

XILINX FPGA XC3S500E 開発ボードのマニュアル

株式会社日昇テクノロジー

http://www.csun.co.jp

info@csun.co.jp

作成日 2014/8/11



copyright@2014



• 修正履歴

NO	バージョン	修正内容	修正日
1	Ver1.0	新規作成	2014/8/11

※ この文書の情報は、文書を改善するため、事前の通知なく変更されることがあります。 最新版は弊社ホームページからご参照ください。「http://www.esun.co.jp」 ※ (株)日昇テクノロジーの書面による許可のない複製は、いかなる形態においても厳重に 禁じられています。



目次

1、注意事項	4
2、開発ボードの主な部品及びハードウェアリソース	5
2.1 主な部品	5
2.2 ハードウェアリソース	5
2.3 PCBとサイズ	7
3、開発ボードの実装と使用	8
3.1 開発ボードの実装	8
3.2 開発ボードの使用	9
3.2.1 ロジック開発プロセスの紹介	9
3.2.2 ダウンロード	3
3.2.3 SPI Flash ダウンロード 1	5
4、開発ボード回路説明	:5
4.1 電源回路	:5
4.2 クロック回路	:5
4.3 リセット回路	:5
4.4 JTAG インタフェース回路 2	:6
4.5 SPI FLASH インタフェース回路 2	:6
4.6 LED 回路 2	27
4.7 メモリーSDRAM 回路 2	:8
4.8 外部拡張 IO 2	:9



1、注意事項

使用要求

- ケーブルを抜き差しする前に、開発ボードを使用中止し、電源を切る。
- 不明なもの又は液体が開発ボードと接触した場合に、すぐに開発ボードを使用中止し、電源を切って開 発ボードに接続するケーブルを全部抜き出す。
- 異常状況が出る場合に、例えば、設備から煙が出る、異臭を放つ時に、すぐに開発ボードを使用中止し、 電源を切る。
- 長時間に使用しない場合に、電源を切る。

静電気安全及び保護

静電気はしばしば開発ボードのチップを損傷する発生誘因となっている。そのため、開発ボードを触る 前に十分な保護対策を行わなければならない。

ケーブルの挿抜安全

- ケーブルを挿抜する場合に、フラットケーブルの方向を間違い無い様に注意してください。間違った場合、ロジックチップ又はダウンロードケーブルを損傷する恐れがある。
- 電源を入れたまま JTAG ケーブルを挿抜操作禁止。通電状態操作すればロジックチップに内蔵された設 定回路に致命的な損傷をもたなす。(コンピュータマザーボードのボードを挿抜と同じ)

外部電源の使用

開発ボードが DC 5V 電源入力をサポートする。DC 電源インタフェースを使う場合に、電圧が間違い又は 極性が逆接するで開発ボードを損傷しないように、電源アダブタ出力が 5V、かつ内側が正極、外側が負極 である事を確認する。

開発ボードは UBS 電源給電もサポートする。標準携帯電話の充電器及び Mini USB 線により開発ボード に給電する。

開発ボードを給電する場合に、DCインタフェースでもMini USBインタフェースでも使える。(自由に一つを選ぶ)

開発ボードとインタフェースボードの接続

3列ソケットにより開発ボートとインタフェースボードを接続する。開発ボードにはピンヘッダ、イン タフェースボードにはソケットがある。単独に本ボードを使わない場合は、出来るだけ抜き出さないでくだ さい。

本ボードを取り出す時に、片手でインタフェースボートを固定しながら本ボードを握り適当な力で取り 出す。(ボードを握る力を適当に調整する)

本ボードを取り付ける時、ボードの方向を十分に注意してください。ピンヘッダとソケットが1対1対 応するかを確認し、取り付ける。



2、開発ボードの主な部品及びハードウェアリソース

2.1 主な部品

開発ボードのコアチップはXILINX Spartan®-3E シリーズのFPGA XC3S500EPQ208を使用する。 Spartan®-3E FPGAはロジック最適化、低コスト、完全な機能を持っているプラットフォームの一つである。 100Kから1.6Mまでのシステムゲート、66から376までのI/0インタフェースがある。詳細は以下のように:

Table 1: Summary of Spartan-3E FPGA Attributes

Davica	System	System Equivalent (One CLB Array		ces)	Distributed	Block	Dedicated	DCMe	Maximum	Maximum		
Device	Gates	Logic Cells	Rows	Columns	Total CLBs	Total Slices	RAM bits ⁽¹⁾	bits ⁽¹⁾	Multipliers	DCIVIS	User I/O	I/O Pairs
XC3S100E	100K	2,160	22	16	240	960	15K	72K	4	2	108	40
XC3S250E	250K	5,508	34	26	612	2,448	38K	216K	12	4	172	68
XC3S500E	500K	10,476	46	34	1,164	4,656	73K	360K	20	4	232	92
XC3S1200E	1200K	19,512	60	46	2,168	8,672	136K	504K	28	8	304	124
XC3S1600E	1600K	33,192	76	58	3,688	14,752	231K	648K	36	8	376	156
Notoo												

1. By convention, one Kb is equivalent to 1,024 bits.

FPGAにとってはロジックチップの論理ゲート数とRAM bit数はFPGAの重要な参考指標である。それに対して、IO数はそれほど重要ではない。

2.2 ハードウェアリソース

▶メインチップはXILINX 会社のSpartan®-3E シリーズのFPGA XC3S500EPQ208を採用。

- ▶50MHz 水晶発振器、システムが動作するメインクロックを提供する。
- ▶M25P16シリアル・コンフィギュレーション・チップを搭載する。FPGAロジックも、SOPCプログラムも保存できる。
- ▶64Mbit のSDRAM (HY57V64162) を搭載する。SOPC設計もできる。
- >大電力 LDO 電源管理チップ LM1085-3.3V を採用し、3Aの3.3v 電圧出力までサポートする。
- ▶AMS1117-1.2V/2.5Vレギュレータチップを使用し、FPGAコアに電圧を提供する。
- ▶IN5819 高速ショットキータイオード、電源逆接の保護設計を持っている。
- ▶Self-locking 電源スイッチーク
- ▶5V 直流電源コンセント、電源アダブタでも給電できる。
- ▶MINI USB コネクタ、携帯電話の標準充電器でも給電できる。
- ▶赤 LED 一つ、電源表示灯とする。
- ▶ブルーLED 二つ、LED 点滅試験とする。
- >リセットキー-->、ユーザーキーとする。
- ▶再設定きーがあり、FPGAのロジック再設定に利用する。
- ▶JTAGインタフェースをサポートする。
- ▶JTAGでM25P16のプログラミングをサポートする。
- ▶デカップリング設計。数多くのデカップリングコンデンサーを採用する。
- ▶全ての IO を 2.54mm ピッチ拡張ピンヘッダで引き出す。

イメージは下図の通り:



低価格、高品質が不可能? 日昇テクノロジーなら可能にする





2.3 PCB とサイズ





3、開発ボードの実装と使用

本開発ボードは以下の部品が含んでいる。

- 開発ボード 1個
- Mini USB ケーブル 1本
- スタッドとネジ 4本ずつ

3.1 開発ボードの実装

- スタッドを開発ボードの穴に取り付ける。
- ダウンロードケーブルを開発ボードに接続する。(注意: JTAG フラットケーブルの赤い側が PCB のプラ イング1ピンに接続する。)
- Mini USBを開発ボードに接続し、ボードに電源を提供する。
- 実装された開発ボードをデスクの上に平らに置く。
- パソコンに関連ソフトウェアをインストールする。例えば ISE、テキスト編集ソフトウェア UltraEdit、 コーディングチェックソフトウェア Nlint など。

開発ボードの接続図:



下図がデータケーブルと開発ボードの接続詳細図:







注意:データケーブルを接続する時に、方向を注意してください。

3.2 開発ボードの使用

開発ボードが正確に接続したあと、ボードに電源を入れる。

3.2.1 ロジック開発プロセスの紹介

ロジック開発は、ハードウェア設計とソフトウェア設計二つの部分が含んでいる。ハードウェアはロジ ックチップ回路、JTAG、クロック、リセット・メモリ、入出力インタフェース回路及び他のインタフェース が含んでいる。これは本ボードで提供している。ソフトウェアはHDL プロセスである。この部分はお客様の 事情によって自分で設計する。ここでご参考までいくつかの例を提供する。

