

インテル[®] MAX[®] 10 FPGA On-Chip Flash を使用するときの注意点

Ver.18.0



インテル[®] MAX[®] 10 FPGA - On-Chip Flash を使用するときの注意点

<u>目次</u>

は	じめに	3
1.	MAX® 10 FPGA デバイスの On-Chip Flash	3
2.	MAX® 10 FPGA で Nios® II を使用する際の On-Chip Flash での注意点	4
-	2-1. プログラムが UFM より大きいサイズの場合でも Error にならずビルドが通ってしまう	5
	2-2. サイズオーバーした elf ファイルは HEX 変換や POF の生成を行っても Error にならない	6
3.	対処方法	7
改	版履歴	.12

<u>はじめに</u>

インテル® MAX® 10 FPGA にはフラッシュメモリーが内蔵されており、後述で示す通り CFM/UFM という形で提供されます。この時 Nios® II のソフトウェアをフラッシュメモリーに格納する場合 UFM に実装することになりますが、フラッシュメモリーのサイズとしては CFM を含めた全体のサイズで定義されます。このため Nios® II を使用する際にプログラム・サイズが On-Chip Flash の UFM 領域のサイズを超えていても Error として通知されません。

その為、MAX® 10 FPGA の UFM にユーザープログラムを格納する場合には、ユーザー側で常にサイズに注 意する必要があります。

この資料では、MAX[®] 10 FPGA で Nios[®] II を使用する際の On-Chip Flash での注意点と対処方法について説 明します。

- 1. MAX[®] 10 FPGA デバイスの On-Chip Flash
- 2. MAX® 10 FPGA で Nios® II を使用する際の On-Chip Flash での注意点
- 3. 対処方法

1. MAX[®] 10 FPGA デバイスの On-Chip Flash

インテル[®] MAX[®] 10 FPGA は、低コスト、シングルチップ、スモール・フォーム・ファクターの不揮発性プログラマ ブル・ロジック・デバイスです。

アナログ・デジタル・コンバーター (ADC) や、2 つのイメージを格納してダイナミックに切り替えることが可能 なデュアル・コンフィグレーション・フラッシュメモリーなどの機能をシングルチップ上に搭載しています。

また、Nios® II ソフト・コア・エンベデッド・プロセッサーのサポート、デジタル信号処理(DSP)ブロック、ソフト DDR3 メモリー・コントローラーといった、フル装備の FPGA 機能を備えています。

インテル® MAX® 10 FPGA についての詳細は、以下のページを参照ください。

インテル[®] MAX[®] 10 FPGA

https://www.intel.co.jp/content/www/jp/ja/products/programmable/fpga/max-10.html

MAX® 10 FPGA デバイスには、On-Chip Flash が 2 つの部分に分割されています。

- Configuration Flash Memory (CFM)
 MAX® 10 FPGA のハードウェア・コンフィギュレーション・データを格納します。
- User Flash Memory (UFM)

ユーザーデータまたはソフトウェア・アプリケーションを格納します。

MAX® 10 FPGA デバイスの UFM についての詳細は、以下の資料を参照ください。

Intel® MAX® 10 User Flash Memory User Guide

https://www.intel.com/content/dam/www/programmable/us/en/pdfs/literature/hb/max-10/ug m10 ufm.pdf

MAX® 10 ユーザー・フラッシュメモリー・ユーザーガイド (日本語版)

https://www.intel.co.jp/content/dam/altera-www/global/ja_JP/pdfs/literature/hb/max-10/ug_m10_ufm_j.pdf



2. MAX® 10 FPGA で Nios® II を使用する際の On-Chip Flash での注意点

Platform Designer にて On-Chip Flash Intel FPGA IP (Altera On-Chip Flash IP) を下図のように設定した場合、 Nios® II Software Build Tools (SBT) for Eclipse では UFM 領域のみならず、CFM の領域まで On-Chip Flash として 見えています (この資料では、MAX® 10 FPGA デバイス 10M08 を例として説明しています)。

