



Gowin goConfig I2C IP

ユーザーガイド

IPUG795-2.1J, 2023-09-27

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かに拘わらず)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
2022/03/09	1.0J	初版。
2022/10/17	2.0J	● サポートされるデバイスの情報を更新。 ● GW1N(R)-2C シリーズの説明を更新。 ● GW2AN-9X シリーズの説明を更新。
2023/09/27	2.1J	● 一部の説明を最適化。 ● チャプター「6 活用例の説明」を追加。

目次

目次	i
図一覧.....	iii
表一覧.....	v
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	2
2 IP の紹介	3
2.1 アップグレード機能の概要.....	4
2.2 I ² C インターフェース	4
2.2.1 I ² C の一般的な特性.....	5
2.2.2 I ² C のタイミング特性	5
2.3 IP の特徴.....	7
2.4 リソース使用量.....	8
3 機能の説明	9
3.1 goConfig I2C の構造	9
3.2 Gowin goConfig I2C IP の外部回路の接続	10
3.3 Gowin goConfig I2C IP のバックグラウンドアップグレード命令	11
3.3.1 2C シリーズ.....	11
3.3.2 9X シリーズ	17
4 信号の定義	23
4.1 信号の定義	23
5 IP の使用法.....	26
5.1 2C シリーズ.....	26
5.1.1 ソフトウェアでの構成	26

5.1.2 手順の説明.....	27
5.1.3 注意事項.....	30
5.2 9X シリーズ.....	30
5.2.1 ソフトウェアでの構成.....	30
5.2.2 手順の説明.....	32
5.2.3 注意事項.....	33
6 活用例の説明.....	35
6.1 2C シリーズでの活用例.....	35
6.1.1 MCU と IP 間の通信が正常かどうかの確認.....	35
6.1.2 バックグラウンドアップグレードの手順.....	36
7 GUI での構成.....	42
8 リファレンスデザイン.....	45
8.1 リファレンスデザイン.....	45
8.1.1 FPGA プロジェクトの説明.....	45
8.1.2 リファレンスデザイン.....	46
8.1.3 リファレンスデザインの注意事項.....	47

図一覧

図 2-1 I ² C 7-bit アドレスプロトコルの典型的なタイミング例	5
図 2-2 I ² C インターフェースのタイミング図	5
図 3-1 goConfig I2C ブロック図 (2C シリーズ)	10
図 3-2 goConfig I2C ブロック図 (9X シリーズ)	10
図 3-3 外部抵抗回路の接続	11
図 3-4 ステータスレジスタ読み出し命令の例	13
図 3-5 IP リセット命令の例	14
図 3-6 リードバック命令の例	15
図 3-7 FIFO データ読み出し命令の例	15
図 3-8 待機命令の例	15
図 3-9 データを送信して Flash からデータをリードバックしない命令の例	16
図 3-10 Reconfign 命令の例	16
図 3-11 IP バージョン命令	16
図 3-12 CSCR レジスタの使用例	18
図 3-13 DICN レジスタ書き込みの使用例	18
図 3-14 DICN レジスタ読み出しの使用例	19
図 3-15 DOCN レジスタ書き込みの使用例	19
図 3-16 DOCN レジスタ読み出しの使用例	19
図 3-17 DIFR レジスタの使用例	20
図 3-18 SRSTR レジスタの使用例	20
図 3-19 SR レジスタの使用例	21
図 3-20 DOFR レジスタの使用例	22
図 5-1 Configuration ウィンドウを開く	26
図 5-2 バックグラウンドアップグレードのインターフェースとして I2C を選択	27
図 5-3 Configuration ウィンドウを開く	31
図 5-4 バックグラウンドアップグレードのインターフェースとして I2C を選択	31
図 5-5 IO の定義例	34
図 6-1 構造	35

図 6-2 MCU と IP 間の通信が正常かどうかを確認するためのフローチャート	36
図 6-3 バックグラウンドアップグレードのフローチャート	37
図 6-4 オンチップ Flash の消去のフローチャート	39
図 6-5 オンチップ Flash のプログラミングのフローチャート	41
図 7-1 IP Core Generator を開く方法	42
図 7-2 goConfig I2C IP コアを開く	43
図 7-3 goConfig I2C IP の構成画面	44
図 8-1 リファレンスデザインの基本構造	46

表一覧

表 1-1 用語、略語	2
表 2-1 Gowin goConfig I2C IP の概要	3
表 2-2 2C シリーズの I ² C インターフェースのタイミングの説明	6
表 2-3 9X シリーズの I ² C インターフェースのタイミングの説明	6
表 2-4 2C シリーズのリソース使用量	8
表 2-5 9X シリーズのリソース使用量	8
表 3-1 2C シリーズの命令の定義	12
表 3-2 ステータスレジスタ	13
表 3-3 レジスタの定義	17
表 3-4 CSCR-Data の定義	18
表 3-5 DICN-Data の定義	18
表 3-6 DOCN-Data の定義	19
表 3-7 DIFR-Data の定義	20
表 3-8 SRSTR-Data の定義	20
表 3-9 SR-Data の定義	21
表 3-10 DOFR-Data の定義	22
表 4-1 2C シリーズの I2C インターフェースの定義	23
表 4-2 9X シリーズの I2C インターフェースの定義	25
表 5-1 対応関係	33
表 8-1 リファレンスデザインの src フォルダのコンテンツ一覧(2C シリーズ)	46
表 8-2 リファレンスデザインの src フォルダのコンテンツ一覧(9X シリーズ)	47

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 GOWIN セミコンダクターの Gowin goConfig I2C IP を使いこなせるように作成されています。本マニュアルには、主に IP の紹介、機能の説明、信号の定義、使用法、活用例、GUI での呼び出しなどが含まれます。

注記：

Gowin goConfig I2C IP の機能は、さまざまなデバイスによってわずかに異なります。現在、GW1N-1P5C/GW1N(R)-2C シリーズ（以下、2C シリーズ）と GW2AN-9X/GW2AN-18X シリーズ（以下、9X シリーズ）をそれぞれ説明します。特に説明のない限り、共通の特性または機能です。

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がダウンロード、参考できます：

- GW1N シリーズ FPGA 製品データシート([DS100](#))
- GW1NR シリーズ FPGA 製品データシート([DS117](#))
- GW2AN-18X & 9X FPGA 製品データシート([DS971](#))
- Gowin ソフトウェア ユーザーガイド([SUG100](#))
- Gowin プリミティブ ユーザーガイド([SUG283](#))
- Gowin 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([UG290](#))

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
ACK	Acknowledge	応答
FPGA	Field Programmable Gate Array	フィールド・プログラマブル・ゲート・アレイ
I ² C(I2C)	Inter-Integrated Circuit	2 線式シリアルバス
IP	Intellectual Property	設計資産
NACK	Not Acknowledge	否定応答
SRAM	Static Random Access Memory	スタティック RAM

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 IP の紹介

バックグラウンドアップグレードは、近年ユーザーからの要望が非常に高い機能です。例えば、Gowin FPGA 製品に「機能 A」があって、ユーザーがチップの「機能 A」を停止することなく「機能 A」を「機能 B」にアップグレードまたは再プログラムし、かつアップグレードが完了した後に「機能 B」がすぐにチップにロードされるようにしたい場合、バックグラウンドアップグレード技術が必要です。

Gowin goConfig I2C IP は、I²C インターフェースのオンラインアップグレード機能を実現できます。この IP により、ユーザーは、I²C インターフェースを介して FPGA の内部メモリに格納されているコンフィギュレーションデータを置き換えてバックグラウンドアップグレードを実現できます。。

表 2-1 Gowin goConfig I2C IP の概要

Gowin goConfig I2C IP	
IP コアの適用	
ロジックリソース	表2-4と表2-5参照。
提供されるファイル	
設計ファイル	Verilog (暗号化)
リファレンスデザイン	Verilog
テストベンチ	Verilog
テストおよび設計フロー	
合成ソフトウェア	GowinSynthesis®
統合開発環境	Gowin ソフトウェア (V1.9.7Beta以降)

注記：

デバイスのサポート情報は、Gowin セミコンダクターの[ホームページ](#)で確認できます。

2.1 アップグレード機能の概要

アップグレード機能には、プログラミング(Program)とコンフィギュレーション(Configure)の 2 つのプロセスがあります。

- プログラミング (Program Flash) : Gowin ソフトウェアによって生成されたビットストリーム データを Flash に書き込むプロセス (以下、「Flash に書き込む」と呼びます)。
- コンフィギュレーション(Configure SRAM) : Flash 内のビットストリームデータを FPGA の SRAM にロードするプロセス (以下、「SRAM にロードする」と呼びます)。

注記 :

- 通常のアップグレード: プログラミング (Program) を行う場合、「Flash に書き込む」前に SRAM を消去 (つまり、実行中の機能を停止します) する必要があります。
- バックグラウンドアップグレード: プログラミング (Program) を行う場合、SRAM は消去されず (つまり、実行中の機能には影響しません)、直接 Flash に書き込みます。
- コンフィギュレーション (Configure) 段階では、「通常のアップグレード」と「バックグラウンドアップグレード」の操作手順は同じです。

バックグラウンドアップグレード機能の例は次のとおりです。

ユーザーは通常、まず実行中の「機能 A」を介して、更新される「機能 B」を Flash に書き込みます。そして機能を更新したい場合、ユーザーは一連の操作を介して Flash 内の「機能 B」を SRAM にロードできます。

注記 :

- バックグラウンドアップグレードの際、「機能 A」はバックグラウンドアップグレード用のモジュールを持つ必要があります。「機能 A」には Gowin goConfig I2C IP が含まれている必要があります。
- 「機能 B」はバックグラウンドアップグレード用のモジュールを持つ必要がありませんが、持たない場合、後で「機能 B」を介してバックグラウンドアップグレードを実行することはできません。
- 「Flash に書き込む」のプロセスでは、「機能 A」は引き続き正常に実行され、「SRAM にロードする」のプロセスでは、「機能 A」は停止されます。

2.2 I²C インターフェース

Gowin goConfig I2C IP の I2C インターフェースの通信方式は、I2C 公式マニュアルの 7-bit アドレスプロトコルに準拠しています。IP の使用方法の詳細については、3 機能の説明および 5IP の使用法を参照してください。

2.2.1 I²C の一般的な特性

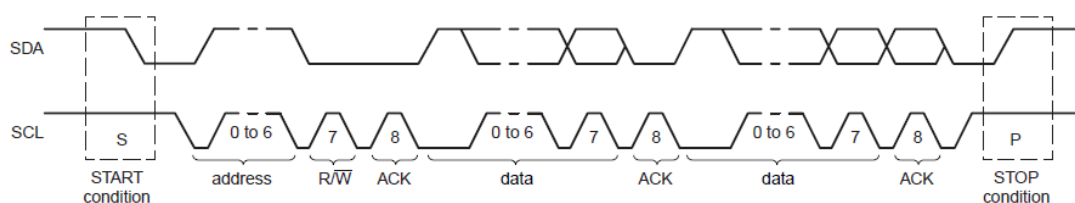
I²C の 7-bit アドレスプロトコルの典型的なタイミング例は図 2-1 のとおりです。その説明を次に示します。

- S : START condition、通信開始フラグ。
- address : スレーブアドレス、幅は 7 ビット。
- R/W : 読み出し書き込み制御ビット。0 の場合は書き込み、1 の場合は読み出し。
- ACK : スレーブ応答ビット。0 の場合は応答(ACK)、1 の場合は否定応答(NACK)。
- data : 読み出し/書き込み際の有効データ、幅は 8 ビット。
- S : STOP condition、通信終了フラグ。

注記 :

マルチビットの幅の場合、上位ビットが最初に送信されます。例: address=7'h58 の場合、ビットが送信される順序は 101_1000 です。data=8'hB1 の場合、ビットが送信される順序は 1011_0001 です。

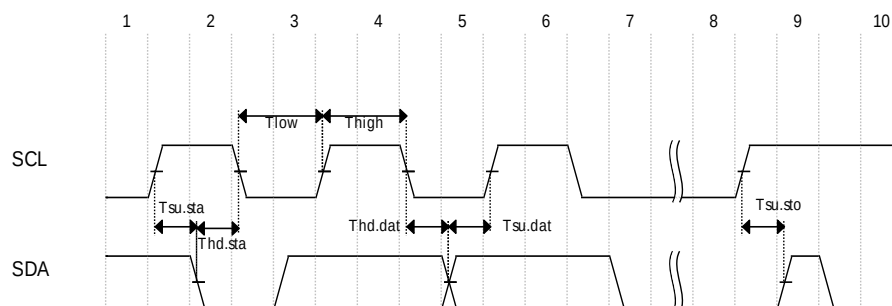
図 2-1 I²C 7-bit アドレスプロトコルの典型的なタイミング例



2.2.2 I²C のタイミング特性

Gowin goConfig I2C IP は、2C シリーズと 9X シリーズで異なるタイミング特性を持っています。そのタイミング図を図 2-2 に示します。

図 2-2 I²C インターフェースのタイミング図



2C シリーズの I2C インターフェースのタイミング

2C シリーズの Gowin goConfig I2C IP のタイミングは、公式の I2C マニュアルの定義と互換性があり、その詳細を表 2-2 に示します。I²C の検証済み最大伝送速度は 400KHz です。

