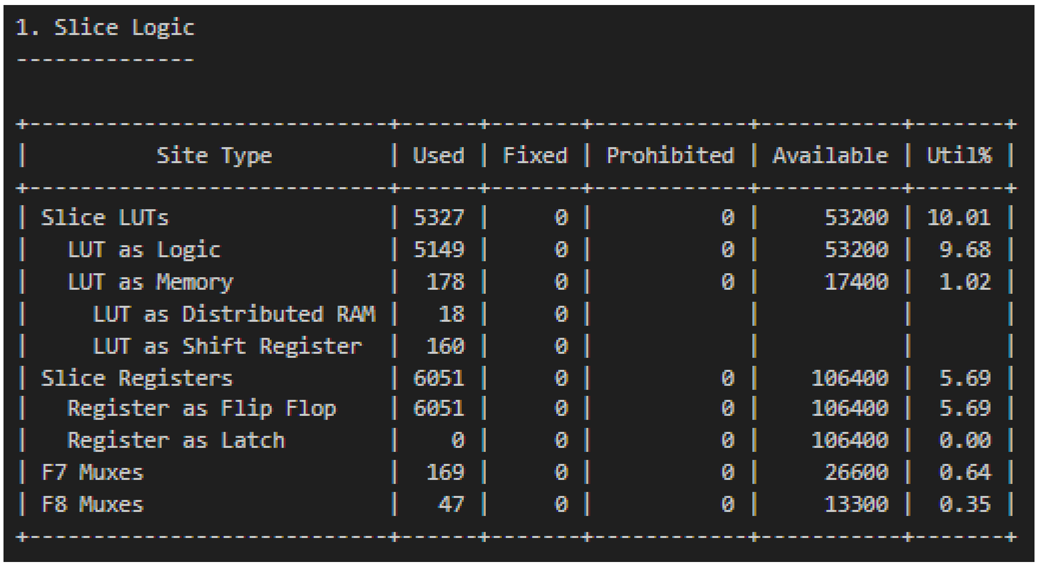


```
4. IO and GT Specific
--------------------

+-----------------------------+------+-------+------------+-----------+--------+
|          Site Type          | Used | Fixed | Prohibited | Available | Util%  |
+-----------------------------+------+-------+------------+-----------+--------+
| Bonded IOB                  |    0 |     0 |          0 |       125 |   0.00 |
| Bonded IPADs                |    0 |     0 |          0 |         2 |   0.00 |
| Bonded IOPADs               |  130 |     0 |          0 |       130 | 100.00 |
| PHY_CONTROL                 |    0 |     0 |          0 |         4 |   0.00 |
| PHASER_REF                  |    0 |     0 |          0 |         4 |   0.00 |
| OUT_FIFO                    |    0 |     0 |          0 |        16 |   0.00 |
| IN_FIFO                     |    0 |     0 |          0 |        16 |   0.00 |
| IDELAYCTRL                  |    0 |     0 |          0 |         4 |   0.00 |
| IBUFDS                      |    0 |     0 |          0 |       121 |   0.00 |
| PHASER_OUT/PHASER_OUT_PHY   |    0 |     0 |          0 |        16 |   0.00 |
| PHASER_IN/PHASER_IN_PHY     |    0 |     0 |          0 |        16 |   0.00 |
| IDELAYE2/IDELAYE2_FINEDELAY |    0 |     0 |          0 |       200 |   0.00 |
| ILOGIC                      |    0 |     0 |          0 |       125 |   0.00 |
| OLOGIC                      |    0 |     0 |          0 |       125 |   0.00 |
+-----------------------------+------+-------+------------+-----------+--------+


5. Clocking
----------

+------------+------+-------+------------+-----------+-------+
|  Site Type | Used | Fixed | Prohibited | Available | Util% |
+------------+------+-------+------------+-----------+-------+
| BUFGCTRL   |    1 |     0 |          0 |        32 |  3.13 |
| BUFIO      |    0 |     0 |          0 |        16 |  0.00 |
| MMCME2_ADV |    0 |     0 |          0 |         4 |  0.00 |
| PLLE2_ADV  |    0 |     0 |          0 |         4 |  0.00 |
| BUFMRCE    |    0 |     0 |          0 |         8 |  0.00 |
| BUFHCE     |    0 |     0 |          0 |        72 |  0.00 |
| BUFR       |    0 |     0 |          0 |        16 |  0.00 |
+------------+------+-------+------------+-----------+-------+
```

