«به نام خدا»

در این جلسه نیز در ادامه‌ی مطالب جلسه پیش ، سعی می‌کنیم کمی بیشتر با نحوه برنامه نویسی در زبان C آشنا شویم .

متغیّر چیست ؟

متغیر قسمتی از حافظه است که ما برای آن یک نام دلخواه انتخاب می کنیم و از آن برای نگه داری اطلاعات مورد نیاز خود در روند اجرای برنامه استفاده می کنیم .

۱- متغیرها با خاموش شدن مدار پاک می شوند و حافظه دائمی نیستند .

۲- باید نوع اطلاعاتی که قرار است در متغیر نگه داری شود ، معلوم گردد ، مثلاً قرار است در آن عدد ذخیره شود یا حروف ، یا عدد اعشاری یا . . . . .

۳- کامپایلر به صورت خودکار بخشی از حافظه را به متغیر مورد نیاز ما اختصاص می دهد و نیازی نیست ما برای آن مشخص کنیم که اطلاعات را در کجای حافظه ذخیره کند . البته می توان در صورت نیاز آدرس بخشی از حافظه را مشخص کرد تا اطلاعات ما در آن جا ذخیره شود (که فعلاً به آن نمی پردازیم) .

تعریف متغیر

برای تعریف یک متغیر ابتدا باید نوع یا (Type) اطلاعاتی که قرار است در آن ذخیره شود ، نوشته شود ، و بعد با یک فاصله (Space) نام متغیر نوشته شود . به مثال زیر دقت کنید :

```
int a ;
```

در اینجا متغیری با نام "a" و از نوع **int**eger یا همان عددی تعریف شده است ، یعنی در این متغیر فقط می توان یک عدد صحیح (غیر اعشاری) را ذخیره کرد .

✓ **نکته :** اگر یک عدد اعشاری در آن ریخته شود ، بخش اعشاری آن حذف می شود .

هر متغیر از جنس int ، دو بایت حافظه را به خود اختصاص می دهد و می توان در آن اعداد در گستره ی۳۲۷۶۷ تا ۳۲۷۶۸- را ذخیره کرد .

برای ذخیره سازی حروف (Character) باید متغیر از نوع Char تعریف شود . متغیرهای Char یک بایت حافظه را به خود اختصاص می دهند و در آن ها می توان تنها یک حرف را ذخیره سازی کرد . حروف را در حافظه به شکل کد شده ذخیره می کنند ، نام سیستم کد کردن حروف کد اسکی (ASCII code) است .

در جدول زیر چند نوع داده (Type Data) ی دیگر نیز معرفی شده است .

Type	اندازه(size)	بازه ی تحت پوشش
Long int	۴بایت	۲۱۴۷۴۸۳۶۴۷ تا ۲۱۴۷۴۸۳۶۴۸ -
Unsigned long int	۴بایت	۰ تا ۴۲۹۴۹۶۷۲۹۵
Float	۴بایت	برای اعداد اعشاری
Unsigned int	۲بایت	۰ تا ۶۵۵۳۵

برای ذخیره سازی اطلاعات در داخل متغیرها نیز از همان عملگر "=" استفاده می کنیم . مثلاً :

```
sum۱=۷۵;
```

می توانیم متغیرها را در همان موقع تعریف مقدار دهی کنیم . به این کار مقدار دهی اولیه یا "Initialize" می گویند . مثلاً :

```
int sum۱=۷۵;
```

اطلاعاتی که در داخل متغیرها ذخیره می شود ثابت نیست و می توان در هر جای برنامه که لازم بود ، مقدار دیگری در متغیر ذخیره کرد . مثلاً :

```
int Cross۱=۳۴;
```

.

.

.

Cross۱= ۶۸;

اگر بخواهیم مقدار متغیر ثابت و غیر قابل تغییر باشد باید قبل از تعیین نوع متغیر ، کلمه ی "const" را بنویسیم . مثلاً

Const float pi=۳ .۱۴;

می توان چند متغیر را با هم تعریف کرد و آنها را مقدار دهی کرد . مثلاً :

char a۱='a', a۲, a۳, a۴='B';

✓ توجه : برای مقدار دهی متغیرهایی که از جنس "char" تعریف می شوند ، باید مقدار در داخل ' ' قرار بگیرد ، به مثال بالا دقت کنید .

