再構成可能集積回路の
効果的な応用に関する研究

システム情報工学研究科
筑波大学

2005年3月

相部 範之

要旨

本研究の目的は，再構成可能集積回路の１つである，FPGA (Field Programmable Gate Array) の「再構成可能」という特徴を生かせる応用（アプリケーション）を探索し，その有効性を実証することである．本論文では，４つの異なる側面からの適用例を用いて，FPGA の「再構成可能」という特徴を生かせるアプリケーションについて論じる．

FPGA は従来より，組み込み製品の製造数が少ない場合に，低コストで実現できるということから，ASIC の代替品として用いられてきた．本研究では確率的ニューラルネットワークを用いた画像認識装置，性能可変 IP パケット・フィルタ装置という２つのアプリケーションを FPGA で実現することを試みた．その結果，適切な処理アルゴリズムや通信プロトコルなどが未確定な応用や，優先する性能が変化する応用など，調整パラメータ数が非常に多い場合において特に優れた性能を発揮し，TAT(Turn Around Time) やコスト面で，ASIC より適した実装方式であることがわかった．

また，FPGA は従来より，専用集積回路開発におけるラピッド・プロトタイピング・ツールとして用いられてきた．本研究では，汎用 I/O インターフェースとハードウェアのオープンソース化という２つのアプリケーションについて研究を行った．その結果，単なるラピッド・プロトタイピング・ツールや ASIC の代替品ではなく，その汎用性と回路構成情報のソフトウェアとしての特徴を活かすことで，I/O インターフェースのユーザビリティの向上やハードウェア開発へのバザール型開発手法の導入を可能にするなど，新たな可能性があることを示した．

目次

図目次

表目次

第 1 章

序論

1.1 研究の目的

本研究の目的は，再構成可能集積回路の 1 つである，FPGA (Field Programmable Gate Array) の「再構成可能」という特徴を生かせる応用（アプリケーション）を探索し，その有効性を実証することである．本論文では，確率的ニューラルネットワークを用いた画像認識装置，性能可変 IP パケット・フィルタ装置，汎用 I/O インターフェース，そして，新しいハードウェア開発手法という，4 つの異なる側面からの適用例を用いて，FPGA の「再構成可能」という特徴を生かせるアプリケーションについて論じる．

FPGA などの再構成可能集積回路は 1980 年代より一般に，LSI 開発における，フォト・マスクを作成する前のロジック／タイミング検証（論理エミュレーション）を行うためのラピッド・プロトタイピング・ツールとして，また，製造数が少ない場合における ASIC (Application Specific Integrated Circuit) や ASSP (Application Specific Standard Product) などの代替として用いられてきた．そして，近年さらに，その「再構成可能」という特徴をハードウェア・アーキテクチャに取り込み，従来の逐次型 MPU (Micro Processor Unit) では実現が困難な処理を効率良く処理しようとする試みが注目を集めている．これはリコンフィギャラブル・コンピューティングとよばれ，特に算術演算の高速化を中心に研究が進められており [1]〜[38]，FPGA よりさらに算術演算向きなアーキテクチャの再構成可能集積回路が次々に開発されている [32][34][37]．一方，本研究では従来の一般的な再構成可能集積回路である FPGA を対象として，その再構成性を活かす応用について検討した．

FPGA は，実装対象となるアルゴリズムの並列度が非常に高い場合，現在注目されている算術演算向きの再構成可能集積回路よりも，高速な処理を実現できる可能性がある．また，I/O インターフェースなど，小規模で，高速な処理を要求する制御回路などのランダム・ロジックは，FPGA での実現に向いている．さらに，FPGA は既に市場に出てから 20 年以上経ってお

り，ASIC/ASSP などの代替として製品に組み込まれ，普及しているため，その応用であれば実用化もし易い．そこで，本研究では FPGA への並列度の高いアルゴリズムの実装，I/O インターフェースへの応用など行い，FPGA の「再構成可能」という特徴を生かせるアプリケーションの有効性を実証した．さらに，FPGA を用いた新しいハードウェア開発手法を提案し，IP (Intellectual Property) の流通への応用を行った．

1.2　論文の構成

以下，第 2 章でまず，再構成可能集積回路の構造と分類について述べる．次に，これまでの再構成可能集積回路の応用である，ラピッド・プトロタイピング・ツールとしての歴史と，ASIC の代替品としての歴史，そして，近年注目されつつある，リコンフィギャラブル・コンピューティングについて述べる．そして，第 3 章以降で今回適用した 4 つの例について述べる．第 3 章では，並列度の高いアルゴリズムである確率的ニューラルネットワークを，市販の FPGA に実装し，その処理速度がアプリケーションの要求速度に達するかを検証した結果について述べる．アプリケーションとしては画像認識を選択し，実用的な認識精度（80% 以上）をビデオレート（33.3ms）で実現可能であることを確かめた．第 4 章では IP パケット・フィルタ装置への適用を行い，ネットワーク装置向けに特化されていない現在の一般的な FPGA で，どの程度の処理性能（処理可能な IP アドレス数と処理速度）を達成できるかについて述べる．第 5 章ではこれまでにない応用の一つとして，I/O インターフェースへの適用を行い，その有効性について述べる．第 6 章では，FPGA のハードウェア・アーキテクチャ上での利用ではなく，新しいハードウェア開発手法という，従来のラピッド・プロトタイピング・ツールとしての利用を発展させた応用について述べる．最後に，第 7 章で 4 つの適用例からわかった，再構成可能集積回路（FPGA）の効果的な応用についてまとめる．

第 2 章

再構成可能集積回路

2.1　FPGA の構造と LUT による論理回路の実現

図 2.1　FPGA の外観

図 2.1 に代表的な再構成可能集積回路の一つである，FPGA の外観を示す．FPGA は外見上は他の集積回路（ASIC/ASSP/MPU など）と何ら変わらず，パッケージにもざまざまなものがある．また，他の集積回路と同様の最新製造技術で製造される．再構成可能集積回路が他の集積回路と大きく異なるのは，製造された集積回路がそのままでは特定の機能を果たさず，回路構成情報を書き込む（プログラミングする）ことで初めて動作する，という点である．再構成可能集積回路は，この，後からプログラミング可能ということによって，さまざまなアプリケーションに容易に対応することができる．

図 2.2 に代表的な FPGA の構造（模式図）を示す．FPGA は図のように，均一な構造をし

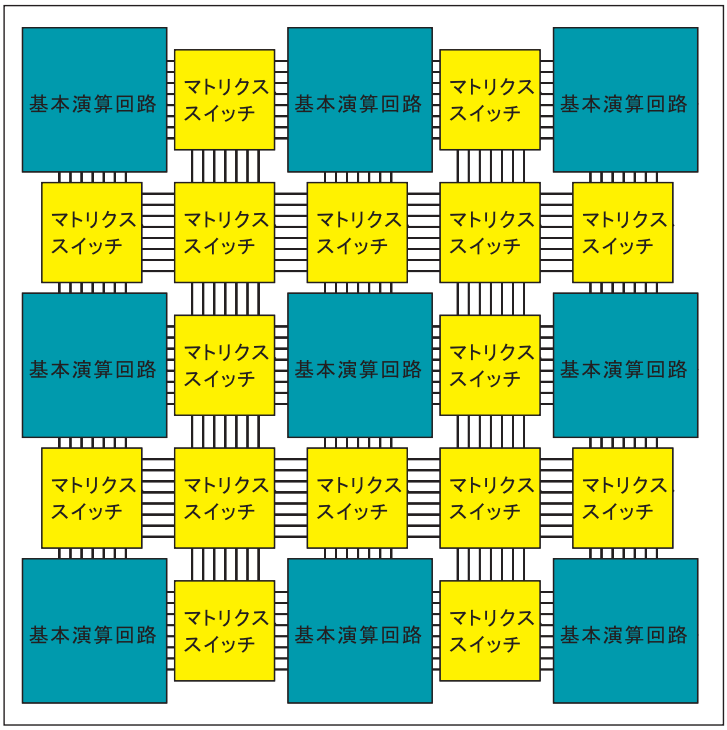


図 2.2　代表的な FPGA の構造（模式図）

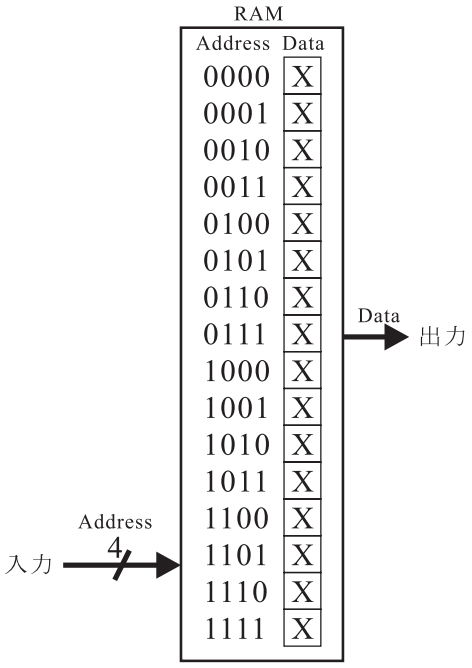


図 2.3　4 入力 1 出力 LUT の構造（模式図）

ており，その構成要素は，LUT (Look Up Table) と呼ばれるメモリとフリップ・フロップから成る基本演算回路及び，その基本演算回路同士を結ぶ配線とその接続を切り替えるマトリクス・スイッチによって構成される．この LUT のデータやマトリクス・スイッチの接続状態は外部から，PC などで構成されたプログラミング装置により，書き換えることができる．図 2.3 に FPGA の LUT の模式図を示す．LUT は，予めデータを書き込んだ RAM のアドレス入力を，データ入力として扱うことで，結果的に書き込まれたデータに従った，任意の組み合わせ回路を実現するものである．現在の主流な FPGA の LUT は 4 入力 1 出力となっており，これにより 2 入力 OR ゲート，3 入力 EXOR ゲート，4 入力 NAND ゲート，など 4 入力 1 出力までの任意のロジック・ゲート（真理値表）を実現できる（図 2.4， 2.5， 2.6）．より大きな組み合わせ回路を必要とする場合には，マトリクス・スイッチを介して，他の LUT と接続することで実現可能である．さらに一般的な FPGA の基本演算回路はフリップ・フロップを持っており，LUT と組み合わせることで複雑な順序回路が実現できる．図 2.7 に具体例として XILINX Virtex シリーズ [65] の基本演算回路（1SLICE）の主要部分を示す．XILINX Virtex シリーズの基本演算回路は，4 入力 1 出力の LUT を 2 個，1bit の D フリップ・フロップを 2 個に，キャリー・ロジック用のパス（図中では省略）とそれらの接続を切り替えるいくつかのマルチプレクサから成る．2005 年現在主流な XILINX 社製の FPGA は，このような基本演算回路を小規模なもの（XC2S15[68]）で 192 個〜大規模なもの（XC2VP100[70]）176,384 個持っており，実現可能な総ゲート数は数万〜数百万ゲート以上と言われている [70].

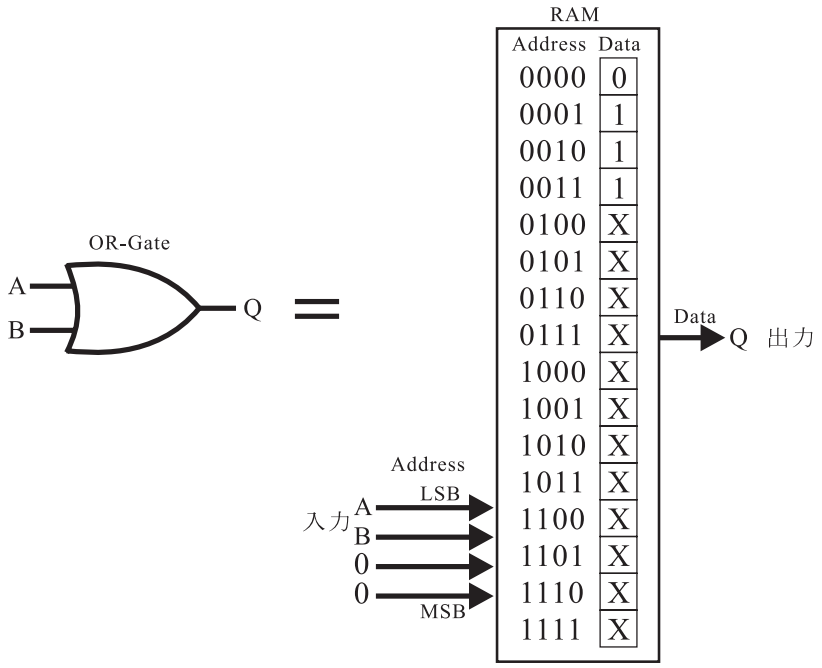


図 2.4　LUT による 2 入力 OR ゲート

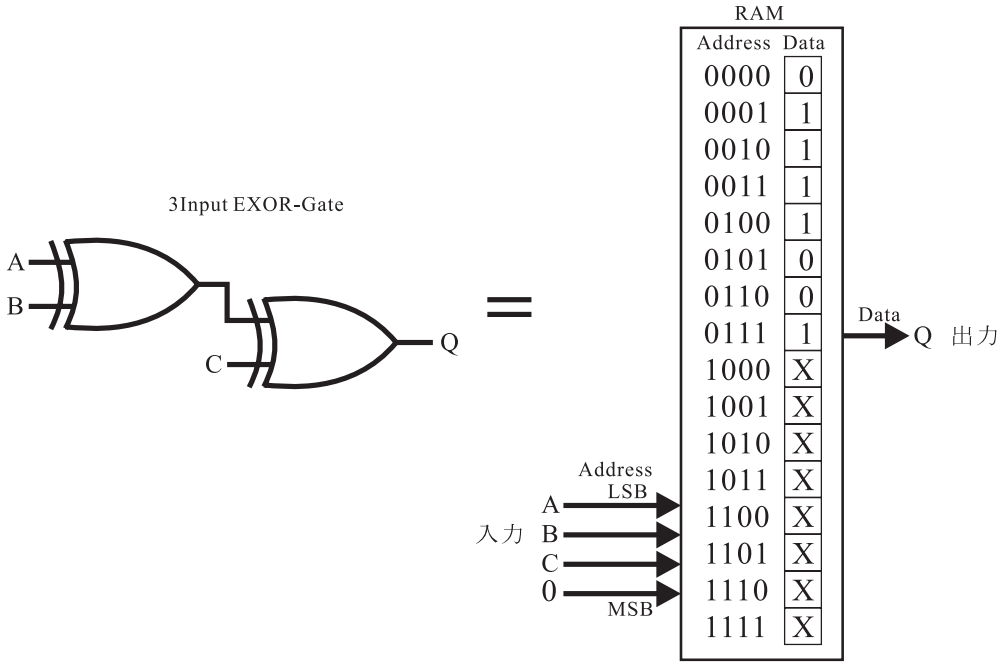


図 2.5　LUT による 3 入力 EXOR ゲート

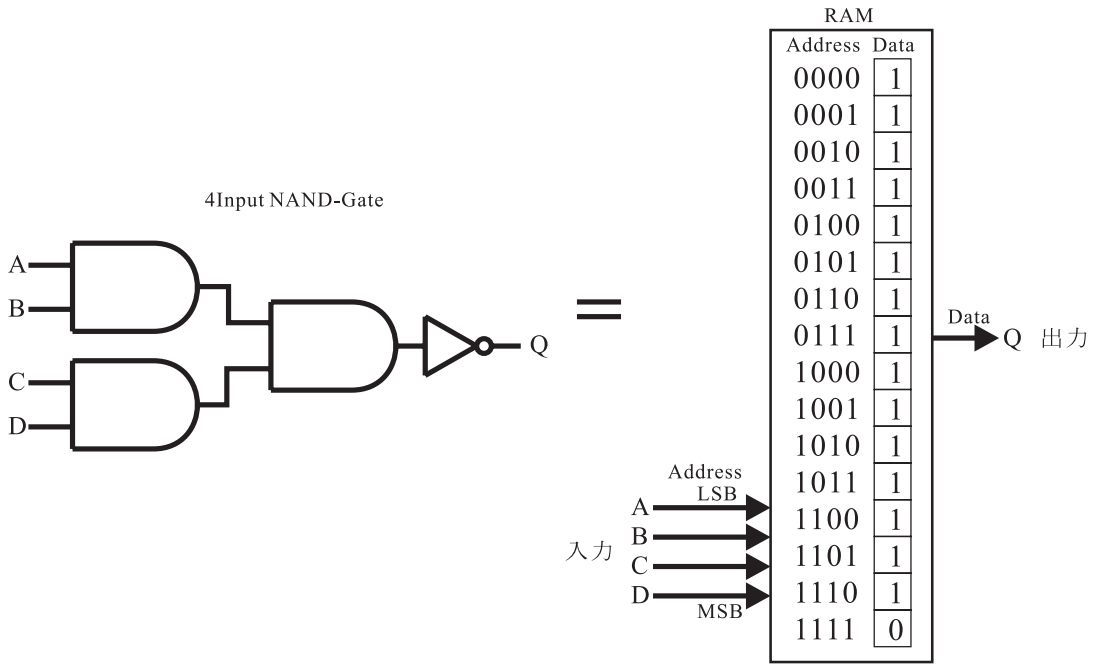


図 2.6 LUT による 4 入力 NAND ゲート

2.2 LUT アレイ型と ALU アレイ型

現在の再構成可能集積回路は主に，図 2.2 の基本演算回路の種類によって，大きく LUT (Look Up Table) アレイ型と ALU (Arithmetic Logic Unit) アレイ型に分類できる（図 2.8）．LUT とは，前節で述べたように，予めデータを書き込んだ RAM のアドレス入力を，データ入力として扱うことで，結果的に書き込まれたデータに従った，任意の組み合わせ回路を実現するものである．従ってその実装形式は SRAM(Static RAM) そのものであり，現在の主流な FPGA の構造である．ALU は，従来から MPU に用いられている算術論理演算回路のことで，AND，OR，NOT などの論理演算に加え，加算，減算，シフト演算，或いは掛け算などの算術演算を行う回路を一つにまとめ，外部からの命令により指定の演算を行うものである．実装形式は通常，トランジスタ・レベルで直接設計・製造される．LUT アレイ型と ALU アレイ型は，粒度という面から見ると LUT アレイ型が細粒度で，ALU アレイ型が粗粒度である．また，従来の単一の ALU のみで構成された逐次処理型の MPU は，ALU アレイ型よりもさらに粒度が粗いと考えることが出来る（表 2.1）．

まず，LUT アレイ型は，基本要素となる回路を，LUT という NAND ゲートのように汎用性の高いものとし，これを通常のゲートアレイのように多数並べて，その接続先を多数のスイッ

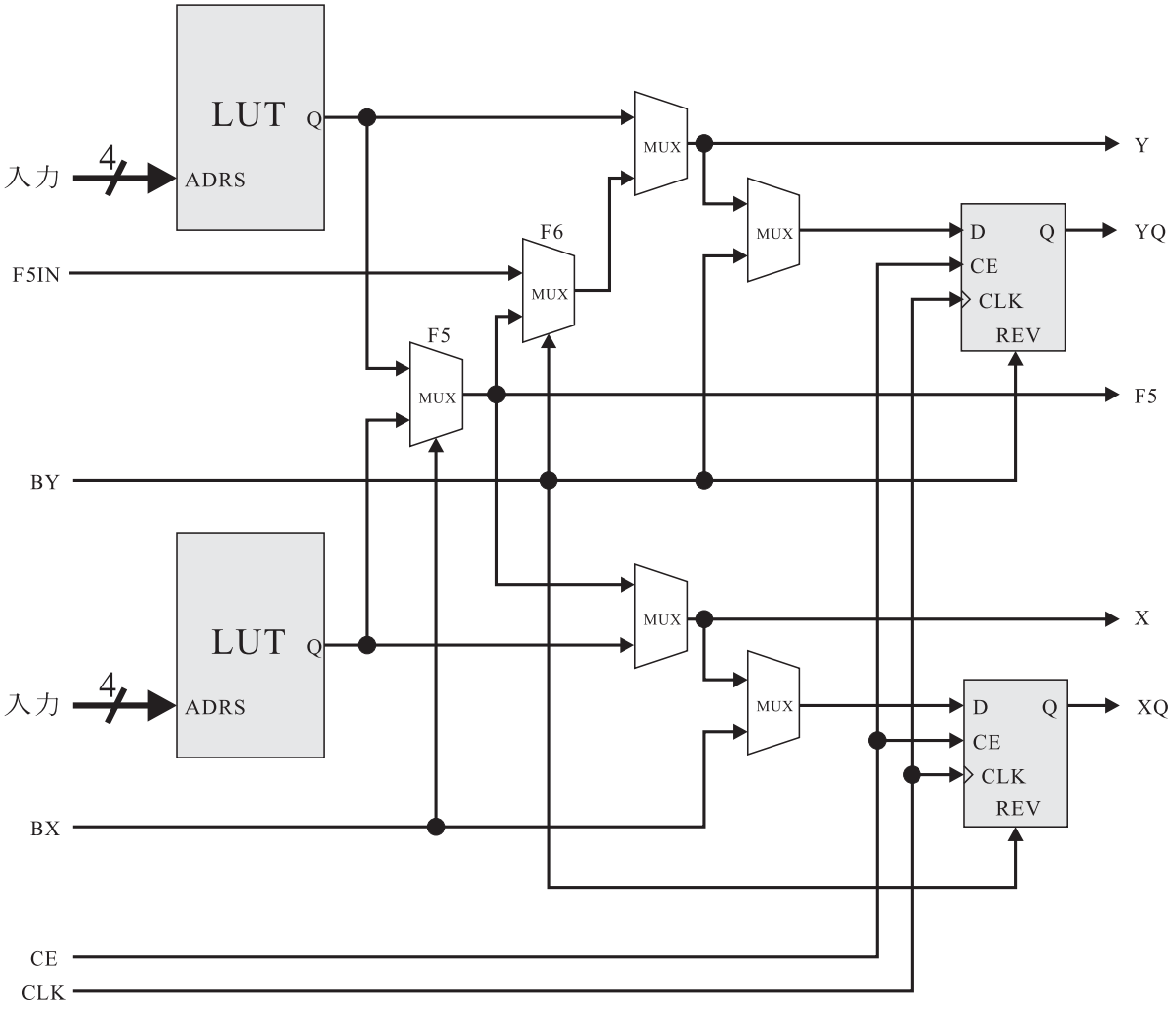
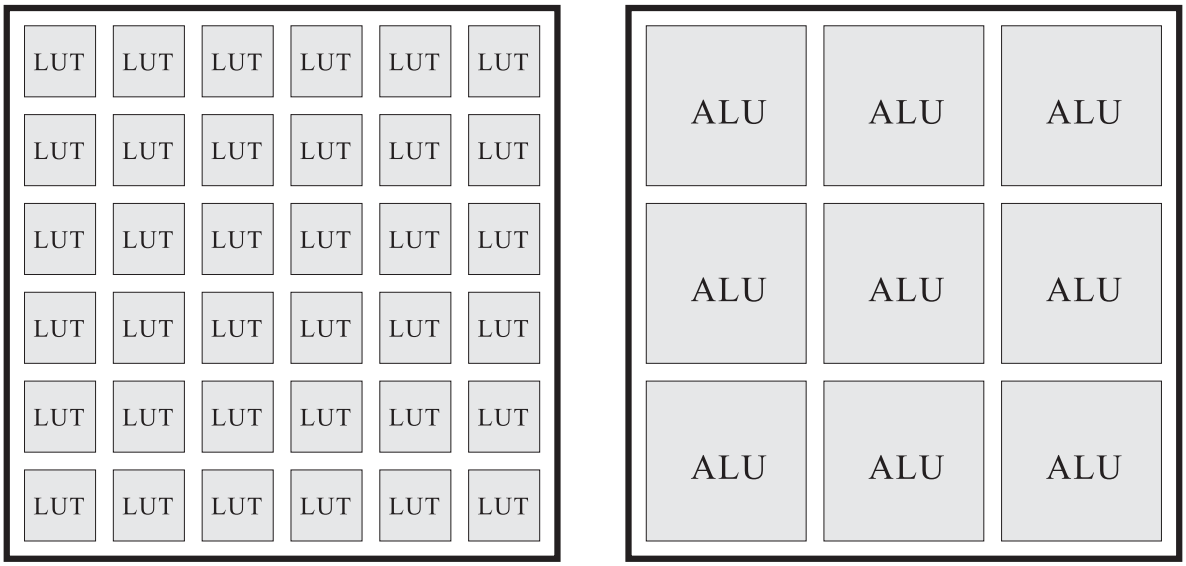


図 2.7　XILINX Virtex シリーズの SLICE（基本演算回路）

表 2.1　実装方式の分類

	LUT アレイ型	ALU アレイ型	MPU
例	XILINX Virtex[65], Spartan[67], ALTERA Stratix[73], Cyclone[74]	広島大 FPAccA[32], NEC DRP[37] IP–Flex DAP/DNA[34]	Intel x86 系, IBM PowerPC 系, Z80, H8, ARM
粒度	細かい	粗い	さらに粗い
適した設計レベル	RTL	演算命令レベル	Behavior Level



(a) LUTアレイ型　　(b) ALUアレイ型

図 2.8　LUT アレイ型と ALU アレイ型

チで切り換える構成（細粒度の構成）で，非常に汎用性の高い再構成可能集積回路である．しかし，この構成では，アプリケーションによっては要求される基本要素回路，スイッチの数が非常に多くなり，信号遅延が大きくなることで，全体の動作周波数を高くできない．XILINX 社や ALTERA 社製の FPGA や CPLD がこの構成にあたる．ALU アレイ型は，ALU という，ある程度大きな演算回路を基本回路としたもの（粗粒度の構成）で，必要なスイッチ数，基本要素回路数共に少なくなり，全体の動作周波数が高められる．しかし，粒度が粗いため，アプリケーションによっては実装が困難となる．このような構成としては，広島大学 越智の FPAccA (Field Programmable Accumulator Array)[32][33]，IP-Flex 社製の DAP/DNA[34] や NEC エレクトロニクス社製の DRP[37] などがある．また，従来の x86 系，PowerPC 系，或いは組み込み機器向けの Z80， H8， ARM など，ノイマン型アーキテクチャの MPU は，一般に単一の ALU で構成される．このため，前述の再構成可能集積回路と異なり，複雑な切り替え機構などがない分，信号遅延が小さく，最も高い周波数での動作が可能となる．しかし，最も演算器の粒度が粗い（演算器が１つしかない）ため，特に並列処理を基本とするアプリケーションは実装効率が悪い．

2.3　２つの再構成方式

再構成可能集積回路は，その回路構成情報の再構成方式によって，静的再構成と動的再構成でに分類できる（表 2.2）．この再構成方式による分類は厳密に行うことは難しいが，再構成（回路構成情報の読み込み）を電源投入時にのみ、一度だけ行うものを静的再構成，電源供給中で回路の一部の動作中に行うものを動的再構成と呼ぶことができる．また読み込まれる回路構成情報は静的再構成の場合には通常１種類であるが，ユーザが一度電源を切って，異なる回路構成を指定し，再び電源投入を行うようなことも考えられる．従って静的再構成では，次の再構成が行われるまでの時間間隔はそれほど短くはなく，数 ms 以上はかかると考えられる．一方，動的再構成は必ず複数の異なる回路構成情報による書き換えが行われる．動的再構成では一連の処理の実行の中で再構成を行うことが考えられるため，再構成に要する時間は短い程良い．ただ，同一の回路構成で処理される時間は長い程，その構成の利用効率は高くなるため，再構成される時間 “間隔” は短い方が良いというわけではない．さらに，回路構成が変更される頻度と全体の処理時間はトレードオフの関係にある場合が多く，その結果，動的再構成で利用される再構成の時間間隔はおおよそ数 us〜数 ms となる．

表 2.2　再構成方式の分類

方式	再構成のタイミング	回路構成情報の種類	再構成の時間間隔目安
静的再構成	電源投入時	１種類，若しくは複数	数 ms 以上
動的再構成	電源供給中 （回路の一部が動作中）	複数	数 us〜数 ms

静的再構成は，従来からの FPGA の利用で広く用いられてきた方式である．現在主流な XILINX や ALTERA 社の FPGA は通常，外部のコンフィギュレーション・メモリに書き込まれた回路構成情報が，電源投入時に読み込まれる．このため，ASIC の代替品として利用される場合には，予めコンフィギュレーション・メモリに，目的の回路構成情報が書き込まれて製品出荷される．ただし，ASIC の場合と異なり，このコンフィギュレーション・メモリを書き換えることで，製品出荷後に，ユーザーサイドで回路構成の修正・機能追加などを行うことができる．また，常に同一の回路構成情報を読み込むのではなく，異なる回路構成情報を必要に応じて切り換えて使用することで，１つのハードウェア資源を複数の用途で共有化することができ，結果的に回路面積の縮小が図れる．

動的再構成ではさらに，これらの利点を回路動作中にも適用できるため，実行時に要求され

る回路実装面積の縮小が図れる．ただし，動的再構成では回路を再構成する時間が重要となり，現在の市販の XILINX や ALTERA 社の FPGA は再構成時間が長いため，アプリケーションによってはこの時間がボトルネックとなり利用できない．そこでこの再構成時間の短縮を図るために，回路構成情報を従来の FPGA のように外部のメモリに持たせるのではなく，再構成可能集積回路内部の分散メモリに持たせた構成で，かつ複数の回路構成情報を即座に切り換え可能とした方式（マルチコンテキスト方式）が研究されている [36]．また，演算器の粒度が粗くなるに従い，回路構成情報の情報量も減るため，ALU アレイ型など粗粒度な構成への適用が行われており，このような構成を持つのが NEC DRP[37] や IP–Flex DNA/DAP[34] などである．

2.4　ラピッド・プロトタイピング・ツールとしての歴史

現在大きく世界シェアを占めている FPGA のメーカは XILINX 社と ALTERA 社で，共に米国のメーカである．XILINX 社が初めて市場に FPGA を販売したのが 1985 年で，XC2064 という 1,000 ゲート規模の FPGA であった．一方 ALTERA 社は 1984 年に EP300 という 300 ゲート規模の PLD (Programmable Logic Device) を製品化している．FPGA と PLD (CPLD: Complex-PLD) はその構造に違いがあり，それぞれに長短があるため，現在では両社共 FPGA と PLD (CPLD) を開発・販売している．これら FPGA や PLD (CPLD) は，1980 年代後半より今日まで，フォト・マスク・ベースの LSI を製造する際の動作検証ツール（ラピッド・プロトタイピング・ツール）として利用され，発展してきた．

ASIC などのフォト・マスク・ベースの LSI を製造する場合，通常図 2.9 のような工程で製造される．PC (Personal Computer) 若しくは WS (Work Station) 上の CAD (Computer Aided Design) で回路設計，LSI のレイアウト設計を行った後，同じく PC/WS 上のシミュレータにより論理検証，タイミング検証，さらには製品の利用される状況（制御対象の状況，設置環境等）をモデル化したシステムレベルでの動作シミュレーションまで行われる．シミュレーションの結果，設計上の不良個所が発見されれば，それを設計に反映し，再度シミュレーションが行われる．この設計・検証サイクルの後，不良個所が除かれた設計図を元に，半導体製造工場にて LSI が製造される．製造された LSI はさらに製品に組み込まれて動作検証が行われるが，このとき途中工程で用いられたシミュレーションのモデルと，実際の製品の状況との間には相違点（シミュレーション・モデルの誤差や，考慮されていない物理現象や制約によって生じる差異）があるため，多くの場合は再度設計の修正，製造が必要となる．このことは開発コストを浪費するばかりでなく，設計開始から，最終製品出荷までに要する時間，いわゆる TAT(Turn Around Time) を長くしてしまう．

一方，FPGA によるラピッド・プロトタイピングを用いた場合の，フォト・マスク・ベース LSI の製造工程を図 2.10 に示す．PC/WS 上の CAD により回路設計を行うところまでは従来と同様だが，従来は回路の動作検証を PC/WS 上でシミュレートしていたのに対し，FPGA を用いた開発では FPGA 上で回路をエミュレートすることで検証を行う．FPGA による論理エミュレーションは，実行時間が PC/WS 上のシミュレーションに比べ遥かに高速であり，PC/WS 上のシミュレーションでは処理時間がかかって検証が困難であった大規模な回路検証も可能である．さらに，シミュレーションと大きく異なることは，FPGA によるエミュレーションでは，より最終製品に近い状況での動作検証が可能であるということである．例えば実際の製品基板上で他の周辺 LSI との接続試験を行う，イン・サーキット・エミュレーションが可能である．このようなことから，FPGA によるエミュレーションでは，より正確に設計上の

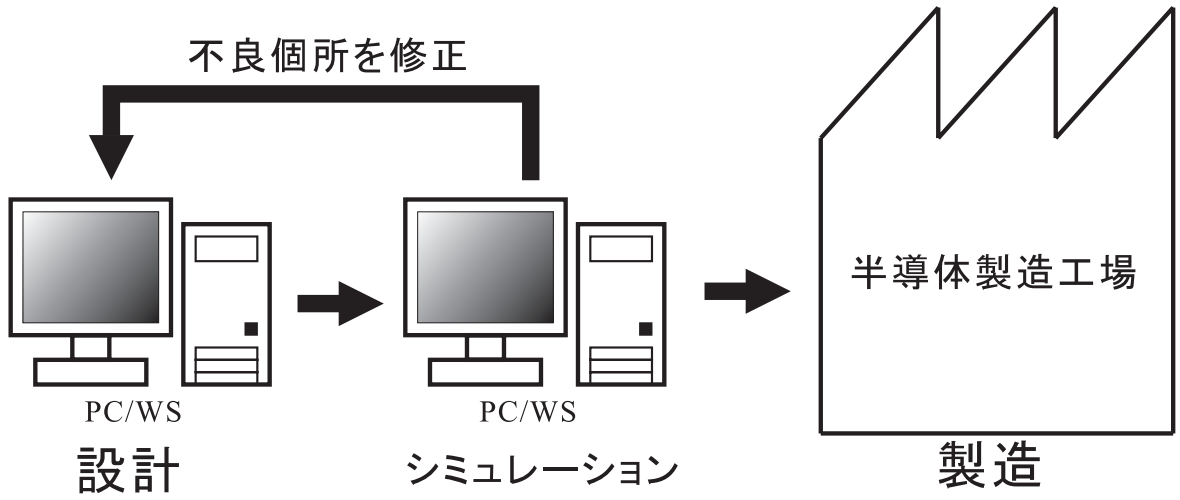


図 2.9　フォト・マスク・ベース LSI の製造工程

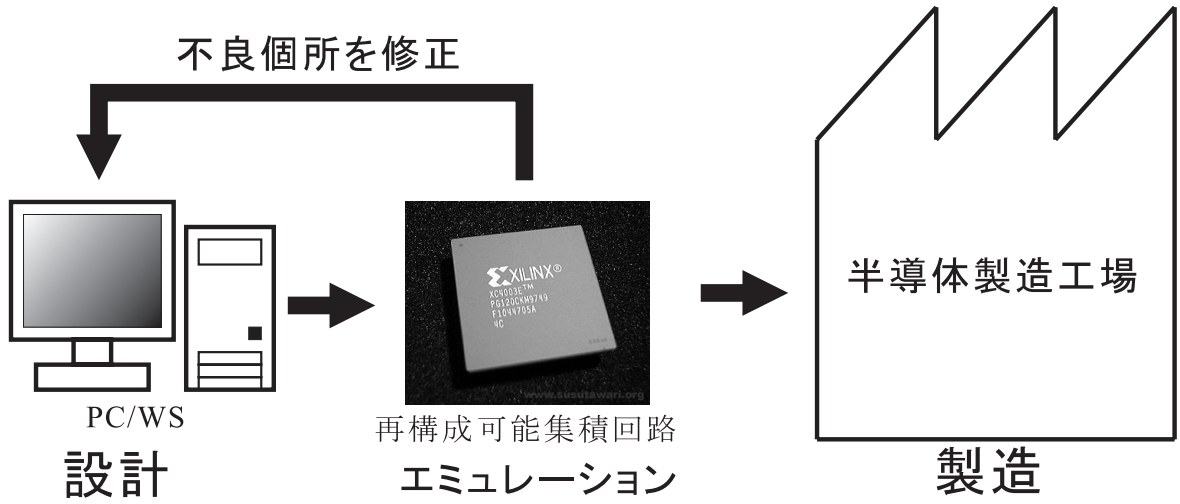


図 2.10　FPGA によるラピッド・プロトタイピング

不良個所修正を行うことができ，結果的に，シミュレーションに比べ TAT を短縮することができる．

このように FPGA は単体でもラピッド・プロトタイピング・ツールとして機能するが，その上に実現可能な回路規模は，他の専用集積回路（ASIC/ASSP/MPU など）に比べ，どうしても小さくなってしまう．これは，同一の製造技術で製造されたとしても，FPGA には汎用性

を実現するための回路（構造）が余計に必要なためである．このため，より大規模な専用集積回路のエミュレーションを可能にするために，複数の FPGA を 1 つの大規模 FPGA として扱えるシステムが必要となる．このような大規模なラピッド・プロトタイピング・ツール（論理エミュレータ）は実際に製品化されており，ASIC/ASSP/MPU の開発で利用されている．例えば，アスキー VM 社製の論理エミュレータ Reale は 1998 年に販売され，ALTERA 社製の FPGA である FLEX EPF10K100[72] を 46 個搭載し，実動動作周波数 30MHz，最大検証可能ゲート数は ASIC 換算で約 140 万ゲートであった．

2.5　ASIC の代替品としての歴史

ある特定のアプリケーションを実現するために，速度，サイズ，消費電力の全ての性能面で最も高い性能を達成するハードウェアを製造する方法は，フル・カスタムのフォト・マスク・ベースで製造する方法である．これは一般的に特定用途向け集積回路 ASIC(Application Specific Integrated Circuit) と呼ばれる．しかし，ASIC はその都度フォト・マスクから設計，製造を行う必要があるため，非常に高度な設計・製造技術が必要となる他，開発・製造期間が長く，開発・製造コストも高い．フォト・マスク製造の期間を短縮し，また製造コストを削減する目的で，フル・カスタム方式の他にスタンダード・セル方式，ゲートアレイ方式，或いはストラクチャード ASIC などがある．しかし，予め製造しておけるのはフォト・マスク・レベルであり，工場で製造する必要があることに変わりない．一方，既存の MPU を用いて処理する場合は，予め MPU を搭載したシステムを作り上げた後で，任意のアプリケーションを実装することができる．従って，その実行性能が多少落ちたとしても，開発期間やコストを含めて評価したときに，MPU での実現が採用される場合が多い．この TAT ／コストに対する実行性能（速度，サイズ，消費電力）を考えると，再構成可能集積回路は ASIC による実現と，MPU による実現の丁度中間に位置する（図 2.11）．従って，製品開発では，アプリケーションの要求性能と，TAT ／コストから適切な “アプリケーションの実現方法” を選択する必要がある．

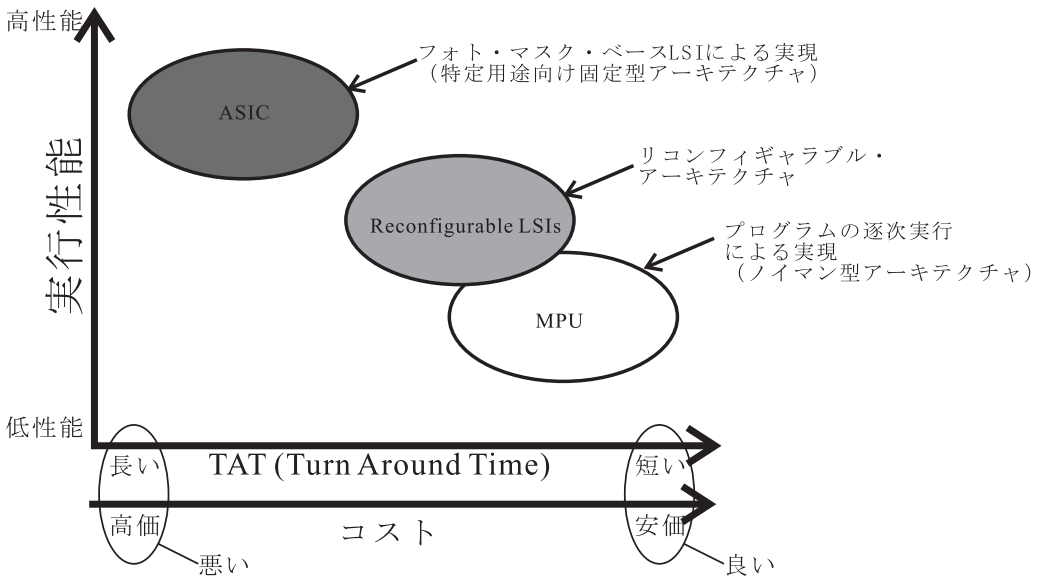


図 2.11　TAT ／コストにおけるリコンフィギャラブル・アーキテクチャの位置付け

FPGA をラピッド・プロトタイピング・ツールとして用いた場合，「プロトタイピング」（試

表 2.3　FPGA/PLD が製品実装された例

採用年度	製品種類	製品メーカ	製品型式	使用 FPGA/PLD
1990	PC 用ビデオカード	Matrox	Raibow Runner	XILINX XC5204
1992	PC 用 CD-ROM ドライブ	Commodore	A570(A690)	XILINX XC2064-50PC68C
1993	ギガビット・スイッチ	Cisco Systems	Catalyst 5000	ALTERA EPF10K10/30[72]
2002 (〜2004)	エレキ・ギター	Gibson Guitar	–	XILINX Spartan-2E[69]/3/ ALTERA Cyclone[74]
2002	簡易血中アルコール検出器	ACS	ELAN	XILINX CoolRunner
2003	自動車制御装置	BMW Williams-F1	FW25 5th VCM	XILINX XCV600E[66]
2004	業務用ビデオカメラ	Panasonic	AG–DVX100A	ALTERA Cyclone[74]
2004	火星探査ロボット用制御装置	NASA JPL	Spirit Rover/ Opportunity MER Rover	XILINX XQVR4000XL

作）から最終製品製造に置き換わる時期は，その LSI が組み込まれる製品の要求性能，マーケット，出荷台数などに大きく影響される．まず，動作速度，回路規模，消費電力などの性能面で，FPGA の性能が最終製品の要求性能に達していなければ，FPGA は完全に試作品としての役割しか果たさない．従って，FPGA を用いて開発した回路図を元にフォト・マスク・ベースの LSI を製造し，完成した LSI が製品に組み込まれて出荷される．ところが，FPGA が先の要求性能を満たしている場合には，FPGA は単なる試作品ではなく，フォト・マスク・ベース・LSI の代替品となり得る．この場合，出荷製品に FPGA をそのまま組み込むことができ，メーカはとりあえず FPGA を組み込んで出荷し，第二ロット製品からフォト・マスク・ベース・LSI に置き換えるということが可能である．この場合のメーカにとっての利点は，開発が完了した時点でフォト・マスク・ベース・LSI の製造を待たずに製品出荷が可能である，つまり TAT をさらに短縮できるという点である（図 2.12）．一方，短所としては一般に FPGA の単価が ASIC などフォト・マスク・ベース・LSI の単価に比べ高いということである．ただし，各 LSI の単価には初期製造コストも含まれ，フォト・マスク・ベース・LSI では特にこの初期製造コストが FPGA に比べ非常に高いため，メーカは製品の種類や市場規模に伴う出荷台数に応じて，FPGA とフォト・マスク・ベース・LSI のどちらを組み込むかを選択する必要がある（図 2.13）．一般に出荷台数が数台〜数百台と少ない場合には FPGA の方が単価が安い．従って，出荷初期には FPGA を使用し，販売数に応じてフォト・マスク製造に移行するということも可能である．そもそも ALTERA 初の PLD である EP300 は，油井用器具制御盤に組み込む目的で開発されたもので，初めから製品に組み込まれる目的であり，ラピッド・プロトタイピング・ツールとして生まれたものではなかった．表 2.3 に実際に，コンシューマ製品や非民生機器に FPGA/PLD(CPLD) が採用された例を示す．

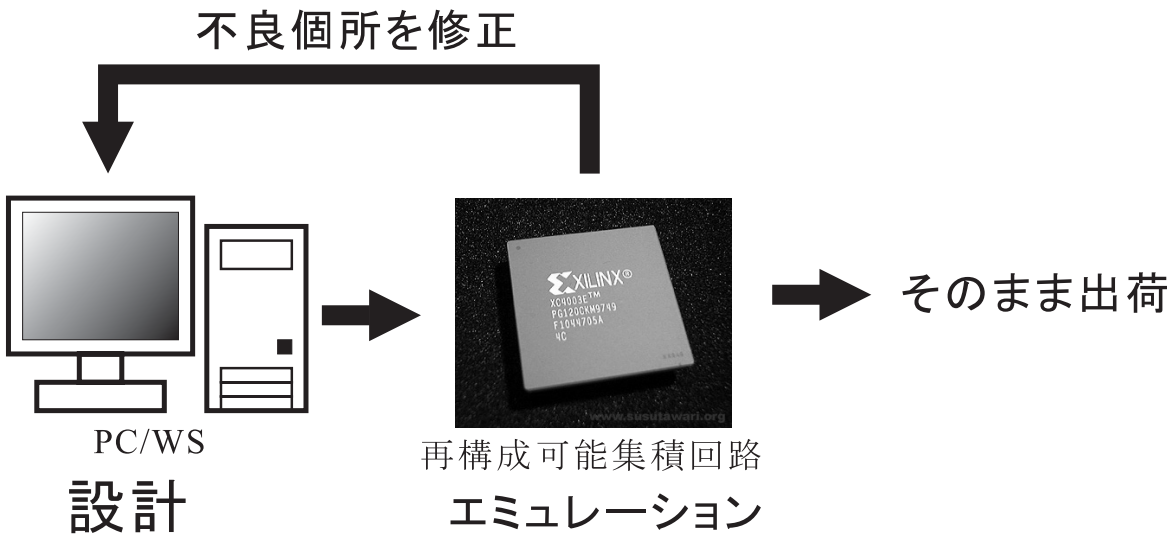


図 2.12　FPGA によりさらなる TAT 短縮可能な工程（ASIC の代替）

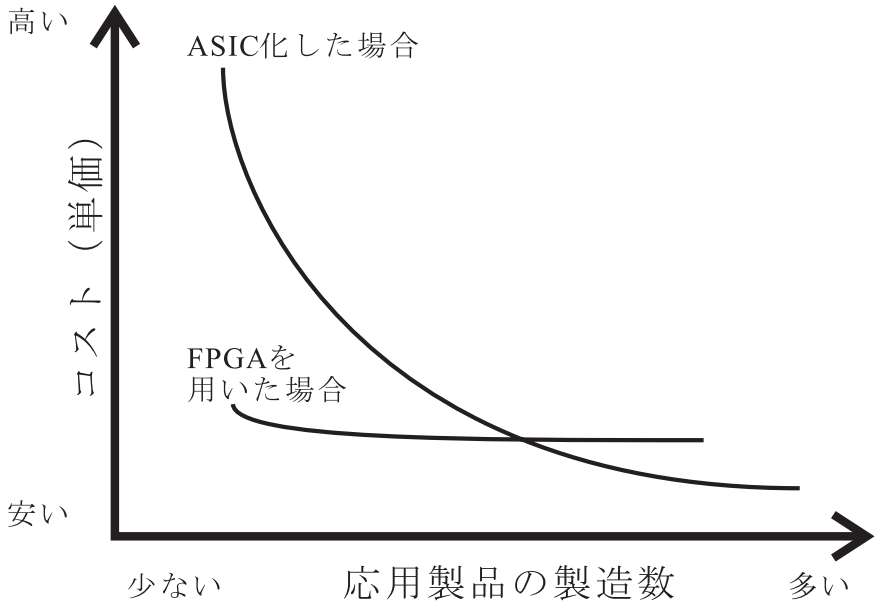


図 2.13　ASIC 化した場合と FPGA を採用した場合の応用製品の製造数とコスト

2.6　リコンフィギャラブル・コンピューティングの歴史

近年，FPGA などの再構成可能集積回路を用いて実現される，リコンフィギャラブル・コンピューティングの研究が盛んに行われている [1]〜[40]．リコンフィギャラブル・コンピューティングとは，再構成可能集積回路の「再構成機構」をハードウェア・アーキテクチャとして取り込み，アプリケーションに適した計算方式を実現するものである．このように「再構成機構」を持つことで，全く異なるアプリケーションに対応可能な汎用ハードウェア・アーキテクチャを，本論文では，リコンフィギャラブル・アーキテクチャと呼ぶ．またこれに対し，「再構成機構」を持たない従来のアーキテクチャを固定アーキテクチャと呼んで区別する．ここでは，この区別についてもう少し詳しく述べる．

集積回路の製造後に，論理回路レベル（RTL）での回路構成の大きな変更の必要がない場合には，複数の要求を満たす回路全てをワイヤード・ロジックで実装し，要求に応じて切り換えられるような回路構成で ASIC 化すれば，最も高性能なハードウェアが実現できる．しかし，要求に応じて回路の構成を大きく変える必要がある場合には，各回路の独立性が高くなる（共有可能部分が減少する）ため，それぞれの回路をワイヤード・ロジックで実装すると，実装に必要な回路規模が大きくなってしまう．さらに，ASIC 化の時点で想定されていなかった全く新しい機能を後から追加するようなことは，特定用途向けの固定アーキテクチャではまず不可能である．このような想定外のことにも対応できるような汎用性を持たせることは，即ちリコンフィギャラブル・アーキテクチャを採用することとなる．広義の意味においては，要求に応じて論理回路レベルでの構成が少しでも切り換えられるような構成は，全てリコンフィギャラブル・アーキテクチャであると言える．例えば外部からの入力信号で切り換え可能な 1 つのマルチプレクサを持つ回路は，1-bit のコンフィギュレーション・データによる再構成可能回路と見なすことができる．しかし，本論文では現在の再構成可能集積回路の LUT アレイ型アーキテクチャ，或いは ALU アレイ型アーキテクチャのように，汎用性の高い基本演算回路を多数並列実装し，後から想定外の全く異なる機能を実現できる，汎用アーキテクチャをリコンフィギャラブル・アーキテクチャと定義する．

再構成可能集積回路は，特定の回路を ASIC 化する場合と同様の最新の製造技術で作られている．しかし汎用性の高い回路を基本要素として多数実装し，その基本要素となる回路同士の接続を自由に切り替えられる機構は，大きな回路面積を必要とする．このため，再構成可能集積回路は，どうしても対象の回路を直接フォト・マスク・ベースで実現する場合（ASIC 化する場合）に比べ，集積密度が低くなり，速度も遅く，消費電力も余計に増加する．つまり，再構成可能集積回路は汎用性を得る代わりに，速度，サイズ，消費電力を犠牲にしているといえる．従って，異なるアプリケーションにどの程度対応できるかという “汎用性” と，速度，サイ

ズ，消費電力などの実行性能を考えると，図 2.14 のようになる．ここで，汎用性において，リコンフィギャラブル・アーキテクチャと，ノイマン型アーキテクチャは，共に互いを実現できることから，同等の汎用性があると言える．しかし，実行性能に関してはアプリケーションに依存して異なり，並列性の低いアルゴリズムを処理する場合はノイマン型アーキテクチャの方が実装効率が高い場合が多い．

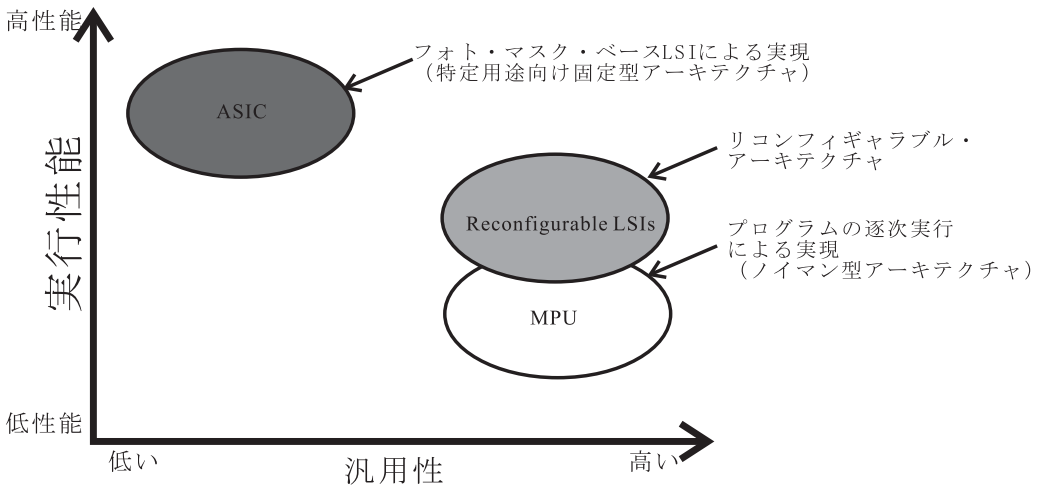


図 2.14　リコンフィギャラブル・アーキテクチャの位置付け

初期のリコンフィギャラブル・コンピューティングとして著名なのは，米国の旧 SRC (Supercomputing Research Center)（1995 年に Center for Computing Sciences[*1] と改名）の 1988 年の Splash 1 及び 1991 年の Splash 2[39][40] である．当時このような演算システムは Custom Computing Machine (CCM) と呼ばれていた．共に複数の XILINX 社製の FPGA で構成され，Splash 1 は 32 個の XC3090 から，Splash 2 は 17 個の XC4010 から成っていた．図 2.15 に Splash 2 の一つのボードの回路構成を示す．各 FPGA は 36-bit 幅のクロスバ・スイッチ（TI SN74ACT8841 が 9 個で構成）で接続されている他，隣同士が 36-bit で，さらに全 FPGA が 32-bit でバス接続されている．また各 FPGA はそれぞれローカルメモリとして，256K × 16-bit (4,096K-bit) の SRAM を持っており，バス接続に使用されているピンのうちの 16-bit が，このメモリとの接続に使用されている．従って総メモリ容量は 8.5M-Byte(4,096K-bit × 17) である．また XC4010 1 つあたりの CLB 数は 400 なので，16 個では 6,400 CLBs となる（17 個あるうちの X0 は制御用に使用されるため除く）．さらに FPGA が 17 個実装されたボードを最大 16 枚でシステムを構築することができる．これは約 256 万ゲート規模（FPGA 換算）の論理エミュレータといえる．しかし，Splash 1 および 2

[*1] http://www.super.org/

は，ラピッド・プロトタイピング・ツールとして開発されたのではなく，従来の逐次型プロセッサを用いたノイマン型アーキテクチャ・コンピュータに代わる，新しいコンピュータ・アーキテクチャとして提案された．ベンチマーク・アプリケーションとして，シストリック・アレイ型の DNA シーケンス，文字列検索，指紋照合などのパターン・マッチ，またメディアン・フィルタ，ガウシアン／ラプラシアン・ピラミッド生成，ハフ変換，2 次元 FFT などの画像処理が行われ，その高速性能が示されている．しかし，これらのアプリケーションは Splash 2 のアーキテクチャの性能を活かしきっておらず，その後もキラー・アプリケーションが見つからなかったために，Splash 2 のプロジェクトは 1994 年で終了している．またノイマン型アーキテクチャである Intel の 8086 系 MPU の処理能力が飛躍的に向上し，パターン・マッチや画像処理のように，本来逐次処理に向かない処理をこなすようになったことも，CCM（初期のリコンフィギャラブル・コンピューティング）研究衰退の原因である．他の例としては，2000 年になって，慶応大学の宇野，柴田，天野らが行った NEC 製の DRL (Dynamically Reconfigurable Logic) の評価報告 [36] の中で，Van der Pol の常微分方程式，及び 4-Queen 問題をベンチマーク・アプリケーションとした結果，従来のノイマン型アーキテクチャの安価な PC よりもよい速度性能を得られなかったことを述べている．

