小規模開発下における zynq の開発環境の提案

1685008 岡村　拓弥

指導教員　木村　誠聡　教授

1　はじめに

FPGA は 1985 年に初めて登場した，回路を何度もプログラムによって書き換え可能なデバイスであり，回路であるため電気信号により実行され，全て並列で動作する．CPU と異なり処理速度はハードウェア依存により一定であり，速度が必要な機能だけを並列化でき，効率的に組み上げることが出来る．またソフトウェアを必要とせず，低コストで大量生産できるというメリットがある．FPGA は並列処理が可能であることから，科学技術計算や画像・映像処理などに用いられることが多く [1]，組み込みの分野で多く利用されるようになってきている．これらのことから，現在 FPGA の需要が高まってきており，世間的に FPGA 技術者が求められている．しかしながら，CPU のプログラムを扱ってきている人にとって，並列処理を感覚的に扱うことが難しく，なかなか手が出しづらいという問題点があり，FPGA への導入に関して技術的な支援が望まれる．また，現時点における種々の処理は常に一定の処理とは限らず，条件によって処理を複雑に変更することが行われる．よって，FPGA のみならず命令によって処理を変更することが出来る CPU の存在が不可欠と考える．そこで本稿は FPGA と CPU が混在した SoC のひとつである Xilinx 社の Zynq に着目し，CPU の状態によって FPGA の処理内容を変更し，簡易的に CPU と FPGA の処理が可能な初心者用システムを検討する．検討するシステムは CPU 側はインタプリタであり，FPGA 側は CPU の命令によって処理内容を切り替えることが出来ることとする．このシステムにより，CPU と FPGA が混在するシステムを扱う初心者が簡易的に扱うことが可能と考える．また一般的には CPU と FPGA 双方の開発環境が必要であるものの，検討するシステムでは，それらが不要となるため，手軽に Zynq を扱うことが可能となる．本稿ではこのシステムの検討と，現時点までの開発状況について報告をする．

2　提案するシステム

2.1　概要

CPU と FPGA が混在したシステムは本来各々の開発環境が必要であり，各々異なる言語での開発となる為，初心者が即時対応することは難しい．そこで本稿では簡易的に CPU と FPGA の処理が可能な初心者用システムを，統合開発環境として提案する．これはインタプリタを利用し，インターフェースを統一することで実現する．また microSD カードの中に開発環境を構築することで，簡単に導入することができ，開発環境用の PC を選ぶことなくどこでも開発できる環境が用意できる．

本システムは microSD カードに開発環境を構築し，PC と Zynq を USB で接続する．その後 PC からの入力で CPU を制御し，そこから FPGA を制御するシステムとなっている．図 1 に本研究のシステム構成図を示す．

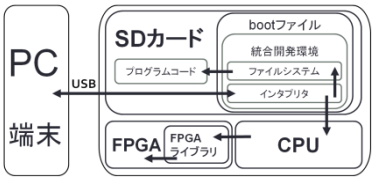

図 1　システム構成図

2.2　構成

Zynq は microSD カードから起動できるため，microSD カードに Boot ファイルを用意し，その中に開発環境を構築する．またファイルシステムを実装することで，インタプリタで編集したコードを保存，読み込みができるようにする．テキスト形式で保存することにより，外部からの編集にも対応させる．FPGA 側の開発はあらかじめ回路を複数用意しておき状況によって切り替える．このため機能は制限されるが，FPGA 側のコードをコンパイルする必要がなく開発時間を短縮できる．本システムの開発ツールとして FPGA 側は Vivado 2016.2，CPU 側は Xilinx SDK を使用する．使用機器は Zynq z-7010 zybo を使用し，CPU は ARM Cortex A-9 650MHz Dual Core を使用する．開発言語は C 言語と VHDL を使用する．また利用者が後から回路を追加，変更出来るようライセンスフリーの開発環境を利用する．

3　実装内容

実際に実装する回路として，小規模開発を想定し，ボードで実験，開発できるものとし，論

理回路と 8bit の GPIO を用意する．また，ボード上の LED，スライドスイッチ，ボタンにアクセスできるようにする．

実装したインタプリタコマンドを表 1 に，プログラムコマンドを表 2 に示す．インタプリタコマンドでは，プログラムの修正，追加，削除，挿入，実行等が可能であり，プログラムコマンドでは，外部入力，四則演算，分岐，繰り返し等が実行できる．

表 1 インタプリタコマンド　表 2　プログラムコマンド

コマンド	
A	追加
I	挿入
E	編集
del	削除
lst	プログラム表示
run	実行
sav	保存
lod	読み込み
pls	プログラム一覧表示
ico	インタプリタコマンド一覧
pco	プログラムコマンド一覧

