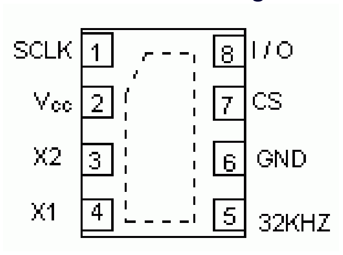
Oct 26, 2004

App Note 3382: Interfacing a MAX6901 RTC With an 8051-Type Microcontroller

This app note demonstrates how to interface a MAX6901 to an 8051-type microcontroller and provides example code showing basic interface routines. The microcontroller used in this example is the DS2250, and the software is written in C.

MAX6901 Pin Assignment



Description

This app note demonstrates how to interface a MAX6901 Real Time Clock (RTC) to an 8051-type microcontroller and provides example code showing basic interface routines. The microcontroller used in this example is the DS2250, and the software is written in C.

Operation
The program uses three general-purpose port pins on the microcontroller to control the 3-wire bus. The microcontroller initiates a data transfer by sending a command/address byte to the MAX6901. The microcontroller then sends additional data and/or SCLKs to the MAX6901, which transmits or receives data based upon the command byte.

A schematic of the circuit is shown in Figure 1. The software is shown in Figure 2.

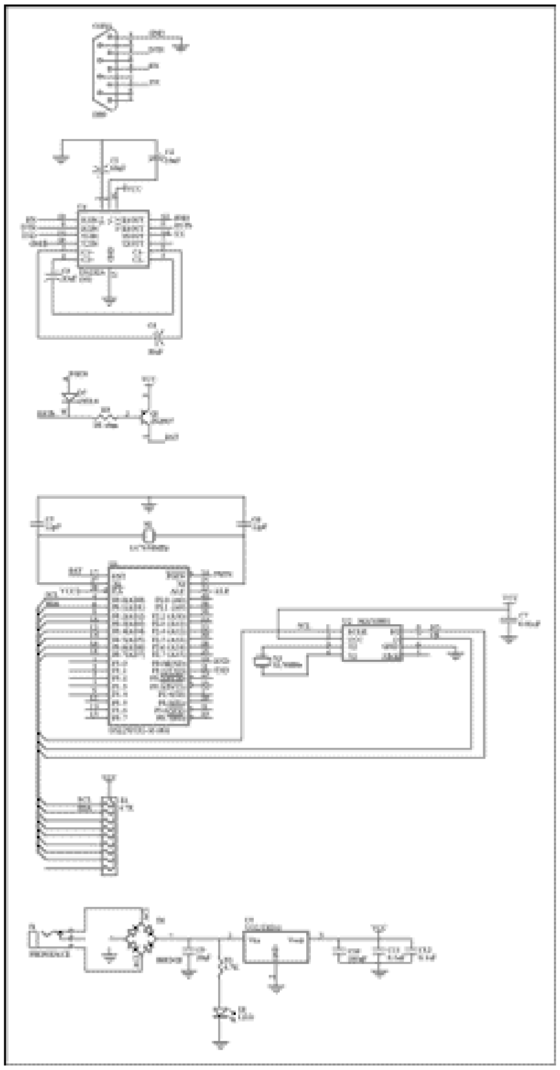


Figure 1. Schematic of Daughter Card *(PDF Download)*

Figure 2. Code Listing

```
/***************************************************************************/
/* DEMO6901.c                                                              */
/***************************************************************************/
/* #pragma code symbols debug */
#include                /* Prototypes for I/O functions     */
#include                /* Register declarations for DS5000 */
/****************************** Defines ********************************/
sbit CE      = P0^0;
sbit SCLK    = P0^1;
sbit IO     = P0^2;
/************************ Function Prototypes **************************/
void    writebyte();
void    initialize();
void    disp_clk_regs();
void    burstramread();
void    burstramwrt();
/************************** Global Variables ***************************/
uchar   cy, yr, mn, dt, dy, hr, min, sec, msec;

void reset_3w() /* ---- reset the 3-wire interface ---- */
{
        SCLK = 0;
        CE = 0;
        IO = 0; /* program IO pin to 0V */
}
void wbyte_3w(uchar W_Byte)     /* --- write one byte to the DUT --- */
{
uchar i;

        CE = 1;
        for(i = 0; i < 8; ++i)
        {
                if(W_Byte & 0x01)
                        IO = 1; /* set port pin high to read data */
                else
                        IO = 0;
                SCLK = 0;
                SCLK = 1;
                W_Byte >>= 1;
      }
}
uchar rbyte_3w()        /* --- read one byte from the DUT --- */
{
uchar i;
uchar R_Byte = 0;

        CE = 1;
        IO = 1;
        for(i=0; i<8; ++i)
        {
                SCLK = 1;
                SCLK = 0;
                if(IO)
                {
                        R_Byte >>= 1;
                        R_Byte += 0x80;
                }
                else
                        R_Byte >>= 1;
        }
        return R_Byte;
}
void writebyte()        /* ----- write one byte, prompt for address and data ------ */
{
uchar add;
uchar dat;
        /* Get Address & Data */
        printf("
Enter the Write Address (h)
ADDRESS (80,82,84...):");
        scanf("%bx", &add);
        printf("DATA (0-ff):");
        scanf("%bx", &dat);

        reset_3w();
        wbyte_3w(add);
        wbyte_3w(dat);
        reset_3w();
}
void initialize()       /* ----- init clock data using user entries ----- */
/* Note: NO error checking is done on the user entries! */
{
        reset_3w();
        wbyte_3w(0x0f);         /* control register write address */
        wbyte_3w(0x00);         /* clear write protect */
        reset_3w();

        printf("
Enter the year (0-99): ");
        scanf("%bx", &yr);
        printf("Enter the month (1-12): ");
        scanf("%bx", &mn);
        printf("Enter the date (1-31): ");
        scanf("%bx", &dt);
        printf("Enter the day (1-7): ");
        scanf("%bx", &dy);
        printf("Enter the hour (1-23): ");
        scanf("%bx", &hr);
        hr = hr & 0x3f; /* force clock to 24 hour mode */
        printf("Enter the minute (0-59): ");
        scanf("%bx", &min);
        printf("Enter the second (0-59): ");
        scanf("%bx", &sec);

        reset_3w();
        wbyte_3w(0xbe); /* clock burst write */
        wbyte_3w(sec);
        wbyte_3w(min);
        wbyte_3w(hr);
        wbyte_3w(dt);
        wbyte_3w(mn);