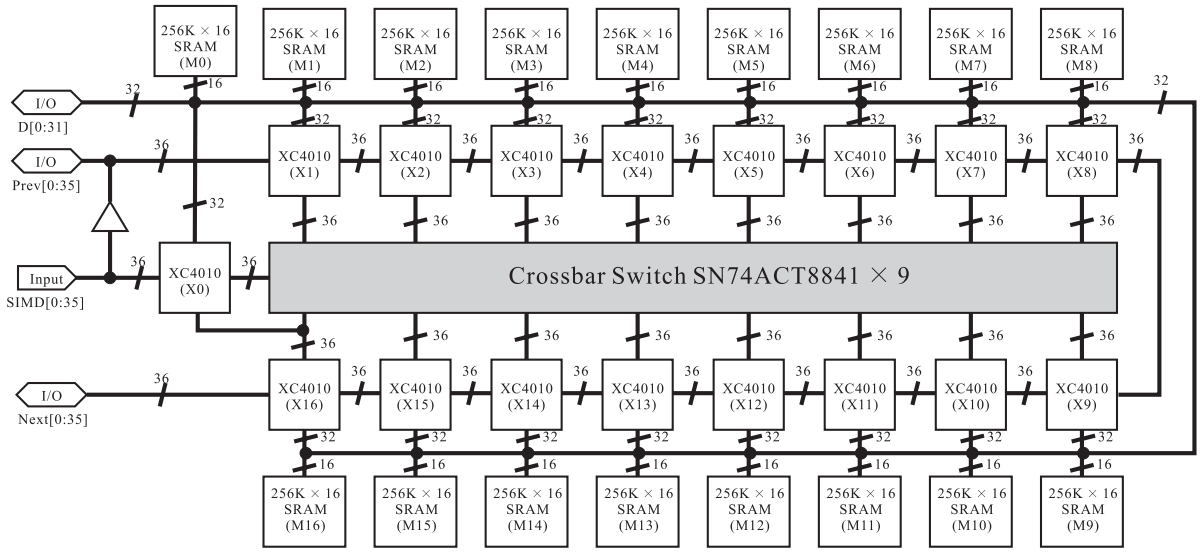


図 2.15　Splash2 の回路構成

しかし，1997 年に広島大学の越智が提案した FPAA(Field Programmable Accumulator Array)[32]（後に FPAccA と改名 [33]），2004 年の慶応大学の長谷川，山田，出口，天野らが行った NEC DRL の後継にあたる，NEC DRP (Dynamically Reconfigurable Processor)[37] の評価 [38]，同年の NTT の片山，甲斐，山田，塩本らの IP–Flex 社の DAPDNA–2[34] の評

価 [35] では，リコンフィギャラブル・アーキテクチャが，特定のアプリケーションにおいて，従来のノイマン型アーキテクチャよりも優れた速度性能を得られることを示している．これらのリコンフィギャラブル・アーキテクチャは FPGA のような LUT アレイ型のアーキテクチャではなく，ALU(Arithmetic and Logical Unit) アレイ型のアーキテクチャであるということが特徴である．また Splash 1/2 と同様に LUT アレイ型のアーキテクチャ (FPGA) を用いたものとしては，1999 年に三菱電機の浅見らの RASH による DES 暗号解析の高速化 [23][24]，同年，名古屋大学の山口らにより同様のアプリケーションでの評価 [25] の結果，ノイマン型アーキテクチャよりも優れた速度性能が得られることが報告されている．このように，近年再びリコンフィギャラブル・アーキテクチャが注目されており，従来の FPGA などの LUT アレイ型のアーキテクチャと，ノイマン型アーキテクチャの中間に位置するアーキテクチャ（ALU アレイ型など）まで製品化され始めている．

2.7 研究課題と対象

現在，リコンフィギャラブル・コンピューティング研究の主流は，ノイマン型アーキテクチャに近い ALU アレイ型アーキテクチャに関するものに移行しつつある．これは，ノイマン型アーキテクチャの MPU の処理性能の向上率が，衰え始めている（ムーア則が崩壊しつつある）ことも影響していると考えられる．従って，今後は従来のノイマン型アーキテクチャの一部に，リコンフィギャラブル・アーキテクチャが取り込まれていくことも予想される．このような方向で進められているリコンフィギャラブル・コンピューティング研究の目的の多くは，現在のノイマン型コンピュータよりも高性能な “汎用コンピュータ” を作ることにある．しかし，本質的にリコンフィギャラブル・アーキテクチャのキラー・アプリケーションが見つかっていないことは，1994 年に Splash 2 が示したことから変わっていない．

先に述べたように，速度，サイズ，消費電力という，アプリケーションの実行性能のみで評価する場合には ASIC 化することで最大性能を得られる．しかし，TAT やコストも含めて評価した場合には再構成可能集積回路や MPU での実現が最良となる場合があり，これはアプリケーションによって決まる．ここで，再構成可能集積回路による実現と MPU による実現を比較したとき，現状では MPU による実現の方が TAT，コストの面で良い上に，アプリケーションによっては実行性能も MPU が上回る場合がある．このため，アプリケーションの性質を問わず，ほとんどの場合において MPU による実装が採用されている．しかし，本来，再構成可能集積回路で実現した方が実装効率の高いアプリケーションが多くあるため，MPU と再構成可能集積回路の実装方式選択の精度を高めることができれば，結果的にアプリケーションの実行性能の向上が期待できる．

本研究の対象は，実用に十分な実行性能（速度，サイズ，消費電力）と実現容易性（TAT，

コスト）を得られる FPGA のアプリケーションであり，その性質を調べることにある．従って，FPGA を用いて実現される特定用途向けシステムの，ボード・レベルでのアーキテクチャ（FPGA とその周辺 LSI の接続構成），FPGA に実装する回路（LUT で実現する回路）の構成，或いは FPGA と回路構成情報の関係などが研究対象である．

第 3 章

確率的ニューラルネットワークを用いた画像認識装置への適用

本章では，アルゴリズム並列性が非常に高く，FPGA のような LUT アレイ型のアーキテクチャに向いていると考えられるアプリケーションの例として，確率的ニューラルネットワークのハードウェア化と，それを用いた画像認識システムについて述べる．本的用例の目的は，現在市販されている一般的な FPGA で MPU が苦手としている処理をどの程度高速化できるかを調べ，ASIC の代替として十分な性能を得られるかを検討することである．本章では，

1. 確率的ニューラルネットワークのハードウェア化,
2. 学習処理の高速化アーキテクチャの提案,
3. 各回路構成,
4. 画像認識システムの構築,
5. 速度性能と認識精度の実装評価,

の順で述べる．

3.1 確率的ニューラルネットワークのハードウェア化

確率的ニューラルネットワーク（PNN: Probabilistic Neural Network）は D.F.Specht によって提案された 3 層構造のフィード・フォワード・ネットワークで，ベイズの識別定理に基づいた，統計的なパターン認識手法の一つである [41]〜[44]．近年その応用事例として，K.Z.Mao らの顔認識問題への適用 [45]， Bin Tian らの衛星からの気象画像の認識問題への適用 [46][47] など，実用的な大規模問題への適用事例が数多く報告されており，その認識精度（認識率）の高さが注目されている．PNN による認識器が高い認識精度を示す理由は， PNN がサンプル

パターンから形成されるカーネル関数を重ね合わせることで，各カテゴリ間の真の確率密度分布の関係を高精度に近似するからである．従って，サンプルパターンの数が増す程，その認識精度はベイズ統計に従う理想的な値に近付き，高い認識精度を持つ認識器が実現できる．

一方 PNN では一つの未知パターンを認識するために，毎回非常に多くのカーネル関数計算を必要とする．このため，画像認識などの大規模パターン認識では多大な計算量が発生する．PNN はカーネル関数の演算に最も計算時間を要するが，この演算は各ニューロン毎に独立して行うことができ，アルゴリズムとしての並列性が非常に高い．このため逐次処理を基本とするノイマン型アーキテクチャの MPU での処理には向かない．一般に画像処理ではビデオレート（33.3 ms）に追従することが一つの基準となるが， PNN によるビデオレートに追従した認識のためには多大な計算量を高速に処理するための専用ハードウェアが必要となる．そこで本論文ではまず，PNN のハードウェア化について述べる．

3.1.1　確率的ニューラルネットワーク

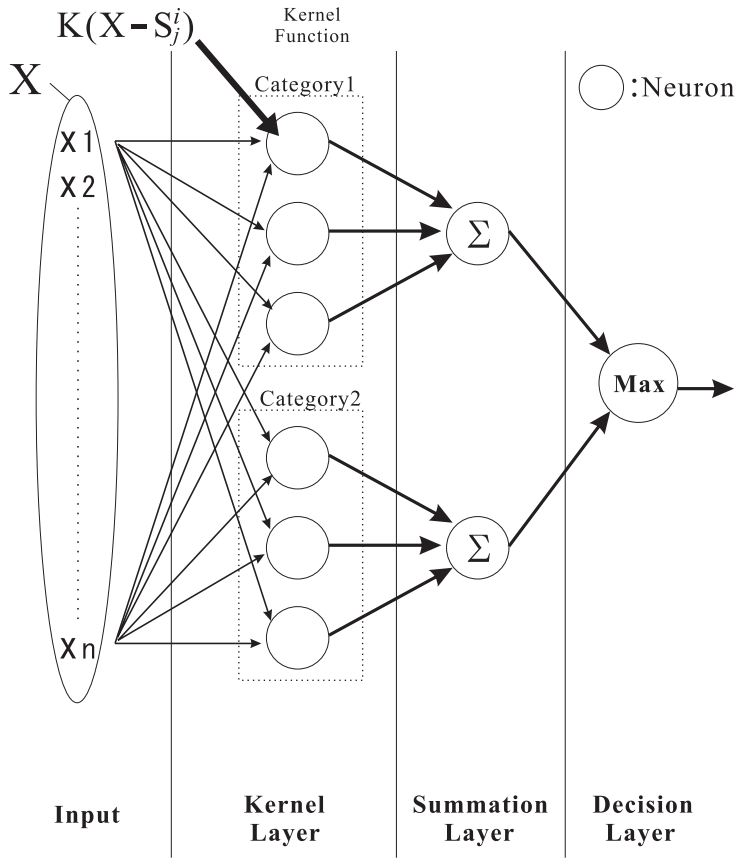


図 3.1　確率的ニューラルネットワークの構成

PNN は図 3.1 に示すように，3 層のフィード・フォワード・ネットワークから成る．PNN によるパターン認識の基本的な考え方は，図 3.1 中のカーネル層（Kernel Layer）のカテゴリ毎のニューロンの出力値を，加算層（Summation Layer）で重ね合わせることで，各カテゴリの確率密度分布を計算し，その大小関係を決定層（Decision Layer）において判定することで未知入力パターンのカテゴリを推定するものである．

未知入力パターン（特徴ベクトル）$\mathbf{X} = (x_1,\ x_2, ...,\ x_n)$ は，カーネル層のカテゴリ毎のニューロン群へ入力され，各ニューロンはサンプルパターン $\mathbf{S}_j^i$（カテゴリ C_i の第 j 番目のサンプルパターン）を基に定義された関数値 $K\left(\mathbf{X} - \mathbf{S}_j^i\right)$ を出力する．この $K\left(\mathbf{X} - \mathbf{S}_j^i\right)$ をカーネル関数とよび，この具体的な関数として

$$K\left(\mathbf{X} - \mathbf{S}_j^i\right) = \exp\left(\frac{-(\mathbf{X} - \mathbf{S}_j^i)^T(\mathbf{X} - \mathbf{S}_j^i)}{2\sigma^2}\right) \tag{3.1}$$

のようなガウス関数や，一様関数

$$K\left(\mathbf{X} - \mathbf{S}_j^i\right) = \begin{cases} 1: & \left|x_k - (s_j^i)_k\right| \le \frac{\sigma}{2} \\ 0: & \text{otherwise} \end{cases} \tag{3.2}$$
$$(\text{for all } k,\ k = 1,\ 2, ...,\ n)$$

が用いられる．ただしここで，

$$\mathbf{X} = (x_1,\ x_2, ...,\ x_n),$$
$$\mathbf{S}_j^i = \left((s_j^i)_1,\ (s_j^i)_2, ...,\ (s_j^i)_n\right)$$

で，x_k，$(s_j^i)_k$ はそれぞれ第 k 番目の成分を表し，n はパターンの次元数を表す．また，式 (3.1)，式 (3.2) 中の σ はカーネル関数の広がりを表すパラメータである．このパラメータを高精度に決定することが，PNN の認識精度を決定する上で重要となる．

ここでカーネル関数としてガウシアン関数（式 (3.1)）を用いると浮動小数点演算などが必要となり，ハードウェア化すると回路が複雑となる．一方，一様関数は後述するように数個の加減算回路やフリップ・フロップなどでハードウェア化できるため，回路規模も小さく，しかも演算の高速化が実現できる．従って我々はカーネル関数として一様関数を用いることにした．

なお，カーネル関数として一様関数を用いることで十分な認識精度を得られることは，すでに Minchin らによるシミュレーション実験によっても示されている [48]．その報告によれば，パターンの次元数を n，カテゴリ数を N_c としたとき，$n = 9$，$N_c = 4$ の石の画像抽出を対象とした認識問題，$n = 48$，$N_c = 5$ の形状の異なる鐘の音響周波数スペクトルを対象とした認識問題などで，カーネル関数にガウシアン関数を用いた場合と，一様関数を用いた場合で，認識精度が 0.7%〜2.8% 程度の違いしかないことが示されている．

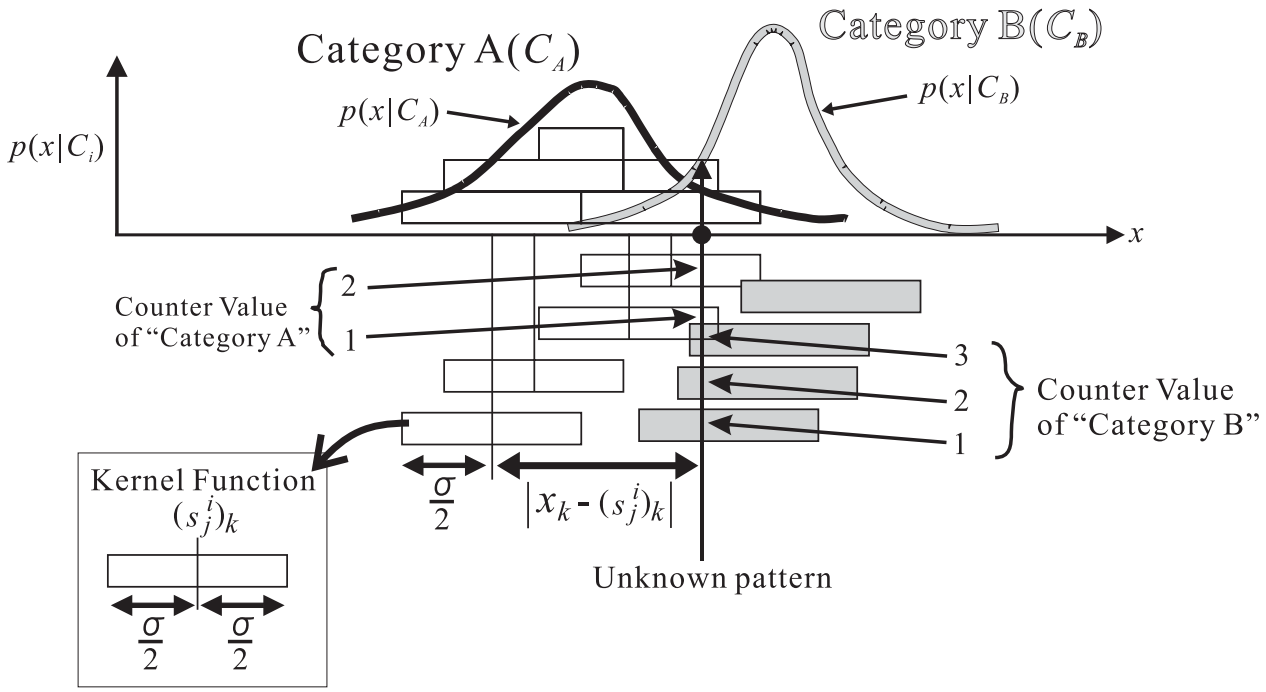


図 3.2　一様関数を用いた確率密度推定

次に加算層（Summation Layer）において，このカーネル関数を各カテゴリ毎に重ね合わせることで，未知パターン $\mathbf{X}$ がカテゴリ C_i に属する確率密度分布 $p(\mathbf{X}|C_i)$ を推定する．即ち，

$$\hat{p}(\mathbf{X}|C_i) = \frac{1}{N_p} \sum_{j=0}^{N_p} K\left(\mathbf{X} - \mathbf{S}_j^i\right) \tag{3.3}$$

と表せる．なお，通常 N_p は各カテゴリで同じとするため， $1/N_p$ の項は 1 と見なして処理することができる．

最後に決定層（Decision Layer）で最も大きな確率値を得たカテゴリを求めることで，未知パターンのカテゴリが推定できる．

図 3.2 に一様関数を用いた場合の確率密度の推定方法を示す．図は 1 次元のパターンにおける確率密度分布を表しており，横軸が特徴量 x を，縦軸が確率 $p(x|C_i)$ を表している．また識別するカテゴリは C_A，C_B の 2 つであり，各カテゴリにおける事前確率は一定としてある．いま未知のパターン（Unknown Pattern）x_k が入力されると，x_k から各サンプルパターンまでの距離 $\left|x_k - \left(s_j^i\right)_k\right|$ に対し，式 (3.2) を適用する．ここで，各カテゴリにおける確率 $p(x_k|C_i)$ の大小関係は，各サンプルから σ の広がりを持つ領域（図中の各ブロック）にそれぞれいくつ入るかを数え，その大小関係を比較することに等しい．この領域内に入るかどうかをカーネル層で求め，その結果を加算層にて各カテゴリ毎に数えることで，決定層にて未知のパターンが所

属している確率が最も高いカテゴリが得られる. 図の例では，カテゴリ C_A に属するサンプルのブロック内に 2 つ，カテゴリ C_B に属するサンプルのブロック内に 3 つ入っているので，この未知パターン x_k はカテゴリ C_B に属するものと推定される.

3.1.2 高速化のための専用プロセッサ " sPNN (serial-PNN) Processor "

PNN の計算において最も時間がかかるのはカーネル層のニューロンの計算，即ち，カーネル関数の計算である．そこで我々は各カーネル関数を高速に計算するための専用プロセッサ（serial-PNN Processor）を提案する．これまでのニューラルネットワークハードウェアではシグモイド関数，ガウシアン関数といった主要な関数を Look-up-table やマイクロプログラムと演算器の組合せによって実現してきた [53]. 一方，我々の提案する sPNN Processor の特徴は，カーネル関数を単純な組合せ回路で直接実現した点にある．これにより後述するようにビデオレート（33.3ms）以内での高速画像認識を実現することができる．なお， sPNN Processor の回路の詳細（カーネル関数の具体的な回路化方法）については 4 章で述べる.

sPNN Processor は 1 つのサンプルパターン S_j^i に対する PNN の 3 層の計算（式 (3.2)，式 (3.3)）を逐次処理によって行うプロセッサである. 単一のプロセッサでも前述したように，その構造の単純さから高速な演算が可能であるが，さらにこのプロセッサを多数並列実装することで，異なるサンプルパターンに対して並列に PNN の計算を実行でき，さらなる認識処理の高速化が図れる. ここでどのカテゴリのどのサンプルに対して演算を行うかは，サンプルパターンを記憶する外部メモリのアドレスとプロセッサ内部のレジスタを管理することにより，認識問題に応じて柔軟に対応できる. このため，同容量の外部メモリを用いても，総サンプル数 $N_p \times N_c$ が等しければサンプル数が多く必要な場合でも，カテゴリ数が多く必要となる場合でも対応が可能である．ただ，多くの認識問題において，高い認識精度を得るためには，カテゴリ数の増加に伴い，各カテゴリに属するサンプル数も必要になるため，この場合には外部メモリを増設する必要がある. このとき，認識時間も一定に保ちたい場合にはメモリと同時にプロセッサも増設する必要がある. しかし，提案する sPNN Processor の接続構成はバスアーキテクチャであるため，容易に増設が可能であり，またそのスケーラビリティも非常に高い（後述）.

3.2 高速学習のための並列アーキテクチャ

認識精度を高めるためには，多くのサンプルパターンが必要であるだけでなく，PNN のパラメータ σ を適切に決定する必要がある．この σ を最適化することが PNN の学習である.

PNN はサンプルパターンが多いため， 1 回のネットワーク計算（認識処理）だけでも多大な計算量を要する．さらに，前述した顔画像認識や気象画像認識といった実用的な大規模問題

では，真の確率密度分布は全く未知であり，このため σ を決定論的に求めることはできない．従って σ をその全空間に渡って探索し，適切な値を得る必要がある．このため σ の学習は，その値を少しずつ変えながら，ネットワーク計算を繰り返す必要があり，その計算時間は認識処理時間とは比べものにならない程多大なものになる．通常，学習はサンプルパターンの追加や変更のたびに行う必要があり，大規模な画像認識システムでは頻繁にこの追加／変更が行われる為，学習時間の高速化は特に重要である．

私の提案する高速学習のための並列アーキテクチャを Sigma-Parallel Architecture（SPA）と呼ぶ．SPA は，複数の PNN のネットワークにそれぞれ異なる認識パラメータ σ を与え，それを並列に計算することで学習の高速化を図るものである．即ち，異なる複数の σ に対して前述の sPNN Processor を多数並列実装する．さらに複数の sPNN Processor をモジュール化し，これらを並列実行させることで更なる高速化を実現する（これを Hybrid-SPA と呼ぶ）．以下，これらの並列アーキテクチャについて述べる．

3.2.1 Sigma-Parallel Architecture (SPA)

前述したように，図 3.1 に示すカーネル層にある各ニューロンは，それぞれカテゴリ C_i に属する第 j 番目のサンプルパターン $\mathbf{S}_j^i$ を基に構成される．サンプルパターン $\mathbf{S}_j^i$ のサイズは，この次元数を n，各要素のビット精度を q bit としたとき， qn bit となる．このサイズのサンプルパターンが 1 カテゴリにつき N_p 個用いられ，それがさらに N_c カテゴリ分あるとすると，全てのサンプルパターンを記憶しておくために必要なメモリの容量は，qnN_pN_c bit となる．ここで，例えば実用的な例として，$q = 8$, $n = 256$, $N_p = 512$, $N_c = 8$ とすると必要なメモリ容量は 1MByte となり，さらにカテゴリ数 N_c，1 カテゴリに含まれるサンプル数 N_p などが増えれば，必要なメモリ容量は数 MByte〜数百 MByte と，大幅に増加する．1 つの集積回路内にこれだけの容量のメモリを演算回路（ランダムロジック）と共に実装することは困難である．このため，これらのサンプルパターンは演算回路とは別に，外部の専用メモリに記憶する必要がある．

従って，図 3.3(a) に示すように，従来よりバックプロパゲーションの並列化などで用いられてきた「ネットワーク上の各ニューロンを専用のプロセッサにハードウェア化し，集積回路内で並列化する手法（これをニューロン並列と呼ぶ）」を PNN にそのまま適用するためには外部メモリが不可欠となる．この場合，カーネル層のニューロンを計算するプロセッサ（図中の Neuron in Kernel Layer）は，それぞれ異なるサンプルパターンが同時に必要となるため，外部メモリのデータバスもそれぞれ独立に必要となる．未知入力パターン $\mathbf{X}$ は各ニューロンに集積回路内でブロードキャストされる．認識パラメータ σ も $\mathbf{X}$ と同様に，集積回路内でブロードキャストされ，同じ値が各ニューロンの演算で使われる．学習ではこの共通な σ を σ_m として

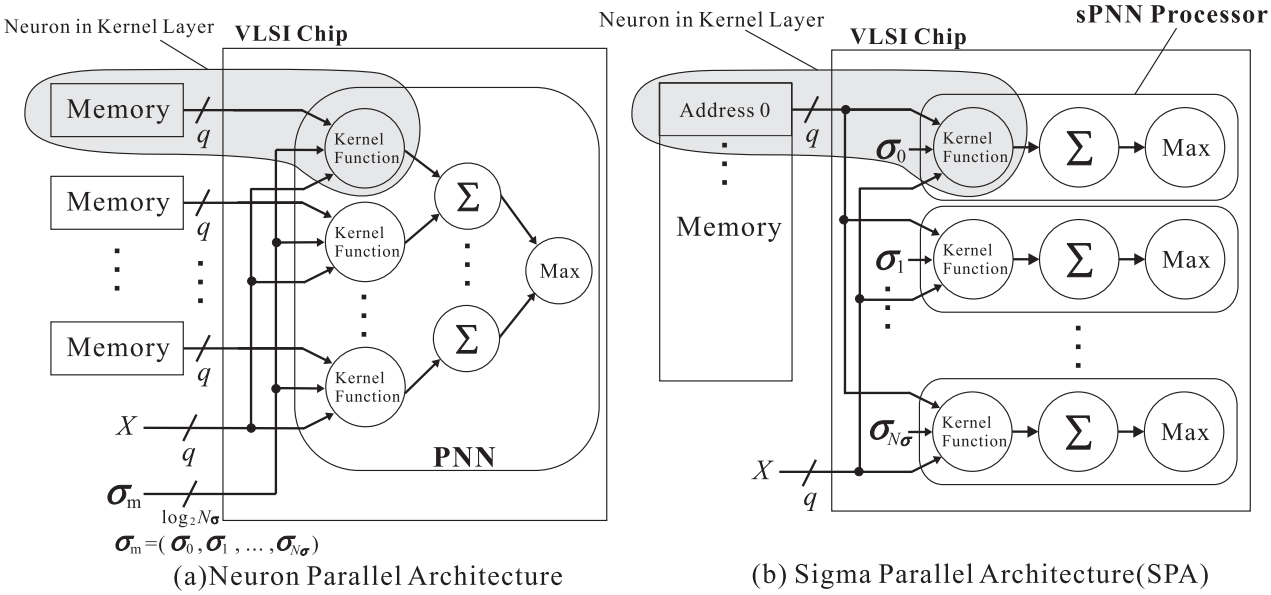


図 3.3　従来手法（ニューロン並列）と SPA の比較

$m = 0$ から $m = N_\sigma$ まで（N_σ は探索する σ の総数）少しずつ変えて，繰り返し演算を行うことでニューロンの並列性を利用した高速な学習が可能となると考えられる．

しかし，このニューロン並列のアーキテクチャには大きな問題がある．図 3.3(a) では，qN_pN_c 本のサンプルパターン用の入力ピン数（メモリのデータバス幅）と，q 本の未知入力ピン数，合計 $q\,(N_pN_c + 1)$ 本の入力ピン数を必要とする．例えば先の例，$q = 8$, $N_p = 512$, $N_c = 8$ のとき，必要なピン数は $q\,(N_pN_c + 1) = 32,776$ 本と，非現実的な要求ピン数となる．

そこで，「異なる σ をニューロンに与え，各ニューロンの下で並列にテストパターンに対して認識精度を計算する」手法を提案する（図 3.3(b)）．これを SPA（Sigma-Parallel Architecture）と呼ぶ．SPA では前述した sPNN Processor が異なる σ に対してそれぞれ並列に動作する．sPNN Processor は次節でその詳細を述べるように，時分割処理で PNN を計算する．実装上，ニューロンは各層とも 1 つのみで構成されており，これを繰り返し実行することで 1 つのネットワーク（PNN）全体を計算する．この sPNN Processor を 1 つの集積回路内に多数，並列に実装し，それぞれに異なる σ を与えることで，並列化をニューロン並列方式（Neuron Parallel Arcitecture）ではなく，σ 並列方式（Sigma Parallel Architecture）とすることができる．σ はサンプルパターンのようにデータサイズが大きくないので（$\log_2 N_\sigma$ は数〜数十ビット），sPNN Processor と同じ 1 つの集積回路内に容易に実装することができる．

各 sPNN Processor は逐次処理をするため，外部メモリや未知パターンの入力に必要となる物理的なピン数も少なくなる．

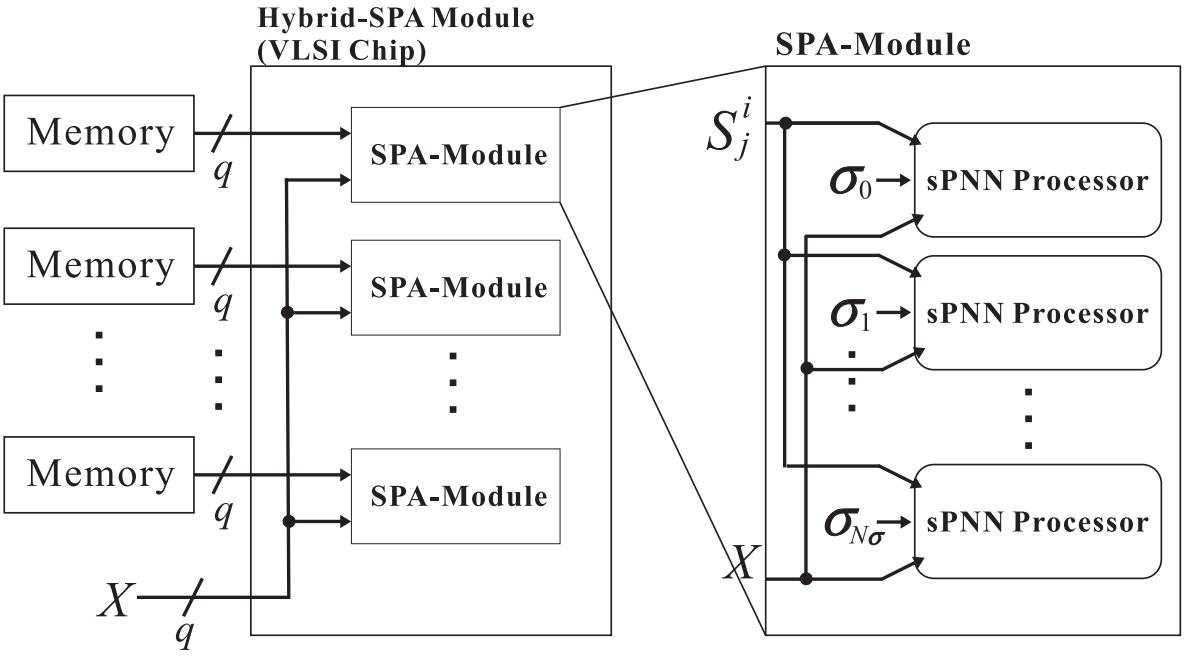


図 3.4 SPA の多重並列化

3.2.2 Hybrid-SPA

図 3.3(b) の SPA は従来のニューロン並列構成と排他的に選択するものではない．SPA で構成した複数の sPNN Processor をモジュール化し（SPA-Module），これを与えられたピン数制限の範囲で図 3.4 のように多重に並列に実装することで，さらなる高速化が期待できる．このような，σ 並列を基本としてさらにこれらをニューロン並列化する構成を Hybrid-SPA と呼ぶこととする．現在この Hybrid-SPA に基づく小型画像認識システムを開発中であり，本論文では後半でその試作評価結果を述べる．

3.3 回路構成

本節では PNN を実際に SPA に基づいてハードウェア化する際の回路構成について，まず sPNN Processor の回路構成について述べ，次に Hybrid-SPA の全体回路構成について述べる．

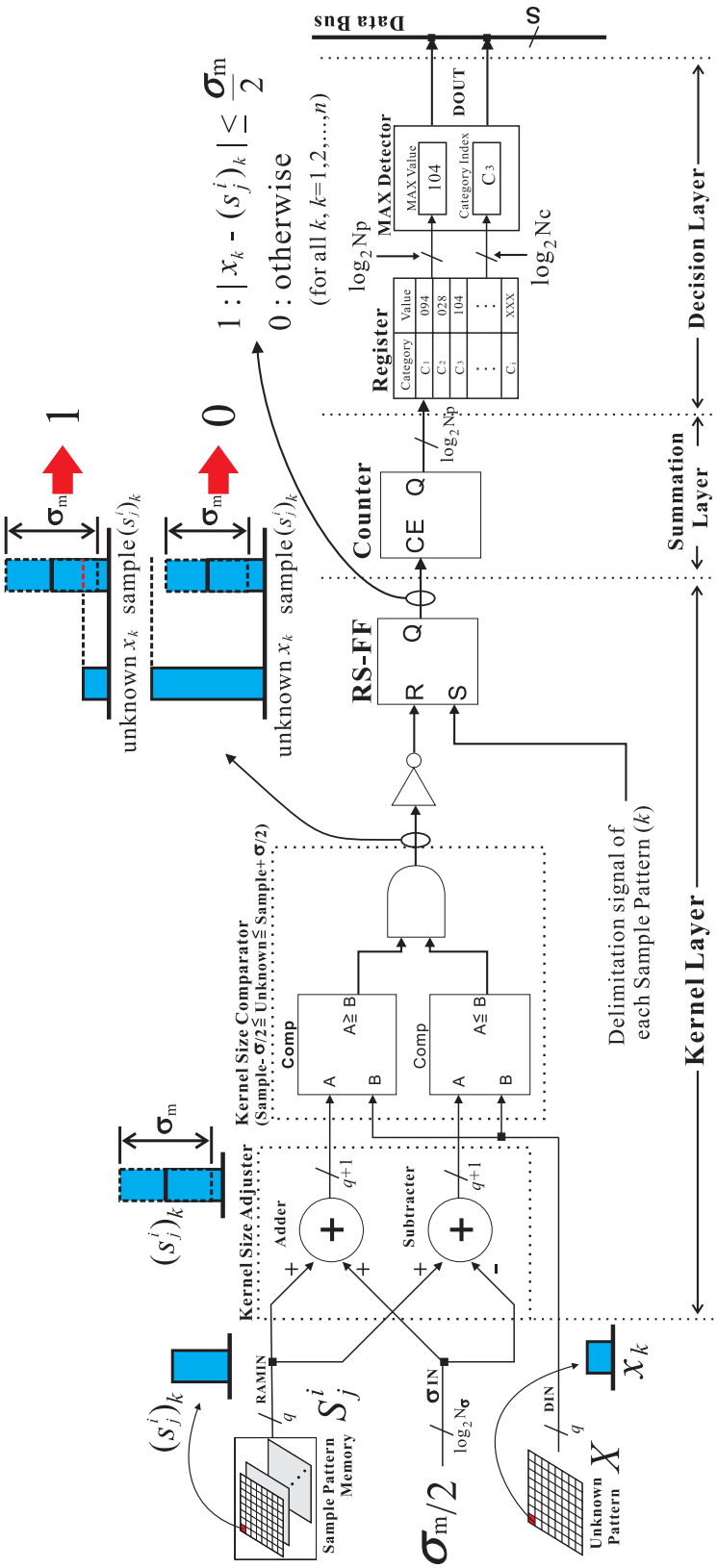


図 3.5 sPNN Processor の回路構成

3.3.1 sPNN Processor の回路構成

sPNN Processor の回路構成を図 3.5 に示す．この回路は入力として未知パターン $\mathbf{X}$，サンプルパターン $\mathbf{S}_j^i$，認識パラメータ σ_m を持ち，PNN の演算を時分割処理で行う．図 3.1 の PNN に対応して 3 層から成るが，物理的に存在する各層のニューロンは 1 つずつで，これらを用いて逐次処理することで PNN を構築し，認識処理を行う．未知パターン $\mathbf{X}$ と予めカテゴリ毎に外部メモリに記録されているサンプルパターン $\mathbf{S}_j^i$ は，それぞれ q bit の各要素 x_1，x_2，..., x_n 及び，$\left((s_j^i)_1, (s_j^i)_2, ..., (s_j^i)_n\right)$ 毎に逐次処理される（即ち，式 (3.2) の演算を各要素毎に行う）．σ_m は第 m 番目の σ で，SPA では各ニューロン毎に異なった値をとる．

対象とするカーネル関数として，前述した一様関数（式 (3.2)）を選択した．これは以下に示すように少ない回路規模で関数を直接回路化することができ，高速かつ小型化が可能であるからである．

3.3.2 カーネル層と加算層の演算回路

外部メモリから読み込まれたカテゴリ C_i の第 j 番目のサンプルパターンの第 1 番目の要素 $\left(s_j^i\right)_1$ は図 3.5 中の “Kernel Size Adjuster” にて $\sigma_m/2$ の加算と減算を行い，各演算結果を出力する．次に “Kernel Size Comparator” にて，この各値，$\left(s_j^i\right)_1 + \sigma_m/2$ 及び，$\left(s_j^i\right)_1 - \sigma_m/2$ と未知入力パターン $\mathbf{X}$ の第 1 番目の要素 x_1 との値の大小関係を比較する（これが式 (3.2) に示す一様関数の $\left|x_k - \left(s_j^i\right)_k\right| \leq \sigma/2$ の比較に対応する）．“Kernel Size Comparator” は，$\left(\left(s_j^i\right)_1 - \sigma_m/2\right) \leq x_1 \leq \left(\left(s_j^i\right)_1 + \sigma_m/2\right)$ であった場合に 1 を出力し，x_1 がこの範囲にないときは 0 を出力する．

この演算をサンプルパターン $\mathbf{S}_j^i$ の全ての要素に対して行う．この出力を後段のフリップ・フロップで監視し，全ての要素において演算結果が 1 となったとき，つまり未知入力がカーネルに入った場合（式 (3.2) の右辺 1）の数をさらに後段の加算層（Summation Layer）にあるカウンタでカウントする．

3.3.3 決定層の演算回路

加算層の出力値は決定層（Decision Layer）でカテゴリ毎にレジスタ（Register）にスタックされる．次に最大値検出器（Max Detector）にてそのレジスタにスタックされた値のうち，最も大きな値と，そのカテゴリが検出される．

図 3.6 に最大値検出器の回路構成を示す．図 3.5 中の Register にスタックされた各カテゴリ毎の加算結果値と，そのカテゴリを表すインデックス番号はそれぞれ図 3.6 の最大値検出器の

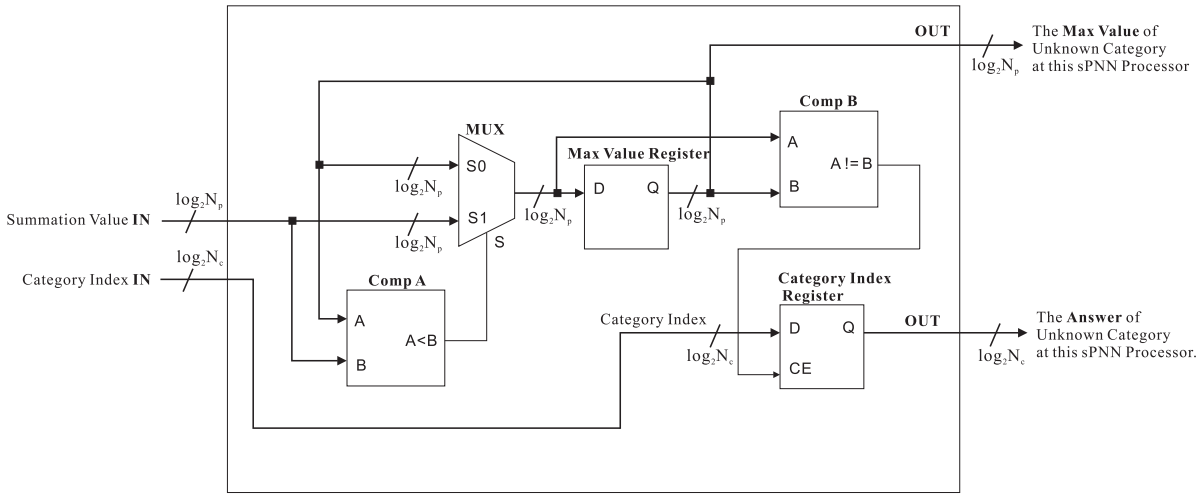


図 3.6 最大値検出回路 (Max Detector)

“Summation Value IN” 及び，“Category Index IN” に入力される．初期状態では，図中の “Max Value Register” の出力値は 0 であるため，“Comp A” の比較結果値（$A < B$ の出力値）は必ず 1 となる. このため次のクロック・サイクルで “Max Value Register” には “Summation Value IN” の入力値がスタックされる. このとき同時に “Comp B” の出力が 1 となり，このとき入力されている “Category Index IN” のインデックス番号が “Category Index Register” にスタックされる. 次のサイクルで 2 番目のカテゴリの加算結果値に対し，まず “Comp A” で初めのカテゴリの加算結果値（先程 “Max Value Register” にスタックされた値）と比較を行い，“2 番目の加算結果値 $>$ 1 番目の加算結果値” であれば 2 番目の値が “Max Value Register” にスタックされ，そうでなければ Register は更新されない. また “2 番目の加算結果値 $>$ 1 番目の加算結果値” であった場合には “Max Value Register” の更新と同時に，“Category Index Register” も 2 番目のカテゴリのインデックス番号に書き換えられる．この処理を全てのカテゴリについて行うことで，最終的に “Max Value Register”， “Category Index Register” にそれぞれ加算結果の最大値と，その最大値となったカテゴリのインデックス番号がスタックされる.

ここで検出されたカテゴリがこの sPNN Processor における認識結果である．sPNN Processor が 1 つの場合にはこの結果が未知パターンのカテゴリを示すが，複数の sPNN Processor で処理を分担している場合には，各プロセッサの演算結果の中からさらに最大の値となったカテゴリを検出する必要がある. その場合，各 sPNN Processor での認識結果はさらにデータバスを経由して，Best σ Detector 内にある最終段の最大値検出器に送られる.

表 3.1 sPNN Processor の回路規模

FPGA	Part	Number of CLBs or SLICEs
XCS30XL	sPNN Processor	120 CLBs(20.8%)
	Control Circuits	160 CLBs(27.8%)
	Whole of Circuits	280 CLBs(48.6%)
XCV800	sPNN Processor	120 SLICEs(1.3%)
	Control Circuits	170 SLICEs(1.8%)
	Whole of Circuits	290 SLICEs(3.0%)

3.3.4 カーネル層最終段の判定回路の小型化

ここで，未知パターンがカーネルに入ったかどうかを判断する回路（図中，Kernel Layer の出力にある RS-FF 部分）はフリップ・フロップで監視する他に，カウンタを用いて各要素毎に “Kernel Size Comparator” の出力に応じてカウントアップし，全ての要素について比較した後，カウンタの値が次元数 n に一致しているかを見ることでも実現できる．しかし，全ての次元 (セグメント) において ‘1’ であるかということは，逆にとると１つの次元 (セグメント) も ‘0’ ではないということであり，これは予め ‘1’ をセットしたフリップ・フロップのリセット入力に “Kernel Size Comparator” の出力のインバートしたものを入れ，全ての次元 (セグメント) の比較を行った後にこのフリップ・フロップの状態を見ることで実現できる. このときフリップ・フロップの出力が ‘0’ ならいずれかの次元 (セグメント) で範囲に入らなかったということで，‘1’ なら全ての次元 (セグメント) の範囲に入った，つまりカーネル内に入ったということになる．このように本回路構成ではカウンタの変わりにフリップ・フロップ１つで済むため，回路規模を抑えることができた.

実際に，1 個の sPNN Processor を XILINX 製の FPGA である, XCS30XL–4PQ240C[67], XCV800–6HQ240C[65] にそれぞれインプリメントした結果を表 3.1 に示す. その結果，XCV800 で sPNN Processor は 120 SLICE 要し，XCV800 は 9,408 SLICE を有する [65] ので，この規模の FPGA でも 64 個以上の sPNN Processor が他の制御回路などを含め実装できると見積もれた.

sPNN Processor はこのようにカーネル関数を簡単な組合せ回路で直接実現しているため，回路規模も小さく，かつ，逐次処理でありながらも結果として後述するような高速処理を実現している.

3.4 Hybrid-SPA による全体回路構成

ここではまず，Hybrid-SPA に基づいたシステムの全体の接続構成について述べ，次に本アーキテクチャのスケーラビリティについて，実際に開発でターゲットとしている範囲から，理論的な限界までのシミュレーション結果を基に述べる．

3.4.1 接続構成

図 3.7 に，Hybrid-SPA に基づいた PNN ハードウェアの全体回路構成を示す．なおこの構成は，後述する SusuPRB–M02 に採用した具体例を用いて説明する．全体は 2 つの FPGA（XILINX 製 XCV1000E-6HQ240C[66] 及び，XCV300E-6PQ240C[66]）と，サンプルパターンを記憶しておくメモリ LSI 群（日立製 4M-bit SRAM HM62W8511HJP）から構成されている．中心となる FPGA（XCV1000E）には，図 3.4 の Hybrid-SPA に基づいて構成された 64 個の sPNN Processor が実装される．この 64 個の sPNN Processor は 4 個の SPA-Module から成り，さらに 1 つの SPA-Module は 16 個の sPNN Processor から構成される（即ち 16 個の σ を並列処理する）．

これら 64 個の sPNN Processor の出力（各 Processor の認識結果）はバスを介してもう 1 つの FPGA（XCV300E）に入力される．この FPGA には最も高い精度を得る σ を求める “Best σ Detector” が実装されており， これを学習時に使用する．Best σ Detector は認識処理においても，各 sPNN Processor の認識結果の中からさらに最終的な認識結果を得るための最大値検出器として働く．即ち，複数の sPNN Processor の認識結果は， s bit のデータバスに結合された Best σ Detector へ逐次転送され，ここでさらに最大値を得たカテゴリが検出される．ここで得られたカテゴリが最終的な認識結果である．

ここで， sPNN Processor と Best σ Detector はバスで接続されており，このため sPNN Processor を容易にこのバス上に増設できる．従って，アプリケーション毎に要求数が異なるカテゴリ数やサンプル数，或は要求される学習時間に応じて， sPNN Processor を増設することができる．なお，Best σ Detector の詳細については次節で述べる．

3.4.2 スケーラビリティ

ここで，学習時間 T_{learn} は，

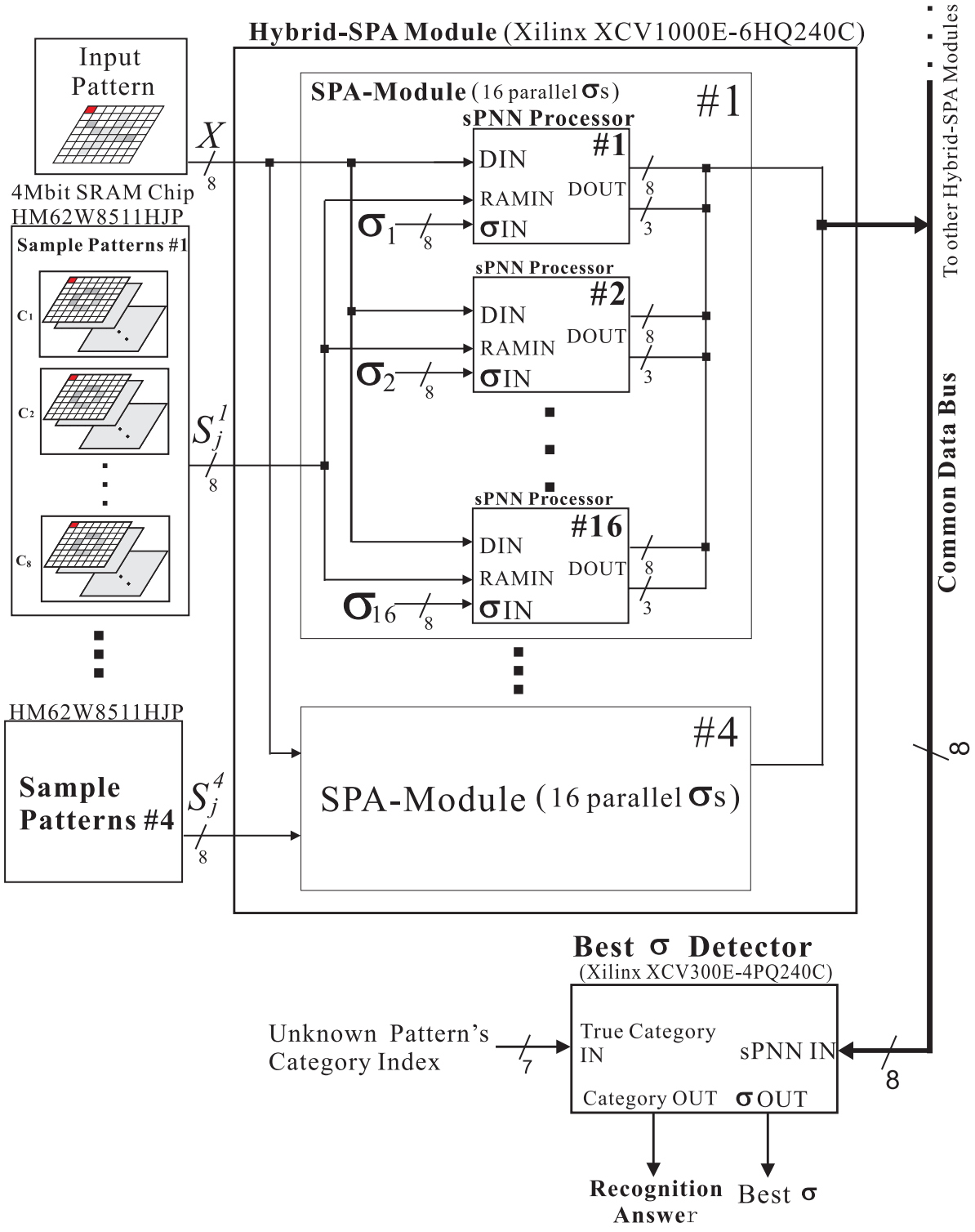


図 3.7　Hybrid-SPA に基づくシステムの構成

$$T_{learn} = \frac{\left(T_{recog.} + \beta N_{sPNN} N_{Mod.} T_{CLK}\right) N_{tp} N_{\sigma}}{N_{sPNN}} \tag{3.4}$$

と算出できる．ただしここで，

n： パターンの次元数，
q： 各要素のビット精度，
r： メモリと sPNN Processor 間の
ビット幅，
s： 共有バスのビット幅，
N_c： カテゴリ数，
N_p： 1 カテゴリのサンプルパターン数，
N_{tp}： テストパターン数，
N_{σ}： 探索する σ の総数，
N_{sPNN}： 1SPA-Module 中の
sPNN Processor 数，
$N_{Mod.}$： SPA-Module 数，
α： sPNN Processor 内の
決定層で生じるオーバーヘッド，
γ： Best σ Detector 内の
決定層で生じるオーバーヘッド，
T_{CLK}： 基本動作周期

である．また $T_{recog.}$ は認識時間を表し，

$$T_{recog.} = \left(1 + \frac{qnN_pN_c}{rN_{Mod.}} + \alpha N_c + \beta N_{Mod.}\right) T_{CLK} \tag{3.5}$$

である．また，

$$\beta = \frac{\log_2 N_p + \log_2 N_c}{s} + \gamma \tag{3.6}$$

で，これは図 3.7 に示した各 sPNN Processor と Best σ Detector を結ぶバスで，データの逐次

転送を行うことによって生じるオーバーヘッド・サイクル数である. sPNN Processor の出力に接続されているデータバスには他の sPNN Processor も接続されるため，その接続数 N_{sPNN} が増えれば，それだけ全ての sPNN Processor の演算結果を “Best σ Detector” へ転送するまでの時間 β が増加する. しかしこのオーバーヘッドは認識処理のサイクル数 $(T_{recog.}/T_{CLK})$ に比べると，はるかに小さいため高いスケーラビリティが得られる.

認識時間 $T_{recog.}$ の第 1 項目は，パイプライン処理の始めのパターンデータの読み込みである 1 サイクルを表し，第 2 項目の分子 qnN_pN_c は基本動作周期で処理できるデータの総数を表し，分母の $rN_{Mod.}$ はメモリの総バス幅を表している. 学習時間の算出式（式 (3.4)）の β の項は，全ての sPNN Processor の出力を Best σ Detector に逐次転送して最大値検出を行うためのオーバーヘッド・サイクル数を表している.

学習処理に要する時間は sPNN Processor 数の増加に伴い短くなる（ただし $N_{sPNN} \leq N_\sigma$）. 実用となるパターン認識では処理するパターン数が sPNN Processor 数に比べ十分に多く（$N_p \times N_c \times N_tp \times N_\sigma \gg N_{sPNN}$），また，提案した SPA は，プロセッサの増加による処理のオーバーヘッドが全体の処理に比べはるかに小さいため，非常に高いスケーラビリティを得られる.

ここで sPNN Processor 数 N_{sPNN} に対する高速化率は 1 つの sPNN Processor で処理する場合の学習時間を基準として $T_{learn}(N_{sPNN}=1)/T_{learn}(N_{sPNN})$ より求まる. これから求めたスケーラビリティを表すグラフを図 3.8，図 3.9 に示す. このグラフから N_{sPNN} =1〜256（図 3.8）では非常に高いスケーラビリティが得られることがわかる. さらに高速化率が落ち始めるのは $n=256$， $N_p=32$， $N_c=32$ のとき，及び $n=32$，$N_p=512$， $N_c=32$ など次元数，サンプル数，カテゴリ数のいづれか 2 つ以上が少ない場合の N_{sPNN} =10,000 以上からであり，実際に開発のターゲットとする範囲（N_{sPNN} =1〜256）ではほぼ線形な高速化率が得られることがわかる.

3.5 Best σ Detector の回路構成

図 3.11 に Best σ Detector の回路構成を示す. Best σ Detector は学習処理に限らず，認識処理においても使用する最終段の “Max Detector” と，学習時にのみ動作する認識精度計算回路の，大きく 2 つのブロックから構成されている. また，学習において， sPNN Processor 数 N_{sPNN} が探索する σ の総数 N_σ より少ない場合には，残りの σ の探索を N_{sPNN} の倍数回分，逐次処理によって行う必要がある. この場合には各 sPNN Processor の持つ σ の値を更新するためにカウンタ（Binary Counter）が用いられる.

認識処理では（最終段）最大値検出器（Max Detector）の検出したカテゴリ・インデックス番号が認識結果である. 学習処理では未知入力として，実際には “既知” のテストパターンを用

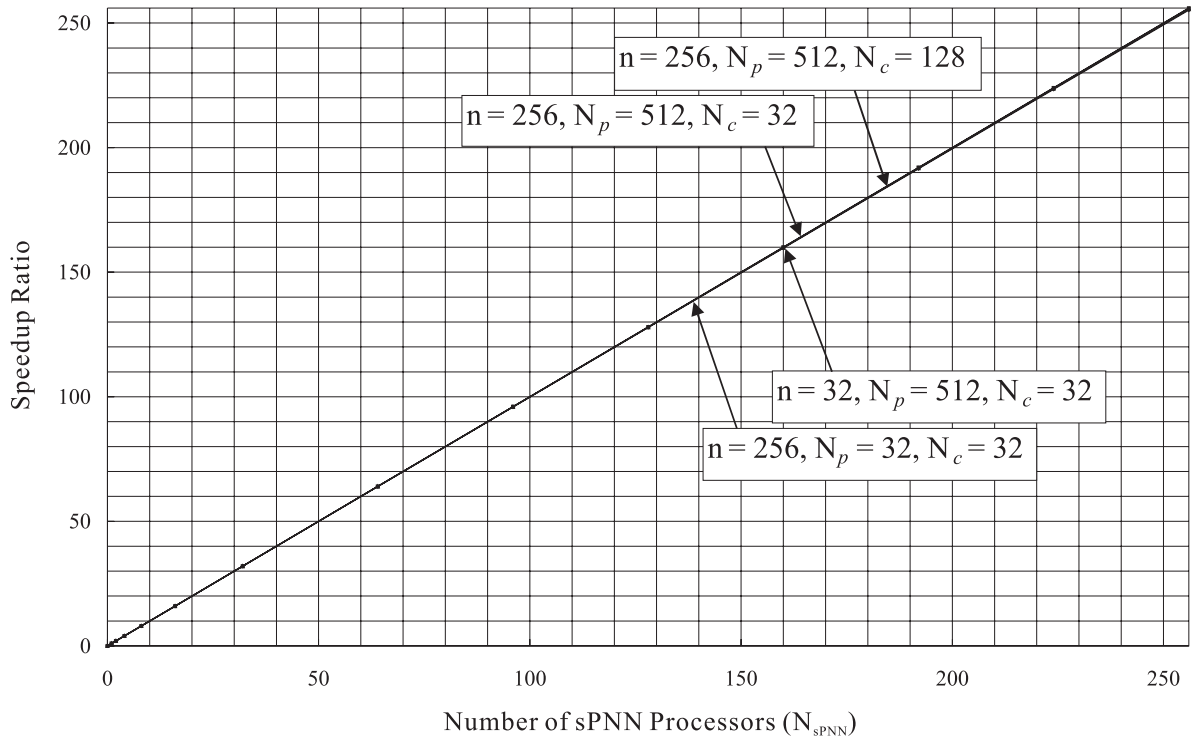


図 3.8　学習処理に関するスケーラビリティ（N_{sPNN} =1〜256）

いてこの認識結果が正しいかどうかを判定し，入力するテストパターンを変更してその正解数をカウントする. これにより，そのとき使われた σ に対する認識精度（認識率）が算出される. さらにこの中から最も高い認識精度を得られる σ を求める. こうして求まった σ の最適値を認識処理で用いる.

