

İLERİ MIKRODENETLEYİCİLER

Ege Üniversitesi Ege MYO
Mekatronik Programı

EK-A

***IDE, Program
Geliřtirme Araçları***

Geliştirme Araçları – Keil C51 Yazılımı

The screenshot displays the Keil C51 development environment with the following components and labels:

- Başlık Çubuğu (Title Bar):** glcd - µVision2 - [C:\Grafik_LCD_Öğreniyorum\glcd_ks108\glcd.c]
- Menü Çubuğu (Menu Bar):** File Edit View Project Debug Peripherals Tools SVCS Window Help
- Araç Çubukları (Toolbars):** A set of icons for file operations, editing, and debugging.
- Çalışma Alanı (Workspace):** The main area containing the source code for 'glcd.c'. The code includes headers, includes, and defines for hardware registers.

```
/* K80108 G_LCD.h */
#include <89c51rd2.H>
#include "resimler.h"

/* Uç Tanımlamaları */
#define DI P3_2 // Register Select ucu
#define RW P3_1 // Oku/Yaz ucu
```
- Proje Penceresi (Project Window):** A table showing the status of registers.

Register	Value
Regs	
r0	0x00
r1	0x00
r2	0x00
r3	0x00
r4	0x00
r5	0x00
r6	0x00
r7	0x00
- Çıktı Penceresi (Output Window):** Shows the command prompt with the command 'Load "C:\\Gra...' and the output 'ASM ASSIGN'.
- Bellek Penceresi (Memory Window):** Displays memory addresses and their contents.

Address	Value
D:0x0C:	00 00 00 00 00 00
D:0x12:	00 00 00 00 00 00
D:0x18:	00 00 00 00 00 00
D:0x1E:	00 00 00 00 00 00
- Watch/Call Stack:** A window for monitoring variables and the call stack.

Geliştirme Araçları – ISIS Programı

ISIS/Proteus programı:

- Hem sayısal hem de analog devreleri birlikte simüle edebilen, mikrodenetleyici tabanlı tasarımları destekleyen çok amaçlı simülasyon programıdır.
- Herhangi bir devreyi fiziksel olarak gerçeklemeksizin gerçek hayatta nasıl çalışacağını test etmeye imkan sağlamaktadır.
- Hem assembly hem de C dili desteği sağlamaktadır.
- Hata ayıklama (debug) işlemlerine imkan tanımaktadır.

Geliştirme Araçları – ISIS Programı

Başlık Çubuğu

Menü Çubuğu

Araç Çubukları

Tasarım Alanı

Ön izleme

Tasarım Araç Çubuğu

Eleman Kutusu

Tasarım Alanı

Devre Çalıştırma Düğmeleri

The screenshot shows the ISIS Professional software interface. The title bar reads "ISIS Keypad_Uygulama_2 - ISIS Professional". The menu bar includes File, View, Edit, Library, Tools, Design, Graph, Source, Debug, Template, System, and Help. The toolbar contains various icons for file operations, editing, and simulation. On the left, there is a component library with a list of devices: 7SEG-MPX2-CA, 80C51, 7447, KEYPAD-PHONE, and RES. The main workspace displays a circuit diagram with a 7-segment display, a keypad, and a microcontroller (80C51). A terminal window shows the following text: "Sayıcı Keypad'ten girilen değere kadar 1 sn aralıktaki sayar", "Saydırmak için iki basamaklı bir sayı girerek # veya * tuşuna basın", and "Sayıya 0-9 arası sayı girmek için önce D'a sonra istenilen sayıya basılmaktadır". The bottom of the workspace contains a footer with the text: "Keypad ile Ayarlı Sayıcı Uygulaması", "Cüneyt BAYILMIŞ Murat ÇAKIROĞLU", "Dr. Ahmet Turan ÖZCERİT", and "C ile 8051 Mikrodenetleyici Uygulamaları http://www.8051turk.com". At the bottom of the window, there are simulation control buttons and a status bar with the text "Form tagged graphics/pads into device and place in library."

