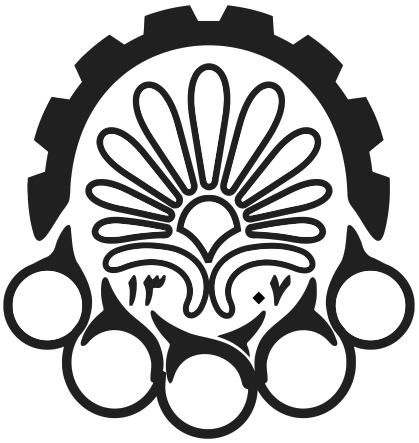
طراحی سیستم‌های قابل بازپیکربندی

دکتر صاحب‌الزمانی



دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

تمرین سری اول

۲۹ مهر ۱۴۰۳

طراحی سیستم‌های قابل بازپیکربندی

تمرین سری اول

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

دانشکده مهندسی کامپیوتر

سوال اول

با ذکر دلیل بیان کنید جملات زیر صحیح هستند یا خیر.

۱. در یک پروژه با زمان محدود بهترین راه جهت پیاده‌سازی الگوریتم پردازشی استفاده از تراشه‌های قابل بازپیکربندی است.

پاسخ

۲. طراحی‌های مبتنی بر پردازنده‌های همه منظوره و تراشه‌های خاص منظوره، دو انتهای بردار کارآیی و انعطاف‌پذیری هستند.

پاسخ

۳. معماری قابل بازپیکربندی جهت حل مشکل دسترسی حافظه در کامپیوتر فون نیومن ارائه شده است.

پاسخ

۴. در کاربردهای فضایی و محیط‌های دارای تشعشعات زیاد، تراشه‌های مبتنی بر FLASH بهترین گزینه انتخابی هستند.

پاسخ

۵. از تراشه‌های مبتنی بر آنتی‌فیوز به دلیل مقاومت مناسب در برابر دمای بالا در کاربردهای صنعتی استفاده می‌شود.

پاسخ

۶. تراشه‌های CGRA با دارابودن واحدهای خاص منظوره بیشتر، توان کمتری نسبت به FPGAها دارند.

پاسخ

۷. استفاده از FPGAها در مقایسه با تولید یک تراشه خاص باعث کاهش هزینه تولید محصول خواهد شد.

پاسخ

۸. یک ASIC همواره سریع‌تر از یک FPGA دستورات پردازشی سطح بالا را انجام خواهد داد.

پاسخ

۹. افزایش تعداد ورودی یک LUT همواره باعث افزایش سرعت مدار پیاده‌سازی شده با استفاده از آن خواهد شد.

پاسخ

۱۰. بلوک‌های UltraRAM درکنار بلوک‌های DSP برای پیاده‌سازی الگوریتم‌های هوش مصنوعی به کمک FPGA خانواده Zynq بسیار مناسب هستند.

پاسخ

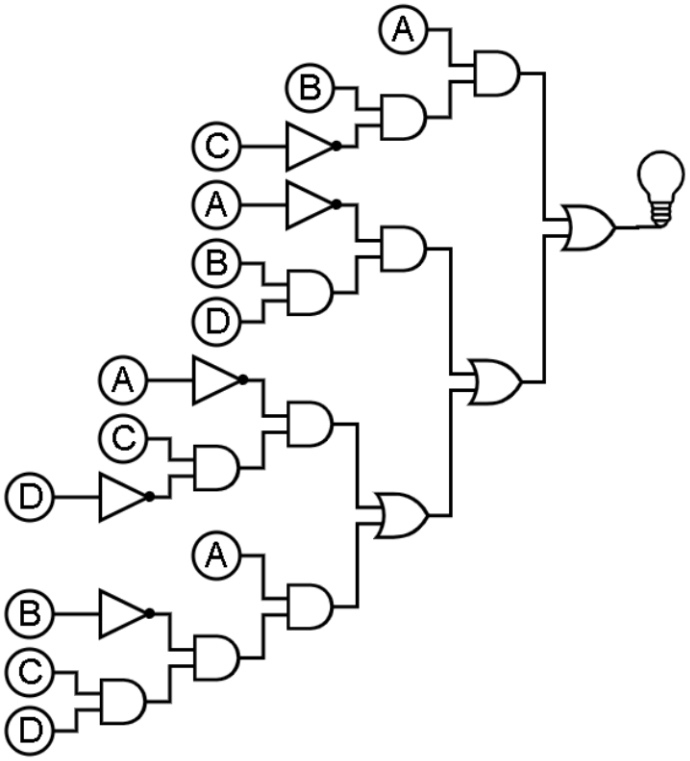
سوال دوم

در یک سیستم ایمنی مرتبط با خودرو نیاز به طراحی یک سیستم ایمنی با قابلیت اطمینان بالا می‌باشد که بایستی دارای امکان به‌روزرسانی الگوریتم ایمنی نیز باشد. همچنین زمان عملکرد سیستم نیز بایستی به صورت Hard Real-time باشد. برای طراحی این سیستم در صورت نمونه‌سازی و در صورتی که ۱ میلیون نسخه از آن نیاز باشد استفاده از چه نوع بستر پردازشی را پیشنهاد می‌نمایید؟ برای انجام محاسبات، هزینه‌های مربوط به ساخت معماری پیشنهادهای خود را از اینترنت استخراج نمایید.

