به نام خداوند هستی­بخش یگانه

|  |  |  |  |
| --- | --- | --- | --- |
| Arm.jpg  دانشگاه صنعتی شریف | طراحی سیستم‏های دیجیتال  Digital System Design  دانشکده مهندسی کامپیوتر  مدرس: دکتر گودرزی | | |
| نام و نام خانوادگی: نگار رحمتی، نیلوفر صالحی | | شماره دانشجویی: 88102026، 88110664 | پروژه بازی تتریس |

**پیاده‏سازی بازی تتریس روی برد FPGA**

* **شرح بازی**

بازی تتریس را همه می­شناسند. در این بازی هفت شکل وجود دارد که به صورت رندم درصفحه ظاهر می­شوند و روی شکل­­های قبلی قرار می­گیرند. هرگاه بک ردیف کامل پر شود، آن ردیف پاک می­شود و بازی تا وقتی که شکل­ها تا بالای صفحه روی هم قرار بگیرند ادامه پیدا می­کند.

* **فاز یک**

در فاز اول پروژه ما با استفاده از بردDE2 بخشی از این بازی را طراحی کردیم که شامل موارد زیر بود:

* در نظر گرفتن بخشی از صفحه نمایش برای انجام بازی. شکل­ها به صورت رندم در بالای این صفحه ظاهر می­شوند، به سمت پایین حرکت می­کنند و از آن خارج می­شوند.
* در نظر گرفتن کلیدهایی برای حرکت دادن شکل­ها به چپ، راست و چرخاندن آن و کلیدی برای افزایش سرعت حرکت شکل به سمت پایین.
* در نظر گرفتن بخشی از صفحه برای نمایش شکل بعدی که قرار است ظاهر شود.

کلیدهای استفاده شده در بازی:

- right key 🡪 حرکت شکل به راست

- left key 🡪 حرکت شکل به چپ

- up key 🡪چرخاندن شکل

- down key 🡪افزایش سرعت حرکت شکل به پایین

- r 🡪 reset

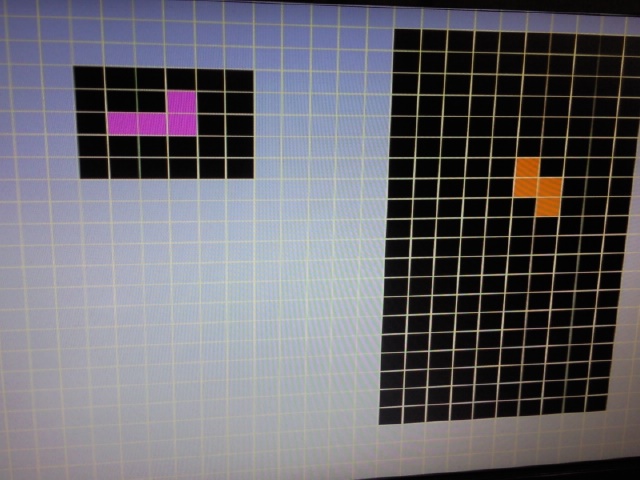
- p 🡪pause

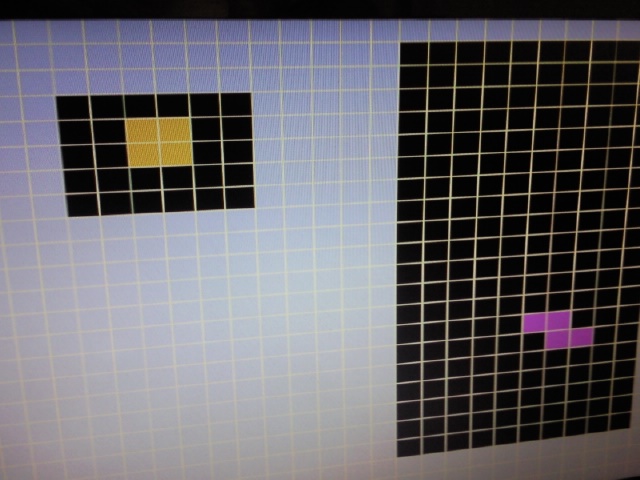
- s 🡪start

توجه!

SW[0] روی برد باید 1 باشد تا بازی شروع شود. بعد از فشردن pause بازی با s شروع می­شود.

شکل بازی در این فاز به صورت زیر است:





* **فاز دو**

در فاز دوم بازی را تکمیل کردیم تابه یک تتریس واقعی تبدیل شود. ابتدا یک ماتریس در نظر گرفتیم که با استفاده از آن شکل ها در جایی که به پایین ترین سطح می رسند نمایش داده شوند. سپس روندی طراحی کردیم که وقتی یک سطر از صفحه پر شد، آن سطر حذف شود و به امتیاز اضافه شود. در انتها امتیاز را نمایش دادیم و به عنوان نمره اضافه مرحله های مختلفی برای بازی در نظر گرفتیم که با افزایش زمان تغییر می کرد.

* **جزئیات پیاده‏سازی**
* **فاز یک**

برای پیاده­سازی بازی ابتدا به کار با برد مسلط شدیم. ماژول­های VGA ,DE2\_default و key board را از سایت DE2 board برداشتیم .

