電子政府推奨暗号回路
および制御ソフトウェア
解説書

第 0.1 版

2010 年 2 月 23 日

目次

１．概要

電子政府推奨暗号回路および制御ソフトウェア解説書（以下、本解説書）は、FPGA ボードに実装した電子政府推奨暗号回路（以下、暗号回路）と FPGA ボード上の USB インターフェースを通して暗号回路を制御するソフトウェアについて解説したものである。

電子政府推奨暗号回路は、平成 20 年度に開発した Virtex-5 FPGA ボードの SASEBO-GⅡ-AES 暗号 FPGA ボード上に実装する。実装する暗号回路は、ブロック暗号 2 種類、ストリーム暗号、ハッシュ関数それぞれ 1 種類ずつの計 4 種類である。以下に、実装する暗号回路を示す。

① ブロック暗号：AES
② ブロック暗号：CMAC
③ ストリーム暗号：MUGI
④ ハッシュ関数：SHA-256 による HMAC

これらの暗号回路は、独立した回路で同時に使用することはできず、変更するためには、FPGA を書き換える必要がある。

制御ソフトウェアに関しても、暗号回路の種類ごとに 4 種類ある。

1.1 SASEBO-GⅡ-AES暗号FPGAボード

SASEBO-GⅡ-AES 暗号 FPGA ボードは、2 個の FPGA と 1 個の USB ターゲットコントローラが実装されている FPGA ボードである。FPGA は、FPGA1 と FPGA2 の 2 個が実装されているが、暗号回路自体は FPGA1 に実装される。FPGA2 は、USB コントローラと FPGA1 とのインターフェースを行う回路が実装され、すべての暗号回路で共通で使用する。図 1 に FPGA2 と FPGA1 および USB コントローラ間の接続を示す。

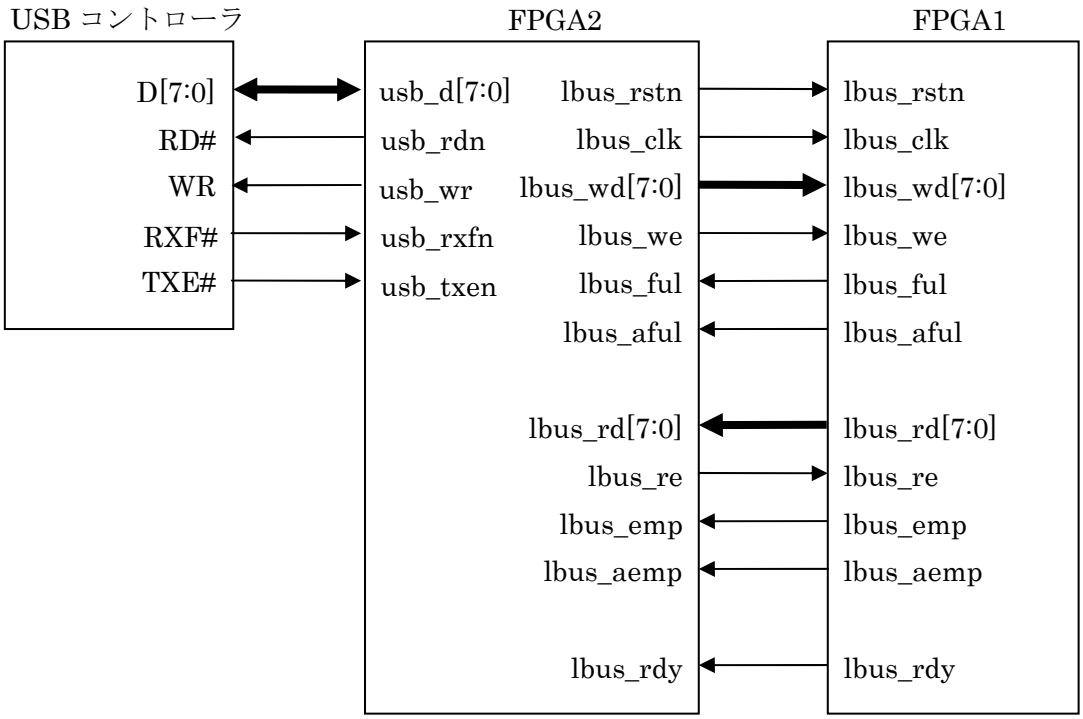


図 1　FPGA2 と FPGA1、USB コントローラの接続

USB コントローラは、マイクロチップ社の FT2232D を使用し、USB コントローラと FPGA2 は、FT2232D のを FIFO モードと呼ばれるモードで接続する。FT2232D の詳細については、データシートを参照のこと。

FPGA2 と FPGA1 は、専用のローカルバスを規定して接続する。SASEBO-GⅡボードの詳細については、SASEBO-GⅡ-AES 暗号 FPGA ボード仕様書を参照のこと。

1.2　制御ソフトウェア

制御ソフトウェアは、Windows PC から、USB インターフェースを通して、暗号回路を制御するためのサンプルプログラム集である。プログラムは、C#言語で記述され開発ツールは、Microsoft Visual C# 2008 Express Edition を使用している。

サンプルプログラムは、以下の 4 種類である。

・SASEBO_AES_sample
・SASEBO_CMAC_sample
・SASEBO_MUGI_sample
・SASEBO_SHA256_sample

２．ローカルバス仕様

FPGA1 と FPGA2 は、専用のローカルバスにてインターフェースを行う。FPGA2 が常にバスマスタで FPGA1 からローカルバスのアクセスを行うことはない。

2.1　信号名と機能

ローカルバスは、すべて単方向の信号線である。の表 1 にローカルバスの信号名と機能を示す。表 1 に示した以外にも FPGA1 と FPGA2 の間には接続されている信号線は存在するが、今回の仕様では使用していない。

表 1　ローカルバス信号と機能

信号名	信号方向	アクティブ	機能
lbus_rstn	FPGA2→FPGA1	Low	ローカルバス・リセット。ハードウェアリセット入力で、この信号が'0'のときに FPGA1 がリセットされる。
lbus_clk	FPGA2→FPGA1	Rise	ローカルバス・クロック。ローカルバスの同期クロックでローカルバスのすべての信号はこのクロックの立ち上りエッジに同期して動作する。また、このクロックは、FPGA1 内のシステムクロックとしても使用される。
lbus_wd[7:0]	FPGA2→FPGA1	－	ローカルバス・ライトデータ。FPGA1 にデータを書き込むためのデータ線である。
lbus_we	FPGA2→FPGA1	High	ローカルバス・ライト。 " lbus_wd[7:0]"上のデータを FPGA1 内に書き込むためのストローブ信号である。
lbus_ful	FPGA2←FPGA1	High	ローカルバス・フル。FPGA1 内にある書き込み用 FIFO がいっぱいであることを示す。この信号が'1'のときは、ローカルバスに書き込みを行ってもデータの書き込みは行われない。
lbus_aful	FPGA2←FPGA1	High	ローカルバス・オールモストフル。FPGA1 内の書き込み用 FIFO があと 1 回の書き込みでいっぱいになること

			を示す。この信号が'1'の状態で書き込みを行うと" lbus_ful"が'1'になる。
lbus_rd[7:0]	FPGA2←FPGA1	－	ローカルバス・リードデータ。FPGA1 内部の読み出し用 FIFO からデータを読み出すためのデータ線である。
lbus_re	FPGA2→FPGA1	High	ローカルバス・リード。FPGA1 内の読み出し用 FIFO からデータを読み出すためのストローブ信号である。
lbus_emp	FPGA2←FPGA1	High	ローカルバス・エンプティ。FPGA1 内にある読み出し用 FIFO に読み出すデータがないことを示す。
lbus_aemp	FPGA2←FPGA1	High	ローカルバス・オールモストエンプティ。FPGA1 内にある読み出し用 FIFO に読み出すデータが 2 個以下であることを示す。
lbus_busy	FPGA2←FPGA1	High	ローカルバス・ビジー。ローカルバスが使用中で、書き込み、読み出し動作が出来ないことを示す。

2.2 バスの動作

ローカルバスは、同期型バスで 16 ビットのアドレス空間を持ち 1 回のアクセスで 16 ビットデータの読み書きが可能である。ローカルバスのバス幅は、8 ビットであるため 1 回のアクセスに 5 サイクルを要する。表 2 に FPGA2 から見た読み出しサイクル、表 3 に FPGA2 からみた書き込みサイクルの詳細を示す。

表 2　読み出しサイクル

バスサイクル	バス情報	信号方向
サイクル 1	読み出しコマンド出力	FPGA2 →　FPGA1
サイクル 2	上位アドレス出力	FPGA2 →　FPGA1
サイクル 3	下位アドレス出力	FPGA2 →　FPGA1
サイクル 4	上位データ入力	FPGA2 ←　FPGA1
サイクル 5	下位データ入力	FPGA2 ←　FPGA1

表 3　書き込みサイクル

バスサイクル	バス情報	信号方向
サイクル 1	読み出しコマンド出力	FPGA2 →　FPGA1
サイクル 2	上位アドレス出力	FPGA2 →　FPGA1
サイクル 3	下位アドレス出力	FPGA2 →　FPGA1
サイクル 4	上位データ入力	FPGA2 →　FPGA1
サイクル 5	下位データ入力	FPGA2 →　FPGA1

読み出しと書き込みは、1 サイクル目に出力するコマンドにより決定する。各コマンドコードは、表 4 の通りである。

表 4　コマンドコード

コマンド名	コード
ローカルバス・リード	0x00
ローカルバス・ライト	0x01

2.3　ローカルバス・タイミング

FPGA2 からの各ローカルバス信号は、”lbus_clk”に同期して出力される。図 2 に書き込みタイミング、図 3 に読み出しタイミングを示す。

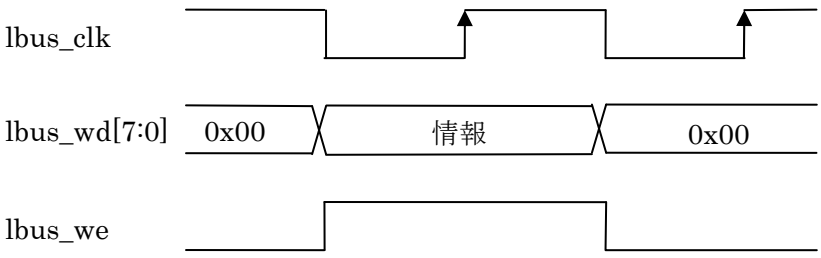


図 2　ローカルバス 書き込みタイミング

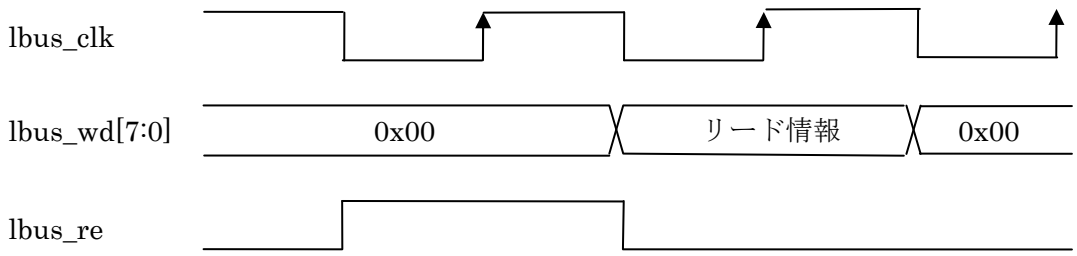


図 3　ローカルバス 読み出しタイミング

３．FPGA1 内部構成

実際に暗号回路が実装される FPGA1 は、各暗号回路により対応した部分とローカルバス・インターフェース回路からなる。暗号回路は、搭載する暗号種類により異なるが、ローカルバス・インターフェース回路は、一部を除いて暗号回路が代わっても共通で使用できる。図 4 に FPGA1 の概略ブロック図を示す。

