بسم الله الرحمن الرحیم

دانشکده
فنی و حرفه ای امام حسین (ع)
اصفهان

میکرو کامپیوتر۱

جلسه۳

مدرس : محمدجواد شاهسوندی

معرفی دستورات وکامپایلر

نخستین ویژگی میکرو قابلیت ذخیره سازی و اجرای برنامه است. یک میکرو تمامی خصوصیات یک کامپیوتر را به صورت محدودتر داراست. میکروکنترلرها تراشه هایی هستند که توسط یک نرم افزار به یکی از زبانهای C یا Basicیا اسمبلی برنامه نویسی می شوند و سپس برنامه نوشته شده (که همان اعمال مورد نظر کاربر از میکروکنترلر است)

توسط کامپایلر یا اسمبلر کامپایل شده و اگر کامپایل بدرستی صورت گیرد فایلی را تولید خواهد کرد که این فایل دقیقا همان دستورات، اما به زبان ماشین (یعنی زبان قابل فهم توسط میکروکنترلر) است. در نهایت این فایل تولید شده توسط ابزاری به نام پروگرمر به میکروکنترلر منتقل می شود . با اتصال منبع تغذیه مناسب به میکرو و ابزارهای جانبی و مورد نیاز به پایه های آن، میکروکنترلر شروع به اجرای دستورات خواهد نمود.

مجموعه دستورات و کامپایلر

بدنه یک برنامه نویسی Basicدرمحیط BASCOM شامل تعیین میکرو مورد استفاده ، کریستال و پایان و گزینه های اختیاری دیگر است که در زیر معرفی می کنیم.

معرفی میکرو: REGFILE= VAR$

برای شروع یک برنامه در محیط BASCOMابتدا بایستی میکرو مورد نظر تعریف گردد. VARنام چیپ مورد استفاده است که می تواند یکی از موارد زیر باشد.

REGFILE= “AT12DEF.DAT” ‘ATTINY 12 MCU$

REGFILE= “AT15DEF.DAT” ‘ATTINY15 MCU$

REGFILE= “2323DEF.DAT” ‘AT90S2323 MCU$

REGFILE= “M16DEF.DAT” ‘MEGA16 MCU$

مجموعه دستورات و کامپایلر

کریستال:

برای مشخص کردن کریستال استفاده شده بر حسب هرتز از دستور زیر استفاده می نماییم.

$CRYSTAL = X

Xفرکانس کریستال استفاده شده بر حسب هرتز است .

اسمبلی و بیسیک:

درصورت نیاز برای نوشتن برنامه اسمبلی در بین برنامه نویسی **Basic** از دستور زیر استفاده می نماییم .

ASM$

ASSEMBLY PROGRAMME

ENDASM$

مجموعه دستورات و کامپایلر

یادداشت (اختیاری):

گاهی نیاز است یادداشتهایی برای اطلاعات بیشتر در برنامه اضافه کنید .

REM‘ یا

در برنامه نویسی Basicیادداشتها و نوشته های بعد از این دستور غیر فعال بوده و در برنامه برای یادداشت به کار می رود و کامپایل نخواهند شد و همچنین به رنگ سبز در می آیند.

مجموعه دستورات و کامپایلر

آدرس شروع برنامه ریزی حافظه FLASH :

گاهی نیاز است که برنامه خود را از آدرسی دلخواه در حافظه ی FLASH ROM قرار دهید.

ROMSTART=ADDRESS$

در برنامه نویسی Basicآدرس مکانی از حافظه است که برنامه HEX از این آدرس در حافظه میکرو کنترولر، شروع به نوشته شدن می شود در صورتی که از این دستور استفاده نشود COMPILER به صورت خودکار آدرس H0000 & را در نظر می گیرد.

مجموعه دستورات و کامپایلر

تعیین کلاک :

با این دستور در میکرو های سری میکرو MEGAAVRاز جمله NEGA103تا MEGA603به صورت نرم افزاری میتوان کلاک سیستم را تغییر داد. تقسیم کلاک به طور مثال برای کاهش مصرف تغذیه استفاده می شود.

