دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر
گزارش پروژه درس ساختار و زبان کامپیوتر

عنوان:

پروژه چهارم ، بازی دایناسور خودکار

Automated Chrome Dino Game using Arduino

نگارش

مایده حیدری و مریم شیران و امیر محمد کوشکی

استاد

دکتر اسدی

بهمن ۱۴۰۱

۱ شرح کلی پروژه

در مرورگر کروم هنگامی که اینترنت قطع میشود بازی دایناسور دینو نمایش داده میشود در این بازی دایناسور در حال حرکت است و به هنگام نزدیک شدن به موانع باید از آنها بپرد تا با آنها برخورد نکند پرش با استفاده از کلید اسپیس صورت میگیرد با برخورد به موانع بازی متوقف شده و امتیاز نهایی نمایش داده میشود در بخش دوم به شرح قسمت عملی پروژه و در بخش سوم به شرح کد میپردازیم.

۱-۱ وسایل مورد نیاز

۱. برد UNO Arduino
۲. یک عدد موتورservo
۳. یک عدد LDR
۴. یک مقاومت K۱۰
۵. برد بورد

۱-۲ شرح قسمت عملی پروژه

.

لپ تاپ وArduino برد را از طریق کابل Type A متصل میکنیم ، برق مورد نیاز برد و دستورات ارسالی به آن برای اجرا از طریق این کابل تامین میشوند. در ادامه گراند و VDD را از طریق یکی از دو پورت گراند وArduino برد و پورت ۵ ولتی وArduino برد میگیریم : پورت ۵ ولتی Arduino برد را از طریق سیم به منفی برد بورد وصل میکنیم و نیز یکی از دو پورت گراند Arduino برد را از طریق سیم به مثبت برد بورد وصل میکنیم تا بتوانیم در ادامه هرگاه نیاز به VDD (ولتاژ ۵ High volt) داشتیم از منفی برد بورد و هر گاه نیاز به گراند داشتیم از مثبت برد بورد سیم بگیریم . موتور یک رشته سه سیمی دارد که یکی از آنها را به VDD و یکی دیگر را به گراند (برای تامین برق موتور اینکار را انجام داده و نیز اتصال به VDD و گراند را از طریق برد بورد انجام میدهیم) و آخری را به پورت شماره ۹ وصل میکنیم ، پورت pwd که به موتور سیگنال روشن و خاموش شدن را برساند تا در زمان مناسب موتور کلید اسپیس را فشار دهد و دایناسور دینو با موفقیت از روی مانع بپرد. LDR با تشخیص سطح روشنایی به برد کمک میکند تا موانع را تشخیص دهد ،

آن را بر روی صفحه نگه میداریم و آن مرتبا در حال ارسال اندازه levels light است که عدد ارسالی هنگام گذر درختان از زیر آن تفاوت زیادی با حالت نرمال (صفحه سفید) دارد و بدین وسیله درختان دیتکت میشوند یک سر LDR از طریق برد بورد به VDD متصل است و سر دیگر آن از طریق برد بورد به وسیله سیم دیگری به پورت A۰ برد آردواینو متصل است (LDR از طریق این connection اندازه levels light را به Arduino بورد میدهد) برای کنترل جریان خروجی از LDR نیز از یک مقاومت ۱۰ اهمی استفاده میکنیم که یک سر آن هم ولتاژ با خروجی LDR بوده و سر دیگر آن از طریق برد بورد به وسیله سیم به گراند متصل است .

مد نظر داشته باشید به هنگام اجرا LDR باید به صورت مناسب بر روی صفحه نمایش قرار گیرد تا کاکتوس ها از زیر آن رد شوند و موتور سروو نیز باید پایین کلید اسپیس قرار گیرد تا با چرخش آن را بفشارد.