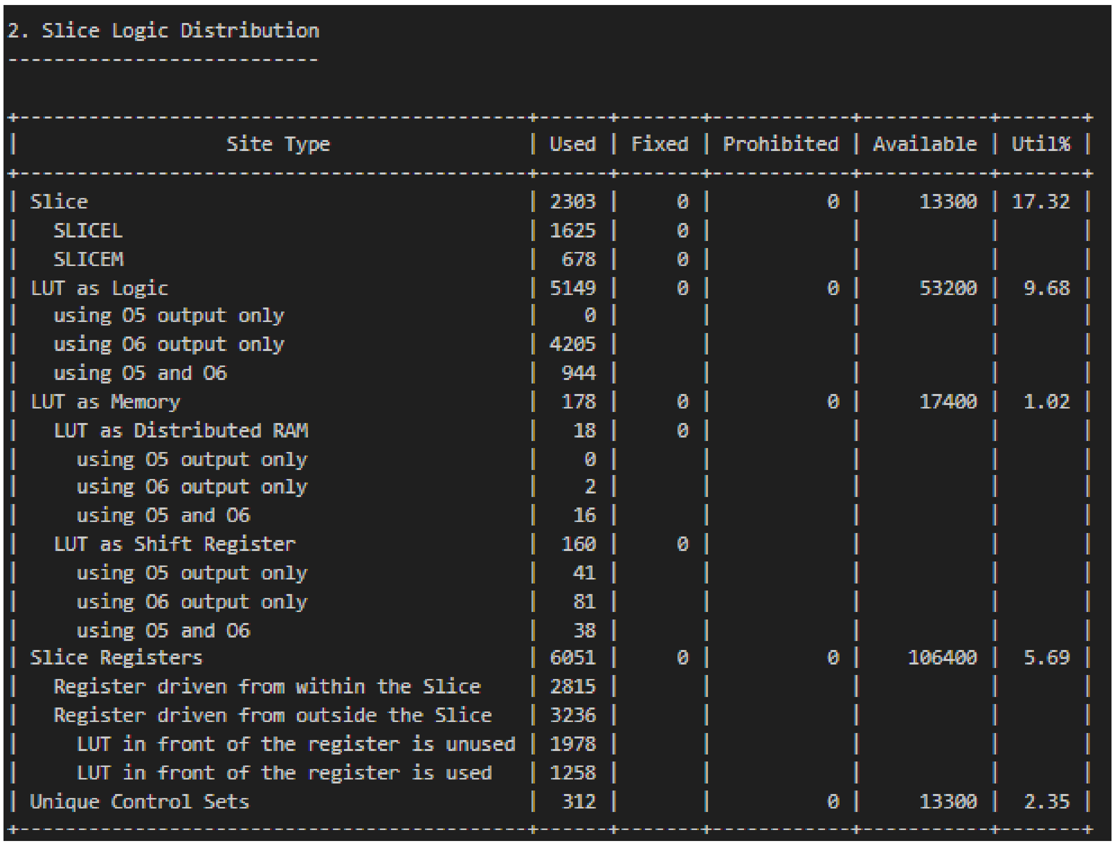


```
7. Primitives
-------------

+----------+------+----------------------+
| Ref Name | Used |  Functional Category |
+----------+------+----------------------+
| BIBUF    |  130 |                   IO |
| LUT1     |   24 |                  LUT |
| PS7      |    1 | Specialized Resource |
| BUFG     |    1 |                Clock |
+----------+------+----------------------+
```

