



Virtual Array Architecture for eFPGA

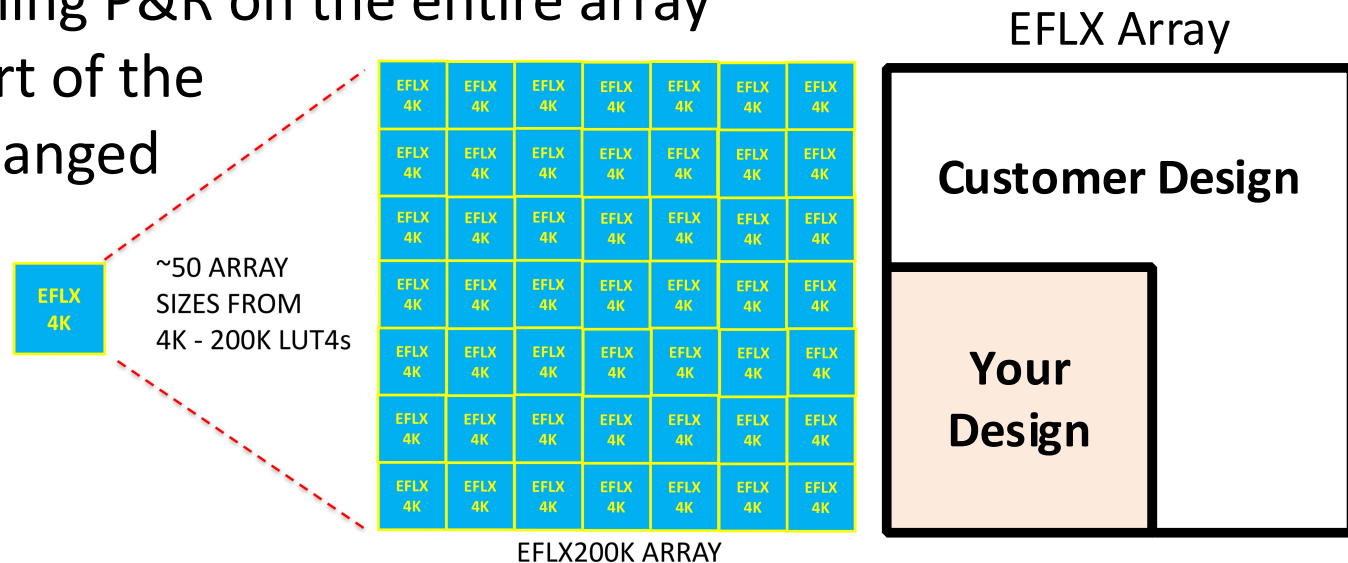
April 5, 2018

Cheng C. Wang – SVP of Engineering

Copyright 2014-2018 Flex Logix Technologies, Inc.

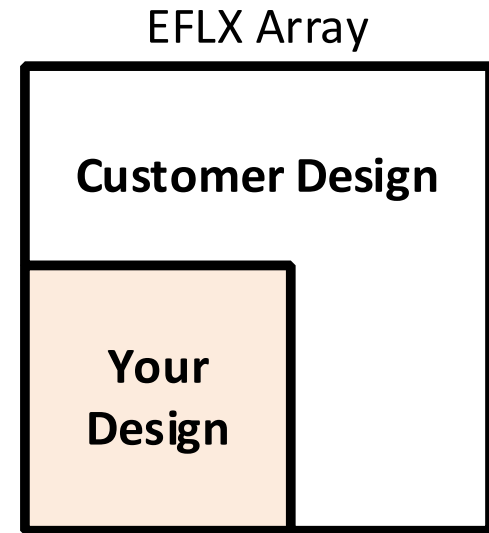
Why the Need for Virtual Arrays?

- Security:
 - Keep your “secret sauce” design masked from the customer
- Preserve Performance & Density:
 - “Harden” your optimized design for best performance & density on EFLX arrays, even when the rest of the design changes
- Runtime:
 - Avoid performing P&R on the entire array when your part of the design is unchanged