CLOCKDIVISON=VAR

VARمقادیر معتبر بین اعداد ۲ تا ۱۲۸ می تواند باشد.

پایان برنامه:

END

در برنامه نویسی Basicاین دستور در انتهای برنامه قرار می گیرد و اجرای برنامه را متوقف می کند. با دستور ENDتمام وقفه ها غیر فعال شده و یک حلقه بی نهایت تولید و برنامه خاتمه می یابد.

مجموعه دستورات و کامپایلر

دستور CONST :

برای تعریف یک ثابت از این دستور استفاده می شود.

```
CONST SYMBOL=NUMCONST

CONST SYMBOL=STRINGCONST

CONST SYMBOL=EXPRESSION
```

دستور ALIAS :

از این دستور برای تغییر نام متغیر استفاده می شود.

```
DIRECTION ALIAS PORTB.1
```

حال شما می توانید به جای PORTB.1 از متغیر DIRECTION استفاده نمایید.

مجموعه دستورات و کامپایلر

دستور CHR:

از این دستور برای تبدیل متغیر عددی یا یک ثابت به کارکتر استفاده می شود. زمانی که شما قصد دارید یک کارکتر بر روی LCDنمایش دهید از این دستور می توانید استفاده نمایید .

در صورتی که از این دستور بدین صورت استفاده نمایید PRINT CHR (VAR)کاراکتر اسکی VARبه پورت سزیال فرستاده خواهد شد.

دستور INSTR :

این دستور محل و موقعیت یک زیر رشته را در رشته دیگر مشخص می کند .

VAR=INSTR (START,STRING,SUBSTR)

VAR=INSTR(STRING,SUBSTR)

VARعددی است که مشخص کننده محل SUBSTRدر رشته اصلی STRINGمی باشد و زمانی که زیر رشته مشخص شده در رشته اصلی نباشد صفر برگردانده می شود.

مجموعه دستورات و کامپایلر

دستور INCR :

INCR VAR

این دستور یک واحد به متغیر عددی می افزلید.

دستور DECR :

DECR VAR

این دستور متغیر VARرا یک واحد کم می کند.

مجموعه دستورات و کامپایلر

دستور CHECKSUM :

این دستور مجموع کد دسیمال اسکی رشته VARرا برمی گرداند که البته اگر مجموع کد اسکی رشته از عدد۲۵۵ بیشتر شود مقدار ۲۵۶ از مجموع کم می شود.

دستور LOW :

این دستور LSBیک متغییر را برمی گرداند.

VAR = LOW (S)

LSBمتغییر Sدر VARقرار می گیرد.

مجموعه دستورات و کامپایلر

دستور HIGH :

این دستور پر ارزش ترین بایت (MSB) یک متغییر را برمی گرداند.

VAR = HIGH (S)

مقدار MSB متغییر S در VAR جای می گیرد.

دستور LCASE :

این دستور تمام حروف رشته مورد نظر را تبدیل به حروف کوچک می کند.

TARGET = LCASE (SOURCE)

تمام حروف رشته SOURCE کوچک شده و در رشته TARGET جای داده می شوند.

مجموعه دستورات و کامپایلر

دستور UCASE :

این دستور تمام حروف رشته مورد نظر را تبدیل به حروف بزرگ می کند.

TARGET = UCASE (SOURCE)

تمام حروف رشته SOURCEبزرگ شده و در رشته TARGETجای داده می شود.

دستور RIGHT :

با این دستور قسمتی از یک رشته را جدا می کنیم.

VAR = RIGHT (VAR1,n)

از سمت راست رشته VAR1به تعداد کاراکتر nرشته ای جدا شده و در رشته VARقرار می گیرد.

مجموعه دستورات و کامپایلر

دستور LEFT :

این دستور کاراکترهای سمت چپ یک رشته را به تعداد تعیین شده جدا می کند.

VAR = LEFT (VAR1,n)

رشته VAR1 از سمت چپ به تعداد کاراکتر n، رشته ای جدا شده و در رشته VAR قرار می گیرد.