۱-۳ شرح کد

در آی دی ای آردواینو شروع به زدن کد زیر میکنیم کتابخانه سرو را اینکلود میکنیم این کتابخانه به ما کمک میکند تا در ادامه شی ای از سروو بسازیم و تابع های آن به عنوان مثال چرخش سروو استفاده کنیم

```
include <Servo.h>

Servo myservo;
```

یک شی سروو به نام مای سروو میسازیم حداکثر ۱۲ تا از این اشیا میتوان ساخت همان طور که در قسمت عملی توضیح داده شد LDRINPUT به پورت ۱۰ متصل است ، A ۰ یک ماکرو است که معادلا مقدار A۰ را برابر ۱۰ قرار میدهد ، در خط زیر به برد اطلاع میدهیم که ورودی LDR از طریق پورت ۱۰ ام به آن داده میشود .

```
int LDRInput=A0; //Set Analog Input A0 for LDR.
```

موتور سرو یک نقاله بر روی خود دارد که هنگامی که به آن دستور چرخش داده شود به زاویه ی مورد نظر در آن نقاله میرود چون در ابتدا خودمان پره های سروو را به آن وصل میکنیم و یا ممکن

است از دفعات قبلی در وضعیتی مانده باشد که ما از آن اطلاع نداریم وضعیت را برابر ۰ میگذاریم تا در ادامه از طریق آن به زاویه ۰ نقاله چرخش داشته باشیم . درباره باقی متغیرها در ادامه توضیح داده خواهد شد.

```
int pos = ۰;
bool clicked = false;
int clickTime = ۰;
int interval = ۳۰۰;
int threshold = ۷۰۰;
```

برای اجرای برنامه باید یک setup اولیه صورت گیرد و در ادامه یک حلقه خواهیم داشت که برنامه همواره در حال چرخش در آن حلقه خواهد بود در setup اولیه مشخص میکنیم که قرار است از چه پین هایی برای ورودی و از چه پین هایی برای خروجی استفاده شود در اینجا فقط ورودی داریم اما در قسمت امتیازی خواهیم دید که خروجی هم خواهیم داشت .

همان طور که در قسمت عملی توضیح داده شد موتور سروو از طریق یکی از سه سیم آن به برد متصل متصل است تا دستورات چرخشی از طریق آن به موتور داده شود در setup شماره پینی که به آن متصل است را نیز میدهیم ، وضعیت را برابر نود قرار داده و دستور قرار گیری سروو در آن وضعیت را میزنیم. در ابتدا دستورات به برد داده شده و ادامه نیز خواهیم دید که نیاز داریم اعدادی را بر روی صفحه پرینت کنیم در واقع نیاز به نوشتن و خواندن از برد داریم برای اینکار باید از پروتکل ۹۶۰۰ سریال استفاده کنیم پس آن را هم در setup مینویسیم در انتهای قسمت setup دستور میدهیم که اجرا برای مدت کمی قطع شود این برا این است که سروو زمان کافی برای چرخش و قرار گیری در وضعیت اولیه نود درجه داشته باشد و کاربر نیز زمان کافی برای قرار دهی LDR بر روی صفحه و شروع بازی داشته باشد.

```
void setup()
Serial.begin(۹۶۰۰) ;
pinMode(LDRInput،INPUT);
myservo.attach(۹) ;
pos = ۹۰ ;
myservo.write(pos);
delay(۱۰۰۰) ;
```

برد پس از اجرای setup شروع میکند به اجرای قسمت loop و در این قسمت میماند یعنی هر بار پس از پایان آن دوباره آن را شروع میکند . در خط اول از تابع analogRead استفاده میکنیم تا از طریق LDRInput سطح روشنایی را بخواند و آن را در متغیر value ذخیره میکنیم سپس مقدار آن را پرینت میکنیم در واقع آنچه که سبب پرش دایناسور میشود تفاوت مقدار value روی بگگراند و موانع است با استفاده از این پرینت ما این مقادیر را پیدا میکنیم خواهیم دید که این مقادیر برای بک گراند حدود ۸۰۰ و برای موانع حدود ۶۰۰ هستند از این رو مقدار threshold را برابر ۷۰۰ تعریف میکنیم این یکی از همان متغیر هایی بود که گفته شده بود در ادامه توضیح داده میشود . حال در ادامه اگر مقدار value از threshold بیشتر باشد یعنی LDR در حال اسکن پس زمینه میباشد و اگر کمتر باشد یعنی در حال اسکن مانع کاکتوس میباشد لازم به ذکر است این عدد ۷۰۰ وابسته به نور محیط و نور صفحه نمایش امکان تغییر دارد .

```
void loop()
int value=analogRead(LDRInput)؛
Serial.println(”LDR reading: ” + String(value))؛
```