	コマンド	変数1	変数2	変数3
外部入力	scan	変数名		
演算	cal	変数名	演算子	変数or#整数
比較	if	変数名	演算子	変数or#整数
ラベル	labl	ラベル名		
移動	goto	ラベル名		
コメント	rem	コメント文		
表示	prt	変数名or"文字列		
回路選択	F	回路名	変数1	変数2
終了	end			

選択できる回路として以下のものを用意する．

表 3　回路

GPIO 8bit入力 x2	7seg x1	ANDx10	NORx10
GPIO 8bit出力 x3	SW	ORx10	XORx10
counter x1	LED	NANDx10	NOTx10

論理回路は複数実装し，利用者が組み合わせて利用できるようにする．

4　実行結果

開発した統合開発環境の実行手順として，作成した開発環境を microSD カードに導入し，PC と Zynq を USB で接続する．そこで Zynq のドライバをインストールし，その後 PC 側でターミナルソフトを起動し，インタプリタによる開発を行う．

```
  --- --- Interpreter Command List ---

A       program Add
I       program Insert
E       program 1 line Edit
del     program 1 line Delete
lst     program List
run     program Run
sav     program Save
lod     program Load
pls     program List in SD card
pco     program Command List
ico     interpreter Command List

  1 : scan val 0
  2 : scan val 1
  3 : cal val_0 + #65
  4 : prt val 0
  5 : prt val 1
  6 : labl p1
  7 : prt "test
  8 : goto p1 3
  9 : if val_0 <= val_1
 10 : cal val_0 + val_1
 11 : cal val_0 - val_1
 12 : end
 13 : scan val 2
 14 : scan val 3
 15 : cal val_2 - val_3
 16 : scan val 4
 17 : scan val 5
 18 : cal val_4 + val_5
 19 : end

command -> ▌
```

図 2　開発画面

```
command -> run
 -- program run --

000:[scan][val][0][]
val[0] -> 12

001:[scan][val][1][]
val[1] -> 45

002:[cal][val_0][+][#65]
12 + 65 = 77

003:[prt][val][0][]
77

004:[prt][val][1][]
45

005:[labl][p1][][]

006:[prt]["test][][]
"test

007:[goto][p1][3][]

006:[prt]["test][][]
"test

007:[goto][p1][3][]

006:[prt]["test][][]
"test

007:[goto][p1][3][]

006:[prt]["test][][]
"test
```

図 3　実行画面

実際にインタプリタで開発している画面を図 2 に示す．また実行画面の一部を図 3 に示す．

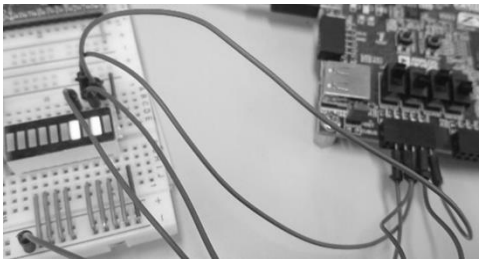

図 4　GPIO からの出力

GPIO 出力の例として，ボード上のスイッチの入力を GPIO から出力し，外部の LED を光らせている様子を図 4 に示す．また，既存の開発環境での開発工程は，開発環境の導入，プロジェクトの作成，開発ボードの設定，コーディング，コンパイル，配置配線，書き込み，実行の順で行う必要があるが，提案する統合開発環境では，コーディングと実行だけで開発が可能であり，大幅に開発工程の手間を削減した．

5　まとめ

本稿では，小規模開発における Zynq の開発環境として，microSD カードの中にインタプリタ形式による統合開発環境を構築する事を提案した．インタプリタを実装し，記述したコードをファイルシステムにより保存することができることを確認した．また，FPGA 側のアドレスへアクセスして回路の切り替えが可能であり，出来ることは限られるが開発ボードを使った簡単な実験，開発が可能であることを確認した．また，開発工程を削減したことによって，容易に実験・開発が可能になり，FPGA 開発への敷居が低くなったと考える．今後の課題として回路の種類を増やし，汎用性を広げる事が必要だと考える．

参考文献

1) 「【 基礎調査 】FPGA は 産業界・科学技術研究の現場で、どのように活用されているのか？」<http://qiita.com/HirofumiYashima/items/40d33390221e8db4e98f> (2016.10.14)
2) 鈴木 量三郎,片岡 啓明:ARM Cortex-A9x2! Zynq でワンチップ Linux on FPGA, CQ 出版社(2014.11.15)
3) 石原 ひでみ:FPGA マガジン No.5 Linux/Android x FPGA,CQ 出版社(2014.5.1)
4) 森岡 澄夫:HDL による高性能ディジタル回路設計,CQ 出版社(2007.2.1)
5) 長谷川 裕恭:VHDL によるハードウェア設計入門,CQ 出版社(2004.4.15)
6) 鳥海 佳孝:世界制覇！最強 ARM2016,CQ 出版社(2016.4.1)