🎦 Parameters 🛛			- 5 🗆			
System: nios_system Path: onchip_flash						
On–Chip Flash Intel FPGA IP altera_onchip_flash			Details			
Parameters						
Data interface:	Parall 👻					
Read burst mode:	Incrementi 👻					
Read burst count:	8 🗸	8 💌				
Configuration Mode						
Configuration Scheme: Internal Configurati 👻						
Configuration Mode:	Single Uncompr	essed Image	•			
Elsoh Momoru						
		diana Maratan	T			
Sector ID Access Mode	· /	address Wapping	Туре			
Read and w	rite US	00000 - 0x03111 04000 - 0x07444	UEM			
3 Bead and w	rite 00	:08000 - 0x1c7ff	LIEM			
4 Read and w	rite 0>	:1c800 - 0x2afff	CFM			
5 Read and w	rite 🛛	26000 - 0×4dfff	CFM			
+ -						
Clock Source	500					
The on-chip flash megafunction will be run with 100000000 Hz clock frequency.						
Flash Initialization						
🛄 Initialize flash content						
Enable non-default initialization file						
User created hex or mif file:	altera_onchip_fla	shhex				
User created dat file for simulation:	altera_onchip_fla	sh.dat				
The on-chip flash is not initialized during device programming.						

【図 2-1】On-Chip Flash IP の設定

Nios[®] II SBT の BSP Editor から見ても、UFM と CFM の区別はしておらず、CFM も含めたサイズ (319456 Bytes) として On-Chip Flash 領域が見えています。

ile Edit Tools Help						
Main Software Packages Drivers	Linker Script Enable File Genera	ation Target BSP Dir	ectory			
Linker Section Mappings						
Linker Section Name	Linker Region Name		Memory	Device Name		Add
.bss onchip ram		0	onchip_	ram		Remove
entry reset		c	onchip_	flash_data		Restore Defaults
.exceptions	onchip_ram	c	onchip_	ram		
.heap	onchip_ram	c	onchip_	ram		
.rodata onchip_ram		onchip_ram				
.rwdata onchip_ram		onchip_ram				
.stack onchip_ram		onchip_ram				
.text	onchip ram		onchip	ram		
Linker Memory Regions						
Linker Region Name	Address Range	Memory Device Na	me	Size (bytes)	Offset (bytes)	Add
onchip ram	0x01010020 - 0x01018FFF	onchip ram	-	36832	32	Remove
onchip ram BEFORE EXCEPTION	0x01010000 - 0x0101001F	onchip ram		32	0	Restore Defaults
onchip_flash_data	0x00000020 - 0x0004DFFF	onchip_flash_dat	a	319456	32	
reset	0×00000000 - 0×0000001F	onchip flash dat	a i	32	U	Add Memory Device
		CFM も含め	たサ.	イズとして	見える	Remove Memory Device
		010				Memory Usage

【図 2-2】 Nios® II SBT から見た On-Chip Flash 領域

2-1. プログラムが UFM より大きいサイズの場合でも Error にならずビルドが通ってしまう

【注意点 1】

上記のことから、.text を On-Chip Flash に配置する際は、プログラム (.elf ファイル) が UFM に収まるかで はなく、On-Chip Flash に配置できるかでしかビルド時に判定されません。

つまりプログラムが UFM より大きいサイズの場合でも Error の表示無くビルドが通ってしまいます。

そのため UFM のサイズを考慮してプログラムを作成する必要があります。

Table 4: UFM and CFM Sector Size

This table lists the dimensions of the UFM and CFM arrays.									
	Pages per Sector					Page	Maximum	Total	
Device	UFM1	UFM0	CFM2	CFM1	CFM0	Size (Kbit)	User Flash Memory Size (Kbit) ⁽⁴⁾	Configura- tion Memory Size (Kbit)	OCRAM Size (Kbit)
10M02	3	3	0	0	34	16	96	544	108
10M04	0	8	41	29	70	16	1248	2240	189
10M08	8	8	41	29	70	16	1376	2240	378
10M16	4	4	38	28	66	32	2368	4224	549
10M25	4	4	52	40	92	32	3200	5888	675
10M40	4	4	48	36	84	64	5888	10752	1260
10M50	4	4	48	36	84	64	5888	10752	1638

