

**T.C.
Yeditepe University**

**Department of Electrical-Electronics
Engineering**

Microcontrollers Term Project Report

Digital Frequencymeter

Prof.Dr.Herman Sedef

Enis Ürel

Fatih Erdem

25.12.2008

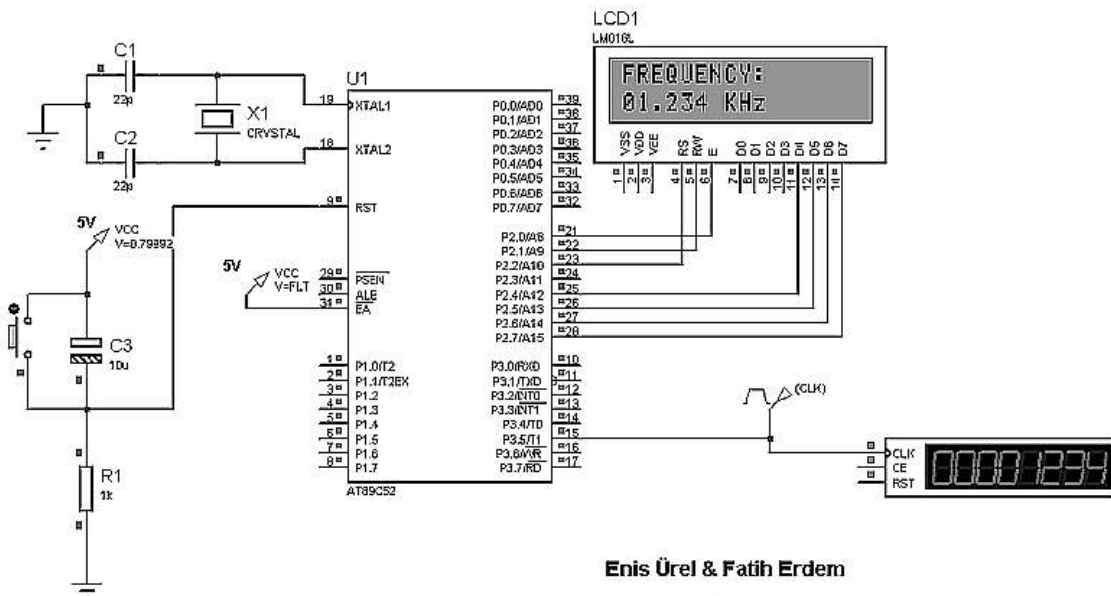
Purpose:

To design a AT89S52 based digital frequencymeter with a bandwidth of 65 KHz.

Hardware:

In hardware we have used one AT89S52 microcontroller and one lm016l lcd module.

Schematic in ISIS:

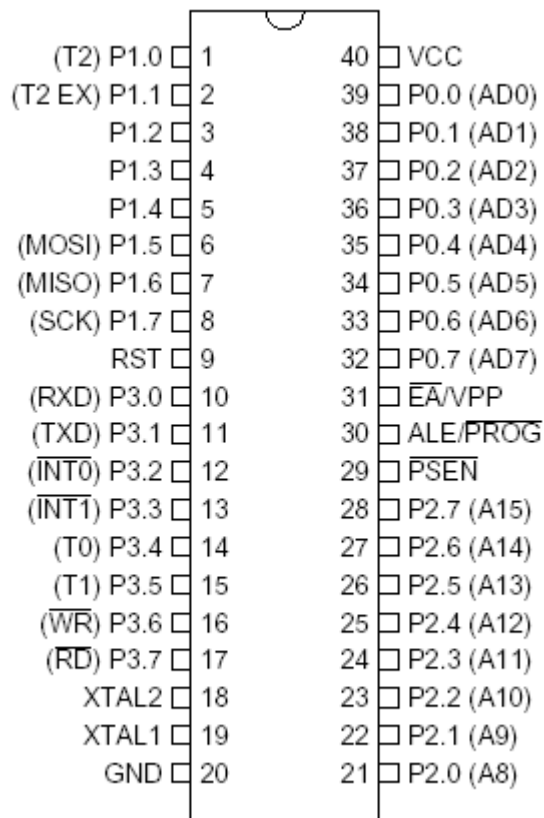


Enis Ürel & Fatih Erdem

Frequency Meter / Term Project

Please see appendix for large schematic and realized circuit.

AT89S52



AT89S52 is a low power, 8bit CMOS based 40 pin microcontroller having intel 8051 core, it has three 16 bit timers, serial port module, two external interrupts, 256 byte internal ram, 8k flash rom.