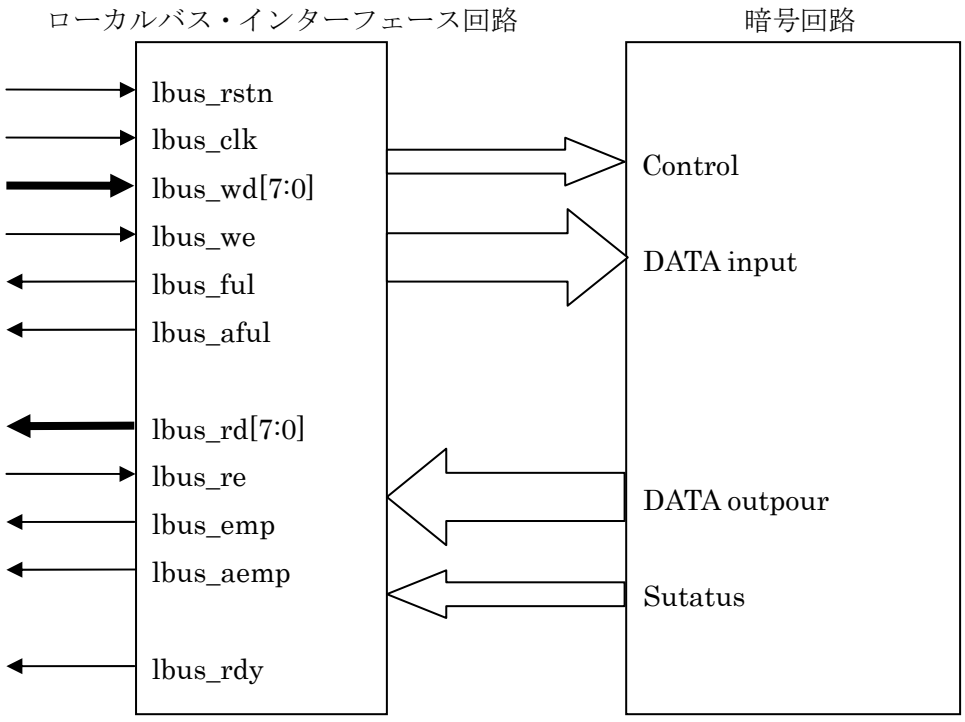


図 4　FPGA1 概略ブロック図

４．ブロック暗号 AES

AES 暗号回路は、NIST Special Publication 800-38A の仕様書に則った AES モードに対応する暗号回路である。共通鍵は、128 ビット、192 ビットおよび 256 ビットのすべてをサポートし、AES 暗号回路は、以下の 5 種類をサポートする。

・ECB（Electronic Codebook）モード
・CBC（Cipher Block Chaining）モード
・CFB（Cipher Feedback）モード
・OFB（Output Feedback）モード
・CTR（Counter）モード

4.1　ファイル構成

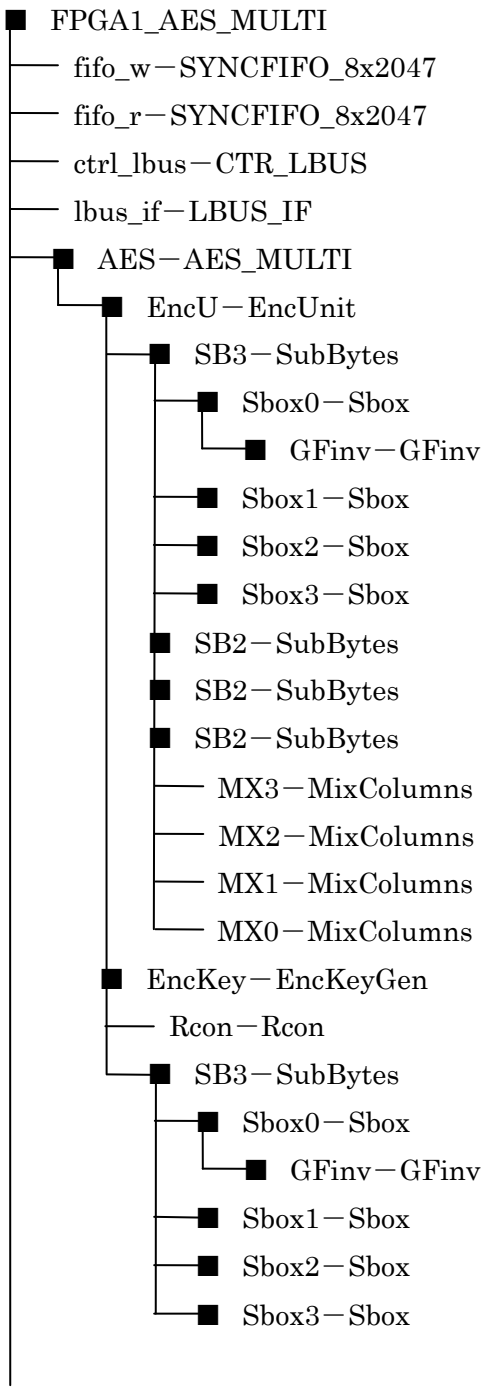
FPGA1_AES を構成するファイルを表 5 に示す。

表 5　FPGA1_AES ファイル構成

ファイル名	種類	内容
FPGA1_AES_MULTI.v	Verilog-HDL	FPGA1_AES の最上位となる HDL ファイル
syncfifo_8x2047.v	Verilog-HDL	8 ビット幅で深さが 2047 ワードの同期型 FIFO を記述した HDL ファイル
ctrl_lbus.v	Verilog-HDL	FPGA2 とのローカルバス･インターフェースを記述した HDL ファイル
lbus_if.v	Verilog-HDL	暗号モジュールの制御を記述した HDL ファイル
AES_MULTI.v	Verilog-HDL	AES 暗号モジュールのトップ HDL ファイル
AES_MULTI_CORE.v	Verilog-HDL	AES 暗号回路の本体を記述した HDL ファイル
FPGA1_AES_MULTI_TB1.v	Verilog-HDL	シミュレーション・ファイル
AES_MULTI_TB1.v	Verilog-HDL	AES モジュールをシミュレーション・ファイル
pin_sasebo_gii_lx50.ucf	構成設定	FPGA1_AES の構成を規定したファイル

4.2　FPGA1_AESモジュール構成

FPGA1_AES の各モジュールは、以下のような構成で接続される。



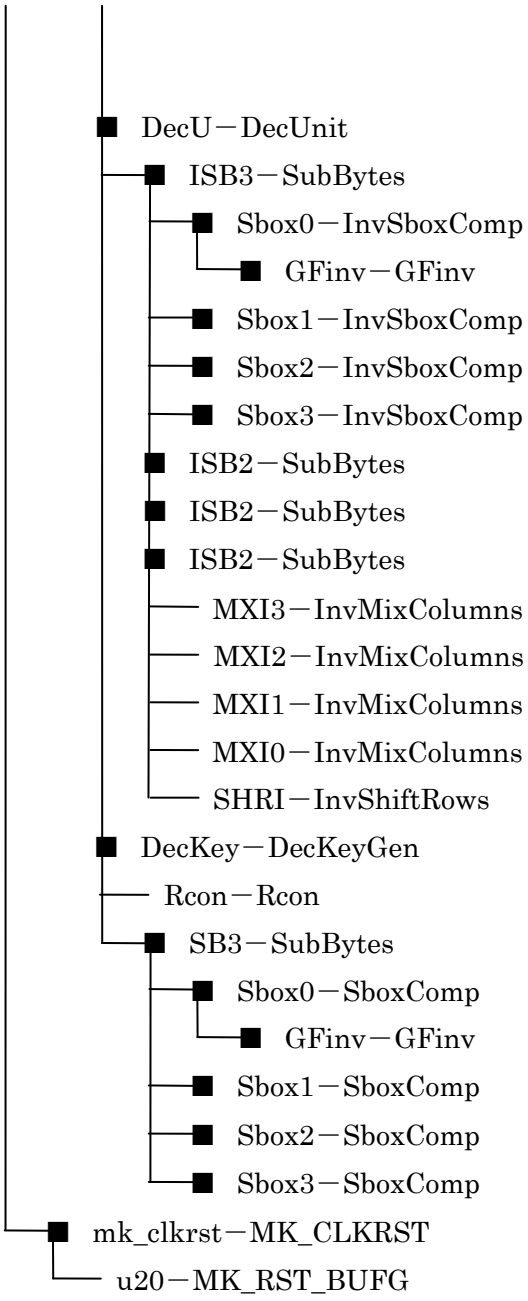


図 5　FPGA1_AES モジュール構成

4.3　暗号回路入出力信号

表 6 に暗号回路の入出力信号を示す。これらの信号は、”lbus_if”モジュールに接続され制御される。

表 6　AES_MULTI モジュール入出力信号

信号名	方向	アクティブ	機能
RSTn	In	Low	リセット入力。 AES モジュールのすべての回路が初期化される。
CLK	In	Rise	システムクロック入力。AES モジュールは、CLK の立ち上りエッジに同期して動作する。
EN	In	High	動作イネーブル信号。”EN”を’1’にすることにより AES モジュールが動作可能となる。
Busy	Out	High	“Busy”が’1’のとき AES モジュールが動作中であることを示めす。
EncDec	In	High/Low	暗号化と復号化の切り替えを行う。’0’のときに暗号化。’1’のときに復号化となる。
Kwidth[1:0]	In	－	使用する鍵のビット幅を指定する。’0’のときに 128 ビット鍵、’1’で 192 ビット鍵、’2’で 256 ビット鍵となる。
AESmode[2:0]	In	－	AES のモードを指定する。’0’で ECB モード、’1’のときに CBC モード、’2’で CFB モード、’3’で OFB モード、’4’で CTR モードとなる。
Kin[127:0]	In	－	鍵入力
Din[127:0]	In	－	平文または暗号文入力。EncDec が’0’のときは平文、’1’のときは暗号文入力となる。
Iin[127:0]	In	－	イニシャルベクター入力
Cin[127:0]	In	－	カウンタ入力
Krdy	In	High	鍵入力が完了し、AES モジュールに鍵生成を開始させる。
Drdy	In	High	データ入力が完了し、AES モジュールに暗号化または復号化を開始させる。
Irdy	In	High	イニシャルベクターを AES モジュールにセットする。
Crdy	In	High	カウンタ値を AES モジュールにセットする。
Dout	Out	－	暗号文または平文出力。
Kvld	Out	High	鍵生成が終了したことを示す。

Dvld	Out	High	Dout に暗号文または、平文が出力されたことを示す。

4.4 アドレスマップ

”lbus_if”モジュール内には、暗号モジュールを操作するためのレジスタ類が複数設けられている。これらのレジスタは、ローカルバス上にマッピングされている。表 7 にアドレスマップを示す。