10M08 の場合 UFM サイズは、((8*16)+(8*16))/8=32 KByte

・ UFM0 のページサイズ : 8 (page) * 16 (Kbit)

・ UFM1 のページサイズ : 8 (page) * 16 (Kbit)

上記 2 つの値を合算し、単位を Byte に直すために ÷8 で 32 KByte

【図 2-3】 UFM と CFM のアレイサイズ

[BSP build complete] Info: Compiling hello_world.c to obj/default/hello_world.o nios2-elf-gcc -xc -MP -MMD -c -I../hello_bsp//HAL/inc -I../hello_bsp/ -I../hello_bsp//drivers/in Info: Linking hello.elf nios2-elf-g++ -T'../hello_bsp//linker.x' -msys-crt0='../hello_bsp//obj/HAL/src/crt0.o' -msys-l: nios2-elf-insert hello.elf --thread_model hal --cpu_name nios2 --qsys true --simulation_enabled Info: (hello.elf) 33 KBytes program size (code + initialized data). Info: 2872 Bytes free for stack + heap. Info: Creating hello.objdump nios2-elf-objdump --disassemble --syms --all-header --source hello.elf >hello.objdump [hello build complete] 19:47:07 Build Finished (took 20s.247ms)

【図 2-4】 Nios® II SBT でビルドしたプログラムが 10M08 の UFM サイズ 32 KB を超えた例



2-2. サイズオーバーした elf ファイルは HEX 変換や POF の生成を行っても Error にならない

【注意点 2】

.elf ファイルはその後、

- HEX に変換
- POF を生成
- プログラムする

などに使用されますが、サイズオーバーした elf ファイルは、これらのどの過程でも Error は発生しません。



【図 2-5】 HEX に変更した場合のログ



【図 2-6】 POF を生成した場合のログ

【参考】

Nios® II SBT でプログラムを HEX ファイルに変換する方法や、Quartus® Prime で SOF ファイルと UFM 用 HEX ファイルから POF ファイルを生成する方法については、以下のページが参考になります。

MAX® 10 の UFM で Nios® II をブートさせてみよう [前編]

https://service.macnica.co.jp/library/118989

MAX[®] 10 の UFM で Nios[®] II をブートさせてみよう [後編] https://service.macnica.co.jp/library/119173

3. <u>対処方法</u>

Nios® II SBT の BSP Editor 上であらかじめ配置できるセクション・サイズに変更します。 セクション・サイズを変更することでビルド時に Error として出力されるようになります。

① Platform Designer の On-Chip Flash Intel FPGA IP (Altera On-Chip Flash IP) において UFM として使用で きる領域を確認します。

例) この例では、0x0 ~ 0x1c7ff まで使用可能です(0x1c800 → 116736 Byte)

