




Arora V 物理制約 ユーザーガイド

SUG1018-1.2J, 2023-08-18

著作権について(2023)

著作権に関する全ての権利は、**Guangdong Gowin Semiconductor Corporation** に留保されています。

 **GOWIN**高云、Gowin、及びGOWINSEMIは、当社により、中国、米国特許商標庁、及びその他の国において登録されています。商標又はサービスマークとして特定されたその他全ての文字やロゴは、それぞれの権利者に帰属しています。何れの団体及び個人も、当社の書面による許可を得ず、本文書の内容の一部もしくは全部を、いかなる視聴覚的、電子的、機械的、複写、録音等の手段によりもしくは形式により、伝搬又は複製をしてはなりません。

免責事項

当社は、GOWINSEMI Terms and Conditions of Sale (GOWINSEMI取引条件)に規定されている内容を除き、(明示的か又は黙示的に拘わらず)いかなる保証もせず、また、知的財産権や材料の使用によりあなたのハードウェア、ソフトウェア、データ、又は財産が被った損害についても責任を負いません。当社は、事前の通知なく、いつでも本文書の内容を変更することができます。本文書を参照する何れの団体及び個人も、最新の文書やエラッタ(不具合情報)については、当社に問い合わせる必要があります。

バージョン履歴

日付	バージョン	説明
2023/04/20	1.0J	初版。
2023/06/30	1.1J	JTAGSEL_N Net、JTAG Net、SSPI Net、MODE Net、および I2C Net 制約を追加。
2023/08/18	1.2J	<ul style="list-style-type: none">● JTAG Net、SSPI Net、MODE Net、および I2C Net 制約の説明を更新。● MIPI_DPHY_RX プリミティブの制約例を追加。● Timing Paths 機能を削除。

目次

目次	i
図一覧	iii
表一覧	vi
1 本マニュアルについて	1
1.1 マニュアル内容	1
1.2 関連ドキュメント	1
1.3 用語、略語	2
1.4 テクニカル・サポートとフィードバック	3
2 物理制約構文仕様	4
2.1 I/O 位置制約	4
2.2 I/O 属性制約	5
2.3 プリミティブの位置制約	6
2.4 グループ制約	9
2.4.1 プリミティブグループ制約	9
2.4.2 相対位置グループ制約	11
2.5 リソースリザーブ制約	12
2.6 リファレンス電圧制約	12
2.7 グローバルクロック割り当て制約	13
2.8 グローバルクロック・プリミティブ制約	15
2.9 高速クロック・プリミティブ制約	16
2.10 その他の制約	17
2.10.1 ADC 入力電圧源制約	17
2.10.2 JTAGSEL_N Net の制約	18
2.10.3 JTAG Net の制約	18
2.10.4 SSPI Net の制約	19
2.10.5 MODE Net の制約	20
2.10.6 I2C Net の制約	21
3 FloorPlanner	23
3.1 概要	23

3.2 起動.....	24
3.3 インターフェース	25
3.3.1 メニューバー	26
3.3.2 Summary ウィンドウと Netlist ウィンドウ	37
3.3.3 Package View ウィンドウ	41
3.3.4 Chip Array ウィンドウ	45
3.3.5 Constraint 編集ウィンドウ	51
3.3.6 Message ウィンドウ	56
4 FloorPlanner の使用.....	57
4.1 制約ファイルの新規作成	57
4.2 制約ファイルの編集.....	59
4.2.1 制約編集の例	59
4.2.2 I/O 制約の編集.....	61
4.2.3 図プリミティブ制約の編集.....	62
4.2.4 グループ制約の編集.....	63
4.2.5 リザーブ制約の編集.....	67
4.2.6 グローバルクロック割り当て制約の編集.....	68
4.2.7 グローバルクロック制約の編集	69
4.2.8 高速クロック制約の編集	70
4.2.9 リファレンス電圧制約の編集	71

図一覧

図 3-1 メニューバーから起動	24
図 3-2 Process ウィンドウから起動	24
図 3-3 Start Page ウィンドウから起動	25
図 3-4 FloorPlanner の GUI	26
図 3-5 File メニュー	27
図 3-6 Open Physical Constraints	27
図 3-7 Constraints メニュー	28
図 3-8 プリミティブ検索ダイアログボックス	28
図 3-9 プリミティブグループの新規作成	30
図 3-10 正しいプリミティブグループ	30
図 3-11 無効な位置	31
図 3-12 無効な位置	31
図 3-13 相対位置グループの作成	32
図 3-14 正しい相対位置グループ	32
図 3-15 リザーブ制約	32
図 3-16 クロック制約	33
図 3-17 グローバルクロック制約の作成	34
図 3-18 高速クロック制約の作成	35
図 3-19 リファレンス電圧制約	35
図 3-20 Tools メニュー	35
図 3-21 Back-annotate Physical Constraints ダイアログボックス	36
図 3-22 Port の配置情報	36
図 3-23 View メニュー	37
図 3-24 Windows メニュー	37
図 3-25 Summary ウィンドウ	38
図 3-26 Netlist ウィンドウ	39
図 3-27 BUS と非 BUS を組み合わせた表示	40
図 3-28 階層表示	40
図 3-29 Netlist ウィンドウでの右クリックメニュー	41
図 3-30 GW5AT-138-FPBGA676A の Package view ウィンドウ	42

図 3-31 Package View の右クリックメニュー	43
図 3-32 差動ペア表示	43
図 3-33 Top View	44
図 3-34 Bottom View	44
図 3-35 Chip Array ウィンドウ	45
図 3-36 グリッドモードでの制約	46
図 3-37 マクロセル・モードでの制約	46
図 3-38 プリミティブモードでの制約	47
図 3-39 Chip Array の右クリックメニュー	49
図 3-40 Show Place View の表示	50
図 3-41 マウスオーバー表示	50
図 3-42 右クリックメニューによるハイライト表示	51
図 3-43 I/O 制約ウィンドウ	52
図 3-44 プリミティブ制約ウィンドウ	53
図 3-45 グループ制約ウィンドウ	53
図 3-46 リザーブ制約ウィンドウ	54
図 3-47 クロック制約ウィンドウ	55
図 3-48 グローバルクロック制約ウィンドウ	55
図 3-49 高速クロック制約ウィンドウ	56
図 3-50 Vref 制約ウィンドウ	56
図 3-51 Message ウィンドウ	56
図 4-1 制約ファイルの新規作成	57
図 4-2 デバイスの選択	58
図 4-3 出力ファイルの保存	59
図 4-4 Chip Array にドラッグして I/O Constraints を作成	61
図 4-5 Package View にドラッグして I/O Constraints を作成	62
図 4-6 Chip Array にドラッグして Primitive Constraints を作成	63
図 4-7 Group Constraints ウィンドウの右クリックメニュー	63
図 4-8 Primitive Group Constraints の作成	64
図 4-9 Primitive Group Constraints	65
図 4-10 Relative Group Constraints の作成	66
図 4-11 Relative Group Constraints	67
図 4-12 Resource Reservation 制約の作成	67
図 4-13 Resource Reservation	68
図 4-14 Clock Net Constraints の作成	69
図 4-15 Clock Net Constraints	69
図 4-16 GCLK Primitive Constraints の作成	70
図 4-17 GCLK Primitive Constraints	70

図 4-18 HCLK Primitive Constraints の作成	71
図 4-19 HCLK Primitive Constraints	71
図 4-20 Vref Constraints の作成	72
図 4-21 Chip Array ウィンドウにドラッグして Vref Constraints Location を生成	72
図 4-22 Package View ウィンドウにドラッグして Vref Constraints Location を生成	73
図 4-23 Vref Constraints 名前の重複	74

表一覧

表 1-1 用語、略語	2
表 2-1 DCS/DCE の制約可能な Position.....	16

1 本マニュアルについて

1.1 マニュアル内容

本マニュアルは、ユーザーが物理制約を迅速に実現できるよう、Gowin ソフトウェアの **FloorPlanner** の使用法と **Arora V FPGA** 製品の物理制約の構文仕様について説明します。ソフトウェアのアップデートにより、一部の内容が変更される場合があります。

1.2 関連ドキュメント

GOWIN セミコンダクターのホームページ www.gowinsemi.com/ja から、以下の関連ドキュメントをダウンロード及び閲覧できます。

- GW5AT シリーズ FPGA 製品データシート([DS981](#))
- GW5A シリーズ FPGA 製品データシート([DS1103](#))
- GW5AST シリーズ FPGA 製品データシート([DS1104](#))
- Gowin ソフトウェア ユーザーガイド([SUG100](#))
- GW5AT-138 デバイス Pinout([UG982](#))
- GW5A-25 デバイス Pinout([UG985](#))
- GW5AST-138 デバイス Pinout([UG986](#))
- Arora V Clock ユーザーガイド([UG306](#))
- Arora V FPGA 製品プログラミング・コンフィギュレーション ユーザーガイド([UG704](#))

1.3 用語、略語

表 1-1 に、本マニュアルで使用される用語、略語、及びその意味を示します。

表 1-1 用語、略語

用語、略語	正式名称	意味
BSRAM	Block SRAM	ブロック SRAM
CFU	Configurable Function Unit	コンフィギュラブル機能ユニット
CLKDIV	Clock Divider	クロック分周器
CLS	Configurable Logic Section	コンフィギュラブル論理セクション
DCS	Dynamic Clock Selector	ダイナミック・クロック・セクタ
DDRDLL	Double Data Rate Delay-locked Loop	ダブル・データ・レート遅延同期回路
DLLDLY	DLL Delay	DLL 遅延
DQS	Bidirectional Data Strobe Circuit for DDR Memory	双方向データストロブ回路
FloorPlanner	FloorPlanner	物理制約エディタ
FPGA	Field Programmable Gate Array	フィールド・プログラマブル・ゲート・アレイ
GCLK	Global Clock	グローバルクロック
I/O	Input/Output	入力/出力
IDE	Integrated Development Environment	統合開発環境
LUT	Look-up Table	ルックアップテーブル
LW	Long Wire	ロングワイヤ・リソース
PLL	Phase-locked Loop	位相同期回路
SSRAM	Shadow SRAM	分散 SRAM
VREF	Voltage Reference	リファレンス電圧

1.4 テクニカル・サポートとフィードバック

GOWIN セミコンダクターは、包括的な技術サポートをご提供しています。使用に関するご質問、ご意見については、直接弊社までお問い合わせください。

ホームページ : www.gowinsemi.com/ja

E-mail : support@gowinsemi.com

2 物理制約構文仕様

2.1 I/O 位置制約

I/O 位置制約により、**port** を指定の IO 位置に制約することができます。IO 位置の詳細については、対応する **Pinout** マニュアルを参照して下さい。

- GW5AT-138 デバイス Pinout([UG982](#))
- GW5A-25 デバイス Pinout([UG985](#))
- GW5AST-138 デバイス Pinout([UG986](#))

構文

```
IO_LOC "obj_name" obj_location [exclusive];
```

制約要素

obj_name

obj_name は、例えば **port** の **name** です。

obj_location

obj_location は IO 位置("A11"、"B12"など)で、複数の位置を指定する場合、"A11,B2"のように英語のコンマで区切る必要があります。

exclusive

exclusive はオプションであり、制約位置の後にあります。これは、制約ステートメントの **obj_location** には、**obj_name** で指定されたプリミティブのみを配置できることを示します。

注記：

obj_name が **escaped name** フォーマット(スラッシュで始まり、スペースで終わる)の場合、**obj_name** の前後に英語の引用符が必要です。

適用例

例 1

```
IO_LOC "io_1" A1;
```

// io_1 を A1 に制約します。

例 2

```
IO_LOC "io_1" A1, B14, A15;
```

// io_1 を A1、B14、A15 に制約し、配置の時、3 つの位置の中の最初の妥当な位置に配置します。

例 3

```
IO_LOC "io_2" A1 exclusive;
```

// io_2 を A1 に制約し、かつ A1 位置は io_2 専用です。

例 4

```
IO_LOC "io_2" A1, B14, A15 exclusive;
```

// io_2 を pin A1、B14、A15 に制約し、かつ A1、B14、A15 は io_2 専用です。

2.2 I/O 属性制約

I/O 属性制約は、I/O の属性値の設定に使用されます。例えば、port のレベル規格(IO_TYPE)、プルアップ/プルダウンモード(PULL_MODE)、ドライブ強度(DRIVE)等。属性設定の詳細については、『GW5AT シリーズ FPGA 製品データシート(DS981)』を参照してください。

構文

```
IO_PORT " obj_name " attribute = attribute_value;
```

1 つの制約ステートメントにおいては複数の属性を設定できます。各属性はスペースで区切ります。

制約要素

obj_name

obj_name は、例えば port の name です。

attribute 和 attribute value

属性制約が必要な Port の属性および属性値です。制約可能な属性およびその属性値の詳細については、『Arora V プログラマブル汎用 IO(GPIO)

ユーザーガイド([UG304](#))』を参照してください。

適用例

例 1

```
IO_PORT "port_1" IO_TYPE = LVTTL33;
```

// port_1 の IO_TYPE を LVTTL33 に設定します。

例 2

```
IO_PORT "port_2" IO_TYPE = LVTTL33 PULL_MODE =KEEPER;
```

// port_2 の IO_TYPE を LVTTL33 に設定し、PULL_MODE の属性値を KEEPER に設定します。

2.3 プリミティブの位置制約

Primitive Constraints は、プリミティブを指定の GRID に配置するために使用されます。Primitive Constraints により LUT、BSRAM、SSRAM、DSP、DDRDL、PLL、DQS、MIPI_DPHY_RXなどを制約することができます。

構文

```
INS_LOC "obj_name" obj_location [exclusive];
```

制約要素

obj_name

制約されるプリミティブの name です。

obj_location

obj_location には、以下の数タイプがあります：

1. LUT の位置制約

- LUT に指定される単一の位置情報(RxCy[0-3][A-B]など)。
- 以下のように複数の行または列を指定した位置情報の範囲：
 - 複数の CLS または LUT を含む："RxCy"、"RxCy[0-3]"。
 - 複数の行を指定する："R[x:y]Cm"、"R[x:y]Cm[0-3]"、"R[x:y]Cm[0-3][A-B]"。
 - 複数の列を指定する："RxC[m:n]"、"RxC[m:n][0-3]"、"RxC[m:n][0-3][A-B]"。
 - 複数の行及び列を指定する："R[x:y]C[m:n]"、"R[x:y]C[m:n][0-3]"、"R[x:y]C[m:n][0-3][A-B]"。

注記：

- LUT 位置情報の **x,m** は、GRID の行情報です。
- LUT 位置情報の **y,n** は、GRID の列情報です。
- LUT 位置情報の **R** は GRID の行を、**C** は GRID の列を意味します。
- LUT 位置情報の **0-3** は、GRID の特定の **CLS** の番号を意味します。
- LUT 位置情報の **A-B** は、CLS 位置の特定の LUT 位置の番号を意味します。

2. PLL の位置制約

PLL 位置制約の形式は"**PLL_L**"または"**PLL_R**"です。左側に複数の PLL を配置できる場合、"**PLL_L[0]**"、"**PLL_L[1]**" …のように、右側に複数の PLL を配置できる場合、"**PLL_R[0]**"、"**PLL_R[1]**"…のように設定することができます。

3. BSRAM の位置制約

BSRAM 位置制約の形式は"**BSRAM_R10[0]**"(10 行目の最初の BSRAM)、"**BSRAM_R10[1]**"…です。

4. DSP の位置制約

DSP 位置制約の形式は"**DSP_R37[0]**"(37 行目の最初の DSP Block)、"**DSP_R37[1]**"…です。MULT12X12 を指定したい場合、**DSP_R37[0][A]**または **DSP_R37[0][B]**のようにマークすることができます。

注記：

各 DSP Block は 2 つのマクロ(macro)で構成され、1 つのマクロは 1 つの MULT12X12 を配置可能な位置を指します。

5. DDRDLL の位置制約

DDRDLL 位置制約の形式は"**DDRDLLM_TL**"、"**DDRDLLM_TR**"…です。

6. MIPI_DPHY_RX の位置制約

MIPI_DPHY_RX 位置制約の形式は “**QUAD[0]**”、“**QUAD [1]**”。

注記：

MIPI_DPHY_RX の制約は、GW5A(S)(T)-138 でサポートされ、また、**exclusive** 機能をサポートしません。

exclusive

"**exclusive**"はオプションであり、制約位置の後にあります。これは、制約ステートメントの **obj_location** には、**obj_name** で指定されたプリミティブのみを配置できることを示します。

注記：

1つの制約ステートメントに複数の `obj_location` を含めることができます。この場合、`;` で区切ります。

適用例

例 1

```
INS_LOC "lut_1" R5C10[0][A];
```

// lut_1 を R5C10 の 1 つ目の CLS の 1 つ目の LUT の位置に制約します。

例 2

```
INS_LOC "ins_2 " R5C6[2] exclusive;
```

// ins_2 を R5C6 の 3 つ目の CLS の位置に制約し、かつこの位置は当該プリミティブ専用です。

例 3

```
INS_LOC "ins_3" R[2:6]C2;
```

// ins_3 を 2 行目から 6 行目、かつ 2 列目の範囲に制約します。

例 4

```
INS_LOC "ins_4" R[2:4]C[2:6] exclusive;
```

// ins_4 を 2 行目から 4 行目、かつ 2 列目から 6 列目の範囲に制約し、かつこの範囲は当該プリミティブ専用です。

例 5

```
INS_LOC "ins_5" R[2:4]C[2:6][1];
```

// ins_5 を 2 行目から 4 行目、かつ 2 列目から 6 列目の範囲内の何れか 1 つの GRID 位置の 2 つ目の CLS に制約します。

例 6

```
INS_LOC "reg_name" B14;
```

// reg_name を IO の B14 に制約します。

例 7

```
INS_LOC "pll_name" PLL_L[0];
```

// PLL の INS_LOC 制約により pll_name を左側 PLL の 1 つ目の位置に制約します。

例 8

```
INS_LOC "bsram_name" BSRAM_R10[2];
```

// BSRAM の INS_LOC 制約により bsram_name を 10 行目の 3 つ目の BSRAM に制約します。

例 9

```
INS_LOC "dsp_name" DSP_R19[2];
```

// DSP の INS_LOC 制約により dsp_name を 19 行目の 3 つ目の DSP Block に制約します。

例 10

```
INS_LOC "ddrdll_name" DDRDLLM_TL;
```

// DDRDLL の INS_LOC 制約により ddrdll_name をトップの左側 DDRDLL の位置に制約します。

例 11

```
INS_LOC "mipi_dphy_rx_name" QUAD[0];
```

// MIPI_DPHY_RX の INS_LOC 制約により mipi_dphy_rx_name を最初の MIPI_DPHY_RX の位置に制約します。

2.4 グループ制約

Group Constraints には、Primitive Group Constraints と Relative Group Constraints があります。

2.4.1 プリミティブグループ制約

Primitive Group 制約は、グループ制約の定義に使用されます。グループは、さまざまな Instance オブジェクトのコレクションです。Primitive Group 制約により、LUT、DFF、BSRAM、SSRAM、DSP、PLL、DDRDL、DQS などの Instance、または Buffer、IOLOGICなどを 1 つのグループに追加でき、このグループの位置を制約することでグループ内の全てのオブジェクトを位置制約できます。

構文

GROUP の定義 :

```
GROUP group_name = { "obj_names" } [exclusive];
```

Instance をグループに追加します :

```
GROUP group_name += { "obj_names" } [exclusive];
```

グループの位置を制約します：

```
GRP_LOC group_name group_location[exclusive];
```

注記：

`group_name` が **escaped name** フォーマット(スラッシュで始まり、スペースで終わる)の場合、`group_name` の前後に引用符が必要です。

制約要素

group_name

グループの名前。

obj_name

`obj_name` は指定の **Instance** オブジェクトをグループに追加するために使用されます。

group_location

この `group` の制約位置を指定し、`group_location` を IOB、GRID、BSRAM、DSP、PLL、DDRDLL の位置にできます。

exclusive

キーワード"**exclusive**"はオプションで、これはグループ定義ステートメントまたは位置制約ステートメントの後にあります。

1 つのオブジェクトを複数のグループに含めることができますが、グループ定義ステートメントの最後の "**exclusive**" キーワードは、グループ内のオブジェクトをこのグループにのみ含められることを表します。

位置制約ステートメントの後の "**exclusive**" は、この制約位置がこのグループ内のオブジェクトにより独占されることを表します。

適用例

例 1

```
GROUP group_1 = { "ins_1" "ins_2" "ins_3" "ins_4" };
```

// `group_1` という名前のグループを作成し、オブジェクト `ins_1`、`ins_2`、`ins_3`、`ins_4` をこのグループに追加します。

例 2

```
GROUP group_2 = { "ins_5" "ins_6" "ins_7" } exclusive;
```

// `group_2` という名前のグループを作成し、オブジェクト `ins_5`、`ins_6`、`ins_7` はこのグループにのみ含まれます。

例 3

```
GROUP group_1 += { "io_1" "io_2"};
```

// io_1、io_2 を group_1 に追加します。

例 4

```
GRP_LOC group_1 R3C4, A14, B4;
```

// group_1 内のオブジェクトを R3C4、A14、B4 に配置できます。

例 5

```
GRP_LOC group_2 R[2:3]C[2:4] exclusive;
```

// group_2 内の Instance オブジェクトを R[2:3]C[2:4]の範囲内に配置でき、かつこの範囲は group_2 内の Instance オブジェクト専用です。