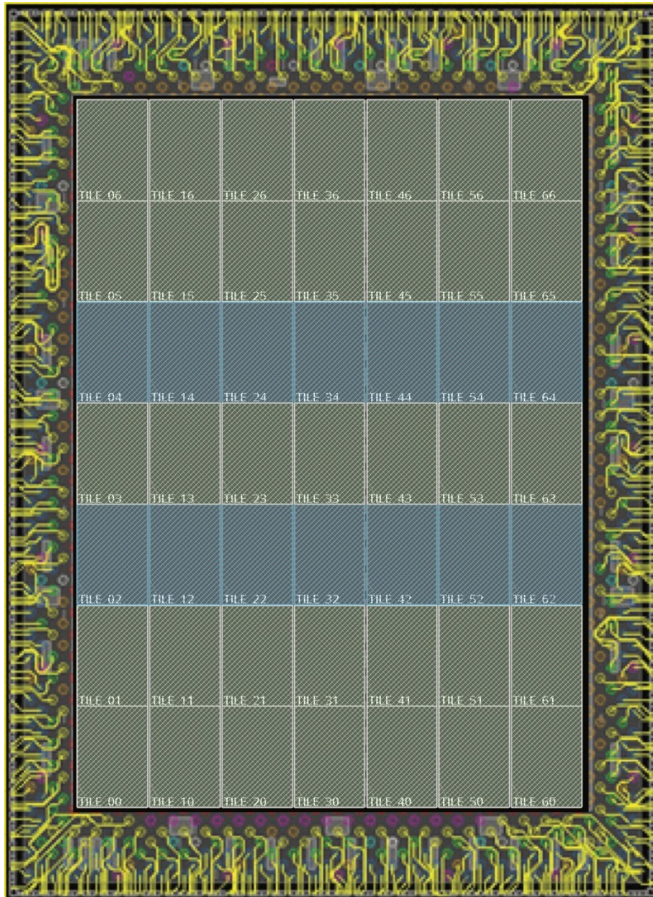
Virtual Array Solution

- Have your “secret sauce” IP and customer design on one EFLX array
 - Designs inside black-box tiles are not visible by end-user
 - Remaining EFLX tiles remain user-programmable
- Maintain optimal performance and density of EFLX designs
 - Preserve P&R performance in black-box tiles
 - Black-box bitstream is merged directly into the final bitstream
- Improve EFLX Compiler runtime
 - Designs in black-box tiles are not P&R’ed again



Example: EFLX200K Test Chip in TSMC16FFC

- 7x7 Array
 - 114,240 6-LUTs (~183K LUT4s)
 - 560 22x22 MACs

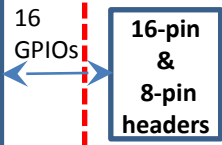
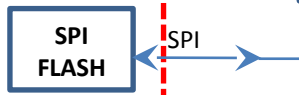
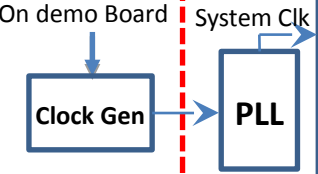
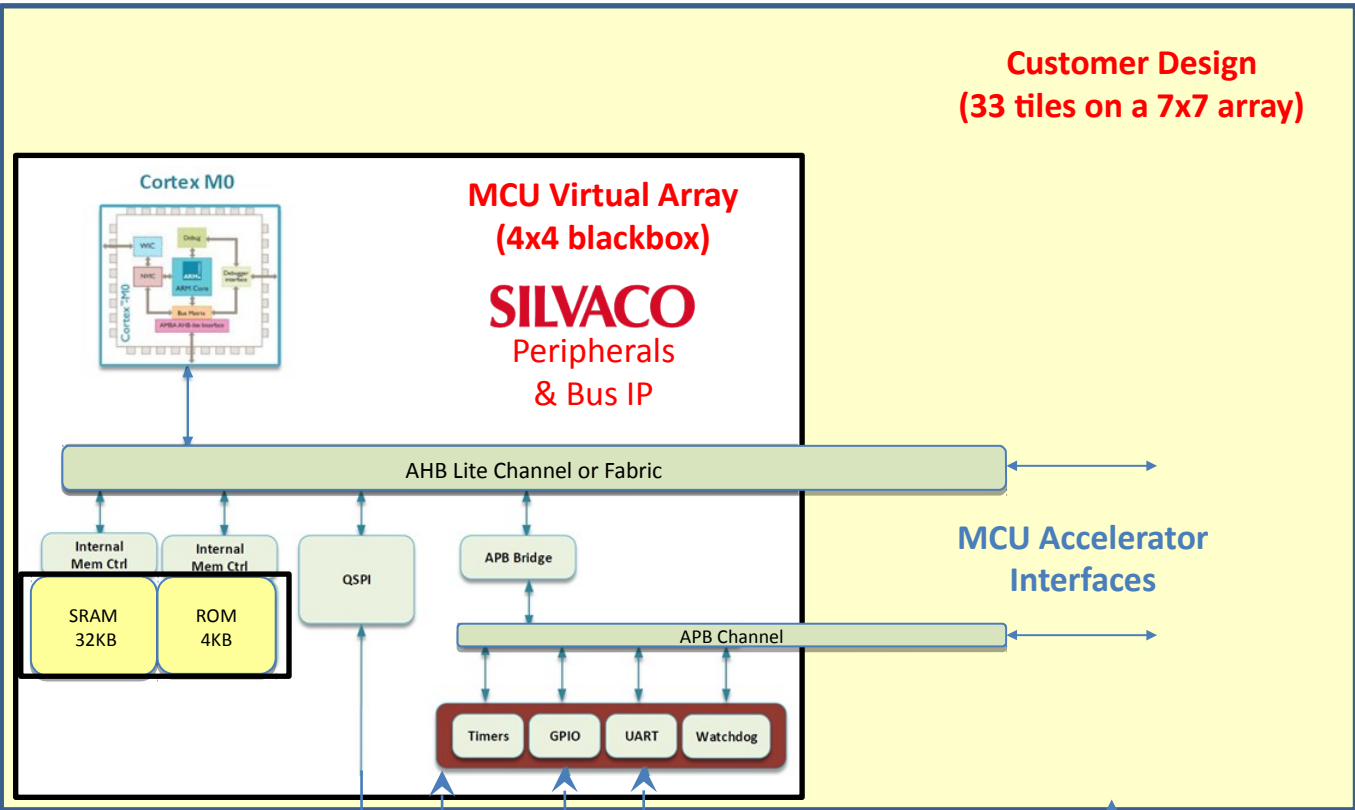
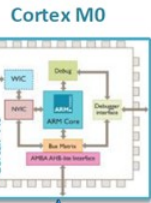


