طراحی و پیاده‌سازی ضرب‌کنندهٔ ماتریس توسط Verilog

احمد سلیمی [۱]، کیمیا نوربخش [۱]، ساعی سعادت [۱]، علیرضا حسین‌پور [۱]
[۱] دانشگاه صنعتی شریف، دانشکده مهندسی کامپیوتر

چکیده—

کلمات کلیدی—

۱. مقدمه

۲. معماری سیستم

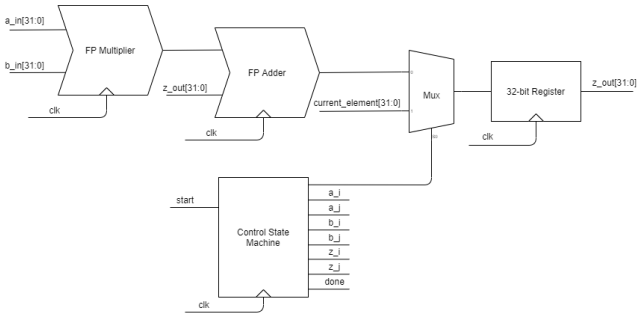
معماری این سیستم، از سه لایهٔ اصلی تشکلی شده‌است. در ادامه، معماری و جزئیات هر یک از این لایه‌ها، توضیح داده شده است.
برای ارتباط بین تمامی ماژول‌ها، برای اطمینان از این که ورودی و خروجی‌ها هنگام استفاده شدن تغییر نمی‌کنند و مقدار صحیحی دارند، برای هر کدام دو سیگنال stable و acknowledge در نظر می‌گیریم. نحوه‌ی استفاده از آن‌ها بدین گونه است که ماژولی که مقدار را دارد و می‌خواهد آن‌را پاس بدهد، با استفاده از سیگنال stable به ماژول گیرنده اعلام می‌کند که ورودی آماده‌ی استفاده است، سپس ماژول گیرنده با استفاده از سیگنال acknowledge اعلام می‌کند که ورودی را با موفقیت دریافت کرده و ماژول فرستنده می‌تواند آن را تغییر دهد.

۲.۱. *ضرب‌کنندهٔ ماتریس ترتیبی*

در این ماژول مانند ضرب ماتریسی عادی، دو ماتریس $m \times m$ را در هم ضرب می‌کنیم. می‌دانیم که برای به دست آوردن درایه ij حاصلضرب، باید سطر i ام ماتریس اول را در ستون j ام ماتریس دوم ضرب کنیم. برای این موضوع به ازای هر $1 \le i \le m, 1 \le j \le m$، داریم:

$$R_{ij} = \sum_{k=0}^{m} A_{ik} \times B_{kj}$$

که در آن، R ماتریس $m \times m$ حاصلضرب است.
در این ماژول برای محاسبه جمع و ضرب‌ها، از ماژول‌های جمع‌کننده و ضرب‌کننده اعشاری [۱] استفاده می‌کنیم.
ماژول ضرب‌کنندهٔ ماتریس ترتیبی [۲] این فرایند را در



شکل ۱: بلوک دیاگرام ضرب‌کنندهٔ ماتریس ترتیبی.

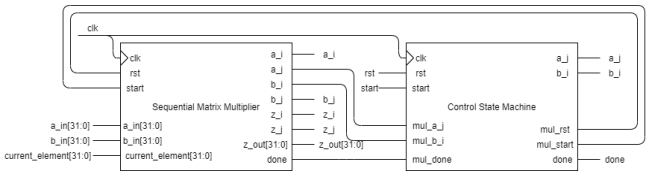
قالب یک ماشین حالت انجام می‌دهد. برای محاسبه درایه i, j ام، یک accumulator برای نگه‌داری جواب نهایی در نظر می‌گیریم و سپس به ازای هر k، ابتدا با استفاده از ماژول FP_multiplier حاصل $A_{ik} \times B_{kj}$ را محاسبه می‌کنیم و با استفاده از ماژول FP_Adder ، به ازای k های مختلف جواب را آپدیت می‌کنیم.