دستور LEN :

این دستور طول یا به عبارتی تعداد کاراکترهای یک رشته را بر می گرداند.

VAR = LEN (STRING)

دستور LTRIM :

این دستور فضای خالی رشته را حذف می کند.

VAR = LTRIM (ORG)

دستور SWAP :

SWAP VAR1 , VAR2

با اجرای این دستور محتوای متغییر VAR1در متغییر VAR2و محتوای متغییر VAR2در متغییر VAR1 قرار می گیرد.

مجموعه دستورات و کامپایلر

دستور MID :

با این دستور می توان قسمتی از یک رشته را برداشت. و یا قسمتی از یک رشته را با قسمتی از یک رشته دیگر عوض کرد.

۱-VAR = MID (VAR1 , ST [,L])

۲-MID(VAR , ST [,L]) = VAR1

۱- قسمتی از رشته VAR1، با شروع از کاراکتر ST ام و طول L برداشته شده و در متغییر VAR قرار می گیرد.

۲- رشته VAR1در رشته VARبا شروع از کاراکتر STام و وطول Lقرار می گیرد.

در صورت قید نکردن گزینه اختیاری L، بیشترین طول در نظرگرفته می شود.

مجموعه دستورات و کامپایلر

دستور ROTATE :

دستور بیتها را به چپ یا راست منتقل می کند ولی تمام بیتها محفوظ هستند و هیچ بیتی بیرون فرستاده نمی شود.

ROTATE VAR, LEFR/RIGHT [,SHIFTS]

دستور SPACE :

برای ایجاد فضای خالی از این دستور استفاده می شود.

VAR=SPACE(X)

تابع FORMAT:

این دستور یک رشته عددی را شکل دهی می کند .

TARGET= FORMAT(SOURCE,"MASK")

رشته ای است که شکل دهی شود. SOURCE قرار می گیرد. TARGETنتایج در نوع شکل دهی است.MASK

مجموعه دستورات و کامپایلر

تابع FUSING :

از این دستور برای روند کردن رشته های عددی استفاده می شود.

TARGET = FUSING (SOURCE , “MASK”)

SOURCEرشته مورد نظر برای شکل دهی و MASKنوع شکل دهی است سپس نتایج این شکل دهی در TARGETقرار می گیرد. عمل MASKحتما باید با علامت # شروع شود و حد اقل باید یکی از علامت های # و & را بعد از ممیز داشته باشد. علامت های # و& با یکدیگر بعد از ممیز استفاده نمی شود. با استفاده از علامت # عدد روند می شود و در صورت استفاده از & روندی صورت نمی گیرد.

جدول LOOKUP :

توسط این جدول می توان مقدار دلخواهی را از جدول برگرداند.

VAE = LOOKUP (VALUE , LABEL)

LABELE برچسب جدول و VALUE اندیس داده دلخواه در جدول است. داده برگشتی از جدول در متغیر VAR قرار می گیرد. VALUE = 0 اولین داده در جدول را برمی گرداند. تعداد اندیس ها و مقدار داده برگشتی به ترتیب نهایتا می تواند ۲۵۵ و ۶۵۵۳۵ باشند.

جدول LOOKUPSTR :

توسط این جدول می توان رشته دلخواهی را از جدولی برگرداند.

VAR = LOOKUPSTR (VALUE , LABEL)

LABLEبرچسب جدول و VALUEاندیس رشته دلخواه در جدول است. رشته برگشتی از جدول در متغییر رشته ای VARقرار می گیرد. VALUE = 0اولین رشته در جدول را برمی گرداند.

تعداد اندیس ها نهایتا می تواند ۲۵۵ باشد.

توابع ریاضی و محاسباتی در برنامه نویسی Basic

از عملگرهای ریاضی زیر می توانید در محیط BASCOMاستفاده نمایید و عملیات ریاضی خود را راحتر انجام دهید.