例 6

```
GRP_LOC group_3 PLL_L[0],DDRLLM_TL,BSRAM_R10[0],  
DSP_R19[0];
```

// group_3 内のオブジェクトを、PLL_L[0]、DDRLLM_TL、BSRAM_R10[0]、DSP_R19[0]に配置できます。

2.4.2 相対位置グループ制約

Relative Group Constraints により、Instance オブジェクト(例えば LUT、REG、MUX)に対して相対位置制約をすることができます。

構文

Relative 制約のグループを定義します：

```
REL_GROUP group_name = { "obj_names" };
```

Instance を定義済みのグループに追加します：

```
REL_GROUP group_name += { "obj_names"};
```

グループ内の instance に対して相対位置制約を実行します：

```
INS_RLOC "obj_name" relative_location;
```

制約要素

obj_name

制約オブジェクトの名前。

relative_location

行と列の相対位置情報。

適用例

```
REL_GROUP grp_1 = { "ins_1" "ins_2" "ins_3" "ins_4" };
```

```
INS_RLOC "ins_1" R0C0;
```

```
INS_RLOC "ins_2" R2C3;
```

```
INS_RLOC "ins_3" R3C5;
```

// grp_1 という名前のグループ制約を定義し、ins_1、ins_2、ins_3、ins_4 を grp_1 に追加します。ins_1 を相対位置の原点 R0C0 とし、ins_2 を ins_1 に対して R2C3 に制約し、ins_3 を ins_1 に対して R3C5 に制約します。

2.5 リソースリザーブ制約

Resource Reservation 制約により、指定の位置または範囲の配置を回避できます。

構文

```
LOC_RESERVE location [ res_obj ];
```

適用例

例 1

```
LOC_RESERVE R2C3[0][A] -LUT;
```

```
LOC_RESERVE R2C3[0][A] -REG;
```

例 2

```
LOC_RESERVE IOR3, IOR6, R2C3, R3C4;
```

例 3

```
LOC_RESERVE R[2:5]C[3:6], R3C[8:9];
```

// 上記の例の制約の位置は配置の段階で予約されます。

2.6 リファレンス電圧制約

チップは、BANK 全体で有効な外部リファレンス電圧の入力をサポートします。Vref Constraints を使用して、外部リファレンス電圧の入力ピンの名前と位置を制約することができます。

注記：

- 外部リファレンス電圧の入力ピンの位置には、IOLOGIC リソースが必要です。

- **Vref Constraints** は、**Port** 属性制約と併用する必要があります。入力または入出力タイプのシングルエンド **PORT** の **IO** 規格が **SSTL / HSTL** の場合、**Vref** 属性を作成された **Vref Constraints** に設定できます。これは、**PORT** のリファレンス電圧として **Vref Constraints** 位置の外部入力リファレンス電圧が使用されることを意味します。

構文

```
USE_VREF_DRIVER vref_name [location];
```

制約要素

vref_name

ユーザーカスタマイズの **VREF** pin name。

location

GRID の任意の **I/O** 位置(**IOLOGIC** リソースを含む)は、**VREF** pin 制約の **location** として使用できます。

適用例

例 1

```
USE_VREF_DRIVER vref_pin;
```

```
IO_PORT "port_1" IO_TYPE = SSTL18_I VREF=vref_pin;
```

```
IO_PORT "port_2" IO_TYPE = SSTL18_I VREF=vref_pin;
```

// "vref_pin" という名前の **VREF** pin を定義し、**port_1** と **port_2** の **VREF** 属性を **vref_pin** に設定します。

例 2

```
USE_VREF_DRIVER vref_pin E16;
```

```
IO_LOC "port_1" C16;
```

```
IO_PORT "port_1" IO_TYPE = SSTL18_I VREF=vref_pin;
```

// "vref_pin" という名前の **VREF** pin を定義し、それを **E16** に制約します。**port_1** の **VREF** 属性を **vref_pin** に設定して **C16** に制約します。**port_1** が制約された位置は、**E16** と同じ **Bank** 上にある必要があります。

2.7 グローバルクロック割り当て制約

Clock Net Constraints は、特定の **net** の、グローバルクロック配線または非クロック配線の制約です。

- **BUFG[0-15]** は、**net** のグローバルクロック・ライン配線の制約を表し

ます。

- **LOCAL_CLOCK** は、**net** を非クロック配線に制約することを示します。

CLK 信号はクロックピンに接続される信号、**CE** 信号はクロックイネーブルピンに接続される信号、**SR** 信号は **SET**、**RESET**、**CLEAR**、および **PRESET** ピンに接続される信号、**LOGIC** はロジック入力ピンに接続される信号です。

構文

```
CLOCK_LOC "net_name" global_clocks = signal_type;
```

制約要素

net_name

net の名前。

global_clocks

BUFG[0-15] : 特定のグローバルクロック・ラインに配線します。

BUFG : グローバルクロック・ラインに配線します。

LOCAL_CLOCK : クロック・ラインに配線しません。

signal_type

CLK : **signal_type** がクロックピンである **net**。

CE : **signal_type** がクロックイネーブルピンである **net**。

SR : **signal_type** が **SET**、**RESET**、**CLEAR**、**PRESET** ピンである **net**。

LOGIC : **signal_type** が上記 **signal_type** でない **net**。

複数の **signal_type** を指定する場合、**|** を使用して区切ることができます。

注記 :

global_clocks として **LOCAL_CLOCK** を選択した場合、**signal_type** は選択不可になります。

適用例

例 1

```
CLOCK_LOC "net" BUFG = CLK|CE;
```

```
NET_LOC "net" BUFG = CLK|CE;
```

// **signal_type** がクロックピンまたはクロックイネーブルピンである **net** をグローバルクロック・ライン配線のように制約します。

例 2

```
CLOCK_LOC "net" LOCAL_CLOCK;
```

// "net" を非クロック・ライン配線のように制約します。

例 3

```
CLOCK_LOC "net" BUFG[0] = CLK;
```

// **signal_type** がクロックピンである **net** を 1 つ目のグローバルクロック・ラインに配線制約します。

2.8 グローバルクロック・プリミティブ制約

GCLK Primitive Constraints は、DCS、DCE などのグローバルクロックオブジェクトを指定された位置に制約するために使用されます。

構文

```
INS_LOC "obj_name" position;
```

制約要素

obj_name

制約オブジェクトの名前。

position

表 2-1 DCS/DCE の制約可能な Position

デバイス	位置	
	DCE	DCS
GW5AT-138	PTR0、PTR1、PTR2 PTR3、 PTR0[0~5]、PTR1[0~5]、 PTR2[0~5]、PTR3[0~5]、 PBR0、PBR1、PBR2、PBR3、 PBR0[0~5]、PBR1[0~5]、 PBR2[0~5]、PBR3[0~5]、PG、 SG、PG[0~11]、SG[0~15]	PTR0、PTR1、PTR2 PTR3、 PTR0[0~1]、PTR1[0~1]、 PTR2[0~1]、PTR3[0~1]、PBR0、 PBR1、PBR2、PBR3、 PBR0[0~1]、PBR1[0~1]、 PBR2[0~1]、PBR3[0~1]、PG、 PG[0~3]
GW5A-25	TOPLEFT、TOPRIGHT、 BOTTOMLEFT、 BOTTOMRIGHT、STOP、 SBOTTOM	TOPLEFT、TOPRIGHT、 BOTTOMLEFT、 BOTTOMRIGHT

注記：

DCS/DCE の制約可能な Position の詳細については、『Arora V Clock ユーザーガイド (UG306)』を参照して下さい。

適用例

```
INS_LOC "dcs_name" PTR0[1];

// DCS オブジェクト dcs_name を PTR0[1]位置に制約します。
```

2.9 高速クロック・プリミティブ制約

HCLK Primitive Constraints 制約により、CLKDIV、DLLDLY を指定の高速クロック位置に制約できます。CLKDIV、DLLDLY の制約位置は、通常の instance オブジェクトの制約位置とは異なります。"BOTTOMSIDE"、"LEFTSIDE"、および"RIGHTSIDE"は、制約位置の側を示します。

構文

```
INS_LOC "obj_name" position;
```

制約要素

obj_name

obj_name は、CLKDIV/DLLDLY の instance name です。

position

CLKDIV の制約可能な Position (GW5AT-138) : BOTTOMSIDE[0~7]、LEFTSIDE[0~7]、RIGHTSIDE[0~7]

CLKDIV の制約可能な Position (GW5A-25) : LEFTSIDE[0~3]、RIGHTSIDE[0~3]、BOTTOMSIDE[0~3]、TOPSIDE[0~3]

DLLDLY の制約可能な Position (GW5AT-138) : BOTTOMSIDE[0~3]、LEFTSIDE[0~3]、RIGHTSIDE[0~3]

DLLDLY の制約可能な Position (GW5A-25) : BOTTOMSIDE[0~1]、LEFTSIDE[0~1]、RIGHTSIDE[0~1]、TOPSIDE[0~1]

適用例

```
INS_LOC "clkdiv_name" LEFTSIDE[0];  
  
// clkdiv_name を LEFTSIDE[0]に配置します。
```

2.10 その他の制約

2.10.1 ADC 入力電圧源制約

ADC 入力電圧源は外部 IO から取得することができ、ADC 入力電圧源制約を使用して Bank0/2/3/4/5/6/7 の IO 位置を ADC 入力電圧源の入口として指定することができます。

ADC 入力電圧源制約には、2 つの制約構文(bus0 と bus1)があります。ここで、bus0 は Bank0/6/7 の IO 位置に対応し、bus1 は Bank2/3/4/5 の IO 位置に対応します。

注記：

ADC 入力電圧源制約対応デバイス：GW5A-25。

構文

```
USE_ADC_SRC bus0 location
```

```
USE_ADC_SRC bus1 location
```

制約要素

location

location は、IO 位置であり、IOB のみ(例えば、IOT48)をサポートします。

注記：

IO 位置 IOL25/IOL31/IOB45/IOR31/IOR5/IOT43/IOT3 は使用不可です。

適用例

```
USE_ADC_SRC bus0 IOT48
```

// IO 位置 IOT48 を ADC 入力電圧源用の外部入力位置として使用します。

USE_ADC_SRC bus1 IOR24

// IO 位置 IOR24 を ADC 入力電圧源用の外部入力位置として使用します。

2.10.2 JTAGSEL_N Net の制約

FPGA の内部ロジックで JTAGSEL_N 機能を制御する場合、すなわち、JTAGSEL_N を Low にプルダウンすることで JTAG ピンをコンフィギュレーション機能に切り替え、電源を落とさずに再度ダウンロードできるようにするには、JTAGSEL_N の net の物理制約を追加する必要があります。

構文

```
NET_LOC "obj_name" V_JTAGSELN;
```

制約要素

obj_name

obj_name として内部ロジックの配線可能な net を指定します。

適用例

```
NET_LOC "netname" V_JTAGSELN;
```

// この netname という net を使用して JTAGSEL_N の機能を制御します

2.10.3 JTAG Net の制約

JTAG ピンには、TCK、TMS、TDI、TDO が含まれます。ここで、TCK、TMS、TDI 機能は、専用コンフィギュレーション IO で実装するか、FPGA 内部ロジックによる制御で実装することができます。ただし、この 2 つの方法は、互いに排他的であり、つまりこれらの機能ピンは同じ方法でしか実装できません。また、TDO は専用コンフィギュレーション IO でしか実装できず、TCK、TMS、TDI の実装方法に影響されません。

FPGA の内部ロジックで JTAG の TCK、TMS、TDI の機能を制御する場合、JTAG の net の物理制約を追加する必要があります。JTAG の使用については、『Arora V FPGA 製品プログラミング・コンフィギュレーションユーザーガイド([UG704](#))』を参照してください。

構文

```
NET_LOC " obj_Name" V_TCK;
```

```
NET_LOC " obj_Name" V_TMS;
```

```
NET_LOC " obj_Name" V_TDI;
```

制約要素

obj_name

obj_name として内部ロジックの配線可能な net を指定します。

適用例

例

```
NET_LOC "netname" V_TCK;
```

```
// この netname という net を使用して TCK の機能を制御します
```

```
NET_LOC " netname " V_TMS;
```

```
// この netname という net を使用して TMS の機能を制御します
```

```
NET_LOC " netname " V_TDI;
```

```
// この netname という net を使用して TDI の機能を制御します
```

2.10.4 SSPI Net の制約

SSPI ピン(SI、SO、SSPI_WPN、CLKHOLD_N、SSPI_CLK、SSPI_CS_Nを含む)は、専用コンフィギュレーション IO で実装するか、FPGA 内部ロジックによる制御で実装することができます。ただし、この2つの方法は、互いに排他的であり、つまりこれらの機能ピンは同じ方法でしか実装できません。

FPGA の内部ロジックで SSPI の機能を制御する場合、SSPI の net の物理制約を追加する必要があります。SSPI の使用については、『Arora V FPGA 製品プログラミング・コンフィギュレーション ユーザーガイド ([UG704](#))』を参照してください。

構文

```
NET_LOC " obj_Name" V_SSPI SI;
```

```
NET_LOC " obj_Name" V_SSPI SO;
```

```
NET_LOC " obj_Name" V_SSPI WPN;
```

```
NET_LOC " obj_Name" V_SSPI CLKHOLDN;
```

```
NET_LOC " obj_Name" V_SSPI CLK;
```

```
NET_LOC " obj_Name" V_SSPI CSN;
```

制約要素

obj_name

obj_name として内部ロジックの配線可能な net を指定します。

適用例

例

```
NET_LOC "netname" V_SSPI SI;
```

```
// この netname という net を使用して SI の機能を制御します
```

```
NET_LOC "netname" V_SSPI SO;
```

```
// この netname という net を使用して SO の機能を制御します
```

```
NET_LOC "netname" V_SSPI WPN;
```

```
// この netname という net を使用して SSPI_WPN の機能を制御しま  
す
```

```
NET_LOC "netname" V_SSPI CLKHOLDN;
```

```
// この netname という net を使用して CLKHOLD_N の機能を制御し  
ます
```

```
NET_LOC "netname" V_SSPI CLK;
```

```
// この netname という net を使用して SSPI_CLK の機能を制御しま  
す
```

```
NET_LOC "netname" V_SSPI CSN;
```

```
// この netname という net を使用して SSPI_CS_N の機能を制御しま  
す
```

2.10.5 MODE Net の制約

MODE ピン(MODE0、MODE1、MODE2 を含む)は、専用コンフィギュレーション IO で実装するか、FPGA 内部ロジックによる制御で実装することができます。ただし、この 2 つの方法は、互いに排他的であり、つまりこれらの機能ピンは同じ方法でしか実装できません。

FPGA の内部ロジックで MODE の機能を制御する場合、MODE0、MODE1、MODE2 に加え、ロード信号という MODE_LD 信号があります。内部ロジックで制御される MODE0～MODE2 の値は MODE_LD 信号の立ち上がりエッジでロードされ、つまり MODE_LD が立ち上がりエッジの際に MODE0～MODE2 の値が切り替えられます。また、最初の MODE_LD の立ち上がりエッジが来る前に使用されるのは、ビットストリームのダウ

ンロード際の **MODE0**～**MODE2** の設定値です。FPGA の内部ロジックで **MODE** の機能を制御する場合、**MODE** の **net** の物理制約を追加する必要があります。**MODE** の使用については、『Arora V FPGA 製品プログラミング・コンフィギュレーション ユーザーガイド([UG704](#))』を参照してください。

構文

```
NET_LOC " obj_Name" V_MODE0;  
NET_LOC " obj_Name" V_MODE1;  
NET_LOC " obj_Name" V_MODE2;  
NET_LOC "obj_Name" V_MODE_LD;
```

制約要素

obj_name

obj_name として内部ロジックの配線可能な **net** を指定します。

適用例

例

```
NET_LOC " netname" V_MODE0;  
// この netname という net を使用して MODE0 の機能を制御します  
NET_LOC " netname" V_MODE1;  
// この netname という net を使用して MODE1 の機能を制御します  
NET_LOC " netname" V_MODE2;  
// この netname という net を使用して MODE2 の機能を制御します  
NET_LOC " netname" V_MODE_LD;  
// この netname という net を使用して MODE_LD の機能を制御しま  
す
```

2.10.6 I2C Net の制約

I2C ピン(**SCL**、**SDA** を含む)は、専用コンフィギュレーション **IO** で実装するか、FPGA 内部ロジックによる制御で実装することができます。ただし、この 2 つの方法は、互いに排他的であり、つまりこれらの機能ピンは同じ方法でしか実装できません。

FPGA の内部ロジックで **I2C** の機能を制御する場合、**I2C** の **net** の物理制約を追加する必要があります。

構文

```
NET_LOC " obj_Name" V_SCL;
```

```
NET_LOC " obj_Name" V_SDA;
```

制約要素

obj_name

obj_name として内部ロジックの配線可能な **net** を指定します。

適用例

例

```
NET_LOC " netname" V_SCL;
```

```
// この netname という net を使用して SCL の機能を制御します
```

```
NET_LOC " netname" V_SDA;
```

```
// この netname という net を使用して SDA の機能を制御します
```

注記：

I2C net の制約をサポートするデバイス：GW5A-25。

3FloorPlanner

3.1 概要

FloorPlanner は GOWIN セミコンダクターが市場向けに独自に研究開発した物理制約エディタです。I/O、プリミティブ、Group などの属性及び位置情報の読み出しと編集をサポートすると同時に、ユーザーの構成に基づき新しい配置ファイルと制約ファイルを生成できます。これらのファイルは、I/O の属性情報と位置情報、プリミティブや Group の位置情報などを指定しています。FloorPlanner は、シンプルで使いやすい配置および制約編集機能を提供して、物理制約ファイルの作成の効率を向上させています。

以下は、FloorPlanner の特徴です。

- ユーザーデザインの読み込み、制約ファイルの読み込み、編集、および出力をサポート
- ユーザーデザインの IO Port、Primitive、Group 制約情報などの表示をサポート
- ユーザーによる制約情報の新規作成、編集、および変更をサポート
- Chip Array のグリッドモード、マクロセル・モード、およびプリミティブモードでの表示をサポート
- Package 情報に基づく Package View 表示をサポート
- Chip Array と Package View の同期表示をサポート
- 制約位置情報のリアルタイム表示および差別化表示をサポート
- ドラッグによる位置情報の設定をサポート
- IO Port の属性のバッチ構成をサポート
- Clock Net Constraints の表示および編集をサポート
- 制約情報の有効性チェックをサポート

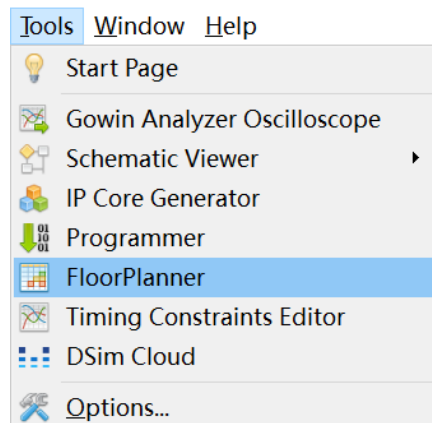
- Back-annotate Physical Constraints 機能をサポート

3.2 起動

FloorPlanner は、次の 3 つの方法で起動できます。

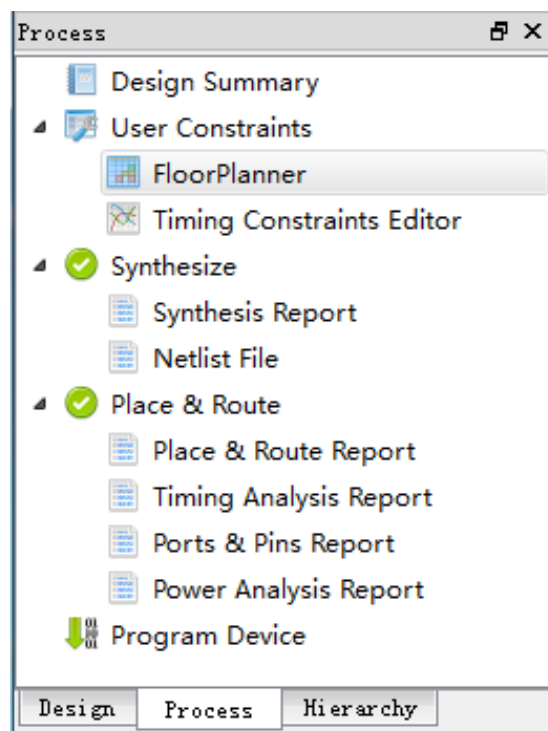
1. "Gowin ソフトウェア > Tools > FloorPlanner"をクリックして開きます (図 3-1)。

