システムＬＳＩ設計技術　106－8
(2002.　5.　24)

システムLSI向け8ビットプロセッサコアIPの開発

真島　優輔†　　安浦　寛人††

† 九州大学大学院 システム情報科学府 情報工学専攻
〒816-8580 福岡県春日市春日公園 6-1
電話：092-583-7622　　FAX:092-583-1338
†† 九州大学大学院 システム情報科学研究院 情報工学部門
〒816-8580 福岡県春日市春日公園 6-1
電話：092-583-7620　　FAX:092-583-1338
E-mail: †{majima,yasuura}@c.csce.kyushu-u.ac.jp

あらまし　現在 VDEC を通じて，教育および研究を目的とした多くの LSI が設計されるようになっている．本研究では，VDEC を通じて行われる，大学における教育・研究を目的としたシステム LSI の開発に利用可能なプロセッサコア IP の開発を行った．開発したプロセッサコア IP は Zilog 社の Z80 と完全互換である．Z80 は無償で利用可能なソフトウェア開発環境が充実しており，また，過去のソフトウェア資産が数多く存在する．開発したプロセッサコア IP は，ソフト IP としての VDEC への提供を目的としている．本研究では，VDEC を通じて利用可能な 2 つのテクノロジをターゲットとして，ソフトコア IP として提供する RTL の設計データに対してテクノロジマッピングを行い，回路規模や動作周波数を評価した．

キーワード　IP，Z80，システム LSI，ソフト IP，VDEC

Development of 8-bit proccessor IP for SoC

Yusuke MAJIMA† and Hiroto YASUURA††

† Graduate School of Information Science and Electrical Engineering, Kyushu University
6-1 Kasuga-Koen, Kasuga, Fukuoka 816-8580 JAPAN
PHONE:（+81）92-583-7622　　FAX:（+81）92-583-1338
†† The Department of Computer Science and Communication Engineering
6-1 Kasuga-Koen, Kasuga, Fukuoka 816-8580 JAPAN
PHONE:（+81）92-583-7620　　FAX:（+81）92-583-1338
E-mail: †{majima,yasuura}@c.csce.kyushu-u.ac.jp

Abstract　We developed processor IP core compatible with Zilog Z80 for education and researches in university. Many free software development environments are available for Z80. We are going to put up the IPcore for VDEC as soft IP. Experimental mapping with two process parameters to layout of the IP shows the clock frequency,area and transistor counts.

Key words　IP, Z80, soft IP,VDEC

1. はじめに

近年の半導体微細加工技術の進歩に伴い，システムを構成する多様な機能をひとつのチップに集積したSoC（System on a Chip）の実現が可能となり，現在、IP（Intellectual Property）の利用体制を整えることが産学ともに急務となっている．

大学や高等専門学校などでの教育および研究の場におけるLSI設計のインフラを整備してきたVDEC（東京大学大規模集積システム設計教育研究センター）では，現在，IPの利用体制の整備を推進するIPプロジェクト体制が採られている[1]．VDECの各ユーザーが独自回路を含むシステムLSIを試作する場合に利用可能なIPの開発を推進するものである。本研究では，IPプロジェクトの開発テーマのひとつ，ディジタルプロセッサコアの開発の一環として，VDECを通じて利用可能となるプロセッサコアIPの開発を行った．開発したプロセッサコアIPはZilog社のZ80と完全互換である．

一般にプロセッサを利用したシステムを開発する場合，システムの多様な機能の多くはソフトウェアで実現される．ソフトウェアの開発には，デバッガやコンパイラ，シミュレータなどの開発環境が必要となる．Z80はPCの黎明期によく採用された実績のあるプロセッサであるため，無償で利用できるソフトウェア開発環境も多い．また、Z80は現在でもパフォーマンスを要しない組み込み機器などでも用いられている．既存のソフトウェア資産を利用してシステムの開発を行うことができる．

本稿は次のような構成となっている．2章では、開発したIPの仕様を述べる．3章では，特定のプロセスを対象としてレイアウト処理を行い、得られたレイアウトデータより回路の諸評価を行い，4章で本稿をまとめる．