MCU Virtual Array on EFLX 200K Test Chip

EFLX 200K Test Chip

Customer Design
(33 tiles on a 7x7 array)

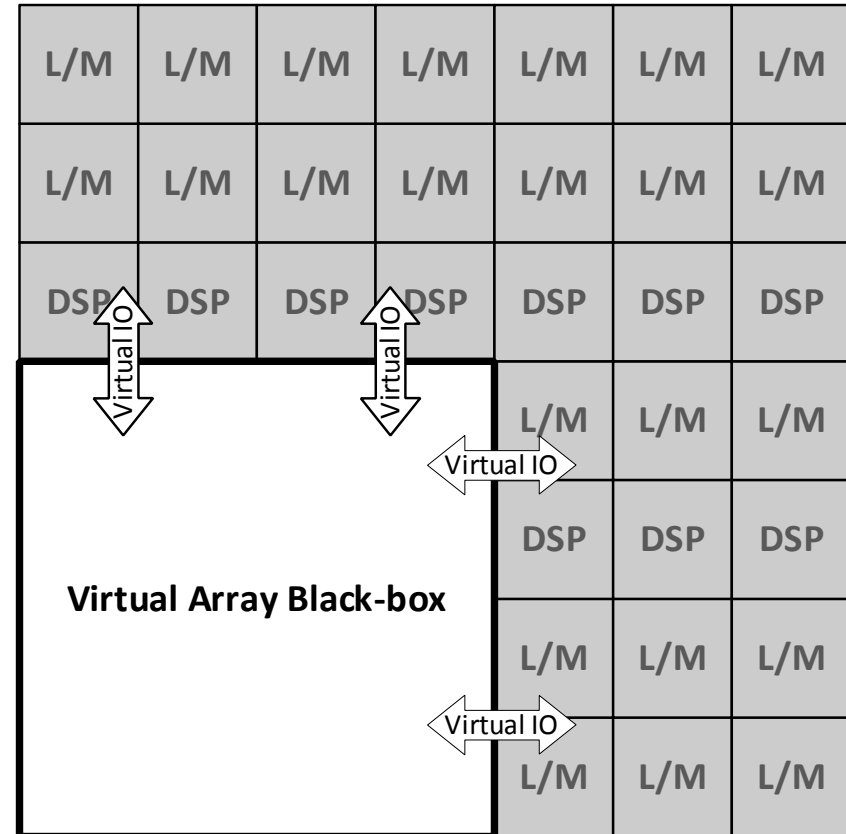
MCU Virtual Array
(4x4 blackbox)
SILVACO
Peripherals
& Bus IP



