



SHAPING THE NEXT GENERATION OF ELECTRONICS

JUNE 23-27, 2024

MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA



JUNE 23-27, 2024

MOSCONE WEST CENTER
SAN FRANCISCO, CA, USA

No-code Power and Clock System Design

Hoyeon Jeon, Ahchan Kim, Ingyu Kim, Jongbae Lee

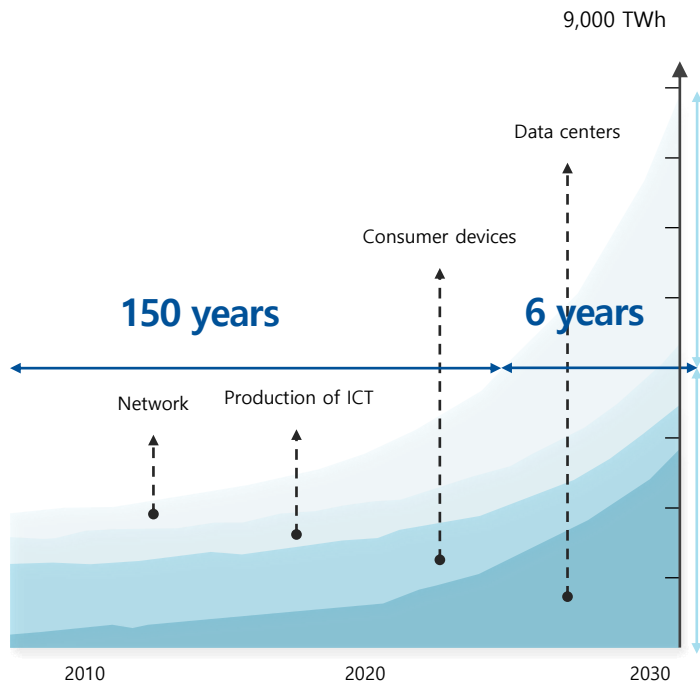
ITDA co., Ltd.



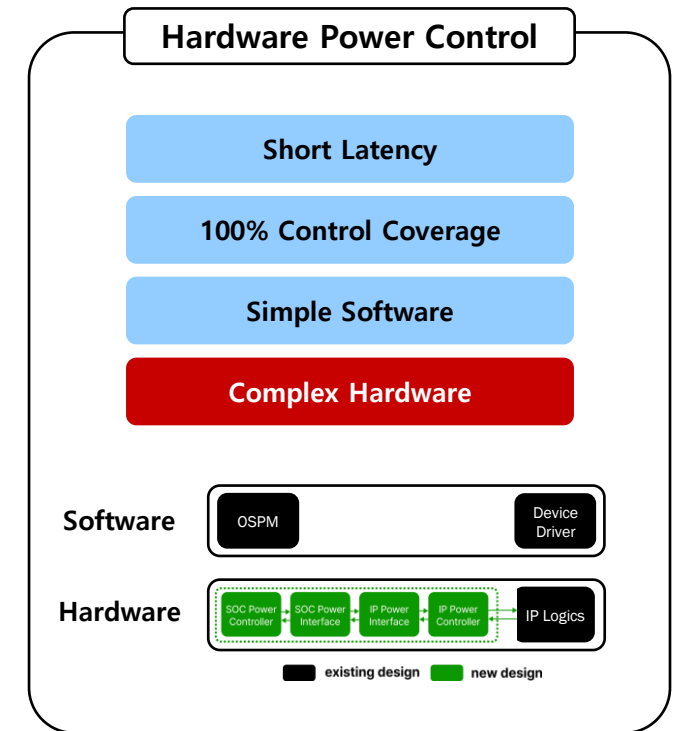
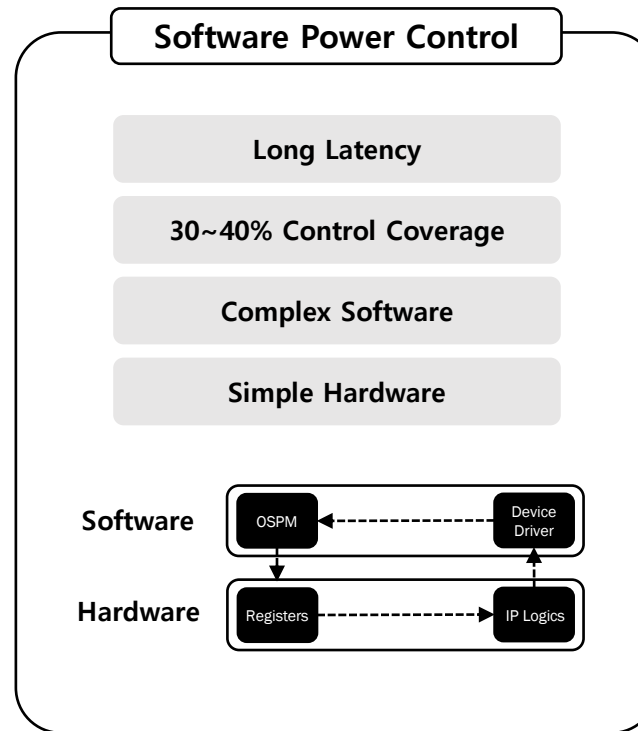
Motivation