表 2-2 2C シリーズの I²C インターフェースのタイミングの説明

名称	説明	最小値	最大値	単位	備考
F _{SCL}	Clock Frequency	-	400	KHz	-
T _{low}	LOW period of the SCL	1.3	-	μ S	
T _{high}	HIGH period of the SCL	0.6	-	μ S	
T _{hd.sta}	Start Hold Time	0.6	-	μ S	
T _{su.sta}	Start Setup Time	0.6	-	μ S	
T _{hd.dat}	Data In Hold Time	0	-	ns	
T _{su.dat}	Data In Setup Time	100	-	ns	
T _{su.sto}	Stop Setup Time	0.6	-	μ S	

9X シリーズの I2C インターフェースのタイミング

9X シリーズの Gowin goConfig I2C IP のタイミングは、一部のパラメータが公式の I2C マニュアルの定義と異なります。その詳細を表 2-3 に示します。I²C の検証済み最大伝送速度は 400KHz です。

表 2-3 9X シリーズの I²C インターフェースのタイミングの説明

名称	説明	最小値	最大値	単位	備考
F _{SCL}	Clock Frequency	-	400	KHz	-
T _{low}	LOW period of the SCL	1.3	-	μ S	
T _{high}	HIGH period of the SCL	0.6	-	μ S	
T _{hd.sta}	Start Hold Time	0.6	-	μ S	
T _{su.sta}	Start Setup Time	0.6	-	μ S	
T _{hd.dat}	Data In Hold Time	50 ^[1]	-	ns	
T _{su.dat}	Data In Setup Time	100	-	ns	
T _{su.sto}	Stop Setup Time	0.6	-	μ S	

注記：

[1] T_{hd.dat} は I²C 公式マニュアルの内容と異なります。

2.3 IP の特徴

Gowin goConfig I2C (2C シリーズ) IP の特徴は次のとおりです：

- I²C インターフェースはスレーブモード
- 最大 400Kbps の I²C 伝送速度をサポート
- I²C インターフェースは 7-bit アドレスモードをサポート
- I²C インターフェースのスレーブアドレスは自由に構成可能
- 内部ストレージの読み出しと書き込みをサポート
- ユーザーデザイン用の 25MHz クロック出力をサポート
- 基本的な I²C 読み出しおよび書き込みプロトコルのみをサポート
- クロックとリセット信号を内部で生成
- 完全に合成可能

Gowin goConfig I2C (9X シリーズ) IP の特徴は次のとおりです：

- I²C インターフェースはスレーブモード
- 最大 400Kbps の I²C 伝送速度をサポート
- I²C インターフェースは 7-bit アドレスモードをサポート
- I²C インターフェースのスレーブアドレスは自由に構成可能
- 内部ストレージの読み出しと書き込みをサポート
- 20MHz のクロック出力をサポート
- クロックとリセット信号を内部で生成
- 完全に合成可能

2.4 リソース使用量

例えば、GW1N-2C デバイスの場合、Gowin goConfig I2C IP（2C シリーズ）のリソース使用量は表 2-4 に示すようになります。

表 2-4 2C シリーズのリソース使用量

デバイス	プログラミング言語	LUT	REG	BSRAM
GW1N-2C	Verilog	608	241	1

例えば、GW2AN-9X デバイスの場合、Gowin goConfig I2C IP（9X シリーズ）のリソース使用量は表 2-5 に示すようになります。

表 2-5 9X シリーズのリソース使用量

デバイス	プログラミング言語	LUT	REG	BSRAM
GW2AN-9X	Verilog	444	247	2

3 機能の説明

3.1 goConfig I2C の構造

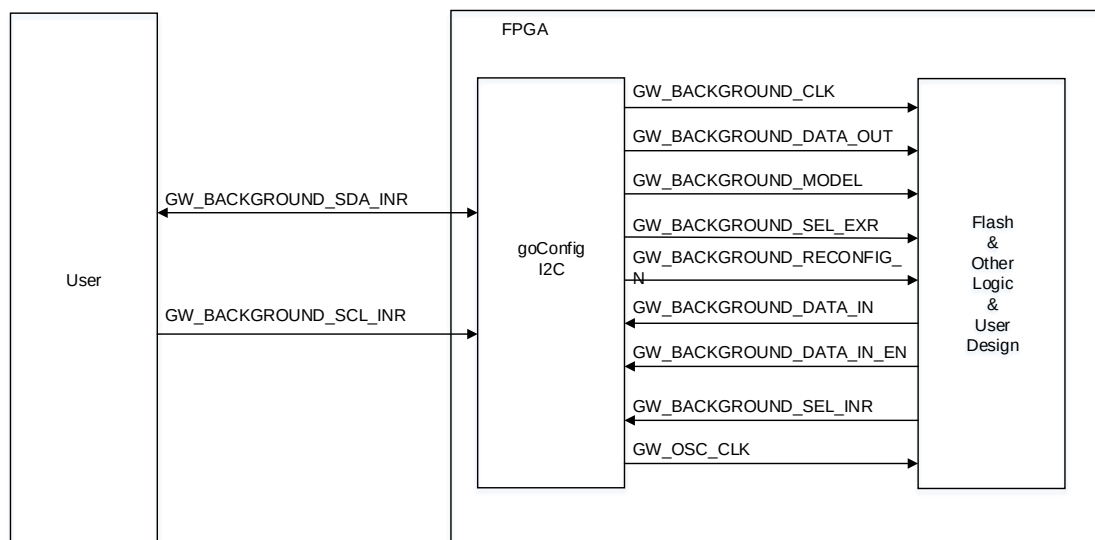
Gowin goConfig I2C IP のアプリケーションは、ユーザー(User)側と FPGA 側に分けられます。User 側は、I2C インターフェースと他のユーザー機能を制御するために使用されます。FPGA 側は、Flash と、ユーザーが FPGA 上で実装したい機能に接続します。IP のみを考慮する場合、User 側は、I2C インターフェース経由で FPGA 内の IP を制御することにより Flash を制御して、バックグラウンドアップグレードを実装します。

注記：

本マニュアルでは、主に User 側インターフェースの操作について説明しており、Flash インターフェースの詳細には言及しません。

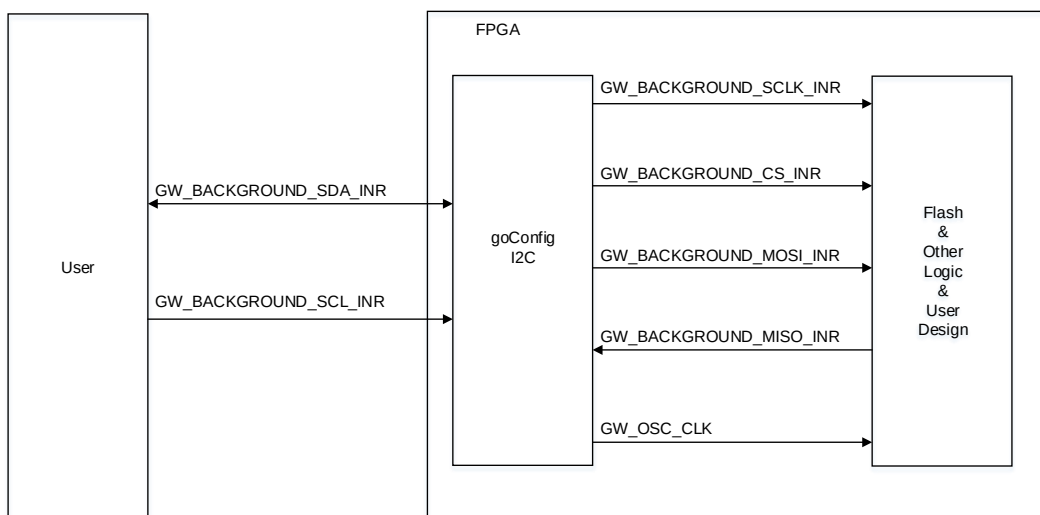
2C シリーズの goConfig I2C ブロック図を図 3-1 に示します。I2C インタフェース(GW_BACKGROUND_SCL_INR、GW_BACKGROUND_SDA_INR)は User 側で制御する必要があり、その他の信号は実際のニーズに応じて制御または使用します。

図 3-1goConfig I2C ブロック図 (2C シリーズ)



9X シリーズの goConfig I2C ブロック図を図 3-2 に示します。I2C インタフェース(GW_BACKGROUND_SCL_INR、GW_BACKGROUND_SDA_INR)は User 側で制御され、その他の信号も必要に応じて制御または使用されます。

図 3-2goConfig I2C ブロック図 (9X シリーズ)



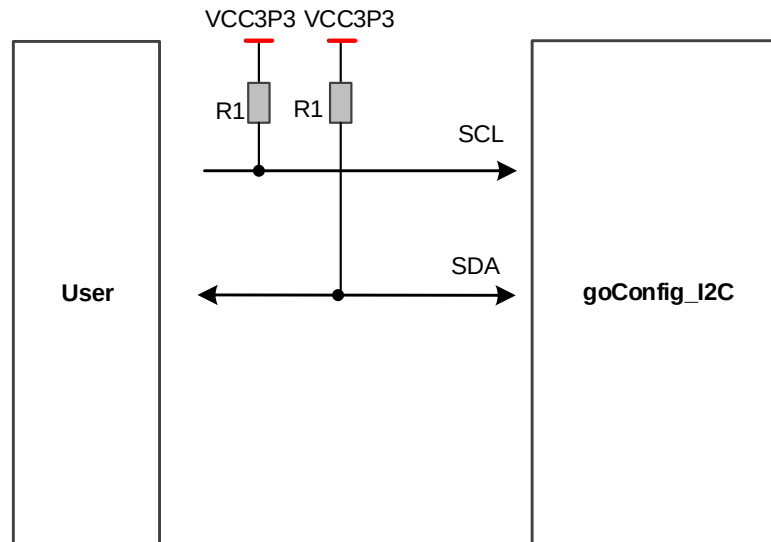
3.2 Gowin goConfig I2C IP の外部回路の接続

注記：

このセクションの説明は、2C シリーズと 9X シリーズの両方に適用されます。

Gowin goConfig I2C IP の外部抵抗回路の接続を図 3-3 に示します。

図 3-3 外部抵抗回路の接続



R1:4.7K

注記：

- IO ポートの属性制約は次のとおりです。
 - GW_BACKGROUND_SCL_INR : IO_TYPE=LVC MOS33 PULL_MODE= UP
 - GW_BACKGROUND_SDA_INR : IO_TYPE=LVC MOS33 PULL_MODE= UP
DRIVE=8
- I/O が所在する Bank の電源電圧は 3.3V です。

3.3 Gowin goConfig I2C IP のバックグラウンドアップグレード命令

Gowin goConfig I2C IP は、2C シリーズと 9X シリーズで同じように動作しないため、以下は、2C シリーズと 9X シリーズ別で個別に説明します。

3.3.1 2C シリーズ

命令の定義

Gowin goConfig I2C IP 2C シリーズでサポートされている命令(コマンド)を表 3-1 に示します。

表 3-1 I2C シリーズの命令の定義

番号	命令名	命令のアドレス (CMD)	説明	命令のフォーマット	備考
1	Status	0x44	ステータスレジスタ読み出し命令	CMD	-
2	Reset	0x55	IPリセット命令	CMD	
3	Send (data back)	0x66	データを送信してFlashからデータをリードバックする命令	CMD + Length + Data + Check_sum	
4	Waiting	0x77	待機命令	CMD + Length + Check_sum	
5	Send (No data back)	0x88	データを送信してFlashからデータをリードバックしない命令	CMD + Length + Data + Check_sum	
6	Reconfign	0x99	Reconfign(再度コンフィギュレーション)命令	CMD	-
7	IP Version	0xAA	IPバージョン命令	CMD	

命令フォーマットの定義および説明は次のとおりです。

- **CMD**: サポートされる命令については表 3-1 参照。
 - 定義: 命令アドレス。
 - 幅: 8 ビット。
- **Length**: 最小値は 2
 - 定義: 送信する有効バイト長^[1]。
 - 幅: 16 ビット。下位 8 ビット (**Length_L**) は上位 8 ビット (**Length_H**) より先に送信されます。
 - 計算式: **Length**=有効データ長+2。

