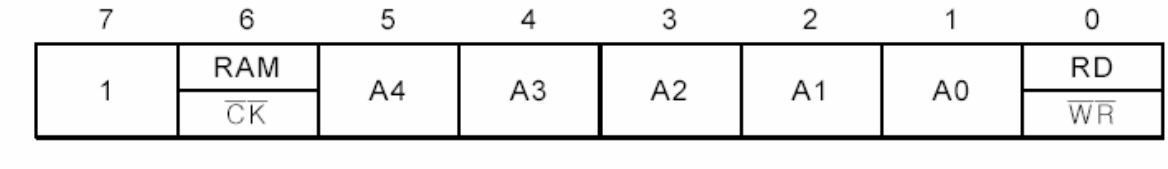
Konu:DS1302 RTC Kullanımı
Tarih:18 Mart 2005
Yazar: Tolga TAŞTAN

DS1302; RTC, takvim ve 31 byte'lık RAM'a (pil korumalı) sahiptir. Mikroişlemci ile basit seri bir ara yüzle haberleşir. Gerçek zaman saati (RTC) saniyeyi, dakikayı, saati, günü, ayı, yılı, haftanın gününü sayar ve 2100 yılına kadar tarih bilgileri yüklüdür. 2.0 V ile 5.0 V arasında gerilimlerde çalışabilir. 2.0 V'ta 300 nA'den daha az akım tüketir. Saat ve RAM hafızasına yazma ve okuma için seri olarak okuma veya yazma işlemi yapabilme (burst) moduna sahiptir. TTL ile uyumludur (Vcc = 5 V). - 40 °C ile +85 °C arasında çalışabilir.

DS1302'nin iki farklı besleme (VCC1, VCC2) giriş ucu vardır. Bu uçlardan bir tanesine bağlanan bir pil vasıtasıyla devreye enerji verilmediğinde dahi DS1302 saat ve zaman bilgilerini hafızasında tutmaya devam eder. Vcc1, Vcc2 + 0.2V olduğunda DS1302 beslemesini Vcc1 üzerinden yapar. Aynı şekilde Vcc2, Vcc1 + 0.2V veya daha büyük bir değere ulaştığında DS1302 otomatik olarak, besleme gerilimi için Vcc2 pinini kullanmaya başlar. DS1302 ile yapılan tüm haberleşmeler komut baytı ile başlar. Komut baytının en anlamlı biti (MSB) her zaman lojik "1" olmalıdır.

Bu bit lojik "0" olarak DS1302'ye gönderilirse, DS1302'den herhangi bir yanıt alınamaz. Komut byte'ı Şekil-1'de gösterilmiştir.