図 3-1 メニューバーから起動



2. プロジェクトの合成に成功した後、**Process** ウィンドウで “FloorPlanner” をダブルクリックします(図 3-2)。

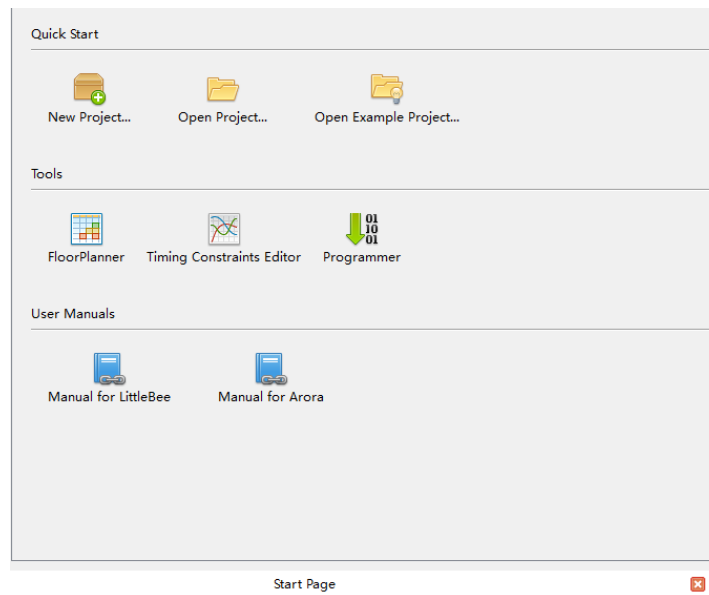
図 3-2 Process ウィンドウから起動



3. "IDE > Start Page > Tools > FloorPlanner"をクリックして

"FloorPlanner"を起動します(図 3-3)。

図 3-3 Start Page ウィンドウから起動



注記：

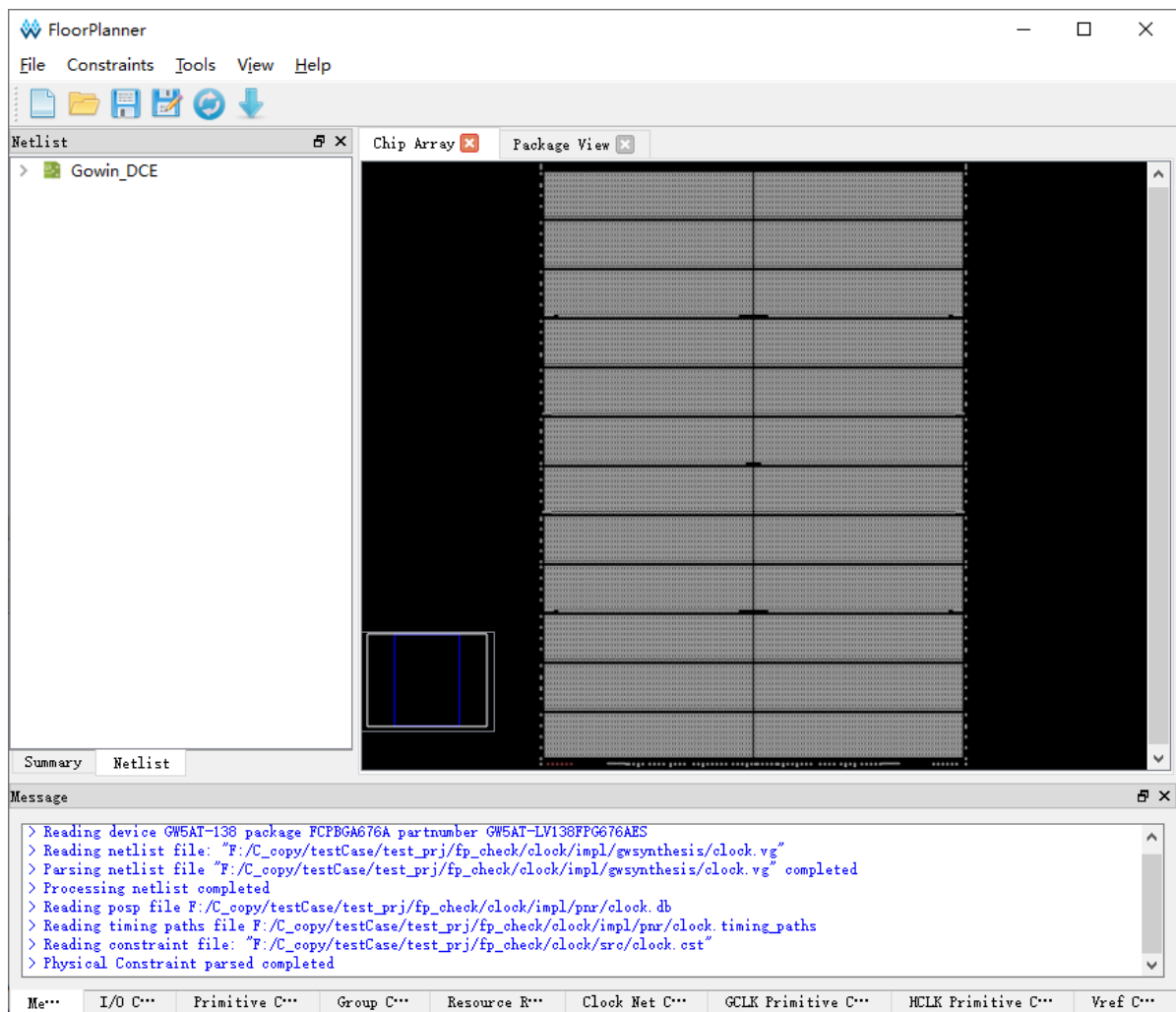
- FloorPlanner による制約が必要な場合、まずネットリストファイルをロードしてください。
- 1つ目の方法と2つ目の方法で FloorPlanner を起動する場合、プロジェクトのネットリストファイルが自動的にロードされます。
- 3つ目の方法で FloorPlanner を起動する場合、"File>New"でネットリストファイルをロードする必要があります。

3.3 インターフェース

FloorPlanner の GUI を図 3-4 に示します。

GUI には、メニューバー、ツールバー、Netlist ウィンドウ、Summary ウィンドウ、Chip Array ウィンドウ、Package View ウィンドウ、Message ウィンドウ、及び各制約編集ウィンドウなどがあります。

図 3-4 FloorPlanner の GUI



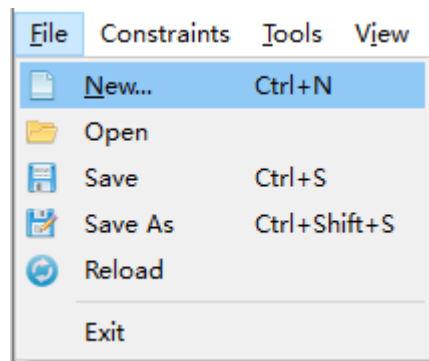
3.3.1 メニューバー

FloorPlanner のメニューバーには、"File"、"Constraints"、"Tools"、"View"、および"Help"の 5 つのサブメニューがあります。

File メニュー

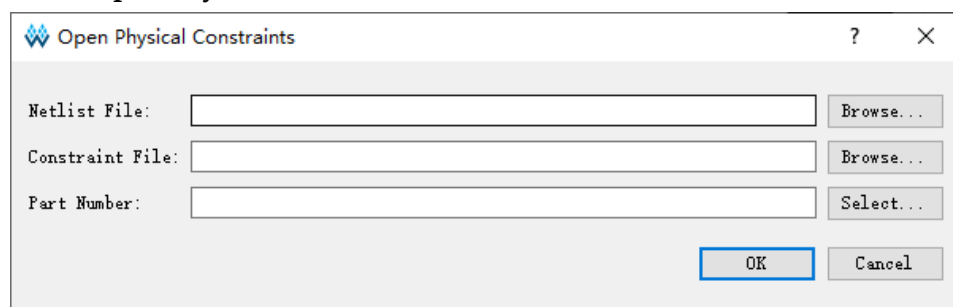
File メニューを図 3-5 に示します。

図 3-5 File メニュー



- **New** : 制約を新規作成します。
- **Open** : 制約を開きます(図 3-6)。
- **Reload** : 物理制約ファイルや配置ファイルなどを変更した場合、ファイルをリロードできます。
- **Save** : 現在の制約情報の変更によって元の制約ファイルを上書きします。
- **Save As** : 現在の制約情報の変更をユーザー指定のファイルに出力します。デフォルトの制約ファイル名はネットリストのファイル名で、変更可能です。
- **Exit** : FloorPlanner を終了します。

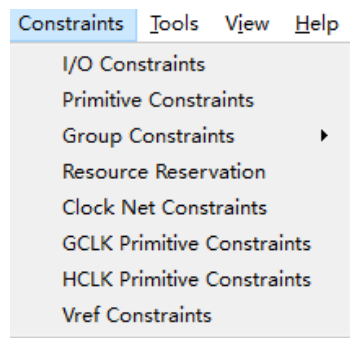
図 3-6 Open Physical Constraints



Constraints メニュー

Constraints メニューを図 3-7 に示します。

図 3-7 Constraints メニュー



Primitive Constraints

Primitive を選択して対応する制約を作成します。右クリックして **"Select Primitives"** をクリックすると、図 3-8 に示すダイアログボックスがポップアップします。

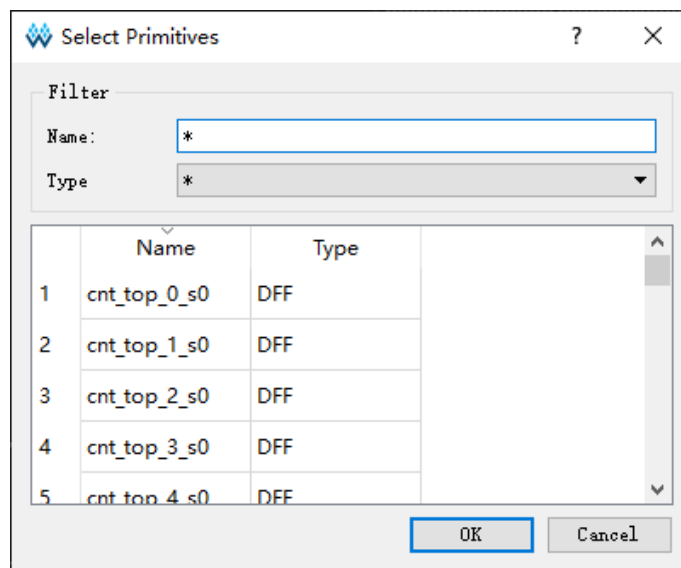
Primitives の名前またはタイプで検索し、対応する **Primitive** を選択します。

1. **"OK"** をクリックして制約情報を生成します。制約情報は、メイン画面下部の **"Primitive Constraints"** 制約編集ウィンドウに表示されます。
2. ユーザーは、編集ウィンドウで手入力またはドラッグによって位置情報を設定できます。

注記：

制約された位置は、**Chip Array** ウィンドウでライトブルーのハイライトで表示されます。



図 3-8 プリミティブ検索ダイアログボックス



Group Constraints

Group Constraints には、New Primitive Group および New Relative Group が含まれます。

Primitive Group を作成します。

1. Group Constraints ウィンドウで右クリックして New Primitive Group を選択すると、図 3-9 に示すダイアログボックスがポップアップします。
2. ユーザーは、Group の名前、含まれる Primitive、位置情報、及び Group の Exclusive 情報を設定できます。""と""の 2 つのボタンで Primitive を追加及び削除できます。図 3-10 は、正しく作成された Primitive Group です。

注記：

- Group の名前、含まれる Primitive、Group の位置は入力必須項目です。
 - 以下の方法で、Group の位置情報を入力できます。
 - 手入力します。
 - Group 制約の作成前に、"Chip Array"ウィンドウで位置をコピーし、"New Primitive Group>Locations"にペーストします。
3. Primitive Group を作成した後、"OK"をクリックすると、Group の位置情報が構文チェックされます。
 - 位置情報が不適切または無効の場合、図 3-11 と図 3-12 のプロンプトがポップアップし、ユーザーは位置情報を変更する必要があります。
 - エラーがない場合、"OK"をクリックすると、Chip Array に使用可能な位置が表示されます。
 4. 新しく生成されたグループ制約は、メイン画面の下部にある"Group Constraints"制約編集ウィンドウに表示されます。"Group Constraints"制約編集ウィンドウで、Primitive Group 制約をダブルクリックすると、図 3-10 に示すダイアログボックスが開き、再度編集できます。

図 3-9 プリミティブグループの新規作成

New Primitive Group

Group Name:

Members

Name	Type
------	------

☐ Exclusive

Locations

☐ Exclusive

図 3-10 正しいプリミティブグループ

New Primitive Group

Group Name:

Members

Name	Type
cout_cZ	LUT4
out_Z[1]	DFFE

☐ Exclusive

Locations

☐ Exclusive

図 3-11 無効な位置

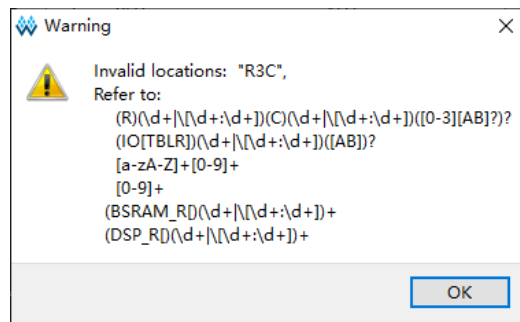
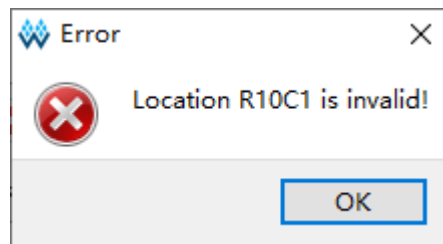




図 3-12 無効な位置



Relative Group を作成します。

1. "New Relative Group"を選択すると、図 3-13 に示すダイアログボックスがポップアップします。
2. ユーザーは、Group の名前、含まれる Primitive、位置情報、及び各

Primitive の相対位置情報を設定できます。"  "と"  "で Primitive を追加及び削除できます。図 3-14 は、作成された Relative Group 制約です。

注記：

- Group の名前、含まれる Primitive、及びその Relative Location は入力必須項目です。
 - 以下の方法で、Group の位置情報を入力できます。
 - － 手入力します。
 - － Group 制約の作成前に、"Chip Array"ウィンドウで位置をコピーし、"New Relative Group>Relative Location"にペーストします。
3. 構成が完了したら、“OK”をクリックして制約情報を生成します。
 4. 生成される制約情報は、メイン画面下部の"Group Constraints"制約編集ウィンドウに表示されます。編集ウィンドウで制約をダブルクリックすると、図 3-14 に示すダイアログボックスが開き、再度編集できます。

図 3-13 相対位置グループの作成

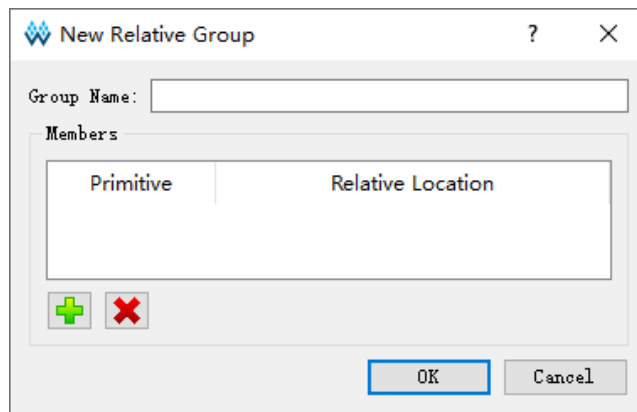
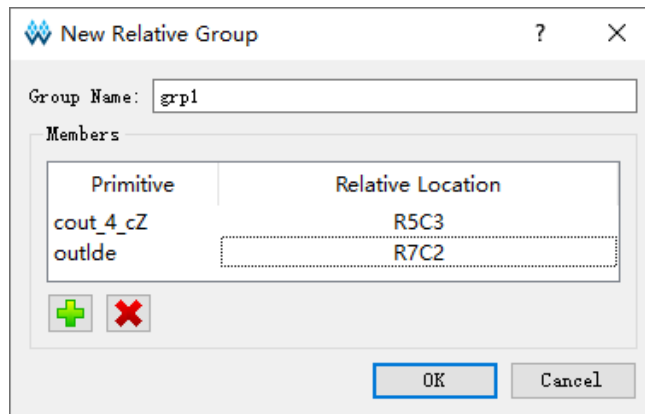


図 3-14 正しい相対位置グループ



Resource Reservation

1. Resource Reservation 制約を作成します。メイン画面下部の"Resource Reservation"制約編集ウィンドウで右クリックして **Reserve Resources** を選択し、制約を新規作成します。
2. 手入力またはドラッグによって位置情報を設定できます。
3. "Attribute"列をダブルクリックし、そのドロップダウン・リストから、リザーブ位置の属性を設定できます(図 3-15)。

注記：

Name 属性は、リザーブ制約を区別するために使用され、変更できません。

図 3-15 リザーブ制約

	Name	Locations	Attribute
1	reserve_0	drag or type t...	ALL
			ALL
			LUT
			REG

Clock Net Constraints

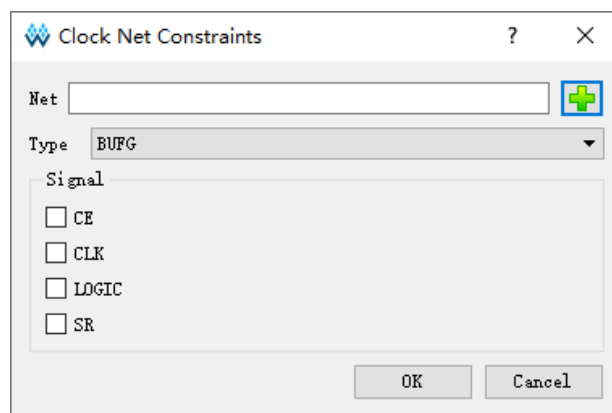
グローバルクロック割り当て制約を作成します。この制約の数に制限があるため、制約の有効性のチェックが行われます。"Clock Net Constraints"を選択すると、図 3-16 に示すダイアログボックスが表示されます。

1. "+" ボタンをクリックし、Net を選択します。
2. "Type"のドロップダウン・リストから"BUFG"、"BUFG[0]~[15]"、LOCAL_CLOCK"を選択します。
3. "CE"、"CLK"などのチェックボックスで Signal を構成します。構成完了後、"OK"をクリックすると、制約情報がメイン画面下部の"Clock Net Constraints"制約編集ウィンドウに表示されます。編集ウィンドウでダブルクリックすると、制約情報は再度編集できるようになります。

注記：

Type として LOCAL_CLOCK が選択されている場合、Signal チェックボックスはグレーアウトします。

図 3-16 クロック制約



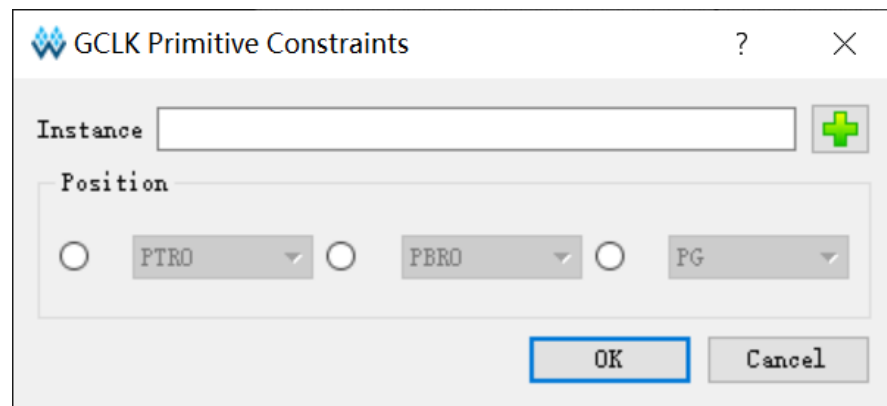
GCLK Primitive Constraints

グローバルクロック・プリミティブ制約の作成に使用されます。デバイスのグローバルクロックの分布に基づき、指定される Instance を特定のグローバルクロックに制約します。例えば、GW5AT-138 デバイスの場合、メイン画面下部の"GCLK Primitive Constraints"制約編集ウィンドウで右クリックして"Select GCLK Primitive"を選択すると、図 3-17 に示すダイアログボックスがポップアップします。以下は、その操作です。

1. "+" ボタンをクリックして対応する GCLK プリミティブを選択します。
GCLK プリミティブがデザインにない場合は、追加できません。

2. **Position** の下にあるラジオボタンと対応するドロップダウンリストを使用して、グローバルクロックの位置を構成します。
3. **"OK"**をクリックすると、制約情報が生成され、メイン画面下部の**"GCLK Primitive Constraints"**制約編集ウィンドウに表示されます。編集ウィンドウで制約をダブルクリックすると、制約情報は再度編集できるようになります。

図 3-17 グローバルクロック制約の作成



注記：

- **"Instance"**を選択すると、**"Position"**がハイライト表示されます。
- 使用可能な **Position** はデバイスとグローバルクロック・プリミティブによって異なります。

HCLK Primitive Constraints

HCLK プリミティブ制約を作成してデバイスの高速クロックの位置に制約します。例えば、GW5AT-138 デバイスの場合、メイン画面下部の**"HCLK Primitive Constraints"**制約編集ウィンドウで右クリックして **Select HCLK Primitive** を選択すると、図 3-18 に示すダイアログボックスがポップアップします。以下は、その操作です。