注記:

- [1] 有効バイト長は、有効データ (**Data**) のバイト数+2 です。例えば、0x66 命令の場合、**Length**=**Data** のバイト長+2。
- 例えば、予想入力バイト長が 10 の場合、**Length**=10 + 2=12 になります。
- **Data**:
 - 定義: 送信される有効データ。
 - 幅: 8 ビットで 1 データ。
 - **Length**: 送信データの実際の長さは **Length-2** です。**Data_0** は最初

の有効データ、Data_1 は 2 番目の有効データ、Data_N は最後の有効データを示します。

● Check_sum :

- 定義：チェックサム
- 幅：16 ビット。上位 8 ビット (Check_sum_H) は下位 8 ビット (Check_sum_L) より先に送信されます。
- 計算式：Check_sum = CMD+ Length_L + Length_H + Data_0+... + Data_N。

注記：

Check_sum がオーバーフローする場合、下位 16 ビットだけが留保され、残りは破棄されます。例えば、計算結果が Check_sum=0x10F11 の場合、Check_sum_H=0x0F、Check_sum_L=0x11 となり、残りは破棄されます。

Status(0x44)

ステータスレジスタ読み出し命令。読み出し専用。読み出されるデータはステータスレジスタの値です(表 3-2)。

表 3-2 ステータスレジスタ

ビット	デフォルト値	説明	備考
Bit[7:3]	0	Reserved	-
Bit [2]	0	1 : Flash インターフェース動作中 0 : Flash インターフェースがアイドル	
Bit [1]	0	1 : チェックサムエラー 0 : チェックサムが正しい	
Bit [0]	0	1 : I ² C ステータスエラー 0 : I ² C ステータス正常	-

ステータスレジスタ読み出し命令の例を図 3-4 に示します。

図 3-4 ステータスレジスタ読み出し命令の例

S	Address	W	A	0x44	Sr	Address	R	A	Data	N	P
---	---------	---	---	------	----	---------	---	---	------	---	---

注記：

以下の同様の図は、特に断りのない限り、この例と同じ意味です。

- 図中の「白いシェーディング」が「I2C マスター」から「I2C スレーブ」への送信、「灰色のシェーディング」が「I2C スレーブ」から「I2C マスター」への送信です。
- 図の中で使われる名前は、以下のように定義されています。

- S : I²C バスの "開始(START)"。
- Address : I²C バスの"スレーブアドレス (Slave Address)"。
- W : I²C バスの"書き込み(WRITE)"。
- A : I²C バスの"応答(ACK)"。
- Sr : I²C バスの"再度開始(Repeated START)"。
- R : I²C バスの"読み出し(READ)"。
- Data : I²C バスで転送される有効データ。
- N : I²C バスの"否定応答(NACK)"。
- P : I²C バスの"停止(STOP)"。
- Length_L : Length の下位 8 ビット。
- Length_H : Length の上位 8 ビット。
- Data_0 : 最初の有効データ。
- Data_N : 最後の有効データ。
- Check_sum_H : Check_sum の上位 8 ビット。
- Check_sum_L : Check_sum の下位 8 ビット。

Reset(0x55)

IP リセット命令。IP の内部状態をリセットし、内部 FIFO をクリアします。

注記：

この処理は IP にのみ関係し、バックグラウンドアップグレードのプロセスには関係しません。

図 3-5IP リセット命令の例

S	Address	W	A	0x55	A	P
---	---------	---	---	------	---	---

Send(0x66,data back)

データを送信して Flash からデータをリードバックする命令。I²C インターフェースは、リードバックされるデータのバイト数 (Length) と同時にデータを送信し、チェックサムが正しければ、IP は Flash インターフェースに変換されてデータを FIFO にリードバックします (図 3-6 参照)。このとき、ユーザー側でこのデータが必要であれば、I²C インターフェースを介して読み出し命令を送信して FIFO 内のデータを読み出すことができます (図 3-7 参照)。

簡単に説明すると、リードバック操作には 2 つのステップがあります。

1. 読み出されるデータを IP の FIFO に読み込みます。
2. FIFO のデータを I²C バス経由で読み出します。

注記：

- ステップ 1 とステップ 2 の間に、例えばステータスを読み出す命令などを挿入することができます。
- 図 3-6 では、Data_0~Data_N は、どのような値でも構いませんが、0x00 または 0xFF を書き込むことが推奨されます。
- 図 3-7 では、Data_0~Data_N は実際にリードバックされた値です。

図 3-6 リードバック命令の例

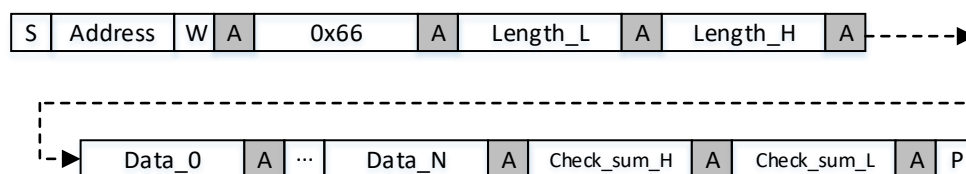
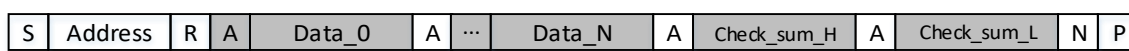


図 3-7FIFO データ読み出し命令の例



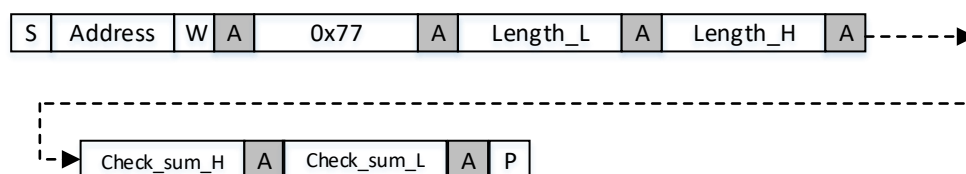
Waiting(0x77)

待機命令。I²C インターフェースは、待機バイト数 (Length) を送信し、チェックサムが正しければ、「待機バイト数」のカウントが完了するまで、Flash インターフェースはアイドルクロック (データは転送されない) を送信し続けることになります。

注記：

2C シリーズでは、Flash インターフェースの周波数は 2.5MHz で、すなわち 1 サイクルは 400ns です。1 バイトの場合は $8 \times 400 \text{ ns} = 3200 \text{ ns} = 3.2 \mu\text{s}$ 。例えば、150ms 待機する場合、待機バイト数は $46875 = 0xB71B$ となります。入力バイト数が $0xB71B + 2 = 0xB71D$ の場合、Length_L=0x1D、Length_H=0xB7 ということが分かります。

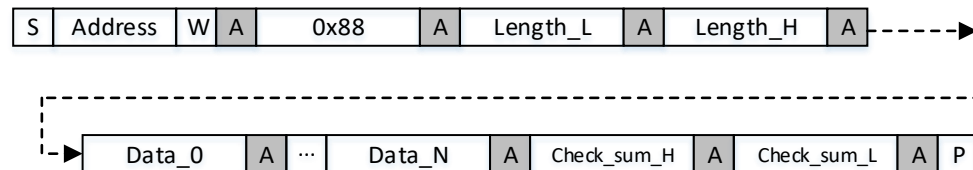
図 3-8 待機命令の例



Send(0x88, No data back)

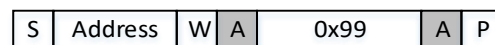
データを送信して Flash からデータをリードバックしない命令。I2C インターフェースは、まず送信されるデータを FIFO に格納し、データカウントが終了してチェックサムが正しい場合、FIFO 内のデータは Flash インターフェース経由で一括して送信されます。

図 3-9 データを送信して Flash からデータをリードバックしない命令の例

**Reconfig(0x99)**

Reconfig 命令。I2C インターフェースは、この命令を送信することにより、"Reconfig_N"ピンを Low にプルダウンすることを可能にします。

図 3-10 Reconfig 命令の例

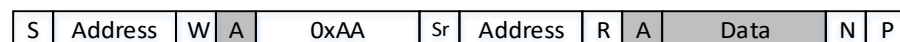
**IP Version(0xAA)**

IP バージョン命令。I2C インターフェースはこの命令を送信して固定値の Data を読み出します。

注記：

現在のバージョンでは、Version_Data=0x20 です。

図 3-11 IP バージョン命令



3.3.2 9X シリーズ

9X シリーズと 2C シリーズのデバイスには、明確な違いがあります。従って、ご使用の際は特にご注意ください。9X シリーズは、主にレジスタで通信を行います。

レジスタの定義

関連レジスタの定義を表 3-3 レジスタの定義に示します。

表 3-3 レジスタの定義

番号	レジスタ名	レジスタアドレス (ADDR)	アクセスタイプ	説明	フォーマット	備考
1	CSCR	0x00	書き込み	$\overline{CS}$ 信号制御レジスタ	ADDR+Data	-
2	DICN	0x01	書き込み/読み出し	書き込み操作バイト数レジスタ	ADDR+Data	
3	DOCN	0x02	書き込み/読み出し	読み出し操作バイト数レジスタ	ADDR+Data	
4	DIFR	0x03	書き込み	書き込みFIFOレジスタ	ADDR+Data	
5	SRSTR	0x08	書き込み	ソフトリセットレジスタ	ADDR+Data	
6	SR	0x80	読み出し	ステータスレジスタ	ADDR+Data	
7	DOFR	0x81	読み出し	読み出しFIFOレジスタ	ADDR+Data	

フォーマットは、レジスタアドレス (ADDR) + データ (Data) です。レジスタアドレスにより、データの長さ (バイト数) が異なります。

以下では、各レジスタのデータを「レジスタ名-Data」で区別します。

例えば、CSCR-Data は、 $\overline{CS}$ 信号制御レジスタの Data を示します。

CSCR(0x00)

$\overline{CS}$ 信号制御レジスタのフォーマットは ADDR+ CSCR-Data (1 バイト)

です。CSCR-Data の長さは 1 (バイト) であり、CSCR-Data の定義は次のとおりです。

表 3-4 CSCR-Data の定義

ビット	デフォルト値	アクセスタイプ	説明	備考
Bit[7:0]	0x00	書き込み	0xFF : CS 信号を Low にします (Flash インターフェースが動作する) 0x00: CS 信号を High にします (Flash インターフェースがアイドル状態になる) その他 : Reserved	-

$\overline{CS}$ 信号制御レジスタの使用例は図 3-12 に示すとおりです。この例では、CSCR-Data=0xFF で、CS 信号は Low にされます。

図 3-12 CSCR レジスタの使用例

S	Address	W	A	0x00	A	Data	A	P
---	---------	---	---	------	---	------	---	---

DICN(0x01)

書き込み操作バイト数レジスタのフォーマットは ADDR+ DICN-Data (2 バイト)です。ここで、DICN-Data の長さは 2 バイトです。値の有効範囲は 1~300 で、0x0001 は 1 バイト、0x00FF は 255 バイトを示し、その他は同様です。

表 3-5 DICN-Data の定義

ビット	デフォルト値	アクセスタイプ	説明	備考
Bit[15:0]	0x0000	書き込み/読み出し	1~300 : 書き込み操作バイト数 その他 : Reserved	-

2 バイトの DICN-Data を送信する場合、下位バイト (Bit[7:0]) は上位バイト (Bit[15:8]) より先に送信されます。従って、DICN-Data=0x0001 の場合、まず 0x01 が送信され、次に 0x00 が送信されます。

DICN レジスタ書き込みの使用例を図 3-13 に示します。DICN-Data = 0x0001 の場合、Data_0 = 0x01, Data_1 = 0x00 となります。

図 3-13 DICN レジスタ書き込みの使用例

S	Address	W	A	0x01	A	Data_0	A	Data_1	A	P
---	---------	---	---	------	---	--------	---	--------	---	---

DICN レジスタ読み出しの使用例を図 3-14 に示します。DICN-Data = 0x0001 の場合、Data_0 = 0x01, Data_1 = 0x00 となります。

図 3-14 DICN レジスタ読み出しの使用例

S	Address	W	A	0x01	Sr	Address	R	A	Data_0	A	Data_1	N	P
---	---------	---	---	------	----	---------	---	---	--------	---	--------	---	---

DOCN(0x02)

読み出し操作バイト数レジスタのフォーマットは ADDR+ DOCN-Data (2 バイト)です。ここで、DOCN-Data の長さは 2 バイトです。値の有効範囲は 1~300 で、0x0001 は 1 バイト、0x00FF は 255 バイトを示し、その他は同様です。

