

FPGA を用いた教育向け 16-bit コンピュータと その開発環境の構築

著者 中後 和希

指導教員 斉藤 徹

1. 研究目的

今日の情報技術の発達により、一般ユーザーだけでなくソフトウェア技術者であってもコンピュータをブラックボックスとして扱うことが多くなった。しかし、複雑化した現代のコンピュータや高度化したソフトウェアからそれらの仕組みを学ぶことは非常に困難である。そのため本研究では、ソフトウェア技術者のための CPU アーキテクチャ教育環境を構築する。その教育環境として、16-bit コンピュータとその CPU を対象とするアセンブラと簡易 C 言語コンパイラを作成する。これらは極めてシンプルに作られており、コンピュータの基礎部分の理解を深めることに役立つ。

16-bit コンピュータの開発には FPGA とハードウェア記述言語 Verilog HDL を用いた。ハードウェアの内部構造や動作をプログラミング言語のようなコンピュータ言語で記述することで、ソフトウェア技術者でもその仕組みを直感的に理解できるようにしている。

ソフトウェア技術者であれば、ハードウェアをソフトウェア面から学ぶこともできる。特にアセンブリ言語は CPU アーキテクチャと密接な関係にある。そのため本研究では、教育向けのアセンブラと簡易 C 言語コンパイラを開発環境として提供し、アセンブリ言語の習得を手助けする。

開発する 16-bit コンピュータとアセンブラの仕様には、情報処理技術者試験におけるプログラミング能力試験のために仕様策定された仮想計算機 COMET II と、そのアセンブリ言語 CASL II を採用する。それにより、ハードウェア理解と同時に情報処理技術者資格の取得を支援することを目標としている。今回考案した教育向け 16-bit コンピュータを以下「FPGA 版 COMET II」と表記する。

最終的には、FPGA 版 COMET II と CASL II アセンブラの実装が完了し、論理シミュレーションによる総合テストを行うことができた。簡易 C 言語コンパイラは完成には至らなかったが、コンパイラを開発する上で必要となる呼出規約の定義を行った。

2. システム概要

2.1. FPGA 版 COMET II

命令セットアーキテクチャには情報処理技術者試験の COMET II の命令セットアーキテクチャ^[1]を

採用した。COMET II の動作仕様を満たす CPU アーキテクチャを設計し、その回路を Verilog HDL で実装した。FPGA 版 COMET II のブロック図を図 1 に示す。

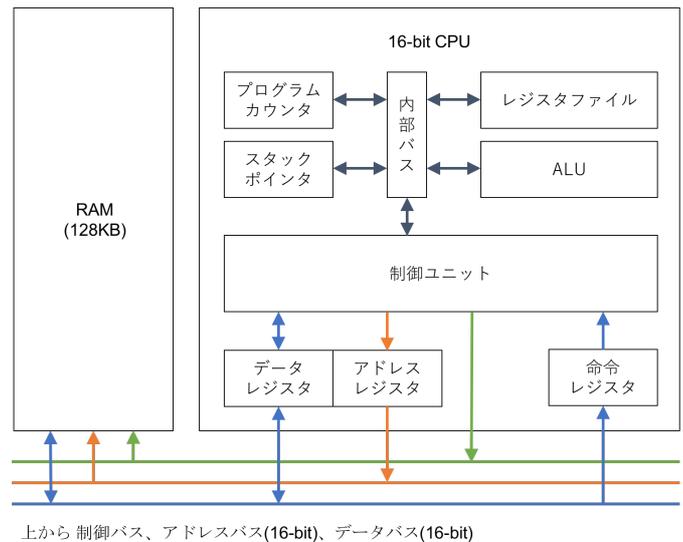


図 1 FPGA 版 COMET II のブロック図

FPGA 版 COMET II はノイマン型コンピュータであり、16-bit CPU や RAM、入出力ポートで構成されている。16-bit CPU はプログラムカウンタやスタックポインタの他に、演算に必要なデータを一時的に記憶するための汎用レジスタを 8 個備えている。RAM には実行するプログラムを格納し、実行時はデータメモリとしても使用する。RAM の容量は 65536 語 (1 語は 2-byte) で、そのアドレス空間は 0 から 65535 番地である。メモリアクセスの際は、メモリとのデータの受け渡しを仲介するデータレジスタと、アクセスする番地を指定するためのアドレスレジスタを使用する。