Keil C51 Derleyicisindeki Veri Tipleri

Veri Tipi	Bit	Bayt	Değer Aralığı
Bit	1		0 to 1
Signed char	8	1	-126 to +127
Unsigned char	8	1	0 to 256
Enum	16	2	-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	+/-1.7549E+38 to +/-3.402823E+38
Sbit	1		0 to 1
Sfr	8	1	0 to 256
Sfr16	16	2	0 to 65535

Bit

- ❑ Tek bitlik (bir bit) bir değişken tanımlamak için kullanılır.
- ❑ Tanımlanan bütün *bit* değişkenleri 8051 mikrodenetleyicisinin **dahili hafızasında** bulunan **bit adreslenebilir** bölgede tutulur.
- ❑ Bit adreslenebilir bölgenin toplam büyüklüğü **16 Bayt** uzunluğundadır dolayısıyla bir uygulama içerisinde toplam **128 adet** bit değişkeni kullanılabilir.
- ❑ Bit veri tipi ANSI C standardında bulunmayan bir veri tipidir.

Örnek:

```
bit led;          /*led değişkeninin bit olarak  
                  tanımlanması*/
```

```
led=0;          /*led değişkenine bir değer atılması*/
```

signed char/unsigned char

- ❑ Bu veri tipleri, standart C dilinde de bulunmaktadır.
- ❑ 1 bayt (8 bit) uzunluğundadır.
- ❑ *signed char* veri tipinin aralığı -128 ile +127 arasındadır
- ❑ *unsigned char* veri tipinin aralığı 0 ile 255 arasındadır.

Örnek:

```
unsigned char veri; /* veri değişkeninin işaretsiz  
                    karakter olarak tanımlanması*/
```

```
veri=0xFF;          /*veri değişkenine  
                    0FFh değerinin atılması*/
```

sbit

- ❑ SFR adres bölgesindeki her hangi bir bite erişmek için kullanılır.
- ❑ SFR hafıza bölgesinin tamamı bit adreslenebilir değildir.

Örnek:

```
sbit CY=PSW^7;          /* PSW kayıtçısının 7.bitinin CY
                        deęişkeni olarak tanımlanması*/

sbit EA=0xAF;          /*EA deęişkenin
                        tanımlanması */

sbit EA=IE^7;          /*EA deęişkenin tanımlanması*/
```

sfr

- ❑ SFR bellek bölgesinden 8 bitlik herhangi bir değişkeni tanımlamak için kullanılan veri tipidir.
- ❑ sfr veri tipi de standart C dilinde mevcut değildir.
- ❑ Yapı olarak diğer 8 bitlik veri tiplerine benzemektedir.

Örnek:

```
sfr P0=0x80;          /* Port-0'in tanımlanması*/
```

```
sfr P1=0x90;          /* Port-1'in tanımlanması*/
```

```
sfr P2=0xA0;          /* Port-2'nin tanımlanması*/
```

sfr16

- ❑ *sfr16* çoğu bakımdan *sfr* veri tipine benzemektedir.
- ❑ Aralarındaki tek fark *sfr16* veri tipi ile SFR adres bölgesinde bulunan 16 bitlik bir değişkeni tanımlayabilmenin mümkün olmasıdır.
- ❑ Bu sayede 16 bitlik bir saklayıcının düşük değerlikli bayt ile bir sonraki adres bölgesinde olan yüksek değerlikli baytı tek bir değişken olarak tanımlanabilir.

Örnek:

```
sfr16 T2=0xCC;    /* Düşük değerlikli baytı  
                  T2L=0CCh, yüksek değerlikli baytı  
                  T2H=0CDh olan Timer 2'nin sfr16  
                  veri tipi ile tanımlanması */
```

C51 Derleyicisinde Bellek Organizasyonu

Bellek Tipi	Boyut	Adres Aralığı	İlgili Komut
Kod	64 KB	0000h-FFFFh	code
External Data (Harici)	64 KB	0000h-FFFFh	xdata
External Data (paged)	256 B	00h-FFh	pdata
Dahili Data (lower RAM)	128 Byte	00h-7Fh	data
Dahili Data (indirect) upper	256 Byte	00h-FFh	idata
Dahili RAM Bit adreslenebilir alan	128 Bit (16 byte)	20h-2Fh	bdata

Belirli Bellekte Değişken Tanımlama

[işaret] [data tipi] [bellek tipi] [değişken adı]

❑ Program (Kod) belleğinde değişken tanımlama

❑ Örnek:

```
code int sayi=55; /* sayi değişkeninin program
belleğinde tanımlanması ve 55 değerinin yüklenmesi */
```

```
char code dizi[6]={0xFF,0xAA,0x56,0x87,0x45,0x89};
/* Program belleğinde dizi tanımlama */
```

