Vol.2015-SLDM-173 No.23
2015/12/2

道路速度標識認識システムの Rapid Prototyping Platform への実装

佐藤光　Anh-Tuan Hoang　小出哲士　岡本拓巳

広島大学 ナノデバイス・バイオ融合科学研究所，〒739-8527 東広島市鏡山 1-4-2

E-mail: {sato-hikaru, anhtuan, koide } @hiroshima-u.ac.jp

あらまし　本稿では，先進運転支援システム （ADAS: Advanced Driver Assistance System） で必要とされている道路速度標識認識システムのハードウェア実装について報告する．我々はこれまでに，標識認識のためリアルタイム処理を実現するハードウェア向けアルゴリズムとアーキテクチャを開発している．本研究では，標識認識処理の中で, 道路速度標識候補領域の抽出のため矩形パターンマッチング処理（RPM: Rectangle Pattern Matching）を Rapid Prototyping Platform へ実装し，ホスト PC との連携によるハードウェアとソフトウェアの協調設計を行った．これにより開発期間の短縮とアルゴリズムの検証をインタラクティブに実機検証することで，開発期間の短縮とシステムの最適化が可能となった．

キーワード　先進運転支援システム （ADAS)，道路速度標識認識，ハードウェア・ソフトウェアの協調設計，Rapid Prototyping Platform

An Implementation of Speed Limit Traffic Sign Recognition System on Rapid Prototyping Platform

Hikaru SATO　Anh-Tuan HOANG　Tetsushi KOIDE　Takumi OKAMOTO

Research Institute for Nanodevice and Bio Systems (RNBS) , Hiroshima University

1-4-2 Kagamiyama , Higashi-Hiroshima , 739-8527 , Japan

E-mail: {sato-hikaru, anhtuan, koide }@hiroshima-u.ac.jp

Abstract　This paper presents a hardware implementation of a speed-traffic sign recognition system, which is one of important issues in practical and effective Advanced Driver Assistance System (ADAS). We have developed a hardware oriented algorithm and architecture for real-time speed traffic sign recognition. In this paper, we implemented the Rectangle Pattern Matching (RPM) processing in our speed-traffic sign recognition system on a rapid prototyping system and attempted the hardware / software co-design by a linux-based PC for software processing. From the implementation on the rapid prototyping system, the verification of the proposed algorithm can be easily and efficiently performed by using the interactive online testing and the optimization of the system can be achieved.

Keywords　ADAS, Speed Traffic-Sign Recognition, Hardware/Software Co-design, Rapid Prototyping Platform

1. 研究背景

近年，自動車普及台数増加による交通事故の増加が深刻化している [1]. 交通事故の主な原因として発見の遅れ，判断・操作ミスなど運転者によるものが挙げられる. これらの事故の要因を取り除くため，運転手の操作をアシストすることで事故そのものを防ぐ先進運転支援システム (ADAS: Advanced Driver Assistance System)が注目されている [2]. また，ADAS は車載向けアプリケーションの中でも，最も急速な成長を遂げている分野の一つである. 特に，車載カメラを通して道路標識を認識し，ドライバーに通知するシステムに関する研究が近年盛んである. その背景として，政府や保険会社などによって ADAS の大規模導入が推進されていることが挙げられ，図 1 に示すような速度標識認識や交差点警告表示，並びに車線キープなどもロードマップに示されている [3].

本研究では，ADAS システムの 1 つとして，車両前方に取り付けた単眼カメラからの画像から速度標識を検出して，標識の速度をリアルタイム（15 ～ 30 fps 以内）に読

Vol.2015-SLDM-173 No.23
2015/12/2

み取る組込みシステムの開発を目的としている．本稿では，我々の提案している道路速度標識認識ハードウェアアルゴリズム [4,5,6] に対して，標識検出候補領域を抽出するハードウェアアルゴリズムを Rapid Prototyping Platform へ実装したものについて報告する．第 2 節では速度標識認識アルゴリズムの概要について順を追って説明し，第 3 節ではそのアルゴリズムの前半の処理である矩形パターンマッチング(RPM)について詳細を説明する．第 4 節では，RPM のハードウェア実装方法について述べ，第 5 節でまとめと今後の課題について述べる．