clicked و clickedTime نیز دو متغیر دیگر بودند که گفتم در ادامه توضیح میدهم فلسفه وجود این متغیر ها به مشکل زیر بر میگردد . شاخه های کاتوس ها و کاکتوس های کنار هم ، در واقع مشاهده کردیم که گاهی به این مشکل بر میخوریم که در حین اسکن یک مانع واحد پس زمینه سفید و دوباره سیاه میشود و این سبب میشد موتور سروو چرخش قبلی خود را بعضا نا کامل رها کند و زود ببازیم و از این رو این دو متغیر را ایجاد کردیم اولی یک متغیر بول است که مقدار اولیه ی غلط دارد و دومی نیز یک عدد صحیح است که مقدار اولیه ۰ دارد. وقتی به شروع یک مانع برسیم وارد به دلیل کمتر بودن مقدار value از threshold وارد این قسمت شده و چون clicked غلط است وضعیت برابر ۱۳۵ درجه شده و سرو به آن وضعیت چرخش میکند حال تابع millis() فراخوانی میشود این تابع زمان فعلی را از لحظه اجرای برنامه بر حسب میلی ثانیه را در clickedTime ذخیره میکند. حال چون که سروو چرخید به زاویه ۱۳۵ درجه و بر روی اسپس کلیک کرد clicked را برابر صحیح قرار میدهیم حال لوپ تمام شده و دوباره اجرا میشود اگر همچنان بر روی مانع باشیم وارد if اول میشود ولی چون clicked را برابر صحیح است وارد if دوم نشده و کلیکی صورت نمیگیرد همان طور ادامه پیدا میکند تا به پس زمینه برسیم وقتی به پس زمینه رسیدیم از کجا بفمهیم که مانع

واقعا تمام شده و یا این در حین همان مانع است ؟ مثلا از کاکتوس به کلی عبور کردیم و یا در بین شاخه های کاکتوس هستیم ؟ فرض کنید به پس زمینه رسیده ایم پس مقدار آن از threshold بیشتر است و وارد else میشود در اینجا بررسی میشود که آیا تفاضل زمان فعلی با زمان ورود به مانع از interval بیشتر است یا خیر اگر بیشتر باشد clicked برابر false شده بدین معنی که مانع تمام شده است . در واقع هدف ما این بود مقداری برای interval پیدا کنیم که حدودی از طول موانع باشد با تلاش های مختلف مقدار ۳۰۰ را پیدا کردیم بدین معنی که بازه زمانی که در حال اسکن مانع هستیم حداکثر ۳۰۰ میلی ثانیه میباشد و فاصله ی زمانی اسکن دو مانع مجزا حداقل ۳۰۰ میباشد. حال اگر آن تفاضل از ۳۰۰ بزرگ تر باشد یعنی که مانع تمام شده و وضعیت clicked تغییر میکند تا دفه بعدی که به مانع میرسیم کلیک رخ دهد و اگر کمتر مساوی باشد تغییری نخواهیم داشت و فرض بر این است که در حال اسکن همان مانع هستیم ولی مثلا در بین شاخه های کاکتوس ها قرار داریم .

```
    if(value<threshold)



    Serial.println(String(value))؛
if (!clicked)


    pos = ۱۳۵؛
myservo.write(pos)؛
delay(۱۵۰) ؛
pos = ۹۰؛
myservo.write(pos)؛
clickTime = millis()؛
clicked = true؛


else if (millis() - clickTime >= interval)
clicked = false؛
```

۲ شرح قسمت امتیازی

اگر سایت https://chromedino.com/ را نگاه کنید میبینید که حالت های دیگری نیز برای بازی وجود دارد که رنگ موانع و پس زمینه ها تغییر میکند مانند Night و یا Color ، عددی که LDR در حالت Color برای پس زمینه و موانع دیتکت میکند تقریبا یکسان است و لذا در این حالت موانع و پس زمینه قابل تشخیص نیستند اما در حالت Night اعداد پس زمینه و موانع مانند قبل تفاوت زیادی دارند و قابل تشخیص اند.
حال میخواهیم کاری کنیم که برنامه ما جدای از حالت بازی درست کار کند از طرفی در قسمت قبلی ما عدد ۷۰۰ را به صورتی دستی پیدا و تنظیم میکردیم و گفتیم که در شرایط مختلف نیز متفاوت است این عدد برای حالت Night نیز در شرایط مختلف متفاوت خواهد بود . میخواهیم کاری کنیم که نیاز به تنظیم دستی این عدد نیز نباشد و با گرفتن LDR به مدت ۴ ثانیه بر روی پس زمینه و ۴ ثانیه بر روی موانع عدد threshold به صورت خودکار تنظیم گردد.