Virtual Array Definition

Each EFLX array can have one or more black-box virtual arrays

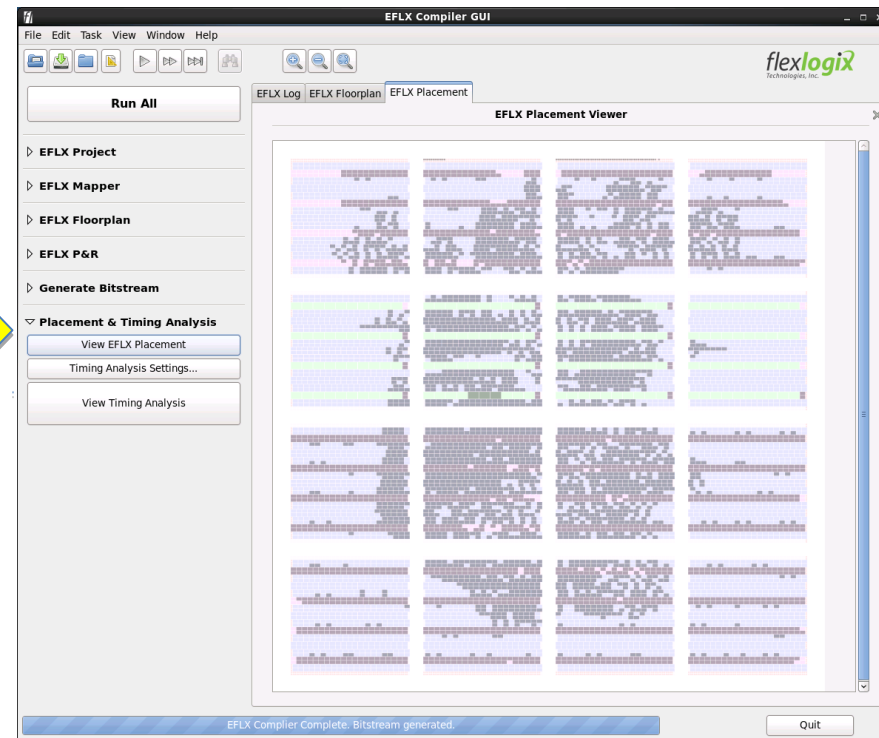
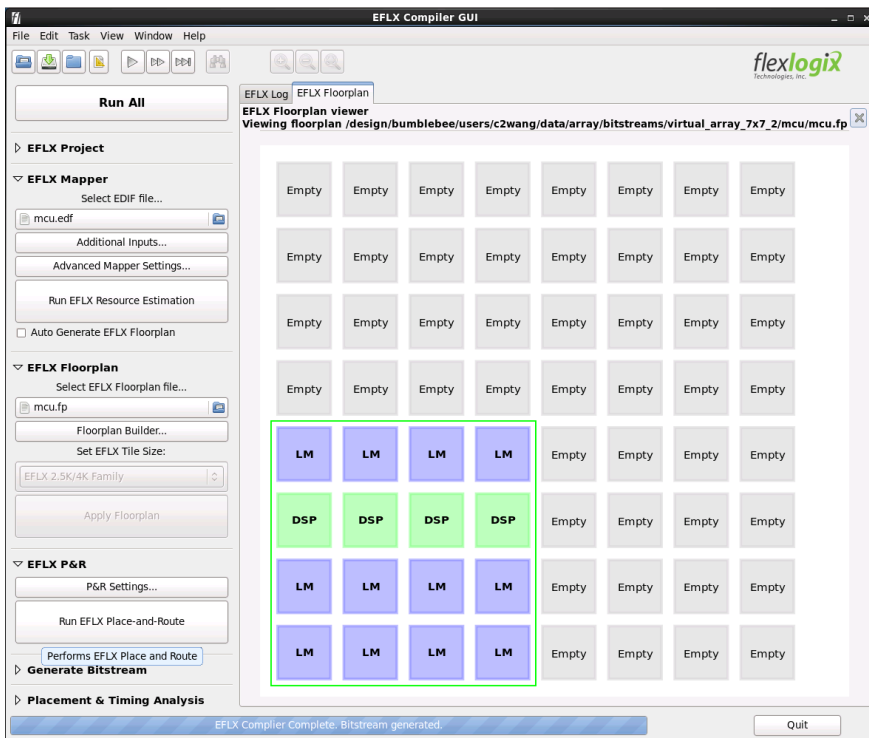
- Each virtual array is a sub-module of the top-level, no direct IO access is needed
- Each virtual array should have its virtual I/Os placed towards the other tiles of the EFLX array



Virtual Array Compilation

Other than pin placement, compiling a 4x4 black-box virtual array is no different from a regular 4x4 array

