Lecture 7

Programming the C8051F020

Using C Language

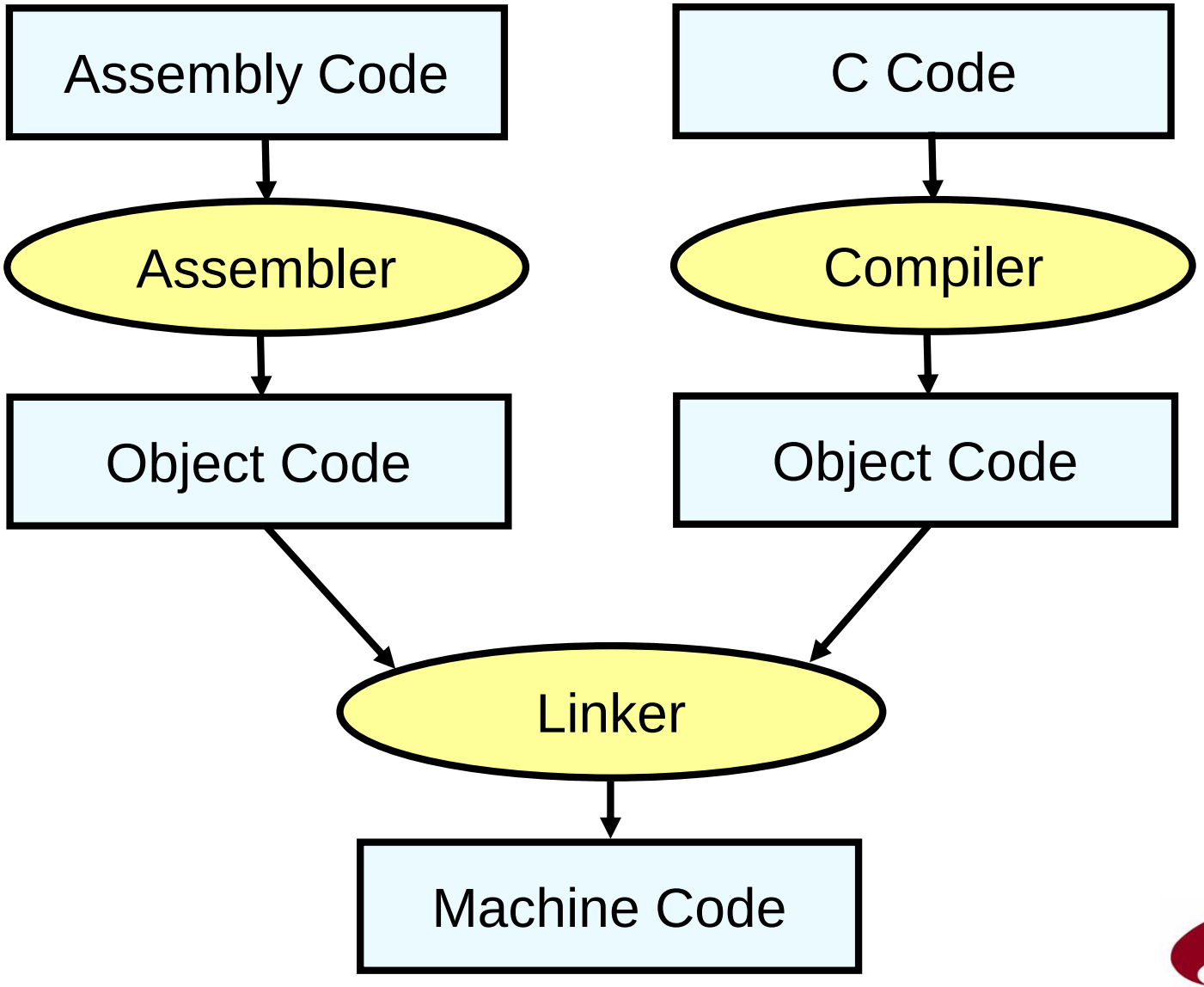
www.elecstudy.org markalex012@gmail.com טסליצקי מרק

תכנות מיקרו בקר C8051F020 בעזרת שפת C.

- בניית קוד
- מבנה של תוכנה בסיסית בשפת C.
- הגדרת אוגרים
- הגדרת אוגרים מיוחדים באורך 16 סיביות.
- סוגי משתנים
- זיכרון פנימי
- זיכרון מסוג סיבית
- זיכרון חיצוני
- אופרטורים: יחס, לוגיים, סיביות.