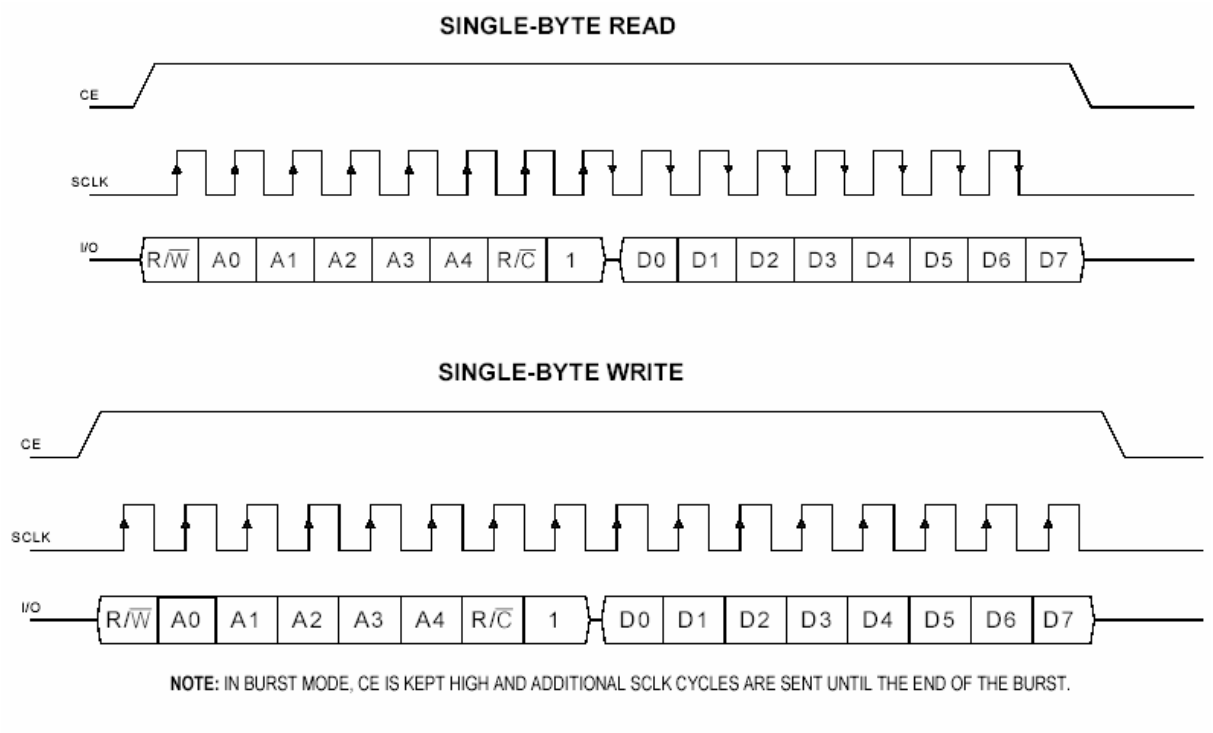


Şekil.1. Komut Byte'ı

Bit 6 lojik "0" olarak gönderilirse, saat / takvim üzerinde okuma veya yazma işlemi yapılır. Bu bit lojik "1" olarak DS1302 'ye gönderilirse, DS1302'nin 31 bitlik RAM hafızasında okuma veya yazma işlemi yapılır. Bit 1 den bit 5'e kadar olan bölüm ise ulaşmak istediğimiz registerın adresinin belirtilmesinde kullanılır. Bit 0 ise DS1302'ye yazma veya okuma işlemlerinden hangisini yapacağımızı bildirir. Bu bit lojik "0" ise DS1302'ye yazma işlemi; lojik "1" ise DS1302'den okuma işlemi yapılır. Komut baytının DS1302 'ye iletimi ilk önce en az anlamlı bit ile başlanarak yapılır.

DS1302 'nin RST ucu (5 nolu pin) normalde lojik"0" olarak tutulmaktadır. DS1302 ile veri transferi yapılacağı zaman bu uç lojik "1" yapılmalıdır. RST lojik "0" iken veri giriş / çıkış ucu yüksek empedans durumunda bulunmaktadır. Ayrıca RST ucu lojik "1" yapılmadan önce SCLK ucu mutlaka lojik "0" olmalıdır.

Komut baytının DS1302 'ye aktarılması ve DS1302'ye istenen herhangi bir verinin yazılması SCLK ucunun çıkan kenarlarında olmaktadır. SCLK ucu lojik "0" 'dan lojik "1" 'e çekildiğinde DS1302'ye iletilmek istenen bitin DS1302'nin I / O ucunda hazır olması gerekmektedir. Benzer şekilde DS1302 'den herhangi bir verinin okunması için gönderilecek olan komut baytı SCLK ucunun çıkan kenarlarında DS1302'ye aktarılmakta, okuma işlemi ise son çıkan kenardan itibaren düşen kenarlarda gerçekleşmektedir. DS1302'ye veri yazma ve okuma işlemleri şekil-2'de gösterilmiştir.



Şekil.2. DS1302'den okuma ve DS1302'ye yazma

DS1302 'ye herhangi bir yazma işleminden önce kontrol kaydedicilerinin (register) 7. biti lojik "0" konumuna alınması gereklidir. Aksi taktirde DS1302'ye herhangi bir yazma işlemi gerçekleştirilemez. Ayrıca saniye kaydedicisinin 7. biti saat durma bayrağı olarak tanımlıdır. Bu bit lojik "1" olduğunda DS1302 osilatörü durur ve DS1302 bekleme konumuna geçer. Bunun için DS1302'ye ilk defa enerji verildiğinde bu bitin lojik "0" olup olmadığı DS1302'den öğrenilmelidir. Eğer lojik'1' ise lojik'0' yapılması sağlanarak osilatörün çalışmaya başlaması sağlanmalıdır. Eğer DS1302'den gelen bilgi ilgili bitin lojik '0' olduğu yönünde ise herhangi bir değişiklik yapılmasına gerek yoktur. Bu bit lojik "0" olarak atandıktan sonra DS1302 pil ile yedeklendiğinden zaman bilgisini kaybetmez.

RTC

READ	WRITE	BIT 7	BIT 6	BIT 5	BIT 4	BIT 3	BIT 2	BIT 1	BIT 0	RANGE
81h	80h	CH	10 Seconds			Seconds				00–59
83h	82h		10 Minutes			Minutes				00–59
85h	84h	12/$\overline{24}$	0	10 $\overline{\text{AM/PM}}$	Hour	Hour				1–12/0–23
87h	86h	0	0	10 Date		Date				1–31
89h	88h	0	0	0	10 Month	Month				1–12
8Bh	8Ah	0	0	0	0	0	Day			1–7
8Dh	8Ch	10 Year				Year				00–99
8Fh	8Eh	WP	0	0	0	0	0	0	0	—
91h	90h	TCS	TCS	TCS	TCS	DS	DS	RS	RS	—

CLOCK BURST

BFh	BEh

RAM

C1h	C0h		00-FFh
C3h	C2h		00-FFh
C5h	C4h		00-FFh
. . .	. . .		. . .
FDh	FCh		00-FFh

RAM BURST

FFh	FEh

Şekil.3. DS1302 Register Tablosu

Kütüphane kodunun açıklanması
Kütüphanede 6 farklı fonksiyon kullanılmıştır. Bu kütüphane CC5X pic compiler için hazırlanmıştır. Bu fonksiyonlar ve işlemleri aşağıdaki gibidir.