پاسخ

سوال سوم

می‌خواهیم مدار زیر را یک بار با LUTهای ۳ ورودی و بار دیگر با LUTهای ۴ ورودی پیاده‌سازی کنیم به طوری که در هر حالت تعداد LUTهای مورد استفاده کمینه باشد.



شکل ۱: مدار مورد نظر

پاسخ

سوال چهارم

معماری سوئیچ‌های Wilton و Disjoint را توضیح داده و میزان F_s را در هر یک گزارش نمایید. آیا معماری دیگری برای اتصال سوئیچ‌ها می‌شناسید؟

پاسخ

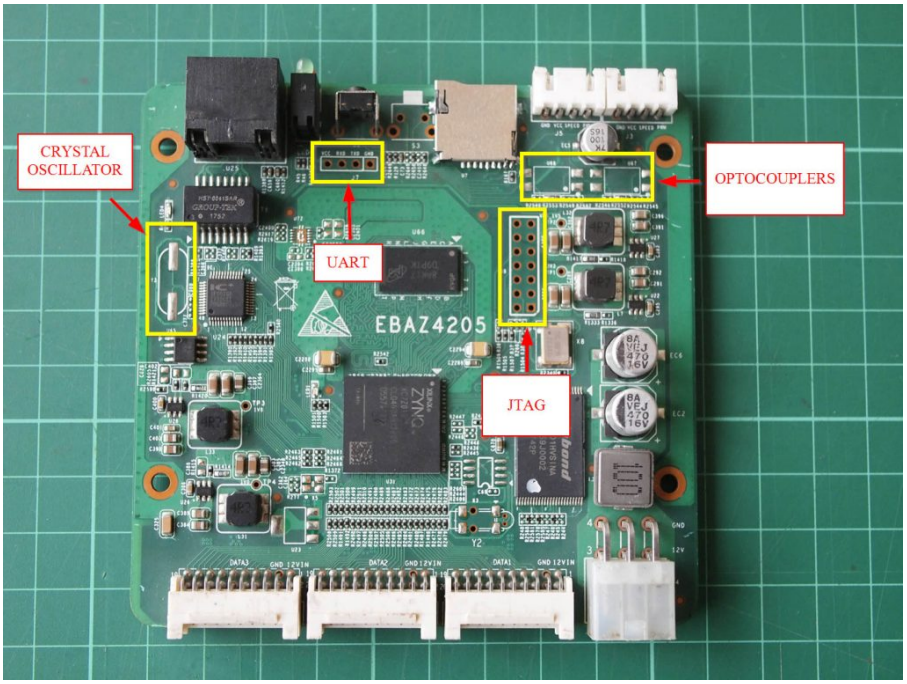
سوال پنجم

آشنایی اولیه با ابزار ویوادو: در این درس دانشجویان با استفاده از ابزار ویوادو از شرکت زایلینکس به انجام پروژه‌ها خواهند پرداخت. هدف از انجام پروژه‌ها، آشنایی عملی با طراحی توأم بر روی سیستم‌های قابل بازپیکربندی است. برای این منظور در این بخش در ابتدا دانشجویان می‌بایست نرم‌افزار ویوادو را بر روی سیستم خود نصب کنند. سپس با بررسی لینک زیر در ارتباط با نحوه طراحی توأمان و نحوه کار با ابزار آشنایی لازم را کسب کرده و توضیحات موردنیاز را در ارتباط با این نوع طراحی ارائه دهند.

- Link (I)
- Link (II)

پروژه مشابه موارد یاد شده در دو ویدئو نیز بایستی به همراه پاسخ تمرین‌ها بارگذاری شود. جهت دانلود نرم‌افزار ویوادو از این لینک استفاده نمایید. نسخه پیشنهادی ۲۰۲۰.۲ به بعد می‌باشد. به دلیل مشکل احتمالی در فعال‌ساز بهتر است از نسخه ۲۰۲۴ استفاده نشود.

همانطور که در ویدئو نیز بیان شد، هدف در این قسمت، طراحی Co-Design است. بدین منظور، برای طراحی یک گیت NAND ساده، گیت AND را با استفاده از Logic Block های FPGA طراحی می‌کنیم و ماژول NOT را در هسته پردازشی یعنی CPU طراحی کرده و اتصالات بین این دو طراحی را برقرار می‌کنیم.
ذکر این نکته الزامی است که در این تمرین ما از بورد EBAZ4205 که تراشه موجود بر روی آن Zynq 7000 است استفاده نمودیم. این آیسی در نرم‌افزار Vivado با پارت‌نامبر xc7z010clg400-3 شناخته می‌شود. تصویر این برد در «شکل ۲» آورده شده است:



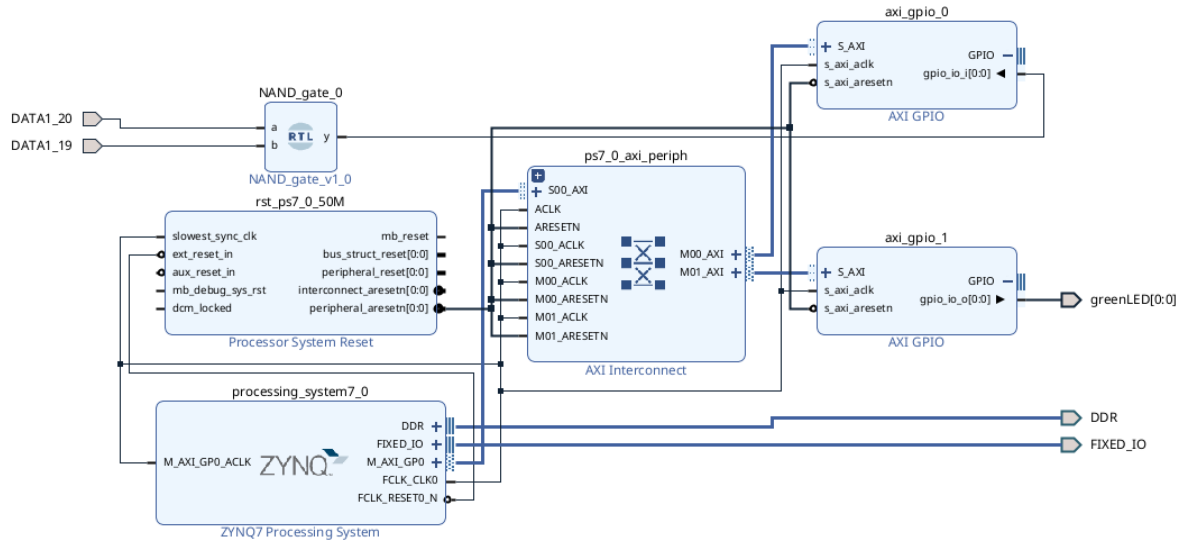
شکل ۲: بورد مورد استفاده در این تمرین

همچنین فایل Constrain مربوط به این بورد را می‌توان از اینجا دانلود کرد.
در ابتدا پس از نصب نرم‌افزار و انتخاب آیسی، طراحی سمت PL را انجام می‌دهیم. در این قسمت صرفا یک گیت AND را طراحی می‌کنیم. کد نوشته شده برای ماژول AND به‌صورت زیر است:

Listing 1: AND Module for PL

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;


entity NAND_gate is
    Port ( a, b: in std_logic;
           y: out std_logic );
end NAND_gate;

architecture Behavioral of NAND_gate is

begin

    y <= a and b;

end Behavioral;
```

پس اط طراحی ماژول AND می‌بایست نحوه Interconnection سمت PL و PS را در درون تراشه برقرار کنیم.
بدین منظور از قسمت طراحی دیاگرامی Vivado پردازنده ZYNQ را انتخاب می‌کنیم و همچنین از ماژول AND خودمان نیز یک بلوک می‌سازیم.
با استفاده از بلوک‌های AXI GPIO می‌توانیم ارتباطات بین PL و PS را برقرار کنیم.
درنهایت سیستم طراحی شده به‌صورت «شکل ۳» می‌شود.



شکل ۳: طراحی PL و PS انجام شده

پس از تکمیل شدن طراحی، می‌بایست گیت NOT را نیز به‌صورت نرم‌افزاری (به زبان c) طراحی کنیم و سپس طراحی را سنتز نهایی کنیم.
برای انجام این کار، نرم‌افزار Vitis Clasic را اجرا می‌کنیم و یک Application Project جدید می‌سازیم و فایل xsa. ساخته شده در مرحله قبل را به این پروژه اضافه می‌کنیم.
پس از ایجاد پروژه کد گیت NOT را به‌صورت زیر می‌نویسیم:

Listing 2: AND Module for PL

```
#include <stdio.h>
// #include "platform.h"
#include "xgpio.h"
#include "xparameters.h"
#include "xil_printf.h"
```

```c

int main()
{
    init_platform();

    XGpio input, output;
    int a;
    int y;

    XGpio_Initialize(&input, XPAR_AXI_GPIO_0_DEVICE_ID);
    XGpio_Initialize(&output, XPAR_AXI_GPIO_1_DEVICE_ID);

    XGpio_SetDataDirection(&input, 1, 1);
    XGpio_SetDataDirection(&output, 1, 0);

    /* print("debug the code"); */

    while(1)
    {
        a = XGpio_DiscreteRead(&input, 1);

        if(a == 1)
        {
            y = 0;
        }
        else
        {
            y = 1;
        }

        XGpio_DiscreteWrite(&output, 1, y);
    }

    cleanup_platform();
    return 0;
}
```

سپس بعد از سنتز کد، با قرار دادن فایل Bitstream ایجاد شده در مرحله PL به عنوان فایل پروگرام، طراحی Integrate شده را بر روی بردمان پروگرم می‌کنیم.