Platform Designer - nios_system.qsys (C:¥Temp¥m File Edit System Generate View Tools Help	nax10_evakit_v	18.0¥nios_system.qsys)							
📑 IP Catalog 🙁 🗕 🗗 🗖	📰 System Cont	ents 🕴 Address Map	없 Interconnect Requi	rements 🛛					
	× • •	System:nios_system F	ath: onchip_flash						
Project	+ Use Con	nections	Name	Description	Export	Clock	Base	End	IRQ
New Component	× /		⊟ clk_0 clk_in clk_in_reset	Clock Source Clock Input Reset Input	clk reset	exported			
Basic Functions			clk clk reset	Clock Output Reset Output	Double-click to export	clk_0			
Interface Protocols Interface Protocols Interface Protocols	<u> </u>		⊡ onchip_ram	On-Chip Memory (RAM or ROM) Inte					
Memory Interfaces and Controllers Processors and Peripherals		$\begin{array}{c} \bullet \bullet$	clk1 s1	Clock Input Avalon Memory Mapped Slave	Double-click to export Double-click to export	clk_0 [clk1]	≠ 0×0101 0000	0×0101 8fff	
Qsys Interconnect		• • • • •	reset1	Reset Input	Double-click to export	[clk 1]			
	✓		⊟ jtag_uart clk	Clock Input	Double-click to export	clk_0			
			reset	Reset Input	Double-click to export	[c]k]	0.0100 1010	0.0100 1017	
		•	avalon_stag_slave irq	Avaion Memory Mapped Slave Interrupt Sender	Double-click to export	[clk]	= 0X0102_1010	0x0102_1017	<u>→</u> 0
			🗆 🛄 nios 2	Nios II Processor	Haublandunk im annak				
		• 	reset	Reset Input	Double-click to export	[clk]			
			data_master	Avalon Memory Mapped Master	Double-click to export	[clk] [elk]			
			instruction_master	Interrupt Receiver	Double-click to export	[clk]	1	IRQ 0	IRQ 31←
New Edit			debug_reset_request	Reset Output	Double-click to export	[c]k]			
			custom_instruction_m.	Cu	Double-click to export	ICIK]	• 0x0102_0800	0x0102_0111	
Hierarchy & Device Family &			□ mm_bridge clk	nchip_flash を	ダブルクリッ	クします	f		
B ➡ led		$\bullet \bullet $	s0	Avalon Memory Mapped Slave	Double-click to export	(cik)	= 0×0000_0000	0×000f_ffff	
tereset tereset	57		T anakin flack	Avaion Memory Mapped Master	Double-click to export	[clk]			
iê-e⊐⊧ jtag_uart ie-e⊐⊧ led	•			Clock Input	Double-click to export	clk_0			
teren mm_bridge			nreset	Reset Input	Double-click to export	[clk]	- 00000 0000	0-0004 4666	
ter-ter onchip_flash		$ \stackrel{\bullet}{\leftarrow} \stackrel{\bullet}{\longrightarrow} $	csr	Avalon Memory Mapped Slave	Double-click to export	[clk]	= 0x0008_0000	0×0008_0007	
B-= Connections			⊟ led	PIO (Parallel I/O) Intel FPGA IP		- 11- 0			
		• • • • • • • • • • • • • • • • • • •	cik reset	Reset Input	Double-click to export Double-click to export	CIK_U [clk]			
		••••••	s1	Avalon Memory Mapped Slave	Double-click to export	[clk]	₽ 0×0102_1000	0×0102_100f	
		۲M	external_connection	Conduit	lea				
						1			
🞦 Parameters 🛛					- 6 🗆				
System: nios_system Pa	ath: onchip_f	flash							
On-Chip Flash Int	el FPGA	IP							
altera_onchip_flash	Volume and Annual Annua	2.00448 - 1			Details				
Parameters									
Data interface:		Parall	-						
Read burst mode:		Increment	- -						
Bead burst count:		0							
		• •							
Configuration Mode									
Configuration Scheme:		Internal C	onfigurati 🚽						
Configuration Mode:		Single Un	compressed Image						
				UFM	として使用で	きる領	域を確認しる	ます	
Flash Memory									
Sector ID	Acces	ss Mode	Address Map	ping Type					
1	Read	and write	0×00000 - 0×	03fff UFM					
2	Read	and write	0×04000 - 0×	07fff UFM					
3	Read	and write	0×08000 - 0×	1c7ff UFM					
4	Read	and write	0×1c800 - 0×3	2attt CFM					
5	Read	and write	0×2b000 - 0×	4dfff CFM					