全ての σ に対する探索を並列に行える場合（$N_{sPNN} = N_\sigma$ のとき），図中のマルチプレクサ（MUX）は “S1” 側に接続されており，認識処理でこの σ が用いられる. 一方，$N_{sPNN} < N_\sigma$ のときは σ を変更して，逐次処理を行うため，MUX は “S0” 側に接続され，SPA-Module へカウンタ（Binary Counter）の値が出力される. SPA-Module 内では，このカウンタの値が各 sPNN Processor へブロードキャストされ，それぞれが持つ σ の値に加算されて認識処理が行われる. このときカウンタの出力ビット数を z bits とすると SPA-Module を実装する集積回路にはこの z 本分余計にピン数が必要となるが，このピン数のオーダーは数本〜十数本程度であるためそれほど問題とはならない. z は，$z = \log_2 (N_\sigma / N_{sPNN})$ より求まる. ただし，N_σ / N_{sPNN} が割り切れない場合は余りを繰り上げるものとする. 従って，例えば $N_{sPNN} = 16$，$z = 4$ であれば，$N_\sigma = 256$ まで探索できる.

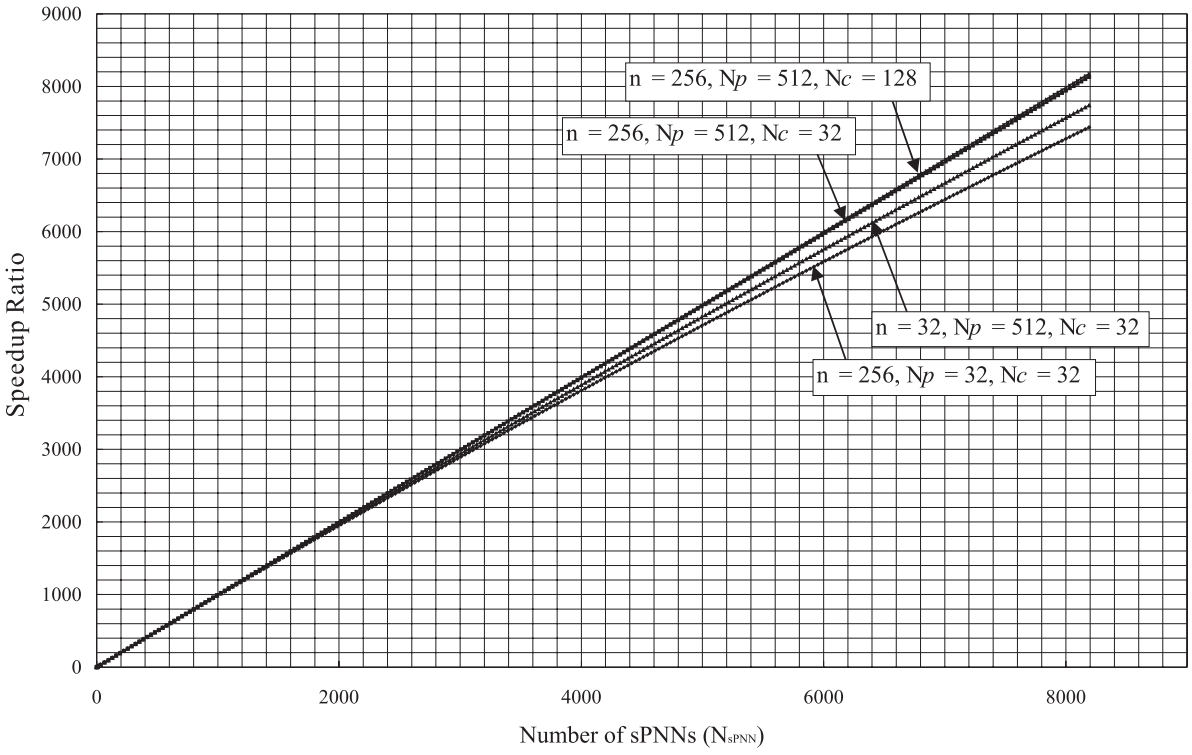


図 3.9　学習処理に関するスケーラビリティ（N_{sPNN} =1〜8，192）

3.5.1　最終段最大値検出器の回路構成

Best σ Detector はまず，各 sPNN Processor の認識結果から，さらに最大値検出を行い，最終的な認識結果を求める．図 3.12 に Best σ Detector 内の最大値検出器の回路構成を示す．基本的な動作は sPNN Processor 内の最大値検出器（図 3.5 の Max Detector）と同じだが，各 sPNN Processor の出力を検出した最大値と，カテゴリのインデックスを同一のデータバス経由で受けるため，入力段にこれらを分離する Shift Register を設けている．また，この Shift Register はデータバスの幅 s bits と各データのビット数 $\log_2 N_p$ bits，及び $\log_2 N_c$ bits とのシリアル-パラレル変換（逐次転送処理）を行う役割も担う．その後の最大値検出の回路動作は，sPNN Processor 中の “Max Detector”（図 3.6）と同様である．

3.5.2　認識精度計算回路

図 3.13 に最大認識精度計算回路の構成を示す．この回路はテストパターンを用いて，各 σ に対する認識精度を算出すると共に，その中から最も高い認識精度を達成した σ を求めるも

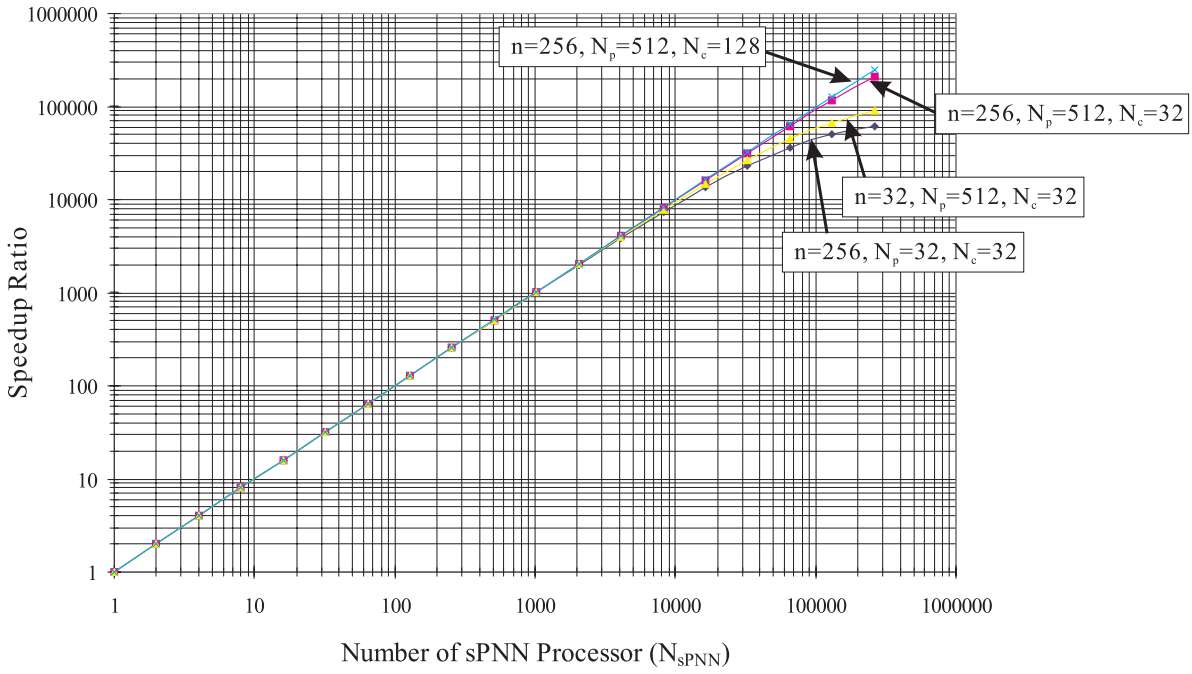


図 3.10 学習処理に関するスケーラビリティ（〜低下領域）

のである．認識精度の算出は入力にある一致検出器（Comp）とその後のカウンタ（Binary Counter）によって行う．Comp の入力 A には認識器が認識した結果（テストパターンのカテゴリのインデックス番号）が入力され，入力 B にはテストパターンの真のカテゴリのインデックス番号が入力される．従って， Comp は認識器の認識結果が正しかったときにのみ 1 を出力する．テストパターンを変更しながらこの 1 の数をカウントすることで，そのときの σ に対する正解数が求まる．ここで，認識精度 $P_{correct}$ は，

$$P_{correct} = \frac{N_{correct}}{N_{tp}} \times 100(\%) \tag{3.7}$$

である．ただしここで，

$N_{correct}$: Number of correct answers
N_{tp} : Number of test patterns

であり， “σ” の最適値はこの $P_{correct}$ を最大にする “σ” である．

式 (3.7) において $P_{correct}$ の最大値を求めることは，テストパターン数 N_{tp} が全ての認識精度計算において一定であれば， 各 σ における正解数 $N_{correct}$ の最大値を求めることに等しくなる．従って，各 σ の正解数である，カウンタの出力値の大小関係を σ 毎に比較することで，最終的に求めたい σ の最適値が求まる．以降の最大値検出の回路動作は，sPNN Processor の最大値検出（図 3.6），及び，最終段最大値検出（図 3.12）の動作と同様である．

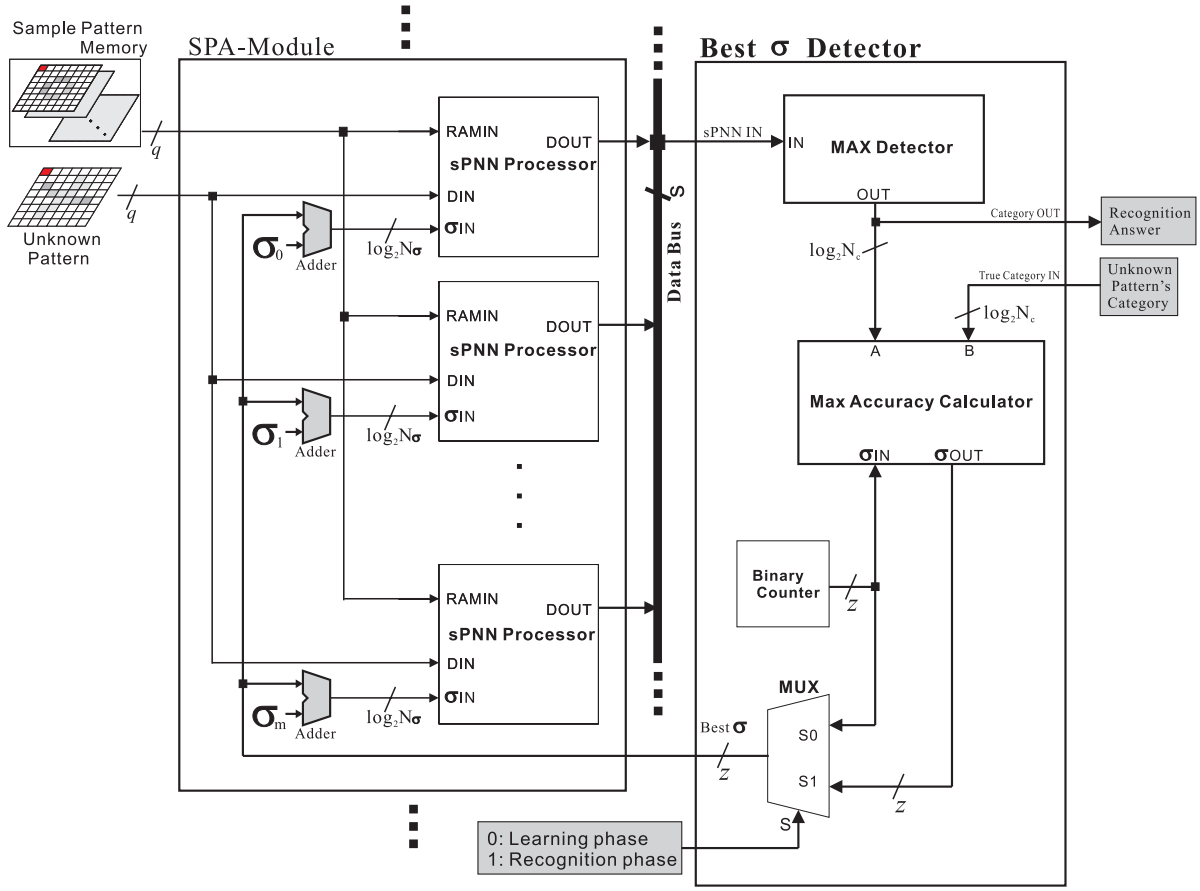


図 3.11　Best σ Detector の回路構成

3.6　画像認識システムの実装と評価

PNN による画像認識システムの開発を，下記のように 3 段階で行った．

1. 画像前処理（特徴抽出）評価システム（SusuPRB–P1），
2. sPNN Processor 評価システム（SusuPRB–M01），
3. 小型高速画像認識システム（SusuPRB–M02）．

本節では，各システムの詳細と評価結果について述べる．

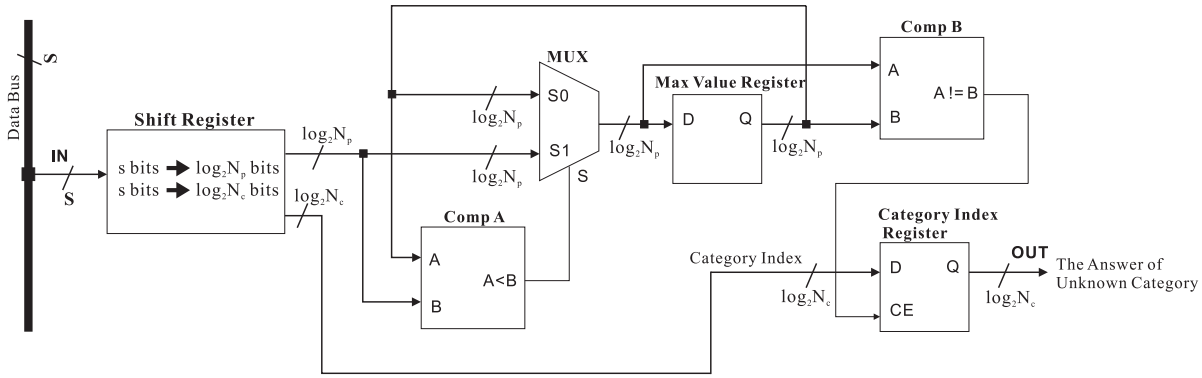


図 3.12　Best σ Detector 内の最大値検出器

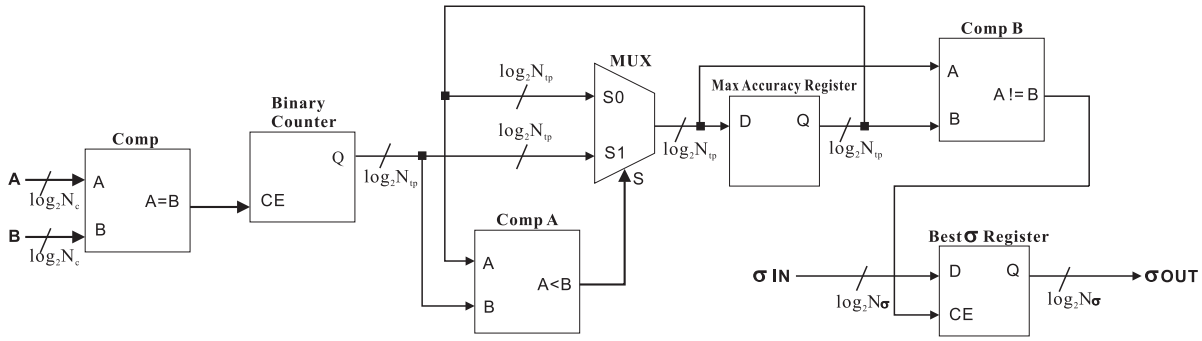


図 3.13　最大認識精度計算回路の構成

3.6.1　画像前処理（特徴抽出）評価システム（SusuPRB–P1）とその評価結果

システム構成

パターン認識のターゲットとしている “画像” を，認識システムへ取り込む最も適した入力方法は，ビデオカメラを用いることである．現在，一般的によく利用されているビデオカメラの出力信号は，NTSC(National Television System Committee) コンポジット信号，及び RS-170A であり，またこれらの規格に基づいたビデオカメラは，秒間 30 フレームの撮像，及び画像出力が可能である．従って，これらのカメラを用いることで，高速な画像認識が可能となる．そこでまず，この NTSC コンポジット信号を入力として，33.3ms 毎に画像を取り込む “NTSC デコーダ”，取り込んだ画像から特徴抽出を行う “プリプロセッサ”，さらに特徴抽出の

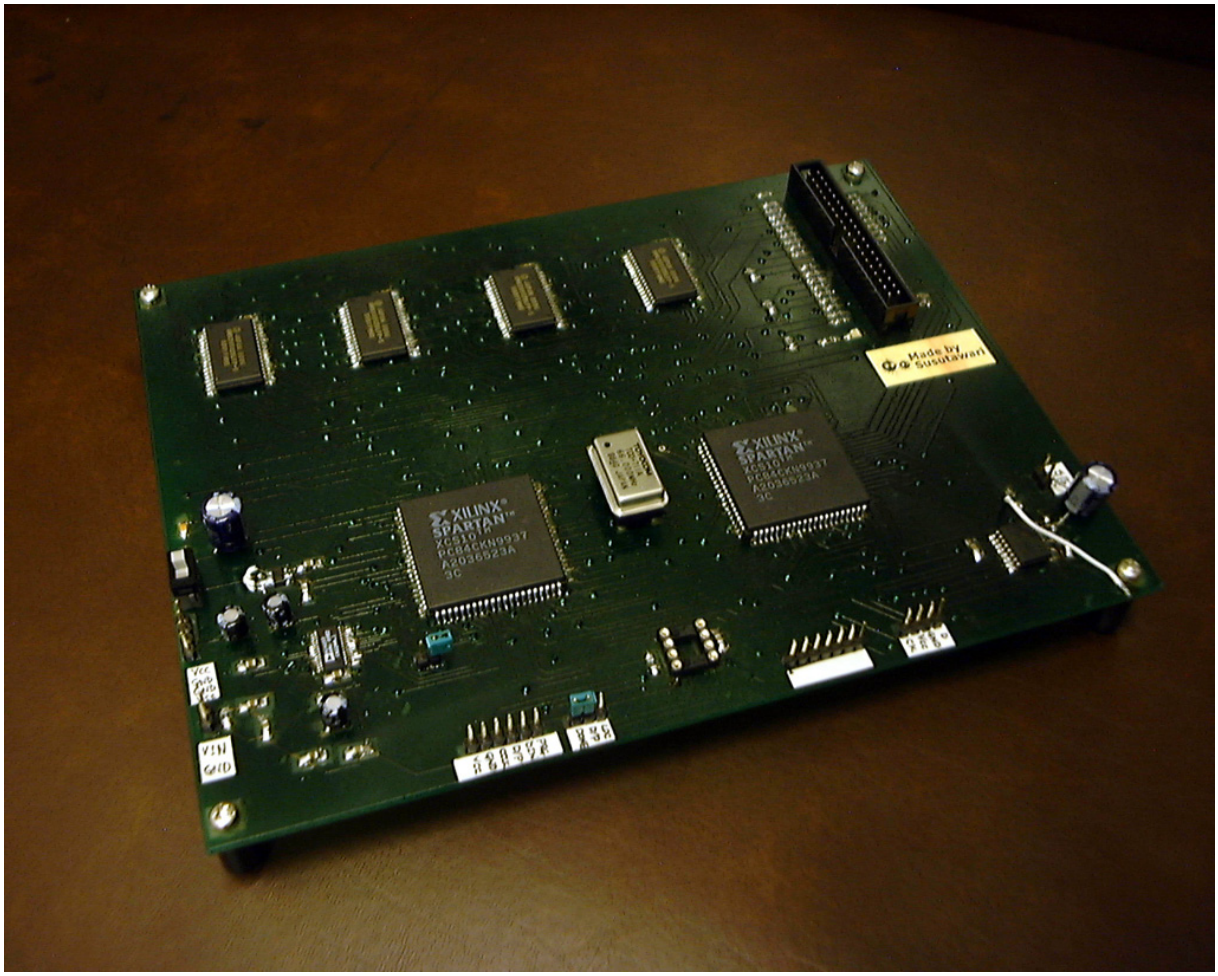


図 3.14　画像前処理（特徴抽出）評価システム（SusuPRB–P1）の写真

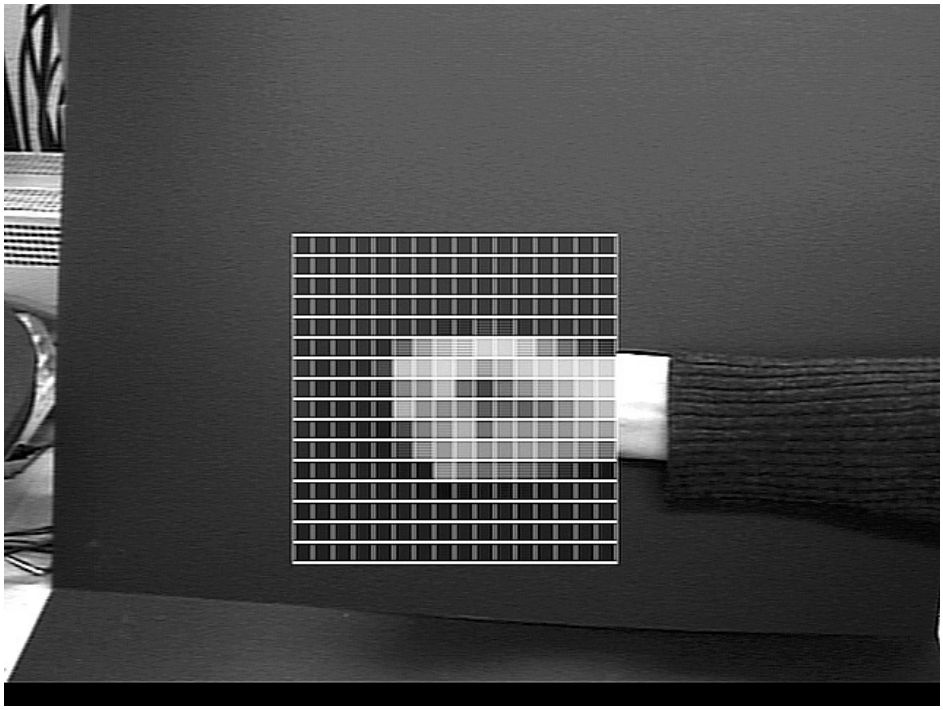
結果を確認するために再び NTSC 信号へ変換する “NTSC エンコーダ” を開発した．

図 3.14 に開発した画像前処理評価システムの写真を示す．NTSC デコーダ／エンコーダ，及びプリプロセッサは XILINX 社の FPGA XCS30–4PLCC84C[67] を 2 つ用いて製作した．本ボードはこの他に，NTSC 信号の入出力段にそれぞれ A/D コンバータ，及び D/A コンバータを，さらに画像を記憶するための SRAM を搭載している．

評価結果

本システムで処理した画像の前処理結果の例を図 3.15，3.16，3.17 に示す（これらは NTSC エンコーダの出力をパソコンにつけたビデオキャプチャボードで直接取り込んだものである）．本画像前処理評価システムより，開発したプリプロセッサが，任意のモザイク・パターンの生成（特徴抽出）を 16.7ms（1 フィールド）毎に行うことができることがわかった．

Original Picture



Pre-processed Picture

図 3.15　画像前処理結果の例 1

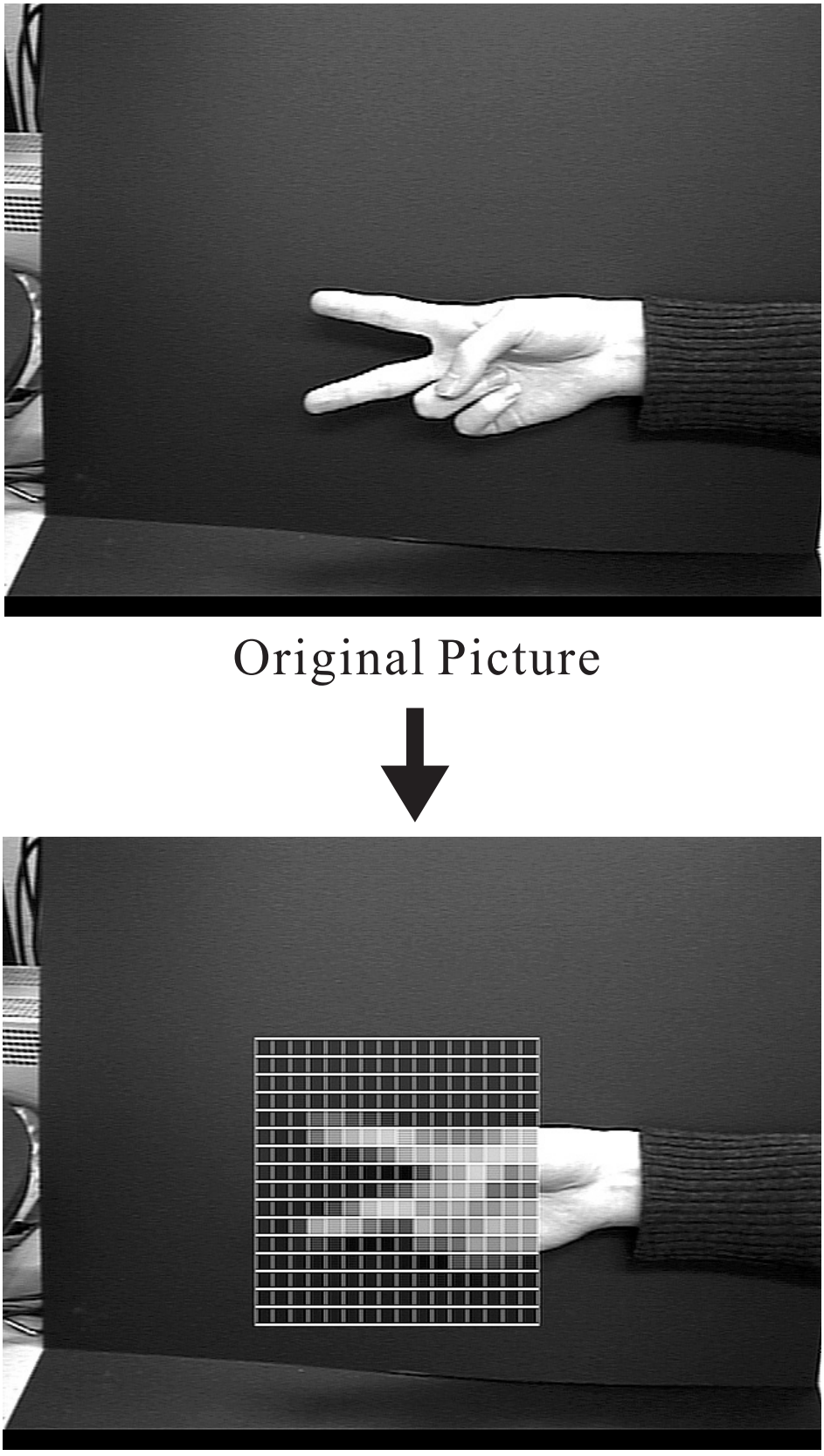
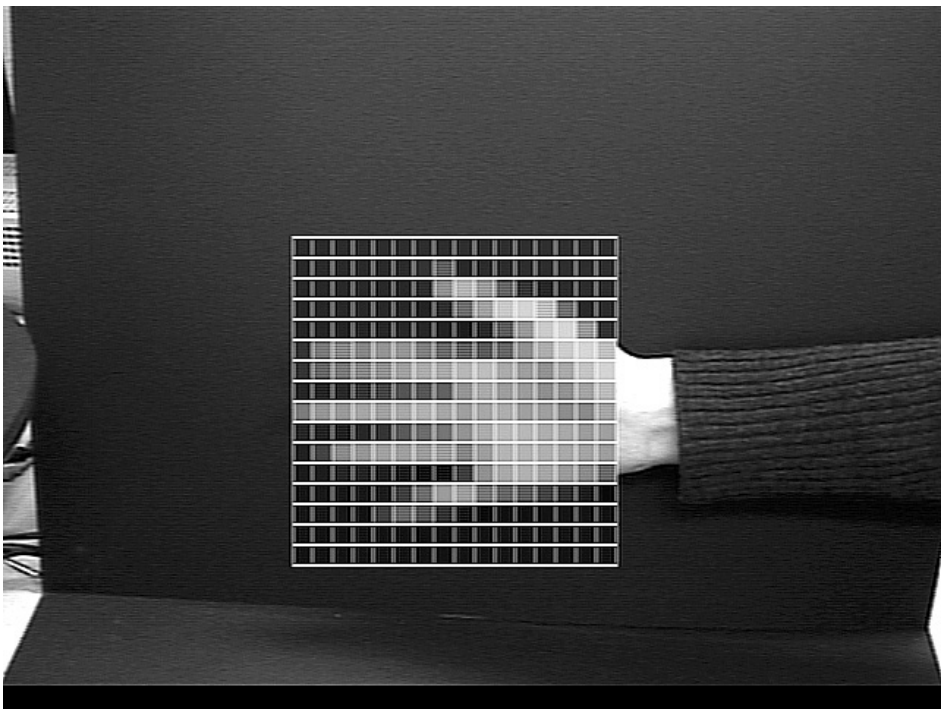


図 3.16　画像前処理結果の例 2

Original Picture



Pre-processed Picture

図 3.17　画像前処理結果の例 3

3.6.2 sPNN Processor 評価システム（SusuPRB–M01）とその評価結果

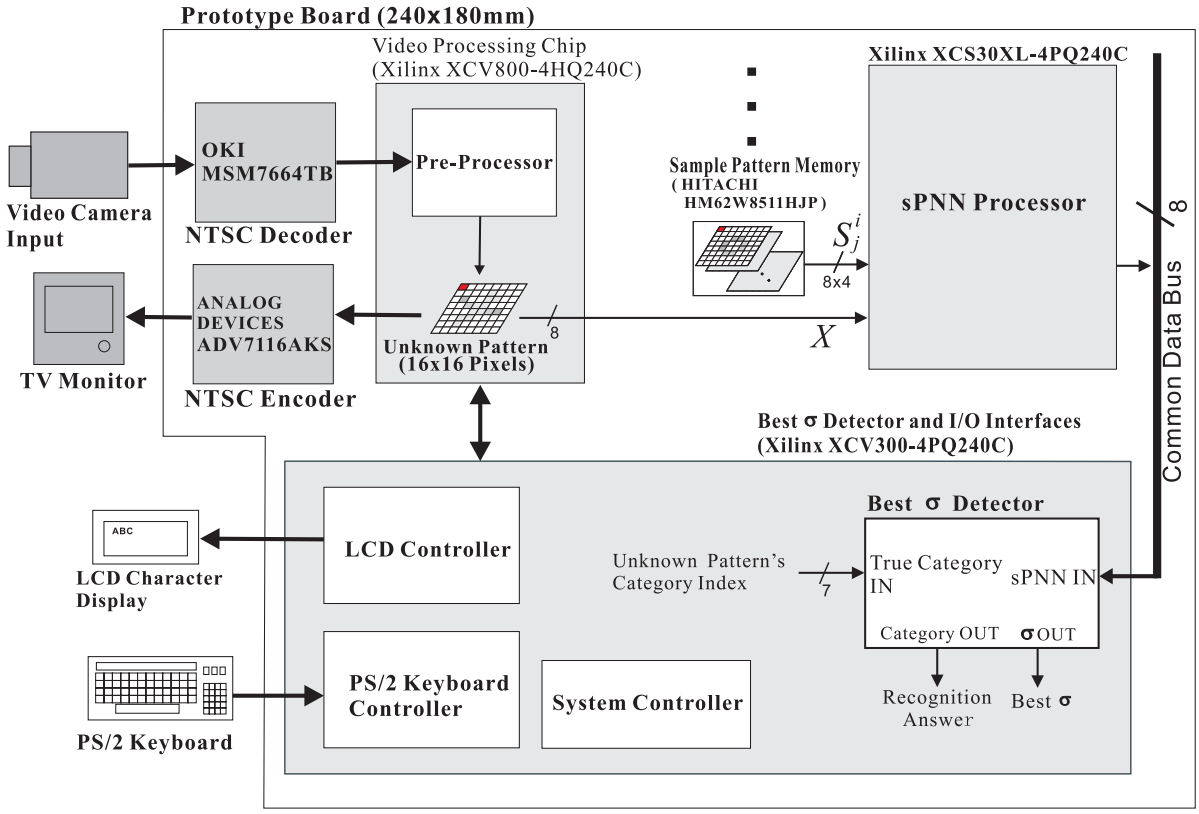


図 3.18 sPNN Processor 評価システム（SusuPRB–M01）の構成

システム構成

次に，sPNN Processor の実装評価を行うために，NTSC ビデオカメラからの入力パターンを取り込み，画像前処理回路も行う評価システムを作成した．図 3.18 にシステムの全体構成図を示す．また，図 3.19，3.20 に試作システムの写真を，表 3.2 にその諸元を示す．sPNN Processor 評価システムは sPNN Processor を実装する FPGA として，XCS30XL–4PQ240C（約 30,000 ゲート規模 [67]）を用いており， XCS30XL には約 3 つの sPNN Processor が実装可能である．

図 3.18 に示すように，ビデオカメラからの入力信号は NTSC デコーダ（OKI 製 MSM7674）にて A/D 変換され，前処理用の FPGA（XILINX 製 XCV800–4HQ240C[65]）に入力される．前処理では特定のエリア（今回は画面の中心付近とした）を 16×16 セグメントにモザイク化 (複数画素の平均化) することで情報量の低減を行っている．前処理の結果は同じく

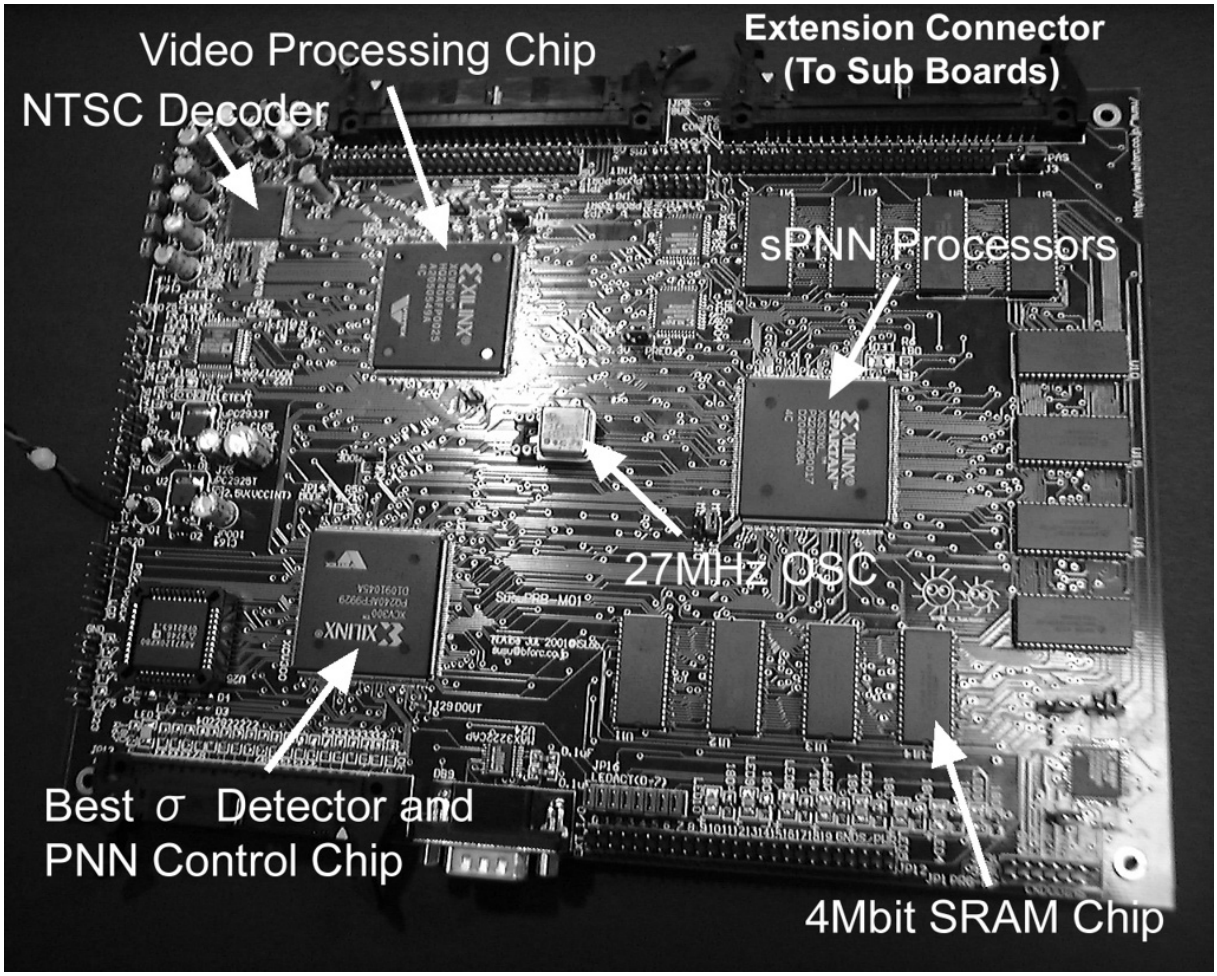


図 3.19　sPNN Processor 評価システム（SusuPRB–M01）の写真 1（ボード単体）

XCV800–4HQ240C に接続された NTSC エンコーダ（AnalogDevices 製 ADV7176AKS）によりテレビモニタに表示して確認することができる．16×16 セグメントにモザイク化された未知入力画像は sPNN Processors に送られる．sPNN Processors では，サンプルパターンと比較を行い，その結果から最も正しいと推測される未知入力パターンのカテゴリ・インデックス番号が出力される．サンプルパターンを記憶しておくためのメモリとして，日立製の 4Mbit-SRAM（HM62W8511HJP-12）を 12 個搭載しており，各データバスは全て独立に XCS30XL–4PQ240C と接続されている（各データバス幅 8bit×12=計 96bit 幅）．前述の Best σ Detector と，全体の制御を行う回路は，XILINX 製 XCV300–4PQ240C[65] にて実現している．

この FPGA を通して，外部からのキーボード操作によるコマンド実行や，認識及び学習結果のディスプレイ装置への表示などを行うことができる. 今回，外部からのコマンド入力装置と

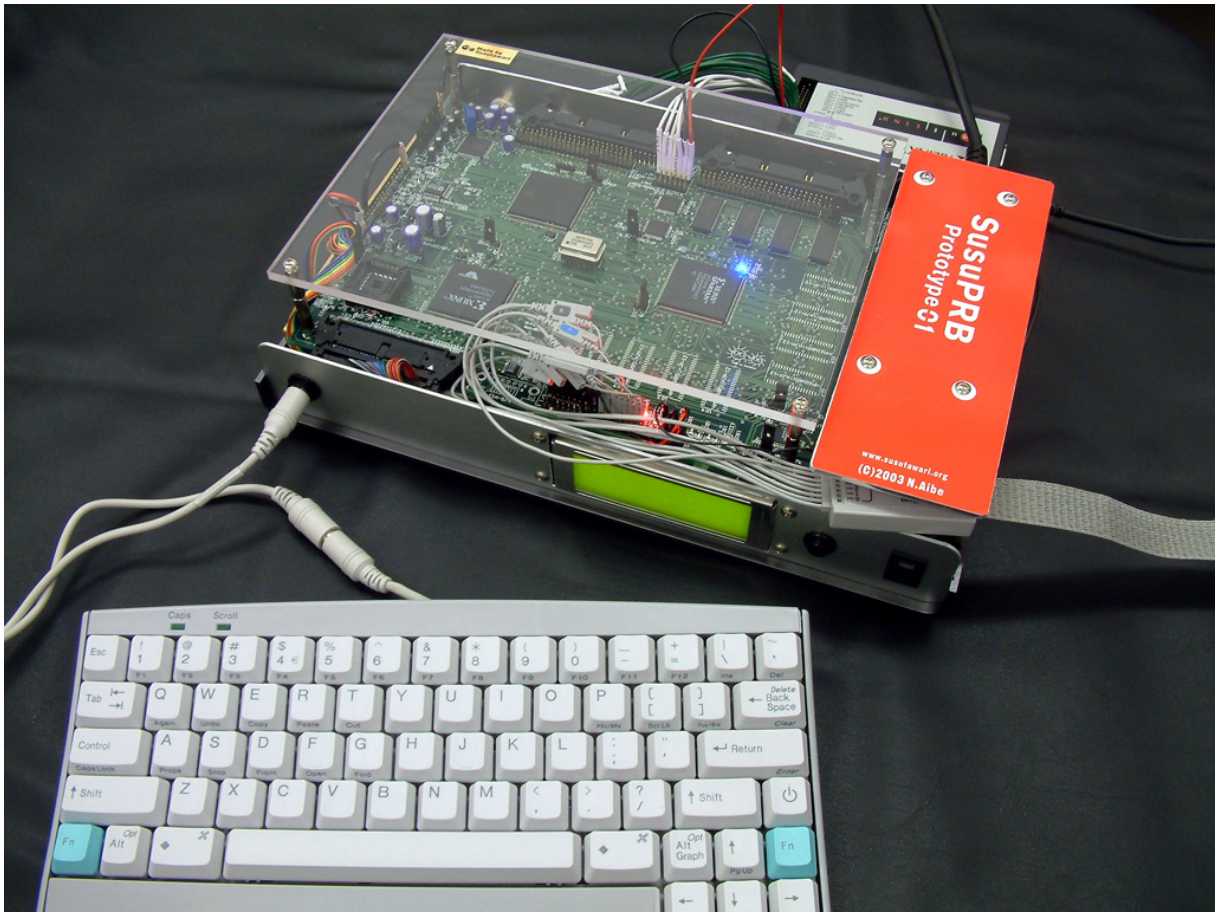


図 3.20　sPNN Processor 評価システム（SusuPRB–M01）の写真 2（稼動中）

してキーボードを選んだのは，要求される物理的なピン数が少ないためである．これにより，データバスなど，全体の処理性能に関係するピンへ割り当てることができ，結果としてバス幅を増やすことができた．なお，入力インターフェースとして使用するキーボードは，一般的に利用されている PS/2 キーボードを使用し，XCV300 内にデコーダを作成した．

認識時間の測定結果

NTSC ビデオカメラから取り込んだ 1 パターン（16 × 16 segment ×8bit のモザイク画像）について，1 つの sPNN Processor でカーネル関数の演算が完了するまでに要する時間を測定した．このときの測定条件を表 3.3 に，測定結果を表 3.4 に示す．これから，ビデオレート時間（33.3ms）では約 1,750 パターンを処理できることがわかった．また，1 カテゴリあたり 256 サンプルパターン（$N_p = 256$）とすると，1 つの sPNN Processor で，6〜7 カテゴリ分（$N_c = 6 \sim 7$）の処理をビデオレート時間内に実行することができることがわかった．

また式 (3.5) において，$n = 256$，$q = r = s = 8$ bit，$N_c = 1$，$N_p = 256$，$N_{Mod.} = 1$，

表 3.2 sPNN Processor 評価システム（SusuPRB–M01）の諸元

PNN 用 FPGA	XILINX XCS30XL–4PQ240C
Pre–Processor 用 FPGA	XILINX XCV800–4HQ240C
I/O Interface 用 FPGA	XILINX XCV300–4PQ240C
基本動作周波数	27MHz(sPNN Processor:13.5MHz)
プロセッサ数	3 sPNN Processors (Max)
メモリ容量	48Mbit (4Mbit×12)(Max)
メモリバス幅	96bit (8bit×12)(Max)
基板サイズ	240×180mm (4 層基板)
インターフェース	NTSC コンポジット入出力, PS/2 キーボード， ATA， キャラクタ LCD, Analog-RGB 出力， RS-232C

$\alpha = \gamma = 0$，$T_{CLK} = 1/13.5\text{MHz}$（$\simeq 74\text{ns}$）とすると，$T_{recog.} = 4.85$ ms で，これは実際に試作システムで測定した結果と等しく，この算出式（3.5）が正しいことを示している．

表 3.3 認識処理時間の測定条件

パターン次元数 (n)	256 （16 × 16 セグメント モザイク画像）
各要素のビット精度 (q)	8（bit）
サンプルパターン数 (N_p)	256
基本処理時間 (T_{CLK})	約 74ns(1/13.5MHz)

表 3.4 測定結果

測定範囲	処理時間
1 サンプル当たり	18.96us
1 カテゴリ当たり	4.85ms

学習時間の推定と PC との比較結果

学習時間は基本的に，前処理の繰り返し処理であるため，前述した認識処理の測定結果と，学習時間の算定式 (3.4) より見積もることができる．式 (3.4) から算出した学習時間，および参考比較のために行った，PC (Personal Computer) での処理時間を表 3.5 に示す．また，使用した PC の諸元を表 3.6 に示す．この結果から SPA によって構成された認識システムが，PC 上で処理される場合に比べ約 68 倍高速であることが導かれる． ただし，より厳密な比較を行う為にはプログラムの実行環境のチューニングなどを行ってさらに検証する必要がある．

表 3.5 学習時間

	$N_c = 8$	$N_c = 32$
64 sPNN Processor	2.5[s]	10.0[s]
PC	170[s]	682[s]

認識精度の評価結果

SusuPRB–M01 を用いて，白熱灯（500W ハロゲンランプを使用）の下で，ビデオカメラの前に任意に提示された手の動画像から，“ジャンケン” に関する 3 つのカテゴリ（C_1: グー，C_2: チョキ， C_3: パー）を認識する実験を行った．実験で用いた各カテゴリのサンプルパターンの一部を図 3.21，3.22 に示す．各サンプル画像は原画像と前処理（モザイク化）後の画像で，前処理後の画像では，画面の中央に 16×16 セグメントにモザイク化された領域があり，この領域に入った手の形状が認識される．各パターンは $q = 8$，$n = 16$，$N_p = N_{tp} = 256$，$N_c = 3$ で，また $N_\sigma = 256$，$N_{sPNN} = N_{Mod.} = 1$，$r = s = 8$ である．実験の結果，画面のほぼ中央に任意に提示したグー，チョキ，パーをほとんどの場合において正しく認識することができた．しかし，本評価システムではサンプルパターンをビデオカメラからの入力画像以外から生成できない，実験試行毎に取り込み直す必要がある，などの理由から正確な認識精度の測定が行えなかった．そこで次に，PC との接続を考慮し，PC からサンプルパターンや未知入

表 3.6 PC の諸元

CPU	Intel Pentium III–1GHz
OS	Linux
使用言語	ANSI C (GNU C Compiler)

力パターンを転送できる評価システム SusuPRB–M02 を開発した．SusuPRB–M02 は sPNN Processor 実装用の FPGA の回路規模を大きくし，Hybrid–SPA（図 3.7）の実装評価を可能とした．SusuPRB–M02 が，本研究の FPGA の PNN への適用性能を調べるための最終システムとなった．

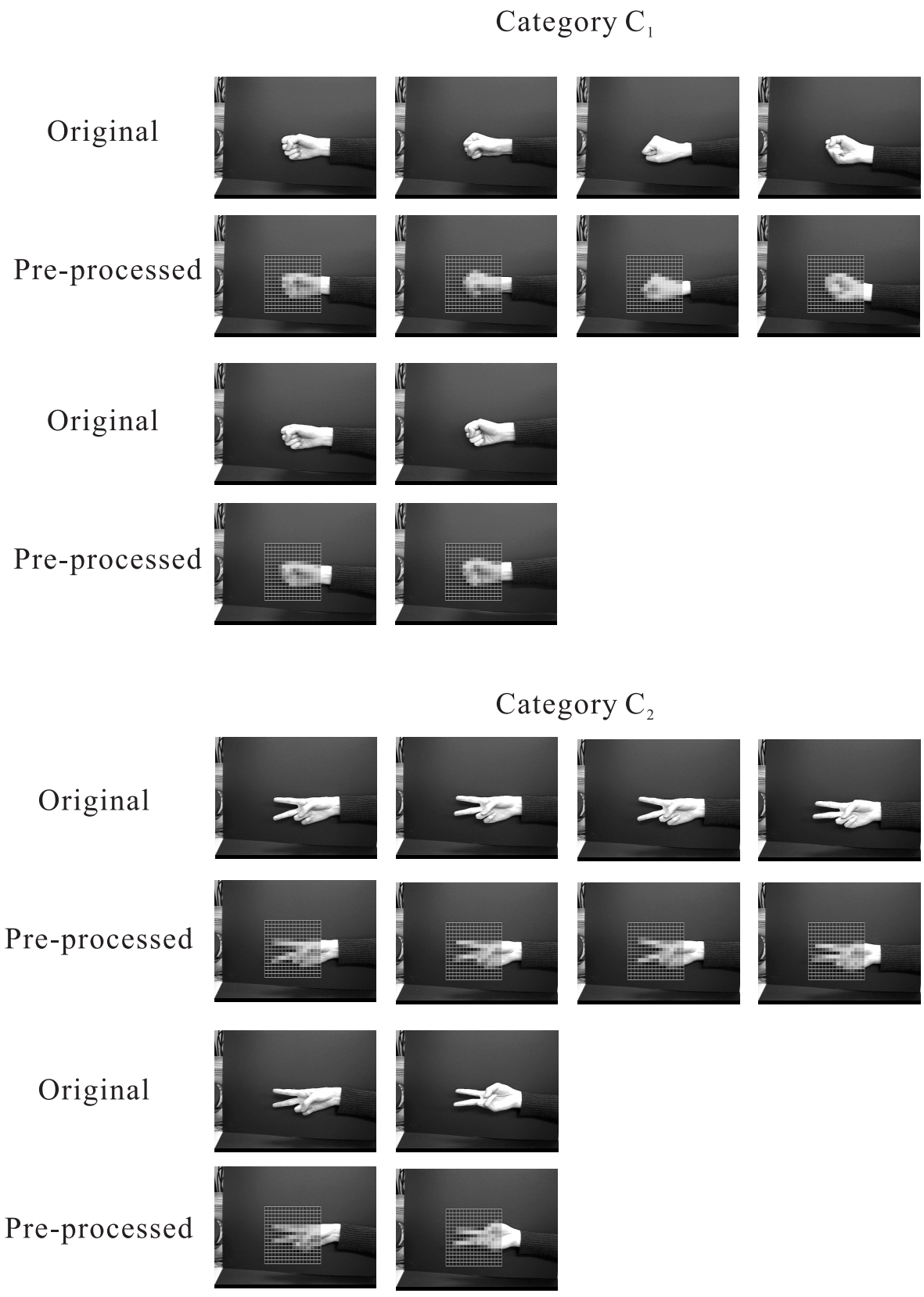


図 3.21　認識実験に用いた画像パターンの例 1

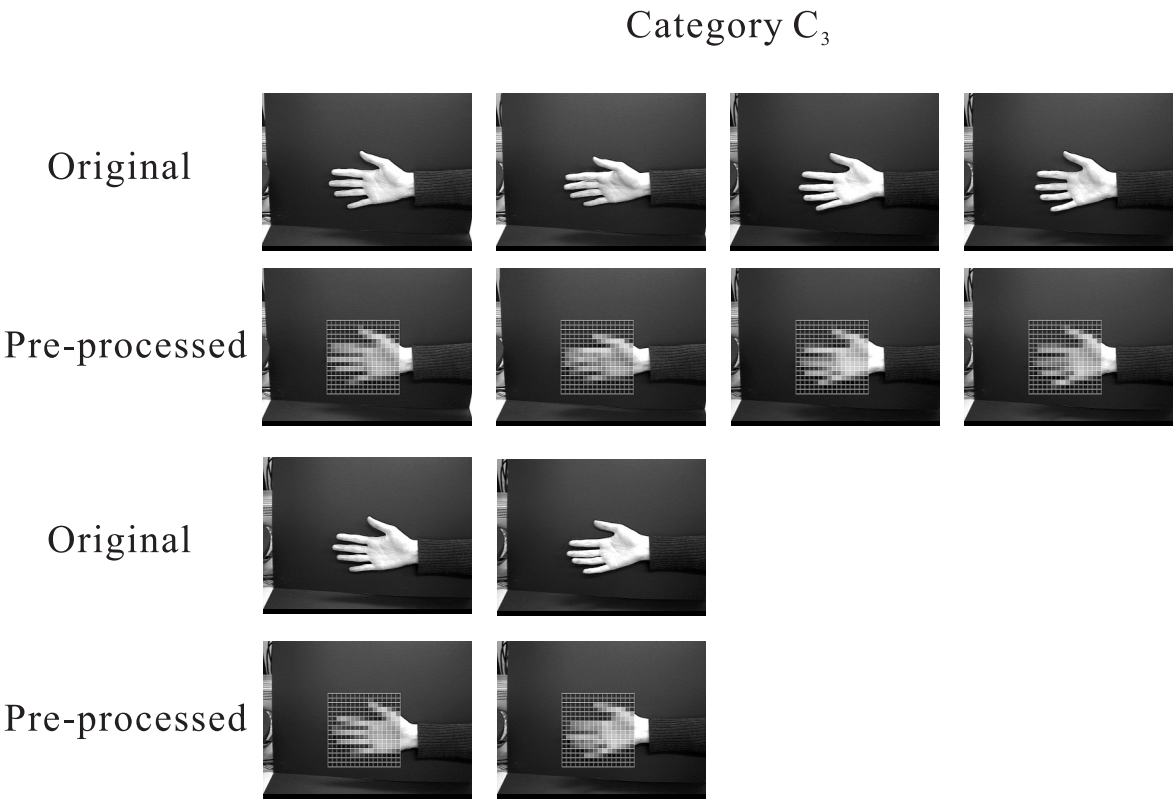


図 3.22　認識実験に用いた画像パターンの例 2

3.6.3 小型高速画像認識システム（SusuPRB–M02）と評価結果

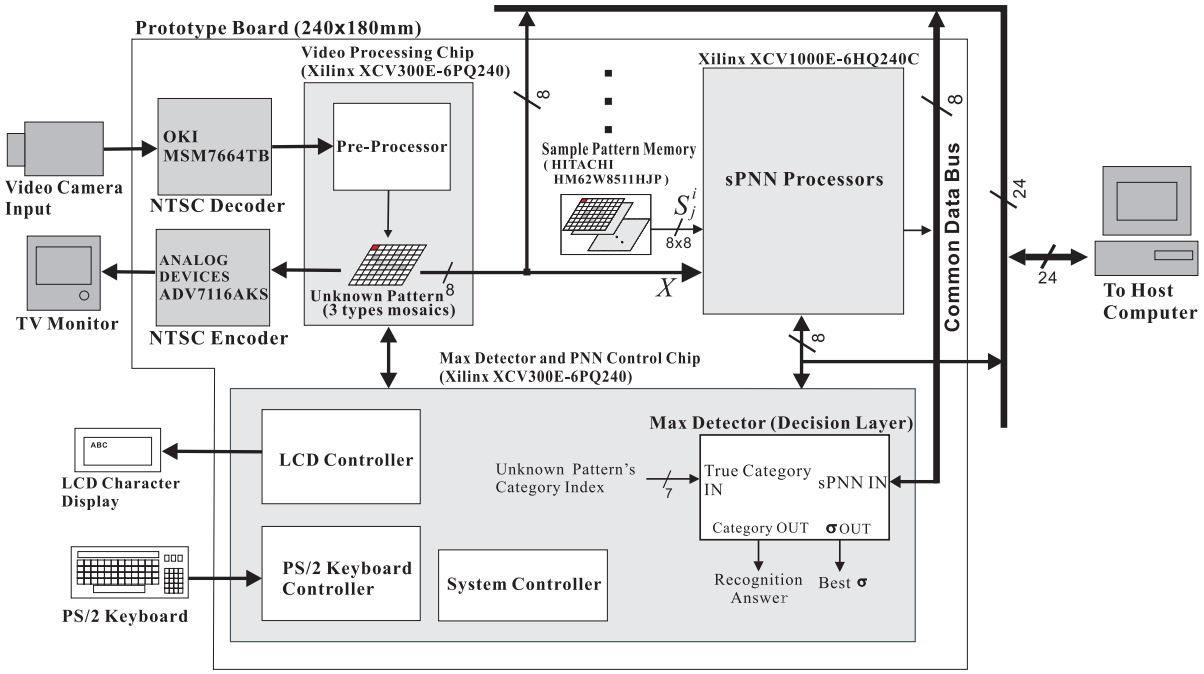


図 3.23　小型高速画像認識システムの構成

システム構成

最終システムとして，図 3.23 に示す小型高速画像認識システムを製作した．最終システムの諸元を表 3.7 に，写真を図 3.24，3.25 に示す．最終システム（SusuPRB–M02）は，SusuPRB–M01 の XCS30XL を XCV1000E[66] に変更し，PC（Host Computer）との専用インターフェースを搭載したことが主な変更である．これにより，64 個の sPNN Processor が実装可能となった他，PC との接続による評価（PC 上で広く用いられているベンチマークの適用など）が可能になった．PC との接続には PC 側に Interface 社製の PCI・パラレル I/O ボード PCI–2703 を搭載し，10-bit TTL（データ 8-bit，制御 2-bit）による接続を行った（SusuPRB–M02 の仕様としては最大 24-bit I/O の接続が可能）．このとき FPGA の I/O を効率良く利用するために，認識／学習処理実行時に使用する I/O を共有化して（バス構成として）PC との接続に用いた（図 3.23）．