表 7　AES アドレスマップ

アドレス	名称	R/W	機能
0x0002	コントロール・レジスタ	R/W	AES 回路に対して制御信号を出力するためのレジスタ
0x000C	ENC/DEC レジスタ	W	AES を Encrypt か Decrypt で動作させるかを選択するレジスタ
0x000A	AES モード・レジスタ	W	AES のモードを指定するレジスタ
0x000E	鍵幅レジスタ	W	使用する鍵幅の選択をするレジスタ
0x0100	鍵入力レジスタ 1	W	鍵入力のビット 127-112
0x0102	鍵入力レジスタ 2	W	鍵入力のビット 111-96
0x0104	鍵入力レジスタ 3	W	鍵入力のビット 95-80
0x0106	鍵入力レジスタ 4	W	鍵入力のビット 79-64
0x0108	鍵入力レジスタ 5	W	鍵入力のビット 63-48
0x010A	鍵入力レジスタ 6	W	鍵入力のビット 47-32
0x010C	鍵入力レジスタ 7	W	鍵入力のビット 31-16
0x010E	鍵入力レジスタ 8	W	鍵入力のビット 15-0
0x0120	イニシャルベクター・レジスタ 1	W	イニシャルベクター値入力のビット 127-112
0x0122	イニシャルベクター・レジスタ 2	W	イニシャルベクター値入力のビット 111-96
0x0124	イニシャルベクター・レジスタ 3	W	イニシャルベクター値入力のビット 95-80
0x0126	イニシャルベクター・レジスタ 4	W	イニシャルベクター値入力のビット 79-64
0x0128	イニシャルベクター・レジスタ 5	W	イニシャルベクター値入力のビット 63-48

0x012A	イニシャルベクター・レジスタ 6	W	イニシャルベクター値入力のビット 47-32
0x012C	イニシャルベクター・レジスタ 7	W	イニシャルベクター値入力のビット 31-16
0x012E	イニシャルベクター・レジスタ 8	W	イニシャルベクター値入力のビット 15-0
0x0130	カウンタレジスタ 1	W	カウンタ値入力のビット 127-112
0x0132	カウンタレジスタ 2	W	カウンタ値入力のビット 111-96
0x0134	カウンタレジスタ 3	W	カウンタ値入力のビット 95-80
0x0136	カウンタレジスタ 4	W	カウンタ値入力のビット 79-64
0x0138	カウンタレジスタ 5	W	カウンタ値入力のビット 63-48
0x013A	カウンタレジスタ 6	W	カウンタ値入力のビット 47-32
0x013C	カウンタレジスタ 7	W	カウンタ値入力のビット 31-16
0x013E	カウンタレジスタ 8	W	カウンタ値入力のビット 15-0
0x0140	データ入力レジスタ 1	W	平文／暗号文入力のビット 127-122
0x0142	データ入力レジスタ 2	W	平文／暗号文入力のビット 111-96
0x0144	データ入力レジスタ 3	W	平文／暗号文入力のビット 95-80
0x0146	データ入力レジスタ 4	W	平文／暗号文入力のビット 79-64
0x0148	データ入力レジスタ 5	W	平文／暗号文入力のビット 63-48
0x014A	データ入力レジスタ 6	W	平文／暗号文入力のビット 47-32
0x014C	データ入力レジスタ 7	W	平文／暗号文入力のビット 31-16
0x014E	データ入力レジスタ 8	W	平文／暗号文入力のビット 15-0
0x0180	データ出力レジスタ 1	R	暗号／復号結果のビット 127-112
0x0182	データ出力レジスタ 2	R	暗号／復号結果のビット 111-96
0x0184	データ出力レジスタ 3	R	暗号／復号結果のビット 95-80
0x0186	データ出力レジスタ 4	R	暗号／復号結果のビット 79-64
0x0188	データ出力レジスタ 5	R	暗号／復号結果のビット 63-48
0x018A	データ出力レジスタ 6	R	暗号／復号結果のビット 47-32
0x018C	データ出力レジスタ 7	R	暗号／復号結果のビット 31-16
0x018E	データ出力レジスタ 8	R	暗号／復号結果のビット 15-0
0xFFFC	バージョン・レジスタ	R	FPGA1 のバージョンが 16 進 4 桁で書かれている読み出し専用レジスタ