2. 道路速度標識認識ハードウェア向けアルゴリズム

我々が提案している速度標識認識システムは，Xilinx Automotive Zynq 7020 に実装し，認識率 98%，スループット 60 fps 以上を達成している [4,5,6]．表 1 に提案しているシステムと他研究との認識率と処理速度の比較と結果を示す．図 2 はシステムのフローチャートを示す．システムは大きく分けて 5 つの処理を経て道路速度標識を認識する．以下にその概要を説明する．

(1) 画像入力

本研究では，単眼車載カメラによって撮影される画像は Full HD グレースケール画像(1920×1080 pixel)の高解像度なものから VGA 画像(640×480)の低解像度のものまで同じ方法で処理することができる．提案するアルゴリズムは天候や照明環境といった変化する色情報は用いず，グレースケール画像であっても実用十分な認識性能を得ることが可能である．また，Full HD 画像に加え，図 3(a)，(c)に示すように，オリジナルサイズの画像(1920×1080 pixel) をダウンサンプリングした 640×360 pixel (約 23 万画素)の画像でも十分な制度で認識することが可能なため，低解像度画像に対しても適用可能であるという特徴を持つ．

(2) 矩形パターンマッチング (Rectangle Pattern Matching: RPM)

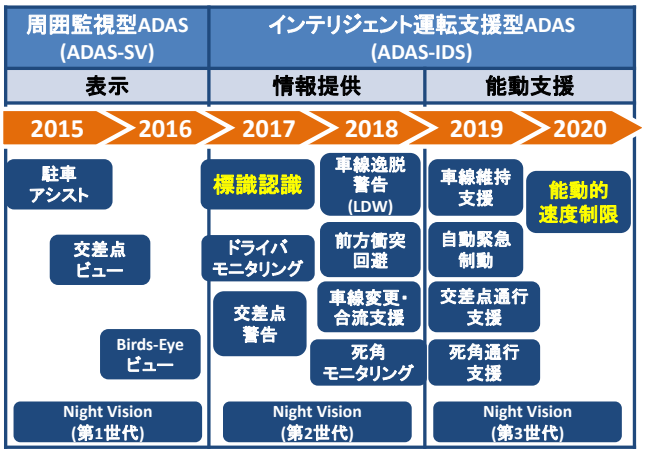


図 1. ヨーロッパにおける車載カメラベースの ADAS ロードマップ [3]．

提案するアルゴリズムの最初の処理は，入力のグレースケール画像の中から標識の候補領域を決定する矩形パターンマッチングである [4]．この RPM において画像をスキャンする領域を Scan Window(SW)と定義する．SW サイズは (20×20)～(50×50)pixel であり．31 種類のサイズ(20, 21, 22, ･･･,, 50 pixel)の SW を用いて，標識候補を検出する．

(3) 道路標識強調処理

標識部分の特徴量を際立たせるために，我々が新たに開発した標識強調フィルタ [4,5,6] を用いて，8 bit グレースケール画像を図 3 (b)，(d)に示すように 2 値画像に変換する．この標識強調フィルタにより，道路標識中のエッジ部分がさらに強調され，数字検出の精度が向上する．

(4) 円検出処理 (Circle Detection: CD)

円検出処理 [4,5,6] では，日本国内の道路速度標識が円形であることに着目して標識候補領域の絞り込みを行う．図 4 に示すように 9 つのブロックに SW サイズを分割し，ブロックごとにエッジ方向を抽出し，円形の標識の場合にはエッジ方向が中心に向く．そのため，各ブロックでエッジの方向の数を調べることで道路標識の円形の縁の有無を判定する．エッジ方向の投票処理は，

表 1. 関連研究と本研究の性能比較 [4].

手法	Condition (解像度)	Frame 誤認識数	Scene 正認識率 [%]	処理速度
SIFT*1 [7]	Daytime (Full HD)	$\frac{16}{215,200}$	90.4 $\left(\frac{359}{397}\text{scenes}\right)$	0.67 fps (CPU*2)
Hough変換 [8]	Daytime (Full HD)	–	91.4 $\left(\frac{533}{583}\text{scenes}\right)$	6.7 fps (CPU*3)
本研究	Daytime (640x360 pixel) (640x390 pixel)	$\frac{32}{81,140}$	100 $\left(\frac{125}{125}\text{scenes}\right)$	>60fps
	Night (640x360 pixel)	$\frac{46}{106,016}$	84.0 $\left(\frac{58}{69}\text{scenes}\right)$	
	Night (Full HD)	–	100 $\left(\frac{69}{69}\text{scenes}\right)$	

