سنسورهای اثر انگشت شرکت علم و صنعت :

این سنسورها به دو دسته سنسورهای خازنی و سنسورهای نوری تقسیم می شوند . طول تمپلیت ایجاد شده و نحوه کار با این دو نوع سنسور مشابه می باشد ولی امکان استفاده از تمپلیت های تولید شده در هر نوع در نوع دیگر وجود ندارد .

طریقه استفاده از EosClock برای برقراری ارتباط و استفاده از سنسورها :

لازم به ذکر است برنامه مذکور به زبان C# و در نسخه ۲ این زبان نوشته و کامپایل شده است و توسط این نسخه از زبان و ویرایش های بالاتر قابل استفاده است.

برای برقراری ارتباط با سنسور ابتدا بایستی به ساعت متصل شد به این منظور ابتدا یک ساعت را در کد برنامه تعریف می نماییم و به آن ساعت وصل می شویم که طریقه ساخت ساعت و اتصال به آن به شکل زیر است :

- ایجاد یک Connection با استفاده از کلاس ConnectionFactory :

در این حالت با توجه به اینکه طریقه ارتباط با ساعت به چه صورت است از یکی از ۳ متد زیر استفاده می نماییم :

۱. ارتباط سریال :

```
connection = ConnectionFactory.CreateSerialConnection(
portNumber,baudRate, System.IO.Ports.Parity.None ,
readTimeout , writeTimeout , waitBeforeread);
```

۲. ارتباط شبکه :

```
connection = ConnectionFactory.CreateTCPIPConnection(ip,
portNumber , waitBeforeread, readTimeout , writeTimeout);
```

۳. ارتباط مودم :

```
connection = ConnectionFactory.CreateModemConnection(
portNumber,baudRate, System.IO.Ports.Parity.None ,
```

```
readTimeout , writeTimeout , waitBeforeread ,
telephoneNumber , isPulseDialing);
```

شرح پارامترهای استفاده شده به شرح زیر است :

`portNumber` :(نوع `int`) این پارامتر در ارتباط مودمی و سریال شماره پورت سریال را مشخص میکند و در ارتباط شبکه شماره پورت ارتباطی شبکه را.

`baudRate` :(نوع `int`) پهنای باند در ارتباط مودمی و سریال میباشد .

`parity`: این مقدار بایستی مقدار گفته شده باشد (اطلاعات مربوط به ارتباط سریال می باشد که در ساعتها به این شکل تنظیم گردیده است)

`readTimeout`:(نوع `int`)زمان مجاز برای گرفتن جواب زمان خواندن اطلاعات را در ارتباط نشان می دهد .

`writeTimeout` :(نوع `int`) زمان مجاز برای گرفتن جواب زمان نوشتن اطلاعات را در ارتباط نشان می دهد .

`waitBeforeread` :(نوع `int`) مدت زمانی که ارتباط , قبل از هر خواندن اطلاعات بایستی صبر کند را نمایش می دهد.

`Ip` :(نوع `string`) این پارامتر آدرس ساعت را در ارتباط شبکه مشخص می کند.

`telephoneNumber` :(نوع `string`) در ارتباط مودمی شماره تلفن مودم سمت ساعت را مشخص می کند .

`isPulseDialing` :(نوع `bool`) پالس بودن نوع تلفن را مشخص می نماید.

- پس از ایجاد یک شیء connection نوبت به ایجاد ساعت می رسد که روش آن به شرح زیر است :

```
device = new Clock(connection , protocolType , trtAddress ,
sensorType);
```

شرح پارامترهای استفاده شده به شرح زیر است :

connection : که شیء ارتباط ساخته شده در قسمت بالا می باشد.
protocolType : که پروتوکل ارتباطی ساعت را نشان می دهد .
trtAddress :(نوع int) آدرس trt ساعت را نشان می دهد که در ساعتهای سریال با پروتکل RS۴۸۶ کاربرد دارد
sensorType : این پارامتر پروتکل سنسور را نشان میدهد .

لازم ب ذکر است که دو پارامتر و از نوع EosClock.ProtocolType می باشند.

- بعد از این مراحل بایستی به ساعت وصل شویم و مطمئن شویم که اتصال بدرستی برقرار گردیده است :
  ۱. با استفاده از دستور Connect :

```
device.Connect();
if(!_device.Connected)
{
     MessageBox.Show("error in connection");
     return;
}
```

  ۲. با استفاده از دستور TestConnection :

```
if(!_device.TestConnection())
{
     MessageBox.Show("error in connection");
     return;
}
```

در هر دوی این موارد ارتباط با ساعت برقرار میگردد .
لازم بذکر است در ارتباط مودمی ابتدا بایستی شماره گیری انجام گیرد . این کار از طریق دستورات بالا انجام می گیرد ولی زمان بیشتری نسبت به سایر روشها دارد .

- در ادامه اتصال به سنسور توسط دستور زیر صورت می پذیرد :

```
device.ConnectToSensor();
```

یا

```
Device.Sensor.Connect()
```

در این حالت ساعت به حالت PCFPM می رود و بر روی نمایشگر ساعت پیام "منتظر باشید" نمایش داده می شود و امکان حضور و غیاب وجود ندارد .