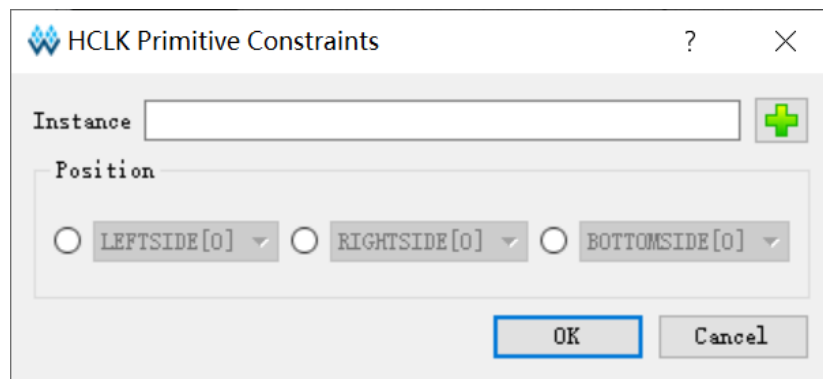
1. **"** ボタンをクリックして対応する **HCLK** プリミティブを選択します。
デザインに対応するプリミティブがない場合は、追加できません。
2. **Position** の下にあるラジオボタンと対応するドロップダウンリストを使用して、高速クロックの位置を構成します。
3. **"OK"**をクリックすると、制約情報が生成され、メイン画面下部の**"HCLK Primitive Constraints"**制約編集ウィンドウに表示されます。編集ウィンドウで制約をダブルクリックすると、制約情報は再度編集できるようになります。

図 3-18 高速クロック制約の作成



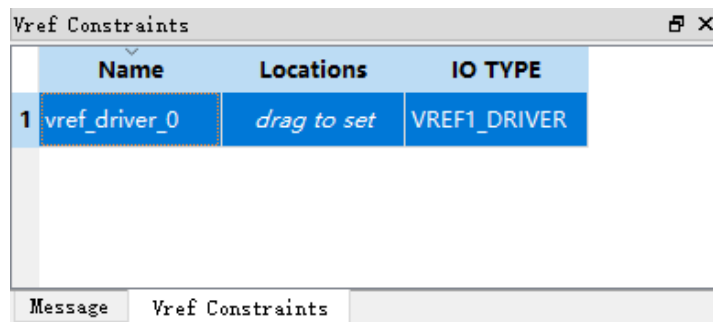
注記：

- "Instance"を選択すると、"Position"がハイライト表示されます。
- 使用可能な Position はデバイスと高速クロック・プリミティブによって異なります。

Vref Constraints

IO Port のリファレンス電圧の構成に使用される Vref Driver を作成します。メイン画面下部の"Vref Constraints"制約編集ウィンドウで右クリックして Define Vref Driver を選択し、制約を作成します(図 3-19)

図 3-19 リファレンス電圧制約



注記：

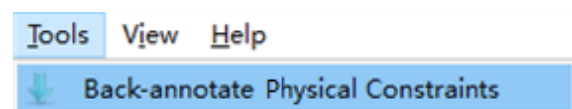
- ドラッグで Vref 制約位置を指定できます。
- Vref の名前は、ダブルクリックして変更できます。

Tools メニュー

Tools メニューを図 3-20 に示します。

Back-annotate Physical Constraints : 各プリミティブと IO Port の配置情報を物理制約ファイルにバックアノテーションします。

図 3-20 Tools メニュー



1. "Tools > Back-annotate Physical Constraints"をクリックすると、バックアノテーション対象選択ダイアログボックスが表示されます(図 3-21)。Back-Annotate Physical Constraints 機能を有効にするには、まずプロジェクトで Place&Route を正常に実行する必要があります。
2. Back-annotate Physical Constraints ダイアログボックスでは 1 つ以上のオブジェクトを選択できます。OK ボタンをクリックすると、"Save as"ダイアログボックスがポップアップし、その配置情報を物理制約ファイルに出力することができます。
3. Back-annotate Physical Constraints ダイアログボックスで Port および Port Attribute をチェックした後に生成された物理制約ファイルは図 3-22 に示すとおりです。

図 3-21 Back-annotate Physical Constraints ダイアログボックス

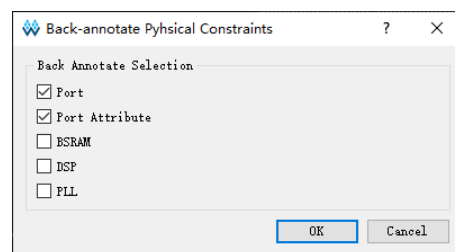
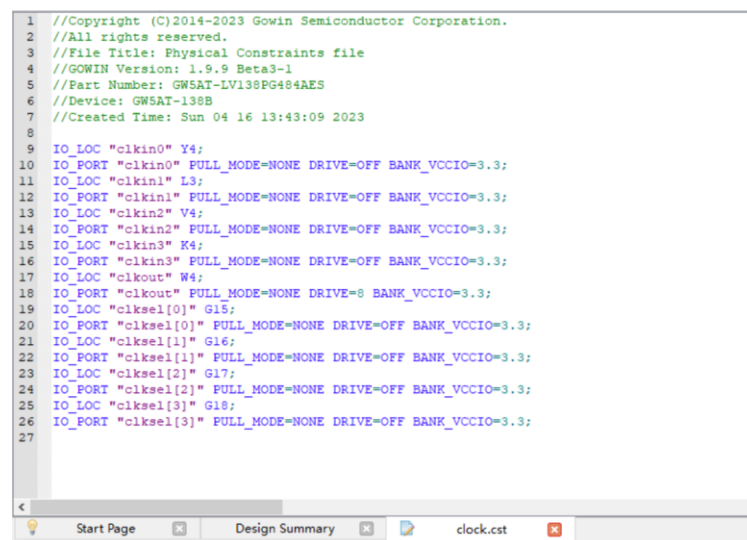


図 3-22 Port の配置情報



View メニュー

View メニューは、主にツールバー、ウィンドウの表示の設定及び Chip Array と Package View ウィンドウの拡大・縮小のなどに使用されます(図 3-23)。以下は、各サブメニューの紹介です。

- Toolbars : ツールバーのボタンの表示を制御するために使用されます。

- **Windows** : 各ウィンドウの表示を制御するために使用されます(図 3-24)。
- **Zoom In**: Chip Array ビューまたは **Package View** ビューを拡大します。
- **Zoom Out** : Chip Array ビューまたは **Package View** ビューを縮小します。
- **Zoom Fit** : ウィンドウのサイズに合わせて **Chip Array** ビューまたは **Package View** ビューを拡大または縮小します。

図 3-23 View メニュー

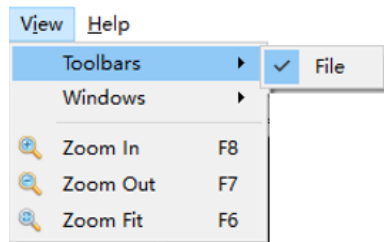
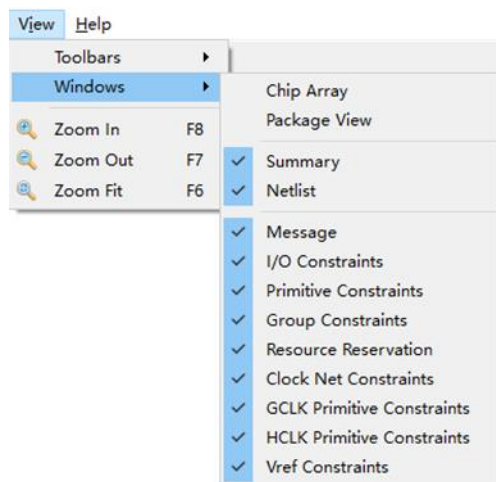


図 3-24 Windows メニュー



Help メニュー

Help メニュー > **About** をクリックすると、ソフトウェアのバージョン番号と著作権情報が表示されます。

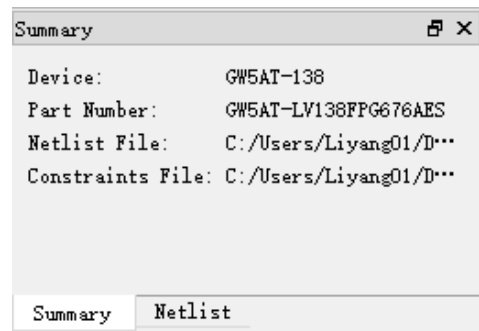
3.3.2 Summary ウィンドウと Netlist ウィンドウ

Summary ウィンドウと Netlist ウィンドウには、現在のプロジェクトの **Device**、**Part Number**、ユーザーデザインと制約ファイルのパス情報、および **Netlist** 情報などが表示されます。

Summary ウィンドウ

図 3-25 に示すように、現在のプロジェクトで使われるデバイスの情報 (Device、Part Number)、設計ファイルと制約ファイルのパスなどが表示されます。

図 3-25 Summary ウィンドウ



Netlist ウィンドウ

図 3-26 に示すように、Netlist ウィンドウは、ツリー構造でユーザーデザインの Ports、Primitives、Nets、Module、及びそれらの数の情報を表示します。

注記：

- Port、Primitive などの名前はフルパスの形式で表示され、デフォルトではアルファベットの昇順で並べ替えられます。
- Port と Net は、Bus と非 Bus の組み合わせで表示されます(図 3-27)。
- Module は階層表示されます。各 Module の後の括弧に各種 Instance の数が表示されます(図 3-28)。

図 3-26 Netlist ウィンドウ

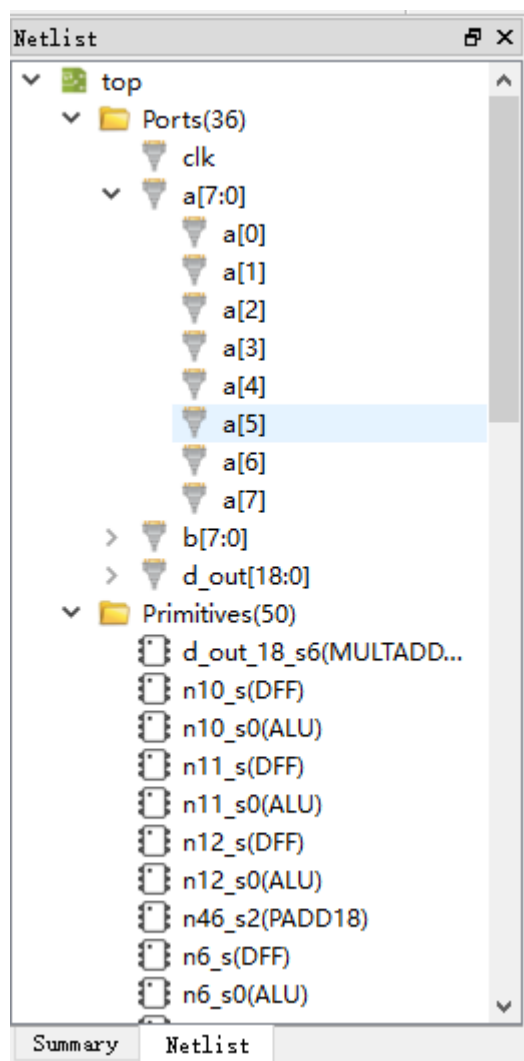


図 3-27 BUS と非 BUS を組み合わせた表示

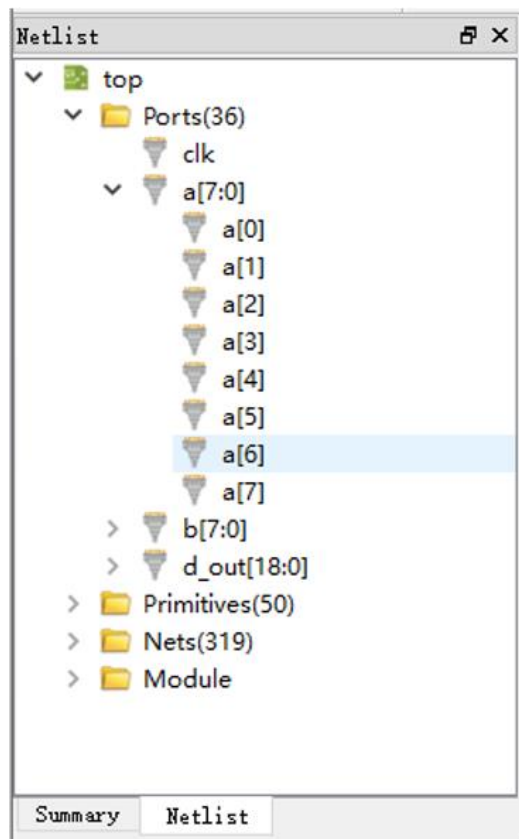
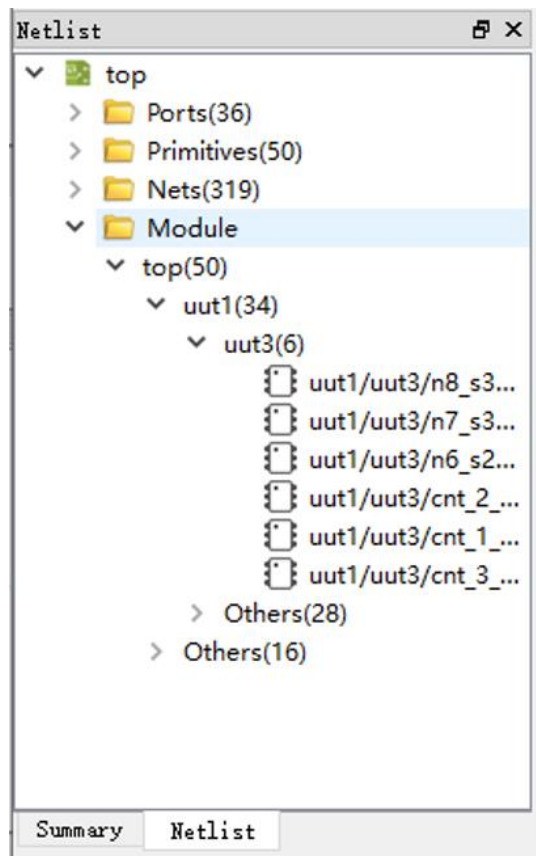


図 3-28 階層表示



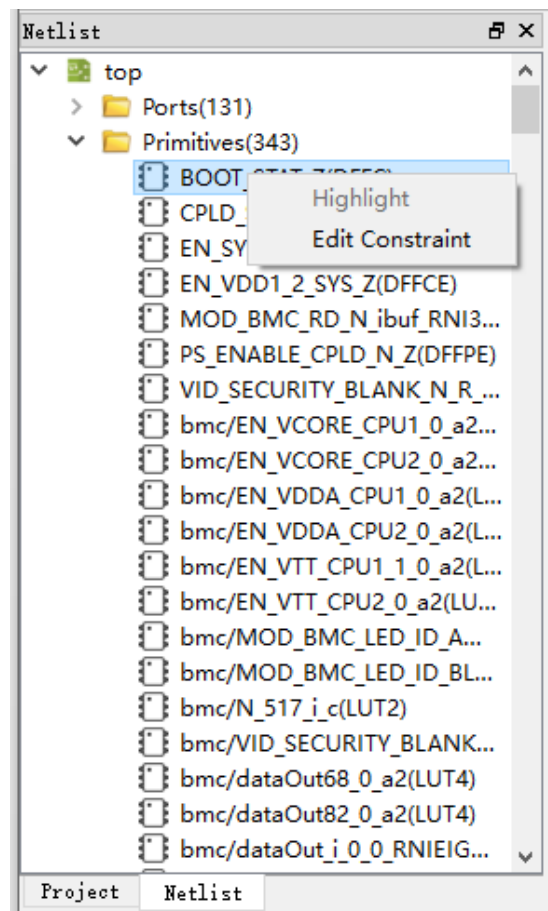
Netlist ウィンドウは右クリックメニューを提供し、その右クリックメニューには以下の機能があります。

- **Highlight** : Chip Array で対応する制約位置をハイライト表示できます。
- **Edit Constraint** : 対応する制約情報を編集します。

注記 :

現在 **Primitive** または **Port** に位置制約がない場合、ハイライト機能は使用できません (図 3-29)。

図 3-29 Netlist ウィンドウでの右クリックメニュー






3.3.3 Package View ウィンドウ

例えば、GW5AT-138-FCPBGA676A の場合、Package View は図 3-30 に示すとおりです。このウィンドウには、当該デバイスとパッケージの組み合わせの場合のユーザー I/O、電源、およびグランドピンが表示されます。カーソルを特定の位置に置くと、その位置の I/O 情報 (I/O のタイプ、Bank、LVDS 情報など) が表示されます。

図 3-30 GW5AT-138-FPBGA676A の Package view ウィンドウ



ユーザーI/O、電源、およびグラウンドピンは異なる記号と色により区別されています。IO ピンの色は BANK により異なります。上図のピンの記号の説明は次のとおりです。

- "  "はユーザーI/O を表します。
- "  "は VCCIO を表します。
- "  " は VSS を表します。

Package View は、右クリックメニューをサポートします(図 3-31)。以下は、その右クリックメニューの機能の説明です。

- Zoom In : Package View を拡大します。
- Zoom Out : Package View を縮小します。
- Zoom Fit : ウィンドウのサイズに合わせて Package View を拡大または縮小します
- Show Differential IO Pairs : 差動ペアを表示します。赤線で繋がっているのは差動ペアです(図 3-32)。

- Top View : トップビュー(デフォルト)。図 3-33 は、GW5AT-138-FCPBGA676A のトップビューで、座標の原点は左上隅にあります。
- Bottom View : ボトムビュー。図 3-34 は、GW5AT-138-FCPBGA676 のボトムビューで、座標の原点は左上隅にあります。

図 3-31 Package View の右クリックメニュー

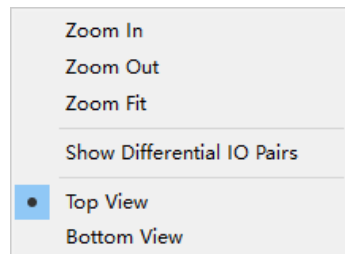


図 3-32 差動ペア表示



図 3-33 Top View

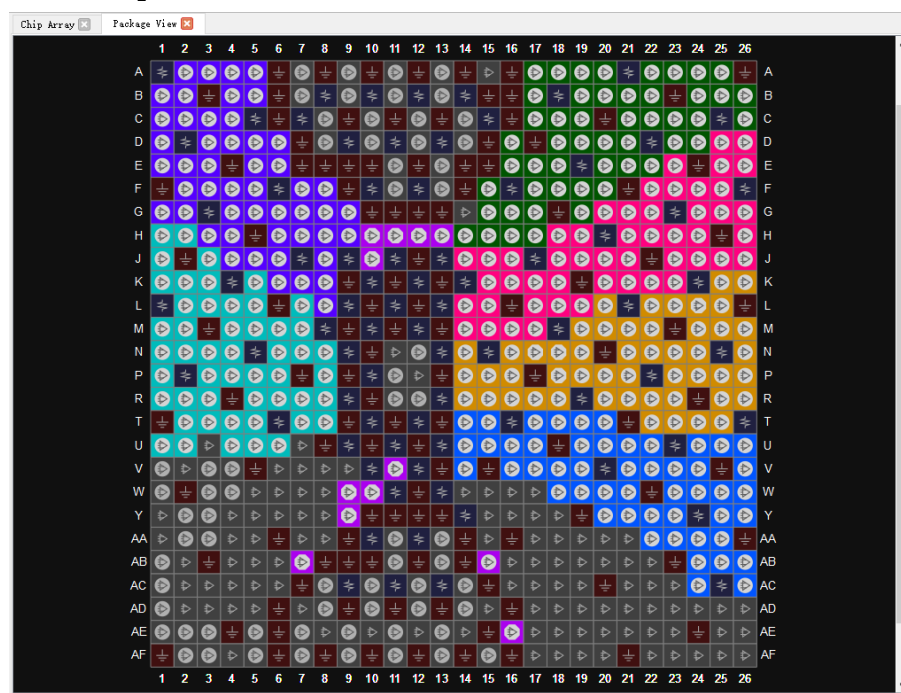
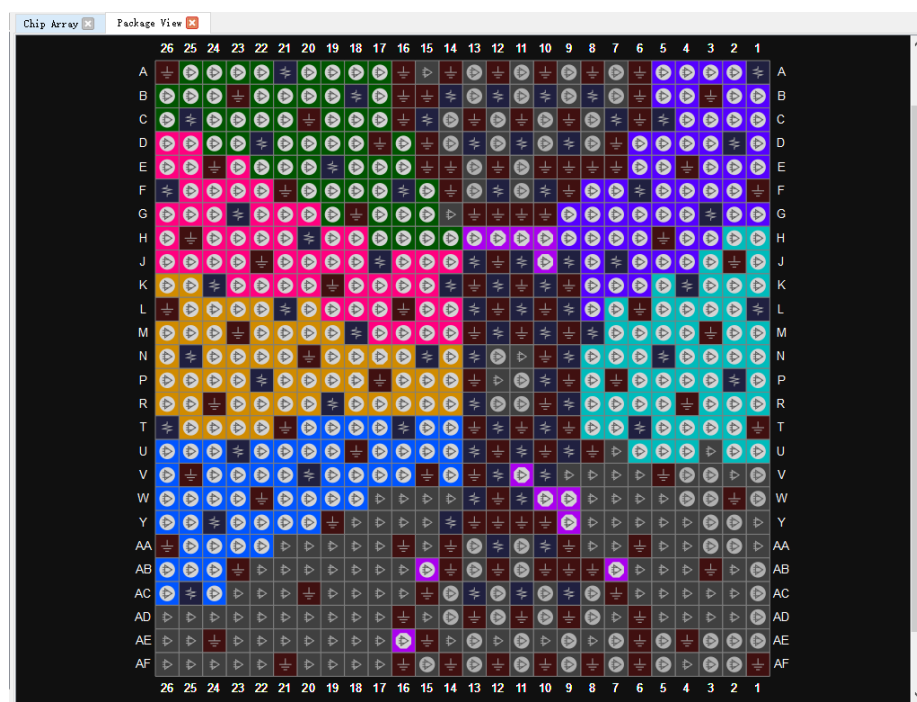