表 3-6 DOCN-Data の定義

ビット	デフォルト値	アクセスタイプ	説明	備考
Bit[15:0]	0x0000	書き込み/読み出し	1~300 : 書き込み操作バイト数 その他 : Reserved	-

2 バイトの DOCN-Data を送信する場合、下位バイト (Bit[7:0]) は上位バイト (Bit[15:8]) より先に送信されます。従って、DOCN-Data=0x0001 の場合、まず 0x01 が送信され、次に 0x00 が送信されます。

DOCN レジスタ書き込みの使用例を図 3-15 に示します。DOCN-Data = 0x0001 の場合、Data_0 = 0x01, Data_1 = 0x00 となります。

図 3-15 DOCN レジスタ書き込みの使用例

S	Address	W	A	0x02	A	Data_0	A	Data_1	A	P
---	---------	---	---	------	---	--------	---	--------	---	---

DOCN レジスタ読み出しの使用例を図 3-16 に示します。DOCN-Data = 0x0001 の場合、Data_0 = 0x01, Data_1 = 0x00 となります。

図 3-16 DOCN レジスタ読み出しの使用例

S	Address	W	A	0x02	Sr	Address	R	A	Data_0	A	Data_1	N	P
---	---------	---	---	------	----	---------	---	---	--------	---	--------	---	---

DIFR(0x03)

書き込み FIFO レジスタのフォーマットは ADDR+ DIFR-Data(1~300 バイト)です。DIFR-Data の長さの範囲は 1~300 バイトです。1 バイトずつ送信され、実際に送信されるデータのバイト数は、書き込み操作バイト

数レジスタの値と一致させる必要があります。

表 3-7 DIFR-Data の定義

ビット	デフォルト値	アクセスタイプ	説明	備考
Bit[7:0]	-	書き込み	0x00~0xFF	-

例えば、DICN-Data=0x0011 の場合、DIFR は 0x0011=17 バイトのデータを送信することになります。

DIFR レジスタの使用例を以下に示します。Data_0 は最初に書き込まれたデータ、Data_N は最後に書き込まれたデータを示します。ここで、書き込まれたバイト数は、DICN-Data の値と同じである必要があります。

図 3-17 DIFR レジスタの使用例

S	Address	W	A	0x03	A	Data_0	A	...	Data_N	A	P
---	---------	---	---	------	---	--------	---	-----	--------	---	---

SRSTR(0x08)

ソフトリセットレジスタのフォーマットは ADDR+ SRSTR-Data(1 バイト)です。ここで、SRSTR-Data の長さは 1 バイトであり、SRSTR-Data = 0xFF のときのみリセットが有効です。

表 3-8 SRSTR-Data の定義

ビット	デフォルト値	アクセスタイプ	説明	備考
Bit[7:0]	0x00	書き込み	0xFF : ソフトリセット有効 その他 : Reserved	-

ソフトリセットレジスタの使用例は図 3-18 に示すとおりです。この例では、SRSTR-Data=0xFF です。

図 3-18 SRSTR レジスタの使用例

S	Address	W	A	0x08	A	Data	A	P
---	---------	---	---	------	---	------	---	---

SR(0x80)

ステータスレジスタのフォーマットは ADDR+ SR-Data(3 バイト)です。ここで、SR-Data の長さの範囲は 3 バイトです。

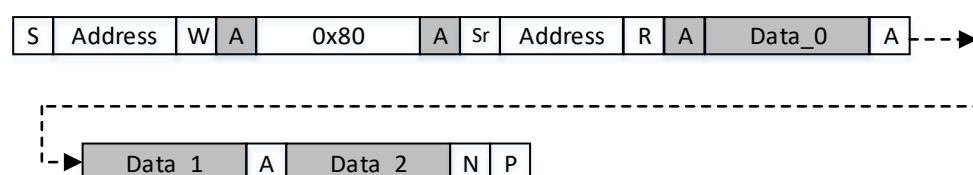
表 3-9 SR-Data の定義

ビット	名称	アクセス タイプ	説明	備考
Bit[23:15]	TXLVL	読み出し	TX FIFOの有効データ数。値の範囲：1 ~ 300。	-
Bit [14:6]	RXLVL	読み出し	RX FIFOの有効データ数。値の範囲：1 ~ 300。	
Bit [5:4]	FIFOの状態	読み出し	Bit [5] : 1 : TXLVLにデータあり 0 : TXLVLにデータなし Bit [4] : 1 : RXLVLにデータあり 0 : RXLVLにデータなし	
Bit [3:0]	Flashインターフェースの状態	読み出し	Bit [3:2] : Reserved Bit [1:0] : 11 : Reserved 10 : Flashインターフェース動作完了 01 : Flashインターフェース動作中 00 : Flashインターフェースがアイドル	-

3 バイトの SR-Data を送信する場合、まず下位バイト(Bit[7:0])、2 バイト目 (Bit[15:8])、上位バイト(Bit[23:15])の順に送信します。従って、SR-Data = 0x000102 の場合、最初に 0x02 が送信され、次に 0x01 が送信され、そして 0x00 が送信されます。

ステータスレジスタの使用例を図 3-19 に示します。SR-Data=0x000102 の場合、Data_0 = 0x02, Data_1 = 0x01, Data_2= 0x00 となります。

図 3-19 SR レジスタの使用例



DOFR(0x81)

読み出し FIFO レジスタのフォーマットは ADDR+ DOFR-Data(1~300 バイト)です。DOFR-Data の長さの範囲は 1~300 バイトです。1 バイトずつ読み出され、実際に読み出されるデータのバイト数は、読み出し操作バイト数レジスタの値と一致させる必要があります。

表 3-10 DOFR-Data の定義

ビット	デフォルト値	アクセス タイプ	説明	備考
Bit[7:0]	-	読み出し	0x00~0xFF	-

DOFR レジスタの使用例を以下に示します。Data_0 は最初に読み出されたデータ、Data_N は最後に読み出されたデータを示します。ここで、読み出されたバイト数は、DOCN-Data の値と同じである必要があります。

図 3-20 DOFR レジスタの使用例

S	Address	W	A	0x81	A	Data_0	A	...	Data_N	N	P
---	---------	---	---	------	---	--------	---	-----	--------	---	---

4 信号の定義

4.1 信号の定義

Gowin goConfig I2C IP は、2C シリーズと 9X シリーズで IO インターフェース定義に違いがあるため、以下は、2C シリーズと 9X シリーズ別で個別に説明します。2C シリーズの IO は表 4-1 に示すとおりです。

表 4-1 2C シリーズの I2C インターフェースの定義

番号	信号名	方向	説明	IO割り当て 制約	備考
1	GW_BACKGROUN D_SCL_INR	入力	I ² Cバスクロック	名称制限	すべての 信号の入 出力方向 は、IPを 基準とし ていま す。
2	GW_BACKGROUN D_SDA_INR	双方向	I ² Cバスデータ	名称制限	
3	GW_BACKGROUN D_SEL_INR	入力	バックグラウン ドアップグレー ドのイネーブル 信号(入力) 1 : イネーブル (デフォルト) 0 : ディセーブ ル	-	
4	GW_OSC_CLK	出力	25MHz出力ク ロック	-	
5	GW_BACKGROUN D_CLK	出力	IP-Flashインタ ーフェースクロ ック	自動	
6	GW_BACKGROUN	出力	IP-Flashインタ	自動	

番号	信号名	方向	説明	IO割り当て 制約	備考
	D_MODEL		ーフエースモード選択		
7	GW_BACKGROUND_DATA_OUT	出力	IP-Flashインターフェースデータ出力	自動	
8	GW_BACKGROUND_DATA_IN	入力	IP-Flashインターフェースデータ入力	自動	
9	GW_BACKGROUND_DATA_IN_EN	入力	IP-Flashインターフェース入力データ有効信号	自動	
10	GW_BACKGROUND_SEL_EXR	出力	バックグラウンドアップグレードのイネーブル信号(出力)	自動	
11	GW_BACKGROUND_RECONFIG_N	出力	RECONFIG_N 信号	自動	

注記：

- IO 割り当て制約はそれぞれ以下のように定義されます。
 - "-"：制約なし。自由に名前を付けられ、自由に位置を制約できます。
 - "名称制限"：名称の制限。自由に名前を付けられないが、自由に位置を制約できます。つまり、トップレベル IO の名前は、"信号名"と"方向"を含め、IP での定義と全く同じである必要があります。
 - "自動"：自動的な位置制約。自由に名前を付けることができず、位置を制約することもできません。すなわち、トップレベル IO に、特定の IO 名を付ける必要があります。さらに、手動で位置を制約することができず、Gowin ソフトウェアが自動的に位置を制約します。
- "GW_BACKGROUND_SEL_INR"：イネーブルを変更する必要がある場合、この入力を "1"(High)のままにしなければなりません。
- "2C シリーズ" の場合、"特定の IO 名" はつまり IP の IO 名です。つまり、トップレベル IO の名前は、IP での定義と全く同じである必要があります。
- ここの IO 割り当て制約の定義は、"9X シリーズ"の IO 割り当て制約にも適用されます。

9X シリーズの IO は表 4-2 に示すとおりです。

表 4-2 9X シリーズの I2C インターフェースの定義

番号	信号名	方向	説明	IOの割り当て ^[1]	備考
1	GW_BACKGROUND_SCL_INR	入力	I ² Cバスクロック	-	すべての信号の入出力方向は、IPを基準としています。
2	GW_BACKGROUND_SDA_INR	双方向	I ² Cバスデータ	-	
3	GW_OSC_CLK	出力	20MHz出力クロック	-	
4	GW_BACKGROUND_SCLK_INR	出力	IP-Flashインターフェースクロック	自動 ^[2]	
5	GW_BACKGROUND_CS_INR	出力	IP-FlashインターフェースのCS出力	自動 ^[2]	
6	GW_BACKGROUND_MOSI_INR	出力	IP-Flashインターフェースデータ出力	自動 ^[2]	
7	GW_BACKGROUND_MISO_INR	入力	IP-Flashインターフェースデータ入力	自動 ^[2]	

注記：

[1] IO 割り当て制約の定義については、“2C シリーズ” の対応説明を参照してください。

[2] トップレベルで使用する場合、一部の IO インターフェース名を 9X シリーズの"特定 IO 名" (Flash インターフェース専用名) に変更する必要があり、Flash インターフェース専用名の定義および対応関係は次のとおりです。

IP IO 名 > Flash インターフェース専用名 (IO 方向)

GW_BACKGROUND_SCLK_INR > O_flash_ck(Output)

GW_BACKGROUND_CS_INR > O_flash_cs_n(Output)

GW_BACKGROUND_MOSI_INR > IO_flash_do(InOut)

GW_BACKGROUND_MISO_INR > IO_flash_di(Output)

5_{IP} の使用法

注記：

この章では、IP に関する操作のみが対象として説明されています。

Gowin goConfig I2C IP は、2C シリーズと 9X シリーズで使用する際に違いがあります。本章ではシリーズ別に説明し、それぞれ Gowin ソフトウェアでの構成、推奨される IP 手順、および注意事項を説明します。

5.1 2C シリーズ

5.1.1 ソフトウェアでの構成

バックグラウンドアップグレード IP を持つプロジェクトを合成 (Synthesize) する前に、Gowin goConfig I2C IP が正常に動作できるように、Gowin ソフトウェアでの構成を変更する必要があります。

Gowin ソフトウェアで、2C シリーズのデバイスを選択した後、"Project > Configuration > BitStream > Background Programming > I2C "をクリックします(図 5-1)。