2. プロセッサの仕様

2.1 主な仕様

本研究において開発したプロセッサコアは，Zilog社のZ80と完全互換である．ただし，オリジナルのZ80に関して一般に知られているような，実行可能だが公式にサポートされていない未定義命令に関してはサポートしていない．

Z80の特徴を、以下にあげる．

- 8ビットのCISC（Complex Instruction Set Computer）プロセッサである．
- メモリアドレス空間は64Kである．
- Intel社8080と命令セット上位互換である．
- 158種類の命令を持ち，1命令は最大4ワードからなる．
- 1命令は，最小4クロックサイクル，最大23クロックサイクルからなる．
- ノンマスカブル割込みと，3種類のモードからなるマスカブル割込みを持つ．マスカブル割込みの種類は以下の通りである．
  - Intel社8080と互換性のある割込みモード（モード0）
  - リスタート割込み（モード1）
  - ディジーチェーン構造の割込み（モード2）
- 汎用レジスタのそれぞれに自身と交換可能な補助レジスタを持つ．
- DRAM用リフレッシュ回路が内蔵されている．
- 10種類のアドレッシングモードが利用可能である．

論理算術演算器の主な機能は以下の通りである．

- 8ビット加算・減算
- 8ビットインクリメント・デクリメント
- 8ビット論理演算
- 8ビットシフト・ローテイト
- ビット操作
- 拡張16ビット加算・減算
- 拡張16ビットインクリメント・デクリメント
- 8ビットBDD(2進化10進）変換

トップモジュールの入出力信号を表1に示す．全ての入出力信号は，オリジナルのZilog社Z80と互換性を持つ．オリジナルのZ80の詳細な外部仕様に関してはZ80ユーザーズマニュアル[2]を参考にして頂きたい．

表1　プロセッサの外部入出力信号
Table 1　external I/O signals of this IP

信号名	極性	方向	機能
CLK	-	入力	クロック信号
RESET	L	入力	リセット信号
INT	L	入力	マスカブル割り込み要求信号
NMI	L	入力	ノンマスカブル割り込み要求信号
BUSREQ	L	入力	周辺回路のバス利用要求信号
WAIT	L	入力	ウェイト信号
AB	-	出力	16ビットのアドレスバス
DB	-	双方向	8ビットのデータバス
HALT	L	出力	停止状態でアクティブ
MREQ	L	出力	メモリアクセス要求時にアクティブ
IORQ	L	出力	入出力要求時にアクティブ
RFSH	L	出力	DRAMリフレッシュ状態でアクティブ
M1	L	出力	M1サイクル時にアクティブ
BUSACK	L	出力	バス利用要求のアクノリッジ
WR	L	出力	メモリまたは周辺回路への出力要求
RD	L	出力	メモリまたは周辺回路への入力要求

2.2 内部構造

本研究では、プロセッサの開発にあたってオリジナルのZ80の詳細な内部仕様が公開されていなかったため，Z80ユーザーズマニュアルにある外部仕様を基に開発した．以下，我々が開発したZ80互換プロセッサコアの内部構造の詳細について述べる．

CISCプロセッサであるオリジナルのZ80は，1つの命令で複雑な処理を行う命令の一連の処理をマルチサイクルパスで実現することにより，演算器などのハードウェア資源の有効利用を図っている．開発したプロセッサもハイパフォーマンスを求めるアーキテクチャを採用せず，オリジナルのZ80と同じクロックサイクル数で命令の実行が完了するようなマルチサイクルパスを採用し，ハードウェア資源が有効に利用されるよう工夫した．