表 3.7 小型高速画像認識システム（SusuPRB–M02）の諸元

PNN 用 FPGA	XILINX XCV1000E–6PQ240C
Pre–Processor 用 FPGA	XILINX XCV300E–6PQ240C
I/O Interface 用 FPGA	XILINX XCV300E–6PQ240C
基本動作周波数	27MHz(sPNN Processor:13.5MHz)
プロセッサ数	64 sPNN Processors (Max)
メモリ容量	32Mbit (4Mbit×8)(Max)
メモリバス幅	64bit (8bit×8)(Max)
基板サイズ	240×180mm (4 層基板)
インターフェース	NTSC コンポジット入出力, PS/2 キーボード， ATA， キャラクタ LCD, 10BASE–T Ethernet， RS–232C 専用パラレル PC インターフェース

より正確な認識精度の評価

PC との接続が可能となったことにより，SusuPRB–M01 の場合より，より正確な認識精度の測定が可能となった．SusuPRB–M01 の場合と同様に，3 カテゴリの手形状（ジャンケン）画像の認識を行った結果を図 3.26 に示す．図は横軸がパラメータ $\sigma/2$ で，縦軸が認識精度である．SusuPRB–M02 では前処理を 16×16 のモザイク化に加え，4×4 及び，8×8 の場合についても行った．その結果，8×8 の場合が最も高い認識精度を得られ，最高 98.3% であった．またこのときの $\sigma/2$ は 37 であった．

PC 向けベンチマークによる評価

さらに，PC 向けの顔認識用ベンチマークである，オリベッティ研究所 (ORL: Olivetti Research Laboratory) の顔画像データベースを用いた認識精度の測定を行った．ORL の顔画像は 40 人（40 カテゴリ）で，1 人あたりのサンプル数は 10 枚である．従って総計 400 サンプルであり，これを 40 枚のテストパターンと，残り 360 枚のサンプルパターンに分けて，認識精度の測定を行った．前処理は手形状認識の場合と同様に，4×4，8×8，16×16 の 3 種類のモザイク化を行った（図 3.27）．各セグメントのビット精度，$\sigma/2$ も同様に 8-bit である．ただし，手形状認識実験の場合と異なり，リアルタイムではなく，予め PC 上で前処理を行い，SusuPRB–M02 上の SRAM にカテゴリ別に転送することで行った．結果，手形状認識の場合と同様に，8×8 のモザイク化の場合が最高認識精度を達成し，90.3% であった．またこのときの $\sigma/2$ は 36 であった．

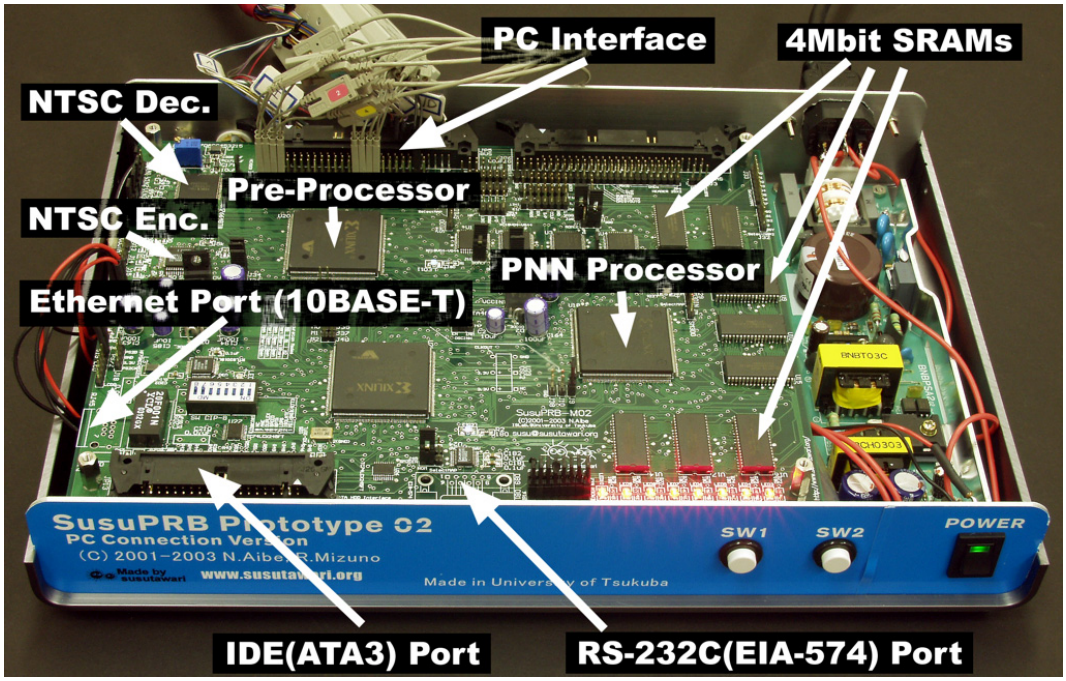


図 3.24　小型高速画像認識システムの写真

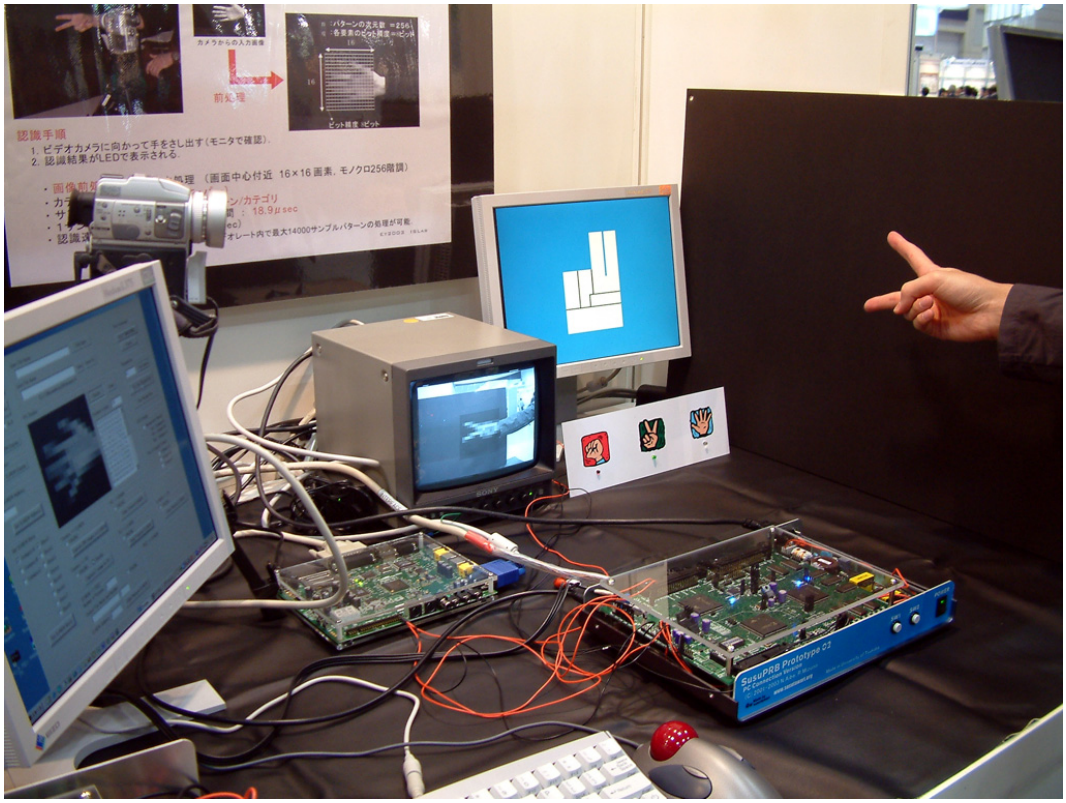


図 3.25　小型高速画像認識システムによる手形状認識の様子

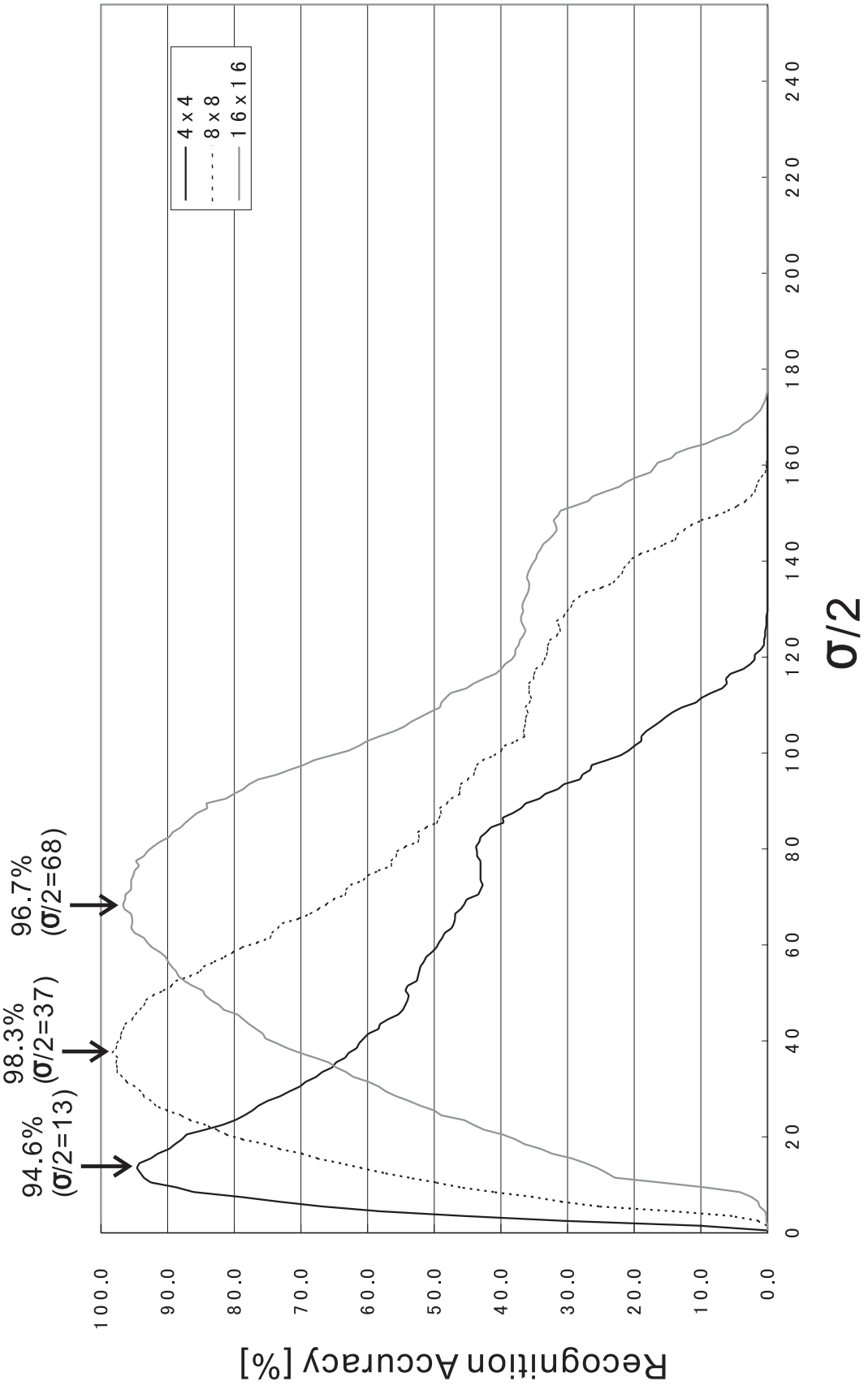


図 3.26　手形状（ジャンケン）画像の認識精度測定結果

図 3.27　オリベッティ顔画像と前処理後

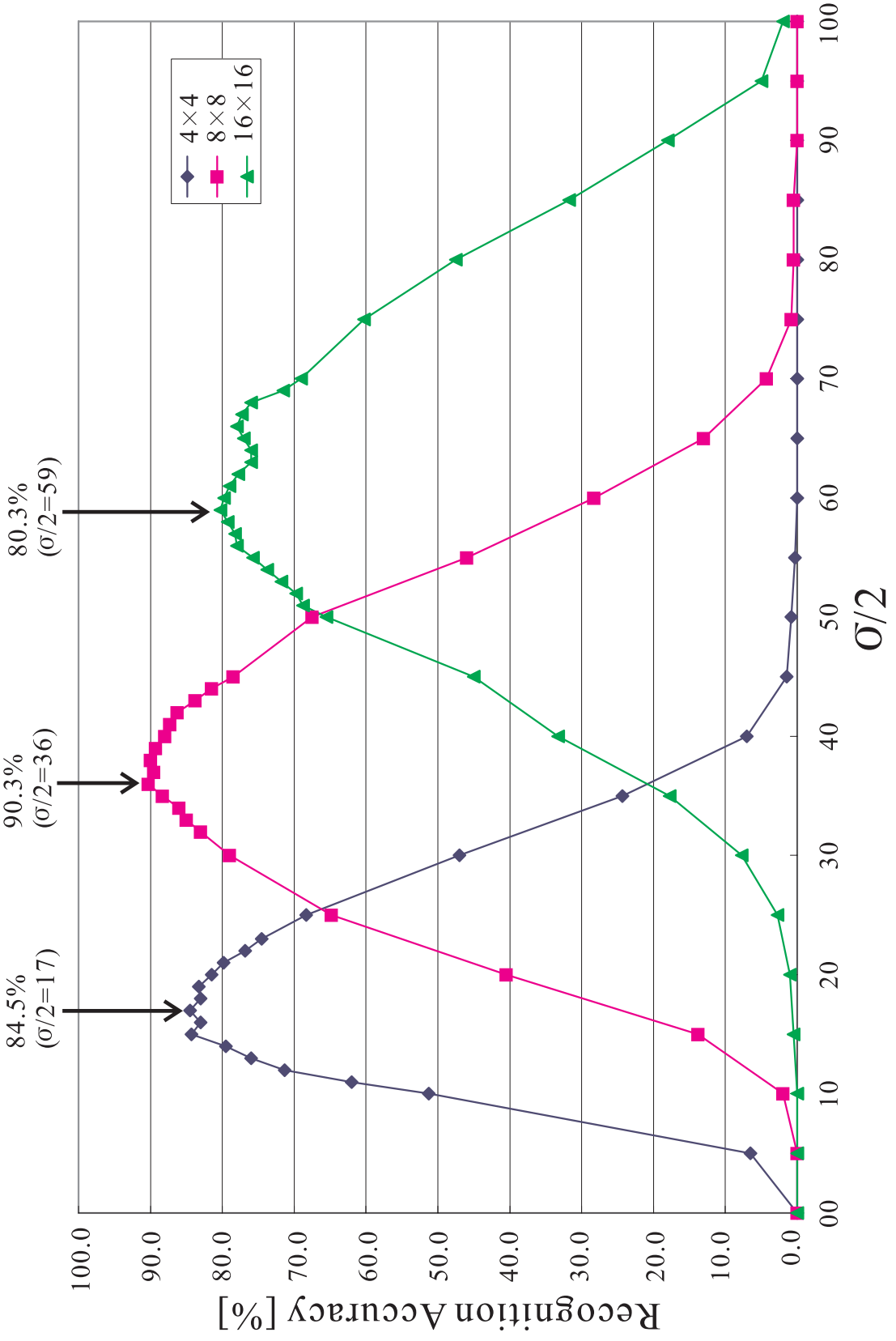


図 3.28　オリベッティ顔画像の認識精度測定結果

3.7 まとめ

本章ではまず，画像認識問題に限らず，カーネル・ベースの統計的なパターン認識で常に問題となる，パラメータの決定を高速学習するために以下の回路とその構成を提案した．

1. 確率的ニューラルネットワークのカーネル関数を単純な組合せ回路で実現する専用プロセッサ sPNN（serial-PNN）Processor.
2. 異なる学習パラメータを並列に処理する学習アーキテクチャ Sigma-Parallel Architecture（SPA）.
3. さらに従来の並列化手法（ニューロン並列）も組み合わせることで，より高速な学習を実現するアーキテクチャ Hybrid-SPA.

また，FPGA 上に実装された sPNN Processor，及び SPA と，それによって構成される認識システムの性能評価を行うために，以下の実装評価を行った．

1. NTSC ビデオカメラからリアルタイム（33.3ms）に特徴抽出（4×4，8×8，16×16 セグメントのモザイク・パターンの生成）を行うプリプロセッサの開発（SusuPRB–P1）.
2. sPNN Processor の認識時間の測定と学習時間の推定，及びプリプロセッサを用いた認識システムの開発（SusuPRB–M01）.
3. 正確な認識精度の測定と一般的なベンチマークによる評価（SusuPRB–M02）

これらの評価から以下の結論を得た．

定量的に，

1. **市販の** *FPGA* **である** XILINX XCV1000E-6HQ240C **を用いて，** *3* **カテゴリの手形状（ジャンケン）認識において，**33.3ms **で** 98.3%**，オリベッティ顔画像ベンチマークにおいて** 90.3% **の認識精度を得られる画像認識システムが実現可能であることがわかった．**
2. **上記性能を達成するために同** *FPGA* **で実現可能な学習時間は** 10.0[s] **であり，これは，**PentiumIII-1GHz **の** *PC* **の約** 68 **倍の処理速度が得られることがわかった**.

今回用いた特徴抽出方法は，モザイク化（低周波数での標本化）という単純な手法であったにも関わらず，定量的に上記のような結果を得た．これは *1* **カテゴリあたりのサンプル数が多いために，特徴抽出手法が最適でなくても，真の確率密度関数の近似精度が高まったためと推測できる．ただし，今回のサンプル群に対しモザイク化という特徴抽出がどの程度適切であったかは定量的に評価できていないので，今後評価方法も含めて検討する必要がある．従って，**

定性的に，

特徴抽出が困難であっても，多くのサンプルパターンを用意できる認識処理において，高い認識精度を実現できる見通しを得た．また，適切な特徴抽出を行える問題に対しては，同様に多くのサンプルパターンを用意できる場合に，従来より高い認識精度を実現できる見通しを得た．

第 4 章

IP パケット・フィルタ装置への適用

本章でも，前章と同様に，アルゴリズム並列性が非常に高く，FPGA のような LUT アレイ型のアーキテクチャに向いていると考えられるアプリケーションへの適用例として，ネットワーク分野でのアプリケーションについて述べる．本的用例の目的は，IP パケット処理における FPGA の実行性能を調べることである．本章では，

1. アクティブネットワークと IP パケット・フィルタ，
2. 性能可変アーキテクチャの提案，
3. 提案アーキテクチャを用いた IP パケット・フィルタ装置の実装評価，

の順で述べる．

4.1 アクティブネットワークと IP パケット・フィルタ

今日，我々が利用しているインターネットなどのコンピュータ・ネットワークは，年々その伝送速度や利用形態の種類が増しており，それに伴って，ネットワーク上のトラフィック制御や，サーバに対する負荷分散・低減技術はますます重要となってきている．そこで近年，ルータ装置などネットワークの中継装置などに，柔軟性をもたせ，個別に，より細かな制御を行わせることでトラフィックやサーバ負荷の低減を行う，アクティブネットワークという技術が注目されている [60][61]．アクティブネットワークは，ルータ装置などのパケット制御機構をプログラマブルにすることで，個別に，その設置エリアや用途などに応じて，適切な制御を行うことを可能にするものである．しかし，このような装置の実現には装置に高い柔軟性が要求され，また，ネットワークの分野では日々新しいプロトコルやデータ・フォーマットが提案されるため高い拡張性能が要求される．このような要求には固定アーキテクチャの ASIC を用いるよりも，従来の MPU ベースの処理の方が柔軟性という面では向いていると考えられる．しか

し，ネットワークの分野は現在最も伝送速度（及び各ノードでの処理速度）が望まれている分野の一つであり，従来の MPU ベースの逐次処理では速度面で実現が困難である．このようにアクティブネットワークの装置は柔軟性と高速性が共に要求されることから，リコンフィギャラブル・アーキテクチャ向けのアプリケーションであるといえる [35][61]〜[63]．本章では市販の FPGA で実現できるアクティブネットワーク向け IP パケット・フィルタ装置を対象とした．

IP パケット・フィルタ装置は，一般にルータなどのネットワーク機器の中に実装されているもので，普段我々がその存在を意識するものではないが，ネットワークのトラフィック制御やセキュリティ面で非常に重要な役割を果たしている．また，ネットワークの上流になればなるほどパケットのトラフィック量が増すため，より高速な処理が必要となる．逆に下流ほどトラフィック量は少ないが，詳細なパケットの制御が要求される．このため現在は上流（基幹ネットワーク）向けの IP パケット・フィルタ装置（ルータ装置）と下流（エンドユーザよりのネットワーク）向けのものが個別に製品化されている．ここでまず速度性能は，上流向けのものは ASIC 化されているため高速で，下流向けのものは MPU により逐次処理されているため低速である．一方コストに関しては，上流向けの装置には ASIC が採用されている上に，販売台数が少ないため非常に単価が高い．下流向けの装置は低価格の MPU が用いられている上に，量産されるため単価が安い．ここで特に基幹からエンドユーザを結ぶ，中流のネットワークには，速度性能及びコストで評価した場合に，適切な装置がないのが現状である．

そこで，ネットワークの状況に応じてその優先性能を変更できる，アクティブネットワーク向けの IP パケット・フィルタ装置とそのリコンフィギャラブル・アーキテクチャを提案する．また，実行性能とコストの両面から評価したとき，特に中流向けの装置として，FPGA による実装が向いているものと考えられる．

4.2　RPCA (Reconfigurable Parallel Comparation Architecture)

IP パケット・フィルタの主要な演算は，予め装置のメモリに記憶されたフィルタリング・ルールと，入力される IP パケットのもつ情報との比較演算である．そこで，この比較演算の高速化を図り，要求に応じて速度性能と最大比較可能なフィルタリング・ルールの量（データ量）のどちらを優先するかを動的に変更可能なアーキテクチャ，RPCA（Reconfigurable Parallel Comparation Architecture）を提案する．RPCA は FPGA を用いて，要求に応じた適切な回路構成を実現し，FPGA の持つ性能を最大限に活用する比較演算器の並列高速化アーキテクチャである．

入力データ（IP パケット）と被比較データ（フィルタリング・ルール）とを比較する比較演算器の並列化では，被比較データを予め回路内部のレジスタ（メモリ）に記憶しておく必要がある．回路内部に記憶せずに，外部メモリを用いることは処理速度の低下を招くだけでなく，必

要となる I/O 数不足の問題から非現実的である．ここで，フィルタリング・ルールを IP アドレスの一致検出とした場合，仮に IP アドレスを外部メモリに展開し，直接比較する IP パケット・フィルタ装置を考えると，IPv4 で 1IP アドレスあたり 32bit の I/O 数となり，128IP アドレスでも $32 \times 128 = 4,096$ も必要となる．一方，外部メモリを用いず，FPGA 内部に全てのフィルタリング・ルールを記憶する構成を考えると，システム全体に必要な回路規模 S は，比較演算器 1 つ当りの回路規模を $Comp.$，その並列度を n，被比較データ用レジスタの回路規模を $Reg.$，制御回路の回路規模を $Ctrl.$ とすると，

$$S = Comp. \times n + Reg. + Ctrl. \tag{4.1}$$

となる．このため限られた FPGA の回路規模 S の中で，演算器の数 $Comp. \times n$ と，レジスタの数 $Reg.$ のどちらを優先するかという構成上の選択が出てくる．ただしここで，制御回路の回路規模 $Ctrl.$ は構成によらずほぼ一定である．図 4.1 に RPCA が実現する処理速度とデータ量とのトレードオフ関係の概念を示す．図では 3 種類の回路構成を示しており，それぞれ，完全並列構成（A），混在構成（B），完全逐次構成（C）となっている．完全並列構成（図中 Completely Parallelized Configuration）は演算器と内部レジスタが被比較データ数分あり，入力データに対し同時並列に演算を行うため最も演算速度が高速だが，必要な回路規模が最も大きくなる．一方，対極的な関係にある完全逐次構成（図中 Completely Serialized Configuration）は演算器が 1 つだけで，内部レジスタは共有バスにシリアル接続されており，逐次読み出しされて演算を行う．このため最も回路規模が小さくなるが，最も演算が低速となる．混在構成（図中 Hybrid Configuration）はこれら 2 つの構成の中間的な構成で，逐次構成された演算器との内部レジスタ群が複数存在し，並列に動作する（i 個の演算器にそれぞれ j 個のレジスタが接続されるため，レジスタの総数は $j \times i$ 個）．このため，限られた回路規模の中で，処理速度（演算器数及び並列度）を優先するか，データ量（内部記憶容量）を優先するかを用途に応じて選択し，それに合う回路構成を実装する必要がある．

以上のように，RPCA で用いる「再構成」とは単に TAT 短縮，低コスト化するだけではなく，FPGA 上に実現された比較演算器のアーキテクチャ（構成）自体を変更することで，その再構成可能集積回路の使用率を上げ，処理性能を最大限に引き出すものである．具体的には 1 つのチップで，データ量が必要な場合と，速度が必要な場合にそれぞれ，必要に応じて対応することができる．まとめると，RPCA は次の場合に有効である．1）多量の比較演算処理が必要となる場合で，2）その比較演算処理の並列性が高く，3）データ量と速度のトレードオフ選択をユーザサイドで行う必要がある場合．また，RPCA は比較演算器の並列化アーキテクチャであるが，アプリケーションによっては同様の手法で，比較器以外でも，例えば加減算器，ALU などの構成を並列構成／逐次構成と要求に応じて再構成することで高い回路使用率と高速化／大容量化が望める可能性がある．

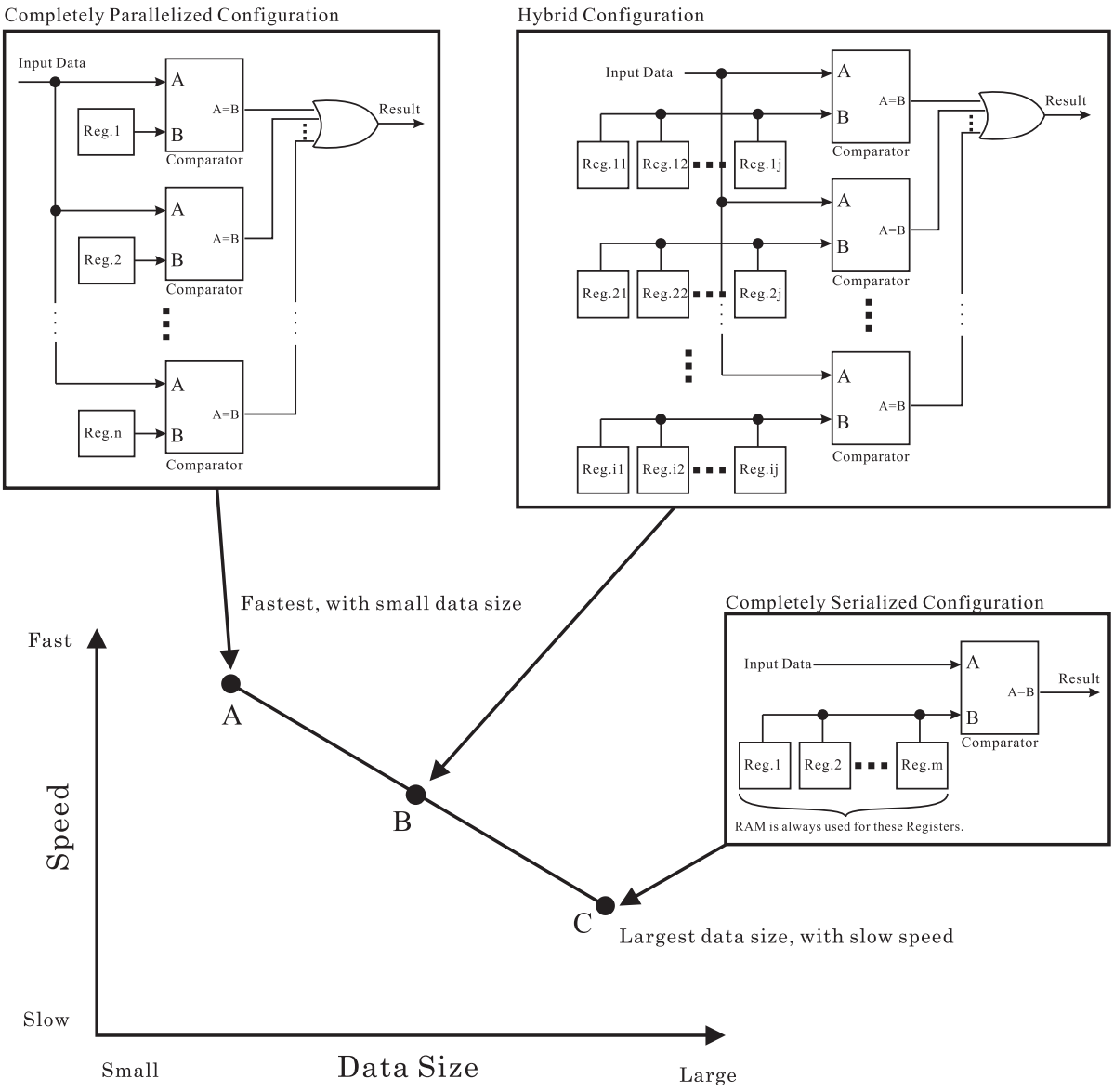


図 4.1　RPCA における速度とデータ量のトレードオフ関係

4.3　性能可変 IP パケット・フィルタ装置の実装評価

図 4.2 に RPCA を用いた IP パケット・フィルタ装置の構成を，図 4.3 に試作機の写真を示す．今回対象としたインターフェースは 10BASE–T で，EthernetII パケット・フレーム及び RFC791[78] IP パケット・フレームに準じた．フィルタリング・ルールは IP アドレスの一致検出によるものとした．

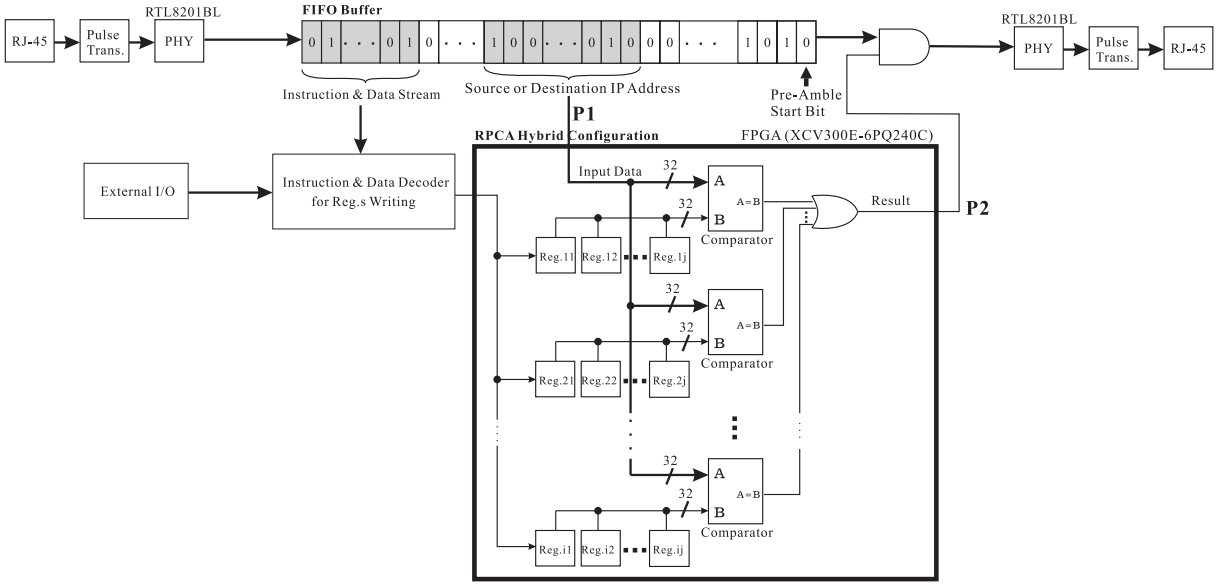


図 4.2　RPCA を用いた IP パケット・フィルタ装置の構成

回路の動作はまず，RJ–45 コネクタから入力された IP パケット信号がパルストランス，及び PHY チップ（電圧レベル変換，マンチェスタ符号・デコードなどを行うチップ）を通って FIFO（First In First Out）バッファに入力される．フィルタ処理では，ターゲットとなる IP アドレスが FIFO バッファに書き込まれた直後に RPCA で構成された比較器群により，その IP パケットを通過するか破棄するかを決定する．その結果に応じて FIFO バッファの出力にあるゲートが開閉し，IP パケットの通過／破棄制御を行う．このためフィルタの処理速度はターゲット IP アドレス獲得から比較演算結果出力までにかかる時間（図 4.2 中 P1 に入力されてから P2 から出力されるまでの時間）による．この時間を，XILINX 社製の FPGA である，XCV300E–6PQ240C[66] に完全同期設計で実装して測定した結果を図 4.4，4.5 に示す．図 4.4，4.5 はそれぞれ，完全並列構成（128IP アドレス）と最大逐次構成（2,048IP アドレス）の場合で，波形は上から，入力 IP パケットからデコードされたクロック（RXC），デコードされたデータ（RXD），ターゲット IP アドレス・ラッチ信号（IPDET），比較演算の出力（ANS）となっており，IPDET の立ち上がりが図 2 中の入力 P1 に，ANS の立ち上がりが P2 の出力に該当する．また，これは比較結果により IP パケットが破棄される場合を観測したものであり，処理時間は，P1–P2 間の時間となる．測定結果はそれぞれ完全並列構成（128IP アドレス）のとき 120ns，最大逐次構成（2,048IP アドレス）のとき 1,360ns であった．

IP パケットの最小長は 64Byte であるため，最初のパケットがフィルタ処理されてから次のパケットの処理が要求されるまでの最短時間 T_p[ns] は，データ伝送速度を X[Mbps] とすると，

$$T_p = \frac{1}{X} \times 64 \times 8 \times 1000 \tag{4.2}$$

となる．ここで，仮にパケットが絶え間なく入力されても On-The-Fly（遅延時間なし）でフィルタ処理できるために必要な条件は，フィルタ処理時間（P1–P2 間の時間）を T_{Comp}[ns] とすると，

$$T_{Comp} \leq T_p \tag{4.3}$$

である．従って On-The-Fly 処理を実現できる最大データ伝送速度 X_{Max}[Mbps] は，

$$X_{Max} = \frac{1}{T_{Comp}} \times 64 \times 8 \times 1000 \tag{4.4}$$

となる．今回の実装での測定結果は完全並列構成のとき $T_{Comp1} = 120$[ns]，最大逐次構成のとき $T_{Comp2} = 1,360$[ns] であることから，式（4）よりそれぞれの最大データ伝送速度 X_{Max1}，X_{Max2} は

$$X_{Max1} \simeq 4266.7[\text{MHz}]$$
$$X_{Max2} \simeq 376.5[\text{MHz}]$$

と，それぞれ 128IP アドレスのとき約 4.27Gbps，2,048IP アドレスのとき約 377Mbps のデータ伝送速度で On-The-Fly フィルタ処理を実現できることがわかった．なお，本測定で使用した比較演算器のベースクロックは 25MHz であり，使用している FPGA はより高速な動作も可能であることから，この動作周波数を上げることで 10Gbps 以上の On-The-Fly 処理を実現できる見通しも得た．

また，IP アドレス数が 128IP アドレスの場合と 2,048IP アドレスの場合の構成はそれぞれ処理速度を優先した構成と，データ量（IP アドレス数）を優先した構成である．理想的には，図 4.1 の理論図における完全並列構成と完全逐次構成の結ぶ線分がトレードオフの範囲となるが，実際に

XCV300E に 32bit の演算器とレジスタで構成した場合には図 4.6 のような関係になった．処理速度優先の構成は完全並列構成となるが，データ量優先の場合，その最大データ量を実現する構成は，完全逐次構成とならず，混在型（32 レジスタ ×64 並列構成即ち，図 4.1 中 Hybrid Configuration で $j = 32$, $i = 64$，また各レジスタ・サイズは 32bit）となった．これは，各比較演算器の出力に 3-State Buffer を用いたため，少ない 3-State Buffer 数がネックとなり，LUT を使い切れなかったことが原因と考えられる．このため，今後の課題として，出力にシフト・レジスタを用いて，高速なシリアル転送を行う方法など，3-State Buffer 数，配線リソース・ネックにならずに，全ての LUT を演算器，或いは IP アドレスを記憶するためのメモリとして使いきれるような構成の検討が必要である．尚，今回の 3-State Buffer を用いた回路において，IP アドレス数 128 と 2,048 の場合の構成の回路使用率はそれぞれ 99%，94% であった．

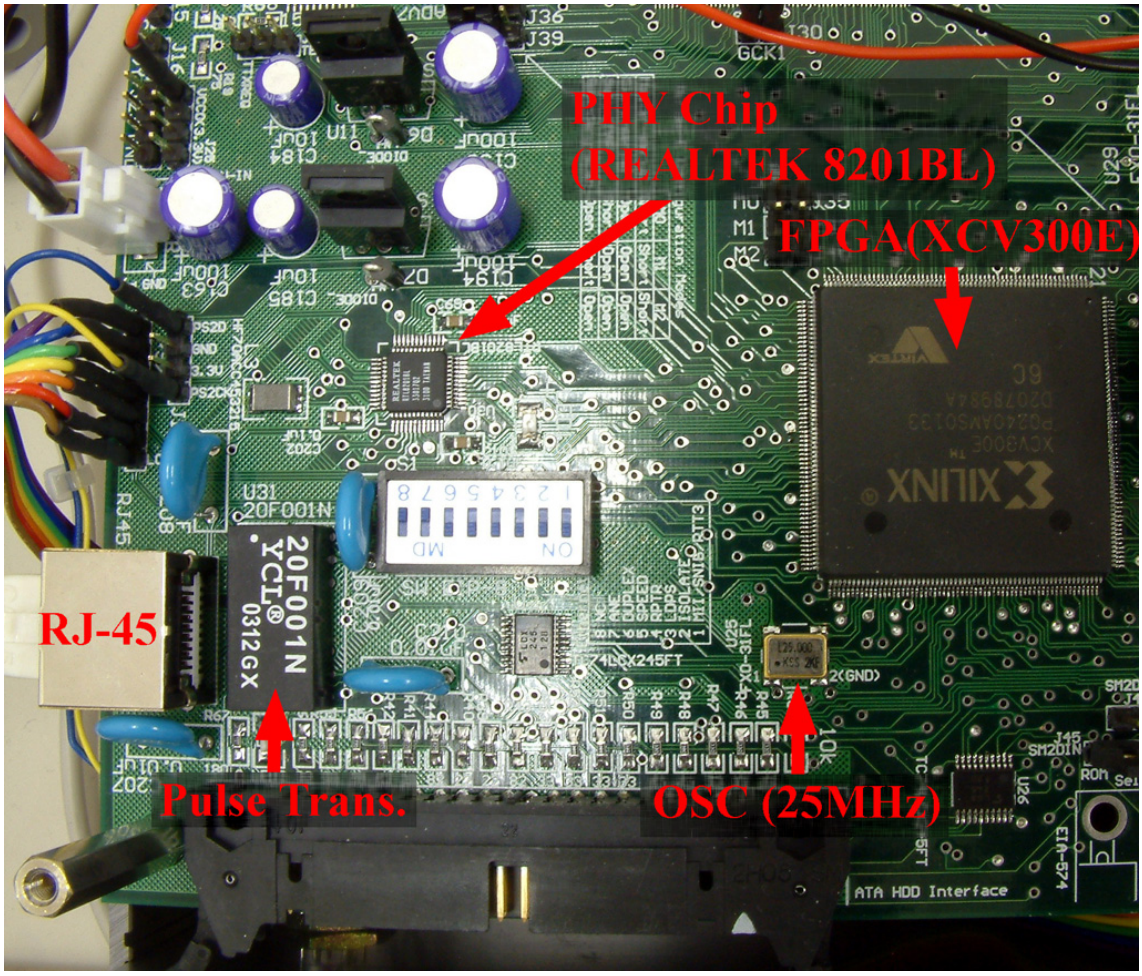


図 4.3　性能可変 IP パケットフィルタ装置の試作機の写真

4.4　まとめ

本章では，FPGA での比較演算処理において，その処理速度とデータ量のどちらを優先するかを選択可能なアーキテクチャ RPCA（Reconfigurable Parallel Comparation Archtecture）を提案し，その実装例としてアクティブネットワーク向けの性能可変 IP パケット・フィルタ装置を示した．実装評価の結果，次のことがわかった．

定量的に，

市販の FPGA **である** XILINX XCV300E-6PQ240C **を用いて，基本動作周波数** 25MHz **にて，速度優先で** 128IP **アドレス処理可能な場合の処理時間は** 120ns**，データ量優先で** 2,048IP **ア**

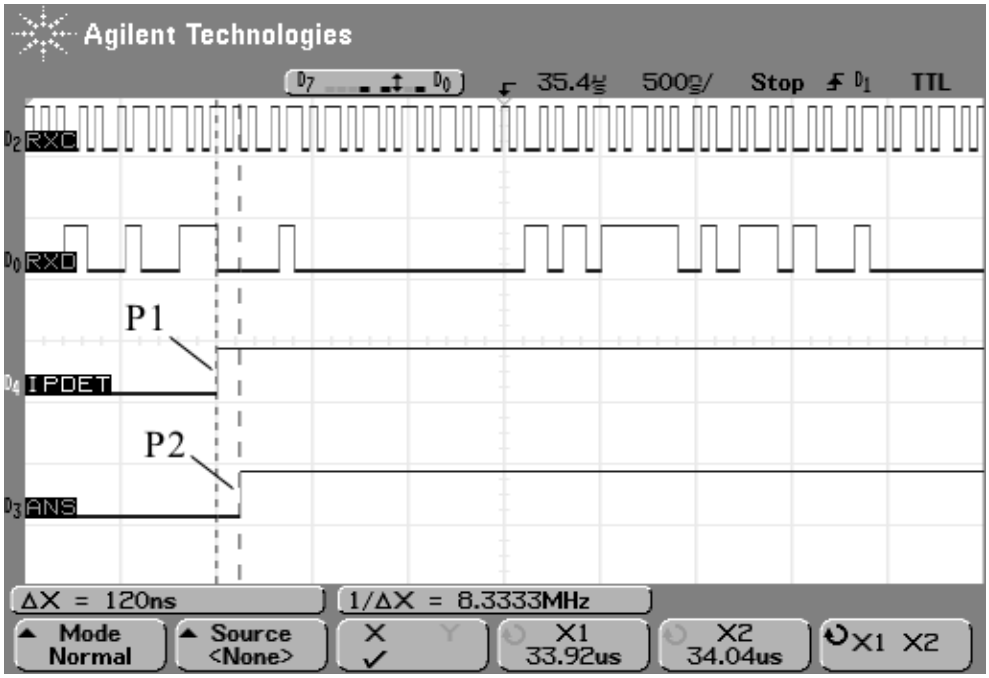


図 4.4　完全並列構成（128IP アドレス）での処理時間観測波形

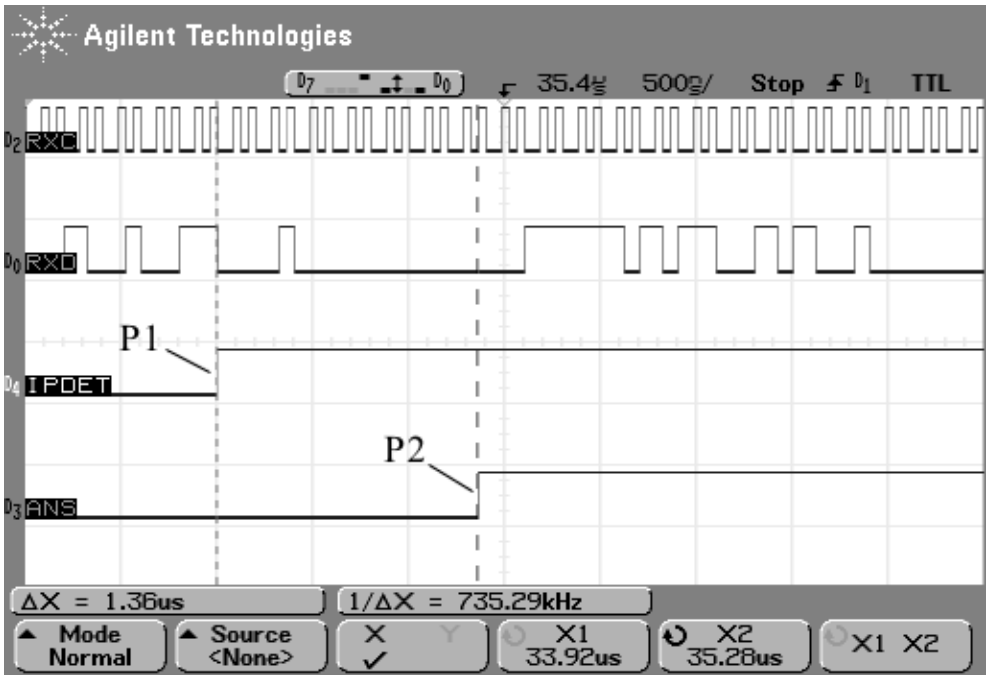


図 4.5　混在型構成（2,048IP アドレス）での処理時間観測波形

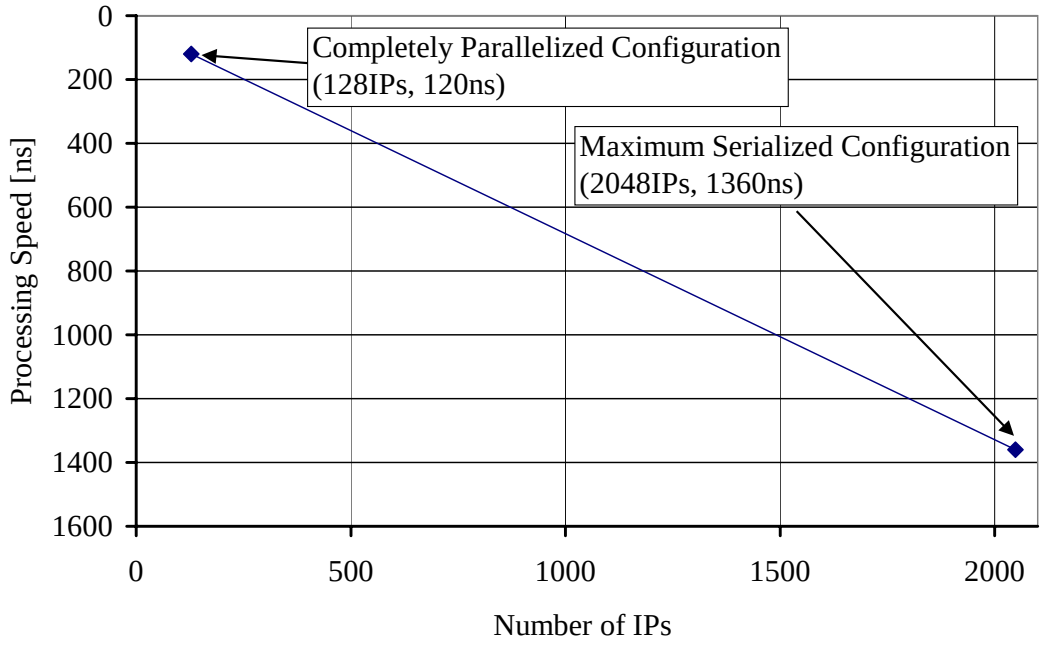


図 4.6 XILINX XCV300E に IP フィルタを実装した場合の処理速度と IP アドレス数の関係

ドレス処理可能な場合の処理時間は 1,360ns **であった．これはそれぞれ約** 4.27Gbps**，**377Mbps **のデータ伝送速度で** *On-The-Fly* **フィルタ処理を実現できることを示している．**

定性的に，

現在一般にエンドユーザで用いられている Ethernet NIC(Network Interface Card) **は** 100BASE–TX **から** 1000BASE–TX(TIE/EIA–854) **に置き換わりつつある．**1000BASE–TX **の最大データ伝送速度は** 1Gbps **であることから，**377Mbps**〜**4.27Gbps **の伝送性能はエンドユーザから中流向けの性能であるといえる．このことから，**XCV300E **という中規模の** FPGA **で中流ネットワーク（基幹〜エンドユーザを結ぶネットワーク）向けの** IP **パケット・フィルタ装置が実現できることがわかった．**

第 5 章

汎用 I/O インターフェースへの適用

本章では，これまでの 2 つの適用例のように並列性が非常に高いアルゴリズムの高速化を目的とするものではないアプリケーションとして，I/O インターフェースへの適用について述べる．本適用例は，I/O インターフェースという，コンピュータ・システムなどの一部分に関するものであり，回路サイズの縮小，低コスト化，及びユーザビリティ（利便性）の向上を目的としている．本的用例の目的は，リコンフィギャラブル・アーキテクチャの新たな可能性を探ることと，現在の FPGA で実現可能な I/O インターフェースを調べることである．本章では，

1. 標準規格化された I/O インターフェース，
2. 提案手法であるメタ・I/O インターフェース，
3. ユーザビリティの向上を図る提案手法 SID(Self-Informational Device)，
4. 実装評価，

の順で述べる．

5.1 標準規格化された I/O インターフェース

現在，パーソナル・コンピュータの I/O インターフェースには，EIA–232，PS/2，USB[76]，Ethernet などさまざまなものがある．これらの I/O インターフェースは，メーカ，或いは標準化団体の先導により作られた，“標準規格” であり，コンピュータ・メーカやその周辺装置を開発しているメーカは，この “標準規格” に準じて製品開発を行っている．このように，I/O インターフェースが統一されているために，コンピュータやその周辺装置は，メーカ毎に内部の回路構成が異なっていても，相互に接続が可能となっている．また，携帯電話，PDA などの携帯端末装置はまだ，パーソナル・コンピュータほど “標準規格” 化が進んでいないが，各メーカ独自規格の I/O インターフェースの中で，EIA–232 互換のプロトコルが使用されていたり，

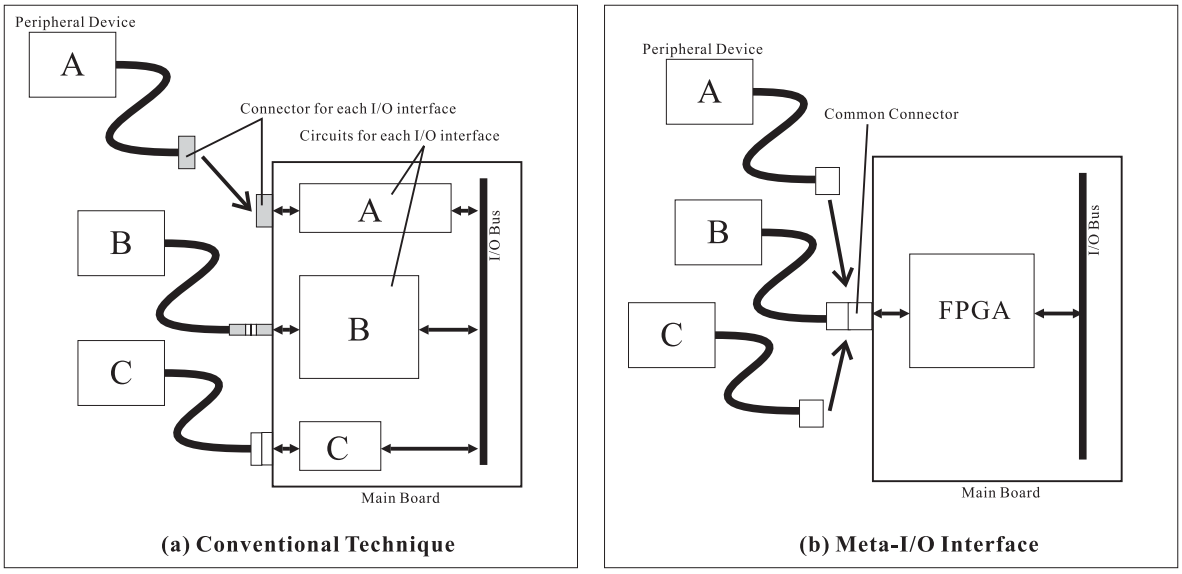


図 5.1　メタ・I/O インターフェースの実装形態

USB を搭載する携帯電話が製品化されるなど，やはり “標準規格” 化の方向へ進んでいる．

しかし一方で，この “標準規格” はさまざまな利用を考慮して，余裕を持った仕様となっているため，各周辺装置に最適な仕様となっていないことが多い．ある周辺装置にはオーバ・スペックであるにもかかわらず，“標準規格” に合わせるために余計な構造や部品が必要となり，結果として周辺装置のサイズ，消費電力，コストなどを上げてしまう．また，“標準規格” に合わせて周辺装置の開発を行うことは，メーカの開発期間，開発費用なども余計に費やすことになる．これは主に，“標準規格” で用いられるプロトコルが汎用性を高めるために煩雑であることが原因であるが，これは周辺装置の実行性能自体を低下させる危険性もある．

従って，各メーカが，各製品毎に異なる要求性能に合う独自のプロトコルを用いた “非標準規格” の I/O インターフェースでありながら，メーカを問わず相互接続可能な I/O インターフェースが実現できれば，より短期間，かつ低価格で製品開発が可能となるばかりか，より小型，省電力で高性能な製品の実現が可能となるはずである．そこで，現在市販されている FPGA などの一般的な再構成可能集積回路を用いて，高い互換性が要求される I/O インターフェース上で独自プロトコルの利用を可能にする，メタ I/O インターフェースを提案する．また，実際にキーボード入力装置に適用した例を用いて，現在一般的に用いられている “標準規格” の I/O インターフェースとの比較評価を行い，提案手法の有用性を示す．