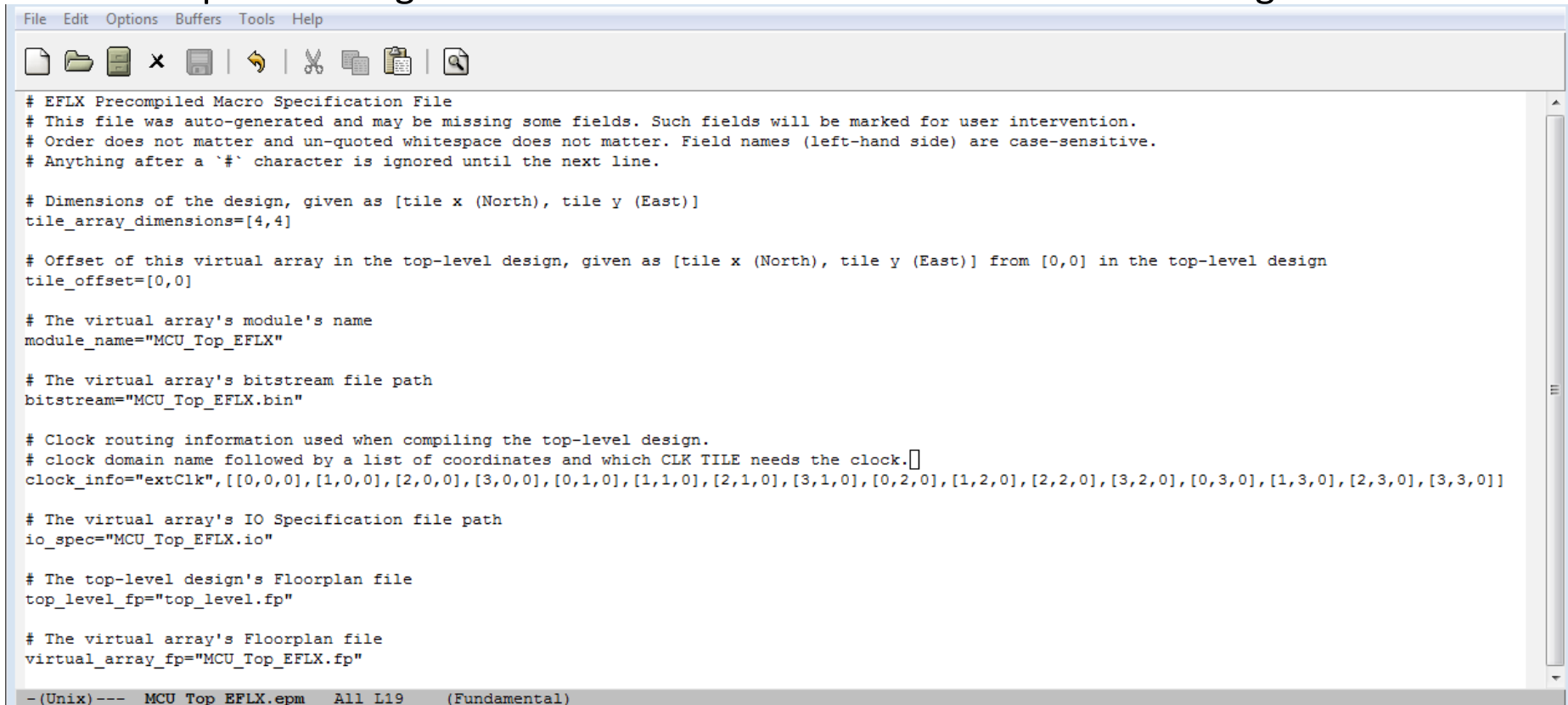
- EFLX Compiler offers complete netlist-to-bitstream solution



Black-box Definition (Auto-Generated)

A black-box definition: EFLX Precompiled Macro (.epm)

- Defines black-box essentials: module name, footprint, IO location, clock utilization
- The epm is auto-generated with “-GENERATE VIRTUAL ARRAY” flag



```
File Edit Options Buffers Tools Help
# EFLX Precompiled Macro Specification File
# This file was auto-generated and may be missing some fields. Such fields will be marked for user intervention.
# Order does not matter and un-quoted whitespace does not matter. Field names (left-hand side) are case-sensitive.
# Anything after a '#' character is ignored until the next line.

# Dimensions of the design, given as [tile x (North), tile y (East)]
tile_array_dimensions=[4,4]

# Offset of this virtual array in the top-level design, given as [tile x (North), tile y (East)] from [0,0] in the top-level design
tile_offset=[0,0]

# The virtual array's module's name
module_name="MCU_Top_EFLX"

# The virtual array's bitstream file path
bitstream="MCU_Top_EFLX.bin"

# Clock routing information used when compiling the top-level design.
# clock domain name followed by a list of coordinates and which CLK TILE needs the clock.[]
clock_info="extClk", [[0,0,0], [1,0,0], [2,0,0], [3,0,0], [0,1,0], [1,1,0], [2,1,0], [3,1,0], [0,2,0], [1,2,0], [2,2,0], [3,2,0], [0,3,0], [1,3,0], [2,3,0], [3,3,0]]

# The virtual array's IO Specification file path
io_spec="MCU_Top_EFLX.io"

# The top-level design's Floorplan file
top_level_fp="top_level.fp"

# The virtual array's Floorplan file
virtual_array_fp="MCU_Top_EFLX.fp"

-(Unix)--- MCU_Top_EFLX.epm All L19 (Fundamental)
```


Instantiating a Black-box in Customer Design

The black-box module content is removed from the customer design, and replaced with the .epm inference

- .epm black box is automatically preserved through Synplify

The screenshot displays the Synplify Pro J-2015.03 interface. The main editor shows Verilog code for a module named `MCU_Top_EFLX.v`. The code includes a list of ports (lines 118-143) and a parameter `INSERT_PRECOMPILED_MACRO` (lines 149-151) set to `"mdu_demo_virtual_array_20180330.epm"`. A blue callout box highlights the parameter assignment with the text "Black Box Inferred through .epm".