ロジック設計プロセスは EDA 開発ソフトと編集ツールを利用して、ロジックチップに対する開発する。 代表的なロジック開発プロセスは機能定義・デバイス選択、入力設計、機能シミュレーション、合成最適化、 最適化後のシミュレーション、実現及び配線後のシミュレーション、ボードレベルのシミュレーション、チ ッププログラミング及びデバッグなどのステップを含んでいる。下図の通り:







機能定義:実現したい機能を定義する。例えば、開発ボード上の二つの LED の点滅。

入力設計:ハードウェア記述言語(HDL)で実現したい機能を記述する(つまり、テキスト編集ソフトウェア UltraEdit に HDL 言語で記述)。回路図モードを推奨していない(非効率的な方法、維持するのが困難、 モジュール構造と再利用も難しい。かつ、回路図のポータビリティが悪い)。例えば、下図は一つの LED 点 滅の設計記述、つまり LED が 50000000 ○の clk ずつ一回点滅する。



parameter TIME_COUNT = 32'd100000000; //input input clk; input reset_n; //output output [1:0] led flash; //LEDは1 s 間隔で点灯する。1: 点灯; 0: 消灯 reg [1:0] led_flash; reg [31:0] cnt; always @(posedge clk or negedge reset_n) begin if (reset_n = 1'b0) cnt <= 32 'b0; else if (cnt >= TINE_COUNT - 1'bl) cant <= #U_DLY 32'b0; else cnt <= #U_DLY ent + 1 'b1; end always @(posedge clk or negedge reset_n) begin if (reset_n - 1'b0) led_flash <= 2'b0; else if (cnt = TIME_COUNT - 1'bl) led_flash <= #U_DLY 2'bll;</pre> else if (cnt = (TIME_COUNT>>1) - 1'b) led_flash <= #U_DLY 2 'b0;</pre> else 2 and

機能シミュレーション:前シミュレーションとも呼ばれる。コンパイルの前にユーザーにより設計された回路のロジック機能を検証する。一般的には Modesim 又は Questa でシミュレーションする。下図が LED 点滅のシミュレーション波形である。Led_flashの値が一定な時間を間隔に1になり、LED を点灯する。



低価格、高品質が不可能?

日昇テクノロジーなら可能にする

M ModelSin SE PLUS 6.	1f							
<u>F</u> ile <u>E</u> dit <u>V</u> iew F <u>o</u> rmat <u>C</u> om	npile <u>S</u> imulate <u>A</u> dd <u>T</u> ool	s <u>W</u> indow <u>H</u> elp						
🗋 🖆 🔚 🎒 🐰 🛍 🛍	22 🗚 🖺 🐧 🕇	· 로추 100 ps 🕈 트나 트	î 📑 79 07 🕱 🖥	Con	tains:	□ -2 X•>	X 🗈 🛛 上 🤉	
Workspace == ::::::= ± 2 ×	💼 wave - default							
Tinstance [🚽 🔶 u led/clk 🛛 🕻	0						
tb_led t	🤞 u_led/reset_n 🔤	1						
	🖃 🚽 u_led/led_flash 🛛 (0 <u>3)</u> 0	(3 (0	(3	<u>)(o</u>)(3)(0	
	(1)	0						
	+	00001ro3						
4 #INITIAL#18 t								
-								
	Now 0	0000 ps	Госсолого со се со с ОП наs			108 ms		
	Cursor 1 7	7300 ps			10796238	7300 ps		
• •	I P		Ц					
👫 Library 🐉 sim 🖺 🚺	l 📰 wave							
Transcript								
# Top level modules:				_				
# tb_led								
# vsim work.tb_led #Loading.l:/Mu_Workspace/led_fl	lash/ut_testbench/testcase1/wo	uk the led						
# Loading J:/My_Workspace/led_fl	lash/ut_testbench/testcase1/wo	ork.led						
# Break key hit	# Break key hit							
# Simulation stop requested. # Simulation Breakpoint: Simulation stop requested.								
# MACRO J:\My_Workspace\led_fl	lash\UT_Testbench\Testcase1\	vrun.tcl PAUSED at line 145						
VSIM(paused)>								
Now: 768,683,720 ns Del	lta: 0 sim:/tb_led							