شکل ۲ بلوک دیاگرام ضرب‌کنندهٔ ماتریس ترتیبی را نشان می‌دهد. باید توجه کرد که حافظه‌ای که حاوی ماتریس‌های ورودی و ماتریس جواب است، در خارج این ماژول قرار دارد. در نتیجه، واحد کنترل در این ماشین حالت محدود [۳] اندیس‌های درایه‌های موردنیاز خود، یعنی $a_i, a_j, b_i, b_j, z_i, z_j$ را تعیین می‌کند، و مقادیر مربوط به هر درایه در ماتریس‌های ورودی در a_{in} و b_{in} قراره گرفته، و مقدار z_{out} نیز در ماتریس جواب قرار داده‌می‌شود.

۲.۲. *ضرب‌کنندهٔ ماتریس سطری در ستونی*

وظیفهٔ این ماژول، این است که با استفاده از یک ماژول ضرب‌کنندهٔ ماتریس ترتیبی، حاصل‌ضرب یک ماتریس سطری $m \times n$ در یک ماتریس ستونی $n \times m$ را محاسبه‌کند. حاصل این ضرب، یک ماتریس $m \times m$ خواهد بود.
برای انجام این کار، ابتدا ماتریس $m \times n$ خود را به $\lceil \frac{n}{m} \rceil$ تا

[1] floatng point adder and multiplier
[2] sequential matrix multiplier
[3] Finite State Machine



شکل ۲: بلوک دیاگرام ضرب‌کنندهٔ ماتریس سطری در ستونی.

ماتریس $m \times m$ کنار هم تقسیم‌بندی می‌کنیم. مشابها ماتریس $n \times m$ خود را نیز به $\lceil \frac{n}{m} \rceil$ تا ماتریس $m \times m$ بالای هم تقسیم‌بندی می‌کنیم. حال با استفاده از ماژول ضرب‌کننده ماتریس ترتیبی و با توجه به این که قاعده ضرب بلوکی در ماتریس‌ها برقرار است، هر یک از این ماتیس‌های $m \times m$ را به مانند یک عدد در نظر می‌گیریم و ماتریس‌های متناظر را در هم ضرب می‌کنیم.

۲.۳. *ضرب‌کنندهٔ ماتریس موازی*

۳. شبیه‌سازی و نتایج

۴. سنتز و نتایج

۵. نتیجه‌گیری

در صورتی که مقدار n بر m بخش پذیر نباشد، ماتریس را به جای $n \times n$ ، $\lceil \frac{n}{m} \rceil \times m$ در نظر می‌گیریم و درایه‌های اضافی را با ۰ پر می‌کنیم. در نظر می‌گیریم و درایه‌های اضافی را با ۰ پر می‌کنیم در نهایت برای ذخیره کردن جواب اندیس‌ها از n فراتر نخواهند رفت در نتیجه جواب $n \times n$ باقی خواهد ماند.

parallel_matrix_multiplier Project Status			
Project File:	Matrix_multiplier.xise	**Parser Errors:**	No Errors
Module Name:	parallel_matrix_multiplier	**Implementation State:**	Synthesized
Target Device:	xc6slx150-3fgg484	• **Errors:**	No Errors
Product Version:	ISE 14.7	• **Warnings:**	122 Warnings (0 new)
Design Goal:	Balanced	• **Routing Results:**	
Design Strategy:	Xilinx Default (unlocked)	• **Timing Constraints:**	
Environment:	System Settings	• **Final Timing Score:**	

Device Utilization Summary (estimated values)			[-]
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slice Registers	2932	184304	1%
Number of Slice LUTs	4508	92152	4%
Number of fully used LUT-FF pairs	2261	5179	43%
Number of bonded IOBs	40	338	11%
Number of BUFG/BUFGCTRLs	1	16	6%
Number of DSP48A1s	16	180	8%

Detailed Reports					[-]
Report Name	**Status**	**Generated**	**Errors**	**Warnings**	**Infos**
Synthesis Report	Current	Sat Feb 6 17:29:47 2021	0	122 Warnings (0 new)	423 Infos (0 new)
Translation Report					
Map Report					
Place and Route Report					
Power Report					
Post-PAR Static Timing Report					
Bitgen Report					

Secondary Reports		[-]
Report Name	**Status**	**Generated**
Post-Synthesis Simulation Model Report	Current	Sat Feb 6 17:45:44 2021

Date Generated: 02/06/2021 - 17:50:30