To the right, a separate window titled "Black Box .epm in EDIF netlist" shows the corresponding EDIF netlist. The netlist defines a cell `MCU_Top_EFLX` with a generic cell type and lists all the ports and their directions, matching the Verilog code. A blue arrow points from the parameter assignment in the Verilog code to the EDIF netlist.

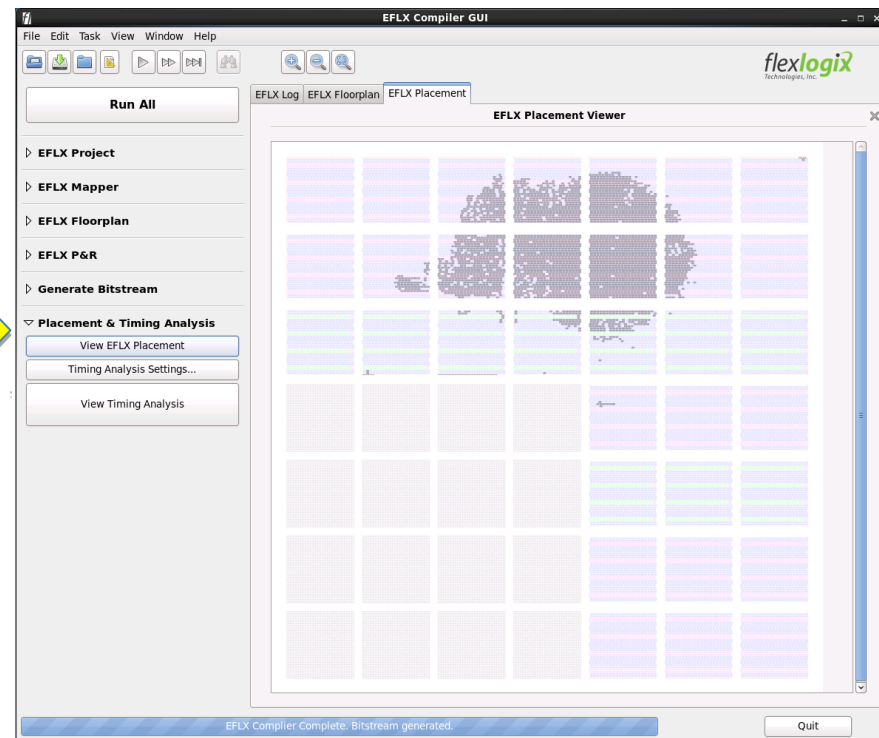
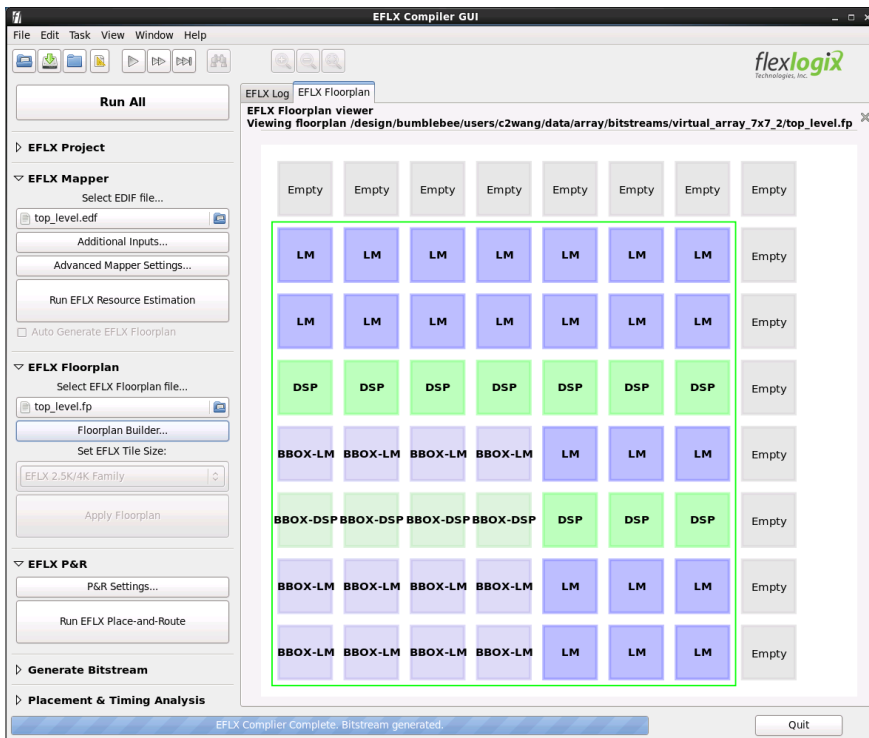
```
118
119 output      HRESETn_M1;
120 input       HBUSREQ_M1;
121 input       HLOCK_M1;
122 output      HGRANT_M1;
123 input [31:0] HADDR_M1;
124 input [31:0] HWDATA_M1;
125 input       HWRITE_M1;
126 input [1:0] HTRANS_M1;
127 input [2:0] HSIZE_M1;
128 input [2:0] HBURST_M1;
129 input [3:0] HPROT_M1;
130 output [31:0] HRDATA_M1;
131 output      HREADY_M1;
132 output [1:0] HRESP_M1;
133
134 output      PRESETn_1_S3;
135 output      PSEL_1_S3;
136 output      PWRITE_1_S3;
137 output      PENABLE_1_S3;
138 output [31:0] PADDR_1_S3;
139 output [31:0] PWDATA_1_S3;
140 input [31:0] PRDATA_1_S3;
141 input       AccIntReq;
142
143
144 ////////////////////////////////////////////////////
145 // Black-box instantiation
146 ////////////////////////////////////////////////////
147
148
149 parameter INSERT_PRECOMPILED_MACRO = "mdu_demo_virtual_array_20180330.epm" ;
150
151 endmodule
```

```
(cell MCU_Top_EFLX (cellType GENERIC)
(view Verilog (viewType NETLIST)
(interface
(port (array (rename spiSdataMO "spiSdataMO[3:0]") 4) (direction OUTPUT))
(port (array (rename spiSdataMI "spiSdataMI[3:0]") 4) (direction INPUT))
(port (array (rename gpioIn "gpioIn[7:0]") 8) (direction INPUT))
(port (array (rename gpioOut "gpioOut[7:0]") 8) (direction OUTPUT))
(port (array (rename HADDR_M1 "HADDR_M1[31:0]") 32) (direction INPUT))
(port (array (rename HWDATA_M1 "HWDATA_M1[31:0]") 32) (direction INPUT))
(port (array (rename HTRANS_M1 "HTRANS_M1[1:0]") 2) (direction INPUT))