合成最適化:設計入力を、実際のゲート回路ではなく、ANDゲート、ORゲート、NANDゲート、RAM、トリガーなどの基本ロジックユニットから組み立てられた論理接続ネットリストにコンパイルされる。実際のゲート回路はメーターのレイアウトソフトウェアを利用して、合成に生成された標準ゲートレベル構造に応じて生成する。XILINX 会社の部品は、設計を ISE に導入し、ISE ソフトウェアで合成最適化する。

合成後のシミュレーション:合成後の結果と当初の設計が同じかどうかを確認する。シミュレーリョンの時、 合成に生成された標準遅延ファイルを合成のシミュレーションモデルにマークし、ゲート遅延の影響を推定 できる。

配線レイアウトと実装: つまり、ロジックメーカーのソフトウェアを利用して、ロジックを目標デバイス構造のリソースにマッピングレ、ロジックのベストレイアウトを決定する。ロジックと入出力機能の接続配線 チャネルを選択する上に、接続し、対応的なファイル(例えば設定ファイル、関連報告)を生成し、合成に 生成されたロジックネットリストを具体的なロジックチップに配置する。XILINX 会社の部品は、設計を ISE に導入し、ISE ソフトウェアで配線レイアウトを行って、設定ファイルを生成する。下図は LED 点滅で配線 レイアウトの例である。



📕 ISE Project Navigator (N.70d) - J:\Ny_Vorkspace	e\io_t
📡 File Edit View Project Source Process Tools Window	Lagou
🖸 🎯 🗟 🕼 🖧 🛍 🛍 🗙 🕪 🛯 🖉 🧭 🛄	BP
Design ↔ □	×
📑 View: 💿 🏟 Implementation 🔿 🎆 Simulation	0
Hierarchy	
🚰 🕂 🔄 io_test	
□□	(5)
the var io_test (io_test.v)	ime
No Processes Running	30
Processes: io_test	A
Design Summary/Reports Design Utilities User Constraints Create Timing Constraints I/O Pin Planning (PlanAhead) - Pre-Synthesis I/O Pin Planning (PlanAhead) - Post-Synthesis Floorplan Area/IO/Logic (PlanAhead) Synthesize - XST Implement Design Configure Target Device Analyze Design Using ChipScope	

タイミングシミュレーリョン:後シミュレーションとも呼ばれる。配線レイアウトの遅延情報を設計ネット リストにマークし、タイミングの違反(即、タイミング約束条件又はデイバイス固有のタイミング規則に満 足しない、例えば、セットアップ時間、ホールド時間など)を検出する。タイミングシミュレーリョンの遅 延情報は最も正確、最も完全である。

ボードレベルのシミュレーション:主に高速回路設計に使用される。高速システムの信号完全性、電磁妨害 などを分析し、一般的には第二者によりシミュレーションと検証を行われる。

チッププロフラミングとデバッグ:チッププロフラミングは配線レイアウトと実現後に生成された bit ファ イルをロジックチップ又は設定チップにダウンロードし、定義された機能を実現させる。デバッグは FPGA チップのオンラインエンベデッドロジックアナライザで(例えば、XILINX ISE における chipScope, ALTERA Quartus における SignalTap)波形を採集し、かつ分析することである。

3.2.2 ダウンロード

本ボードは JTAG モードと SPI FLASH モードをサポートする。デバッグでは、JTAG モードを使用する。 次に、この二種類のモードについて紹介する。

A JTAG モードで FPGA を設定

ダウンロードするファイルはbitファイルである(ダウンロードスピードが速い)。JTAGインタフェース でロジックをJTAGの内蔵SRAMにダウンロードする。しかし、電源切れの場合に、SRAMの内容を失って、電源 を入れ再設定される。勉強とデバッグの時に、JTAGモードを推奨する。デバッグ成功後、SPI FLASHモード