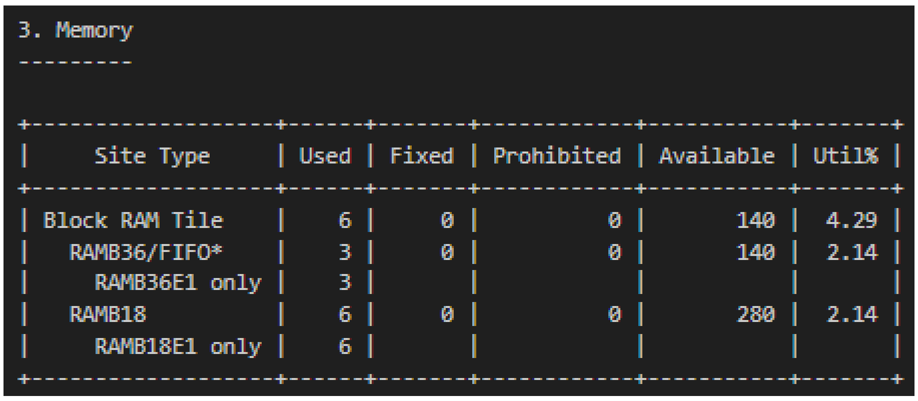
Overall implementation:

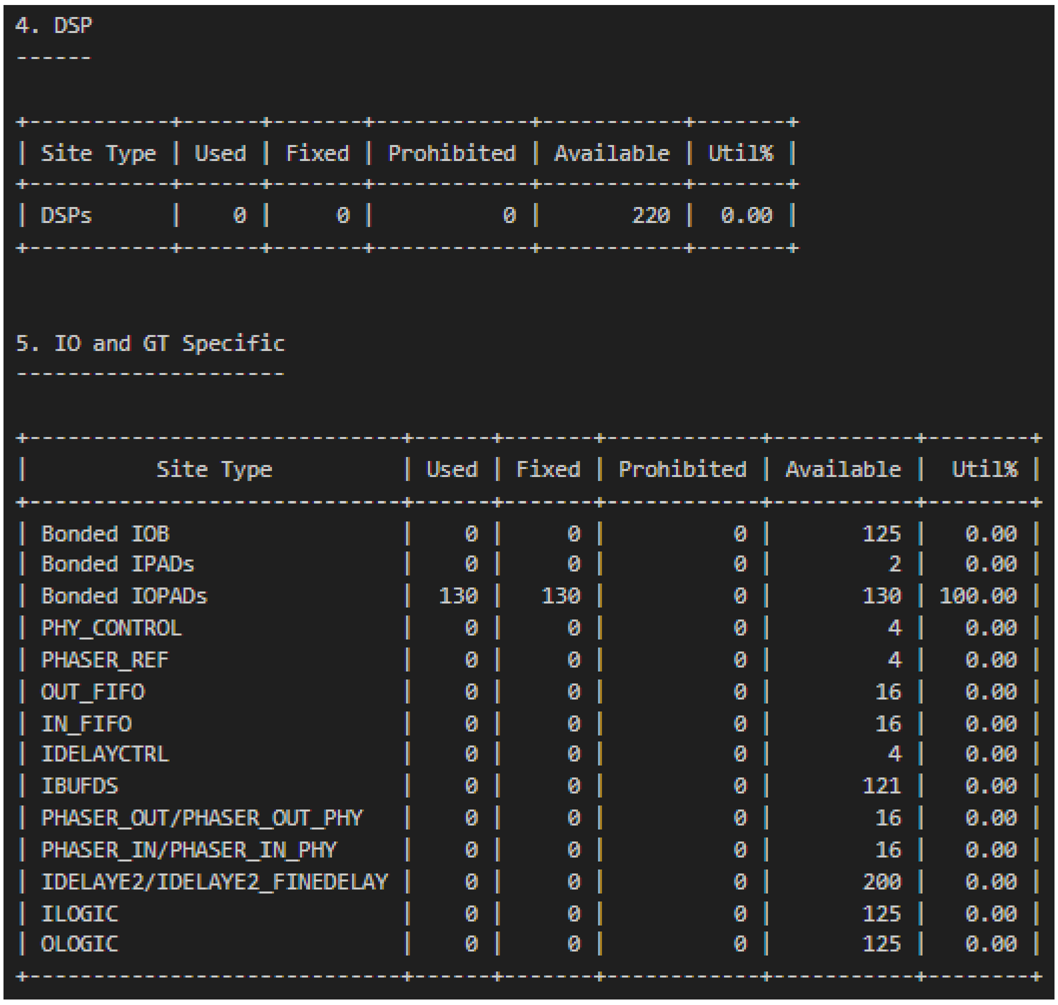


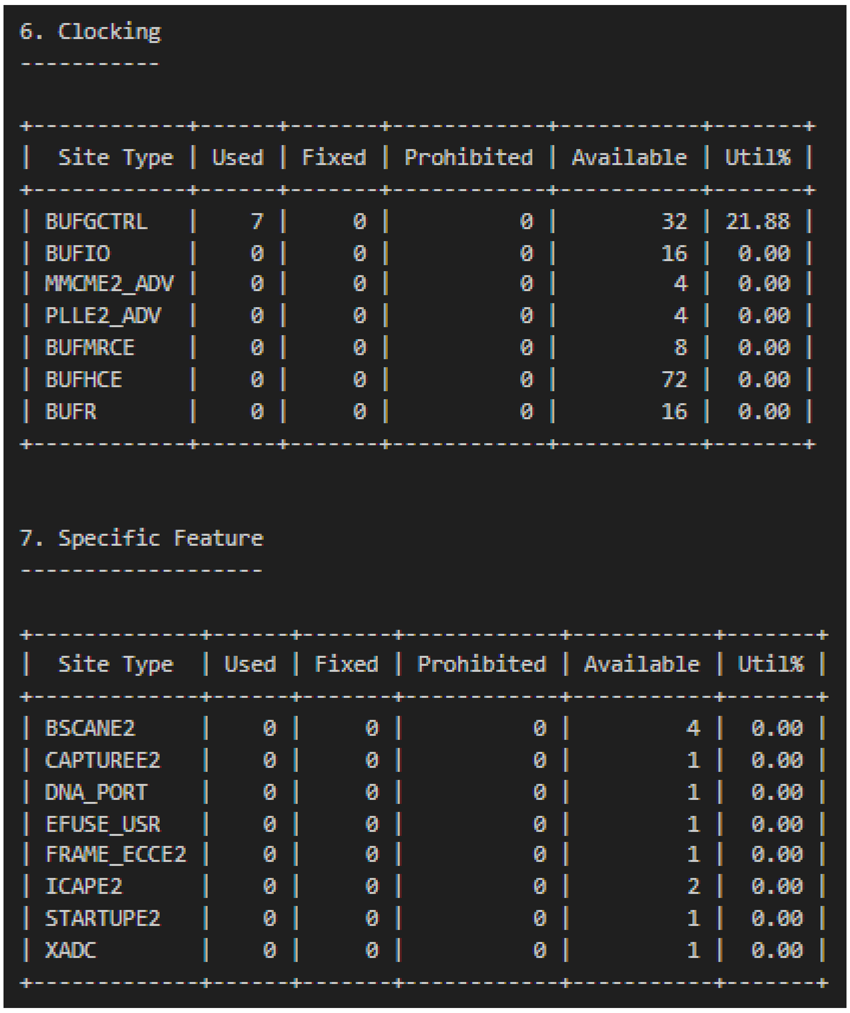
```
1. Slice Logic
--------------

+----------------------------+------+-------+------------+-----------+-------+
|          Site Type         | Used | Fixed | Prohibited | Available | Util% |
+----------------------------+------+-------+------------+-----------+-------+
| Slice LUTs                 | 5327 |     0 |          0 |     53200 | 10.01 |
|   LUT as Logic             | 5149 |     0 |          0 |     53200 |  9.68 |
|   LUT as Memory            |  178 |     0 |          0 |     17400 |  1.02 |
|     LUT as Distributed RAM |   18 |     0 |            |           |       |
|     LUT as Shift Register  |  160 |     0 |            |           |       |
| Slice Registers            | 6051 |     0 |          0 |    106400 |  5.69 |
|   Register as Flip Flop    | 6051 |     0 |          0 |    106400 |  5.69 |
|   Register as Latch        |    0 |     0 |          0 |    106400 |  0.00 |
| F7 Muxes                   |  169 |     0 |          0 |     26600 |  0.64 |
| F8 Muxes                   |   47 |     0 |          0 |     13300 |  0.35 |
+----------------------------+------+-------+------------+-----------+-------+
```