Hardware level power management is essential for AI computing ERA

Power consumption exponentially ↑



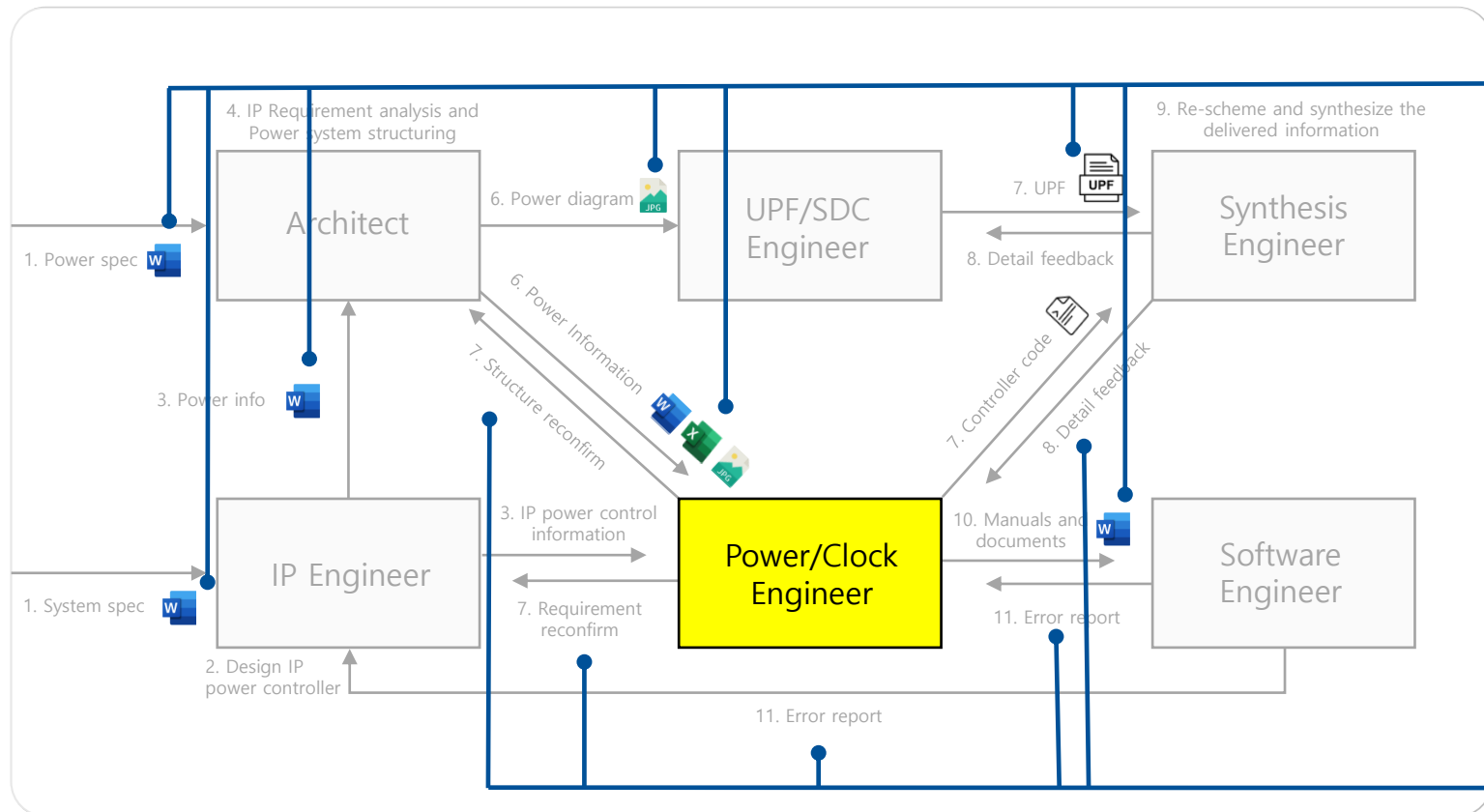
Hardware level power management is efficient but
Hardware design complexity ↑