علامت ضرب ASTERISKS	*
علامت جمع PLUS SIGN	+
علامت تفریق MINUS SIGN	–
علامت ممیز PERIOD	.
علامت تقسیم SLASH	/
علامت کوچکتر از LESS THAN	<
علامت تساوی EQUAL SIGN	=
علامت بزرگتر از GREATER THAN	>
علامت بتوان EXPONENT	^
علامت کوچکتر یا مساوی با LESS THAN OR EQUAL TO	<=
علامت بزرگتر یا مساوی با GREATER THAN OR EQUAL TO	=>
علامت مخالف INEQUALITY	<>

عملگر های منطقی در برنامه نویسی Basic

عملگر های منطقی در BASCOMبه قرار زیر است:

CONJUNCTION AND

DISJUNCTION OR

EXCLUSIVE OR XOR

LOGICAL COMPLEMENT NOT

تابع ABS :

VAR = ABS (VAR2)

این دستور به معنای ریاضی VAR = |VAR2| قدر مطلق است.

توابع ریاضی و محاسباتی در برنامه نویسی Basic

تابع EXP :

TARGET = EXP(SOURCE)

TARGET برابر با eبه توان SOURCE است. TARGET مغییری از نوع داده SINGLE است.

تابع LOG10 :

لگاریتم پایه ۱۰ متغییر یا ثابت SOURCE در متغییر TARGET قرار می گیرد. TARGETو SOURCE هر دو داده نوع SINGLE هستند.

تابع LOG :

این دستور لگاریتم طبیعی یک داده از نوع SINGLEرا برمی گرداند.

TARGET = LOG (SOURCE)

لگاریتم متغییر یا ثابت SOURCE از نوع داده SINGLE گرفته می شود و در متغییر TARGET از نوع داده SINGLE قرار می گیرد. این تابع برای اجرا شدن وقت زیادی می برد مخصوصا زمانی که اعداد بزرگ استفاده می شود. همچنین با بزرگتر شدن اعداد دقت نیز پایین خواهد آمد.

رجیستر ها و آدرس های حافظه در برنامه نویسی Basic

تمام میکروهای AVR دارای ۳۲ رجیستر ۸ بیتی R0-R31 همه منظوره در CPU خود هستند. رجیسترهای R31(MSB)با R30(LSB)، R29(MSB)و R27(MSB)با R26(LSB)تشکیل سه رجیستر ۱۶ بیتی به ترتیب با نام های Z، Yو Xرا می دهند.

دستور SET :

توسط این دستور می توان یک بیت را یک کرد.

SET BIT/PIN

SET VAR.X

BIT می تواند یک بیت و یا یک SFR مانند PORB.1 باشد و VAR متغییری از نوع داده BYTE، INTEGER، WORDیا LONG است. Xبرای BYTE می تواند ۰ تا ۷، ۰ تا ۱۵ برای WORD و برای LONG می تواند ۰ تا ۳۱ باشد.

دستور TOGGLE :

این دستور مقدار منطقی یک پایه یا یک بیت را معکوس می کند.

TOGGLE PIN/BIT

PINمی تواند یک SFRمانند PORTB.1و یا یک بیت باشد.

رجیستر ها و آدرس های حافظه در برنامه نویسی Basic

دستور RESET :

توسط این دستور می توان یک بیت را صفر کرد.

RESET BIT/PIN

RESET VAR.X

BIT می تواند یک بیت و یا یک **SFR** مانند **PORTB.1** باشد و **VAR** مغییری از نوع داده **WORD,** **INTEGER, BYTE**یا **LONG** است و **X**برای **BYTE**می تواند ۰ تا ۷، ۰ تا ۱۵ برای **WORD**و برای **LONG** می تواند ۰ تا ۳۱ باشد.

دستور BITWAIT :

BITWAIT X, SET/RESET

توسط این دستور اجرای برنامه تا زمانی که بیت **X,** (1=) **SET**یا **RESET**شود در خط جاری متوقف می ماند. در صورت **TRUE**شدن شرایط، اجرای برنامه از خط بعد ادامه می یابد. **X** می تواند یک بیت رجیستر داخلی مانند **PORTB.Y** باشد که **Y**می تواند بین می تواند بین اعداد ۰ تا ۷ تغییر کند.