でロジックを設定チップにダウンロードする。

- 1) ISEソフトウェアでbit設定ファイルを生成する。(bit接尾辞)
- 2) ダウンロードケーブルを開発ボードのJTAGコネクタJ7に接続する。
- 3) 開発ボードに電源を入れる。

4) ISEの「Tools」に「iMPACT]をクリックし、JEDファイルを選択してから右側のXILINXアイコンをク

リックし、次に「program」をダブルクリックしダウンロードする。



B SPI FLASHモードでFPGAを設定

ダウンロードするファイルはmcsファイルである(ダウンロードスピードが遅い)。 SPI FLASHモ ードでロジックを設定チップにダウンロードする(本ボードはM25P16を使用している)。電源切れの場 合に、設定チップの内容が失わない。電源を入れ、FPGAが自動的に設定チップのロジックをFPGAにロー ドする。

ISE 10.1を初め、シリアルチップに対する直接のプログラミングをサポートしないが、JTAGで間接 にプログラミングする。(上図と下図の右側のアイコンの色を注意してください)





Mcsファイルを生成する方法と、spi flashプログラムプロジェクトを構築する方法は次を参照ください。

3.2.3 SPI Flash ダウンロード

環境: ▶ISE10.1バージョン以上。 ▶ダウンロードケーブル ▶Windows XP ▶開発ボード ▶M2:M0=001、MASTER SPI MODE(本ボードではハードウェア回路上設定済み) ▶VS2:VS0=111(本ボードではハードウェア回路上設定済み)

ステップ1、iMPACTを起動

Xilinx ISE Design Suite 10.1->ISE->Accessories->iMPACT、起動後の画面



ステップ2、mcsファイルを生成

上記ウィンドウの左側「Flows」の一番下にある「PROM File Formatter」をダブルクリックして、下記の 様に設定する:



低価格、高品質が不可能?

日昇テクノロジーなら可能にする

IMPACT		×
Eile Edit View Operations Options Output	Debug <u>Wi</u> ndow <u>H</u> elp	
🔁 🖥 🗶 🛱 🗶 🖬 🗶 🖬	🔅 : 🚔 🖶 O 😺 😽	
Flows ×		
Boundary Scan SlaveSerial SlaveSerial Direct SPI Configuration SystemACE PROM File Formatter	I wank to target a Xilmx PR0M Generic Parallel PR0M 3rd-Party SPI PR0M PR0M Supporting Multiple Design Versions:	
Modes MPACT Processes X	PROM File Format MCS TEK UFP (C" format) EX0 BIN ISC HEX Swap Bits	
	Checksum Fill Value (2 Hex Digits): FF	
	PROM File Name: demo_spi_flash	
	Location: E:/ Browse	×
· · · · · · · · · · · · · · · · · · ·		
Operations	< <u>B</u> ack Next> Cancel	<u>nttp://www.xilinx.com</u>
X Welcome to iNPACT iNPACT Version: 10.1.03		×

「Next」をクリックして、提示された画面で「Select SPI PROM Density (bits)」の所のリストから「16M」 を選択する。本開発ボードで搭載しているSPI FlashはM25P16で容量は16M bitsである。



「Next」をクリックすると:





低価格、高品質が不可能?

日昇テクノロジーなら可能にする



「OK」をクリックする:



低価格、高品質が不可能?