قوانین نام گذاری شناسه ها (Identifiers) در زبانC

شناسه ها همان نام هایی هستند که برای متغیرها ، توابع و . . . انتخاب می شوند .

برای انتخاب یک شناسه فقط می توانیم از حروف زیر استفاده کنیم :

۱- اعداد ۹ تا ۰

۲- حروف z تا a (حروف کوچک)

۳- حروفZ تا A (حروف بزرگ)

۴- زیرخط / underline "_"

۵- علامت $

به غیر از این کاراکترها مجاز به استفاده از هیچ کاراکتر دیگری (حتی فاصله(Space)) نیستیم .

همچنین در ابتدای شناسه ها نمی توانیم از اعداد استفاده کنیم . مثلاً شناسه ۲loop غلط است ، ولی Loop۲ درست است .

طول شناسه ها نیز نمی تواند بیش از ۳۲ کاراکتر باشد .

بعضی کلمات در این زبان جزو **کلمات رزرو شده** (Reserved word) هستند و نمی توانند به عنوان شناسه استفاده شوند مانند : int, float, void, char, while, if ، و . . .

نکات مهم در مورد برنامه نویسی در زبان C

۱- در پایان هر دستور باید یک ";" گذاشته شود .

۲- جملات و عبارات غیر عددی را باید داخل " " قرار دهیم . مثلاً اگر می خواهیم کارکتر B را در داخل متغیری با نام Temp که از جنس char تعریف شده است ذخیره کنیم ، باید بنویسیم :

```
Temp='B';
```

۳- زبان C در اصطلاح یک زبان Case sensitive است ، یعنی در این زبان بین حروف بزرگ و کوچک تفاوت وجود دارد . مثلاً در یک برنامه ما می توانیم دو متغیر با نام های "temp" و "Temp" داشته باشیم که ارتباطی هم با یکدیگر ندارند .

۴- اگر بخواهیم در هر قسمت از برنامه توضیحاتی را بنویسیم ، باید یک "//" در ابتدای جمله بنویسیم . مثلاً :

```
int a; // etelaate porte C dar in moteghayer rikhte mishavad
```

همچنین اگر بخواهیم چند خط پشت سر هم را موقتاً از روند اجرای برنامه حذف کنیم ، باید علامت"/*" را در ابتدا ، و "*/" را در انتهای آن خطوط قرار دهیم . هرگاه این ۲ علامت را پاک کنیم ، دوباره آن قسمت ، به روند اجرای برنامه اضافه می شود . این روش هم برای افزودن توضیحات به برنامه کاربرد دارد .

۵- در ساختار زیر ، هردستور یا دستوراتی که در داخل{} نوشته شود ، بی نهایت بار انجام می شود . در حقیقت while(۱) ، یک حلقه ی بی پایان است که دستورات داخل آن تا وقتی که مدار فعال باشد ، تکرار می شود . در جلسات آیند شما با ساختار حلقه ها بیشتر آشنا خواهید شد .

```
while(۱)

{

PORTD .۳=PINA .۲;

PORTD .۴=PINA .۳;

}
```

این ۲ دستور مکرراً تا زمانی که میکروکنترلر فعال باشد ، اجرا می شوند .

قراره کمی در مورد Codevision توضیحاتی بدیم .

آشنایی باCodevision ، سربرگ های Port و Chip ، آشنایی با Pullup و Output value و. . .

در ابتدا دوستان عزیز برای اینکه بتونن مطلب را با ما دنبال کنند ، لازم است که این نرم افزار را تهیه کرده و روی کامپیوتر شخصی خود نصب کنند . در اینترنت نسخه های Crack شده‌ی این نرم افزار ، برای دانلود وجود دارد .