*1 Scale-Invariant Feature Transform
*2 Intel Pentium D 3.2 GHz　*3 Intel Pentium4 2.8 GHz

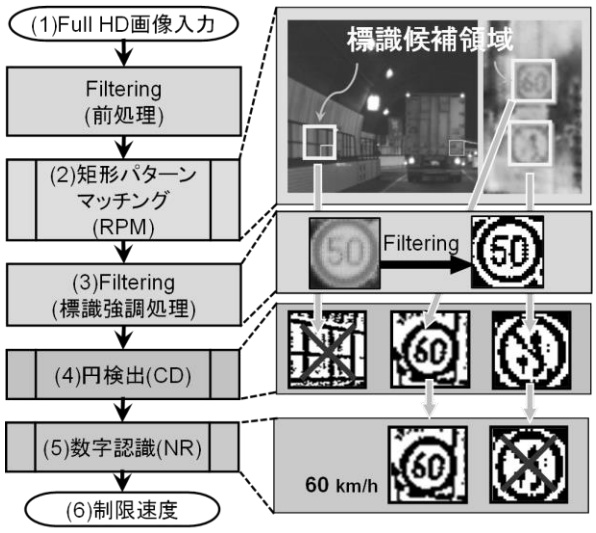


図 2. 速度標識検出システムの処理フロー [4].

Vol.2015-SLDM-173 No.23
2015/12/2

図 4 のような図 2.(2)の RPM 処理で決定した標識候補領域に対して 3×3 pixel のウィンドウを走査していき，図 5 に示すテンプレートとのマッチングを行うことで高速に処理することができる．テンプレートのマッチングがあらかじめ指定した回数あれば，円検出が行われる．

(5) **数字認識処理**（Number Recognition: NR）

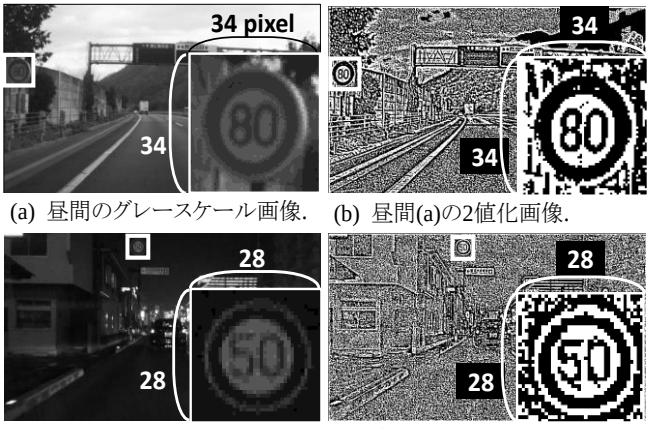
数字認識処理 [5]は，矩形パターンマッチングと円検出で絞り込まれた標識候補領域（Scan Window）から速度数字特有の局所特徴量を抽出して，その局所特徴量をあらかじめ決めた判定基準とマッチングさせることで，(10，20，…，80)の 8 種類の 2 桁の速度標識の数字を判定する．図 6（a）に示すように，Scan Window（SW）から Search-Area(数字候補領域）を絞り込み，その領域の特徴量を抽出する．SW サイズを S×S pixel とすると，SW に対する Search Area の領域は，図 6(b)に示すように定義される．本研究で用いる SW サイズは，31 種類(20,21,22,･･･,50 pixel)とした．最大 SW サイズが 50×50 pixel であるため，最大の Search Area サイズは 27×27 pixel となる．

(6) **速度標識の検出**

認識された制限速度は，運転手に通知されるとともに運転支援システムで利用される．

3. 矩形パターンマッチング（RPM）

矩形パターンマッチング処理（RPM）[4,5,6]のアルゴリズムについて図 7 を用いて説明する．まず，SW（Scan Window）内に白領域 W と黒領域 B の対になった矩形を 8 つ，図 7 のように配置する．SW を入力画像の左上から図 8 のように走査する際，SW の枠が図 7 のように道路速度標識部分の「赤い縁」のところに「黒領域 B」，その内側の「白い領域」に「白領域 W」が重なるよう 8 つの矩形が配置されたとする．この場合，周りの部分が黒く，その内側が白い画像が標識の候補として選択される．つまり，標識の縁部分の白黒特徴を捉えることにより，標識の候補検出が可能となる．



(a) 昼間のグレースケール画像.　(b) 昼間(a)の2値化画像.