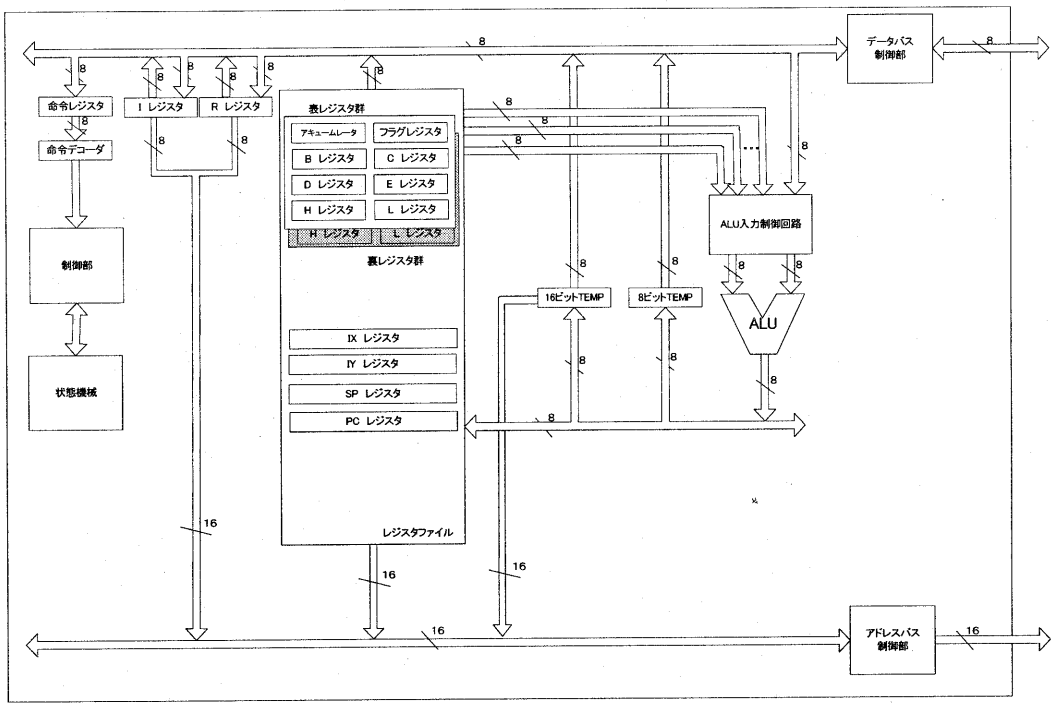

図 1　プロセッサコアのブロック図

Fig. 1　the orgnization of our IP

2.3　設計データの提供方法

本研究において設計したプロセッサは，VDEC を通じてソフト IP として利用可能になる予定である．提供する設計データは，合成可能な Verilog-HDL による RTL 記述である．ソースコードは 21 個のファイルからなり，全体で 4,144 行である．合成可能であることを確認した CAD ツールは，Synopsys 社の Design Compiler である．本稿の最後に，Verilog-HDL ソースコードの一部を示す．

2.4　システム開発

本研究において開発したプロセッサを利用してシステムを開発する際には、無償で提供されているようなソフトウェア開発環境が利用できる。命令レベルのシミュレータでは YAZE がよく知られているし，コンパイラやアセンブラに関してもさまざまなクロス開発環境が存在する．表 2 にその一例を示す．システムの開発者にとって必要なことは，独自に開発している回路に対して，命令レベルのシミュレータとタイミング検証するための Verilog-HDL によって記述された動作モデルを用意することである．

本研究において開発したプロセッサコアはオリジナルの Z80 と完全互換である．シリアル/パラレルインターフェイスや割り込みコントローラなどの周辺回路を別途設計する必要が生じた場合は，Z80 ファミリーペリフェラルの詳細なドキュメントを参考にすることができる．

3.　回路評価

VDEC を通じて利用できるいくつかのテクノロジをターゲットとして，論理合成および自動配置配線処理を施し，最大動作周波数，コアのサイズ，および，ゲート数等を調べた．対象としたテクノロジは，日立社 0.18μmCMOS テクノロジと ROHM 社 0.35μmCMOS テクノロジである．表 3 に各テクノロジをターゲットとした場合の結果を示す．論理合成時，最適化の制約条件は遅延時間が最小になるように設定し，階層構造を無くし全て平坦化して最適化を施した．図 2 に ROHM0.35μm テクノロジをターゲットとした時のプロセッサコアのレイアウト図を示す．図 2 は演算器等のデータパス部分，および，制御部分など全て論理合成により得られたネットリストを基に自動配置配線処理を施して得られた回路である．

最大動作周波数は，レイアウトの結果から実配線長を考慮した配線遅延情報をネットリストにバックアノテートしてクリティカルパス遅延を求めることで算出した．最大動作周波数は，日立社 0.18μm プロセスで 456MHz と ROHM 社 0.35 μm プロセスで 248MHz となった．コアサイズは，ともに VDEC を通じて利用できる試作チップのセル配置可能面積に対して，10%未満の面積占有率であった．研究開発用として，ユーザーの独自回