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

בניית קוד הרצה



www.elecstudy.org markalex012@gmail.com טסליצקי מרק

דוגמא לתוכנה בסיסית בשפת C.

```
//-----------------------------------------------------------------
// Basic blank C program that does nothing
// other than disable the watch dog timer
//-----------------------------------------------------------------

//-----------------------------------------------------------------
// Includes
//-----------------------------------------------------------------

#include <C8051F020_defs.h> // Include SFR declarations

void main (void)
{
   EA = 0;              // Disable global interrupts

   WDTCN = 0xde;        // Disable watchdog timer
   WDTCN = 0xad;

   EA = 1;              // Enable global interrupts

   while(1);            // Stops the program from terminating and restarting
}
```


www.elecstudy.org markalex012@gmail.com טסליצקי מרק

הגדרת אוגרים

- הגדרת אוגרים עושים בעזרת קבצים מיוחדים (include files).
- קובץ **C8051F020_defs.h** מכיל הגדרות של כל האוגרים מיוחדים שיש במערכת (SFR).
- דוגמא:

```
sfr    P0=0x80;      // Port 0
sfr    SBUF0=0x99;   // Serial Port 0 Buffer
sfr    IE=0xA8;      // Interrupt Enable
sfr    WDTCN=0xFF;   // Watchdog Timer Control
sbit   EA=IE^7;      // Global Interrupt enable
```


www.elecstudy.org markalex012@gmail.com טסליצקי מרק

אוגרים מיוחדים באורך 16 סיביות

- משתמשים בשני אוגרים שכנים שנמצאים אחד ליד שני כדי להגדיר אוגר 16 סיביות.
- קובץ **C8051F020_defs.h** מכיל גם אוגרי 16 סיביות כמו שהוא מכיל אוגרים 8 סיביות.
- אוגרי 16 סיביות **לא ניתנות לגישה מבחינת סיביות**, אפילו שכתובות שלהם מסתיימות ב-0H או ב-8H .

הגדרת משתנים בשפת C 51.

ANSI C

8051 Compiler Specific

Data Type	Bits	Bytes	Value Range
signed char	8	1	-128 to +127
unsigned char	8	1	0 to 255
enum	8/16	1 or 2	-128 to +127 or -32768 to +32767
signed short	16	2	-32768 to +32767
unsigned short	16	2	0 to 65535
signed int	16	2	-32768 to +32767
unsigned int	16	2	0 to 65535
signed long	32	4	-2147483648 to 2147483647
unsigned long	32	4	0 to 4294967295
float	32	4	±1.175494E-38 to ±3.402823E+38
bit	1	-	0 to 1
sbit	1	-	0 to 1
sfr	8	1	0 to 255
sfr16	16	2	0 to 65535

Some compilers use 4 bytes for these

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

זיכרון פנימי

- **תיאור כללי**
  - ניתן להשתמש ב-256 בית של זיכרון פנימי.
  - 128 בית ראשונים (תחתונים) ניתנים לגישה ישירה וגם לגישה בצורה אקיפה.
  - 128 בית עליונים (0x80 עד 0xFF) ניתנים לגישה רק בצורה אקיפה.
  - יש גם אזור של 16 בית שמתחיל בכתובת 20h שניתן לגישה בצורת סיביות בודדות.

- גישה לזיכרון פנימי מהירה מאוד עקב גישה דרך כתובת 8 סיביות.
  - גודל של זיכרון פנימי מוגבל ומכיל עד 256 בית. ($2^8 = 256$)

- בשפת C ניתן לשים משתנה בכל אזור רצוי. במידה ומשתמש לא עושה הגדרת מיקום, אז קומפיילר עושה עבודה הזאת במקומו בהתאם למודל זיכרון הנבחר מראש.
  - **דוגמא:** `int ADC_Result;`
    - SMALL memory model: this variable is placed in DATA space
    - COMPACT memory model: this variable is placed in IDATA space
    - LARGE memory model: this variable is placed in XDATA space

Internal Data Memory זיכרון פנימי

- ניתן לחלק אזור זיכרון פנימי ל-3 תת אזורים: **idata**, **data**, **bdata**.
- אזור זיכרון **data** הוא אזור של 128 בית תחתונים. משתנים שנמצאים באזור זה ניתנים לגישה בשיטה ישירה (direct addressing) ובברירת מחדל משתמשים במודל קטן של זיכרון SMALL.
- אזור **idata** מתייחס לכל 256 בית של זיכרון פנימי.
  - גישה לאזור הזה ניתן אך ורק בעזרת גישה עוקפת וזה מוריד את המהירות הגישה במספר רמות לאומת גישה ישירה.
- אזור **bdata** זה אזור באורך 16 בית (20h - 2Fh). באזור הזה ניתן להגיע לנתונים לפי סיביות ולא רק לפי בית שלם.