図 5-1 Configuration ウィンドウを開く

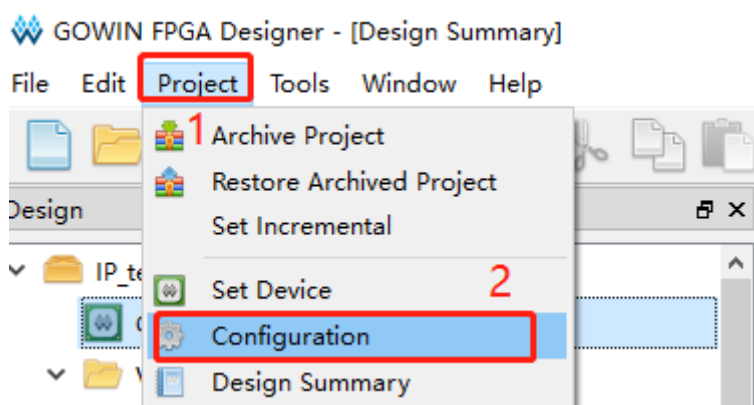
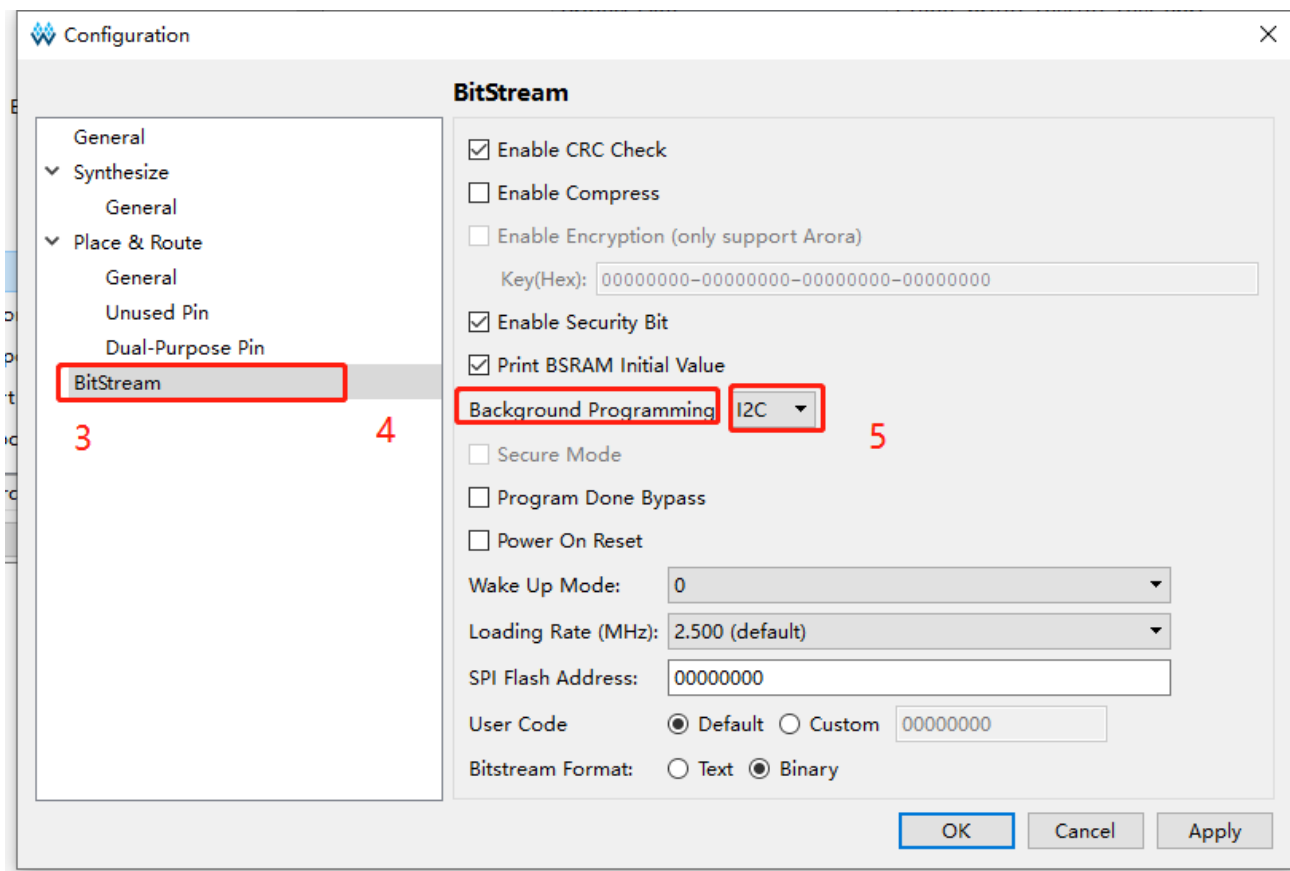


図 5-2 バックグラウンドアップグレードのインターフェースとして I2C を選択



5.1.2 手順の説明

Gowin goConfig I2C IP(2C シリーズ)の命令には、読み出し/書き込み命令と操作命令があります。

- 読み出し/書き込み命令：CMD および Length を送信する必要があります。例えば、"Send(data back)"、"Waiting"、"Send(No data back)"などの命令です。
- 操作命令：CMD を送信するだけで、Length を送信する必要がありません。例えば、"Reset"や "Reconfig"などの命令です。

2C シリーズには、以下の操作があります。

1. 事前準備。
2. 書き込み。
3. 読み出し。
4. RECONFIG_N のトリガ。

注記：

読み出し/書き込み命令が完了した後、IP は直接 Flash インターフェースを操作すること

になります。操作命令は、主に IP の状態を制御/確認するために使用されます。

事前準備

IP が正常に動作することを検証/確保するための事前準備は、以下のよう
に実行されます

1. リセット命令 (0x55) を送信して IP を初期状態にします。
2. IP Version 命令 (0xAA) を送信し、読み出されたバージョン値
(Version_Data=0x20) を確認し、期待通りの IP バージョンであるこ
とを確認します。
3. Status 命令 (0x44) を送信し、読み出されたステータス値が 0x00 で
あることを確認します。

操作完了後の読み出されたデータが期待通りのものであれば、IP との
通信を開始することができます。

書き込み命令の手順

"事前準備"が正しいと判断されれば、書き込み命令の手順を開始する
ことができます。書き込み命令の手順は、一般的に次のように行われま
す。

1. "Send(No data back)"(0x88)命令を送信し、データを書き込みます。
2. "Status" (0x44) 命令を送信し、IP 動作が完了して他の操作が可能で
あることを確認します。

操作完了後の読み出されたステータスデータが期待通りのものであれ
ば、IP との他の通信を開始することができます。

リードバック命令の手順

"事前準備"が正しいと判断されれば、リードバック命令の手順を開始
することができます。リードバック命令の手順は、一般的に次のように行
われます。

1. "Send(data back)"(0x66)命令を送信し、読み出されるデータの長さを
書き込みます。
2. "Status" (0x44) 命令を送信し、Flash 側の動作が完了したことを確認
します。
3. I2C インターフェースを介してデータを読み出します
(Send(0x66,data back)参照)。
4. "Status" (0x44) 命令を送信し、IP 動作が完了して他の操作が可能で
あることを確認します。

操作完了後の読み出されたステータスデータが期待通りのものであれ
ば、IP との他の通信を開始することができます。

RECONFIG_N のトリガ

RECONFIG_N をトリガします。その機能は次のとおりです：バックグラウンドアップグレードの第 1 ステップ（Flash への書き込み）が完了した後、"RECONFIG_N" をトリガして第 2 ステップ（Flash の内容を SRAM に読み込む）を実行することで、バックグラウンドアップグレードを完了させます。

5.1.3 注意事項

Gowin goConfig I2C IP を使用する際の注意事項は次のとおりです。

IO の制約

"信号の定義"で説明したように、Gowin goConfig I2C IP(2C シリーズ)の IO 信号は、制約なしの IO と制約ありの IO (IO 名の限定と自動を含む) の 2 つに分類できます。

- 制約なしの IO : 通常の利用のように、その名称は自由に構成でき、その位置も自由に制約できます。
- 制限ありの IO : IO 名が限定され、トップレベルに置く必要があります。

注記 :

Gowin ソフトウェアはこの"制限ありの IO"を検出します。Background Programming > I2C が選択されているが、IO が存在しない、または該当しない場合、“Place & Route”の実行時にエラーが報告されます。

OSC の出力

Gowin goConfig I2C IP は、デバイス内の OSC モジュールを使用します。また、本 IP は 25MHz のクロック出力を提供し、ご希望により使用することができます。

バッファの制限

Gowin goConfig I2C IP は、データバッファリングに 1 個の BSRAM を使用するので、一度に転送されるデータの総量は 1 個の BSRAM を超えてはいけません。

5.2 9X シリーズ

5.2.1 ソフトウェアでの構成

バックグラウンドアップグレード IP を使用したプロジェクトを合成 (Synthesize) する前に、Gowin goConfig I2C IP が正常に動作するように、Gowin ソフトウェアでの構成を変更する必要があります。

Gowin ソフトウェアで、9X シリーズのデバイスを選択した後、“Project > Configuration > BitStream > Background Programming > Internal”をクリックします。

図 5-3 Configuration ウィンドウを開く

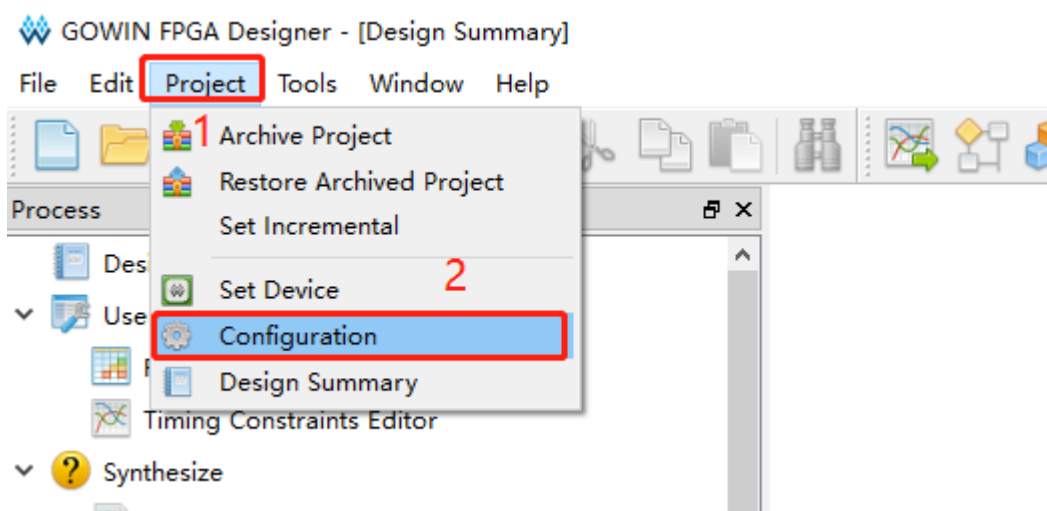
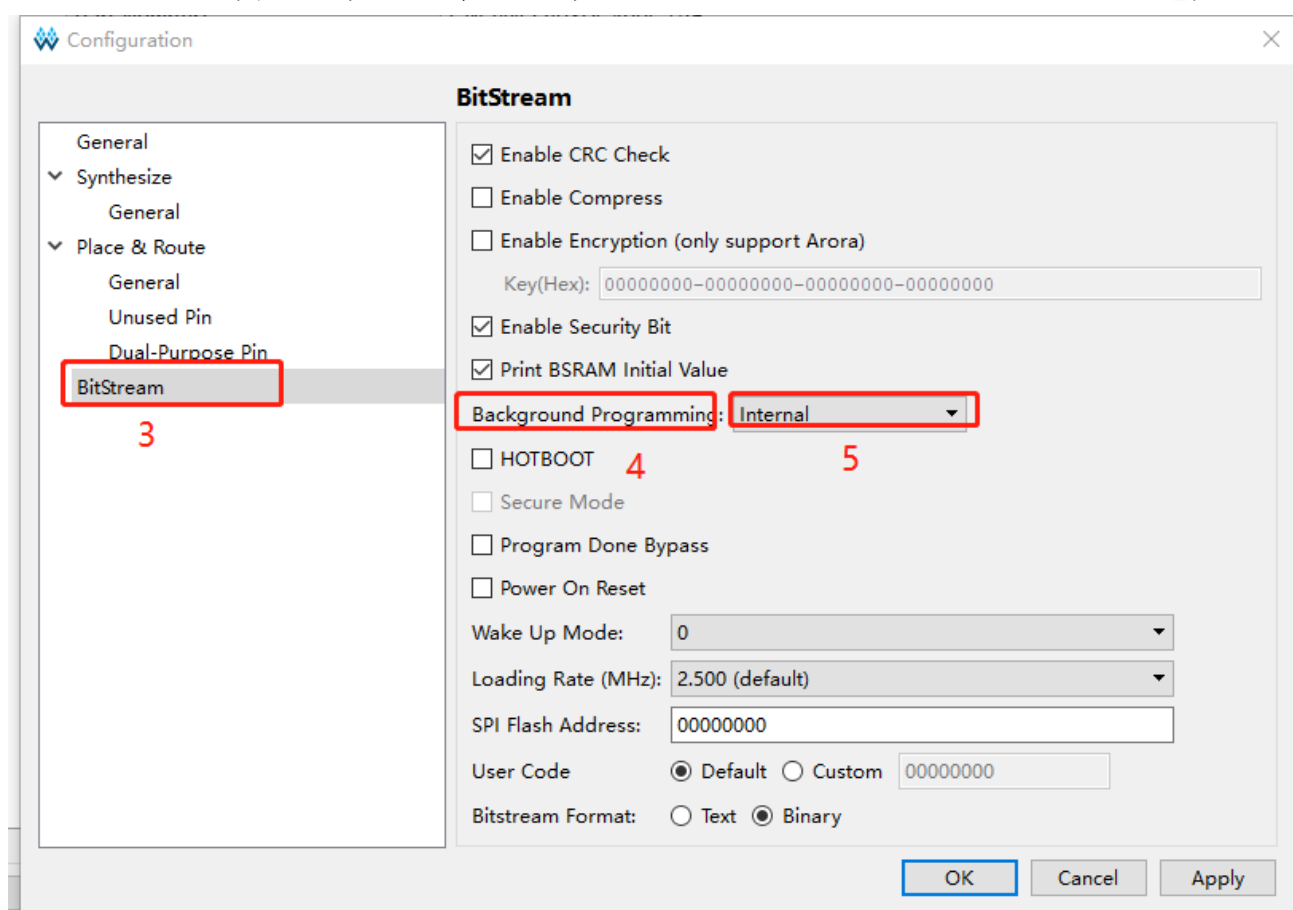