System Design Complexity

Conventional design process is inconsistent in data, slow and complex

Conventional Power/Clock Design Process



Problems of Design

Data Inconsistency

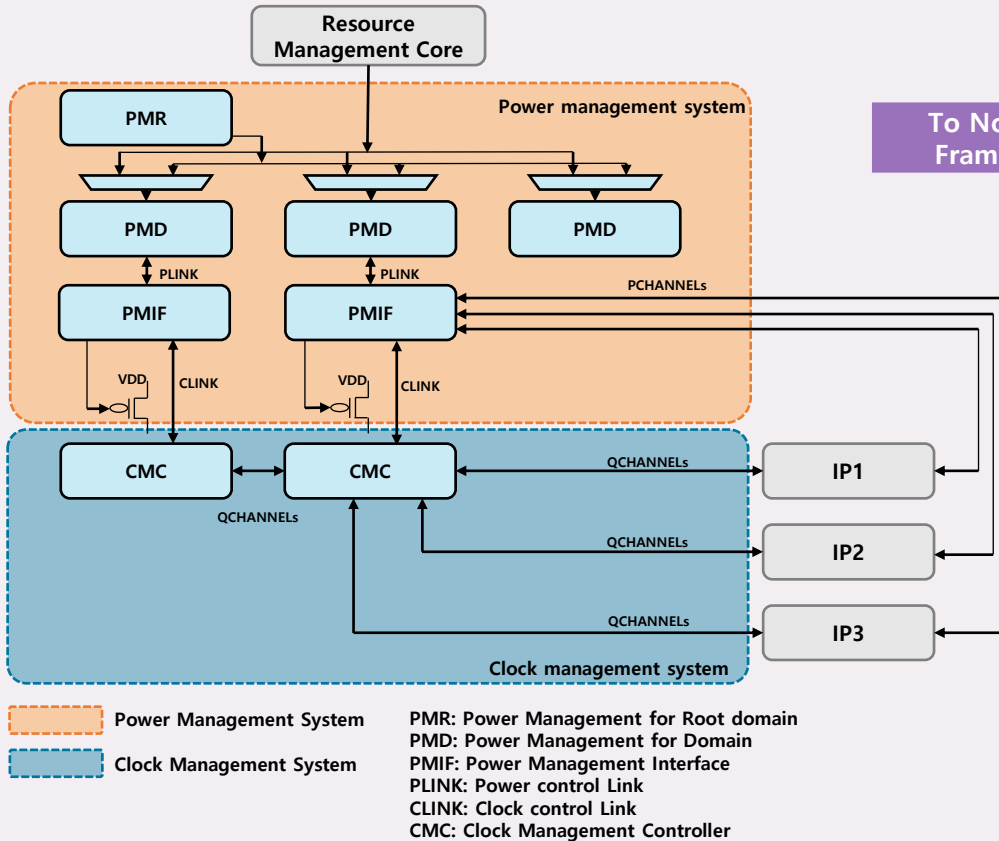
Project Duration Delay

Design Complexity

Difficult to meet power target

No-Code System Design Platform

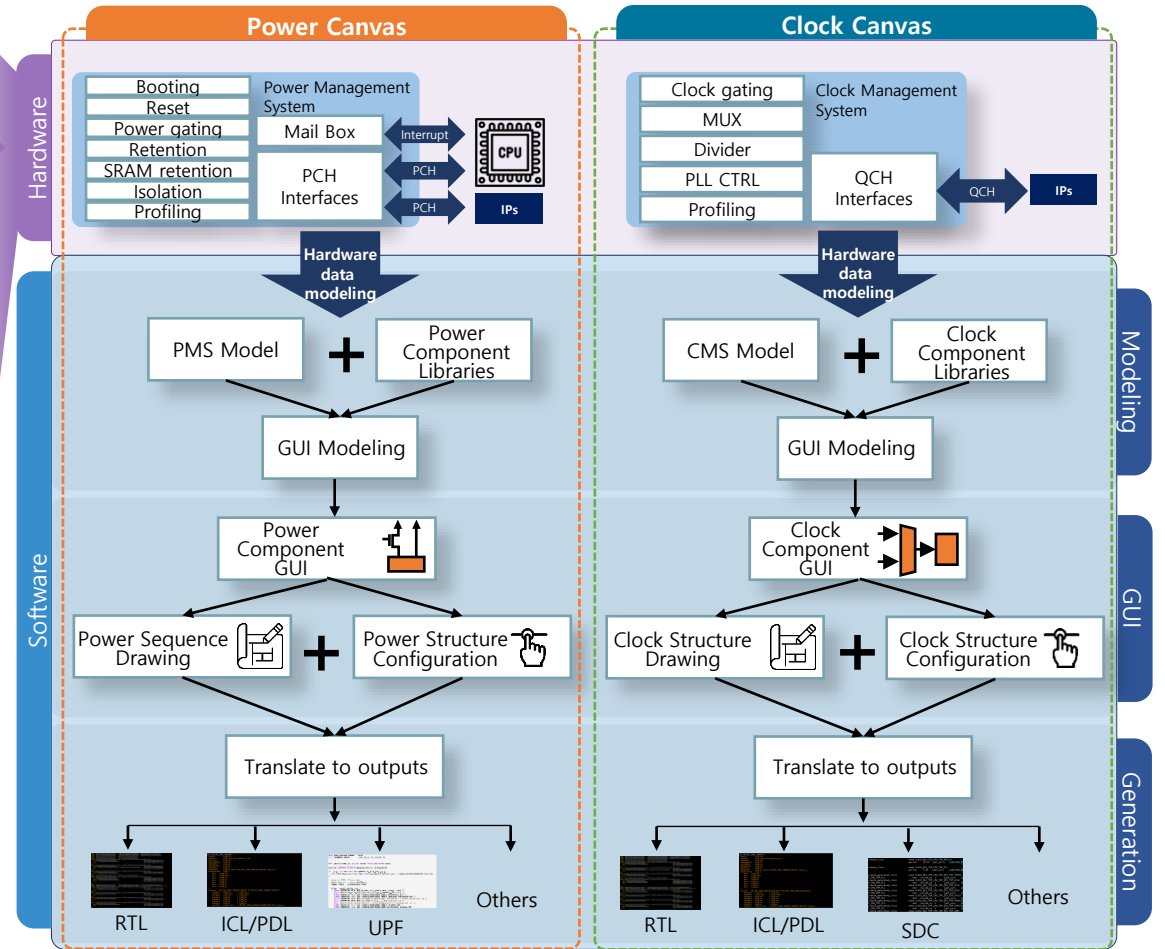
Target Power and Clock Management Architecture