図 3-34 Bottom View



Package View は、IO Port の制約位置の表示をサポートしています。IO ポートを Netlist ウィンドウまたは下部の I/O Constraints ウィンドウから Package View ウィンドウにドラッグすることで IO Port の位置を制約することができます。ドラッグすると、ドラッグされたポートの名前はマウ

スのそばに表示され、制約できないピンはグレーアウトします。

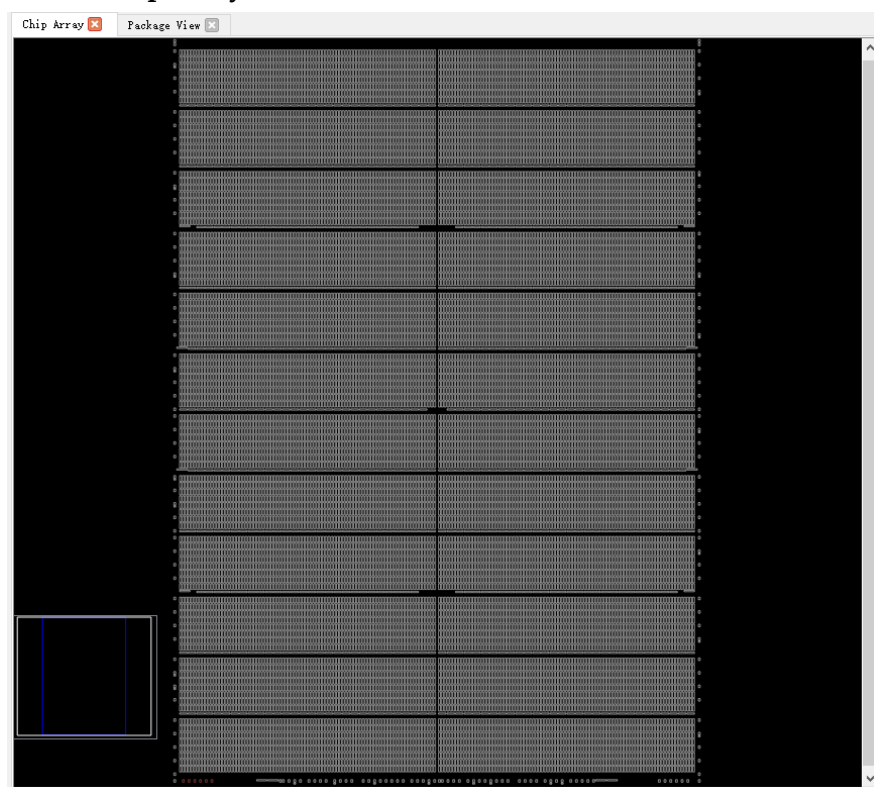
3.3.4 Chip Array ウィンドウ

図 3-35 は、FloorPlanner の Chip Array ウィンドウです。Chip Array ウィンドウはチップの行と列情報によってチップの I/O、CFU、DSP、PLL、BSRAM、および DQS などの分布を表示し、すべての制約位置のリアルタイム表示を実現すると同時に、拡大・縮小、位置コピー、マウスオーバー表示、ドラッグなどの機能をサポートします。

そのうち、I/O は色によって区別されます。

- 白色：このパッケージでボンディングされている I/O。
- 赤色：このパッケージでボンディングされていない IO。

図 3-35 Chip Array ウィンドウ



Chip Array にはグリッドモード、マクロセル・モード、プリミティブモードの 3 つの表示モードがあります。

- グリッドモード：GRID 単位で制約位置を表示します(図 3-36)。
- マクロセル・モード：CLS、BLOCK など単位で制約位置を表示します(図 3-37)。
- プリミティブモード：REG、LUT など単位で制約位置を表示します(図 3-38)。

図 3-36 グリッドモードでの制約

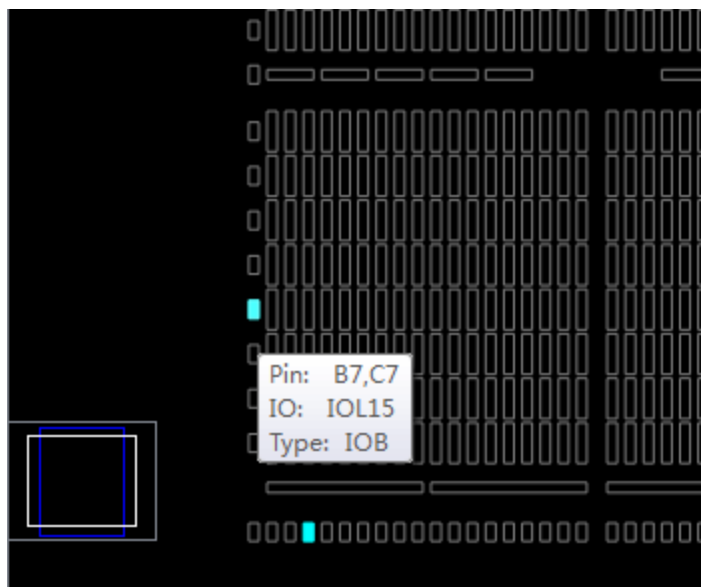


図 3-37 マクロセル・モードでの制約

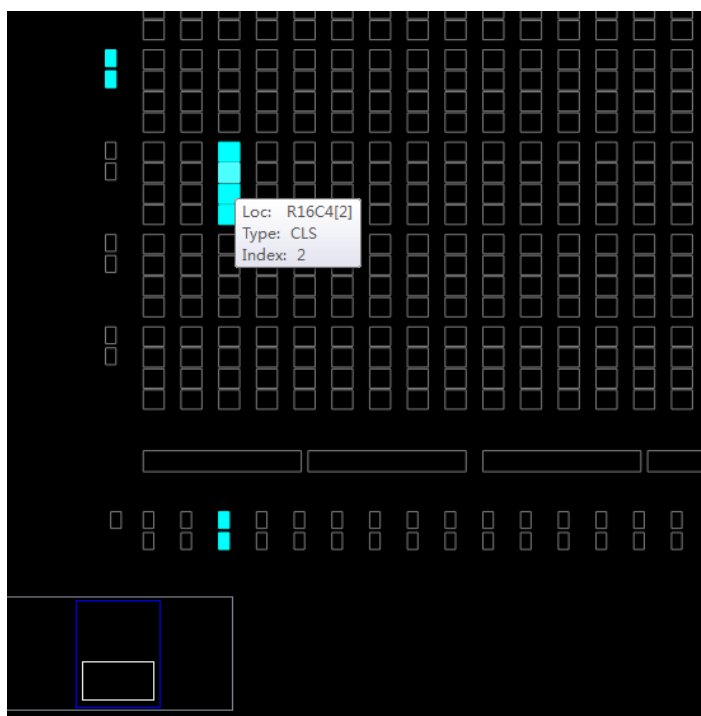
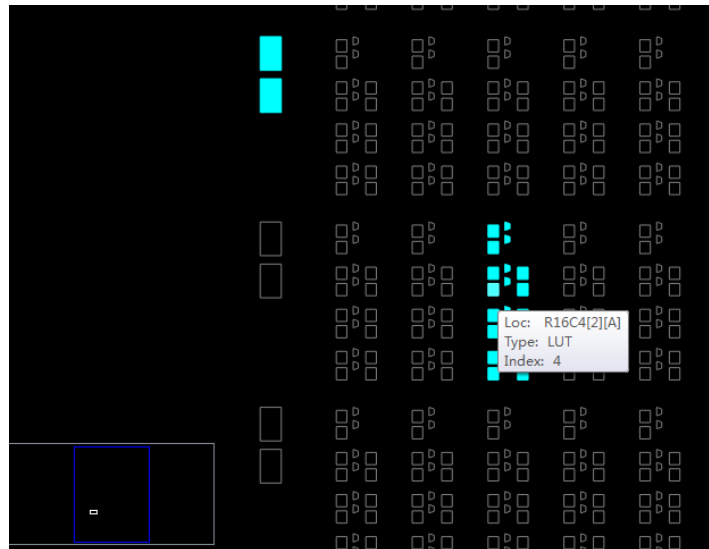


図 3-38 プリミティブモードでの制約



Chip Array は以下のドラッグ機能をサポートします。

- Netlist ウィンドウから Array ウィンドウまでドラッグ：制約の生成と位置の指定に使用されます。
- 制約編集ウィンドウから Array ウィンドウまでドラッグ：制約位置の指定に使用されます。

Chip Array ウィンドウには、デバイス全体に対する現在のウィンドウの位置をリアルタイムで表示するために使用される chip サブウィンドウがあります。サブウィンドウの白いボックスをドラッグすると、Chip Array のビューも一緒に移動します。また、Chip Array ウィンドウは、さまざまな色を使用して制約タイプを区別します。以下は、各色の意味です。

- 白色：選択状態にある制約位置、またはハイライト表示されている制約位置を表示します。
- ネイビー：リザーブ制約の位置を表示します。この位置を占有できないことを示します。
- 水色：あるグリッドまたは範囲内に制約されている I/O とプリミティブの位置を表示します。

Chip Array ウィンドウは右クリックメニューをサポートし、その機能は以下のとおりです。

- Zoom In : Chip Array ビューを拡大します。
- Zoom Out : Chip Array ビューを縮小します。
- Zoom Fit : ウィンドウのサイズに合わせて Chip Array ビューを拡大または縮小します。

- **Show Constraints View** : Chip Array の instance 制約ビューを表示します。
- **Show Place View** : Chip Array の instance 配置ビューを表示します。Place & Route が実行されて FloorPlanner が起動した場合にのみ有効です。それ以外の場合は、グレースアウトします。
- **Show Multi-View** : Chip Array の instance 制約および配置ビューを表示します。Place & Route が実行されて FloorPlanner が起動した場合にのみ有効です。それ以外の場合は、グレースアウトします。
- **Show In-Out Connection** : Place View で instance の入力および出力接続の instance 位置を表示および選択します。これは、**Show Place View > All Instance** の時にある instance が選択されている場合にのみ使用できます。それ以外の場合は、グレースアウトします。
- **Show In Connection** : Place View で instance の入力接続の instance 位置を表示および選択します。これは、**Show Place View > All Instance** の時にある instance が選択されている場合にのみ使用できます。それ以外の場合は、グレースアウトします。
- **Show Out Connection** : Place View で instance の出力接続の instance 位置を表示および選択します。これは、**Show Place View > All Instance** の時にある instance が選択されている場合にのみ使用できます。それ以外の場合は、グレースアウトします。
- **Unhighlight All** : すべてをハイライト表示解除します。
- **Copy Location** : 選択した対象の位置をコピーします。Chip Array ウィンドウで GRID、Block などが選択されている場合にのみ使用できます(図 3-39)。

Show Place View で、Lut、Reg の密度を表示できます(図 3-40)。詳細は次のとおりです。

- **ALL Instance** : すべての Instance の配置状況を表示します。5 個以下の場合には薄緑色、6-10 個の場合には緑色、11 個以上の場合には濃い緑色です。
- **Only Lut** : すべての Lut の配置状況のみを表示します。2 個以下の場合には薄緑色、3-4 個の場合には緑色、5 個以上の場合には濃い緑色です。
- **Only Dff** : すべての Reg の配置状況を表示します。2 個以下の場合には薄緑色、3-4 個の場合には緑色、4 個以上の場合には濃い緑色です。

Show Place View > ALL Instance で、デザイン内のすべての instance の配置を確認できます。

- Chip Array ウィンドウで、マウスを instance の配置位置に合わせると、その instance の名前が表示されます。

- Netlist ウィンドウで特定の instance を右クリックして Highlight を選択すると、この instance の配置位置が Chip Array ウィンドウでハイライト表示されます(図 3-42)。

注記：

"Ctrl"キー+左クリックでドラッグすることで、領域を選択できます。領域を右クリックして"Copy Location"を選択すると、その領域の位置情報をコピーできます。コピーされた位置は、任意の制約編集ウィンドウにペーストできます。

図 3-39 Chip Array の右クリックメニュー

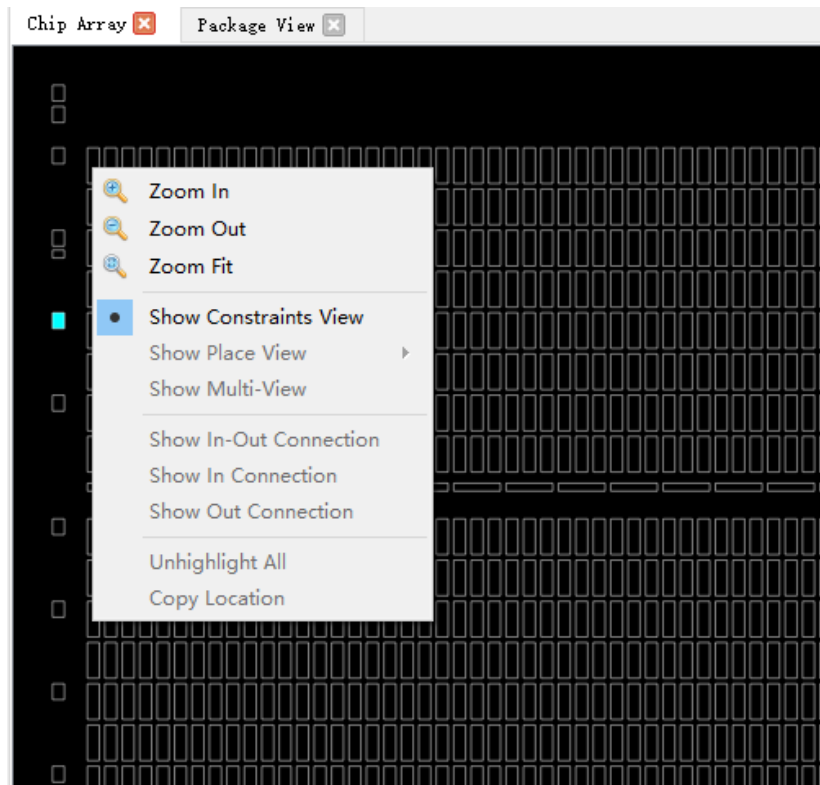


図 3-40 Show Place View の表示

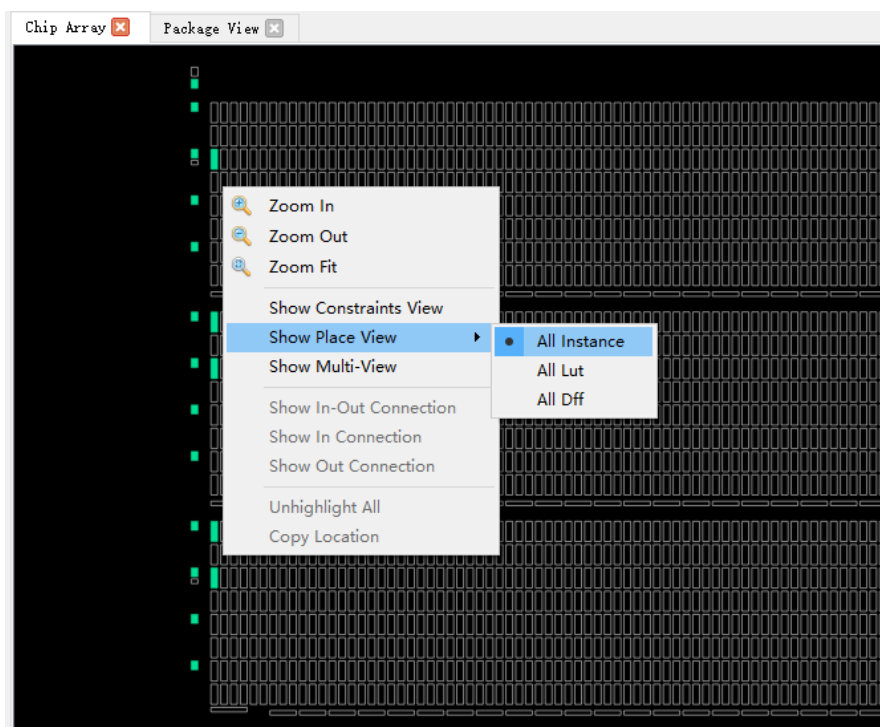


図 3-41 マウスオーバー表示

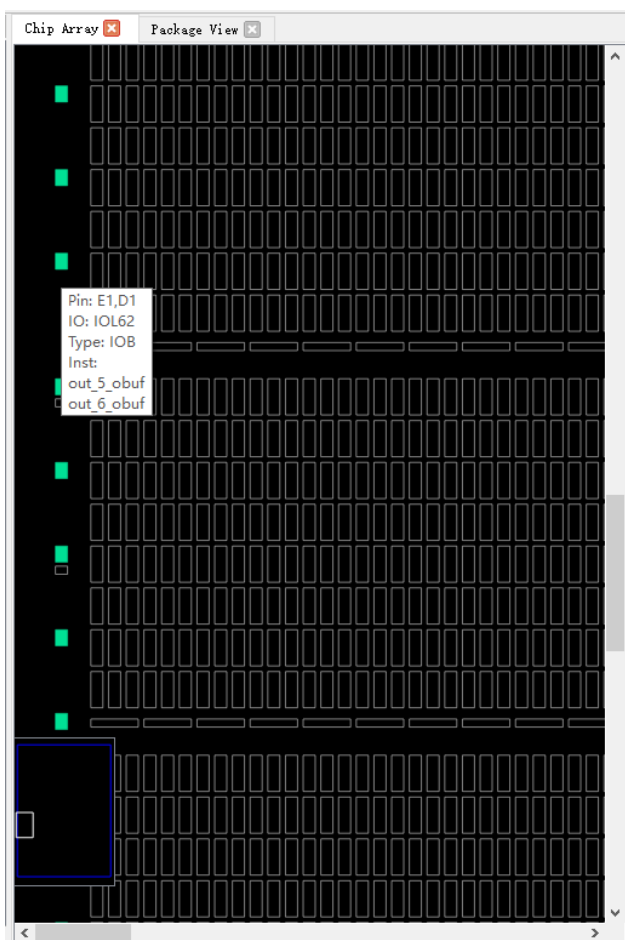
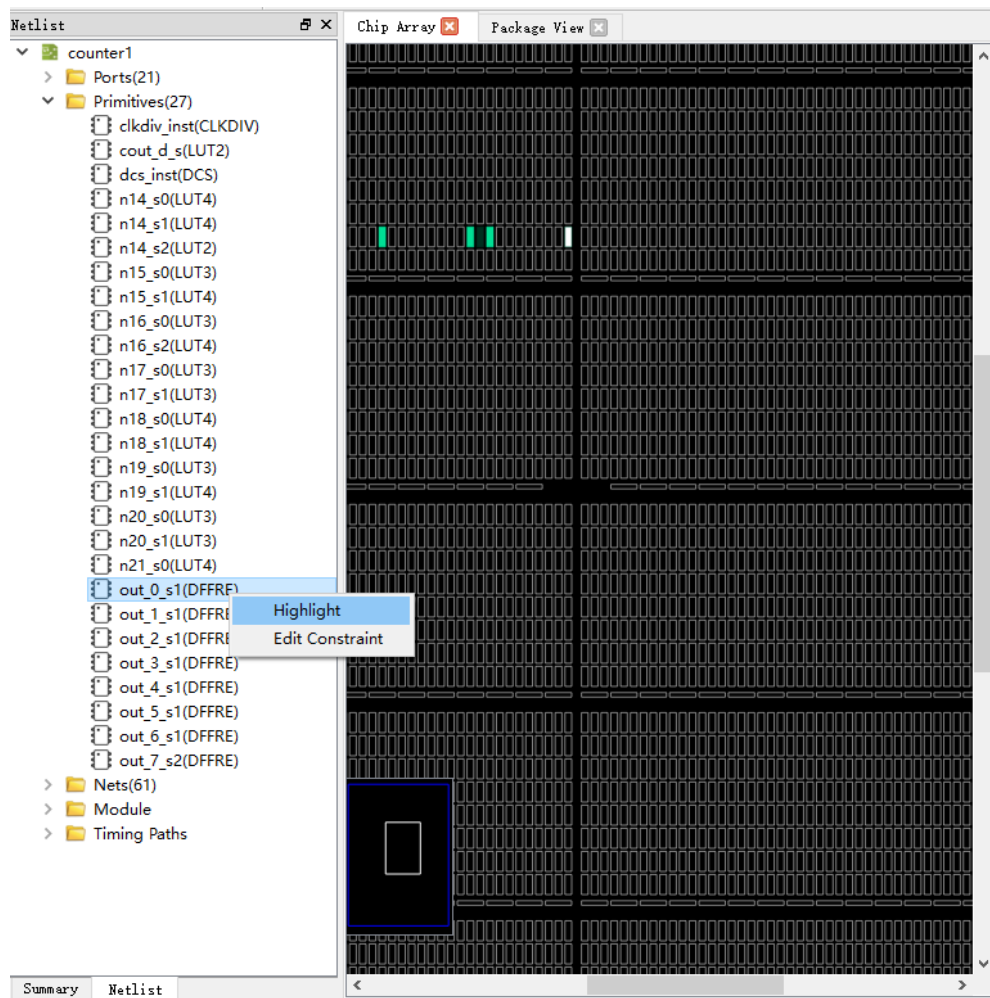