(port (array (rename HSIZE_M1 "HSIZE_M1[2:0]") 3) (direction INPUT))
(port (array (rename HBURST_M1 "HBURST_M1[2:0]") 3) (direction INPUT))
(port (array (rename HPROT_M1 "HPROT_M1[3:0]") 4) (direction INPUT))
(port (array (rename HRDATA_M1 "HRDATA_M1[31:0]") 32) (direction OUTPUT))
(port (array (rename HRESP_M1 "HRESP_M1[1:0]") 2) (direction OUTPUT))
(port (array (rename PADDR_1_S3 "PADDR_1_S3[31:0]") 32) (direction OUTPUT))
(port (array (rename PWDATA_1_S3 "PWDATA_1_S3[31:0]") 32) (direction OUTPUT))
(port (array (rename PRDATA_1_S3 "PRDATA_1_S3[31:0]") 32) (direction INPUT))
(port (array (rename exRstn "exRstn") 1) (direction INPUT))
(port extCLK (direction INPUT))
(port rs232Atx (direction OUTPUT))
(port rs232Arx (direction INPUT))
(port spiSclkOut (direction OUTPUT))
(port spiSsOut (direction OUTPUT))
(port JTKC_i (direction INPUT))
(port JTDI (direction INPUT))
(port JTMS (direction INPUT))
(port JTRSTn (direction INPUT))
(port JTDO (direction OUTPUT))
(port l2csClout_1_P0 (direction OUTPUT))
(port l2csDAOE_R_1_P0 (direction OUTPUT))
(port l2csDAout_1_P0 (direction OUTPUT))
(port l2csDAin_1_P0 (direction INPUT))
(port selkOut_1_P2 (direction OUTPUT))
(port sdataIn_1_P2 (direction INPUT))
(port sdataOut_1_P2 (direction OUTPUT))
(port HRESETn_M1 (direction OUTPUT))
(port HBUSREQ_M1 (direction INPUT))
(port HLOCK_M1 (direction INPUT))
(port HGRANT_M1 (direction OUTPUT))
(port HWRITE_M1 (direction INPUT))
(port HREADY_M1 (direction OUTPUT))
(port PRESETn_1_S3 (direction OUTPUT))
(port PSEL_1_S3 (direction OUTPUT))
(port PWRITE_1_S3 (direction OUTPUT))
(port PENABLE_1_S3 (direction OUTPUT))
(port AccIntReq (direction INPUT))
)
(property langParams (string "INSERT_PRECOMPILED_MACRO"))
(property INSERT_PRECOMPILED_MACRO (string "mdu_demo_virtual_array_20180330.epm"))
(property orig_inst_of (string "MCU_Top_EFLX"))
)
```