To No-Code Framework

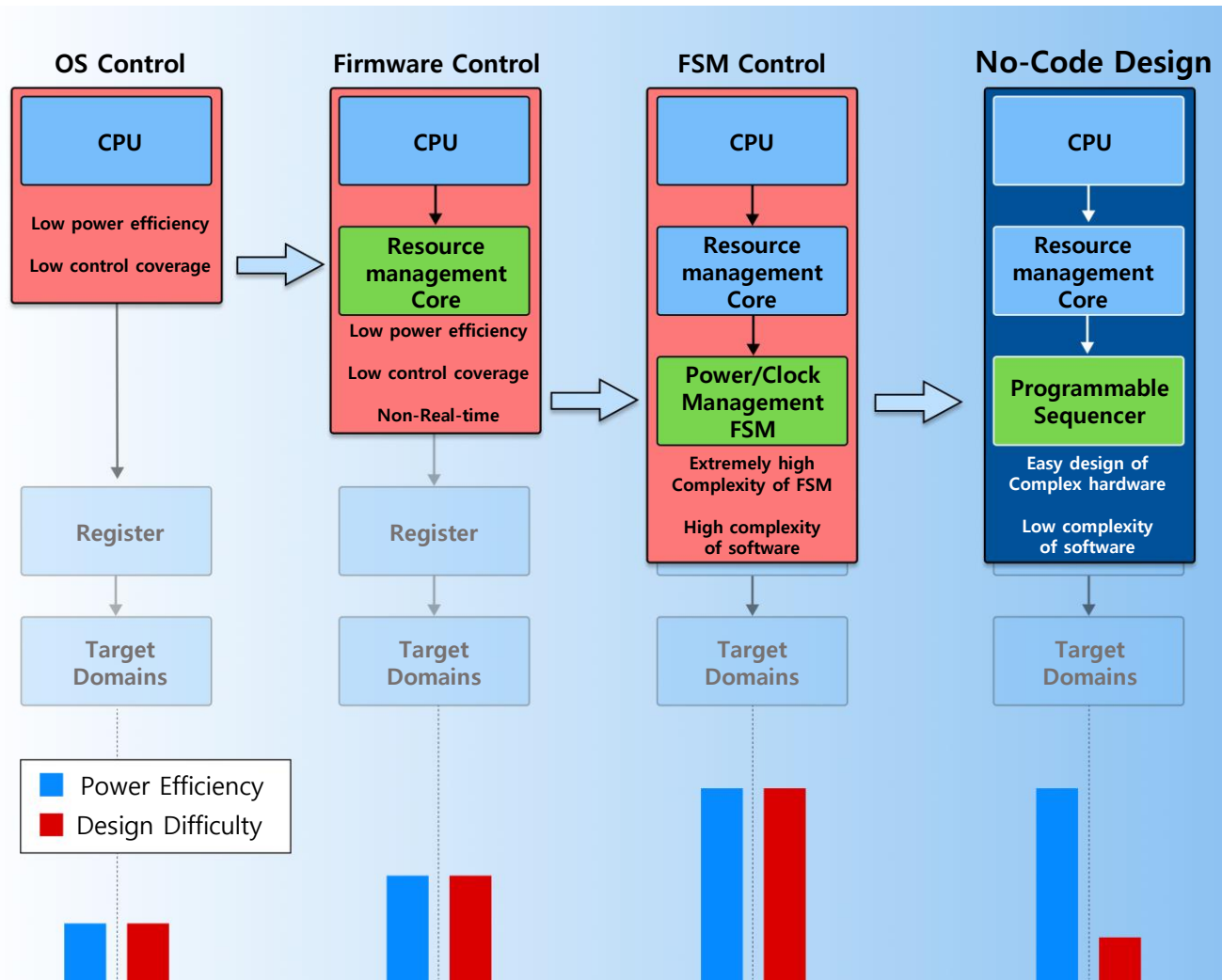
For power and clock systems, **domain dedicated controllers** are defined in system level.

Implementation using No-Code design platform



The Solution of Power Reduction

No-code design is highly efficient and easy at the same time



OS Control

- Non-real-time management
- Restricted control scope

Firmware Control

- Resource management core control
- Still non-real-time due to firmware latency

FSM Control

- Domain dedicated FSM
- Real-time management
- Complex hardware and software structures

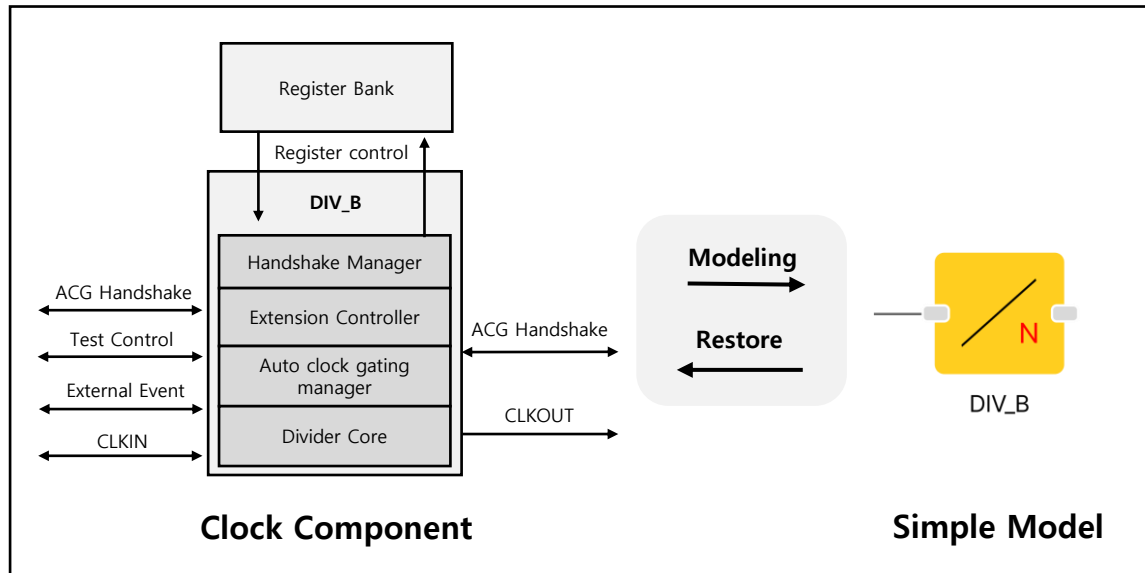
No-Code Design

- Power reduction based on real-time management
- Easy to design with simple model
- Consistent data for accurate outputs
- System design time reduction