図 3-42 右クリックメニューによるハイライト表示



3.3.5 Constraint 編集ウィンドウ

Constraint 編集ウィンドウは、"I/O Constraints"、"Primitive Constraints"、"Group Constraints"など 8 つの編集ウィンドウで構成され、制約編集機能と位置ドラッグ機能を提供します。以下は、各ウィンドウの紹介です。

I/O Constraints

I/O Constraints はデザインの port を制約します。I/O 制約ウィンドウは図 3-43 に示すとおりで、その各機能は以下のとおりです。

- Port 的 Direction、Bank、IO Type、Pull Mode など、ユーザーデザインのすべての IO Port の属性及び制約情報を表示します。
- 制約位置、属性などの編集機能を提供します。
- ドラッグするか、ダブルクリックして入力することで制約情報を変更できます。

注記：

- I/O の位置は、ドラッグで設定するか、ダブルクリックして入力できます。

- IO をドラッグする時、ドラッグされる IO の名前が表示されます。
- IO を Chip Array ウィンドウにドラッグすると、配置できる位置が明るくなり、配置できない位置の明るさは変わりません。
- IO を Chip Array ウィンドウにドラッグする時、配置できる位置の明るさは変わらず、配置できない位置の明るさは暗くなります。
- 設定完了後、Chip Array ウィンドウの制約の位置は水色でハイライト表示され、Package View ウィンドウの制約の位置はオレンジ色でハイライト表示されます。

ウィンドウは右クリックメニューを提供します。詳細は次のとおりです。

- Unplace : 配置をキャンセルします。
- Reset Properities : Port の属性の設定をリセットします。
- Highlight : 制約位置をハイライト表示します。
- IO Type : レベル規格を設定します。
- Drive : ドライブ強度を設定します。
- Pull Mode : プルモードを設定します。
- PCI Clamp : PCI プロトコルのオン/オフを設定します。
- Hysteresis : ヒステリシスを設定します。
- Open Drain : オープンドレイン回路のオン/オフを設定します。
- Vref : 外部リファレンス電圧を設定します。
- Single Resistor : シングルエンド抵抗のオン/オフを設定します。
- Diff Resistor : 差動抵抗のオン/オフを設定します。
- Bank Vccio : BANK 電圧を設定します。

注記 :

右クリックメニューで、ユーザーはバッチで Port 属性を変更できます。ユーザーは複数 Port を選択できます。複数の Port に同じ構成可能な属性値がある場合、右クリックメニューで一括構成できます。詳しくは、『GW5AT シリーズ FPGA 製品データシート(DS981)』を参照してください。

図 3-43 I/O 制約ウィンドウ

I/O Constraints										
	Port	Direction	Diff Pair	Location	Bank	Exclusive	IO Type	Drive	Pull Mode	PCI Clamp
2	cin	input				False	LVC MOS18	N/A	UP	N/A
3	clk	input				False	LVC MOS18	N/A	UP	N/A
4	clko	output				False	LVC MOS18	8	UP	N/A
5	cout	output				False	LVC MOS18	8	UP	N/A
6	data[0]	input				False	LVC MOS18	N/A	UP	N/A
7	data[1]	input				False	LVC MOS18	N/A	UP	N/A

Primitive Constraints

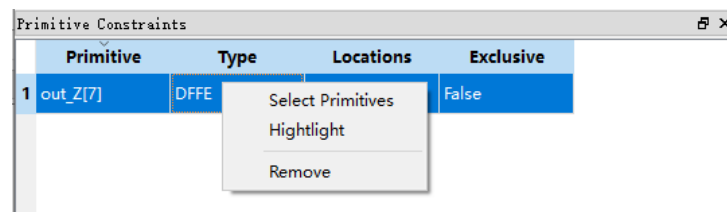
Primitive Constraints により、プリミティブの位置を制約できます。図 3-44 にプリミティブ制約ウィンドウを示します。

- 現在のすべての **Primitive** 制約の名前、タイプ、位置、および **Exclusive** 情報を表示します。
- このウィンドウは、制約位置のハイライト表示、制約の削除と追加をサポートする右クリックメニューを提供します。

注記：

- ドラッグするか、ダブルクリックして入力することで位置情報を変更できます。
- ダブルクリックして **Exclusive** 属性を設定することができます。
- **Primitive** 制約位置を手入力する時、位置の構文と有効性がチェックされます。図 3-11 と図 3-12 はそのエラーメッセージです。

図 3-44 プリミティブ制約ウィンドウ

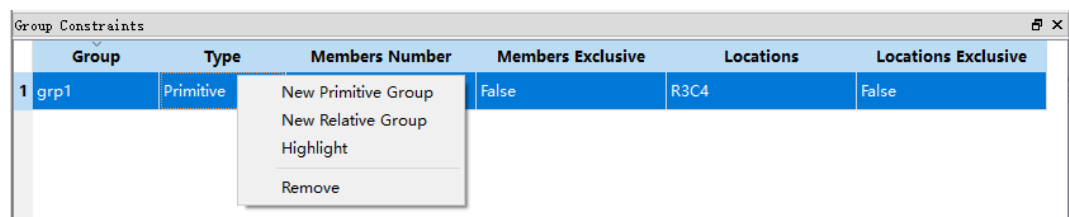


Group Constraints

Group Constraints により、デザインの I/O および一部のプリミティブに対してグループ制約を実行できます。グループ制約ウィンドウを図 3-45 に示します。その機能は次のとおりです。

- **Primitive** と **Relative** の 2 つの **Group** があります。現在のすべての **Group** 制約の名前、タイプ、含まれる **Primitive** の数、位置、及び **Exclusive** 情報を表示します。**Group** をダブルクリックすると、図 3-10 または図 3-14 に示すダイアログボックス表示され、制約情報を編集および変更できます。
- このウィンドウは、制約位置のハイライト表示、制約の削除と追加をサポートする右クリックメニューを提供します。

図 3-45 グループ制約ウィンドウ



Resource Reservation

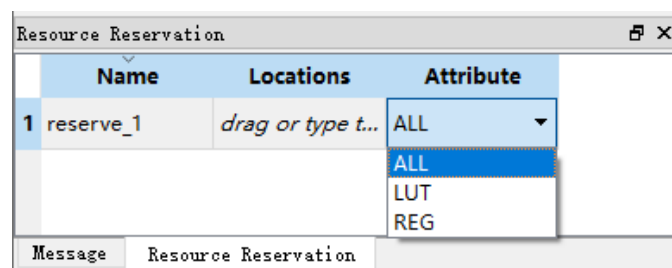
Resource Reservation は現在のパッケージにおける利用可能なリソースをリザーブ制約します。リザーブ制約ウィンドウを図 3-46 に示します。その機能は次のとおりです。

- 現在のすべてのリザーブ制約の位置情報を表示します。
- このウィンドウは、制約位置のハイライト表示、制約の削除と追加をサポートする右クリックメニューを提供します。
- **Name** 属性は、各リザーブ制約を区別するために使用され、ユーザーはそれを変更できません。

注記：

ドラッグするか、ダブルクリックして入力することで位置情報を変更できます。

図 3-46 リザーブ制約ウィンドウ



Clock Net Constraints

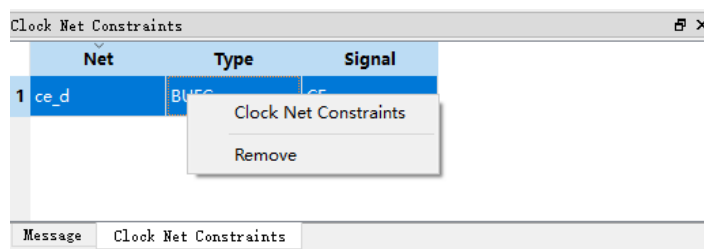
Clock Net Constraints により、**net** に対してグローバルクロック割り当て制約を実行できます。クロック割り当て制約ウィンドウを図 3-47 に示します。

- 現在のすべてのクロック割り当て制約に関する情報を表示します。
- このウィンドウは、クロック割り当て制約の追加と削除をサポートする右クリックメニューを提供します。

注記：

- 制約をダブルクリックして編集することができます。
- **CLOCK Net** に位置情報がないため、ドラッグ機能はサポートされません。
- 図 3-16 はグローバルクロック制約の新規作成のウィンドウです。

図 3-47 クロック制約ウィンドウ



GCLK Primitive Constraints

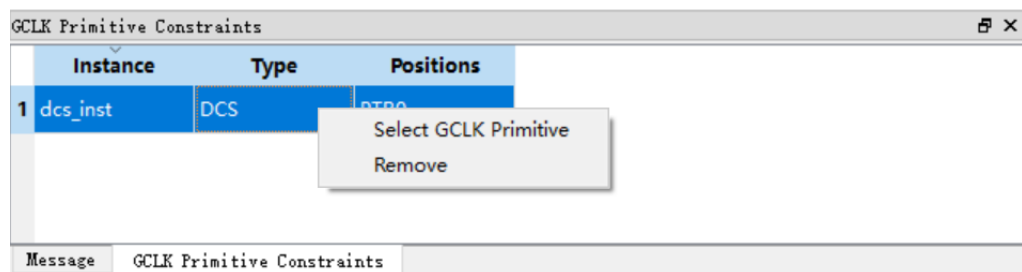
GCLK Primitive Constraints により、グローバルクロック・プリミティブ制約を行うことができます。グローバルクロック制約ウィンドウを図 3-48 に示します。

- Instance の名前、タイプ、グローバルクロックの位置を含むすべてのグローバルクロック制約を表示します。
- ウィンドウは、グローバルクロック制約の追加と削除をサポートする右クリックメニューを提供します。

注記：

図 3-17 はグローバルクロック制約の新規作成のウィンドウです。

図 3-48 グローバルクロック制約ウィンドウ



HCLK Primitive Constraints

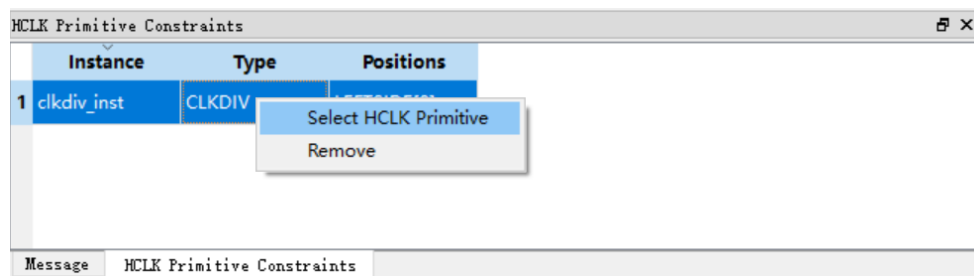
HCLK Primitive Constraints により、高速クロック・プリミティブに対して高速クロック制約を行うことができます。高速クロック制約ウィンドウを図 3-49 に示します。

- 高速クロック関連 Instance の位置制約を表示します。これには Instance の名前、タイプ、高速クロックの位置が含まれます。
- ウィンドウは、高速クロック制約の追加と削除をサポートする右クリックメニューを提供します。

注記：

図 3-18 はグローバルクロック制約の新規作成のウィンドウです。

図 3-49 高速クロック制約ウィンドウ



Vref Constraints

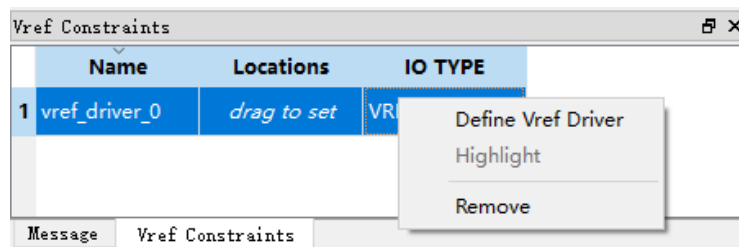
Vref Constrains により、Bank の外部リファレンス電圧を制約できます。Vref 制約ウィンドウを図 3-50 に示します。その機能は次のとおりです。

- ユーザーカスタマイズの Vref Driver 情報の表示に使用され、ユーザーは Vref の名前、位置情報をカスタマイズできます。
- このウィンドウは、制約の位置のハイライト表示、制約情報の追加と削除をサポートする右クリックメニューを提供します。

注記：

位置情報は、ドラッグでのみ設定できます。

図 3-50 Vref 制約ウィンドウ



3.3.6 Message ウィンドウ

Message ウィンドウは、出力結果を表示します(図 3-51)。

図 3-51 Message ウィンドウ



4 FloorPlanner の使用

FloorPlanner は、制約を作成および編集し、配置配線で使用する物理制約ファイルを生成できます。

4.1 制約ファイルの新規作成

FloorPlanner は、新規作成した物理制約ファイルと変更した物理制約ファイルを出力できます。以下は、その操作手順です。

1. 3.2 起動の説明に従い、FloorPlanner を起動します。
2. "File>New"をクリックして"New"ダイアログを開きます(図 4-1)。

注記：

または、以下の 2 つの方法で"New"ダイアログボックスを開きます。

- ショートカットキーCtrl+N を使用します。
- ツールバーの"New"アイコンをクリックします。

3. プロジェクトのネットリストファイルとデバイスを選択し、"OK"をクリックします。

図 4-1 制約ファイルの新規作成

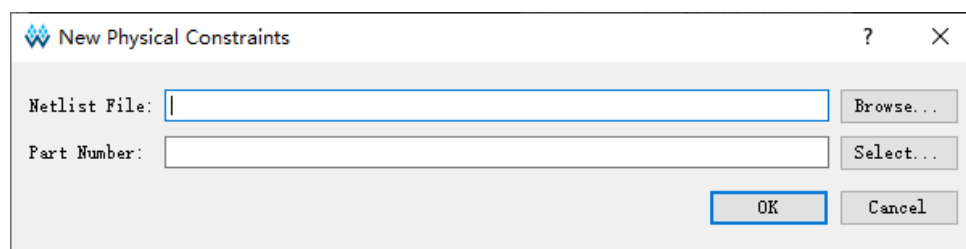


図 4-2 デバイスの選択

The "Select Device" dialog box contains a filter section with the following settings:

- Series: GW5AT
- Package: Any
- Device: Any
- Speed: Any
- Device Version: Any

Below the filter is a table with the following data:

Part Number	Device	Device Version	Package	Speed	Voltage	IO	LUT	
GW5AT-LV138PG484AES	GW5AT-138	B	PBGA484A	ES	LV	297	138240	1
GW5AT-LV138GW391AES	GW5AT-138	B	GW391A	ES	LV	324	138240	1
GW5AT-LV138GW391ES	GW5AT-138	B	GW391	ES	LV	324	138240	1
GW5AT-LV138FPG676AES	GW5AT-138	B	FCPBGA676A	ES	LV	312	138240	1
GW5AT-LV138GW391ES	GW5AT-138		GW391	ES	LV	324	138240	1
GW5AT-LV138FPG676AES	GW5AT-138		FCPBGA676A	ES	LV	312	138240	1

At the bottom right of the dialog are "OK" and "Cancel" buttons.

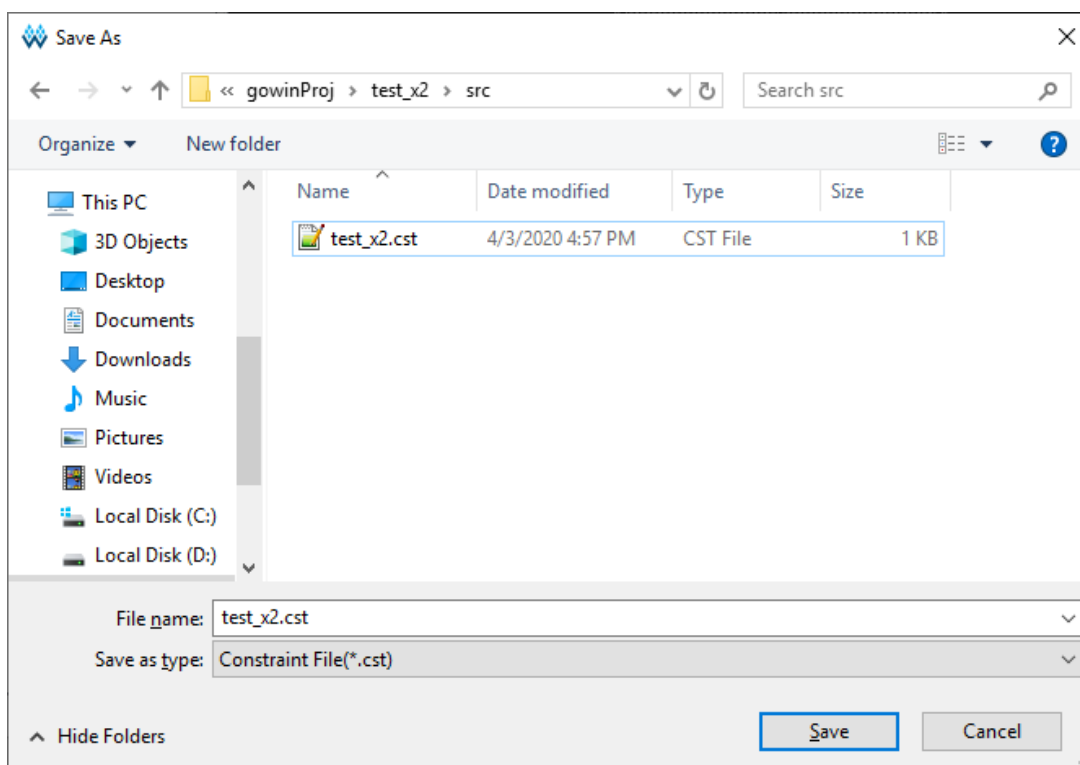
注記：

- **Select** …ボタンは、デバイスおよびパッケージの選択に使用されます。GOWIN セミコンダクターのすべての **FPGA** デバイスがサポートされます(図 4-2)。
- 3.2 起動の 3 つ目の方法を使用して **FloorPlanner** を起動します。

物理制約を新規作成した後、**FloorPlanner** のメイン画面で以下のような操作が可能です。

1. ドラッグなどによってピン位置を割り当てます。
2. ツールバーの**"Save"**アイコンをクリックして制約ファイルを出力できます。
3. ポップアップした**"Save"**ダイアログボックスで、ファイル名を変更できます(図 4-3)。

図 4-3 出力ファイルの保存



4.2 制約ファイルの編集

FloorPlanner は、I/O 制約、プリミティブ制約、グループ制約、リソースリザーブ制約、グローバルクロック割り当て制約、リファレンス電圧制約などの作成をサポートします。**Constraints** メニューから **Constraints** を編集および作成できます。詳細は [3.3.1 メニューバー](#) を参照してください。

