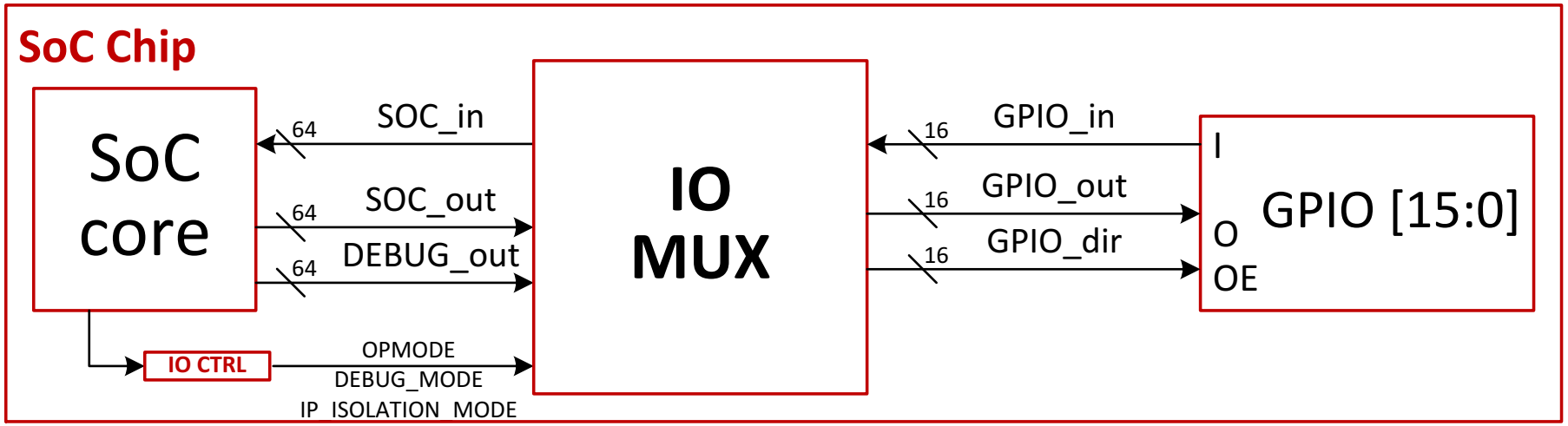
I/O Pin Mux using Embedded FPGA

I/O Pin Mux in RTL

- **In most SoCs, I/O Mux MUST be decided at design time**
  - The designer must choose which (large) set of SoC pins to bring out to a (limited) set of IO pins (e.g. GPIOs)
  - Due to different operation modes and debug modes, the IO mux can become a complex multiplexer



IO Mux in RTL

- **Complex IO Mux can be difficult to track**
  - × Often results in massive Excel sheets to capture every IO mode
  - × Must carefully control both data *and* direction
  - × Large, clumsy, hard-to-read, and **easy to miswire**!
  - × Any changes to IO configuration requires silicon re-spin