We have used timer1 to count edges and timer2 to keep 1s time interval. We have used port1 to drive lcd module.

2x16 LM016L LCD Module

LCD displays can be considered readable and writable memory, in other words they are like RAMs. They contain their own driver circuit to interpret incoming commands or data from external drivers such as microcontrollers.

There are three main control line of lcd modules RS, RW, E and 8 bit data line.

In order to send a command or data to lcd, we keep RS and RW low and we put command(or data) byte to data pins then we trigger a high to low transition on E pin. Commands or data are only sent to lcd module on this transition.

Main features of our frequencymeter:

- * 65 Khz bandwidth, minimum measurable frequency is 1 Hz, maximum measurable frequency is 65535 Hz

- * Input signal can be any TTL Sinwave, squarewave, triangle wave, sawtooth...

- * Refreshing measurements every 1.5 second

- * LCD Display

- * Reset button

Main algorithm:

We have set timer2 to take interrupt every 50ms so that obtain $20 \times 50\text{ms} = 1\text{s}$ time interval. We count number of interval and when it reaches to 20 we trigger a sub routine which is `one_sec_interval_elapsed`. We used timer1 to count edges on P3.5.

In order to start measurement we start timer1 by setting bit TR1 and timer2 by setting bit TR2 simulatenously(with 1us delay).

Software:

We have used demo version of pinnacle to develop, simulate and debug our source code.

`freq_met.asm`

; AT89S52 Based Frequencymeter with LCD Display
; Enis Ürel & Fatih Erdem

;TIMER1-16 bit mod yükselen kenarları saymaya ayarlandı
;TIMER2-auto reload mod 50ms lik zaman tutmak için ayarlandı

;R1 50ms lik aralıkları saymak için tahsis edildi, 20 tane sayınca 1s dolmuş olur.
;40H, 41H frekansın ilk alındığı bölgeler, hex cinsten.
;Bütün bekleme fonksiyonları milisaniye cinsinden.
;P1 LCD ye tahsis edildi.

ORG 0000
LJMP START

ORG 001BH
LJMP T1INT

ORG 002BH
LJMP T2INT

;Timer2 initial
LB EQU 0AFH
HB EQU 3CH

; LCD Pins
E EQU P2.0
RW EQU P2.1
RS EQU p2.2

ORG 100H
START:
;initializations
SETB IE.7; interrupts enable
SETB IE.5; tmr2 int enable
MOV 89H,#50H ; timer1 16 bit counter olarak ayarlandı, P3.5
;(FFFF-3CAF)+ Timer2 INT rutin=50ms
MOV RCAP2L,#LB ; timer2 50m sn de bir
MOV RCAP2H,#HB ; reload etmek üzere ayarlandı
MOV TL2,#LB ; timer2 ye ilk
MOV TH2,#HB ; değeri verildi
MOV R1,#00 ; R1 temiz

LCALL LCD_INIT; lcd yi başlatır ve "FREQUENCY:" yazar.

LCALL START_MEASURE
MAIN:
SJMP MAIN

START_MEASURE:
MOV TH1,#00 ; timer1 i temizle
MOV TL1,#00

MOV TL2,#LB ; timer2 ye ilk
MOV TH2,#HB ; değeri verildi

MOV R1,#00

```
SETB TR1
SETB TR2
RET
```

```
;Timer1 interrupt routine
T1INT:
INC 30H
```

```
;Timer2 interrupt routine 50ms lik aralıkları sayar, 20. de 1s dolmuştur.
T2INT:
CLR TF2; CLR int flag
INC R1
CJNE R1,#14H,CONT ; 20 tane 50ms geldiyse 1s dolmuştur
ACALL ONE_SEC_INTERVAL_ELAPSED
CONT:
RETI
```