図 5-4 バックグラウンドアップグレードのインターフェースとして I2C を選択



5.2.2 手順の説明

Gowin goConfig I2C IP(9X シリーズ)は、レジスタを操作することにより Flash の読み出し/書き込みを行います。その手順は次のとおりです。

書き込み手順

Gowin goConfig I2C IP(9X シリーズ)の動作際、まず一部のレジスタを構成する必要があります。書き込み手順の例は以下のとおりです。

1. SR ステータスレジスタ (0x80) に問い合わせ、SPI ポートがアイドルの場合 (SR-Data[1:0]=00)、操作を開始することができます。
2. 送信するバイト数を DICN(0x01)に書き込みます。
3. 書き込まれるデータを DIFR (0x03) レジスタに順次書き込みます。
4. CSCR (0x00)レジスタの値を 0xFF に設定し、操作を開始します (Flash インターフェースが操作を開始します)。
5. SR (0x80)ステータスレジスタに問い合わせで操作の状態を読み出します。
6. Flash インターフェースの操作が完了した後(SR-Data[1:0]=10)、CSCR (0x00)レジスタの値を 0x00 に設定すれば、Flash インターフェースはアイドル状態に入り、この操作は終了します。

読み出し手順

Gowin goConfig I2C IP(9X シリーズ)の動作際、まず一部のレジスタを構成する必要があります。読み出し手順の例は以下のとおりです。

1. SR ステータスレジスタ (0x80) に問い合わせ、SPI ポートがアイドルの場合 (SR-Data[1:0]=00)、操作を開始することができます。
2. 送信するバイト数を DICN(0x01)に書き込みます。
3. 受信するバイト数を DOCN(0x02)に書き込みます。
4. 書き込まれるデータを DIFR (0x03) レジスタに順次書き込みます。
5. CSCR (0x00)レジスタの値を 0xFF に設定し、操作を開始します。
6. SR (0x80)ステータスレジスタに問い合わせで操作の状態を読み出します。
7. Flash インターフェースの操作が完了した後(SR-Data[1:0]=10)、DOFR (0x81)レジスタを読み出すことによってデータを読み出します。
8. CSCR (0x00)レジスタの値を 0x00 に設定すれば、Flash インターフェースはアイドル状態に入り、この操作は終了します。

5.2.3 注意事項

Gowin goConfig I2C IP(9X シリーズ)を使用する際の注意事項は次のとおりです。

IO 制約

表 4-2 で説明したように、Gowin goConfig I2C IP(9X シリーズ)の IO 信号は、制約なしの IO と制約ありの IO（自動）の 2 つに分類できます。

- 制約なしの IO：通常の利用のように、その名称は自由に構成でき、その位置も自由に制約できます。
- 制限ありの IO：IO 名が限定され、トップレベルに置く必要があります。

トップレベルで"制限ありの IO"を使用する場合、9X シリーズの IO インターフェース名を"特定 IO 名"（Flash インターフェース専用名）に変更する必要があります。Flash インターフェース専用名の定義および対応関係は表 5-1 に示すとおりです。

表 5-1 対応関係

番号	IPインターフェース名	Flashインターフェース		説明	備考
		専用名	方向		
1	GW_BACKGROUND_SCLK_INR	O_flash_ck	出力	IP-Flashインターフェースクロック	トップレベルで使用する場合、インターフェースの方向は、Flashインターフェースを基準とする必要があります。
2	GW_BACKGROUND_CS_INR	O_flash_cs_n	出力	IP-FlashインターフェースのCS出力	
3	GW_BACKGROUND_MOSI_INR	IO_flash_do	双方向	IP-Flashインターフェースデータ出力	
4	GW_BACKGROUND_MISO_INR	IO_flash_di	出力	IP-Flashインターフェースデータ入力	

トップモジュールで "制限ありの IO"を定義する場合、IO 方向は Flash インターフェースの方向と同じにする必要があります。IO の定義例を図 5-5 に示します。

図 5-5 IO の定義例

```

//
module goConfig_I2C_Top_wrap
(
    //i2c
    inout  GW_BACKGROUND_SDA_INR,
    input  GW_BACKGROUND_SCL_INR,
    //flash IO
    output O_flash_ck,      //
    output O_flash_cs_n,    //
    inout  IO_flash_do,     //
    output IO_flash_di,     //
    //Others
    output GW_OSC_CLK
);

goConfig_I2C_Top inst_goConfig_I2C(
    .GW_BACKGROUND_SDA_INR(GW_BACKGROUND_SDA_INR),
    .GW_BACKGROUND_SCL_INR(GW_BACKGROUND_SCL_INR),
    .GW_BACKGROUND_SCLK_INR(O_flash_ck),
    .GW_BACKGROUND_CS_INR(O_flash_cs_n),
    .GW_BACKGROUND_MISO_INR(IO_flash_do),
    .GW_BACKGROUND_MOSI_INR(IO_flash_di),

    .GW_OSC_CLK(GW_OSC_CLK)
);
endmodule

```

注記：

Gowin ソフトウェアはこの"制限ありの IO"を検出します。Background Programming > Internal が選択されているが、IO が存在しない、または該当しない場合、“Place & Route”後の結果が期待したものと異なることになります。

OSC 出力

Gowin goConfig I2C IP は、デバイス内の OSC モジュールを使用します。本 IP は 20MHz の出力クロックも提供し、ご希望により使用することができます。

バッファの制限

Gowin goConfig I2C IP は、データをバッファリングするために、受信データ用と送信データ用の 2 つの BSRAM を使用します。一度に読み出し/書き込みできるデータの範囲は 1～300 バイトです。

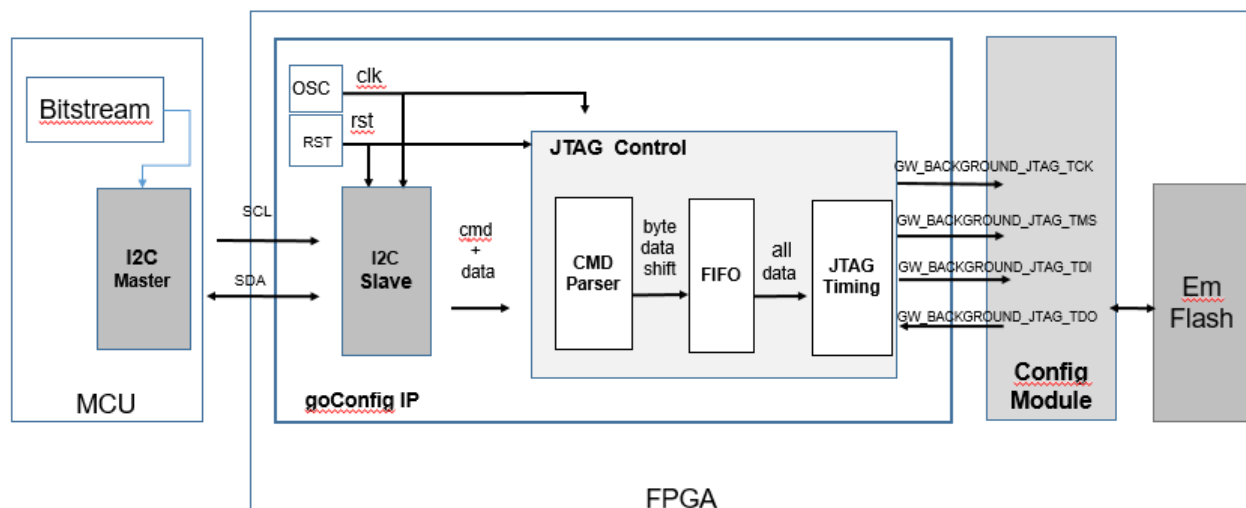
6 活用例の説明

より詳細な例については、『Gowin goConfigIP (I2C) サンプルコード ユーザーガイド(TN715)』を参照してください。サンプルコードを入手するには、当社の FAE にお問い合わせください。

6.1 2C シリーズでの活用例

この例のサンプルコードは MCU で実行され、そして I²C インターフェースを介して goConfigIP を使用してオンチップ Flash の消去とプログラムや、FPGA の ID code, user code, status code の読み出しなどを実現します。

図 6-1 構造



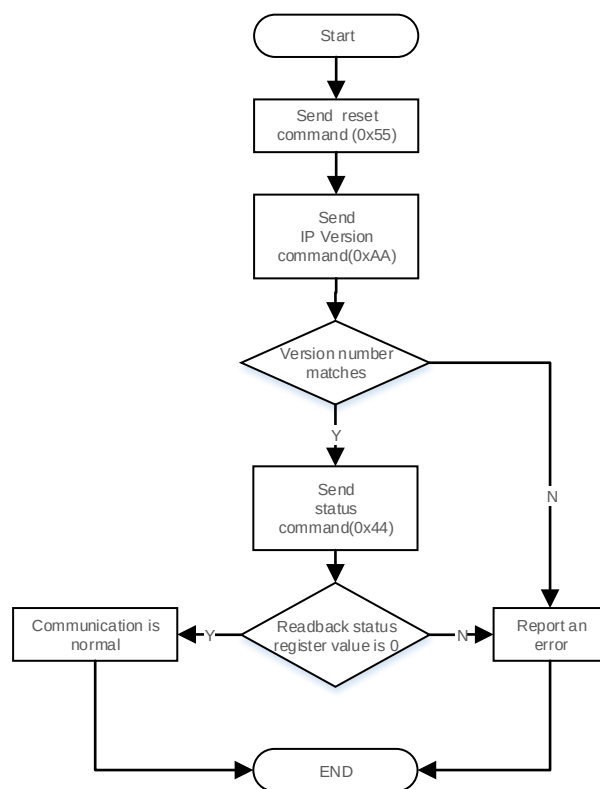
6.1.1 MCU と IP 間の通信が正常かどうかの確認

バックグラウンドアップグレードを開始する前に、まず MCU と IP 間の通信が正常であるかどうかを確認します。

1. Reset 命令 (0x55) を送信して IP を初期状態にします(API : gw_i2c_send_reset)。
2. IP Version 命令(0xAA)を送信して、読み出されたバージョン値 (例えば Version Data=0x20)が予期したバージョンと一致しているかを確認します (API:gw_i2c_read_version)。
3. Status 命令 (0x44) を送信し、読み出されたステータス値を確認します(API: gw_i2c_read_reg_status)。
4. 操作が完了し、読み出されたデータが期待どおりであれば、IP との通信を開始できます。

MCU と IP 間の通信が正常かどうかを確認するためのフローチャートを図 6-2 に示します：

図 6-2 MCU と IP 間の通信が正常かどうかを確認するためのフローチャート



6.1.2 バックグラウンドアップグレードの手順

バックグラウンドアップグレードの手順は次のとおりです。

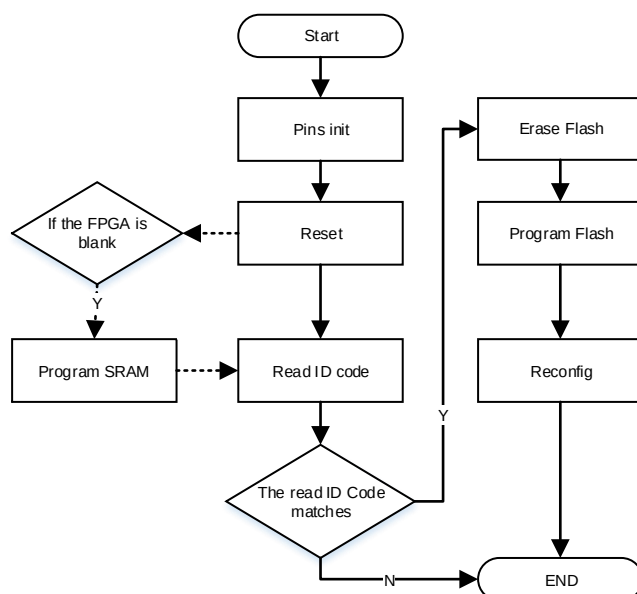
1. MCU プラットフォームのピンを初期化します (I2C、debug_uart)。
2. IP をリセットします (API:gw_i2c_send_reset)。
3. FPGA がブランクチップでない場合は、このステップをスキップしてください。FPGA がブランクチップの場合は、まず SRAM をコンフィ