表 2　無償で利用可能なソフトウェア開発環境
Table 2　free software development environment

種類	ツール名	ホスト	参考
アセンブラ	ZASM v.1.1	DOS	http://us.share.geocities.com/SiliconValley/Peaks/3938
シミュレータ	YAZE-AG	UNIX/LINUX	http://www.mathmatik.uni-ulm.de/users/ag/yaze-ag/
C コンパイラ	ZCC	DOS	http://us.share.geocities.com/SiliconValley/Peaks/3938
逆アセンブラ	DASM	DOS	http://us.share.geocities.com/SiliconValley/Peaks/3938

表 3　回路の各種評価
Table 3　evaluation of circuit

テクノロジ	ROHM 社 0.35μm	日立社 0.18μm
最大動作周波数 (MHz)	248	456
コアサイズ		
縦 (μm)	941	691
横 (μm)	966	700
面積 (mm^2)	1.057	0.4838
セル数	3768	3650
トランジスタ数	78,604	55,469

表 4　プロセッサコアのチップ面積占有率
Table 4　core/chip ratio

テクノロジ	ROHM 社 0.35μm	日立社 0.18μm
コアの面積 mm^2	1.057	0.4838
試作チップのチップサイズ	4.9mm 角	5.8mm 角
試作チップのセル配置 mm^2 可能面積（概算）	12.6	11.5
面積占有率 %	8.3	4.2

路を配置する面積に大きな影響を与えずにシステムをひとつのチップ上に実現できることが確認できた．
実際，前年度に行われた日立社 0.18μm テクノロジの試作時は，VDEC を通じて日立より提供されたメモリおよび PLL のマクロセルを配置しても余りある面積が確認できた．図 3 に，前年度試作時のレイアウト図を示す．図 3 において，①がプロセッサコアである．②，③は日立から提供されたマクロセルで，それぞれ，②は 2 種の PLL，③が 1 ワード 16 ビット・16K エントリの SRAM である．メモリを搭載してもチップ面積に余裕があることが確認できる．

4. おわりに

本研究では VDEC の IP プロジェクトの一環として，Zilog 社の Z80 と完全互換のプロセッサコア IP を開発した．プロセッサコア IP には，

- ソフトウェア開発環境
- 基本的な周辺回路
- コア部分のテスト容易性

が要求される．本研究において開発したプロセッサコア IP は Z80 と互換性を持つので，ソフトウェア開発環境に関して非常に充実している．今後は，基本的な周辺回路の充実を図るとともに，コア部分のテストを容易にするテスト容易化設計施す必要がある．周辺回路群に関しては，VDEC ユーザーがを用いたシステム開発を行う上で，別途に設計する必要が生じた回路を IP として遂次提供されるような仕組みが確立されることが望まれる．コア部分のテストはピン数が制限されることから，適切なテスト容易化設計を施す必要がある．どのようなテスト容易化設計を施すか今後開発する．

前年度に日立社 0.18μmCMOS テクノロジで試作したチップは，近々テスターを用いて最大動作周波数を測定する予定である．テスターによる測定を行ったあと，測定結果に基づいて設計の修整および改良を行い，本年度の日立 0.18μm プロセスの第 1 回目試作で再度チップを試作する予定である．本年度の試作でリファレンスモデルを提示することができれば，VDEC へ提供したいと考えている．

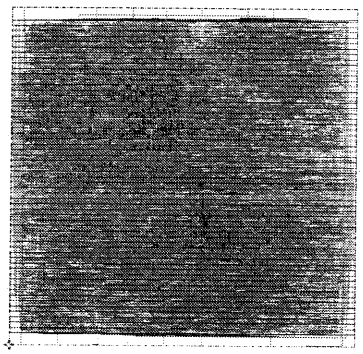

図 2　プロセッサコアのレイアウト図
Fig. 2　the layout of our IP

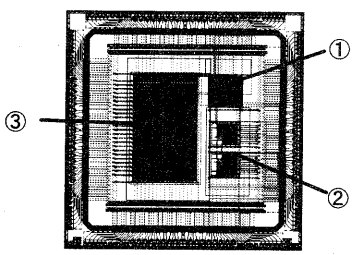

図 3　前年度の日立 0.18μm テクノロジでの試作時のレイアウト図
Fig. 3　layout of last prototype with HITACHI0.18μm