(c) 夜間のグレースケール画像.　(d) 夜間(c)の2値化画像.

※(a)と(c)は，1920×1080のサイズを640×360にダウンサンプリングした画像.

図 3. 入力画像と 2 値化画像処理の例［4］.

これら 8 つの矩形$\{W_1, W_2, \cdots, W_8\}$と$\{B_1, B_2, \cdots, B_8\}$の各々で W_i の領域の輝度値の総和 $\mathrm{I}(W_i)$ とB_iの領域の輝度値の総和 $\mathrm{I}(B_i)$を計算し，対応する矩形の輝度値の差 $D_i = \mathrm{I}(W_i) - \mathrm{I}(B_i)$, $i=(1, 2, \ldots, 8)$を算出する．その後，算出された D_i とあらかじめ決めておいた輝度値のしきい値 Th_1 を比較する．このしきい値 Th_1 を大きく設定した場合，より縁が黒く，内側が白い領域が検出される．この輝度値の差 D_i は標識における白領域と黒領域の相対的な差をとって判定を行うため，撮影条件や環境に対して広く対応することができる．また，本アルゴリズムは各領域の輝度値の総和の算出，しきい値との比較のみの処理で構成されるためハードウェア化を意識した構成となっている．

道路速度標識認識の大きさは，自動車からの位置と自動車速度により大きさは，刻々と変化する．そこで入力画像に対して様々な SW サイズでラスタスキャンを行う必要がある．各 SW のラスタスキャンを行う場合，以下に示す 2 種類の方法で B/W 領域の輝度値の総和の計算結果を再利用できる．初めに，ローカルオーバラップについて説明する(図 9(a))．

1 pixel だけ SW を右方向に動かしたとき，算出する領域は直前の領域と両端の1 pixel を除いて重なっている．よって，直前の領域の輝度値の総和に，直前の領域の左端の輝度値を減算し，右端の輝度値を加算することで領域の輝度値の総和を算出できる．これによって，重なった部分の領域の計算結果を再利用することができる．輝度値を減算し，右端の輝度値を加算することで領域の輝度値の総和を算出できる．これによって，重なった部分の領域の計算結果

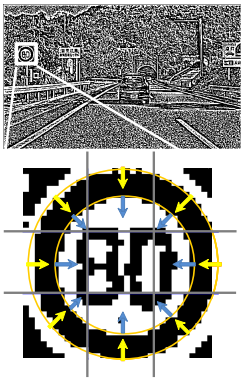


図 4. エッジ方向検出.

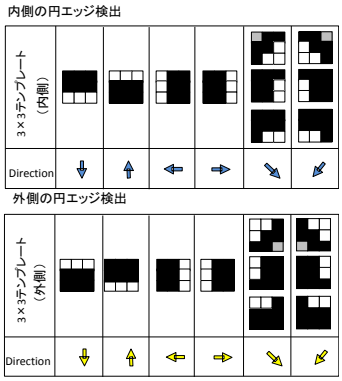
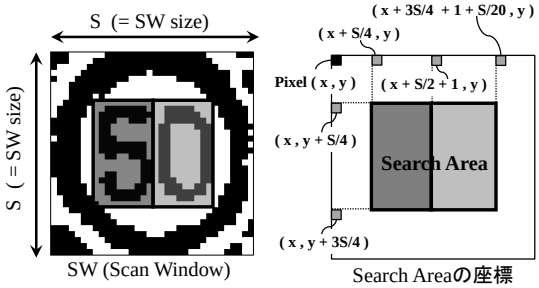


図 5. エッジ検出のための 3x3 テンプレート［5］.



（a）Scan Window の定義.（b）Search Area の定義.

図 6. Scan Window 中の Search Area の座標の定義.

を再利用することができる.