4.5　レジスタ

4.5.1　コントロール・レジスタ

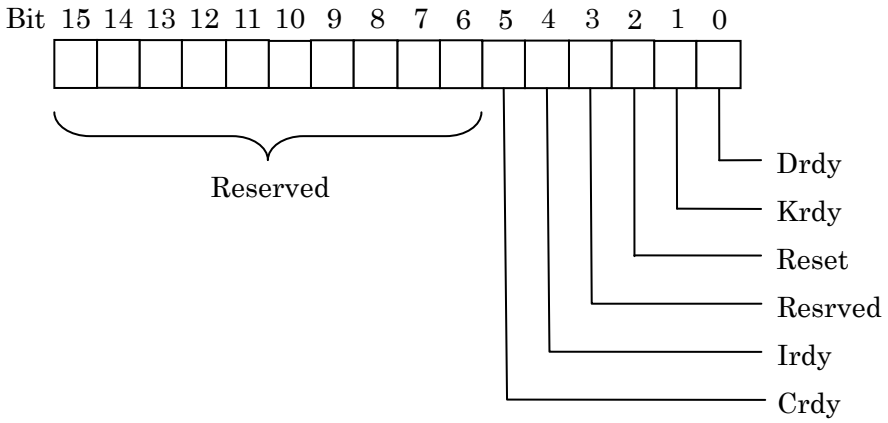


図 6　コントロール・レジスタ ビットアサイン

4.5.2　ENC/DECレジスタ

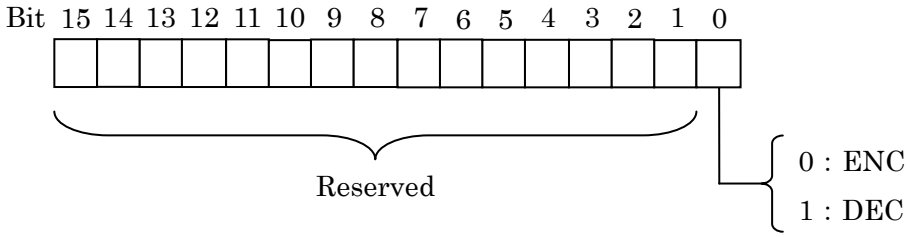


図 7　ENC/DEC レジスタ ビットアサイン

4.5.3　AESモード・レジスタ

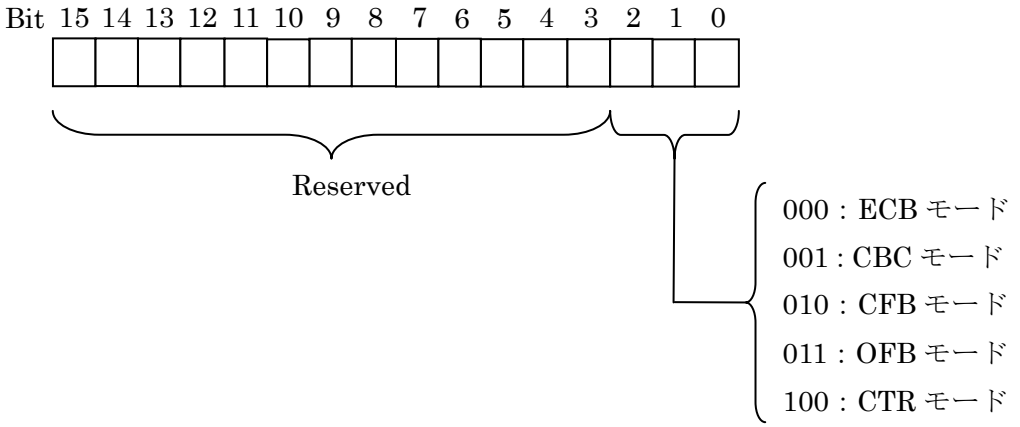


図 8　鍵幅レジスタ ビットアサイン

4.5.4　鍵幅レジスタ

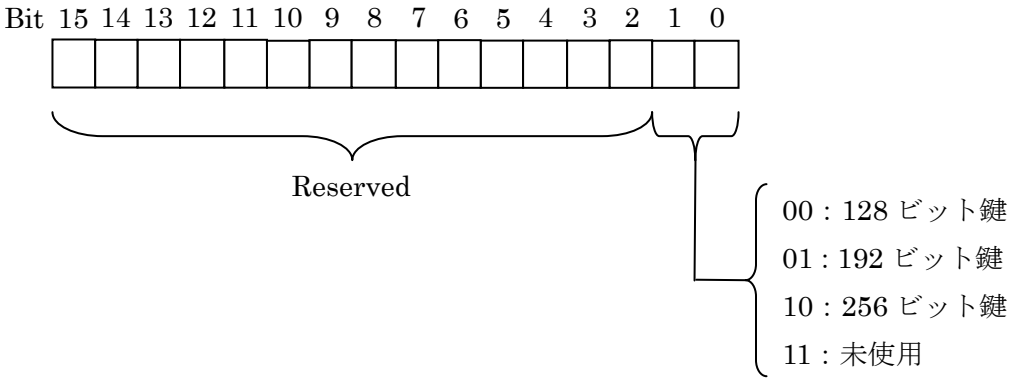


図 9　鍵幅レジスタ ビットアサイン

4.5.4　鍵入力レジスタ 1-8

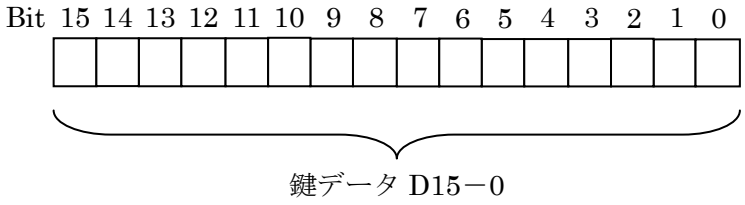


図 10　鍵入力レジスタ 1-8 ビットアサイン

4.5.5　イニシャルベクター入力レジスタ 1-8

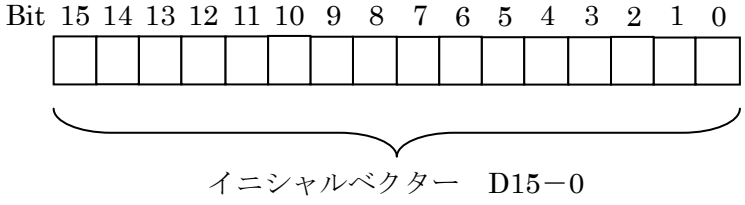


図 11　イニシャルベクター入力レジスタ 1-8 ビットアサイン

4.5.6　データ入力レジスタ 1-8

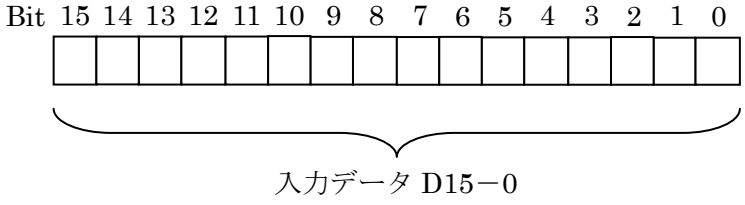


図 12　イニシャルベクター入力レジスタ 1-8 ビットアサイン

4.5.7　データ出力レジスタ 1-8

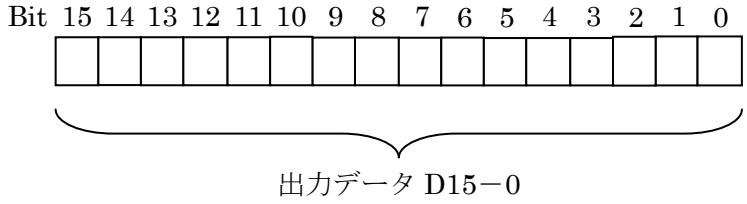


図 13　データ出力レジスタ 1-8 ビットアサイン

4.5.8　FPGA1 バージョン・レジスタ

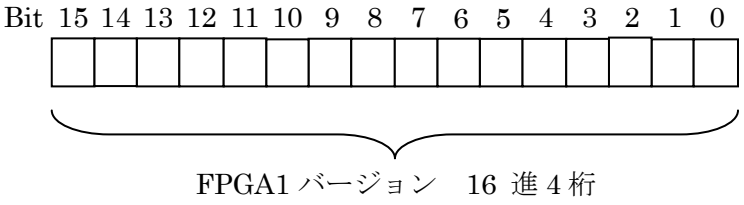


図 14　FPGA1 バージョン・レジスタ ビットアサイン

4.6　動作手順

AES 回路を動作させるには、以下の手順に従って操作を行う。手順は、AES モードおよび鍵幅の違いによりセットするレジスタに違いがある。

4.6.1　ECBモード Encrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに'0'をセット（ECB モード）
手順 3：鍵幅を設定
手順 4：ENC/DEC レジスタに'0'をセット（Encrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット
手順 6："Krdy"入力
手順 7： 鍵幅が 128 ビットならば手順 10 に移動

手順 8： 鍵入力 2

鍵入力レジスタの 1 から 8 に鍵 2 をセット

手順 9：”Krdy”入力

手順 10：データ入力

データ入力レジスタの 1 から 8 に平文をセット。

手順 11：“Drdy”入力

手順 12：暗号文の読み出し

データ出力レジスタの 1 から 8 を読み出だし暗号文を得る

手順 13：終了でなければ手順 10 に戻る

手順 14：終了

4.6.2 ECBモード Decrypt

手順 1： リセット入力

手順 2：AES モード・レジスタに’0’をセット（ECB モード）

手順 3：鍵幅を設定

手順 4：ENC/DEC レジスタに’1’をセット（Decrypt モード）

手順 5： 鍵入力 1

鍵入力レジスタの 1 から 8 に鍵をセット

手順 6：”Krdy”入力

手順 7： 鍵幅が 128 ビットならば手順 10 に移動

手順 8： 鍵入力 2

鍵入力レジスタの 1 から 8 に鍵 2 をセット

手順 9：”Krdy”入力

手順 10：データ入力

データ入力レジスタの 1 から 8 に暗号文をセット。

手順 11：“Drdy”入力

手順 12：平文の読み出し

データ出力レジスタの 1 から 8 を読み出だし平文を得る

手順 13：終了でなければ手順 10 に戻る

手順 14：終了

4.6.3 CBCモード Encrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに'1'をセット（CBC モード）
手順 3：鍵幅を設定
手順 4：ENC/DEC レジスタに'0'をセット（Encrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット
手順 6："Krdy"入力
手順 7： 鍵幅が 128 ビットならば手順 12 に移動
手順 8： 鍵入力 2
　　鍵入力レジスタの 1 から 8 に鍵 2 をセット
手順 9："Krdy"入力
手順 10：イニシャルベクター入力
　　イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット
手順 11："Irdy"入力
手順 12：データ入力
　　データ入力レジスタの 1 から 8 に平文をセット。
手順 13："Drdy"入力
手順 14：暗号文の読み出し
　　データ出力レジスタの 1 から 8 を読み出だし暗号文を得る
手順 15：終了でなければ手順 12 に戻る
手順 16：終了

4.6.4 CBCモード Decrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに'1'をセット（CBC モード）
手順 3：鍵幅を設定
手順 4：ENC/DEC レジスタに'1'をセット（Decrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット
手順 6："Krdy"入力
手順 7： 鍵幅が 128 ビットならば手順 12 に移動
手順 8： 鍵入力 2

鍵入力レジスタの 1 から 8 に鍵 2 をセット

手順 9：”Krdy”入力

手順 10：イニシャルベクター入力

イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット

手順 11：”Irdy”入力

手順 12：データ入力

データ入力レジスタの 1 から 8 に暗号文をセット。

手順 13：“Drdy”入力

手順 14：暗号文の読み出し

データ出力レジスタの 1 から 8 を読み出だし平文を得る

手順 15：終了でなければ手順 12 に戻る

手順 16：終了

4.6.5 CFBモード Encrypt

手順 1： リセット入力

手順 2：AES モード・レジスタに’2’をセット（CFB モード）

手順 3：鍵幅を設定

手順 4：ENC/DEC レジスタに’0’をセット（Encrypt モード）

手順 5： 鍵入力 1

鍵入力レジスタの 1 から 8 に鍵をセット

手順 6：”Krdy”入力

手順 7： 鍵幅が 128 ビットならば手順 12 に移動

手順 8： 鍵入力 2

鍵入力レジスタの 1 から 8 に鍵 2 をセット

手順 9：”Krdy”入力

手順 10：イニシャルベクター入力

イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット

手順 11：”Irdy”入力

手順 12：データ入力

データ入力レジスタの 1 から 8 に平文をセット。

手順 13：“Drdy”入力

手順 14：暗号文の読み出し

データ出力レジスタの 1 から 8 を読み出だし暗号文を得る

手順 15：終了でなければ手順 12 に戻る

手順 16：終了

4.6.6　CFBモード Decrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに'2'をセット（CFB モード）
手順 3：鍵幅を設定
手順 4：ENC/DEC レジスタに'1'をセット（Decrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット
手順 6："Krdy"入力
手順 7： 鍵幅が 128 ビットならば手順 12 に移動
手順 8： 鍵入力 2
　　鍵入力レジスタの 1 から 8 に鍵 2 をセット
手順 9："Krdy"入力
手順 10：イニシャルベクター入力
　　イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット
手順 11："Irdy"入力
手順 12：データ入力
　　データ入力レジスタの 1 から 8 に暗号文をセット。
手順 13："Drdy"入力
手順 14：暗号文の読み出し
　　データ出力レジスタの 1 から 8 を読み出だし平文を得る
手順 15：終了でなければ手順 12 に戻る
手順 16：終了

4.6.7　OFBモード Encrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに'3'をセット（OFB モード）
手順 3：鍵幅を設定
手順 4：ENC/DEC レジスタに'0'をセット（Encrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット

手順 6：”Krdy”入力
手順 7： 鍵幅が 128 ビットならば手順 12 に移動
手順 8： 鍵入力 2
　　鍵入力レジスタの 1 から 8 に鍵 2 をセット
手順 9：”Krdy”入力
手順 10：イニシャルベクター入力
　　イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット
手順 11：”Irdy”入力
手順 12：データ入力
　　データ入力レジスタの 1 から 8 に平文をセット。
手順 13：“Drdy”入力
手順 14：暗号文の読み出し
　　データ出力レジスタの 1 から 8 を読み出だし暗号文を得る
手順 15：終了でなければ手順 12 に戻る
手順 16：終了

4.6.8　OFBモード Decrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに’3’をセット（OFB モード）
手順 3：鍵幅を設定
手順 4：ENC/DEC レジスタに’1’をセット（Decrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット
手順 6：”Krdy”入力
手順 7： 鍵幅が 128 ビットならば手順 12 に移動
手順 8： 鍵入力 2
　　鍵入力レジスタの 1 から 8 に鍵 2 をセット
手順 9：”Krdy”入力
手順 10：イニシャルベクター入力
　　イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット
手順 11：”Irdy”入力
手順 12：データ入力
　　データ入力レジスタの 1 から 8 に暗号文をセット。
手順 13：“Drdy”入力

手順 14：暗号文の読み出し
　　データ出力レジスタの 1 から 8 を読み出だし平文を得る
手順 15：終了でなければ手順 12 に戻る
手順 16：終了

4.6.9　CTRモード Encrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに'4'をセット（CTR モード）
手順 3：鍵幅を設定
手順 4：ENC/DEC レジスタに'0'をセット（Encrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット
手順 6："Krdy"入力
手順 7： 鍵幅が 128 ビットならば手順 12 に移動
手順 8： 鍵入力 2
　　鍵入力レジスタの 1 から 8 に鍵 2 をセット
手順 9："Krdy"入力
手順 10：カウンタ入力
　　カウンタ入力レジスタの 1 から 8 にカウンタ値をセット
手順 11："Crdy"入力
手順 12：データ入力
　　データ入力レジスタの 1 から 8 に平文をセット。
手順 13："Drdy"入力
手順 14：暗号文の読み出し
　　データ出力レジスタの 1 から 8 を読み出だし暗号文を得る
手順 15：終了でなければ手順 12 に戻る
手順 16：終了

4.6.4　CTRモード Decrypt

手順 1： リセット入力
手順 2：AES モード・レジスタに'4'をセット（CTR モード）
手順 3：鍵幅を設定

手順 4：ENC/DEC レジスタに'1'をセット（Decrypt モード）
手順 5： 鍵入力 1
　　鍵入力レジスタの 1 から 8 に鍵をセット
手順 6："Krdy"入力
手順 7： 鍵幅が 128 ビットならば手順 12 に移動
手順 8： 鍵入力 2
　　鍵入力レジスタの 1 から 8 に鍵 2 をセット
手順 9："Krdy"入力
手順 10：カウンタ値入力
　　カウンタ入力レジスタの 1 から 8 にカウンタ値をセット
手順 11："Crdy"入力
手順 12：データ入力
　　データ入力レジスタの 1 から 8 に暗号文をセット。
手順 13："Drdy"入力
手順 14：暗号文の読み出し
　　データ出力レジスタの 1 から 8 を読み出だし平文を得る
手順 15：終了でなければ手順 12 に戻る
手順 16：終了

５．ブロック暗号 CMAC

CMAC 暗号回路は、NIST Special Publication 800-38B の仕様書に則った暗号回路である。共通鍵は、128 ビット、192 ビットおよび 256 ビットのすべてをサポートする。

5.1　ファイル構成

表 7　FPGA1_CMAC ファイル構成

ファイル名	種類	内容
FPGA1_CMAC.v	Verilog-HDL	FPGA1_CMAC の最上位となる HDL ファイル
syncfifo_8x2047.v	Verilog-HDL	8 ビット幅で深さが 2047 ワードの同期型 FIFO を記述した HDL ファイル

ctrl_lbus.v	Verilog-HDL	FPGA2 とのローカルバス・インターフェースを記述した HDL ファイル
lbus_if.v	Verilog-HDL	暗号モジュールの制御を記述した HDL ファイル
CMAC.v	Verilog-HDL	CMAC 暗号モジュールのトップ HDL ファイル
FPGA1_CMAC_TB1.v	Verilog-HDL	FPGA1_CMAC のテストベンチを記述した HDL ファイル
CMAC_TB1.v	Verilog-HDL	CMAC 回路単体のテストベンチを記述した HDL ファイル
pin_sasebo_gii_lx50.ucf	構成設定	FPGA1_CMAC の構成を規定したファイル