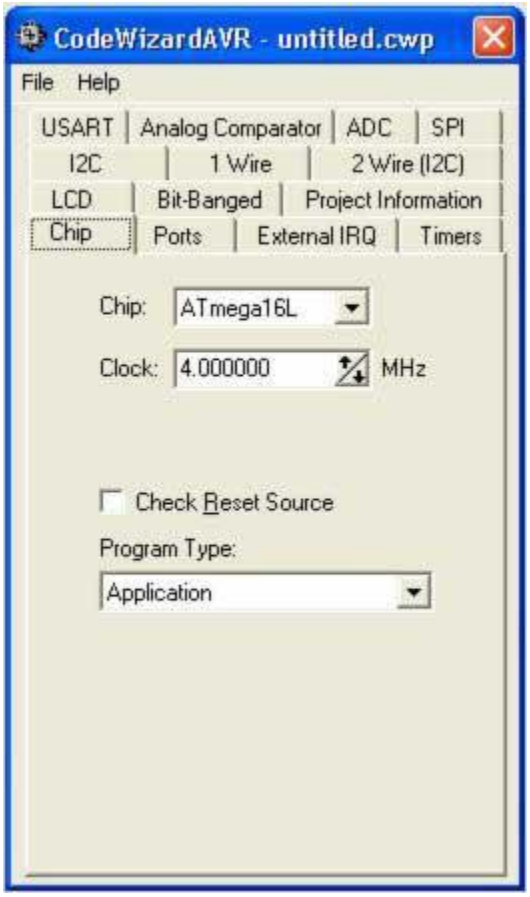
چگونه یک پروژه جدید تعریف کنیم؟

برای نوشتن یک پروژه ی جدید ، باید ابتدا از منوی File ، گزینه ی New را انتخاب کنید . یک پنجره ی کوچک در وسط صفحه باز می شود که در آن باید گزینه ی Project را انتخاب کرده و تایید کنید . بلافاصله پنجره ی دیگری باز می شود که از شما سوال می کند آیا تمایل دارید برای انجام پروژه ی خود از CodeWizard استفاده کنید؟

همانطور که گفته شد ، CodeWizard یکی از نرم افزارهای جانبی CodeVision است که به وسیله ی یک واسط گرافیکی ، در نوشتن برنامه ی اصلی و انجام تنظیمات اولیه پورت ها و . . . . ، کمک بسیار زیادی به ما می کند .

پس گزینه ی Yes را انتخاب می کنیم و CodeWizard باز می شود .

چگونه از CodeWizard استفاده کنیم ؟



شکل زیر ، نمای کلی از CodeWizard است :

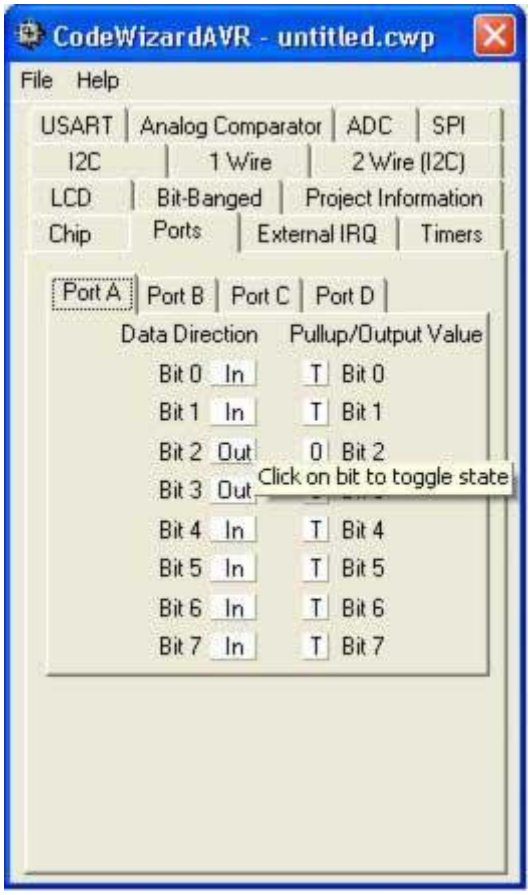
همانطور که می بینید ، لبه های متعددی برای انجام تنظیمات مختلف میکروکنترلر در آن وجود دارد .

سربرگ : Chip

اولین سربرگی که ما با آن کار داریم سربرگ Chip است . در این قسمت ما باید نوع میکروکنترلر خودمان را انتخاب کنیم . همانطور که در شکل بالا می بینید ، میکرو کنترلر ATmega۱۶L را انتخاب کرده ایم .

قسمت Clock مربوط به تنظیم فرکانس کاری آی سی است که ما فعلاً وارد این مبحث نمی‌شویم و آن را با همان مقدار پیش فرض می پذیریم . با قسمت های دیگر این سربرگ هم ما کاری نداریم و آن ها را به همان صورت پیش فرض می پذیریم .