次にグローバルオーバラップを示す(図 9(b)). ラスタスキャンが進むにつれて, 図 7 の B_3, B_8 と B_4, B_7 のように領域が一致する場合がある. この場合, すでに算出されている B_3, B_4 の輝度値の総和をそれぞれ B_8, B_7 に代入し, 再利用することで総和が求めることができる. 以上の 2 種類のオーバラップ領域の計算結果の再利用によって, 輝度値の総和の算出処理が削減できるため, RPM のハードウェア実装においてハードウェアコストを削減することができる. 図 10 (a) に, グローバルオーバラップ再利用のためのアーキテクチャを示す. 図 10 (b) に, 並列に 31 種類全ての SW サイズでのラスタスキャンを行う際の設計を示す.

4. 道路速度認識アーキテクチャの実装

4.1 Protium: Rapid Prototyping Platform の概要

Protium™: Rapid Prototyping Platform は, Cadence© 社が提案する大容量 FPGA (最新の FPGA をベース) ボードと機能実装とデバック・ソフトウェア・フローを統合したことで, デザインの早期立ち上げと使い勝手の良さを実現したプロトタイプソリューションである [9]. そのため, Protium はソフトウェアの開発期間の短縮とコストの削減を可能にする. その結果, プロトタイプの立ち上げ期間の短縮が可能となり, 性能の向上へとつながる. 今回使用した Protium の特長の一部を紹介する [9].

- 高速なプロトタイプの立ち上げ. 複数 FPGA への自動分割, 自動メモリ変換とモデリング, 並びに制限なしのデザインクロック数のサポート.
- 高いモデリングの正確さ : 複雑な ASIC スタイルのクロッキングのサポート, FPGA 分割後の Palladium™データベースの自動生成.

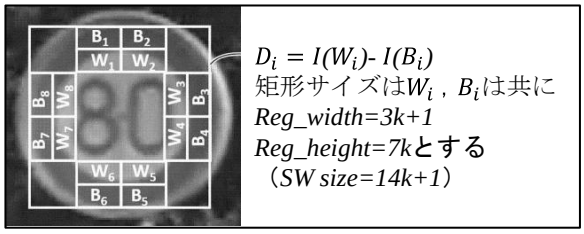


図 7. SW 内の矩形パターンの配置.

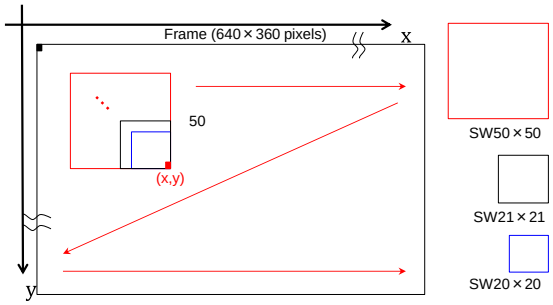


図 8. SW のラスタスキャン.

- 優れたデバッキング機能 : 波形キャプチャー, セーブ機能, シグナル・フォース/リリース機能, メモリ・アップロード/ダウンロード, スタート/ストップ, N サイクル実行を含んだ完全なクロック・コントロール機能, 並びに, インタラクティブなデバック機能.
- 卓越した柔軟性 : ボードに実装されたもっとも一般的なインターフェース, カスタムあるいは既製品のドーターボードの為の拡張コネクタ, 及び, ケイデンスの SpeedBridge™ アダプターとの互換性.

以上の特長により, 実行時のインタラクティブなデバッグ機能を提供するため, コンパイルの前に観測したい信号を選択し, データ・キャプチャー開始のトリガー条件の定義ができる。また, オフラインでの観測・解析を行うために, 実行中, 選択した信号をキャプチャーしセーブすることができる。更に、いくつかのユニークなデバック機能として back-door を利用した memory の upload/download, 信号を “1”,” 0” に force, クロックのスタート/ストップ、あるいは N サイクル実行が可能である [9]. 使用する Protium に搭載されている FPGA は 68Mbit の組込メモリを有する Xilinx-Virtex-7 2000T デバイスであり, 電源, クーリング, そして必要なインターフェース, ケーブルがすべて組み込まれたカスタム筐体となっている. 使用した

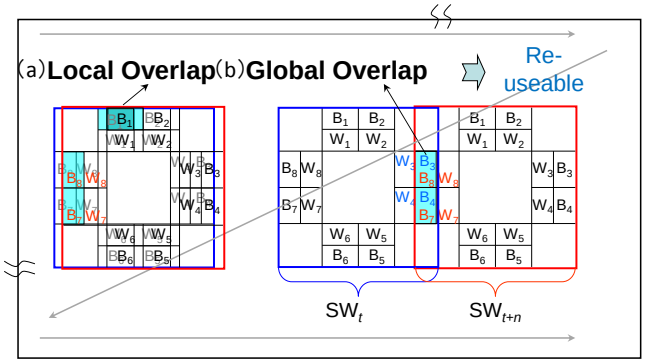


図 9. ローカルとグローバルオーバラップ.