5.2　FPGA1_CMACモジュール構成

FPGA1_CMAC の各モジュールは、以下のような構成で接続される。

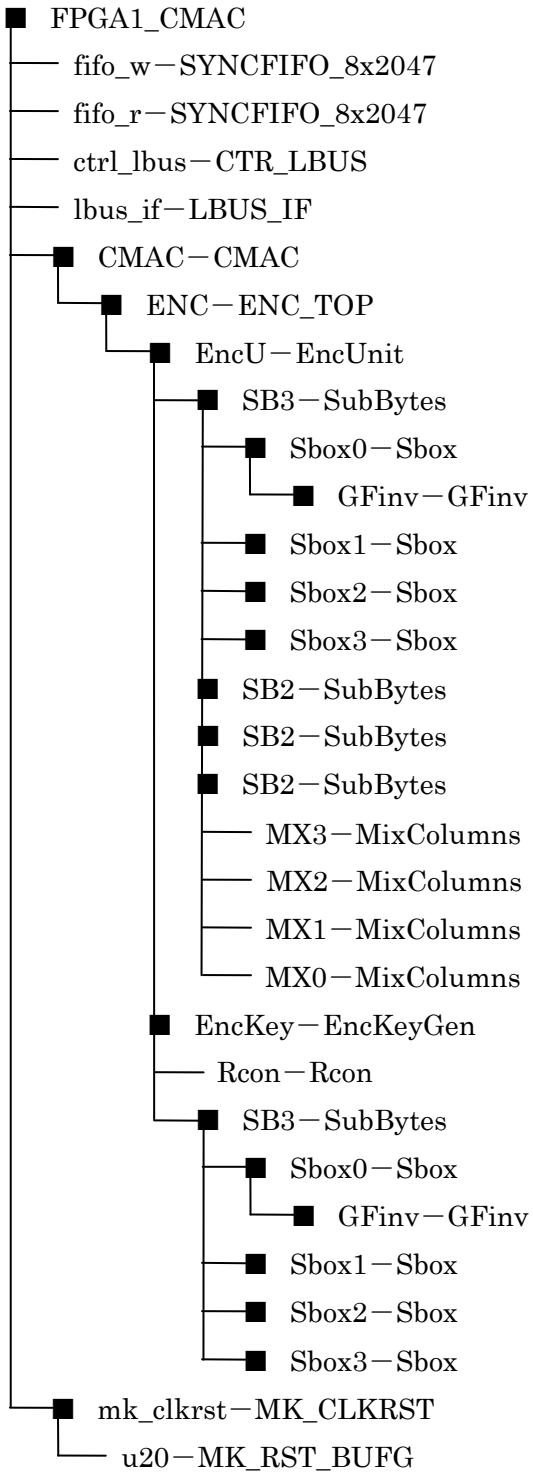


図 15　FPGA1_CMAC モジュール構成

5.3 CMACモジュール入出力信号

表 8 に暗号回路の入出力信号を示す。これらの信号は、”lbus_if”モジュールに接続され制御される。

表 8　CMAC モジュール入出力信号

信号名	方向	アクティブ	機能
RSTn	In	Low	リセット入力。　CMAC モジュールのすべての回路が初期化される。
CLK	In	Rise	システムクロック入力。CMAC モジュールは、CLK の立ち上りエッジに同期して動作する。
EN	In	High	動作イネーブル信号。”EN”を’1’にすることにより AES モジュールが動作可能となる。
Busy	Out	High	“Busy”が’1’のとき CMAC モジュールが動作中であることを示めす。
Kwidth[1:0]	In	－	使用する鍵のビット幅を指定する。’0’のときに 128 ビット鍵、’1’で 192 ビット鍵、’2’で 256 ビット鍵となる。
Kin[127:0]	In	－	鍵入力
Din[127:0]	In	－	平文または暗号文入力。EncDec が’0’のときは平文、’1’のときは暗号文入力となる。
Lin[127:0]	In	－	レングス入力
Krdy	In	High	鍵入力が完了し、CMAC モジュールに鍵生成を開始させる。
Drdy	In	High	データ入力が完了し、CMAC モジュールに暗号化または復号化を開始させる。
Irdy	In	High	イニシャルベクターを CMAC モジュールにセットする。
Lrdy	In	High	レングス値を CMAC モジュールにセットする。
Dout	Out	－	暗号文または平文出力。
Kvld	Out	High	鍵生成が終了したことを示す。
Dvld	Out	High	Dout に暗号文または、平文が出力されたことを示す。

5.4 アドレスマップ

”lbus_if”モジュール内には、暗号モジュールを操作するためのレジスタ類が複数設けられている。これらのレジスタは、ローカルバス上にマッピングされている。表 9 にアドレスマップを示す。

表 9　CMAC アドレスマップ

アドレス	名称	R/W	機能
0x0002	コントロール・レジスタ	R/W	AES 回路に対して制御信号を出力するためのレジスタ
0x000E	鍵幅レジスタ	W	使用する鍵幅の選択をするレジスタ
0x0080	レングス・レジスタ 1	W	レングス入力のビット 32-16
0x0082	レングス・レジスタ 2	W	レングス入力のビット 15-0
0x0100	鍵入力レジスタ 1	W	鍵入力のビット 127-112
0x0102	鍵入力レジスタ 2	W	鍵入力のビット 111-96
0x0104	鍵入力レジスタ 3	W	鍵入力のビット 95-80
0x0106	鍵入力レジスタ 4	W	鍵入力のビット 79-64
0x0108	鍵入力レジスタ 5	W	鍵入力のビット 63-48
0x010A	鍵入力レジスタ 6	W	鍵入力のビット 47-32
0x010C	鍵入力レジスタ 7	W	鍵入力のビット 31-16
0x010E	鍵入力レジスタ 8	W	鍵入力のビット 15-0
0x0140	データ入力レジスタ 1	W	平文／暗号文入力のビット 127-122
0x0142	データ入力レジスタ 2	W	平文／暗号文入力のビット 111-96
0x0144	データ入力レジスタ 3	W	平文／暗号文入力のビット 95-80
0x0146	データ入力レジスタ 4	W	平文／暗号文入力のビット 79-64
0x0148	データ入力レジスタ 5	W	平文／暗号文入力のビット 63-48
0x014A	データ入力レジスタ 6	W	平文／暗号文入力のビット 47-32
0x014C	データ入力レジスタ 7	W	平文／暗号文入力のビット 31-16
0x014E	データ入力レジスタ 8	W	平文／暗号文入力のビット 15-0
0x0180	データ出力レジスタ 1	R	暗号／復号結果のビット 127-112
0x0182	データ出力レジスタ 2	R	暗号／復号結果のビット 111-96
0x0184	データ出力レジスタ 3	R	暗号／復号結果のビット 95-80
0x0186	データ出力レジスタ 4	R	暗号／復号結果のビット 79-64
0x0188	データ出力レジスタ 5	R	暗号／復号結果のビット 63-48
0x018A	データ出力レジスタ 6	R	暗号／復号結果のビット 47-32

0x018C	データ出力レジスタ 7	R	暗号／復号結果のビット 31-16
0x018E	データ出力レジスタ 8	R	暗号／復号結果のビット 15-0
0xFFFC	バージョン・レジスタ	R	FPGA1 のバージョンが 16 進 4 桁で書かれている読み出し専用レジスタ