ONE_SEC_INTERVAL_ELAPSED: ; BDC hesaplar, lcd ye yazar, yeni olcum alır

```
CLR TR1; timer1 i durdur.
CLR TR2; timer2 yi durdur.
CPL P1.1
MOV 40H,TH1 ; timer 1 in içeriğini
MOV 41H,TL1 ; 40H ve 41H te sakla.

;;BDC basliyor;;;
MOV r1,41H ; 175d
MOV r2,40H ; 91AFH = 37295d

; Result: R7=3, R6=7, ..., R3=5
```

```
MOV R5,#0
MOV R6,#0
MOV R7,#0
```

```
MOV A,R1
MOV B,#10
DIV AB
```

```
MOV R3,B
MOV A,R2
JZ SPRING
MOV B,#10
DIV AB
```

```
MOV R5,B
MOV R6,A
```

```
MOV R0,#05 ;
ACALL TRANSF
INC R0 ;
ACALL TRANSF
```

```
SPRING:    MOV  R0,#03
           ACALL DECADJ
```

```
TRANSF:    MOV  A,@R0
           MOV  B,#6
           MUL  AB
           DEC  R0
           DEC  R0
           ADD  A,@R0
           MOV  @R0,A
```

```
           INC  R0
           INC  R0
           MOV  A,@R0
           MOV  B,#5
           MUL  AB
           DEC  R0
           ADD  A,@R0
           MOV  @R0,A
```

```
           INC  R0
           MOV  A,@R0
           RL   A           ;
           MOV  @R0,A
```

```
           RET
```

```
DECADJ:    MOV  A,@R0
DECLOP:    MOV  B,#10
           DIV  AB
           MOV  @R0,B
           INC  R0
           ADD  A,@R0
           MOV  @R0,A
```

```
           CJNE R0,#7,DECLOP
           ;;;;;;;;;;BCD bitti;;;;;;;;;;
           ;Sonucu LCD de gosterelim
```

```
           ;;;;;;;;;;Ultimate Display;;;;;;;;;
           MOV  A,#11000000B ;bu komutla LCD
           LCALL SEND_COMMAND; 2. satirin basina gecer
```

```
           MOV  A,R7
           ADD  A,#30H
           LCALL SEND_DATA
```

```
           MOV  A,R6
           ADD  A,#30H
           LCALL SEND_DATA
```

```
           MOV  A,#2EH
           LCALL SEND_DATA
```

```
           MOV  A,R5
```

```

ADD      A,#30H
LCALL   SEND_DATA

MOV      A,R4
ADD      A,#30H
LCALL   SEND_DATA

MOV      A,R3
ADD      A,#30H
LCALL   SEND_DATA

MOV      A,#' '
LCALL   SEND_DATA

MOV      A,#'K'
LCALL   SEND_DATA

MOV      A,#'H'
LCALL   SEND_DATA

MOV      A,#'z'
LCALL   SEND_DATA
.....
LCALL   WAIT_500
LCALL   START_MEASURE; interruptan cikmadan yeni olcum baslativer

```

```
RET
```

```

WAIT_1:
PUSH 30H
MOV   30H,#0FFH
MOV   31H,#47H
LOOPX:
NOP
DJNZ 30H,LOOPX
POP 30H
RET

```

```

WAIT_20:
PUSH 30H
PUSH 31H
MOV   30H,#0FFH
MOV   31H,#47H
LOOPA:
LOOPB:DJNZ 31H,LOOPB
DJNZ 30H,LOOPA
POP 31H
POP 30H
RET

```

```
WAIT_500:
```

```

PUSH 30H
PUSH 31H
MOV     30H,#0FFH
MOV     31H,#0C0H
LOOP1:
LOOP2:DJNZ 31H,LOOP2
DJNZ 30H,LOOP1
POP     31H
POP     30H
RET

```

```

;::::::::::LCD RELEATED FUNCTIONS;::::::::::
;::::::::::

```

```

LCD_INIT:

```

```

CLR     RS
CLR     RW
MOV     P2,#21H; 4 bit mod
        LCALL WAIT_20
CLR     E
SETB   E

```

```

MOV     A,#2FH ; 2 line mod
LCALL  SEND_COMMAND

```

```

MOV     A,#01H; ekrani temizle ki resetten sonra bir şey kalmasin
LCALL  SEND_COMMAND

```

```

MOV     A,#0CH; "cursor not blink and not underlined" olarak ayarlandi
LCALL  SEND_COMMAND

```

```

MOV     A,#'F'
LCALL  SEND_DATA
MOV     A,#'R'
LCALL  SEND_DATA
MOV     A,#'E'
LCALL  SEND_DATA
MOV     A,#'Q'
LCALL  SEND_DATA
MOV     A,#'U'
LCALL  SEND_DATA
MOV     A,#'E'
LCALL  SEND_DATA
MOV     A,#'N'
LCALL  SEND_DATA
MOV     A,#'C'
LCALL  SEND_DATA
MOV     A,#'Y'
LCALL  SEND_DATA
MOV     A,#':'
LCALL  SEND_DATA

```

```

RET

```

```

SEND_COMMAND:

```


Test Results:

According to our test with equipment agilent signal generator and multimeter our device is quite accurate regardless of which type wave is applied.