Belirli Bellekte Değişken Tanımlama

[işaret] [data tipi] [bellek tipi] [değişken adı]

❑ Dahili veri belleğinde değişken tanımlama

Örnek:

```
char data oku; /* ilk 128 baytlık dahili veri
                belleğinde oku değişkeninin tanımlanması */

int idata sicaklik; /* sicaklik değişkeninin ikinci
                    128 baytlık dahili veri
                    belleğinde (upper RAM) tanımlanması */

int bdata bayrak; /* bayrak değişkeninin bit
                  adreslenebilir tamsayı olarak tanımlanması */

char bdata bit_dizi[5]; /* bit adreslenebilir dizi */
```

Belirli Bellekte Değişken Tanımlama

[işaret] [data tipi] [bellek tipi] [değişken adı]

□ Harici veri belleğinde değişken tanımlama

Örnek:

```
unsigned char xdata veri; /* veri değişkeninin  
                           harici RAM belleğinde  
                           (64 KB'lık bir alanda  
                           her hangi bir yerde) tanımlanması */
```

```
int pdata yaz; /* yaz değişkeninin harici RAM  
                belleğinde (256 bayt'lık bir  
                alanda her hangi bir yerde)  
                tanımlanması */
```

C51 Derleyicisindeki Bellek Modelleri

❑ **SMALL** bellek modeli:

- ❑ Tüm değişkenler standart olarak dahili veri belleğinde saklanır (Tüm değişkenlerin *data* tanımlaması kullanılarak ifade edilmesi gibi).
- ❑ Mümkün olan en az kod üretilir ve en etkin kullanım sağlanır.

❑ **COMPACT** bellek modeli:

- ❑ Tüm değişkenler harici veri belleğinin bir sayfasında tanımlanmaktadır (*pdata* ile)

❑ **LARGE** bellek modeli:

- ❑ Tüm değişkenler, harici veri belleğinin 64 KBayt'lık her hangi bir bölgesinde tanımlanmaktadır (*xdata* ile)

C51 Derleyicisinde Kesme Rutinlerinin Tanımlanması

Kesme Kaynağı	Kesme Adresi	C51'deki Kesme Numarası
Harici Kesme 0	0003h	0
Zamanlayıcı / Sayıcı 0	000Bh	1
Harici Kesme 1	0013h	2
Zamanlayıcı / Sayıcı 1	001Bh	3
Seri Haberleşme	0023h	4

```
void Fonksiyon_Adı (void) interrupt kesme_no [using depo no]
{
    Kesme oluştuğunda yapılması gereken işlemler
}
```

Örnek:

```
void Kesme_T0 () interrupt 1
{
    Kesme oluştuğunda yapılması gereken işlemler
}
```

Mikrodenetleyici Tabanlı C Program Yapısı

```
#include <89v51rd2.H> } kütüphane dosyalarının yazılması
```

```
#define ...
```

```
sfr ...  
sbit ...  
int... } Değişkenlerin  
tanımlanması
```

```
void kesme_0() interrupt 0 } Kesme rutinlerinin  
{  
.....  
{  
tanımlanması
```

```
void alt_program() } Alt programların yazılması  
{  
...  
}
```

```
main() } Ana programın yazılması  
{  
.....  
}
```

Örnek 1 Buton ile LED denetimi

```
#include <REG52.H>
#include <stdio.h>
sbit buton = P2^0; // Butonu'ü P2.0'a bađla
sbit led = P1^0; // LED'i P1.0'a bađla
void main(void)
{
led=0;
while(1)
{
while(!buton);
while(buton);
led=!led;
}
}
```

Örnek 2

/*Bu program bir LED'i bir saniye yakar ve söndürür*/

```
#include <reg52.h> // SFR tanımlamaları
#include <stdio.h>

#define SYSCLK 12000000 // SYSCLK frequency in Hz
sbit led = P1^0; //LED'i P1.0'a bağla

void msec(unsigned int Delay) // 1 msaniye geçiktirir.
{
    int i, j;
    for(j=0; j<Delay; j++)
    {
        for (i=0; i<500; i++);
    }
}

void main (void) // ana program
{
    while (1)
    {
        led=!led;
        msec (1000);
    }
}
```