ギュレーションします(API: i2c_configmode_program_sram)。

4. MCU(I2C)によって読み出された ID code がビットストリームファイルの ID code と一致するかどうかを比較します(API: mach_code)。一致する場合は次のステップへ移行し、一致しない場合はバックグラウンドアップグレードを終了します。
5. Flash を消去します(API: gw_i2c_erase_flash_tsmc)。
6. Flash をプログラムします(API: gw_i2c_prog_flash_tsmc)。
7. 再コンフィギュレーションします(gw_i2c_send_reconfig_n)。

バックグラウンドアップグレードのフローチャートを図 6-3 に示します。

図 6-3 バックグラウンドアップグレードのフローチャート



FPGA Code の読み出し

FPGA code を読み出すには、API: gw_i2c_read_code を使用します。MCU と IP 間の通信が正常であれば、I2C を使用して FPGA code を読み出すことができます。

- ID Code(即ち JEDEC ID Code)は、FPGA デバイスの基本的な ID です。
- user code は、使用する FPGA デバイスのユーザー識別子です。
- status code は、FPGA デバイスがビットストリームファイルのロードを完了した後のステータス識別子です。

例：

- ID code 読み出しの API は、gw_i2c_read_code(0x11)です。

- user code 読み出しの API は、`gw_i2c_read_code(0x13)`です。
- status code 読み出しの API は、`gw_i2c_read_code(0x41)`です。

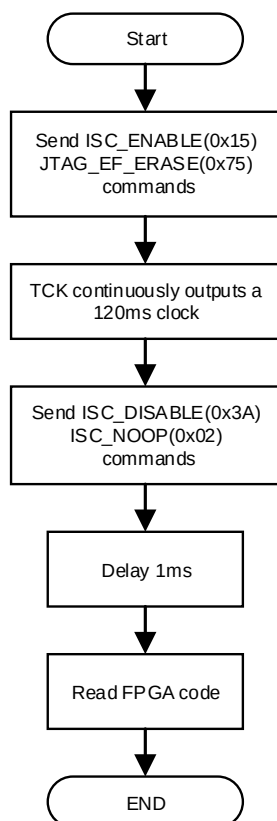
オンチップ Flash の消去

オンチップ Flash を消去するための API は、`gw_i2c_erase_flash_tsmc` です。オンチップ Flash を消去する手順は次のとおりです。

1. `ISC_ENABLE(0x15)`と `JTAG_EF_ERASE(0x75)`を送信します。この 2 つの命令は API (`write_inst_and_data`)によってパッケージ化され、MCU の I2C インターフェース(API: `i2c_write_buffer`) を介して FPGA に送信されます。
2. TCK でクロックを 120ms 継続的に出力します(API: `gw_i2c_send_runtest_xus`)。
3. `ISC_DISABLE(0x3A)`と `ISC_NOOP(0x02)`を送信します。この 2 つの命令は API (`write_inst_and_data`)によってパッケージ化され、MCU の I2C インターフェース(API: `i2c_write_buffer`) を介して FPGA に送信されます。
4. 1ms 遅延します(API: `gw_i2c_send_runtest_xus`)。
5. FPGA code を読み出します(API : `gw_i2c_read_code`)。

オンチップ Flash を消去するためのフローチャートを図 6-4 に示します。

図 6-4 オンチップ Flash の消去のフローチャート



オンチップ Flash のプログラミング

オンチップ Flash をプログラムするための API は、`gw_i2c_prog_flash_tsmc` です。オンチップ Flash をプログラムする手順は次のとおりです。

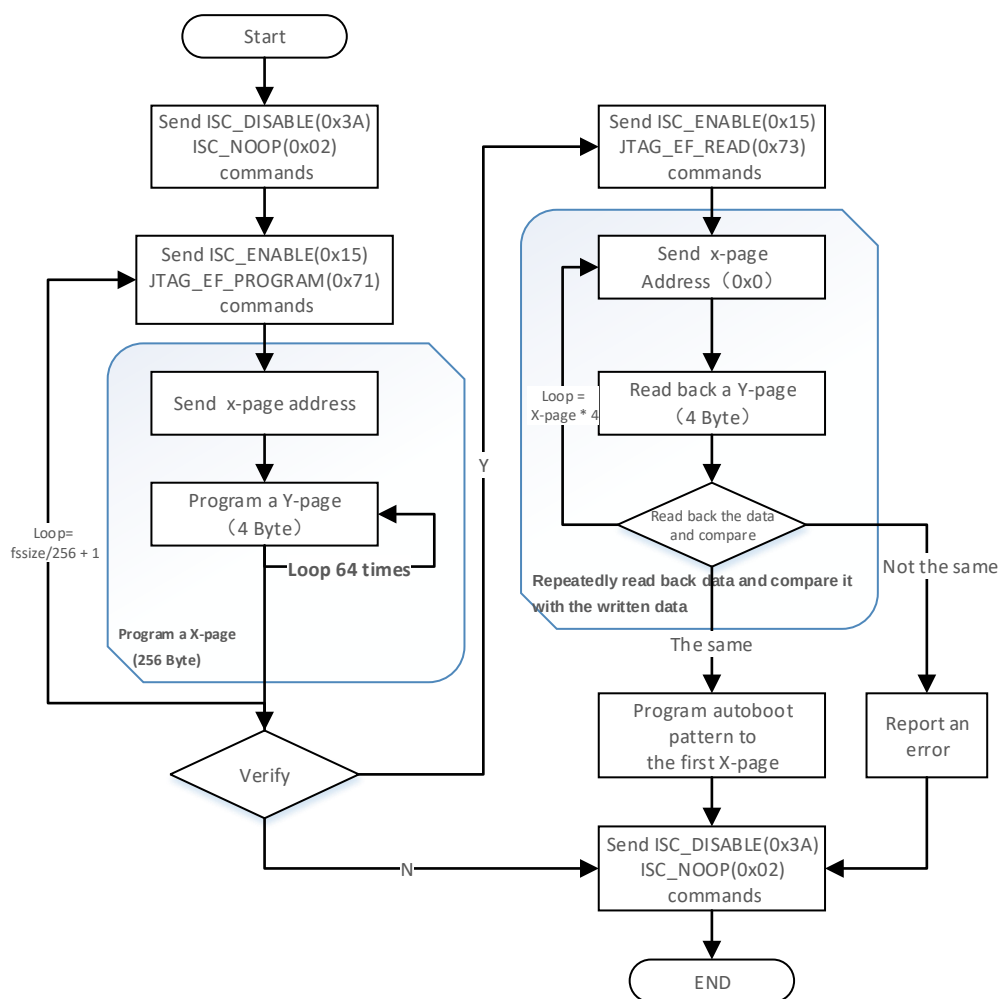
1. `ISC_DISABLE(0x3A)`と `ISC_NOOP(0x02)`を送信します。この 2 つの命令は API (`write_inst_and_data`)によってパッケージ化され、MCU の I2C インターフェース(API: `i2c_write_buffer`) を介して FPGA に送信されます。
2. `ISC_ENABLE(0x15)`と `JTAG_EF_PROGRAM(0x71)`を送信します。この 2 つの命令は API (`write_inst_and_data`)によってパッケージ化され、MCU の I2C インターフェース(API: `i2c_write_buffer`) を介して FPGA に送信されます。
3. X-pgae のアドレスを送信します。このアドレスは、API (`write_one_x_address`)によってパッケージ化され、MCU の I2C インターフェース(API : `i2c_write_buffer`) を介して FPGA に送信されます。
4. 1 つの Y-page をプログラムします。1 つの X-page には 64 個の Y-page が含まれています。この Y-page のデータは、API (`write_one_y_page`)によってパッケージ化され、MCU の I2C インター

フェース(API : i2c_write_buffer) を介して FPGA に送信されます。

5. リードバックするには、ISC_ENABLE(0x15)と JTAG_EF_READ(0x73)を送信する必要があります。これら 2 つの命令は API (write_inst_and_data)によってパッケージ化され、MCU の I2C インターフェース(API: i2c_write_buffer) を介して FPGA に送信されます。
6. X-pgae のアドレス(0x00)を送信します。このアドレスは、API (write_one_x_address)によってパッケージ化され、MCU の I2C インターフェース (API : i2c_write_buffer) を介して FPGA に送信されます。
7. Flash から繰り返しデータをリードバックし、書き込まれたビットストリームファイルと比較します (API:recv_one_y_page_and_verify)。
8. 繰り返しリードバック・比較が完了したら、autoboot-pattern データを最初の X-page に書き込みます。autoboot-pattern データは API (i2c_prog_xpage)によってパッケージ化され、MCU の I2C インターフェース (API : i2c_write_buffer)を介して FPGA に送信されます。
9. ISC_DISABLE(0x3A)と ISC_NOOP(0x02)を送信します。この 2 つの命令は API (write_inst_and_data)によってパッケージ化され、MCU の I2C インターフェース(API: i2c_write_buffer) を介して FPGA に送信されます。