Some of test results:

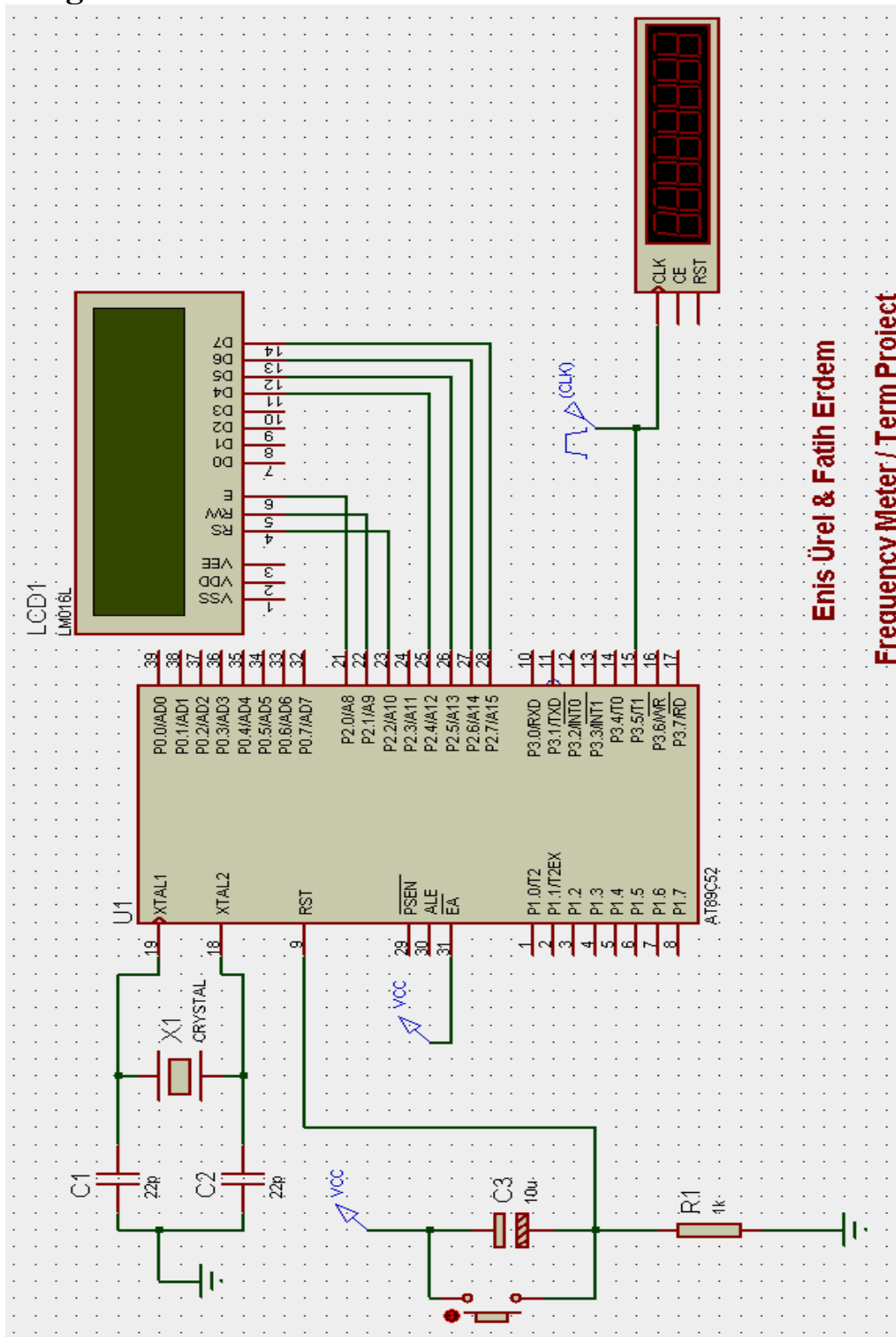
Agilent Multimeter	Our Device
9.89 hz	00.010 khz
13.24678 khz	13.247 khz
46.78712 khz	46.788 khz
34.09245 khz	34.092 khz
22.67892 khz	22.679 khz
67.23312 khz	1.698 khz(oflow)
100.000 khz	34.465 khz(oflow)

Conclusion:

In this project we have learned details of AT89S52 microcontroller and had experience in designing a system being microcontroller based. In addition, developing the software in assembly has refreshed our software knowledge and it helped us to learn the hardware well.