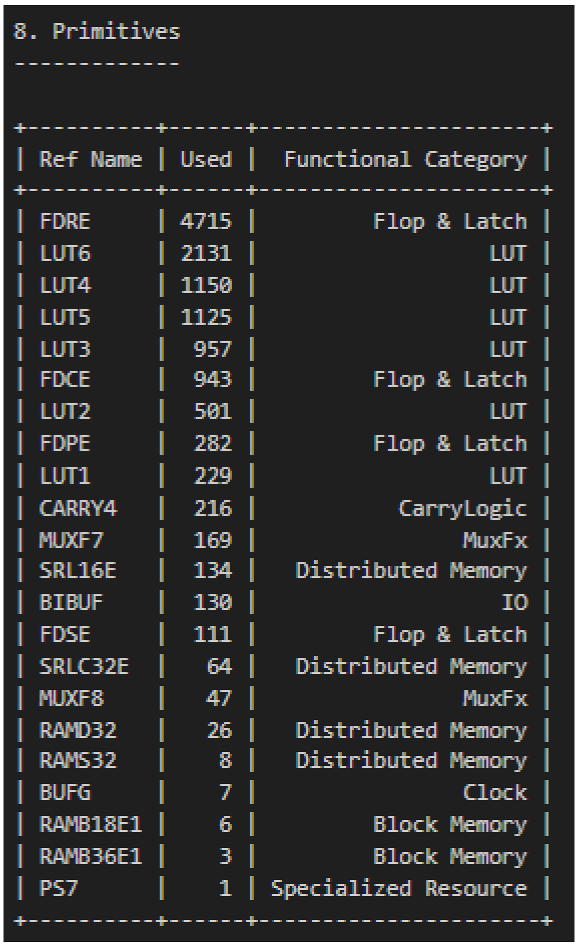
```
2. Slice Logic Distribution
---------------------------

+-------------------------------------------+------+-------+------------+-----------+-------+
|                 Site Type                 | Used | Fixed | Prohibited | Available | Util% |
+-------------------------------------------+------+-------+------------+-----------+-------+
| Slice                                     | 2303 |     0 |          0 |     13300 | 17.32 |
|   SLICEL                                  | 1625 |     0 |            |           |       |
|   SLICEM                                  |  678 |     0 |            |           |       |
| LUT as Logic                              | 5149 |     0 |          0 |     53200 |  9.68 |
|   using O5 output only                    |    0 |       |            |           |       |
|   using O6 output only                    | 4205 |       |            |           |       |
|   using O5 and O6                         |  944 |       |            |           |       |
| LUT as Memory                             |  178 |     0 |          0 |     17400 |  1.02 |
|   LUT as Distributed RAM                  |   18 |     0 |            |           |       |
|     using O5 output only                  |    0 |       |            |           |       |
|     using O6 output only                  |    2 |       |            |           |       |
|     using O5 and O6                       |   16 |       |            |           |       |
|   LUT as Shift Register                   |  160 |     0 |            |           |       |
|     using O5 output only                  |   41 |       |            |           |       |
|     using O6 output only                  |   81 |       |            |           |       |
|     using O5 and O6                       |   38 |       |            |           |       |
| Slice Registers                           | 6051 |     0 |          0 |    106400 |  5.69 |
|   Register driven from within the Slice   | 2815 |       |            |           |       |
|   Register driven from outside the Slice  | 3236 |       |            |           |       |
|     LUT in front of the register is unused | 1978 |       |            |           |       |
|     LUT in front of the register is used  | 1258 |       |            |           |       |
| Unique Control Sets                       |  312 |       |          0 |     13300 |  2.35 |
+-------------------------------------------+------+-------+------------+-----------+-------+
```