オンチップ Flash をプログラムするためのフローチャートを図 6-5 に示します。

図 6-5 オンチップ Flash のプログラミングのフローチャート



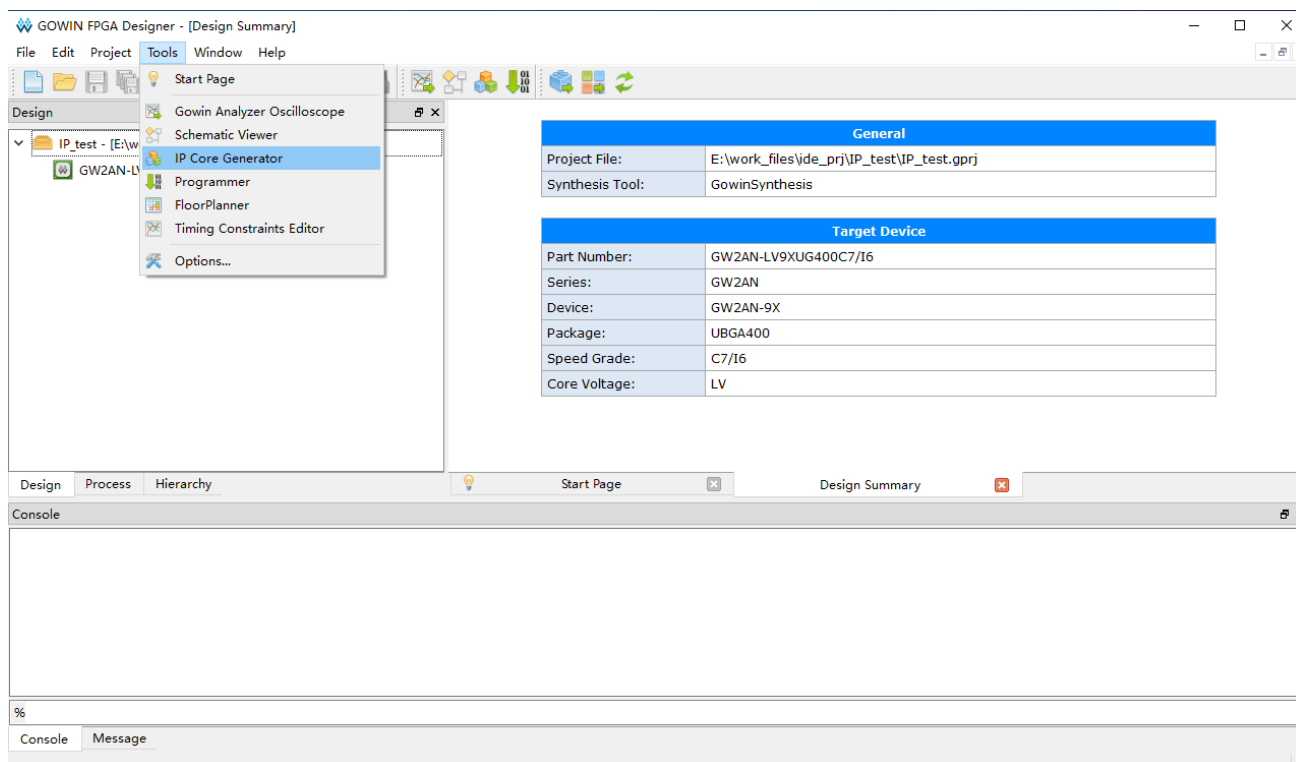
7 GUI での構成

Gowin ソフトウェア GUI のメニューバー > Tools > IP Core Generator を使用して IP を呼び出し・構成できます。

1. IP Core Generator を開く

プロジェクトが作成された後、“Tools > IP Core Generator”を開きます(図 7-1)。

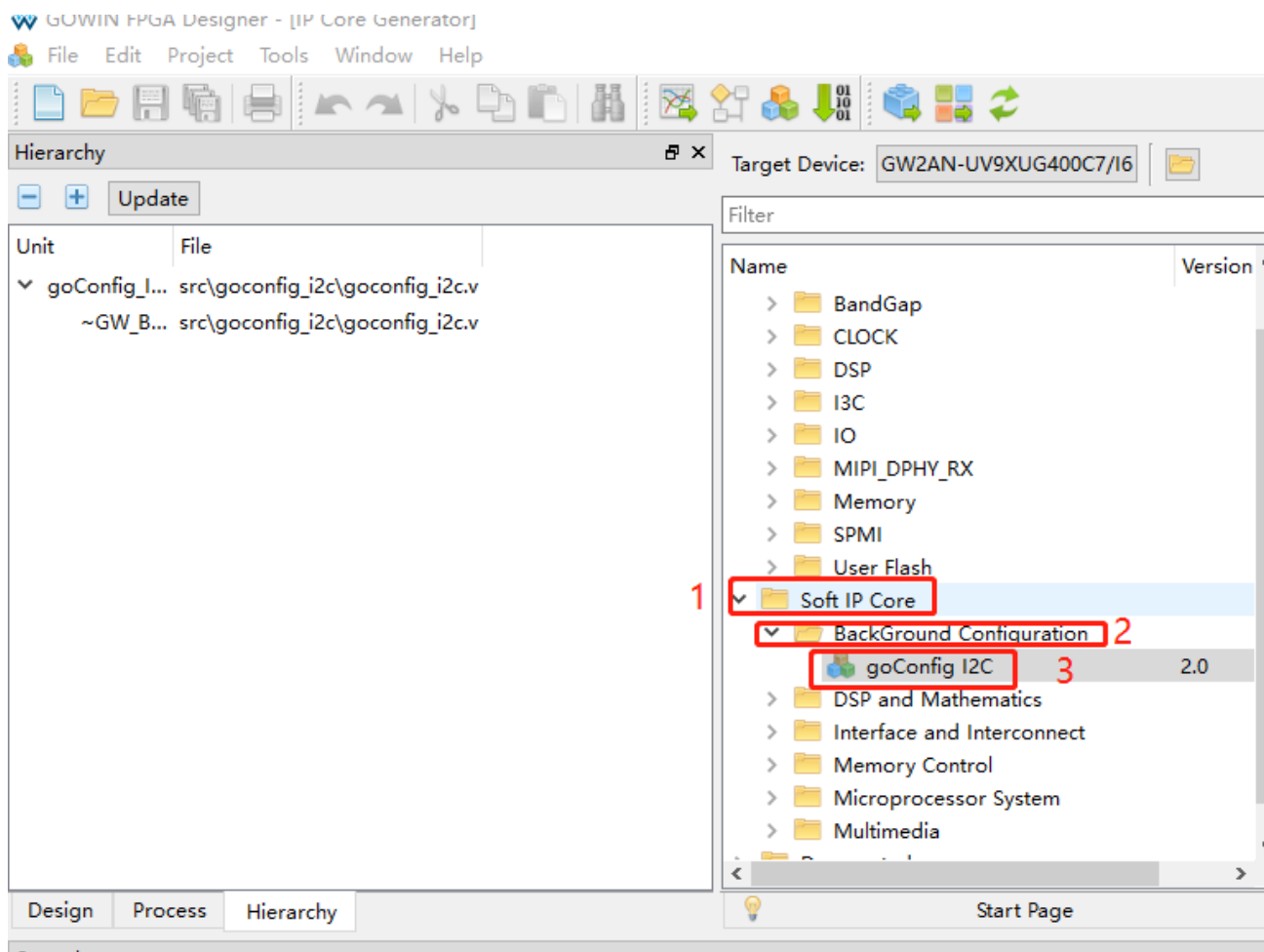
図 7-1 IP Core Generator を開く方法



2. Gowin goConfig I2C IP コアを開く

“Soft IP Core > BackGround Configuration > goConfig I2C” 選択してダブルクリックします(図 7-2)。

図 7-2 goConfig I2C IP コアを開く

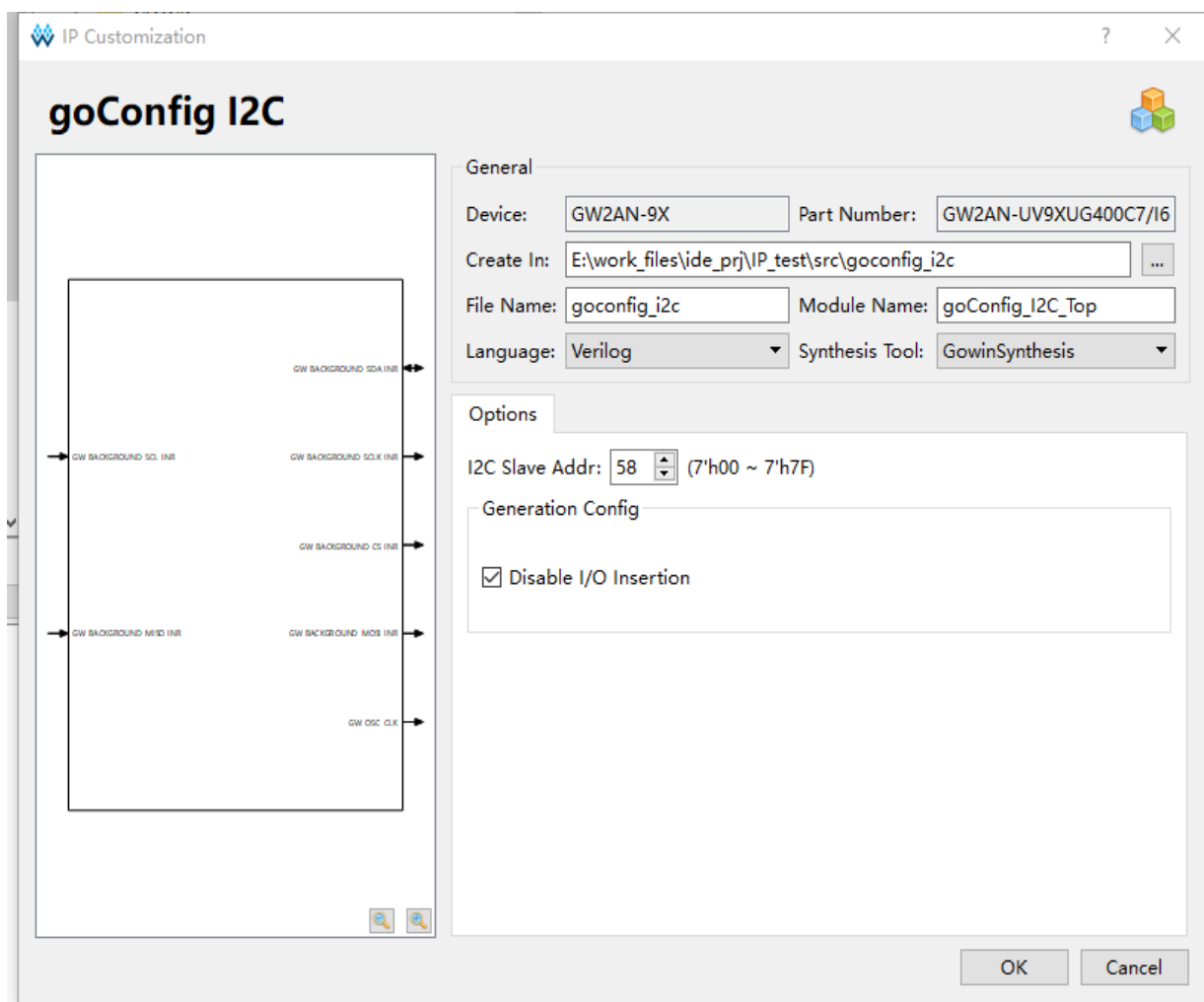


3. Gowin goConfig I2C IP コアの構成画面

Gowin goConfig I2C IP コアの構成画面は、図 7-3 に示すとおりです。左側は、goConfig I2C IP コアのインターフェースの説明図です。右側は、構成可能なオプションです。

- "File Name"フィールドでファイルの名前を構成することができます。
- "Module Name"フィールドでトップモジュールの名前を構成することができます。
- I2C Slave Addr フィールドでスレーブのアドレスを構成することができます。

図 7-3 goConfig I2C IP の構成画面



8 リファレンスデザイン

このセクションでは、Gowin goConfig I2C IP のリファレンスデザインの構成とその使用方法について説明します。詳細については、GOWIN のホームページのリファレンスデザインを参照してください。

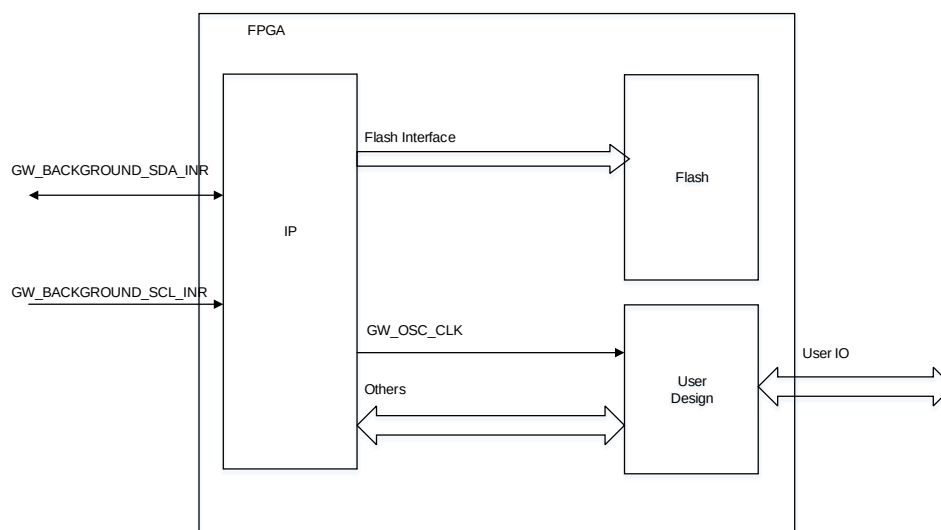
8.1 リファレンスデザイン

8.1.1 FPGA プロジェクトの説明

Gowin goConfig I2C IP の FPGA リファレンスデザインには、主に IP モジュールとユーザーモジュールが含まれています。

- IP モジュール：その使用例が提供されています。モジュールの説明は以下の通りです。
 - IP を使用するには、Gowin ソフトウェアでの構成を変更する必要があり、その例はリファレンスデザインに含まれています。詳細な手順については、第 5 章を参照してください。
 - IP の IO の一部には使用制限があり、リファレンスデザインでの使い方はその参考例です。詳しくは、第 4 章および第 5 章内の注意事項を参照してください。
- ユーザーモジュール：簡単な LED テスト例です。

図 8-1 リファレンスデザインの基本構造



8.1.2 リファレンスデザイン

Gowin goConfig I2C IP リファレンスデザインの関連ファイルを表 8-1 および表 8-2 に示します。

表 8-1 リファレンスデザインの src フォルダのコンテンツ一覧(2C シリーズ)

名称	説明
Top.v	トップモジュール(LEDテスト例を含む)
goConfig_I2C_1p5c.cst	物理制約ファイル
goconfig_i2c	IPフォルダ
----goconfig_i2c.v	生成されたIPファイル(暗号化)
----goconfig_i2c.vo	生成されたIPのネットリストファイル(シミュレーション用)

表 8-2 リファレンスデザインの src フォルダのコンテンツ一覧(9X シリーズ)

名称	説明
goConfig_I2C_9X.cst	物理制約ファイル
LED_test	LED テストフォルダ
----LED_TOP.v	トップモジュール
----LED.v	LED テストモジュール
goconfig_i2c	IP フォルダ
----goconfig_i2c.v	生成された IP ファイル(暗号化)
----goconfig_i2c.vo	生成された IP のネットリストファイル(シミュレーション用)

8.1.3 リファレンスデザインの注意事項

Gowin goConfig I2C IP のリファレンスデザインには、FPGA プロジェクトと一部のドキュメントのみが含まれており、シミュレーション環境や I2C の駆動例は提供されません。

"I2C の駆動例"については、弊社の FAE までお問い合わせください。