5.2 メタ・I/O インターフェース

高い互換性が要求される I/O インターフェース上で，独自プロトコルの利用を可能にする新しい I/O インターフェースを実現するため，FPGA を用いた I/O インターフェースである，メタ・I/O インターフェースを提案する．図 5.1 にコンピュータの主基板とその周辺装置を例にした，従来方式（図 5.1(a)）と，提案するメタ・I/O インターフェースの実装形態（図 5.1(b)）を示す．現在，コンピュータ側の主基板と各周辺装置は共に，標準規格の I/O インターフェースを実現するため，その内部にそれぞれの専用のインターフェース回路を実装している（図 5.1(a)）．このため，コンピュータの主基板と周辺装置間を結ぶケーブル中は標準規格に準じた信号が流れ，同一規格であれば周辺装置のメーカ，種類が異なっても，相互に接続が可能となっている．一方，提案手法（図 5.1(b)）では，コンピュータ側の主基板に搭載された再構成可能集積回路（FPGA）に，接続される各周辺装置に応じて最適なインターフェース回路を実現することで，周辺装置側に実装するインターフェース回路を極力抑えることができる．また，コンピュータの主基板側のインターフェース回路を再構成可能集積回路（FPGA）によって実現することで，同一のハードウェア・リソースを用いて，さまざまな種類の I/O インターフェースを実現することが可能となる．これは，単にあるときはキーボード・インターフェースの回路を，あるときはディスプレイ・インターフェースの回路を再構成可能集積回路上に実現して対応するというものではなく，周辺装置の内部で処理する必要のあったものを，コンピュータの主基板側で処理することを可能にするものであり，コンピュータの主基板と周辺装置のインターフェース（境界）をどこにするかを，状況に応じて自由に選択可能にするものである（図 5.2）．この境界の適切な位置は周辺装置により細かく異なるため，これを可動とすることによって，各周辺装置に最適な I/O インターフェースを実現することが可能となる．その結果として，周辺装置のサイズ，消費電力，製造コストをより小さくすることができ，より多様な形体の製品が実現可能となる．

また開発においても，“標準規格” という制約から開放されることにより，各周辺装置のメーカは装置毎に最適な，独自のプロトコルを利用可能となる．一方，コンピュータ・メーカは従来のように多種の I/O インターフェース用チップを実装する必要がなく，一つの再構成可能集積回路を実装するだけで済む．さらにその再構成可能集積回路の内部回路設計は各周辺装置メーカによって行われるため，コンピュータ・メーカはその内部設計について検証する必要がない．このことは互いの開発コストの低減につながる．また，周辺装置の制御回路が可能な限り再構成可能集積回路に実装され，再構成可能な部分が増えることで，製品出荷後のディバグが容易となるばかりでなく，新たな I/O インターフェース方式やアプリケーションへの対応を即時行うことが可能となる．

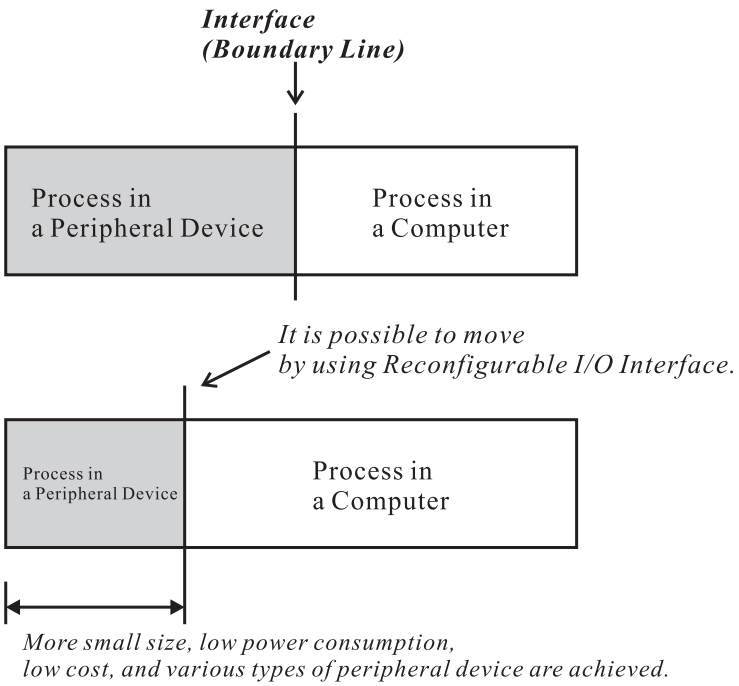


図 5.2　メタ I/O インターフェースの概要 1

ただし，“標準規格” に捉われないといっても，コネクタの形状や電気的特性などは可変ではないため統一する必要がある．また再構成可能集積回路の接続構成も統一する必要がある．これを従来の USB と比較すると図 5.3 のようになる．USB も 1 つの共有コネクタで異なる種類の周辺装置の接続が可能であるが，USB の場合上位のドライバ・ソフトウェアは書き換え可能であっても，ハードウェアのプロトコルなどは固定である．これはそのプロトコルの仕様が周辺装置の要求仕様の範囲内であれば接続可能となるが，逆にオーバー・スペックとなっている場合もあり，決して各周辺装置に最適なものではない．一方，メタ・I/O インターフェースの場合は，前述したコネクタや再構成可能集積回路などの物理的特性を除く，ハードウェア自体が書き換え可能であるため，接続される装置に応じて最適な回路，プロトコルを実現可能となる．

図 5.2 をより具体的に，4 つの処理 (Process1〜4) を行う 2 つの装置 (device1，2) 間のインターフェースにおいて，従来の実装形体に，メタ I/O インターフェースを適用する例を図 5.4，5.5 に示す．図 5.4 中の Step0 が従来の実装形体を示しており，Process1〜4 までの処理のうち，Process1，2 が装置 1(device1) にて，Process3，4 が装置 2(Device2) にて処理されるものとする．各処理間には次の処理へデータを受け渡すためのプロトコル (Protocol A〜C) が存在し，例では装置 1 と 2 の境界（インターフェース）が Process2 と 3 の間となっている

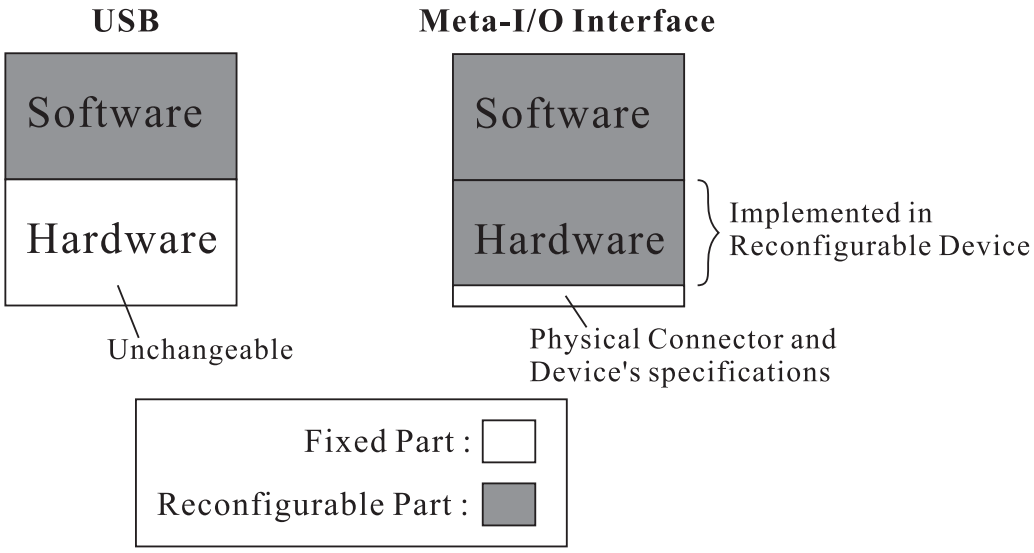


図 5.3　USB との比較

ため，境界におけるプロトコルは Protocol B となっている．また従来の実装形体では，この Protocol B を標準規格化することで，装置の種類，あるいはメーカなどが異なっていても相互に接続が可能となっている．このため，Protocol B を処理する Process2，3 の回路も，標準規格化されたプロトコルを実現するための専用集積回路となっている．ここで，Step1 に示すように，Process3 を行っていた専用集積回路を FPGA などの再構成可能集積回路に置き換えると，装置 2 の I/O インターフェースは Protocol B 以外のさまざまなプロトコルに対応可能となるため，結果としてプロトコルの異なるさまざまな装置 1 に同一の I/O インターフェースで対応可能となる．

このとき，装置 2 が Protocol A を扱うように再構成可能集積回路を書き換えると，図 5.5 の Step2 に示すように，境界（インターフェース）自体を移動することができる．これは，Process 2 を装置 2 へ取り込むことによるが，これはさらにほとんどの回路において Step3 のように Process 2 と 3 を最適化した Process 2’ とすることができ，結果として装置 1，2 共に回路サイズの縮小，省電力化，低コスト化が可能となる．

5.3　携帯端末装置向けの実装

図 5.1(b) に示すように，一方の装置（図 5.1(b) では周辺装置側）の回路を削減し，他方（図 5.1(b) ではコンピュータ側）の再構成可能集積回路に最適化した回路を実装するためには，ま

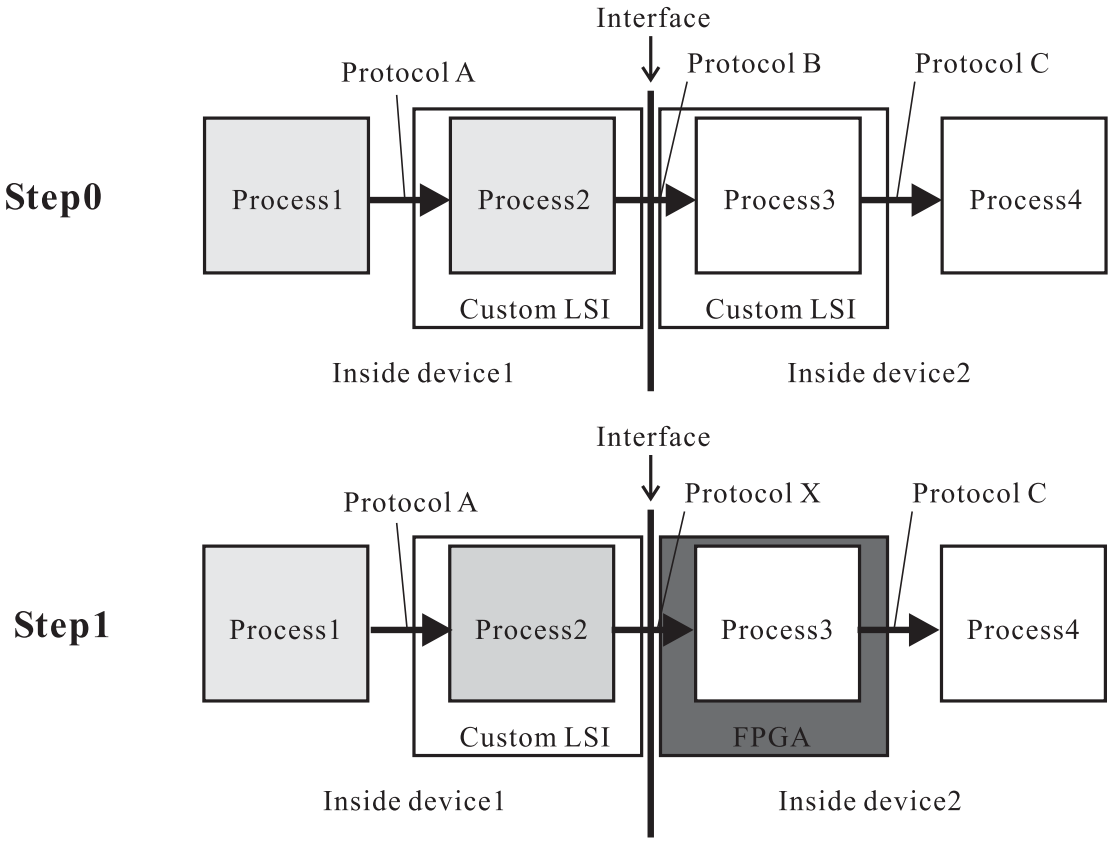


図 5.4　メタ I/O インターフェースの概要 2

ず一方の装置内の処理を他方の装置内の再構成可能集積回路で代行できる必要がある．そして装置間の物理的な接続構成（配線のピン数，電気的特性など）により，図 5.2 に示す境界線の可動範囲が制約を受け，実現可能な I/O インターフェースが決定される．そこで本研究ではまず，現在一般的に利用されている携帯端末などのコネクタの仕様（表 5.1）にて，市販の再構成可能集積回路を用いて実現可能な “非標準規格” の I/O インターフェースを対象とした．具体的には，キーボード入力装置の I/O インターフェースにおいて既存の “標準規格” の I/O インターフェースと，数種類の独自構成のインターフェースとの比較評価を行った．

5.3.1　キーボード入力インターフェース

まず，キーボード入力装置の一般的な “標準規格”I/O インターフェースとしては PS/2 や USB などがある．これらは共にシリアル通信方式であり，最も少ない配線数で多くのキースイッチの状態を伝送可能である．しかし，キースイッチの数が少ない場合はより単純な通信方式を採用することで，キーボード，端末共により小型，省電力，低コスト化が可能である．最

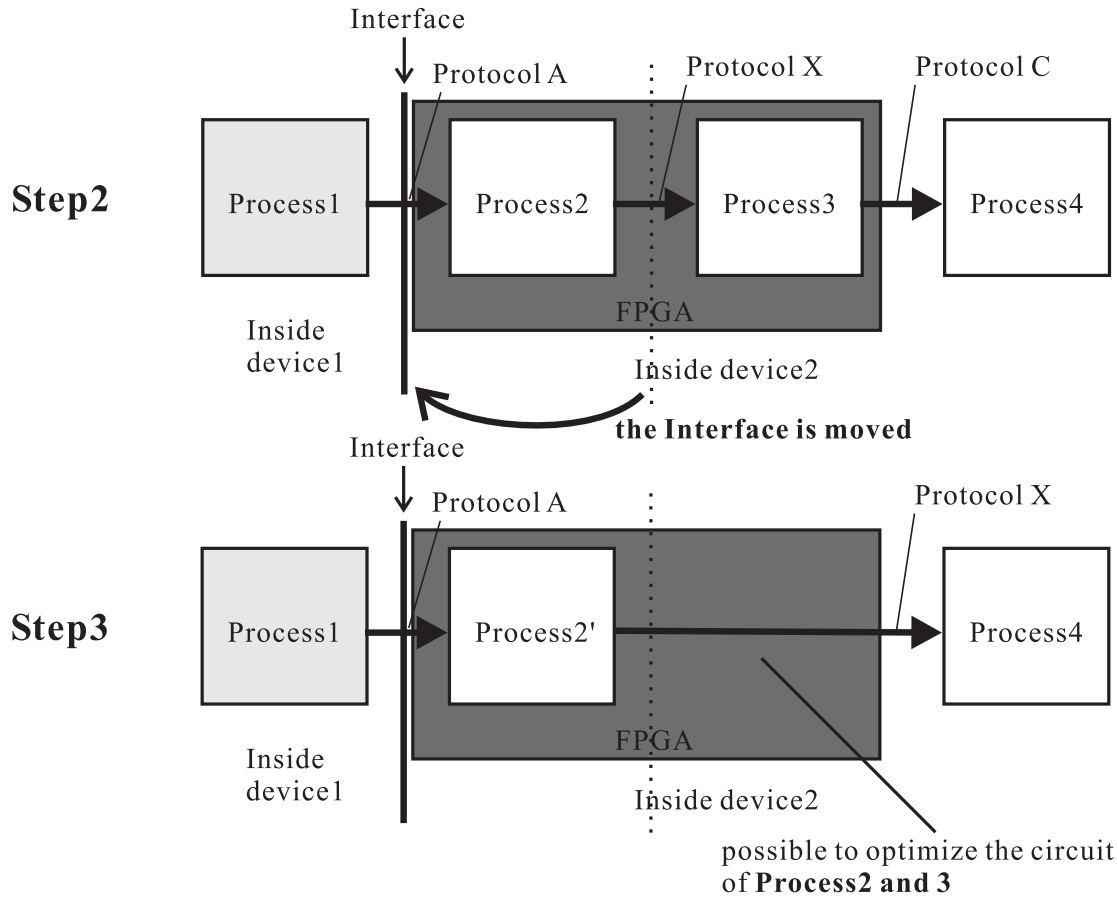


図 5.5　メタ I/O インターフェースの概要 3

も単純な方式は，各キースイッチの出力をそのまま伝送線に配線したものであるが，この場合表 5.1 に示すような十数ピン程度では十数個のキーが最大となる．これは丁度，現在製品化されている携帯電話などのキー数である．次に，キースイッチをマトリクス配線した場合について考える．マトリクス配線にはキースイッチの数だけダイオードが必要になるが，この場合，例えば 10 ピンの I/O で，$5 \times 5 = 25$ 個，13 ピンで $7 \times 6 = 42$ 個，16 ピンで $8 \times 8 = 64$ 個のキーが実現できる．さらにキー数が必要となる場合，あるいはより少ないピン数で実現する必要がある場合には，シリアライザなどの半導体部品を追加する必要があるが，それでも現在の PS/2 や USB インターフェースに比べれば遥かに簡単で小規模な回路構成である．ただし，これら “非標準規格” の I/O インターフェースを用いるためには，再構成可能集積回路へ書き込む回路情報を誰がどのように用意するかという問題がある．ユーザにとって煩雑とならないような仕組みがないと，今までよりも使い難いものとなってしまう．この解決手法については後で論じる．

図 5.6 にメタ・I/O インターフェースによるキーボード入力装置インターフェースの実装例

表 5.1　市販製品の外部インターフェース・コネクタ仕様

種類	製品名	ピン数	I/O インターフェース
PDA	Palm Pilot, III, VII, V, m100, m105	10	TIA/EIA-562
PDA	Palm m500, m505, m515, m125, m130, i705	16	TIA/EIA-232, USB
PDA	SONY CLIE PEG-N, S	13	TIA/EIA-232, USB
PDA	SONY CLIE PEG-NX, NZ, TG, NR, T, SJ, SL	18	TIA/EIA-232, USB
携帯電話	PDC 方式ディジタル携帯電話機標準	16	シリアル・データ, RF 音声信号用 I/O
携帯電話	CdmaOne 方式携帯電話機標準	18	シリアル・データ, RF 音声信号用 I/O
携帯電話	IMT-2000 携帯電話用コネクタ A	10	USB, RF

を示す．図 5.6 はキーボード側にシリアライザを，PC 側にその受信回路と MPU から読み出されるバッファを持った構成例である．従来の手法（PS/2 や USB など）では図中の I/O インターフェース回路は専用 LSI や MPU などで実現されている．メタ・I/O インターフェースでは，このうち PC 側の I/O インターフェース回路を再構成可能集積回路で実現する．これは図 5.4 の Step1 に該当する．回路の動作はまず，キーボード入力装置側の各キースイッチの状態を，制御回路 1 によりマルチプレクサを通じてバッファ 1 に蓄える．これを PC 側の制御回路が逐次読み出し，PC 側のバッファ 2 に蓄える．バッファ 2 に蓄えられたキーの情報を MPU が読み出し，OS，或いはアプリケーション・ソフトウェアなど上位のソフトウェアで利用する．この構成にすることで，PC 側の I/O インターフェース回路を自由に書き換え可能となるので，I/O インターフェースの物理的特性（I/O のピン数や信号線の電気的特性など）の範囲内であれば，シリアライザの種類（クロック同期式，調歩同期式，ビット数，クロック周波数，バッファ容量など）を自由に変更できる．しかし，このような回路構成の場合はさらに，図 5.7 のようにインターフェースをバッファ 1 と 2 の間から，キースイッチ群とシリアライザの間へ移動することができる．このインターフェースの移動により，キーボード入力装置側はキースイッチの直接配線，もしくはマトリックス構成とすることで LSI などの高価な部品を一切用いずに実現することができる．また PC 側もキーボード側と PC 側にそれぞれ実装されていた回路を合わせて最適化できるため，結果としてキーボード入力装置，PC 共に回路の小規模化，省電力化，低コスト化が可能となる．なお，これは図 5.5 の Step2，3 に該当する．

5.3.2　試作基板

実際に PC 側の再構成可能集積回路として XILINX 社の FPGA である，XC2S50E[69] を用いた試作基板を作成した（図 5.8）．試作基板には 3 つの共有コネクタが搭載されているが，

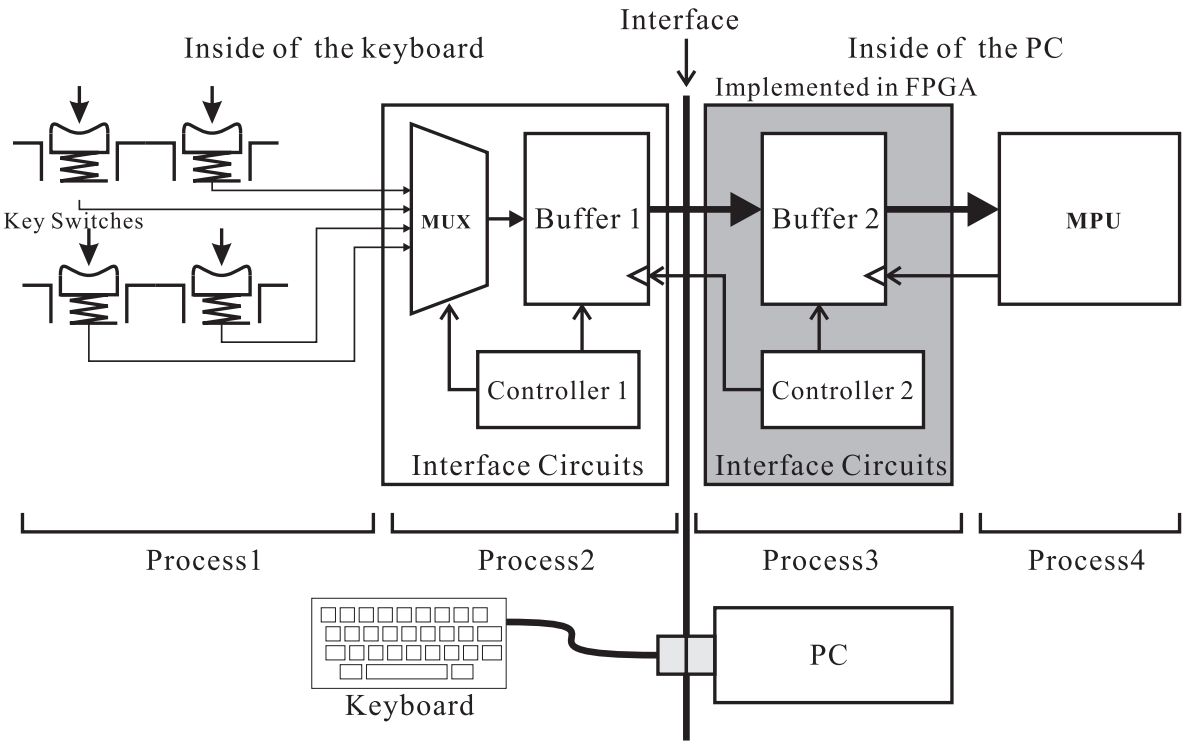


図 5.6　キーボード入力装置と情報処理装置間への実装例 1

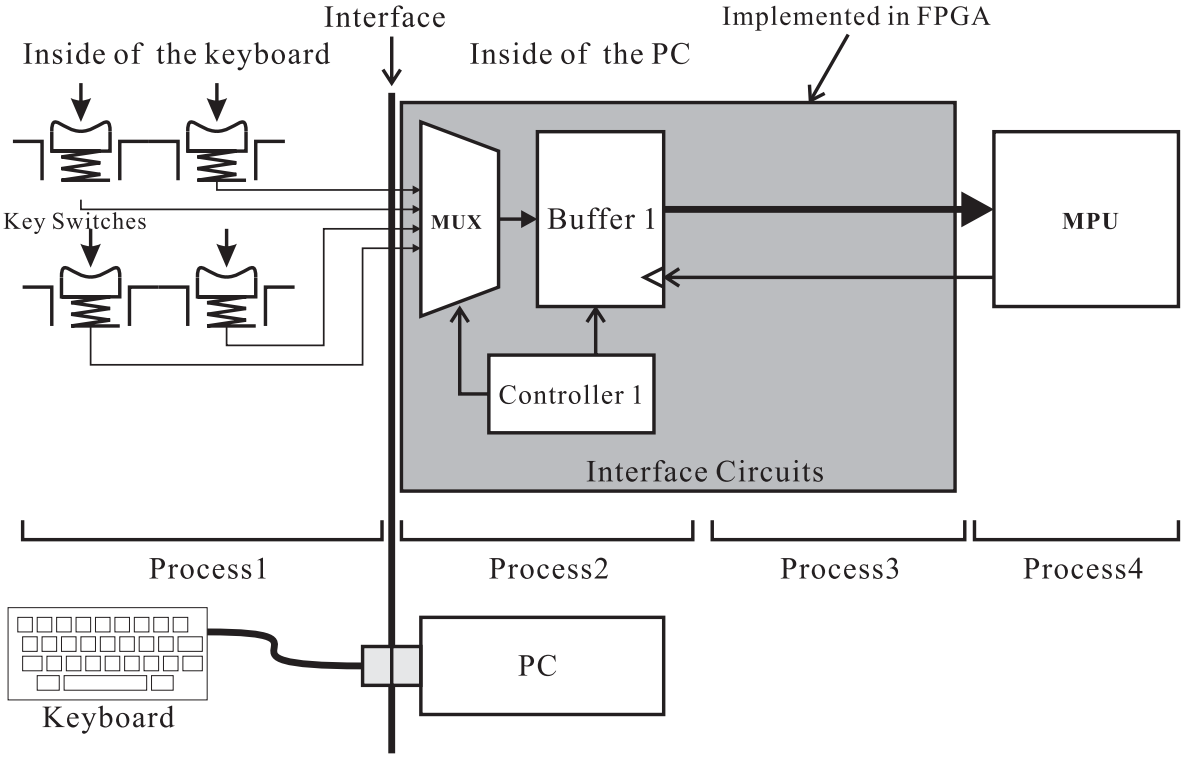


図 5.7　キーボード入力装置と情報処理装置間への実装例 2

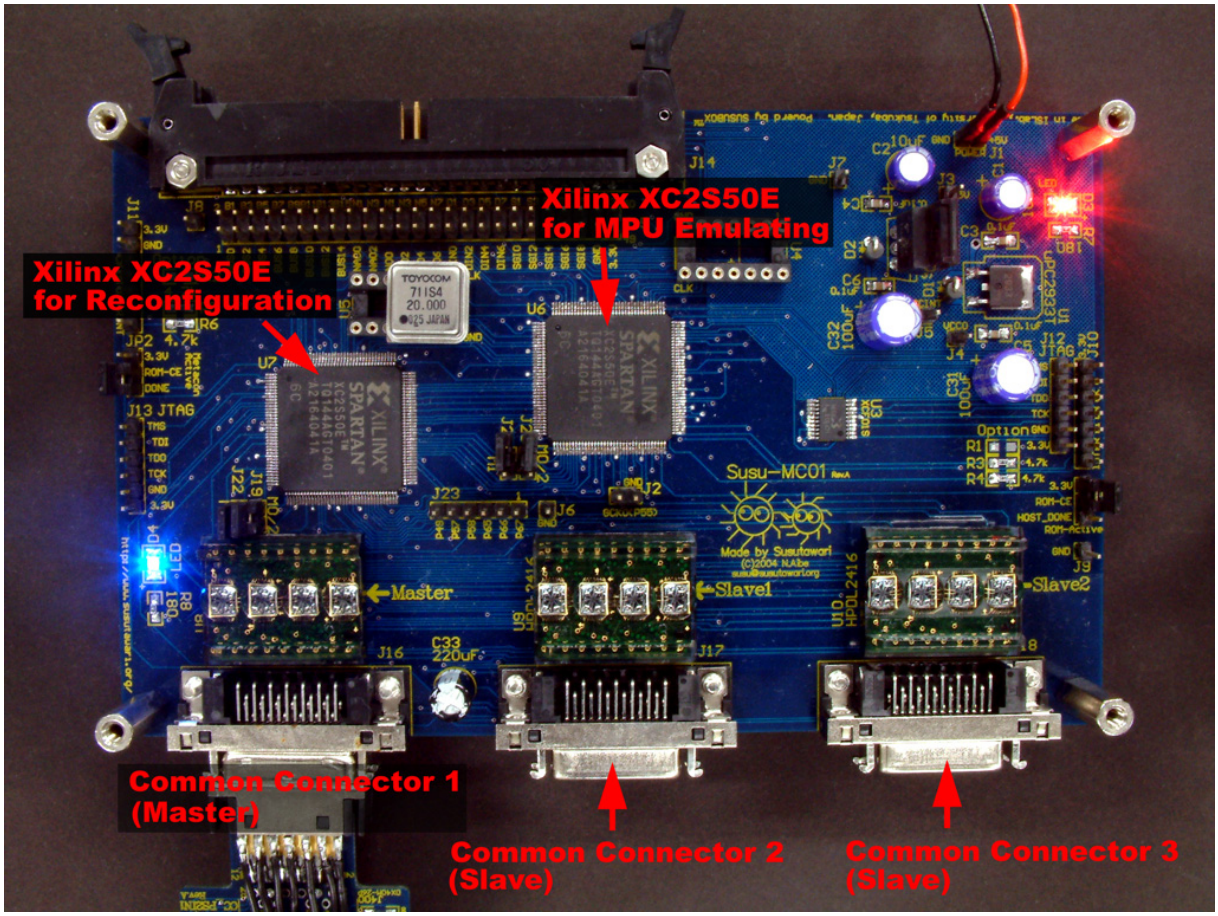


図 5.8　メタ・I/O インターフェースの試作基板

このうちの一つ（Common Conector1(Master)）にのみ FPGA のコンフィギュレーション・ポートが接続されており，ここに接続された SID の回路構成情報のみがロードされる．他の 2 つの共有コネクタ（Common Connector2，3）は同時に複数の周辺装置を利用するためのもので，ここには SID を接続してもその回路構成情報は読み込まれない．また，試作基板に搭載された FPGA は 2 つあり，SID によりインターフェース回路が構築されるものの他に，MPU などの固定された回路との動作検証を行うため，これらをエミュレートする FPGA がある．

5.4　さまざまな I/O インターフェースの回路規模

メタ・I/O インターフェースでは，コネクタの I/O 数や電気的特性といった制約の範囲内であれば，未知の I/O インターフェースに対応できる．しかし，利用したい複数の未知の I/O インターフェース間で，その回路規模に大きな差があると，コンピュータ側に搭載しておく FPGA は余裕を見て大きな回路規模のものである必要があり，これは結果的にコストを上げる

表 5.2　5種類のI/Oインターフェース回路の実装結果

I/O Interface	SLICEs	XC2S15 [%] (192 SLICEs)	XC2S30 [%] (432 SLICEs)	XC2S50 [%] (768 SLICEs)
PS/2 Keyboard Decoder	21	10	4	2
PS/2 Keyboard Decoder with Key-Code Decoder	100	52	23	13
I^2C Drv. with RAMs	51	26	11	6
Character LCD (SC2004C) Driver	109	56	25	14
B/W NTSC Enc.	64	33	14	8

ことに繋がる．従って，現実的に利用されるI/Oインターフェースがどのくらいの回路規模を要求し，また異なるI/Oインターフェース間で，その回路規模にどの程度の差があるかを調べる必要がある．厳密には，さらにI/Oインターフェースの種類ごとに回路の性質を調べる必要があるが，本研究ではまずFPGAで実現可能な，一般的なI/Oインターフェース回路をいくつか実装し，その回路規模と必要なFPGAの規模を求めた．PS/2キーボード・デコーダ回路，およびキーコードのデコード回路，I^2Cドライバ回路とI^2Cデータを記憶するRAM，キャラクタ・LCDディスプレイ・ドライバ回路，モノクロNTSCエンコーダ回路，の合計5種類のI/Oインターフェース回路を，3種類のFPGAに実装した結果を表5.2に示す．FPGAにはXILINX XC2S15，XC2S30，XC2S50[68]を用いた．この結果，最も差が大きかったのは，PS/2キーボード・デコーダ回路単体と，キャラクタ・LCDディスプレイ・ドライバ回路間で，その差は88 SLICEであった．しかし，PS/2キーボード・デコーダは通常，キーコード・デコーダ回路も併用されるため，この場合はその差は9 SLICEと少なく，FPGAの回路使用効率が高くなる．また今回の5種類のI/Oインターフェースは全てXC2S15という最小規模のFPGAに実装可能で，要求回路規模は50〜100 SLICE程度であることがわかった．ただし，本結果からは入出力の種類の違いと回路規模の間の相関関係は特に得られなかった．より多種のI/Oインターフェース回路で比較を行う必要があるが，この程度の小規模であれば，将来的に現在のMPUの一部に100 SLICE程度の再構成機構（LUTアレイ）を組み込むことで，コンシューマ向けのコンピュータ・システムにメタ・I/Oインターフェースを組み込める見通しが得られた．

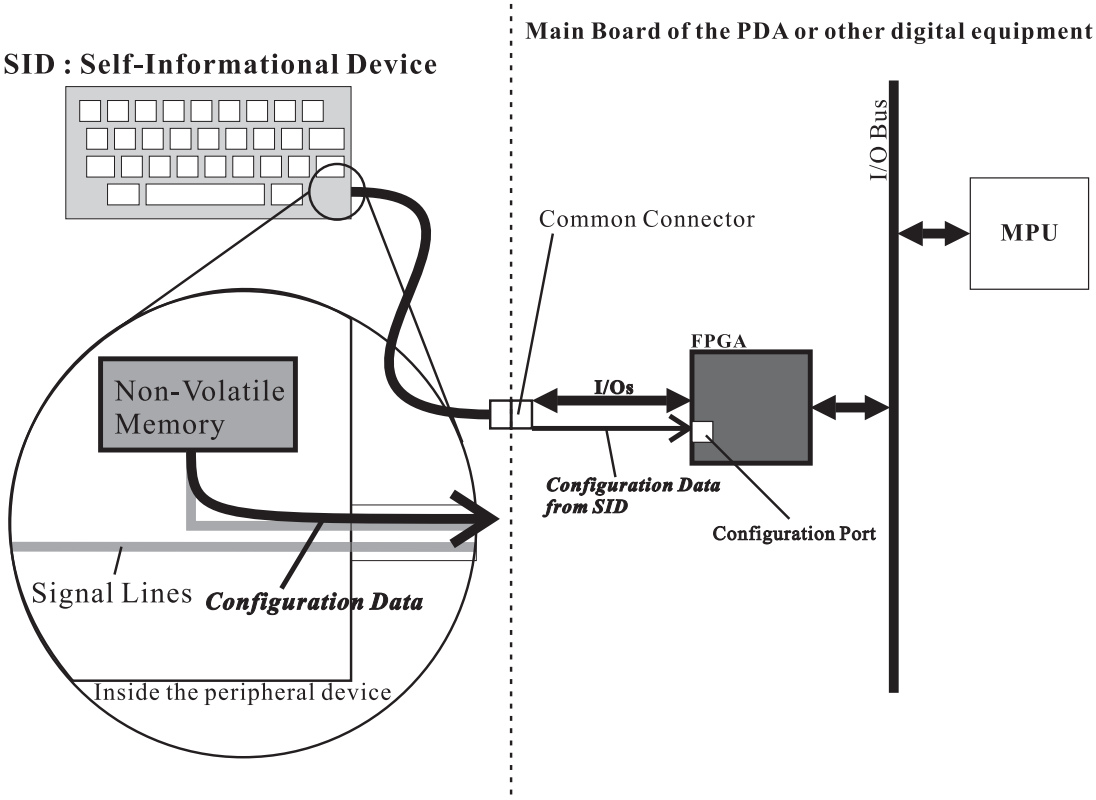


図 5.9　SID とメタ・I/O インターフェースの関係

5.5　SID (Self-Informational Device)

提案するメタ・I/O インターフェースは一方の装置に再構成可能集積回路を持ち，主要な処理をこの再構成可能集積回路上で行うことで，全体の性能向上を図るものである．従って，例え周辺装置メーカが独自に開発したインターフェースであっても，その処理を行う再構成可能集積回路の回路構成情報をターゲット側にロードさえすれば，その時点で装置間に独自の “非標準規格”I/O インターフェースが実現される．これは，前節で示したように，各装置の用途や役割に応じて最適な I/O インターフェースを実現するが，さらに各メーカの開発した I/O インターフェースを “非公開” にすることも可能とする．また，再構成可能集積回路は何度でも書き換え可能であるため，I/O インターフェースのディバグ，性能向上，機能の追加なども容易に行うことが可能である．これらを実現するための再構成可能集積回路の回路構成情報は，各装置がインターネットなどに接続されていれば，各装置メーカの Web サイトからダウンロードし，ユーザが書き換えるような手法も考えられるが，ここではさらにユーザの利便性を高め

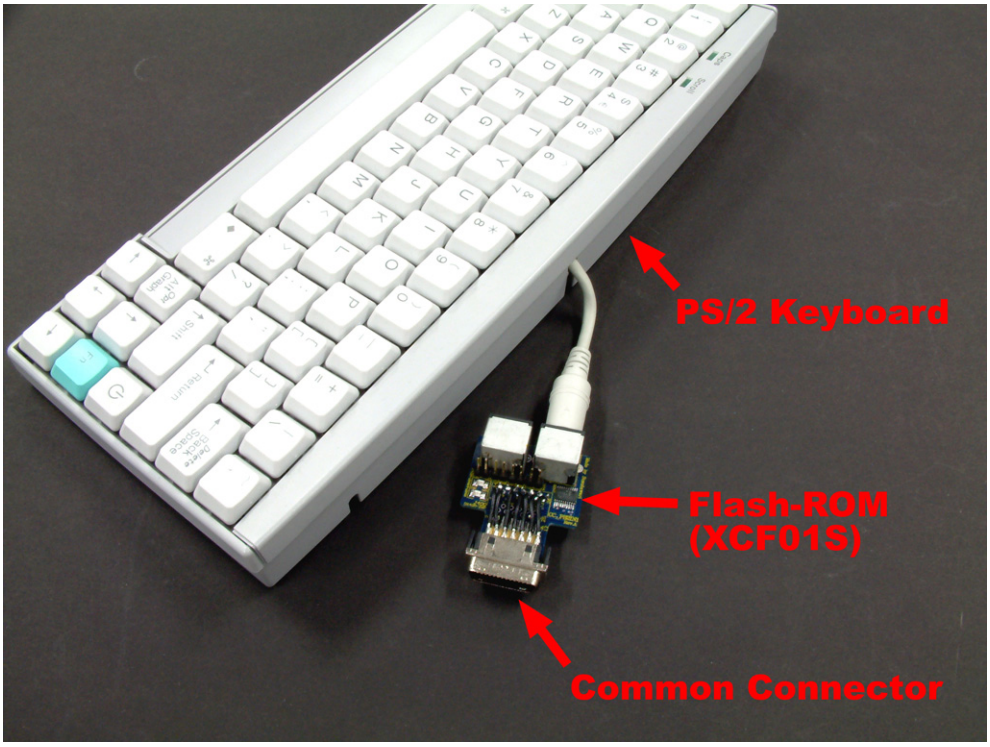


図 5.10 SID の試作器（全体）

るための実装手法である，SID (Self-Informational Device) を提案する．SID を用いることでユーザは回路構成情報を全く意識することなく，従来の I/O インターフェースと同様に装置同士を接続するだけで利用可能となる．

図 5.9 に SID とメタ・I/O インターフェースの関係を，図 5.10，5.11 に SID の実装例を示す．図 5.9 で，中央の破線が装置間の境界を表している．破線の右側が PDA などの情報処理装置で，再構成可能集積回路を持っている．左側がキーボード入力装置などの周辺装置で，この周辺装置が SID である．SID は内部に再構成可能集積回路の回路情報を記憶した，不揮発メモリを持っており，これが SID (Self-Informationa Device) と言う名前の由来である．二つの装置は破線上の共有コネクタによって接続される．この共有コネクタには再構成可能集積回路の回路情報を書き込むための専用のピンがあり，SID が接続されると，内蔵の不揮発メモリから回路情報がロードされる．このことにより再構成可能集積回路に各周辺装置 I/O インターフェース用の制御回路が構成され，予め各周辺装置用の I/O インターフェースを持っていない装置で，その周辺装置の利用が可能となる．従って，装置メーカは新しい I/O インターフェースをもつ装置を提供する際，装置にその回路情報を記憶させるだけでよい．また，ユーザは利用したい周辺装置をただ接続するだけでよく，回路構成情報などを全く意識する必要がない．

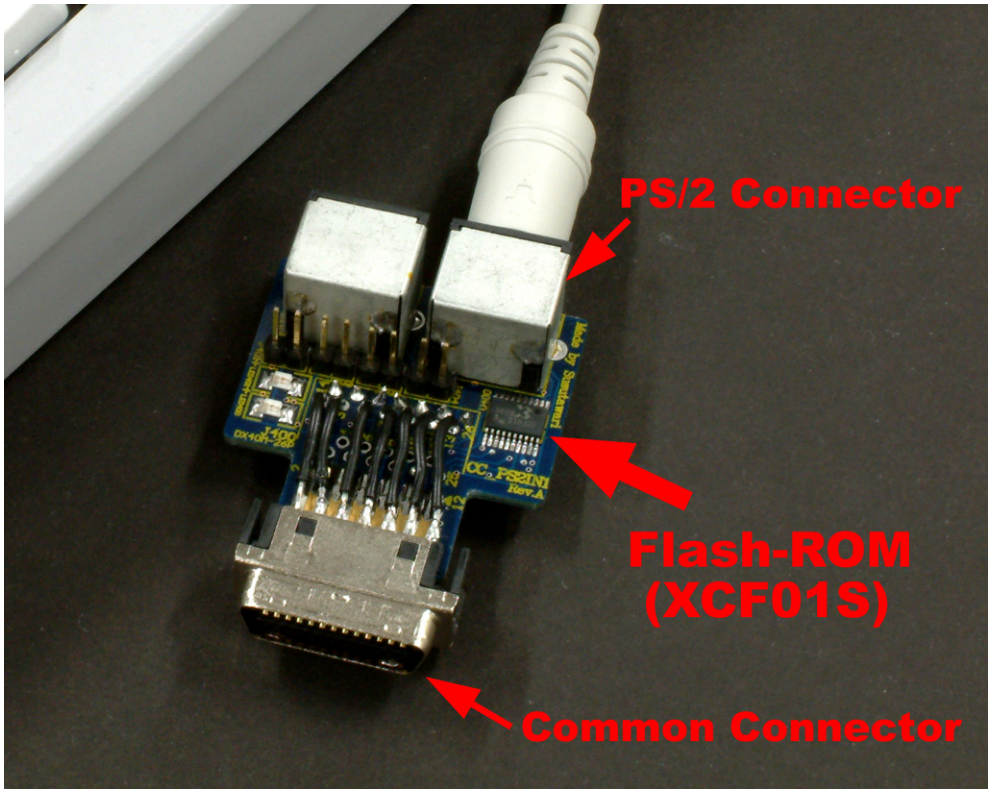


図 5.11　SID の試作器（拡大：共有コネクタ部）

5.6　まとめ

本章では，携帯端末装置やパーソナル・コンピュータと，その周辺装置の接続を今までよりも柔軟にする新しい I/O インターフェースである，メタ・I/O インターフェースを提案した．メタ・I/O インターフェースは FPGA の再構成機構を，現在リコンフィギャラブル・コンピューティングの研究で注目されている演算の高速化以外に活かしたアプリケーションである．また，メタ・I/O インターフェースと組み合わせて使用することで，ユーザに余分な負担を強いらない実装形体である，SID (Self-Informational Device) を提案し，キーボード入力装置を例とした試作機により，その実現性の高さと有効性を示した．以上より，

定量的に，

一般的な *I/O* **インターフェースである，** *PS/2* **キーボード・デコーダとキーコード・デコーダ回路，** I^2C **ドライバとデータ用** *RAM*，**キャラクタ・** *LCD* **・ドライバ，モノクロ** *NTSC* **エンコーダ，それぞれが，**50〜100 SLICE **の回路規模で** *FPGA* **上に実現可能であり，これは最小**

規模の XILINX XC2S15 **に実装できることがわかった．**

定性的に，

1. **１つの共有コネクタで，異なる種類のプロトコルを持つ周辺装置の接続が可能，**
2. **高い相互接続性を維持したまま，“標準規格”外の各周辺装置に最適な独自プロトコルの使用が可能，**
3. **周辺装置の小型化，省電力化，低コスト化を実現し，周辺装置の多様化を実現可能，**
4. **ユーザーが** I/O **インターフェースの種類を意識せずに利用できるインターフェースが実現可能，**

であることがわかった．

第 6 章

新しいハードウェア開発手法への適用

本章では，これまでの 3 つの適用例のように，ハードウェア・アーキテクチャに関する応用ではなく，新しいハードウェアの開発手法という，FPGA の本来のラピッド・プロトタイピング・ツールとしての利用を，さらに発展させたアプリケーションについて述べる．本的用例の目的は，FPGA 本来の，ラピット・プロトタイピング・ツールとしての利用方法の中で，新たな可能性を探ることである．本章では，

1. IP(Intellectual Property) 流通,
2. オープンソース・ソフトウェア,
3. 提案するオープンソース・ハードウェアという概念,
4. 実装評価,

の順で述べる．

6.1 IP 流通

Intellectual Property (IP) は，知的財産（権）と訳せるが，半導体の分野では一般に，集積回路の設計図を中心とした資産を指す．IP には ASIC などのフォト・マスク・ベース LSI を対象としたものと，FPGA や CPLD などの再構成可能集積回路を対象としたものがある．また，その表記方法にも紙面に印刷した回路図から，フォト・マスク， HDL (Hardware Description Language) コード，ネットリスト，半導体メモリに記憶可能なビットストリーム・データまで，さまざまな形式がある．中でも HDL コードやネットリストで記述された IP は ASIC でも FPGA でも利用可能なものがあり，非常に再利用性が高い．さらに最近では，従来のチップメーカのように，これら IP を専門に売買する企業や，仲介する組織が出てきている．このため，電機メーカなども自社の製品に搭載する ASIC を製造，或いは FPGA の内部設計を行

う際，他社が開発した IP を購入して製品に採用するケースが増えてきている．電機メーカにとって，従来の汎用ロジック，或いは ASSP (Application Specific Standard Product) のようなチップ単位ではなく，IP 単位で選定できることの利点には次のようなことが挙げられる．

1. 従来よりさらに用途に適した回路を実現可能
2. 回路の集積による実装の高密度化
3. 部品点数の削減による低コスト化

まず，従来の汎用ロジックや ASSP の中から用途に合うものを選定する場合に比べ，IP 単位で回路を選択できることは，より適切な回路を選択できる可能性が高い．これは一般に IP の方がチップよりも回路の持つ機能の粒度が細かいことが理由であるが，さらに IP の場合は記述形式によっては独自のカスタマイズが可能であることが大きい．また，従来必要な機能を実現するために複数のチップが必要であったものに比べ，IP を用いて単一の ASIC，FPGA に集積することで，回路実装密度の高密度化が可能である．さらにこの集積化は実装の高密度化と同時に部品点数を削減し，低コスト化にもつながる．

一方，チップメーカにとっては，より競争の激化が予想される．これは従来の汎用ロジック/ASSP の初期製造コストに比べ，IP（ソフト・マクロ・コア IP）の開発（製造）は遥かに低価格であり，資金の少ないベンチャー企業などの参入が可能であるからである．さらに，今後は IP ビジネスにおいて，価格競争が進む他，特許，専門性の強化，サポート体制などによる差別化が進み，IP の多様化が期待される．

以上のことから，これからますます高性能で再利用性の高い IP の開発が重要となり，この IP こそが今後の半導体産業の発展の鍵となることだろう．そこで本章では，IP 開発，流通の新しいモデルとして，IP をオープンソース化した「オープンソース・ハードウェア」を提案する．本提案は「オープンソース・ソフトウェア」に深く関係しており，次節ではまず一般的に知られているオープンソース・ソフトウェアについて述べる．

6.2　オープンソース・ソフトウェア

パーソナル・コンピュータの世界では，古くから様々なオープンソース・ソフトウェアがある．著名なものとしては，Linux，BSD，X-Window，emacs，Apache，Perl，GNU C Compiler，OpenOffice など OS からアプリケーション・ソフトウェアに至るまで様々なものがある．これらオープンソース・ソフトウェアは，その名の通りソースコードが公開されているが，さらに特徴的なことは「自由な再頒布」が保障されているということである．これらは利用側，提供（開発）側双方に以下のような利点をもたらす．

利用側

1. 低コスト（ほぼ無料）で利用可能
2. 様々な関連ソフトウェアや派生物が期待できる
3. コミュニティの発生，活性化が期待できる
4. 高い透明性を得られる
5. 構造，動作の理解，改良が容易

提供（開発）側

1. 急速な普及が期待できる
2.（自律的な）アウトソーシングが期待できる
3. 保証・サポートの放棄が可能

まず，利用側にとって一番大きいことは，オープンソース・ソフトウェアはほぼ無料で利用できるということである．ここで,「ほぼ」無料というのは，対象ソフトウェアの価値自体の対価以外は要求することができるためである．つまり，頒布に必要な通信費用，メディア代，輸送費用などは頒布側が請求することができる．

「オープンソース」という言葉に関して，米国の非営利団体 OSI (Open Source Initiative) では次のように定義している．

「オープンソース」とは，単にソースコードが閲覧あるいは入手できるということだけを意味するのではありません．「オープンソース」であるプログラムの頒布条件は，以下の基準を満たしていなければなりません．

1. 自由な再頒布
   「オープンソース」であるライセンス (以下「ライセンス」と略) は，出自の様々なプログラムを集めたソフトウェア頒布物 (ディストリビューション) の 一部として，ソフトウェアを販売あるいは無料で頒布することを制限してはなりません．ライセンスにおいて，このような販売に関して印税やその他の報酬を要求してはなりません．
2. ソースコード
   「オープンソース」であるプログラムはソースコードを含んでいなければならず ，コンパイル済形式と同様にソースコードでの頒布も許可されていなければなりません．何らかの事情でソースコードと共に頒布しない場合には， ソースコードを複製に要するコストとして妥当な額程度の費用で入手できる方法を用意し，それをはっきりと公表しなけ

ればなりません．方法として好ましいのはインターネッ トを通じての無料ダウンロードです．ソースコードは， プログラマがプログラムを変更しやすい形態でなければなりません．意図的にソースコードを分かりにくくすることは許されませんし，プリプロセッサや変換プログラムの出力のような中間形式は認められません．

3. 派生ソフトウェア
   ライセンスは，ソフトウェアの変更と派生ソフトウェアの作成，並びに派生ソフトウェアを元のソフトウェアと同じライセンスの下で頒布することを許可しなければなりません．
4. 作者のソースコードの完全性 (integrity)
   バイナリ構築の際にプログラムを改変するため，ソースコードと一緒に「パッチファイル」を頒布することを認める場合に限り，ライセンスによって改変されたソースコードの頒布を制限することができます．ライセンスは，改変されたソースコードから構築されたソフトウェアの頒布を明確 に許可していなければなりませんが，派生ソフトウェアに元のソフトウェアとは異なる名前やバージョン番号をつけるよう義務付けるのは構いません．
5. 個人やグループに対する差別の禁止
   ライセンスは特定の個人やグループを差別してはなりません．
6. 利用する分野 (fields of endeavor) に対する差別の禁止
   ライセンスはある特定の分野でプログラムを使うことを制限してはなりま せん．例えば，プログラムの企業での使用や，遺伝子研究の分野での使用を制限してはなりません．
7. ライセンスの分配 (distribution)
   プログラムに付与された権利はそのプログラムが再頒布された者全てに等しく認められなければならず，彼らが何らかの追加的ライセンスに同意することを必要としてはなりません．
8. 特定製品でのみ有効なライセンスの禁止
   プログラムに付与された権利は，それがある特定のソフトウェア頒布物の一部であるということに依存するものであってはなりません．プログラムをその頒布物 から取り出したとしても，そのプログラム自身のライセンスの範囲内で使用あるいは頒布される限り，プログラムが再頒布される全ての人々が，元のソフトウェア頒布物において与えられていた権利と同等の権利を有することを保証しなければなりません．
9. 他のソフトウェアを制限するライセンスの禁止
   ライセンスはそのソフトウェアと共に頒布される他のソフトウェアに制限を課してはなりません．例えば，ライセンスは同じ媒体で頒布される他のプログラムが全てオープンソースソフトウェアであることを要求しては なりません．

10. ライセンスは技術中立的でなければならない
    ライセンス中に，特定の技術やインターフェースの様式に強く依存するような規定があってはなりません．

※以上 「オープンソースの定義 八田 真行訳，2004 年 2 月 21 日 バージョン 1.9」[83] より転載．

この定義に当てはまるソフトウェアが，OSI の定義するオープンソース・ソフトウェアであるといえるが，これは直接，法的効力を持つものではない．法的な効力があるのは，各ソフトウェアに適用されているライセンスである．従って，各ソフトウェアのライセンスが「オープンソースの定義」に当てはまるとき，そのソフトウェアはオープンソース・ソフトウェアとしての実用的な価値を持つ．OSI では各ライセンスが「オープンソースの定義」に準拠しているかどうかの承認を行っており，承認されたものには認定マーク (http://www.opensource.org/docs/certification_mark.php) を発行している．

6.3 オープンソース・ハードウェア

パーソナル・コンピュータの上で動作するプログラム，いわゆるソフトウェアはディジタル・データという形で保存されているため，非劣化，容易複製可能，電気伝送可能などの特徴をもつ．一方，コンピュータ本体（MPU，メモリ，I/O とそれらの制御回路を含む）の方はシリコンを主体とした「物質」で構成されたハードウェアであり，その製造，流通はエネルギー面でもコスト面でも，ソフトウェアのようにはいかない．しかしここで，MPU，メモリ，I/O とそれらの制御回路を全て FPGA で構成し，FPGA 上にコンピュータを構築すると，もはやコンピュータ本体はハードウェアではなく，FPGA に実装されるソフトウェアに過ぎない．そして先に挙げたソフトウェアの非劣化，容易複製可能，電気伝送可能などの特徴を適用できるのである．

オープンソース・ハードウェアとは，前節のオープンソース・ソフトウェアの行ってきたことを FPGA という媒体を通して実現しようというものである．厳密には FPGA がハードウェアであり，その回路情報はソフトウェアであるので，その回路情報を公開，配布することはオープンソース・ソフトウェアそのものであるが，狭義の意味ではハードウェアという言葉がコンピュータ本体を指すことから，オープンソース・ハードウェアとする方が一般に意味を通しやすい．ただし，これは FPGA 上にコンピュータを実現する場合の話で，それ以外の専用回路などを実装する場合には適切でない．また，この “オープンソース・ハードウェア” という言葉は，既に IBM PC 互換機などのように，ハードウェア・アーキテクチャがオープンであるという意味で用いられている．このようなことから，“オープンソース・IP” の方が正確である．

従って、本文ではこのように，ハードウェアをオープン・ソフトウェア化したものを，オープンソース・IP と呼ぶことにする．

このようなオープンソース・IP は，既にいくつかの Web サイトで公開されている [84]〜[86]．しかし，これらの IP は現在，まだあまり普及していない．その理由には次のようなことが考えられる．