- **דוגמאות:**

```
unsigned char data name;
int idata count;
int bdata status;
```

גישה באזור סיביות בודדות.

- אזור **bdata** זה אזור באורך 16 בית (20h - 2Fh). באזור הזה ניתן להגיע לנתונים לפי סיביות ולא רק לפי בית שלם.

- כדי להגדיר אותם משתמשים במצביעים **sbit ,bdata, bit.**

- **דוגמא:**

```
int bdata X;              // 16-bit bit-addressable variable X
bit flag;                 // bit-valued variable flag
```

- משתנה X הוא 16 סיביות והוא ניתן לגישה של כל סיבית וסיבית שלו בנפרד.
- גודל משתנה ***flag*** *הוא סיבית בודדת והוא יכול להכיל או ערך 0 או ערך 1.*

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

גישה באזור סיביות בודדות.

- הגדרה **sbit** מיועדת לשיוך סיביות למשתנים או לאוגרים שניתנים לגישה של סיביות בודדות:

- **Example:**

```
int bdata X;            // 16-bit bit-addressable variable X
sbit X7_Flag = X^7;     // bit 7 of X (bit variable)
sbit Red_LED = P0^1;    // bit 1 of Port P0 (bit-addressable SFR)
```

- משתנה מסוג **sbit חייב להיות מוגדר אך ורק בצורה גלובלית ולא מקומית.**
- *בדוגמא ראשונה משתנה **X7_Flag** זה סיבית מספר 7 בתוך משתנה X בגודל 16 סיביות.*
- *בדוגמא השנייה משתנה **Red_LED** שייך לפורט P0 ששייך לקבוצת SFR.*

גישה באזור סיביות בודדות.

- עוד דוגמא:

```
int bdata status;
bit s2 = status^5;
```

- אי אפשר להגדיר מצביע מסוג bit pointer או מערך סיביות.
- אזור סיביות מוגדר ל-16 בית או 128 סיביות בסה"כ.

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

אזור זיכרון חיצוני

- גודל של זיכרון חיצוני הוא עד 64 kB, ונמצא מחוץ יחידת עיבוד מרכזי (CPU).
- גישה לאזור חיצוני (אזור נמצא באזור XDATA) לוקחת זמן רב בהשוואה לגישה לאזור פנימי של זיכרון.
  - זה קורה עקב כך שמצביע לכתובת הוא אוגר 16 סיביות DPTR ומבחינת CPU צריכים לעשות מספר פעולות אם אוגר זה לפני שניגשים לזיכרון עצמו.

- מבחינת שפת **Cx51** ישנם 2 סוגים שונים של זיכרון חיצוני: **xdata ו- pdata.**
  - **xdata** - זה אזור זיכרון כולו (כל 64 kB של זיכרון חיצוני). ניתן להגיע עליו, ברגע שמגדירים מודל LARGE .
  - **pdata** - זה אזור זיכרון מצומצם של עמוד בודד (256 בית) וזה מתאים למודל מסוג COMPACT.
    - **pdata** מוגדרת ע"י אוגרים R0 ו- R1 (@R1 , @R0) במקום DPTR ובגלל זה זמן הגישה לאזור pdata הוא קצר יותר מ- **xdata** (@DPTR) . זה גם מגביל את אזור הזיכרון **pdata ל- 256 בית (אוגרים R1 ו- R0 הם 8 בית).**

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

פעולות מתמטיות

- כל הפעולות מתמטיות פרט ל-"-" אונרי עובדות עם שני אופרנדים בלבד. (פעולה – או ++ ניתן לתרגם בתור -1 או +1).

Operator	Description
+	Add
–	Subtract
*	Multiply
/	Divide
%	Modulo (remainder of division)
–	Negation (unary minus)

- פעולת "-" אונרי מחזירה ערך שלילי של מספר בשיטת משלים ל-2.
  - דוגמא:

```
unsigned int count = 0x0F;
// TMR2RL gets 0xFFFF-0x0F+1 = 0xFFF1
TMR2RL = -count;
```

אופרטורים של יחס (השוואה)

- אופרטורים בקבוצה הזאת מחזירים ערך אמת או שקר (True or False). 0 או לא 0.