5.5 レジスタ

5.5.1 コントロール・レジスタ

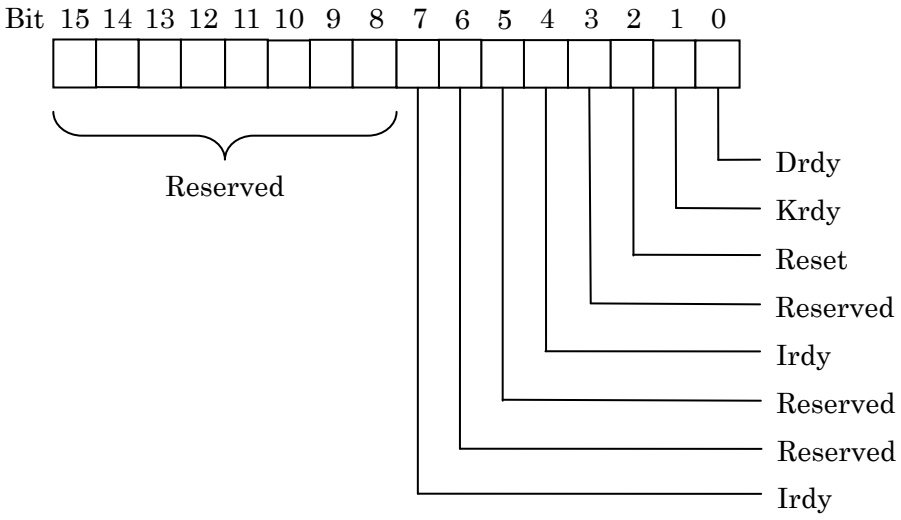


図 16　コントロール・レジスタ ビットアサイン

5.5.2　鍵幅レジスタ

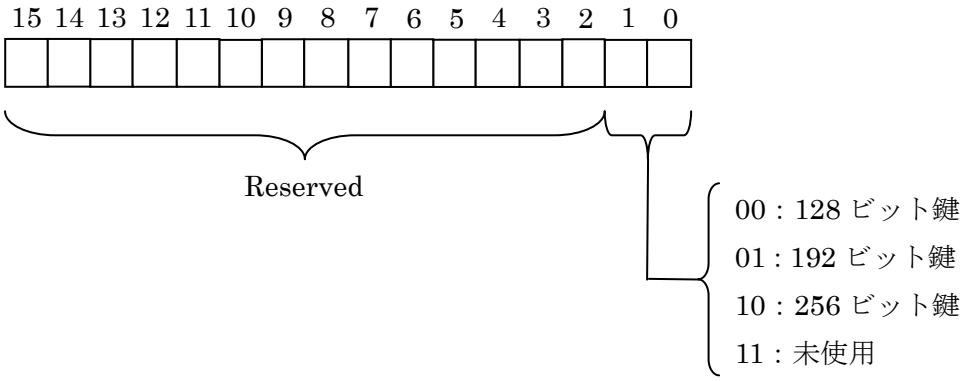


図 17　鍵幅レジスタ ビットアサイン

5.5.3　レングス入力レジスタ 1，2

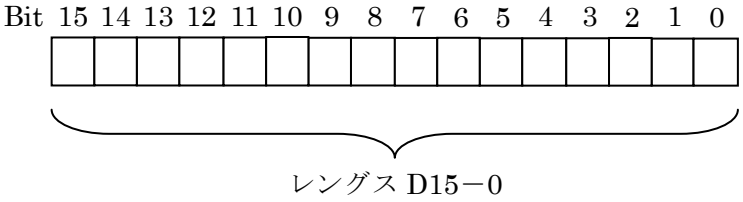


図 18　レングス入力レジスタ 1，2 ビットアサイン

5.5.4　鍵入力レジスタ 1-8

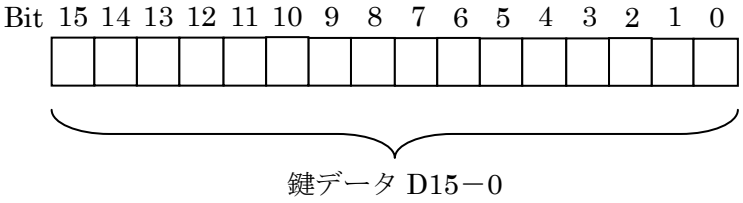


図 19　鍵入力レジスタ 1-8 ビットアサイン

5.5.5　イニシャルベクター入力レジスタ 1-8

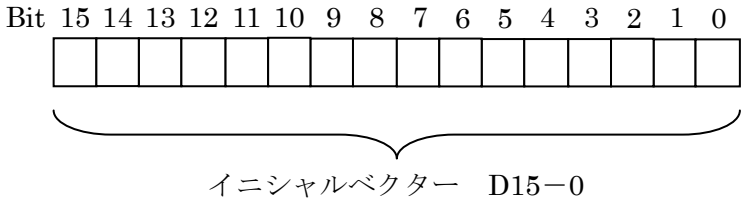


図 20　イニシャルベクター入力レジスタ 1-8 ビットアサイン

5.5.6　データ入力レジスタ 1-8

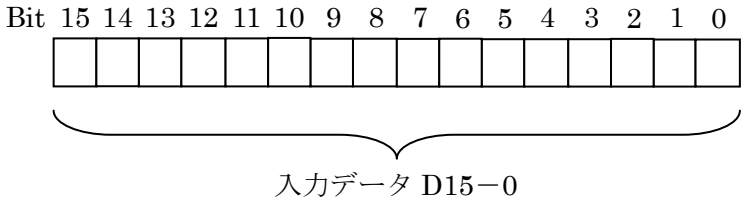


図 21　イニシャルベクター入力レジスタ 1-8 ビットアサイン

5.5.7　データ出力レジスタ 1-8

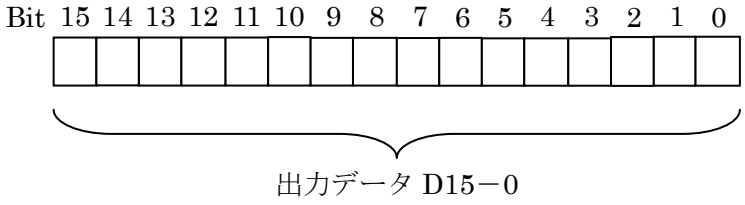


図 22　データ出力レジスタ 1-8 ビットアサイン

5.5.8　FPGA1 バージョン・レジスタ

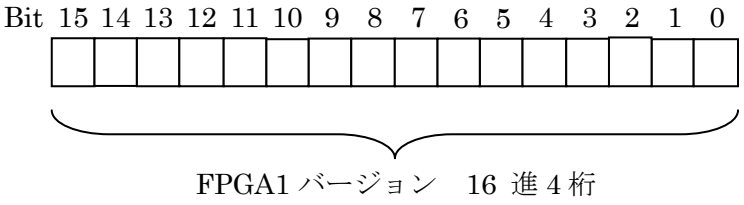


図 23　FPGA1 バージョン・レジスタ ビットアサイン

5.6　動作手順

CMAC 回路を動作させるには、以下の手順に従って操作を行う。

手順 1： リセット入力
手順 2：鍵幅を設定
手順 3： 鍵入力
　　鍵入力レジスタの 1 から 8 に鍵 1 をセット
手順 4：”Krdy”入力
手順 5： 鍵幅が 128 ビットならば手順 10 に移動
手順 6： 鍵入力 2
　　鍵入力レジスタの 1 から 8 に鍵 2 をセット
手順 7：”Krdy”入力
手順 8： イニシャルベクター入力
　　イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット
手順 9：”Irdy”入力
手順 10：レングス入力
　　レングス入力レジスタの 1，2 にレングス値をセット
手順 11：”Lrdy”入力
手順 12：データ入力
　　データ入力レジスタの 1 から 8 にデータをセット。
手順 13：“Drdy”入力
手順 14：終了でなければ手順 12 に戻る
手順 15：CMAC 値の読み出し
　　データ出力レジスタの 1 から 8 を読み出だす。
手順 16：終了

６．ストリーム暗号MUGI

MUGI は、ストリーム暗号向けの疑似乱数生成器である。128 ビットの秘密鍵と 128 ビットの公開イニシャルベクターをパラメータとして持つ。

6.1　ファイル構成

表 10　FPGA1_MUGI ファイル構成

ファイル名	種類	内容
FPGA1_MUGI.v	Verilog-HDL	FPGA1_MUGI の最上位となる HDL ファイル
syncfifo_8x2047.v	Verilog-HDL	8 ビット幅で深さが 2047 ワードの同期型 FIFO を記述した HDL ファイル
ctrl_lbus.v	Verilog-HDL	FPGA2 とのローカルバス・インターフェースを記述した HDL ファイル
mugi_lbus_if.v	Verilog-HDL	暗号モジュールの制御を記述した HDL ファイル
MUGI.v	Verilog-HDL	MUGI 暗号モジュールのトップ HDL ファイル
FPGA1_MUGI_TB1.v	Verilog-HDL	FPGA1_MUGI 暗号回路のテストベンチを記述した HDL ファイル
MUGI_TB.v	Verilog-HDL	MUGI 暗号回路単体のテストベンチを記述した HDL ファイル
pin_sasebo_gii_lx50.ucf	構成設定	FPGA1_MUGI の構成を規定したファイル

6.2　FPGA1_MUGIモジュール構成

FPGA1_MUGI の各モジュールは、以下のような構成で接続される。

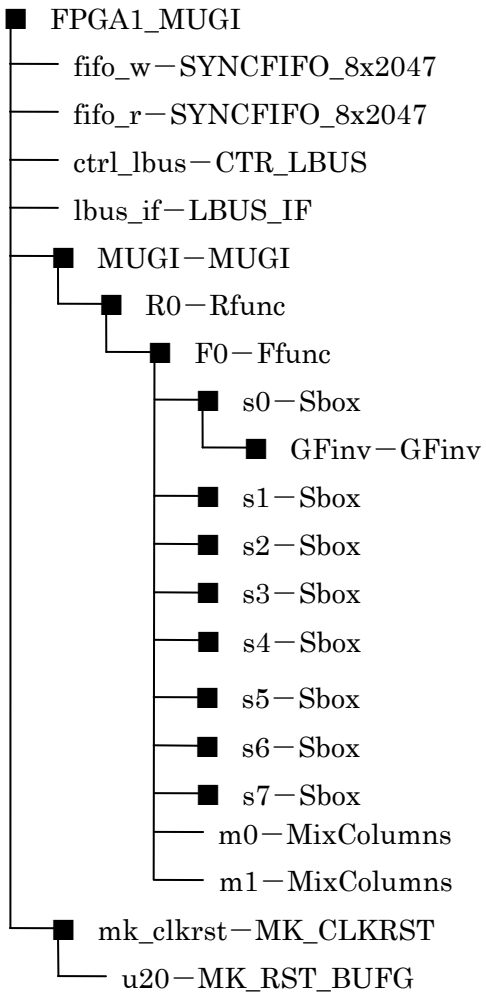


図 24　FPGA1_MUGI のモジュール構成

6.3　MUGIモジュール入出力信号

表 11 に暗号回路の入出力信号を示す。これらの信号は、”lbus_if”モジュールに接続され制御される。

表 11　MUGI モジュール入出力信号

信号名	方向	アクティブ	機能
RSTn	In	Low	リセット入力。MUGI モジュールのすべての回路が初期化される。
CLK	In	Rise	システムクロック入力。MUGI モジュールは、CLK の立ち上りエッジに同期して動作する。
EN	In	High	動作イネーブル信号。”EN”を’1’にすることにより AES モジュールが動作可能となる。
Busy	Out	High	“Busy”が’1’のとき AES モジュールが動作中であることを示めす。
Kin[127:0]	In	－	鍵入力
Iin[127:0]	In	－	イニシャルベクター入力
Krdy	In	High	鍵入力を MUGI モジュールにセットし、初期化 1 を開始させる。
Irdy	In	High	イニシャルベクターを MUGI モジュールにセットし、初期化、アップデートを開始する。
Rrdy	In	High	ラウンドを一つ進めて乱数を発生させる。
Dout	Out	－	乱数出力。
Kvld	Out	High	鍵生成が終了したことを示す。
Rvld	Out	High	Dout に乱数が出力されたことを示す。

6.4　アドレスマップ

”lbus_if”モジュール内には、暗号モジュールを操作するためのレジスタ類が複数設けられている。これらのレジスタは、ローカルバス上にマッピングされている。表 12 にアドレスマップを示す。

表 12　MUGI アドレスマップ

アドレス	名称	R/W	機能
0x0002	コントロール・レジスタ	R/W	MUGI 回路に対して制御信号を出力するためのレジスタ
0x0100	鍵入力レジスタ 1	W	鍵入力のビット 127-112
0x0102	鍵入力レジスタ 2	W	鍵入力のビット 111-96
0x0104	鍵入力レジスタ 3	W	鍵入力のビット 95-80
0x0106	鍵入力レジスタ 4	W	鍵入力のビット 79-64

0x0108	鍵入力レジスタ 5	W	鍵入力のビット 63-48
0x010A	鍵入力レジスタ 6	W	鍵入力のビット 47-32
0x010C	鍵入力レジスタ 7	W	鍵入力のビット 31-16
0x010E	鍵入力レジスタ 8	W	鍵入力のビット 15-0
0x0120	イニシャルベクター・レジスタ 1	W	イニシャルベクター値入力のビット 127-112
0x0122	イニシャルベクター・レジスタ 2	W	イニシャルベクター値入力のビット 111-96
0x0124	イニシャルベクター・レジスタ 3	W	イニシャルベクター値入力のビット 95-80
0x0126	イニシャルベクター・レジスタ 4	W	イニシャルベクター値入力のビット 79-64
0x0128	イニシャルベクター・レジスタ 5	W	イニシャルベクター値入力のビット 63-48
0x012A	イニシャルベクター・レジスタ 6	W	イニシャルベクター値入力のビット 47-32
0x012C	イニシャルベクター・レジスタ 7	W	イニシャルベクター値入力のビット 31-16
0x012E	イニシャルベクター・レジスタ 8	W	イニシャルベクター値入力のビット 15-0
0x0180	データ出力レジスタ 1	R	暗号／復号結果のビット 127-112
0x0182	データ出力レジスタ 2	R	暗号／復号結果のビット 111-96
0x0184	データ出力レジスタ 3	R	暗号／復号結果のビット 95-80
0x0186	データ出力レジスタ 4	R	暗号／復号結果のビット 79-64
0x0188	データ出力レジスタ 5	R	暗号／復号結果のビット 63-48
0x018A	データ出力レジスタ 6	R	暗号／復号結果のビット 47-32
0x018C	データ出力レジスタ 7	R	暗号／復号結果のビット 31-16
0x018E	データ出力レジスタ 8	R	暗号／復号結果のビット 15-0
0xFFFC	バージョン・レジスタ	R	FPGA1 のバージョンが 16 進 4 桁で書かれている読み出し専用レジスタ