謝辞

本チップ試作は，東京大学大規模集積システム設計教育研究センターを通し，Avant!社，Synopsys 社，および，Cadence 社のツールを用いて行われたものである．

付録

以下，Verilog-HDL のソースコードの一部を示す．

```
  —— include 部（省略）——
module Z80(CLK,RESET,INT,NMI,BUSREQ,WAIT,
      AB,DB,HALT,MREQ,IORQ,RFSH,M1,BUSACK,WR,RD);
  input CLK;
  input RESET;
  input INT;
  input NMI;
  input BUSREQ;
  input WAIT;

  output [15:0] AB;
  output   HALT;
  output   IORQ;
  output   MREQ;
  output   RFSH;
  output   M1;
  output   BUSACK;
  output   WR;
  output   RD;

  inout [7:0]   DB;

  //==internal signal==========
  —— wire 宣言部（省略）——

  RESET_CTRL RESET_CTRL(
.RESET_IN(RESET),.CLK(CLK),
.RESET(RESETint)
);

  DB_CTRL DB_CTRL(
  .RESET(RESETint),.BUSACK(BUSACK),
  .IORQ(IORQ),
  .WR(WR),.RD(RD),.DBIN(DBint),
  .DBOUT(DBint),.EX_DB(DB)
  );

  AB_CTRL AB_CTRL(
  .RESET(RESETint),.BUSACK(BUSACK),
  .ABIN(ABint),
  .ABOUT(AB)
  );

   ALUlogic ALU(
.A(ALUa),.B(ALUb),
.CIN(ALUcin),.HIN(ALUhin),.NIN(ALUnin),
.SETBIT(SETBIT),.ALUCTRL(ALUCTRL),
.TCR(TCR),.exALU(exALU),
.relative_high(relative_high),
.Y(ALUOUT),.FLAGOUT(ALU_FLAGOUT)
);

   ALUinputCTRL ALUinputCTRL(
     .DBIN(DBint),
     .A(A),.F(F),.B(B),.C(C),.D(D),
     .E(E),.H(H),.L(L),
     .IX(IX),.IY(IY),.SP(SP),.PC(PC),
     .select_a(select_a),
     .select_b(select_b),
     .aout(ALUa),.bout(ALUb)
     );

   FLAG FLAG(
     .CLK(CLK),.RESET(RESETint),
     .ALUFLAGOUT(ALU_FLAGOUT),.IN(ALUOUT),
     .CTRL(FLAGctrl),
     .cset(cset),.nset(nset),.pvset(pvset),
     .hset(hset),
     .zset(zset),.sset(sset),.tcfset(tcfset),
     .IFF(IFF),.EXAF(EXAF),.BO(BO),.TCR(TCR),
     .FLAGOUT(FLAGOUT),.DBOUT(DBint)
     );

   TEMP8 TEMP8(
      .CLK(CLK),.RESET(RESETint),
      .IN(ALUOUT),.CTRL(TEMP8ctrl),
      .OUT(DBint)
      );
   TEMP TEMP(
     .CLK(CLK),.RESET(RESETint),
     .IN(ALUOUT),.DIGITIN(A[3:0]),
     .INTVlatch(INTVlatch),.RLD(RLD),.RRD(RRD),
     .CTRL(TEMPctrl),
     .DIGITOUT(TEMP_DIGIOUT),
     .OUT(TEMPOUT),.DBOUT(DBint),.ABOUT(ABint)
     );

   ACC ACC(
   .CLK(CLK),.RESET(RESETint),
   .IN(ALUOUT),.DIGITIN(TEMP_DIGIOUT),
   .RLD(RLD),.RRD(RRD),.CTRL(ACCctrl),.EXAF(EXAF),
   .ALUOUT(A),.DBOUT(DBint)
   );
```

```
  REG16 BC(
   .CLK(CLK),.RESET(RESETint),
   .IN(ALUOUT),.CTRL(BCctrl),.EXX(EXX),
   .ZERO16(BC0),.ZERO8(B0),
   .ALUOUT({B,C}),.ABOUT(ABint),.DBOUT(DBint)