سربرگ: Port



این سربرگ مربوط به تنظیمات ورودی خروجی پایه هاست . همانطور که می بینید هر پایه از هر پورت را در این قسمت می توان به راحتی به صورت In (ورودی) و یا Out (خروجی) تنظیم کرد . فقط کافیست سربرگ مربوط به پورت مورد نظر را انتخاب کنید ، حالا برای تغییر وضعیت هر پایه باید روی آن کلیک کنید .

Output value یا مقدار اولیه

وقتی پایه ای را به صورت خروجی تنظیم می کنیم ، می توان با تنظیم رجیستری PORTx تعیین کرد که سطح ولتاژ خروجی این پایه به صورت پیش فرض ۰ باشد یا ۱ . یعنی در زمانی که هنوز برنامه ما برای پایه ها تعیین وضعیت نکرده است ، می توان به این طریق سطح ولتاژ اولیه ی پایه را تعیین نمود .

CodeWizard در این جا هم کار ما را راحت تر کرده است ، در ستون مقابل یعنی ستون " Pullup/Output value" برای پایه هایی که به صورت **خروجی** تعریف شده اند ، می توان با یک کلیک وضعیت خروجی پایه را مشخص کرد . مثلاً الآن پایه ی شماره ی ۲ از پورت A به صورت خروجی تعریف شده و در ستون مقابل نیز مقدار پیش فرض خروجی ۰ تعیین شده است .

Pullup

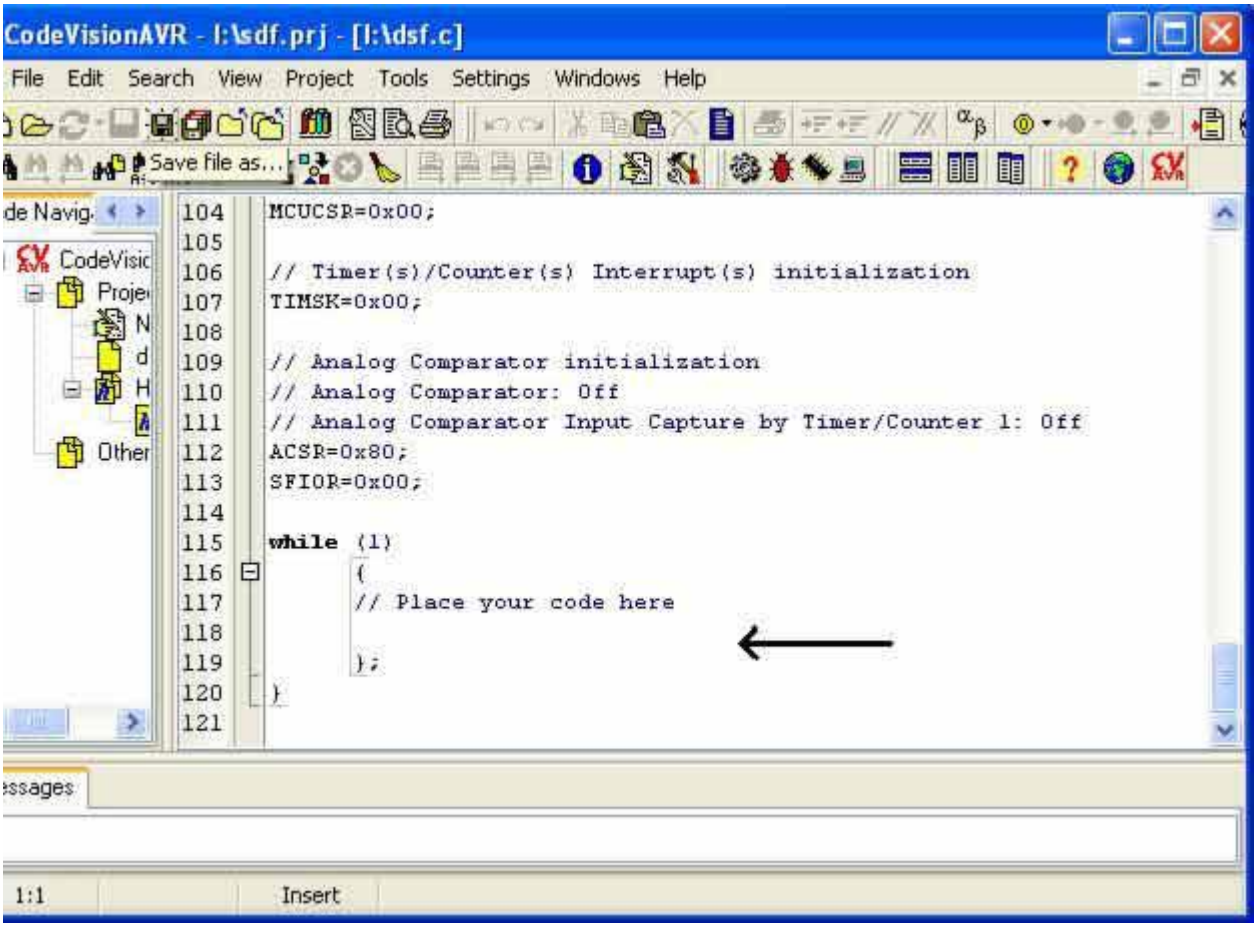
این قابلیت سخت افزاریست و در خانواده ی AVR نیز وجود دارد . Pullup کردن به این معناست که پایه ای را با یک مقاومت بالا (مثلاً ۱۰ کیلو اهم) به + وصل کنیم . اگر هم پایه را با این مقاومت به GND وصل کنیم ، می گوییم پایه را Pulldown کرده ایم . مقاومت بزرگ باعث می شود که جریان عبوری به حداقل کاهش یابد ، ولی به وسیله ولتاژی که بر روی پایه قرار می گیرد ، می‌توان ورودی مورد نظر را در هنگامی که هنوز ورودی از خارج دریافت نکرده به صورت پیش فرض ۰ یا ۱ کرد .