reset_3w() , DS1302'yi resetlemeye yarar
clk_pulse(), pulse gönderir
uns8 read_byte(), 8bit lik bilgi okur
write_byte(uns8 command), 8bit'lik command komutunu DS1302'ye yazar
send_command(uns8 cmd, uns8 data), DS1302'nin cmd register'ına data bilgisini yazar
uns8 get_command(cmd), cmd komutunun karşılığındaki bilgiyi okur

Bu fonksiyonların kullanımıyla DS1302 ile Pic arasında 3 bacak üzerinden iletişim sağlanır.
Verilen kod ve kodların karşılıları Şekil 3'de verilen Tablo ile karşılaştırıldığında DS1302'nin çalışma mantığı daha da netlik kazanacaktır.
İyi çalışmalar dilerim.

Tolga TAŞTAN

KÜTÜPHANE (ds1302.c)

```
/*
////////////////////////////////////////////////////////////
//////////////////////Tolga TASTAN ////////////////////////
////////////////////////////////////////////////////////////
////////////////// For CC5X Compilers ////////////////////
////////////////////////////////////////////////////////////
////////////////////// DS 1302 Library ////////////////////
////////////////////////////////////////////////////////////
DS1302 Real time clock library is written by Tolga TASTAN at
27 Sept.2004. It could be used in CC5X Pic Compiler. The compiler
could be dowloaded from http://www.bknd.com/cc5x/ address.
Firstly DS1302 initialized by controller with rtc_init() function.
Then time is set and clock started with Write Protect.
I hope its help to you for improve your codes.

                                                tolga@aadf.net
////////////////////////////////////////////////////////////
Functions                Operations
-------------------------------------------------------------
reset_3w(),                        Reset the chip
clk_pulse(),                       Send Clock Pulse to DS1302
uns8 read_byte(),                  Read one byte from DS1302
write_byte(data),                  Write one byte to DS1302
send_command(cmd,data),                Send data to cmd register at DS1302
uns8 get_command(cmd),                 Send the command address then receive the
DS1032 answer
////////////////////////////////////////////////////////////
*/

#pragma bit RTC_SCLK @ PORTB.5
#pragma bit RTC_IO @ PORTB.6 //should be set in below
#pragma bit RTC_E @ PORTB.7

uns8 ret_hr,ret_min,ret_sec;

// Reset 3W'is using for Enable bit
void reset_3w(void)
{
        RTC_E=0;
        Delay1ms(2);
        RTC_E=1;
}

// Clock Pulse is usign for data transfer clocks
void clk_pulse(void)
{
        RTC_SCLK=1;
        Delay1ms(2);

        RTC_SCLK=0;
        Delay1ms(2);
}

// Write a Byte to DS1302
```

```
void write_byte(uns8 command)
{
	TRISB.6=0; //Set IO bit for output
	RTC_SCLK=0;
//Start with LSB
	RTC_IO=command.0; nop();
	clk_pulse();
	RTC_IO=command.1; nop();
	clk_pulse();
	RTC_IO=command.2; nop();
	clk_pulse();
	RTC_IO=command.3; nop();
	clk_pulse();
	RTC_IO=command.4; nop();
	clk_pulse();
	RTC_IO=command.5; nop();
	clk_pulse();
	RTC_IO=command.6; nop();
	clk_pulse();
	RTC_IO=command.7; nop();
	clk_pulse();

}


// Read byte from DS1302
uns8 read_byte(void)
{
	uns8 got_com; //define return value
	got_com=0;

//Set IO Port for input
	TRISB.6=1;
	RTC_SCLK=0;
//Start with LSB
	got_com.0=RTC_IO; nop();
	clk_pulse();
	got_com.1=RTC_IO; nop();
	clk_pulse();
	got_com.2=RTC_IO; nop();
	clk_pulse();
	got_com.3=RTC_IO; nop();
	clk_pulse();
	got_com.4=RTC_IO; nop();
	clk_pulse();
	got_com.5=RTC_IO; nop();
	clk_pulse();
	got_com.6=RTC_IO; nop();
	clk_pulse();
	got_com.7=RTC_IO; nop();
	clk_pulse();

return(got_com);
}

void send_command(uns8 cmd, uns8 data)
{
```

```
        RTC_SCLK=0;
        reset_3w(); nop();
        write_byte(cmd);
        write_byte(data);
        reset_3w(); nop();
        RTC_E=0;
}

uns8 get_command(uns8 cmd)
{
uns8 rcv_command;
rcv_command=0;

        RTC_SCLK=0;
        reset_3w(); nop();
        write_byte(cmd);
        rcv_command = read_byte();
        reset_3w(); nop();
        RTC_E=0;

return (rcv_command);
}

void rtc_init(uns8 batt_stat)
{

reset_3w();

//Disable Write Protect
send_command(0x8e,0x00);

send_command(0x80,0b10000000); //Clock Halted

send_command(0x90,batt_stat); //Diode and resistor select

RTC_E=0;
}
```