- **משתמשים בהם בפקודות: for ,if ,while**

Operator	Description
==	Equal to
!=	Not Equal to
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

פעולות לוגיות

- במידה ויש מספר תנאים ואנו צריכים לאחד אותם לתנאי אחד ניתן להשתמש בפונקציות לוגיות.

Operator	Description
&&	Logical AND
||	Logical OR
!	Logical NOT

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

פעולות עם סיביות

- בשפת C ישנם מספר פעולות עם סיביות:

Operator	Description
&	Bitwise AND
|	Bitwise OR
~	Bitwise NOT (1's Compliment)
^	Bitwise Exclusive OR
<<	Shift Left
>>	Shift Right

- דוגמא:

```
Result = Value1 & Value2;
```

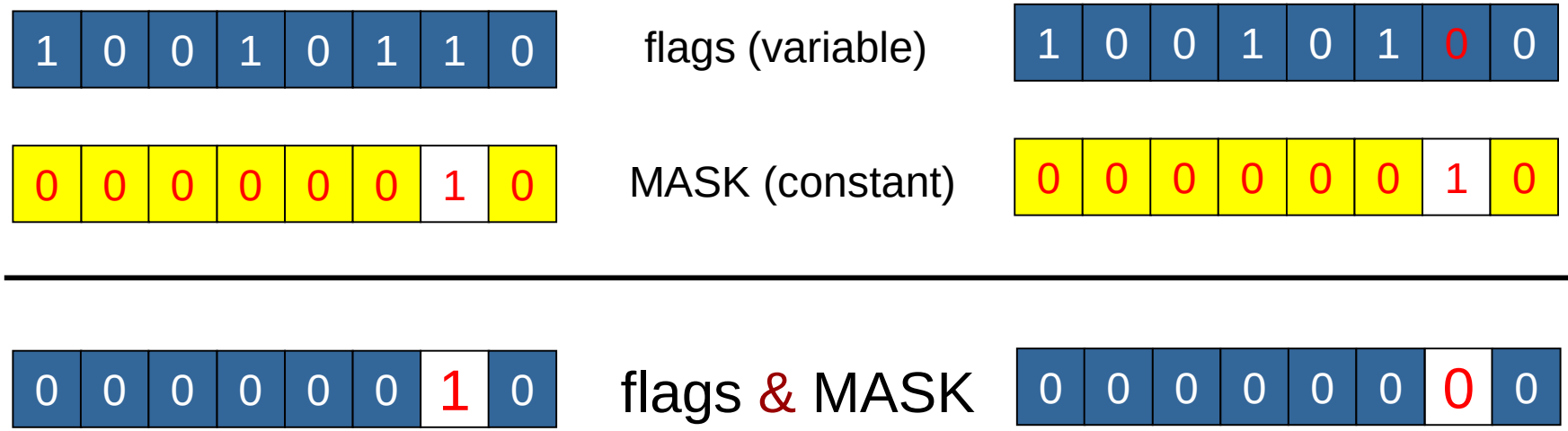
- רק ערך של **`Result`** **משתנה**
- אם **`Value1`** = 00100100b וערך של **`Value2`** = 10100000b, אז תשובה של פעולת
  00100100b & 10100000b = 00100000b

שימוש בפעולות עם סיביות

- הדלקת סיביות.
  - ניתן להדליק סיבית רצויה ע"י פעולת OR עם "1" לוגי.
- כיבוי סיביות.
  - ניתן לכבות סיבית רצויה ע"י פעולת AND עם "0" לוגי.
- הפיכת סיביות.
  - ניתן להפוך סיבית רצויה ע"י פעולת XOR עם "1" לוגי.

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

בדיקת מצב הסיבית




```
if ( (flags & MASK) == 0 )
   printf("flags.1 is OFF");
else
   printf("flags.1 is ON");
```

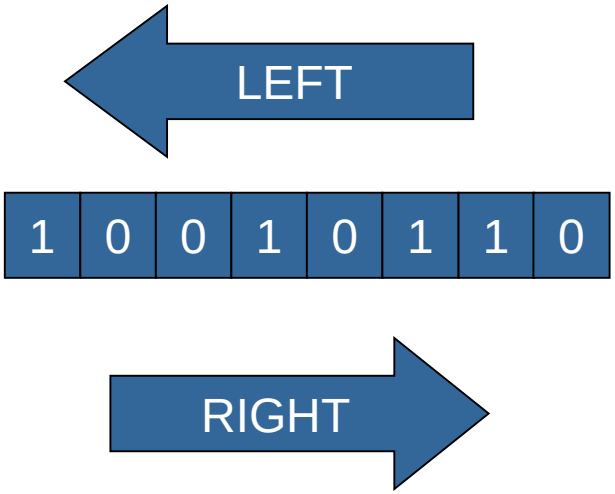
```
flags.1 is ON
```

```
flags.1 is OFF
```