入出力機能は、COMET II の規格に定義されていないが FPGA にて実装が容易となるようにメモリマップド I/O 方式で実装した。アドレス空間の一部を入出力ポートに割り当てることで、命令セットを拡張せずに既存のロード・ストア命令で入出力を可能にした。

FPGA 版 COMET II の CPU 処理方式にはマルチクロックサイクル方式を採用した。最も単純な逐次制御方式として単一クロックサイクル方式があり、その方式では 1 命令を 1 クロックで実行する。しかし、動作周波数は命令セットの中で最も処理時間が長い命令に合わせる必要があり、処理時間が

短い命令では余り時間が発生し処理効率が悪い。マルチクロックサイクル方式では、1 命令の実行を同程度の処理量の工程に分割し各工程を 1 クロックで実行していく。各命令について必要な工程だけ実行することで、処理時間が短い命令で余り時間が発生せず、また動作周波数を高くすることができる。FPGA 版 COMET II では、命令実行工程を (1) 命令フェッチ、(2) オペランドフェッチ、(3) デコード/レジスタフェッチ (4) メモリアクセス、(5) レジスタへのライトバックという 5 つの工程に分割している。

2.2. 簡易 C 言語コンパイラ

制御構文や静的変数・動変数、関数呼出といった最低限の機能を持つ C 言語コンパイラの開発を想定し、本研究ではそのための呼出規約を定義した。

例えば、C 言語プログラム上で foo という名前の関数が定義されている時、表 1 に示す CASL II プログラムを出力する。

表 1 関数 foo に対応する CASL II プログラム

```
_foo START
; 関数の処理
RET
END
```

関数や静的変数の定義がある時、その識別子の前にアンダースコア文字を付加した文字列をラベルとするサブルーチンを生成する。このラベルはアセンブリ上で一意であるので、その関数や静的変数を参照する時はこのラベルを参照する。アンダースコア文字以外の英文字で始まるラベルは、処理系で自由に使用できるようにしている。

自動変数は通常スタック領域に配置するが、FPGA 版 COMET II にはスタックポインタを用いたアドレッシング命令が無いので、汎用レジスタをスタックポインタとした自動変数用のスタック領域(以下、疑似スタック)を使用する。アセンブラ命令の DS 命令でメモリ上に固定サイズの領域を確保し、それを疑似スタック領域とした。疑似スタックの最下段とスタック内の一番上のデータのアドレスを表すため、最下段にはラベル、疑似スタックポインタとして汎用レジスタ GR7 を割り当てた。

また、汎用レジスタ GR6 をフレームポインタとして割り当てることで、疑似スタック内に配置されている自動変数を参照できるようにした。フレームポインタの値とフレームポインタからのオフセットの和が、その自動変数の先頭アドレスとなる。ある自動変数の値を汎用レジスタ GR1 にロー

ドする CASL II プログラムを表 2 に示す。

表 2 自動変数をロードする CASL II プログラム

```
LD GR1, <オフセット>, GR6
```

3. 動作検証

FPGA 版 COMET II と CASL II アセンブラの動作確認のため、Icarus Verilog を用いて論理シミュレーションを行った。

表 3 に示す CASL II プログラムを用意し、作成した CASL II アセンブラでアセンブルする。シミュレーション開始前に FPGA 版 COMET II の RAM にその機械語を格納し実行した。そのシミュレーション結果を図 2 に示す。

表 3 動作検証に用いた CASL II プログラム

```
PROG START
LAD GR1, 5
LAD GR2, 2
PUSH 0, GR1
ADDA GR1, GR2
POP GR3
END
```

```
GR0: 0 GR4: 0
GR1: 7 GR5: 0
GR2: 2 GR6: 0
GR3: 5 GR7: 0
```

図 2 実行後の汎用レジスタの値

動作検証の結果、仕様どおりに動作していることが確認できた。

4. まとめ

本研究はソフトウェア技術者のための CPU アーキテクチャ教育環境を構築することを目標し、今回の研究では FPGA 版 COMET II と CASL II アセンブラの実装が完了した。C 言語コンパイラは、構文や演算子、データ型を限定して開発中である。

FPGA 版 COMET II を論理合成可能なソースコードに修正し、コンパイラも含めた FPGA 上での総合テストの実施を今後の課題としたい。

参考文献

- [1] 「試験で使用する情報技術に関する用語・プログラム言語など Ver3.0」
 <https://www.jitec.ipa.go.jp/1_13download/shiken_yougo_ver3_0.pdf> (2018 年 2 月参照).