وقتی پایه ای را به صورت **ورودی** تعریف می کنیم ، با تنظیم رجیستری PORTx می توان تعیین کرد که پایه ورودی به صورت پیش فرض Pullup باشد یا نباشد . دقت کنید که در اینجا نمی توان تنظیم کرد که مقدار ورودی پیش فرض ۰ باشد ، چون خانواده ی AVR قابلیت Pulldown ندارند وفقط می توان آنرا به صورت Pullup تنظیم نمود ، و در نتیجه پایه ای که Pullup شده است در هنگامی که هنوز از خارج مقداری را دریافت نکرده است ، به صورت پیش فرض ۱ منطقی شود .

حالا بعد از انجام تنظیمات اولیه پورت ها و خود آی سی ، باید از Codewizard بخواهیم تا یک برنامه نیمه آماده با توجه به تنظیماتی که انجام داده ایم در اختیار ما بگذارد .

برای این کار از منوی File گزینه ی "Generate, Save and Exit" را انتخاب کنید . حالا باید جایی که می خواهید برنامه ی شما Save شود را مشخص کنید . Codevision در اینجا ۳ فایل برای برنامه ی شما می سازد که باید آن ها را نام گذاری کنید . بهتر است نام این ۳ فایل و محل ذخیره سازی آن ها یکی باشد .

بعد از ساخته شدن این ۳ فایل توسط Codevision برنامه آماده است ، حالا شما باید دستورات خود را در محل تعیین شده بنویسید .



بعد از نوشتن برنامه باید آن را کامپایل کرده و و سپس فایل Hex آن را بسازید و بعد از آن ، فایل Hex را در میکرو کنترلر Load کنید . حالا میکروکنترلر شما پروگرام شده و آماده استفاده است .

موضوع : مراحل کامپایل کردن ، پروگرام کردن میکروکنترلر و رفع نقص برنامه

همانطور که گفته شد فقط «زبان ماشین»(Machine Language) ، زبان قابل فهم برای پردازنده ی کامپیوتر است، و برنامه هایی که در زبان های دیگر می نویسیم برای اینکه بتوانند توسط پردازنده اجرا شوند باید حتماً توسط کامپایلرها به «زبان ماشین» ترجمه شوند. اما نوشتن برنامه در این زبان برای ما بسیار مشکل است، زیرا دستورات قابل فهم برای این زبان بسیار ابتدایی و ساده هستند و به سختی می توان برنامه های حرفه ای و الگوریتم های پیچیده را در آن پیاده سازی کرد. مثلاً حتی برای انتقال داده از یک متغیر به متغیر دیگر، باید چندین خط برنامه بنویسید، اما در زبان C این کار در ۱ عبارت انجام می شود. برنامه نویسی در این زبان دشواری های مختلفی دارد که فعلاً به آن ها نمی پردازیم .