رجیستر ها و آدرس های حافظه در برنامه نویسی Basic

دستور CPEEK :

از این دستور برای برگرداندن بایتی که در آدرسی از حافظه کدی ذخیره شده است استفاده می کنیم. با این دستور می توانید به رجیسترهای داخلی نیز دسترسی پیدا کنید. لازم به تذکر است که با این دستور شما نمی توانید در حافظه داخلی بنویسید.

دستور OUT :

توسط این دستور می توان یک بایت به یک پورت سخت افزاری یا آدرس حافظه داخلی /خارجی ارسال کرد.

OUT ADDRESS , VALUE

VALUEبه آدرس ADDRESSکه می تواند بین ۰ Hتا FFFFHباشد فرستاده می شود. دستور OUT می تواند در تمام مکانهای حافظه AVRبنویسید.در ضمن برای نوشتن در مکان حافظه خارجی (XRAMباید در محیط BASCOMو در منوی (OPTION?COMPILER ? CHIP، گزینه (EXTERNAL ACCESS ENABLEرا فعال کنید.

دستور العمل های حلقه و پرش در برنامه نویسی Basic

دستور GOTOو JMP

GOTO LABEL

JMP LABLE

با این دستورات می توان به برچسب LABEL پرش کرد. برچسب LABEL باید با علامت: COLLONپایان یابد و می تواند تا۳۲ کاراکتر طول داشته باشد. به خاطر داشته باشید زمانی که از دو LABELهم نام استفاده شود کامپایلر به شما هشدار WARNING می دهد. دستور RETURNبرای برگشت از برچسب وجود ندارد.

دستور العمل DO-LOOPدر برنامه نویسی Basic

فرم کلی دستور DO...LOOPبه صورت زیر می باشد.

DO

Statements

LOOP [UNTIL EXPRESSION]

دستورالعمل Statements تا زمانی که EXPRESSION دارای ارزش TRUEیا غیر صفر است تکرار خواهد شد بنابراین این نوع حلقه، حداقل یکبار تکرار شود. DO-LOOPبه تنهایی یک حلقه بینهایت ایت که با EXIT DO می توان از درون حلقه خارج شد و اجرای برنامه در خط بعد از حلقه ادامه یابد.

دستور العمل های حلقه و پرش در برنامه نویسی Basic

دستورالعمل IFدر برنامه نویسی Basic

در کلیه حالتهای زیر عبارت Statements می تواند یک دستورالعمل ساده یا چند دستورالعمل مرکب باشد.

حالت ۰:

IF EXPRESSION THEN Statements

دستورالعمل Statementsزمانی اجرا می شود که عبارت EXPRESSIONدارای ارزش TRUEباشد.

حالت ۱:

IF EXPRESSION THEN

Statements1

ELSE

Statements2

END IF

دستور العمل های حلقه و پرش در برنامه نویسی Basic

در صورتی که عبارت EXPRESSION1 دارای ارزش TRUE باشد دستورالعمل Statements1 اجرا خواهد شد، در غیر این صورت دستورالعمل Statements2 اجرا می شود.

IF EXPRESSION1 THEN

Statements1

ELSE IF [EXPRESSION2 THEN]

Statements2

ELSE

Statements3

ELSE IF

دستور العمل های حلقه و پرش در برنامه نویسی Basic

در صورتی که عبارت EXPRESSION1 دارای ارزش TRUE باشد دستورالعمل Statements1 اجرا خواهد شد. در صورتی که عبارت EXPRESSION1 دارای ارزش FALSE ولی عبارت اختیاری EXPRESSION2 دارای ارزش TRUE باشد، دستورالعمل Statements2 اجرا خواهد شد و در غیر این صورت یعنی در حالتی که هر دو عبارت EXPRESSION2و EXPRESSION1 دارای ارزش FALSEباشند دستورالعمل Statements3 اجرا خواهد شد.

همچنین با دستور IFمی توان یک یا صفر بودن یک بیت از یک متغییر را امتحان کرد.

IF BIT = 1 THEN OR IF BIT = 0 THEN