        wbyte_3w(dy);
        wbyte_3w(yr);
        wbyte_3w(0);            /* control */
        reset_3w();
        wbyte_3w(0x92);
        wbyte_3w(0x20); /* century data */
        reset_3w();
}
void disp_clk_regs()            /* --- loop reading clock, display when secs change ---
*/
{
uchar mil, pm, prv_sec = 99;

while(!RI)      /* Read & Display Clock Registers */
{
        reset_3w();
        wbyte_3w(0xbf); /* clock burst read */
        sec = rbyte_3w();
        min = rbyte_3w();
        hr = rbyte_3w();
        dt = rbyte_3w();
        mn = rbyte_3w();
        dy = rbyte_3w();
        yr = rbyte_3w();
        cy = rbyte_3w();        /* dummy read of control register */
        reset_3w();
        wbyte_3w(0x93); /* century byte read address */
        cy  = rbyte_3w();
        reset_3w();

        if(hr & 0x80)
                mil = 0;
        else
                mil = 1;

        if(sec != prv_sec)      /* display every time seconds change */
        {
                if(mil)
                {
                        printf("
%02bX%02bX/%02bX/%02bX %01bX", cy, yr, mn, dt, dy);
                        printf(" %02bX:%02bX:%02bX", hr, min, sec);
                }
                else
                {
                        if(hr & 0x20)
                                pm = 'P';
                        else
                                pm = 'A';
                        hr &= 0x1f;     /* strip mode and am/pm bits */
                        printf("
%02bx%02bx/%02bx/%02bx %02bx", cy, yr, (mn & 0x1f), dt, dy);
                        printf(" %02bx:%02bx:%02bx %cM", hr, min, sec, pm);
                }
        }
        prv_sec = sec;
}
   RI = 0;  /* Swallow keypress to exit loop */
}
void burstramread()             /* ------ read RAM using burst mode ----- */
{
uchar k;

        printf("
MAX6901 RAM contents:
");

        reset_3w();
        wbyte_3w(0xff); /* ram burst read */
        for (k = 0; k < 31; k++)
        {
                if(!(k % 8) )   printf("
");
                printf("%02.bX ", rbyte_3w() );
        }
        reset_3w();
```

```
}
void burstramwrt(uchar Data)    /* ------ write RAM using burst mode ------- */
{
uchar j, k;

        reset_3w();
        wbyte_3w(0xfe); /* ram burst write */
        for (k=0; k < 31; ++k)
        {
                wbyte_3w(Data);
        }
        reset_3w();
}
main (void)             /* ----------------------------------------------------- */
{
uchar i, M, M1;

        while (1)
        {
                printf("
MAX6901 build %s
", __DATE__);
                printf("CI Clock Init
");
                printf("CR Clock Read CW Clock Write
");
                printf("RR Read RAM   RW RAM Write
");
                printf("Enter Menu Selection: ");

                M = _getkey();

                switch(M)
                {
                        case 'C':
                        case 'c':
                        printf("\rEnter Clock Routine to run:C");
                        M1 = _getkey();

                        switch(M1)
                        {
                                case 'I':
                                case 'i':       initialize();
                                                break;

                                case 'R':
                                case 'r':       disp_clk_regs();
                                              break;

                                case 'W':
                                case 'w':       writebyte();
                                                break;
                        }
                        break;

                        case 'R':
```

```
                        case 'r':
                        printf("\rEnter Ram Routine to run:R");
                        M1 = _getkey();

                        switch(M1)
                        {
                                case 'R':
                                case 'r':       burstramread();
                                                break;
                                case 'W':
                                case 'w':       printf("
Enter the data to write: ");

                                                scanf("%bx", &i);
                                                burstramwrt(i); break;
                        }
                        break;
                }
        }
}
```

More Information

MAX6901: QuickView -- Full (PDF) Data Sheet -- Free Samples