日昇テクノロジーなら可能にする

😺 іМРАСТ				
Add Device			l	
Look jn:	: 🗀 demoprj	★ È [™] ■.		
<u>i</u>	demo_spi_flash.bit			
My Recent Documents				
Desktop			S	
Mu Desumente				
my Documents				
Mu Computer				
My Network	File name: der	no spi flash.bit 🔻 Open		
Places	Files of type: FP	GA Bit Files (*.bit)		
			-	
			_	http://www.xilipx.com
Operations				
× PI	ROM Name : 16M	PROM Size : 16777216 Bytes		
// ***]	BATCH CMD : set.	END of Report Attribute -design -attr name -value "O"		
nscript				×
⊖ Output Em	ror Warning			No Cable Connection No File Open
hitファ	イルを選択	マーマ「Onen」をクリックす	Z·	
	1/* 2 201/		· ·	
Eile Edit View O	Operations Window Hel	p		
Elows	à â X # % ∷ ×			
Boundary S	ican	PROM File Formatter		
SelectMAP				
Direct SPI C	onfiguration Configuration			
SystemACE	Formatter			
		2.00 % ruii x03s1200e demo_spi_flash.bt		
Modes				
iMPACT Processes	x	Add D	evice 🛛	
Generate File	N GIO.		would you like to add another device rile to Data Stream: 0	
			Yes No	
				http://www.xilinx.com
Operations		PROM File Formatter		
X INFO: iMH	PACT:501 - '1':	Added Device xc3s1200e successfully.		<u>^</u>
Add one	device.			
Output Em	or Warning			2
			PROM File Ceneration Target SPI DROM	1 3 841 184 Bits used File: demo. spi flash in Location: E:/demonri/

「Would you like to add another device file to Data Stream:0」の提示画面で「No」をクリックする:



低価格、高品質が不可能?

日昇テクノロジーなら可能にする



左側の「iMPACT Process」の「Generate File...」をダブルクリックして、mcsファイル生成画面が出る:



低価格、高品質が不可能?

日昇テクノロジーなら可能にする

Ele Edit View Ogerations Window Help	
Flows X SelectMAP Desktop Configuation SplexeSPI configuation SPI	http://www.xilinx.com
Operations 😵 PROM File Formatter	
<pre>X Using user-specified prom size of 2048K Writing file "E:/demoprj//demo_spi_flash.mcs". Writing file "E:/demoprj//demo_spi_flash.prm". Output Etro: Warning</pre>	

これで、SPI FLASHにダウンロード用のmcsファイルが生成される。

ステップ3、bitファイル及びmcsファイルをSPI FLASHにダウンロード

左側の「Flows」の「Boundary Scan」をダブルクリックする。 🛱 アイコンが緑色になる。このアイコン をクリックする。

😵 iMPACT - [Boundary Scan]			
😼 File Edit View Operations Output Deb	ug Window Help		l d l
🔢 🖉 🖬 🖓 🖬 🗶 🗄 🗶	Assign New Configuration File	? 🔀	
Flows			
Boundary Scan	Look in: E:/demoprj/		
- BSlaveSerial			
	demo_spi_flash.bit		
Desktop Configuration			
Direct SPI Configuration			
SystemACE			
PRUM File Formatter			
· · · · · · · · · · · · · · · · · · ·		Upen	
	File type: All Design Files (*.bit *.rbt *.nky *.isc *.bsd)	Cancel	
	in the second	Lancel All Bypass	
Modes	🔿 None		
×	Enable Programming of SPI Flash Devi	ce Attached to this FPGA	
done.	Enable Programming of PPI Flash Douis	es Attached to this EPGA	-
PROGRESS_END - End Opera		CE Allached to this HT UA	
Elapsed time = 0 set			
// *** BATCH CMD : ident:	ifyMPM		
sci			~
			2
		[C an	investion Distance Cable UCD 6 Mile Luck ha

bitファイルを選択する。また下側にある



None

- Enable Programming of SPI Flash Device Attached to this FPGA
- C Enable Programming of BPI Flash Device Attached to this FPGA

デフォルトでは「None」が選択されているが、SPI FLASHにダウンロードするので、必ず「Enable Programming of SPI Flash Device Attached to this FPGA」を選択必要。「Open」をクリックする:

😺 iMPACT - [Bo	undary Scan]		
Add PROM File		? 🔀	
Look in:	🗁 demoprj 💌	- 🗈 📸 🎟	
à	demo_spi_flash.mcs		operations
My Recent			
Desktop			e sh bit
My Documents			
My Computer			
My Network	File name:	▼ Open	the bitstream stored in memory, 🔄
Places	Files of type: MCS Files (*.mcs)	✓ Cancel	
<u>ت</u>			
			×
	r_)_waming_)		Configuration Platform Cable USB 6 MHz usb-hs