6.5　レジスタ

6.5.1　コントロール・レジスタ

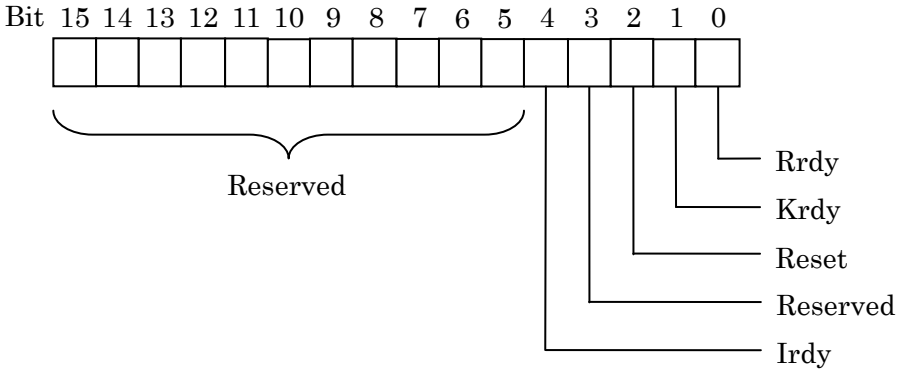


図 25　コントロール・レジスタ ビットアサイン

6.5.2　鍵入力レジスタ 1-8

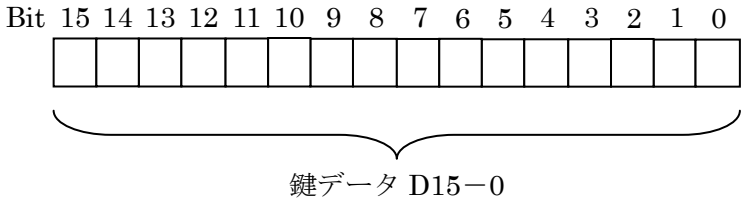


図 26　鍵入力レジスタ 1-8 ビットアサイン

6.5.3　イニシャルベクター入力レジスタ 1-8

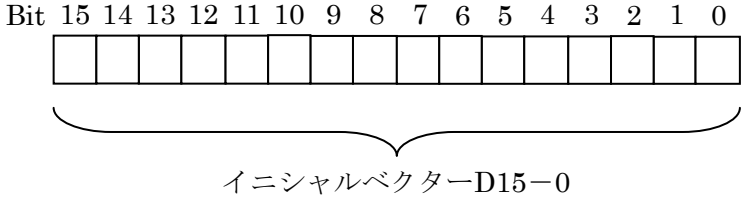


図 27　イニシャルベクター入力レジスタ 1-8 ビットアサイン

6.5.4　データ出力レジスタ 1-8

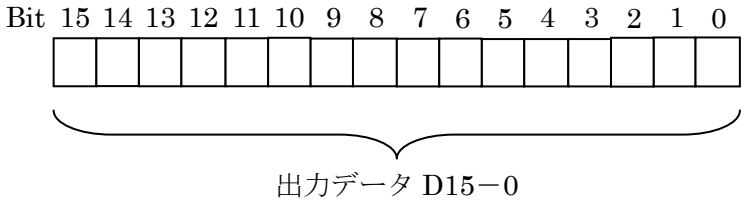


図 28　データ出力レジスタ 1-8 ビットアサイン

6.5.5　FPGA1 バージョン・レジスタ

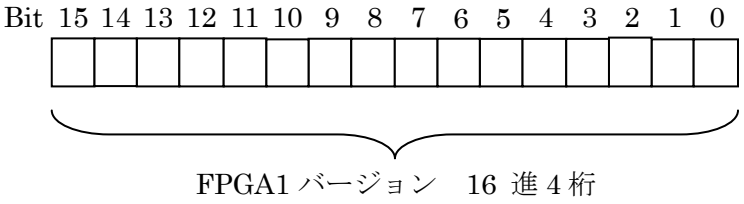


図 29　FPGA1 バージョン・レジスタ ビットアサイン

6.6　動作手順

MUGI 回路を動作させるには、以下の手順に従って操作を行う。

手順 1：リセット入力

手順 2：鍵入力

　　鍵入力レジスタの 1 から 8 に鍵値をセット

手順 3：”Krdy”入力

手順 4：イニシャルベクター入力

　　イニシャルベクター入力レジスタの 1 から 8 にイニシャルベクターをセット

手順 5：”Irdy”入力

手順 6：乱数読み出し

　　データ出力レジスタの 1 から 4 を読み出して乱数値を得る。

手順 7：“Rrdy”入力

手順 8：乱数読み出し

データ出力レジスタの 1 から 4 を読み出して乱数値を得る。

手順 9：終了でなければ手順 6 に戻る

手順 10：終了

7．HMAC

7.1　ファイル構成

表 13　FPGA1_SHA256 ファイル構成

ファイル名	種類	内容
FPGA1_SHA256.v	Verilog-HDL	FPGA1_SHA256 の最上位となる HDL ファイル
syncfifo_8x2047.v	Verilog-HDL	8 ビット幅で深さが 2047 ワードの同期型 FIFO を記述した HDL ファイル
ctrl_lbus.v	Verilog-HDL	FPGA2 とのローカルバス・インターフェースを記述した HDL ファイル
sha_lbus_if.v	Verilog-HDL	暗号モジュールの制御を記述した HDL ファイル
SHA256.v	Verilog-HDL	SHA256 モジュールのトップ HDL ファイル
FPGA1_SHA_TB1.v	Verilog-HDL	FPGA1_SHA256 回路のテストベンチを記述した HDL ファイル
SHA256_TB.v	Verilog-HDL	SHA256 回路単体のテストベンチを記述した HDL ファイル
pin_sasebo_gii_lx50.ucf	構成設定	FPGA1_MUGI の構成を規定したファイル

7.2　FPGA1_SHA256 モジュール構成

FPGA1_の各モジュールは、以下のような構成で接続される。

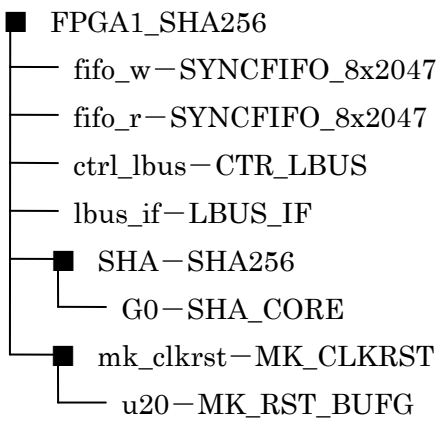


図 30　FPGA1_SFA256 モジュール構成

7.3　FPGA1_SHA256 入出力信号

表 11 に暗号回路の入出力信号を示す。これらの信号は、”lbus_if”モジュールに接続され制御される。

表 14　SHA-256 モジュール入出力信号

信号名	方向	アクティブ	機能
RSTn	In	Low	リセット入力。SHA-256 モジュールのすべての回路が初期化される。
CLK	In	Rise	システムクロック入力。SHA-256 モジュールは、CLK の立ち上りエッジに同期して動作する。
EN	In	High	動作イネーブル信号。”EN”を’1’にすることにより SHA-256 モジュールが動作可能となる。
Busy	Out	High	“Busy”が’1’のとき SHA-256 モジュールが動作中であることを示めす。
INIT	In	High	SHA-256 モジュールを初期化する。
MSGin[31:0]	In	－	メッセージ入力。
Mrdy	In	High	メッセージ入力が完了し、SHA256 回路に動作要求を行う。
Hout[255:0]	Out	－	ハッシュ値の出力。
Hvld	Out	High	ハッシュ値の生成が終了したことを示す。

7.4 アドレスマップ

”lbus_if”モジュール内には、暗号モジュールを操作するためのレジスタ類が複数設けられている。これらのレジスタは、ローカルバス上にマッピングされている。表 15 にアドレスマップを示す。

表 15　HMAC アドレスマップ

アドレス	名称	R/W	機能
0x0002	コントロール・レジスタ	R/W	SHA256 回路に対して制御信号を出力するためのレジスタ
0x0140	データ入力レジスタ 1	W	平文入力のビット 31-16
0x0142	データ入力レジスタ 2	W	平文入力のビット 15-0
0x0180	データ出力レジスタ 1	R	暗号／復号結果のビット 255-240
0x0182	データ出力レジスタ 2	R	暗号／復号結果のビット 239-224
0x0184	データ出力レジスタ 3	R	暗号／復号結果のビット 223-208
0x0186	データ出力レジスタ 4	R	暗号／復号結果のビット 191-176
0x0188	データ出力レジスタ 5	R	暗号／復号結果のビット 175-160
0x018A	データ出力レジスタ 6	R	暗号／復号結果のビット 159-144
0x018C	データ出力レジスタ 7	R	暗号／復号結果のビット 143-128
0x018E	データ出力レジスタ 8	R	暗号／復号結果のビット 15-0
0x0190	データ出力レジスタ 9	R	暗号／復号結果のビット 127-112
0x0192	データ出力レジスタ 10	R	暗号／復号結果のビット 111-96
0x0194	データ出力レジスタ 11	R	暗号／復号結果のビット 95-80
0x0196	データ出力レジスタ 12	R	暗号／復号結果のビット 79-64
0x0198	データ出力レジスタ 13	R	暗号／復号結果のビット 63-48
0x019A	データ出力レジスタ 14	R	暗号／復号結果のビット 47-32
0x019C	データ出力レジスタ 15	R	暗号／復号結果のビット 31-16
0x019E	データ出力レジスタ 16	R	暗号／復号結果のビット 15-0
0xFFFC	バージョン・レジスタ	R	FPGA1 のバージョンが 16 進 4 桁で書かれている読み出し専用レジスタ

7.5　レジスタ

7.5.1　コントロール・レジスタ

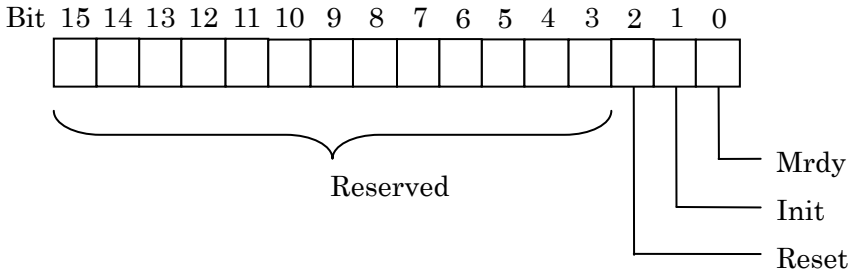


図 31　コントロール・レジスタ ビットアサイン

7.5.2　データ入力レジスタ 1，2

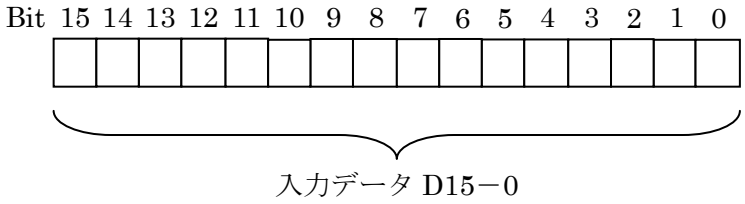


図 32　データ入力レジスタ 1，2 ビットアサイン

7.5.3　データ出力レジスタ 1-16

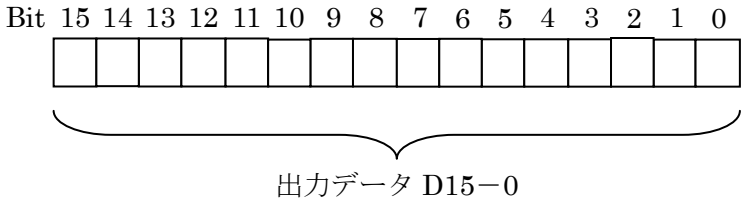


図 33　データ出力レジスタ 1，2 ビットアサイン

7.5.4　FPGA1 バージョン・レジスタ

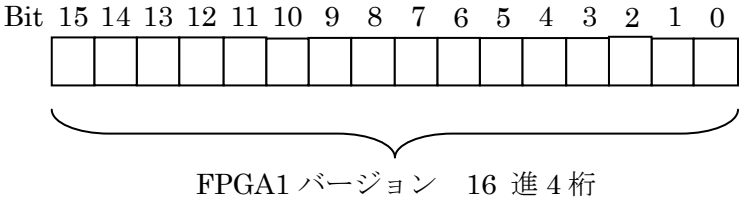


図 34　FPGA1 バージョン・レジスタ ビットアサイン

7.6　動作手順

SHA256 回路を動作させるには、以下の手順に従って操作を行う。　SHA256 は、256 ビットのデータ単位に処理を行うため、データは、256 ビット単位で入力しなければならない。