Running Black-box Design in EFLX Compiler

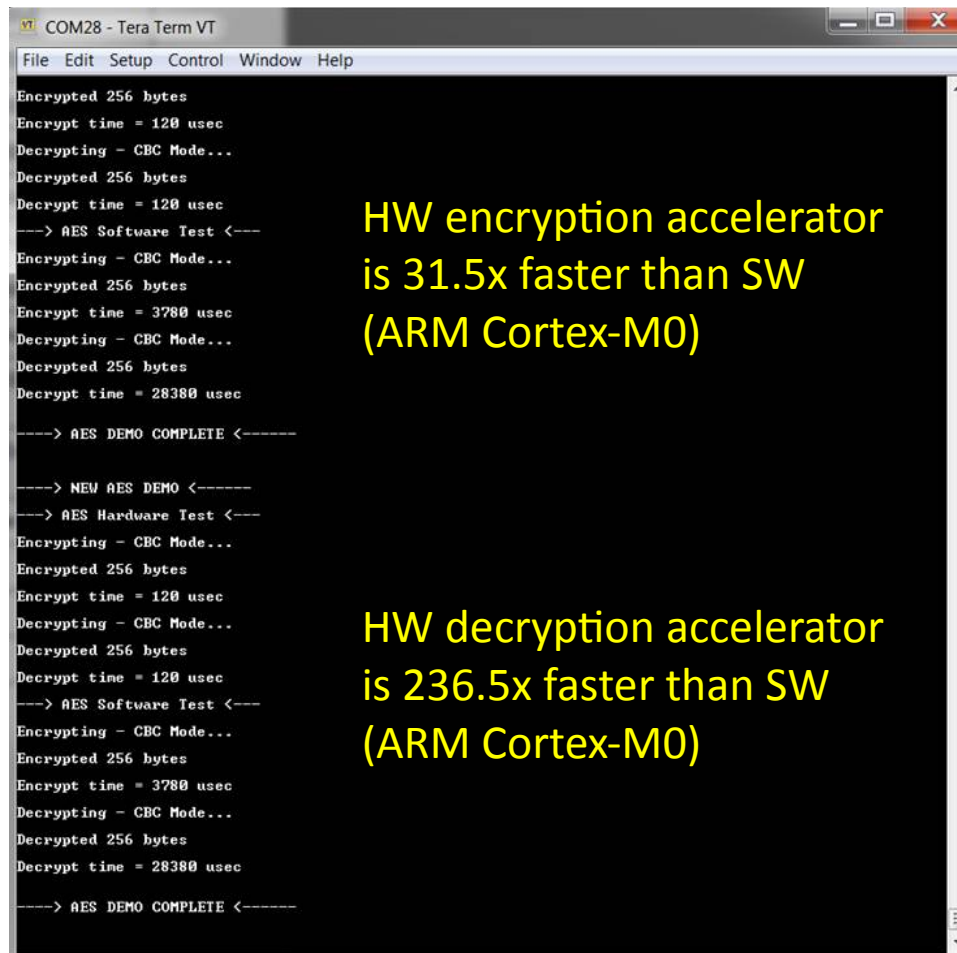
Load a top-level floorplan with the black-box tiles labeled

- EFLX Compiler connects top-level design with black-box design during P&R
- No user visibility into what's inside the black-box



Running Blackbox Designs in Silicon

Black-box MCU (CM0) executes the AES enc/decryption in SW
Custom accelerator executes the same enc/decryption in HW



```
COM28 - Tera Term VT
File Edit Setup Control Window Help
Encrypted 256 bytes
Encrypt time = 120 usec
Decrypting - CBC Mode...
Decrypted 256 bytes
Decrypt time = 120 usec
--> AES Software Test <---
Encrypting - CBC Mode...
Encrypted 256 bytes
Encrypt time = 3780 usec
Decrypting - CBC Mode...
Decrypted 256 bytes
Decrypt time = 28380 usec
----> AES DEMO COMPLETE <-----

----> NEW AES DEMO <-----
--> AES Hardware Test <---
Encrypting - CBC Mode...
Encrypted 256 bytes
Encrypt time = 120 usec
Decrypting - CBC Mode...
Decrypted 256 bytes
Decrypt time = 120 usec
--> AES Software Test <---
Encrypting - CBC Mode...
Encrypted 256 bytes
Encrypt time = 3780 usec
Decrypting - CBC Mode...
Decrypted 256 bytes
Decrypt time = 28380 usec
----> AES DEMO COMPLETE <-----
```

HW encryption accelerator
is 31.5x faster than SW
(ARM Cortex-M0)

HW decryption accelerator
is 236.5x faster than SW
(ARM Cortex-M0)

Conclusion: Virtual Array Benefits

- Virtual array is a simple way to insert a pre-compiled black-box in to the customer design
- **Secure:** User has no visibility into the black-box design
- **Predictable:** Pre-compiled design is frozen, preserves PPA
- **Fast:** Pre-compiled design portion is not P&R'ed again
- **Flexible:**
 - Maintain user-programmability in the rest of the designs
 - User can choose to instantiate the black-box or not
 - Updates to .epm can be downloaded as an image file
- **One of many ways to build a flexible, secure, and high-quality EFLX IP ecosystem**