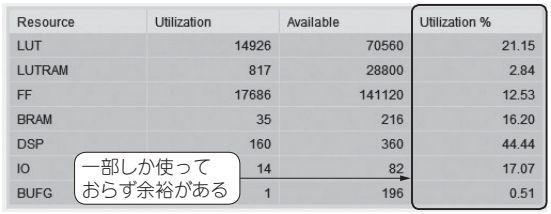
映像認識人工知能のカリカリ高速化に挑戦

奥畑 宏之

表1　処理パートごとの処理時間および実行回数

項　目	時間［秒］
行列B→DMAバッファ	12.87
DMA転送＆行列乗算	46.26
DMAバッファ→行列C	60.64
その他	0.9
行列乗算総処理時間	120.68

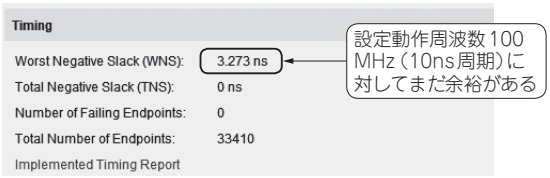
行列乗算実行回数1050704回



Resource	Utilization	Available	Utilization %
LUT	14926	70560	21.15
LUTRAM	817	28800	2.84
FF	17686	141120	12.53
BRAM	35	216	16.20
DSP	160	360	44.44
IO	14	82	17.07
BUFG	1	196	0.51

一部しか使っておらず余裕がある

（a）リソース



Timing

Worst Negative Slack (WNS):	3.273 ns
Total Negative Slack (TNS):	0 ns
Number of Failing Endpoints:	0
Total Number of Endpoints:	33410

Implemented Timing Report

設定動作周波数100MHz（10ns周期）に対してまだ余裕がある

（b）タイミング

図1　生成した行列乗算ロジック回路の残りリソースを確認するVivado レポート

処理時間を解析して高速化の方針を決める

● 処理時間を測定する

ここまでの行列乗算のハードウェア化では，処理時間が思ったより短くなりませんでした．

その原因を調べるため，変更したニューラル・ネットワークDarknetソースコードの行列乗算まわりの処理時間の内訳を測定してみます．行列乗算の実行回数および各処理パートの処理時間合計が**表1**のようになりました．

● 方針を決める

まず，ロジック回路（IP）とのデータ転送，特に，乗算結果をDMAバッファから行列Cにコピーする部分で時間がかかっています．また，32×32行列乗算に関して，ハード化によって高速化がされましたが，まだ46秒ほどかかっています．

そして，32×32行列乗算が，およそ100万回実行されています．一度に乗算する行列のサイズを大きくして実行回数を削減すれば，転送におけるオーバヘッドも小さくできると思われます．

高速化の方針としては次の2点を攻めることになります．

(1) 行列乗算IPの性能（サイズ/スピード）を上げる
(2) データ転送を高速化する

高速化その1：行列乗算ハードウェアIPの性能向上

● コンパイル・レポートを確認

プログラマブル・ロジック部PLの行列乗算回路をコンパイルした結果のレポートは**図1**の通りです．リソースの使用率についてLUTは21%，DSPは44%，タイミングについてWorst Negative Slack（WNS）は3.2nsとかなり余裕があります．行列乗算IPの性能を上げる余地はありそうです．

● 検討1：1回で演算する量を増やす

行列サイズを見直したり，ロジック回路のクロック周波数を上げたりして性能を改善しました．その結果，試行錯誤の細かい過程は省略しますが，行列乗算IPが1回に演算する行列のサイズを32×32から64×64に増やせました．

● 検討2：クロック周波数を上げる

また，クロック周波数は100MHzから299MHzへ上げることができました．どうもUltraScale+ MPSoCプロセッサ・システムに接続するAXIバスのクロック周波数上限は300MHzくらいにあるようです．

● ハードウェア・デザインの変更

乗算する行列のサイズを32×32から64×64に変更するには，Vivado HLSのCソースコードを**リスト1**～