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

+-------------------+------+-------+------------+-----------+-------+
|     Site Type     | Used | Fixed | Prohibited | Available | Util% |
+-------------------+------+-------+------------+-----------+-------+
| Block RAM Tile    |    6 |     0 |          0 |       140 |  4.29 |
|   RAMB36/FIFO*    |    3 |     0 |          0 |       140 |  2.14 |
|     RAMB36E1 only |    3 |       |            |           |       |
|   RAMB18          |    6 |     0 |          0 |       280 |  2.14 |
|     RAMB18E1 only |    6 |       |            |           |       |
+-------------------+------+-------+------------+-----------+-------+
```

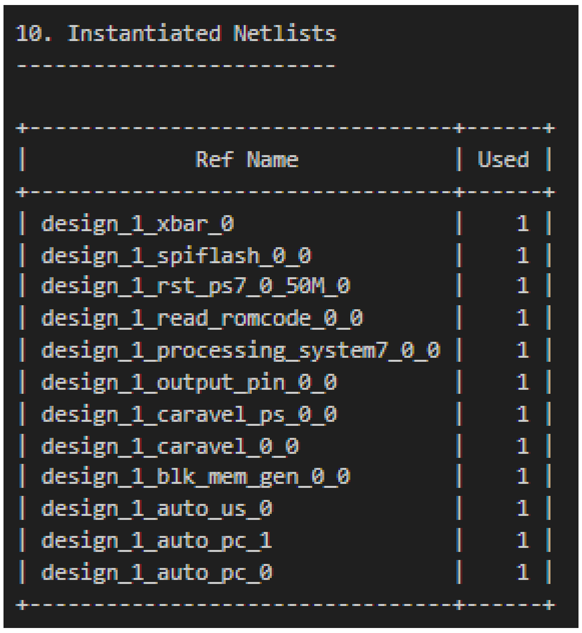

```
4. DSP
------

+-----------+------+-------+------------+-----------+-------+
| Site Type | Used | Fixed | Prohibited | Available | Util% |
+-----------+------+-------+------------+-----------+-------+
| DSPs      |    0 |     0 |          0 |       220 |  0.00 |
+-----------+------+-------+------------+-----------+-------+



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

+-----------------------------+------+-------+------------+-----------+--------+
|          Site Type          | Used | Fixed | Prohibited | Available |  Util% |
+-----------------------------+------+-------+------------+-----------+--------+
| Bonded IOB                  |    0 |     0 |          0 |       125 |   0.00 |
| Bonded IPADs                |    0 |     0 |          0 |         2 |   0.00 |
| Bonded IOPADs               |  130 |   130 |          0 |       130 | 100.00 |
| PHY_CONTROL                 |    0 |     0 |          0 |         4 |   0.00 |
| PHASER_REF                  |    0 |     0 |          0 |         4 |   0.00 |
| OUT_FIFO                    |    0 |     0 |          0 |        16 |   0.00 |
| IN_FIFO                     |    0 |     0 |          0 |        16 |   0.00 |
| IDELAYCTRL                  |    0 |     0 |          0 |         4 |   0.00 |
| IBUFDS                      |    0 |     0 |          0 |       121 |   0.00 |
| PHASER_OUT/PHASER_OUT_PHY   |    0 |     0 |          0 |        16 |   0.00 |
| PHASER_IN/PHASER_IN_PHY     |    0 |     0 |          0 |        16 |   0.00 |
| IDELAYE2/IDELAYE2_FINEDELAY |    0 |     0 |          0 |       200 |   0.00 |
| ILOGIC                      |    0 |     0 |          0 |       125 |   0.00 |
| OLOGIC                      |    0 |     0 |          0 |       125 |   0.00 |
+-----------------------------+------+-------+------------+-----------+--------+
```