Appendix

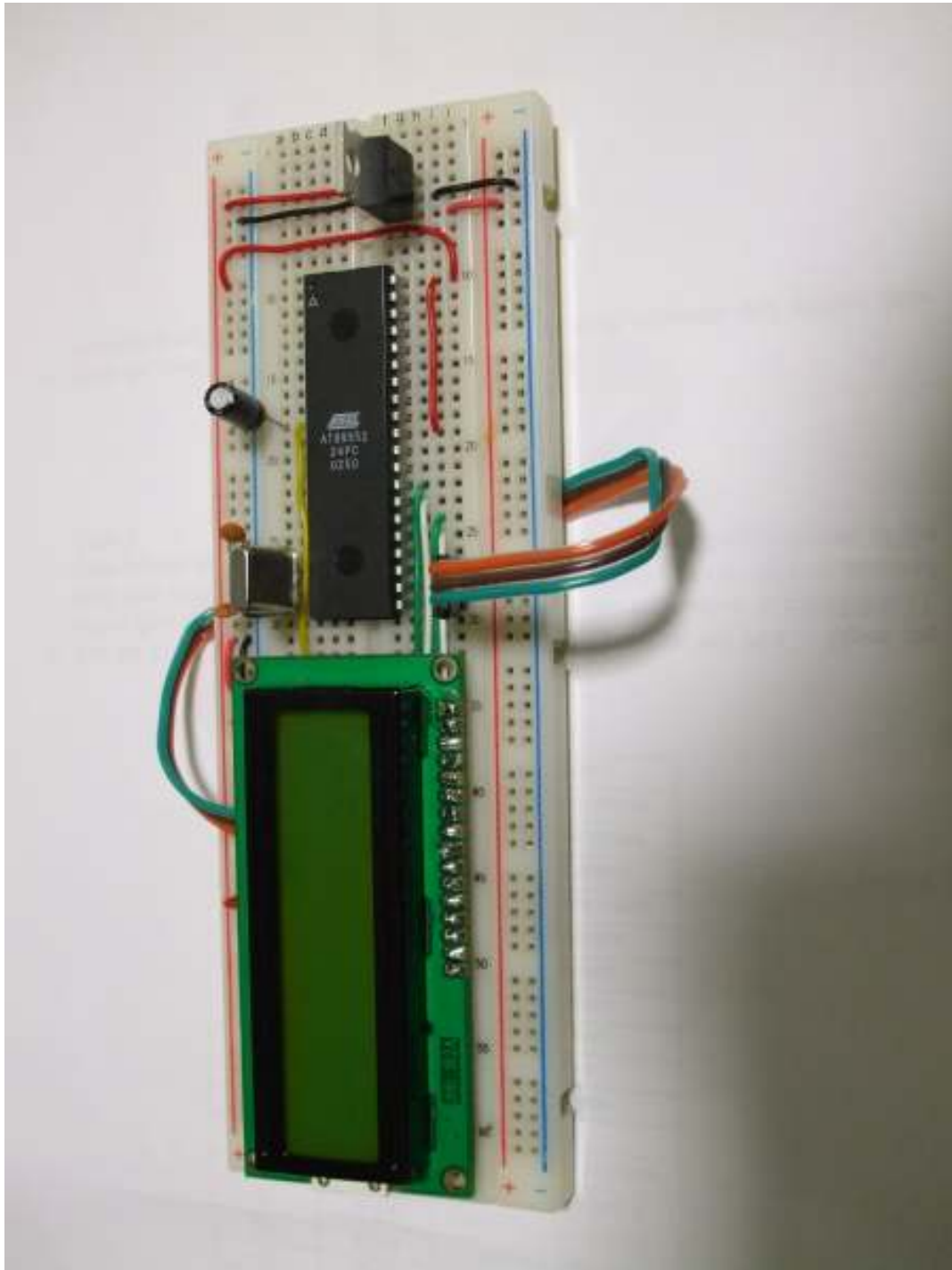
Large schematic in ISIS:



Enis Ürel & Fatih Erdem

Frequency Meter / Term Project

Realized circuit:



References:

MCS(R) 51 Microcontroller Family User's Manual
www.intel.com/design/mcs51/manuals/272383.htm

www.8052.com Code library, hex2BCD examples.

www.fatiherdem.net/sikca-kullanilan-entegrelerin-pin-diagramlari/