	OP_MODE[2]	OP_MODE[1]	OPMODE[0]	DBG_MODE[5]	DBG_MODE[4]	DBG_MODE[3]	DBG_MODE[2]	DBG_MODE[1]	DBG_MODE[0]	IP_ISO_MODE[2]	IP_ISO_MODE[1]	IP_ISO_MODE[0]
GPIO_out[15]	SOC_out[111]	SOC_out[47]	0	DBG_out[127]	DBG_out[111]	DBG_out[95]	DBG_out[79]	DBG_out[63]	DBG_out[47]	DBG_out[31]	DBG_out[15]	SOC_out[15]
GPIO_dir[15]	1	1	0	1	1	1	1	1	1	1	1	1
GPIO_out[14]	SOC_out[110]	SOC_out[46]	0	DBG_out[126]	DBG_out[110]	DBG_out[94]	DBG_out[78]	DBG_out[62]	DBG_out[46]	DBG_out[30]	DBG_out[14]	SOC_out[14]
GPIO_dir[14]	1	1	0	1	1	1	1	1	1	1	1	1
GPIO_out[13]	SOC_out[109]	SOC_out[45]	0	DBG_out[125]	DBG_out[109]	DBG_out[93]	DBG_out[77]	DBG_out[61]	DBG_out[45]	DBG_out[29]	DBG_out[13]	SOC_out[13]
GPIO_dir[13]	1	1	0	1	1	1	1	1	1	1	1	1
GPIO_out[12]	SOC_out[108]	SOC_out[44]	0	DBG_out[124]	DBG_out[108]	DBG_out[92]	DBG_out[76]	DBG_out[60]	DBG_out[44]	DBG_out[28]	DBG_out[12]	SOC_out[12]
GPIO_dir[12]	1	1	0	1	1	1	1	1	1	1	1	1
GPIO_out[11]	SOC_out[107]	SOC_out[43]	0	DBG_out[123]	DBG_out[107]	DBG_out[91]	DBG_out[75]	DBG_out[59]	DBG_out[43]	DBG_out[27]	DBG_out[11]	SOC_out[11]
GPIO_dir[11]	1	1	0	1	1	1	1	1	1	1	1	1
GPIO_out[10]	SOC_out[106]	SOC_out[42]	0	DBG_out[122]	DBG_out[106]	DBG_out[90]	DBG_out[74]	DBG_out[58]	DBG_out[42]	DBG_out[26]	DBG_out[10]	SOC_out[10]
GPIO_dir[10]	1	1	0	1	1	1	1	1	1	1	1	1
GPIO_out[9]	SOC_out[105]	SOC_out[41]	0	DBG_out[121]	DBG_out[105]	DBG_out[89]	DBG_out[73]	DBG_out[57]	DBG_out[41]	DBG_out[25]	DBG_out[9]	SOC_out[9]
GPIO_dir[9]	1	1	0	1	1	1	1	1	1	1	1	1
…	…	…	…	…	…	…	…	…	…	…	…	…