Easy to Design with Simple Model

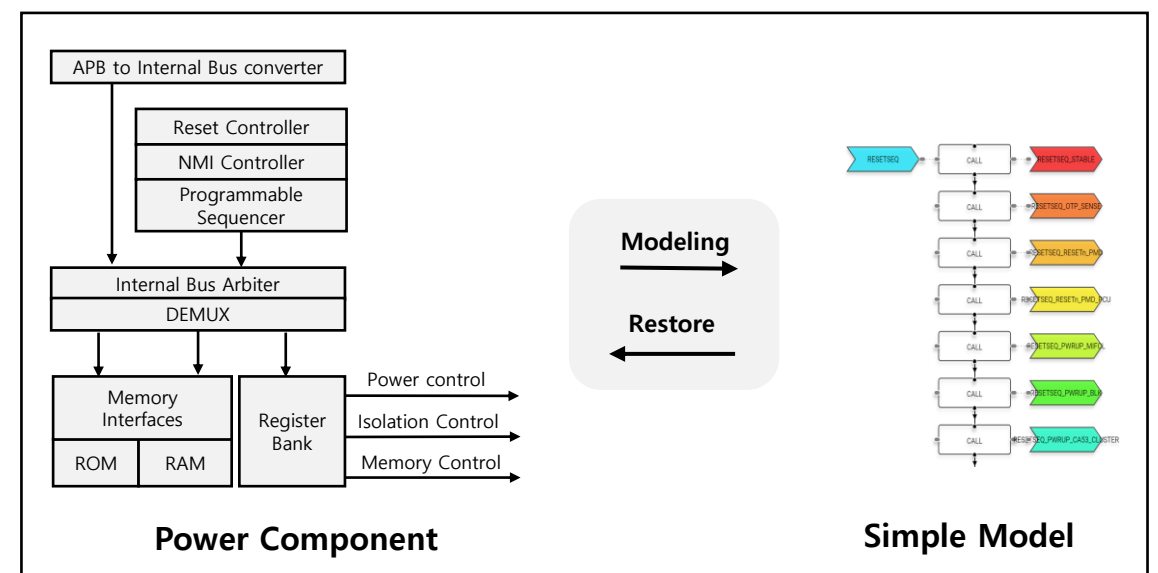
Complex Hardware Design into simple GUI Model

Clock Management System Modeling Example



- ACG handshake configures full network to replace common clock framework
- Other extension layers are pre-designed

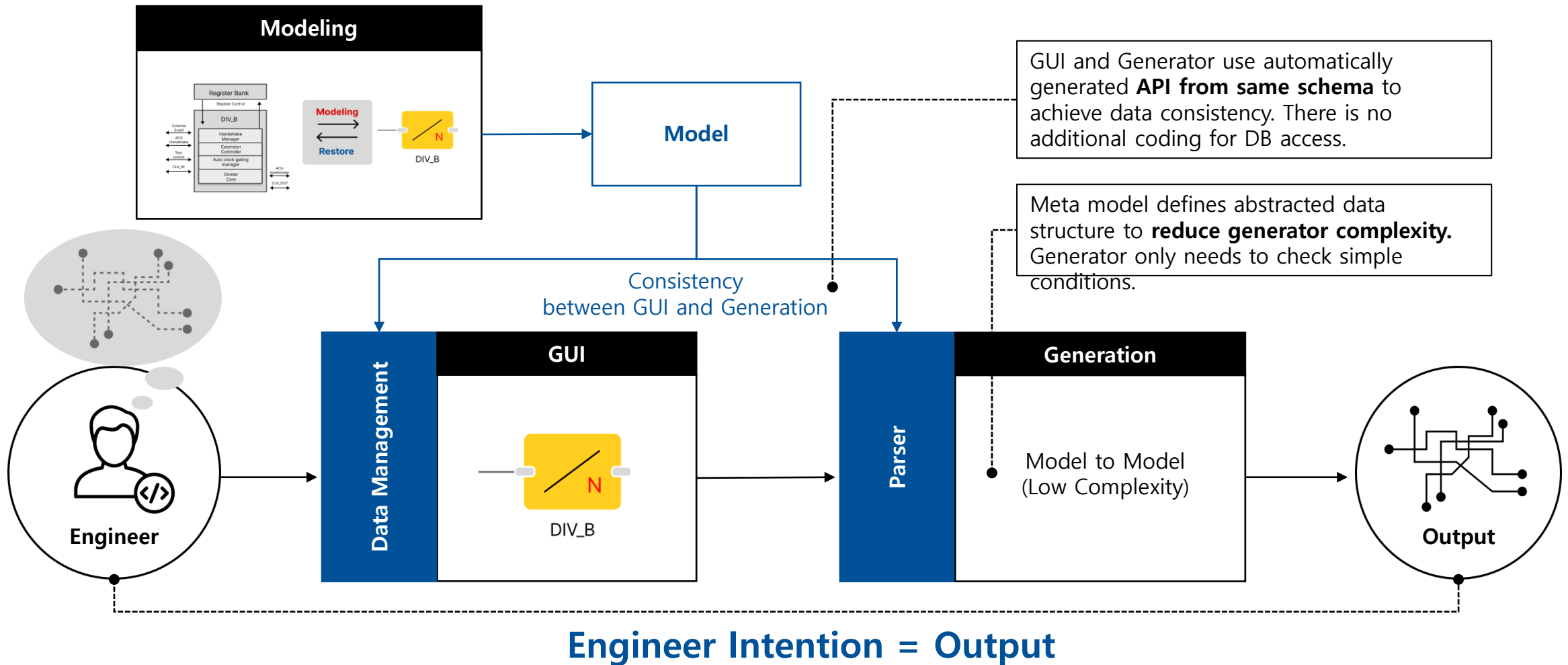
Power Management System Modeling Example