「FPGA SPI Flash Association」の提示画面でボードに搭載されているSPI FLASH型番を選択する。ここでは「M25P16」を選択して、「OK」をクリックする:



Device Programming Properties - Device 1 F	Programming Properties	
Category		
🖻 Boundary-Scan		
Device 1 (FPGA, xc3s1200e)	Property Name	Value
Device 1 (Attached FLASH, M25P16)		
	Venry	
	FPGA Device Specific Programming Properties	
	Assert Cable INIT during programming	
「OK」をクリックする:	OK	Cancel Apply Help
ShillDACT IBoundary Secol		
File Edit View Operations Output Debug Window Help		
		الالا
Flows X IMPACT Processes	× Pickt aliak derige to select executions	
Boundary Scan Available Operations are:	FLASH	
SelectMAP		
BDesktop Configuration		
Blank Check	4	
SystemACE	xc3s1200e	
□ IPHOM File Formatter	demo_spi_flash.bit	

퉳 Boundary Scan

Selected part: M25P16 // *** BATCH CMD : attachflash -position 1 -spi "M25P16" // *** BATCH CMD : assignfiletoattachedflash -position 1 -file "E:/demoprj/demo_spi_flash.mcs" // *** BATCH CMD : attachflash -position 1 -spi "M25P16"

Operations

Modes

<

Output Error Warning

^

>



「FLASH」アイコンをクリックして緑色になる。右クリックして下記画面が表示する:



ダウンロードの進捗が表示される。 ダウンロード完了したら、下記画面が表示する:



🔯 iMPACT - [Boundary Scan]			
虆 File Edit View Operations Output D	ebug Window Help		
🛛 🤔 🔚 🖌 🖻 🕼 🗶 🔡 🏹	H 🗘 H 🍀 😂 🗘 🖓	Ø N ?	
Flows × Image: State Serial State Serial Select MAP Image: Select MAP	MPACT Processes Available Operations are: ➡ Program ➡ Verify ➡ Erase ➡ Blank Check ➡ Readback ➡ Get Device Checksum	Right click device to select operations TDI xc3s1200e demo_spi_flash.bit TDO Program Succeeded	
Modes	Operations	😵 Boundary Scan	
<pre>X '1': Verification compl INFO: MPACT - '1': Chec '1': Programmed success PROGRESS_END - End Oper</pre>	eted. Sking done pindone. Sfully. Sation.		^
Elapsed time = 57 s	sec.		~
			>
- Uutput Error Warning		Configuration Platform Cable USB 6	MHz usb-hs

これまで、SPI FLASHダウンロード完了。ボードを再起動すると、SPI FLASHからデータを読んでFPGAを設定する。



4、開発ボード回路説明

4.1 電源回路

本開発ボードは外部から 5V の電源を投入し、LM1085-3.3V レギュレーターを通じて 3.3v 電源を出力する。主に FPGA IO インタフェース、メモリーSDRAM、シリアル設定デバイス、リセット回路用の電源とする。 また 5V は AMS1117-2.5/1.2V のレギュレータに与え、出力電源は FPGA コアとフェーズロック・ループに 使用される。

SW2 はセルフロック電源スイッチ、D 2 は電源表示灯、D 1 は電力の逆接続を防ぐ指示灯である。

同時に、電力システムの安定性を向上させるために、電源入出力の部分に高品質の電解及びタンタルコ ンデンサを設計した。下図は回路図である。



4.3 リセット回路

ローレベルリセット、普通のボタンとしても使える。XC3S500EPQ208の Pin184 に接続する。下図が回路 図である。





4.4 JTAG インタフェース回路

FPGA をダウンロードし、デバッグする時使用される。ダウンロードスピードは早くて、Chipscope サンプリングもできる。電源切れの場合に、FPGA のロジックが失う。デバッグの時は、JTAG モードで Platform Cable と一緒にダウンロードしデバッグすることを推奨する。