1. ドキュメントの不足による導入のし難さ，
2. 専門性の高さ，
3. 信頼性の低さ, 無保証，

などである．

現在のオープンソース・IP のほとんどは，ただ回路情報のファイルやソースコードが配信されているだけで，その詳細な動作の解説やアプリケーション・ノート（使用例）を添付しているものがほとんどない．しかし，実際にユーザが IP を利用するには ASIC/ASSP などと同等の，十分なデータ・シートや，アプリケーション・ノートの整備が必要不可欠である．これらがあまり整わないのは，回路の開発以上に労力を要することが多いためと考えられる．また，2. の専門性の高さに関しては，2 つの意味がある．まず公開されている IP がオリジナルの MPU であるなど，その適用範囲が狭い場合，アーキテクチャなどが参考になっても，そのもの自体はあまり実用的でないことが多い．また，一般的なインターフェース回路などでも，これから FPGA，或いはディジタル回路を学ぼうとしている学生などには敷居が高過ぎる場合である．これは 1. のドキュメント整備に加え，容易に利用可能なプラットフォーム（開発環境）を整えることが必要である．そして，3. の信頼性や保証に関しては，特に企業などがオープンソース・IP を利用することを困難にしている要因と考えられるが，これも 1. のドキュメント整備によりある程度対処できるものと考えられる．そこで，ドキュメンテーションの整備や，充実したアプリケーションの開発を行うことを前提とした，オープンソース・IP の開発を試みた．

6.4 実装評価：オープンソース・IP ライブラリ SUSUBOX

実際に FPGA 用のオープンソース・IP をいくつか開発し，インターネット上の Web サイト（http://www.susubox.org/）にて配信を行った．開発したオープンソース・IP ライブラリ「SUSUBOX」を表 6.1 に示す．これらのオープンソース・IP は，前章までの適用例の実装でも用いている．対象としている FPGA は，XILINX 社製の Virtex シリーズ，Spartan–II シリーズなどが主なものである．設計は全て Mentor Graphics(Innoveda, 旧 ViewLogic) 社の ViewDraw と呼ばれる回路図入力（CAD）で行い，2005 年現在公開している形式はその CAD 用のソースコード (wir, sch, sym ファイル形式)，EDIF 2.0 ネットリスト形式，及び PDF 形

表 6.1 開発したオープンソース・IP

種類	IP 名
PS/2 キーボード デコーダ	SPS2KEY_DEC*
モノクロ NTSC デコーダ	SNTSC_DEC*
モノクロ NTSC エンコーダ	SNTSC_ENC*
キャラクタ LCD ドライバ	SC2004C_*
RS-232C トランシーバ	S232C_*
I2C ドライバ	SI2C_*
Ethernet_II デコーダ	SETHER_*
基本モジュール集	SBASE_*

式の回路図の 3 形式である．現在既に配信中のオープンソース・IP の一部のドキュメントを付録 B に添付する．

また，本オープンソース・IP の普及を図る目的で，図 6.1，及び表 6.2 に示す開発プラットフォーム SUSUBOARD を開発した．SUSUBOX は現在，XILINX 社製の FPGA であれば，そのプラットフォームを選ばず，ユーザーサイドにある，あらゆる FPGA に実装可能である．しかし I/O インターフェースを実現するためには FPGA 以外に周辺の LSI が必要となる．SUSUBOARD は現在開発中の SUSUBOX で必要となる周辺 LSI を搭載したもので，SUSUBOX を利用したアプリケーションの開発をより容易にするものである．

6.4.1 適用ライセンス

開発した IP をオープンソース・IP とするためには，法的に有効なライセンスを適用しなければならない．オープンソース・IP は前節で述べたように，その実態はソフトウェアであることに違いないため，オープンソース・ソフトウェアのライセンスを適用することが可能であると考えられる．そこで，下記の 3 つの著名なオープンソース・ソフトウェアのライセンスを考察する．

1. GNU GPL ライセンス
2. GNU LGPL ライセンス
3. 修正版 BSD ライセンス

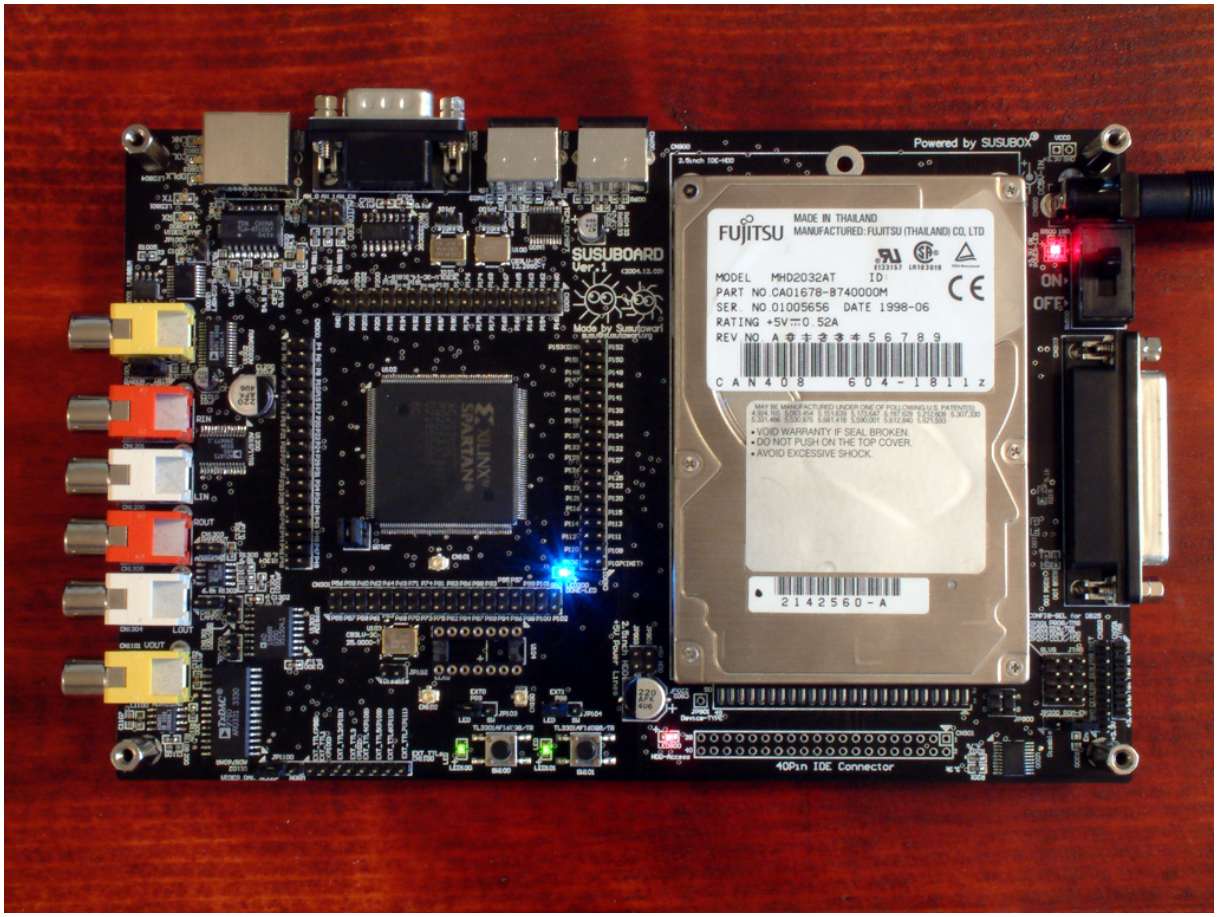


図 6.1　アプリケーション開発プラットフォーム SUSUBOARD Ver.1

まず，GNU GPL のライセンスは，コンピュータ上で実行されるソフトウェア・プログラムに適用することを念頭に書かれており，ユーザにそのプログラムの

1. バージョンの同一性の保証,
2. 無償／有償を問わない再配布の自由の保証,
3. ソースの公開の保証,

をする．中でも，ソース公開の保証というのが特徴的で，これによりユーザはそのプログラムを勉強したり，独自の改良を施すことができる．また，ソースが理解できることでそのプログラムに一定の信頼をおくことができる．さらに一歩進んで，ユーザがそのプログラムに改良を加えて配布するということが繰り返し行われれば，最初のプログラムが多くのユーザ（既に開発者）の手によって，よりよいプログラムへと進化（発展）して行く可能性がある．これを実現するために，GNU GPL は「GNU GPL ライセンスが適用されたソースコード（もとのプログラムのソース）を含むソース（改良したプログラムのソース）は公開しなければならない」

表 6.2 SUSUBOARD Ver.1 の諸元

FPGA	XILINX XC2S200E-6PQ208C
外部メモリ	1MByte SRAM CY7C1049CV33-15VC ×2
PS/2 ポート	2 ポート
RS-232C ポート	MAX3232CSE
Fast Ethernet ポート	DP83846AVHG
NTSC Video OUT	AD9760AR
NTSC Video IN	AD9203ARU
Stereo Audio OUT	AD1866R
Stereo Audio IN	AD1871YRS
ATA HDD Interface	2.5inch/3.5inch 用各 1
OSC (各 1)	12.288MHz, 14.318MHz, 25MHz
基板サイズ	200 × 130mm （4 層）

という制約を課している．しかしこの制約は一方で，商用利用などにおいて，戦略上ソースを公開したくないプログラムを開発したい場合には GNU GPL が適用されたプログラムのソースは利用できない（開発するプログラムに直接は組み込めない）ということになり，企業などは利用を見送る可能性がある．

次に GNU LGPL ライセンスは，GNU GPL と異なり，「独占的なプログラムに組み込むこと」を許可している．これにより，3. の「ソースの公開」が保証されない．従って，GNU GPL のような改良の連鎖が起こりにくくなるものの，企業が独占的に開発するソフトウェアにも組み込んで利用することが可能となる．しかし GNU プロジェクトとしては，開発の連鎖が起こらないライセンスは推奨されていない．

最後に，修正版 BSD ライセンスも GNU LGPL のように「独占的なプログラムに組み込む」利用が可能である（言及していない）．また GNU GPL や GNU LGPL ライセンスに比べると単純で短いため，利用が容易である．しかし GNU LGPL のように改変について触れていないため，「バージョンの同一性の保証」が成される保証がない．

これらのライセンスを参考にオープンソース・IP に適したライセンスを考える．まず，オープンソース・IP は，コンピュータの上で実行されるプログラムではなく，FPGA の回路構成情報である．従って，オープンソース・IP は単一で動作するものではなく，システムの一部に組み込まれて動作することが前提である．そして，そのシステムはオープンソースである保証がないため，これを許可しない GNU GPL は適切でない．従って，GNU LGPL ライセンス若

しくは，修正版 BSD ライセンスのどちらかということになるが，オープンソース・IP はコンピュータ・ソフトウェアの場合と異なり，常にコンピュータ上で取り扱われるとは限らず，ライセンスの文章量が多いとハードウェア資源を余計に占有するため，できるだけ短い文章であることが望ましい．従って，修正版 BSD ライセンスが最もオープンソース・IP 用のライセンスとして適していると考えられ，今回開発した IP には修正版 BSD ライセンスを適用することとした．しかし，コンピュータ・ソフトウェア向けに作られたライセンスは，用語の違いなどからそのまま適用することはできなかったため，これに加筆修正を行った “The SUSUBOX License, Version 1.0” を作成した．付録 A にオープンソース・IP 向けに作成したライセンス “The SUSUBOX License, Version 1.0” を添付する．

6.4.2 評価

以上のように，FPGA 上で動作する回路資産を開発し，オープンソース・ライセンスを適用することで，オープンソース・IP を実際に配信することはできたが，今回は定量的な評価を行うことはできなかった．ただ，評価指標は TAT，コストであるため，評価手法としては同一の回路システムを構築する課題において，本オープンソース・IP を用いて開発する場合と市販の IP を用いる場合，IP を一切用いない場合との間で比較評価することが考えられる．

6.5 まとめ

本章では，FPGA 本来のラピッド・プロトタイピング・ツールという利用方法を発展させたアプリケーションについて述べた．FPGA 上に実現されるハードウェアの実態が回路構成情報というソフトウェアによって仮想的に構成されるものであることを利用した，オープンソース・IP を提案した．これにより，

定性的に，

これまでハードウェアであったものに，オープンソース・ソフトウェアの特徴である，

利用側

1. **低コスト（ほぼ無料）で利用可能**
2. **様々な関連ソフトウェアや派生物が期待できる**
3. **コミュニティの発生，活性化が期待できる**
4. **高い透明性を得られる**

5. **構造，動作の理解，改良が容易**

提供（開発）側

1. **急速な普及が期待できる**
2. **（自律的な）アウトソーシングが期待できる**
3. **保証・サポートの放棄が可能**

などを取り入れることができることがわかった．

第 7 章

結論

本論文では，再構成集積回路の中でも，LUT アレイ型のアーキテクチャを持つ FPGA を用いて，その「再構成可能」という特徴を生かせる，効果的なアプリケーションについて述べた．また，実際に４つのアプリケーションに適用し，その有効性を示した．

再構成可能集積回路の「再構成性」を生かせるアプリケーションは，従来の MPU で実現が困難な高速処理，サイズ，省電力性能を要求し，かつ ASIC などのフォト・マスク・ベース LSI で実現が困難な汎用性，TAT，コストが求められるものである．これは，

条件１.）MPU での実現が困難なものとして

1. 高速処理を要求し，アルゴリズムの並列性が高い応用
2. 小さい回路規模，省電力性能を要求する応用

のいずれかを満たし，かつ

条件２.）ASIC 化が困難なものとして

a. 適切な処理アルゴリズム／プロトコルが未確定な分野での応用，
b. 優先する性能が変化する応用，
c. 回路構成情報のソフトウェアとしての利点を生かせる応用

のいずれかを満たすものである．

また，下記に示す具体的な適用例にて実証した．

満足条件	適用例
1-a.	確率的ニューラルネットワークを用いた画像認識装置
1-b, (1-a).	性能可変 IP パケット・フィルタ装置
1-a, 2-a, 1-c, 2-c.	汎用 I/O インターフェース
1-c, 2-c.	新しいハードウェア開発手法

確率的ニューラルネットワークはアルゴリズムの並列性が非常に高く，これを FPGA に実装して構築した画像認識装置は，認識対象によってはビデオレート時間で 98.3% 以上の認識精度を達成するなど，高い性能を実現することができた．また，このような高い性能を得るためには，認識対象に応じて前処理手法などを変更する必要があり，これはコスト面や TAT から考えると，ASIC 化するよりも FPGA で実現した方が適したアプリケーションであるといえる．さらに，本研究で開発したシステムは，高速に PNN 計算が行えるため，特徴抽出が困難な認識処理において特に有用となる見通しを得た．

また，IP パケット・フィルタも非常に並列性の高いアルゴリズムを持ったアプリケーションであり，FPGA によって実現される IP パケット・フィルタ装置は，特に基幹ネットワーク～エンドユーザ・ネットワークを結ぶ，中流ネットワークにおいて，性能，コスト面から有用であることがわかった．さらにネットワークの状況に応じた処理速度と処理可能な IP アドレス数の優先順位の変更が可能な IP パケット・フィルタ装置が実現できる見通しを得た．

本論文で提案した FPGA によって実現される汎用 I/O インターフェースは，PC や PDA と周辺装置間の I/O インターフェースなど，従来 ASIC/ASSP 化されていたインターフェース回路に適用することで，そのプロトコルを標準規格でない，各メーカ独自のものを使用することを可能とした．これにより，周辺装置毎に最適なプロトコルを実現可能となり，周辺装置の高性能化が期待できる他，開発コストの低減や，周辺装置のユーザビリティの向上が可能となる見通しを得た．

４つ目の適用例は他の適用例と異なり，回路の性能向上などに関するものではなく，ハードウェア開発における，開発効率の向上の可能性を示した．FPGA 上に構築する回路を，オープンソース・IP（オープンソース・ソフトウェア）としてインターネットを通じて配信し，新しいアウトソーシングの形態を構築することで，メーカにおけるハードウェア開発の効率を向上させるだけでなく，学生などを含む，回路設計のアマチュア層へ製品開発を広げることが可能であることを示した．

謝辞

本研究は，筑波大学 大学院 博士課程 システム情報工学研究科 コンピュータ・サイエンス専攻にて行ったものです．本研究を進めるにあたり，筑波大学 大学院 博士課程 システム情報工学研究科 コンピュータ・サイエンス専攻 安永守利教授には，5 年もの長期に渡り，電子回路の基礎から，研究姿勢に至るまで，終始御指導賜りましたことを，心から感謝致します．

また，本研究をまとめるに際し，貴重なお時間を割いて，御指導と御助言下さった，筑波大学 大学院 博士課程 システム情報工学研究科 コンピュータ・サイエンス専攻 西川博昭教授，和田耕一教授，山口喜教教授，ならびに，筑波大学 大学院 博士課程 システム情報工学研究科 知能機能システム専攻 丸山勉助教授に心より感謝致します．

さらに，本研究を進めるにあたり，確率的ニューラルネットワークを用いた画像認識装置に関して，さまざまな助言を戴いた，宮崎大学 大学院 工学研究科 情報工学専攻 吉原郁夫教授，アーカンソン大学 システム工学科 Jung H. Kim 教授，筑波大学 大学院 博士課程 システム情報工学研究科 コンピュータ・サイエンス専攻 平井有三教授に心より感謝致します．また，集積システム研究室の OB にあたる，水野良亮君には同認識システムにおいて，PC との接続回路，制御ソフトウェアの開発と，手形状，オリベッティ・ベンチマーク画像による認識精度の測定を行って戴きました．同じく，同研究室 OB の角谷夏樹君には，開発初期において，矢印画像を対象とした，ソフトウェア・シミュレーションによる認識精度の評価を行って戴きました．

私が FPGA に出会ったのは，1998 年の卒業論文（学士）の時でした．私は当時，東京 新宿にある私立 工学院大学 電子工学科の学生で，自分の卒論テーマに人工知能，知識情報処理に関するものを選択したく，そのような研究を行っている研究室を探していました．しかし，コンピュータ上で高級言語を用いてそのようなテーマで研究を行っているところはいくつかありましたが，自分の好きなハードウェアで実装することを対象に行っているところは，学内にはありませんでした．そこで，結局，知識情報処理に関するテーマを諦め，ハードウェア実装を対象に研究を行っていた 工学院大学 1 部 電子工学科 大類重範助教授の研究室を選択しました．大類先生の電子回路研究室では，アナログ／ディジタル・フィルタに関する研究を行っており，TI 社製の DSP(Digital Signal Processor) を用いた研究がメインでした．しかし，大類

先生は非常に寛容で，私が当時，知識情報処理に関することに興味があることを伝えると，自分で研究テーマを決めて卒論を行うことを認めてくださいました．そんなとき，大学の地下一階にある高層棟用エレベータ・ホールで，理化学研究所の一般公開に関するポスターを目にしました．その中で理化学研究所 脳科学総合研究センター 脳型デバイス・ブレインウェイ・グループ 故 松本元 グループディレクタが，「脳を創る」というテーマで講演を行うことが記されており，非常に興味が湧いたので聴講しに行くことにしました．それまで，理研という名前も知らなかったのですが，このときの行動が私の大きな人生の分岐になったと思っています．

松本先生の講演は，私が予想していたものより遥かにわかりやすく，興味深いものでした．難しい神経モデルの数式が沢山出てくるのではなく，鮭の産卵の話や，野球選手の成功事例を用いた話で，脳が目標を設定し，その目標を満たすように自然と行動を進めていく，仮説立証主義に関するものが中心で，それを実現しようと試みていることに感銘を受けました．理研の一般公開では，さまざまな研究室の公開があったようですが，そのときは松本先生の話を聞きに行くことが目的だったので，あまり他の研究室を見るつもりはありませんでした．ところが，その松本先生の講演が行われた会場の隣にあったプレハブ小屋の研究室で，大きなプリント基板が展示されているのが目に留まりました．そのときは，そこが松本先生の研究室であることを知らなかったのですが，そこで理化学研究所 脳科学総合研究センター 脳型デバイス・ブレインウェイ・グループ 脳創生デバイス研究チーム 市川道教 チームリーダが脳型コンピュータに関する展示を行っていました．そこで展示されていたプリント基板は，初期のニューロ・プロセッサの開発に用いられたもので，沢山の FPGA が実装されていました．いわば，ASIC 化前の論理検証を行う，FPGA エミュレータだったようです．家に戻って，何か研究テーマを決める参考になるような資料がないかと思い，市川先生の研究室の Web サイトを見ていたら，技術系のアルバイトを募集していたので，さっそく市川先生にメールでどのような内容なのかを尋ねてみました（メール.1）．すると，何故か「卒論研究の受け入れ」が可能という，予想もしないお返事を頂きました（メール.2）．現在の「大学の先生が認めれば可能」とのことでしたので，さっそく大類先生に相談してみたところ，そんなに良いお話があるのであれば是非行って来なさいと快く許可してくださいました．その後，一度も直接お会いしてもいないのに，卒論生として受け入れてくださることとなりました．

理化学研究所にて，卒業研究として「FPGA による自然空間認識のための視覚情報処理プロセッサの製作」を行い，FPGA による回路設計の基礎から，視覚情報処理に関する生体の仕組みに関するまで松本先生，市川先生を始め，ブレイン・ウェイグループの研究員の方々から多くのことを教わりました．今日の私があるのも，このとき学んだ多くのことがあるためです．改めまして，理化学研究所 脳科学総合研究センター 脳型デバイス・ブレインウェイ・グループ 故 松本元 グループディレクタ，同グループ 脳創生デバイス研究チーム 市川道教 チームリーダ，大学院受験の際の学習指導まで戴いた，同研究チームの山田整博士を始めとする同グルー

プの研究員の皆様，技術的な助言を戴いた株式会社ヤマモトシステムデザイン 山本頼寿氏に心より感謝致します．また，外で卒業研究を行うことを快く薦めてくださった，工学院大学 1 部電子工学科 大類重範助教授に深く感謝致します．

また，本論文をまとめるにあたり，議論，助言戴いた，田中寛之君，川合浩之君を始めとする，筑波大学 大学院 博士課程 システム情報工学研究科 コンピュータ・サイエンス専攻 安永研究室（集積システム研究室）のメンバーの皆様に深く感謝致します．

最後に，ここに至るまで支えてくださった祖父と母に心より感謝致します．

To: welcome@brainway.riken.go.jp
Subject: Staff
From: c294001@ccs.kogakuin.ac.jp (SUSUTAWARI)

初めまして。工学院大学４年の相部と申します。

本日（15 日）一般公開で研究室に伺い、この URL を教えていただき
「脳型コンピュータに関連した電子回路開発業務」を行うスタッフの募集を
されていることを知りました。
「このお知らせの有効期限は 1997 年 12 月―1998 年 3 月です。」となっていましたが
まだ募集されているのでしょうか？

私は今年度、工学院大学の電子回路研究室に卒論の配属が決まり、
現在、人工知能関連、特に「学習」に関することを卒論にしたいと考えております。
さらに私の与えられたテーマは「IC 応用回路の試作研究」であるため、
「ハードウェア」で実現できるものを探しています。
（もともとうちの大学には人工知能関連でハードをやらせてくれるところがなかった
ので現在のところを希望したのですが）
将来もこの関連の研究ができればと考えております。
もしまだ募集をされているようでしたら一度お話を伺いたいのですが。

ｂｙ すすたわり

|--|
| 工学院大学　電子工学科 電子回路研究室： 相部 範之　　|
| |
	---	** E-Mail:c294001@ccs.kogakuin.ac.jp

メール.1

From: ichikawa@brainway.riken.go.jp (Michinori Ichikawa)
To: "SUSUTAWARI " <c294001@ccs.kogakuin.ac.jp>
Subject: RE: Staff
Date: Wed, 15 Apr 1998 22:14:51 +0900

ご興味をお持ち頂けて光栄です。

さて、スタッフですが、相部さんの場合、卒論学生としてこちらで
勉強されたいということですか？ もし、大学の教授がそれを認めて
下さるようでしたら、受け入れることは可能です。
暇を見つけて遊びに来てください。こちらは、いつでも歓迎します。
もちろん、忙しくて、あまり満足に対応してあげられないこともありますが、
研究の様子を見れば、いろいろと参考になるかもしれません。

市川道教

Michinori Ichikawa
http://brainway.riken.go.jp

メール.2

参考文献

[1] 末吉 敏則, *Reconfigurable Computing System* **の現状と課題** *–Computer Evolution* **へ向けて**–, 電子情報通信学会技報, Vol.96, No.426, pp.111–118, 1996.

[2] 末吉 敏則, **リコンフィギャラブルロジック**, 電子情報通信学会誌, Vo.81, No.11, pp.1100–1106, Nov. 1998.

[3] 末吉 敏則, 飯田 全広, **リコンフィギャラブル・コンピューティング**, 情報処理学会誌, Vol.40, No.8, pp.777-782, Aug. 1999.

[4] 田丸 啓吉, **やわらかい** *LSI* **デバイス：***FPGA*, 情報処理学会誌, Vol.40, No.8, pp.783-788, Aug. 1999.

[5] 今井 正治, *HW/SW* **コデザインとやわらかいハードウェア**, 情報処理学会誌, Vol.40, No.8, pp.789-794, Aug. 1999.

[6] 樋口 哲也, **進化型ハードウェア**, 情報処理学会誌, Vol.40, No.8, pp.795-800, Aug. 1999.

[7] K. Compton and S. Hauck, *Reconfigurable Computing: A Survey of Systems and Software*, ACM Computing Surveys, Vol.34, No.2, pp.171–210, June 2002.

[8] Waugh T.C., *Field Programmable Gate Array Key to Reconfigurable Array Outperforming Supercomputers*, Proc. IEEE CICC, pp.6.6.1–6.6.4, 1991.

[9] Brown S.D., Francis R.J., Rose J. and Vranesic Z.G., *Field-Programmable Gate Array*, Kluwer Acad. Pub., pp.13–43, 1992.

[10] Brown S.D., *Field-Programmable Devices*, Stan Baker Associates, 1995.

[11] Andre DeHon, *The Density Advantage of Configurable Computing*, COMPUTER, IEEE Computer Society, pp.41–49, April 2000.

[12] Wirthlin M.J. and Hutchings B.L., *DISC: The Dynamic Instruction Set Computer, in Field Programmable Gate Arrays (FPGAs) for Fast Board Development and Reconfigurable Computing*, John Schewel, Editor, Proc. SPIE 2607, pp.92–103, 1995.

[13] K. Nagami, K. Oguri, T. Shiozawa, H. Ito and R. Konishi, *Plastic Cell Architecture: Towards Reconfigurable Computing for General-Purpose*, The IEEE Symposium on FPGAs for Custom Computing Machines, 1998.

[14] Kouichi Nagami, Kiyoshi Oguri, Tsunemichi Shiozawa, Hideyuki Ito and Ryusuke Konishi, *Plastic Cell Architecture: A Scalable Device Architecture for General-Purpose Reconfigurable Computing*, IEICE Transactions on Electronics, Vol.E81-C, No.9, pp.1431–1437, September 1998.

[15] 永見 康一, 塩澤 恒道, 小栗 清, **自律再構成可能アーキテクチャ**, 設計自動化研究会, 87–3, 1998.

[16] 塩澤 恒道, Norbert Imling, 永見 康一, 小栗 清, *Communicating Logic* **による汎用情報処理機構**

の実現, 電子情報通信学会技報, Vol.99, No.480, pp.57–64, 1999.

[17] 伊藤 秀之, 小栗 清, 小西 隆介, 中田 広, **非同期式動的再構成論理** *LSI* **の設計**, 電子情報通信学会技報, Vol.99, No.480, pp.65–72, 1999.

[18] Seth Copen Goldstein, Herman Schmit, Mihai Budiu, Srihari Cadambi, Matt Moe, R.Reed Taylor, *PipeRench: A Reconfigurable Architecture and Compiler*, COMPUTER, IEEE Computer Society, pp.70–76, April 2000.

[19] Faura J., Horton C., Duong P.V., Madrenas J., Aguirre M.A. and Insenser J.M., *A Novel Mixed Signal Programmable Device with On-Chip Microprocessor*, Proc. IEEE CICC, pp.103–106, 1997.

[20] 岩田 昌也, 坂無 英徳, 樋口 哲也, **印刷画像データ圧縮用進化型ハードウェアチップ**, 電子情報通信学会論文誌, Vol.J86-C No.8, pp.771-779, Aug. 2003.

[21] Simon D., John Stone, Peter Y.K.Cheung, and Wayne Luk *Video Image Processing with the Sonic Architecture*, COMPUTER, IEEE Computer Society, pp.50–57, April 2000.

[22] Marco Platzner, *Reconfigurable Accelerators for Combinatorial Problems*, COMPUTER, IEEE Computer Society, pp.58–60, April 2000.

[23] 中島, 森, 佐藤, 高橋, 浅見, 水上, 飯田, 新留, *FPGA* **ベース並列マシン** *RASH*, 並列処理シンポジウム JSPP'99, pp.222, 1999.

[24] 浅見 廣愛, 飯田 全広, 中島 克人, 森 伯郎, *FPGA* **ベース並列マシン** *RASH* **での** *DES* **暗号解析処理の改良**, 情報処理学会論文誌, HPS, Vol.41, No.SIG5(HPS1), pp.50–57, Aug. 2000.

[25] 山口 晃由, 橋山 智訓, 大熊 繁, *Reconfigurable Computing System* **の暗号処理への適用**, 電子情報通信学会技報, Vol.99, No.480, pp.25–30, 1999.

[26] 市川 周一, Lerdtanseangtham Udorn, 小西 幸治, **部分グラフ同型判定アルゴリズムの** *FPGA* **による実装と評価**, 情報処理学会論文誌, HPS, Vol.41, No.SIG5(HPS1), pp.39–49, Aug. 2000.

[27] 黒川 恭一, 古市 洋希, 柴田 慎一, *CPLD* **を用いたニューラルネットワークの開発**, 電子情報通信学会論文誌 研究速報, Vol.J83-D-II, No.3, pp.1054–1059, March 2000.

[28] Y. Hori, M. Sonoyama, and T. Maruyama, *An FPGA-Based Processor for Shogi Mating Problems*, IEEE International Conference on Field-Programmabe Technology, pp.117–124, 2002.

[29] 今井 正治, 武内 良典, **ハードウェア／ソフトウェア・コデザイン手法**, 電子情報通信学会誌, Vol.81, No.11, pp.1132–1138, 1998.

[30] Staunstrup J. and Wolf W., *Hardware/Software Co-Design*, Principles and Practice, Kluwer Academic Publishers, 1997.

[31] Imai M., Shiomi A., Takeuchi Y., Sato J. and Honma Y., *Hardware/Software Codesign in the Deep Submicron Era*, Proc. of IWLAS '97, pp.236–248, Grenoble, France, Dec. 1996.

[32] 越智 裕之, *FPAA:* **フィールドプログラマブルアキュームレータアレイ**, 計算機アーキテクチャ, pp.97-102, Oct. 1997.

[33] 越智 裕之, *FPAccA:* **フィールドプログラマブルアキュームレータアレイ***–FPAccA model 1.0* **チップの設計と評価**, 情報処理学会論文誌, Vol.40, No.4, pp.1717-1725, Apr. 1999.

[34] アイピーフレックス株式会社, *DAPDNA-2* **ダイナミック・リコンフィギュラブル・プロセッサ データシート**, http://www.ipflex.com/jp/, 2004.

[35] 片山 勝, 甲斐 英則, 山田 博希, 塩本 公平, 山中 直明, **リコンフィギャラブルプロセッサを用いた**

10Gb/s **ファイアウォール装置の実現**, 電子情報通信学会 技術研究報告 第 4 回リコンフィギャラブルシステム研究会, pp.67-72, Nagasaki, Japan, Sep.15-17 2004.

[36] 宇野 正樹, 柴田 裕一郎, 天野 英晴, 吉田 浩一朗, 藤井 太郎, 本村 真人, **動的な部分再構成デバイスを用いた仮想ハードウェアシステム**, 電子情報通信学会 技術研究報告 コンピュータシステム, VLD99-104, CPSY99-113, Jan. 2000.

[37] NEC エレクトロニクス株式会社, **動的再構成プロセッサ**, http://www.necel.com/ja/techhighlights/drp/, 2004.

[38] 出口 勝昭, 山田 裕, 天野 英晴, *DRP* **でのウェーブレットフィルタの実装**, 情報処理学会研究報告, システム LSI 設計技術, Vol.2003, No.007, Jan. 2003.

[39] Duncan A. Buell, Jeffrey M. Arnold, and Walter J. Kleinfelder, *Splash2 : FPGAs in a Custom Computing Machine*, IEEE Computer Society Press, 1996.

[40] Ernesto Novillo and Paul Lu, *A Case Study of Selected SPLASH-2 Applications and SBT Debugging Tool*, IEEE International Parallel and Distributed Processing Symposium (IPDPS'03), Apr.22-26, 2003.

[41] Specht, D.: Probabilistic Neural Networks, *J.Neural Networks*, Vol. 3, pp. 109–118 (1990).

[42] Specht, D.: Probabilistic Neural Networks and the Polynomial Adaline as Complementary Techniques for Classification, *IEEE Trans. Neural Networks*, Vol. 1, No. 1, pp. 110–111 (1990).

[43] Müller, Klaus-R., Mika, S., Rätsch,G., Tsuda,K. and Schölkopf,B.: An Introduction to Kernel-Based Learning Algorithms, *IEEE Trans. Neural Networks*, Vol. 12, No. 2, pp. 181–201 (2001).

[44] Ruiz, A., *Member*,IEEE. and E.López-de Teruel, P.: Nonlinear Kernel-Based Statistical Pattern Analysis, *IEEE Trans. Neural Networks*, Vol. 12, No. 1, pp. 16–32 (2001).

[45] Mao, K.Z., Tan, K.C. and Ser, W.: Probabilistic Neural-Network Structure Determination for Pattern Classification, *IEEE Trans. Neural Networks*, Vol. 11, No. 4, pp. 1009–1016 (2000).

[46] Tian, B., R.Azimi-Sadjadi, M. *Senior Member*, IEEE. H.Vonder Harr, T. and Reinke, D.: Temporal Updating Scheme for Probabilistic Neural Network with Application on Satellite Cloud Classification, *IEEE Trans. Neural Networks*, Vol. 11, No. 4, pp. 903–920 (2000).

[47] Tian, B., R.Azimi-Sadjadi, M.: Comparison of Two Different PNN Training Approaches for Satellite Cloud Data Classification, *IEEE Trans. Neural Networks*, Vol. 12, No. 1, pp. 164–168 (2001).

[48] Minchin, G. and Zaknich, A.: A Design for FPGA Implementation of the Probabilistic Neural Network.

[49] 岩田, 雨宮: ニューラルネットワーク LSI, 電子情報通信学会編, オーム社 (1995).

[50] Gotarredona, T.S., Barranco, B.L. and Andreou, A.G.: Programmable Kernel Analog VLSI Convolution Chip for Real Time Vision Processing, *Proc. IEEE Int'l Joint Conf. Neural Networks*, CD-ROM (2000).

[51] Szabo, T., Antoni, L., Horvath, G. and Feher, B.: A full-parallel digital implementation for pre-trained NNs, *Proc. IEEE Int'l Joint Conf. Neural Networks*, CD-ROM (2000).

[52] Girau, B. and Lorraine, L.: Digital hardware implementation of 2D compatible neural net-

works, *Proc. IEEE Int'l Joint Conf. Neural Networks*, CD-ROM (2000).

[53] Yasunaga, M., Masuda, N., Yagyu, M., Asai, M., Yamada, M. and Masaki,A.: Design, Fabrication and Evaluation of A 5-inch Wafer Scale Neural Network LSI Composed of 576 Digital Neurons, *IJCNN '90*, II, pp. 527–535 (1990).

[54] 平井 有三, 落合 辰男, 安永 守利: 1000 ニューロン 100 万シナプスで構成されたニューラルネットワークハードウェアシステム, 電子情報通信学会論文誌, J84-D-II, No. 6, pp. 1185–1193 (2001).

[55] Sinencico, E.S. and Lau, C.(Eds.): *Artificial Neural Networks: Paradigms, Applications, and Hardware Implementations*, IEEE Press (1992).

[56] Kohonen, T.: *Chapter 8 (Hardware for SOM) in "Self-Organizing Maps"*, Vol. 1, Springer, 2nd edition, chapter 2, pp. 371–381 (1997).

[57] Wawrzynek, J., Asanovic, K., Kingsbury, B., Johnson, D., Beck, J. and Morgan, N.: Spert-II: A Vector Micro Processro System, *Special Issue of Neural Computing in IEEE COMPUTER*, Vol. 29, No. 3, pp. 79–86 (1996).

[58] Bishop, C.: *Neural Network for Pattern Recognition*, Oxford Press (1995).

[59] 高島 浩二, 中村 太郎, 安永 守利, 吉原 郁夫: 遺伝的アルゴリズムを用いた画像中の顔画像検出とその進化ハードウェア上への実装, 電子情報通信学会技報, PRMU2000-117, pp. 71–78 (2000).

[60] 山本 幹,"アクティブネットワーク", 電子情報通信学会誌 TS, Vol.86, No.7, pp.489–492, July, 2003.

[61] A. G. Fragkiadakis, N. G. Bartzoudis D. J. Parish, and M. Sandford, "Active networking using programable hardware", PGNet2003, June, 2003.

[62] S. Li, J. Torresen, O. Soraasen, "Exploiting reconfigurable hardware for network security", FCCM2003, 2003.

[63] Y. H. Cho, S. Navab, and W. H. Mangione-Smith, "Specialized hardware for deep network packet filtering", FPL2002, pp452-461, 2003.

[64] Sipper M., Mange D. and adn Andres Perez-Uribe (editors), *CAM Brain*, Proc. of Second International Conference on Evolvable Systems ICES98, Lecture Notes on Computer Science, 1478, Springer Verlag, 1998.

[65] XILINX Inc., "Virtex 2.5V Field Programmable Gate Arrays Data Sheet (DS003-1 v2.5)", http://www.xilinx.co.jp/bvdocs/publications/ds003.pdf, April 2001.

[66] XILINX Inc., "Virtex–E 1.8V Field Programmable Gate Arrays Data Sheet (DS022-1 v2.3)", http://www.xilinx.co.jp/bvdocs/publications/ds022.pdf, July 2002.

[67] XILINX Inc., *Spartan and Spartan–XL Families Field Programmable Gate Arrays (DS060 v1.7)*, http://direct.xilinx.com/bvdocs/publications/ds060.pdf, June 2002.

[68] XILINX Inc., *Spartan–II 2.5V FPGA Family: Complete Data Sheet (DS001)*, http://direct.xilinx.com/bvdocs/publications/ds001.pdf, August 2004.

[69] XILINX Inc., *Spartan–IIE 1.8V FPGA Family: Complete Data Sheet (DS077)*, http://direct.xilinx.com/bvdocs/publications/ds077.pdf, July 2004.

[70] XILINX Inc., *Virtex–II Pro and Virtex–II Pro X FPGAs: Complete Data Sheet (DS083 v4.1)*, http://direct.xilinx.com/bvdocs/publications/ds083.pdf, November 2004.

[71] XILINX Inc., *Platform Flash In-System Programmable Configuration PROMs (DS123 v2.5)*, http://www.xilinx.co.jp/bvdocs/publications/ds123.pdf, October 2004.

[72] ALTERA Corp., *FLEX 10K: Embedded Programmable Logic Device Family (DS–F10K–4.2 ver.4.2)*, http://www.altera.com/literature/ds/dsf10k.pdf, January 2003.

[73] ALTERA Corp., *Stratix Device Handbook, Volume 1 and 2 (S5V1–3.1, S5V2–3.1)*, http://www.altera.co.jp/literature/hb/stx/stratix_handbook.pdf, 2004.

[74] ALTERA Corp., *Cyclone Device Handbook, Volume 1 and 2 (CSV1–1.4, CSV2–1.0)*, http://www.altera.co.jp/literature/hb/cyc/cyclone_device_handbook.pdf, 2003.

[75] RealTeK Semiconductor Corp., *REALTEK SINGLE CHIP SINGLE PORT 10/100M FAST ETHERNET PHYCEIVER RTL8201BL (Rev.1.2)*, http://www.realtek.com.tw/, March 2002.

[76] Compaq Computer Corp., Hewlett-Packard Company, Intel Corp., Lucent Technologies Inc., Microsoft Corp., NEC Corp., Koninklijke Philips Electronics N.V., *Universal Serial Bus Specification Rev.2.0*, http://www.usb.org/, April 2000.

[77] HIROSE ELECTRIC Co.Ltd, *Harf pitch interface connector: DXM Series (CL230)*, http://www.hirose.co.jp/, 2004.

[78] J. Postel, “INTERNET PROTOCOL”,RFC791 (STD5), Sep, 1981.

[79] Eric S. Raymond, “The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary”, Oreilly & Associates Inc., Oct., 1999.

[80] 青山 幹雄, “オープンソースソフトウェアの現状”, 情報処理学会会誌, Vol.43, No.12, pp.1319-1324, Dec. 2002.

[81] 河原 正治, 大瀧 保広, “オープンソースから見た超流通技術”, 情報処理学会研究報告, 電子化知的財産・社会基盤, Vol.2000, No.056, Jun. 2000.

[82] Tim O’Reilly, “Lessons From Open-Source Software Development”, Communications of the ACM, Vol.42, No.4, pp.33-37, Apr. 1999.

[83] 八田 真行, **オープンソースの定義，八田 真行訳**, *2004* **年** *2* **月** *21* **日 バージョン** *1.9*, http://www.opensource.jp/osd/osd-japanese_plain.html, 2004.

[84] “OPENCORES”, http://www.opencores.org/, 2004.

[85] Jeung Joon Lee, “CMOSexod”, http://www.cmosexod.com/ , 2004.

[86] 小山忠昭, “FPGA インフォメーション”, http://www.fpga.co.jp/ , 2004.

研究業績

論文誌 （掲載済み：２）

1. Noriyuki Aibe, Ryousuke Mizuno, Masanori Nakamura, Moritoshi Yasunaga, and Ikuo Yoshihara, “Performance Evaluation System for Probabilistic Neural Network Hardware”, The journal of Artificial Life and Robotics (Springer-Verlag), Vol.8, No.2, pp.208–213, 2004.

2. 相部 範之, 安永 守利, “確率的ニューラルネットワーク計算の並列高速化アーキテクチャとその画像認識システムへの適用”, 情報処理学会論文誌：ハイパフォーマンスコンピューティングシステム, Vol.43, No.SIG6(HPS5), pp.206-218, 2002.

国際会議 （発表済み：１０）

1. Noriyuki Aibe, Moritoshi Yasunaga, “Reconfigurable I/O Interface for Mobile Equipments”, IEEE International Conference on Field-Programmable Technology (ICFPT) 2004, The University of Queensland, St Lucia Campus, Brisbane, Australia, pp.359-362, 6-8 December, 2004.

2. Noriyuki Aibe, Moritoshi Yasunaga, “Meta-I/O Interface Using Reconfigurable LSIs”, IEEE International Midwest Symposium on Circuits and Systems (MWSCAS2004), Hiroshima, Japan, pp.I-537-I-540, 25-28 July, 2004.

3. Noriyuki Aibe, Moritoshi Yasunaga, “Reconfigurable Parallel Comparation Architecture and Its Application to IP Packet Filters”, IEEE International Conference on Field-Programmable Technology (ICFPT) 2003, University of Tokyo, Tokyo, Japan, 15-17 December, pp.363-366, 2003.

4. Noriyuki Aibe, Ryousuke Mizuno, Masanori Nakamura, Moritoshi Yasunaga, and Ikuo Yoshihara, “Performance Evaluation System for Probabilistic Neural Network Hardware”,

The Eighth Int. Symp. on Artificial Life and Robotics (AROB 8th '03), Beppu, Oita, Japan, 24-26 January, 2003.

5. Moritoshi Yasunaga, Noriyuki Aibe, Ryosuke Mizuno, Masanori Nakamura, and Ikuo Yoshihara, "Power-Quake Reduction Design in VLSI", Proceeding of 2002 Pacific Rim International Symposium on Dependable Computing (PRDC2002), Fast Abstract, pp.7–8, Tsukuba, Japan, 16-18 December, 2002.

6. Noriyuki Aibe, Moritoshi Yasunaga, Ikuo Yoshihara, and Jung H Kim, "A Probabilistic Neural Network Hardware System Using A Learning-Parameter Paralell Architecture", Proceeding of International Joint Conference on Neural Networks 2002 (IJCNN2002) on IEEE WCCI2002, CD-ROM, Honolulu, Hawaii, 11-17 May, 2002.

7. Moritoshi Yasunaga, Kentaro Ushiyama, Noriyuki Aibe, Hidetoshi Fujiwara, Ikuo Yoshihara, and Jung H Kim, "An Evolutionary Kernel-Based Reasoning System Using Reconfigurable VLSIs: Its Hardware Prototyping and Application to the Splicing Boundary Problem", Proceeding of IEEE Congress on Evolutionary Computation 2002 on IEEE WCCI2002, CD-ROM, Honolulu, Hawaii, 11-17 May, 2002.

8. Noriyuki Aibe, Moritoshi Yasunaga, and Ikuo Yoshihara, "Probabilistic Neural Network Processor for Image Recognition Using Reconfigurable LSIs", 2001 International Symposium on Nonlinear Theory and its Applications (NOLTA2001), Zao, Miyagi, Japan, Vol.1, pp.111-114, Oct.28-Nov.1, 2001.

9. Noriyuki Aibe, Moritoshi Yasunaga, and Ikuo Yoshihara, "Self-learning Probabili stic Neural Network Hardware Using Reconfiguable LSIs", The Sixth Int. Symp. on Artificial Life and Robotics (AROB 6th '01), Tokyo, Japan, Vol.1, pp.89-92, 15-17 January, 2001.

10. Moritoshi Yasunaga, Noriyuki Aibe, Taro Nakamura, and Ikuo Yoshihara, "DP Matching Hardware for High-speed Genome Database Machine", Proceeding of International Conference on Artificial Intelligence in Science and Technology, pp.256–261, December, 2000.

国内会議，研究会

1. 相部 範之, 安永 守利, "メタ・I/O インターフェース", 電子情報通信学会 コンピュータシステム研究会 (第 2 種研究会) 第 4 回リコンフィギャラブルシステム研究会, pp.212-219, 長崎大学, 15-17 September, 2004.

2. 相部 範之, 安永 守利, “再構成型並列比較アーキテクチャ (RPCA) とその IP パケット・フィルタへの適用”, 電子情報通信学会 コンピュータシステム研究会 (第 2 種研究会) 第 1 回リコンフィギャラブルシステム研究会, 熊本大学, 18-19 September, 2003.

3. 相部 範之, 安永 守利, “確率的ニューラルネットワーク計算の並列高速化アーキテクチャとその画像認識システムへの適用”, 情報処理学会 並列処理シンポジウム JSPP2002 論文集 (IPSJ Symposium Series Vol.2002, No.8), pp.27-34, Tsukuba, Japan, 29-31 May, 2002.

4. 相部 範之, 安永 守利, 吉原 郁夫, “確率的ニューラルネットワークを用いた高速画像認識システムの開発”, 電子情報通信学会技術報告, FIIS-02-96, 8 March, 2002.

受賞

1. 相部 範之, “実用的なフリー IP ライブラリ SUSUBOX”, 2004 年 第 6 回 LSI IP デザイン・アワード 開発助成 大学部門 研究助成賞, 日経 BP 社, 2004.

2. 相部 範之, “A Probabilistic Neural Network Hardware System Using A Learning-Parameter Paralell Architecture”, 財団法人 C&C 振興財団 国際会議論文発表者助成 2002 年度 前期, 2002.

その他著書，公表

1. 相部 範之, “フリー IP ライブラリ SUSUBOX と開発プラットフォーム”, IP フリーマーケット, EDS フェア 2005, パシフィコ横浜, 27-28 January, 2005.

2. 相部 範之, “ユビキタスコンピューティングに最適なハードウェアによるパターン認識”, COMUTER & NETWORK LAN 3月号, オーム社, pp.61-66, March, 2004.

3. 相部 範之, “実用的なフリー IP SUSUBOX”, IP フリーマーケット, EDS フェア 2004, パシフィコ横浜, 29-30 January, 2004.

付録 A

The SUSUBOX License, Version 1.0

注意：本ライセンスは，本論文中内容に関する参考であり，本論文に適用するものではありません.

原文

```
/* The SUSUBOX License, Version 1.0
 *
 * URL: http://www.susubox.org/
 * E-mail: susu@susubox.org
 *
 * Copyright (c) 2003 Noriyuki Aibe
 * All rights reserved.
 *
 * In this license, the "source" includes:
 * "circuit diagrams written by CAD Tools and its files, VHDL codes,
 * Verilog-HDL codes, other language codes, and net-lists".
 *
 * The "binary" includes:
 * "bit-stream files that are possible to be implemented (downloaded) to
 * reconfigurable devices (such as FPGAs or CPLDs)".
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
```

```
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 *
 * THIS SOFTWARE IS PROVIDED BY NORIYUKI AIBE AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL NORIYUKI AIBE OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 */
```

日本語訳

```
/* The SUSUBOX License, Version 1.0 （訳）
 * 注：本文は便宜上作成した、"LICENSE"の日本語訳であり、原文では
 * ありません。SUSUBOX に適用するライセンスは英語版の"LICENSE"ファイルに
 * 記載された内容とし、英語版の"LICENSE"と本訳文"LICENSE.euc.jp"及び
 * "LICENSE.sjis.jp"に記載された内容に差異があった場合には英語版の
 * "LICENSE"の内容が優先されます。
 *
 * URL: http://www.susubox.org/
 * E-mail: susu@susubox.org
 *
```

付録 B

SUSUBOX ドキュメント

SUSUBOX™

データシートの見方

Document Ver.0.5J (2004.2.28) Made by Susutawari

本文は「データシート」の見方について解説したものです。

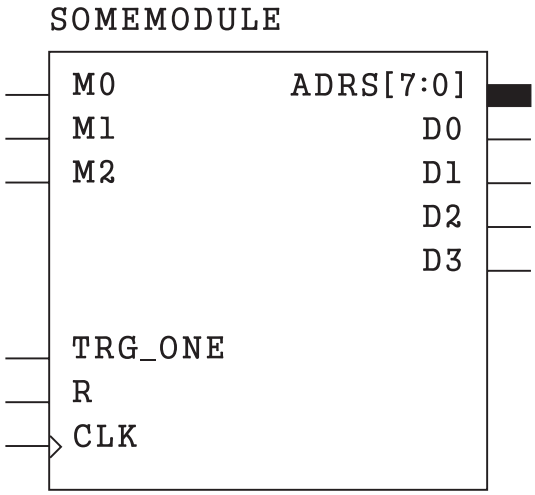


図 1: SOMEMODULE のシンボル

はじめに各機能モジュールの回路図（ViewDraw）上でのシンボルが記載されています。各ピンの機能については２ページ目以降に表（“Pin Function Description”）が記載されています。

1 機能概要

各機能モジュールに関する概要はここに記載されています。

関連モジュール
SOMEMODULE1, SOMEMODULE2, SOMEMODULE3,... ここには一緒に使用するモジュール、関連するモジュールなどの名前が記載されています。

機能仕様

ここには、そのモジュールの機能面での仕様が記載されています。これは、モジュール毎に大きく異なり、モジュールによってはこの項目自体が記載されていないものもあります。

出力サイクル	xx ms
入力データ幅	x bit
Spec. 3	–
Spec. 4	–

:

実装仕様

ここには、各モジュールを FPGA や CPLD にインプリメント（実装）する際の仕様が記載されています。

表は上から、ターゲットとなるデバイスのシリーズ名、現在提供しているソースの形式、回路のサイズ（使用 CLB 数、もしくは SLICE 数）、となっています。回路サイズの見積りは、実際に ISE のインプリメンテーションを行った結果で、その条件及び結果の詳細は後の方に記載してあります。

Target Device	Virtex, Viertex-E, Spartan, SpartanXL, Spartan2, XC4000E, XC9500, etc.
Source Type	ViewDraw Sch, EDIF, etc.
Circuit Size	xxx CLBs or SLICEs

モジュール名

ここに実際のモジュールのファイル名（拡張子を除く）が記載されています。VHDL や Verilog-HDL などから EDIF を呼び出す場合にも、この名前を使用します。名前の構成は、ベースクロックに応じて選択する必要があるモジュールに関しては、“モジュール名_ベースクロック周波数” となっており、最後の M は MHz を表し、数字の途中に入る P は小数点を表しています。また先頭、最後とも数字とならないようにしています。表中の “Base CLK Freq.” に表示がない場合は周波数非依存型のモジュール、記載されている周波数が範囲表記でない場合にはその周波数用のモジュール、範囲表記の場合はその範囲で動作可能なモジュールであることを表しています。

Base CLK Freq.	Module Name
–	SOMEMODULE
4.2MHz	SOMEMODULE_4P2M
～8MHz 以下	SOMEMODULE_8M
8MHz より速く～16MHz 以下	SOMEMODULE_16M
:	:

また、モジュール名の付け方に関する規則は、まだ厳密な定義がありませんが、機能面での意味に関してはおおよそ次のようになっています。

ENC	エンコーダ（符号器）
DEC	デコーダ（復号器）
GEN, G	ジェネレータ（発生器）
CTR, C	コントローラ（制御器）
CONV, C	コンバータ（変換器）
XTOY	X から Y への変換
INIT	イニシャライザ（初期化器）
DRV, D	ドライバ（駆動器）
CMD	コマンド（命令）関連
CHAR	キャラクタ（文字）コード関連
DISP	ディスプレイ（表示器）関連
ROM	データの書き込まれたメモリモジュール、またそのサンプル
SAMPLE	サンプル（参考用）

2 動作解説

各モジュールの入出力信号に対する動作がここに記載されています。必要に応じてタイミングチャートやフローチャート、ロジックアナライザなどによる実測波形などを載せている場合もあります。

表 1: Pin Function Descriptions（ピン機能の解説）

Pin 名	信号方向	機能	備考
シンボルのピン名です	IN、OUT、或いは IN/OUT（双方向）	機能を示します	機能解説、注意すべきことなどが書かれています
:	:	:	:

表 2: Time Measurment Results（時間測定結果）

モジュール名	ベースクロック周波数	t_1[ns]	t_2[ns]	t_3[μs]
SOMEMODULE_4P2M	4MHz	2000	4000	80
SOMEMODULE_16M	10MHz	200	700	15
	16MHz	163	570	12
:	:	:	:	:

表 1 はモジュールの入出力ピンに関する説明をしたものです。表 2 は入出力特性などを表したもので、特定の時間のパルスを生成するモジュールなどで実装測定されたものについてのみ記載しています。また同様に測定条件も記載しています。

パルス時間測定条件

Target Device	Virtex XCVXXX
OSC	16, XXMHz: XTAL-OSC
ISE Version	6.1.03i
Oscilloscope	XXX

各モジュールの回路サイズは “（回路規模測定用）実装条件” に記載された条件でインプリメンテーションを行い、表 3 に Map Report から得られる結果を示しています。また、回路規模を正確に求めるため、I/O などの作成は行わず、モジュールのみのインプリメンテーションを行っています。このため、実際にユーザの回路中に組み込んで使用する場合には他のロジックとのパッキングにより、より小さくなる可能性があります。

（回路規模測定用）実装条件

Target Device	Virtex, Spartan, etc.
ISE Version	6.1.03i
Map Properties	Trim Unconnected Signals: Disable
Place & Route Properties	Effort Level: Standard
	Cost Table: 1
	Place and Route Mode: Normal Place and Route

表 3: Circuit Size Implementation Results

モジュール名	Slices	Gates	FFs	LUTs
SOMEMODULE_4P2M	8	154	11	11
SOMEMODULE_16M	9	172	11	14
:	:	:	:	:

ピン名の付け方に関する規則もモジュール名と同様に、まだ厳密な定義がありませんが、機能面での意味に関してはおおよそ次のようになっています。

DIN	データ入力
DOUT	データ出力
ADRS	アドレス入力／出力
TRG	トリガ入力
E	イネーブル入力／出力
CE	クロック・イネーブル入力
CEO	クロック・イネーブル出力
_ONE	シングルショット・パルス入力／出力（接尾語）
R	リセット入力
RDY	レディ信号出力
CLK	クロック入力

3 HDL でのインスタンシエーション方法

VHDL や Verilog-HDL から本モジュール（IP）を利用（EDIF をインポート）する際のインスタンシエーション例を記載しています。モジュールによっては、このインスタンシエーションを用いたサンプルコードを付属しているものもあるので、そちらも参考にしてください。周波数毎にいくつかモジュールがある場合には、特定の周波数向けのモジュール（SOMEMODULE_4P2M）などを例に記述してあります。

4 使用許諾及び再配布に関して

各モジュールに対するライセンスの参照先などを記載しています。基本的に、各モジュールは修正 BSD ライセンスを基にしたものとなっており、各モジュールのアーカイブにテキストファイルとして同封されています。

その他の解説

ファイル名の長さ制限は十数文字程度とし、8 文字以内には限定していません。データシートの場合は ‘SDS_’ で、本文のような解説やお知らせは ‘SINFO_’ で始まり、サフィックスの番号はバージョンを表します。データシートの場合、SOMEMODULE1_1J などは、機能モジュールのバージョンが 1.0 で、データシートのバージョン（Document Version）が 1.0.1J であることを示しています（最後の J は日本語版の意味です）。

by すすたわり

SUSUBOX™

Datasheet of “SBASE” Ver.1.0
SUSUBOX Base Module Library

Document Ver.1.0J (2004.03.30) Made by Susutawari

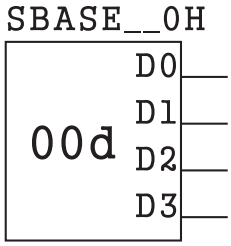


図 2: The symbol of SBASE__0H

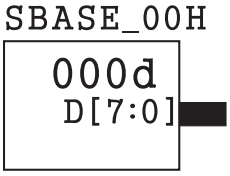


図 3: The symbol of SBASE_00H

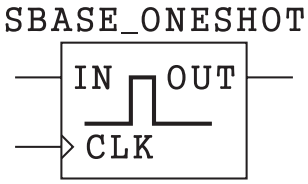


図 4: The symbol of SBASE_ONESHOT

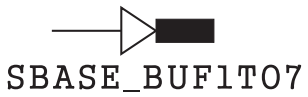


図 5: The symbol of SBASE_BUF1TO7

関連モジュール
SBASE__?H, SBASE_??H,
SBASE_ONESHOT, SBASE_BUF1TO*

5　機能概要

本ライブラリはSUSUBOXの基本となるモジュール群で、他のIPモジュールから利用されているため、常に最新のものをインストールしておく必要があります。ただし、本ライブラリを使用しているモジュールでも、HDLなどからEDIFをインスタンシエーションして利用するだけであれば必要ありません。あくまで回路図用のライブラリです。勿論、各モジュールを単一で使用することも可能です。Ver.1.0では、4bitおよび8bitの定数を出力するSBASE__?H, SBASE_??H (?には0〜Fまでの十六進の値が入ります)、シングル・ショット・パルスを生成する（微分回路）SBASE_ONESHOT、そしてバスの全ての信号線に同じ値を出力するためのSBASE_BUF1TO*（*には7,8,15,16が入り、それぞれがバスのビット幅を示します）があります。またViewDraw(DxDesigner)の回路図上で使われるSUSUBOXのロゴ・ブロックも含まれています。

機能仕様

SBASE_*H	4bit, 8bit 固定値出力
SBASE_ONESHOT	1CLK 幅 微分パルス出力
SBASE_BUF1TO*	1 to 7,8,15,16 変換

実装仕様

Target Device	Virtex, Virtex-E, Spartan2.
Source Type	EDIF 2.0, PDF, ViewDraw Sch.