- Programmable sequencer for power management consists of the base power management unit
- Sequence data can be programmed in ROM or SRAM, and sequence is modeled simply in flow chart
- Related bus systems, controllers and registers are generated based on canvas information

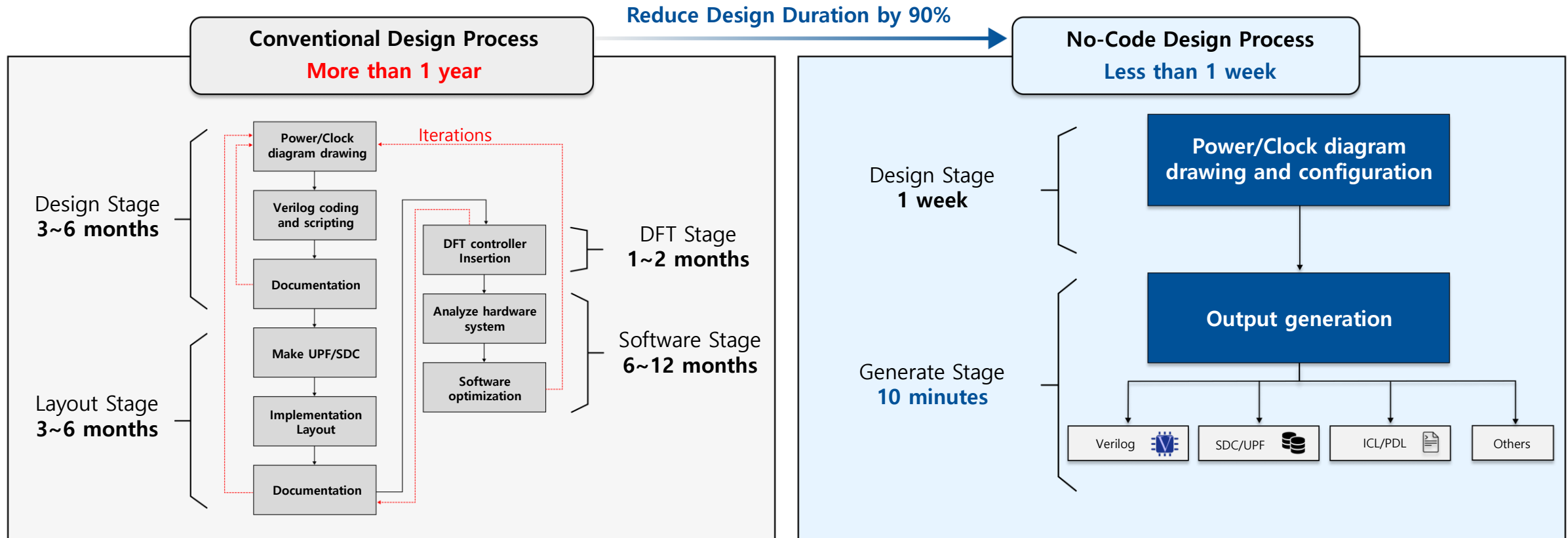
Consistent Data for Accurate Outputs

Software keeps data consistency between hardware design and GUI model



System Design Time Reduction

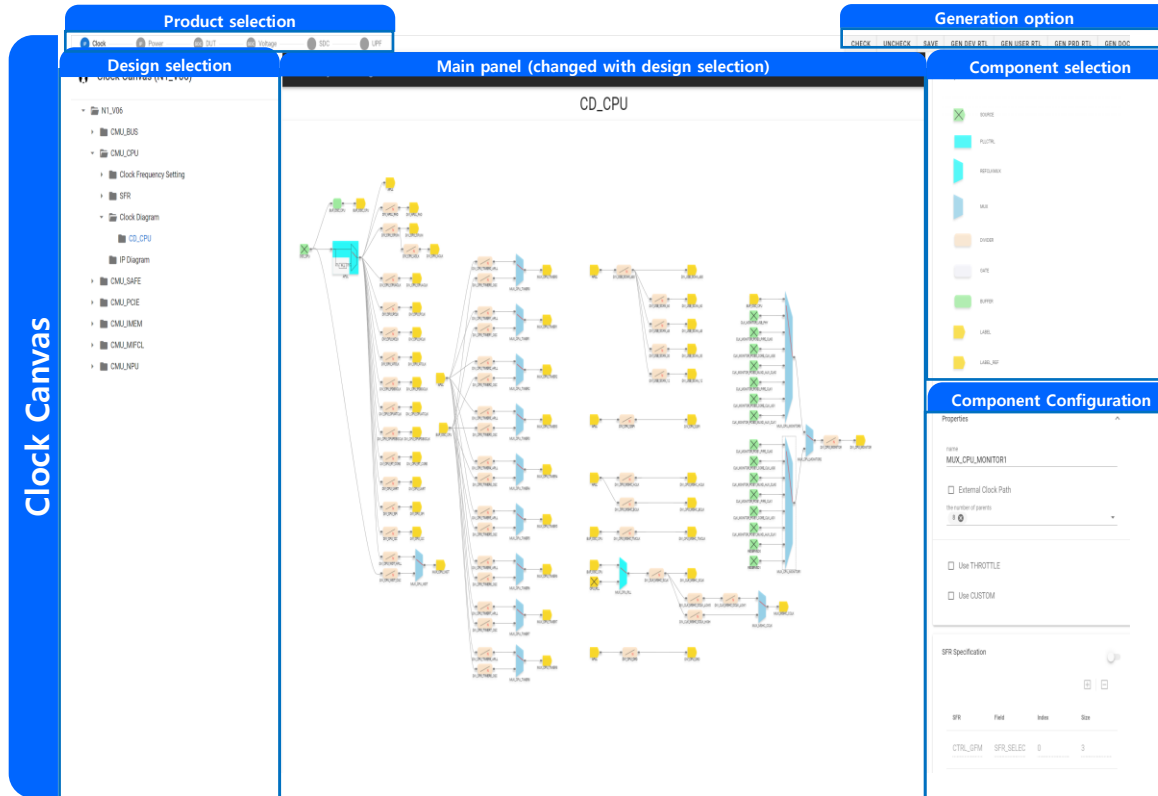
Design time is dramatically reduced in actual project



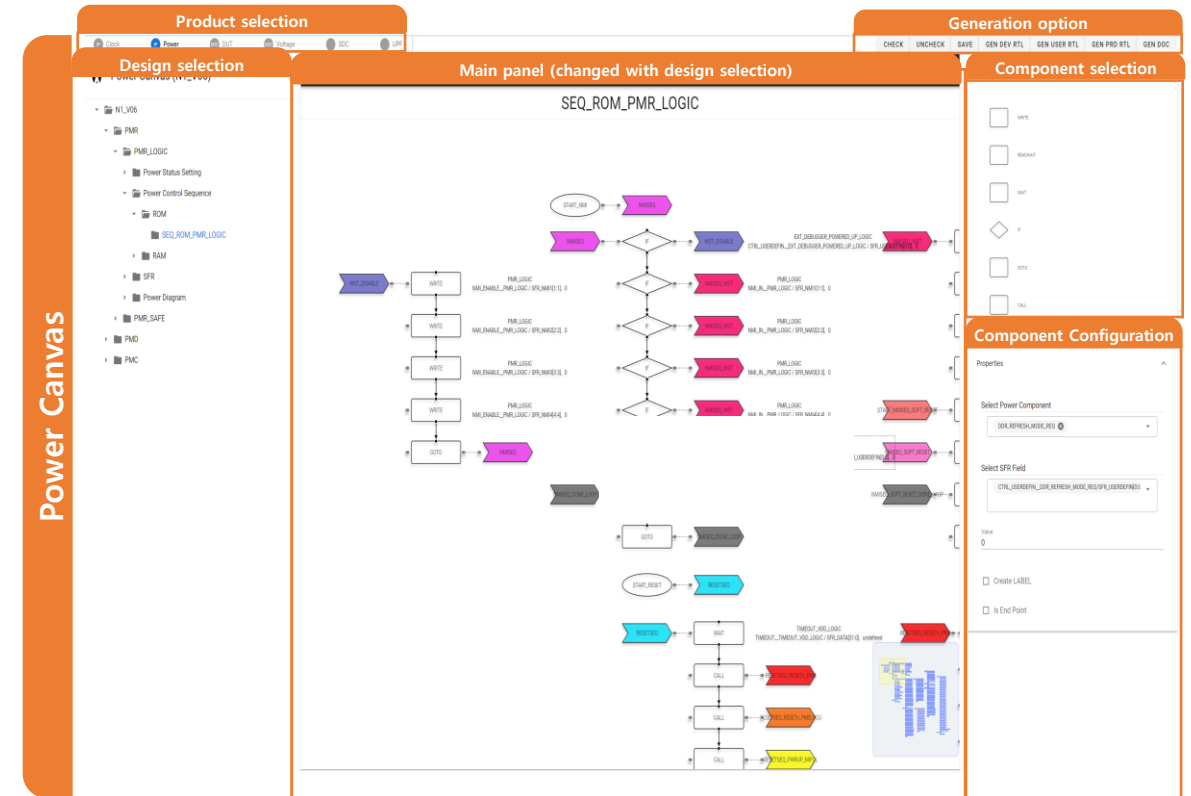
- Hardware design starts with an architectural diagram that usually misses detailed information
- Traditional design may lead to substantial unforeseen costs from data mismatches, especially with complex control schemes.
- A complete architectural diagram and configuration enable the generation of all required outputs for design phases

GUI Page Examples

All design data are represented in GUI pages



Clock Canvas includes **SDC pages** and **gating pages**



Power Canvas includes **UPF pages** and **signal pages**

Output Examples

Power/Clock Canvas generates all system design files w/ data consistency



- 1st customer and partner
- All power, clock management and booting systems are designed by power and clock canvas
- Full specification design changes
✓ less than 1 week
- All IP and system level verifications are cleared
- The designed power and clock systems meet all the regulation standards (LBIST, MBIST, and ASIL-D) for automotive SOC