【図 3-1】 On-Chip Flash IP において UFM として使用できる領域を確認



File Edit Navigate S	Search Project Run Nios II W	indow Help	
	• 🔂 • 🔂 • 🚱 • 🔯 • 🔘	▼ 4 ▼ (⊖ <i>3</i> ° ▼) 2 ▼ 10 ▼ 10 ▼ 10 ▼ 0 ▼ 0 ▼	
陷 Project Explorer 🔀	E <	\$ 2 2 2 2	
> 😂 hello	BSD プロ・		
P Bello_bsp [nios_s]	New New		
	Go Into		
	Open in New Window		
	Сору	Ctrl+C	
	Paste	Ctrl+V	
	💥 Delete	Delete	
	ی _ Remove from Context	Ctrl+Alt+Shift+Down	
	Source	•	
	Move		
	Rename	F2	
	🔤 Import		
	🚵 Export		
	Build Project		
	Clean Project		
, , , , , , , , , , , , , , , , , , ,	Refresh	F5	
		\sim	
	Debug As		
	Profile As Restore from Local Hist	2014	
	Nios II	Nios II Command Shell	
	Run C/C++ Code Analysis	sis Generate BSP	1
	Team	BSP Editor	
	Compare With	Elash Programmer	_

【図 3-2】 Nios II SBT の BSP Editor を開く

- ③ BSP Editor の Linker Script タブにおいて、onchip_flash_data の領域を Platform Designer で確認したサ イズに設定します。このとき、先頭の 32 Byte(リセットベクター領域)を引いた値で設定します。
 - 例)10M08 の場合は、116736-32=116704 Byte

設定前	: onchip	flash	data size = 3	3 19456 Byte
-----	----------	-------	---------------	---------------------

* BSP Editor - settings.bsp				
File Edit Tools Help				
Main Software Packages Drivers Lir	ker Script Enable File Generation Targ	et BSP Directory		
Linker Section Mappings				
Linker Section Name	Linker Region Name		Memory Device Name	
.bss	onchip_ram	0	onchip_ram	
.entry	reset	0	onchip_flash_data	
.exceptions	onchip_ram		onchip_ram	
.heap	onchip_ram	0	onchip_ram	
.rodata	onchip_ram	0	onchip_ram	
.rwdata	onchip_ram		onchip_ram	
.stack	onchip_ram	0	onchip_ram	
.text	onchip ram		onchip ram	
Linker Memory Regions	ダブルクリッ	ックして 116704 に変	更します	
Linker Region Name	Address Range	Memory Device Name	Size (bytes)	Offset (bytes)
onchip_ram	0×01010020 - 0×01018FFF	onchip_ram	368	332 32
onchip_ram_BEFORE_EXCEPTION	0×01010000 - 0×0101001F	onchip_ram		32 0
onchip_flash_data	0×00000020 - 0×0004DFFF	onchip_flash_data	3194	456 32
reset	0x00000000 - 0x0000001F	onchip flash data		32 0

【図 3-3】変更前の onchip_flash_data のサイズ (10M08 の場合の例)



設定後:ONCHID TIASN Data SIZE = 1	16/04 Byte
--------------------------------	------------

💩 BSP Editor - settings.bsp						
File Edit Tools Help						
Main Software Packages Drivers Link	er Script Enable File Generation Targ	et BSP Directory				
Linker Section Mappings						
Linker Section Name	Linker Region Name		Memory Device Name			
.bss	onchip_ram	onchip_ram				
.entry	reset	reset		onchip_flash_data		
.exceptions	onchip_ram	onchip_ram		onchip_ram		
.heap	onchip_ram	onchip_ram		onchip_ram		
.rodata	onchip_ram	onchip_ram		onchip_ram		
.rwdata	onchip_ram	onchip_ram		onchip_ram		
.stack	onchip_ram	onchip_ram		onchip_ram		
.text	.text onchip ram		onchip ram			
Linker Memory Regions						
Linker Region Name	Address Range	Memory Device Name	Size (bytes)	Offset (bytes)		
onchip_ram	0x01010020 - 0x01018FFF	onchip_ram	3683	2 32		
onchip_ram_BEFORE_EXCEPTION	0x01010000 - 0x0101001F	onchip_ram	3	2 0		
pnchip_flash_data	0x00000020 - 0x0001C7FF	onchip_flash_data	11670	4 32		
reset	0x00000000 - 0x0000001F	onchip flash data	3:	2 0		

【図 3-4】変更後の onchip_flash_data のサイズ (10M08 の場合の例)