```
6. Clocking
-----------

+------------+------+-------+------------+-----------+-------+
|  Site Type | Used | Fixed | Prohibited | Available | Util% |
+------------+------+-------+------------+-----------+-------+
| BUFGCTRL   |    7 |     0 |          0 |        32 | 21.88 |
| BUFIO      |    0 |     0 |          0 |        16 |  0.00 |
| MMCME2_ADV |    0 |     0 |          0 |         4 |  0.00 |
| PLLE2_ADV  |    0 |     0 |          0 |         4 |  0.00 |
| BUFMRCE    |    0 |     0 |          0 |         8 |  0.00 |
| BUFHCE     |    0 |     0 |          0 |        72 |  0.00 |
| BUFR       |    0 |     0 |          0 |        16 |  0.00 |
+------------+------+-------+------------+-----------+-------+


7. Specific Feature
-------------------

+-------------+------+-------+------------+-----------+-------+
|  Site Type  | Used | Fixed | Prohibited | Available | Util% |
+-------------+------+-------+------------+-----------+-------+
| BSCANE2     |    0 |     0 |          0 |         4 |  0.00 |
| CAPTUREE2   |    0 |     0 |          0 |         1 |  0.00 |
| DNA_PORT    |    0 |     0 |          0 |         1 |  0.00 |
| EFUSE_USR   |    0 |     0 |          0 |         1 |  0.00 |
| FRAME_ECCE2 |    0 |     0 |          0 |         1 |  0.00 |
| ICAPE2      |    0 |     0 |          0 |         2 |  0.00 |
| STARTUPE2   |    0 |     0 |          0 |         1 |  0.00 |
| XADC        |    0 |     0 |          0 |         1 |  0.00 |
+-------------+------+-------+------------+-----------+-------+
```



```
8. Primitives
-------------


+----------+------+---------------------+
| Ref Name | Used | Functional Category |
+----------+------+---------------------+
| FDRE     | 4715 |         Flop & Latch |
| LUT6     | 2131 |                  LUT |
| LUT4     | 1150 |                  LUT |
| LUT5     | 1125 |                  LUT |
| LUT3     |  957 |                  LUT |
| FDCE     |  943 |         Flop & Latch |
| LUT2     |  501 |                  LUT |
| FDPE     |  282 |         Flop & Latch |
| LUT1     |  229 |                  LUT |
| CARRY4   |  216 |           CarryLogic |
| MUXF7    |  169 |                MuxFx |
| SRL16E   |  134 |   Distributed Memory |
| BIBUF    |  130 |                   IO |
| FDSE     |  111 |         Flop & Latch |
| SRLC32E  |   64 |   Distributed Memory |
| MUXF8    |   47 |                MuxFx |
| RAMD32   |   26 |   Distributed Memory |
| RAMS32   |    8 |   Distributed Memory |
| BUFG     |    7 |                Clock |
| RAMB18E1 |    6 |         Block Memory |
| RAMB36E1 |    3 |         Block Memory |
| PS7      |    1 | Specialized Resource |
+----------+------+---------------------+
```



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


+----------------------------------+------+
|             Ref Name             | Used |
+----------------------------------+------+
| design_1_xbar_0                  |    1 |
| design_1_spiflash_0_0            |    1 |
| design_1_rst_ps7_0_50M_0         |    1 |
| design_1_read_romcode_0_0        |    1 |
| design_1_processing_system7_0_0  |    1 |
| design_1_output_pin_0_0          |    1 |
| design_1_caravel_ps_0_0          |    1 |
| design_1_caravel_0_0             |    1 |
| design_1_blk_mem_gen_0_0         |    1 |
| design_1_auto_us_0               |    1 |
| design_1_auto_pc_1               |    1 |
| design_1_auto_pc_0               |    1 |
+----------------------------------+------+
```