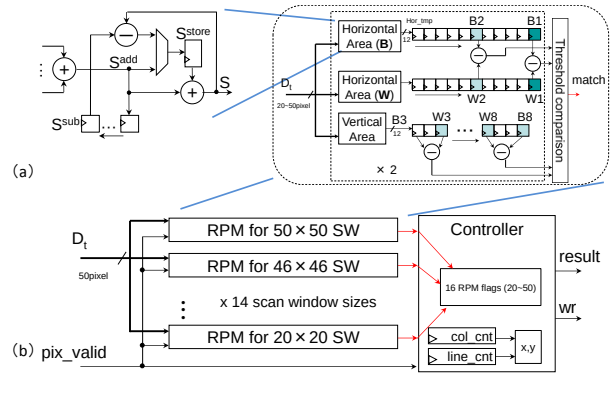


図 10. ローカルオーバラップとグローバルオーバラップを用いた RPM 実装.

Protium のハードウェアコンフィギュレーションを表 2 に示す. 図 11 に Protium のコンパイルフローを示す.

4.2 RPM のハードウェア実装

前述した Protium を使用して矩形パターンマッチング（RPM） 部分のハードウェア実装を行った. RPM の処理としては，入力画像に対して標識候補領域を検出するためのラスタスキャンを行う. この結果，標識候補領域が検出された座標の総数（N） とその座標に対して標識候補領域が検出された各 SW サイズの有無を出力データとして格納する. そのときの座標と各 SW サイズのフラグの有無のデータ構造を図 12 に示す. このとき，入力画像のサイズは y 方向に 360 pixel，x 方向に 640 pixel であるためウィンドウの座標情報は y が 9 bit，x が 10 bit で格納する. SW サイズのフラグは各ウィンドウサイズで 1 bit のデータとして格納する. 標識候補領域座標に対応する SW で矩形が検出されたとき SW のフラグは 1 とする. 座標に対する全 SW サイズフラグが 0 のときは，RPM 処理において標識候補領域が検出されないため標識データの書き込みは行わない. これにより，ソフトウェア部における必要のない入力データはカットされ，読み込みにかかる時間を短縮できる. 図 13 に道路速度標識認識プロトタイプシステムのハードウェアのアーキテクチャを示す. 前述した RPM 処理（RPM processing） を User design として組み込む. ハードウェアと PC 間のデータ転送には PCI Express を介した DMA 転送，データの読み書きのためのバスには AXI バスを使用している.

AXI バス仕様では読み出しと書き込みいずれの場合も，データとアドレスで別々の接続を使用する [10]. AXI では読み出しと書き込みのために 2 本のバスが接続されているため，データの読み書きが独立でデータの同時転送が可能となる. そのため転送データが衝突しないため，それを防ぐためのプロトコルがコンパクトなものになる. またデータの転送が，独立のため双方の転送されるデータサイズが異なっていても問題が起こらない. 以下にアーキテクチャの動作を説明する.

①PC から入力画像データ（1 pixel = 8 bit）を 32 bit のデータにパックしてメモリ 0 へと DMA 転送を行う.

② PC から interface 内の Controller へと start_trigger を送ることで，RPM with AXI interface の動作を開始する.

③ Master AXI interface は前述した AXI バスを経由してメモリ 0 からアドレスを送りデータを受け取り，メモリ 0 から読み出した 32 bit のデータを FIFO に送る.

④ FIFO から 1 pixel （8 bit） 単位のデータを RPM processing の入力として処理をする.

⑤ RPM processing から図 12 のような構造の 32 bit データが出力され，出力データは FIFO と AXI バスを経由しメモリ 1 で保持される.

③～⑤までの一連の動作を並列して行う.

⑥ 全ての入力データに対して RPM processing が終了すると，標識候補領域として検出された座標の総数を出力し同様にメモリ 1 に保持する.