برای نمایش، از VGA در DE2\_default یک نمونه گرفتیم و ورودی­های مربوطه را طوری تعیین کردیم که در هر مرحله شکل مورد نظر را در صفحه بکشد. مختصات گوشه بالا سمت چپ شکل را با دو متغییر Xو Y در نظر گرفتیم. به طوری که X با فشردن کلید های چپ و راست تغییر کند و Y با کلاک به صورت مرتب افزایش یابد.برای مثال:

always @(posedge MYCLOCK[21])

begin

if(SW[0] || Y==20)

begin

X = 500;

end

else if(keyOut == 'b01101011 && 400<X)//left

X = X-20;

else if(keyOut == 'b01110100 )begin //right

if ( rightX<600 )

X = X+20;

end

end

برای کار با key board ابتدا خروجی­های آن را روی برد نمایش دادیم تا کد ASCII مربوط به هر کلید را بفهمیم و سپس یک خروجی از ماژول key board گرفتیم که مشخص می­کرد کاربر چه کلیدی را فشرده است.

برای رسم شکل ها مقادیری را که با توجه به X و Y محاسبه می­شد با آرایه ای از سیم ها assign کردیم. برای هر شکل اگر coord\_X و coord\_Y که VGA می­فرستد داخل محدوده شکل باشد مقدار سیم مربوط به آن شکل یک می­شود و آن نقطه به رنگ شکلی که ما می خواهیم در می آید. برای مثال:

wire [3:0]shape [6:0];

assign shape[0][0]=(Coord\_X<X+40 && Coord\_X>X && Coord\_Y>Y && Coord\_Y<Y+40);

برای کشیدن شکل بعدی در مربع جداگانه هم همین کار را کردیم.

برای این که شکل از صفحه بیرون نزند با توجه به X و Y ، مختصات چپ و راست و پایین اشکال را ذخیره کردیم.

برای این که شکل به راحتی به چپ و راست منتقل شود کلاک بازی را به یک کانتر دادیم و کلاک بخشی را که مربوط به تاثیر کلیدها بود از کلاک مربوط به پایین آمدن شکل سریع­تر در نظر گرفتیم.

برای نمایش رندم اشکال خودمان عدد رندم تولید کردیم.

* **فاز دو**

در این فاز ابتدا سعی کردیم شکل هایی که به پایین صفحه یا روی شکل دیگری می رسد را در صفحه نگه دازیم و ثابت نمایش دهیم. برای این کار یک ماتریس در نظر گرفتیم که ورودی sram را تعیین می کرد. صفحه را مربع های 20 در 20 در نظر گرفتیم و کد مپ شده به هر رنگ در خانه های ماتریس متناظر قرار دادیم.

برای هر شکل یک عدد 16 بیتی در نظر گرفتیم که متناظر با یک جدول 16 در 16 بود که مشخص می­کرد این شکل کدام خانه­ها را با توجه به Xو Y رنگی می کند .باتوجه به این عدد هر جا که لازم بود شکل ثابت شود آن را در ماتریس ذخیره ­کردیم.

در قسمت بعد ماژولی نوشتیم که خانه های ماتریس را چک می کرد، در صورت کامل بودن یک سطر همه سطرهای بالای آن را به پایین شیفت می داد و در نهایت به امتیاز اضافه می کرد.

برای نمایش امتیاز یک باکس زیر مربع شکل در نظر گرفتیم سپس یکان و دهگان امتیاز را به دست آوردیم و برای هر عدد پیکسل هایی از صفحه را که باید رنگ می شد مشخص کردیم.

* **ویژگی‏ها و ایده‏ها**
* **فاز یک**
* برای بازی یک کلید هم برای سریع پایین آمدن در نظر گرفتیم که بازی را جذاب­تر کند.
* در مواردی از دستورات data flow مانند assign استفاده کردیم که سرعت کامپایل را بسیار افزایش داد.
* با استفاده از کلاک­های مختلف حرکت شی را آسان و روان کردیم. به طوری که شکل به دلخواه کاربر و با توجه به نحوه فشردن کلید ها تند یا کند حرکت کند.
* چون $random سنتز نمی­شد خودمان عدد رندم تولید کردیم.
* **فاز دو**
* در این فاز یکی از معضلات اصلی زمان کامپایل بود. این مشکل بد از استفاده از FOR دو چندان می شد بنابراین ما کد c++ نوشتیم که حلقه های FOR را باز کند و معادل آن را چاپ کند. کد معادل را در QUARTUS کپی کردیم و زمان کامپایل خیلی کم شد.
* برای نمایش امتیاز اعداد را مانند SEVEN SEGMENT در نظر گرفتیم و هرکدام را متناظر با چند پاره خط در صفحه در نظر گرفتیم. به این صورت نمایش آن ها در صفحه نمایش آسان تر شد.
* **شرح وظایف**

برای پیاده سازی بازی هر دوی ما در هر زمان ممکن در آزمایشکاه حاضر می­شدیم تا بخش­هایی از بازی را پیاده­سازی کنیم و بلافاصله روی برد امتحان کنیم. منطق کلی بازی را به کمک هم طراحی کردیم. برای بخش­های دیگر تقسیم وظایف در مجموع به صورت زیر بود:

* **فاز یک**
* نیلوفر صالحی: راه اندازی نمایش گر و key board، مشخص کردن رنگ­ها، تنظیم کلاک ها، بیرون نزدن شکل از صفحه، چپ و راست شدن شکل.
* نگار رحمتی: چرخیدن اشکال ، راه اندازی ,reset pause و start ، نمایش شکل بعدی در قسمت مربوطه، پایین آمدن شکل.
* **فاز دو**
* نیلوفر صالحی: نمایش اشکال در صفحه و متناظر کردن آن ها با یک جدول 16 در 16.
* نگار رحمتی:حذف کردن خط و افزایش امتیاز. نمایش اعداد در صفحه.