به همین خاطر ما برنامه های خود را در زبان C می نویسیم و باقی کارها را به کامپایلر می سپاریم. **کامپایلر** ابتدا برنامه ی ما را از زبان C به زبان اسمبلی ترجمه می کند، سپس برنامه ی دیگری به نام **«اسمبلر»**(" Assembler") برنامه ی ما را از اسمبلی به «زبان ماشین» تبدیل می کند.

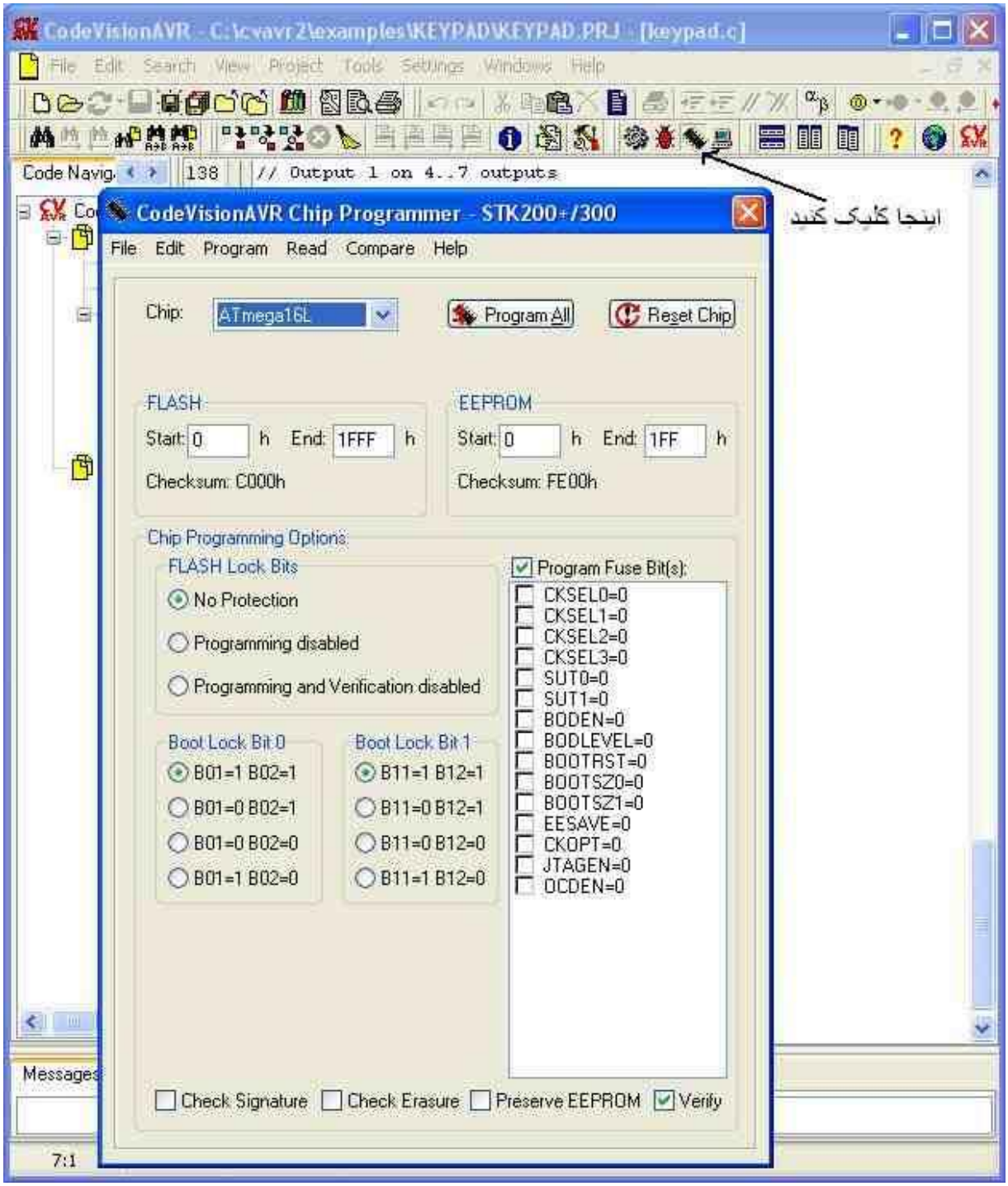
زبان اسمبلی یک پله کاملتر از زبان ماشین است. برنامه نویسی در این زبان بسیار ساده تر از زبان ماشین است و بعضی از مشکلاتی که در زبان ماشین وجود داشت در این زبان برطرف شده و یکی از زبان های رایج فعلی برای برنامه نویسی میکروکنترلرها همین زبان اسمبلی است که بیشتر هم در برنامه نویسی میکروکنترلرهای سری ۸۰۵۱ استفاده می شود. اما برنامه نویسی در این زبان هم بسیار پیچیده تر از زبان C است و نوشتن برنامه های حرفه ای و طولانی در این زبان بسیار دشوار است.