⑦ Controller から PC へと end_trigger を送ることで，メモリ 1 から PC への出力データの転送を開始する.

⑧ 標識候補領域が検出された座標の総数従って，メモリ 1 のデータを PC へと転送する.

4.3 システムの実装

図 14 に Protium™ と Host PC の概観を示す. 図 14 において，PCIe 拡張ケーブルを経由して図 13 の Host PC と Protium 間のデータ転送が PCI- Express を介して行われる. Host PC から画像を転送し，Protium 側で処理されたデータを Host PC 側に返し，ディスプレイに結果を出力している. 図 15 に一般道路と高速道路の入力画像，図 16 に矩形パターンマッチング（RPM），図 17

表 2. Rapid Prototyping Platform Hardware Configurations （39RX712）.

FPGAs	**2 Virtex-7 XC7V2000T**	**FPGA-internal memory**	**136 Mbits**
FPGA boards	**1**	**Clock generators**	**5 programmable synthesizers**
Approx. total capacity	**Up to 25M ASIC gate**	**Mictor connectors**	**2**
On-board memory	**Up to 32GB**	**Expansion connectors**	**4**
On-board interfaces	**PCIe (Gen1/2), SATA, USB 3.0, Ethernet, XAUI, Infiniband**	**User I/Os**	**288 LVDS pairs or 584 single-ended**
Front panel interfaces	**2x 4-lane PCI, Express®, Gen 2, 3X SFP+, 1X QSFP+, 1X USB 3.0**	**Board configuration**	**Ethernet USB PCIe**

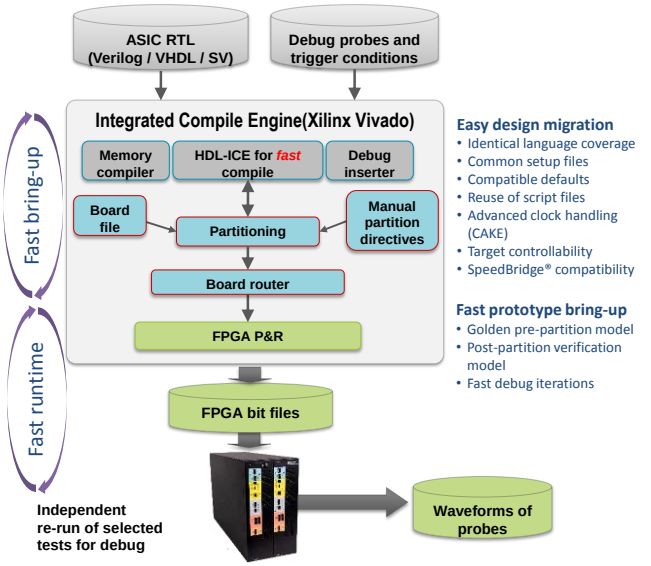


図 11. Protium コンパイルフロー.

に数字認識処理を行った結果を示す.図 16 の矩形は標識候補を表す.図 17 では候補領域から数字認識により認識された速度標識（矢印）を出力した結果である.

Location 19 bit		f lag 1 bit×31				
y 9bit	x 10bit	SW size50	SW size49	・・・	SW size21	SW size20

図 12. RPM 処理の出力データ

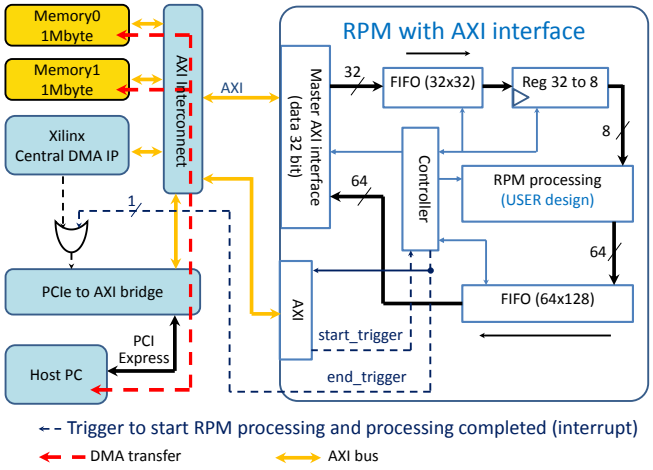


図 13. ハードウェアのアーキテクチャ.