手順 1：リセット入力
手順 2：SHA256 初期化
手順 3：メッセージ入力
　　データ入力レジスタ 1，2 にデータをセット。
手順 4：”Mrdy”入力
手順 5：手順 3,4 を 16 回繰り返す
手順 6：入力メッセージ終了で手順 7 に、メッセージがまだある場合は手順 3 に戻る
手順 7：ハッシュ値の読み出し
　　データ出力レジスタの 1 から 16 を読み出してハッシュ値を得る。

８．制御サンプルソフトウェア

制御サンプルソフトウェアは、SASEBO-GⅡ-AES 暗号 FPGA ボードに実装されている 4 種類の暗号回路を SASEBO-GⅡ-AES 暗号 FPGA ボードの USB ポートを通して PC から制御を行うためのプログラムである。これらのプログラムは、C#言語で記述され開発ツールは、Microsoft Visual C# 2008 Express Edition を使用している。

サンプルプログラムは、Visual C#のプロジェクトの形で以下の 4 種類用意する。

・SASEBO_AES_sample
・SASEBO_CMAC_sample
・SASEBO_MUGI_sample
・SASEBO_SHA256_sample

8.1　プロジェクトの構成ファイル

各プロジェクトは、表 16 に示されるファイルにより構成されている。

表 16　プロジェクトの構成ファイル

ファイル名	内容
Program.cs	プログラムのメインルーチンのソースファイルで、プログラムごとに異なる。
SASEBO_CMD_interface.cs	FPGA1 のローカルバスのアクセス手順を記述したソースファイルで、すべてのプロジェクトで共通に使用する。
ft245rl_interface.cs	USB コントローラ用の DLL プログラムを呼び出すためのラッパ関数のソースファイルで、すべてのプロジェクトで共通に使用する。
FTD2XX_NET.dll	USB コントローラ用の DLL ファイルで、すべてのプロジェクトで共通に使用する。

8.2　プログラムの構造

サンプルプログラムは、暗号回路に対応して 4 種類あるが、基本的に同様の構造である。いくつかの簡単なコマンドを規定し、そのコマンドが書かれたテキスト形式のスクリプトファイルを読み込んで、その内容に従い FPGA1 内のレジスタを読み書きする。図 35 にサンプルプログラムの概略フローチャートを示す。

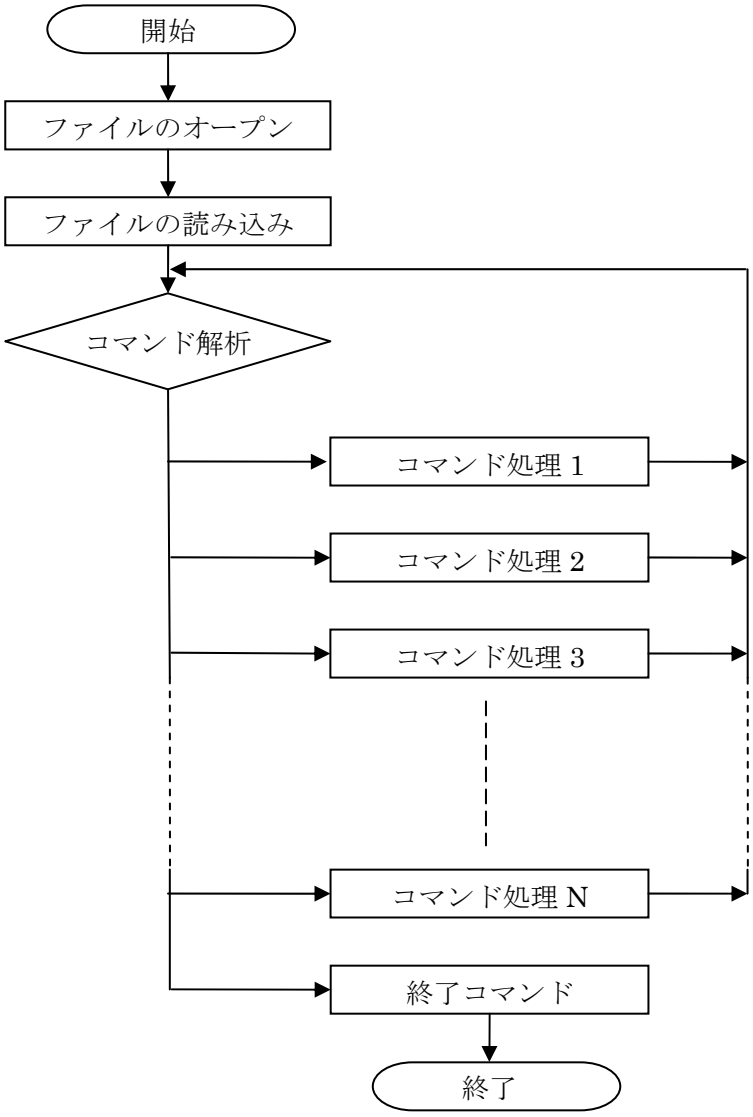


図 35　サンプルプログラム概略フローチャート

暗号回路により、コマンドの数や各コマンド処理に違いがあるが基本な処理内容は同じである。

8.3　スクリプトファイル

スクリプトファイルで使用できるコマンドを表 17 に示す。

表 17　スクリプトファイルのコマンド表

表記	意味	内容	使用暗号
$R	リセット入力	暗号回路にリセットを入力する。	全暗号
$F	スクリプト終了	プログラムを終了する。	全暗号
$O	結果読み出し	暗号回路の結果を読み出して画面に表示する。	全暗号
#K	鍵セット	鍵レジスタに鍵をセットし、”Krdy”を入力する。	AES，CMAC, MUGI, HMAC
#C	カウンタ値入力	カウント・レジスタにカウント値をセットし、”Crdy’を入力する。	AES
#I	イニシャルベクター入力	イニシャルベクター・レジスタにイニシャルベクター値をセットし、”Irdy”を入力する。	AES，MUGI
#L	レングス入力	レングス・レジスタにレングス値をセットし、”Lrdy”を入力する。	CMAC
#D	データ入力	データ入力レジスタに入力データをセットし、”Drdy”を入力する。	AES，CMAC, HMAC
#E	ENC/DEC 入力	ENC/DEC レジスタに動作モードをセットする。	AES
#M	AES モード入力	AES モード・レジスタに動作させる AES モードをセットする。	AES
#W	鍵幅入力	鍵幅レジスタに使用する鍵幅をセットする。	AES，CMAC

8.4　AES

AES サンプルプログラムは、ファイル名 ”aes.txt” のスクリプトファイルを読み込んで実行する。　リスト 1 に AES 暗号のスクリプトファイルの記述例を示す。

リスト 1　AES-ECB-128bitKey　Encrypt サンプルスクリプト

```
$R        ····· リセット入力
#E 0      ····· Encrypt モード
#M 0      ····· ECB モード
#W 0      ····· 128 ビット鍵
#K 2b7e151628aed2a6abf7158809cf4f3c     ····· 鍵入力
#D 6bc1bee22e409f96e93d7e117393172a     ····· 平文入力
$O        ····· 暗号文出力
$F        ····· スクリプト終了
```

リスト 1 の実行結果を図 36 に示す。

```
SASEBO AES control sample program start
Number of FTDI device   : 2
Successful to open device.
ver   : 0001

AES Reset!!
Encrypt
MODE: 0
KEY WIDTH: 0
KEY : 2B7E151628AED2A6ABF7158809CF4F3C
DATA: 6BC1BEE22E409F96E93D7E117393172A
OUT : 3AD77BB407DA3660A89ECAF32466EF97

Press enter.
```

図 36　リスト 1 実行結果

リスト 2　AES-ECB-128bitKey　Decrypt サンプルスクリプト

```
$R      ····· リセット入力
#E 1    ····· Decrypt モード
#M 0    ····· ECB モード
#W 0    ····· 128 ビット鍵
#K 2b7e151628aed2a6abf7158809cf4f3c    ····· 鍵入力
#D 3ad77bb40d7a3660a89ecaf32466ef97    ····· 暗号入力
$O      ····· 平文出力
$F      ····· スクリプト終了
```

8.5　CMAC

CMAC サンプルプログラムは、ファイル名 ”cmac.txt” のスクリプトファイルを読み込んで実行する。リスト 3 に CMAC スリプトファイルの記述例を示す。

リスト 3　CMAC-128bitKey サンプルスクリプト

```
$R      ····· リセット入力
#W 0    ····· 128 ビット鍵
#L 00000040    ····· レングス入力
#K 2b7e151628aed2a6abf7158809cf4f3c    ····· 鍵入力
#D 6bc1bee22e409f96e93d7e117393172a    ····· 平文入力
#D ae2d8a571e03ac9c9eb76fac45af8e51    ····· 平文入力
#D 30c81c46a35ce411e5fbc1191a0a52ef    ····· 平文入力
#D f69f2445df4f9b17ad2b417be66c3710    ····· 平文入力
$O      ····· 暗号文出力
$F      ····· スクリプト終了
```

8.6 MUGI

MUGI サンプルプログラムは、ファイル名 ”mugi.txt” のスクリプトファイルを読み込んで実行する。リスト 4 に MUGI スリプトファイルの記述例を示す。

リスト 4　MUGI サンプルスクリプト

```
$R      ····· リセット入力
#K 000102030405060708090A0B0C0D0E0F      ····· 鍵入力
#I F0E0D0C0B0A090807060504030201000      ····· 初期ベクター入力
$O      ····· 乱数出力
$T      ····· 乱数発生
$O      ····· 乱数出力
$T      ····· 乱数発生
$O      ····· 乱数出力
$T      ····· 乱数発生
$O      ····· 乱数出力
$F      ····· スクリプト終了
```

8.7 HMAC

HMAC サンプルプログラムは、ファイル名 ”sha256.txt” のスクリプトファイルを読み込んで実行する。平文の入力は、256 ビット単位で入力しなければならない。リスト 5 に HMAC スリプトファイルの記述例を示す。

リスト 5　HMAC サンプルスクリプト

```
$R       ····· リセット入力
#D 61626380    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000000    ····· 平文入力
#D 00000018    ····· 平文入力
$O       ····· ハッシュ出力
$F       ····· スクリプト終了
```