۲-۱ وسایل مورد نیاز

همان وسایل قبلی به همراه یک مقاوت ۵۰۰ اهمی و یک لامپ LED

۲-۲ شرح قسمت عملی

تمامی کانکشن ها مانند قبل میباشد تنها با این تفاوت که یک LED اضافه شده است پایه کاتد LED را از طریق برد بورد به گراند متصل کرده و پایه آند آن را از طریق برد بورد به یک سر مقاومت متصل کرده و سر دیگر مقاومت را به پین شماره ۱۱ متصل کرده ایم علت استفاده از مقاومت این است که ۵ ولت برای LED زیاد است و ممکن است بسوزد .
چرا پین شماره ۱۱ : پین شماره ۱۱ این قابلیت را دارد که زمان هاییی که ما میخواهیم ولتاژ High بدهد و زمان هایی که نمیخواهیم ولتاژ low بدهد ، در واقع ما میخواهیم روی خاموش و روشن شدن LED کنترل داشته باشیم .

۲-۳ شرح کد

کلیت کد بسیار شبیه قبل است و تنها قسمت های جدیدی به آن اضافه شده ، در اینجا تنها به توضیح قسمت های جدید میپردازیم:

```
include <Servo.h>

Servo myservo;

int LDRInput=A۰;
int pinled = ۱۱;
```

همانطور که در ۲-۲ شرح داده شد از طریق پین شماره ۱۱ به یک سر مقاومت جریان داده میشود و سر دیگر آن به پایه آند LED متصل است در خط چهارم برنامه این را مشخص میکنیم .

متغیر های background و obstacle و متفیر بول darkmode را نیز تعریف میکنیم. اینکه چرا background و obstacle به صورت long long تعریف شده اند به این دلیل است که از int اور فلو میکرد لازم به ذکر است که int در آردواینو ۲ بایتی میباشد.

```
int pos = ۰;
bool clicked = false;
int clickTime = ۰;
int interval = ۳۰۰;
int threshold = ۰; // ۷۸۰
long long int backGround = ۰;
long long int obstacle = ۰;
bool darkmode = false;
```

LED در واقع نقش خروجی را بازی میکند در ادامه خواهید دید در ابتدا به مدت ۴ ثانیه در حالت چشمک زن است تا کاربر آن را بر روی بکگراند قرار دهد سپس به مدت ۲ ثانیه ممتد روشن است تا کاربر آن را روی بکگراند نگه دارد و بعد دوباره به مدت ۴ ثانیه چشمک زن است تا کاربر آن را روی موانع قرار دهد و بعد دوباره به مدت ۲ ثانیه ممتد روشن است تا کاربر آن را روی مانع نگه

دارد و در ادامه خاموش خواهد شد از این رو پین آیدی آن را به عنوان خروجی ست خواهیم کرد .

```
    void setup()
Serial.begin(۹۶۰۰) ؛
pinMode(pinled، OUTPUT)؛
pinMode(LDRInput،INPUT)؛
myservo.attach(۹) ؛
myservo.write(pos = ۹۰) ؛
delay(۱۰۰۰) ؛
```

در زیر به مدت ۲۰ بار LED را روشن کرده به مدت ۱۰۰ میلی ثانیه صبر تا روشن شود بعد آن را خاموش کرده و به مدت ۱۰۰ میلی ثانیه صبر کرده تا خاموش شود در واقع LED به مدت ۲۰*۲۰۰ میلی ثانیه معادل ۴ ثانیه چشمک میزند .

```
    for (int i = ۰؛ i > ۲۰؛ i++)
digitalWrite(pinled، HIGH)؛
delay(۱۰۰) ؛
digitalWrite(pinled، LOW)؛
delay(۱۰۰) ؛
```

در زیر به آردواینو میگوییم که خروجی High بدهد و ۴۰ بار میزان value را خوانده و به متغیر backgrand اضافه میکنیم و هر بار ۵۰ نانو ثانیه صبر میکنیم در واقع ایده این است که ما با یک بار خواندن بکگراند و یک بار خواندن مانع احتمال خطا داریم سعی میکنیم با ۴۰ با خواندن هر یک و میانگین گفتن هر کدام خطا را کم کنیم در اینجا به مدت ۴۰*۵۰ نانوثانیه معادل ۲ ثانیه لامپ روشن خواهد بود .

```
    digitalWrite(pinled، HIGH)؛
for (int i = ۰؛ i > ۴۰؛ i++)
int value=analogRead(LDRInput)؛
Serial.println(”background value reading: ” + String(value))؛
backGround + = value؛
```

```
delay(۵۰) ؛
```