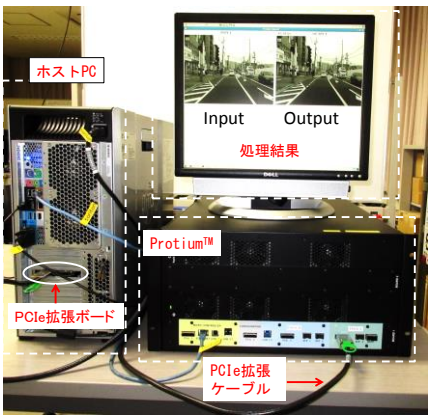


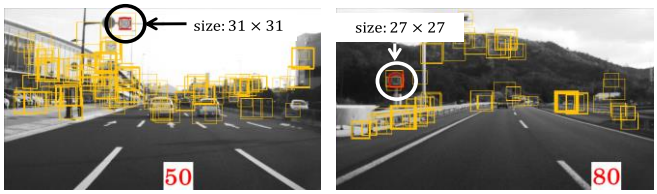
図 14. Host PC と Rapid Prototyping Platform.

(a) 一般道路　　(b) 高速道路

図15. Input画像(640×360).

(a) 一般道路（193個検出）　　(b) 高速道路（144個検出）

図16. 矩形パターンマッチング(RPM)出力結果(正方形領域).



(a) 一般道路（50: size31 × 31）　　(b) 高速道路（80: size27 × 27）

図17. 道路速度認識結果(NR処理後).

5. まとめ

数字標識認識システムのうち矩形パターンマッチング処理(RPM)の Protium™ Rapid Prototyping Platform への実装を行った．今後の課題は，RPM 以外の処理の Protium™ への実装と評価が挙げられる.

謝　辞

本研究の一部は, JSPS 科研費基盤研究(B) 26280015 の助成を受けたものです．また，Protium™ Rapid Prototyping Platform は，日本ケイデンス・デザイン・システムズ社の協力で行われ, Vivado™ FPGA 設計ツールによる開発は，Xilinx University Program（XUP）の提供で行われました.

文　献

[1] 加藤良文，他,“デンソーの先進安全技術動向”，デンソーテクニカルレビュー, Vol.18, pp. 12-22, 2013.
[2]運転支援システム(トヨタ自動車, 富士重工業, ホンダ, 日産自動車, 仏ヴァレオ），“君も名ドライバーになれる ”，日経ビジネス/日経 BP 社 編 (1713): 2013.10.28. 114-117 ISSN 0029-0491, 2013.
[3] Euro NCAP, “Euro NCAP 2020 Roadmap” http://euroncap.blob.core.windows.net/media/16472/euro-ncap-2020-roadmap-rev1-march-2015.pdf , 2015-10-24 accessed.
[4] A. T. Hoang, et al., “Pipeline scanning architecture with computation reduction for rectangle pattern matching in real-time traffic sign detection”, Proc. of the IEEE International Symposium on Circuits and Systems (ISCAS2014), pp.1532-1535, 2014.
[5] M Yamamoto, et al., “Compact hardware oriented number recognition algorithm for real-time speed traffic-sign recognition,” Proc. of the IEEE International Symposium on Circuits and Systems（ISCAS2014), pp. 2535-2538, 2014.
[6] A. T. Hoang, et al., “Real-time Speed Limit Traffic Sign Detection System for Robust Automotive Environments”,Proc. of IEIE Transactions on Smart Processing and Computing, Volume 4, No. 4, pp.237-250, August, 2015.
[7] 高木雅成，他, “SIFT 特徴量を用いた交通道路標識認識”, SSII2007 論文集, LD2-06, 2007.
[8] 石塚裕 他, “Opponent-Color フィルタを用いた道路交通標識認識システム”, 信学技報, No. 103（737）, pp. 13-18, 2004
[9] “Protium: Rapid Proto Platform”, Cadence Design Systems, Inc., https://www.cadence.co.jp/products/pdf/Protium_DS%20_J.pdf, 2015-10-22 accessed.
[10] Xilinx AXI リファレンスガイド https://japan.xilinx.com/support/documentation/ip_documentation/axi_ref_guide/v13_2/j_ug761_axi_reference_guide.pdf, 2015-10-22 accessed.