④ [Generate] ボタンをクリックし、その後 [Exit] ボタンをクリックします。



【図 3-5】 [Generate] ボタンをクリックし、その後 [Exit] ボタンをクリック

⑤ Nios[®] II SBT の Application プロジェクトを右クリックし、「Build Project」を実行します。



【図 3-6】 Application プロジェクトのビルド



本問題に対応するために、MAX 10 の UFM 内にソフトウェアが収まるかどうかを簡易的にチェックするソフト ウェア check_size.exe (check_size.c)を用意しました (別途、ダウンロードしてください)。

※ ソフトウェア check_size.exe (check_size.c) は、参考として提供するものであり、運用した結果の影響について は責任を負いかねますので、あらかじめご了承ください。

このソフトウェアでは、Nios[®] II SBT の Application プロジェクト内でビルド時に作成された .objdump ファイル 内の記載から対象セクションのサイズを抜き出し足し合わせて、最終的にどれくらいのサイズが必要なのかを出 力しています。

⑥ Nios[®] II SBT によるビルド後、Nios[®] II Command Shell より下記コマンドを入力し check_size.exe を実行して、作成したソフトウェアが指定した UFM に収まるかを確認します。

\$./check_size.exe <アプリケーション名.objdump> <UFM サイズ> 🛽

※ check_size.exe を実行する前に、Application プロジェクト・ディレクトリーに check_size.exe をコピーしてお きます。

UFM に収まる場合

<pre>/cygdrive/c/Temp/max10_evakit_v18.0/software/hello_small</pre>
Altera Nios2 Command Shell [GCC 4]
Version 18.0, Build 614 Application プロジェクトのディレクトリーに移動します
1114994811149A /cygdrive/c/intelFPGA/18.0 \$ cd "C:¥Temp¥max10_evakit_v18.0¥software¥hello_small"
11149@HD11149A /cygdrive/c/Temp/max10_evakit_v18.0/software/hello_small \$ cp_/cygdrive/c/Temp/check_size.exe ./check_size.exe
11149@HD11149A /cygdn Application プロジェクト・ディレクトリーに check_size.exe をコピーします
\$ Is Makefile hello_small.elf hello_world_small.c system check_size.exe hello_small.map obj create-this-app hello_small.objdump readme.txt
11149@HD11149A /cygdrive/c/Temp/max10_evakit_v18.0/software/bello_small \$./check_size.exe hello_small.objdump 0x8000 ح check_size.exe を実行します
ufm_size : 0x8000
entry size : 0x0 exceptions size : 0x0 text size : 0xd54 rodata size : 0x30 rwdata size : 0x118
[Result] All Section size : 0xe9c
000000000000000000000000000000000000000
OK ,Can be stored! - You can use more 29028 byte
000000000000000000000000000000000000000
11149@HD11149A /cygdrive/c/Temp/max10_evakit_v18.0/software/hello_small \$_

【図 3-7】 check_size.exe ツールによる UFM に収まる場合の表示例



UFM に収まらない場合



【図 3-8】 check_size.exe ツールによる UFM に収まらない場合の表示例

<u> 改版履歴</u>

Revision	年月	概要
1	2018年12月	初版

免責およびご利用上の注意

弊社より資料を入手されましたお客様におかれましては、下記の使用上の注意を一読いただいた上でご使用ください。

- 1. 本資料は非売品です。許可無く転売することや無断複製することを禁じます。
- 2. 本資料は予告なく変更することがあります。
- 本資料の作成には万全を期していますが、万一ご不明な点や誤り、記載漏れなどお気づきの点がありましたら、本資料を入手されました下記代理店までご一報いただければ幸いです。
 株式会社マクニカ アルティマ カンパニー <u>https://www.alt.macnica.co.jp/</u> 技術情報サイト アルティマ技術データベース <u>http://www.altima.jp/members/</u>
- 4. 本資料で取り扱っている回路、技術、プログラムに関して運用した結果の影響については、責任を負いかねますのであらかじめご了承ください。
- 5. 本資料は製品を利用する際の補助的な資料です。製品をご使用になる際は、各メーカ発行の英語版の資料もあわせてご利用ください。