5A-01

RISC-V における AES 向けカスタム命令の試作と評価

岡田 正純† 山下 雄也† 鈴木 弘成† 外山 正勝†
三菱電機株式会社 情報技術総合研究所†

1. はじめに

オープンな命令セットアーキテクチャである RISC-V の特徴の 1 つとしてカスタム命令の実装が挙げられ，CPU アーキテクチャの用途に応じた処理性能の向上が期待されている．一方で，カスタム命令の実装による処理時間の削減と回路規模の増加はトレードオフの関係にある．そのため，カスタム命令を設計する際には，このトレードオフを理解しておく必要がある．

そこで本稿では，Advanced Encryption Standard (AES)処理を題材として，カスタム命令の実装による処理時間の削減と回路規模の増加を評価した結果を述べる．

2. カスタム命令化対象処理の選定

前述したように，カスタム命令化には処理時間と回路規模に関するトレードオフが存在する．そのため，システム全体の最適化を考える上では，適切な処理に絞ってカスタム命令化する必要がある．

本試作では，回路規模の増加を抑えるため，実行する全ての処理へのカスタム命令化の適用は行わない．加えて，処理時間の高速化比/回路規模の増加比を 1 以上にすることを目標としたとき，以下の 3 条件を満たす処理が，カスタム命令化に適すると仮定した．

① ホットスポット処理である．
② CPU バウンドな処理である．
③ 並列実行可能なループを含む処理である．

上記条件を満たす処理を AES 処理の中から探し，その処理をカスタム命令化対象とする．

まず，AES 処理のうち，①を満たすものを探す．図 1 に，AES 処理を構成する 4 つの処理とその演算負荷を示す[1]．

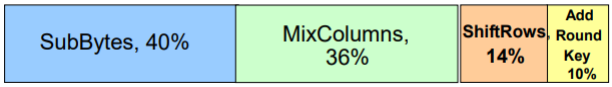

図 1 AES 処理の各モジュールの演算負荷

図 1 より，ホットスポットとなっている処理は，SubBytes と MixColumns である．したがって，①を満たす処理は，SubBytes と MixColumns である．

次に，SubBytes と MixColumns のうち，②を満たす処理を探す．SubBytes の処理内容を図 2 に，MixColumns の処理内容を図 3 に示す．

```
void SubBytes( unsigned char SBox_Table[16][16]){
    int x,y,i;
    for(i=0;i<16;i++){
        x= STATE[i] & 0x0f;
        y= STATE[i] >> 4;
        STATE[i] = SBox_Table[y][x];
    }
}
```

図 2 SubBytes の処理

```
void MixColumns(void){
    int i;
    unsigned char tmp[16];
    for(i=0;i<16;i++){
        tmp[i]=STATE[i];
    }
    for(i=0;i<4;i++){
        STATE[i*4]   = mul2(tmp[i*4  ]) ^ (mul2(tmp[i*4+1])
              ^ tmp[i*4+1]) ^ tmp[i*4+2] ^ tmp[i*4+3];
        STATE[i*4+1] = mul2(tmp[i*4+1]) ^ (mul2(tmp[i*4+2])
              ^ tmp[i*4+2]) ^ tmp[i*4+3] ^ tmp[i*4  ];
        STATE[i*4+2] = mul2(tmp[i*4+2]) ^ (mul2(tmp[i*4+3])
              ^ tmp[i*4+3]) ^ tmp[i*4  ] ^ tmp[i*4+1];
        STATE[i*4+3] = mul2(tmp[i*4+3]) ^ (mul2(tmp[i*4  ])
              ^ tmp[i*4  ]) ^ tmp[i*4+1] ^ tmp[i*4+2];
    }
}
```

図 3 MixColumns の処理

図 2 より，SubBytes の主な処理は配列の要素アクセスであるため，メモリバウンドである．一方，図 3 より，MixColumns の主な処理は行列積演算であるため，CPU バウンドである．したがって，②を満たす処理は MixColumns である．

最後に，MixColumns が③を満たすかを確認する．図 3 より，MixColumns には並列実行可能なループが含まれている．したがって，MixColumns は③を満たす．

以上より，本稿におけるカスタム命令化対象処理として，MixColumns を選定した．

3. 実装

RV32IM を実装しているオープンソースの RISC-V CPU コア[2][3]を基に，AES 暗号化処理回路内の MixColumns モジュールをカスタム回路として追加した．

3.1 カスタム命令の定義

MixColumns モジュール用カスタム回路を S/W から起動するために，カスタム命令を定義する．命令定義は RISC-V の仕様に沿って作成した．

図 4 に追加する命令フォーマットの表記を示す．

U 形式の命令仕様として，出力にデスティネー

Prototyping and Evaluation of Custom Instruction for AES on RISC-V
Masazumi Okada, Yuya Yamashita, Kosei Suzuki, Masakatsu Toyama
† Information Technology R&D Center, Mitsubishi Electric Corporation