در زیر میانگین مقدار value برای بک گراند را محاسبه میکنیم چیزی که شاید عجیب باشد کست کردن این مقدار به int و سپس کست کردن آن به String برا پرینت است دلیل این امر این است که کست مستقیم int long long به String ممکن نیست

```
    digitalWrite(pinled، LOW)؛
backGround / = ۴۰؛
Serial.println(”Final background Calibration: ” + String(int(backGround)))؛
```

در زیر به مدت ۲۰ بار LED را روشن کرده به مدت ۱۰۰ میلی ثانیه صبر تا روشن شود بعد آن را خاموش کرده و به مدت ۱۰۰ میلی ثانیه صبر کرده تا خاموش شود در واقع LED به مدت ۲۰*۲۰۰ میلی ثانیه معادل ۴ ثانیه چشمک میزند .

```
    for (int i = ۰؛ i > ۲۰؛ i++)
digitalWrite(pinled، HIGH)؛
delay(۱۰۰) ؛
digitalWrite(pinled، LOW)؛
delay(۱۰۰) ؛
```

در زیر به آردواینو میگوییم که خروجی High بدهد و ۴۰ بار میزان value را خوانده و به متغیر اضافهobstacle میکنیم و هر بار ۵۰ نانو ثانیه صبر میکنیم در اینجا به مدت ۴۰*۵۰ نانوثانیه معادل ۲ ثانیه لامپ روشن خواهد بود .

```
    digitalWrite(pinled، HIGH)؛
for (int i = ۰؛ i > ۴۰؛ i++)
int value=analogRead(LDRInput)؛
Serial.println(”obstacle value reading: ” + String(value))؛
obstacle + = value؛
```

```
delay(۵۰) ؛
```

در زیر میانگین مقدار value برای موانع را محاسبه میکنیم چیزی که شاید عجیب باشد کست کردن این مقدار به int و سپس کست کردن آن به String برا پرینت است دلیل این امر این است که کست مستقیم int long long به String ممکن نیست.

```
digitalWrite(pinled، LOW)؛
obstacle / = ۴۰؛
Serial.println(”Final Obstacle Calibration: ” + String(int(obstacle)))؛
```

مقدار threshold را برابر میانگین background و obstacle قرار میدهیم اگر در حالت نرمال باشیم هنگامی که value از threshold کمتر است یعنی به مانع رسیدیم و باید بپریم اما اگر در نایت مود باشیم هنگامی که value از threshold بیشتر است یعنی به مانع رسیدیم و باید بپریم پس از ست کردن متغیر بولین darkmode استفاده میکنیم تا متوجه شویم که در حالت نرمال هستیم و یا نایت مود سپس به مدت ۲ ثانیه صبر کرده تا کاربر بازی را شروع و LDR را در مکان مناسب قرار دهد.

```
threshold = (obstacle + backGround) / ۲؛

darkmode = obstacle < backGround؛

Serial.println (”Darkmode is ” + String(darkmode) + ” and threshold is set
at ”+String(threshold) + ” ” ) ؛
delay(۲۰۰۰) ؛
```

کد قسمت loop همانند قبل است تنها با این تفاوت که اکنون تشخییص شروع مانع به حالت بستگی دارد همانطور که بالا تر توضیح داده شد در هر یک از حالت mode night و normal متفاوت است که این با استفاده از and و or در زیر پیاده سازی شده است.

```
    void loop()
int value=analogRead(LDRInput)؛
Serial.println(”LDR reading: ” + String(value))؛
if((!darkmode  value<threshold) || (darkmode  value>threshold))

Serial.println(String(value))؛
if (!clicked)
pos = ۱۳۵؛
myservo.write(pos)؛
delay(۱۵۰) ؛
pos = ۹۰؛
myservo.write(pos)؛
clickTime = millis()؛
clicked = true؛


else if (millis() – clickTime =< interval)
clicked = false؛
```

۳ جمع بندی

در این پروژه با استفاده از یک برد آردواینو و LDR و موتور servo بازی dino را خودکار کردیم بدین شکل که زمانی که یک درخت مشاهده می شود توسط LDR ، به صورت خودکار موتور servo، کلید space از کیبورد را فشار میدهد و دایناسور میپرد .

References