注記：

制約は他の方法でも作成できます。このセクションでは主にドラッグアンドドロップおよび編集で制約を生成する方法を紹介します。

4.2.1 制約編集の例

例としてユーザーデザイン **counter.v** を使用して、制約の作成方法を示します。

```
module counter1(out, cout, data, load, cin, clk, clko);  
output [7:0] out;  
output cout;  
output clko;  
input [7:0] data;
```

```
input load, cin, clk;
reg [7:0] out;
always @(posedge clk)
begin
    if (load)
        out = data;
    else
        out = out + cin;
end
assign cout = &out & cin;
wire clkout;
CLKDIV clkdiv_inst (
    .CLKOUT(clkout),
    .HCLKIN(clk),
    .RESETN(1'b1),
    .CALIB(1'b0)
);
defparam clkdiv_inst.DIV_MODE = "2";

DCS dcs_inst (
    .CLKOUT(clko),
    .CLKSEL(4'b0000),
    .CLKIN0(clkout),
    .CLKIN1(clkout),
    .CLKIN2(clkout),
    .CLKIN3(clkout),
    .SELFORCE(1'b0)
);
```

```
defparam dcs_inst.DCS_MODE = "RISING";

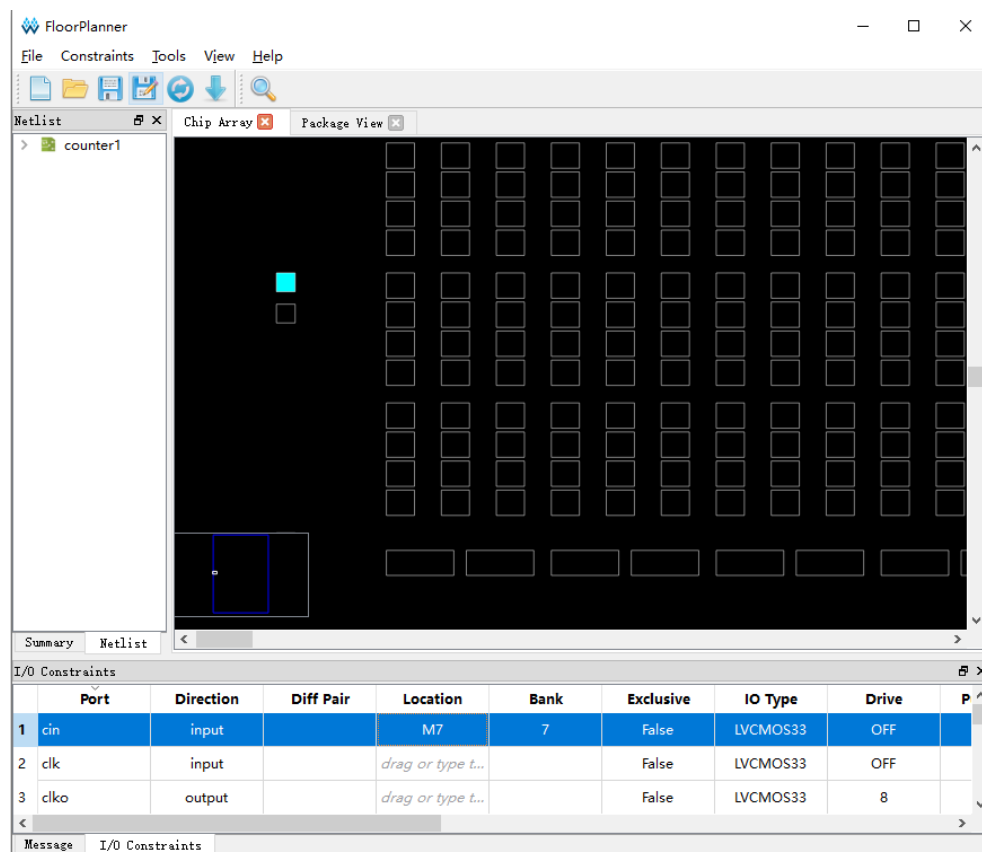
endmodule
```

4.2.2 I/O 制約の編集

Chip Array にドラッグして I/O 制約を作成します。その手順は次のとおりです。

1. Chip Array ウィンドウをマクロセル・モードに拡大します。
2. 図 4-4 に示すように、Port "cin"を Chip Array ウィンドウの"M7"の位置にドラッグします。
3. Port "cin"の位置情報は M7 になります。

図 4-4 Chip Array にドラッグして I/O Constraints を作成



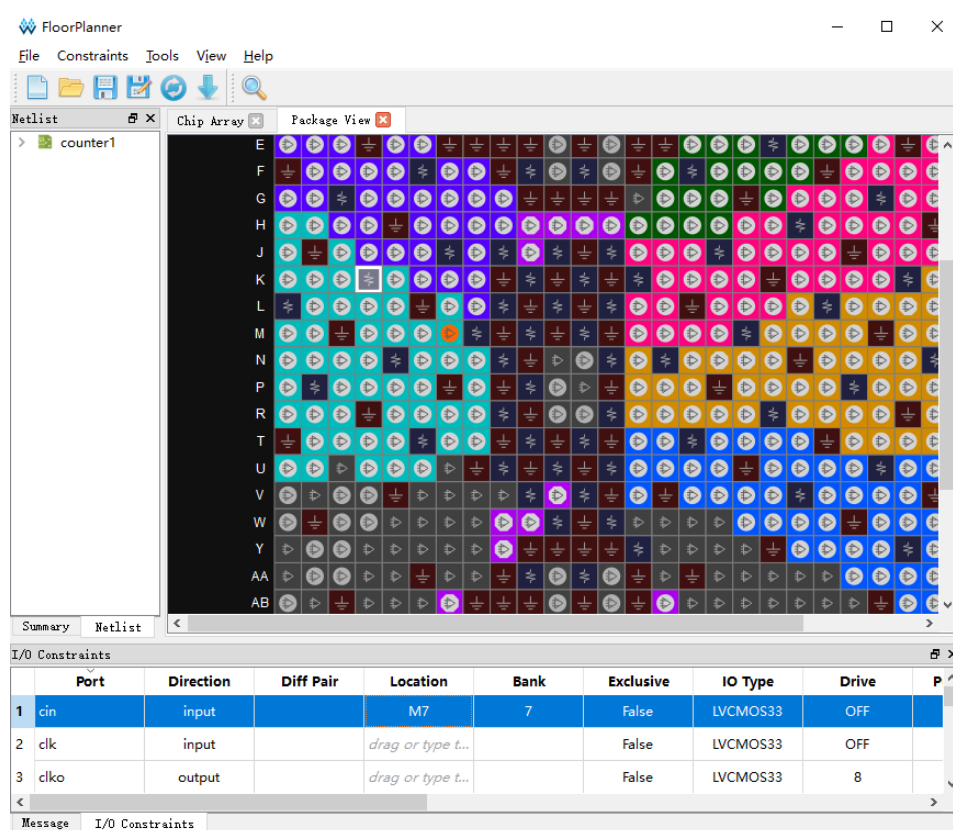
Package View にドラッグして I/O Constraints を作成します。手順は次のとおりです。

1. IO Constraints 編集ウィンドウをクリックします。
2. 図 4-5 に示すように、Port "cin"を Package View ウィンドウの M7 の位

置にドラッグします。

3. Port "cin"の位置情報は M7 になります。

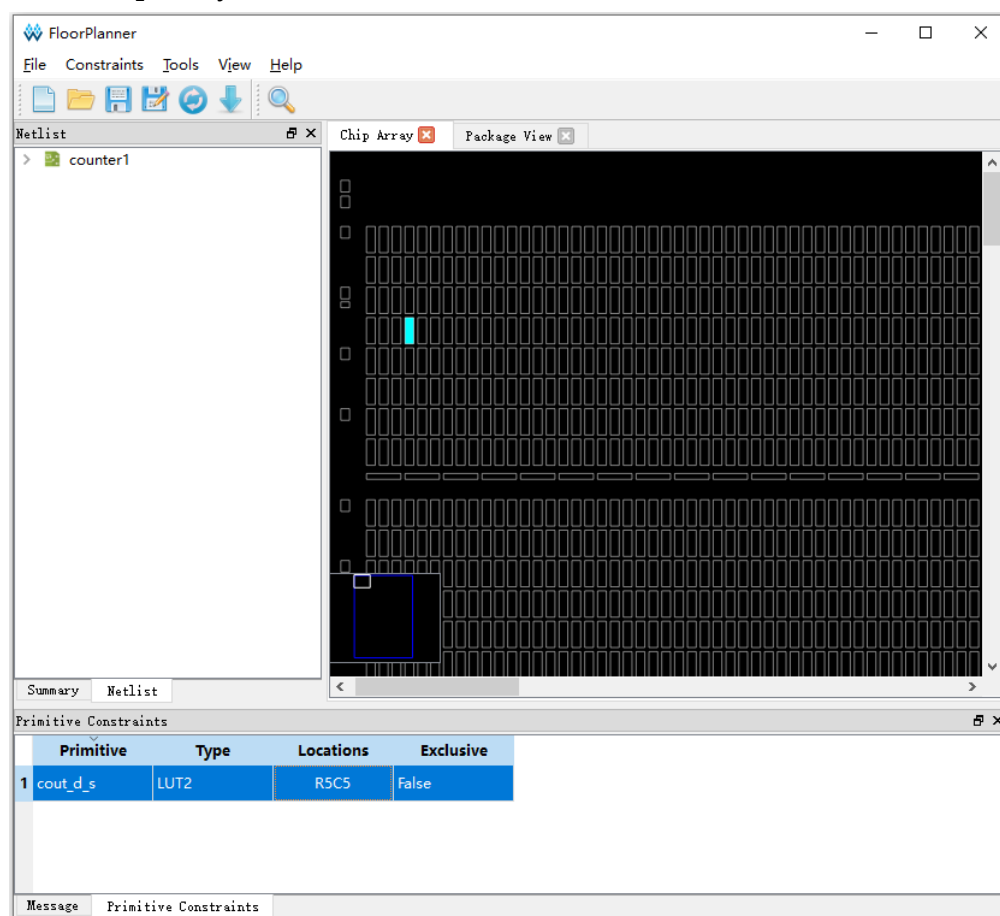
図 4-5 Package View にドラッグして I/O Constraints を作成



4.2.3 図プリミティブ制約の編集

1. Primitive Constraints 編集ウィンドウで、右クリックして “Select Primitives” を選択します。Primitive Finder ダイアログボックスが表示されたら、Primitive “cout_d_s” を選択して “OK” をクリックします。
2. 作成されたプリミティブ制約を Chip Array ウィンドウの "R5C5"位置にドラッグします(図 4-6)。
3. Primitive "cout_d_s"の位置情報は R5C5 になります。

図 4-6 Chip Array にドラッグして Primitive Constraints を作成



4.2.4 グループ制約の編集

図 4-7 に示すように、Group Constraints ウィンドウで右クリックし、Primitive Group 制約と Relative Group 制約を作成します。

図 4-7 Group Constraints ウィンドウの右クリックメニュー



プリミティブグループ制約の編集

1. Group Constraints 編集ウィンドウで右クリックして "New Primitive Group" を選択すると、"New Primitive Group" ダイアログボックスがポップアップします。
2. Group Name "grp1" を入力し、"⊕" をクリックすると、Primitive Finder ダイアログボックスがポップアップします。

3. 設定したい Primitive “n17_s0”、“cout_d_s” を選択して “OK” をクリックし、Members リストに追加します。
4. Locations で制約位置“R5C10”を入力します(図 4-8)。
5. "OK"をクリックして Primitive Group Constraints を作成します(図 4-9)。

図 4-8 Primitive Group Constraints の作成

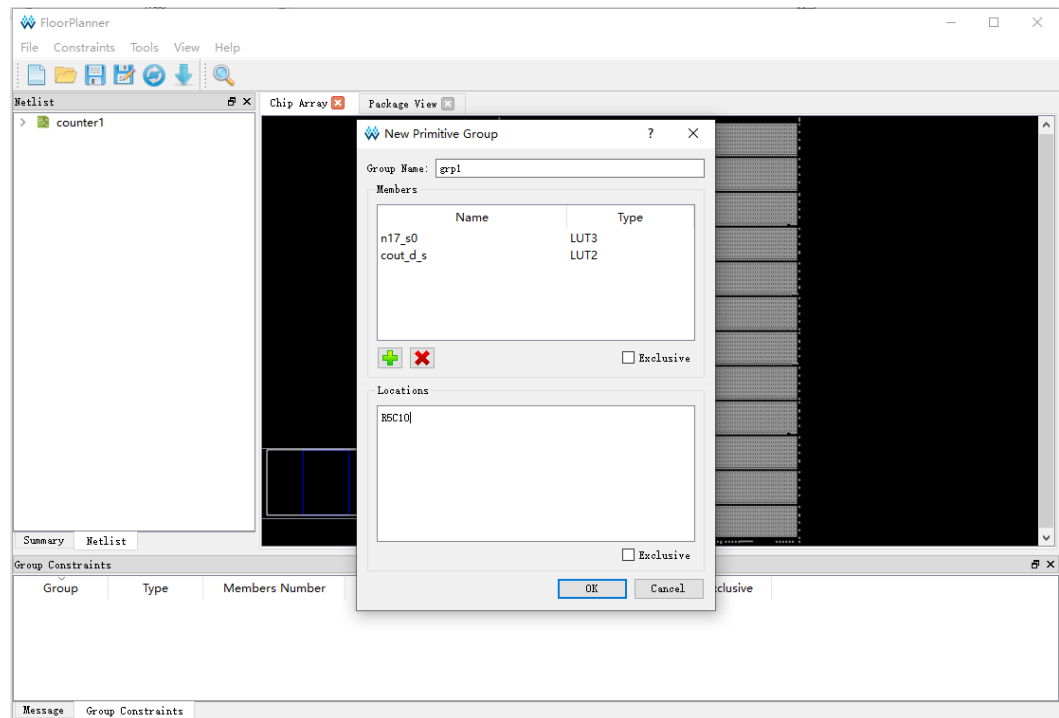
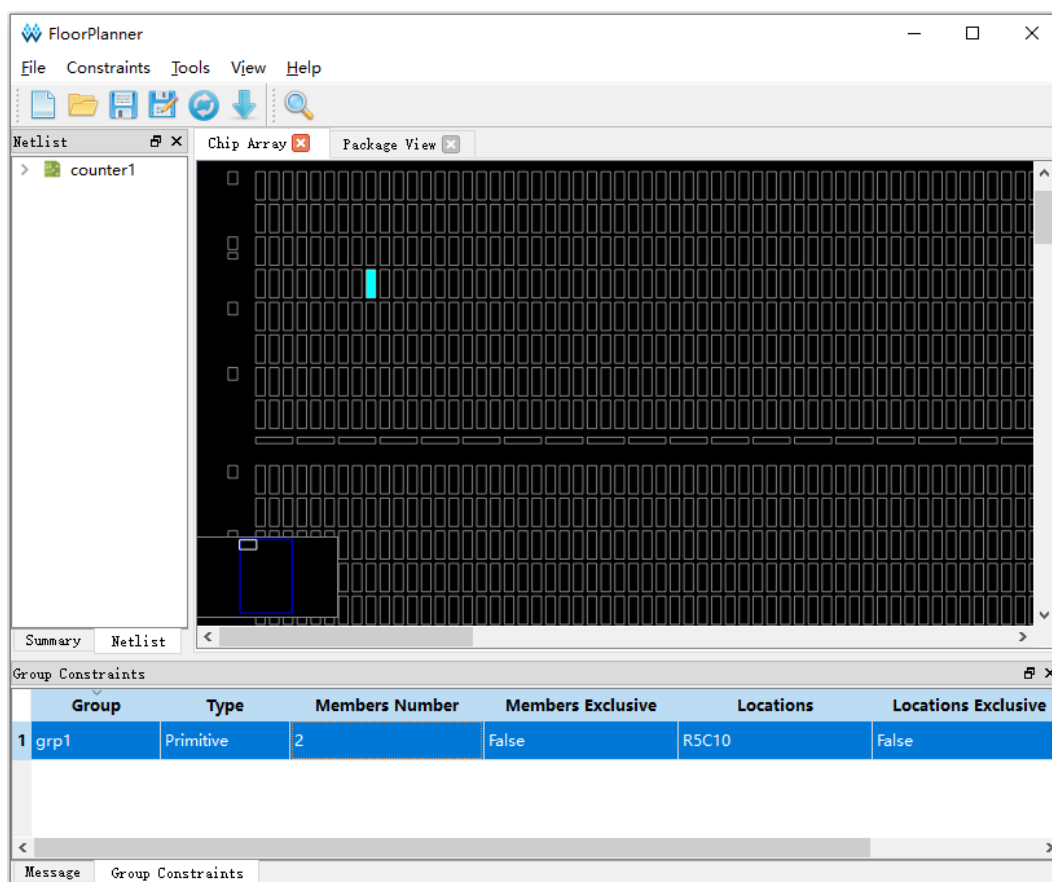


図 4-9 Primitive Group Constraints



注記：

Primitive Group Constraints の Location 情報は、手動で入力するか、Chip Array ウィンドウからコピーしてペーストします。ドラッグでは入力できません。

相対位置グループ制約の編集

1. "Group Constraints"編集ウィンドウで右クリックして"**New Relative Group**"を選択すると、"**New Relative Group**"ダイアログボックスがポップアップします。
2. Group の名前“rel_grp”を入力し、“”をクリックすると、Primitive Finder ダイアログボックスがポップアップします。
3. Primitive Finder ダイアログボックスで設定したい Primitives “cout_0_s1”、“cout_1_s1”を選択して“OK”をクリックします。
4. これらの Primitive に相対位置"R0C0"、"R4C5"を追加します(図 4-10)。
5. "OK"をクリックして Relative Group Constraints を作成します(図 4-11)。

図 4-10 Relative Group Constraints の作成

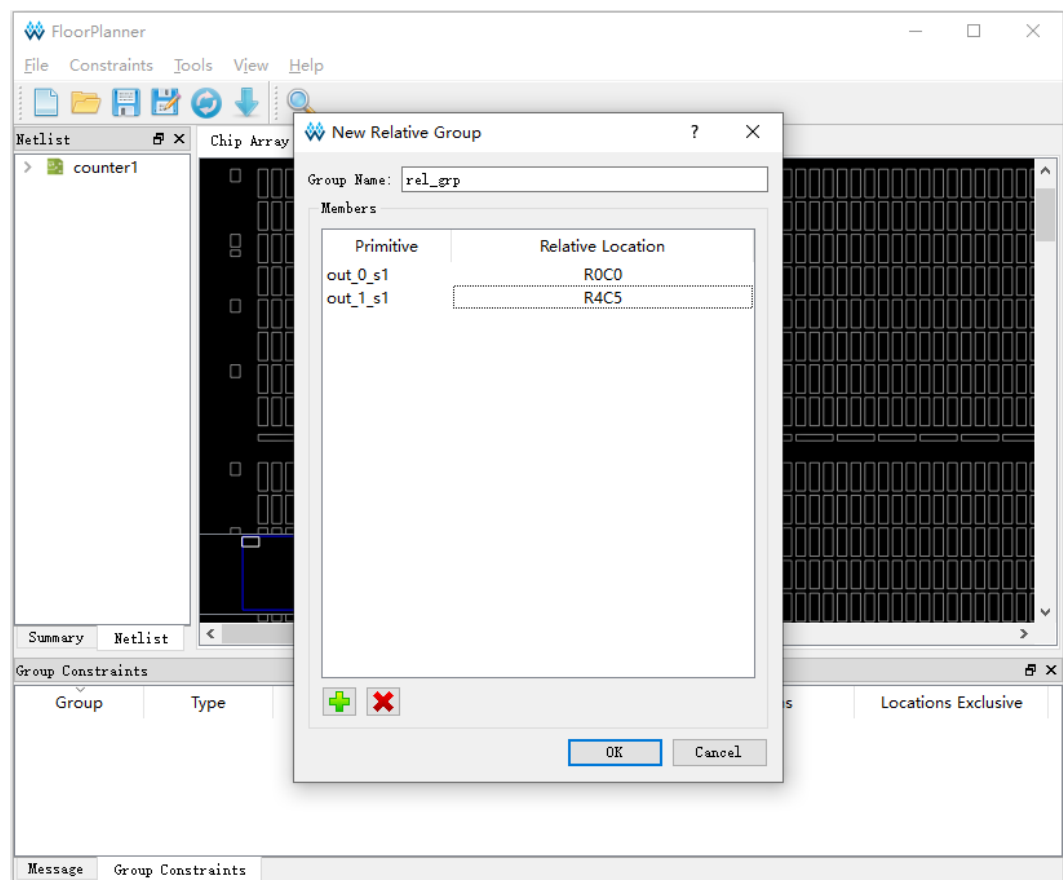
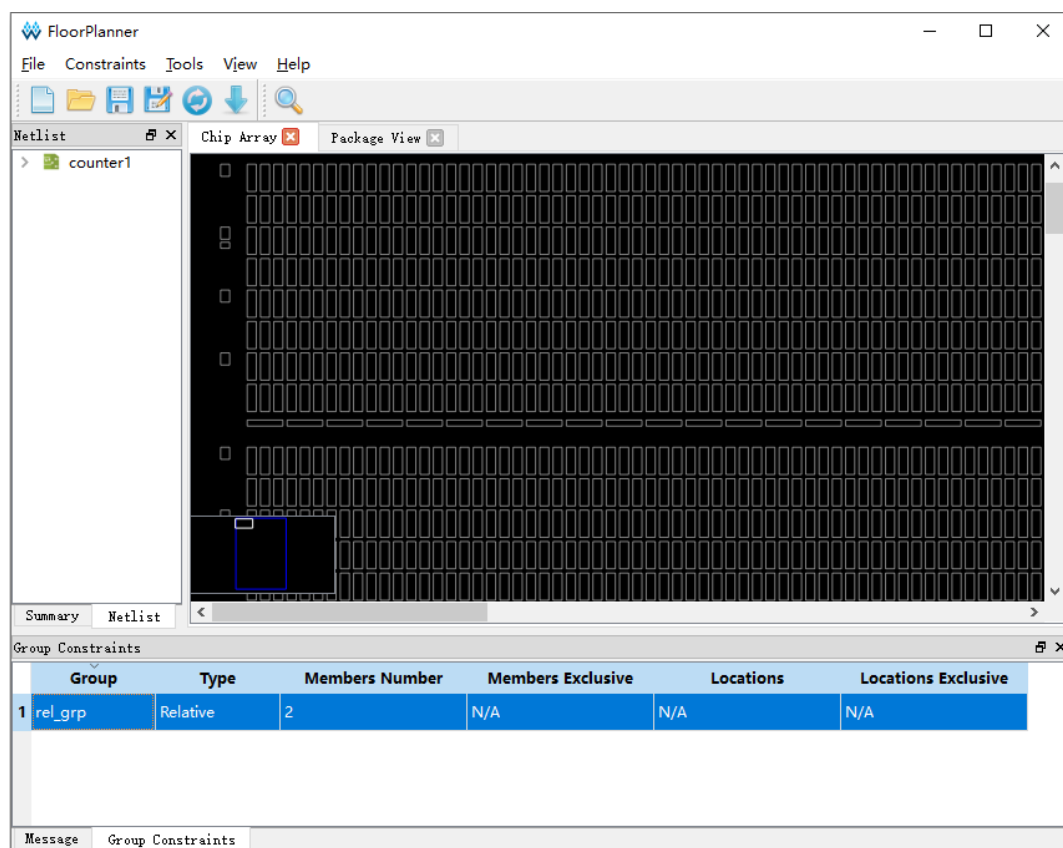


図 4-11 Relative Group Constraints



4.2.5 リザーブ制約の編集

1. "Resource Reservation"編集ウィンドウで、右クリックメニューから "Reserve Resources"をクリックして Resource Reservation 制約を追加します(図 4-12)。
2. 作成された Resource Reservation 制約を Chip Array ウィンドウにドラッグします。図 4-13 に示すように、BSRAM_R10[1]にドラッグアンドドロップして、Resource Reservation 制約を完了します。

図 4-12 Resource Reservation 制約の作成

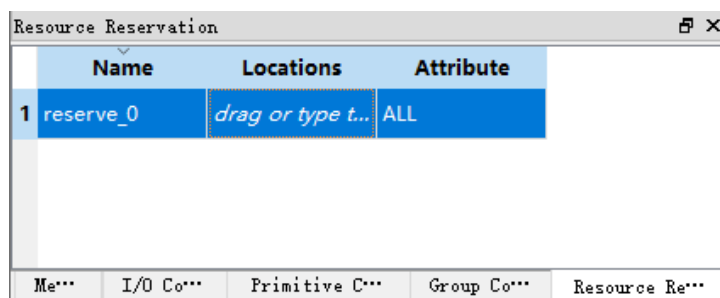
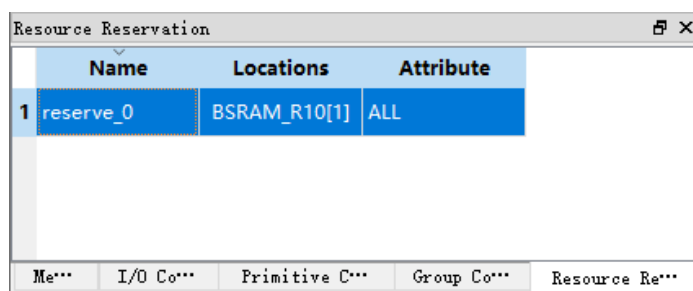


図 4-13 Resource Reservation



	Name	Locations	Attribute
1	reserve_0	BSRAM_R10[1]	ALL

Me... I/O Co... Primitive C... Group Co... Resource Re...