SBASE__?H (4bit 版) モジュール名

Output Value (decimal)	Module Name
00d	SBASE__0H
01d	SBASE__1H
02d	SBASE__2H
03d	SBASE__3H
04d	SBASE__4H
05d	SBASE__5H
06d	SBASE__6H
07d	SBASE__7H
08d	SBASE__8H
09d	SBASE__9H
10d	SBASE__AH
11d	SBASE__BH
12d	SBASE__CH
13d	SBASE__DH
14d	SBASE__EH
15d	SBASE__FH

SBASE_?H (8bit 版) モジュール名

Output Value (decimal)	Module Name
000d	SBASE_00H
001d	SBASE_01H
002d	SBASE_02H
:	:
254d	SBASE_FEH
255d	SBASE_FFH

SBASE_ONESHOT モジュール名

Base CLK Freq.	Module Name
–	SBASE_ONESHOT

SBASE_BUF1TO* モジュール名

# of Output bits	Module Name
7	SBASE_BUF1TO7
8	SBASE_BUF1TO8
15	SBASE_BUF1TO15
16	SBASE_BUF1TO16

表 1: Pin Function Descriptions of SBASE__?H

Pin 名	信号方向	機能	備考
D0〜3	OUT	データ出力	各定数値が 4bit で出力される。

表 2: Pin Function Descriptions of SBASE_??H

Pin 名	信号方向	機能	備考
D[7:0]	OUT	データ出力	各定数値が 8bit で出力される。

表 3: Pin Function Descriptions of SBASE_ONESHOT

Pin 名	信号方向	機能	備考
IN	IN	パルス入力	ここに入力されたパルスの立ち上がりエッジで、1CLK 幅の微分パルスが生成される。
CLK	IN	クロック入力	周波数非依存
OUT	OUT	微分パルス出力	出力パルスは入力エッジから 1CLK 分遅れる。

表 4: Pin Function Descriptions of SBASE_BUF1TO*

Pin 名	信号方向	機能	備考
IN（非表示）	IN	データ入力	1bit のデータ入力。この入力値が出力バスの全 bit に出力される。
OUT[?:0]（非表示）	OUT	データ出力	それぞれ 7,8,15,16bit のデータ出力

6　使用許諾及び再配布に関して

“SBASE” Ver.1.0 のライセンスはアーカイブ・ファイル内に同封された “LICENSE” ファイルに記載されている SUSUBOX License Ver.1.0 の内容に準じます。SUSUBOX License Ver.1.0 は修正 BSD ライセンスを基に作られており、本モジュールを組み込んだ回路の公開などは強要していません。また、ライセンスを満たす限り、個人、商用を問わず自由に再配布が可能です。

本モジュールが有用であった場合、どんなところに利用されたか、また、感想などをを頂ければ幸いです。（連絡先：susu@susubox.org）

ｂｙ すすたわり

SUSUBOX™

Datasheet of “SC2004C_4DRIVERL” Ver.0.7 LCD Module SC2004C Driver (4 bit length, Long Cycle Version)

Document Ver.0.7.1J (2004.01.28) Made by Susutawari

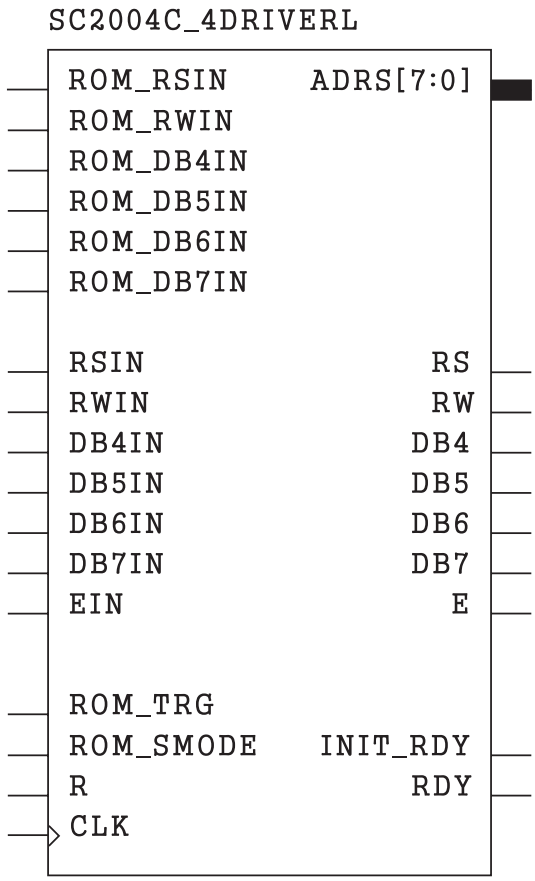


図 6: The symbol of SC2004C_4DRIVERL

関連モジュール
SC2004C_4DRIVER, SC2004C_MC, SC2004C_EGENL, SC2004C_EGEN, SC2004C_4INIT, SC2004C_4ROM, SC2004C_8TO4CC

7　機能概要

SUNLIKE 社製のキャラクタ・ディスプレイ LCD である、 SC2004C, SC1602B, 及び、セイコーインスツルメンツ社製の M1632, L1634 などをドライブするためのモジュールです。本モジュールにより、容易にこれらのキャラクタ LCD を利用できるようになります。本モジュールは LCD に表示する文字列や、インストラクションを記録したメモリからデータを読み出し、LCD に渡すためのコントローラ “SC2004C_MC”、初期化処理を行う “SC2004C_4INIT”、Enable パルスを生成する “SC2004C_EGENL” から構成されています（各モジュールの詳細についてはそれぞれのデータシートを参照してください)。尚、本モジュールのインターフェースデータ長は 4bit です。内蔵のメモリ・コントローラは、アドレス空間が 8bit あるので、128 インストラクション（128 文字）まで出力可能です。SC2004C は 20 文字 ×4 行なので、80 文字＋ 48 インストラクションの出力が可能です。また、２種類の出力速度を持っており、タイプ入力のような速度の文字表示ができます。

なお、LCD の規定にあったパルスを生成するため、使用するベースクロックに合わせてモジュールを選択する必要があります。規格はセイコーインスツルメンツ社（http://www.sii.co.jp/）の L1634 の「キャラクタタイプ液晶表示モジュール取り扱い説明書」（AN.No.CLCM-137）を参考にしています。

機能仕様

動作確認済み LCD	SC2004C rev.0
インターフェースデータ長	4 bit
Enable パルス出力サイクル (Min)	50μs
Enable パルス・セットアップ時間 (Min)	140ns
Enable パルス幅 (Min)	460ns
電源投入後の初期化待ち時間	45ms(Min)
初期化インストラクション Cycle	4.1ms(Min)
Duty Cycle 設定値	1/16
Character Size 設定値	5×7 Dots
カーソル・モード設定値	カーソル表示, インクリメント, シフトなし, 点滅あり
データ出力サイクル時間 (Min)	1.64ms/14.5ms
メモリアドレス空間	8bit (128 Instructions)

実装仕様

Target Device	Virtex, Virtex-E, Spartan2.
Source Type	EDIF 2.0, PDF, ViewDraw Sch.
Circuit Size	99〜121 Slices

モジュール名

Base CLK Freq.	Module Name
〜7.14MHz 以下	SC2004C_4DRIVERL_07P14M
7.14MHz より速く 〜14.28MHz 以下	SC2004C_4DRIVERL_14P28M
14.29MHz より速く 〜21.43MHz 以下	SC2004C_4DRIVERL_21P43M
21.44MHz より速く 〜28.57MHz 以下	SC2004C_4DRIVERL_28P57M
28.58MHz より速く 〜35.71MHz 以下	SC2004C_4DRIVERL_35P71M
35.71MHz より速く 〜49.99MHz 以下	SC2004C_4DRIVERL_49P99M
49.99MHz より速く 〜71.42MHz 以下	SC2004C_4DRIVERL_71P42M

※注：本モジュールは使用するベースクロックに合わせて選択する必要があります。表中の動作周波数を超える、高いベースクロックを使用した場合は動作しない可能性がありますが、低い周波数を入力した場合は、処理が遅くなるだけで論理動作には問題ありません。このため必要に応じてより高い周波数向けのモジュールを選択することで、動作マージンを大きくすることが可能です。

表 1: Pin Function Descriptions

Pin 名	信号方向	機能	備考
ROM_RSIN	IN	データ入力	メモリからの RS データ入力。
ROM_RWIN	IN	データ入力	メモリからの RW データ入力。LCD からの読み出しを行わない場合は LOW（0）に固定。
ROM_DB4〜DB7IN	IN	データ入力	メモリからの DB4〜7 データ入力。
RSIN	IN	データ入力	RS データ入力。
RWIN	IN	データ入力	RW データ入力。LCD からの読み出しを行わない場合は LOW（0）に固定。
DB4〜7IN	IN	データ入力	DB4〜7 データ入力。
EIN	IN	Enable パルストリガ入力	RSIN, RWIN, 及び DB4〜7 にデータをセットし、この EIN にパルスを入力すると、LCD へデータが出力される。INIT_RDY, 及び RDY が HIGH のときに有効。
ROM_TRG	IN	メモリ出力開始トリガ入力	メモリ・データ出力開始トリガ入力。アドレス 00h〜FFh までのデータが出力される。INIT_RDY, 及び RDY が HIGH のときに有効。
ROM_SMODE	IN	メモリ出力速度設定入力	メモリ出力の速度を設定する入力。LOW のときは通常速度、HIGH のときは低速で出力。
R	IN	初期化開始トリガ入力	LCD の初期化を開始するためのトリガ入力。電源投入時（FPGA リセット時）以外に、再度初期化したい場合に使用する。
CLK	IN	クロック入力	各周波数に合ったモジュールを選択する必要あり。
ADRS[7:0]	OUT	メモリ・アドレス出力	データを格納したメモリのアドレス端子に接続。00h〜FFh まで一定間隔でインクリメントされる。
RS	OUT	データ出力	RS データの出力。LCD の RS 端子に接続する。
RW	OUT	データ出力	RW データの出力。LCD の RW 端子に接続する。
DB4〜7	OUT	データ出力	DB4〜7 データの出力。LCD の DB4〜7 端子にそれぞれ接続する。
E	OUT	Enable パルス出力	Enable パルスの出力。LCD の E 端子に接続する。
RDY	OUT	レディ信号出力	この信号が HIGH になったら、次のトリガ入力、或いは LCD へのデータ出力が可能。従って、この信号のポジティブ・エッジで出力の完了がわかる。

8 動作解説

電源投入（FPGA がリセット）されると、本モジュールはまず LCD の初期化処理を開始します。初期化完了までに要する時間は使用するベースクロック周波数により異なりますが、最低 98.3[ms]（およそ 100〜140[ms] 前後）となっています（詳しくは “SC2004C_4INIT” のデータシートを参照してください)。初期化が完了すると “INIT_RDT” が HIGH となります。その後の動作は大きく 2 つあり、ユーザの目的に合わせて利用できます。1 つ目は LCD に送りたいデータを毎回ユーザが準備する場合です。この場合、4bit 長のデータを “RSIN”, “RWIN”, 及び “DB4〜DB7IN” にセットし、“EIN” にパルスを入力すると規定のセットアップ時間後に Enable パルスが出力され、LCD にデータが書き込まれます。次のデータの書き込みが可能となると “RDY” が HIGH となるので、このポジティブ・エッジを用いて、次のデータ出力のためのユーザ・シーケンスを開始できます。2 つ目は、LCD に出力したい文字列やインストラクションを予めメモリに書き込んでおき、それを連続出力する場合です。本モジュールの扱うアドレス空間は 8bit となっているので、4bit×256 のメモリを用いることで、128 インストラクション（128 文字）の出力が可能です。アドレスの制御は、単純に 8bit のバイナリカウンタが 00h〜FFh までインクリメントするだけなので、128 インストラクション未満で利用する場合には NOP（No OPeration）命令などでメモリを埋める必要があります。逆に、この NOP を利用することで文字表示のタイミングなどをある程度制御することもできます。なお、NOP は「表示オン／オフコントロール」命令などで現在と同じモードを再設定することで実現できます。“ROM_TRG” にパルスが入力されると、メモリのアドレス 00h 番地のデータから順に、規定時間の Enable パルスにより、LCD へ出力されます。メモリの全アドレスの出力（FFh）が完了し、再トリガが可能な状態になると “RDY” が HIGH になります。なお、初期化処理中、メモリ出力中、Enable パルス・サイクル規定時間内は “EIN” 及び “ROM_TRG” 入力は無視されます。

また、下記の条件で各モジュールの回路規模を求めました。Map Report から得た結果を表 2 に示します。I/O などは作成せず、モジュールのみインプリメンテーションしました。ISE のオプション設定は、基本的に Map Properties の “Trim Unconnected Signals” を行わないようにする以外はデフォルト値のままです。

実装条件

Target Device	Virtex, Virtex-E, Spartan2
ISE Version	6.1.03i
Map Properties	Trim Unconnected Signals: Disable
Place & Route Properties	Effort Level: Standard
	Cost Table: 1
	Place and Route Mode: Normal Place and Route

表 2: Circuit Size Implementation Results

モジュール名	Slices	Gates	FFs	LUTs
SC2004C_4DRIVERL_07P14M	99	2,204	99	150
SC2004C_4DRIVERL_14P28M	101	2,256	101	156
SC2004C_4DRIVERL_21P43M	105	2,358	107	165
SC2004C_4DRIVERL_28P57M	109	2,422	112	169
SC2004C_4DRIVERL_35P71M	111	2,458	115	171
SC2004C_4DRIVERL_49P99M	113	2,480	117	172
SC2004C_4DRIVERL_71P42M	121	2,570	123	179

9 HDL での インスタンシエーション方法

9.1 VHDL の場合

```
Component sc2004c_4driverl_35p71m
  Port(
        ROM_RSIN  : in std_logic;
        ROM_RWIN  : in std_logic;
        ROM_DB4IN : in std_logic;
        ROM_DB5IN : in std_logic;
        ROM_DB6IN : in std_logic;
        ROM_DB7IN : in std_logic;
        RSIN      : in std_logic;
        RWIN      : in std_logic;
        DB4IN     : in std_logic;
        DB5IN     : in std_logic;
        DB6IN     : in std_logic;
        DB7IN     : in std_logic;
        EIN       : in std_logic;
        ROM_TRG   : in std_logic;
        ROM_SMODE : in std_logic;
        R         : in std_logic;
        CLK       : in std_logic;
        ADRS      : out std_logic_vector
                    (7 downto 0);
        RS        : out std_logic;
        RW        : out std_logic;
        DB4       : out std_logic;
        DB5       : out std_logic;
        DB6       : out std_logic;
        DB7       : out std_logic;
        E         : out std_logic;
        INIT_RDY  : out std_logic;
        RDY       : out std_logic
      );
end Component;

INSTANCE_NAME : sc2004c_4driverl_35p71m
  Port map(
           ROM_RSIN  => ROM_RS,
           ROM_RWIN  => ROM_RW,
           ROM_DB4IN => ROM_DB4,
           ROM_DB5IN => ROM_DB5,
           ROM_DB6IN => ROM_DB6,
           ROM_DB7IN => ROM_DB7,
           RSIN      => DRIVERL_RSIN,
           RWIN      => DRIVERL_RWIN,
           DB4IN     => DRIVERL_DB4IN,
           DB5IN     => DRIVERL_DB5IN,
           DB6IN     => DRIVERL_DB6IN,
           DB7IN     => DRIVERL_DB7IN,
           EIN       => DRIVERL_EIN,
           ROM_TRG   => TRG_IN,
           ROM_SMODE => DRIVERL_ROM_SMODE,
           R         => DRIVERL_R,
           CLK       => INTERNAL_CLK,
           ADRS      => ROM_ADRS,
           RS        => RS_OUT,
           RW        => RW_OUT,
           DB4       => DB4_OUT,
           DB5       => DB5_OUT,
           DB6       => DB6_OUT,
           DB7       => DB7_OUT,
           E         => E_OUT,
           INIT_RDY  => DRIVERL_INIT_RDY,
           RDY       => DRIVERL_RDY
         );
```

9.2 Verilog-HDL の場合

```
module sc2004c_4driverl_35p71m
  (
   ROM_RSIN,
   ROM_RWIN,
   ROM_DB4IN,
   ROM_DB5IN,
   ROM_DB6IN,
   ROM_DB7IN,
   RSIN,
   RWIN,
   DB4IN,
   DB5IN,
   DB6IN,
   DB7IN,
   EIN,
   ROM_TRG,
   ROM_SMODE,
   R,
```

```
  CLK,
  ADRS,
  RS,
  RW,
  DB4,
  DB5,
  DB6,
  DB7,
  E,
  INIT_RDY,
  RDY
 );

  input ROM_RSIN;
  input ROM_RWIN;
  input ROM_DB4IN;
  input ROM_DB5IN;
  input ROM_DB6IN;
  input ROM_DB7IN;
  input RSIN;
  input RWIN;
  input DB4IN;
  input DB5IN;
  input DB6IN;
  input DB7IN;
  input EIN;
  input ROM_TRG;
  input ROM_SMODE;
  input R;
  input CLK;
  output [7:0] ADRS;
  output RS;
  output RW;
  output DB4;
  output DB5;
  output DB6;
  output DB7;
  output E;
  output INIT_RDY;
  output RDY;

endmodule
```

```
sc2004c_4driverl_35p71m INSTANCE_NAME
  (
   .ROM_RSIN (ROM_RS),
   .ROM_RWIN (ROM_RW),
   .ROM_DB4IN (ROM_DB4),
   .ROM_DB5IN (ROM_DB5),
   .ROM_DB6IN (ROM_DB6),
   .ROM_DB7IN (ROM_DB7),
   .RSIN (DRIVERL_RSIN),
   .RWIN (DRIVERL_RWIN),
   .DB4IN (DRIVERL_DB4IN),
   .DB5IN (DRIVERL_DB5IN),
   .DB6IN (DRIVERL_DB6IN),
   .DB7IN (DRIVERL_DB7IN),
   .EIN (DRIVERL_EIN),
   .ROM_TRG (TRG_IN),
   .ROM_SMODE (DRIVERL_ROM_SMODE),
   .R (DRIVERL_R),
   .CLK (INTERNAL_CLK),
   .ADRS (ROM_ADRS),
   .RS (RS_OUT),
   .RW (RW_OUT),
   .DB4 (DB4_OUT),
   .DB5 (DB5_OUT),
   .DB6 (DB6_OUT),
   .DB7 (DB7_OUT),
   .E (E_OUT),
   .INIT_RDY (DRIVERL_INIT_RDY),
   .RDY (DRIVERL_RDY)
   );
```

10　使用許諾及び再配布に関して

“SC2004C_4DRIVERL” Ver.0.7 のライセンスはアーカイブ・ファイル内に同封された “LICENSE” ファイルに記載されている SUSUBOX License Ver.1.0 の内容に準じます。SUSUBOX License Ver.1.0 は修正 BSD ライセンスを基に作られており、本モジュールを組み込んだ回路の公開などは強要していません。また、ライセンスを満たす限り、個人、商用を問わず自由に再配布が可能です。

本モジュールが有用であった場合、どんなところに利用されたか、また、感想などをを頂ければ幸いです。（連絡先：susu@susubox.org）

by すすたわり

SUSUBOX™

Datasheet of “SC2004C_4INIT” Version 0.7
LCD Module SC2004C Initializer (4bit Data Interface Version)

Document Ver.0.7.1J (2004.01.23) Made by Susutawari

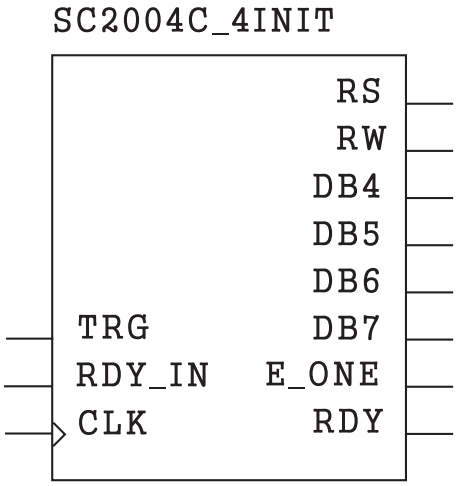


図 7: The symbol of SC2004C_4INIT

11 機能概要

SUNLIKE 社製のキャラクタ・ディスプレイ LCD である、 SC2004C, SC1602B, 及び、セイコーインスツルメンツ社製の M1632, L1634 などをドライブするためのモジュールの一部で、TOP モジュールである、SC2004C_4DRIVER(L) 中で使われています。LCD をドライブするには他に、SC2004C_MC, SC2004C_EGEN(L), SC2004_8TO4CC などが必要です。

本モジュールは上記 LCD を初期化するためのモジュールで、電源投入時やシステム・リセット時などに LCD に初期化コマンドを発行します。なお、本モジュールのインターフェースデータ長は 4bit です。LCD の規定にあったパルスを生成するため、使用するベースクロックに合わせてモジュールを選択する必要があります。規格はセイコーインスツルメンツ社（http://www.sii.co.jp/）の L1634 の「キャラクタタイプ液晶表示モジュール取り扱い説明書」（AN.No.CLCM-137）を参考にしています。ただし、SC2004C や M1632 は L1634 に比べると動作が遅いようで、秋月電子の付属資料などによると、倍以上のパルス幅が必要なようです。本モジュールはこれら全ての LCD を駆動できるように設計されています。

関連モジュール
SC2004C_4DRIVER, SC2004C_4DRIVERL
SC2004C_EGEN, SC2004C_EGENL,
SC2004C_MC, SC2004C_4ROM,
SC2004C_8TO4CC

機能仕様

インターフェースデータ長	4 bit
電源投入直後の待ち時間	45ms(Min)
インストラクション Cycle	4.1ms(Min)
Duty Cycle	1/16
Character Size	5×7 Dots
カーソル・モード	カーソル表示, インクリメント, シフトなし, 点滅あり

実装仕様

Target Device	Virtex, Virtex-E, Spartan2.
Source Type	EDIF 2.0, PDF, ViewDraw Sch.
Circuit Size	48〜56 Slices

モジュール名

Base CLK Freq.	Module Name
〜7.14MHz 以下	SC2004C_4INIT_07P14M
7.14MHz より速く 〜14.28MHz 以下	SC2004C_4INIT_14P28M
14.29MHz より速く 〜21.43MHz 以下	SC2004C_4INIT_21P43M

21.44MHz より速く 〜28.57MHz 以下	SC2004C_4INIT_28P57M
28.58MHz より速く 〜35.71MHz 以下	SC2004C_4INIT_35P71M
35.71MHz より速く 〜49.99MHz 以下	SC2004C_4INIT_49P99M
49.99MHz より速く 〜71.42MHz 以下	SC2004C_4INIT_71P42M

※注：本モジュールは使用するベースクロックに合わせて選択する必要があります。表中の動作周波数を超える、高いベースクロックを使用した場合は動作しない可能性がありますが、低い周波数を入力した場合は、処理が遅くなるだけで論理動作には影響ありません。このため必要に応じてより高い周波数向けのモジュールを選択することで、動作マージンを大きくすることが可能です。

表 1: Pin Function Descriptions

Pin 名	信号方向	機能	備考
TRG	IN	トリガ入力	初期化インストラクション発行開始トリガ入力。
RDY_IN	IN	RDY 入力	SC2004C_EGEN(L) の RDY 出力を接続する。
CLK	IN	クロック入力	使用する周波数に合ったモジュールを選択する必要あり。
RS	OUT	データ出力	常に LOW(0) を出力。
RW	OUT	データ出力	常に LOW(0) を出力。
DB4〜7	OUT	4bit データ出力	4bit インストラクション・データ長の出力。 これらデータ出力と RS, RW は直接 LCD の各入力端子と接続すればよい。
E_ONE	OUT	Enable パルス用トリガ出力	Enable パルス用のトリガ出力で、 シングルショット・パルス出力。 “SC2004C_EGEN(L)” の “TRG_ONE” に接続して使用する。
RDY	OUT	レディ信号出力	モジュールが動作中のとき LOW となる。 このポジティブ・エッジを見ることで、初期化完了のタイミングを得られる。

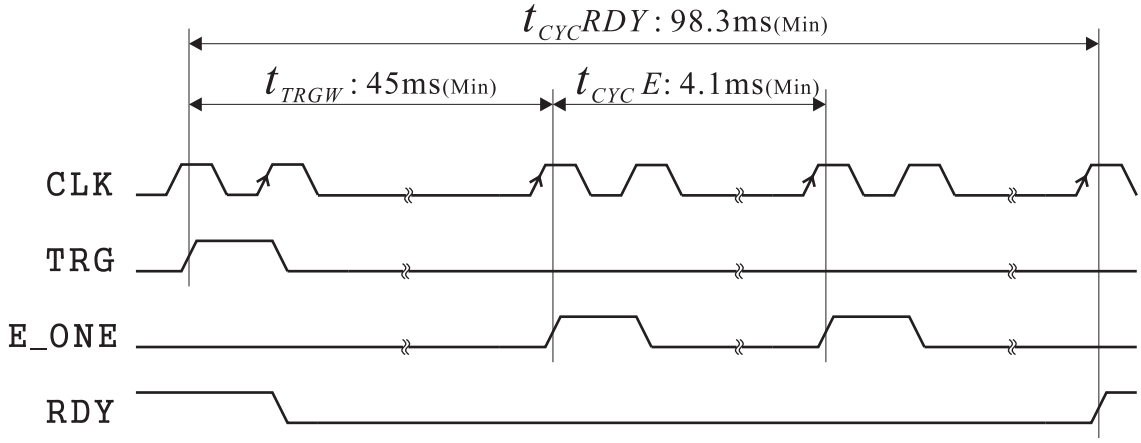


図 8: Timing Chart of SC2004C_4INIT

12 動作解説

電源が投入され、FPGA がリセットされた場合、或いは ‘TRG’ 入力があると、モジュールは ‘RDY’ 信号を LOW にして初期化インストラクションを発行します。図 8 に主要な信号のタイミングを示します。‘TRG’ 入力があると、次のクロックで ‘RDY’ 出力が LOW になり、‘TRG’ の立ち上がりから t_{TRGW}（45ms 以上）空けて、‘E_ONE’ パルスの出力が開始されます。‘E_ONE’ パルスは 1 クロック幅のシングルショット・パルスで、$t_{CYC}E$（4.1ms 以上）の間隔で 14 回出力されます。最後の ‘E_ONE’ パルスが出力された後、‘RDY_IN’ 入力が HIGH になるのを待ち、‘RDY’ 出力が HIGH に戻ります（$t_{CYC}RDY \geq 45\text{ms} + 4.1\text{ms} \times 13 +$“RDY 待ち時間”）。

ベースクロックが 28.6(36)MHz の場合のモジュール “SC2004C_4INIT_35P71M” 及び “SC2004C_EGENL_35P71M (Ver.1.0)” を用いた信号観測結果を図 9, 10 に示します。図はそれぞれ電源投入後、及び ‘TRG’ 信号入力後の波形です。尚、観測には Agilent Mixed Signal Oscilloscope 54622D を使用しました。また、図 11 は図 10 の後半部分を拡大したものです。図 9, 10 を見ると、電源投入後、或いは ‘TRG’ 入力後から約 56ms（45ms 以上）経過してから初期化インストラクションの発行が始まっていることがわかります。これはセイコーのキャラクタタイプ液晶表示モジュール総合取扱説明書 第 8 版（AN.No.CLCM-137）に基づくものです。従来の SC2004C, M1632 などはデータシートによると、15ms 以上の待ち時間で良いものがあるようですが、より汎用性を高めるため、本モジュールでは 45ms 以上の待ち時間を生成しています。また図 11 で示すように、各インストラクション（4bit 長なので、正確には 1/2 インストラクション）の発行サイクルは 4.1ms 以上となっています。実際に 4.1ms 以上の待ち時間が必要なのは、初めのインストラクション発行後のみですが、回路規模を抑えるため、全てのインストラクション発行サイクルで同一の間隔となっています。

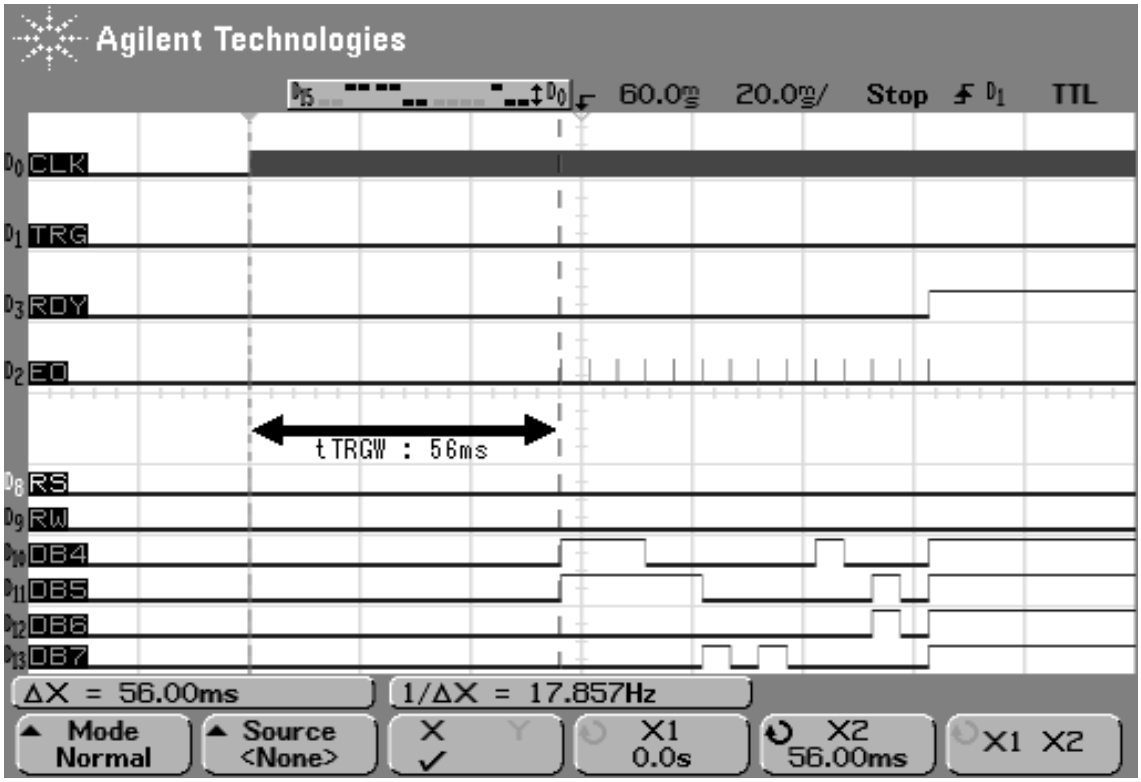


図 9: After Power On State of SC2004C_4INIT_35P71M

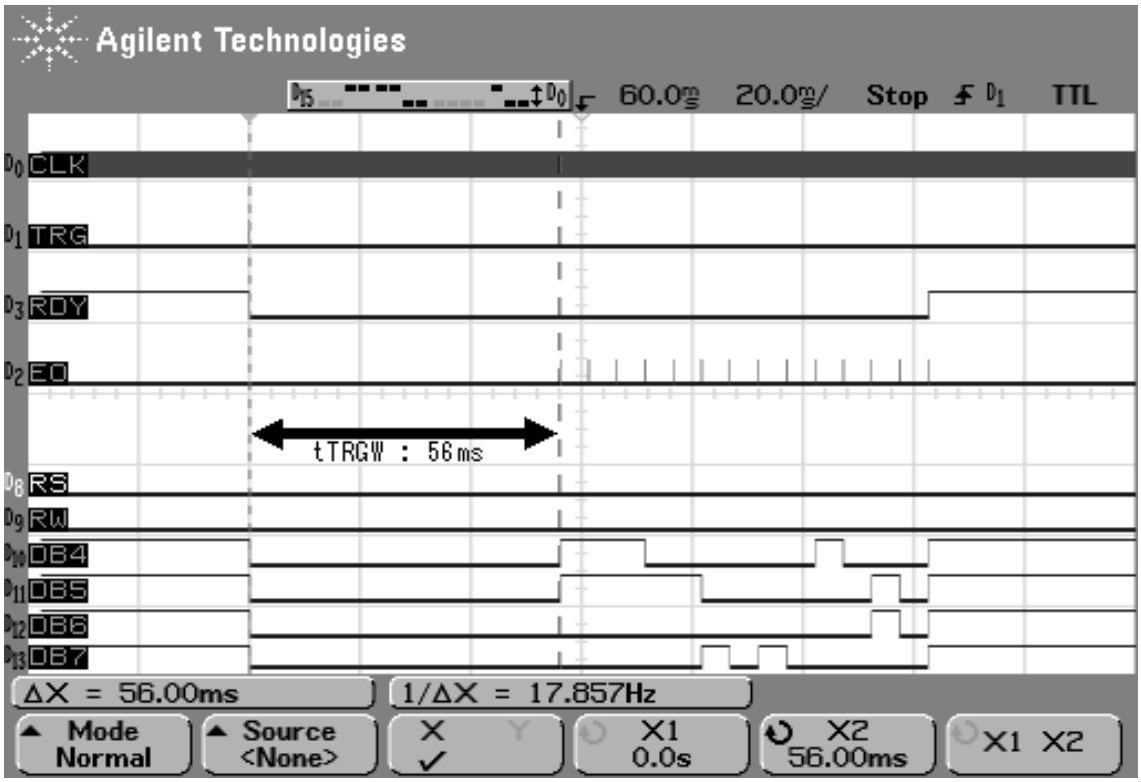


図 10: After TRG Input State of SC2004C_4INIT_35P71M

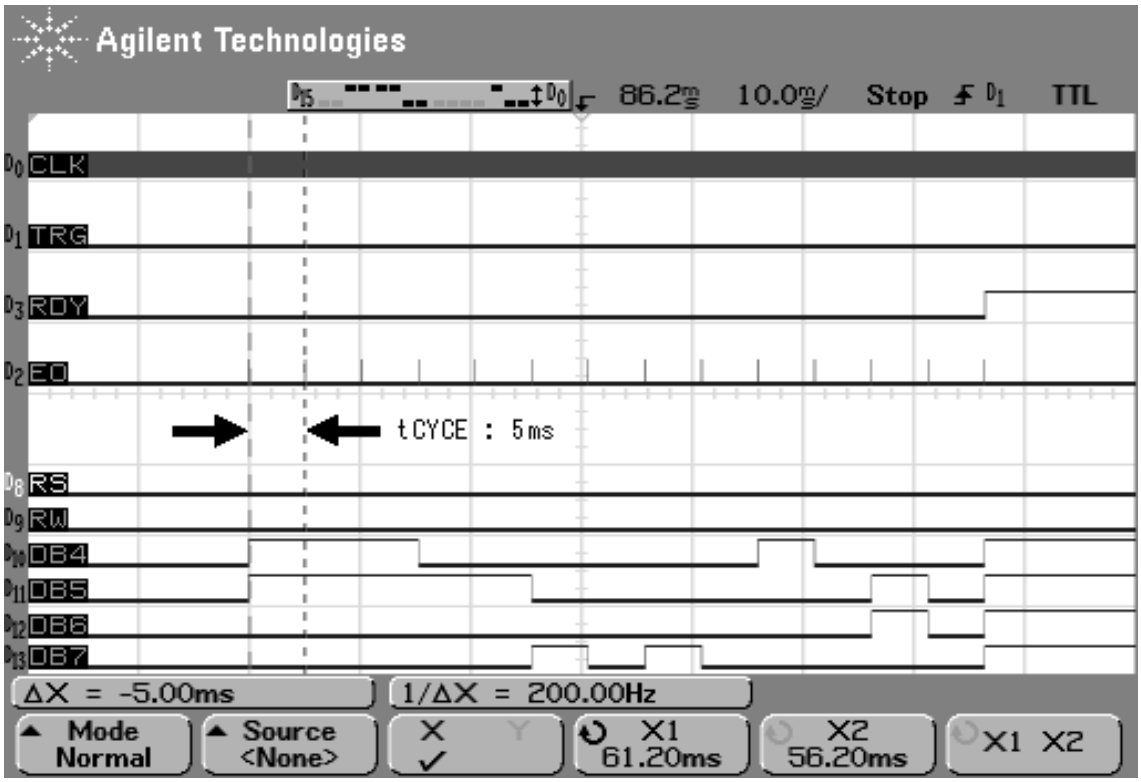


図 11: A Part(Magnified) of SC2004C_4INIT_35P71M's After TRG Input State

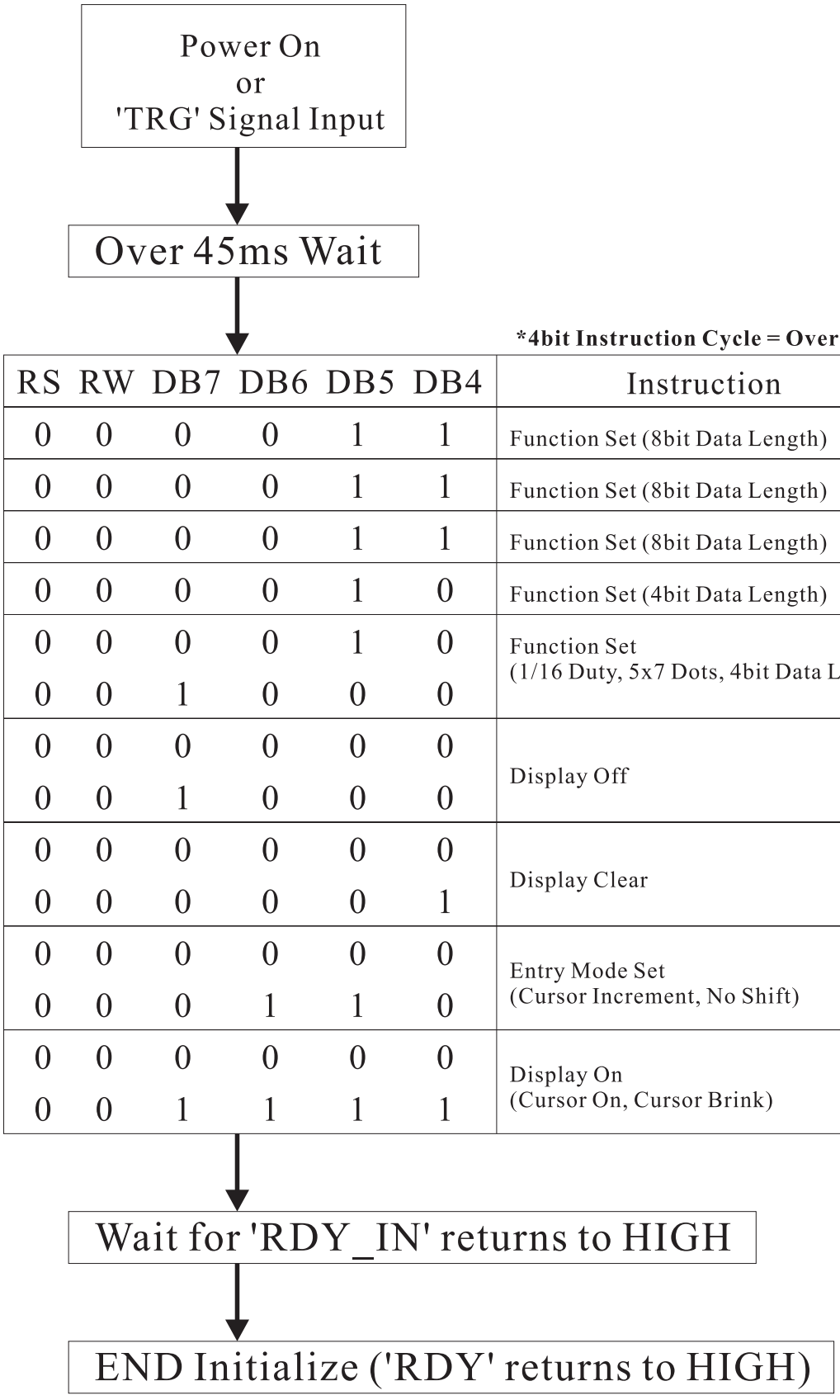


図 12: Instruction Flow

表 2: Time Measurment Results

モジュール名	ベースクロック周波数	t_{TRGW}[ms]	$t_{CYC}E$[ms]	$t_{CYC}RDY$[ms]
SC2003_4INIT_07P14M	1MHz	321.3	29.27	702.3
	5MHz	64.3	5.86	140.5
SC2004C_4INIT_14P28M	10MHz	64.3	5.86	140.5
	12.2(88)MHz	52.3	4.76	114.3
SC2004C_4INIT_21P43M	16MHz	60.3	5.49	131.7
	20MHz	48.2	4.39	105.4
SC2004C_4INIT_28P57M	25MHz	51.4	4.68	112.4
	27MHz	47.6	4.34	104.1
SC2004C_4INIT_35P71M	28.6(36)MHz	56.1	5.11	122.6
	30MHz	53.5	4.88	117.1
	35MHz	45.9	4.18	100.4
SC2004C_4INIT_49P99M	36MHz	62.5	5.69	136.6
	40MHz	56.2	5.12	122.9
	45MHz	50.0	4.55	109.3
SC2004C_4INIT_71P42M	50MHz	64.3	5.85	140.5
	60MHz	53.5	4.88	117.1
	70MHz	45.9	4.18	100.4

図 12 にインストラクション発行シーケンスを示します。図中では省略されていますが、各インストラクション（4bit 長のところは 1/2 インストラクション）間は 4.1ms 以上のウェイト時間が入っています。最後の「表示オン」インストラクション以外はデータシート通りです。また初期化が完了しても、Enable Pulse Generator（SC2004C_EGEN(L)）の 'RDY' が HIGH になるまで、本モジュールの 'RDY' 信号も HIGH に戻らないようになっています。

表 2 に各モジュールの動作周波数範囲内のベースクロックで動作させた場合の、各時間の測定結果を示します。$t_{CYC}RDY$ の測定にはそれぞれの周波数に合った SC2004C_EGENL （Ver.1.0）を使用しています。測定条件は下記の通りです。25MHz 以上のベースクロックには Design Gateway 社（http://www.dgway.com/）製の VariClock という可変オシレータを使用しました。ただ、これは設定可能周波数が 1MHz ステップなので、28.6(36)MHz などは従来の DIP タイプのオシレータを用いてテストしました。

測定条件

Target Device	Virtex-E XCV300E-6PQ240C
OSC	～24, 28.6(36)MHz: XTAL-OSC (DIP-14) 25～70MHz: DG VC250M14P
CLKDLL	Unused
ISE Version	6.1.03i
Oscilloscope	Agilent 54622D

また、次の条件で各モジュールの回路規模を求めました。Map Report から得た結果を表 3 に示します。I/O などは作成せず、モジュールのみインプリメンテーションしました。ISE のオプション設定は、基本的に Map Properties の “Trim Unconnected Signals” を行わないようにする以外はデフォルト値のままです。

実装条件

Target Device	Virtex, Virtex-E, Spartan2
ISE Version	6.1.03i
Map Properties	Trim Unconnected Signals: Disable
Place & Route Properties	Effort Level: Standard
	Cost Table: 1
	Place and Route Mode: Normal Place and Route

表 3: Circuit Size Implementation Results

モジュール名	Slices	Gates	FFs	LUTs
SC2004C_4INIT_07P14M	48	1,352	48	76
SC2004C_4INIT_14P28M	48	1,352	48	76
SC2004C_4INIT_21P43M	49	1,372	49	78
SC2004C_4INIT_28P57M	52	1,428	53	82
SC2004C_4INIT_35P71M	54	1,454	54	85
SC2004C_4INIT_49P99M	54	1,454	54	85
SC2004C_4INIT_71P42M	56	1,494	56	89

13 HDL でのインスタンシエーション方法

13.1 VHDL の場合

```
Component sc2004c_4init_35p71m
  Port(
        TRG     : in std_logic;
        CLK     : in std_logic;
        RS      : out std_logic;
        RW      : out std_logic;
        DB4     : out std_logic;
        DB5     : out std_logic;
        DB6     : out std_logic;
        DB7     : out std_logic;
        E_ONE   : out std_logic;
        RDY     : out std_logic
      );
end Component;

INSTANCE_NAME : sc2004c_4init_35p71m
  Port map(
            TRG     => TRG_IN,
            CLK     => INTERNAL_CLK,
            RS      => RS_OUT,
            RW      => RW_OUT,
            DB4     => DB4_OUT,
            DB5     => DB5_OUT,
            DB6     => DB6_OUT,
            DB7     => DB7_OUT,
            E_ONE   => E_ONE_OUT,
            RDY     => RDY_OUT
          );
```

13.2 Verilog-HDL の場合

```
module sc2004c_4init_35p71m
 (TRG, CLK, RS, RW, DB4, DB5, DB6,
  DB7, E_ONE, RDY);
  input TRG;
  input CLK;
  output RS;
  output RW;
  output DB4;
  output DB5;
  output DB6;
  output DB7;
```

```
  output E_ONE;
  output RDY;
endmodule


sc2004c_4init_35p71m INSTANCE_NAME
  (
   .TRG (TRG_IN),
   .CLK (INTERNAL_CLK),
   .RS (RS_OUT),
   .RW (RW_OUT),
   .DB4 (DB4_OUT),
   .DB5 (DB5_OUT),
   .DB6 (DB6_OUT),
   .DB7 (DB7_OUT),
   .E_ONE (E_ONE_OUT),
   .RDY (RDY_OUT)
  );
```

14 使用許諾及び再配布に関して

“SC2004C_4INIT” Ver.0.7 のライセンスはアーカイブ・ファイル内に同封された “LICENSE” ファイルに記載されている SUSUBOX License Ver.1.0 の内容に準じます。SUSUBOX License Ver.1.0 は修正 BSD ライセンスを基に作られており、本モジュールを組み込んだ回路の公開などは強要していません。また、ライセンスを満たす限り、個人、商用を問わず自由に再配布が可能です。

本モジュールが有用であった場合、どんなところに利用されたか、また、感想などをを頂ければ幸いです。（連絡先：susu@susubox.org）

by すすたわり

SUSUBOX™

Datasheet of “SC2004C_EGENL” Ver.1.0 LCD Module SC2004C Enable Pulse Generator (Long Cycle Version)

Document Ver.1.0.1J (2004.01.20) Made by Susutawari

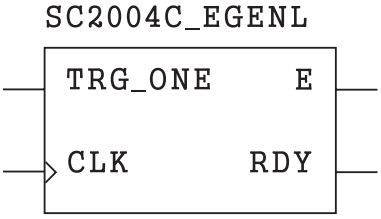


図 13: The symbol of SC2004C_EGENL

15 機能概要

SUNLIKE 社製のキャラクタ・ディスプレイ LCD である、 SC2004C, SC1602B, 及び、セイコーインスツルメンツ社製の M1632, L1634 などをドライブするためのモジュールの一部で、TOP モジュールである、SC2004C_4DRIVERL 中で使われています。LCD をドライブするには他に、SC2004C_MC, SC2004C_4INIT, SC2004_8TO4CC などが必要です。SC2004C_EGEN との違いは、LCD での命令実行時間を考慮し、パルス出力サイクルが 50μs 以上と長いことです。本モジュールは前述の LCD をドライブするために必要な、Enable パルスを生成します。モジュールの出力 “E” を LCD の “E” 端子に接続して使用します。LCD の規定にあったパルスを生成するため、使用するベースクロックに合わせてモジュールを選択する必要があります。なお、規格はセイコーインスツルメンツ社（http://www.sii.co.jp/）の L1634 の「キャラクタタイプ液晶表示モジュール取り扱い説明書」（AN.No.CLCM-137）を参考にしています。ただし、SC2004C や M1632 は L1634 に比べると動作が遅いようで、秋月電子の付属資料などによると、倍以上のパルス幅が必要なようです。本モジュールはこれら全ての LCD を駆動できるように設計されています。

関連モジュール
SC2004C_4DRIVERL, SC2004C_EGEN, SC2004C_4DRIVER, SC2004C_4INIT, SC2004C_MC, SC2004C_4ROM, SC2004C_8TO4CC

機能仕様

動作確認済み LCD	SC2004C rev.0
パルス出力サイクル (Min)	50μs
セットアップ時間 (Min)	140ns
パルス幅 (Min)	460ns

実装仕様

Target Device	Virtex, Virtex-E, Spartan2.
Source Type	EDIF 2.0, PDF, ViewDraw Sch.
Circuit Size	12〜25 Slices

モジュール名

Base CLK Freq.	Module Name
〜7.14MHz 以下	SC2004C_EGENL_07P14M
7.14MHz より速く 〜14.28MHz 以下	SC2004C_EGENL_14P28M
14.29MHz より速く 〜21.43MHz 以下	SC2004C_EGENL_21P43M
21.44MHz より速く 〜28.57MHz 以下	SC2004C_EGENL_28P57M
28.58MHz より速く 〜35.71MHz 以下	SC2004C_EGENL_35P71M
35.71MHz より速く 〜49.99MHz 以下	SC2004C_EGENL_49P99M
49.99MHz より速く 〜71.42MHz 以下	SC2004C_EGENL_71P42M

※注：本モジュールは使用するベースクロックに合わせて選択する必要があります。上記の動作周波数を超える、高いベースクロックを使用した場合は動作しない可能性がありますが、低い周波数を入力し

た場合は、処理が遅くなるだけで論理動作には問題ありません。このため必要に応じてより高い周波数向けのモジュールを選択することで、動作マージンを大きくすることが可能です。

16 動作解説

図 16 に各入出力ピンのタイミング・チャートを示します。初期状態は “E” が LOW, “RDY” が HIGH です。“TRG_ONE” にシングルショット・パルスが入力されると、規定のセットアップ時間 t_{AS}（140ns 以上）の後に Enable パルスが規定時間 PW_{EH}（460ns 以上）出力されます。L1634 等では Enable パルスの幅は 230ns で良いようですが、SC2004C, M1632 に対応するため、倍の時間になっています。また、ここで生成しているセットアップ時間は、TRG_ONE 入力からの時間で、LCD の RS, 及び R/W 端子への入力はこの TRG_ONE 入力以前に確定しているものとしています（AN.No.CLCM-137 中の t_{AS}: 40ns に相当）。さらに、“TRG_ONE” の入力直後（2CLK 後）から “RDY” は LOW となり、規定 Enable パルス・サイクル時間 PW_{EH}（50μs 以上）後に HIGH に戻ります。この RDY 信号の立ち上がりをトリガとして次のシーケンスに移ることで、ほとんどのインストラクション（表示クリアと、カーソルホームの実行待ちは、さらに 3 Cycle 時間以上必要）の実行時間以上のイネーブルサイクル時間 $t_{CYC}E$ を確保できるようになっています。

ベースクロックに合ったモジュールを使用した場合、出力されるパルスは、LCD（L1634）のデータ・シート（AN.No.CLCM-137）で定められている規定時間以上でかつ、SC2004C や M1632 で動作可能と思われる時間以上となるように設計されています。

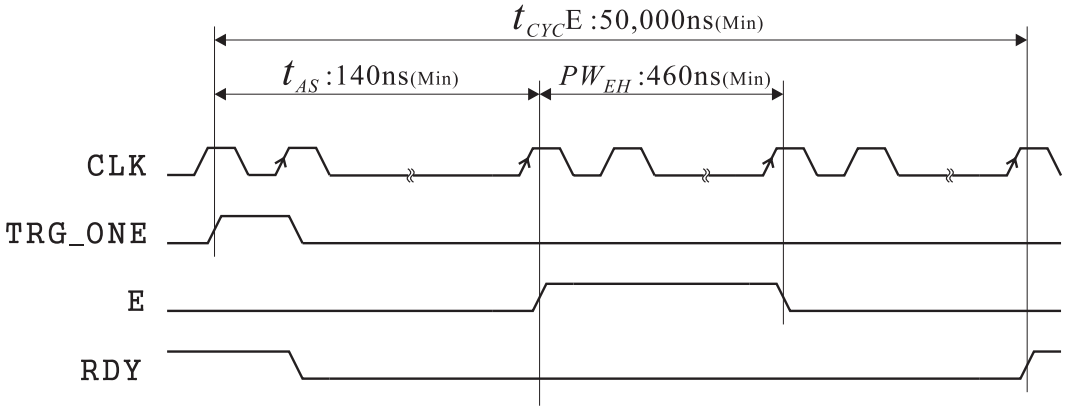


図 14: Timing Chart of SC2004C_EGENL

表 1: Pin Function Descriptions

Pin 名	信号方向	機能	備考
TRG_ONE	IN	トリガ入力	シングルショット・パルスを入力する必要あり。
CLK	IN	クロック入力	各周波数に合ったモジュールを選択する必要あり。
E	OUT	Enable パルス出力	LCD の “E” 端子に接続。
RDY	OUT	レディ信号出力	この信号が HIGH になったら、次のトリガ入力が可能。従って、この信号のポジティブ・エッジでインストラクション・メモリのアドレスカウンタなどをインクリメントする。ただし、インストラクションによっては、より長い処理時間を要するので注意が必要。