www.elecstudy.org markalex012@gmail.com טסליצקי מרק

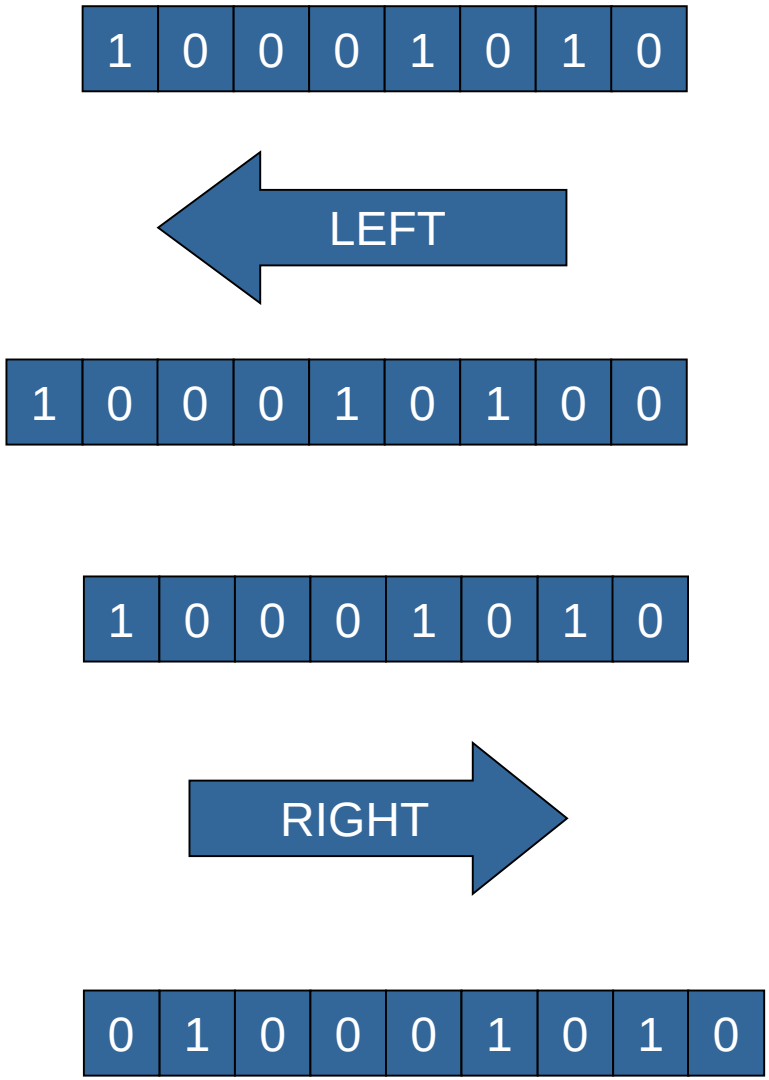
הזזת סיביות

- פעולת הזזה מזיזה סיביות לפי סוג פקודה (שמאלה או ימינה) ולמקום פנוי מכניסה 0.



www.elecstudy.org markalex012@gmail.com טסליצקי מרק

Left (<<) and Right Shift (>>)



משפטים מתמטיים מקוצרים

Operator	Description	Example	Equivalent
+=	Add to variable	`x += 2`	`x=x+2`
-=	Subtract from variable	`x -= 1`	`x=x-1`
/=	Divide variable	`x /= 2`	`x=x/2`
*=	Multiply variable	`x *= 4`	`x=x*4`

www.elecstudy.org markalex012@gmail.com טסליצקי מרק

משפטים לוגיים מקוצרים

Operator	Description	Example	Equivalent
&=	Bitwise AND with variable	`X &= 0x00FF`	`X = X & 0x00FF`
|=	Bitwise OR with variable	`X |= 0x0080`	`X = X | 0x0080`
^=	Bitwise XOR with variable	`X ^= 0x07A0`	`X = X ^ 0x07A0`

```
//-- Enable P1.6 as push-pull output
P1MDOUT |= 0x40;

//-- wait till XTLVLD pin is set
while ( !(OSCXCN & 0x80) );
```

www.silabs.com/MCU