Pin Mux in RTL

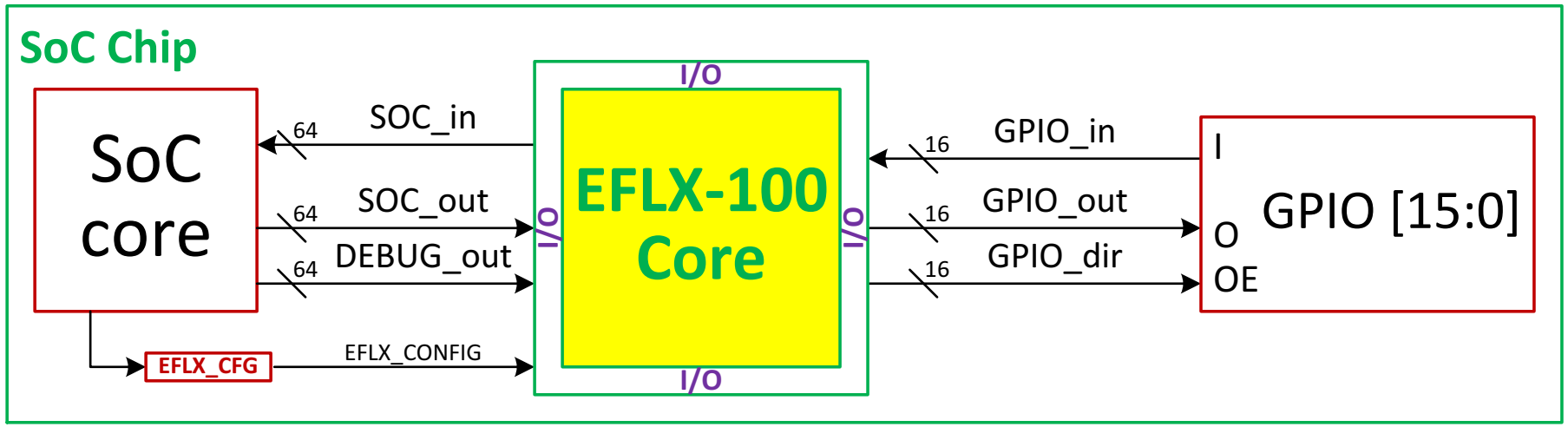
- **Generated RTL is hard-coded**
  - × Any changes to IO configuration requires silicon re-spin

```
module IO_MUX (
 input  [63:0] SOC_in,
output  [63:0] SOC_out,
output  [63:0] DBG_out,
output  [15:0] GPIO_out,
output  [15:0] GPIO_dir,
 input  [15:0] GPIO_in,
// different operation modes
 input   [2:0] OPMODE,
 input   [5:0] DBG_MODE,
 input   [2:0] IP_ISO_MODE );

 assign GPIO_out[15] = OPMODE[2] ? SOC_out[111] : OPMODE[1] ? SOC_out[47] : OPMODE[0] ? 0 :
               DBG_MODE[5] ? DBG_out[127] : DBG_MODE[4] ? DBG_out[111] : DBG_MODE[3] ? DBG_out[95] :
               DBG_MODE[2] ? DBG_out[79]  : DBG_MODE[1] ? DBG_out[63]  : DBG_MODE[0] ? DBG_out[47] :
               IP_ISO_MODE[2] ? DBG_OUT[31] : IP_ISO_MODE[1] ? DBG_OUT[15] : IP_ISO_MODE[0] : SOC_out[15];
 assign GPIO_dir[15] = OPMODE[2] ? 1 : OPMODE[1] ? 1 : OPMODE[0] ? 0 :
               DBG_MODE[5] ? 1 : DBG_MODE[4] ? 1 : DBG_MODE[3] ? 1 :
               DBG_MODE[2] ? 1  : DBG_MODE[1] ? 1  : DBG_MODE[0] ? 1 :
               IP_ISO_MODE[2] ? 1 : IP_ISO_MODE[1] ? 1 : IP_ISO_MODE[0] : 1;

...
```

Repeat 16 times

IO Mux in EFLX

- **Using EFLX, the IO Mux can be fully reconfigurable**
  - ✓ Just wire in all the I/O pins to EFLX
  - ✓ The smallest EFLX (EFLX-100) is 0.12 mm^2 in TSMC 40nm, has:
    152 inputs + 152 outputs
    120 look-up-tables (LUTs) for combinatorial logic and 240 FFs
  - ✓ Re-configurable to ANY new IO configuration, in field, in < 1ms



IO Mux in EFLX - RTL

- **Only need to write RTL for the individual IO Mux mode**
  - ✓ Below is 1 example IO mode (OP_MODE[1])

```
module IO_MUX (           // For example, MODE: OP_MODE[1]
output  [63:0] SOC_in,
 input  [63:0] SOC_out,
 input  [63:0] DBG_out,
output  [15:0] GPIO_out,
output  [15:0] GPIO_dir,
 input  [15:0] GPIO_in);

assign  GPIO_out[15:8] = SOC_out[47:40];
assign  GPIO_dir[15:8] = 8'hff;
assign  GPIO_dir[7:0]  = 8'h00;
assign  GPIO_out[7:0]  = 8'h00;
assign  SOC_in[39:32]  = GPIO_in[7:0];
assign  SOC_in[7:0]    = GPIO_in[15:8];

// Can ALSO do combinatorial + sequential logic and arithmetic/comparators
assign SOC_in[8] = (GPIO_in[15:0] >= 8'x80);
assign SOC_in[9] = GPIO_in[7:0] + GPIO_in[15:8];
endmodule
```

Room for more sophisticated IO processing

Resource Usage (1 EFLX-100 core)	EFLX-100 Input	EFLX-100 Output	Performance (Any IN→OUT)	EFLX-100 Look-up-tables
Total Area: 0.12mm^2 (40nm)	144 out of 152 95%	96 / 152 63%	< 2ns (40nm)	6 / 120 5%