[illegible]

```

comment (ADAPTER_ADR_HANDLE signals [15:0]_AND are open when AGC is
filter (Module == "PLL_ADR"
    (Statement == "ITDAS1318870730c17730c17540ab55ae911" OR
    Statement == "ITD406459fa6216802a25d5083672d" OR
    Statement == "ITD458a8dc375730ca140ab8086929dc2" ))
-tsg (M523)

mainve_line
_and (ITDA_COMMON_WAVE_M523_CMU_TOP_PLL_BB_ADAPTER_AGC_HANDSHAKE
comment (ADAPTER_AGC_Handshake signals [15:0]_AND are open when AGC is
filter (Module == "PLL_ADR"
    (Statement == "ITDAS131887299134044ce2513608a532d" OR
    Statement == "ITD406459fa6216802a25d5083672d" OR
    Statement == "ITD484c16e3c82027a61e081772171a" OR
    Statement == "ITD484c16e3c82027a61e081772171a" ))
-tsg (M523)

mainve_line
_and (ITDA_COMMON_WAVE_M523_CMU_TOP_MUX_ADR_ADAPTER_AGC_HANDSHAKE
comment (ADAPTER_AGC_Handshake signals [15:0]_AND are open when AGC is
filter (Module == "MUX_ADR" AND
    (Statement == "ITDAS131887299134044ce2513608a532d" OR
    Statement == "ITD406459fa6216802a25d5083672d" OR
    Statement == "ITD484c16e3c82027a61e081772171a" OR
    Statement == "ITD484c16e3c82027a61e081772171a" ))
-tsg (M523)

```

```
create_clock -name clock_clk_top_cmu_top_refclk  
-period 30.46 [get_ports {DESIGN}]  
  
create_clock -name clock_clk_top_cmu_top_pll  
-period 0.66 [get_pins {DESIGN}]  
  
create_generated_clock  
-cmu_top_pll  
-cmu_top_clk_divide_by 2 [get_pins  
-cmu_top_pll  
-cmu_top_pll] -cmu_top_refclk  
-cmu_top_refclk_divide_by 2 [get_pins  
-cmu_top_refclk  
-cmu_top_refclk] create_generated_clock  
-cmu_top_pll  
-cmu_top_clk_divide_by 4 [get_pins  
-cmu_top_pll  
-cmu_top_pll] -cmu_top_refclk  
-cmu_top_refclk_divide_by 4 [get_pins  
-cmu_top_refclk  
-cmu_top_refclk] create_generated_clock  
-cmu_top_pll  
-cmu_top_clk_divide_by 8 [get_pins  
-cmu_top_pll  
-cmu_top_pll] -cmu_top_refclk  
-cmu_top_refclk_divide_by 8 [get_pins  
-cmu_top_refclk  
-cmu_top_refclk]
```

SDC

[illegible][illegible]

```
# =====
# Power Domain
# =====
create_power_domain      BLK_PCIE_WRRAPPER

# =====
# BLK_PCIE_WRRAPPER::Power Supply - create_supply_port
# =====
create_supply_port      VDD0_PCIE_A0M      -domain
create_supply_port      VDD0M_PCIE          -domain
create_supply_port      VDD0D0_PCIE         -domain
create_supply_port      VDD0_PCIE          -domain
create_supply_port      VSS_PCIE_gd        -domain
create_supply_port      VSS_PCIE_vsscore    -domain
create_supply_port      VSSi               -domain
```

```

spirit::component mXns::spirit::http://www.spiritconsortium.org
+TDTA_Register_Map; spirit::name=sprite::baseAddress0x0/sprite
2c/spirit::size=spirit::volatile=true/spirit::volatile=spirit
+sprite::bitWidth8/sprite::bitWidth=8/spirit::access=read-only/s
+sprite::field=0/sprite::fieldWidth=8/spirit::access=read-only/s
+sprite::bitWidth=8/spirit::access=read-write/sprite::access=ro
+sprite::value=spirit::mask0xfffffff/sprite::mask=0/sprite
+sprite::bitWidth=1/sprite::bitWidth=8/spirit::access=read-only/s
+sprite::description=spirit::bitOffset16/spirit::bitOffset=spiri
+write=spirit::access=read-write/sprite::access=read-write/spr
+field=16/sprite::fieldWidth=8/spirit::FIELD_ENABLE=spirit::fi
+fieldWidth=20/spirit::bitWidth=spirit::access=read-write/spir
+bit::value=0x00/sprite::value=spirit::mask0xfffffff/sprite::m
+PIL_AAn/sprite::name=sprite::addressOffset0x10/sprite::add
+description=/spirit::bitOffset31/sprite::bitOffset=spirit::ac
+colorFormat=fields=channels=568 SPI_MCU_FCN_0

```

[illegible][illegible]

Summary

Complete power and clock system can be designed within No-code platform

- **Easy design**

- ✓ Engineer can use only simple GUI

- **Accurate design**

- ✓ Data consistency is guaranteed 100%

- **Fast design**

- ✓ All engineering outputs are generated in few minutes
- ✓ No communication loss

- **Power reduction**

- ✓ Hardware level power and clock management is fully supported
- ✓ No-Code platform can optimize SOC power very efficiently and effectively