注意:ロジック内部がプルアップされたため、TMSとTDIはプルアップ抵抗を使用しない。

4.5 SPI FLASH インタフェース回路

SPI FLASH インタフェース、M25P16 設定チップ、M[2:0]設定モードの選択、VS[2:0] SPI Flash 種類の 選択、再設定キーなどがある。ボードに電源を入れ、または再設定キーSW2 を押すと、FPGA が自動的に M25P16 のロジックをロードし FPGA を再設定する。

SPI FLASH ダウンロードモードでは、M25P16 チップにプログラミングする。本ボードは 16Mbit チップ を設定し、FPGA ロジックと SOPC プログラムを設定チップに格納する。詳細: a JTAG インタフェースで間接 に M25P16 をプログラミングする。b 電源を入れ、又は再設定した後、FPGA が自動的に M25P16 のロジックを ロードし FPGA を再設定する。c FPGA がロジックを実行し、動作する。以下は回路図である。

			V 1 L
	/PROG	B 1	PROCR
			FROGB
M2	IP81	81	
M1	1084	84	IP_L10N_2/M2/GCLK1
M0	1086	86	
			10_L12P_2/M0
SS_B	1055	55	10 1018 2/050 B
	1056	56	
	IO60	60	
MOSI	IO61	61	
MISO	1087	87	
SCK	IO103	103	
	DONE	104	IO_L1/N_2/CCLK
			DONE





4.6 LED 回路

LED 二つ、ハイレベルで点灯する。LED 試験に利用する。例えば、点灯、点滅など。下図が回路図である。





4.7 メモリーSDRAM 回路

開発ボードのメモリーは SDRAM 、タイプは HY57V641620FTP、64Mbit である。SDRAM アドレス線は A0 ~A11、データワイドビットは 16 ビット、0.1 μ F のコンデンサでフィルタ処理を行って、チップが安定に動作する。



NE	ET "sdram_ba[0]" LOC = P103;
NE	ET "sdram_ba[1]" LOC = P104;
NE	ET ″sdram_cas_n″LOC = P99 ;
NE	ET ″sdram_cke″ LOC = P117;
NE	ET "sdram_clk" LOC = P116;
NE	ET ″sdram_dq[0]″ LOC = P151;
NE	ET ″sdram_dq[1]″ LOC = P150;
NE	ET "sdram_dq[2]" LOC = P149;
NE	ET "sdram_dq[3]" LOC = P147;
NE	ET "sdram_dq[4]" LOC = P146;
NE	ET "sdram_dq[5]" LOC = P145;
NE	ET ″sdram_dq[6]″ LOC = P144;
NE	ET "sdram_dq[7]" LOC = P143;
NE	ET "sdram_dq[8]" LOC = P114;
NE	ET "sdram_dq[9]" LOC = P113;
NE	ET "sdram_dq[10]" LOC = P112;
NE	ET ″sdram_dq[11]″ LOC = P110;



NET "sdram_dq[12]" LOC = P108; NET "sdram_dq[13]" LOC = P107; NET "sdram_dq[14]" LOC = P106; NET "sdram_dq[15]" LOC = P105; NET "sdram_dqm[0]" LOC = P142; NET "sdram_dqm[1]" LOC = P115; NET "sdram_ras_n" LOC = P101; NET "sdram_sa[0]" LOC = P139; NET "sdram_sa[1]" LOC = P138; NET "sdram_sa[2]" LOC = P137; NET "sdram_sa[3]" LOC = P135; NET "sdram_sa[4]" LOC = P134; NET "sdram_sa[5]" LOC = P133; NET "sdram_sa[6]" LOC = P128; NET "sdram_sa[7]" LOC = P127; NET "sdram_sa[8]" LOC = P120; NET "sdram_sa[9]" LOC = P119; NET "sdram_sa[10]" LOC = P141; NET "sdram_sa[11]" LOC = P118; NET "sdram_we_n" LOC = P97 ; NET "sdram_cs_n" LOC = P102;

4.8 外部拡張 IO

開発ボードは三つのピン配列で外部に接続する。連続ピンはダブル列、40pin、ピンの間隔は2.54mm ピッチ、ピンの定義は下図の通り:



以上。