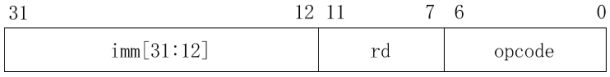

図 4 U 形式の命令フォーマット

ションレジスタの rd，入力に即値オペランドの imm[x]が定義される．本稿で実装するカスタム回路上で即値オペランドは未使用のため，任意の値とした．また，opcode には仕様で定義されているカスタム命令用のオペコードを用いた．

3.2 C ソースへの呼び出しの実装

C ソースコードのコンパイル時に 3.1 節で定義したカスタム命令を呼び出すため，バックエンドを改変して命令の追加を行った．本稿では LLVM の Target Description ファイルを改変し，オペコード定義と命令定義を追加することで，LLVM Clang からカスタム命令を含むソースコードを出力するようにした．

4. 評価

4.1 動作確認

まず，改変した LLVM Clang でカスタム命令非実装/実装のテスト用 C ソースコードをコンパイルし，実行ファイルを生成した．実行ファイルの逆アセンブル結果から，想定通りカスタム命令用オペコードが呼ばれていることを確認した．

次に Vivado 2017.2 を用いて，評価基板の Arty A7-35T に RISC-V CPU コアと生成した実行ファイルを bit ファイルとしてダウンロードした．ここでカスタム命令が Vivado シミュレーション上でも波形として呼び出されていることを確認した．

また，テスト用 C ソースコードではカスタム命令非実装/実装で同一の入力を与え，出力の比較を行った．シリアル出力により，出力結果の値が両者で一致することを確認した．

4.2 性能計測

本稿で用いた RISC-V CPU コアの場合，本処理で使用する全ての命令は 1 命令 5 ステージで完了する．また，クロック周波数は 500MHz 固定であり，パイプライン機構も存在しない．よって，実行時間は命令数に正比例する．そこで，MixColumns モジュール処理の実行ファイルを逆アセンブルし，カスタム命令化によって削減された命令数の比較から高速化比を導出した．また，カスタム命令化による高速化の評価を行うため，性能計測の条件として，最適化オフのオプションを使用してコンパイルを実行した．

命令数を導出すると，カスタム命令非実装で 238 命令，カスタム命令化によって 14 命令に削減されることが分かった．これにより，カスタム命令を用いることで，約 17 倍の高速化比を得ることが分かった．また，命令数から求めた MixColumns 処理における高速化比を，AES 処理全体を実行する際の高速化比の算出に用いると，Amdahl の法則より，約 1.5 倍の高速化比になることが分かった．

次に，Vivado の utilization の結果からカスタム命令非実装/実装での回路規模の比較を行う．カスタム命令非実装での LUT，FF，BRAM，DSP，IO，BUFG の utilization の合計は 5502 に対し，カスタム命令化によって 5610 に増加することが分かった．これより，カスタム回路の実装による回路規模の増加率は約 1.02 倍になることが分かった．

4.3 考察

他のシステムと比較するため，以上の結果を標準化し，回路規模の増減に対する処理時間の高速化効率を求める．この計算効率を評価する目安として，費用対効果=高速化比/回路規模比を導入すると，MixColumns 単体で 16.7，AES 処理全体として 1.47 の値を得ることが出来た．

よって，2 つの費用対効果の値が 1 より大きいことから，回路規模と処理時間のトレードオフが考慮されたカスタム命令を実装できたと考えられる．また，カスタム命令非実装の性能と比べて，AES 処理の全てをカスタム命令化せずに，処理時間を大きく削減できたと考え，2 章の仮定は本稿の測定条件下では有意であると言える．

5. まとめ

本稿では 2 章で述べたカスタム命令化における 3 条件の仮定に基づき，選定した MixColumns 処理をカスタム回路として実装し，MixColumns 処理と AES 処理全体の高速化比，カスタム命令非実装/実装を費用対効果の視点で評価した．

今後の課題として，回路規模と処理時間のトレードオフを考えるとき，カスタム命令化した際の各モジュールの回路規模の増加率を考慮して，選定を行うことがあげられる．また，2 章の条件を満たさない処理のカスタム命令化や，カスタム命令化の規模や処理の種類を増やすことで，2 章の 3 つの条件と評価式の有効性を検証したい．

参考文献

[1]梅原直人，古川達久，的場督永，ほか：ハード/ソフト最適分割を考慮した AES 暗号システムと JPEG エンコーダの設計と検証，第 4 回情報科学技術フォーラム講演論文集，pp．257-260(2005)．

[2]石原ひでみ：俺々RISC-V へのカスタム命令実装，FPGA マガジン，Vol. 18，pp. 24-63 (2017)．

[3]aquaxis: GitHub(online), available from <https://github.com/aquaxis/FPGAMAG18>(accessed 2020-12-14)．