حال چگونه باید این مراحل را در محیط CodeVision انجام داد :

بعد از نوشتن برنامه، شما می توانید با فشار دادن کلید f۹ برنامه ی خود را کامپایل کنید. با فشار دادن همزمان Shift+F برنامه ی شما ابتدا کامپایل شده و به اسمبلی تبدیل می شود وسپس توسط اسمبلر، به زبان ماشین تبدیل می شود. سپس فایلی با پسوند hex. در محلی که شما مشخص کرده اید(در هنگام ساختن پروژه) ساخته می شود.این فایل همان برنامه ی شماست و شما باید این فایل را طی مراحلی که در ادامه توضیح داده می شود، در میکروکنترلر Load کنید.

در اینجا ما نیاز به نرم افزارِ پروگرامر "Programmer" داریم تا اطلاعات ما رو با پرتوکل های مشخصی که در جلسات آینده در مورد آن ها توضیح خواهیم داد، به میکروکنترلر منتقل کند.

همانطور که در جلسات پیش مطرح شد، CodeVision مجموعه ای از چند برنامه مختلف است که در کنار هم جمع شده اند تا همه ی نیازهای کاربر را برطرف کنند. در اینجا هم پروگرامر CodeVision مشکل ما رو حل می کند. برای استفاده از پروگرامر، باید در نوارابزار بالا روی "Chip Programmer" کلیک کنید تا پنجره ای به شکل زیر باز شود.


CodeVisionAVR - C:\cvavr2\examples\KEYPAD\KEYPAD.PRJ - [keypad.c]
File Edit Search View Project Tools Settings Windows Help
Code Navig.
138
// Output 1 on 4..7 outputs
اینجا کلیک کنید
CodeVisionAVR Chip Programmer - STK200+/300
File Edit Program Read Compare Help
Chip: ATmega16L
Program All
Reset Chip
FLASH
Start: 0 h End: 1FFF h
Checksum: C000h
EEPROM
Start: 0 h End: 1FF h
Checksum: FE00h
Chip Programming Options
FLASH Lock Bits
No Protection
Programming disabled
Programming and Verification disabled
Boot Lock Bit 0
B01=1 B02=1
B01=0 B02=1
B01=0 B02=0
B01=1 B02=0
Boot Lock Bit 1
B11=1 B12=1
B11=0 B12=1
B11=0 B12=0
B11=1 B12=0
Program Fuse Bit(s):
CKSEL0=0
CKSEL1=0
CKSEL2=0
CKSEL3=0
SUT0=0
SUT1=0
BODEN=0
BODLEVEL=0
BOOTRST=0
BOOTSZ0=0
BOOTSZ1=0
EESAVE=0
CKOPT=0
JTAGEN=0
OCDEN=0
Check Signature
Check Erasure
Preserve EEPROM
Verify
Messages
7:1

نکته : نرم افزار کدویژن به دو علت عمده زیر کامپایلر مناسبی نمی باشد :

۱. عدم رعایت استاندارد های برنامه نویسی به زبان سی
۲. میزان صحت عملکرد نهایی کد کامپایل شده

ولی به دلیل سادگی و فراوانی منابع آموزشی چه کتاب و چه در اینترنت و ... به ویژه در سطح مبتدی بیشتر آموزش داده می شود .