4.2.6 グローバルクロック割り当て制約の編集


1. Clock Net Constraints 編集ウィンドウで右クリックして**"Clock Net Constraints"**を選択すると、**"Clock Net Constraints"**ダイアログボックスがポップアップします。
2.  をクリックすると、**"Select Net"**ダイアログボックスが表示されます。制約したい **Net** を選択して**"OK"**をクリックします。
3. **Type** ドロップダウン・リストからタイプを選択し、**Signal** タイプを設定します(図 4-14)。
4. **"OK"**をクリックして **Clock Net Constraints** の作成を完了します(図 4-15)。

図 4-14 Clock Net Constraints の作成

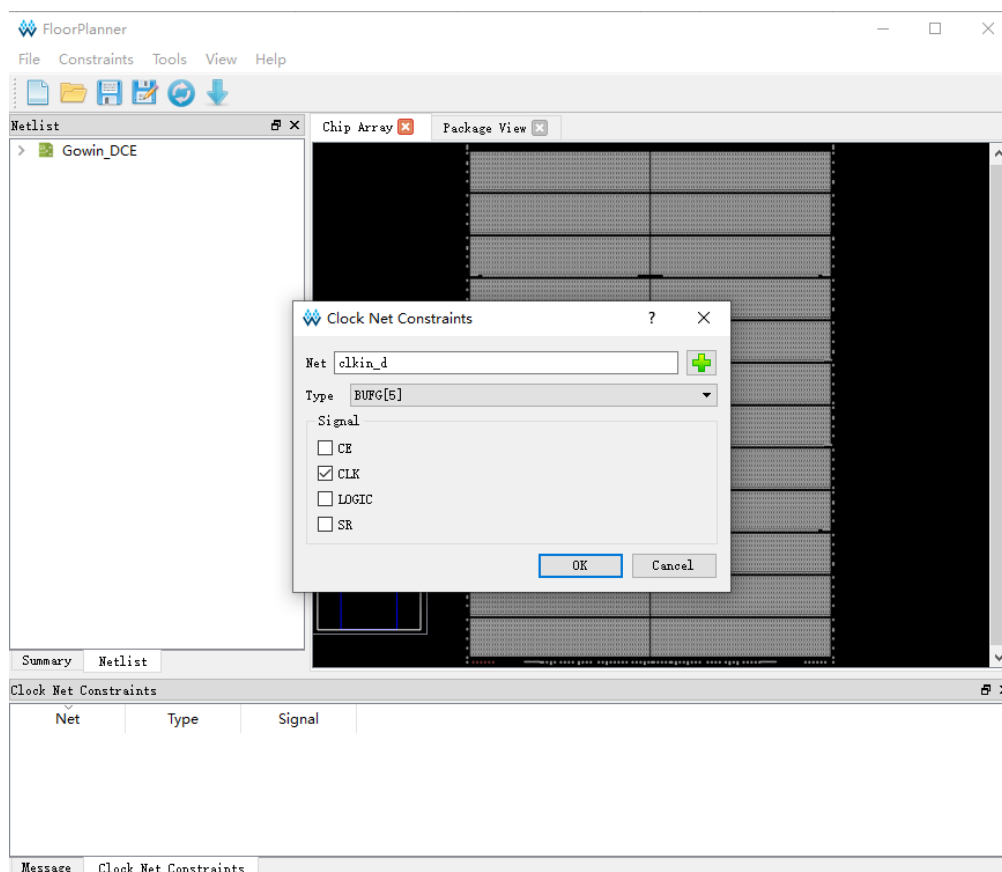
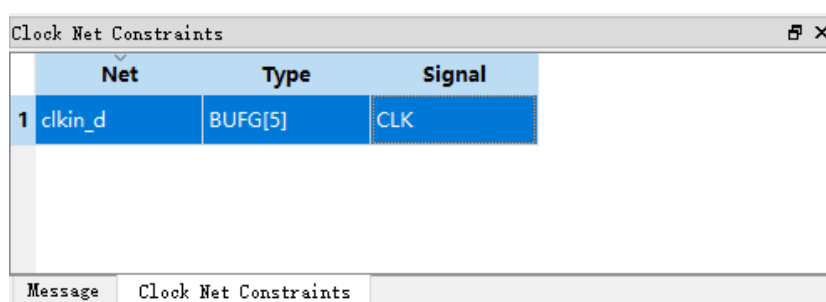


図 4-15 Clock Net Constraints



4.2.7 グローバルクロック制約の編集

GCLK Primitive Constraints は、DCS と DCE に対する制約のみをサポートします。

GCLK Primitive Constraints の作成手順は以下のとおりです。

1. GCLK Primitive Constraints 編集ウィンドウで右クリックして “Select GCLK Primitive” を選択すると、“GCLK Primitive Constraints” ダイアログボックスが表示されます。
2. “” をクリックすると GCLK 選択ダイアログボックスがポップアップ

プします。Instance を選択して “OK” をクリックし、Instance の設定を完了します。

- 図 4-16 に示すように、“GCLK Primitive Constraints”ダイアログボックスの“Position”で制約したいグローバルクロック位置を選択します。
- GCLK Primitive Constraints ダイアログボックスで“OK”をクリックして、制約を GCLK Primitive Constraints 編集ウィンドウに追加します(図 4-17)。

図 4-16 GCLK Primitive Constraints の作成

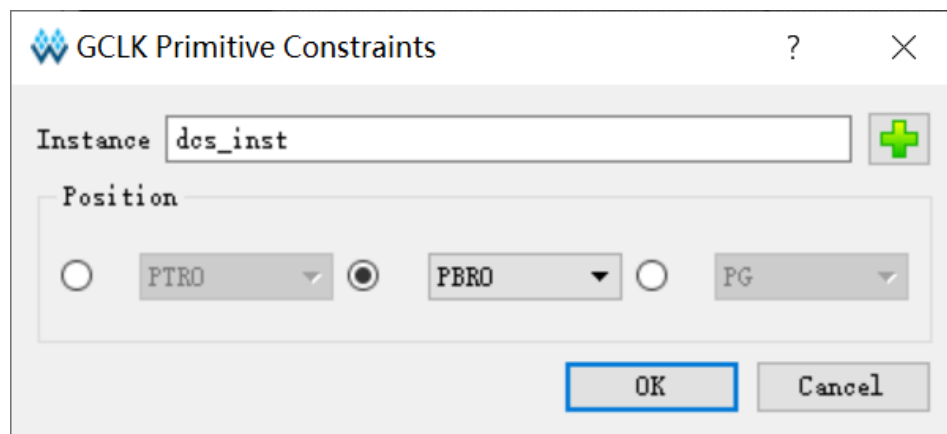



図 4-17 GCLK Primitive Constraints



4.2.8 高速クロック制約の編集

HCLK Primitive Constraints は、CLKDIV と DLLDLY Instance に対する制約のみをサポートします。

HCLK Primitive Constraints の作成手順は以下のとおりです。

- HCLK Primitive Constraints 編集ウィンドウで右クリックして “Select HCLK Primitive” を選択すると、“HCLK Primitive Constraints” ダイアログボックスが表示されます。
- “” をクリックすると HCLK ダイアログボックスがポップアップします。Instance を選択して “OK” をクリックし、Instance の設定を完了します。

3. 図 4-18 に示すように、"HCLK Primitive Constraints"ダイアログボックスの"Position"で制約したい高速クロック位置を選択します。
4. HCLK Primitive Constraints ダイアログボックスをクリックして"OK"をクリックして、制約を HCLK Primitive Constraints 編集ウィンドウに追加します(図 4-19)。

図 4-18 HCLK Primitive Constraints の作成

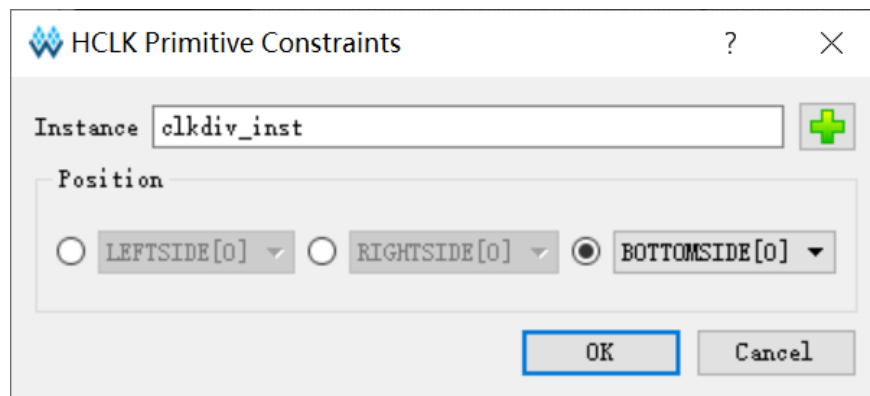
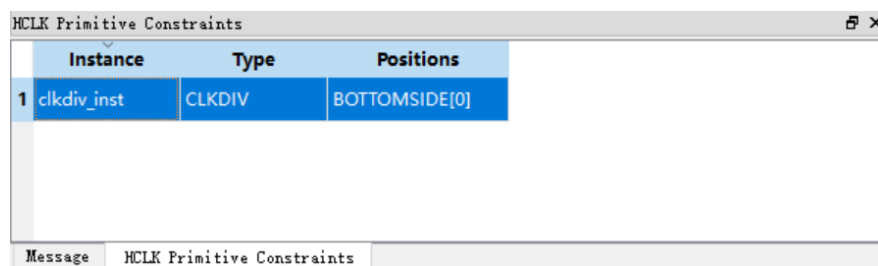


図 4-19 HCLK Primitive Constraints

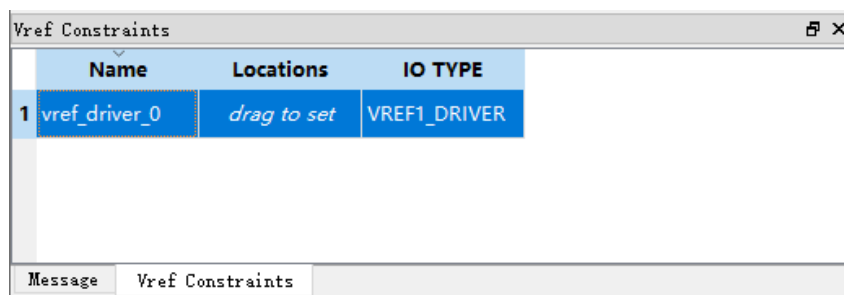


4.2.9 リファレンス電圧制約の編集

Chip Array ウィンドウにドラッグして **Constraints** を作成します。手順は次のとおりです。

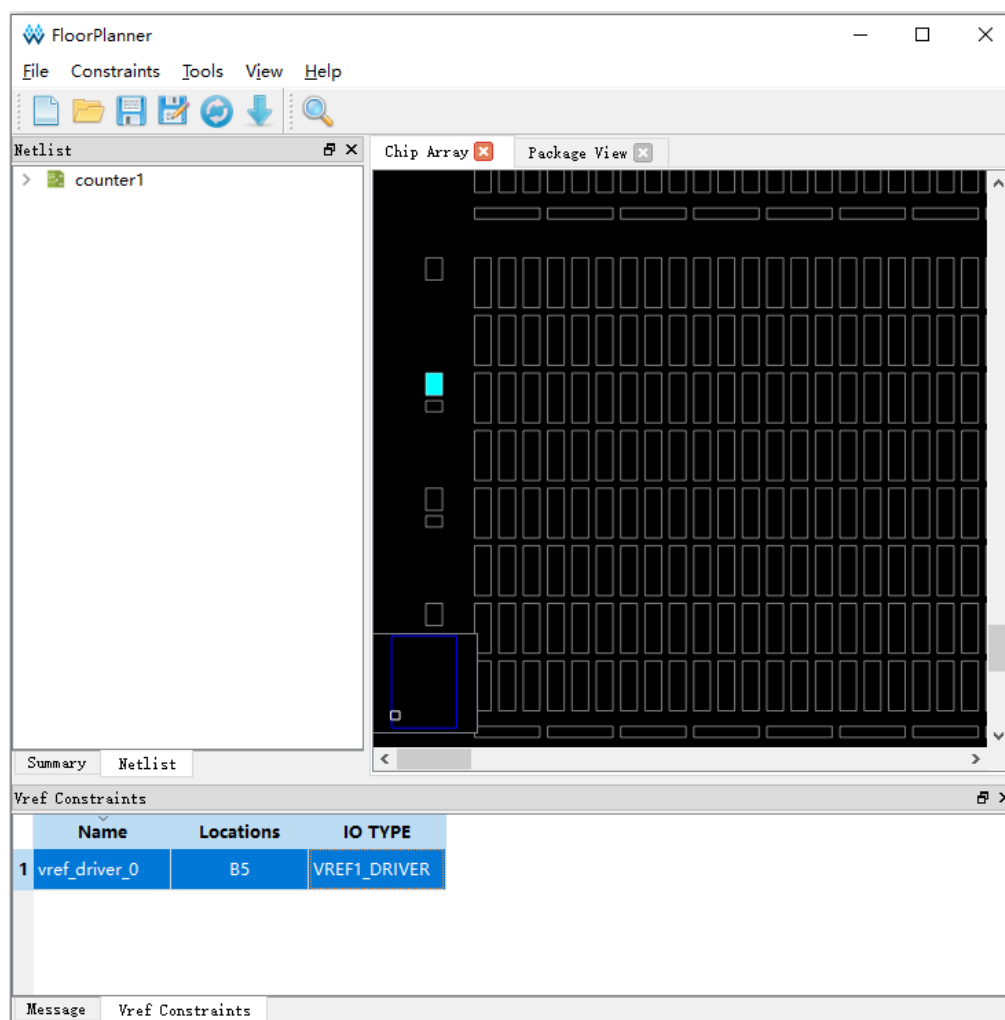
1. "Vref Constraints"編集ウィンドウで右クリックメニューから"Define Vref Driver"を選択してこの Vref Constraints 制約を追加します(図 4-20)。
2. Chip Array ウィンドウをマクロセル・モードに拡大し、Vref Constraints 編集ウィンドウで新しく作成された Vref Constraints を選択して Chip Array ウィンドウの B5 の位置にドラッグします。Vref Constraints の Location 情報は、B5 になります(図 4-21)。

図 4-20 Vref Constraints の作成



Vref 制約名はカスタマイズできます。Vref 名の重複は許可されません。設定中に名前が重複する場合、プロンプトが表示されます(図 4-23)。

図 4-21 Chip Array ウィンドウにドラッグして Vref Constraints Location を生成



Package View にドラッグして Vref Constraints を作成します。手順は次のとおりです。

1. "Vref Constraints"編集ウィンドウで右クリックメニューから"Define

Vref Driver"を選択してこの Vref Constraints 制約を追加します(図 4-20)。

2. "Vref Constraints"編集ウィンドウで新しく作成された Vref Constraints を選択して Package View ウィンドウの B5 の位置にドラッグします。Vref Constraints の Location 情報は、B5 になります(図 4-22)。

図 4-22 Package View ウィンドウにドラッグして Vref Constraints Location を生成

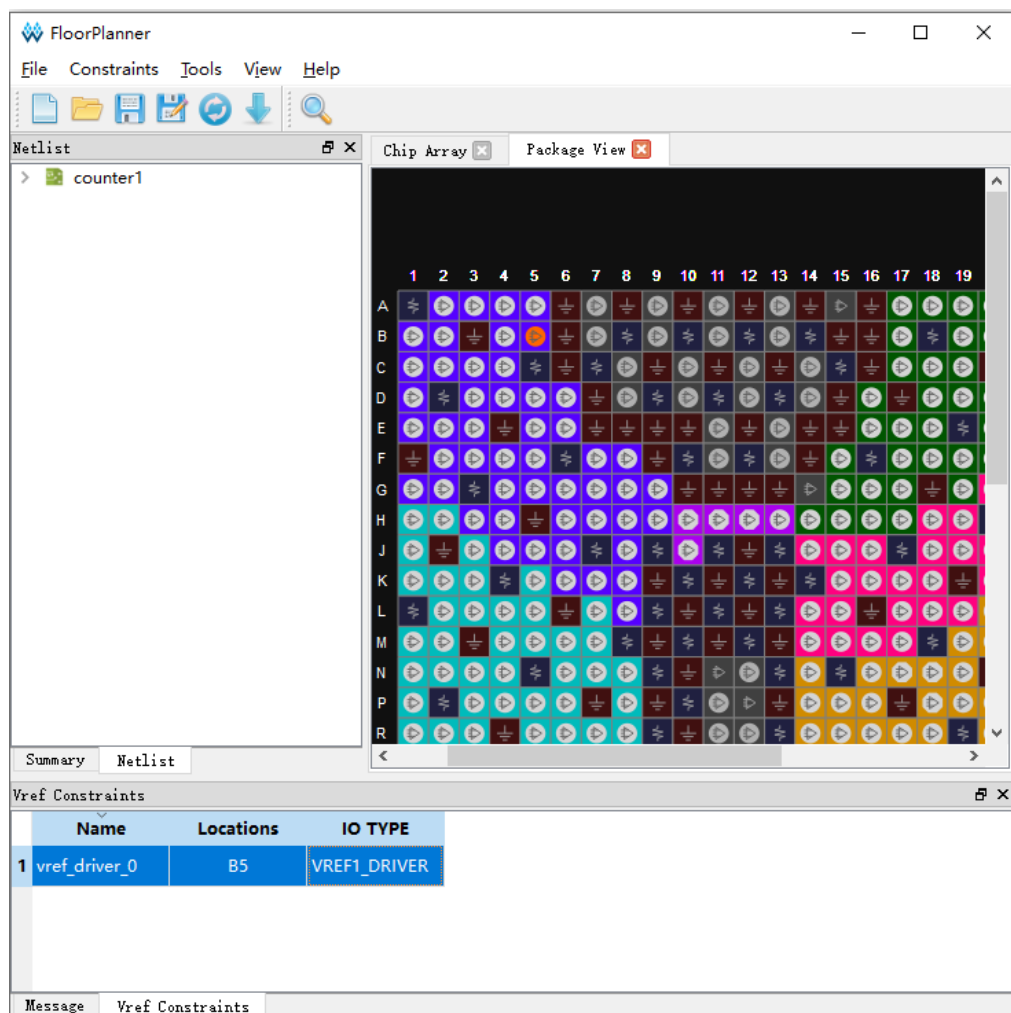


図 4-23 Vref Constraints 名前の重複