- در ادامه عملیات مورد نظر با سنسور انجام میگردد که در انتها به آنها خواهیم پرداخت .
- پس از اتمام عملیات بایستی ارتباط با سنسور قطع شود تا ساعت بتواند به کار خود ادامه دهد و به اصطلاح از حالت PCFPM خارج شود .
- `Device.DisconnectFromSensor();`
- یا
- `Device.Sensor.Disconnect()`

- در ادامه بایستی ارتباط با ساعت نیز قطع شود تا پورت مورد استفاده آزاد شود . این کار با دستور زیر صورت می پذیرد :

```
device.Disconnect();
```

لیست عملیات سنسور :

- `ApplySettings` : برای فعالسازی تنظیمات انجام شده بر روی ساعت از این دستور استفاده می شود .
- `CancelOperation` : برای کنسل کردن دستوری که به سنسور فرستاده شده به کار میرود . لازم بذکر است عملیات `RollBack` نمی شود و تنها متوقف می شود .
- `AbortRead` : برای چشم پوشی از خواندن از سنسور به کار می رود .
- `DeleteAllTemplates` : برای پاک کردن تمامی اثر انگشتهای گرفته شده به کار میرود و امکان بازگردانی این اطلاعات وجود ندارد . در استفاده از این دستور بایستی جوانب احتیاط رعایت شود .
- `DeleteByID` : برای پاک کردن اثر انگشتهای فردی با `Id` مشخص به کار می رود .
- `DeleteByScan` : برای پاک کردن اثر انگشت فرد با اسکن استفاده می شود به این صورت که فرد انگشت خود را اسکن کرده و در صورت وجود اثر انگشت وی آن اثر انگشت پاک می شود .
- `DeleteTemplate` : برای پاک کردن اثر انگشت خاص به دوصورت اعلام `id` و اثر انگشت و یا `id` و ایندکس استفاده می شود .
- `EnrollByImage` : برای گرفتن اثر انگشت از روی عکس انگشت مورد استفاده قرار می گیرد .
- `EnrollByScan` : برای گرفتن اثر انگشت توسط اسکن انگشت مورد استفاده قرار می گیرد .

- `EnrollByTemplate` : برای گرفتن اثر انگشت توسط اثر انگشت ذخیره شده مورد استفاده قرار می گیرد.
- `GetAvailableFingerpringTemplatesCount` : تعداد ظرفیت خالی برای گرفتن اثر انگشت را نشان می دهد .
- `GetEnrolledFingerprintTemplatesCount` : برای مشخص کردن تعداد اثر انگشتهای گرفته شده مورد استفاده قرار می گیرد .
- `GetEnrolMode` : نوع اسکن را مشخص میکند که یکی از حالات زیر است :
  ۱. `SensorEnrollMode.OneTime` : حالت عادی که از هر انگشت یک اثرانگشت نگه می دارد.
  ۲. `SensorEnrollMode.TwoTime` : مانند حالت قبل با این تفاوت که برای ذخیره این اثر انگشت , انگشت را دو بار پشت سر هم اسکن می نماید.
  ۳. `SensorEnrollMode.TwoTimeWith2Request` : مشابه حالت قبلیست با این تفاوت که برای اسکن بار دوم بایستی دستور به سنسور ارسال شود .
  ۴. `SensorEnrollMode.TwoTemplates` : حالتی که ۲ اثر انگشت از یک انگشت نگه می دارد و مزیت آن این است که یکی از آنها در طول زمان با تغییرات شکل اثر انگشت بروز شده و قدرت شناسایی سنسور را افزایش میدهد.
  ۵. `SensorEnrollMode.TwoTemplatesWith2Request` : مشابه حالت قبلیست با این تفاوت که برای اسکن دوم نیاز به ارسال فرمان دارد.
- `GetFingerprintImage` : سنسور در صورت تنظیم از آخرین انگشت اسکن شده تصویری ذخیره می نماید . این کار در صورتی است که کیفیت تصویر `none` نباشد . با این دستور عکس آخرین انگشت اسکن شده برگردانده می شود .
- `GetFreeScanState` : حالت سنسور را نشان میدهد . منظور از حالت اسکن اتوماتیک برای حضور و غیاب و یا اسکن دستی می باشد
- `GetImageFormat` : فرمت عکس گرفته شده از انگشت را نشان می دهد.
- `GetUsersIdList` : لیست `id` هایی که برای آنها اثر انگشتی در ساعت موجود است را نشان می دهد.
- `GetUserTemplate` : اثر انگشت ایندکس خاص یک کاربر را برمی گرداند.
- `GetUserTemplates` : لیست اثر انگشت های یک فرد را بر می گرداند .
- `GetUserTemplatesCount` : تعداد اثرانگشت های ثبت شده برای یک فرد را بر می گرداند.
- `Identify` : برای شناسایی یک اثر انگشت توسط اسکن آن استفاده می شود .

- `IsBusy` : مشخص می کند که سنسور مشغول است یا نه .
- `IsFreeScan` : مشخص می کند که سنسور در حالت `FreeScan` قرار دارد یا نه.
- `SaveSystemParameter` : برای ذذیره سازی پارامترهای سیستم استفاده می شود.
- `SetBaudRate` : برای تنظیم پهنای باند سنسور مورد استفاده قرار می گیرد . لازم بذکر است این پهنای باند و پهنای باند ارتباط با ساعت بایستی یکسان باشد.
- `SetEnrollMode` : نوع اسکن (یکی از ۵ مورد بالا) را مشخص می کند.
- `SetFreeScanState` : حالت اسکن حضور و غیاب را تنظیم می کند
- `SetImageFormat` :فرمت تصویر انگشت (در اسکن اخر) را تنظیم می کند.
- `Verify` : برای تایید یک اثر انگشت با اسکن به کار می رود .
- `VerifyByTemplate` : برای تایید یک اثر انگشت با اثر انگشت ذخیره شده به کار می رود .