表 2: Time Measurment Results

モジュール名	ベースクロック周波数	t_{AS}[ns]	PW_{EH}[ns]	$t_{CYC}E$[μs]
SC2004C_EGENL_07P14M	1MHz	2000	4000	357.0
	5MHz	400	800	71.4
SC2004C_EGENL_14P28M	10MHz	200	700	71.4
	12.2(88)MHz	163	570	58.1
SC2004C_EGENL_21P43M	16MHz	188	625	67.0
	20MHz	150	500	53.6
SC2004C_EGENL_28P57M	25MHz	160	562	57.2
	27MHz	150	520	52.9
SC2004C_EGENL_35P71M	28.6(36)MHz	175	595	62.4
	30MHz	168	568	59.5
	35MHz	143	487	51.0
SC2004C_EGENL_49P99M	36MHz	195	640	69.4
	40MHz	175	575	62.5
	45MHz	155	512	55.5
SC2004C_EGENL_71P42M	50MHz	200	660	71.4
	60MHz	168	552	59.5
	70MHz	143	475	51.0

表 2 に各モジュールの動作周波数範囲内のベースクロックで動作させた場合の、各時間の測定結果を示します。測定条件は下記の通りです。25MHz 以上のベースクロックには Design Gateway 社（http://www.dgway.com/）製の VariClock という可変オシレータを使用しました。ただ、これは設定可能周波数が 1MHz ステップなので、28.6(36)MHz などは従来の DIP タイプのオシレータを用いてテストしました。

測定条件

Target Device	Virtex-E XCV300E-6PQ240C
OSC	～24, 28.6(36)MHz: XTAL-OSC (DIP-14) 25～70MHz: DG VC250M14P
CLKDLL	Unused
ISE Version	6.1.03i
Oscilloscope	Agilent 54622D

また、下記の条件で各モジュールの回路規模を求めました。Map Report から得た結果を表 3 に示します。I/O などは作成せず、モジュールのみインプリメンテーションしました。ISE のオプション設定は、基本的に Map Properties の “Trim Unconnected Signals” を行わないようにする以外はデフォルト値のままです。

実装条件

Target Device	Virtex, Virtex-E, Spartan2
ISE Version	6.1.03i
Map Properties	Trim Unconnected Signals: Disable
Place & Route Properties	Effort Level: Standard
	Cost Table: 1
	Place and Route Mode: Normal Place and Route

表 3: Circuit Size Implementation Results

モジュール名	Slices	Gates	FFs	LUTs
SC2004C_EGENL_07P14M	12	236	16	18
SC2004C_EGENL_14P28M	14	268	17	22
SC2004C_EGENL_21P43M	17	316	20	26
SC2004C_EGENL_28P57M	17	324	21	26
SC2004C_EGENL_35P71M	19	340	23	26
SC2004C_EGENL_49P99M	19	356	25	26
SC2004C_EGENL_71P42M	25	406	29	29

17 HDL での インスタンシエーション方法

17.1 VHDL の場合

```
Component sc2004c_egenl_35p71m
  Port(
        TRG_ONE : in std_logic;
        CLK     : in std_logic;
        E       : out std_logic;
        RDY     : out std_logic
      );
end Component;

INSTANCE_NAME : sc2004c_egenl_35p71m
  Port map(
            TRG_ONE => TRG_ONE_IN,
            CLK     => INTERNAL_CLK,
            E       => E_OUT,
            RDY     => RDY_OUT
          );
```

17.2 Verilog-HDL の場合

```
module sc2004c_egenl_35p71m
 (TRG_ONE, CLK, E, RDY);
  input TRG_ONE;
  input CLK;
  output E;
  output RDY;
endmodule

sc2004c_egenl_35p71m INSTANCE_NAME
  (
   .TRG_ONE (TRG_ONE_IN),
   .CLK (INTERNAL_CLK),
   .E (EOUT),
   .RDY (RDY_OUT)
  );
```

18 使用許諾及び再配布に関して

“SC2004C_EGENL” Ver.1.0 のライセンスはアーカイブ・ファイル内に同封された “LICENSE” ファイルに記載されている SUSUBOX License Ver.1.0 の内容に準じます。SUSUBOX License Ver.1.0 は修正 BSD ライセンスを基に作られており、本モジュールを組み込んだ回路の公開などは強要していません。また、ライセンスを満たす限り、個人、商用を問わず自由に再配布が可能です。

本モジュールが有用であった場合、どんなところに利用されたか、また、感想などをを頂ければ幸いです。(連絡先：susu@susubox.org)

by すすたわり

SUSUBOX™

Datasheet of “SC2004C_MC” Ver.0.8
LCD Module SC2004C Memory Controller

Document Ver.0.8.1J (2004.01.24) Made by Susutawari

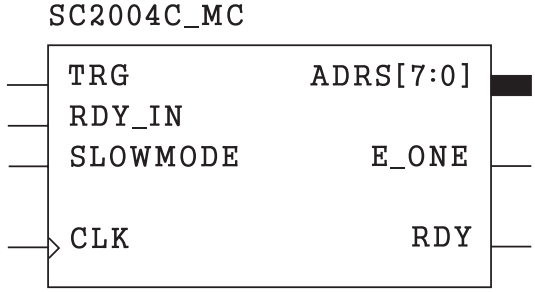


図 15: The symbol of SC2004C_MC

19 機能概要

SUNLIKE 社製のキャラクタ・ディスプレイ LCD である、 SC2004C, SC1602B, 及び、セイコーインスツルメンツ社製の M1632, L1634 などをドライブするためのモジュールの一部で、TOP モジュールである、SC2004C_4DRIVER(L) 中で使われています。LCD をドライブするには他に、SC2004C_EGEN(L), SC2004C_4INIT, SC2004_8TO4CC などが必要です。本モジュールは前述の LCD に表示する文字列や、インストラクションを記録したメモリからデータを読み出し、LCD に渡すためのコントローラです。アドレス空間が 8bit あるので、8bit データ長のときは 256 インストラクション（256 文字）、4bit データ長のときは 128 インストラクション（128 文字）まで出力可能です。SC2004C は 20 文字 ×4 行なので、80 文字＋ 176/48 インストラクションの出力が可能です。また、２種類の出力速度を持っており、タイプ入力のような速度の文字表示ができます。

関連モジュール
SC2004C_4DRIVERL, SC2004C_4DRIVER, SC2004C_EGENL, SC2004C_EGEN, SC2004C_4INIT, SC2004C_4ROM, SC2004C_8TO4CC

なお、LCD の規定にあったパルスを生成するため、使用するベースクロックに合わせてモジュールを選択する必要があります。規格はセイコーインスツルメンツ社（http://www.sii.co.jp/）の L1634 の「キャラクタタイプ液晶表示モジュール取り扱い説明書」（AN.No.CLCM-137）を参考にしています。

機能仕様

動作確認済み LCD	SC2004C rev.0
データ出力サイクル (Min)	1.64ms/14.5ms
メモリアドレス空間	8bit
8bit 長のとき	256 Instructions
4bit 長のとき	128 Instructions

実装仕様

Target Device	Virtex, Virtex-E, Spartan2.
Source Type	EDIF 2.0, PDF, ViewDraw Sch.
Circuit Size	34〜37 Slices

モジュール名

Base CLK Freq.	Module Name
〜7.14MHz 以下	SC2004C_MC_07P14M
7.14MHz より速く 〜14.28MHz 以下	SC2004C_MC_14P28M
14.29MHz より速く 〜21.43MHz 以下	SC2004C_MC_21P43M
21.44MHz より速く 〜28.57MHz 以下	SC2004C_MC_28P57M
28.58MHz より速く 〜35.71MHz 以下	SC2004C_MC_35P71M
35.71MHz より速く 〜49.99MHz 以下	SC2004C_MC_49P99M
49.99MHz より速く 〜71.42MHz 以下	SC2004C_MC_71P42M

※注：本モジュールは使用するベースクロックに合わせて選択する必要があります。上記の動作周波数を超える、高いベースクロックを使用した場合は動作しない可能性がありますが、低い周波数を入力した場合は、処理が遅くなるだけで論理動作には問題ありません。このため必要に応じてより高い周波数向けのモジュールを選択することで、動作マージンを大きくすることが可能です。

20 動作解説

図 16 に主なピンのタイミング・チャートを示します。初期状態は “E_ONE” が LOW, “RDY” が HIGH, “ADRS[7:0]”=00h です。“TRG” 入力があると、“RDY” は LOW となり、$t_{CYC}E$（Normal=1.64ms 以上, Slow=27.5ms 以上）間隔で “E_ONE” が出力されます。2 回目以降、“E_ONE” の出力と同時に “ADRS[7:0]” がインクリメントされ、FFh になった後、$t_{CYC}E$ 待って “RDY” が HIGH に戻ります。

ベースクロックに合ったモジュールを使用した場合、出力されるパルスは、LCD（L1634）のデータ・シート（AN.No.CLCM-137）で定められている規定時間以上でかつ、SC2004C や M1632 で動作可能と思われる時間以上となるように設計されています。

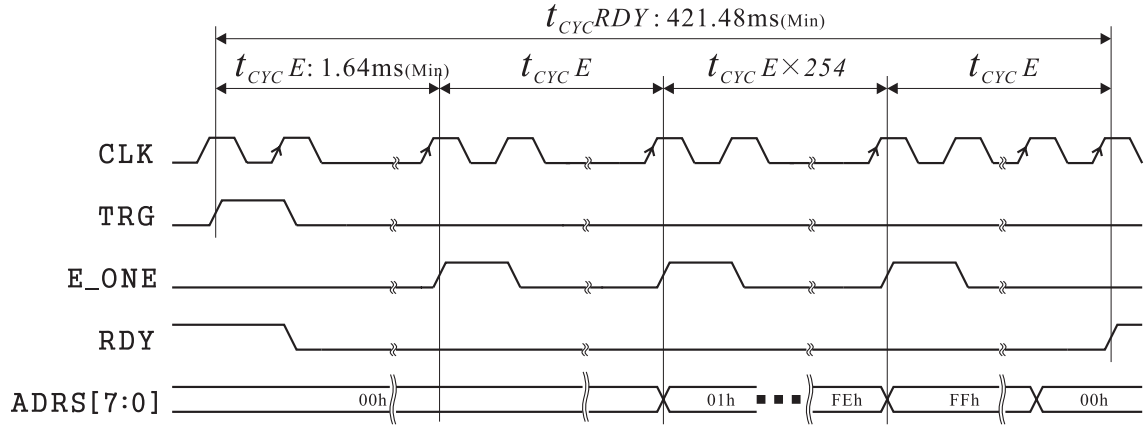


図 16: Timing Chart of SC2004C_MC

表 1: Pin Function Descriptions

Pin 名	信号方向	機能	備考
TRG	IN	トリガ入力	メモリ・データ出力開始トリガ入力。 RDY, 及び RDY_IN が HIGH のときに有効。
RDY_IN	IN	レディ入力	Enable Pulse Generator の RDY 信号を入力。
SLOWMODE	IN	出力速度設定	LOW のときは通常速度、HIGH のときは低速で出力。
CLK	IN	クロック入力	各周波数に合ったモジュールを選択する必要あり。
ADRS[7:0]	OUT	メモリ・アドレス出力	メモリのアドレス端子に接続。00h〜FFh まで一定間隔でインクリメントされる。
E_ONE	OUT	Enable パルス用トリガ出力	Enable Pulse Generator の “TRG” 入力に接続。シングルショット・パルス出力。
RDY	OUT	レディ信号出力	この信号が HIGH になったら、次のトリガ入力が可能。従って、この信号のポジティブ・エッジで出力の完了がわかる。

表 2: Time Measurment Results

モジュール名	ベースクロック周波数	$t_{CYC}E$[ms] (Normal*)	$t_{CYC}RDY$[ms] (Normal*)	$t_{CYC}E$[ms] (Slow*)	$t_{CYC}RDY$[s] (Slow*)
SC2004C_MC_07P14M	1MHz	11.78	3,027	104.5	26.84
	5MHz	2.36	605	20.9	5.37
SC2004C_MC_14P28M	10MHz	2.36	605	20.9	5.37
	12.2(88)MHz	1.92	493	17.0	4.37
SC2004C_MC_21P43M	16MHz	2.20	565	19.4	5.00
	20MHz	1.76	452	15.6	4.00
SC2004C_MC_28P57M	25MHz	1.88	484	16.7	4.29
	27MHz	1.74	448	15.5	3.98
SC2004C_MC_35P71M	28.6(36)MHz	2.07	533	18.3	4.70
	30MHz	1.98	509	17.5	4.49
	35MHz	1.70	436	15.0	3.85
SC2004C_MC_49P99M	36MHz	2.27	585	20.3	5.20
	40MHz	2.05	527	18.2	4.68
	45MHz	1.82	468	16.2	4.16
SC2004C_MC_71P42M	50MHz	2.38	611	20.7	5.33
	60MHz	1.98	509	17.3	4.44
	70MHz	1.70	436	14.8	3.81

*Normal: SLOWMODE = LOW(0), Slow: SLOWMODE = HIGH(1).

表 2 に各モジュールの動作周波数範囲内のベースクロックで動作させた場合の、各時間の測定結果を示します。測定条件は下記の通りです。25MHz 以上のベースクロックには Design Gateway 社（http://www.dgway.com/）製の VariClock という可変オシレータを使用しました。ただ、これは設定可能周波数が 1MHz ステップなので、28.6(36)MHz などは従来の DIP タイプのオシレータを用いてテストしました。

測定条件

Target Device	Virtex-E XCV300E-6PQ240C
OSC	～24, 28.6(36)MHz: XTAL-OSC (DIP-14) 25～70MHz: DG VC250M14P
CLKDLL	CLKDLLE
ISE Version	6.1.03i
Oscilloscope	Agilent 54622D

また、下記の条件で各モジュールの回路規模を求めました。Map Report から得た結果を表 3 に示します。I/O などは作成せず、モジュールのみインプリメンテーションしました。ISE のオプション設定は、基本的に Map Properties の “Trim Unconnected Signals” を行わないようにする以外はデフォルト値のままです。

実装条件

Target Device	Virtex, Virtex-E, Spartan2
ISE Version	6.1.03i
Map Properties	Trim Unconnected Signals: Disable
Place & Route Properties	Effort Level: Standard
	Cost Table: 1
	Place and Route Mode: Normal Place and Route

表 3: Circuit Size Implementation Results

モジュール名	Slices	Gates	FFs	LUTs
SC2004C_MC_07P14M	34	546	33	47
SC2004C_MC_14P28M	34	566	34	49
SC2004C_MC_21P43M	36	600	36	52
SC2004C_MC_28P57M	36	600	36	52
SC2004C_MC_35P71M	35	594	36	51
SC2004C_MC_49P99M	35	600	36	52
SC2004C_MC_71P42M	37	600	36	52

21 HDL でのインスタンシエーション方法

21.1 VHDL の場合

```
Component sc2004c_mc_35p71m
  Port(
        TRG      : in std_logic;
        RDY_IN   : in std_logic;
        SLOWMODE : in std_logic;
        CLK      : in std_logic;
        ADRS     : out std_logic_vector
                   (7 downto 0);
        E_ONE    : out std_logic;
        RDY      : out std_logic
      );
end Component;

INSTANCE_NAME : sc2004c_mc_35p71m
  Port map(
            TRG      => TRG_IN,
            RDY_IN   => MC_RDY_IN,
            SLOWMODE => MC_SLOWMODE,
            CLK      => INTERNAL_CLK,
            ADRS     => ROM_ADRS,
            E_ONE    => MC_E_ONE,
            RDY      => MC_RDY
          );
```

21.2 Verilog-HDL の場合

```
module sc2004c_mc_35p71m
 (TRG, RDY_IN, SLOWMODE, CLK, ADRS, E_ONE, RDY);
  input TRG;
  input RDY_IN;
  input SLOWMODE;
  input CLK;
  output [7:0] ADRS;
  output E_ONE;
  output RDY;
endmodule

sc2004c_mc_35p71m INSTANCE_NAME
  (
   .TRG (TRG_IN),
   .RDY_IN (MC_RDY_IN),
   .LOWSPEED (MC_SLOWMODE),
   .CLK (INTERNAL_CLK),
   .ADRS (ROM_ADRS),
   .E_ONE (MC_E_ONE),
   .RDY (MC_RDY)
  );
```

22 使用許諾及び再配布に関して

“SC2004C_MC” Ver.0.8 のライセンスはアーカイブ・ファイル内に同封された “LICENSE” ファイルに記載されている SUSUBOX License Ver.1.0 の内容に準じます。SUSUBOX License Ver.1.0 は修正 BSD ライセンスを基に作られており、本モジュールを組み込んだ回路の公開などは強要していません。また、ライセンスを満たす限り、個人、商用を問わ

ず自由に再配布が可能です。

本モジュールが有用であった場合、どんなところに利用されたか、また、感想などをを頂ければ幸いです。（連絡先：susu@susubox.org）

ｂｙ すすたわり

SUSUBOX™

Datasheet of “SC2004C_4ROM” Ver.1.0 LCD Module SC2004C Instruction ROM (4 bit × 256 Version)

Document Ver.1.0.1J (2004.01.28) Made by Susutawari

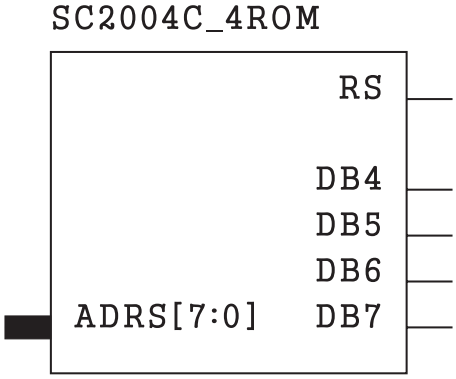


図 17: The symbol of SC2004C_4ROM

23 機能概要

SUNLIKE 社製のキャラクタ・ディスプレイ LCD である、 SC2004C, SC1602B, 及び、セイコーインスツルメンツ社製の M1632, L1634 などをドライブするためのモジュール “SC2004C_4DRIVER(L)” や “SC2004C_MC” で使用する、書き込み済み？の ROM のサンプルです。（LCD への出力専用なため、LCD の R/W 端子は LOW（0）に保つ必要があります。）主に DxDesigner（ViewDraw）で利用するためのモジュールですが、データのコーディングが大変なので、ViewDraw でも HDL で記述してモジュール化したものを埋め込んだ方がいいかもしれません。或いは EDIF を直接書き換えるか ‥

本モジュールを使う場合は、いったん別の名前でコピーして、内部 RAM の “INIT” 属性の値を変更して使います。

関連モジュール
SC2004C_4DRIVERL, SC2004C_4DRIVER, SC2004C_MC

機能仕様

動作確認済み LCD	SC2004C rev.0
インターフェースデータ長	4 bit
メモリ容量	4bit × 256 (1,024bit)
格納命令数	128 Instructions

実装仕様

Target Device	Virtex, Virtex-E, Spartan2.
Source Type	EDIF 2.0, PDF, ViewDraw Sch.
Circuit Size	44 Slices

モジュール名

Base CLK Freq.	Module Name
–	SC2004C_4ROM

表 1: Pin Function Descriptions

Pin 名	信号方向	機能	備考
ADRS[7:0]	IN	アドレス入力	“SC2004C_DRIVER(L)”, “SC2004C_MC” の ADRS[7:0] 出力を接続する。
RS	OUT	データ出力	RS データの出力。 “SC2004C_DRIVER(L)”, “ROM_RSIN” などに接続する。
DB4〜7	OUT	データ出力	DB4〜7 データの出力。“SC2004C_DRIVER(L)”, の “ROM_DB4〜DB7IN” などにそれぞれ接続する。

※ “SC2004C_DRIVER(L)” の “ROM_RWIN” は LOW（0）に固定する必要あり。

次の条件で各モジュールの回路規模を求めました。Map Report から得た結果を表 2 に示します。I/O などは作成せず、モジュールのみインプリメンテーションしました。ISE のオプション設定は、基本的に Map Properties の “Trim Unconnected Signals” を行わないようにする以外はデフォルト値のままです。

実装条件

Target Device	Virtex, Virtex-E, Spartan2
ISE Version	6.1.03i
Map Properties	Trim Unconnected Signals: Disable
Place & Route Properties	Effort Level: Standard
	Cost Table: 1
	Place and Route Mode: Normal Place and Route

表 2: Circuit Size Implementation Results

モジュール名	Slices	Gates	FFs	LUTs	32×1 RAMs
SC2004C_4ROM	44	10,528	0	8	80

24 HDLでのインスタンシエーション方法

24.1 VHDLの場合

```
Component sc2004c_4rom
  Port(
        ADRS      : in std_logic_vector
                    (7 downto 0);
        RS        : out std_logic;
        RW        : out std_logic;
        DB4       : out std_logic;
        DB5       : out std_logic;
        DB6       : out std_logic;
        DB7       : out std_logic
      );
end Component;

INSTANCE_NAME : sc2004c_4rom
  Port map(
           ADRS      => ROM_ADRS,
           RS        => ROM_RS,
           RW        => ROM_RW,
           DB4       => ROM_DB4,
           DB5       => ROM_DB5,
           DB6       => ROM_DB6,
           DB7       => ROM_DB7
          );
```

24.2 Verilog-HDLの場合

```
module sc2004c_4rom
 (ADRS, RS, DB4, DB5, DB6, DB7);

        input [7:0] ADRS;
        output RS;
        output DB4;
        output DB5;
        output DB6;
        output DB7;

endmodule
```

```
sc2004c_4rom INSTANCE_NAME
        (
         .ADRS (ROM_ADRS),
         .RS (ROM_RS),
         .DB4 (ROM_DB4),
         .DB5 (ROM_DB5),
         .DB6 (ROM_DB6),
         .DB7 (ROM_DB7)
        );
```

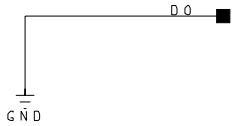
25 使用許諾及び再配布に関して

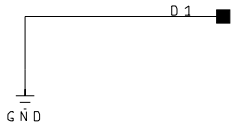
“SC2004C_4ROM” Ver.1.0 のライセンスはアーカイブ・ファイル内に同封された “LICENSE” ファイルに記載されている SUSUBOX License Ver.1.0 の内容に準じます。SUSUBOX License Ver.1.0 は修正 BSD ライセンスを基に作られており、本モジュールを組み込んだ回路の公開などは強要していません。また、ライセンスを満たす限り、個人、商用を問わず自由に再配布が可能です。

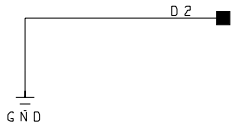
本モジュールが有用であった場合、どんなところに利用されたか、また、感想などをを頂ければ幸いです。（連絡先：susu@susubox.org）

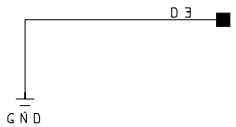
by すすたわり

SBASE__0H



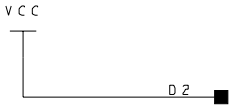






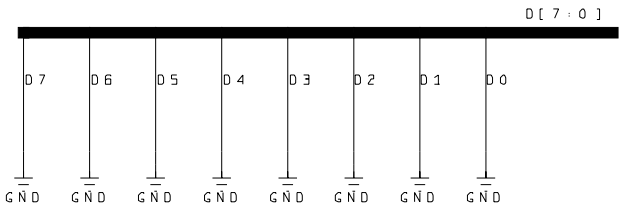
SUSUBOX™	
WWW.SUSUBOX.ORG	
Title: SBASE__0H	
Comments: Constant Value Generator (4bit Version)	
Date: 2004.02.03	Ver: 1.0
(C)2003-2004 N.Aibe	Rev:

SBASE__FH



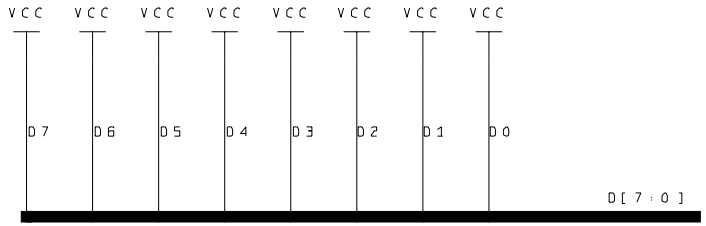
SUSUBOX™	
WWW.SUSUBOX.ORG	
Title: SBASE__FH	
Comments: Constant Value Generator (4bit Version)	
Date: 2004.02.03	Ver: 1.0
(C)2003-2004 N.Aibe	Rev:

SBASE_00H



SUSUBOX™		
WWW.SUSUBOX.ORG		
Title: SBASE_00H		
Comments: Constant Value Generator (8bit Version)		
Date: 2004.02.01		Ver: 1.0
		Rev:

SBASE_FFH



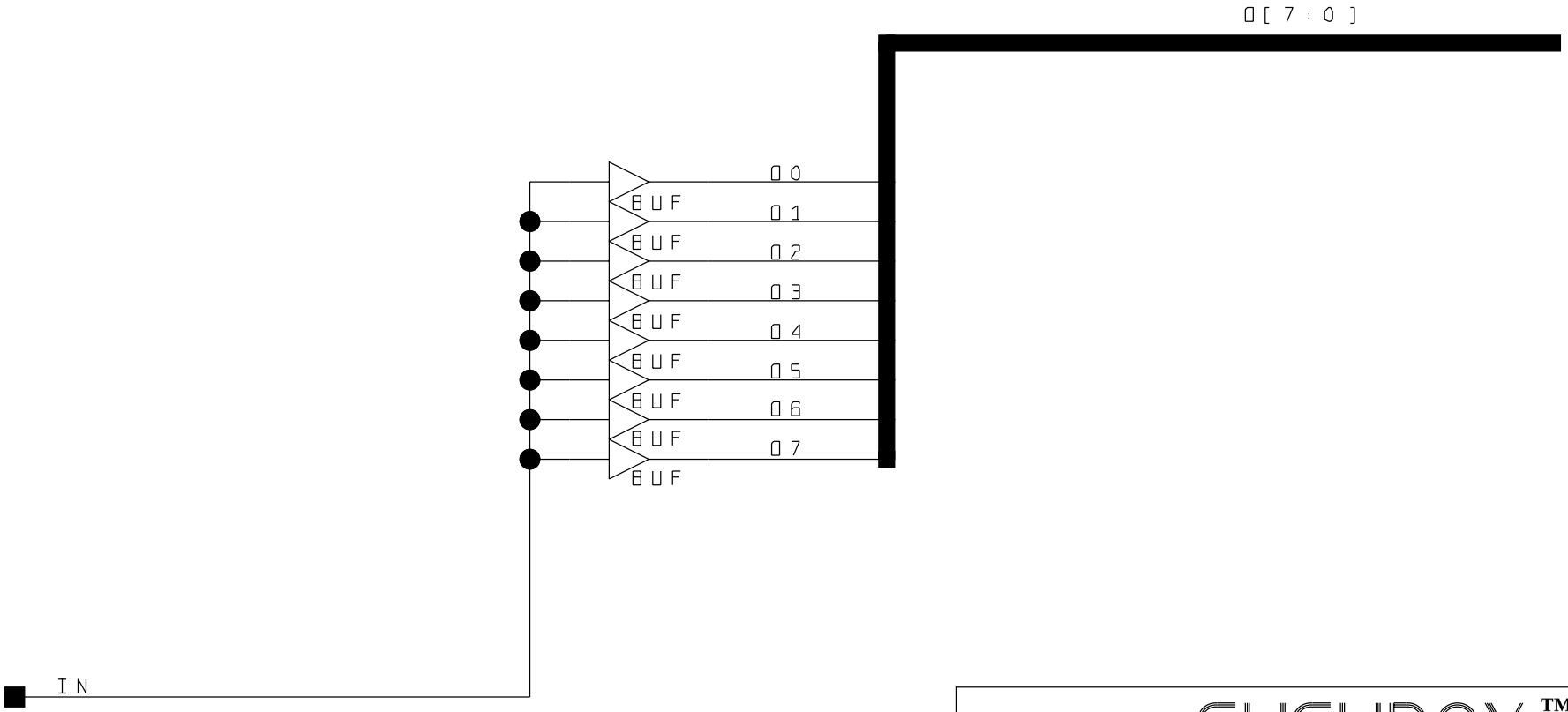
FFh = 255d = 11111111b

SUSUBOX™

WWW.SUSUBOX.ORG

Title:	SBASE_FFH	
Comments:	Constant Value Generator (8bit Version)	
Date:	2004.02.03	Ver: 1.0
	(C)2003-2004 N.Aibe	Rev:

SBASE_BUF1TO8 Ver.1.0



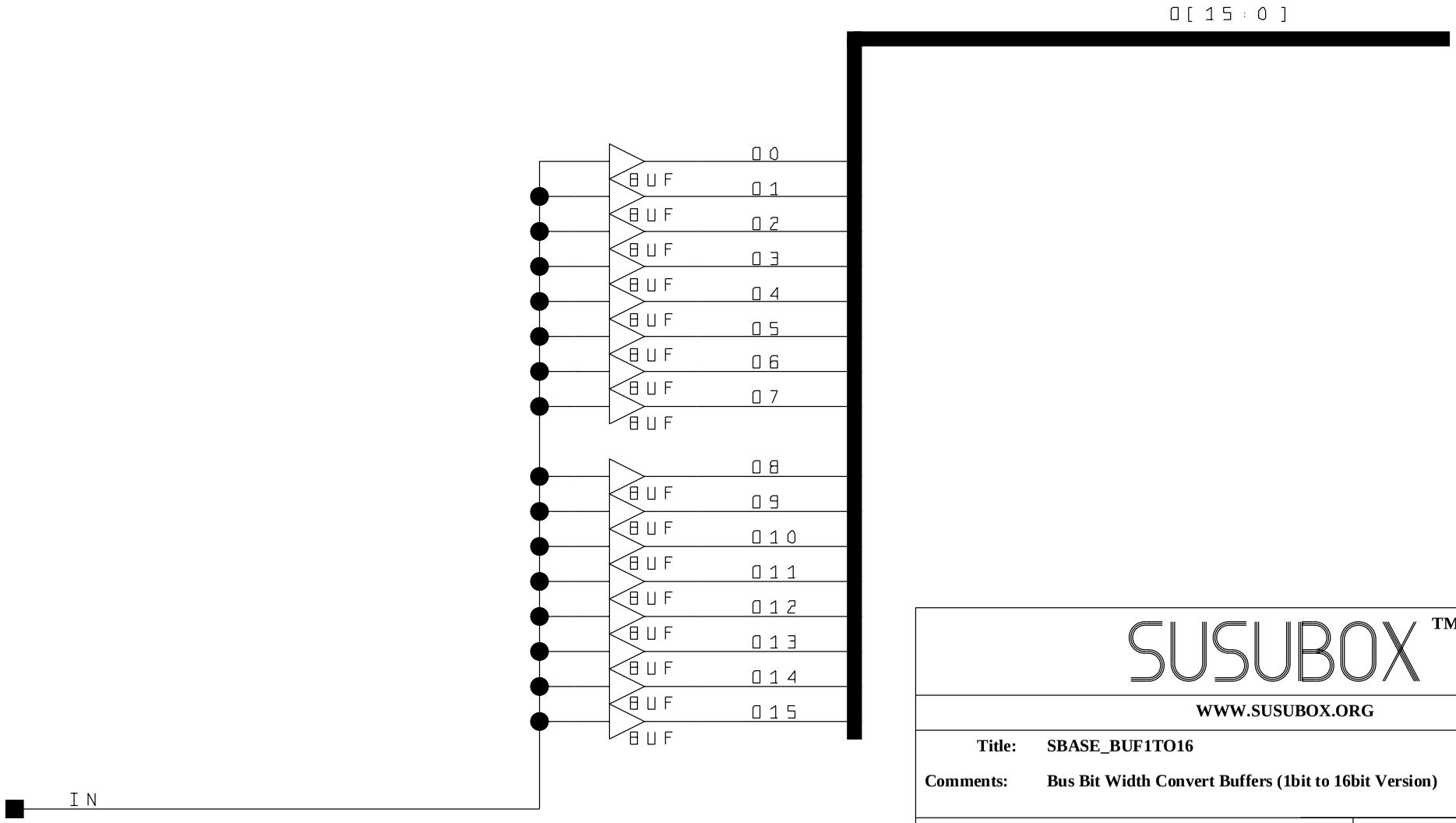
SUSUBOX™

WWW.SUSUBOX.ORG

Title:	**SBASE_BUF1TO8**
Comments:	**Bus Bit Width Convert Buffers (1bit to 8bit Version)**

Date: 2004.01.21	**Ver:** 1.0
(C)2003-2004 N.Aibe	**Rev:**

SBASE_BUF1TO16 Ver.1.0

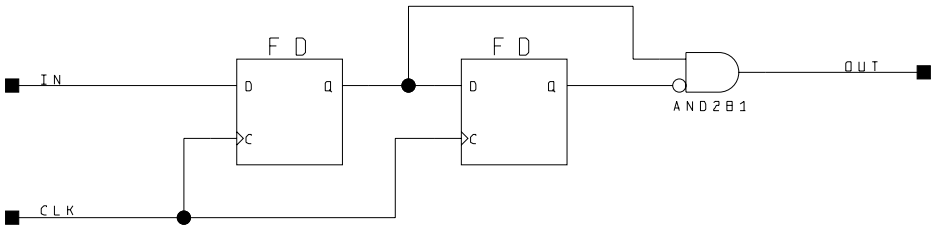


SUSUBOX™

WWW.SUSUBOX.ORG

Title:	**SBASE_BUF1TO16**	
Comments:	**Bus Bit Width Convert Buffers (1bit to 16bit Version)**	
Date:	**2004.01.21**	**Ver: 1.0**
	(C)2003-2004 N.Aibe	**Rev:**

SBASE_ONESHOT Ver.1.0



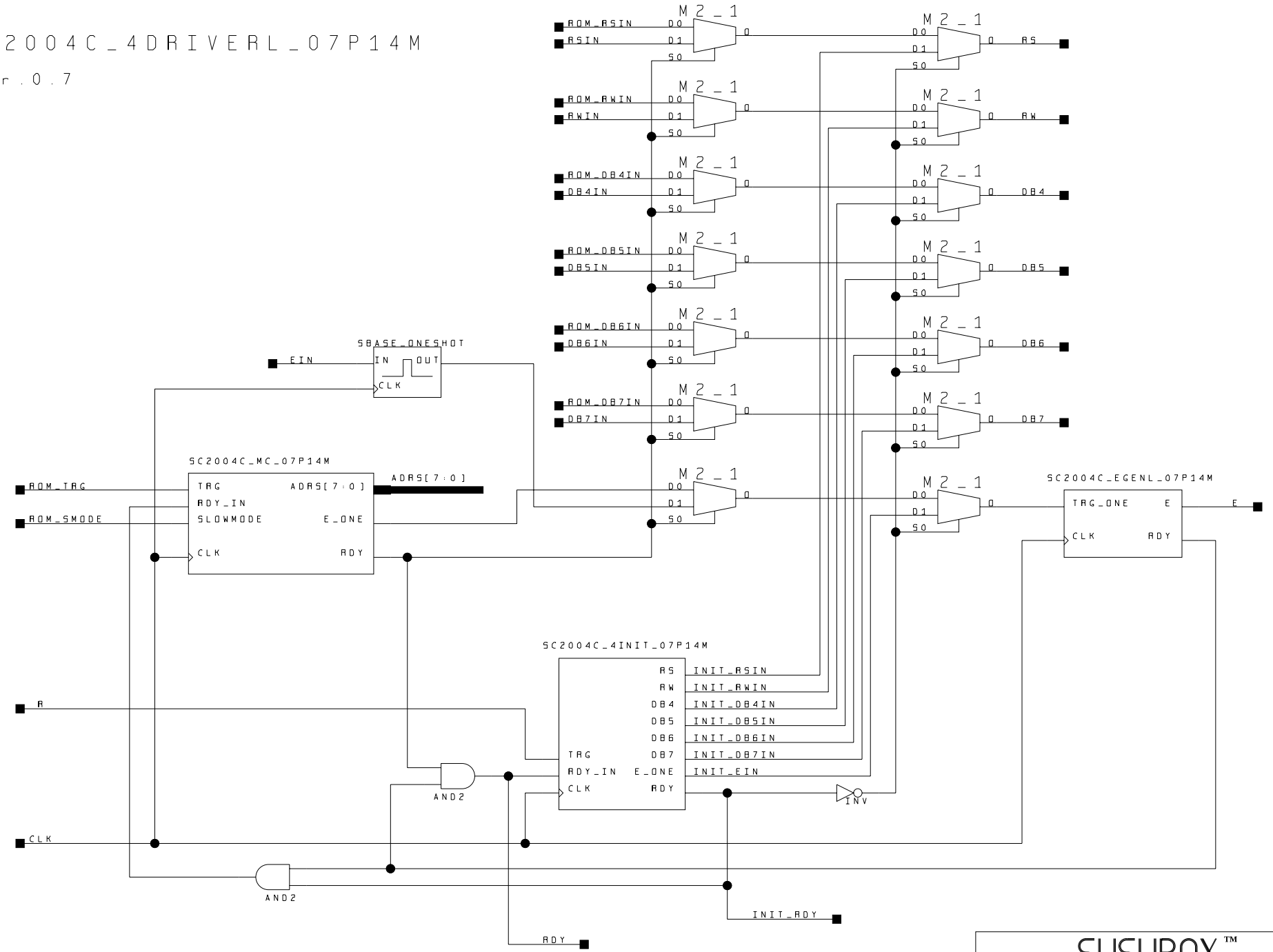
SUSUBOX™

WWW.SUSUBOX.ORG

Title: SBASE_ONESHOT	
Comments: Single Shot Pulse Generator	
Date: 2004.01.20	Ver: 1.0
(C)2003-2004 N.Aibe	Rev:

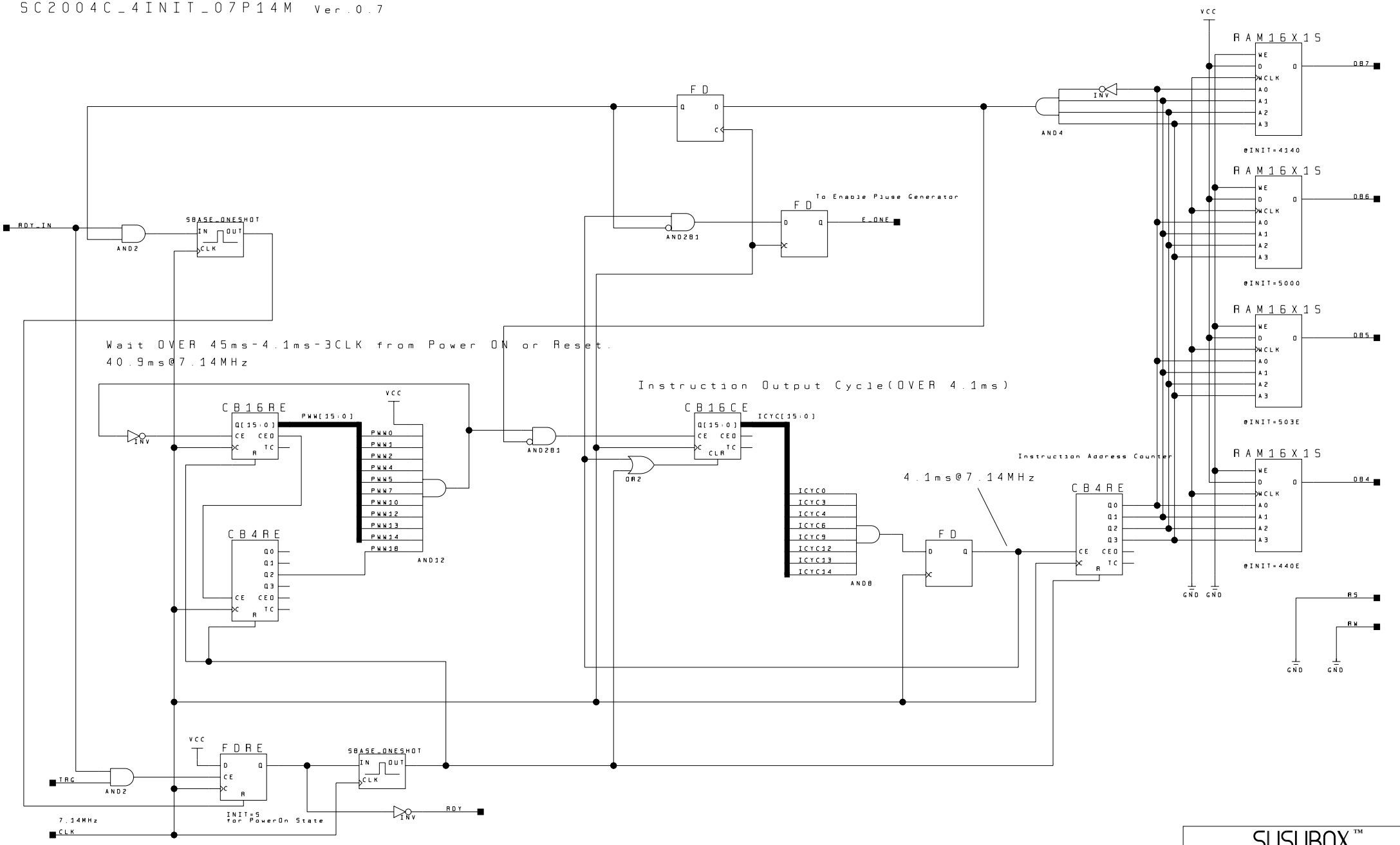
SC2004C_4DRIVERL_07P14M

Ver.0.7



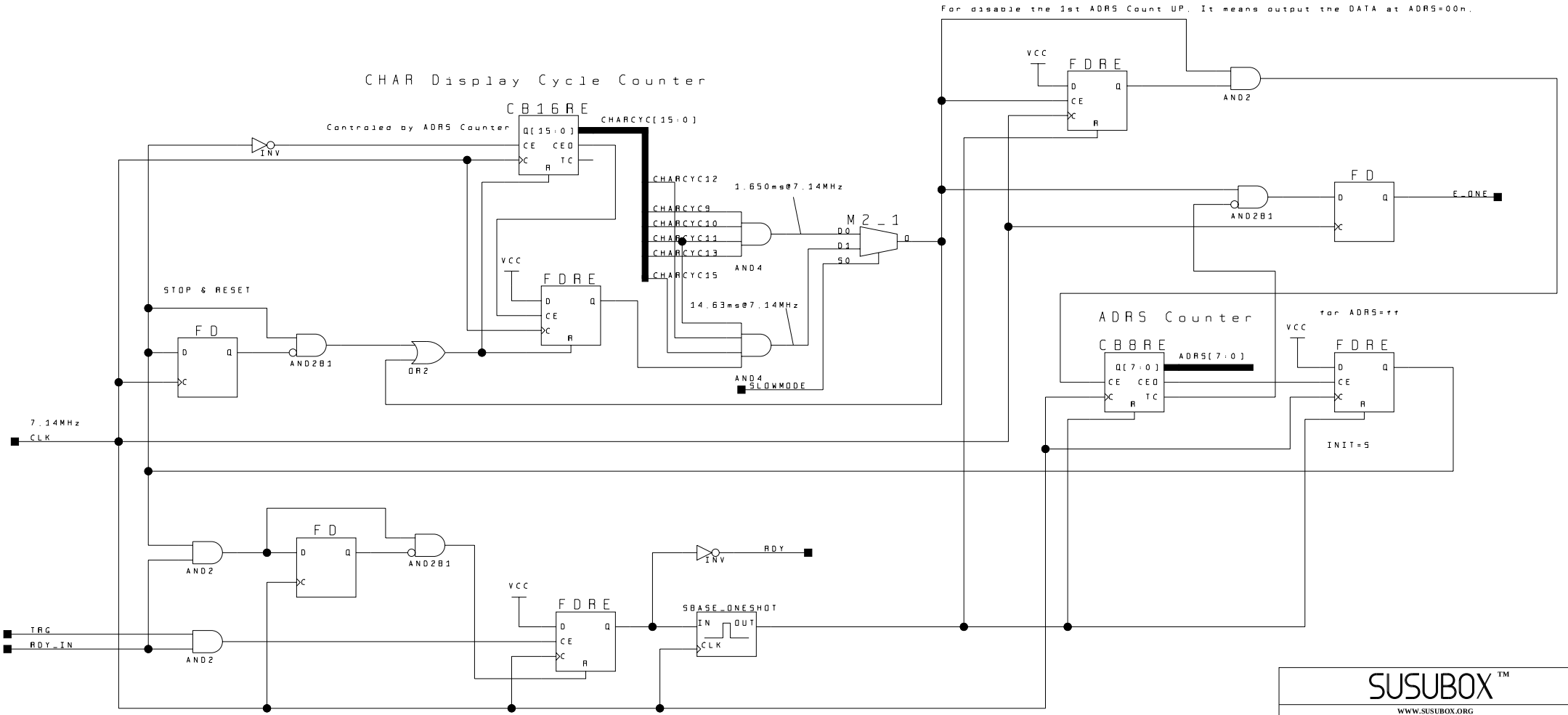
SUSUBOX™	
WWW.SUSUBOX.ORG	
Title:	SC2004C_4DRIVERL_07P14M
Comments:	LCD Module SC2004C DRIVER (4 bit length, Long Cycle Version)
Date: 2004.01.27	Ver: 0.7
(C)2003-2004 N.Aibe	Rev:

SC2004C_4INIT_07P14M Ver.0.7



SUSUBOX™		
WWW.SUSUBOX.ORG		
Title:	SC2004C_4INIT_07P14M	
Comments:	LCD Module SC2004C Initializer	(4bit Data Interface Version)
Date:	2004.01.23	Ver: 0.7
(C)2003-2004 N.Aibe		Rev:

SC2004C_MC_07P14M Ver.0.8

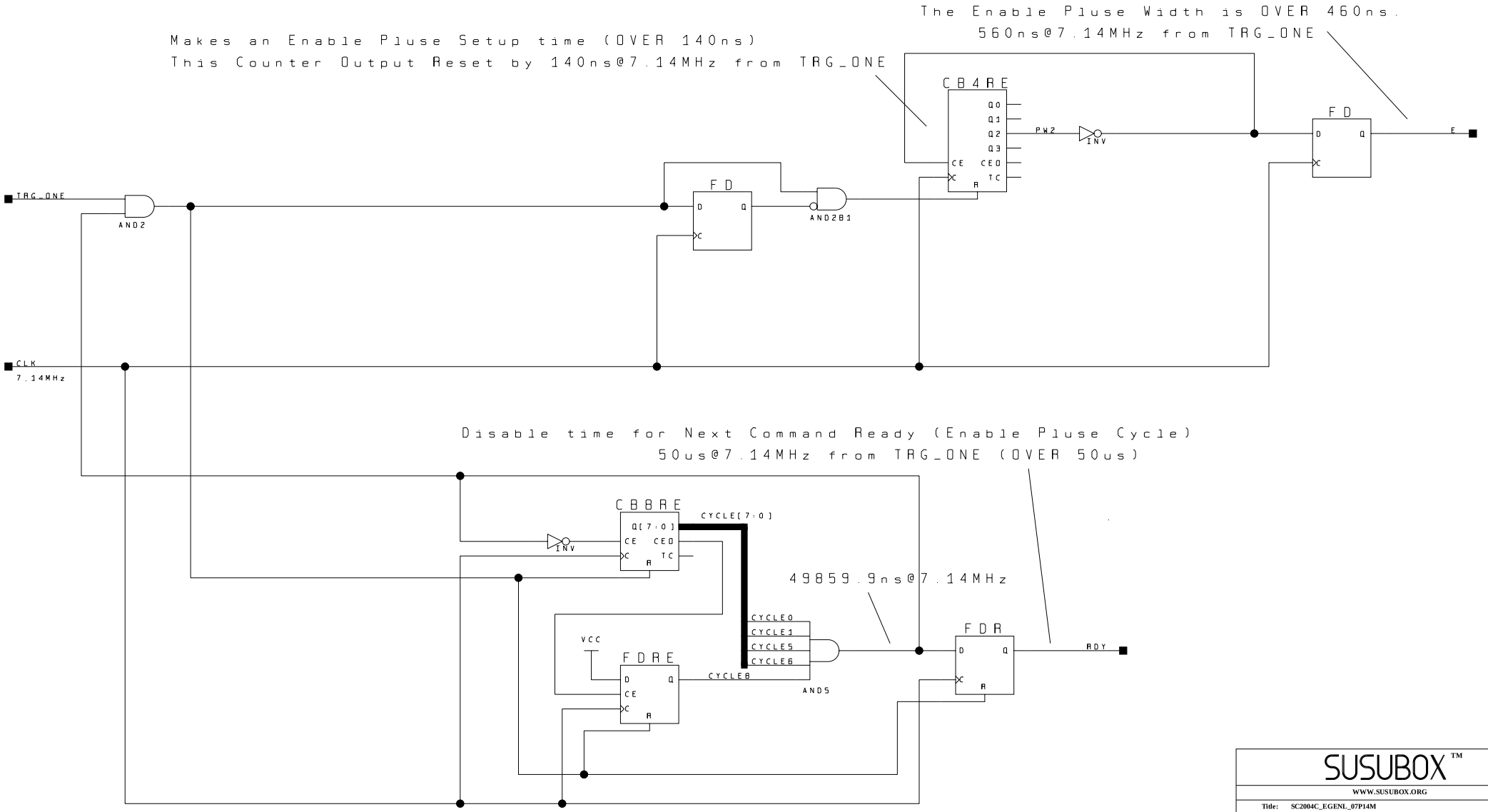


SUSUBOX™

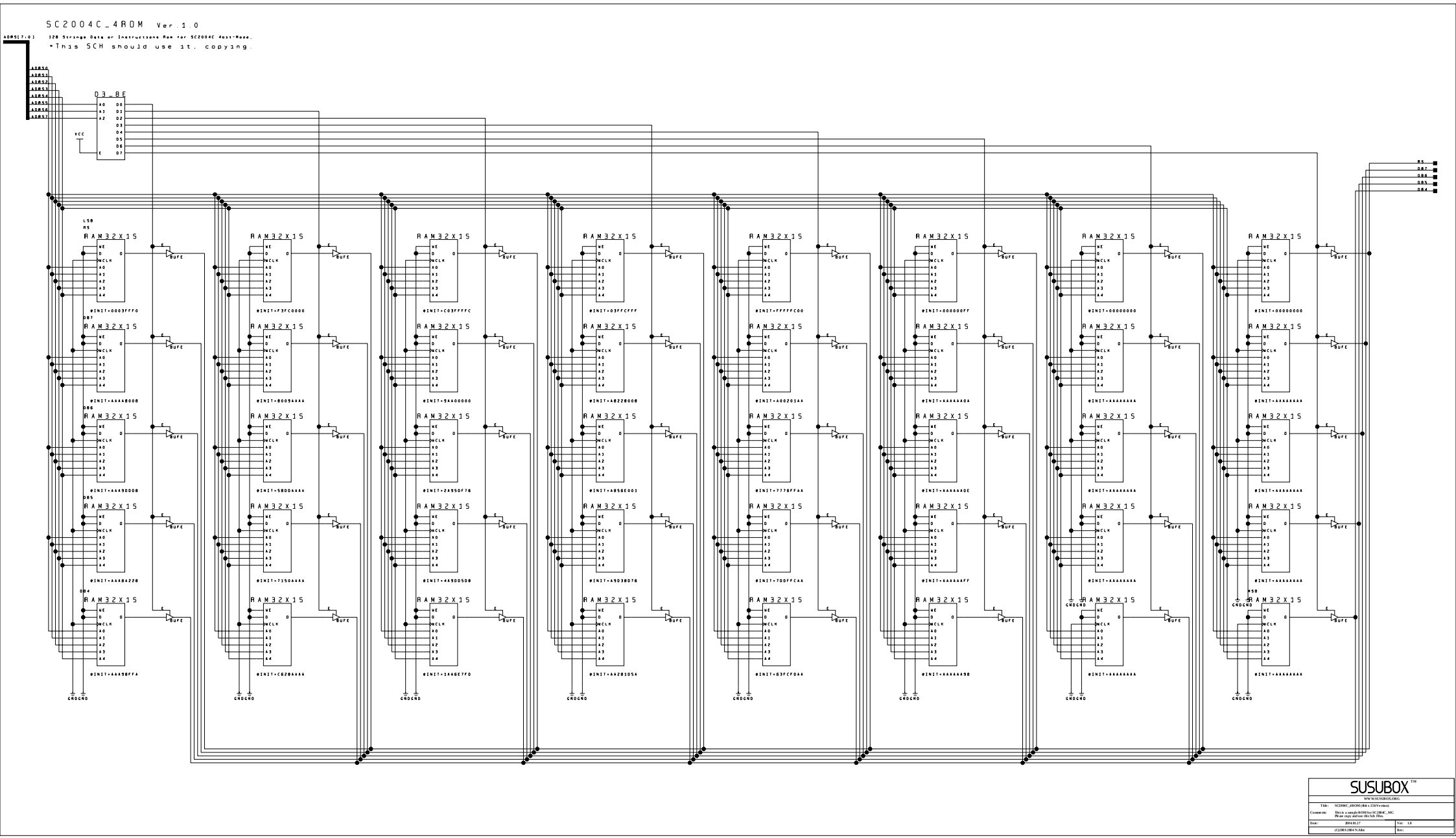
WWW.SUSUBOX.ORG

Title:	SC2004C_MC_07P14M
Comments:	LCD Module SC2004C Memory Controller
Date: 2004.01.26	Ver: 0.8
(C)2003-2004 N.Aibe	Rev:

SC2004C_EGENL_07P14M Ver.1.0



SUSUBOX™	
WWW.SUSUBOX.ORG	
Title: SC2004C_EGENL_07P14M	
Comments: LCD Module SC2004C Enable Pulse Generator (Long Cycle Version)	
Date: 2004.01.20	Ver: 1.0
(C)2003-2004 N.Aibe	Rev:



SUSUBOX™

WWW.SUSUBOX.ORG

Title:	SC2004C_4ROM (4bit x 256Version)
Comments:	This is a sample ROM for SC2004C_MC. Please copy and use this Sch' Files.
Date:	2004.01.27
	Ver: 1.0
(C)2003-2004 N.Aibe	Rev:

著者略歴

相部 範之 （あいべ のりゆき）

1975 年　東京生まれ
1994 年　多摩大学附属 聖ヶ丘高等学校 卒業
同年　工学院大学 1 部 電子工学科 電子工学コース 入学
1999 年　工学院大学 1 部 電子工学科 電子工学コース 卒業
同年　理化学研究所 脳科学総合研究センター
脳型デバイス・ブレインウェイ・グループ
脳創生デバイス研究チーム テクニカル・スタッフ
2000 年　同研究所 退所
同年　筑波大学 大学院 修士課程 理工学研究科 理工学専攻 入学
2002 年　筑波大学 大学院 修士課程 理工学研究科 理工学専攻 修了
同年　筑波大学 大学院 博士課程 システム情報工学研究科
コンピュータ・サイエンス専攻 3 年次編入学
2005 年　筑波大学 大学院 博士課程 システム情報工学研究科
コンピュータ・サイエンス専攻 修了

本論文は Microsoft Corp. Microsoft Windows 2000 上で，宮下尚 Meadow 1.15 (GNU Emacs 20.7.1 (i386-msvc-nt5.0.2195))，D.E.Knuth. pLaTeX2e （pTeX, Version 3.14159-p3.0.5 (sjis) (Web2C 7.3.11))，奥村晴彦 jsbook.cls, 井上浩一 mediabb.sty v1.8, David Carlisle Sebastian Rahtz graphicx.sty v1.0f, ASCII Corp. ascmac.sty, 大島利雄 (SHIMA) dviout for Windows Ver.3.14, dvipdfmx prj. team dvipdfmx-20021031, Adobe Systems Inc. Adobe Photoshop 6.0.1J Educational Version, Adobe Acrobat 5.0.5, 及び，Corel Corp. Corel DRAW Ver.11.640 によって作成しました.

印刷・製本所：株式会社オーエム (http://www.takuhaiprint.com/)