   );

  REG16X DE(
    .CLK(CLK),.RESET(RESETint),
    .IN(ALUOUT),
    .CTRL(DEctrl),.EXX(EXX),.EXDEHL(EXDEHL),
    .EXBUSIN(EXBUS_HLOUT),.EXBUSOUT(EXBUS_DEOUT),
    .ALUOUT({D,E}),.ABOUT(ABint),.DBOUT(DBint)
    );

  REG16X HL(
    .CLK(CLK),.RESET(RESETint),
    .IN(ALUOUT),
    .CTRL(HLctrl),.EXX(EXX),.EXDEHL(EXDEHL),
    .EXBUSIN(EXBUS_DEOUT),.EXBUSOUT(EXBUS_HLOUT),
    .ALUOUT({H,L}),.ABOUT(ABint),.DBOUT(DBint)
    );

  INDEXREG ix(
      .CLK(CLK),.RESET(RESETint),
      .IN(ALUOUT),
      .CTRL(IXctrl),
      .ALUOUT(IX),.DBOUT(DBint)
      );

  INDEXREG iy(
      .CLK(CLK),.RESET(RESETint),
      .IN(ALUOUT),
      .CTRL(IYctrl),
      .ALUOUT(IY),.DBOUT(DBint)
      );

  SP sp(
.CLK(CLK),.RESET(RESETint),
.IN(ALUOUT),
.CTRL(SPctrl),
.ALUOUT(SP),.DBOUT(DBint),.ABOUT(ABint)
);

  PC pc(
.CLK(CLK),.RESET(RESETint),
.ALUOUT_IN(ALUOUT),
.HLIN({H,L}),.IXIN(IX),.IYIN(IY),.TEMPIN(TEMPOUT),
.CTRL(PCctrl),.ZEROPAGE(ZEROPAGE),
.ALUOUT(PC),.ABOUT(ABint),.DBOUT(DBint)
);

   I_R I_R(
   .CLK(CLK),.RESET(RESETint),
   .DBIN(DBint),
   .CTRL(I_Rctrl),
   .ABOUT(ABint),.DBOUT(DBint)
   );

   CTRL CTRL(
     .CLK(CLK),.RESET(RESETint),
     .FLAG(FLAGOUT),.DBIN(DBint),
     .WAIT(WAIT),.INT(INT),.NMI(NMI),.BUSREQ(BUSREQ),
     .BC0(BC0),.B0(B0),
     .ZEROPAGE(ZEROPAGE),.SETBIT(SETBIT),
     .ALUCTRL(ALUCTRL),
     .ACCCTRL(ACCctrl),.TEMPCTRL(TEMPctrl),
     .TEMP8CTRL(TEMP8ctrl),
     .RLD(RLD),.RRD(RRD),.INTVlatch(INTVlatch),
     .cset(cset),.nset(nset),.pvset(pvset),
     .hset(hset),
     .zset(zset),.sset(sset),.tcfset(tcfset),
     .IFF(IFF),
     .BCctrl(BCctrl),.DEctrl(DEctrl),.HLctrl(HLctrl),
     .IXctrl(IXctrl),.IYctrl(IYctrl),.SPctrl(SPctrl),
     .PCctrl(PCctrl),.I_Rctrl(I_Rctrl),
     .FLAGctrl(FLAGctrl),
     .EXDEHL(EXDEHL),.EXX(EXX),.EXAF(EXAF),
     .exALU(exALU),
     .relative_high(relative_high),
     .select_a(select_a),.select_b(select_b),
     .HALT(HALT),.MREQ(MREQ),.IORQ(IORQ),.RFSH(RFSH),
     .M1(M1),.BUSACK(BUSACK),.WR(WR),.RD(RD)
     );
endmodule // Z80_top
```

文　献

[1] “VDEC IP プロジェクトの現状と今後の展望” 浅田邦博電子情報通信学会技術研究報告 VLSI 設計技術 VLD2001-85-88

[2] “Z80 family users manual” Zilog 社 UM08001-1000

[3] Thomas Scherrer Home of the Z80 CPU “http://us.share.geocities.com/SiliconValley/Peaks/3938”